

Use R!

J. Christopher Westland

Audit Analytics

Data Science for
the Accounting Profession

EXTRAS ONLINE

 Springer

Use R!

Series Editors

Robert Gentleman, 23andMe Inc., South San Francisco, USA
Kurt Hornik, Department of Finance, Accounting and Statistics, WU
Wirtschaftsuniversität Wien, Vienna, Austria
Giovanni Parmigiani, Dana-Farber Cancer Institute, Boston, USA

Use R!

This series of inexpensive and focused books on R is aimed at practitioners. Books can discuss the use of R in a particular subject area (e.g., epidemiology, econometrics, psychometrics) or as it relates to statistical topics (e.g., missing data, longitudinal data). In most cases, books combine LaTeX and R so that the code for figures and tables can be put on a website. Authors should assume a background as supplied by Dalgaard's Introductory Statistics with R or other introductory books so that each book does not repeat basic material.

More information about this series at <http://www.springer.com/series/6991>

J. Christopher Westland

Audit Analytics

Data Science for the Accounting Profession

J. Christopher Westland 
Department of Information & Decision Sciences
University of Illinois
Chicago, IL, USA

ISSN 2197-5736 ISSN 2197-5744 (electronic)
Use R!
ISBN 978-3-030-49090-4 ISBN 978-3-030-49091-1 (eBook)
<https://doi.org/10.1007/978-3-030-49091-1>

© Springer Nature Switzerland AG 2020
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG.
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Foreword by Erik Brynjolfsson

I have known my colleague Chris Westland for nearly 25 years, and during this time, he has contributed a wealth of innovations in areas relevant to my own research. So it gives me great pleasure to write this foreword for his new book *Audit Analytics* which offers substantial contributions toward automation of the accounting profession. *Audit Analytics* addresses a number of topics that have been pivotal in my own research as Director of the MIT Initiative on the Digital Economy. I have investigated productivity contributions of information technology and the key roles of organizational capital and intangible assets all the while wishing that the accounting data on which I was so dependent had been more complete, informative, and reliable. Westland’s book puts forth many of the tools and concepts that will allow accounting to provide future economists with more relevant, complete, and reliable data.

Chris is well-equipped to write this book. He is a Certified Public Accountant who has worked professionally as an auditor for Touche Ross (now Deloitte Touche Tohmatsu); he has been a prolific researcher in information technology, in statistics, in data science, and in financial accounting. He is the author of seven well-received books and is editor-in-chief of our top journal in electronic commerce. I can think of no one better suited to write a path-breaking book that brings together the latest innovations in data science, statistical information technology, and the practice of auditing.

This book is both timely and needed. Technological progress through improvements in computer hardware, software, and networks has been so rapid, and so surprising, that accounting and auditing, along with many other industries, are not keeping up. Accounting is now both global and technology-centric, yet accounting rules and regulations change slowly. Any change yields winners and losers—a difficult prospect in times of rising income inequality. Technology’s promise of cost-effective audit automation accompanies wrenching change: jobs and skillsets transform, and wages diverge even as productivity and profits soar. The accounting profession is not spared this transformation.

Audit Analytics is directed toward today’s practicing auditor, and consequently, its structure parallels that of an audit. It includes several topics that I have not seen previously in accounting texts—analysis of the complex statistical distributions of accounting transactions, and methods for using accounting reports in predicting future adverse events through statistical time-series and machine learning models, as well as an entertaining history of double-entry accounting prior to Pacioli. Chris’s presentation of blockchain technology, along with R-code to implement a blockchain, is a welcomed reprieve from the vendor hype surrounding blockchain. This book is not just theory—Westland provides a wealth of software code to implement machine learning models, blockchains, “test decks,” interactive scripts for management of the audit, and software to implement nearly all of the important tasks and decisions in an audit. The book provides a solid and concise founding in audit theory, along with practical implementations of auditing tasks and decisions using R-code.

Audit Analytics invokes a scientific perspective to support accounting and auditor opinions, based on treating audits as natural experiments. This perspective creates a statistical foundation for auditor hypotheses, evidence collection, and integration into decisions. *Audit Analytics* innovates in applying the latest machine learning models to identify subtle patterns in audit evidence that might escape traditional auditing methods. Rather than subjecting audits to intuitive assessments that can vary widely and inconsistently across auditors and audits, Westland applies modern methods in data science to make auditing a more rational, replicable, and reliable practice. In the best sense of the concept, Westland has learned how auditing can better race with the machines, using computers as allies rather than adversaries, concentrating on ways to accelerate organizational innovation and enhance human capital.

The growth of the digital economy has many consequences, among them a need for new and extended accounting systems. Income inequality and the rise in income of the world’s top earners are controversial consequences of our move to a digital economy. Governments, tax policy, corporations, and social institutions will never be effective in addressing these without complete and reliable accounting data. We are not there yet.

I conveyed in my books with Andy McAfee, *Race Against the Machine* and *The Second Machine Age*, as well as my articles with Tom Mitchell, the dangers to policymakers and managers when they “fly blind” without good data. This has consequences when knowledge and skills divides lead to ever-widening wage gaps and lifestyles. Challenges confronting contemporary economists—intangible assets, network effects, wage, and skills inequalities—present significant research problems precisely because there is so little reliable data.

Over the years, my own research agenda has faced the challenge of a paucity of reliable economic evidence on the value of technology and information assets. I have needed to construct alternative non-accounting metrics, procure datasets from independent survey and investigative organizations, and put forth conjectures on underlying economic mechanisms to fill gaps created by underreporting of financial information in corporate accounting reports. Yet, any scientific study in economics and finance will benefit from reliable accounting data. Auditors are key arbiters and guarantors of that reliability—they verify that accounting systems are functioning, and that transaction capture reliably mirrors reality.

I believe that our future is bright, but only if it is inclusive, scientific, and informed. Otherwise, we will choose unrealistic futures based on convenient fictions, invented lies, and debunked ideas. The danger is that we will manage ourselves toward larger wage and skills gaps, wasteful “investments,” and counterproductive policy. Reliable accounting and auditing are the first steps in realizing this future.

Westland’s book will not be the final word in this quest, but it provides a first step by addressing shortcomings, gaps, and weaknesses in our audit systems. These, in turn, assure that accounting data feeds the larger universe of financial and economic analysis with reliable data, assuring accurate, reliable, credible predictions and policy recommendations. With accounting reports that are accurate, reliable, and complete, economics can do what it does best—provide the big-picture, strategic insights that yield a richer, more inclusive and more meaningful world for all.

Peter Drucker observed over a half-century ago in the *Effective Executive* that “Working on the right things is what makes knowledge work effective. This is not capable of being measured by any of the yardsticks for manual work.” Audit Analytics offers a cornucopia of yardsticks, implementing technology that makes them accessible to auditors—guarantors of the data that fuels much of our knowledge work. Westland’s prose is entertaining, with stories and insights to keep you turning pages. The software implementations may be challenging, but to auditors as well as researchers like myself, who regularly confront large, sparse, poorly formatted business datasets, the advances proffered by this book promise a future completeness of reporting and reliability of accounting data that in the past would not have been conceivable.

Audit Analytics epitomizes the future of auditing. We can no longer afford to pave the old cow-paths by computerizing audit methods that originated in the nineteenth century. Rather the fundamental tenets of accounting and auditing must be revisited in light of advances in data science and information technology, to address the needs of modern investors and stakeholders. Both in structure and content, Audit Analytics does an excellent job of defining the new tenets of accounting and implementing software solutions. Read it, run the code, and apply the ideas to your own work!

Director of the Stanford Digital Economy Lab
Stanford Institute for Economic Policy Research (SIEPR)
Research Associate at the National Bureau of Economic Research (NBER)
Stanford Graduate School of Business
Stanford Department of Economics
Stanford, CA
January 2020

Erik Brynjolfsson

Preface

I began my career in 1973 as a staff accountant in the Chicago office of Touche Ross (a predecessor firm to Deloitte Touche Tohmatsu). Computers were new to accounting in those days. IBM had introduced its System 360 in 1964 (S/360 referred to the architecture's wide scope: 360 degrees to cover the entire circle of possible uses). By 1973, these were becoming fixtures in the accounting systems of large corporations. Prior to Touche Ross, I had earned a large portion of my college expenses providing assistance in the use of the Compustat tapes (they were actually physical 9-track tapes in those days and also came as decks of 80-column Hollerith punched cards) on a Control Data 6600. Touche Ross decided I was well-suited to teach and develop applications for their bespoke auditing software—STRATA (System by Touche Ross for Audit Technical Assistance) and this is largely what I did during the rest of my tenure at Touche Ross.

My Compustat work provided me a prescient foundation in statistical computing which I would later apply to my auditing work. In those days, people were still calling computers “electronic brains” reminiscent of today’s use of “artificial intelligence” to refer to machine learning. Applications of statistical computing to accounting were poorly taught, and reading matter on the subject was non-existent.

The Compustat story is itself worth retelling, as it played an important role in my education and has strong ties to Chicago. Compustat was the brainchild of Alfred Cowles III who came from an established publishing family; his father and uncle founded the Chicago Tribune and Cleveland Leader, respectively. For a short time after the First World War, Cowles successfully ran a Chicago investment firm that acquired and restructured small railroads. His firm also published a stock market newsletter providing fundamental analysis and recommendations on railroad stock issues as well as other investments. This was long before the Securities and Exchange Commission existed, let alone required annual audits and reliable financial reports. Ford Motor Company at the time, for example, only produced a balance sheet (unaudited) but not an income statement, despite being one of the largest firms in the USA. It was difficult to be a financial analyst in those days.

Diagnosed with tuberculosis in the late 1920s, Cowles consolidated his investments (just prior to the 1929 crash) and moved to Colorado Springs to improve his health. There, he filled his time developing linear regression models that simultaneously compared the predictions of 24 stock market newsletters to actual stock performance. Cowles quickly came to the conclusion that forecasters were guessing; that they offered little useful investment information and were more often wrong than right (he also applied his regression skills to investigate whether good climates, like Colorado Springs, improved the outcome of tuberculosis, with somewhat more optimistic results). The pen and paper calculation required at the time, for the regression formulas he used, soon exceeded his capabilities as a lone researcher. At this point, he made a decision to invest some of his fortune to create the Cowles Commission, an institution dedicated to linking economic theory to mathematics and statistics. To that end, its mission was to develop a specific, probabilistic framework for estimating simultaneous regression equations to model the U.S. economy. The Cowles Commission moved from Colorado Springs to the University of Chicago in 1939, where economist Tjalling Koopmans developed the systems of regression tools that Cowles originally had sought. This period also expanded Cowles personal files into what ultimately became the Compustat and CRSP databases and created the market index that eventually became the Standard & Poor’s 500 Index. Cowles researchers developed many new statistical methods such as the indirect least squares, instrumental variable methods, full information maximum likelihood method, and limited information maximum likelihood. Eleven Cowles associates ultimately received the Nobel Prize in Economics.

The days of bespoke auditing software have passed; platforms, systems, and standards are too varied and change too quickly for this small market to be attractive to developers. The good news is that readily available open-source software now has much more power, flexibility, and ease-of-use than has ever been available, and these can be applied in the support of audit tasks. Circa 2020, Python, Java, and the R statistical packages are widely used open-source platforms for data analytics; TensorFlow and PyTorch are open-source packages for machine learning; and open-source packages such as LibreOffice provide spreadsheet capabilities. There currently are no comprehensive texts for their application in auditing, a fact which

motivated the writing of this book. This book and its methods have grown out of an Audit Analytics course that I have taught at the University of Illinois—Chicago and previously at the Hong Kong University of Science and Technology. I have switched software platforms several times over the years but have settled on R, which has become my software package of choice for data analytics (I talk about motivations for my choice later in this book).

I am honored to be able to contribute my ideas and work in Gentleman, Hornik, and Parmigiani’s *Use R!* series. Robert Gentleman and Ross Ihaka developed the R programming language at The University of Auckland in the mid-1990s. Hadley Wickham, who holds adjunct professorships at University of Auckland, Rice University, and Stanford University, has played a major role in developing the `tidyverse` packages, which I use throughout this book, and J.J. Allaire’s `RStudio` has put the tools of data analytics within easy reach of nearly everyone. Researchers such as myself, who regularly confront large, sparse, poorly formatted business datasets, owe all of them multiple debts of gratitude for not only making our lives infinitely easier but also for making some of our research even possible at all.

The *Use R!* series is designed to make statistical computing tools and relevant algorithms readily available to practitioners. Since these books are written in `LaTeX` and R’s `Bookdown`, code for figures and tables are easily placed on a website for sharing. I assumed, in writing this book, that the reader has a basic background in statistics (for example, as would be offered in Dalgaard’s *Introductory Statistics with R*) and in data analytics (for example, as would be offered in Grolemund and Wickham’s *R for Data Science*) and with basic principles of accounting (for example, as would be offered in Walther’s *Principals of Accounting*). For the most part, I focus on the tasks faced by an auditor in public accounting, though offer some material that addresses other important topics in auditing.

How to Use This Book

The R language is not a typical language with a single core of developers and guidelines; rather, it is a sharing platform for a wide range of state-of-the-art data analytic functions—features which make it useful, dynamic, and sometimes messy. The effort put into familiarizing oneself with the R ecosystem will pay off many times over in access to the latest, most sophisticated algorithms that exist anywhere in industry.

The reader is assumed to have a basic knowledge of the R language, to be familiar with `help` screens, with online package support and documentation sites, and to be linked in with the large group of R experts at Internet resources like Stack Overflow and R-bloggers. These will be essential resources as the reader customizes the code examples offered in this text to their own particular needs. Much of the code in this text, particularly that presented in the Analytical Review chapter, depends on an ever-changing set of databases, streams, and standards. In some cases, packages will need to be loaded from GitHub rather than the official Comprehensive R Archive Network (CRAN) repositories; it is important that the reader consult these resources as well as CRAN.

In writing this book, I hope to provide the reader with an inventory of basic algorithms which can be easily expanded and customized to fit an auditor’s specific challenges. Accessible and comprehensive tools for these additional approaches are covered in this book, as are research approaches to take advantage of the additional explanatory power that these approaches offer to auditing. Coverage of software, methodologies, and fit statistics provide a concise reference for the conduct of audits, helping assure that audit opinions and decisions are defensible, fair, and accurate.

The book files and chapter code are available at <https://github.com/westland/auditanalytics> and this can be installed as an R package as well.

Chicago, IL, USA
January 2020

J. Christopher Westland

Contents

Foreword by Erik Brynjolfsson	v
Preface.....	vii
Fundamentals of Auditing Financial Reports.....	1
Auditing.....	1
Computers in Auditing and the Birth of Audit Analytics	1
The Roots of Modern Financial Accounting and Auditing	3
Al-Khwarizmi Algebra of Double-Entry.....	4
The Renaissance.....	6
The Industrial Revolution	6
The Birth of Modern Auditing.....	7
Public Accounting.....	8
Emerging Technologies and Intangible Assets.....	9
Financial Accounting.....	12
The Products of Accounting: Financial Statements	12
The Methodology of Accounting	14
The Accounting Process and Major Document Files	15
Code and Data Repositories for Audit Analytics	16
R Packages Required for This Book.....	17
References	18
Foundations of Audit Analytics.....	19
Business and Data Analytics	19
Accounting Data Types	24
Numerical vs. Categorical.....	25
Categorical (Enums, Enumerated, Factors, Nominal, Polychotomous) Data	26
Binary (Dichotomous, Logical, Indicator, Boolean) Data	28
Ordinal (Ordered Factor) Data	30
Data Storage and Retrieval	33
Further Study	38
R Packages Required for This Chapter	39
References	39
Analysis of Accounting Transactions	41
Audit Procedures and Accounting Cycles	41
The Origin of Accounting Transactions	42
Audit Tests as Learning Models	42
Working with Dates	43
Accounting Transactions	44
Couching Institutional Language in Statistical Terms	47
Transaction Samples and Populations	48
Accounting Cycles	49

Substantive Testing	50
Metrics and Estimates	50
Important Concepts in Probability and Statistics	52
Machine Learning Methods	58
Statistical Perspectives on Audit Evidence and its Information Content	60
Support and the Additivity of Evidence: The Log-Likelihood	60
The “Score”	61
Fisher Information	61
Reference	61
Risk Assessment and Planning	63
Auditing	63
Risk Assessment in Planning the Audit	63
Accessing the SEC’s EDGAR Database of Financial Information	65
Caveats on accessing EDGAR information with R	73
Audit Staffing and Budgets	76
The Risk Assessment Matrix	77
Using Shiny to create a Risk Assessment Matrix Dashboard	77
Generating the Audit Budget from the Risk Assessment Matrix	81
Technical Sampling Structure of the Audit Program	81
Sample Sizes for Budgeting	82
Notable Audit Failures and Why They Occurred	84
Auditing: A Wicked Problem	85
Final Thoughts on Audit Planning and Budgets	86
References	86
Analytical Review: Technical Analysis	87
Analytical Review	87
Institutional Context of Analytical Review	87
Technical Measures of a Company’s Financial Health	90
Purpose and Types of Ratios	90
Common Technical Metrics	91
Accessing Financial Information from EDGAR (https://www.sec.gov/edgar/)	92
Accessing Financial Information from EDGAR (https://www.sec.gov/edgar/) with the <code>finreportr</code> Package ..	95
Computing Technical Metrics	97
Visualization of Technical Metrics	98
Internet Resources for Analytical Review	100
US Census Data	101
R and Application Programming Interfaces (API)	104
Technical Analysis of Product and Customer News Sources on the Web	105
Vocabulary-Based Vectorization	106
References	112
Analytical Review: Intelligence Scanning	113
Intelligence Scanning of Internet Resources	113
Sentiment Analysis with Tidy Data	113
Scanning of Uncurated News Sources from Social Networks	121
Example: Extracting Tweets About General Motors and the Auto Industry	123
Intelligence Scanning of Curated News Streams	125
Accessing General Web Content through Web Scraping	131
Final Comments on Analytical Review with R	136
Design of Audit Programs	139
Audit Programs as Natural Experiments	139
Collecting and Analyzing Audit Evidence: Sampling	139
AICPA Guidelines on audit sampling	140
Sampling for Interim Tests of Compliance	141

Discovery Sampling	141
Coefficient of Variation Formulas for Sample Size	142
Rules of Threes to Calculate 95% Upper Confidence Bounds	142
Attribute Sampling	143
Acceptance Sampling	143
Acceptance Sampling with Poisson Data	146
A Statistical Decision Framework for Auditing.....	146
Audit Tests as Learning Models	147
Materiality	149
Risk	149
The AICPA on Sampling Approaches and Risks	150
AICPA Pronouncements on Generally Accepted Auditing Standards	151
Types of Sampling Allowed or Discussed by AICPA	152
Judgmental Sampling	153
Fixed-Interval Sampling	153
Cell or Random-Interval Sampling	153
Random Sampling.....	153
Conditional Sampling	153
Stratified Sampling	153
Monetary Unit Sampling.....	154
Transaction or Record Sampling	155
Non-statistical Sampling.....	155
Accounting Transaction Distributions	156
The Audit Cycle	157
The Context of Auditing and Information Technology	158
Auditors' Opinion: The Product of an Audit	159
References	160
Interim Compliance Tests	161
Interim Compliance Tests and the Management Letter	161
The SAS 115 Letter to Management	161
Three Methods of Sampling	162
Discovery Sampling	163
Statistics for Interim Tests	165
Attribute Sampling with t-Tests	168
Audit of Collection Transactions	171
Machine Learning Models for Audit of Controls	175
Autoencoders and Unbalanced Datasets	178
Final Thoughts on Machine Learning Applications in Auditing	187
References	187
Substantive Tests	189
Substantive Tests	189
Objective of Substantive Tests	189
Exploratory Substantive Tests	189
Creating Trial Balance Figures in One Step	194
Accounts Receivable Auditing	198
Footing and Agreeing to the Trial Balance	198
Tests of Supporting Evidence	200
Acceptance Sampling	202
Accounts Receivable Confirmation	204
Confirmations and Experimental Design	205
Audit Program Procedures for Accounts Receivable Confirmation	205
Timing of Confirmation Request	205
Confirming Prior to Year-End	205
Steps in Confirmation Process	206

Non-response to Confirmation Requests	208
Confirmation Responses Not Expected	208
Confirmation and Estimation of Error in Account.....	209
Post-Confirmation Tests	211
Probability Proportional to Size (PPS) Sampling.....	211
Estimation and Accrual of the Allowance for Doubtful Accounts	212
GLM-Exponential Regression	215
Time-Series Forecasting	216
Forecasting Accounts Receivable Collections	219
Calculating Allowance for Uncollectable Accounts	222
Stratified Samples and Monetary Unit Sampling	223
PPS	224
Stringer's Perspective on Monetary Unit Sampling	225
“Taintings” and the Poisson–Poisson Compound Distribution	226
The Audit of Physical Inventory	227
Periodic Inventory Systems	227
Perpetual Inventory Systems.....	227
Counting Inventory When Preparing Financial Statements	227
Inventory Systems	228
Physical Inventory (Counting) Process	228
Physical Inventory Count Versus Cycle Counts	229
Inventory Audit Procedures	229
Computer Analytic Workpaper Support	230
Footing, Reconcile the Inventory Count to the General Ledger	232
Cutoff Analysis	233
Duplicates and Omissions	234
Physical Count Exceptions to Perpetual Inventory Ledger	236
Test for Lower of Cost or Market (LOCOM)	237
Audit a Subset of High-Value Items	238
Inventory Allowances	239
Other Inventory Tests	240
References	241
Sarbanes–Oxley Engagements	243
The Sarbanes–Oxley Act: Security, Privacy, and Fraud Threats to Firm Systems	243
Academic Research on SOX Effectiveness	244
Evidence from Industry on SOX Effectiveness	245
Using R to Assess SOX Effectiveness in Predicting Breaches, and Identifying Control Weaknesses	246
Exploratory Analysis of the SOX-Privacy Clearinghouse Dataset	249
Using an Autoencoder to Detect Control Weaknesses	252
Preprocessing	253
Tensorflow Implementation of the Autoencoder	255
The H2O Implementation of Autoencoders and Anomaly Detection for Fraud Analytics	257
Anomaly Detection	265
Pre-trained Supervised Model	266
Measuring Model Performance on Highly Unbalanced Data	268
Fama–French Risk Measures and Grid Search of Machine Learning Models to Augment Sarbanes–Oxley Information	271
Fama–French Risk Factors	272
Final Thoughts on Sarbanes–Oxley Reports	277
References	278
Blockchains, Cybercrime, and Forensics	279
Blockchains for Securing Transactions	279
The “Block”	279
Hashing	279
Proof-of-Work (PoW)	280

Adding Transactions (New Blocks) to the Blockchain	281
Cybercrime and Forensics	283
Forensic Analytics: Benford's Law	287
References	290
Special Engagements: Forecasts and Valuation	291
Special Engagements for Assurance and Valuation	291
The Role of Valuations in the Market	291
Hi-Tech, High Risk	292
Strategy Drivers and Figures of Merit	292
Corporate Figures of Merit	293
The ROI Figure of Merit	293
The Profit Figure of Merit	293
The Sales Revenue Figure of Merit	294
Opportunity Costs	294
Valuation Models	295
The Behavioral (Historical) Model	295
Data: Transaction Stream Time Series	296
How Much Information Is in a Dataset?	297
Forecast Models	298
Discount Model	302
Terminal Dividend	303
Generating a Current Valuation	303
Other Approaches to Valuation	303
Real Options	303
Scenario Analysis and Decision Trees	304
Monte Carlo Simulations	304
Further Study	305
References	305
Simulated Transactions for Auditing Service Organizations	307
“Test Decks” and Their Progeny	307
Service Organization Audits	307
Accounting Cycles, Audit Tasks, and the Generation of Audit Data	308
Generation of Sales and Procurement Cycle Databases for A/B Testing of Audit Procedures	308
Setting Up the Simulation	309
Code and Data Repositories for <i>Audit Analytics</i>	310
Assumptions Used in the Generation of Simulated Data	310
Document Generation	311
Strategy for Document Generation	313
Statistical Assumptions and the Distribution of Accounting Transactions	313
General Parameters of the Simulation	314
The Sales Journal	315
Cash and Bank Deposits	318
Inventory and Purchase Orders for Inventory Replenishment	318
Inventory	319
Perpetual Inventory, Accounts Payable, and Other Inventory Related Accounts	323
Customer Credit Limits and Outstanding Accounts Receivable	323
Accounts Receivable	324
Accounts Receivable Aging	325
Employee Expenditures	326
Omissions, Duplications, and Monetary Errors in Transactions	327
Audit Tasks: Inventory and Accounts Receivable	332
Accounts Receivable Confirmations	333
Accounting Files for Audit	334
Auditing with Simulated Accounting Transactions	335

References	335
Index.....	337

Fundamentals of Auditing Financial Reports



Auditing

An audit is an independent examination of the records of an organization to ascertain how far the financial statements as well as non financial disclosures, present a true and fair view of the concern. It also provides assurance that the systems of record-keeping are well-controlled and accurate as required by law. Auditing has become such a ubiquitous phenomenon in the corporate and the public sector that academics started identifying an “Audit Society” (Power 1997).

The earliest surviving mention of a public official charged with auditing government expenditure is a reference to the Auditor of the Exchequer in England in 1314 (Matthews 2006). The Auditors of the Imprest were established under Queen Elizabeth I in 1559 with formal responsibility for auditing Exchequer payments. Modern concepts of “auditing” grew out of practices from the sixteenth century English manorial accounting. The medieval Latin term was “auditus compotī” from Latin *auditus* “a hearing,” at a time that literacy levels were low, and an official examination of accounts was presented orally.

Auditing initially existed primarily for governmental accounting and was concerned mostly with record-keeping rather than accounting procedures (Matthews 2006). It was not until the Industrial Revolution in the eighteenth century that auditing began evolving into a field of fraud detection and financial accountability.

The early twentieth century saw the standardization of ‘auditors’ testing methods and reporting practices and a growing role by banks and shareholders for public companies. Auditors developed a system for examining a representative sample of a company’s transactions, rather than examining each transaction in detail, allowing audits to be completed in less time and at lower costs. By that time, audit findings were regularly presented as standard “Independent Auditor’s Reports” accompanying a firm’s financial statements.

Statistical sampling of transactions is now the industry standard in performing audits. As a consequence, computer tools to conduct statistical samples and opinions based on them are an integral part of modern auditing. It is only when gross errors or fraudulent activities are uncovered that comprehensive audits are performed. As businesses have increased in complexity, such “risk-based” auditing has evolved to make auditing more efficient and economical. Risk-based auditing starts by assessing whether an audit is even needed, based on a review of information in the financial statements. If the review finds adverse trends or irregularities, audit scope may be expanded accordingly. Through audits, stakeholders may effectively evaluate and improve the effectiveness of risk management, control, and corporate governance.

In the USA, audits of publicly traded companies are governed by rules laid down by the Public Company Accounting Oversight Board (PCAOB), which was established by Section 404 of the Sarbanes–Oxley Act of 2002. Such an audit is called an integrated audit, where auditors, in addition to an opinion on the financial statements, must also express an opinion on the effectiveness of a company’s internal control over financial reporting, in accordance with PCAOB Auditing Standard No. 5.

Computers in Auditing and the Birth of Audit Analytics

When I received my undergraduate degree in statistics in 1971, the job market offered very little employment in statistics. Mainframe computers had just been introduced, and data resided on 80-column Hollerith cards. All data analyses were done on paper ledgers with adding machines. Microcomputers, cloud computing, and remote storage were yet to be invented. What a difference 50 years has made. For a growing number of people, data analysis is a central part of their job today.

Increased data availability, more powerful computing, and an emphasis on analytics-driven decision in business has created a plethora of jobs in data science. There are around 2.5 million openings (and increasing) for data analytics jobs in the USA at any time.

A significant share of employees analyze data with Microsoft Excel or other spreadsheet programs like Google Sheets. Others use proprietary statistical software like SAS, Stata, or SPSS that they often first learned in school. While Excel and SAS are widely used tools, they have serious limitations. Excel conflates metadata (headings and descriptions) with formulas and data; and all are called “cells” in Excel. Furthermore Excel is horrendously slow, cannot handle datasets above a certain size, and does not easily allow for reproducing previously conducted analyses on new datasets. The main weaknesses of programs like SAS are that they were developed for very specific uses in the mainframe era of the 1970s and 1980s and have missed out on many of the newest developments in analytics. Nor do they have a large community of contributors constantly adding new tools.

For those who have reached the limits of Excel, SAS, and their ilk, there is a next step: learn R or Python. R and Python are the two most popular programming languages used by data analysts and data scientists. Both are free and open source and were developed in the early 1990s—R (an open-source version of the “S” language) for statistical analysis and Python (after Monty Python) as a general-purpose programming language. For anyone interested in machine learning, working with large datasets, or creating complex data visualizations, both have become standard tools for analysis.

Corporate computing offers more data analytics jobs requiring Python. In contrast, in academe, consulting, and finance, R offers an expanded set of data cleaning and analysis tools that are useful in one-off data analysis. Learning either requires a significant time investment—particularly if you have never coded before. Python is better for data manipulation and repeated tasks, while R is good for ad hoc analysis and exploring datasets.

Python tends to be in demand in companies run by computer scientists and with a code base, partly because it is easy to learn once you know other languages. Python is most useful for relatively routine, predictable processes. From pulling data, to running automated analyses over and over, to producing visualizations like maps and charts from the results, Python is the better choice.

R tends to be in demand in consulting fields where every report is ad hoc. Neither is a particularly efficient language from a purely operational standpoint; for that you would turn to C++. Over the past 5 years there has been substantial competition in offerings of interactive development environments (IDEs) to promote the use of either language. The R IDE is called R-Studio and is managed by J. J. Allaire (originally from South Bend, IN, and who developed the ColdFusion web development software many years ago). Python is supported on NetBeans and Eclipse IDEs through plugins; but the go to Python environment is Anaconda, which has its own version of Python, and supports R as well as other programs.

R is good for statistics-heavy projects and one-time analyses of a dataset. R is also more difficult to learn, as many of its conventions assume you have a solid background in statistics. Python conceptually uses the same control structures and data types that you will find in other languages, which is why computer scientists prefer Python. They see R as being more of a statistical tool that also happens to have a language, rather than a well-designed language before anything else (as Python arguably is). R has *Reticulate* that allows any program written in Python to be used in R; it also has *Rcpp* (written and maintained by Chicago’s Dirk Eddelbuettel) that allows C++ programs to be used in R. Although powerful, R as a programming language may seem very unfamiliar and quirky to developers in more general-purpose languages.

No matter how brilliant your analyses are, if you cannot communicate them, they are worthless. Report writing, publication on webs, GitHub, and other outlets are all essential parts of being a data analyst. In support of this, R has the report writing tool called *knitr* (the author himself spells it with a lower case letter k). Python has its counterpart, *Jupyter notebooks* (which can be used with R now), and although they are conceptually similar, their intended purpose is quite different. Though not particularly well suited for exploratory analysis, *knitr* is a powerful tool for dynamic data report generation so much so that it is a worthy addition to any programmer’s toolbox.

For example, assume you have a CSV file with some entries that you want to present to someone (or yourself) in some graphical way on regular basis. You can open it in Excel, add a pivot table and graphs; but assume then that you need to combine this data with one or more XML files and some more data in a database—this becomes a scripting task. Excel can still help, but it would be laborious. Python can also accomplish this, but you will need to learn quite a few libraries before you are ready. But *knitr* and *R Markdown* can easily generate PDF reports with all the graphical and data-processing tools of R included. Additionally, R segues statistics with machine learning, which is important since the concepts of machine learning and artificial intelligence derive from statistics. Much of the Python-based literature on machine learning completely misses underlying statistical concepts and consequently fails to be as effective as R in developing machine learning algorithms. It is written by computer scientists who understand well the Python scientific stack, and who will demonstrate the steps in rote manner of how to achieve a certain goal, but would not be able to impart much insight into what happens behind the scenes.

Much of the popular press “artificial intelligence” news you may read is a product of hackers who may understand “how” to program something, but along the way seem to have missed the “why?” and “what?” In contrast, books by authors who actually teach statistical curricula on which the machine learning practice is based and which skillfully elucidate deep theoretical foundations, all seem to exclusively presume basic to intermediate knowledge of R. If you are planning to work in machine learning, R is a much more worthwhile investment than Python.

The preceding are some of my admittedly biased opinions. I think R is the better language; but with the caveat that “your mileage may vary” and you should choose your own path. I firmly believe, though, that the extra effort in learning R will be rewarded tenfold if you intend to work in data analytics.

The Roots of Modern Financial Accounting and Auditing

Accounting and auditing are ancient, with many modern practices that are rooted in traditions that date back millennia. This section details the evolution of accounting, with explanations of how historical developments live on in idiosyncrasies of accounting and auditing today.

The earliest extant record of humans keeping track of numbers are the Lebombo Bones (several dozen have been found to date), which have been carbon dated to about 35,000 BCE. These are arguably the oldest known accounting computers. They are tally sticks with counting notches carved into a baboon’s fibula, found in the Lebombo mountains located between South Africa and Swaziland.

In the middle of the ninth millennium BCE, communities began coalescing around marketplaces that eventually became the cities of antiquity. Accounting was practiced in the ancient Middle East by means of small counters—tokens modeled in clay in different shapes, each symbolizing a particular commodity. Their shapes include spheres, flat and lenticular disks, cones, tetrahedrons, and cylinders. These forms seem fully arbitrary and were decided based on the requirement of least production effort. They were easy to identify and replicate and embodied daily life commodities.

With stylized signs, all information could be recorded directly on clay tablets, which were cheaper and more portable than tokens. The earliest scribes served as lawyers and accountants in the early cities. They would record trade transactions, debts, and advances, in duplicate on clay tablets to be retained by the parties to the transaction. The earliest texts were pictographs on tablets written with a stylus, with accounting-specific writing systems evolving into more expressive systems that described politics, religion, and news.

Many commodities were exchanged in the early barter communities—silver, grain, and so forth based on standardized weights, and thus establishing an absolute value for any particular item was complex and inexact. Under King Croesus of Lydia (in modern Turkey), the *touchstone* (mined from local riverbeds) was used to standardize the content of gold alloys, allowing standardized coinage and a single denomination for the price of any item. The earliest coins were blank droplets of metal of standard weight, issued by the treasury; later coins were stamped with punches for easy accounting. It was the custom of unmarried ladies in Lydia to sell their virtue in exchange for coins, in order to accumulate a dowry sufficient for them to marry. In this regard, accounting can credibly lay claim to being “the oldest profession.”

The Kingdom of Lydia was short lived; but their idea of standardized coinage lived on, first in the Athenian silver drachma, and later the Roman Solidus, which provided Mediterranean trade with an unprecedented assurance of quality and value for a numéraire commodity—one which could be traded with all others. Issuing standardized coinage was a profitable business for government. Governments essentially take out non-interest bearing loans on the money they mint; in turn, this begot profitable industries in money changing, accepting deposits, and lending money at interest. Rome developed sophisticated financial and contracting systems for transport and trade, far-reaching tax collection, and a vast bureaucratic network.

Parallel developments in Asia saw China evolve from Cowry shells as standard coinage in the fifth century BCE, to copper coins in the Qin Dynasty period (third century BCE). Qin coins were practical—round with a square hole in the middle (for stringing together), which remained the common design for most Chinese copper coins until the twentieth century. Because of the low value of the coinage, seigniorage and government profits were substantial, as was the tax on transactions. The Qin Dynasty took a no nonsense attitude toward executing anyone caught making transactions with local coins, or any currency not sanctioned by the government; a policy revived in the Tang Dynasty when they introduced paper (tree bark) currencies.

The Roman Empire sank into medieval chaos after about the third century. Predicting the collapse of civilization, Boethius and Cassiodorus in the sixth century collected existing studies in mathematics (including accounting) into the quadrivium, comprising the four subjects of arithmetic, geometry, music, and astronomy. These were to be taught to Roman patricians in the monastic schools, which before Saint Benedict of Nursia were more or less safe country estates for offspring of the Roman elite. Until the fifteenth century, European accounting was static and unchanged from the Roman practices. Meanwhile in the Middle East, firstly the Umayyad Caliphate, and then the Abbasid Caliphate of Baghdad rapidly gained

power over Central Asia by adopting our modern Arabic number system from India. This was a substantial improvement over the cumbersome notation of the Romans. It allowed easier additions and subtractions, as well as allowing accounting to extend itself to multiplication (for prices and cost allocations) and division (for rates). Baghdad became the world's center of accounting innovations.

Al-Khwarizmi Algebra of Double-Entry

In the Islamic world, the word account took on religious significance, relating to one's obligation to account to God on all matters pertaining to human endeavor. According to the Qur'an, followers are required to keep records of their indebtedness. Thus Islam provides general approval and guidelines for the recording and reporting of transactions. The Islamic law of inheritance defines exactly how the estate is calculated after death of an individual. The power of testamentary disposition is basically limited to one-third of the net estate (i.e., the assets remaining after the payment of funeral expenses and debts), providing for every member of the family by allotting fixed shares not only to wives and children but also to fathers and mothers. Clearly this requires ratios, multiplication, and division that were well beyond the scope of Roman numerals and abaci.

The complexity of Islamic inheritance law served as an impetus behind the development of algebra by medieval Islamic mathematicians. Al-Khwarizmi's "The Compendious Book on Calculation by Completion and Balancing" devoted a chapter on the solution to the Islamic law of inheritance using linear equations. In the twelfth century, Latin translations of al-Khwarizmi's "Book of Addition and Subtraction According to the Hindu Calculation" on the use of Indian numerals, introduced the decimal positional number system to the Western world. Hindu–Arabic numerals and algebra were introduced to Europe from Arab mathematics at the end of the tenth century by Pope Sylvester II and in the twelfth century by Fibonacci.

Al-Khwarizmi (Latinized as *Algorithmi* from which we derived the word "algorithm") introduced algebra (from the Arabic *al-jabr* meaning "restoration") to accounting, leading to three fundamental *accounting – algebraic* concepts:

1. *Debits = Credits*: The "Bookkeeping equation" for error control, which is the accounting equivalent of algebraic manipulations on the left-hand and right-hand side of an equation.
2. *Assets = Liabilities + Owner's Equity*: "Real" accounts for tracking wealth, and the "basic accounting equation." An elaborate form of this equation is presented in a balance sheet that lists all assets, liabilities, and equity, as well as totals to ensure that it balances.
3. *Closing process* where "Nominal" accounts for tracking activity are closed to *Owner's Equity*: Closing out these accounts at year end yields the *net income* (the owner's increment in wealth)—arguably the most important single statistic produced in the accounting process.

Algebra manipulates formulas around an equal sign, the only constraint being the formula on the right of the equal sign must have the same value as the formula on the left. Double-entry bookkeeping manipulates debit and credit balances around an equal sign, the only constraint being that the debits must have the same value as the credits. Accounting is applied algebra.

Though not specific in this regard, Al-Khwarizmi's book hinted at what were to become the standards for later accounting: error control (through double-entry), nominal accounts (which appear on the Income Statement) and real accounts (which appear on the Balance Sheet). Nominal accounts are revenue or expense accounts that are closed to a zero balance at the end of each accounting period. They start with a zero balance at the beginning of a new accounting period, accumulate balances during the period, and return to zero at the year end by means of closing entries. Nominal accounts are income statement accounts and are also called "temporary accounts" in contrast to balance sheet (asset, liability, and owners' equity) accounts that are called "permanent accounts" or "real accounts." Real accounts are asset, liability, reserve, and capital accounts that appear on a balance sheet. The balances of real accounts are not canceled out at the end of an accounting period but are carried over to the next period.

Al-Khwarizmi's method was an immediate hit and was widely disseminated throughout the educated world. In 756, the Abbasid Caliph Al-Mansur sent 4000 Arab mercenaries to assist the Chinese Tang Dynasty in the An Shi Rebellion. After the war, they remained in China and established an alliance with the Tang court where they were known as the Black-robed Arabs. The Tang Dynasty brought in Arab scholars and adopted Abbasid innovations in money and accounting. Trade flourished: the Tangs reopened the Silk Road and hugely expanded their maritime presence into the Persian Gulf and the Red Sea, into Persia, Mesopotamia, (sailing up the Euphrates River in modern-day Iraq), into Arabia, Egypt, Ethiopia, and Somalia in the Horn of Africa. They established the most extensive taxation system to date. Each citizen was taxed on iron and salt usage and owed the Dynasty corvée labor of 20 days a year. The state practiced the "equal-field system" in which most land was state owned and granted to individual farmers to prevent the formation of large estates. This allowed greater

government control over the individual farmers but required a huge professional bureaucracy, including accountants selected by Imperial examination. The banknote was first developed in China in the Tang Dynasty during the seventh century, with local issues of paper currency. Its roots were in merchant receipts of deposit as merchants and wholesalers desired to avoid the heavy bulk of copper coinage in large commercial transactions. The Tang Dynasty paid local merchants with money certificates called “flying cash,” because of its tendency to blow away. These certificates bearing different amounts of money could be converted into hard cash on demand at the capital. Since they were transferable, they were exchanged among merchants almost like currency. The threat of penalties and possibly execution also encouraged merchants to use “flying cash.”

Abbasid and Tang innovations in accounting would have been much less influential were it not for the rise of the Mongol conqueror Genghis Khan and later his grandson Kublai Khan who were deeply influenced by the bureaucracy of the Tang Dynasty. Genghis and his progeny nearly conquered the known world in the thirteenth century. Germany lost nearly a million soldiers (most of its adult males) defending Europe against their invasion. Genghis Khan gave his accountants an unprecedented position of power in the Mongol Empire. When a city was sacked by Mongols, the accountants were the first to enter, tallying up the total asset value of the city (before soldiers could loot or pillage) from which the Mongols took 10%, to be allocated between the troops on well defined principles. Conquered cities were subjugated, then encouraged to remain going-concerns, so that they could be reliably tithed—in this the accountants also played a pivotal role.

The Caliphate of Baghdad had provided fertile ground for the advancement of mathematics at a time that Europe was stuck in the quadrivium. Many of the ancient libraries of Rome and Greece had been transferred to Baghdad, where scholars integrated the Hindu numerical notation into geometry and the algebra of linear and quadratic equations. These works were disseminated throughout the Abbasid Caliphate, including into Moorish Spain, making this body of knowledge accessible to Europeans in the colleges at Granada, Cordova, and Seville around the beginning of the twelfth century. The Italian mathematician Fibonacci became exposed to this knowledge traveling extensively throughout Egypt, Syria, Greece, and Sicily. He eventually returned to Italy to publish his *Liber Abbaci* (book of calculation), which promoted the use of Arabic numerals for calculations. The book was written in Latin during the year 1202 and was influential and widely read.

The introduction of double-entry bookkeeping into Europe is an interesting one in itself. Arabic numerals were known in Europe, but it was considered sinful to use them. For example, the statutes of the *Arte del Cambio* of 1299 prohibited the bankers of Florence from using Arabic numerals (even though they had been slipping them into documents since the tenth century just to ease calculations). Pope Sylvester II who reigned for three years at the turn of the millennium, endorsed and promoted study of Arabic arithmetic, mathematics, and astronomy, reintroducing the abacus to Europe. After his death, he was widely denounced as a sorcerer in league with the devil for exactly this reason.

It was into this politically and religiously charged atmosphere that the innovations in Fibonacci's *Liber Abbaci* started with merchants and bankers synthesizing alternatives to the abacus and daybooks of unwieldy Roman numerals. Fibonacci himself was a merchant as well as a mathematician. He traveled extensively throughout the Mediterranean as a merchant prior to 1200. Not only did he encourage adoption of the Hindu–Arabic numerals for commercial accounting, but he actually set out an account contrasting completely the Roman figures versus the Arabic numerals.

Importantly, the Hindu–Arabic system mathematics also covered systems of equations, and these were in addition introduced into Spain by the Moors. From an accounting perspective, the conceptualization of debits and credits provided a generalization at the accounting systems (financial statement) level in addition to details at the transaction (journal entry) level. Equations—by definition—require an equality of two sides of the equation—on one side dependent variables, on the other independent. In an exchange transaction there is a natural dichotomization into independent variables (assets) and dependent variables (claims). These are reflected in the Latin words *debere* (to owe) and *credere* (to entrust), which are the basis for our modern words, *debit* and *credit*.

The Hindu–Arabic systems of equations were focused on equilibrium—in the balance sheet at any date, an equilibrium of exchange transactions is preserved. The use of an equilibrium device may at first have been pro forma. But it is easy to see how the change from ad hoc single-entry to equilibrium bookkeeping could lead to a complete double-entry system (particularly given European bankers' early penchant for slipping Arabic numerals into their financial records, despite the fact that they might be accused of colluding with the devil). Once figures began to be disposed in a single column, instead of being scattered all over the page and reduced to order only outside the account-book on the abacus or in the mind, then the advantages of having two clearly separated columns, simply to facilitate computation, would quickly become apparent (De Roover 1956, 1955, 1938).

The Renaissance

For two centuries, during the rise of the great Italian banking centers of Genoa, Florence, and Venice, Italian banking recorded transactions, made loans, issued scripts, and carried on with numerous other financial activities that we would recognize today. Initially they used single-entry recording (date, and account affected), but this proved error prone as transaction volume increased. Fibonacci's *Liber Abbaci* was widely read, and influential. Giovanni di Bicci de' Medici introduced double-entry bookkeeping for the Medici bank in the fourteenth century. By the end of the fifteenth century, merchant ventures in Venice used this system widely.

The fifteenth century also saw the invention of the printing press in Germany, and affordable, widely available reading glasses from Venetian glassmakers. Leo X's Vatican was an enthusiastic customer, where printing presses were used to churn out indulgences, and subsequently, Italy became a center of printing, book publishing, and literature. One of the first great texts on mathematics was published by a close friend of Leonardo da Vinci, Venetian Luca Pacioli, in 1494. *Summa de Arithmetic, Geometria, Proportioni et Proportionalita* (Everything About Arithmetic, Geometry and Proportion) formally described the widely used but still controversial system of double-entry for a much wider audience. Pacioli's Summa included one chapter titled, Details of Accounting and Recording that popularized the Method of Venice for accounting (an evolution of Al-Khwarizmi's method). Pacioli's Summa described the components of bookkeeping as: a memorandum book, journal, and ledger, with the journal and ledger similar to modern equivalents. A trial balance was used when the books were closed. The profit or loss was entered into the capital account to balance the balance sheet. Thanks to Gutenberg's printing press, Summa was published throughout Europe.

Pacioli's Summa was translated into the most commonly read vernaculars in Europe and was influential not only in investment and merchant ventures but in accounting in the great estates of Europe. Over the next 400 years it became the standard, in only slightly modified form, for accounting in all realms of business in Europe. Before the Industrial Revolution, China and India were the wealthiest economies in the world. Mughal India's annual revenues were twenty times that of France. In China, Qianlong demanded that Britain's ambassador Lord McCartney kowtow and informed him that China was in need of nothing from the West.

The Industrial Revolution

Europe was only a minor economic player in the world at the start of the Industrial Revolution in the late eighteenth century. England, and specifically Charles II, set out to change that. After the death of Oliver Cromwell in 1658, which resulted in his restoration, one of Charles II first tasks was to set up a body to move mathematics and the science forward in England. The Royal Society of London for Improving Natural Knowledge, known simply as the Royal Society, was founded in November 1660. Members of the Royal Society were initially chosen almost evenly from Parliamentarians, and Monarchists (leading some to believe that the Royal Society may have been a Masonic society). England was the first European country to place science and mathematics at the center of national policy. This was not merely an abstract goal. As a country with growing colonial and industrial ambitions, it needed better tools for navigation (clocks, sextants), for business (accounting, patents, methods), and government (statistics).

Modern businesses grew out of the cottage industries of the day. Home sewing grew into textile mills; mines became larger and more efficient. Goldsmiths' safes were used to hold the monetary deposits of merchants and others, using gold notes as receipts. These receipts evolved into banknotes and these new bankers loaned money and performed other banking tasks. Adam Smith chronicled the merits of specialization. Stock in businesses allowed operations to grow without limit as the technology became available for greater scale. The Bank of England was founded as a joint stock company and became the central bank of Britain.

Much of Britain's wealth depended on textiles and specifically the wool trade. The wool of British sheep was highly priced, and sheep owners in the Cotswolds and other areas became rich. Instead of selling all wool to continental merchants, the domestic system of textile manufacturing developed. Small farmers did the spinning, cleaning, and weaving in their homes. About half of the manufactured cloth was exported and a wealthy merchant class developed.

At this time manufacturing productivity levels were similar around the world and changed little over time. But the Industrial Revolution inched Britain's productivity growth up an average about 2% a year, pushing it to the largest economy in the world at the end of the nineteenth century. The factory system was invented for the textile industry. Power-driven machines required an army of laborers, working long hours at monotonous tasks in unsafe conditions for low wages. By 1800 most cotton manufacturing was done at factories. Cotton provided about half of Britain's exports well into the nineteenth

century. By 1850 over 1900 factories were in the cotton industry with 330,000 workers, about 85% using steam and 15% using water power. The human toll was excruciating—the average age of death for a factory worker in 1800 was 17 (Bryson 2010).

The potter Josiah Wedgwood was keenly interested in the scientific advances of his day. It was this interest that underpinned his adoption of its approach and methods to revolutionize the quality of his pottery. His unique glazes began to distinguish his wares from anything else on the market. He was perhaps the most famous potter of all time. He built a successful business, but the depression years of 1770–1772 were financially difficult for Wedgwood—demand dropped, inventories rose, and prices had to be cut. He also found that his clerks were embezzling funds, while ignoring debt payments and other paperwork.

Wedgwood turned his creative genius toward a detailed examination of his company's transactions. He was interested in developing a system that would allow him to quickly discover inefficient operations and to identify sources of overhead. In the process, he discovered a long history of embezzlement by his head clerk. This resulted in a new clerk and weekly accounting reviews. Eventually, Wedgwood was able to calculate detailed costs for materials and labor for each step of manufacturing for each product. Overhead costs were analyzed and allocated to specific products. Wedgwood's early work in cost accounting influenced the understanding of industrial economics, as businessmen discovered that costs to produce some products were considerably more than for others, with profound effects on profitability. Economists evolved concepts of economies of scale and sunk costs. Wedgwood's own factory started to differentiate markets. Based on his cost accounting, demand became a primary factor for production and pricing. The market could be divided between high-price, high-quality, and high-cost products for elite customers, while a mass market was developed with low-cost and low-price pottery. Technology and mechanization were given roles in economic scaling. Wedgwood's cost accounting innovations allowed him to survive economic downturns where competitors could not.

The Birth of Modern Auditing

At the start of the nineteenth century, eleven Londoners listed their occupation as the archaic “Accomptants.” As industry, mass transportation and capital markets expanded so did the need for accountants. Business regulation increased, and industry was often taxed heavily, promoting the need for professionals. The Bankruptcy Act of 1831 allowed accountants to be appointed “Official Assignees,” the first government recognition of the new profession. A primary role became the preparation of accounts and the balance sheet of public companies. Bankrupt firms were especially likely to use their services, which increasingly served an audit function.

The British Companies Act of 1844 established the incorporation of business by a formal registration process. It required annual appointment of auditors to examine the accounts and balance sheet of all public companies (the role of accountants under the British Companies Act would change substantially over the century). The Companies Act of 1862 required banks to be audited and established the practice of limited cash dividends to be paid only out of profits. By 1900, the audit was the central practice of accountants. The earliest of the *Big Six* accounting firms were started in mid-nineteenth century London. William Deloitte opened a London firm in 1845. Samuel Price and Edwin Waterhouse formed their partnership in 1849. William Cooper started his firm in 1854, to be joined by his brothers in 1861. William Peat started in 1867. These men were active in establishing the Institute of Accounting in the 1870s, and a royal charter was granted in 1880. With the Institute and professional requirements to become Chartered Accountants, the profession of accountants was firmly established.

Many of the current methods in cost accounting were developed after 1880. The *Scientific Management* analysis of mass production was a major factor. Engineers using job analysis as well as time and motion studies determined “scientific” standards of material and labor to produce each unit of output. Complex machines required complex engineering and efficient use of workers to perform specialized and repetitive tasks. Standard costs became a significant efficiency measure. Frederick Taylor analyzed the best ways to use labor and machines, and standards were determined to minimize waste. The focus was on cost cutting rather than product quality. Actual costs could be compared to standard costs to measure performance, and the variances between actual and standard costs analyzed to determine potential corrective action. Measuring and allocating overhead costs also were major concerns of *Scientific Management*.

Prior to the 1950s, management used cost accounting information primarily for planning purposes; while operating control normally was based on non-accounting information. For example, Du Pont used ROI for planning, but control was based on factors such as timeliness of delivery to customers and product quality. Multidivisional business organizations became more pronounced, and businesses turned to decoupling—different production processes took place at different sites, perhaps thousands of miles away.

Public Accounting

Until the nineteenth century, firms tended to be small and raised money from partnerships, short-term bank loans, or profits. New industries from the early nineteenth century including canals, railroads, and utilities needed large amounts of capital from private sources. The English and American organized exchanges which provided the mechanisms for the needed cash. Parliament regulated business and capital markets through British Companies Acts. American regulations were based on state laws, which were not particularly effective, yet by the turn of the twentieth century, America overtook Britain as the leading industrial power in the world. Financiers provided the capital to establish large corporations and combinations and, as demonstrated by J. P. Morgan, maintained substantial power over industries. Insiders benefited from price fixing, stock manipulation, and various schemes of questionable legality. Accountants usually aided managers and speculators. Mergers, cutthroat competitions, railroad rebates, and bribery were some of the techniques used by businesses. Beginning with Standard Oil, the trust was used to acquire businesses across state lines. By 1890, 300 trusts controlled 5000 companies. By 1900, the largest dozen trusts had capital of over \$1 billion. The investment bankers became directors (78 companies at J. P. Morgan).

Government regulation was demanded by the public, often subject to monopoly pricing and various predatory practices alone. Federal regulations began with the Interstate Commerce Commission (ICC) Act of 1887 to regulate railroads. Federal antitrust began with the Sherman Act of 1890. Following the panic of 1907, the USA experienced a decade of reform in legislation. The Clayton Act of 1913 prohibited interlocking directorships. The Federal Reserve was established the next year, as was the Federal Trade Commission (FTC).

Disclosure of financial information was voluntary. During the nineteenth and early twentieth century, the balance sheet was paramount. The income statement was neglected because it was open to abuse (no accounting standard regulations existed), and the concept of earnings power had to wait to the post-World War I period. At the turn of the century, the stock exchanges were dominated by the railroad industry (60% of NYSE firms at 1900, for example). They were considered safe investments because they paid fairly consistent dividends—ignoring bankruptcies that did occur in this industry. Industrial firms did catch on in the 1920s (44% of NYSE firms in 1920), and the income statement became more important as industrials tended to pay dividends based on earnings. Unfortunately, the Roaring Twenties also was a time of rampant speculation, with securities bought on credit and with little regard to underlying earnings power. Until 1910, the NYSE had an “unlisted department” for firms that disclosed no financial information. Although 90% of NYSE firms had audits in some form by 1926, audited financial statements were not required until 1933—when stock prices stood at 10% of 1929 highs. Insider trading was common and not illegal; “preferred list” sales of new securities at discounted prices were made before the issues went public; syndicated stock pools manipulated stock prices.

The U.S. economy, which hinged on farming, had been in trouble since the early 1920s. Waves of immigrant farmers plowed increasingly marginal land, creating dustbowls, and impoverishing local banks that lent to them. But since there were no national records, knowledge about the problem was scarce. When the stock market crashed, the regulators panicked. The Federal Reserve cut back the money supply, turning a recession and stock panic into a major depression. From 1929 until 1932, 11,000 banks failed, gross national product fell about 10% a year, steel production dwindled to 12% of capacity, and the unemployment rate hit 25%. The Hawley–Smoot Tariff Act dramatically increased tariffs, with effect being to limit world trade—which effectively made this a global depression. Franklin D. Roosevelt was President Hoover’s opponent in 1932, promising a New Deal. He won in a landslide and delivered. During the first 100 days, a remarkable amount of legislation was passed (some of it a holdover from the Hoover administration). The Glass–Steagall Act separated commercial from investment banking and created the Federal Deposit Insurance Corporation to insure bank deposits; the Social Security Act established retirement and disability pensions funded with payroll taxes.

New Deal legislation led to federal responsibility for protecting investors from malpractice in the investment markets with the Securities Act of 1933. The Act was modeled on state regulations, British Companies Acts, and earlier Congressional legislation. The Securities and Exchange Commission (SEC) Act of 1934 created the SEC to administer the legislation. The Securities Act required companies to present registration statements with new public offerings of stocks, bonds, and other securities, to make “full and fair” disclosure of financial information. Information relevant to the “prudent investor” was to be disclosed in the registration statement. Antifraud and liability regulations increased the legal responsibilities of accountants, who became liable to the public as well as the management of the firms audited. Modern financial reporting and the development of generally accepted accounting principles (GAAP) started with this federal legislation. The SEC has the authority to regulate accounting. This was delegated to the private sector and the American Institute of Accountants (AIA—later the American Institute of Certified Public Accountants or AICPA). It gave the Committee on Accounting Procedure (CAP) the responsibility to issue accounting standards in 1938. The CAP’s standards were called Accounting Research Bulletins (ARBs) and from 1938–1959, 51 ARBs were issued. Few ARBs are still in effect, but the basic structure of financial

accounting has not changed much from these initial standards. Both the accounting profession and the financial community cited the CAP and this was replaced by the Accounting Principles Board (APB) in 1959. This was another AICPA committee and had many of the same problems. The APB would issue 31 Opinions until they were superseded in 1973.

Emerging Technologies and Intangible Assets

Information technology plays a pivotal role in financial control and audit today. Most, if not all financial data is now digitally recorded, and dispersed among servers, clouds, and networks of computers, over which the audited firm has no control. Additionally, firm data particularly in finance, IT, insurance, and medicine now comprise most of the audited value of the firm. Financial audits are critical mechanisms ensuring the integrity of information systems and the reporting of organizational finances. They help avoid the abuses that led to passage of legislation such as the Foreign Corrupt Practices Act, and the Sarbanes–Oxley Act. They help provide assurance that International Accounting Standards are consistently applied in calculating firm value, while controlling asset market bubbles. Expanding audit challenges have fueled a growth market in auditing throughout the past decade. Both undergraduate and graduate accounting enrollments are at their highest level in the past 40 years, with accounting students making up 40% of business school enrollments (Fig. 1).

Graduate degrees are increasingly important to the profession, around 40% of accounting graduates hired by public accounting firms have graduate degrees. The second largest group and hired into public accounting had degrees in information technology and computer science. The most prestigious of the accounting firms, the largest accounting firms, the *Big Four*, have increased their intake substantially (Table 1).

The financial information sector of the USA and world economies are major contributors to jobs' growth and wealth. In 2018, it represented around 15% of the US economy (about 2 trillion dollars) and around 10% of the global economy (around 7 trillion dollars). In addition to accounting firms, large banks, rating agencies, funds, private investment, and many other fields require accounting expertise to function competitively in current markets. Increasingly that means expertise in information technologies as well.

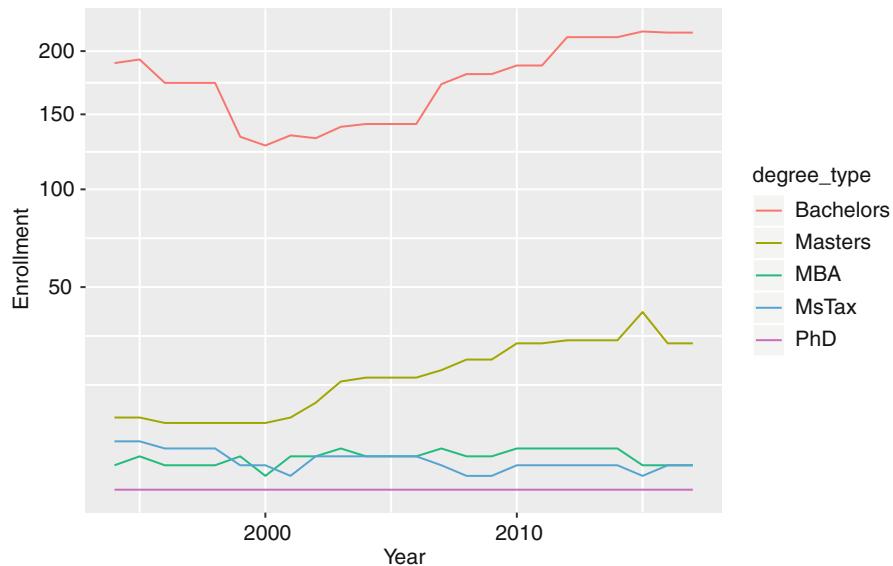


Fig. 1 Enrollments in audit degree programs, USA

Table 1 The 'Big Four' accounting firms (y/e 2018)

Firm	Revenues	Employees	Rev. per. emp	Headquarters
Deloitte	\$43.2 bn	286,200	\$150,943	United Kingdom
PwC	\$41.3 bn	250,930	\$164,588	United Kingdom
EY	\$34.8 bn	260,000	\$133,846	United Kingdom
KPMG	\$29.0 bn	207,050	\$139,870	Netherlands

Auditors have been grappling for many decades with calls to increase the scope of their audits to non-traditional measures of corporate performance financial forecasts, brand valuations, valuations of intangible assets, social accountability, and so forth. For the most part, they have rebuffed any efforts to expand auditing because of concerns over objectivity and the ability to provide sufficient reliability of their opinions (Aboody and Lev 1998; Lev 2001, 2004).

The stock market valuation of the Fortune 500 companies (which include many traditional industries such as oil and autos) has grown increasingly higher than the accountants book valuations over the past three decades. This is reflected in the growth of intangible assets, which may be loosely defined as valuable things the firm owns, like patents, skills, and so forth, that are not explicitly assessed in an audit, nor presented on the financial reports.

Tobin's q is the ratio between a physical asset's market value and its replacement value. It was first introduced by Kaldor (1966) and popularized by Nobel Laureate James Tobin. Values are supplied in the Federal Reserve Z.1 Financial Accounts of the United States, which is released quarterly. The average (arithmetic mean) q -Ratio is about 0.70. The chart below shows the q -Ratio relative to its arithmetic mean of 1 (i.e., divided the ratio data points by the average). This gives a more intuitive sense to the numbers. For example, the all-time q -Ratio high at the peak of the Tech Bubble was 1.61—which suggests that the market price was 136% above the historic average of replacement cost. The all-time lows in 1921, 1932 and 1982 were around 0.30, which is approximately 55% below replacement cost. The latest data point is 55% above the mean and now in the vicinity of the range of the historic peaks that preceded the Tech Bubble (Fig. 2).

Since the 1980s, accounting has evolved to focus less on physical things and more on intangibles and information. Information is increasingly the asset of value in firms, and IT provides the tools to manage, process, and deliver information products and services. In 1980, most of firm value was comprised of physical goods accounted for in the financial statements; by 2018, 40–80% of firms value was in unmeasured *intangible* assets. Much of this *intangible* value is held in databases, intellectual property, proprietary processes, and individual expertise.

Historically, book value was relatively accurate in assessing the value of a firm—up until the 1970s. The 1970s introduced “financial engineering” as exemplified by the conglomerate fad initiated by firms like LTV and Textron. Firms became harder to value from both an investment and an accounting perspective. The 1970s was also a period when Treasury rates approached 15%, and firms were feeling new competition from Asia. This pushed many firms into a negative equity position, which was only remedied by the availability of easy money in the 1980s. Since then, the growth of stock market valuations have steadily outpaced book value, at least partly reflecting the growth of intangible “knowledge” assets that are of value to investors but poorly tracked in accounting systems.

Information technology now plays a pivotal role in financial control and audit: most, if not all financial data is now digitally recorded and dispersed among servers, clouds, and networks of computers over which the audited firm has no control. Intellectual property, accounting, finance, IT, and risk data now comprise most of the value of the firm. Financial audits are critical mechanisms ensuring the integrity of information systems and accurate reporting. Over the past 30 years, the information intensive parts of the economy have been consistently more profitable than industries requiring heavy investment

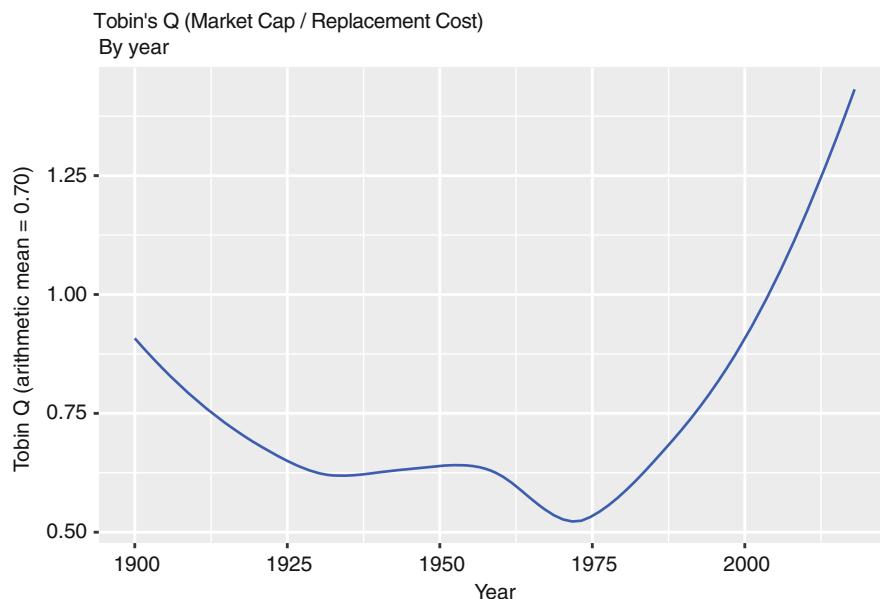
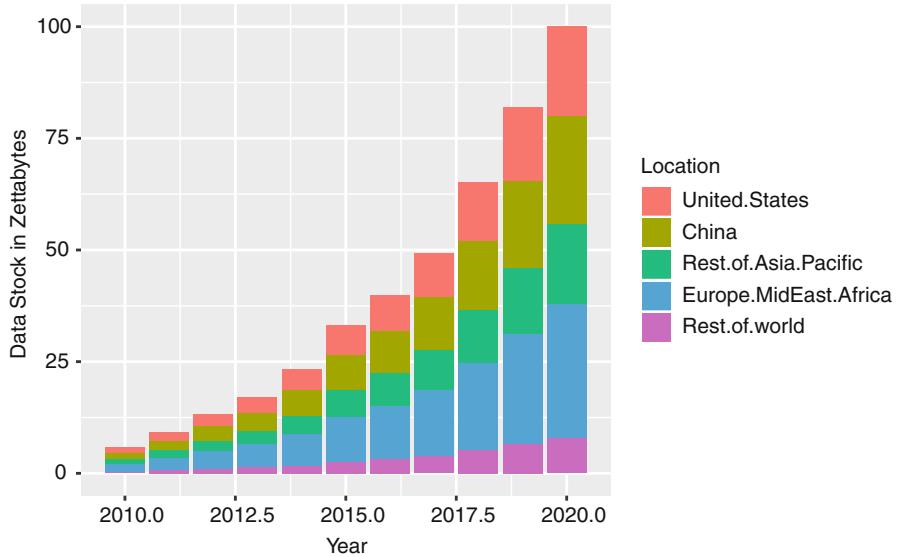
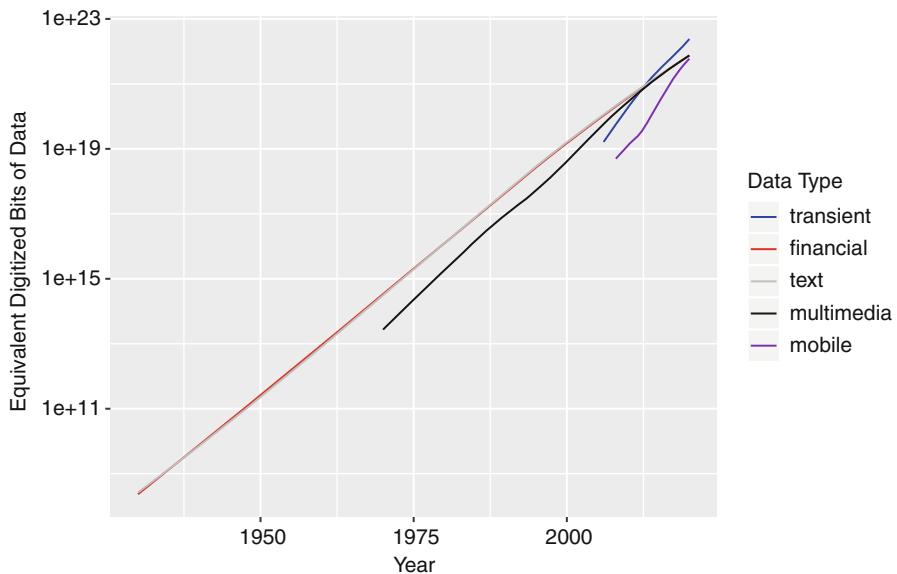


Fig. 2 Tobin's q (market cap/replacement cost)

**Fig. 3** Global stock of data assets (source: IDC)**Fig. 4** Growth of data (source: EMC digital universe project)

in machinery and other fixed assets. Conclusion: return on investment is strongly correlated with investment in intangible assets.

The value of all the data in America is between \$1.5 and \$2 trillion, or around 5% of America's stock of private physical capital. The amount of data in the world has been growing exponentially for decades. The first human genome (three gigabytes of data, which nearly fills a DVD) was sequenced in 2003; by 2020 more than 25 million people had had their genomes sequenced. The latest autonomous vehicles produce up to 30 TB for every 8 hours of driving. IDC estimates that the world will generate over 90 ZB per year in 2020 (1 ZB = 1 trillion billion bytes), more than all data produced since the advent of computers (Figs. 3 and 4).

Data is approximately doubling every 2 years, in 2020, with about 20 trillion bytes of data for every person on earth. Spending on IT hardware, software, services, telecommunications, and staff, that could be considered the "infrastructure" of Big Data, is growing around 40% annually in the USA and China, and approaching that in the rest of the world. There has been a rapid migration of data from local islands of technology such as corporate servers, mainframes, personal computers,

and portable devices toward data investment in targeted areas like storage management, security, and cloud computing. Data management often centers around the partitioning of data residence between “edge” computers and “cloud” servers.

The mode in which data is presented has changed dramatically over the past century. Most digitized data was in columnar spreadsheet format up through the 1980s. Increased bandwidth, storage, and processing power has allowed richer formats to dominate over the past two decades. Much of the information needed for effective auditing is now encapsulated in these more complex modalities.

Much of the growth in data has been driven by user creation through social networks and multimedia (videos, pictures, music). There is also a rapid growth in transient data. Phone calls that are not recorded, digital TV images that are consumed but not saved, packets temporarily stored in routers, digital surveillance images purged from memory when new images come in, and so forth.

Financial Accounting

Accountancy is the process of communicating financial information about a business entity to users such as shareholders and managers. The communication is generally in the form of financial statements that show, in money terms, the economic resources under the control of management; the art lies in selecting the information that is relevant to the user and is reliable. The principles of accountancy are applied to business entities in three divisions of practical art, named accounting, bookkeeping, and auditing. Accounting is defined by the American Institute of Certified Public Accountants (AICPA) as “the art of recording, classifying, and summarizing in a significant manner and in terms of money, transactions and events which are, in part at least, of financial character, and interpreting the results thereof.” Today, accounting is called “the language of business” because it is the vehicle for reporting financial information about a business entity to many different groups of people. Accounting that concentrates on reporting to people inside the business entity is called management accounting and is used to provide information to employees, managers, owner-managers, and auditors. Management accounting is concerned primarily with providing a basis for making management or operating decisions. Accounting that provides information to people outside the business entity is called financial accounting and provides information to present and potential shareholders, creditors such as banks or vendors, financial analysts, economists, and government agencies. Because these users have different needs, the presentation of financial accounts is very structured and subject to many more rules than management accounting. The body of rules that governs financial accounting in a given jurisdiction is called Generally Accepted Accounting Principles, or GAAP. Other rules include International Financial Reporting Standards, or IFRS, or US GAAP. Accounting standards have historically been set by the American Institute of Certified Public Accountants (AICPA) subject to Securities and Exchange Commission regulations. The AICPA first created the Committee on Accounting Procedure in 1939 and replaced that with the Accounting Principles Board in 1951. In 1973, the Accounting Principles Board was replaced by the Financial Accounting Standards Board (FASB) under the supervision of the Financial Accounting Foundation with the Financial Accounting Standards Advisory Council serving to advise and provide input on the accounting standards. Other organizations involved in determining United States’ accounting standards include the Governmental Accounting Standards Board (GASB), formed in 1984, and the Public Company Accounting Oversight Board (PCAOB).

In 2008, the FASB issued the FASB Accounting Standards Codification, which reorganized the thousands of US GAAP pronouncements into roughly 90 accounting topics. In 2008, the Securities and Exchange Commission issued a preliminary “roadmap” that may lead the USA to abandon Generally Accepted Accounting Principles in the future and to join more than 100 countries around the world instead of using the London-based International Financial Reporting Standards. As of 2010, the convergence project was underway with the FASB meeting routinely with the IASB. The SEC expressed their aim to fully adopt International Financial Reporting Standards in the USA by 2014. With the convergence of the US GAAP and the international IFRS accounting systems, as the highest authority over International Financial Reporting Standards, the International Accounting Standards Board is becoming more important in the US.

The Products of Accounting: Financial Statements

There are four main financial statements. They are (1) balance sheets, (2) income statements, (3) cash flow statements, and (4) statements of shareholders’ equity.

Balance sheets show what a company owns and what it owes at a fixed point in time. Income statements show how much money a company made and spent over a period of time. Cash flow statements show the exchange of money between a

company and the outside world also over a period of time. The fourth financial statement, called a “statement of shareholders’ equity,” shows changes in the interests of the company’s shareholders over time.

The Balance Sheet

A balance sheet provides detailed information about a company’s assets, liabilities, and shareholders’ equity. Assets are things that a company owns that have value. This typically means they can either be sold or used by the company to make products or provide services that can be sold. Assets include physical property, such as plants, trucks, equipment, and inventory. It also includes things that cannot be touched but nevertheless exist and have value, such as trademarks and patents. And cash itself is an asset. So are investments a company makes. Liabilities are amounts of money that a company owes to others. This can include all kinds of obligations, like money borrowed from a bank to launch a new product, rent for use of a building, money owed to suppliers for materials, payroll a company owes to its employees, environmental cleanup costs, or taxes owed to the government. Liabilities also include obligations to provide goods or services to customers in the future. Shareholders’ equity is sometimes called capital or net worth. It is the money that would be left if a company sold all of its assets and paid off all of its liabilities. This leftover money belongs to the shareholders, or the owners, of the company. A company’s balance sheet is set up like the basic accounting equation shown above. On the left side of the balance sheet, companies list their assets. On the right side, they list their liabilities and shareholders’ equity. Sometimes balance sheets show assets at the top, followed by liabilities, with shareholders’ equity at the bottom. Assets are generally listed based on how quickly they will be converted into cash. Current assets are things a company expects to convert to cash within 1 year.

A good example is inventory. Most companies expect to sell their inventory for cash within 1 year. Noncurrent assets are things a company does not expect to convert to cash within 1 year or that would take longer than 1 year to sell. Noncurrent assets include fixed assets. Fixed assets are those assets used to operate the business but that are not available for sale, such as trucks, office furniture, and other property. Liabilities are generally listed based on their due dates. Liabilities are said to be either current or long term. Current liabilities are obligations a company expects to pay off within the year. Long-term liabilities are obligations due more than 1 year away.

Shareholders’ equity is the amount owners invested in the company’s stock, plus or minus the company’s earnings or losses since inception. Sometimes companies distribute earnings, instead of retaining them. These distributions are called dividends. A balance sheet shows a snapshot of a company’s assets, liabilities, and shareholders’ equity at the end of the reporting period. It does not show the flows into and out of the accounts during the period.

The Income Statement

An income statement is a report that shows how much revenue a company earned over a specific time period (usually for a year or some portion of a year). An income statement also shows the costs and expenses associated with earning that revenue. The literal “bottom line” of the statement usually shows the company’s net earnings or losses. This tells you how much the company earned or lost over the period. Income statements also report earnings per share (or “EPS”). This calculation tells you how much money shareholders would receive if the company decided to distribute all of the net earnings for the period.

Cash Flow Statements

Cash flow statements report a company’s inflow and outflow of cash. This is important because a company needs to have enough cash on hand to pay its expenses and purchase assets. While an income statement can tell you whether a company made a profit, a cash flow statement can tell you whether the company generated cash. A cash flow statement shows changes over time rather than absolute dollar amounts at a point in time. It uses and reorders the information from a company’s balance sheet and income statement. Cash Flow Statements may be generalized to Working Capital Statements, which replace cash in the calculations, with more general Current Assets and Current Liabilities, both of which are expected to expire within the operating cycle (typically 1 year) and thus are what would be called “near-cash.” The bottom line of the cash flow statement shows the net increase or decrease in cash for the period. Cash flow statements are divided into three main parts. Each part reviews the cash flow from one of three types of activities: (1) operating activities, (2) investing activities, and (3) financing activities.

The Methodology of Accounting

Financial accounting is the vehicle through which firms can record and report important economic transactions that affect their wealth. It is a quasi-axiomatic system where fundamental “principles” are loosely applied to the recording of economic events (transactions) that affect firm wealth. These detailed transactions are summarized into accounts based on a firm-specific classification called the Chart of Accounts. The accounts are further organized and summarized for reporting in accordance with generally accepted accounting principles (GAAP).

The accounting perspective in practice today derives from Al-Khwarizmi’s eighth century system of algebraic balancing and arithmetic manipulation. The system is inherently linear—prices and costs are assumed to be additive and generally are considered fixed across transactions. But in practice, many economic processes are non-linear and data may be incomplete and inaccurate. Consequently, modern accounting reports are economic approximations that are made to facilitate tractability and scalability in accounting systems. The perspective adopted by modern auditors reflects trade-offs made in accounting reports, and many standard audit procedures accommodate the inherent uncertainty of financial accounting.

Generally Accepted Accounting Principles (GAAP)

Generally Accepted Accounting Principles (GAAP) refer to the standard framework of guidelines for financial accounting used in any given jurisdiction, generally known as accounting standards. GAAP includes the standards, conventions, and rules accountants follow in recording and summarizing, and in the preparation of financial statements. Financial Accounting is information that must be assembled and reported objectively. Third-parties who must rely on such information have a right to be assured that the data is free from bias and inconsistency, whether deliberate or not. For this reason, financial accounting relies on GAAP. Principles derive from tradition, such as the concept of matching. In any report of financial statements (audit, compilation, review, etc.), the preparer/auditor must indicate to the reader whether or not the information contained within the statements complies with GAAP.

Theory

The basic accounting equation is assets = liabilities + stockholders' equity. This is the balance sheet. The foundation for the balance sheet begins with the income statement, which is revenues – expenses = net income or net loss. This is followed by the retained earnings statement, which is beginning retained earnings + net income – dividends = ending retained earnings or beginning retained earnings – net loss – dividends = ending retained earnings.

Assumptions

1. Accounting Entity: assumes that the business is separate from its owners or other businesses. Revenue and expense should be kept separate from personal expenses.
2. Going Concern: assumes that the business will be in operation indefinitely. This validates the methods of asset capitalization, depreciation, and amortization. only when liquidation is certain is this assumption not applicable.
3. Monetary Unit principle: assumes a stable currency is going to be the unit of record. The FASB accepts the nominal value of the US Dollar as the monetary unit of record unadjusted for inflation.
4. The Time-Period principle implies that the economic activities of an enterprise can be divided into artificial time periods.

Principles

1. Historical cost principle requires companies to account and report based on acquisition costs rather than fair market value for most assets and liabilities. This principle provides information that is reliable (removing opportunity to provide subjective and potentially biased market values), but not very relevant. Thus there is a trend to use fair values. Most debts and securities are now reported at market values.
2. Revenue recognition principle requires companies to record when revenue is (1) realized or realizable and (2) earned, not when cash is received. This way of accounting is called accrual basis accounting.

3. Matching principle. Expenses have to be matched with revenues as long as it is reasonable to do so. Expenses are recognized not when the work is performed, or when a product is produced, but when the work or the product actually makes its contribution to revenue. Only if no connection with revenue can be established, cost may be charged as expenses to the current period (e.g., office salaries and other administrative expenses). This principle allows greater evaluation of actual profitability and performance (shows how much was spent to earn revenue). Depreciation and Cost of Goods Sold are good examples of application of this principle.
4. Full Disclosure principle. Amount and kinds of information disclosed should be decided based on trade-off analysis as a larger amount of information costs more to prepare and use. Information disclosed should be enough to make a judgment while keeping costs reasonable. Information is presented in the main body of financial statements, in the notes or as supplementary information

Constraints

1. Objectivity principle: the company financial statements provided by the accountants should be based on objective evidence.
2. Materiality principle: the significance of an item should be considered when it is reported. An item is considered significant when it would affect the decision of a reasonable individual.
3. Consistency principle: means that the company uses the same accounting principles and methods from year to year.
4. Conservatism principle: when choosing between two solutions, the one that will be least likely to overstate assets and income should be picked.

The Accounting Process and Major Document Files

Accounting Entries and Document Files

Al Khwarizmi's algebraic method of balancing accounting entries developed into the standards for error control, nominal accounts, and real accounts that became universal in accounting. By the twelfth century, Fibonacci had codified and circulated these in Latin. This work was popularized by another Italian, Luca Pacioli, in the fifteenth century in one of the first printed and widely distributed texts on business mathematics. This is the double-entry bookkeeping method we use today, which is conceived in terms of debits and credits. Pacioli's most important contributions were

1. In deciding which account has to be debited and which account has to be credited, the processes of balancing and completion follow the accounting equation: $\text{Equity} = \text{Assets} - \text{Liabilities}$.
2. The accounting equation serves as an error detection tool. If at any point, the sum of debits for all accounts does not equal the corresponding sum of credits for all accounts, an error has occurred.
3. In the double-entry accounting system, each accounting entry records related pairs of financial transactions for asset, liability, income, expense, or capital accounts.
4. A document file that records economic events that result in an increase or decrease in wealth of the firm is called a Journal, and the transaction is the Journal Entry.
5. Pacioli's term for a Journal is a "Daybook"—a term that was used up to the twentieth century in accounting.

The system popularized by Pacioli has become the standard fare for introductory accounting courses around the world. For the purpose of the accounting equation approach, all the accounts are classified into the following five types: assets, liabilities, income/revenues, expenses, or capital gains/losses. If there is an increase or decrease in one account, there will be equal decrease or increase in another account.

Books of Accounts

Each financial transaction is recorded in at least two different real ledger accounts within the financial accounting system, so that the total debits equals the total credits in the general ledger, i.e., the accounts balance. The transaction is recorded as a "debit entry" (Dr.) in one account, and a "credit entry" (Cr.) in a second account. The debit entry will be recorded on the debit side (left-hand side) of a general ledger, and the credit entry will be recorded on the credit side (right-hand side) of a

general ledger account. If the total of the entries on the debit side of one account is greater than the total on the credit side of the same real account, that account is said to have a debit balance.

Double-entry is used only in real ledgers. It is not used in journals (daybooks), which normally do not form part of the real ledger system. The information from the journals will be used in the general ledger and it is the real ledgers that will ensure the integrity of the resulting financial information created from the journals (provided that the information recorded in the journals is correct). For very frequent transactions, such as Sales, a special purpose “Sales journal” is usually kept, and the totals are posted to the general ledger on a periodic (e.g., daily) basis.

A real account in a business is a record of the amount of asset, liability, or owners’ equity at a precise moment in time. Nominal accounts summarize a business’s revenue and expenses over a period of time, such as a year. The recordkeeping process for bookkeepers is fundamentally the same: Adopt a chart of accounts, make original entries using debits and credits to keep the books in balance, make adjusting entries to get profit for the period right, and close the books at the end of the year. Businesses keep two types of accounts:

1. Real accounts are those reported in the balance sheet, which is the summary of the assets, liabilities, and owners’ equities of a business. The label ‘real’ refers to the continuous, permanent nature of this type of account. Real accounts are active from the first day of business to the last day. (A real account could have a temporary zero balance, in which case it is not reported in the balance sheet.) Real accounts contain the balances of assets, liabilities, and owners’ equities at a specific point in time, such as at the close of business on the last day of the year. The balance in a real account is the net amount after subtracting decreases from increases in the account.
2. Nominal (revenue and expense) accounts are closed at the end of the year. After these accounts have done their jobs accumulating amounts of sales and expenses for the year, their balances are closed. Their balances are reset to zero to start the new year. Nominal accounts are emptied out to make way for accumulating sales revenue and expenses during the following year.

The “Trial Balance” lists all the ledger account balances. The list is split into two columns, with debit balances placed in the left-hand column and credit balances placed in the right-hand column. Another column will contain the name of the ledger account—these accounts are determined for the firm in the “Chart of Accounts.” The total of the debit column must equal the total of the credit column, which is why this is called a “Balance.”

When the total revenues equal total expenses, the accounting equation must be true: assets = liabilities + equity: i.e., the equality of the “nominal” accounts implies the equality of the “real” accounts. The system of recording debits and credits is designed to keep the “nominal” and “real” accounts synchronized.

The modern term for “Capital” is “Owners Equity” and consists of equity shares plus retained earnings of the firm, less dividends paid out. The change in Capital from period to period is “Income” or “Loss,” and the portions of the Capital that this represents are called Retained Earnings (the rest from equity shares).

For the accounts to remain in balance, a change in one account must be matched with a change in another account. These changes are made in debit and credit recording system applied to the chart of accounts. Each transaction recorded in a Journal Entry has a debit and credit side, and these sides have the following effect depending on the particular classification of an account in the chart of accounts.

Though the need for such “error checking” systems has since been obsoleted by computers, it is still part of the tradition of accounting, and we can expect it to be for some time to come. Each of these constructs: debits, credits, journals, ledgers, and so forth are reflected in the processes and file systems of modern computing systems.

Code and Data Repositories for Audit Analytics

Code chunks in many places in *Audit Analytics* text contain references to external .CSV formatted files. These files are hosted on GitHub in the `westland/auditanalytics` repository. A conscious decision was made, in writing the code in this book, to keep files in a .CSV format. Many sources of data that accountants encounter in practice retain data in Excel or other spreadsheet formats. Where they are not, as with database tables, they may be reformatted and presented to auditors as spreadsheets. In this vein, the last chapter creates simulated “test decks” of accounting transactions in .CSV formats because

such formats are in their own way, the “native” formats for accounting and financial data. Presentation of *Audit Analytics* data in .CSV formats allows me to show examples of loading/parsing the kind of raw data encountered in practice.

This is not how files are typically stored and formatted in R. Nonetheless, for pedagogical purposes, I have chosen to retain .CSV formatting for the files that are downloaded from the book’s repositories. Otherwise GitHub’s `westland/auditanalytics` repository for this book follows the repository standards adopted by R and RStudio, and eventually I intend to release these as CRAN repositories.

Raw (.CSV) data files used in the *Audit Analytics* text are made available to the R workspace by installing the `westland/auditanalytics` package from GitHub, and using `system.file()` to refer to specific files. The path to the file is returned by `system.file()` which can then simply be inserted into any other commands that need this path. The following code chunk provides a simple example.

```
devtools::install_github("westland/auditanalytics")
library(auditanalytics)

fyear_end_ar_ledger <- read.csv(
  system.file("extdata", "fyear_end_ar_ledger.csv", package = "auditanalytics", mustWork = TRUE))

head(fyear_end_ar_ledger)

##   X customer_no invoice_no amount shipper_no shipper_date
## 1 1      c00005     i00147    462      s00136 2020-03-01
## 2 2      c00007     i00302    504      s00313 2020-05-02
## 3 3      c00004     i00334   1122      s00318 2020-05-03
## 4 4      c00010     i00348   1292      s00345 2020-05-13
## 5 5      c00005     i00437    756      s00437 2020-06-15
## 6 6      c00005     i00461   1470      s00481 2020-07-01
```

R Packages Required for This Book

The code in the chapters in this book requires R packages that are specified in the `library("package_name")` commands. These will be packages such as `tidyverse`, `ggplot2`, `lubridate`, and `keras`. `Keras` is the API for the Tensorflow machine learning language and requires a separate `keras` installation with `install_keras`; general notes on Tensorflow installation are provided below. Tensorflow itself is a collection of algorithms written in multiple languages inside a Python wrapper. It requires a Python environment be installed; further information can be found at <https://www.tensorflow.org/>.

Package Installation There are two steps to using a package. First it must be *installed*, i.e., copied to a location on your computer where R can access it. Then it must be *loaded* into the working memory of R. To install, for example, the `tidyverse` package, type `install.packages("tidyverse")` and then press the *Enter/Return* key. To load the previously installed package, type `library(tidyverse)`. After these commands, the `tidyverse` package will now be available for use by your program code.

Tensorflow Installation Tensorflow is a machine learning package used in this book; commands to Tensorflow are called using the Keras API. Prior to using the `tensorflow` R package, you need to install a version of TensorFlow on your system using the R `install_tensorflow()` function, which provides an easy-to-use wrapper for the various steps required to install TensorFlow. You can also choose to install TensorFlow manually (as described at <https://www.tensorflow.org/install/>).

TensorFlow for R is tested and supported on the following 64-bit systems:

1. Ubuntu 16.04 or later
2. Windows 7 or later
3. macOS 10.12.6 (Sierra) or later (no GPU support)

First, install the tensorflow R package from GitHub or the CRAN repository (search to find the site), and then use the `install_tensorflow()` function to install TensorFlow. Note that on Windows you need a working installation of Anaconda. `install_tensorflow()` is a wrapper around `reticulate::py_install`.

```
install.packages("tensorflow")
library(tensorflow)
install_tensorflow()
```

You can confirm that the installation succeeded with:

```
library(tensorflow)
tf$constant("Hello Tensorflow")
## tf.Tensor(b'Hello Tensorflow', shape=(), dtype=string)
```

This will provide you with a default installation of TensorFlow suitable for use with the tensorflow R package. There is much more to TensorFlow, and interested readers should review the materials at <https://tensorflow.rstudio.com/> and at <https://www.tensorflow.org/>.

A caveat on code 'copy & paste' Though code from this book is available in the package repositories, readers may be tempted in some situations to 'copy and paste' code directly from the book. Unfortunately, many built-in machine editors try to convert quotation marks to distinct opening and closing forms, which will cause the resulting code to throw an error. Be aware that 'copy & paste' code may require some additional editing.

References

- Aboody, David, and Baruch Lev. 1998. The Value Relevance of Intangibles: The Case of Software Capitalization. *Journal of Accounting Research* 36: 161–91.
- Bryson, Bill. 2010. *At Home: A Short History of Private Life*. New York: Doubleday.
- De Roover, Raymond. 1938. Characteristics of Bookkeeping Before Paciolo. *The Accounting Review* 13 (2): 144–449.
- . 1955. New Perspectives on the History of Accounting. *The Accounting Review* 30 (3): 405–420.
- . 1956. The Development of Accounting Prior to Luca Pacioli According to the Account-Books of Medieval Merchants. In *Studies in the History of Accounting*, 114–174. London: Sweet & Maxwell.
- Kaldor, Nicholas. 1966. Marginal Productivity and the Macro-Economic Theories of Distribution: Comment on Samuelson and Modigliani. *The Review of Economic Studies* 33 (4): 309–119.
- Lev, Baruch. 2001. *Intangibles: Management, Measurement and Reporting*. Washington: Brookings Institution Press.
- . 2004. Sharpening the Intangibles Edge. *Harvard Business Review* 6: 109–118.
- Matthews, Derek. 2006. *A History of Auditing: The Changing Audit Process in Britain from the Nineteenth Century to the Present Day*. London: Routledge.
- Power, Michael. 1997. *The Audit Society: Rituals of Verification*. Oxford: Oxford University Press.

Foundations of Audit Analytics



Business and Data Analytics

The modern science of data analytics evolved from early projects to add scientific rigor in two fields—(1) games of chance and (2) governance (Stigler 1986). The latter application, in governance, focused on summarizing the demographics of nation-states and lent its name, *statistics*, to the field. The field has steadily evolved to include exploratory analysis, which used computers to improve graphics and summarizations; and to include the computationally intensive methods termed machine learning.

The mathematical foundations of statistics evolved from the seventeenth to nineteenth centuries based on work by Thomas Bayes, Pierre-Simon Laplace, and Carl Gauss. Statistics as a rigorous scientific accelerated at the turn of the twentieth century under Francis Galton, Karl Pearson and R.A. Fisher, introducing experimental design and maximum likelihood estimation (Stigler 1986).

Exploratory data analysis (EDA) arose from seminal work in data summarization presented in Cochran et al. (1954) and Tukey (1980). EDA built on the emerging availability of computing power and software. EDA is now an essential part of data analytics, and is important for determining how much, and what kind of information is contained in a dataset. In contrast, the statistics of Pearson and Fisher tended to focus on testing of models and prediction. EDA complements such testing by assessing whether data obtained for testing is actually appropriate for the questions posed; in the vernacular of machine learning, EDA helps us *extract the features* that are contained in the data. In our current era of massive, information rich datasets, where often we have no control, and limited information about how the data was obtained, EDA has become an essential precursor of model testing and prediction.

A typical first step in analytical review of a company might be to review the industry statistics to find out what part, if any, of our client's financial accounts are outliers in the industry. Unusual account values or relationships can indicate audit risks that would require a broader scope of auditing. Exploratory statistics are easy to implement, yet are ideal for quickly highlighting unusual account values. We will first load a dataset of industry statistics that have been downloaded from the Wharton Research Data Service repository, and conduct simple summaries of the data. The R package *plotluck* automatically chooses graphs that are appropriate for the data, and is an ideal tool for quick visualization of data. We ask: What variables are contained and how they are distributed? In general, R lets the dot symbols stands for “all variables in the data set” and “~1” regresses the variables against the intercept, thus giving the distribution (essentially a smoothed histogram) (Fig. 1)

```
library(tidyverse)
library(plotluck)
library(broom)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

industry_stats <- read.csv(
  system.file("extdata", "ch_2_dataset.csv", package = "auditanalytics", mustWork = TRUE))
summary(industry_stats)
```

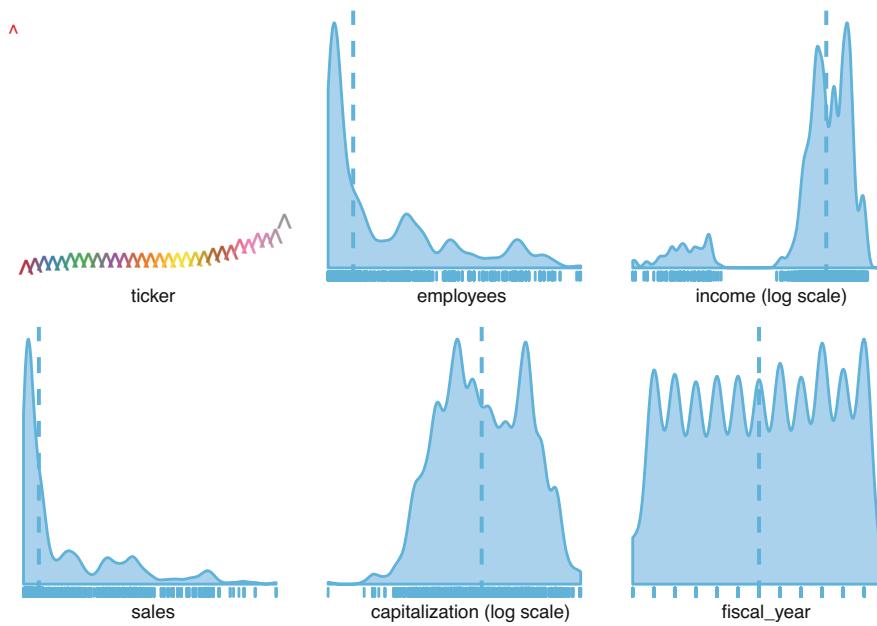


Fig. 1 Exploratory data analysis (EDA) using R's plotluck package

```

##      ticker     employees      income      sales
##  HD    : 84   Min.   : 1.00   Min.  :-38732.0   Min.   : 336.4
##  HRB   : 60   1st Qu.: 12.00   1st Qu.: 165.2   1st Qu.: 4260.4
##  BA    : 55   Median  : 44.00   Median : 592.2   Median : 12928.3
##  HNT   : 54   Mean    : 95.56   Mean   : 2139.2   Mean   : 34185.2
##  CVS   : 44   3rd Qu.:152.00   3rd Qu.: 2803.5   3rd Qu.: 51411.2
##  EFX   : 44   Max.    :434.00   Max.   :104821.0   Max.   :207349.0
##  (Other):939 NA's    :2
##      capitalization      fiscal_year
##  Min.   : 49   Min.   :2004
##  1st Qu.: 3815  1st Qu.:2007
##  Median : 11389  Median :2010
##  Mean   : 33641  Mean   :2010
##  3rd Qu.: 47400  3rd Qu.:2013
##  Max.   :382421   Max.   :2016
##
```

```
plotluck(industry_stats, .~1)
```

There are four continuous (employees, income, sales, and capitalization) and one categorical (fiscal year) variable; we will get rid of the ticker. Where there is a significant range of values in a variable, plotluck automatically presents the x-axis on a log scale. Let us explore income and the influence of other variables. Output is set to “verbose” in order to estimate conditional entropies for ordering the plot (lower values indicate stronger informational value). There is a clear correlation of income with capitalization, which we can assess by applying frequency weights (Fig. 2).

How has the industry’s profit changed over the recent past? (Fig. 3).

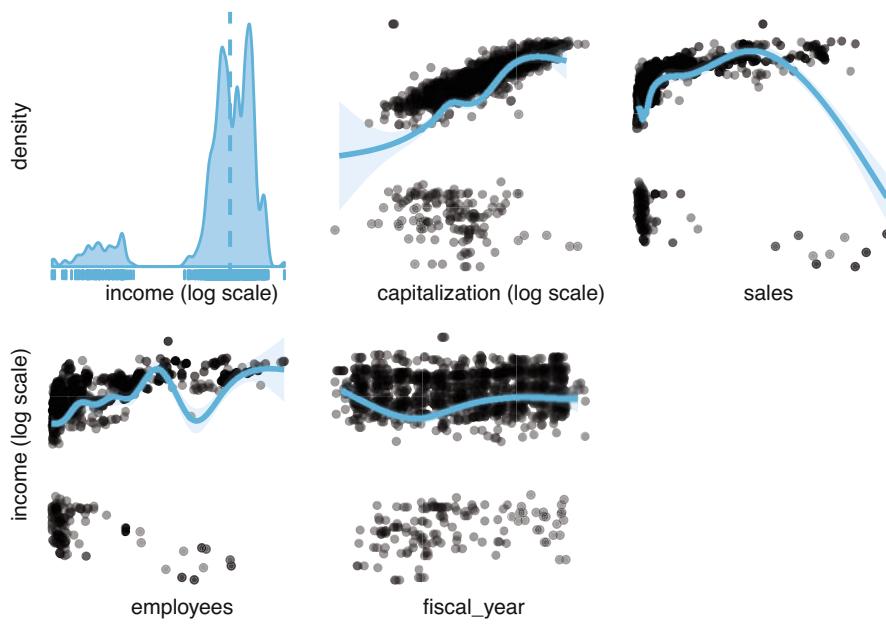


Fig. 2 EDA of income vs. other variables

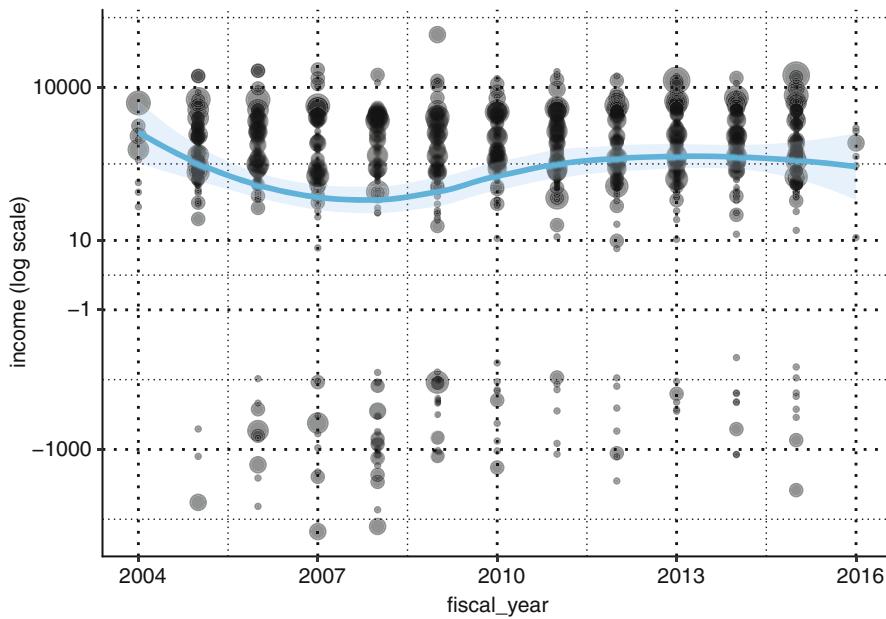


Fig. 3 Income distribution over time

How do sales and headcount influence income? This chart is three-dimensional, with color providing a third dimension for assessing correlations. During analytical review, the auditor can use such analyses to find whether the client's accounts vary from industry averages (Fig. 4).

How has this distribution changed over time?

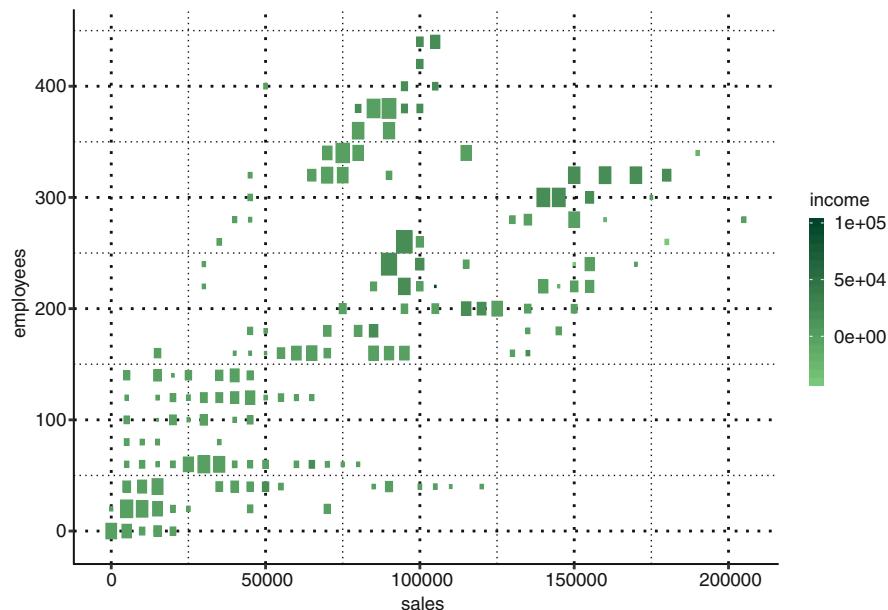
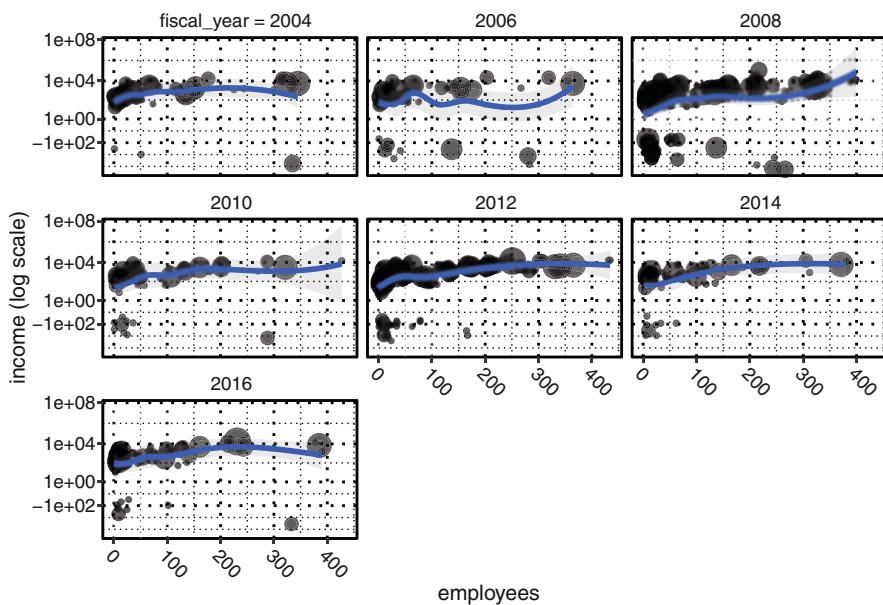


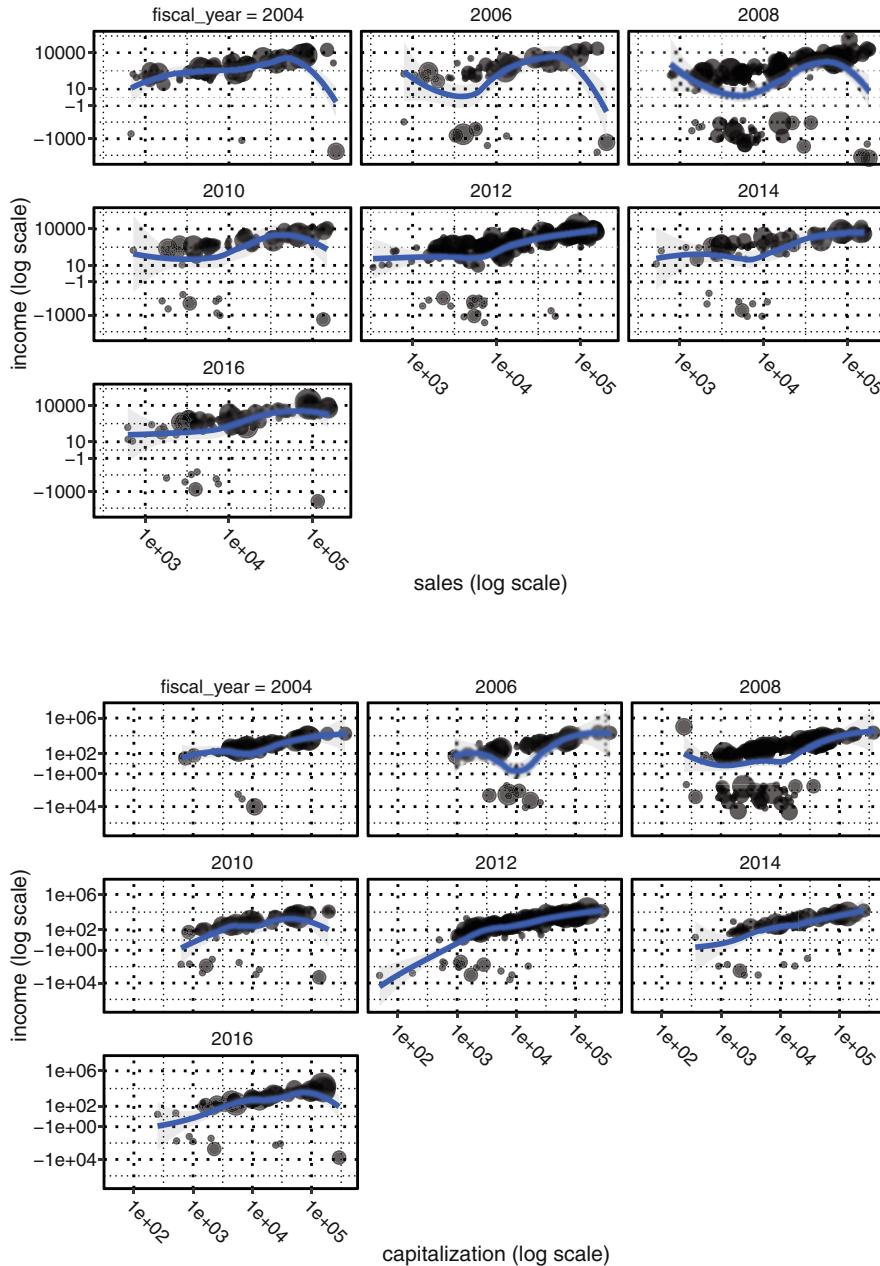
Fig. 4 Three-dimensional plots of income, employees, and sales



New methods for analyzing massive datasets have emerged in the past decade, and are continually evolving, as a product of the machine learning revolution. The three most common terms used to describe these tools are reserved for a nested set of three technologies:

- Artificial Intelligence = “any attempt to mimic human learning/intelligence”
- Machine Learning = “computational methods for learning from data”
- Deep Learning = “machine learning methods that mimic human neural networks (perceptrons)”

The emphasis on “learning” instructs us on the differences between these technologies and more traditional statistical approaches. “Learn” is a transitive verb (i.e., takes an object) and involves



- A **subject** (who makes a *decision*)
- **learns** (from reviewing data) about some
- **construct** (parameter value, classification, etc.)

The essential components of learning are: the decision, the learning method, the construct about which we are learning, and the quality assessment (how well we learned). The field of artificial intelligence was motivated by Vannevar Bush and Bush's (1945) article "As we may think" and initiated projects in rule-based and symbolic programs such as early chess programs that involved hardcoded rules crafted by programmers. Despite considerable but sporadic funding, the field only began flourishing with the machine learning competitions hosted by website Kaggle in 2010. Random forests quickly became a favorite on the platform, but by 2014, gradient boosting machines were outperforming them, and by 2016, perceptron models, in particular building on the successes at Google, began dominating the field.

Machine learning re-envisioned several of the fundamental methods of statistics. Computationally intensive stochastic gradient boosting replaces the first-order conditions in calculus used in statistics search for solutions. The easy to compute, squared-error loss function of parametric statistics gave way to a larger selection of solution concepts. Machine learning is not limited to fitting statistical depictions of probability distributions typically defined with 1, 2, or 3 parameters; highly customizable distributions can be crafted by altering the weights of successive layers of machine learning networks.

For classification problems, cross-entropy measures offered much better fit than, say, comparable logit or probit models. One particular challenge, *ImageNet* was notoriously difficult in 2012, consisting of classifying high-resolution color images into 1000 different categories after training on 1.4 million images. In 2011, the top-five accuracy of the winning model, based on classical approaches to computer vision, was only 74.3%. In 2012, a deep-learning method was able to achieve a top-five accuracy of 83.6%. By 2015, the winner reached an accuracy of 96.4%, and the classification task on ImageNet was considered to be a completely solved problem.

Business analytics' embrace of machine learning is inspired by the numerous successes: near-human-level image classification, language translation, speech recognition, handwriting transcription, text-to-speech conversion, autonomous driving and better than human Go and Chess games. Despite these successes, there are reasons for a balanced, holistic perspective on the tools of data analytics. Machine learning outperforms twentieth century statistics for:

- Large datasets: These tend to overfit simple statistical models and may be a basis for p-hacking
- Large complex construct spaces: Where a model is estimating many parameters such as in image processing, 1, 2, or 3 parameter probability distributions are not sufficiently rich
- Feature extraction: the major contribution of exploratory data analysis was feature extraction from data, which could be used to decide where a dataset was appropriate for model testing.

But statistical models still outperform machine learning models (though this is steadily evolving) where there is a:

- clear interpretation of results of analysis
- consistency
- replicability
- formal definition of “information”
- clear roles for data
- clear demands on constructs
- clear (though disputed) philosophies on the meaning of “learn”
- formal logic and notation

The rest of this chapter is devoted to classifying types of data, which in turn says something about the particular importance that we attach to an entity, and way that we measure it. In the process I will share some of the excellent graphic tools that the R language offers the auditor for understanding accounting data.

Accounting Data Types

McCarthy (1979, 1982) proposed a design theory of accounting systems that applies the Chen (1976) framework to accounting. In it, accounting transactions are measurements of economic events involving an *entity's* contractual activities with a *related* party. These measurements result in the recording of numbers, time, classifications, and descriptive information. Classifications are dictated by a firm's *Chart of Accounts* which delineates the types of economic activities in the firm. *Measurements* are in monetary units, thus require some method of valuing an economic event. The ubiquity of information assets makes valuation one of the most difficult challenges facing modern accountants. *Time* in the form of date stamps ensures that economic events are recorded in the correct accounting period. *Descriptive* information was originally entered in notes to a journal entry. But social networks and news outlets provide auditors with a plethora of relevant information in textual form. Without methods of interpreting this, auditors substantially increase their risk of failures.

The processing and interpretation of each of these types of information is different. Information technology, statistical methods, and software specify *data types* to differentiate the processing and storage of data.

The R language excels at managing data. Indeed, this particular strength sets R, as an audit language, above any other software language. This means, when using R, that the auditor never has to worry that some part of the client's data will remain inaccessible. Packages exist for any commercially important data structure and format, whether real-time stream, web-based, cloud-based, or on client's bespoke system, can be analyzed with R code.

The next sections of this chapter discuss the most important data and file types, with examples of how these are represented in R. The examples here use several databases ranging from financial reports of industry firms over time, of Sarbanes-Oxley reports and of control breaches. In the process, the management of various types of information are highlighted: e.g., ticker information is categorical, fee information is continuous, breach and SOX-audit decision data is binary, and so forth.

Numerical vs. Categorical

There are two basic types of structured data: numeric and categorical. Numeric data comes in two forms: continuous, such as wind speed or time duration, and discrete, such as the count of the occurrence of an event. Categorical data takes only a fixed set of values, such as a type of TV screen (plasma, LCD, LED, etc.) or a state name (Alabama, Alaska, etc.). Binary data is an important special case of categorical data that takes on only one of two values, such as 0/1, yes/no, or true/false. Another useful type of categorical data is ordinal data in which the categories are ordered; an example of this is a numerical rating (1, 2, 3, 4, or 5).

Why do we bother with a taxonomy of data types? It turns out that for the purposes of data analysis and predictive modeling, the data type is important to help determine the type of visual display, data analysis, or statistical model. In fact, data science software, such as R and Python, uses these data types to improve computational performance. More importantly, the data type for a variable determines how software will handle computations for that variable.

```
library(tidyverse)
library(plotluck)
library(broom)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

sox_stats <-
  read.csv(
    (system.file("extdata", "ch_2_data_types.csv", package = "auditAnalytics",
    mustWork = TRUE)))

summary(sox_stats)

##      ticker        date       card       disc       hack
##  UNH     : 46   Min.   :2005      : 1   Min.   :0.0000   Min.   :0.0000
##  C      : 45   1st Qu.:2007     0 :1460   1st Qu.:0.0000   1st Qu.:0.0000
##  T      : 40   Median  :2010     1 : 52   Median  :0.0000   Median  :0.0000
##  AAPL    : 35   Mean    :2010     card:  1   Mean    :0.1739   Mean    :0.2467
##  HD      : 35   3rd Qu.:2013          3rd Qu.:0.0000   3rd Qu.:0.0000
##  HNT     : 30   Max.   :2016          Max.   :1.0000   Max.   :1.0000
##  (Other):1283 NA's    :2          NA's   :2   NA's   :2
##      insd        phys       port       stat
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000000
##  Median :0.0000   Median :0.0000   Median :0.0000   Median :0.000000
##  Mean   :0.1382   Mean   :0.0463   Mean   :0.2837   Mean   :0.03704
##  3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:0.000000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000000
##  NA's   :2         NA's   :2       NA's   :2       NA's   :2
##      unkn        effective_303   mat_weak_303   sig_def_303
##  Min.   :0.000000   Min.   :0.0000   Min.   :0.000000   Min.   :0.0000
##  1st Qu.:0.000000   1st Qu.:1.0000   1st Qu.:0.000000   1st Qu.:0.0000
##  Median :0.000000   Median :1.0000   Median :0.000000   Median :0.0000
##  Mean   :0.03968   Mean   :0.9616   Mean   :0.03241   Mean   :0.1157
##  3rd Qu.:0.000000   3rd Qu.:1.0000   3rd Qu.:0.000000   3rd Qu.:0.0000
##  Max.   :1.000000   Max.   :1.0000   Max.   :1.000000   Max.   :1.0000
```

```

##  NA's    :2          NA's    :2          NA's    :2          NA's    :2
##    auto_1      auditor_agrees_303 effective_404 audit_fee
##  Min.   :0.0000  Min.   :0.00000  Min.   :0.0000  Min.   : 100000
##  1st Qu.:0.0000  1st Qu.:0.00000  1st Qu.:0.0000  1st Qu.: 3785984
##  Median :0.0000  Median :0.00000  Median :0.0000  Median : 7956000
##  Mean   :0.2037  Mean   :0.08003  Mean   :0.3935  Mean   : 15321536
##  3rd Qu.:0.0000  3rd Qu.:0.00000  3rd Qu.:1.0000  3rd Qu.: 17800000
##  Max.   :1.0000  Max.   :1.00000  Max.   :1.0000  Max.   :112200000
##  NA's    :2          NA's    :2          NA's    :2          NA's    :2
##  non_audit_fee      tax_fees
##  Min.   :     0  Min.   :     0
##  1st Qu.: 477686  1st Qu.:  93140
##  Median : 1068000  Median : 370000
##  Mean   : 3679039  Mean   : 1451705
##  3rd Qu.: 3400000  3rd Qu.: 1311000
##  Max.   :41100000  Max.   :20800000
##  NA's    :2          NA's    :2

```

Continuous (Interval, Float, Numeric) Data

Many of our key financial metrics are continuous measures. Conceptually, we can subdivide continuous data without end. We can look at the density functions of continuous data using the plotluck package.

Discrete (Integer, Count) Data

In practice, our minimum level of resolution in accounts is one dollar (or single monetary unit). This resolution limit is in fact the basis of audit procedures such as dollar-unit sampling. We are not limited to dollar levels of resolution; annual reports may use thousands or millions of dollars as their level of resolution. In this example we use ggplot to produce a histogram of discrete values. The level of resolution determines the number of bins, in this case 20. Human cognition tends to be overwhelmed by large numbers of bins, which is why we often see financial statistics summarized at levels of resolution larger than one dollar.

```

library(ggplot2)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

sox_stats <-
  read.csv(
    system.file("extdata", "ch_2_data_types.csv", package = "auditanalytics",
               mustWork = TRUE))
sox_stats[,c("audit_fee", "non_audit_fee", "tax_fees")] %>%
  ggplot(aes(x=audit_fee)) + geom_histogram(bins=20)

```

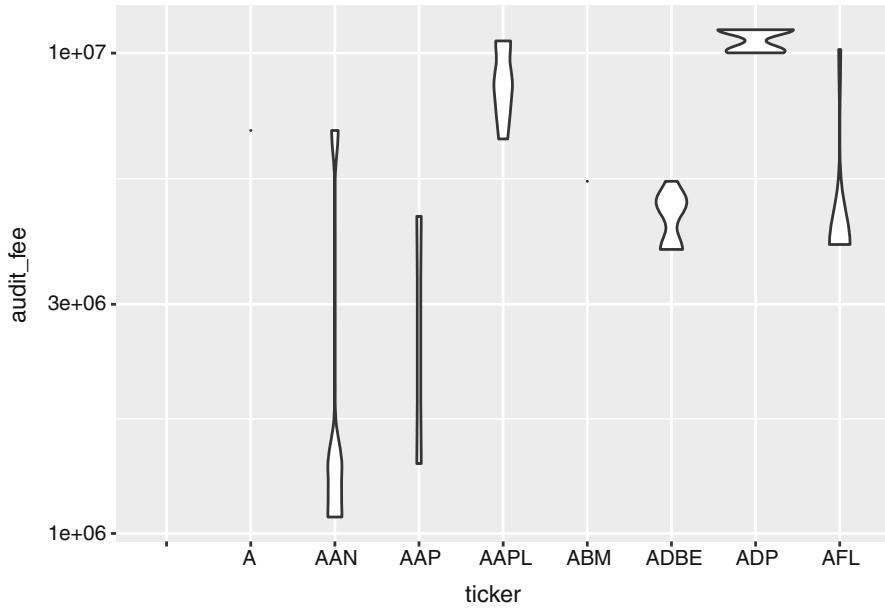
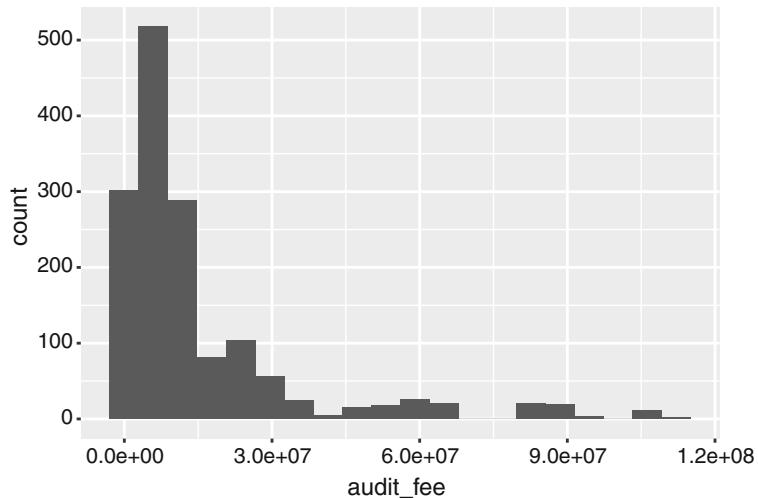
Categorical (Enums, Enumerated, Factors, Nominal, Polychotomous) Data

Categorical data can take on only a specific set of values representing a set of possible categories. The firm's Chart of Accounts imposes a categorical data scheme on the economic events which are the basis of journal entries in the accounting system. In our Sarbanes-Oxley data, we might be interested in the variation over time of audit fees, categorized by the firms they were charged to.

```

sox_stats <-
  sox_stats[,c("ticker", "audit_fee", "non_audit_fee", "tax_fees")]
sox_stats[which(as.character(sox_stats$ticker)<'AI'), ] %>%
  ggplot(aes(x=ticker, y= audit_fee)) +
  geom_violin() +
  scale_y_continuous(trans = "log10")

```



```
library(tidyverse)
library(plotluck)
library(broom)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

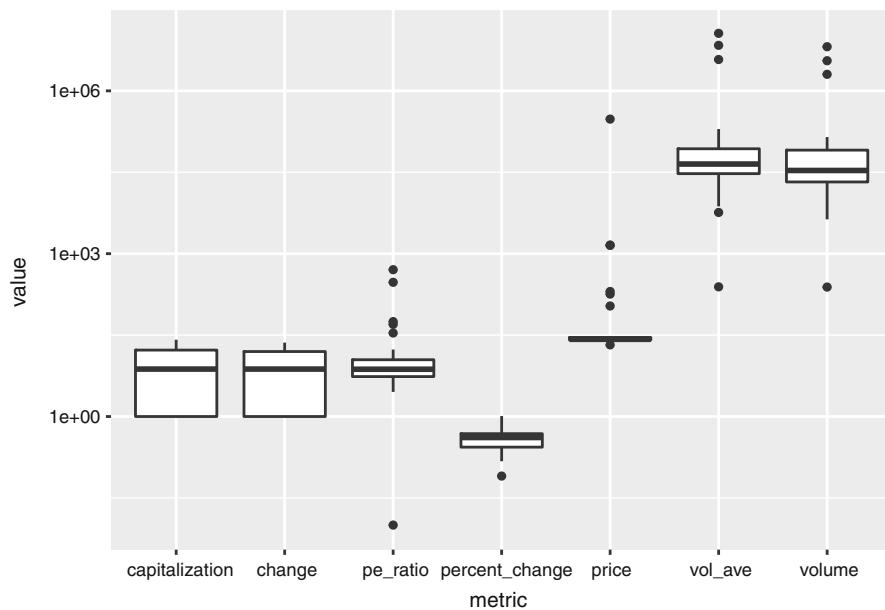
bank_fin <-
  read.csv(system.file("extdata", "ch_2_yahoo_fin.csv", package =
  "auditanalytics", mustWork = TRUE))
bank_fin$change <-
  as.numeric(bank_fin$change)
bank_fin$capitalization <-
  as.numeric(bank_fin$capitalization )
bank_fin <-
```

```

gather(bank_fin,
       key="metric",
       value = value,
       price,
       change,
       percent_change,
       volume,
       vol_ave,
       capitalization,
       pe_ratio)

bank_fin %>% ggplot(aes(x=metric, y=value)) +
  geom_boxplot() +
  scale_y_continuous(trans = "log10")

```



Binary (Dichotomous, Logical, Indicator, Boolean) Data

Binary data represents a special case of categorical data with just two categories. These are data's way of providing answers to *yes/no* or *true/false*. An audit opinion will provide a yes/no answer concerning whether the financial statements are fairly presented. Usually we are not just interested in the answer, but why particular circumstances, treatments, or parameter settings resulted in this answer. In the following figure, we are interested in whether credit card fraud is influenced by the fees paid to the auditor. We analyze a binary variable by looking at the variation in other variables under a 0 or 1 value of the binary variable.

```

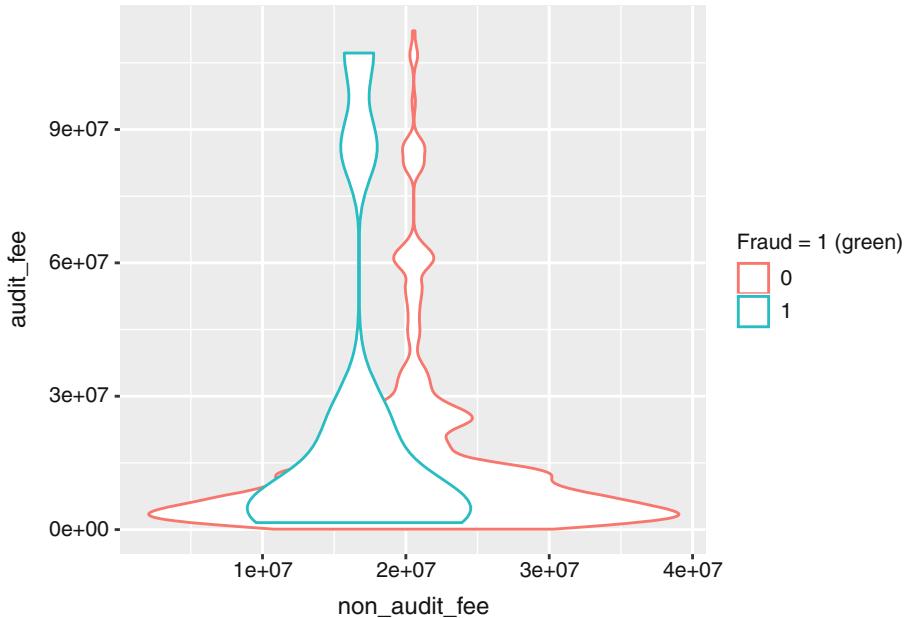
library(tidyverse)
library(plotluck)
library(broom)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

```

```
sox_stats <-
  read.csv(
    "/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_data_types.csv")

ggplot(sox_stats, aes(x=non_audit_fee, y=audit_fee, col=card)) +
  geom_violin() +
  labs(col = "Fraud = 1 (green)")
```



We can similarly inspect the influence of audit fees on all of the binary variables from SOX reporting in the SOX dataset.

```
library(tidyverse)
library(plotluck)
library(broom)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

sox_stats <-
  read.csv(
    system.file("extdata", "ch_2_data_types.csv", package = "auditanalytics",
    mustWork = TRUE))

sox_stats$card <-
  as.integer(sox_stats$card)

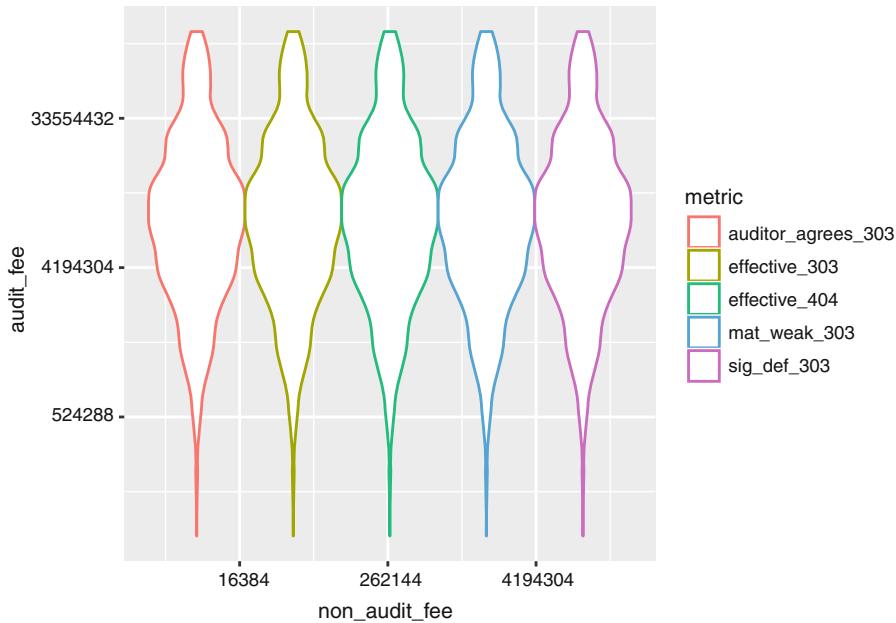
sox_stats1 <-
  gather(sox_stats,
    key="metric",
    value = value,
    effective_303,
    mat_weak_303,
```

```

sig_def_303,
effective_404,
auditor_agrees_303)

ggplot(sox_stats1, aes(x=non_audit_fee, y=audit_fee, col=metric)) +
  geom_violin() +
  scale_x_continuous(trans = "log2") +
  scale_y_continuous(trans = "log2")

```



Ordinal (Ordered Factor) Data

Ordinal data is categorical data with an explicit ordering. Ordinal data provides an important control over documents of original entry in accounting systems. When a journal entry of any sort is generated, it must be uniquely identifiable, and generally the sequence of identifying numbers is assigned in chronological sequence. In the days of paper transaction processing, accountants expected firms to initiate transactions on prenumbered forms, often with numerous copies made simultaneously through the use of (messy) carbon paper. Modern systems assign these numbers internally, but auditors still consider sequential number of input documents to be one of the more important internal controls in a system.

```

library(tidyverse)
library(lubridate)
library(kableExtra)

## create a function to generate a random date in the current year

```

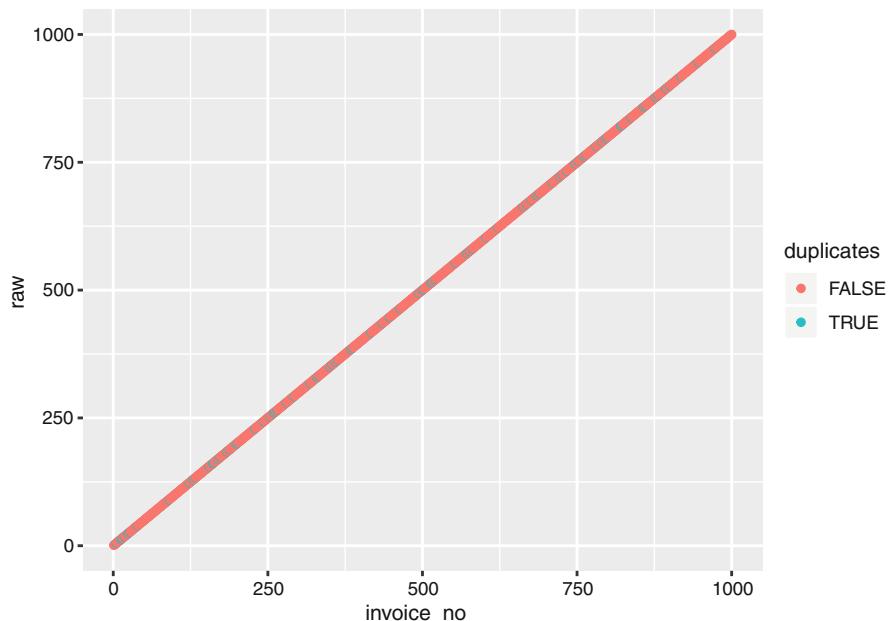
```
rdate <- function(x,
  min = paste0(format(Sys.Date(), '%Y'), '-01-01'),
  max = paste0(format(Sys.Date(), '%Y'), '-12-31'),
  sort = TRUE) {
  dates <- sample(seq(as.Date(min), as.Date(max), by = "day"), x, replace = TRUE)
  if (sort == TRUE) {
    sort(dates)
  } else {
    dates
  }
}

## generate 1000 invoices with dates

invoice_no <- date <- 1:1000 ## placeholder
journal_ent_no <- cbind.data.frame(invoice_no, date)
date <- rdate(1000)
journal_ent_no$date <- date[order(date)]
journal_ent_no$invoice_no <- seq(1,1000) + rbinom(1000,1,.1) # add some errors

duplicates <- duplicated(journal_ent_no$invoice_no)
raw <- seq(1,1000)
journal_dups <- cbind.data.frame(raw, duplicates)

ggplot(journal_dups, aes(x=invoice_no, y=raw, col=duplicates)) +
  geom_point()
```

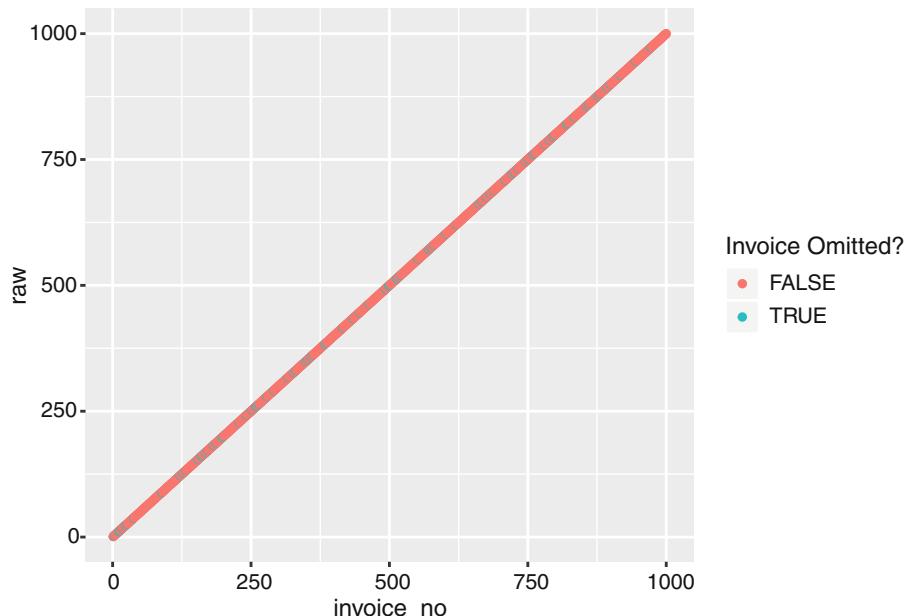


Tables are always an option for omissions and duplications

```
library(tidyverse)
library(kableExtra)
library(reshape2)

journal_ent_no[journal_dups$duplicates==TRUE,] %>%
  kable(longtable=T, caption = "Duplicated Invoices") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"),
                full_width = F, font_size=10,) %>%
  column_spec(1, width = "3em") %>%
  column_spec(2, width = "4em")
```

But graphs provide immediate access to the degree of problem, by looking at the number of exceptions in the graph, and are most visible if exceptions are rendered in a contrasting color such as the green used in the accompanying charts. Often this is all that the auditor needs to render an opinion on internal controls, as it is the client's responsibility to correct these problems.



```
journal_dups$omit <- 0
journal_dups$invoice <- journal_ent_no$invoice_no
journal_dups[journal_dups$raw[!journal_dups$raw %in% journal_ent_no$invoice],
"omit"] <- 1
journal_dups$omit <- as.logical(journal_dups$omit)

ggplot(journal_dups, aes(x=invoice_no, y=raw, col=omit)) +
  geom_point() +
  labs(col = "Invoice Omitted?")
```

Data Storage and Retrieval

The amount of recorded data produced by human activity has probably been growing exponentially for more than a millennium. Most data today is digitized, both for archival storage as well as display. This is good for the environment (newsprint alone in the 1970s accounted for 17% of US waste) but it also means that this data is potentially available for computer processing. The current amount of digitized data is around 20 trillion GB. Much of this increase has been fueled by new structures for storing and retrieving data—video, text, and vast streams of surveillance data—that have arisen since the commercialization of the internet in the 1990s.

This was not always the case. In the nineteenth century, vectors, matrices, and determinants were central tools in engineering and mathematics. As accounting developed professional standards in this period, they naturally gravitated to representations that were matrix-like: accounts \times values for financial reporting, and transactions \times values for journal entries and ledgers. In the twentieth century, spreadsheet tools and rectangular tables of data brought matrices into the computer domain (Fig. 5).

Terminology for matrix data can be confusing. Statisticians and data scientists use different terms for the same thing. For a statistician, predictor variables are used in a model to predict a response or dependent variable. For a data scientist, features are used to predict a target. One synonym is particularly confusing: computer scientists will use the term sample for a single row; a sample to a statistician means a collection of rows.

Within the past decade, twentieth-century concepts of data storage and retrieval have given way to richer modalities. Storage of text has learned from the older discipline of library science; but entirely new approaches are demanded by music and video data.

One reason that machine learning is able to take on new tasks is their ability to analyze data that would be impossible with the matrix based methods of statistics. Statistics evolved in the early twentieth century, building on the matrix algebra that dominated most of science then. Einstein's work with tensors initiated a revolution in physics. Physics "tensors" are not exactly the same as the computer science discipline's "tensor"; the computer science tensor might more correctly be called an array (a multidimensional extension of a matrix). In the language of R, tensors are called arrays, which is probably the more acceptable term in mathematics. Until recently, tensor methods were not available for data analysis. Tensor representation of data is now standard in machine learning models. For example:

- Vector data—2D tensors of shape (samples, features)
- Timeseries data or sequence data—3D tensors of shape (samples, timesteps, features)
- Images—4D tensors of shape (samples, height, width, channels) or (samples, channels, height, width)
- Video—5D tensors of shape (samples, frames, height, width, channels) or (samples, frames, channels, height, width)

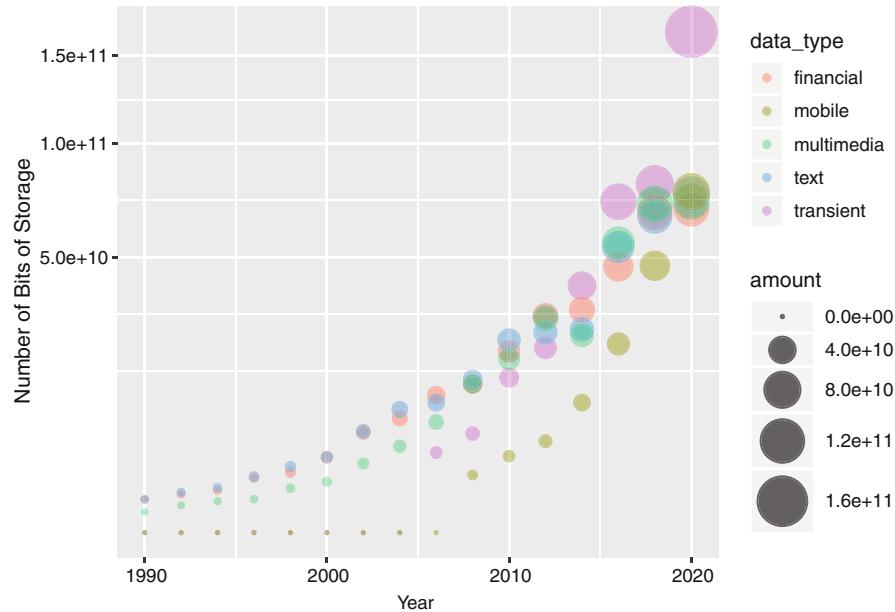


Fig. 5 Growth of stored data (source: EMC Digital Universe)

Tensor data comes from many sources: sensor measurements, events, text, images, and videos. Social networks, drones, surveillance cameras, and other Internet of Things devices are generating massive volumes of data which, from a traditional statistical standpoint, would be considered unstructured. These data streams need the extensions to statistics that are provided by machine learning.

The R language uses seven main storage formats for data: vectors, matrices, arrays, data frames, tibbles (tidyverse), lists, and factors.

Vectors

```
a <- c(1,2,5,3,6,-2,4) # numeric vector
b <- c("one","two","three") # character vector
c <- c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE) #logical vector

## Refer to elements of a vector using subscripts.

a[c(2,4)] # 2nd and 4th elements of vector
```

```
## [1] 2 6
```

Matrices

All columns in a matrix must have the same mode (numeric, character, etc.) and the same length. The general format is

```
vector <- seq(1, 25)
r <- 5
c <- 5

char_vector_rownames <- as.character(seq(1, 5))
char_vector_colnames <- as.character(seq(1, 5))

mat <- matrix(vector, nrow=r, ncol=c, byrow=FALSE,
               dimnames=list(char_vector_rownames, char_vector_colnames))

mat

##      1   2   3   4   5
## 1  1   6  11  16  21
## 2  2   7  12  17  22
## 3  3   8  13  18  23
## 4  4   9  14  19  24
## 5  5  10  15  20  25

mat <- matrix(vector, nrow=r, ncol=c, byrow=TRUE,
               dimnames=list(char_vector_rownames, char_vector_colnames))

mat

##      1   2   3   4   5
## 1  1   2   3   4   5
## 2  6   7   8   9  10
## 3 11  12  13  14  15
## 4 16  17  18  19  20
## 5 21  22  23  24  25

# byrow=TRUE indicates that the matrix should be filled by rows.
# byrow=FALSE indicates that the matrix should be filled by columns (the default).
# dimnames provides optional labels for the columns and rows.
```

```
# generate 5 x 4 numeric matrix
y <- matrix(1:20, nrow=5, ncol=4)

# another example
cells <- c(1, 26, 24, 68)
rnames <- c("R1", "R2")
cnames <- c("C1", "C2")
mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=TRUE,
  dimnames=list(rnames, cnames))

# Identify rows, columns or elements using subscripts.

mat[, 4] # 4th column of matrix

##   1   2   3   4   5
##   4   9  14  19  24

mat[3,] # 3rd row of matrix

##   1   2   3   4   5
## 11  12  13  14  15

mat[2:4,1:3] # rows 2,3,4 of columns 1,2,3

##      1   2   3
## 2   6   7   8
## 3  11  12  13
## 4  16  17  18
```

Arrays

Arrays are similar to matrices but can have more than two dimensions.

```
# Create two vectors of different lengths.
vector1 <- c(5, 9, 3)
vector2 <- c(10, 11, 12, 13, 14, 15)
column.names <- c("COL1", "COL2", "COL3")
row.names <- c("ROW1", "ROW2", "ROW3")
matrix.names <- c("Matrix1", "Matrix2")

# Take these vectors as input to the array.
result <- array(c(vector1, vector2), dim = c(3, 3, 2), dimnames =
list(row.names, column.names,
  matrix.names))
print(result)

## , , Matrix1
##
##      COL1  COL2  COL3
## ROW1      5     10    13
## ROW2      9     11    14
## ROW3      3     12    15
##
## , , Matrix2
##
##      COL1  COL2  COL3
## ROW1      5     10    13
```

```
## ROW2      9     11    14
## ROW3      3     12    15
```

Data Frames, Data Tables, and Tibbles

A data frame is more general than a matrix, in that different columns can have different modes (numeric, character, factor, etc.) similar to SAS and SPSS datasets. Data frames represent the most widely used format for data storage and retrieval in the R language

```
d <- c(1,2,3,4)
e <- c("red", "white", "red", NA)
f <- c(TRUE,TRUE,TRUE,FALSE)
mydata <- data.frame(d,e,f)
names(mydata) <- c("ID", "Color", "Passed") # variable names

# There are a variety of ways to identify the elements of a data frame .

mydata[1:2] # columns 1 and 2 of data frame

##   ID Color
## 1  1   red
## 2  2 white
## 3  3   red
## 4  4   <NA>

mydata[c("ID","Color")] # columns ID and Age from data frame

##   ID Color
## 1  1   red
## 2  2 white
## 3  3   red
## 4  4   <NA>

mydata$x1 # variable x1 in the data frame

## NULL
```

Tibbles are the tidyverse's improvements on data frames, designed to address problems in data analysis at an earlier point. Otherwise they are used exactly as data frames.

In R, the basic rectangular data structure is a `data.frame()`. A `data.frame()` also has an implicit integer index based on the row order. While a custom key can be created through the `row.names()` attribute, the native R `data.frame()` does not support user-specified or multilevel indexes. To overcome this deficiency, two new packages are gaining widespread use: `data.table` and `dplyr`. Both support multilevel indexes and offer significant speedups in working with a `data.frame`.

Lists

An ordered collection of objects (components). A list allows you to gather a variety of (possibly unrelated) objects under one name.

```
a <- b <- "seven"
z <- y <- 0

# example of a list with 4 components -
# a string, a numeric vector, a matrix, and a scalar
w <- list(name="Fred", mynumbers=a, mymatrix=y, age=5.3)

# example of a list containing two lists
list1 <- list(name="Fred", mynumbers=a, mymatrix=y, age=5.3)
list2 <- list(name="Joe", mynumbers=b, mymatrix=z, age=10,11)
```

```
v <- c(list1,list2)

## Identify elements of a list using the [[]] convention.

w[[1]] # 1st component of the list

## [1] "Fred"

v[["mynumbers"]] # component named mynumbers in list

## [1] "seven"
```

Factors

Tell R that a variable is nominal by making it a factor. The factor stores the nominal values as a vector of integers in the range [1... k] (where k is the number of unique values in the nominal variable), and an internal vector of character strings (the original values) mapped to these integers.

```
# variable gender with 20 "male" entries and
# 30 "female" entries
gender <- c(rep("male",20), rep("female", 30))
gender <- factor(gender)
# stores gender as 20 1s and 30 2s and associates
# 1=female, 2=male internally (alphabetically)
# R now treats gender as a nominal variable

summary(gender)

## female     male
##      30      20

# An ordered factor is used to represent an ordinal variable.

# variable rating coded as "large", "medium", "small"

rating <- c( "medium","large", "small")
rating <- ordered(rating)
rating

## [1] medium large  small
## Levels: large < medium < small

# recodes rating to 1,2,3 and associates
# 1=large, 2=medium, 3=small internally
# R now treats rating as ordinal
```

R treats factors as nominal (i.e., label or naming) variables and ordered factors as ordinal variables in statistical procedures and graphical analyses. You can use options in the factor() and ordered() functions to control the mapping of integers to strings (overriding the alphabetical ordering). You can also use factors to create value labels.

Useful Functions for Dataset Inspection and Manipulation

```
library(kableExtra)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

arry <- read.csv(system.file("extdata", "morph_array.csv", package = "auditanalytics",
mustWork = TRUE))
```

```

array %>%
  kable(longtable=T) %>%
  kable_styling(
    bootstrap_options = c("striped", "hover", "condensed"),
    full_width = F,
    font_size=10)

length(arr) # number of elements or components
str(arr) # structure of an object
class(arr) # class or type of an object
names(arr) # names

list1 <- list("a", "b", "c")

c(list1, list1)      # combine objects into a vector
cbind(list1, list1) # combine objects as columns
rbind(list1, list1) # combine objects as rows

list1 # prints the object
newobject <- edit(list1) # edit copy and save as newobject
fix(list1)           # edit in place

ls()      # list current objects
rm(list1) # delete an object

```

Other Data Types

R supports almost any conceivable type of data structure. A few additional structures that are important to account are:

- Time series data records successive measurements of the same variable. It is the raw material for statistical forecasting methods, and it is also a key component of the data produced in surveillance.
- Spatial data structures, which are used in mapping and location analytics, are more complex and varied than rectangular data structures. In the object representation, the focus of the data is an object (e.g., a house) and its spatial coordinates. The field view, by contrast, focuses on small units of space and the value of a relevant metric (pixel brightness, for example).
- Graph (or network) data structures are used to represent physical, social, and abstract relationships.

Further Study

This chapter should be seen as a survey of what is possible for auditors interested in an analytical, algorithmic approach to auditing. I have tried to delineate some of the parts of data analytics you should familiarize yourself with. Additionally, Wikipedia is a great resource for statistics; some years ago, there were many disparate sites which publicized a great deal of useful information, and most of their content has been transferred to Wikipedia. The R package vignettes and package writeups are excellent sources of empirical statistical methods which might be motivated by particular data structures unique to accounting and auditing. The Journal of Statistical Software was founded by Jan de Leeuw in 1996, the year before the Comprehensive R Archive Network (CRAN) first made R and contributed R packages widely available on the Internet. Within a few years, R came increasingly to dominate contributions to JSS, which is often the definitive source for information behind the packages in R. Many very helpful solutions, articles and tutorials are continually being created by the large group of R experts that contribute to Internet resources like [stackoverflow](#) and [R-bloggers](#). Searches of the Internet will also find an increasing number of “data camp” type resources that are parts of larger educational programs. Whatever your challenge, you should be able to find assistance through one or more of these resources.

R Packages Required for This Chapter

This chapter's code requires the following packages to be installed: `tidyverse`, `kableExtra`, `plotluck`, `broom`, `ggplot2`, `lubridate`, and `reshape2`.

Note: There are two steps to using a package. First it must be *installed*, i.e., copied to a location on your computer where R can access it. Then it must be *loaded* into the working memory of R. To install, for example, the `tidyverse` package, type `install.packages("tidyverse")` and then press the *Enter/Return* key. To load the previously installed package type `library(tidyverse)`. The `tidyverse` package is now available for use by your program code.

References

- Bush, Vannevar, and Vannevar Bush. 1945. As We May Think. *Resonance* 5(11).
- Chen, Peter Pin-Shan. 1976. The Entity-Relationship Model—Toward a Unified View of Data. *ACM Transactions on Database Systems (TODS)* 1 (1): 9–36.
- Cochran, William G., Frederick Mosteller, and John W. Tukey. 1954. Principles of Sampling. *Journal of the American Statistical Association* 49 (265): 13–35.
- McCarthy, William E. 1979. An Entity-Relationship View of Accounting Models. *Accounting Review* LIV (4): 667–686.
- _____. 1982. The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. *Accounting Review* LVII (3): 554–578.
- Stigler, Stephen M. 1986. *The History of Statistics: The Measurement of Uncertainty Before 1900*. Cambridge: Harvard University Press.
- Tukey, John W. 1980. We Need Both Exploratory and Confirmatory. *The American Statistician* 34 (1): 23–25.

Analysis of Accounting Transactions



Audit Procedures and Accounting Cycles

Audit procedures are designed around the accounting cycles. There are both practical and theoretical reasons for this. The accounting cycles define the particular processing of transactions, and interim tests focus entirely on particular transactions and their processing. The same economic event will be reflected in multiple transactions, and there are economies in designing audits around this fact. In substantive year-end tests, the income statement accounts are transaction based, and their accurate estimation depends on understanding the accounting cycles.

Guidance from the PCAOB's AS 2100 through 2600 (available on the Public Company Accounting Oversight Board's website at pcaobus.org) can be used to structure the sampling and analysis decision-making in an audit:

1. Planning and risk assessment phase: Using the audit risk assessment matrix, determine which transaction flows will have predicted error rates greater than tolerable rate, or predicted error amounts greater than tolerable misstatement amount. Their function is to provide the foundation for determining initial audit scope, the budget and predicted contract cost of the audit, and for selecting particular audit tests that will be performed. This work typically uses little or no transaction data from the client, rather relies on the prior years audit conclusions and tests, benchmarks from similar firms in the client's industry and available audit resources.
2. Interim compliance tests (tests of transaction, mid-year tests): Each significant transaction processing step may potentially be tested, but the audit program typically restricts scope of the interim tests to those found to be "at risk" in the planning and risk assessment phase of the audit. Where risk assessment matrix predictions exceed tolerable rates or amounts, plan to use attribute sampling (to find actual rate of error); for other tests, use discovery sampling to detect the possibility that error rate might be intolerable. The product of this phase is the internal control report (specified in AU section 319 and PCAOB AS No. 5) which may also be the basis for assertions in the Sarbanes–Oxley letter signed by management.
3. Substantive tests: Substantive tests assess the probability of material error in the accounts summarized in the various financial reports distributed to shareholders. Based on error rates discovered in interim compliance tests (and reported in the internal control memo) and expand the minimum tests of monetary balances on the trial balance.

Corporate accounting systems are universally implemented on computer systems today—either in-house systems, or increasingly through service bureaus and software as a service (SaaS) infrastructure. Auditing is fundamentally about drawing conclusions on the financial statement balances (which are just a particular variety of summary statistic) from detailed investigations of individual accounting transactions (i.e., the economic events that record the data for these account balances). Since both balances and transactions are recorded in the computer databases (not on paper) today, audit sampling can only be completed with computer programs. Additionally, corporations are retaining more accounting data than ever, partly because of new regulations from e.g. Sarbanes–Oxley, and partly because investors value more information, and corporations strive to provide it. Though this creates some up-front costs for auditors, it also makes the selection, audit, and analysis of transaction samples much more efficient. The following provides the basic statistics that you need to understand to properly conduct an audit—either test of transaction (mid- year, internal control) or substantive (year-end, balance) tests.

The Origin of Accounting Transactions

Accounting transactions begin as real-world events involving a firm—processes like sales that are localized in value, time, and space. The vast majority of these events are irrelevant to accountants—events that have no impact on the wealth of the firm. Eligibility for an event to become an accounting transaction is determined by commercial law—whatever events will create income, financial obligations, or financial allocations are likely to become accounting transactions.

The route from event to accounting transaction involves three components:

1. The definition of the boundaries of the firm. Thus the same event may be an accounting transaction on a divisional financial statement, but will not be incorporated into the consolidated financial statements.
2. The relevant commercial law applying to the event-transaction. Often this pertains to the definition of “ownership” of an asset, or “obligation” for a promise, warranty, or contract. Where ownership and obligations cannot be determined, the firm may need to report “contingent” liabilities or assets in a note. Before the nineteenth century, law mainly related to product transactions, but the twentieth century witnessed an explosion of intricacies in the law which have made this determination much more difficult.
3. The capture system for recording the event-transaction. Auditor responsibility for capture of transactions has expanded significantly over the past two decades, particularly with the Sarbanes–Oxley Act of 2002. Firms are expected to have capture systems to assure that where there is a legal ownership or liability, that it shows up in the accounting reports.

Once an event has successfully bridged the transition from real world to captured and recorded accounting transaction, it needs a name and a sign. This is the responsibility of the *“Chart of Accounts.”* The Chart of Accounts is an artificial categorization system that is set up by the firm’s controller or financial executive. It varies from firm to firm, reflecting the idiosyncrasies of the firm’s particular business.

There are general guidelines for constructing any Chart of Accounts, guided by generally accepted accounting principles (GAAP) and needed to assure that investors can easily compare investment in one firm with another. High volume transactions such as sales and payroll will have their own journals—today these are typically computer systems dedicated to processing a single type of transaction. Less common transactions may receive less attention, and thus may be more error prone.

In addition, there may be specific idiosyncrasies of GAAP that modify the application and interpretation of any of these four filters.

Thus when auditors speak of a “transaction stream” they are fundamentally referencing some real-world event located in time, space, and value, but which has been processed through a set of filters:

1. capture.
2. legal ownership and valuation.
3. firm boundaries.
4. classification.
5. GAAP.

This complexity is fundamental to process of accounting, and is necessary for assuring that stakeholders have informative and comparable financial statements. The remainder of the accounting process is a matter of stylized summarization of the transactions processed through these filters. The summarization is inherently linear-additive with a few non-linear exceptions applied to allocations over time or output. Sadly, the most consequential part of accounting—this five stage filtering process—is also the part that receives very little formal attention in auditing.

Audit Tests as Learning Models

Auditing may be seen as a sequence of evidence collection and analyses that help the auditor learn enough to render an opinion.

Conceptually, the tests involve a sequence of inferences:

1. Audit planning will have set scope and sample size values for audit tasks that in general ‘seek to discover’ (i.e., discovery sampling) whether or not transaction processing is or is not in control. This is usually stated as an assessment of error rates rather than dollar magnitudes of error.

2. If errors are discovered in a particular transaction flow, then sufficient additional evidence must be gathered to assess the rate of error occurrences in the transaction flow (i.e., attribute sampling).
3. Transaction flows with error rates that are sufficiently high to be deemed “out-of-control” (i.e., intolerable from the audit perspective) are listed on the Internal Control Memo generated at the end of the Interim compliance audit.
4. Financial statement account balances affected by any transaction flows with error rates that are sufficiently high to be deemed “out-of-control” will have their sample sizes in the audit program for year-end substantive tests adjusted to assure an accurate assessment of the existence of material error in the audited financial statements.

Various ad hoc and formal models have been proposed for the gradual accumulation of knowledge of the accounts throughout the audit. My own preference is for systems of audit task evidence accumulation modeled as Bayesian conjugate priors. The most general of these involves the Gamma distribution, but we will simplify some of the steps using the simpler one-parameter Poisson distribution. Both distributions take on values on the positive quadrant, and thus are more suitable for modeling accounting distributions than the Normal distribution.

Working with Dates

Time is an essential component of auditing. Income statement accounts represent sums of transactions within strictly set periods, and balance sheet accounts are stated at a specific time. Auditors routinely test timing of transactions for cutoff errors and frauds such as check kiting allows fraudsters to build up a balance in one bank by writing bad checks on another bank; or lapping involves stealing a customer payment and using any additional payments from that customer to cover the theft. Transactions are always time-stamped, and the internal control, processing, and auditing of client systems are heavily dependent on timing of transactions.

Unfortunately, in computer environments, date-time data can be frustrating, error prone, and difficult to work with, even though they are an essential element of all accounting systems. Fortunately, the R package *lubridate* simplifies date-time manipulation and calculations in a fashion unparalleled in other software.

The *lubridate* package makes it easier to do the things R does with date-times and possible to do the things that Base-R does not.

```
library("lubridate")

## 
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
## 
##     date

## parsing of date-times: ymd(), ymd_hms, dmy(), dmy_hms, mdy(), e.g.,
ymd(20101215)
```

```
## [1] "2010-12-15"
```

```
mdy("4/1/17")
```

```
## [1] "2017-04-01"
```

```
## simple functions to get and set components of a date-time, such as year(),
month(), mday(), hour(), minute() and second():

bday <- dmy("14/10/1979")
month(bday)
```

```
## [1] 10
```

```
wday(bday, label = TRUE)

## [1] Sun
## Levels: Sun < Mon < Tue < Wed < Thu < Fri < Sat

year(bday) <- 2016
wday(bday, label = TRUE)

## [1] Fri
## Levels: Sun < Mon < Tue < Wed < Thu < Fri < Sat

## lubridate has helper functions for handling time zones: with_tz(), force_tz()
time <- ymd_hms("2010-12-13 15:30:30")
time

## [1] "2010-12-13 15:30:30 UTC"

with_tz(time, "America/Chicago")

## [1] "2010-12-13 09:30:30 CST"

force_tz(time, "America/Chicago")

## [1] "2010-12-13 15:30:30 CST"
```

Lubridate also expands the type of mathematical operations that can be performed with date-time objects, and defines three time span classes that play roles in auditing:

- durations, which measure the exact amount of time between two points.
- periods, which accurately track clock times despite leap years, leap seconds, and day light savings time.
- intervals, a protean summary of the time information between two points.

Accounting Transactions

Accounting transactions are the “raw data” in an accounting system. They represent measurements of economic events such as sales or purchases, that cross the border of the firm. Transactions are “measured” and “recorded” on *journal entries*, the documents of original entry. Journal entries were originally simply reminders and descriptive notes recorded in a journal book. But with more formal, mathematical developments in accounting, they evolved into their current form of documents of original entry. For high volume transactions, there will typically be specialized journals, e.g., the Sales Journal, or Purchase Journal, that consolidate the recording of these types of transactions in the accounting systems.

Before the twentieth century, accounting was concerned mainly with computing the asset value of the firm, and focused on economic events that made the firm richer or poorer. The twentieth century saw a transition to a more holistic perspective, where transactions were recorded to track, in addition to wealth, managerial performance, and stakeholder interests.

In practice, our minimum level of resolution in transactions and accounts is one dollar (a single monetary unit). This resolution limit is in fact the basis of audit procedures such as dollar-unit sampling. We are not limited to dollar levels of resolution; annual reports may use thousands or millions of dollars as their level of resolution. Additionally, accounting and double-entry bookkeeping are methods of generating summarizations of transactions, and these often take the form of $value \times quantity$ statistics.

In the histogram below which was constructed from a real-world transaction journal, we use qplot (an abbreviated form of ggplot) to produce a histogram of discrete transaction values at a resolution of one dollar. The level of resolution determines the number of histogram bins, in this case 30. Human cognition tends to be overwhelmed by large numbers of bins, which is why we often see financial statistics summarized at levels of resolution larger than one dollar (Fig. 1).

This histogram displays several characteristics commonly seen in actual transaction probability distributions:

1. the distribution is left-bounded at zero.

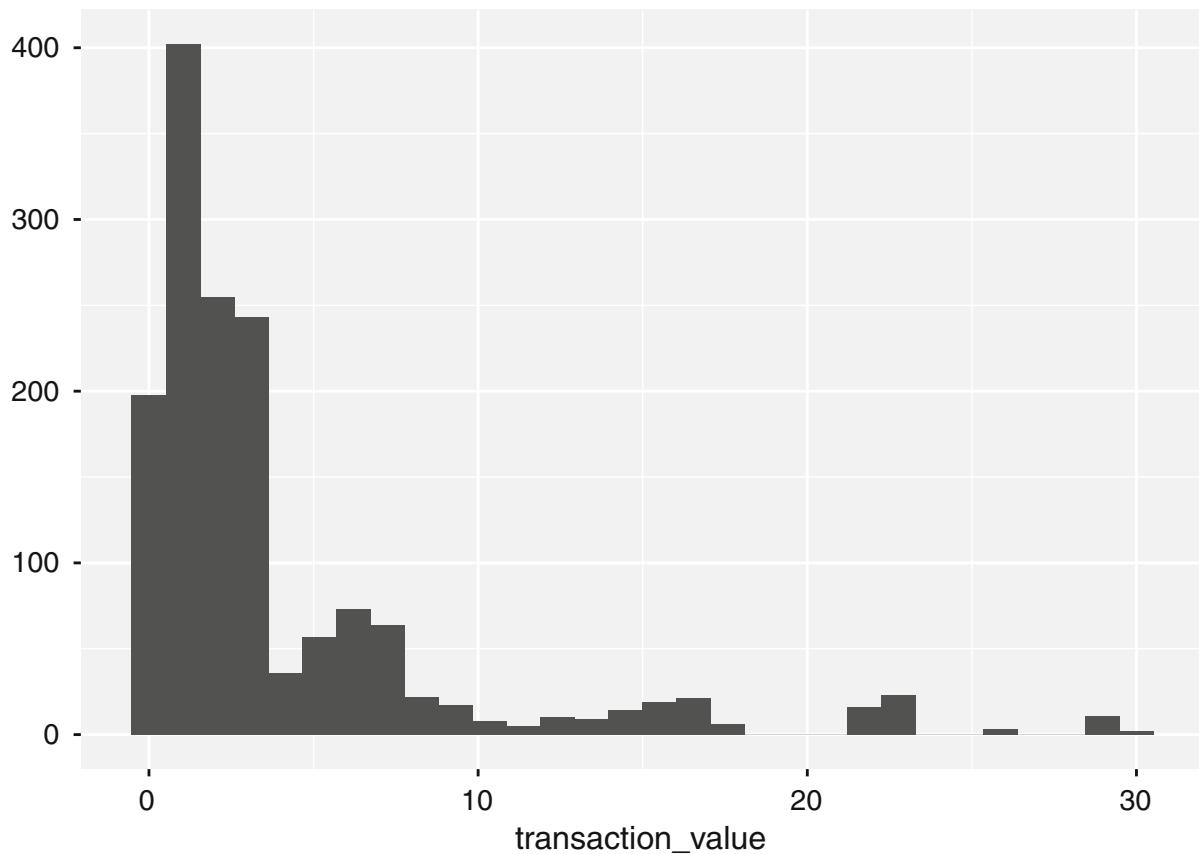


Fig. 1 Typical distribution of transaction values at dollar-unit level of resolution

2. the distribution may be zero-inflated, i.e., it may have a substantial number of zero measurements.
3. the distribution is multimodal, with a succession of value clusters trailing off to the right.

The first characteristic is built into accounting systems by design; account entries should not contain negative values. Rather, when a transaction reduces an account balance, e.g., a reduction of Sales when there is a Sales Return, the reduction should be recorded in the “contra” account Sales Returns and Allowances. This convention goes back to the birth of accounting. In ninth century Baghdad, Al-Khwarizmi solved linear and quadratic equations using algebraic methods derived from the work of the Indian mathematician Brahmagupta, who invented negative numbers. Nonetheless, Al-Khwarizmi felt that negative results were meaningless. In his treatise on the laws of inheritance (where the double-entry system was first published) Al-Khwarizmi instructed his readers to represent negative quantities as debts—i.e., contra-accounts.

The second characteristic reflects the existence of “convenience” transactions that record an economic event that has no effect on firm wealth. These could reflect estimates, adjustments, or other ad hoc entries.

The final characteristic is most interesting. The verdict is still out on why we see this in transaction streams, but it seems to have several sources. Transactions may be produced by several independent processes, perhaps representing different locations, product values, or other factor. Another explanation arises from the mathematics of many transaction amounts—the unit price times the number of items in the transaction. Even where values and quantities are unimodally distributed, their product can be multimodal, as is seen in the following graph (Fig. 2).

```
library(tidyverse)
library(ggplot2)

price <- rpois(1000, 2)
quantity <- rpois(1000, 10000)
```

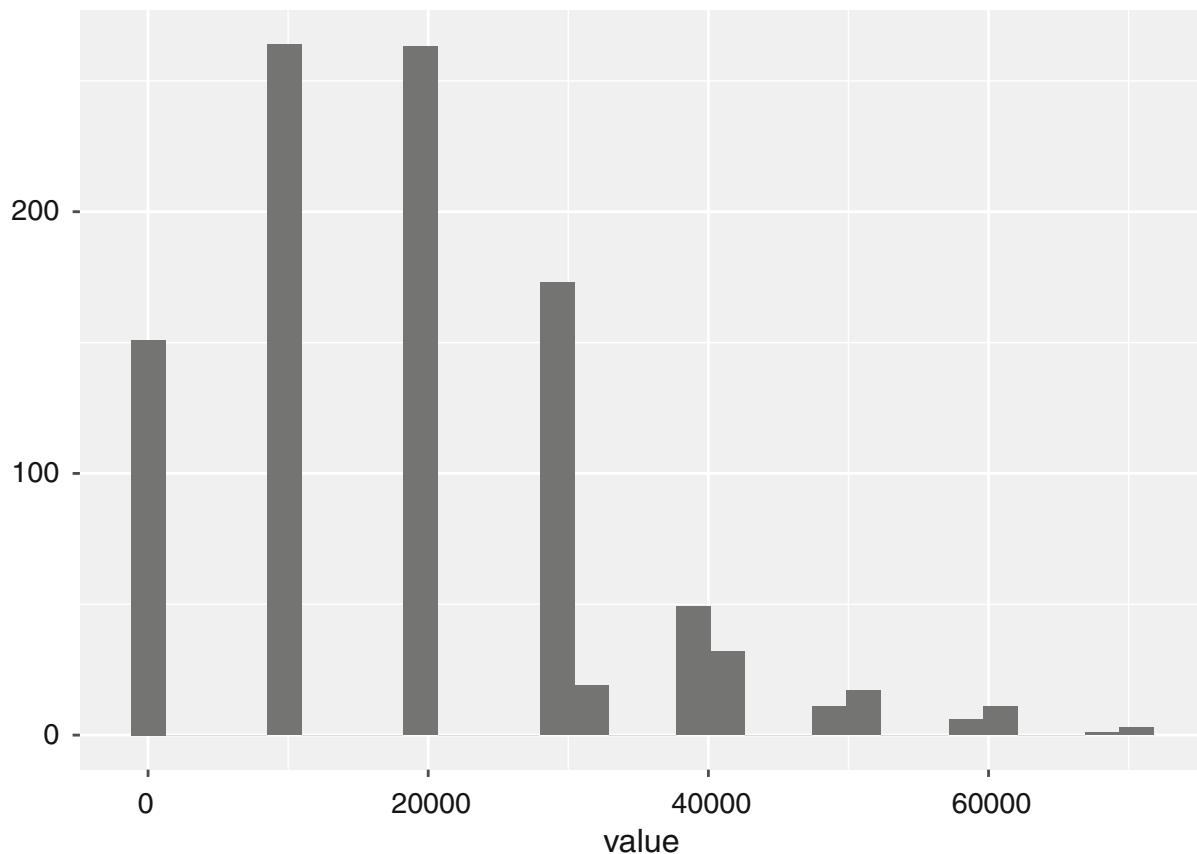


Fig. 2 Artificially Generated transactions from value × quantity

```
value <- price*quantity
qplot(value, geom="histogram")
```

```
qplot(value, geom="density")
```

Because accounting systems use linear operators, and summation is central to most accounting computations, it is natural to assume—when sampling, summarizing, or estimating errors—that errors are Normally distributed, because of the Central Limit Theorem (CLT). The CLT provides useful approximations, and is appropriate in many situations. Nonetheless, the auditor should not blindly invoke CLT convergence to justify Normality assumptions in the search for material errors in the financial statements (Fig. 3).

In a search for material errors, it may not be the entire transaction or account distribution that is important, but just the upper tail. Audit decisions tend to focus on the existence of material misstatements, which limit them to the upper tail. For a broad range of distributions, the asymptotic upper tail distribution can be inferred from the Fisher–Tippett–Gnedenko (FTG) theorem, an extreme value counterpart to the CLT. The FTG Theorem proves that extreme values can only converge to either the Gumbel, Fréchet, or Weibull distributions or they do not converge at all. FTG convergence applies only to the right tail of the distribution, and is widely invoked in insurance risk models. This can be very useful in practice, empirical distributions in finance can be complicated, with multiple modes which may not be stable over time. Being able to more accurately characterize accounting distributions that matter allows the auditor to potentially reduce sample sizes, and lower the cost of auditing.

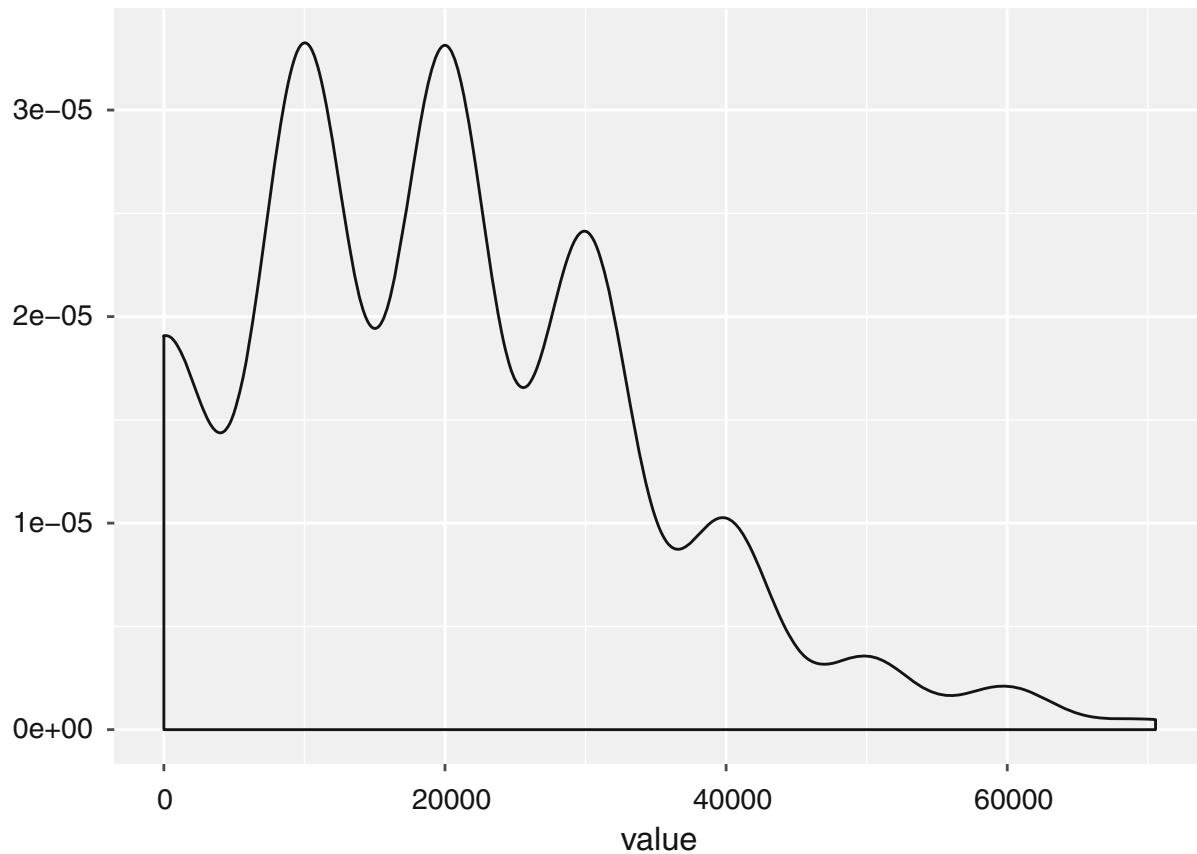


Fig. 3 Artificially Generated transactions from value \times quantity

Couching Institutional Language in Statistical Terms

Generally, auditing assumes underlying Gaussian distributions of account and transaction values. Evidence has existed for some time (Loebbecke and Steinbart 1987) that empirical distributions of account and transaction values are kurtotic and can be multimodal. Gaussian assumptions are made for convenience, and there has been no clear evidence to date this assumption is unreasonable. Auditors are concerned with the potential errors in audit decision-making that can arise from mis-classifying audit asset and transaction distributions. If these are large, then auditors are either collecting more evidence than needed (driving up audit costs) or unnecessarily incurring additional risk of an incorrect audit opinion (and thus potential litigation). The AICPA defines two aspects to sampling risk when performing tests of controls:

1. The risk of assessing control risk too low represents the risk that an audit sample supports the conclusion that the design and operation of an internal control is effective when in fact it is not.
2. The risk of assessing control risk too high represents the risk that an audit sample supports the conclusion that the design and operation of an internal control is not effective when in fact it is effective.

The AICPA defines two aspects to sampling risk when performing substantive tests:

- (1) The risk of incorrect acceptance represents the risk that an audit sample supports the conclusion that a material misstatement does not exist when in fact a material misstatement does exist. This risk is similar to the risk of assessing control risk too low.
- (2) The risk of incorrect rejection represents the risk that an audit sample supports the conclusion that a material misstatement exists when in fact a material misstatement does not exist. This risk is similar to the risk of assessing control risk too high.

Ultimately, an audit must provide shareholders in publicly traded securities an independent verification of the “fairness” of accounting reports. Audit procedures are dictated by two objectives: cost efficiency in data collection and analysis, and “fairness” in reporting. Fairness is generally interpreted as absence of “material error” in the accounting reports. For audits performed by an outside audit firm, risk assessment is a very crucial stage before accepting an audit engagement.

It is an integral part of determining the audit tasks that will be performed in the audit program. According to ISA315, Understanding the Entity and its Environment and Assessing the Risks of Material Misstatement, “the auditor should perform risk assessment procedures to obtain an understanding of the entity and its environment, including its internal control.” Evidence relating to the auditor’s risk assessment of a material misstatement in the client’s accounting statements. Then, the auditor obtains initial evidence regarding the classes of transactions at the client and the operating effectiveness of the client’s internal controls. In auditing, audit risk includes inherent risk (IR), control risk (CR), and detection risk (DR).

The audit risk model expresses the risk of an auditor providing an inappropriate opinion of a commercial entity’s accounting statements and is calculated: $AR = IR \times CR \times DR$. In this formula, IR refers to the risk involved in a business or transaction. Example, transactions involving exchange of cash may have higher IR than transactions involving settlement by checks. CR refers to the risk that a misstatement could occur but may not be detected and corrected or prevented by entity’s internal control mechanism. DR is the probability that the audit procedures may fail to detect existence of a material error or fraud. While CR depends on the strength or weakness of the internal control procedures, DR is either due to sampling error or human factors.

Due to the breadth of qualitative and quantitative material that analytical procedures consider, audit risk models have typically been interpreted as abstract heuristics that provide a framework for auditor judgment, but not methods to explicitly process the external data. But this need not be the case, as there are formal statistical models for assessment of financial reports using external data used by investors. The most influential of these is the efficient markets hypothesis (EMH) which states that asset prices reflected in these markets fully reflect all available information, and provides a strong motivation for reliance on asset markets for decision-making. The distribution of abnormal returns in US asset markets provides complete and unbiased information about corporate performance. From the auditor’s perspective, in acquiring plausible financial and non-financial evidence prior to and during actual fieldwork, inexpensive evidence that can fully reflect all available information about a firm and the industry in which it operates should be extremely attractive. Analytical procedures incorporating information-rich asset market statistics hold the potential to substitute for increased audit scope or risk without incurring additional fieldwork. Tests of market efficiency have typically been carried out jointly with the Capital Asset Pricing Model (CAPM). Both market efficiency and audit models in practice favor an assumption that transaction time series are Gaussian distributed. Decision makers understand that this is only an approximation of reality, but it is often considered “good enough” with the Central Limit Theorem (CLT) commonly invoked as justification. But as was argued earlier, CLT convergence is most appropriate when trying to generate unbiased estimates for accounts, rather than for assessing extreme value situations such as material errors in financial statements.

Transaction Samples and Populations

Auditing depends heavily on sampling to control the cost of auditing. A statistical sample is a subset of the population. The population is all of the transactions relevant to a particular audit or procedure. For example, in confirmation of accounts receivable (A/R) the population is all of the accounts receivable from customers buying on credit for the year (assuming an annual audit). Audit samples are usually several orders of magnitude smaller than the population. For example, auditors might take a sample of 100 units (transactions or dollars) from a population of 1 million accounts receivable or 0.01% of the items in the population.

The sampling base or metric is the unit used to draw the sample. Monetary unit sampling (one dollar is a sample item) is applied at year end for substantive testing—determining whether the financial statement account balances contain material errors. Transaction or Record sampling (one accounting transaction is a sample item) is applied prior to year-end for internal control testing—determining whether the inherent level of transaction processing error risk that exists in each accounting cycle.

The AICPA provides guidelines on sampling in several standards:

- Risk assessment standards (SAS Nos. 104-111).
- Defining Professional Requirements Standard (SAS No. 102).
- Guidance on audit documentation (SAS 103).
- Communicating internal control related matters (SAS 112 and SAS 115).
- Audit Risk Alerts such as Communicating Internal Control Related Matters in an Audit—Understanding SAS No. 115

These discuss several approaches to the audit sampling process.

- Statistical audit sampling in tests of controls and for substantive tests of details.
- Non-statistical audit sampling in tests of controls and for substantive tests of details.

- Monetary unit sampling.
- Attribute sampling.
- Multi-location sampling considerations.

Additionally, the AICPA Audit and Accounting Manual addresses problems in small- and medium size CPA practices. It explains engagement steps from planning, to performing procedures, to issuing reports:

- Guidance on internal controls.
- Audit documentation guidelines.
- Illustrative confirmation letters.
- Illustrative engagement and representation letters.
- Illustrative auditor's reports.

The AICPA discusses the following types of sampling in drawing conclusions from audit evidence, which are discussed in depth later in the chapter on *Design of Audit Programs*:

1. Judgmental sampling: Items are audited based on personal hunches and convenience involving a subjective selection of items for testing and a subjective evaluation of the results.
2. Random sampling: Applies random number generators to assure that each population unit or item has an equal probability of being selected for the sample.
3. Fixed-interval sampling: Specify an interval and a random starting point which must be greater than zero and less than or equal to the selection interval.
4. Cell or random-interval sampling: Specify an interval and a random seed which is the basis for a series of pseudo-random numbers.
5. Conditional sampling: Apply a condition to subject only a portion of the population to selection.
6. Stratified sampling: A process of dividing a population in subgroups each of which is a set of sampling units with similar characteristics. Stratified sampling may be considered a particular form of conditional sampling—conditioned on item size (in dollars, etc.).
7. Transaction or Record sampling: Treats each recorded transaction equivalently.
8. Monetary Unit Sampling: Treats each recorded dollar equivalently.
9. Estimation sampling: Sampling with an objective to estimate the proportion of errors to see if it is less than some acceptable level.
10. Acceptance sampling: Sampling with an objective to reject or accept the population under certain conditions.
11. Discovery sampling: Sampling with an objective to determine if the population is error free.

Accounting Cycles

An accounting cycle begins when accounting personnel create a transaction from a source document and ends with the completion of the financial reports and closing of temporary accounts in preparation for a new cycle. The five accounting cycles (with main functions) are:

- (a) Revenue cycle.
 - (1) Sales orders,
 - (2) Cash receipts.
- (b) Expenditure cycle. This cycle focuses on two separate resources; inventory and human resources and is often considered two separate cycles; purchasing and payroll/HR.
 - (1) Inventory/purchasing;
 - (2) Accounts payable;
 - (3) Payroll;
 - (4) Cash payments.
- (c) Conversion cycle (Production cycle).
 - (1) Production;
 - (2) Cost accounting.

- (d) Financing (Capital Acquisition and repayment).
 - (1) Borrowing/repayment;
 - (2) Issuing stock;
 - (3) Dividends;
 - (4) Cash management.
- (e) Fixed assets.

Substantive Testing

The purpose of substantive procedures is to provide audit evidence as to the completeness, accuracy, and validity of the information contained in the accounting records or in the financial statements. Substantive testing involves detailed examination of the monetary value of the account balances to determine their accuracy and to draw conclusions about the materiality of the error amounts in the accounts. The extent and nature of substantive testing depends upon the decision taken about the effectiveness of the systems of internal control. In substantive testing, statistical sampling is used to obtain monetary estimates of the total error amount or confidence limits for the total error amount in a particular account. The objective is to obtain reliable confidence limits.

Metrics and Estimates

Statisticians estimate; business analysts measure. Statisticians often use the terms *statistic* and *estimate* for values calculated from the data, to draw a distinction between interpretations of the data, and the “true” state of affairs. Data scientists and business analysts are more likely to refer to such values as a *metric*. The difference reflects the approach of statistics versus data science: accounting for uncertainty lies at the heart of the discipline of statistics. Business or organizational objectives are the focus of data science.

In the past, the auditors initial step when confronted with a new database is to “foot” the dataset (i.e., compute a total) and “agree” that total to the client’s records (i.e., see whether client’s records agree with the computed total). This is done with the “sum” command in R.

```
disburse <- read.csv(system.file("extdata", "ch_3_AP_disbursements_journal.csv",
  package = "auditAnalytics", mustWork = TRUE))

devtools::install_github("westland/auditAnalytics")
library(auditAnalytics)

summary(disburse)

cat('\n\n Footed total of disbursements journal = ', sum(disburse$amount_paid))
```

R has a number of packages that generate basic statistics from the data, beyond that of the built-in *summary* command. Here are three of the most useful, applied to our banking industry dataset.

```
devtools::install_github("westland/auditAnalytics")
library(auditAnalytics)

library(Hmisc)

bank_fin <- read.csv(system.file("extdata", "ch_2_yahoo_fin.csv",
  package = "auditAnalytics", mustWork = TRUE))

Hmisc::describe(bank_fin)
```

The *describe* and *describeBy* functions of the *psych* package offer rich reporting of summary statistics, though much of this may be superfluous in auditing.

```
library(psych)
## The psych package allows specific summary-by-group variation, describeBy

psych::describe(bank_fin)
psych::describeBy(bank_fin, bank_fin$name)
```

If the auditor wishes to use summary statistics for further processing, these can be formatted into data frames using the *tidy* function in *broom*; alternatively, the data frames can be used to print formatted tables, which may be included in audit papers.

```
library(kableExtra)
library(broom)
library(pastecs)

## Tidy these up and write them as a formatted table, using kableExtra

pastecs::stat.desc(bank_fin) %>%
  broom::tidy() %>%
  kable(longtable=T, caption = "Summary Statistics") %>%
  kable_styling(full_width = F)
```

However the auditor chooses to summarize a dataset, there are certain common statistics that will be generated.

1. Count: How many transactions (or rows) does the dataset contain.
2. Information (in a dataset): Potentially the maximum amount that the data can tell you about is the variability of a particular construct. Information is context specific (you cannot use a weather database to predict poker outcomes) and there is no absolute measure. In accounting transactions, the count of data is often not a good measure of information content. Accounting data is highly multicollinear—the same economic event (e.g., a sale) is repeatedly recorded in the data (e.g., as an account receivable of the same amount, as an inventory reduction of a related amount, as a collection, and so forth)
3. Mean: The basic estimate of location. The arithmetic mean (or just mean) is the sum of all the values divided by the number of values.
4. Trimmed mean: An arithmetic mean that removes a small designated percentage of the largest and smallest values before calculating the mean. After removing the specified *outlier* observations, the trimmed mean is found using a standard arithmetic averaging formula.
5. Outlier: An outlier is any value that is very distant from the other values in a dataset. The exact definition of an outlier is somewhat subjective. Outliers are exceptions that should be further investigated. Being an outlier in itself does not make a data value invalid or erroneous. Nonetheless, outliers are often the result of data errors. When outliers are the result of bad data, the mean will result in a poor estimate of location, while the median will still be valid. Outliers are sometimes informative and sometimes a nuisance, in anomaly detection the points of interest are the outliers, and the greater mass of data serves primarily to define the “normal” against which anomalies are measured.
6. Median: The median is the middle number on a sorted list of the data. It provides a robust estimate of location since it is not influenced by outliers (extreme cases) that could skew the results. An outlier is any value that is very distant from the other values in a dataset. The exact definition of an outlier is somewhat subjective, although certain conventions are used in various data summaries and plots. Being an outlier in itself does not make a data value invalid or erroneous. Still, outliers are often the result of data errors such as mixing data of different units (kilometers versus meters) or bad readings from a sensor. When outliers are the result of bad data, the mean will result in a poor estimate of location, while the median will be still valid. In any case, outliers should be identified and are usually worthy of further investigation.
7. Variance: The primary variability statistic (though less commonly used than standard deviation). It is the sum of squared deviations from the mean divided by $n - 1$ where n is the count. Location is just one dimension in summarizing a feature. A second dimension, variability, also referred to as dispersion, measures whether the data values are tightly clustered or spread out. At the heart of statistics lies variability: measuring it, reducing it, distinguishing random from real variability, identifying the various sources of real variability, and making decisions in the presence of it.

8. Deviations (errors, residuals): The difference between the observed values and the estimate of location.
9. Standard deviation (l2-norm, Euclidean norm): The square root of the variance. Standard deviation (sd) is more commonly used than variance, because its values are easy to compare to linear estimators such as the mean. Measures of variability are borrowed from the field of mechanics, where a moment is an expression involving the product of a distance and another physical quantity. In statistics, the first moment of a distribution is the mean, the second moment the variance, and third and fourth moments describe skewness and kurtosis. Higher moments are not particularly robust, so tend not to be used.
10. Degrees of freedom (df): Another concept borrowed mechanics. Where a mechanism is made of several parts, it is the number of possible independent relative motions between the parts of the mechanism. In statistics, the number of degrees of freedom is the number of values in the final calculation of a statistic that are free to vary. It is a rough way to determine whether a dataset is large enough to render a decision based on a particular hypothesis or statistic. Degrees of freedom considerations are important in computing sample statistics, as adjustments are required to assure the statistic is not biased.
11. Mean absolute deviation (mad, l1-norm, Manhattan norm): The mean of the absolute value of the deviations from the mean.
12. Range: The difference between the largest (max) and the smallest (min) value in a dataset.
13. Skew and Kurtosis: In statistics, the first moment of a distribution is the mean, the second moment the variance, and third and fourth moments describe skewness and kurtosis. Higher moments are not particularly robust, so tend not to be used. Typically central moments are reported, which means that the distance is computed from the mean of the distribution of transactions, rather than from zero.
14. Order statistics: Metrics based on the data values sorted from smallest to biggest.
15. Percentile (quantile): The value such that $p\%$ of the values take on this value or less and $(100 - p)\%$ take on more than this value. The difference between the 75th percentile and the 25th percentile is called the interquartile range (IQR).
16. Correlation coefficient: A metric that measures the extent to which numeric variables are associated with one another (ranges from -1 to $+1$).
17. Scatterplot: A plot in which the x-axis is the value of one variable, and the y-axis the value of another.

Important Concepts in Probability and Statistics

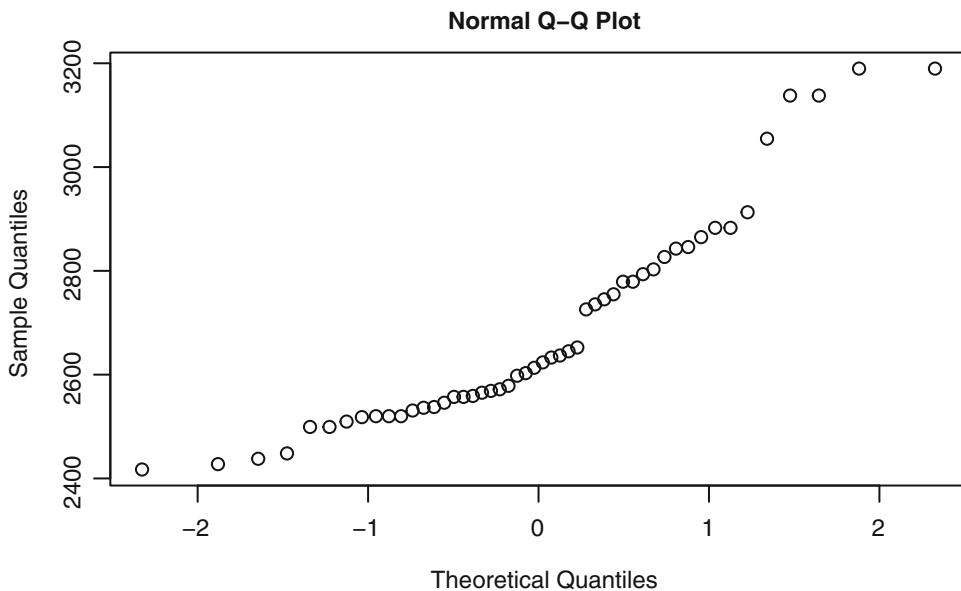
Probability distributions are models describing the variability of data or the underlying population from which the data is drawn. There are perhaps 50 or 60 different distributions that are used in characterizing population statistics, but only half a dozen are commonly used. Quite often when we refer to parametric statistics, we are making an assumption of Normal (Gaussian) distributions for the data. We will see later that this assumption may not be warranted for accounting and financial transactions.

1. Normal (Gaussian) distribution: The bell-shaped normal distribution is iconic in traditional statistics. Related terms are:
 - Standardization: Subtract the mean and divide by the standard deviation.
 - z-score: The result of standardizing an individual data point.
 - Standard normal: A normal distribution with mean = 0 and standard deviation = 1.
 - QQ-Plot: A quantile (of the sample) by quantile (of a Normal distribution) plot to visualize how close a sample distribution is to a Normal distribution. For the disbursement data, an example of a QQ-plot follows

```
disburse <- read.csv(system.file("extdata", "ch_2_AP_disbursements_journal.csv",
  package = "auditAnalytics", mustWork = TRUE))

d <- as.numeric(as.character(disburse[, "amount_paid"]))

qqnorm(d, main = "Normal Q-Q Plot",
       xlab = "Theoretical Quantiles", ylab = "Sample Quantiles",
       plot.it = TRUE)
```

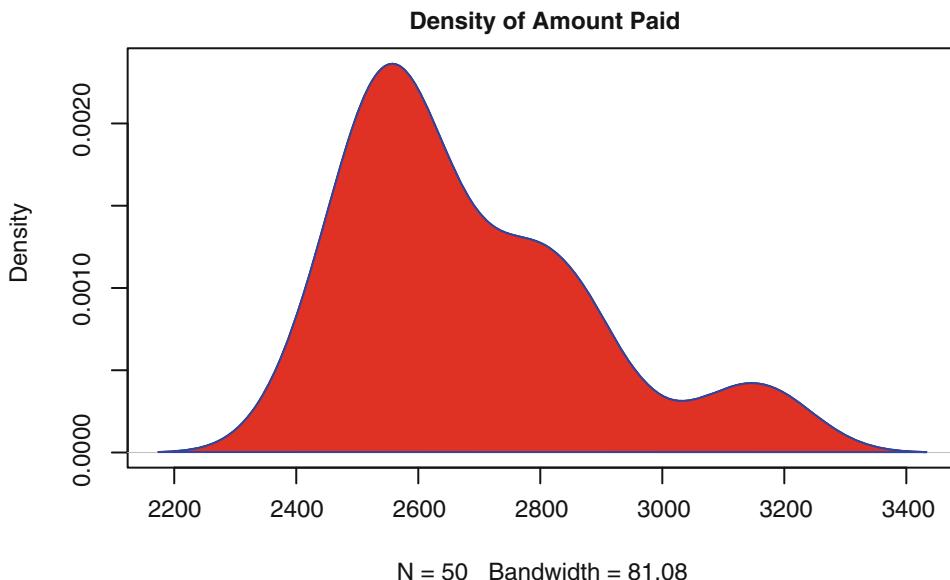


2. Binomial Distribution: A binomial distribution with parameters n and p is the discrete probability distribution of the number of successes in a sequence of n independent experiments, each asking a yes–no question, and each with its own Boolean-valued outcome: success/yes/true/one (with probability p) or failure/no/false/zero (with probability $q = 1 - p$). A single success/failure experiment is also called a Bernoulli trial or Bernoulli experiment and a sequence of outcomes is called a Bernoulli process; for a single trial, i.e., $n = 1$, the binomial distribution is a Bernoulli distribution. The binomial distribution is the basis for the popular binomial test of statistical significance.
3. Bernoulli distribution: A Bernoulli distribution is the discrete probability distribution of a random variable which takes the value 1 with probability p and the value 0 with probability $q = 1 - p$. In other words, the probability distribution of any single experiment that asks a yes–no question; the question results in a Boolean-valued outcome, a single bit whose value is success/yes/true/one with probability p and failure/no/false/zero with probability q .
4. Poisson distribution: the Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time or space if these events occur with a known constant rate and independently of the time since the last event. The waiting time between Poisson events follows an Exponential distribution.
5. Density and Cumulative Distribution: Probabilities have densities (in continuous distributions such as the Normal) or mass functions (in the case of discrete distributions such as the Poisson) that describe the probability of a given outcome. Cumulative distribution functions describe the probability up to a given value. An extension of summary statistics computes estimates of density and cumulative distribution functions for data, as the following example using our disbursements data will show.

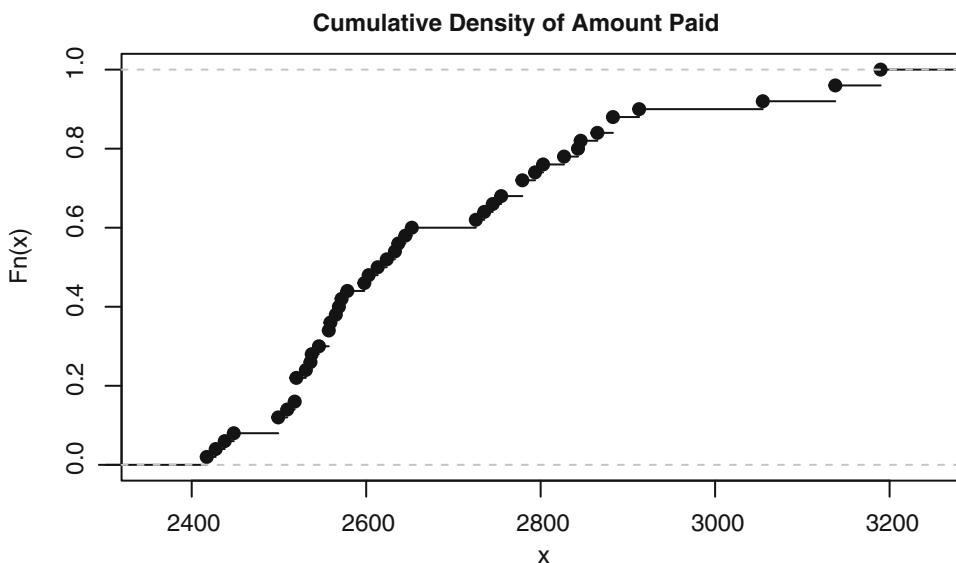
```
devtools::install_github("westland/auditanalytics")
library(auditanalytics)

disburse <- read.csv((system.file("extdata", "ch_2_AP_disbursements_journal.csv",
  package = "auditAnalytics", mustWork = TRUE)))

d <- density(disburse[, "amount_paid"])
plot(d, main = "Density of Amount Paid")
polygon(d, col="red", border="blue")
```



```
~plot(ecdf(disburse[, "amount_paid"]), main = "Cumulative Density of Amount Paid")
```



6. Random Sampling and Sample Bias: A sample is a subset of data from a larger dataset; statisticians call this larger dataset the population—a large, defined but sometimes theoretical or imaginary set of data.
7. Central Limit Theorem (CLT): Proves that sums of independent variables asymptotically approach a Normal distribution as sample size goes to infinity.
8. Standard Error: This is roughly the standard deviation of the sample data. Standard deviation (which measures the variability of individual data points) is distinct from standard error (which measures the variability of a sample metric).
9. Bootstrap and resampling: A bootstrap sample is taken with replacement from an observed dataset. Resampling takes repeated samples from observed data; includes both bootstrap and permutation (shuffling) procedures.
10. Hypothesis Tests (significance tests): Hypotheses are ways of turning complex problems into yes/no questions. They are another way of defining an A/B test. The method grew commonplace after Neyman and Pearson (1933) provided an efficient hypothesis test. Most audit procedures can be couched as hypothesis tests. Here are some key concepts in hypothesis testing:
 - *p*-value: In statistical hypothesis testing, the *p*-value or probability value is, for a given hypothesis test, the probability that, when the null hypothesis is true, the statistical summary (such as the sample mean difference between two groups) would be equal to, or more extreme than, the actual observed results. The use and misuse of *p*-values in statistical hypothesis testing has become a controversial topic.

- Confidence Intervals: Sampling leads to uncertain results, and that uncertainty is reduced through larger, higher quality samples. Confidence intervals are one way of describing the uncertainty of an estimate or prediction based on a sample. Confidence intervals are the only remaining tool from the old fiducial statistical inference.
 - Null and Alternative hypotheses: Ways of dividing the decision space into two parts.
 - One-way and two-way tests: Hypothesis test that considers one tail or two of the distribution.
 - significance (alpha): Critical value for selecting one or the other hypothesis. This is the probability that you will make a type I error.
 - power (1-beta): Critical value for selecting one or the other hypothesis. This is the probability that you will not make a type II error.
 - *t*-test: An inferential statistic used to assess whether there is a significant difference between the means of two groups.
11. Overfitting: Most measurements, and all samples misrepresent, in some way, the population. There is always danger of fitting models to these misrepresentations, thus generating incorrect decisions. This is controlled by better sample size and quality, and also by resampling to ensure robustness of estimation.
12. Linear Regression: Fits a model of the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The following example builds a linear regression model to test the occurrence of insider breaches with audit fees and auditor decisions on section 404 audits, using the Sarbanes–Oxley data.

```
devtools::install_github("westland/auditanalytics")
library(auditanalytics)

lr <-
  read.csv(system.file("extdata", "ch_2_data_types.csv", package = "auditAnalytics",
mustWork = TRUE) %>%
  lm(formula=insd~audit_fee+effective_404)
summary(lr)

## 
## Call:
## lm(formula = insd ~ audit_fee + effective_404, data = .)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -0.1590 -0.1475 -0.1411 -0.1229  0.9292 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 1.486e-01 1.325e-02 11.217   <2e-16 ***
## audit_fee   -9.521e-10 4.449e-10 -2.140    0.0325 *  
## effective_404 1.066e-02 1.816e-02   0.587    0.5574  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.3449 on 1509 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.003242,  Adjusted R-squared:  0.001921 
## F-statistic: 2.454 on 2 and 1509 DF,  p-value: 0.08627
```

Since the dependent variable is dichotomous (binary), results can be improved by using a logit model (from the built-in *glm* function). The following example also showcases R's analysis of the residual errors (differences between the dependent variable values, and the estimated model on the right-hand side). Leverage and distance provide measures of how particular transactions influence the estimation, and are important in identifying outliers.

```

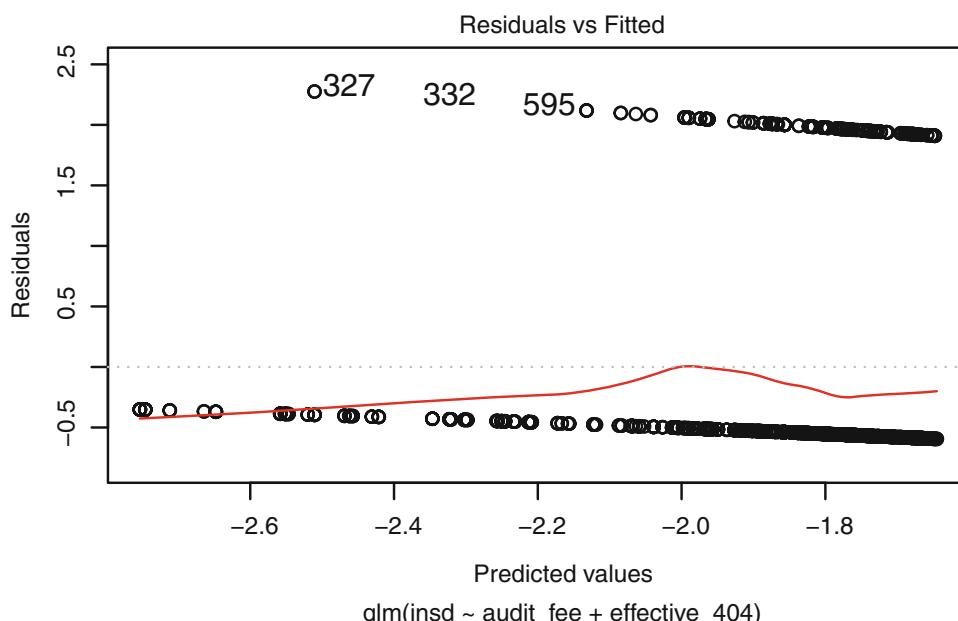
lgt <-
devtools::install_github("westland/auditanalytics")
library(auditanalytics)

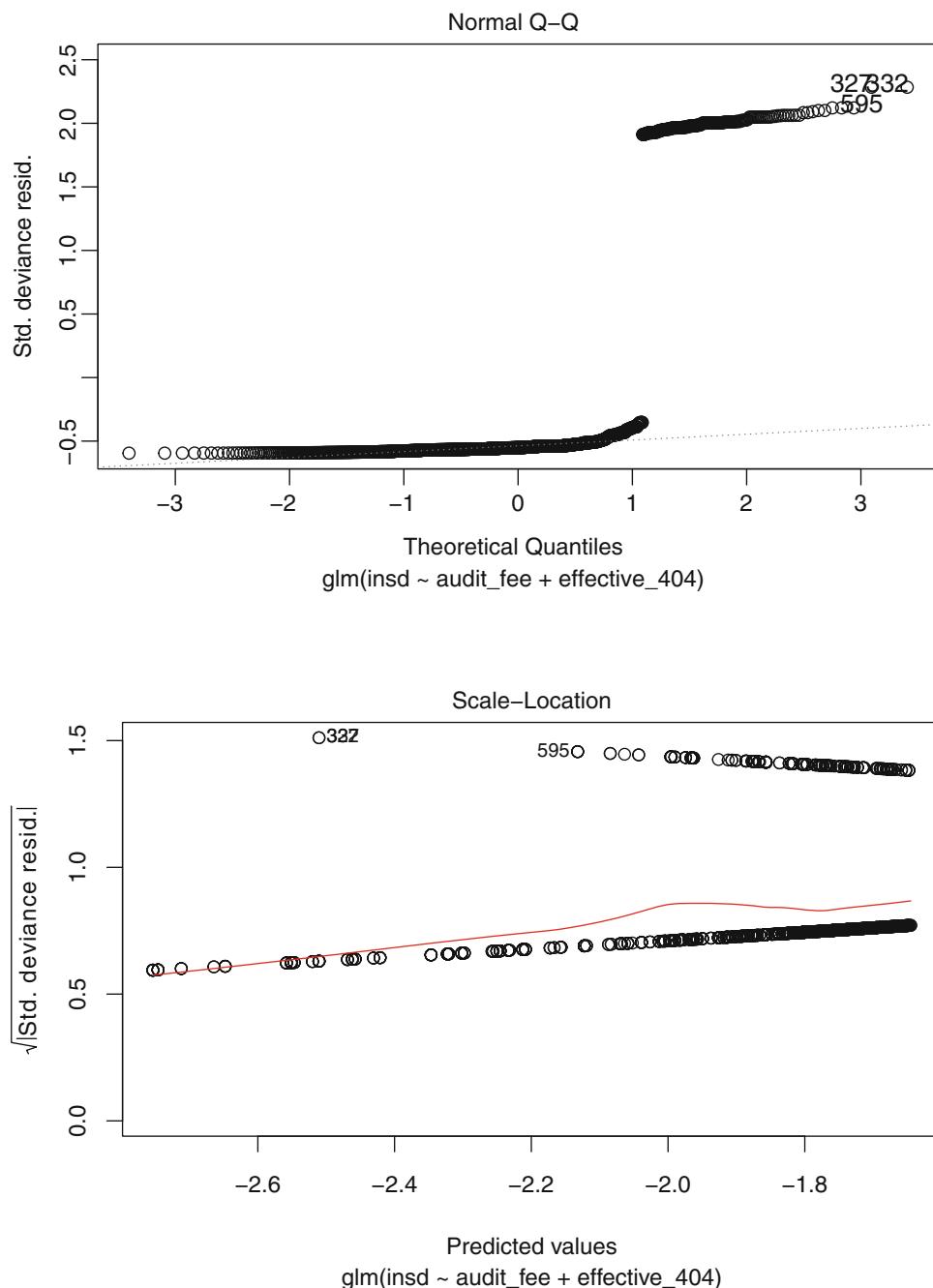
read.csv(system.file("extdata", "ch_2_data_types.csv", package = "auditanalytics",
mustWork = TRUE) %>%
  glm(formula=insd~audit_fee+effective_404, family = "binomial")
summary(lgt)

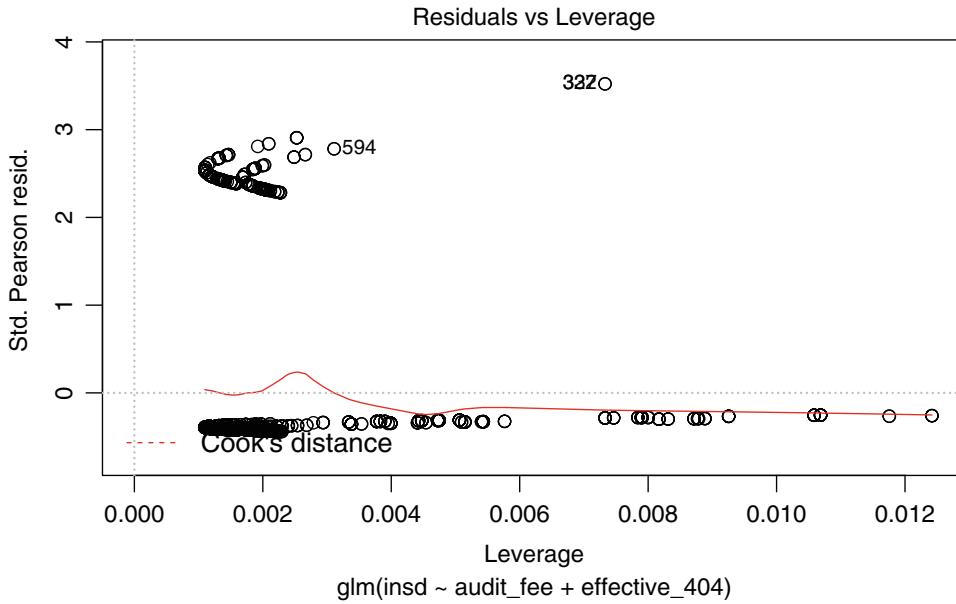
## 
## Call:
## glm(formula = insd ~ audit_fee + effective_404, family = "binomial",
##      data = .)
## 
## Deviance Residuals:
##    Min      1Q   Median      3Q     Max
## -0.5941 -0.5675 -0.5510 -0.5059  2.2754
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.732e+00 1.128e-01 -15.354 <2e-16 ***
## audit_fee    -9.530e-09 4.496e-09  -2.120  0.034 *
## effective_404 8.928e-02 1.518e-01   0.588  0.557
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
## Null deviance: 1214.8 on 1511 degrees of freedom
## Residual deviance: 1209.4 on 1509 degrees of freedom
## (2 observations deleted due to missingness)
## AIC: 1215.4
## 
## Number of Fisher Scoring iterations: 4

```

```
plot(lgt)
```







Machine Learning Methods

Over the past decade, very useful extensions to twentieth century statistical models have been provided by advances in machine learning, and in particular deep-learning. Deep learning is a branch of artificial intelligence that is computationally intensive, but highly flexible for inference from data. Inference is a decision δ (estimation, prediction) based on data $x \in X$ that hopefully contains information about a particular set of constructs. Inference may be about:

1. Classification—e.g., identifying faces, threats.
2. Estimation—e.g., a vector $\theta = \{\theta_1, \dots, \theta_n\}$.
3. Other decisions that may or may not be carried out in real time; e.g., driving a car.

The implicit goal of machine learning is construction of decision strategies that minimize risk. Risk is an informal concept inherited from gambling, and roughly implies the expected loss from using a given decision strategy δ . Frequentist and Bayesian statisticians both base their decision strategies on real-world data, but sharply divided on the actual implementation and interpretation of decision risk.

In practice, risk presents deep learning (and AI in general) with its greatest challenges. For example, self-driving cars are trained on massive datasets extracted from many other cars; that knowledge of how to drive is encapsulated in weights in a network model that is firm-wired into a computer unit in a car. It matters whether that unit was trained to save pedestrians, or to save property, or to save the driver—each implies a different decision risk.

Traditionally, statistics have used a squared-error loss function $l(\theta, \delta(X)) = E_\delta[(\delta(X) - \theta)^2]$ where $\delta(X) \equiv \hat{\theta}$ in the case of estimation. This is easy to compute using pencil and paper, particularly when optimization relied on first-order conditions from calculus. But most real-world decisions are not optimally made with squared-error loss functions. Although squared-error loss is still widely used, machine learning practitioners will also use more complex, sometimes asymmetric loss functions that more closely fit the real-world problems. In particular for classification problems such as image recognition, squared-error loss makes little sense, and practitioners will tend to use cross-entropy, drawn from information theory, where the Kraft–McMillan theorem establishes the rationale for its application. Business situations prefer asymmetric loss that penalizes costs and rewards revenues.

Whereas traditional statistics relies heavily on first-order conditions from calculus, deep learning uses compute-intensive search algorithms that explore the response surface of the risk function. The following example presents the “Hello World” of machine learning: recognizing handwritten numbers on the National Institute of Standards dataset.

```
# The following example assumes that you have already installed Tensorflow have
# been installed (see https://www.tensorflow.org/install/)

library(keras)
install_keras()

## Creating virtualenv for TensorFlow at ~/.virtualenvs/r-tensorflow
## Installing TensorFlow ...
##
## Installation complete.

mnist <- dataset_mnist()
train_images <- mnist$train$x
train_labels <- mnist$train$y
test_images <- mnist$test$x
test_labels <- mnist$test$y

network <- keras_model_sequential() %>%
  layer_dense(units = 512, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")

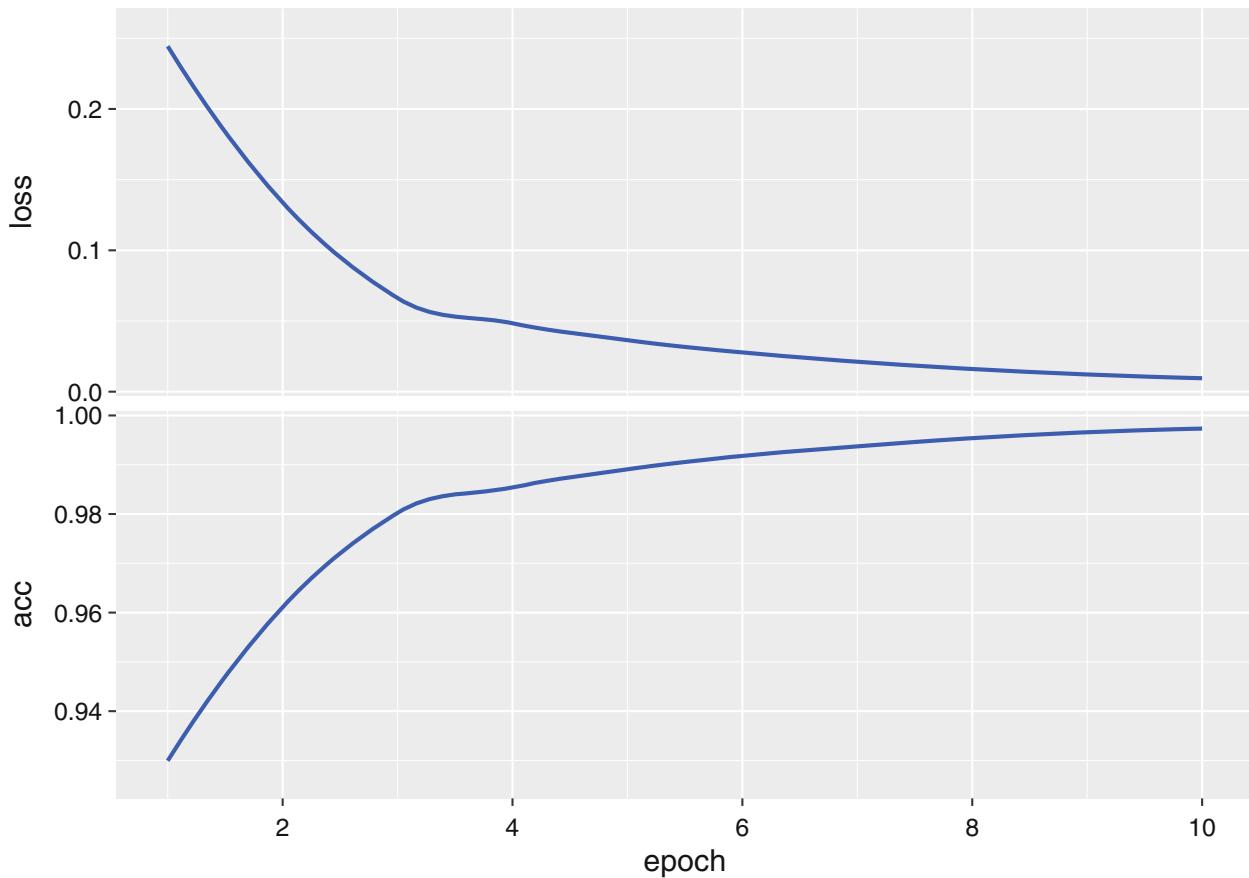
network %>% compile(
  optimizer = "rmsprop",
  loss = "categorical_crossentropy",
  metrics = c("accuracy")
)

train_images <- array_reshape(train_images, c(60000, 28 * 28))
train_images <- train_images / 255

test_images <- array_reshape(test_images, c(10000, 28 * 28))
test_images <- test_images / 255

train_labels <- to_categorical(train_labels)
test_labels <- to_categorical(test_labels)

history <- network %>% fit(train_images, train_labels, epochs = 10, batch_size = 128)
plot(history)
```



```
metrics <- network %>% evaluate(test_images, test_labels, verbose = 0)
metrics
```

```
## $loss
## [1] 0.07689612
##
## $acc
## [1] 0.9803
```

This particular Keras model overfits the MNIST data, with a final accuracy used of 98% and loss of 7%.

Machine learning is a vast and rapidly evolving field. It is increasingly used for the analysis of social network and other text based intelligence required for the analytical review portion (and other parts) of the audit. In the future, expect its role in auditing to expand, as suitable models are developed in the field of auditing and accounting.

Statistical Perspectives on Audit Evidence and its Information Content

Support and the Additivity of Evidence: The Log-Likelihood

The *log-likelihood* has an intuitive interpretation, as suggested by the term “support.” Given independent events, the overall log-likelihood is the sum of the log-likelihoods of the individual events, just as the overall log-probability is the sum of the log-probability of the individual events. Viewing data as evidence, this is interpreted as “support from independent evidence adds,” and the log-likelihood is the “weight of evidence.” Interpreting negative log-probability as information content or surprisal, the support (log-likelihood) of a model, given an event, is the negative of the surprisal of the event, given the model: a model is supported by an event to the extent that the event is unsurprising, given the model.

Just as the likelihood, given no event, being 1, the log-likelihood, given no event, is 0, which corresponds to the value of the empty sum: without any data, there is no support for any models.

The log-likelihood is particularly convenient for maximum likelihood estimation. Because logarithms are strictly increasing functions, maximizing the likelihood is equivalent to maximizing the log-likelihood.

Since concavity plays a key role in the maximization, and since most common probability distributions—in particular the exponential family—are only logarithmically concave, it is usually more convenient to work with a logarithmic transformation of the likelihood function, known as the log-likelihood function.

The “Score”

In statistics, the score (or informant) is the gradient of the log-likelihood function with respect to the parameter vector. Evaluated at a particular point, the score indicates the steepness of the log-likelihood function and thereby the sensitivity to infinitesimal changes to the parameter values. If the log-likelihood function is continuous over the parameter space, the score will vanish at a local maximum or minimum; this fact is used in maximum likelihood estimation to find the parameter values that maximize the likelihood function.

Since the score is a function of the observations that are subject to sampling error, it lends itself to a test statistic known as score test in which the parameter is held at a particular value. Further, the ratio of two likelihood functions evaluated at two distinct parameter values can be understood as a definite integral of the score function.

Fisher Information

In mathematical statistics, the Fisher information (sometimes simply called information) is a way of measuring the amount of information that an observable random variable X carries about an unknown parameter θ of a distribution that models X . Formally, it is the variance of the score, or the expected value of the observed information. In Bayesian statistics, the asymptotic distribution of the posterior mode depends on the Fisher information and not on the prior (according to the Bernstein–von Mises theorem, which was anticipated by Laplace for exponential families). The role of the Fisher information in the asymptotic theory of maximum likelihood estimation was emphasized by the statistician Ronald Fisher (following some initial results by Francis Ysidro Edgeworth). The Fisher information is also used in the calculation of the Jeffreys prior, which is used in Bayesian statistics.

The Fisher-information matrix is used to calculate the covariance matrices associated with maximum likelihood estimates. It can also be used in the formulation of test statistics, such as the Wald test.

Statistical systems of a scientific nature (physical, biological, etc.) whose likelihood functions obey shift invariance have been shown to obey maximum Fisher information. The level of the maximum depends upon the nature of the system constraints.

The Fisher information is a way of measuring the amount of information that an observable random variable X carries about an unknown parameter θ upon which the probability of X depends. Let $f(X; \theta)$ be the probability density function (or probability mass function) for X conditional on the value of θ . It describes the probability that we observe a given outcome of X , given a known value of θ . If f is sharply peaked with respect to changes in θ , it is easy to indicate the “correct” value of θ from the data, or equivalently, that the data X provides a lot of information about the parameter θ . If the likelihood f is flat and spread out, then it would take many samples like of X to estimate the actual “true” value of θ that would be obtained using the entire population being sampled.

Reference

- Neyman, Jerzy, and Egon Sharpe Pearson. 1933. IX. On the Problem of the Most Efficient Tests of Statistical Hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 231 (694–706): 289–337.
Loebbecke, James K., and Paul J. Steinbart. (1987). An Investigation of the Use of Preliminary Analytical Review to Provide Substantive Audit Evidence. *Auditing-a Journal of Practice & Theory* 6 (2): 74–89.

Risk Assessment and Planning



Auditing

Auditing of financial accounts is the process of verifying that economic events in the real world are accurately summarized in the financial statements of a legal entity. An audit opinion, the product of such auditing provides:

1. Reasonable assurance.
2. By an independent third party.
3. That the financial statements are presented fairly in all material respects.
4. In accordance to some financial reporting framework, e.g., generally accepted accounting principles.
5. Applied consistently so that year-to-year trends and comparisons are possible.

In the USA, the auditing framework is dictated by generally accepted accounting principles (GAAP) and generally accepted accounting standards (GAAS). Internationally the International Standards on Auditing (ISA) issued by the International Auditing and Assurance Standards Board (IAASB) is considered as the benchmark for audit process. Under US generally accepted accounting principles (GAAP) auditors must release one of three types of opinions of the overall financial statements:

1. An unqualified auditor's opinion is the opinion that the financial statements are presented fairly.
2. A qualified opinion is that the financial statements are presented fairly in all material respects in accordance with GAAP:
 - (a) Except for a material misstatement that does not pervasively affect the users ability to rely on the financial statements.
 - (b) Or with a scope limitation that is of limited significance.
3. An adverse audit opinion is issued when the financial statements do not present fairly due to departure from US GAAP with an explanation of the nature and size of the misstatement.

Additionally an auditor can issue a disclaimer which is considered a type of qualified opinion because there is insufficient and appropriate evidence to form an opinion or because of lack of independence. In a disclaimer the auditor explains the reasons for withholding an opinion and explicitly indicates that no opinion is expressed. The wording of these reports typically appears as follows:

1. Unqualified ("Clean") Opinion: We believe these financial statements are (1) fairly presented in (2) accordance with GAAP (3) consistently applied.
2. Qualified Opinion: We believe these financial statements are (1) fairly presented in (2) accordance with GAAP (3) consistently applied; except for (4) A List of Exceptions.
3. Adverse Opinion: We DO NOT believe these financial statements are (1) fairly presented in (2) accordance with GAAP (3) consistently applied; THE AUDITORS DISAGREE WITH A List of Exceptions.

Risk Assessment in Planning the Audit

Risk assessment in auditing is the determination of quantitative value of risk of a particular subset of accounting operations related to the rendering of the audit opinion—i.e., on whether the accounts have material error when GAAP has been

consistently applied. Quantitative risk assessment requires calculations of two components of risk—the magnitude of the potential loss, and the probability that the loss will occur.

The economic events of the accounting cycles are called accounting transactions, and are traditionally recorded in journals. An accounting cycle begins when accounting personnel create a transaction from a source document and ends with the completion of the financial reports and closing of temporary accounts in preparation for a new cycle. The five accounting cycles are:

1. Revenue cycle.
2. Expenditure cycle (this cycle focuses on two separate resources: inventory and human resources and often considers two separate cycles: purchasing and payroll/HR).
3. Conversion cycle (Production cycle).
4. Financing (Capital Acquisition and repayment).
5. Fixed assets.

Problems in any transaction generated in these cycles are the sources of “loss” considered in the Risk Assessment Matrix (RAM). Risk assessment consists of subjective and objective evaluations of risk in which assumptions and uncertainties are clearly considered and presented. Part of the difficulty in risk management is that measurement of both potential loss and probability of occurrence is error prone and subjective.

Risk with a large potential loss and a low probability of occurring is often treated differently from one with a low potential loss and a high likelihood of occurring. In theory, both are of nearly equal priority, but in practice it can be very difficult to manage when faced with the scarcity of resources, especially time, in which to conduct the risk management process. This is one conundrum engendered by auditing’s inherently “wicked” character.

A Risk Assessment Matrix (RAM) is a calculation spreadsheet that is used in risk assessment to define, estimate, argue, and support particular risks involved in decision-making. In auditing, the objective of the RAM is to construct a set of audit tasks (the audit program) that cost-effectively assures that the eventual audit decision will be correct with a particular level of confidence.

Different statistical philosophies dictate the particular mathematical approach which is applied in the RAM. We will briefly review the alternatives here, with the goal of describing a simple, somewhat objective approach appropriate for the initial stages of auditing planning.

Mathematicians recognize three broad approaches to risk calculations for decision-making: (1) the Neyman–Pearson hypothesis testing framework; (2) the Minimax game theoretic framework; and (3) Bayes Risk. This chapter proposes a simplified Bayes Risk as most appropriate for early planning of audits.

The Neyman–Pearson lemma allows performing a hypothesis test between two simple hypotheses using a likelihood-ratio test, and provides a method of statistical inference from evidence. In statistics, a result is called statistically significant if it has been predicted as unlikely to have occurred by chance alone, according to a pre-determined threshold probability, the significance level. The phrase “test of significance” was coined by statistician Ronald Fisher, and plays an important part in our substantive tests of account balances and material error. Statistical hypothesis testing is sometimes called confirmatory data analysis, in contrast to exploratory data analysis, which may not have pre-specified hypotheses. Statistical hypothesis tests define a procedure that controls (fixes) the probability of incorrectly deciding that a default position (null hypothesis) is incorrect based on how likely it would be for a set of observations to occur if the null hypothesis were true. Note that this probability of making an incorrect decision is not the probability that the null hypothesis is true, nor whether any specific alternative hypothesis is true. This contrasts with other possible techniques of decision theory in which the null and alternative hypothesis are treated on a more equal basis.

Minimax is a decision rule published in 1928 by John von Neumann and is used in decision theory, game theory, statistics, and philosophy for minimizing the possible loss for a worst case (maximum loss) scenario. Originally formulated for two-players, zero-sum game theory, covering both the cases where players take alternate moves and those where they make simultaneous moves, it has also been extended to more complex games and to general decision-making in the presence of uncertainty.

Bayes risk is the expected value of a loss function. It is typically optimized when a Bayes estimator decision rule is designed to minimize the posterior expected loss. The most common risk function used for Bayesian estimation is the mean square error (MSE), or squared-error risk $E_x[(\hat{\theta}(x) - \theta)^2]$ for parameter θ .

The probability of something happening multiplied by the resulting cost or benefit if it does is commonly known as the “expectation value” and is used to compare levels of risk and commensurate audit task scope and sample size in the subsequent audit program design.

For audits performed by an outside audit firm, risk assessment is a very crucial stage before accepting an audit engagement. It is an integral part of determining the audit tasks that will be performed in the audit program. According to ISA315 “the

auditor should perform risk assessment procedures to obtain an understanding of the entity and its environment, including its internal control.” The auditor obtains initial evidence regarding the classes of transactions at the client and the operating effectiveness of the client’s internal controls. In auditing standards, audit risk is stated to include inherent risk (IR) control risk (CR) and detection risk (DR) (Srivastava and Shafer 2008; Jones 2017; Bedard, Graham, and Jackson 2005; Knechel 2007).

The audit risk model expresses the risk of an auditor providing an inappropriate opinion of a commercial entity’s financial statements and is calculated:

$$AR = IR \times CR \times DR$$

In this formula, IR refers to the risk involved in the nature of business or transaction. Example, transactions involving exchange of cash may have higher IR than transactions involving settlement by checks. CR refers to the risk that a misstatement could occur but may not be detected and corrected or prevented by entity’s internal control mechanism. DR is the probability that the audit procedures may fail to detect existence of a material error or fraud. While CR depends on the strength or weakness of the internal control procedures, DR is either due to sampling error or human factors.

This formula is an extreme simplification of the occurrence of loss in the real world, and is only suitable for the early, exploratory stages of planning an audit. As evidence is collected at mid-year and year-end, audit tasks will need to be updated, and audit scope changed to meet the interim findings of the audit. There are several problems with the simplified formula:

1. Poor Resolution. Typical risk matrices can correctly and unambiguously compare only a small fraction (typically less than 10%) of randomly selected pairs of hazards. They can assign identical ratings to quantitatively very different risks (“range compression”).
2. Errors. Risk matrices can mistakenly assign higher qualitative ratings to quantitatively smaller risks. For risks with negatively correlated frequencies and severities, they can be “worse than useless,” leading to worse-than-random decisions.
3. Suboptimal Resource Allocation. Effective allocation of resources to risk-reducing countermeasures cannot be based on the categories provided by risk matrices.
4. Ambiguous Inputs and Outputs. Categorizations of severity cannot be made objectively for uncertain consequences. Inputs to risk matrices (e.g., frequency and severity categorizations) and resulting outputs (i.e., risk ratings) require subjective interpretation, and different users may obtain opposite ratings of the same quantitative risks.

Accessing the SEC's EDGAR Database of Financial Information

One of the first steps in planning an analytical review is a review of current and prior year filings with the SEC. These will include annual and quarterly financial statements, restatements, proxy statements, lawsuits, and numerous other documents, where their acquisition and incorporation into workpapers is an essential prerequisite of audit planning. Fortunately, complete information is available on the SEC’s website at sec.gov. Many of the most relevant documents to an audit are maintained by the SEC in XBRL format (as .XML files) which can be downloaded into the working papers from the Internet. XBRL is *eXtensible Business Reporting Language*, a freely available, global markup language for exchanging business information. XBRL allows the expression of semantic meaning, which lends to an unambiguous definition of accounts and other financial information. XBRL representations of financial reports are more reliable and less subject to misinterpretation than any disseminations in other formats. XBRL also allows for automated parsing of information, which can greatly improve the efficiency of audit ratio and statistical analysis.

The following code chunk accesses the SEC’s XBRL databases to acquire current and prior year filings for any listed company, and read it as a dataset that can be manipulated by R. For this example, we extract General Motors’ 2016 and 2017 financials from the EDGAR database at sec.gov. I use the finstr package to access EDGAR files.

```
library(devtools)
install_github("bergant/finstr")
library(finstr)
library(XBRL)

# Locate the XBRL format 10-K reports for years 2016 and 2017
# Search sec.gov for General Motors, choose the 10-K for the year
```

```

# Note that .xml is the XBRL file indicator)

xbrl_url2016 <-
"https://www.sec.gov/Archives/edgar/data/1467858/000146785817000028/gm-20161231.xml"

xbrl_url2017 <-
"https://www.sec.gov/Archives/edgar/data/1467858/000146785818000022/gm-20171231.xml"

# Get EDGAR data in XBRL format from the sec.gov site
# parse XBRL (GM 10-K reports)
old_o <- options(stringsAsFactors = FALSE)
xbrl_data_2016 <- xbRLDoAll(xbRL_url2016)
xbrl_data_2017 <- xbRLDoAll(xbRL_url2017)
options(old_o)

## With xbRL_get_statements convert sec.gov's XBRL data to a list of lists

st2017 <- xbRL_get_statements(xbRL_data_2017)
st2016 <- xbRL_get_statements(xbRL_data_2016)

st2017

## Financial statements repository
##                                     From      To Rows Columns
## ConsolidatedBalanceSheets        2016-12-31 2017-12-31  2    44
## ConsolidatedIncomeStatements     2015-12-31 2017-12-31  3    29
## ConsolidatedStatementsOfCashFlows 2015-12-31 2017-12-31  3    42
## ConsolidatedStatementsOfComprehensiveIncome 2015-12-31 2017-12-31  3    11

```

The 10-K XBRL file is a list of four lists—balance sheet, income statement, cash flow, and comprehensive income. Content and names used will vary widely from company to company, and from particular filing or statement. Thus it is important to inspect the files that are retrieved to determine the correct variable names, data formats, and structure of each XBRL file prior to any subsequent analysis. Once the financial statements are loaded into the R session, the `finstr` package contains several commands to check consistency of the reports and to display data in a format suitable for auditing. The following code chunks provide examples of some of the most useful commands in the `finstr` package (Table 1).

```

library(tidyverse)
library(kableExtra)

## To get a single statement, assign one of the four lists, e.g.

balance_sheet2017 <- st2017$ConsolidatedBalanceSheets
balance_sheet2016 <- st2016$ConsolidatedBalanceSheets
income2017 <- st2017$ConsolidatedIncomeStatements
income2016 <- st2016$ConsolidatedIncomeStatements

## Print the balance sheet;
## capture the output to a NULL file, and
## reformat with the kableExtra package

capture.output(
  bs_table <-
  print(
    balance_sheet2017,
    html = FALSE,
    big.mark = ",",
    dateFormat = "%Y"),

```

```

file='NUL')

bs_table %>%
  kable(longtable=T,
        caption="Balance Sheet",
        "latex",
        booktabs = T) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width =
F, font_size=10)

```

Table 1 Balance sheet

Element	2017-12-31	2016-12-31
Assets =	212,482	221,690
+ AssetsCurrent =	68,744	76,203
+ CashAndCashEquivalentsAtCarryingValue	15,512	12,574
+ MarketableSecuritiesCurrent	8313	11,841
+ AccountsNotesAndLoansReceivableNetCurrent	8164	8700
+ InventoryNet	10,663	11,040
+ gm_AssetsSubjecttoorAvailableforOperatingLeaseNetCurrent	1106	1110
+ OtherAssetsCurrent	4465	3633
+ AssetsOfDisposalGroupIncludingDiscontinuedOperationCurrent	0	11,178
+ NotesAndLoansReceivableNetCurrent	0	0
+ AssetsNoncurrent =	143,738	145,487
+ EquityMethodInvestments	9073	8996
+ PropertyPlantAndEquipmentNet	36,253	32,603
+ IntangibleAssetsNetIncludingGoodwill	5849	6149
+ DeferredIncomeTaxAssetsNet	23,544	33,172
+ OtherAssetsNoncurrent	4929	3849
+ DisposalGroupIncludingDiscontinuedOperat...	0	9375
+ NotesAndLoansReceivableNetNoncurrent	0	0
+ PropertySubjectToOrAvailableForOperatingLeaseNet	42,882	34,342
LiabilitiesAndStockholdersEquity =	212,482	221,690
+ Liabilities =	176,282	177,615
+ LiabilitiesCurrent =	76,890	85,181
+ AccountsPayableCurrent	23,929	23,333
+ AccruedLiabilitiesCurrent	25,996	25,893
+ LiabilitiesOfDisposalGroupIncludingDisco...	0	12,158
+ DebtCurrent	0	0
+ LiabilitiesNoncurrent =	99,392	92,434
+ OtherPostretirementDefinedBenefitPlanLiabilitiesNoncurrent	5998	5803
+ DefinedBenefitPensionPlanLiabilitiesNoncurrent	13,746	15,264
+ OtherLiabilitiesNoncurrent	12,394	12,415
+ LiabilitiesOfDisposalGroupIncludingDisco...	0	7626
+ LongTermDebtAndCapitalLeaseObligations	0	0
+ CommitmentsAndContingencies	0	0
+ StockholdersEquityIncludingPortionAttrib... =	36,200	44,075
+ StockholdersEquity =	35,001	43,836
+ CommonStockValue	14	15
+ AdditionalPaidInCapital	25,371	26,983
+ RetainedEarningsAccumulatedDeficit	17,627	26,168
+ AccumulatedOtherComprehensiveIncomeLossNetOfTax	-8011	-9330
+ MinorityInterest	1199	239

Planning review looks for changes from prior years, or trends that may be important in the current year's audit. The `merge()` command consolidates the information from different .XML files into single files (Table 2).

```
library(tidyverse)
library(kableExtra)

## Use merge function to create single financial statement data from two statements.

balance_sheet <- merge(balance_sheet2017, balance_sheet2016)

## Print the balance sheet;
## capture the output to a NULL file, and
## reformat with the kableExtra package

capture.output(bs_table <-
  print(balance_sheet2017,
        html = FALSE,
        big.mark = ",",
        dateFormat = "%Y"),
  file='NULL')

bs_table %>%
  kable(longtable=T,
        caption="Merged Balance Sheet",
        "latex",
        booktabs = T) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width =
F, font_size=10)
```

Table 2 Merged balance sheet

Element	2017-12-31	2016-12-31
Assets =	212,482	221,690
+ AssetsCurrent =	68,744	76,203
+ CashAndCashEquivalentsAtCarryingValue	15,512	12,574
+ MarketableSecuritiesCurrent	8313	11,841
+ AccountsNotesAndLoansReceivableNetCurrent	8164	8700
+ InventoryNet	10,663	11,040
+ gm_AssetsSubjecttoorAvailableforOperatingLeaseNetCurrent	1106	1110
+ OtherAssetsCurrent	4465	3633
+ AssetsOfDisposalGroupIncludingDiscontinuedOperationCurrent	0	11,178
+ NotesAndLoansReceivableNetCurrent	0	0
+ AssetsNoncurrent =	143,738	145,487
+ EquityMethodInvestments	9073	8996
+ PropertyPlantAndEquipmentNet	36,253	32,603
+ IntangibleAssetsNetIncludingGoodwill	5849	6149
+ DeferredIncomeTaxAssetsNet	23,544	33,172
+ OtherAssetsNoncurrent	4929	3849
+ DisposalGroupIncludingDiscontinuedOperat...	0	9375
+ NotesAndLoansReceivableNetNoncurrent	0	0
+ PropertySubjectToOrAvailableForOperatingLeaseNet	42,882	34,342
LiabilitiesAndStockholdersEquity =	212,482	221,690

(continued)

Table 2 (continued)

Element	2017-12-31	2016-12-31
+ Liabilities =	176,282	177,615
+ LiabilitiesCurrent =	76,890	85,181
+ AccountsPayableCurrent	23,929	23,333
+ AccruedLiabilitiesCurrent	25,996	25,893
+ LiabilitiesOfDisposalGroupIncludingDisco...	0	12,158
+ DebtCurrent	0	0
+ LiabilitiesNoncurrent =	99,392	92,434
+ OtherPostretirementDefinedBenefitPlanLiabilitiesNoncurrent	5998	5803
+ DefinedBenefitPensionPlanLiabilitiesNoncurrent	13,746	15,264
+ OtherLiabilitiesNoncurrent	12,394	12,415
+ LiabilitiesOfDisposalGroupIncludingDisco...	0	7626
+ LongTermDebtAndCapitalLeaseObligations	0	0
+ CommitmentsAndContingencies	0	0
+ StockholdersEquityIncludingPortionAttrib... =	36,200	44,075
+ StockholdersEquity =	35,001	43,836
+ CommonStockValue	14	15
+ AdditionalPaidInCapital	25,371	26,983
+ RetainedEarningsAccumulatedDeficit	17,627	26,168
+ AccumulatedOtherComprehensiveIncomeLossNetOfTax	-8011	-9330
+ MinorityInterest	1199	239

The `check_statement()` command in `fistr` will automatically validate internal consistency of transaction lines and summary lines in the EDGAR filings.

```
library(tidyverse)

## Recalculate higher order concepts from basic values and check for errors.

check <- check_statement(balance_sheet2017)
check

## Number of errors: 8
## Number of elements in errors: 4
##
## Element: AssetsCurrent = + CashAndCashEquivalentsAtCarryingValue +
##           MarketableSecuritiesCurrent + AccountsNotesAndLoansReceivableNetCurrent +
##           InventoryNet + gm_AssetsSubjecttoOrAvailableforOperatingLeaseNetCurrent +
##           OtherAssetsCurrent + AssetsOfDisposalGroupIncludingDiscontinuedOperation
##           Current + NotesAndLoansReceivableNetCurrent
##           date original calculated error
## 3 2016-12-31 7.6203e+10 6.0076e+10 1.6127e+10
## 4 2017-12-31 6.8744e+10 4.8223e+10 2.0521e+10
##
## Element: AssetsNoncurrent = + EquityMethodInvestments + PropertyPlantAnd
##           EquipmentNet + IntangibleAssetsNetIncludingGoodwill + DeferredIncomeTax
##           AssetsNet + OtherAssetsNoncurrent + DisposalGroupIncludingDiscontinued
##           OperationAssetsNoncurrent + NotesAndLoansReceivableNetNoncurrent + Property
##           SubjectToOrAvailableForOperatingLeaseNet
##           date original calculated error
## 5 2016-12-31 1.45487e+11 1.28486e+11 1.7001e+10
## 6 2017-12-31 1.43738e+11 1.22530e+11 2.1208e+10
```

```

## 
## Element: LiabilitiesCurrent = + AccountsPayableCurrent + AccruedLiabilities
## Current + LiabilitiesOfDisposalGroupIncludingDiscontinuedOperationCurrent +
## DebtCurrent
##           date   original calculated      error
## 11 2016-12-31 8.5181e+10 6.1384e+10 2.3797e+10
## 12 2017-12-31 7.6890e+10 4.9925e+10 2.6965e+10
##
## Element: LiabilitiesNoncurrent = + OtherPostretirementDefinedBenefit
## PlanLiabilitiesNoncurrent + DefinedBenefitPensionPlanLiabilitiesNoncurrent +
## OtherLiabilitiesNoncurrent + LiabilitiesOfDisposalGroupIncludingDiscontinued
## OperationNoncurrent + LongTermDebtAndCapitalLeaseObligations
##           date   original calculated      error
## 13 2016-12-31 9.2434e+10 4.1108e+10 5.1326e+10
## 14 2017-12-31 9.9392e+10 3.2138e+10 6.7254e+10

## In case of error the numbers with errors will be presented along with elements:

check_statement(
  within(balance_sheet2017, InventoryNet <- InventoryNet * 2)
)

## Number of errors: 8
## Number of elements in errors: 4
##
## Element: AssetsCurrent = + CashAndCashEquivalentsAtCarryingValue +
## MarketableSecuritiesCurrent + AccountsNotesAndLoansReceivableNetCurrent +
## InventoryNet + gm_AssetsSubjecttoorAvailableforOperatingLeaseNetCurrent +
## OtherAssetsCurrent + AssetsOfDisposalGroupIncludingDiscontinuedOperation
## Current + NotesAndLoan ReceivableNetCurrent
##           date   original calculated      error
## 3 2016-12-31 7.6203e+10 7.1116e+10 5.087e+09
## 4 2017-12-31 6.8744e+10 5.8886e+10 9.858e+09
##
## Element: AssetsNoncurrent = + EquityMethodInvestments + PropertyPlantAnd
## EquipmentNet + IntangibleAssetsNetIncludingGoodwill + DeferredIncomeTax
## AssetsNet + OtherAssetsNoncurrent + DisposalGroupIncludingDiscontinued
## OperationAssetsNoncurrent + NotesAndLoansReceivableNetNoncurrent +
## PropertySubjectToOrAvailableForOperatingLeaseNet
##           date   original calculated      error
## 5 2016-12-31 1.45487e+11 1.28486e+11 1.7001e+10
## 6 2017-12-31 1.43738e+11 1.22530e+11 2.1208e+10
##
## Element: LiabilitiesCurrent = + AccountsPayableCurrent + AccruedLiabilities
## Current + LiabilitiesOfDisposalGroupIncludingDiscontinuedOperationCurrent +
## DebtCurrent
##           date   original calculated      error
## 11 2016-12-31 8.5181e+10 6.1384e+10 2.3797e+10
## 12 2017-12-31 7.6890e+10 4.9925e+10 2.6965e+10
##
## Element: LiabilitiesNoncurrent = + OtherPostretirementDefinedBenefitPlan
## LiabilitiesNoncurrent + DefinedBenefitPensionPlanLiabilitiesNoncurrent + Other
## LiabilitiesNoncurrent + LiabilitiesOfDisposalGroupIncludingDiscontinuedOperation
## Noncurrent + LongTermDebtAndCapitalLeaseObligations
##           date   original calculated      error

```

```

## 13 2016-12-31 9.2434e+10 4.1108e+10 5.1326e+10
## 14 2017-12-31 9.9392e+10 3.2138e+10 6.7254e+10

## validation returns all calculation results in a readable data frame.
## e.g., operating income from income statement:

check <- check_statement(income2017, element_id = "OperatingIncomeLoss")
check

## Number of errors: 0
## Number of elements in errors: 0

check$expression[1]

## [1] "+ Revenues - CostsAndExpenses"

check$calculated / 10^6

## [1] 5538 9962 10016

```

Rearranging statements is often a useful step before actual calculations. Rearrangements can offer several advantages in ad hoc analyses such as analytical review:

- We can avoid errors in formulas with many variables,
- Accounting taxonomies do change and using many formulas on original statement is harder to support than using custom hierarchy for analysis starting point,
- When sharing analyses it is easier to print fewer values.

To rearrange the statement to simple two-level hierarchy use the `expose` function.

```

expose( balance_sheet,

  # Assets
  `Current Assets` = "AssetsCurrent",
  `Noncurrent Assets` = other("Assets"),

  # Liabilities and equity
  `Current Liabilities` = "LiabilitiesCurrent",
  `Noncurrent Liabilities` = other(c("Liabilities", "CommitmentsAndContingencies")),
  `Stockholders Equity` = "StockholdersEquity"
)

## Financial statement: 3 observations from 2015-12-31 to 2017-12-31
## Element          2017-12-31 2016-12-31 2015-12-31
## Assets =         212482    221690    194338
##   + Current.Assets      48223     54138     51357
##   + Noncurrent.Assets    122530     90237     86258
##   LiabilitiesAndStockholdersEquity = 212482    221690    194338
##   + Current.Liabilities    49925     56153     51655
##   + Noncurrent.Liabilities  32138     36834     39249
##   + Stockholders.Equity      35001     43836     39871
##   + OtherLiabilitiesAndStockholdersEquity_  1199      239      452

```

Here, the balance sheet stays divided by assets, liabilities, and equity. For the second level we are exposing current assets from noncurrent and similarly for the liabilities. We choose to separate equity.

Function `expose` expects a list of vectors with element names. Function `other` helps us identify elements without enumerating every single element. Using `other` reduces potential errors, as the function knows which elements are not specified and keeps the balance sheet complete.

Sometimes it is easier to define a complement than a list of elements. In this case we can use the `%without%` operator. Let us expose, for example, tangible and then intangible assets (Table 3):

```
library(tidyverse)
library(kableExtra)

expose( balance_sheet,

  # Assets
  `Tangible Assets` =
    "Assets" %without% c("AssetsOfDisposalGroupIncludingDiscontinuedOperationCurrent",
  , "NotesAndLoansReceivableNetCurrent", "gm_AssetsSubjecttoorAvailableforOperating
  LeaseNetCurrent"),
  `Intangible Assets` = other("Assets"),

  # Liabilities and equity
  `Liabilities` = c("Liabilities", "CommitmentsAndContingencies"),
  `Stockholders Equity` = "StockholdersEquity"
)

## Financial statement: 3 observations from 2015-12-31 to 2017-12-31
## Element          2017-12-31 2016-12-31 2015-12-31
## Assets           212482     221690     194338
##   + Tangible.Assets      169647     142479     134832
##   + Intangible.Assets     1106       1896       2783
## LiabilitiesAndStockholdersEquity = 212482     221690     194338
##   + Liabilities          82063      92987      90904
##   + Stockholders.Equity    35001      43836      39871
##   + OtherLiabilitiesAndStockholdersEquity_  1199        239        452

## To calculate lagged difference for entire statement use diff function.
## The result is statement of changes between successive years

diff_bs <- diff(balance_sheet)

## Print the lagged differences; capture the output to a NULL file, and reformat
## with the kableExtra package

capture.output(bs_table <- print(diff_bs, html = FALSE, big.mark = ",",
  dateFormat = "%Y"), file='NULL')

bs_table %>%
  kable(longtable=T,
    caption="Lagged Differences in Balance Sheets",
    "latex",
    booktabs = T) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width =
F, font_size=10)
```

Table 3 Lagged differences in balance sheets

Element	2017-12-31	2016-12-31
Assets =	-9208	27,352
+ AssetsCurrent =	-7459	6795
+ CashAndCashEquivalentsAtCarryingValue	2552	-2278
+ MarketableSecuritiesCurrent	-3528	3678
+ AccountsNotesAndLoansReceivableNetCurrent	-1474	1301
+ InventoryNet	-3125	24
+ gm_AssetsSubjecttoorAvailableforOperatingLeaseNetCurrent	-790	-887
+ OtherAssetsCurrent	450	943
+ AssetsOfDisposalGroupIncludingDiscontinuedOperationCurrent	0	0
+ NotesAndLoansReceivableNetCurrent	0	0
+ AssetsNoncurrent =	-1749	20,557
+ EquityMethodInvestments	77	-205
+ PropertyPlantAndEquipmentNet	433	4591
+ IntangibleAssetsNetIncludingGoodwill	-410	312
+ DeferredIncomeTaxAssetsNet	-11,548	-1768
+ OtherAssetsNoncurrent	859	1049
+ DisposalGroupIncludingDiscontinuedOperat...	0	0
+ NotesAndLoansReceivableNetNoncurrent	0	0
+ PropertySubjectToOrAvailableForOperatingLeaseNet	42,882	0
LiabilitiesAndStockholdersEquity =	-9208	27,352
+ Liabilities =	-1333	23,600
+ LiabilitiesCurrent =	-8291	13,964
+ AccountsPayableCurrent	-3032	2899
+ AccruedLiabilitiesCurrent	-3196	1599
+ LiabilitiesOfDisposalGroupIncludingDisco...	0	0
+ DebtCurrent	0	0
+ LiabilitiesNoncurrent =	6958	9636
+ OtherPostretirementDefinedBenefitPlanLiabilitiesNoncurrent	195	118
+ DefinedBenefitPensionPlanLiabilitiesNoncurrent	-4205	-2960
+ OtherLiabilitiesNoncurrent	-686	427
+ LiabilitiesOfDisposalGroupIncludingDisco...	0	0
+ LongTermDebtAndCapitalLeaseObligations	0	0
+ CommitmentsAndContingencies	0	0
+ StockholdersEquityIncludingPortionAttrib... =	-7875	3752
+ StockholdersEquity =	-8835	3965
+ CommonStockValue	-1	0
+ AdditionalPaidInCapital	-1612	-624
+ RetainedEarningsAccumulatedDeficit	-8541	5883
+ AccumulatedOtherComprehensiveIncomeLossNetOfTax	1319	-1294
+ MinorityInterest	960	-213

These are the basic tools that you need to access the information on sec.gov. Note that there are numerous reports on EDGAR; `finstrr` will be able to access and format any financial statements in XBRL format on the EDGAR database. Almost all of the EDGAR information is maintained in HTML format, and I will provide code later in this chapter to access and parse HTML files in EDGAR.

Caveats on accessing EDGAR information with R

One of the most useful functions that R and its packages offer auditors and accountants is the ability to quickly and directly access SEC filings with very simple R code. This comes with caveats, as EDGAR's database formatting and naming

conventions are not always stable, and can change without warning. Additionally, the R packages which support access to SEC databases are not always reliably maintained, and changes at the SEC may not be reflected in the code. When you run into problems using the EDGAR related code chunks in this chapter, it is advisable to visit <https://www.sec.gov/edgar/> to look at the files in the SEC's repositories, and see whether they conform to the expectations of R's packages. Though useful, the EDGAR functions in R require the auditor to be both flexible and willing to debug errant routines.

The following are some problems and workarounds I have found in my own use of these packages. For example, an auditor might run into problems accessing Tesla's data in XML format, during a time that the SEC seems to have changed naming conventions. Access of Tesla data up to 2018 works properly, as shown in the following code chunk.

```
library(finreportr)
# The following commands will directly load
# EDGAR information into the R workspace for analysis
tesla_co <- CompanyInfo("TSLA")
tesla_ann <- AnnualReports("TSLA")
tesla_inc <- GetIncome("TSLA", 2018)
tesla_bs <- GetBalanceSheet("TSLA", 2018)
tesla_cf <- GetCashFlow("TSLA", 2018)
head(tesla_inc)

##
# Metric          Units      Amount
## 1 Sales Revenue Goods Net U_iso4217USD 3431587000
## 2 Sales Revenue Goods Net U_iso4217USD 5589007000
## 3 Sales Revenue Goods Net U_iso4217USD 8534752000
## 4 Operating Leases Income Statement Lease Revenue U_iso4217USD 309386000
## 5 Operating Leases Income Statement Lease Revenue U_iso4217USD 761759000
## 6 Operating Leases Income Statement Lease Revenue U_iso4217USD 1106548000
##   startDate    endDate
## 1 2015-01-01 2015-12-31
## 2 2016-01-01 2016-12-31
## 3 2017-01-01 2017-12-31
## 4 2015-01-01 2015-12-31
## 5 2016-01-01 2016-12-31
## 6 2017-01-01 2017-12-31
```

But this code will not be able to access Tesla's 2019 reports, because it throws an error:

```
Error in fileFromCache(file.inst) :
Error in download.file(file, cached.file, quiet = !verbose) :
cannot open URL
'https://www.sec.gov/Archives/edgar/data/1318605/000156459020004475/tsla-20191231.xml'
```

What has happened: rather than asking for 'tsla-20191231.xml' the package should have asked for 'tsla-10k_20191231_htm.xml'. EDGAR either made a mistake in their index files, or changed naming conventions. You can explore this further by going to their website.

You can also use the xml2 package to read what is in the correct file, bypassing finreportr altogether (or just wait for the repositories to be updated with corrected code). Consider this workaround to access 2019 data.

```
library(xml2)
library(curl)
u1 <-
  'https://www.sec.gov/Archives/edgar/data/1318605/000156459020004475/
  tsla-10k_20191231_htm.xml'
url_file <- curl_download(u1, destfile="~/Downloads/u1.xml") # to download and save
list_url_1 <- as_list(read_xml(u1)) # to read into R
```

If you would rather avoid the challenge of working with XML altogether, there is another workaround. The EDGAR package allows you to download the 10-K in text or HTML in computer readable form on your computer.

```
library(edgar)
library(kableExtra)
cik.no = 0001318605 # Tesla
form.type = '10-K'
filing.year = 2019
quarter = c(1,2,3,4)

# getFilings function saves to directory
# '~/Edgar filings_full text' to store all downloaded filings.

getFilings(
  cik.no,
  form.type,
  filing.year,
  quarter,
  downl.permit="y")
## Downloading filings. Please wait...
##
| | 0%
|=====
| 100%
##      cik company.name form.type date filed quarter filing.year
## 1 1318605  Tesla Inc       10-K 2019-02-19        1      2019
##           accession.number   status
## 1 0001564590-19-003165 Download success

# getFilingsHTML function saves the filing content in
# '~/Edgar filings_HTML view' directory in HTML format.

getFilingsHTML(
  cik.no = cik.no,
  form.type = form.type,
  filing.year = filing.year,
  quarter = quarter
)
## Downloading filings. Please wait...
##
| | 0%
|=====
| 100%
## Scrapping full EDGAR and converting to HTML...
##
| | 0%
|=====
| 100%
## HTML filings are stored in 'Edgar filings_HTML view' directory.
##      cik company.name form.type date filed accession.number
## 1 1318605  Tesla Inc       10-K 2019-02-19 0001564590-19-003165

## HTML filings are stored in 'Edgar filings_HTML view' directory.
```

```
## cik company.name form.type date filed accession.number
## 1 1318605 Tesla Inc 10-K 2019-02-19 0001564590-19-003165
```

Additionally, you may wish to look at stock prices, and this is easy to do with the `tseries` package.

```
library(tseries)
tesla_stock <- get.hist.quote("TSLA")
## time series starts 2010-06-29
## time series ends 2020-09-16

## time series starts 2010-06-29
## time series ends 2020-03-26

plot(tesla_stock)
```

Audit Staffing and Budgets

The audit program lays out in advance of mid-year and year-end tests, the procedures that will be used to collect evidence and to analyze it with the objective of reporting the “correct” audit opinion, while keeping costs within the contracted audit budget. This section provides an example of an audit program that might be created after the risk assessment. In addition to auditing steps on samples drawn from specific computer files, the example demonstrates the sort of results that the audit would produce, and describes the corrective steps or reporting that would accompany the audit results.

Audit budgeting, which primarily is determined by the allocation of audit staff, is too often made in an ad hoc manner. Prior years’ budgets and assignments influence staffing; so does availability of knowledgeable staff. Human resource problems, especially in specialized, knowledge intensive industries such as auditing, will never be an exact science. Nonetheless, management should attempt to instill a reasonable level of cost–benefit discipline in staffing decisions.

At the planning stage, audit managers will determine the scope of each audit test. From a statistical perspective, this can be estimated as a cost proportional to the sample sizes that are decided on for the tests. From a staffing perspective, this is proportional to the number of auditors assigned to the audit tasks. Similarly, the benefit derived from that expenditure can be perceived in terms of the monetary error that could be detected; typically a percentage of the value of the account or years transaction stream.

Audit programs are collections of audit tests that test different critical transaction and systems processing features in the client’s accounting systems. I assume that audit planning required for each of the individual audit tests is set to a scope which overall maximizes the audit risk reduction for a given cost of conducting the audit with this program.

Staffing is “lumpy” in the sense that you typically get whole auditors assigned to an audit. The number of auditors assigned to an audit will be commensurate with their potential for detecting monetary error—the benefit received from an audit test. We would hope to see the following sort of relation between the potential benefit derived from performing a set of audit tasks, and the person-months of audit staff assigned to a specific audit program (and thus the cost) (Fig. 1).

```
library(ggplot2)

benefit <- seq(10, 10000, 10)
staff_allocated <- data.frame(benefit, floor(10*(log(benefit^.06)))) 

ggplot(staff_allocated,
       aes(staff_allocated[,1],staff_allocated[,2]) ) +
  geom_line() +
  labs(x ="Audit Risk Reduction",
       y ="Staff Auditor Person-Months")
```

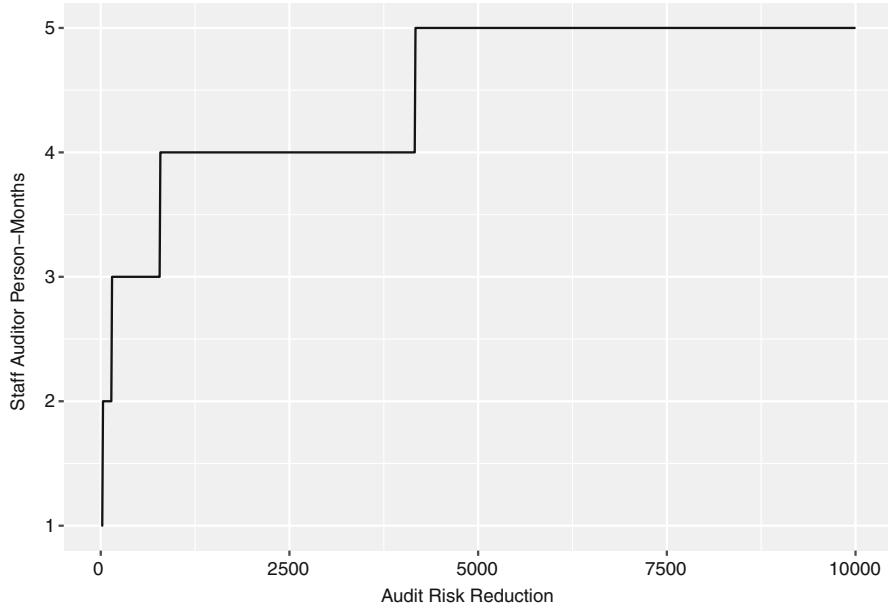


Fig. 1 Audit Budgets and risk reduction

The Risk Assessment Matrix

A Risk Assessment Matrix is constructed prior to and during the analytical review phase of the audit (see Fig. 2 for an example of a RAM dashboard). During this phase, the auditor will scan any business intelligence from media and Internet sources that would be relevant to potential risks to be encountered in the audit of the client. Prior years' working papers will also be perused for ascertain experiences, client specific risks, and application of the "rule of 3's" to adjust any anticipated risks and expectations during the current year audit.

Audit firms tend to enforce firm specific procedures in auditing. Each of the "Big 4" audit firms (which audit almost all of the listed firms on US exchanges) displayed unique biases in rendering adverse attestations: Ernst and Young focused on accounts receivables, revenue recognition, taxes, and fixed assets; PricewaterhouseCoopers focused on accounts receivables, revenue recognition, taxes, and payables; KPMG focused on accounts receivables, revenue recognition, taxes, and inventory; and Deloitte and Touche focused on revenue recognition, taxes, liabilities, inventory, and executive compensation (Cheffers 2012). These biases are likely to reflect signature audit methods, internal forms and checklists, and audit histories that are unique to individual firms. These firms will consequently allocate larger portions of the audit budget to certain accounts at the expense of others. Additionally, auditors tend to allocate more time to auditing debit balance accounts, assuming double-entry will assure the accuracy of the credit accounts. The specific accounts selected for audit depend on firm policy, procedures, and managing partners.

In this section, I will show how to construct a Risk Assessment Matrix on a client-server dashboard. Dashboards are well suited to auditing—they accommodate the information needs of auditors (and their laptops) in the field, while assuring the security, integrity, completeness, and privacy of client and audit records behind firewalls. The Risk Assessment Matrix will assume we are planning the audit of the simulated system presented in chapter "Simulated Transactions for Auditing Service Organizations" (Fig. 2).

Using Shiny to create a Risk Assessment Matrix Dashboard

Auditors face a particular problem in the field, in that much of the information they need may be in prior years' workpapers, in proprietary client files, in central locations in the audit firm, behind firewalls, and on powerful servers or cloud platforms. To secure the clients records and address privacy concerns, it is important that only the necessary information be maintained on mobile platforms such as laptops that are used in the field.

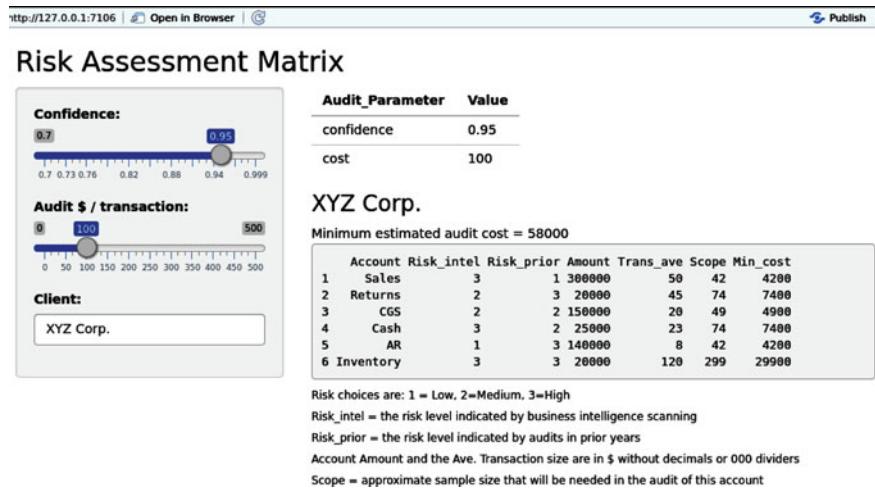


Fig. 2 The risk assessment matrix dashboard

The standard solution to such problems is found in client–server systems that place high-performance, secure systems on a centralized server, and provide the field auditor with light, client software that runs on a laptop, communicating with the server over the Internet. Shiny is the client–server extension of the R language, and as with the rest of the R language, is uniquely suited to handling the ad hoc nature of audits, where each audit often represents an entirely new set of analyses.

Shiny is a tool for fast prototyping of digital dashboards, giving you a large number of HTMLWidgets at your disposal which lend themselves really well to building general-purpose web applications. Shiny is particularly suited for fast prototyping and is fairly easy to use for someone who is not a programmer. Dashboards locally display some data (such as in a database or a file) providing a variety of metrics in an interactive way.

Reactive programming starts with reactive values that change in response to user input (such as positioning the confidence and cost sliders) and builds on top of them with reactive expressions that access reactive values and execute other reactive expressions. Reactivity based code for the Risk Assessment Matrix dashboard appears below. This has two parts: (1) the user interface “ui” which conceivably would operate on the field auditor’s laptop, and (2) the server-side “server” operations that would take place at the audit firms headquarters behind a firewall, and with access to firm and client files. First I will look at the user interface.

```
# Define the User Interface (UI)
ui <- fluidPage(

  titlePanel("Risk Assessment Matrix"),

  sidebarLayout(
    sidebarPanel(
      # Input: statistical confidence level of the audit tests
      sliderInput("confidence", "Confidence:",
                 min = .7, max = .999,
                 value = .95),

      # Input: cost of auditing per transaction sampled
      sliderInput("cost", "Audit $ / transaction:",
                 min = 0, max = 500,
                 value = 100),
    )
  )
)
```

```

textInput(inputId = "caption",
          label = "Client:",
          value = "XYZ Corp.")

),

# Main panel for displaying outputs
mainPanel(
  # Output: slider values entered
  tableOutput("values"),

  # Output: Formatted text for caption
  h3(textOutput("caption", container = span)),

  # Output: total cost of the audit
  textOutput("view"),

  # Output: RAM summary with sample sizes (scope) and cost
  verbatimTextOutput("summary"),

  h6("Risk choices are: 1 = Low, 2=Medium, 3=High"),
  h6("Risk_intel = the risk level indicated by business intelligence scanning"),
  h6("Risk_prior = the risk level indicated by audits in prior years"),
  h6("Account Amount and the Ave. Transaction size are in
      $ without decimals or 000 dividers"),
  h6("Scope = estimated discovery sample size that will be needed in the
      audit of this account"),
  h6("Audit cost = audit labor dollars per sampled transaction"),
  h6("Confidence = statistical confidence")
)
)

```

The mathematics of scope assessment takes place on the server. I used a very simple “discovery sampling” inspired model (see chapter “Design of Audit Programs”) to compute audit scope which I interpret as sample sizes for various transaction flows, computed as:

$$n \approx \frac{\log(1 - confidence)}{\log(1 - \frac{10 - risk_{intelligence} \times risk_{prior}}{100})}$$

These are dynamically (reactively in the Shiny vernacular) updated for changes in confidence level and transaction auditing costs established by the auditor. A total audit cost of field tests is computed, to be incorporated into the overall budget of the audit.

In practice, the server side of the Risk Assessment Matrix will have access to prior years working papers (assuming they are digitized) and to client accounting files, as well as proprietary audit firm data and technologies.

```
# Define Server-size calculations
server <- function(input, output) {
  devtools::install_github("westland/auditanalytics")
  library(auditanalytics)
```

```

# auditors risk assessment matrix generated from prior years' workpapers, etc.
ram <- read.csv(system.file("extdata", "risk_asst_matrix.csv", package =
  "auditanalytics", mustWork = TRUE)

# Reactive expression to create data frame of slider input values
sliderValues <- reactive({
  data.frame(
    Audit_Parameter = c("confidence",
                        "cost"),
    Value = as.character(c(input$confidence,
                           input$cost)),
    stringsAsFactors = FALSE)
})

# Show the values in an HTML table ----
output$values <- renderTable({
  sliderValues()
})

output$caption <- renderText({
  input$caption
})

# Recompute scope and cost whenever input$confidence or input$cost change

output$summary <- renderPrint({
  ram <- ram
  conf <- input$confidence
  cost <- input$cost
  risk <- (10 - (as.numeric(ram[,2]) * as.numeric(ram[,3]))) /100
  Scope <- ceiling( log(1-conf) / log( 1- risk))
  ram <- cbind(ram[,1:5], Scope)
  Min_cost <- Scope * cost
  ram <- cbind(ram[,1:6], Min_cost)
  ram
})

# Recompute minimum audit cost whenever input$confidence or input$cost change

output$view <- renderText({
  ram <- ram
  conf <- input$confidence
  cost <- input$cost
  risk <- (10 - (as.numeric(ram[,2]) * as.numeric(ram[,3]))) /100
  Scope <- ceiling( log(1-conf) / log( 1- risk))
  ram <- cbind(ram[,1:5], Scope)
  Min_cost <- Scope * cost
  minimum_audit_cost <- sum(Min_cost)
  c("Minimum estimated audit cost = ",minimum_audit_cost)
})
}
}

```

R Studio gives you various options for assembling Shiny apps, including apps with server-side code resident on either an R Studio or a bespoke server, and stand-alone client side apps which can be constructed with the following code.

```
#  
# See http://shiny.rstudio.com/ for documentation  
#  
  
library(shiny)  
  
# Define UI for application  
## Insert UI Code Here **  
  
# Define server logic required to draw a histogram  
## Insert Server Code Here **  
  
# Run the application  
shinyApp(ui = ui, server = server)
```

Generating the Audit Budget from the Risk Assessment Matrix

The Risk Assessment Matrix (RAM) will generate qualitative measures of risk along with initial estimates of minimum (discovery) sample sizes. This will generally not be sufficient to accurately budget the audit, since higher risk accounts will require audit scope beyond mere “discovery” of errors. In addition, planning will need to estimate costs associated with:

1. Estimating the rates of errors from control weaknesses of all specific types (in interim tests),
2. Assessing the existence and amount of errors in trial balance accounts (in substantive tests), and
3. Generally assessing structural and qualitative problems in financial information consolidation and presentation.

The third item is beyond the scope of simple technical metrics, and will require the experience and judgment of audit managers. It can probably best be estimated by reviewing prior years’ budgets and assuming similar costs for the current year audit. The first two items, though, can be budgeted through a relatively simple linear model with assumptions which reflects the cost structure of a particular audit firm. Though each RAM will be auditor and client specific, the prior interactive RAM software can easily be programmed to incorporate such a linear model. This makes it possible to write the technical parts of the audit program and budgets (items 1 and 2 above) automatically from the interactive RAM software. The code chunks, models, and algorithms provided below comprise the building blocks to facilitate such an automatic generation.

Technical Sampling Structure of the Audit Program

The technical tests of internal control (interim testing) and account balances (substantive testing) consist of audit work investigating the items in transaction samples—they are transaction-centric. The unit of audit work is a sampled transaction, and each type of transaction will be subject to misaccounting through a variety of control weaknesses.

Consider a sales transaction. A sale could be recorded at the wrong amount, or the wrong item sold could have been recorded, or the sale could have been shipped to the wrong customer, or be recorded in the wrong period. Each of these problems reflect a specific audit risk, control weakness, and audit procedure. Auditors’ concern with such errors differs in the interim and substantive tests. In interim tests, the auditor is concerned with estimating the rate at which each type of error occurs. Where this rate suggests a control weakness is significant, the auditor needs to expand the scope of substantive auditing of account balances that are affected by the control weakness.

Placing these considerations in a more formal mathematical setting, let $T_{i,j}$ represent a particular control weakness j in transaction type i . Let $S_{i,j}$ be the interim testing sample size suggested by the RAM to test for control weakness j in transaction type i . Let $C_{i,j}$ be the audit cost to test for control weakness j in a single transaction of type i . Then a simple linear cost model would be:

$$\text{Total cost of technical interim tests} = \sum_{i,j} T_{i,j} S_{i,j} C_{i,j}$$

The matrix form is typically more useful in writing R language code, because the transaction, cost, and sample values are matrices that can directly use the fast BLAS/LAPACK implementations in R for linear algebra, rather than coding slow, messy, nested for statements. The matrix form is:

$$\text{Total cost of technical interim tests} = 1_{(i)}^T \cdot (T \cdot S^T \cdot C) \cdot 1_{(j)}$$

where $1_{(i)}^T$ is the row i -vector whose entries are all 1's, $1_{(j)}$ is the column j -vector whose entries are all 1's, $T = \{T_{i,j}\}$, $S = \{S_{i,j}\}$, and $C = \{C_{i,j}\}$.

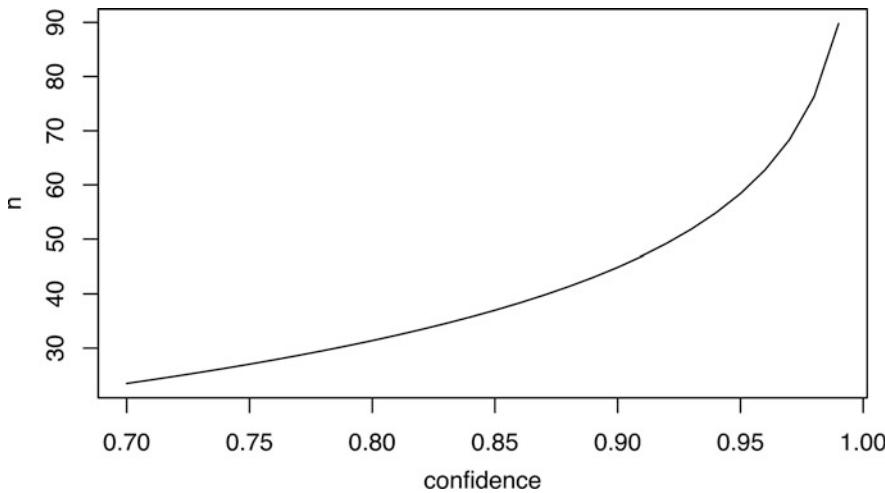
Sample Sizes for Budgeting

There are two types of sampling in interim tests:

1. Discovery sampling for discovery of *out-of-control* transaction streams.
2. Attribute sampling for estimating transaction error rate.

Discovery sampling sets a sample size that is likely to discover at least one error in the sample if the actual transaction error rate exceeds the minimum acceptable error rate (alternatively called the out-of-control rate of error). Discovery tests help the auditor decide whether the systems processing a particular transaction stream are in or out of control. Budgeted sample sizes in interim testing will depend on whether the RAM suggests that control risk is low or high. If it is low, then the discovery sample size plus a “security” factor for cases where error is discovered will estimate the scope of auditing.

```
confidence <- seq(.99,.7,-.01)
n <- (log(1-confidence))/log(1-.05)
plot(confidence,n, type="l")
```



So for a 5% intolerable error rate at 95% confidence we have:

```
confidence <- .95
n <- (log(1-confidence))/log(1-.05)
cat("\n Discovery sample size = ", ceiling(n))

## 
## Discovery sample size = 59
```

Where the RAM assesses control risk to be anything higher, the auditor can assume that scope will be expanded to include attribute sampling. Attribute sampling estimates the error rate in the entire transaction population with some confidence (e.g.,

95%) that the estimate is within the out-of-control error rate cutoff for that transaction stream. If it is found that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Errors estimates from attribute samples may either be *rates* or *amounts* or both.

If discovery sampling suggests that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Attribute sampling size is determined using Cohen's power analysis (Cohen 1992) which is implemented in R's *pwr* package. We compute both in the following code chunk

```

library(readr)
library(pwr)

## Attribute sample for estimating 'rate' of errors

size <- 1000                                     ## number of transactions
Delta <- .05*size                                ## detect 5% occurrence error
sigma <- .3*size                                  ## variability (guess ~1/3 rd)
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")                          ## look for overstatement of earnings

cat("\n Attribute sample size for occurrence of error = ", ceiling(sample$n))

## 
## Attribute sample size for occurrence of error = 224

## Attribute sample for estimating 'amount' of errors

size <- 100000                                    ## total amount of transactions
mu <- 50                                         ## average value of transaction
Delta <- .05*mu                                   ## detect 5% amount intolerable error
sigma <- 30                                         ## variability
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")    ## look for sales value too large

cat("\n Attribute sample size for amount of error = ", ceiling(sample$n))

## 
## Attribute sample size for amount of error = 892

```

The auditor faces different decisions in substantive testing. The particular type of account determines the impact of control weaknesses found in interim testing. For example, a 5% error rate in a \$1 million sales account discovered in interim testing implies a \$50,000 error in annual sales on the trial balance. In contrast, assume that accounts receivable turn over 10 times annually, then that 5% error rate implies only a \$5000 misstatement in accounts receivable. Whether sales or accounts receivable are "fairly stated" depends on the materiality level set by the auditor—a \$10,000 materiality level would imply that sales is not fairly presented, while accounts receivable is fairly stated.

At year-end where there will be a complete set of transactions available for the year, and substantive samples are typically focused on acceptance sampling to determine if the account balance is “fairly stated” (does not contain intolerable or material error). The approach is the same as attribute sampling of amounts, and is inherently more straightforward than interim control tests. Substantive tests estimate the error rate in an account balance with some confidence (e.g., 95%) that the estimate is within the “materiality” or “intolerable error” cutoff for that account balance.

For example, consider sampling sales invoices from the accounts receivable aging report and comparing them to supporting documentation to see if they were billed in the correct amounts, to the correct customers, and on the correct dates. Additionally, auditors might trace invoices to shipping log, and match invoice dates to the shipment dates for those items in the shipping log, to see if sales are being recorded in the correct accounting period. This can include an examination of invoices issued after the period being audited, to see if they should have been included in a prior period.

Acceptance sampling size is determined using Cohen’s power analysis (Cohen 1992) which is implemented in R’s *pwr* package. If discovery sampling suggests that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Errors estimates from attribute samples may either be *rates of erroneous transactions* or from a monetary unit sampling perspective, can be *rates of monetary error in the transaction stream*. We compute both in the following code chunk

```
library(readr)
library(pwr)

## Acceptance sample for estimating 'amount' of error in an account balance

size <- 100000
mu <- 50
Delta <- .05*mu
sigma <- 30
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater") ## look for value too large

cat("\n Attribute sample size for amount of error = ", ceiling(sample$n))

##
## Attribute sample size for amount of error =  892
```

Notable Audit Failures and Why They Occurred

Audit and accounting practice have been strongly influenced by a string of scandals that have occurred nearly every 7 years since the Reagan reforms of the early 1980s. The medical, legal and accounting professions were opened up to free-market forces when all three professions were allowed direct-to-consumer marketing and brand-building. Rules regarding pricing and competition for talent were also relaxed. In auditing the repeal of AICPA Ethics Rules Section 501 on advertising, recruiting from other firms and other free-market innovations moved the industry from its cossetted clubby culture to a profit-oriented business. By the late 1990s audit firms were averaging \$7 of IT revenue for every \$1 of audit revenue.

The year 2001 witnessed a series of financial frauds involving Enron Corporation auditing firm Arthur Andersen, the telecommunications company WorldCom, Qwest, and Sunbeam among other well-known corporations. These problems highlighted the need to review the effectiveness of accounting standards, auditing regulations and corporate governance principles. In some cases, management manipulated the figures shown in financial reports to indicate a better economic performance. In others, tax and regulatory incentives encouraged over-leveraging of companies and decisions to bear extraordinary and unjustified risk.

The Enron scandal deeply influenced the development of new regulations to improve the reliability of financial reporting and increased public awareness about the importance of having accounting standards that show the financial reality of companies and the objectivity and independence of auditing firms.

In addition to being the largest bankruptcy reorganization in American history, the Enron scandal undoubtedly is the biggest audit failure. The scandal caused the dissolution of Arthur Andersen which at the time was one of the five largest accounting firms in the world.

One consequence of these events was the passage of the Sarbanes–Oxley Act in 2002. The act significantly raises criminal penalties for securities fraud for destroying, altering, or fabricating records in federal investigations or any scheme or attempt to defraud shareholders.

Auditing: A Wicked Problem

In theory, independent audits increase the value and credibility of the financial statements, reduce investor risk, and reduce the cost of capital of audited firms. They are often required under securities law and by investors and creditors. But such assertions have grown increasingly contentious over time as the definitions and usage of financial statements have evolved and changed. Scholars are increasingly aware that many of auditing's difficulties reflect its status as a wicked problem.

Wicked problems are difficult or impossible to solve because of incomplete, contradictory and changing requirements that are often difficult to recognize. The term wicked is used not in the sense of evil but rather its resistance to resolution. Moreover because of complex interdependencies, the effort to solve one aspect of a wicked problem may reveal or create other problems. Contrast this with relatively tame soluble problems in mathematics, chess or puzzle solving (Coyne 2005; Ludwig 2001; Rittel and Webber 1973).

Audits are classic “wicked problems” sharing the following characteristics (Rittel and Webber 1973):

1. There is no definitive formulation of auditing.
2. Audits have no stopping rule.
3. Audit products are not true-or-false rather are opinions.
4. There is no immediate and no ultimate test of an audit conclusion.
5. Each audit is a one-shot operation.
6. Audits do not have an enumerable or an exhaustively describable set of potential solutions nor is there a well-described set of permissible operations that may be incorporated into the audit program.
7. Every audit is essentially unique.
8. Every audit procedure can be considered to be a response to some other finding.
9. The existence of an audit discrepancy can be explained in numerous ways. The choice of explanation determines the nature of the problems resolution and the audit opinion.
10. The auditor has no right to be wrong and is ultimately liable for the consequences of audit opinions.
11. The solution depends on how the problem is framed and vice versa, i.e., the problem definition depends on the solution.
12. Stakeholders have radically different world views and different frames for understanding the problem.
13. The constraints that the problem is subject to and the resources needed to solve it change over time.

Roberts (2002) and Roberts (2000) identified three strategies to address wicked problems: competitive, authoritative and collaborative.

Competitive strategies attempt to solve wicked problems by pitting opposing points of view against each other requiring parties that hold these views to come up with their preferred solutions. The advantage of this approach is that different solutions can be weighed up against each other and the best one chosen. The disadvantage is that this adversarial approach creates a confrontational environment in which knowledge sharing is discouraged. Consequently the parties involved may not have an incentive to come up with their best possible solution.

Collaborative strategies try to engage all stakeholders in order to find the best possible solution for all stakeholders. Typically these approaches involve meetings in which issues and ideas are discussed and a common agreed approach is formulated. Such approaches share many of the disadvantages of competitive approaches and can be even more time consuming.

Authoritative strategies concentrate responsibility for solving the problems in the hands of a few people, for example Certified Public Accountants. The reduction in the number of stakeholders reduces problem complexity as many competing points of view are eliminated at the start. The disadvantage is that authorities and experts charged with solving the problem lack all of the information needed to efficiently and completely solve the problem.

Auditing for better or worse has chosen to couch the industrial organization of the audit business in an authoritarian structure. Authority for rendering audit opinions is vested in a small number of firms; the Fortune 500 firms must typically be audited by one of the Big Four audit firms; firm employees must receive extensive training typically 4–5 years of college education and passage of a CPA certification examination.

Audits authoritative strategy for solving its wicked problem offers the advantage of substantially reduced cost and greater efficiency of audits. Its disadvantage is that no matter how well educated the auditors are they will not have as much information about IT storage and processing platforms, financial investment instruments, production operations, international law and so forth as dedicated professionals. Information technology along with the automation of audit tasks and decision-making are helping auditors respond to an increasingly complex business environment in the context of their authoritative strategy.

Final Thoughts on Audit Planning and Budgets

Planning and budgeting of audits is not an exact science, rather it is contextual, with uncertain outcomes, incomplete and asymmetric information, and ongoing negotiations with the client and between audit offices. The judgment and experience of auditors, managers, and the profession are essential to a complete and effective planning process. The prior exposition moves us toward quantification and standardization of the process. It offers audit professionals computational and statistical tools to automate tedious, repetitive tasks, while allowing them to focus on critical issues that demand their expertise. Since audit planning will often be highly collaborative, involving professionals in different offices around the world, computational tools to aid in standardizing and objectifying procedures across languages and cultures can be expected to improve efficiency and effectiveness of auditing, while insuring against significant deficiencies and miscommunications in audit planning and structure.

References

- AICPA. 1980. Statement of Financial Accounting Concepts No. 2 Qualitative Characteristics of Accounting Information.
- Bedard, Jean C., Lynford Graham, and Cynthia Jackson. 2005. Information Systems Risk and Audit Planning. *International Journal of Auditing* 9 (2): 147–163.
- Cheffers, D. Whalen, M. 2012. SOX 404 Dashboard: Year 6 Update. Report. Audit Analytics.
- Cohen, Jacob. 1992. A Power Primer. *Psychological Bulletin* 112 (1): 155.
- Coyne, Richard. 2005. Wicked Problems Revisited. *Design Studies* 26 (1): 5–17.
- Jones, Peter. 2017. *Statistical Sampling and Risk Analysis in Auditing*. London: Routledge.
- Knechel, W. Robert. 2007. The Business Risk Audit: Origins, Obstacles and Opportunities. *Accounting, Organizations and Society* 32 (4–5): 383–408.
- Ludwig, Donald. 2001. The Era of Management Is over. *Ecosystems* 4 (8): 758–764.
- Rittel, Horst W.J., and Melvin M Webber. 1973. Dilemmas in a General Theory of Planning. *Policy Sciences* 4 (2): 155–169.
- Roberts, Nancy. 2000. Wicked Problems and Network Approaches to Resolution. *International Public Management Review* 1 (1): 1–19.
- Roberts, Nancy C. 2002. Keeping Public Officials Accountable Through Dialogue: Resolving the Accountability Paradox. *Public Administration Review* 62 (6): 658–669.
- Srivastava, Rajendra P., and Glenn R. Shafer. 2008. Belief-Function Formulas for Audit Risk. In *Classic Works of the Dempster–Shafer Theory of Belief Functions*, 577–618. Berlin: Springer.

Analytical Review: Technical Analysis



Analytical Review

Analytical procedures are evaluations of financial information made by a study of plausible relationships between both financial and non-financial data. Analytical procedures are used in all stages of the audit including planning, substantive testing, and final review. It serves as a vital planning function in the entirety of the audit procedures.

Analytical review procedures are applied at several points in a firm's audit to establish "plausible relationships among both financial and non-financial data" as a cost-effective test of the conclusions derived from fieldwork. These procedures compare financial and organization information of an audit client such as budgets, industry news, firm guidance and news releases, forecasts, market information, non-financial information such as minutes and contracts, as well as bank and tax information. Their goal is to flag "exceptional" situations that would require more extensive auditing, and commensurately expand the scope of auditing for these accounts. Prior to interim and substantive fieldwork, they play an essential role in assisting the auditor to "adequately plan work" and "obtain a sufficient understanding of the entity and its environment." Prior to fieldwork, auditors have very little access to client transactions (which may not have yet occurred) and analytical procedures based on industry and economy-wide news offer cost-effective tools for audit planning.

The auditor's "analytical review of significant ratios and trends" relies on historical ratios which provide the basis for assessing risk and planning the audit (AICPA, 1988). AICPA statement AU Section 329 states that analytical procedures are used throughout the audit engagement—in audit planning, execution, and review—and specifically at three times during the audit:

1. To assist in planning the nature, timing, and extent of other auditing procedures;
2. As a substantive test to obtain audit evidence about assertions related to account balances or classes of transactions
3. As an overall review of the financial information in the final review stage of the audit.

The AICPA couches its pronouncements on analytical procedures in the language of statistics, borrowing statistical terms such as "precision," "likelihood," and "expectation" in describing analytical tests. As audit technologies evolve, auditor judgment has steadily been augmented or superseded by objective statistical methods that aim to make auditing more cost-effective, and control the risk of audit error. Central to the modern audit's quasi-statistical framework are the assumptions made about data distributions in framing and implementing audit decisions. A likely reason that audits have not embraced statistical approaches more completely is the complexity of financial data distributions. Seldom is it possible to make the standard Gaussian assumption underlying commonly used parametric statistical tests and estimators. Empirical research suggests that financial data distributions vary considerably from firm to firm and account to account, and may be multimodal, fat tailed, or otherwise difficult to use.

Institutional Context of Analytical Review

The AICPA defines analytical review procedures to "... consist of evaluations of financial information made by a study of plausible relationships among both financial and non-financial data." Analytical procedures are used in planning the audit; as a substantive test to obtain audit evidence about assertions related to account balances or classes of transactions; and as an overall review of the financial information in the final review stage of the audit. The AICPA emphasizes that "... in

some cases, analytical procedures can be more effective or efficient than tests of details for achieving particular substantive testing objectives.” Official pronouncements of the AICPA on the nature and objectives of analytical review support the overall objectives of an audit: discovery of material errors in the financial statements. Analytical review procedures flexibly and opportunistically rely on external information available at the time of planning the audit; they depend heavily on non-transaction data sources, and typically need to make explicit or implicit assumptions about the underlying statistical characteristics of such data and its relevance to the financial statements to be audited.

Analytical review procedures generally refer to any qualitative or quantitative audit methods that do not directly relate to tests of the client’s own transactions and balances during the control and substantive testing of the financial system. Analytical procedures involve comparisons of recorded amounts, or ratios developed from recorded amounts, to expectations developed by the auditor. According to the AICPA:

“The auditor develops such expectations by identifying and using plausible relationships that are reasonably expected to exist based on the auditor’s understanding of the client. The expected effectiveness and efficiency of an analytical procedure in identifying potential misstatements depends on, among other things, the precision of the expectation. The expectation should be precise enough to provide the desired level of assurance that differences that may be potential material misstatements, individually or when aggregated with other misstatements, would be identified for the auditor to investigate. As expectations become more precise, the range of expected differences becomes narrower and, accordingly, the likelihood increases that significant differences from the expectations are due to misstatements. Expectations developed at a detailed level generally have a greater chance of detecting misstatement of a given amount than do broad comparisons. Monthly amounts will generally be more effective than annual amounts and comparisons by location or line of business usually will be more effective than company-wide comparisons.”

Analytical review procedures have been in use at least since the 1940s, and were introduced into the authoritative auditing literature in 1970 with the promulgation of Statement on Auditing Procedures (SAP) No. 54 (AU section 320.70). This Statement specified that sufficient competent evidential matter “is obtained through two general classes of auditing procedures:

1. Tests of details of transactions and balances
2. Analytical review of significant ratios and trends.

Statistical terms such as “precision,” “likelihood,” and “expectation” in the audit context are presented but not further defined in the AICPA’s pronouncements. But statistical tests commonly used by auditors do provide specific and practical definitions of these terms. Additionally, “assurance” as used in the context of analytical review for substantive tests has been linked in SAS 56 to the “statistical confidence” that the auditor has in his or her tests. SAS No. 56 required auditors to use analytical procedures to assist in planning the nature, timing, and extent of other auditing procedures and as an overall review of the financial statements in the final review of the audit. The Statement also recommended that analytical procedures be used “as a substantive test to obtain evidence.”

Over the last half century, research in many areas has expanded the statistical toolset available to auditors for analytical procedures. Research assessing account values and transaction flows using statistical time-series methods appear in, e.g., Asare and Wright (1997a,b), Kinney and Uecker (1982), Kinney (1979), Loebbecke and Steinbart (1987), Mueller and Anderson (2002), Nelson (1994), Stringer (1975). The practical application of various judgmental or hybrid approaches has been investigated through an equally rich stream of behavioral audit decision research, which has appeared in, e.g., Ameen and Strawser (1994), Anderson et al. (1995), Anderson et al. (1994), Biggs et al. (1988), Kogan et al. (2010), Libby (1985). Statistical approaches have been augmented with computationally intensive machine learning approaches and this is currently an active area of development in the audit industry; e.g., see, Wilson and Colbert (1989), Coakley (1995), Koskivaara (2004a), Koskivaara (2006), Coakley and Brown (1993), Knechel (1988), Coakley and Brown (1993), Koskivaara and Back (2007), Koskivaara (2004b).

During the planning stage, the analytical review procedure enhances the auditor’s understanding of the client’s business and helps in identifying significant transactions and events that have occurred since the last audit date. It also identifies unusual transactions and events, amounts, ratios, or trends that might be significant to the financial statements and may represent specific risks relevant to the audit.

Risk assessment procedures require understanding the entity and its environment. The whole process assists the auditor in planning the nature, extent and timing of other auditing procedures. Measuring risks to identify significant areas requiring the auditor’s attention is also covered here. Results of the analytical procedures performed during the planning stage help identify risks that may, based on the auditor’s judgment, require special audit consideration. Examples of these are those that are considered as non-routine, unusual and complex transactions, business risks that may result in material misstatement, fraud risk, significant related party transactions, accounting estimates and principles.

Analytical procedures include the review of data aggregated at high levels, such as comparing financial statements to budgeted or anticipated results. Generally, financial data are used, but relevant non-financial data (e.g., number of employees, square footage of selling space, or volume of good produced) may also be considered.

Analytical review procedures typically include a review of the current and prior year's financial statements and the current year's budget. Comparisons are made between the current year's actual and budgeted financial statements. There must also be an analysis to compare the current and previous year's actual financial statements to test for internal consistency.

Auditors must develop independent expectations for comparison to recorded amounts. Examples of these expectations include financial information for comparable prior periods, anticipated results from budgets and forecasts, relationship among data within the current period, industry norms and relationships of financial data with non-financial information. Income statement accounts have more predictable relationships compared to balance sheet accounts. In addition, accounts based on management discretion such as bonus, other employee benefits, and other related expenses are less predictable.

In performing analytical procedures, the auditor may also use financial analysis ratios which may be classified as liquidity ratios, activity ratios, profitability ratios, investors ratios, and long-term debt paying ability ratios. Additional analyses, such as common size analysis (vertical and horizontal), analysis of industry statistics, and trend analysis, may also be valuable depending on the nature of transactions.

Recent years have seen an increased emphasis on the use of analytical procedures, as it helps identify significant audit risks without relying on the client's own attestations. However, analytical review comparisons are based on expected plausible relationships among data. Limitations exist in the use of analytical procedures because differences noted do not necessarily indicate errors or fraud, but simply the need for further analysis and investigation—specifically of the client's own documentation, as well as entities doing business with the client. Changes in an account and in accounting principles, and inherent differences between industry norms and the client all contribute to fluctuations in expected amounts. Hence, the auditor needs to exercise professional judgment in analyzing the results.

Analytical procedures are usually designed to point out audit areas that are indicative of potential risks, need special emphasis or additional attention. Thus there are a number of practical guidelines that should dictate their application:

1. Avoid mechanical computations and comparisons. Instead, determine trends, ratios, and relationships that are most relevant to the business. Develop expectations on plausible or predictable relationships based on historical patterns in the operation of the business. These will serve as benchmarks for comparisons to determine unusual or unexpected changes.
2. Unusual or unexpected relationships would be characterized by anything out-of-the-ordinary or those that do not make sense or at odds with comparable industry data
3. For nonprofit organizations, analytical procedures should lead to information regarding changes in programs, nature of activities, grantors, fund-raising events, political environment, and the impact of the economy in the collection of promises to give.

Some of the typical analytical procedures applied are:

1. Comparison of receipts from annual fund-raising drives to total support. This procedure could help detect improper revenue recognition.
2. Comparison of support and revenue by source for the past 5 years. This procedure may identify revenue sources that require increased attention.
3. Evaluation of the ratio of fund-raising expenses to contribution revenues. This could also help detect improper revenue recognition. If the expense is significantly lower than the amount needed to produce recorded revenue, it might indicate overstating of revenues. If there is little or no fund-raising expense while reporting contribution revenue, this might indicate underreported expenses to maintain favorable expense ratios.
4. Scanning of financial information to identify unusual changes, unexpected relationships, and major fluctuations which could indicate specific areas of risk of material misstatement. For example, a large increase in notes payable might indicate new loans acquired. Also, significant changes in total net assets or changes in net assets by class (unrestricted, temporarily restricted, permanently restricted) might indicate unfavorable trends or going concern problems.
5. Obtaining information from prior audits. This enables the auditor to make preliminary judgments about the inherent and control risks about material accounts and to focus on substantive procedures to reduce the detection risk.

Technical Measures of a Company's Financial Health

Analytical review information is obtained from a variety of sources, internal and external to the firm. There are a wealth of methods available for firm intelligence gathering, a reflection of the rapid growth of Internet resources on financial analysis and firm intelligence. I will start with a brief analysis of internal technical metrics for analytical review—primarily accounting, equations and ratios. These traditionally were the main source of information, alerting the auditor to potential irregularities and accounting risks, as well as control weaknesses. Over the past decade, new information has become available on websites, forums, social network feeds, and databases which is arguably much more informative than crude ratio metrics. Financial analysts routinely monitor the Internet for firm related information, which has become much more useful in assessing firm health and problems than the limited, retrospective metrics reported by accountants.

Technical metrics have their roots in Al-Khwārizmi's method (from *The Compendious Book on Calculation by Completion and Balancing*, c. 825) which introduced algebra to accounting leading to the three fundamental equations:

1. The Bookkeeping equation (double-entry) for error control,
2. "Real" accounts for tracking wealth; this is called the "basic accounting equation." An elaborate form of this equation is presented in a balance sheet which lists all assets, liabilities, and equity, as well as totals to ensure that it balances, and
3. "Nominal" accounts for tracking activity; closing out these accounts at year-end yields the net income amount, which arguably is the most important single statistic produced in the accounting process from a stockholders perspective.

Technical metrics in analytical review involve an exploration of these fundamental equations and their components, both in current and prior years' statements and in the context of information obtained from interviews, news, and other sources, with the aim of identifying areas of risk in the current year's audit. Ratio analysis provides one of the specific ways that these relationships may be expanded to compare with other firms in an industry or between accounting periods for the same firm.

A financial ratio (or accounting ratio) is a relative magnitude of two selected numerical values taken from an enterprise's financial statements. Often used in accounting, there are many standard ratios used to try to evaluate the overall financial condition of a corporation or other organization. Financial ratios may be used by managers within a firm, by current and potential shareholders (owners) of a firm, and by a firm's creditors. Financial analysts use financial ratios to compare the strengths and weaknesses in various companies. If shares in a company are traded in a financial market, the market price of the shares is used in certain financial ratios.

Ratios can be expressed as a decimal value, such as 0.10, or given as an equivalent percent value, such as 10%. Some ratios are usually quoted as percentages, especially ratios that are usually or always less than 1, such as earnings yield, while others are usually quoted as decimal numbers, especially ratios that are usually more than 1, such as P/E ratio; these latter are also called multiples. Given any ratio, one can take its reciprocal; if the ratio was above 1, the reciprocal will be below 1, and conversely. The reciprocal expresses the same information, but may be more understandable: for instance, the earnings yield can be compared with bond yields, while the P/E ratio cannot be: for example, a P/E ratio of 20 corresponds to an earnings yield of 5%.

Values used in calculating financial ratios are taken from the balance sheet, income statement, statement of cash flows or (sometimes) the statement of retained earnings. These comprise the firm's "accounting statements" or financial statements. The statements' data is based on the accounting method and accounting standards used by the organization.

Purpose and Types of Ratios

Financial ratios quantify many aspects of a business and are an integral part of the financial statement analysis. Financial ratios are categorized according to the financial aspect of the business which the ratio measures:

- Liquidity ratios measure the availability of cash to pay debt.
- Activity ratios measure how quickly a firm converts non-cash assets to cash assets.
- Debt ratios measure the firm's ability to repay long-term debt.
- Profitability ratios measure the firm's use of its assets and control of its expenses to generate an acceptable rate of return.
- Market ratios measure investor response to owning a company's stock and also the cost of issuing stock. These are concerned with the return on investment for shareholders, and with the relationship between return and the value of an investment in company's shares.

Financial ratios allow for comparisons:

- between companies
- between industries
- between different time periods for one company
- between a single company and its industry average

Ratios generally are not useful unless they are benchmarked against something else, like past performance or another company, or industry averages. Thus, the ratios of firms in different industries, which face different risks, capital requirements, and competition are usually hard to compare. Financial ratios may not be directly comparable between companies that use different accounting methods or follow various standard accounting practices.

Most public companies are required by law to use generally accepted accounting principles for their home countries, but private companies, partnerships, and sole proprietorships may not use accrual basis accounting. Large multi-national corporations may use International Financial Reporting Standards to produce their financial statements, or they may use the generally accepted accounting principles of their home country.

There is no international standard for calculating the summary data presented in all financial statements, and the terminology is not always consistent between companies, industries, countries, and time periods. Various abbreviations may be used in financial statements, especially financial statements summarized on the Internet. Sales reported by a firm are usually net sales, which deducts returns, allowances, and early payment discounts from the charge on an invoice. Net income is always the amount after taxes, depreciation, amortization, and interest, unless otherwise stated. Other commonly used terms are:

- COGS = Cost of goods sold or cost of sales.
- EBIT = Earnings before interest and taxes
- EBITDA = Earnings before interest, taxes, depreciation, and amortization
- EPS = Earnings per share

Common Technical Metrics

- **Debt-to-equity ratio:** compares a company's total debt to shareholders' equity. Both of these numbers can be found on a company's balance sheet. To calculate debt-to-equity ratio, you divide a company's total liabilities by its shareholder equity, or

$$\text{Debt-to-Equity Ratio} = \text{Total Liabilities} / \text{Shareholders' Equity}$$

If a company has a debt-to-equity ratio of 2 to 1, it means that the company has two dollars of debt to every one dollar shareholders invest in the company. In other words, the company is taking on debt at twice the rate that its owners are investing in the company.
- **Inventory turnover ratio:** compares a company's cost of sales on its income statement with its average inventory balance for the period. To calculate the average inventory balance for the period, look at the inventory numbers listed on the balance sheet. Take the balance listed for the period of the report and add it to the balance listed for the previous comparable period, and then divide by two. (Remember that balance sheets are snapshots in time. So the inventory balance for the previous period is the beginning balance for the current period, and the inventory balance for the current period is the ending balance.) To calculate the inventory turnover ratio, you divide a company's cost of sales (just below the net revenues on the income statement) by the average inventory for the period, or

$$\text{Inventory Turnover Ratio} = \text{Cost of Sales} / \text{Average Inventory for the Period}$$

If a company has an inventory turnover ratio of 2 to 1, it means that the company's inventory turned over twice in the reporting period.
- **Operating margin:** compares a company's operating income to net revenues. Both of these numbers can be found on a company's income statement. To calculate operating margin, you divide a company's income from operations (before interest and income tax expenses) by its net revenues, or

$$\text{Operating Margin} = \text{Income from Operations} / \text{Net Revenues}$$

Operating margin is usually expressed as a percentage. It shows, for each dollar of sales, what percentage was profit.
- **P/E ratio:** compares a company's common stock price with its earnings per share. To calculate a company's P/E ratio, you divide a company's stock price by its earnings per share, or

$$\text{P/E Ratio} = \text{Price per share} / \text{Earnings per share}$$

If a company's stock is selling at \$20 per share and the company is earning \$2 per share, then the company's P/E Ratio is 10 to 1. The company's stock is selling at 10 times its earnings.

- **Working capital:** is the money leftover if a company paid its current liabilities (that is, its debts due within 1-year of the date of the balance sheet) from its current assets.
- Working Capital = Current Assets - Current Liabilities

And so forth. There are numerous ratios available for analysis, but diminishing returns to the auditor for some of the less used ratios. The above ratios are probably sufficient for a quick check of corporate health. Where the auditor or analyst has questions, there is a wealth of information online about how to compute financial ratios and other technical metrics. Do not hesitate to use these resources where specific situations require something more than these most basic metrics.

Accessing Financial Information from EDGAR (<https://www.sec.gov/edgar/>)

The EDGAR database offers a wealth of financial information on industry competitors, but only if they are publicly traded companies. Use package `edgar` to extract this information. The subsequent R code extracts information on Tesla Motors from Edgar; this information is then analyzed for sentiment, which is presented as a bar chart.

```
library(edgar)
```

```
## Registered S3 method overwritten by 'R.oo':
##   method      from
##   throw.default R.methodsS3
```

```
library(tidyverse)
```

```
## -- Attaching packages -----
##   tidyverse 1.3.0 --
## v ggplot2 3.2.1     v purrr    0.3.3
## v tibble   2.1.3     v dplyr    0.8.3
## v tidyr    1.0.0     v stringr  1.4.0
## v readr    1.3.1     vforcats  0.4.0

## -- Conflicts -----
##   tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()
```

```
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
## The following object is masked from 'package:dplyr':
## 
##   group_rows
```

```
cik.no = 0001318605  # Tesla
form.type = '10-K'
filling.year = 2018
quarter = c(1,2,3,4)
```

```
# getFilings function takes CIKs, form type, filing year, and quarter of the filing
# as input. It creates
# new directory '~/Downloads/Edgar filings_full text' to store all downloaded filings.
```

```

getFilings(
  cik.no,
  form.type,
  filing.year,
  quarter,
  downl.permit="y")

## Downloading filings. Please wait...
## | | | 0% | =====
=====| 100%

##      cik company.name form.type date filed quarter filing.year
## 1 1318605 Tesla Inc 10-K 2018-02-23 1 2018
##      accession.number status
## 1 0001564590-18-002956 Download success

# getFilingsHTML function takes CIKs, form type, filing year, and quarter
# of the filing as input. The
# function imports edgar filings downloaded via getFilings function; otherwise,
# it downloads the filings which are not already been downloaded. It then reads
# the downloaded filing, scraps main body
# the filing, and save the filing content in '~/Downloads/Edgar filings_HTML view'
# directory in HTML format.

getFilingsHTML(
  cik.no = cik.no,
  form.type = form.type,
  filing.year = filing.year,
  quarter = quarter
)

## Downloading filings. Please wait...
## | | | 0% | =====
=====| 100%
## Scrapping full EDGAR and converting to HTML...
## |
## HTML filings are stored in 'Edgar filings_HTML view' directory.

##      cik company.name form.type date filed accession.number
## 1 1318605 Tesla Inc 10-K 2018-02-23 0001564590-18-002956

# This function creates a new directory '~/Downloads/Master Indexes' into
# current working directory to save these Rda Master Index.

getMasterIndex(filing.year)

## Downloading Master Indexes from SEC server for 2018 ...
## Master Index for quarter 1
## Master Index for quarter 2
## Master Index for quarter 3
## Master Index for quarter 4

# Management Discussion creates a new directory with name "~/Downloads/MD&A
# section text"

```

```

getMgmtDisc(
  cik.no = cik.no,
  filing.year = filing.year)

## Downloading fillings. Please wait...
## | | | 0% | ======|=====
## ======|===== 100%
## Extracting 'Item 7' section...
## | | | 0% | ======|=====
## ======|===== 100%
## MD&A section texts are stored in 'MD&A section text' directory.

##      cik company.name form.type date filed accession.number extract.status
## 1 1318605    TESLA INC       10-K 2018-02-23 0001564590-18-002956      1

# getSentiment function takes CIK(s), form type(s), and year(s) as input parameters.
# The function first
# imports available downloaded filings in local working directory 'Edgar filings'
# created by getFilings
# function; otherwise, it downloads the filings which is not already been downloaded.
# It then reads the
# filings, cleans the filings, and computes the sentiment measures. The function
# returns a dataframe
# with filing information, and sentiment measures.

sentiment_analysis <-
  getSentiment(
    cik.no,
    form.type,
    filing.year) %>%
  t()

## Downloading fillings. Please wait...
## |
## Computing sentiment measures...
## |

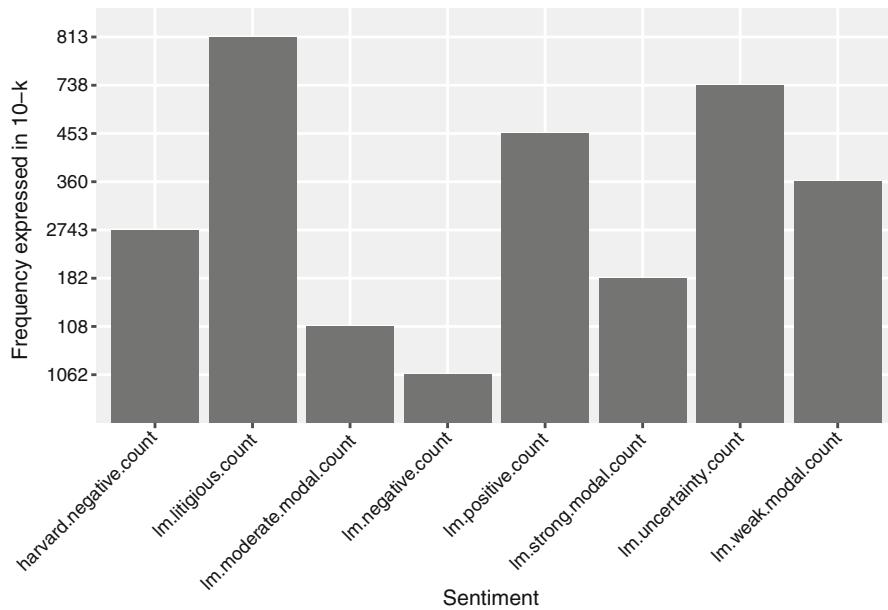
d <- sentiment_analysis
names <- rownames(d)
rownames(d) <- NULL
sentiment_analysis <- cbind(names,d)

colnames(sentiment_analysis) <- c("sentiment", "n")

sentiment_analysis <- as.data.frame(sentiment_analysis[10:nrow(sentiment_analysis),])

ggplot(sentiment_analysis, aes(sentiment, n)) +
  geom_col() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  xlab("Sentiment") +
  ylab("Frequency expressed in 10-k")

```



Accessing Financial Information from EDGAR (<https://www.sec.gov/edgar/>) with the finreportr Package

```
library(finreportr)

# The following commands will directly load EDGAR information into the R
# workspace for analysis

tesla_co <- CompanyInfo("TSLA")
tesla_ann <- AnnualReports("TSLA")
tesla_inc <- GetIncome("TSLA", 2018)
tesla_bs <- GetBalanceSheet("TSLA", 2018)
tesla_cf <- GetCashFlow("TSLA", 2018)

head(tesla_inc)
```

Metric	Units	Amount
Sales Revenue Goods Net	U_iso4217USD	3431587000
Sales Revenue Goods Net	U_iso4217USD	5589007000
Sales Revenue Goods Net	U_iso4217USD	8534752000
Operating Leases Income Statement Lease Revenue	U_iso4217USD	309386000
Operating Leases Income Statement Lease Revenue	U_iso4217USD	761759000
Operating Leases Income Statement Lease Revenue	U_iso4217USD	1106548000
startDate endDate		
1 2015-01-01 2015-12-31		
2 2016-01-01 2016-12-31		
3 2017-01-01 2017-12-31		
4 2015-01-01 2015-12-31		
5 2016-01-01 2016-12-31		
6 2017-01-01 2017-12-31		

The `finreportr` package returns a `data.frame` in “long form.” Because analysis typically benefits from datasets in “short form” with one row per account, `finreportr` `data.frames` need to be reshaped. Hadley Wickham has created a

comprehensive package called `reshape2` that uses metaphors of `melt` and `cast`. You `melt` data so that each row is a unique id-variable combination (i.e., is in “long form”) and then you `cast` the melted data into any shape you would like. There are specific commands for casting data.frames `dcast`, arrays `acast`, and so forth. In the next example, we take the Tesla income statement in ‘long form’ that we acquired with the `finreportr` package and `dcast` it into a more usable form.

```

library(tidyverse)
library(lubridate)

## 
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
## 
##     date

library(finreportr)
library(reshape2) # uses 'melt' and 'cast' to with between long and wide formats

## 
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
## 
##     smiths

tesla_inc <- GetIncome("TSLA", 2018) %>%
  rbind(GetIncome("TSLA", 2017)) %>%
  rbind(GetIncome("TSLA", 2016)) %>%
  rbind(GetIncome("TSLA", 2015))

head(tesla_inc)

##                                     Metric      Units    Amount
## 1                         Sales Revenue Goods Net U_iso4217USD 3431587000
## 2                         Sales Revenue Goods Net U_iso4217USD 5589007000
## 3                         Sales Revenue Goods Net U_iso4217USD 8534752000
## 4 Operating Leases Income Statement Lease Revenue U_iso4217USD 309386000
## 5 Operating Leases Income Statement Lease Revenue U_iso4217USD 761759000
## 6 Operating Leases Income Statement Lease Revenue U_iso4217USD 1106548000
##   startDate   endDate
## 1 2015-01-01 2015-12-31
## 2 2016-01-01 2016-12-31
## 3 2017-01-01 2017-12-31
## 4 2015-01-01 2015-12-31
## 5 2016-01-01 2016-12-31
## 6 2017-01-01 2017-12-31

tesla_inc <-tesla_inc %>%
  filter(month(startDate) == 01 & month(endDate) == 12) %>%
  mutate(Year = year(endDate)) %>%
  group_by(Metric, Year) %>%
  slice(1L) %>%
  dcast(Metric ~ Year, value.var = 'Amount')

head(tesla_inc)

```

```

##                               Metric      2012      2013      2014      2015
## 1 Cost Of Automotive Leasing <NA>       <NA>  87405000 183376000
## 2 Cost Of Goods Sold        371658000 1483321000 2058344000 2639926000
## 3 Cost Of Revenue          383189000 1557234000 2316685000 3122522000
## 4 Cost Of Revenues Automotive <NA>       <NA> 2145749000 2823302000
## 5 Cost Of Services           11531000 13356000 6674000 <NA>
## 6 Cost Of Services And Other <NA>       73913000 166931000 286933000
##                               2016      2017
## 1 481994000 708224000
## 2 4268087000 6724480000
## 3 5400875000 9536264000
## 4 4750081000 7432704000
## 5 <NA>       <NA>
## 6 472462000 1229022000

```

Computing Technical Metrics

In the prior section, I showed you how to acquire financial information from the SEC's repositories. This section provides general guidelines for computing technical metrics such as ratios from that statement data. Consider the calculation of the current ratio which is defined as:

$$\text{Current Ratio} = \frac{\text{Current Assets}}{\text{Current Liabilities}}$$

Here is how to calculate the ratio from our merged balance sheet obtained from EDGAR data using the `finstr` package.

```

library(devtools)
install_github("bergant/finstr")
library(finstr)
library(tidyverse)
library(kableExtra)
library(XBRL)

# Get EDGAR data in XBRL format from the sec.gov site
# parse XBRL (GM 10-K reports)

xbrl_url2016 <-
"https://www.sec.gov/Archives/edgar/data/1467858/000146785817000028/gm-20161231.xml"

xbrl_url2017 <-
"https://www.sec.gov/Archives/edgar/data/1467858/000146785818000022/gm-20171231.xml"

old_o <- options(stringsAsFactors = FALSE)
xbrl_data_2016 <- xbrlDoAll(xbrl_url2016)
xbrl_data_2017 <- xbrlDoAll(xbrl_url2017)
options(old_o)

st2017 <- xbdl_get_statements(xbdl_data_2017)
st2016 <- xbdl_get_statements(xbdl_data_2016)

balance_sheet2017 <- st2017$ConsolidatedBalanceSheets
balance_sheet2016 <- st2016$ConsolidatedBalanceSheets
balance_sheet <- merge(balance_sheet2017, balance_sheet2016)

## calculate current ratio

```

```
balance_sheet %>% transmute(
  date = endDate,
  CurrentRatio = AssetsCurrent / LiabilitiesCurrent
)

##           date CurrentRatio
## 1 2015-12-31    0.9745988
## 2 2016-12-31    0.8946009
## 3 2017-12-31    0.8940564
```

Other ratios may be calculated in a similar straightforward manner using the `dplyr` package in the `tidyverse` library. Note that several other packages (e.g., `edgar`, `finreportr`) are available to extract EDGAR filings. The `finstr` package can only process links to XBRL files, but `finreportr` can access data from both HTML and XBRL files.

Visualization of Technical Metrics

Visualizations are compact, yet can reveal patterns that would not be readily identified in the raw data. This is because the human brain is much more attuned to analyzing visual scenes than to analyzing lists of numbers and characters. Visualizing financial statements exposes a limited number of key values, and emphasizes their relationships and trends. In the following code chunk, I aggregate a balance sheet by selected concepts (Table 1).

```
library(htmlTable)
library(ggplot2)
library(kableExtra)

bs_simple <- expose(balance_sheet,

# Assets
`Current Assets` = "AssetsCurrent",
`Noncurrent Assets` = other("Assets"),
# Liabilities and equity
`Current Liabilities` = "LiabilitiesCurrent",
`Noncurrent Liabilities` = other(c("Liabilities", "CommitmentsAndContingencies")),
`Stockholders Equity` = "StockholdersEquity"
)

## Print the balance sheet;
## capture the output to a NULL file, and
## reformat with the kableExtra package

capture.output(bs_table <-
  print(
    bs_simple,
    html = FALSE,
    big.mark = ",",
    dateFormat = "%Y"),
    file='NUL')

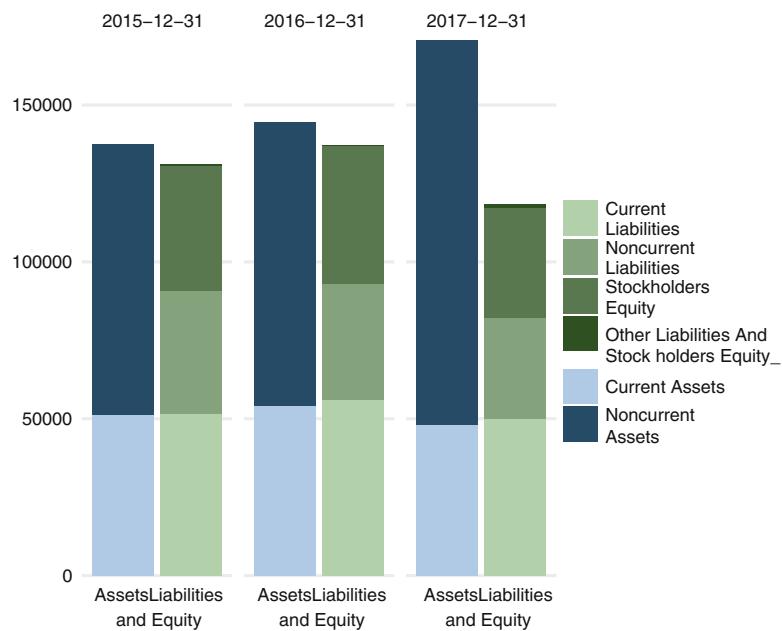
bs_table %>%
  kable(longtable=T,
    caption="Abbreviated Balance Sheet",
    "latex",
    booktabs = T) %>%
  kable_styling(bootstrap_options =
```

```
c("striped", "hover", "condensed"),
full_width = F, font_size=10)
```

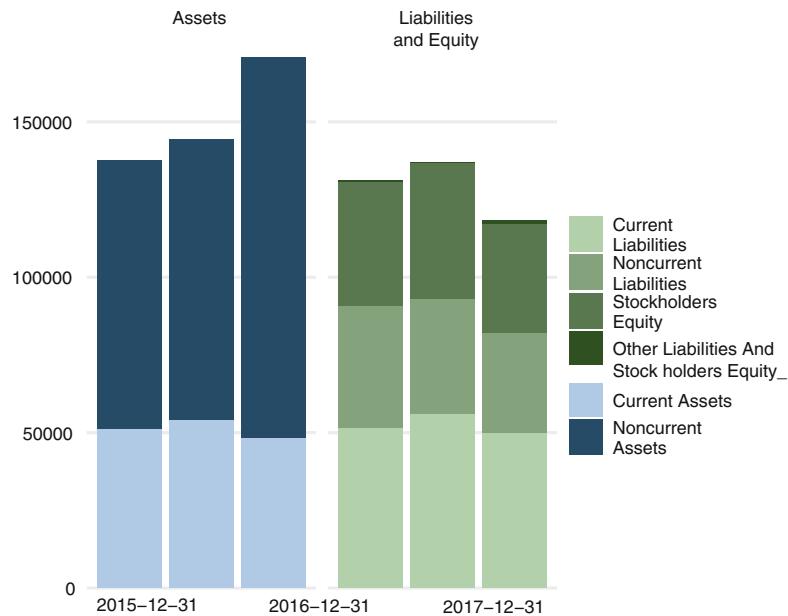
Table 1 Abbreviated balance sheet

Element	2017-12-31	2016-12-31	2015-12-31
Assets =	212,482	221,690	194,338
+ Current.Assets	48,223	54,138	51,357
+ Noncurrent.Assets	122,530	90,237	86,258
LiabilitiesAndStockholdersEquity =	212,482	221,690	194,338
+ Current.Liabilities	49,925	56,153	51,655
+ Noncurrent.Liabilities	32,138	36,834	39,249
+ Stockholders.Equity	35,001	43,836	39,871
+ OtherLiabilitiesAndStockholdersEquity_	1199	239	452

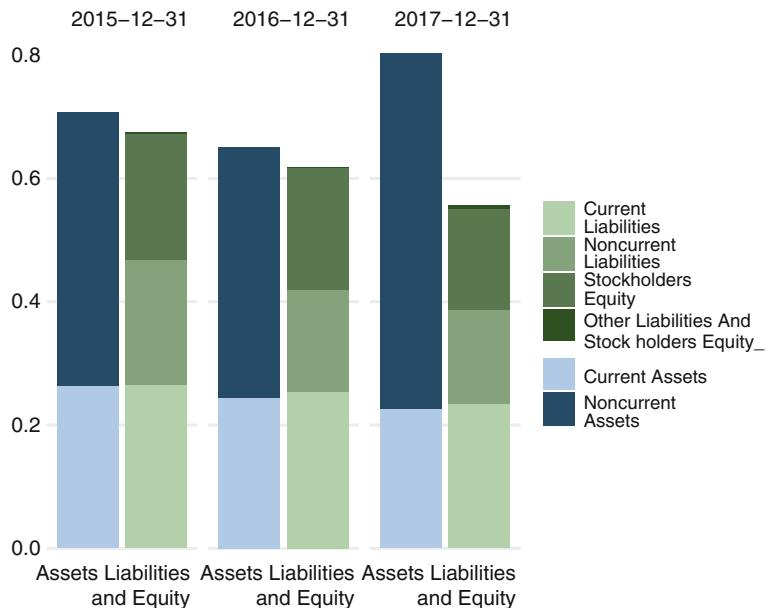
```
## plot the balance sheet
finstr::plot_double_stacked_bar(bs_simple)
```



```
## facet balance sheet DR and CR accounts
finstr::plot_double_stacked_bar(bs_simple, by_date = FALSE)
```



```
## use proportional form to highlight changes in balance sheet structure
bs_simple_prop <- proportional(bs_simple)
finstr::plot_double_stacked_bar(bs_simple_prop)
```



Internet Resources for Analytical Review

The exponential growth of web-accessible financial and firm data has revolutionized analytical review over the past decade. Intelligence scanning of Internet resources is the automated search for financial information, inventory sentiment, discussions, news, and other audit related information. It has eclipsed calculation of technical metrics as a source of internal

control weaknesses, accounting problems or fraud, and audit risk over the past decade. Today, the audit that does not fully take advantage of statistical software and Internet resources leaves itself open to negligence, misconduct, and lawsuits.

Arguably, the most important number on the financial statements is sales revenue. This reflects the audited firm's market strength in the face of competitors products, highlights product or service quality issues that reflect control weaknesses, and provides a general guide to other demand-side measures of firm health. I have focused the following analytical review algorithms on such demand-side measures of customer satisfaction, as these may be the auditors' single best source of qualitative intelligence on control weaknesses and accounting irregularities.

Sales revenues are fundamental drivers of business success—they reflect customers' impressions of the company's products, value proposition, and brand. The difference between a *product* and a *business* is the repeat sales. Customer loyalty to a brand is a major part of the valuation of many companies and spawns innovative "loyalty" programs to lock-in customers. Success or failure of these loyalty programs in subscription businesses (e.g., magazines, cable services, phone services) is measured by customer loyalty and brand recognition. Most organizations have data that can be used to target customers and to understand the key drivers of the revenue stream. Customers are the fuel that powers a business. Loss of customers impacts sales. Furthermore, it is much more difficult and costly to gain new customers than it is to retain existing customers. Customers' impressions of the firm and its products provide important, non-financial metrics for assessing the health of a firm. A firm with many dissatisfied customers is likely to suffer from weak management with high turnover, unreported control weaknesses, and to engage in "aggressive" accounting in reporting to lending institutions and shareholders.

The Internet has numerous resources—social networks, forums, product reviews, industry reports, and investment advice—that can alert the auditor to problems in a client's controls and financials. Internet resources play an essential role in analytical review, and in identifying audit risks that require increased diligence and scope. The remaining part of this chapter explores these Internet resources and how to incorporate them into the analytical review procedures.

US Census Data

Where location and demographics are important in a business, the US Census provides extensive and reliable data. R provides several packages to access and use those repositories. Data from the US Census Bureau is stored in tables, and to find the table for a particular metric you can use the function `acs.lookup` in the `acs` package. Note that to run this code you will need to get and install a census API key which you can request at https://api.census.gov/data/key_signup.html. Install the key with `api.key.install`.

The `acs.fetch` function is used to download data from the US Census American Community Survey. The `acs.lookup` function provides a convenience function to use in advance to locate tables and variables that may be of interest.

`acs.lookup` takes arguments similar to `acs.fetch`—in particular, "table.number," "table.name," and "keyword," as well as "endyear," "span," and "dataset"—and searches for matches in the meta-data of the Census tables. When multiple search terms are passed to a given argument (e.g., `keyword=c("Female", "GED")`), the tool returns matches where ALL of the terms are found; similarly, when more than one lookup argument is used (e.g., `table.number="B01001", keyword="Female"`), the tool searches for matches that include all of the terms (i.e., terms are combined with a logical "AND," not a logical "OR").

Results from `acs.lookup`—which are `acs.lookup` class objects—can then be inspected, subsetted (with [square brackets]), and combined (with `c` or `+`) to create custom `acs.lookup` objects to store and later pass to `acs.fetch` which has the following arguments:

- `endyear` is an integer indicating the latest year of the data in the survey (e.g., for data from the 2007–2011 5-year ACS data, `endyear` would be 2011).
- `span` is an integer indicating the span (in years) of the desired ACS data (should be 1, 3, or 5 for ACS datasets, and 0 for decennial census SF1 and SF3 datasets); defaults to 5, but ignored and reset to 0 if `dataset="sf1"` or `"sf3."`
- `geography` is a `geo.set` object specifying the census geography or geographies to be fetched; can be created "on the fly" with a call to `geo.make()`.
- `table.name` is a string giving the search term(s) to find in the name of the ACS census table (for example, "Sex" or "Age"); accepts multiple words, which must all be found in the returned table names; always case-sensitive. (Note: when set, this variable is passed to an internal call to `acs.lookup`—see `acs.lookup`).
- `table.number` is a string (not a number) indicating the table from the Census to fetch; examples: "B01003" or "B23013"; always case-sensitive. Used to fetch all variables for a given table number; if "table.number" is provided, other lookup variables ("table.name" or "keyword") will be ignored.

- variable is an object of `acs.lookup` class, or a string or vector of strings indicating the exact variable number to fetch. Non-`acs.lookup` examples include “B01003_001” or “B23013_003” or `c("B01003_001", "B23013_003")`.
- keyword is a string or vector of strings giving the search term(s) to find in the name of the census variable (for example, “Male” or “Haiti”); always case-sensitive.
- dataset is either “acs” (the default), “sf1,” or “sf3,” indicating whether to fetch data from in the American Community Survey or the SF1/SF3 datasets.

In the following example, we will compute the 2014–2019 female-to-male populations of the USA across age groups, and plot these as a bar graph.

```
library(acs)
library(tidyverse)

look <- acs.lookup(endyear = 2019,
                    keyword=c("Female"))
i_look <- look@results[1:24,c(1,4)] %>% t()
colnames(i_look) <- i_look[1,]

geo <- geo.make(state = "IL")
fet <- acs.fetch(endyear = 2014,
                  span = 5,
                  table.number="B01001",
                  keyword=c("Female"),
                  geography = geo)

fet_tbl <- fet@estimate
fet_tbl <- rbind(fet_tbl,i_look[2,]) %>% t()
colnames(fet_tbl) <- c("population", "age_group")

fet_tbl <-as.data.frame(fet_tbl)
# make age_group an ordered factor and convert population to numeric
fet_tbl$age_group <- factor(fet_tbl$age_group, levels = fet_tbl$age_group)
fet_tbl$population <- as.numeric(as.character(fet_tbl$population))
fet_fem <-as.data.frame(fet_tbl)

fet <- acs.fetch(endyear = 2014,
                  span = 5,
                  table.number="B01001",
                  keyword=c("Male"),
                  geography = geo)

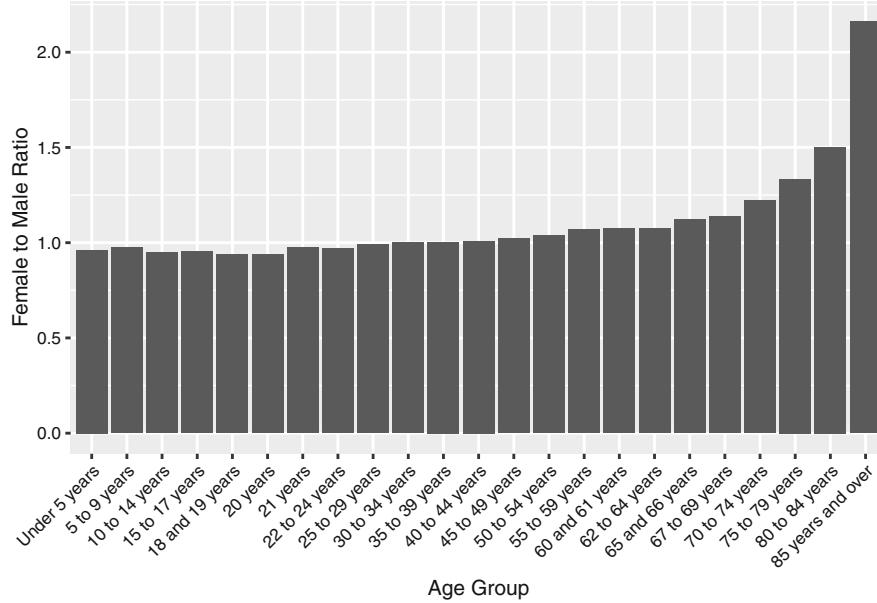
fet_tbl <- fet@estimate

fet_tbl <- rbind(fet_tbl,i_look[2,]) %>% t()
colnames(fet_tbl) <- c("population", "age_group")
fet_tbl <-as.data.frame(fet_tbl)
fet_tbl$age_group <- factor(fet_tbl$age_group, levels = fet_tbl$age_group)
fet_tbl$population <- as.numeric(as.character(fet_tbl$population))
fet_male <-as.data.frame(fet_tbl)

# Compute the ratio of females to males by U.S. county
fet_ratio <- inner_join(fet_fem,fet_male, by = "age_group")
fet_ratio$fem_to_male <- fet_ratio$population.x / fet_ratio$population.y
fet_ratio$age_group <- sub("Female:", "", fet_ratio$age_group)
```

```
fet_ratio$age_group <- factor(fet_ratio$age_group, levels = fet_ratio$age_group)

ggplot(fet_ratio[-1], aes(age_group, fem_to_male)) +
  geom_col() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  xlab("Age Group") +
  ylab("Female to Male Ratio")
```



The first table on the Census Bureau list is “B01001.” Look at `look_tbl = look@results` under `variable.code`, for example, the “_009” at the end indicates the column of the table; each column tabulates a different age range for males or females.

If your end result is to create choropleth maps, using the `choroplethr` package, it is more straightforward to use the function `get_acs_data` inside the `choroplethr` package instead of `acs.fetch` in the `acs` package.

```
library(choroplethr)
library(acs)
library(tidyverse)

# map = one of "state", "county" or "zip"
# column_idx = 15 of table B01001 is "Sex by Age: Male: 45 to 49 years"
# column_idx = 39 of table B01001 is "Sex by Age: Female: 45 to 49 years"

M45 <- get_acs_data(tableId = "B01001", map = "county", column_idx=15)
F45 <- get_acs_data(tableId = "B01001", map = "county", column_idx=39)
M45 <- M45[[1]]
head(M45)

##   region value
## 1    1001 2107
## 2    1003 6509
## 3    1005 1128
## 4    1007 1257
```

```

## 5    1009  2129
## 6    1011   386

F45 <- F45[[1]]
head(F45)

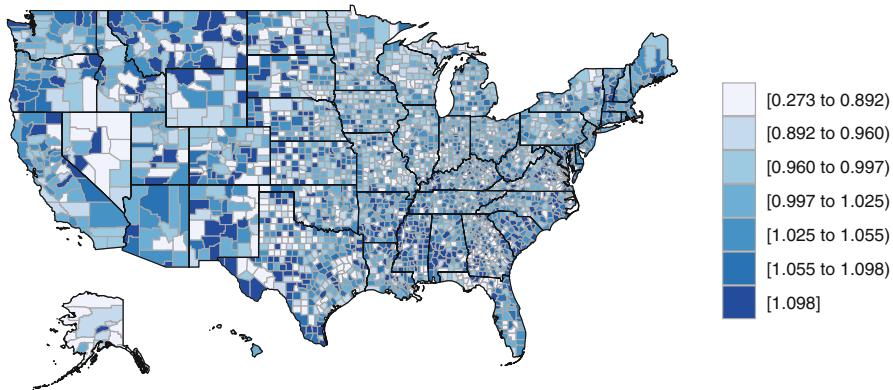
##   region value
## 1    1001 2095
## 2    1003 6848
## 3    1005  941
## 4    1007  848
## 5    1009 1985
## 6    1011  298

R45 <- inner_join(M45, F45, by = "region")
R45$value <- R45$value.y / R45$value.x

county_choropleth(R45[,c(1,4)], title = "Female / Male Ratio: 45 - 49 y.o. by County")

```

Female / Male Ratio: 45 – 49 y.o. by County



R and Application Programming Interfaces (API)

The R statistical language is particularly well-suited to *ad hoc* intelligence scanning, as it include many bespoke packages that support one-off analyses of Internet datasets and datastreams.

Application Programming Interfaces (API) provide access to Internet datastreams. Such data is not static, as users are constantly posting to datastreams and databases; rather these services think in terms of “streaming” dataflows that are constantly updated with changes to the database. Datastreams may be “throttled” by their owners, to prevent overwhelming systems servers with data requests. Three API protocols are used for most web data access:

1. JSON = JavaScript Object Notation is an open-standard file format that uses human-readable text to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value). It is a very common data format used for asynchronous browser–server communication, including as a replacement for XML in some AJAX-style systems.
2. REST = Representational State Transfer (REST) is a software architectural style that defines a set of constraints to be used for creating web services. Web services that conform to the REST architectural style, termed RESTful web services, provide interoperability between computer systems on the Internet. RESTful web services allow the requesting systems to access and manipulate textual representations of web resources by using a uniform and predefined set of stateless operations. Other kinds of web services, such as SOAP web services, expose their own arbitrary sets of operations.
3. SOAP = Simple Object Access Protocol is a messaging protocol specification for exchanging structured information in the implementation of web services in computer networks. Its purpose is to provide extensibility, neutrality, and independence.

It uses XML Information Set for its message format, and relies on application layer protocols, most often Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission.

In addition, access security for most Internet services is managed by OAuth = Open Authorization; pronounced “oh-auth.” OAuth is an open standard for token-based authentication and authorization on the Internet. It allows an end user’s account information to be used by third-party services without exposing the user’s password. OAuth was introduced in 2006 by Twitter, with the OAuth 1.0 protocol published in April 2010. The OAuth 2.0 framework was published in 2012 and is not backwards compatible with OAuth 1.0. OAuth 2.0 provides specific authorization flows for web applications, desktop applications, mobile phones, and smart devices. Facebook’s Graph API only supports OAuth 2.0; Google and Amazon support OAuth 2.0 as the recommended authorization mechanism for all of its APIs; Microsoft supports OAuth 2.0 for various APIs and its Azure Active Directory service, which is used to secure many Microsoft and third party APIs. Because of support from major firms, OAuth 2.0 has become the *de facto* standard for web services.

Technical Analysis of Product and Customer News Sources on the Web

Most text mining and natural language processing (NLP) modeling uses “bag of words” or “bag of n-grams” methods. Despite their simplicity, these models usually demonstrate good performance on text categorization and classification tasks. But in contrast to their theoretical simplicity and practical efficiency, building bag-of-words models involves technical challenges. This is especially the case in R because of its copy-on-modify semantics.

Let us briefly review some of the steps in a typical text analysis pipeline:

1. The auditor usually begins by constructing a document-term matrix (DTM) or term-co-occurrence matrix (TCM) from input documents. In other words, the first step is to vectorize text by creating a map from words or n-grams to a vector space.
2. The auditor fits a model to that DTM. These models might include text classification, topic modeling, similarity search, etc. Fitting the model will include tuning and validating the model.
3. Finally the auditor applies the model to new data.

Texts themselves can take up a lot of memory, but vectorized texts usually do not, because they are stored as sparse matrices. Because of R’s copy-on-modify semantics, it is not easy to iteratively grow a DTM; constructing a DTM, even for a small collections of documents, can be a serious bottleneck. It involves reading the whole collection of text documents into memory and processing it as single vector, which can easily increase memory use by a factor of two to four. The `text2vec` package solves this problem by providing a better way of constructing a document-term matrix.

As an example of NLP using the `text2vec` package, I will parse a dataset that comes with the package—the `movie_review` dataset. It consists of 5000 movie reviews, each of which is marked as positive or negative. First, split the dataset into two parts—train and test.

```
library(text2vec)
library(data.table)
library(magrittr)
data("movie_review")
setDT(movie_review)
setkey(movie_review, id)
set.seed(2017L)
all_ids = movie_review$id
train_ids = sample(all_ids, 4000)
test_ids = setdiff(all_ids, train_ids)
train = movie_review[J(train_ids)]
test = movie_review[J(test_ids)]

head(movie_review)

##           id sentiment
## 1: 10000_8          1
## 2: 10001_4          0
```

```

## 3: 10004_3      0
## 4: 10004_8      1
## 5: 10006_4      0
## 6: 10008_7      1
review
## 1: Homelessness (or Houselessness as George Carlin stated) has been an issue
for years but never a plan to help those on the street that were once considered
human who did everything from going to school, work, or vote for the matter. Most
people think of the homeless as just a lost cause while worrying about things
such as racism, the war on Iraq, pressuring kids to succeed, technology, the
elections, inflation, or worrying if they'll be next to end up on the streets.

```

```
class(movie_review)
```

```
## [1] "data.table" "data.frame"
```

Vocabulary-Based Vectorization

To represent documents in vector space, we first have to create “term” mappings. We call them terms instead of words because they can be, not just single words, but arbitrary n-grams—contiguous sequence of n items, where items can be phonemes, syllables, letters, words or base pairs. We represent a set of documents as a sparse matrix, where each row corresponds to a document and each column corresponds to a term. Create a vocabulary-based DTM by collecting unique terms from all documents and mark each of them with a unique ID using the `create_vocabulary()` function, using an iterator to create the vocabulary.

The following code chunk:

1. creates an iterator over tokens with the `itoken()` function. All functions prefixed with `create_` work with these iterators. R users might find this idiom unusual, but the iterator abstraction allows us to hide most of details about input and to process data in memory-friendly chunks.
2. builds the vocabulary with the `create_vocabulary()` function.

```
# define preprocessing function and tokenization function
prep_fun = tolower
tok_fun = word_tokenizer

it_train = itoken(train$review,
                  preprocessor = prep_fun,
                  tokenizer = tok_fun,
                  ids = train$id,
                  progressbar = FALSE)
vocab = create_vocabulary(it_train)
```

Alternatively, we could create list of tokens and reuse it in further steps. Each element of the list should represent a document, and each element should be a character vector of tokens.

```
library(magrittr)
library(tidyverse)
library(text2vec)

train_tokens = train$review %>%
  prep_fun %>%
  tok_fun
it_train = itoken(train_tokens,
                  ids = train$id,
```

```

# turn off progressbar because it won't look nice in rmd
progressbar = FALSE)

vocab = create_vocabulary(it_train)
vocab

## Number of docs: 4000
## 0 stopwords: ...
## ngram_min = 1; ngram_max = 1
## Vocabulary:
##             term term_count doc_count
## 1:         ufo      1       1
## 2:     bouchet      1       1
## 3:   atherton      1       1
## 4:     cyhyper      1       1
## 5: instalment      1       1
##   ---
## 38450:       to    21891     3796
## 38451:       of    23477     3794
## 38452:       a    26398     3880
## 38453:      and    26917     3868
## 38454:      the    53871     3970

vectorizer = vocab_vectorizer(vocab)
t1 = Sys.time()
dtm_train = create_dtm(it_train, vectorizer)
print(difftime(Sys.time(), t1, units = 'sec'))

## Time difference of 0.4296474 secs

dim(dtm_train)

## [1] 4000 38454

identical(rownames(dtm_train), train$id)

## [1] TRUE

```

Once we have a vocabulary, we can construct a document-term matrix.

```

vectorizer = vocab_vectorizer(vocab)
t1 = Sys.time()
dtm_train = create_dtm(it_train, vectorizer)
print(difftime(Sys.time(), t1, units = 'sec'))

## Time difference of 0.6223795 secs

```

At this point, we are ready to fit our model. Here we will use the `glmnet` (Lasso and Elastic-Net Regularized Generalized Linear Models) package to fit a logistic regression model with an L1 penalty (LASSO = least absolute shrinkage and selection operator) and fourfold cross-validation.

```

library(glmnet)
NFOLDS = 4
t1 = Sys.time()
glmnet_classifier =
  cv.glmnet(x = dtm_train,

```

```

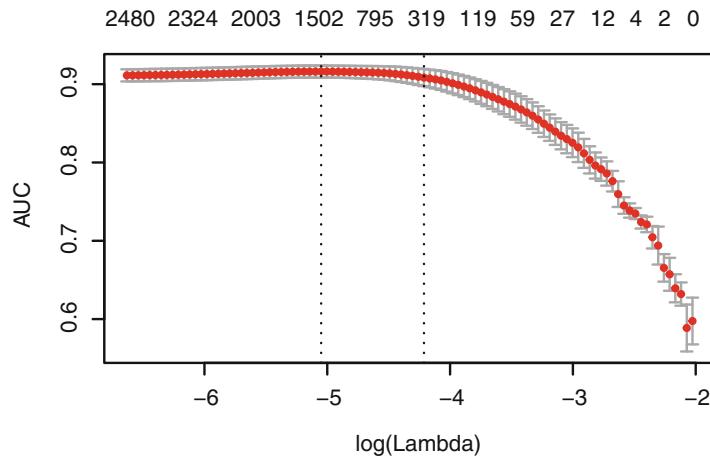
y = train[["sentiment"]],
      family = 'binomial',
      ## for the movie_reviews this is "binomial",
      ## for the food reviews it is 1-5, so "multinomial"

      # L1 penalty
      alpha = 1,
      # interested in the area under ROC curve
      type.measure = "auc",
      # 5-fold cross-validation
      nfolds = NFOLDS,
      # high value is less accurate, but has faster training
      thresh = 1e-3,
      # again lower number of iterations for faster training
      maxit = 1e3)
print(difftime(Sys.time(), t1, units = 'sec')))

## Time difference of 2.242048 secs

plot(glmnet_classifier)

```



```
print(paste("max AUC =", round(max(glmnet_classifier$cvm), 4)))
```

```
## [1] "max AUC = 0.9161"
```

We have successfully fit a model to our DTM. Now we can check the model's performance on test data. Note that we use exactly the same functions from prepossessing and tokenization. Also we reuse/use the same vectorizer—function which maps terms to indices.

```

# Note that most text2vec functions are pipe friendly!
it_test = test$review %>%
  prep_fun %>% tok_fun %>%
  # turn off progressbar because it won't look nice in rmd
  itoken(ids = test$id, progressbar = FALSE)

dtm_test = create_dtm(it_test, vectorizer)

```

```
preds = predict(glmnet_classifier, dtm_test, type = 'response')[,1]
glmnet:::auc(test$sentiment, preds)
```

```
## [1] 0.9151781
```

The result shows that performance on the test data is roughly the same as we expected from cross-validation. Note though that the training time for the model was high. We can reduce it and also significantly improve accuracy by “pruning” the vocabulary. For example, we can find words “a,” “the,” “in,” “I,” “you,” “on,” etc. in almost all documents, but they do not provide much useful information. Usually such words are called “stop words.” On the other hand, the corpus also contains very uncommon terms, which are contained in only a few documents. These terms are also useless, because we do not have sufficient statistics for them. Here we will remove predefined stopwords, very common and very unusual terms.

```
stop_words = c("i",
              "me",
              "my",
              "myself",
              "we",
              "our",
              "ours",
              "ourselves",
              "you",
              "your",
              "yours")
```



```
t1 = Sys.time()
vocab = create_vocabulary(it_train, stopwords = stop_words)
print(difftime(Sys.time(), t1, units = 'sec'))
```

```
## Time difference of 0.2090459 secs
```

```
pruned_vocab = prune_vocabulary(vocab,
                                  term_count_min = 10,
                                  doc_proportion_max = 0.5,
                                  doc_proportion_min = 0.001)
vectorizer = vocab_vectorizer(pruned_vocab)
```

```
# create dtm_train with new pruned vocabulary vectorizer
```

```
t1 = Sys.time()
dtm_train = create_dtm(it_train, vectorizer)
print(difftime(Sys.time(), t1, units = 'sec'))
```

```
## Time difference of 0.209995 secs
```

```
dim(dtm_train)
```

```
## [1] 4000 6542
```

```
# create DTM for test data with the same vectorizer:
```

```
dtm_test = create_dtm(it_test, vectorizer)
dim(dtm_test)
```

```
## [1] 1000 6542
```

This model can be improved by using n-grams instead of words—in the following code chunk, I use up to 2-grams:

```
t1 = Sys.time()
vocab = create_vocabulary(it_train, ngram = c(1L, 2L))
print(difftime(Sys.time(), t1, units = 'sec'))

## Time difference of 0.697365 secs

vocab = prune_vocabulary(vocab, term_count_min = 10,
                         doc_proportion_max = 0.5)

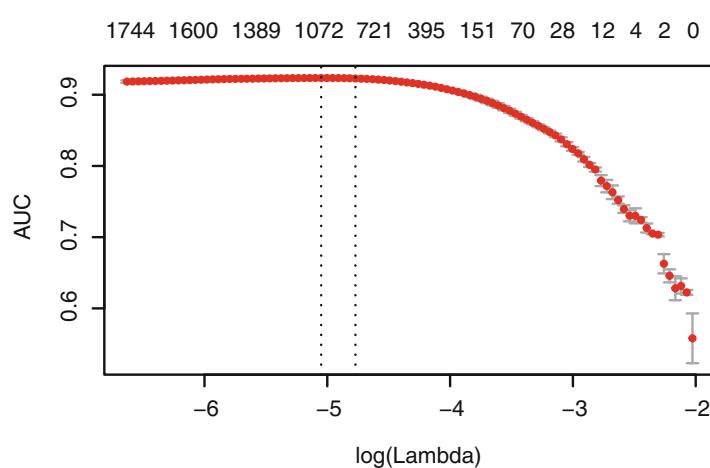
bigram_vectorizer = vocab_vectorizer(vocab)

dtm_train = create_dtm(it_train, bigram_vectorizer)

t1 = Sys.time()
glmnet_classifier = cv.glmnet(x = dtm_train, y = train[['sentiment']],
                               family = 'binomial',
                               alpha = 1,
                               type.measure = "auc",
                               nfolds = NFOLDS,
                               thresh = 1e-3,
                               maxit = 1e3)
print(difftime(Sys.time(), t1, units = 'sec'))

## Time difference of 1.626327 secs

plot(glmnet_classifier)
```



```
print(paste("max AUC =", round(max(glmnet_classifier$cvm), 4)))

## [1] "max AUC = 0.9235"

# apply vectorizer
dtm_test = create_dtm(it_test, bigram_vectorizer)

preds =
  predict(glmnet_classifier,
```

```

  dtm_test,
  type = 'response') [,1]

glmnet:::auc(test$sentiment, preds)

## [1] 0.9294646

```

To further improve performance, we can use “feature hashing” which achieves greater speed by avoiding a lookup over an associative array. Another benefit is that it leads to a very low memory footprint, since we can map an arbitrary number of features into much more compact space, using `text2vec`. The method often makes AUC slightly worse in exchange for improved execution times, which on large collections of documents can provide a significant advantage.

```

h_vectorizer = hash_vectorizer(hash_size = 2 ^ 14, ngram = c(1L, 2L))

t1 = Sys.time()
dtm_train = create_dtm(it_train, h_vectorizer)
print(difftime(Sys.time(), t1, units = 'sec'))

```

Time difference of 0.6586969 secs

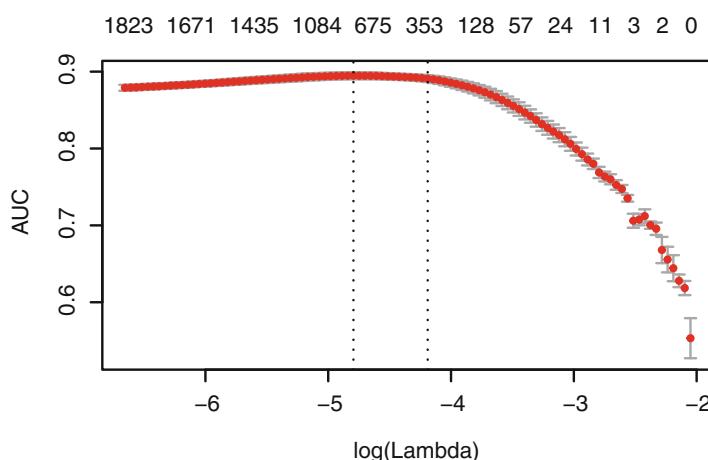
```

t1 = Sys.time()
glmnet_classifier = cv.glmnet(x = dtm_train, y = train[['sentiment']],
                               family = 'binomial',
                               alpha = 1,
                               type.measure = "auc",
                               nfolds = 5,
                               thresh = 1e-3,
                               maxit = 1e3)
print(difftime(Sys.time(), t1, units = 'sec'))

```

Time difference of 2.570303 secs

```
plot(glmnet_classifier)
```



```
print(paste("max AUC =", round(max(glmnet_classifier$cvm), 4)))

```

[1] "max AUC = 0.8947"

```

dtm_test = create_dtm(it_test, h_vectorizer)

preds =
  predict(glmnet_classifier,
    dtm_test ,
    type = 'response') [, 1]

glmnet:::auc(test$sentiment, preds)

## [1] 0.898901

```

Before analysis it can be useful to transform DTM, using the `normalize` function, since lengths of the documents in collection can significantly vary. Normalization transforms the rows of DTM so we adjust values measured on different scales to a not to transform rows so that the sum of the row values equals 1. There is also a `Tfidf` function which not only normalizes DTM, but also increase the weight of terms which are specific to a single document or handful of documents and decrease the weight for terms used in most documents. Both can further improve execution time and accuracy, which may be of value in very large datasets.

References

- Ameen, Elsie C., and Jerry R. Strawser. 1994. Investigating the Use of Analytical Procedures: An Update and Extension. *Auditing* 13 (2): 69. American Accounting Association.
- Anderson, John C., Marianne M. Jennings, Steven E. Kaplan, and Philip M.J. Reckers. 1995. The Effect of Using Diagnostic Decision Aids for Analytical Procedures on Judges' Liability Judgments. *Journal of Accounting and Public Policy* 14 (1): 33–62. Elsevier.
- Anderson, Urton L., Lisa Koonce, and Garry Marchant. 1994. The Effects of Source-Competence Information and Its Timing on Auditors' Performance of Analytical Procedures. *Auditing* 13 (1): 137. American Accounting Association.
- Asare, Stephan K., and A. Wright. 1997. Hypothesis Revision Strategies in Conducting Analytical Procedures. *Accounting, Organizations and Society* 22 (8): 737–755. Elsevier.
- Asare, Stephen K., and Arnold M. Wright. 1997. Evaluation of Competing Hypotheses in Auditing. *Auditing* 16 (1): 1. American Accounting Association.
- Biggs, Stanley F., Theodore J. Mock, and Paul R. Watkins. 1988. Auditor's Use of Analytical Review in Audit Program Design. *The Accounting Review* 63 (1): 148–161. JSTOR.
- Coakley, James R. 1995. Using Pattern Analysis Methods to Supplement Attention Directing Analytical Procedures. *Expert Systems with Applications* 9 (4): 513–528. Elsevier.
- Coakley, James R., and Carol E. Brown. 1993. Artificial Neural Networks Applied to Ratio Analysis in the Analytical Review Process. *Intelligent Systems in Accounting, Finance and Management* 2 (1): 19–39. Wiley Online Library.
- Kinney, William R. 1979. The Predictive Power of Limited Information in Preliminary Analytical Review: An Empirical Study. *Journal of Accounting Research* 17: 148–165. JSTOR.
- Kinney Jr, William R., and Wilfred C. Uecker. 1982. Mitigating the Consequences of Anchoring in Auditor Judgments. *The Accounting Review* 57 (1): 55–69. JSTOR.
- Knechel, W. Robert. 1988. The Effectiveness of Statistical Analytical Review as a Substantive Auditing Procedure: A Simulation Analysis. *The Accounting Review* 63 (1): 74–95. JSTOR.
- Kogan, Alexander, Michael G. Alles, Miklos A. Vasarhelyi, and Jia Wu. 2010. Analytical Procedures for Continuous Data Level Auditing: Continuity Equations 1. Citeseer.
- Koskivaara, Eija. 2004a. Artificial Neural Networks for Analytical Review in Auditing. fi= Turun yliopiston en= University of Turkul.
- _____. 2004b. Visualization of Patterns in Accounting Data with Self-Organizing Maps. In *Business Intelligence Techniques*, 133–147. Berlin: Springer.
- _____. 2006. Integrating Analytical Procedures into the Continuous Audit Environment. *JISTEM-Journal of Information Systems and Technology Management* 3 (3): 331–346. SciELO Brasil.
- Koskivaara, Eija, and Barbro Back. 2007. Artificial Neural Network Assistant (Anna) for Continuous Auditing and Monitoring of Financial Data. *Journal of Emerging Technologies in Accounting* 4 (1): 29–45.
- Libby, Robert. 1985. Availability and the Generation of Hypotheses in Analytical Review. *Journal of Accounting Research* 23 (2): 648–667. JSTOR.
- Loebbecke, James K., and Paul J. Steinbart. 1987. An Investigation of the Use of Preliminary Analytical Review to Provide Substantive Audit Evidence. *Auditing-A Journal of Practice & Theory* 6 (2): 74–89. Amer Accounting Assoc.
- Mueller, Jennifer M., and John C. Anderson. 2002. Decision Aids for Generating Analytical Review Alternatives: The Impact of Goal Framing and Audit-Risk Level. *Behavioral Research in Accounting* 14 (1): 157–177.
- Nelson, Mark W. 1994. The Learning and Application of Frequency Knowledge in Audit Judgment. *Journal of Accounting Literature* 13: 185. Elsevier BV.
- Stringer, Kenneth W. 1975. A Statistical Technique for Analytical Review. *Journal of Accounting Research* 13: 1–9. JSTOR.
- Wilson, Arlette C., and Janet Colbert. 1989. An Analysis of Simple and Rigorous Decision Models as Analytical Procedures. *Accounting Horizons* 3 (4): 79. American Accounting Association.

Analytical Review: Intelligence Scanning



Intelligence Scanning of Internet Resources

Analytical procedures involve the study of plausible relationships between both financial and non-financial data. Before the mid-2000s, sources of such information were spotty, unreliable, and scarce, and that, indeed, was considered a prime reason for markets needing annual, audited financial statements. Today, there are numerous social networks, news, and discussion boards that offer both raw and curated, streaming sources of useful business intelligence. The auditor who does not scan for such client and industry intelligence both increases the cost of auditing, and can be considered negligent in not investigating all available information about the financial health of their client.

Though this information is available, the raw information itself may need considerable interpretation and processing before an auditor may rely on it for audit work. The most useful information for analytical review may be qualitative and textual, so unsuitable for the sort of mathematical analysis offered in spreadsheet packages and statistical programs. Qualitative data may be most useful when it is used to gauge sentiment of customers, investors, partners, employees, and vendors toward the company. It is only in the past decade that tools have become available to glean such intelligence from Internet streams and databases. This chapter provides a set of tools that will assist the auditor in interpreting and understanding the relevant qualitative data accessed through the Internet. I provide algorithms to map qualitative audit intelligence into consumer, investor, and vendor sentiment. I also suggest useful Internet resources for the audit, and provide algorithms for interpreting the scanned intelligence concerning the audit client, its competitors and the industry.

Sentiment Analysis with Tidy Data

When human readers approach a text, we use our understanding of the emotional intent of words to infer whether a section of text is positive or negative, or perhaps characterized by some other more nuanced emotion like surprise or disgust. We can use the tools of text mining to approach the emotional content of text programmatically. We start by representing text in R's "tidy" structure:

- Each variable is a column.
- Each observation is a row.
- Each type of observational unit is a table.

We thus define the tidy text format as being a table with one-token-per-row. A token is a meaningful unit of text, such as a word, that we are interested in using for analysis, and tokenization is the process of splitting text into tokens. This one-token-per-row structure is in contrast to the ways text is often stored in current analyses, perhaps as strings or in a document-term matrix. For tidy text mining, the token that is stored in each row is most often a single word, but can also be an n-gram, sentence, or paragraph. R's `tidytext` package provides functionality to tokenize by commonly used units of text like these and convert to a one-term-per-row format.

One way to analyze the sentiment of a text is to consider the text as a combination of its individual words and the sentiment content of the whole text as the sum of the sentiment content of the individual words. This is not the only way to approach sentiment analysis, but it is an often-used approach, and an approach that naturally takes advantage of the tidy tool ecosystem.

The tidytext package contains several sentiment lexicons in the sentiments dataset. For example, consider the following code chunk.

```
library(tidytext)
sentiments
```

```
## # A tibble: 6,786 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 2-faces    negative
## 2 abnormal    negative
## 3 abolish     negative
## 4 abominable  negative
## 5 abominably  negative
## 6 abominate   negative
## 7 abomination negative
## 8 abort       negative
## 9 aborted     negative
## 10 aborts     negative
## # ... with 6,776 more rows
```

The three general-purpose lexicons are

- bing from Bing Liu and collaborators at University of Illinois—Chicago,
- AFINN from Finn Arup Nielsen, and
- nrc from Saif Mohammad and Peter Turney.

All three of these lexicons are based on unigrams, i.e., single words. These lexicons contain many English words and the words are assigned scores for positive/negative sentiment, and also possibly emotions like joy, anger, sadness, and so forth. All three were constructed via either crowdsourcing (using, for example, Amazon Mechanical Turk) or by the labor of one of the authors, and were validated using some combination of crowdsourcing again, restaurant or movie reviews, or Twitter data.

The nrc lexicon categorizes words in a binary fashion (“yes”/“no”) into categories of positive, negative, anger, anticipation, disgust, fear, joy, sadness, surprise, and trust.

The bing lexicon categorizes words in a binary fashion into positive and negative categories.

The AFINN lexicon assigns words with a score that runs between -5 and 5 , with negative scores indicating negative sentiment and positive scores indicating positive sentiment.

All of this information is tabulated in the sentiments dataset, and tidytext provides a function `get_sentiments()` to get specific sentiment lexicons without the columns that are not used in that lexicon.

```
get_sentiments("afinn")
```

```
## # A tibble: 2,477 x 2
##   word      value
##   <chr>     <dbl>
## 1 abandon    -2
## 2 abandoned  -2
## 3 abandons   -2
## 4 abducted   -2
## 5 abduction  -2
## 6 abductions -2
## 7 abhor      -3
## 8 abhorred   -3
## 9 abhorrent  -3
## 10 abhors    -3
## # ... with 2,467 more rows
```

```
get_sentiments("bing")
```

```
## # A tibble: 6,786 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 2-faces  negative
## 2 abnormal negative
## 3 abolish  negative
## 4 abominable negative
## 5 abominably negative
## 6 abominante negative
## 7 abomination negative
## 8 abort    negative
## 9 aborted  negative
## 10 aborts  negative
## # ... with 6,776 more rows
```

```
get_sentiments("nrc")
```

```
## # A tibble: 13,901 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 abacus   trust
## 2 abandon   fear
## 3 abandon   negative
## 4 abandon   sadness
## 5 abandoned anger
## 6 abandoned fear
## 7 abandoned negative
## 8 abandoned sadness
## 9 abandonment anger
## 10 abandonment fear
## # ... with 13,891 more rows
```

There are also some domain-specific sentiment lexicons available, constructed to be used with text from a specific content area—e.g., for accounting and finance. Dictionary-based methods like the ones we are discussing find the total sentiment of a piece of text by adding up the individual sentiment scores for each word in the text. Not every English word is in the lexicons because many English words are pretty neutral. It is important to keep in mind that these methods do not take into account qualifiers before a word, such as in “no good” or “not true”; a lexicon-based method like this is based on unigrams only.

One last caveat is that the size of the chunk of text that we use to add up unigram sentiment scores can have an effect on results. A text the size of many paragraphs can often have positive and negative sentiment averaged out to about zero, while sentence-sized or paragraph-sized text often works better.

With data in a tidy format, sentiment analysis can be done as an inner join. This is another of the great successes of viewing text mining as a tidy data analysis task; much as removing stop words is an antijoin operation, performing sentiment analysis is an inner join operation. Let us look at the words with a joy score from the NRC lexicon. For this example, we capture an HTML formatted General Motors’ 10-K report for 2017 from SEC’s ECGAR database, demonstrating that HTML documents may be used in the workpapers, as well as those in XBRL format.

```
library(htm2txt)
library(kableExtra)
library(tokenizers)
library(wordcloud)
library(tidyverse)
library(tidytext)
```

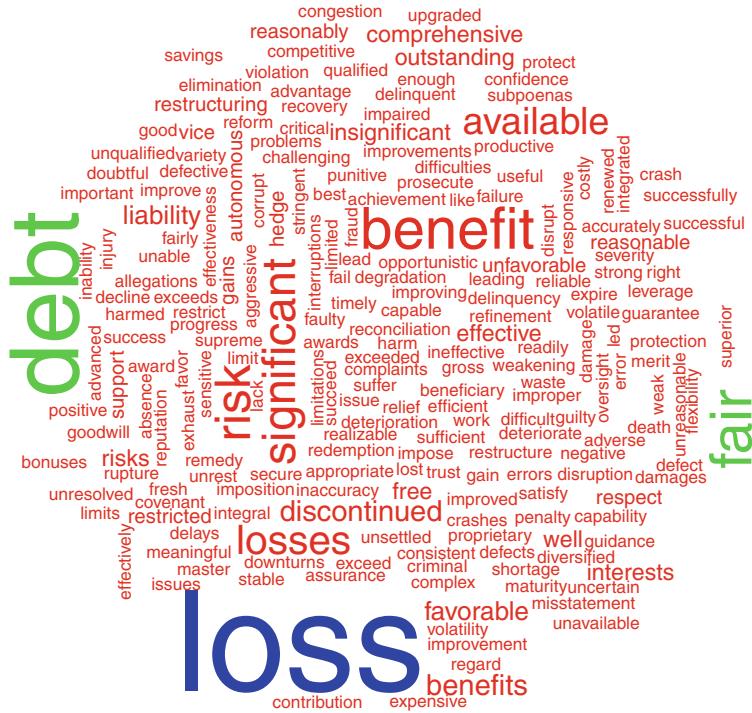
```
txt <- gettxt("https://www.sec.gov/Archives/edgar/data/1467858/000146785818000022/gm201710k.htm",
  encoding = "UTF-8")

text_stuff <- htmtxt(txt) %>% tokenize_words() %>% unlist() %>% as.data.frame()

colnames(text_stuff) <- "word"

stuff_sentiment <- text_stuff %>% inner_join(get_sentiments("bing"), by = "word")

text_stuff %>% anti_join(get_stopwords()) %>% inner_join(stuff_sentiment) %>% count(word) %>%
  with(wordcloud(word, colors = rainbow(3), rot.per = 0.15, n, max.words = 1000))
```



```
net_sentiment <- text_stuff %>% inner_join(get_sentiments("bing"), by = "word") %>%
  count(sentiment) %>% spread(sentiment, n, fill = 0) %>% mutate(net_positive = positive -
  negative, proportion_positive = positive/negative - 1)

net_sentiment %>% kable(longtable = T, "latex", booktabs = T) %>% kable_styling(bootstrap_options = c("striped",
  "hover", "condensed"), full_width = F, font_size = 10)
```

Negative	Positive	Net_positive	Proportion_positive
1138	1316	178	0.1564148

The size of a word's text here is in proportion to its frequency within its sentiment. We can use this visualization to see the most important positive and negative words, but the sizes of the words are not comparable across sentiments.

In other functions, such as `comparison.cloud()`, you may need to turn the data frame into a matrix with `reshape2`'s `acast()`. Let us do the sentiment analysis to tag positive and negative words using an inner join, then find the most common positive and negative words. Until the step where we need to send the data to `comparison.cloud()`, this can all be done with joins, piping, and `dplyr` because our data is in tidy format.

```
library(reshape2)
library(wordcloud)

stuff_sentiment %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                    max.words = 100)
```



With several options for sentiment lexicons, you might want some more information on which one is appropriate for your purposes. Let us use all three sentiment lexicons and examine how the sentiment changes. I use `inner_join()` to calculate the sentiment in different ways. The AFINN lexicon measures sentiment with a numeric score between -5 and 5 , while the other two lexicons categorize words in a binary fashion, either positive or negative. Use integer division (`%/%`) to define larger sections of text that span multiple lines, and we can use the same pattern with the `tidyverse` library's pipe (`%>%`), `count()`, `spread()`, and `mutate()` to find sentiment.

```
library(tidyverse)
library(Hmisc)

cat("\n\n afinn")

## 
## 
##  afinn

get_sentiments("afinn") %>%
  describe()
```

```

## .
## 
## 2 Variables      2477 Observations
## -----
## word
##     n missing distinct
##    2477      0      2477
## 
## lowest : abandon abandoned abandons abducted abduction
## highest: yucky yummy zealot zealots zealous
## -----
## value
##     n missing distinct      Info      Mean      Gmd      .05      .10
##    2477      0      11    0.931   -0.5894    2.303      -3      -3
##    .25      .50      .75      .90      .95
##    -2      -2       2       2       3
## 
## Value      -5      -4      -3      -2      -1       0       1       2       3       4       5
## Frequency  16      43     264    966    309      1     208    448    172     45      5
## Proportion 0.006  0.017  0.107  0.390  0.125  0.000  0.084  0.181  0.069  0.018  0.002
## -----

```

```
cat("\n\n nrc")
```

```

## 
## 
## nrc

## 
```

```

get_sentiments("nrc") %>%
  filter(sentiment %in% c("positive",
                         "negative")) %>%
  describe()

```

```

## .
## 
## 2 Variables      5636 Observations
## -----
## word
##     n missing distinct
##    5636      0      5555
## 
## lowest : abandon abandoned abandonment abba      abduction
## highest: youth   zeal   zealous   zest      zip
## -----
## sentiment
##     n missing distinct
##    5636      0       2
## 
## Value      negative positive
## Frequency  3324      2312
## Proportion 0.59      0.41
## -----
```

```
cat("\n\n bing")
```

```

## 
## 
## bing
```

```
get_sentiments("bing") %>%
  filter(sentiment %in% c("positive",
                           "negative")) %>%
  describe()

## .
## 
## # 2 Variables      6786 Observations
## -----
## word
##     n missing distinct
##   6786      0    6783
## 
## lowest : 2-faces    abnormal    abolish    abominable abominably
## highest: zealously zenith      zest       zippy      zombie
## -----
## sentiment
##     n missing distinct
##   6786      0        2
## 
## Value      negative positive
## Frequency    4781      2005
## Proportion   0.705     0.295
## -----
```

Both bing and NEC lexicons have more negative than positive words, but the ratio of negative to positive words is higher in the bing lexicon than the NRC lexicon. This will contribute to the effect we see in the plot above, as will any systematic difference in word matches.

One advantage of having the data frame with both sentiment and word is that we can analyze word counts that contribute to each sentiment. By implementing count() here with arguments of both word and sentiment, we find out how much each word contributed to each sentiment.

```
bing_word_counts <- stuff_sentiment %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

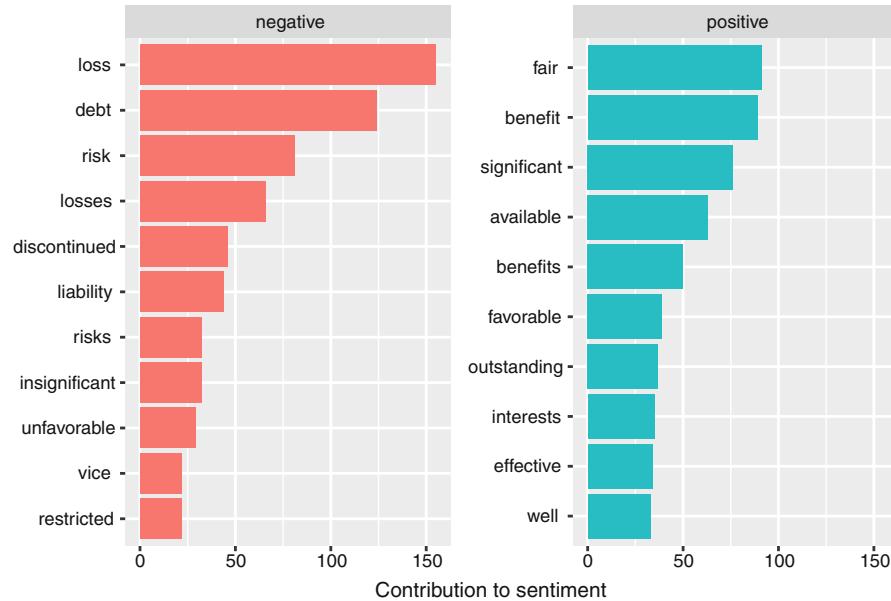
head(bing_word_counts)

## # A tibble: 6 x 3
##   word      sentiment     n
##   <chr>      <chr>     <int>
## 1 loss      negative    155
## 2 debt      negative    124
## 3 fair      positive     91
## 4 benefit   positive     89
## 5 risk      negative     81
## 6 significant positive   76
```

This can be shown visually, and we can pipe (%>%) straight into ggplot2, if we like, because of the way we are consistently using tools built for handling tidy data frames.

```
bing_word_counts %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
```

```
geom_col(show.legend = FALSE) +
facet_wrap(~sentiment, scales = "free_y") +
labs(y = "Contribution to sentiment",
x = NULL) +
coord_flip()
```



This lets us spot an anomaly in the sentiment analysis; the words “loss,” “debt,” and “fair” are generic words used to describe many accounting situations and accounts. If it were appropriate for our purposes, we could easily add “loss” to a custom stop-words list using `bind_rows()`. We could implement that with a strategy such as this.

```
library(tidytext)
library(tidyverse)
library(wordcloud)

custom_stop_words <-
  bind_rows(tibble(word = c("loss", "debt", "fair"),
  lexicon = c("custom")),
  get_stopwords())

head(custom_stop_words)

## # A tibble: 6 x 2
##   word  lexicon
##   <chr> <chr>
## 1 loss   custom
## 2 debt   custom
## 3 fair   custom
## 4 i      snowball
## 5 me     snowball
## 6 my     snowball

## here is our prior wordcloud with custom_stop_words

text_stuff %>%
  anti_join(custom_stop_words) %>%
  inner_join(stuff_sentiment) %>%
```

```
count(word) %>%
  with(wordcloud(word,
                 colors = rainbow(3),
                 rot.per = 0.15, n,
                 max.words = 1000)
  )
```



Much useful work can be done by tokenizing at the word level, but sometimes it is useful or necessary to look at different units of text. For example, some sentiment analysis algorithms look beyond only unigrams (i.e., single words) to try to understand the sentiment of a sentence as a whole. These algorithms try to understand that “I am not having a good day” is a sad sentence, not a happy one, because of negation. R packages, `coreNLP`, `cleanNLP`, and `sentimentr` are examples of such sentiment analysis algorithms. For these, we may want to tokenize text into sentences.

Scanning of Uncurated News Sources from Social Networks

Twitter is arguably the most important of all the social networks for market analysis. Facebook, MySpace, Instagram, Snapchat, LinkedIn, WeChat, and so forth potentially could offer more useful marketing information, but regulations either prohibit the distribution of data completely (e.g., LinkedIn) or they significantly limit it to just the people you may know directly (Facebook). This section shows you how to gain access to Twitter's API (application programming interface).

Twitter has around 130 million daily active users, a number that vastly dwarfs any of the finance specific social platforms (e.g., StockTwits) which have usage measured in hundreds of thousands. Even though it is not finance specific, the volume of financial information exchanged on Twitter is substantially larger than on bespoke finance platforms. In addition, Twitter is more likely to report the market, competitor and product specific problems that tend to drive valuations today. I use the `rtweet` package here to extract information from Twitter.

The “#” symbol is used to refer to individuals on Twitter. A # symbol before a word tells Twitter to “index” all transactions that have this hashtag. Twitter accumulates all posts that have it and retrieves them more quickly than a search with a query engine would.

The “@” symbol is used to refer to individuals on Twitter. It is combined with a username and inserted into tweets to refer to that person or send them a public message. When @ precedes a username, it automatically gets linked to that user’s profile page. Users may be individuals or firms.

It is not necessary to receive a token to use `rtweet`; only a user account is required. But the OAuth authentication gives access to more functions on Twitter, such as posting. If you need full access, go to <https://developer.twitter.com> (Twitter Developer Site) create an app and apply for a consumer key and consumer secret; as the “Callback URL” enter: <http://127.0.0.1:1410>.

The authentication process of Twitter involves creating a *Twitter app* and doing a *handshake*. You have to *handshake* every time you want to get data from Twitter with R. Since Twitter released the Version 1.1 of their API an OAuth handshake is necessary for every request you do. The following steps will get you onto the Twitter API:

1. Create an *app* at Twitter: Go to apps.twitter.com/ and log in with your Twitter Account. From your Profile picture in the upper right corner select the drop-down menu and look for “*My Applications*”. Click on it and then on “*Create new Application*.” As the Callback URL enter:^{*} <http://127.0.0.1:1410>. *Click on Create** you will get redirected to a screen with all the OAuth setting of your new App.
2. Use the `setup_twitter_OAuth()` function which uses the `httr` package. Get your `api_key` and your `api_secret` as well as your `access_token` and `access_token_secret` from your app settings on Twitter (click on the “*API key*” tab to see them). Here is an example.

```
## install devtools package if it's not already
if (!requireNamespace("devtools", quietly = TRUE)) {
  install.packages("devtools")
}

## install dev version of rtweet from github
devtools::install_github("mkearney/rtweet")
install.packages("maps")
## load rtweet package
library(rtweet)
library(maps)

## access token method: create token and save it as an environment variable
create_token()
  app = "Test_of_the_API_platform",
  consumer_key = 'AWsZc3pjFsgAF1BK4OHRlyGtK',
  consumer_secret = 'DTRvorcjSaQQ1goWzynZ2tc226mgRvQ1JPxGur7nQMTesuXw3z',
  access_token = '14122740-FWlOwl04qvhiy6oTcRypgVaIyvmlg1OZLudAToO6c',
  access_secret = 'sYjzQMjFKQFvMVRCU9gYx7bOteiS4XCoLvCgodTJZVm7y'

## <Token>
## <oauth_endpoint>
##   request: https://api.twitter.com/oauth/request_token
##   authorize: https://api.twitter.com/oauth/authenticate
##   access: https://api.twitter.com/oauth/access_token
## <oauth_app> Test_of_the_API_platform
##   key: AWsZc3pjFsgAF1BK4OHRlyGtK
##   secret: <hidden>
## <credentials> oauth_token, oauth_token_secret
## ---

## Google API key for accessing geo location data through Google Maps
westland_api <- 'AIzaSyCERk3aBmPoG1FAKEqNUz6elhD6ZrR2MQtN7W0'

# To test your authentication, search for 18000 tweets using the rstats hashtag
```

```
rt <- search_tweets(
  "#rstats", n = 18000, include_rts = FALSE
)
```

Example: Extracting Tweets About General Motors and the Auto Industry

```
library(tidyverse)
library(rtweet)

##Query used to select and customize streaming collection method.
## There are four possible methods.

## (1) The default, q = "",
## returns a small random sample of all publicly available Twitter statuses.

## (2) To filter by keyword, provide
## a comma separated character string with the desired phrase(s) and keyword(s).

## (3) Track users by providing a comma separated list of user IDs or screen names.

## (4) Use four latitude/longitude bounding box points to stream by geo location.
##This must be provided via a vector of length 4, e.g., c(-125, 26, -65, 49).

stream_tweets(
  q = "auto, car, general motors, GM",
  timeout = 100,      ## the number of seconds that you will access.
                      ## Max 18000 tweets / 15 min
  parse = FALSE,      ## we'll do this later
  file_name = "tweetsaboutGM.json"
)

## read in the data as a tidy tbl data frame
djt <- parse_stream("tweetsaboutGM.json")

djt <- djt[,3:6]  ## just a few things we'd like to see
glimpse(djt)

## Observations: 1,105
## Variables: 4
## $ created_at <dttm> 2019-12-24 16:09:38, 2019-12-24 16:09:38, 2019-12-24 16:...
## $ screen_name <chr> "mickey_theGreat", "dasanilacourr", "javandon", "kyuthuyt...
## $ text         <chr> "He ``accidentally'' fucked her best friend in HER house so...
## $ source       <chr> "Twitter for iPhone", "Twitter for iPhone", "BIGO LIVE", ...
```

Example: Extracting Tweets About Roland Musical Instruments and Their Industry

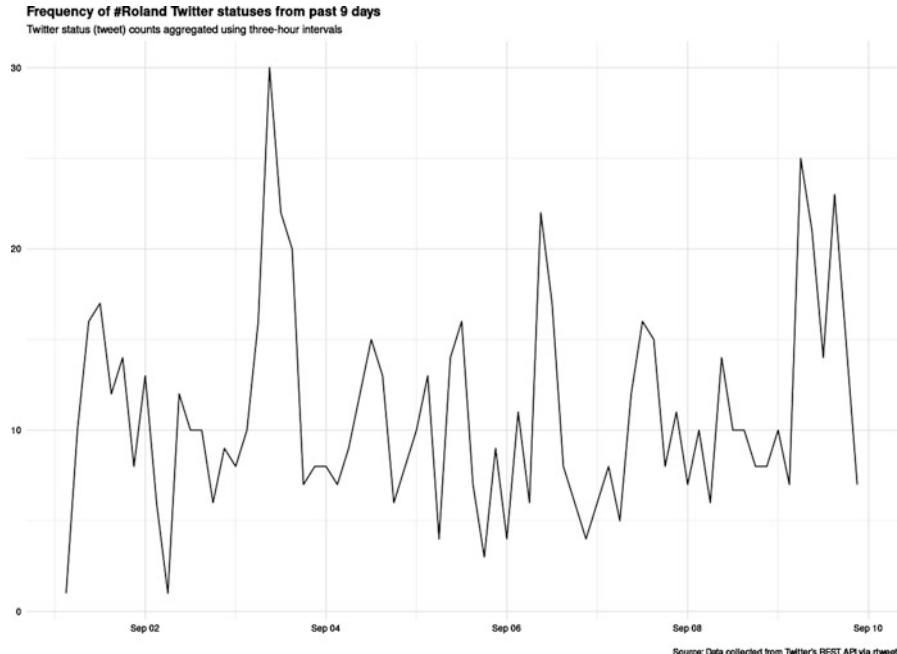
For the following examples of R code, we will be interested in Internet intelligence scanning to support a presumed analytical review of the Roland Corporation. Headquartered in Hamamatsu, Japan, Roland's 3000 employees design, manufacture, and market electronic musical instruments. The firm was publicly traded until 2014 as TYO: 7944. Since it is not a US publicly listed firm, the auditor would not have any access to SEC-EDGAR filings.

```
library(rtweet)
library(httputv)

#Twitter rate limits cap the number of search results returned to 18,000 every 15 minutes.
## To request more than that, set retryonratelimit = TRUE
## and rtweet will wait for rate limit resets for you.
## Here we search for 18000 tweets using the Roland hashtag

roland_tweets <- search_tweets(
  "#Roland", n = 18000, include_rts = FALSE, retryonratelimit = TRUE
)

## plot time series of tweets
roland_tweets %>%
  ts_plot("3 hours") +
  ggplot2::theme_minimal() +
  ggplot2::theme(plot.title = ggplot2::element_text(face = "bold")) +
  ggplot2::labs(
    x = NULL, y = NULL,
    title = "Frequency of #Roland Twitter statuses from past 9 days",
    subtitle = "Twitter status (tweet) counts aggregated using three-hour intervals",
    caption = "\nSource: Data collected from Twitter's REST API via rtweet"
)
```



I can perform a quick sentiment analysis of the Roland tweets using the *NRC Word-Emotion Association* lexicon (available in R in the `textdata` library) which associates words with ten sentiments:

- positive
- negative
- anger
- anticipation
- disgust
- fear
- joy
- sadness
- surprise
- trust

```

library(tidytext)
library(tidyverse)
library(textdata)
library(ggplot2)

reg <- "([^\u00c0-\u00e3-\u00e1-\u00e3\u00f1\u00f3@\u00f1] | (?![\u00c0-\u00e3-\u00e1-\u00e3\u00f1\u00f3@\u00f1]) )"
tweet_words <-
  roland_tweets %>%
  select(user_id, source, created_at, text) %>%
  filter(!str_detect(text, '^')) %>%
  mutate(text = str_replace_all(text, "https://t.co/[A-Za-z\\d]+|&", "")) %>%
  unnest_tokens(word, text, token = "regex", pattern = reg) %>%
  filter(!word %in% stop_words$word,
         str_detect(word, "[a-z]"))

sentmnt <- inner_join(
  get_sentiments("nrc"),
  tweet_words,
  by="word") %>%
  count(sentiment)

ggplot(sentmnt, aes(sentiment, n)) +
  geom_col() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  xlab("Sentiment") +
  ylab("Frequency expressed in tweets")

```

We see from this brief analysis that sentiment is overall very positive for Roland. This clearly provides some insight into the sentiments of consumers and investors, but there is an important caveat. Like most social platforms, Twitter curates very little, and the quality of information conveyed about businesses is highly variable. It is best used to identify potential problems or exceptions that may not appear in other sources. The auditor will want to augment Twitter analysis with curated news sources, and this is the subject of the next section.

Intelligence Scanning of Curated News Streams

Curated intelligence sources, such as Google News, MacRumours, and other news feeds, offer prepackaged information that can be. My favorite is Feedly (stylized as feedly), a news aggregator application that compiles news feeds from a variety of online sources for the user to customize and share with others. Feedly is emblematic of the sort of cloud-based information services that are revolutionizing the audit process.

Start by going to <https://developer.feedly.com/> and then page to <https://feedly.com/v3/auth/dev> to sign in, get a user ID and then follow steps to get your access token. This requires either a “Pro” account or a regular account and you manually requesting OAuth codes. Store it in your `~/.Renviron` in `FEEDLY_ACCESS_TOKEN`.

Feedly will return your access token and refresh token, which looks like the tokens below, and which you can save on your computer, leaving you free to easily access Feedly content.

Congratulations, you have successfully generated a developer access token.

Your user id is 448b1736-7c91-45d2-8a06-1cd797b12edc

Your access token:

`AwPMVtPDxTUC7FAKEIb6_9P:feedlydev`

(expires on 2019-03-15)

Your refresh token (help) :

`AwPMVtPDxTUC44wAQ`

Do not share your tokens!

```
knitr::opts_chunk$set(echo = TRUE)

#' Simplifying some example package setup for this non-pkg example
.pkgenv <- new.env(parent=emptyenv())
.pkgenv$token <- Sys.getenv("FEEDLY_ACCESS_TOKEN")

#' In reality, this is more complex since the non-toy example has to
#' refresh tokens when they expire.
.feedly_token <- function() {
  return(.pkgenv$token)
}
```

For the purposes of this example, consider a “stream” to be all the historical items in a feed. Maximum “page size” (max number of items returned in a single call) is 1000. For simplicity, there is a blanket assumption that if continuation is actually present, we can ask for a large number of items (e.g., 10,000).

```
devtools::install_github("hrbrmstr/seymour")
devtools::install_github("hrbrmstr/hrbrthemes")

library(seymour)
library(hrbrthemes)
library(tidyverse)

# Find writeups for seymour functions at https://github.com/hrbrmstr/seymour/tree/master/man

## useful functions for extracting news are:
# feedly_search_contents: Search content of a stream
# feedly_search_title: Find feeds based on title, url or '#topic'
# feedly_stream: Retrieve contents of a Feedly ''stream''
# feedly_subscribe: Subscribe to an RSS feed
# feedly_subscriptions: Retrieve Feedly Subscriptions

## use Feedly to check news about Apple Inc. and its products
```

```

## (search "feedburner.com" and then the corporation names to see what is available)

apple_feed_id <- "feed/http://feeds.feedburner.com/MacRumors"

## Here is the stream function

feedly_stream(stream_id,
              ranked = c("newest", "oldest"),
              unread_only = FALSE,
              count = 1000L,
              continuation = NULL,
              feedly_token = feedly_access_token())

apple_stream <- feedly_stream(apple_feed_id)
glimpse(apple_stream)

## Here is another function

feedly_search_contents(query,
                       stream_id = NULL,
                       fields = "all",
                       embedded = NULL,
                       engagement = NULL,
                       count = 20L,
                       locale = NULL,
                       feedly_token = feedly_access_token())

f_search <-
  feedly_search_contents(q = "ipod",
                         stream_id = "apple_feed_id",
                         fields = "keywords")
glimpse(f_search)

# preallocate space
streams <- vector("list", 10)
streams[1L] <- list(apple_stream)

# catch all of the content

idx <- 2L
while(length(apple_stream$continuation) > 0) {
  cat(".", sep="") # progress bar, sort of
  feedly_stream(
    stream_id = apple_feed_id,
    ct = 10000L,
    continuation = apple_stream$continuation
  ) -> rb_stream
  streams[idx] <- list(apple_stream)
  idx <- idx + 1L
}
cat("\n")

str(streams, 1)
str(streams[[1]], 1)
glimpse(streams[[1]]$items)

```

Feedly curates numerous news and blog outlets, which distribute a broad array of news items from formal press publications that would have a bearing on the conduct of an audit. I suggest you use the various `lubridate`, `ggplot`, and `tidyverse` tools to analyze and present any insights from this larger dataset.

There are many sources of curated news for conducting analytical reviews in an audit, each with its own merits. The largest consolidator of curated news on the web is arguably Google. The `newsAPI` package is used to access Google's News API using R.

```
## install script
if (!"devtools" %in% installed.packages()) {
  install.packages("devtools")
}
devtools::install_github("mkearney/newsAPI")

## load package
library(newsAPI)

# go to newsapi.org and register to get an API key.
# save the key as an environment variable

## my obscured key
NEWSAPI_KEY <- "079ee9e373894dcfb9a062a85c4ele7e"

## save to .Renvironment file
cat(
  paste0("NEWSAPI_KEY=", NEWSAPI_KEY),
  append = TRUE,
  fill = TRUE,
  file = file.path("~", ".Renvironment")
)

## install script
if (!"devtools" %in% installed.packages()) {
  install.packages("devtools")
}
devtools::install_github("mkearney/newsAPI")

src <- get_sources(category = "", language = "en", country = "", apiKey = NEWSAPI_KEY,
  parse = TRUE)

## load package
library(newsAPI)

df <- lapply(src$id, get_articles, apiKey=NEWSAPI_KEY)

## collapse into single data frame
df <- do.call("rbind", df)

## additional functions allow the parsing of streamed news articles

# get_sources(
#   category = "", language = "", country = "", apiKey = NEWSAPI_KEY,   parse = TRUE)

# x <- get_sources(
#   category = "", language = "", country = "", apiKey = NEWSAPI_KEY,   parse = TRUE)
#parse_sources(x)

# y <- get_articles(
```

```

# source, sortBy = "top", apiKey = NEWSAPI_KEY, parse = TRUE)
#parse_articles(y)

# source: Name of news source.
# sortBy: Name of sorting mechanism must be one of
## latest, top, or popular. Certain methods only work for certain news sources.

# apiKey: Character string API token. Default is to grab it from user R environ.
# parse: Logical indicating whether to parse response object to data frame.

# where x is a response object from get_sources

```

Feedly is a news aggregator application for various web browsers and mobile devices running “iOS” and “Android,” also available as a cloud-based service. It compiles news feeds from a variety of online sources for the user to customize and share with others. Methods are provided to retrieve information about and contents of “Feedly” collections and streams.

Neither `feedly_search()` nor `feedly_stream()` require authentication (i.e., you do not need a developer token) to retrieve the contents of the API call. For `feedly_stream()`, you do need to know the Feedly-structured feed id which is (generally) `feed/FEED_URL` (e.g., `feed/http://feeds.feedburner.com/RBloggers`).

I have generally found Feedly to be more useful than Google News for business intelligence scanning, because it is less hampered by throttling, and the curation extends to industry specific feeds (rather than Google’s algorithmic guess about the topic),

In the following example, consider a “stream” to be all the historical items in a feed. Maximum “page size” (max number of items returned in a single call) is 1000. For simplicity, there is a blanket assumption that if `continuation` is actually present, we can ask for a large number of items (e.g., 10,000).

```

devtools::install_github("westland/auditAnalytics")
library(auditAnalytics)

library(kableExtra)

read.csv(system.file("extdata", "feedly_functions.csv", package =
  "auditAnalytics", mustWork = TRUE)) %>%
  kable("latex", booktabs = T) %>%
  kable_styling()

```

Function	Action
<code>feedly_access_token</code>	Retrieve the Feedly Developer Token
<code>feedly_categories</code>	Show Feedly Categories
<code>feedly_collections</code>	Retrieve Feedly Connections
<code>feedly_continue</code>	Helper function to iterate through a “ <code>feedly_stream()</code> ” result ...
<code>feedly_feed_meta</code>	Retrieve Metadata for a Feed
<code>feedly_opml</code>	Retrieve Your Feedly OPML File
<code>feedly_profile</code>	Retrieve Your Feedly Profile
<code>feedly_search_contents</code>	Search content of a stream
<code>feedly_search_title</code>	Find feeds based on title, url or “#topic”
<code>feedly_stream</code>	Retrieve contents of a Feedly “stream”
<code>feedly_subscribe</code>	Subscribe to an RSS feed
<code>feedly_subscriptions</code>	Retrieve Feedly Subscriptions
<code>feedly_tags</code>	Retrieve List of Tags
<code>global_resource_ids</code>	Global Resource Ids
<code>pipe</code>	Pipe operator
<code>render_stream</code>	Render a Feedly Stream Data Frame to RMarkdown
<code>seymour</code>	Tools to Work with the “Feedly” “API”

```

# devtools::install_github("hrbrmstr/seymour")
# devtools::install_github("hrbrmstr/hrbrthemes")

library(seymour)
library(hrbrthemes)
library(tidyverse)

# Find writeups for seymour functions at https://github.com/hrbrmstr/seymour/tree/master/man

## useful functions for extracting news are:
# feedly_search_contents: Search content of a stream
# feedly_search_title: Find feeds based on title, url or '#topic'
# feedly_stream: Retrieve contents of a Feedly ``stream''
# feedly_subscribe: Subscribe to an RSS feed
# feedly_subscriptions: Retrieve Feedly Subscriptions

## Let's check what is happening at Apple
## (search Google's "feedburner.com" and then the corporation names to see what is available)

## the following should retrieve your Feedly access token, but if not,
## you can log into the site and request the token.

token <- feedly_access_token()

feedly_search_title("roland")

# prefix the URL with 'feed/'
music_feed_id <- "feed/http://feeds.feedburner.com/MusicRadar"
music_feed_id_2 <- "feed/http://feeds.feedburner.com/MusicTech"

## Here is the stream function
feedly_stream(stream_id,
              unt = 1000L,
              continuation = NULL,
              feedly_token = feedly_access_token())

music_stream <- feedly_stream(music_feed_id_2)
glimpse(music_stream)

## Here is another function
feedly_search_contents(query,
                        stream_id = NULL,
                        fields = "all",
                        embedded = NULL,
                        engagement = NULL,
                        count = 20L,
                        locale = NULL,
                        feedly_token = feedly_access_token())

f_search <- feedly_search_contents(q = "ipod", stream_id = "music_feed_id", fields = "keywords")
glimpse(f_search)

# preallocate space
streams <- vector("list", 10)
streams[1L] <- list(music_stream)

# catch all of the content

```

```

idx <- 2L
while(length(music_stream$continuation) > 0) {
  cat(".", sep="") # progress bar, sort of
  feedly_stream(
    stream_id = music_feed_id,
    ct = 10000L,
    continuation = music_stream$continuation
  ) -> rb_stream
  streams[idx] <- list(music_stream)
  idx <- idx + 1L
}
cat("\n")

str(streams, 1)
str(streams[[1]], 1)
glimpse(streams[[1]]$items)

```

API datastreams from social networks, blogs, and other Internet resources tend to be best for qualitative intelligence scanning. They can alert the auditor to information that would not appear in financial news or in the accounting statements and transactions. Such information is an essential part of the analytical review process, but until the advent of Internet accessible resources and automated tools provided by R, has not been accessible to auditors in a cost-effective way.

Accessing General Web Content through Web Scraping

Where relevant intelligence is not available through APIs, but is presented on websites, it is possible to *web scrape* data. This can be difficult and messy, but R provides a number of effective helper tools to scrape and organize data from websites. I provide here a brief introduction to the concept and practices of web scraping in R using the `rvest` package. Tools like `rvest` and *Beautiful Soup* (Python) inject structure into web scraping, which has become important because so few companies are willing to part with their proprietary customer datasets. They have no choice but to expose some of this proprietary data via the web, though, and this is where auditors have an opportunity to accumulate valuable information germane to audit risk. The process of scraping data from the web exemplifies the computer-plus-human model of computing. It is also a nice introduction to building custom software for scraping a specific website.

The basic functions in `rvest` are powerful, and you should try to utilize the following functions when starting out a new web scraping project.

- `html_nodes()`: identifies HTML wrappers.
- `html_nodes(".class")`: calls node based on css class
- `html_nodes("#id")`: calls node based on id
- `html_nodes(xpath="xpath")`: calls node based on xpath
- `html_attrs()`: identifies attributes (useful for debugging)
- `html_table()`: turns HTML tables into data frames
- `html_text()`: strips the HTML tags and extracts only the text

Note on plurals: `html_node()` returns metadata; but `html_nodes()` iterates over the matching nodes. The `html_nodes()` function turns each HTML tag into a row in an R dataframe.

SelectorGadget

SelectorGadget is a *javascript bookmarklet* that allows you to interactively figure out what *css selector* you need to extract desired components from a page. To install it, go to the page:

https://cran.r-project.org/web/packages/rvest/vignettes/selector_gadget.html

Install *selectorgadget* on the Chrome Browser (only at the time of this writing) from <https://selectorgadget.com/>. SelectorGadget is an open-source tool that simplifies CSS selector generation and discovery on complicated sites. Install

the Chrome Extension or drag the bookmarklet to your bookmark bar, then go to any page and launch it. A box will open in the bottom right of the website. Click on a page element that you would like your selector to match (it will turn green). *SelectorGadget* will then generate a minimal CSS selector for that element, and will highlight (yellow) everything that is matched by the selector. Now click on a highlighted element to remove it from the selector (red), or click on an unhighlighted element to add it to the selector. Through this process of selection and rejection, *SelectorGadget* helps you come up with the perfect CSS selector for your needs.

To use it, open the page:

1. Click on the element you want to select. *Selectorgadget* will make a first guess at what css selector you want. It is likely to be bad since it only has one example to learn from, but it is a start. Elements that match the selector will be highlighted in yellow.
2. Click on elements that should not be selected. They will turn red. Click on elements that should be selected. They will turn green.
3. Iterate until only the elements you want are selected. Selectorgadget is not perfect and sometimes will not be able to find a useful css selector. Sometimes starting from a different element helps.

Other important functions:

1. If you prefer, you can use xpath selectors instead of css: `html_nodes(doc, xpath = "//table//td")`.
2. Extract the tag names with `html_tag()`, text with `html_text()`, a single attribute with `html_attr()` or all attributes with `html_attrs()`.
3. Detect and repair text encoding problems with `guess_encoding()` and `repair_encoding()`.
4. Navigate around a website as if you are in a browser with `html_session()`, `jump_to()`, `follow_link()`, `back()`, and `forward()`. Extract, modify, and submit forms with `html_form()`, `set_values()`, and `submit_form()`.

Example: Simple Sentiment Analysis

Here is an example of a simple sentiment analysis for customers' comments on restaurants in Hanoi Vietnam. Start by pointing your browser to <https://www.tripadvisor.com> and searching for "Asian" cuisine in "Hanoi" (as homework, consider other cities or services based on specific audit needs). Click on the "Asian" menu, which brings you to web page <https://www.tripadvisor.com/Restaurants-g293924-Hanoi.html>. Turn on *selectorgadget* in Chrome browser and highlight all of the reviews. In the menu at the bottom of your screen, this will give you an index ".is-9" which is the designator for the CSS code that you have outlined (you can verify this in Chrome by clicking the three dot menu at the upper right-hand corner of the screen, clicking "More Tools" = "Developer Tools" and checking the webpage HTML; or right-click and inspect for a quick look)

```
library(rvest)
library(RColorBrewer)
library(wordcloud)

## Copy the URL of the page you are scraping
url <- "https://www.tripadvisor.com/Restaurants-g293924-Hanoi.html"

## Extract the reviews in the CSS ".is-9" selector
reviews <- url %>%
  read_html() %>%
  html_nodes(".is-9")

## Pull the text out of the reviews
quote <- reviews %>% html_text()

## Turn the character string "quote" into a data.frame and View
data.frame(quote, stringsAsFactors = FALSE) %>% View()

pal2 <- brewer.pal(8,"Dark2") ## from RColorBrewer

wordcloud(quote, colors=pal2)
```



In TripAdvisor, you can use the same methods, in various geographical regions, for: Hotels, Things to do, Restaurants, Flights, Vacation Rentals, Cruises, and other things. Similar methods work for other review and aggregation sites.

Example: Movie Reviews

The next example scrapes information about “The Lego Movie” from IMDB. We start by downloading and parsing the file with `html()`. To extract the rating, we start with *Selectorgadget* to figure out which css selector matches the data we want. We use `html_node()` to find the first node that matches that selector, extract its contents with `html_text()`, and convert it to numeric with `as.numeric()`.

```
library(rvest)
lego_movie <- html("http://www.imdb.com/title/tt1490017/")

lego_movie %>%
  html_node("strong span") %>%
  html_text() %>%
  as.numeric()

## [1] 7.8

## We use a similar process to extract the cast,
## using html_nodes() to find all nodes that match the selector:

lego_movie %>%
  html_nodes("#titleCast .itemprop span") %>%
  html_text()

## character(0)
```

Next find the actors listed on “The Lego Movie” IMDB movie page:

1. Navigate to the page and scroll to the actors list.

2. Click on the selectorgadget link in the bookmarks. The selectorgadget console will appear at the bottom of the screen, and element currently under the mouse will be highlighted in orange.
3. Click on the element you want to select (the name of an actor). The element you selected will be highlighted in green. Selectorgadget guesses which css selector you want (.itemprop in this case), and highlights all matches in yellow.
4. Scroll around the document to find elements that you do not want to match and click on them. For example, we don't want to match the title of the movie, so we click on it and it turns red. The css selector updates to #titleCast .itemprop.

```
library(rvest)

html <- read_html("http://www.imdb.com/title/tt1490017/")
cast <- html_nodes(html, "#titleCast .itemprop")
length(cast)

## [1] 0

cast[1:2]

## {xml_nodeset (0)}

## Looking carefully at this output, we see twice as many matches as we expected.
## That's because we've selected both the table cell and the text inside the cell.
## We can experiment with selectorgadget to find a better match or look at the html directly.

cast <- html_nodes(html, "#titleCast span.itemprop")
length(cast)

## [1] 0

html_text(cast)

## character(0)
```

Example: Tabular Data

Some websites publish their data in an easy-to-read table without offering the option to download the data. Package `rvest` uses `html_table()` for tabular data. Using the functions listed above, isolate the table on the page. Then pass the HTML table to `html_table()`. In the following case, you can go to <https://www.nis.gov.kh/cpi/> and inspect the html.

```
library(rvest)
library(tidyverse)

accounts <- read_html("https://www.nis.gov.kh/cpi/Apr14.html")

table <- accounts %>%
  html_nodes("table") %>%
  html_table(header=T)

# lean up the table
# table[[1]]
dict <- table[[1]][,1:2]
accounts_df <- table[[1]][6:18,-1]

names <- c('id', 'weight.pct', 'jan.2013', 'dec.2013', 'jan.2014', 'mo.pctch', 'yr.pctch', 'mo.cont', 'yr.cont')
colnames(accounts_df) <- names

glimpse(accounts_df)

## Observations: 13
## Variables: 9
## $ id      <chr> "All ITEM (CPI TOTAL)", "FOOD AND NON-ALCOHOLIC BEVER...
## $ weight.pct <chr> "100.000", "44.775", "1.625", "3.036", "17.084", "2.743", ...
## $ jan.2013  <chr> "150.2", "170.4", "127.9", "125.8", "127.8", "130.5", "116...
```

```
## $ dec.2013  <chr> "156.8", "178.7", "136.0", "130.1", "131.1", "141.3", "127...
## $ jan.2014  <chr> "157.7", "180.0", "136.9", "130.4", "131.5", "141.9", "127...
## $ mo.pctch  <chr> "0.5", "0.7", "0.6", "0.2", "0.3", "0.4", "0.3", "0.1", "-...
## $ yr.pctch  <chr> "4.9", "5.6", "7.1", "3.6", "2.9", "8.7", "9.2", "1.2", "-...
## $ mo.cont   <chr> "0.5", "0.4", "0.0", "0.0", "0.0", "0.0", "0.0", "0.0", "0...
## $ yr.cont   <chr> "4.9", "2.9", "0.1", "0.1", "0.4", "0.2", "0.4", "0.1", "0...
```

Example: XPaths

Xpaths are content hierarchies in a website. Sometimes you can get more comprehensive retrieval with an xPath. You can get the xpath that includes some content with the Chrome *xPath Finder* extension (it is like *SelectorGadget* but for xPaths)

```
# example of scraping a table with an XPath
library(rvest)
library(tidyverse)

h <- read_html(
  "https://en.wikipedia.org/wiki/Current_members_of_the_United_States_House_of_Representatives")

reps <- h %>% html_node(xpath = '//*[@id="votingmembers"]') %>%
  html_table(fill=T)
reps <- reps[,c(1:2,4:9)] %>% as_tibble()
```

Example: Extracting Intelligence from Product User Forums

Product user forums are excellent sources of informed consumer and retailer information. This example provides a number of methods that can be used for general web scraping. Applying this to our goal of web scraping for intelligence on Roland's products, we can glean consumer sentiment on Roland's pianos as conveyed by discussions on the Piano World Forum website and display it with wordcloud.

```
#install.packages(c("tm", "SnowballC", "wordcloud", "RColorBrewer", "RCurl",
"XML"))
library(tm)
library(SnowballC)
library(RCurl)
library(tidyverse)
library(rvest)
library(RColorBrewer)
library(wordcloud)
library(stringr)
library(httr)
library(XML)

handle <- handle("http://forum.pianoworld.com//")
path    <- "ubbthreads.php/ubb/login.html?ocu=http%3A%2F%2Fforum.pianoworld.com%2F"

# fields found in the login form.
login <- list(
  amember_login = "westland"
, amember_pass  = "powerpcc"
, amember_redirect_url =
  "http://forum.pianoworld.com//ubbthreads.php/forum_summary.html"
)
```

```
response <- POST(handle = handle, path = path, body = login)

# Copy the URL of the page you are scraping
url <- "http://forum.pianoworld.com/"

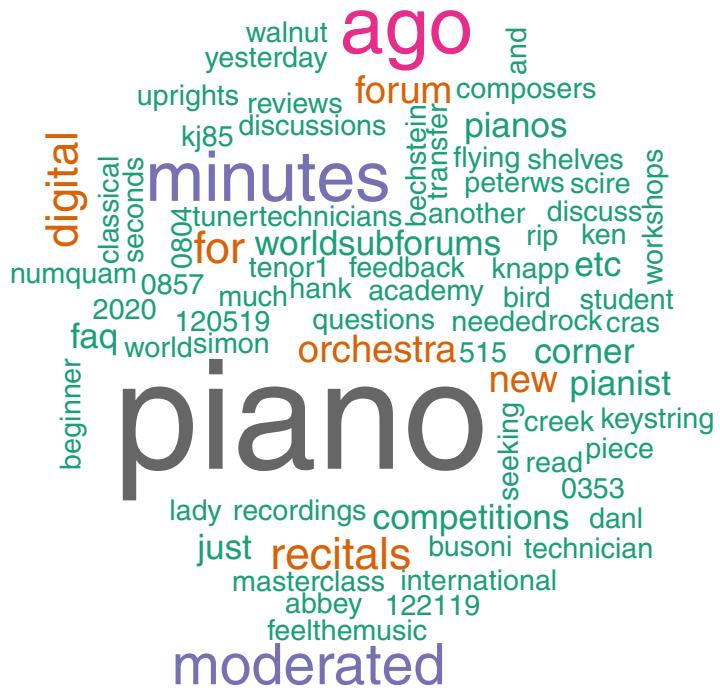
# Extract the reviews in the CSS selector
reviews <- url %>%
  read_html() %>%
  html_nodes("#cat2 div")

# Pull the selected text out of the reviews
quote <- reviews %>%
  html_text() %>%
  as.tibble()

quote <- filter(quote[str_detect(quote, "Roland")])

pal2 <- brewer.pal(8, "Dark2") # from RColorBrewer

wordcloud(unlist(quote), colors=pal2)
```



Final Comments on Analytical Review with R

The previous algorithms and examples should give the reader an overview of some of the tools available to assist with analytical review. This is, of course, not a complete list of tools, or of opportunities. Much of the code in this chapter draws from dynamic and continually updated databases, streams, and standards. New tools and repositories are constantly coming online, particularly in accounting and finance, where the market is large and rich. New packages are often introduced on

GitHub rather than the official Comprehensive R Archive Network (CRAN) repositories, and auditors should always be on the lookout for new offerings.

This chapter provides a start, but there are many opportunities for an ambitious auditor to develop new methodologies for plumbing Internet resources. This will require a level of patience and curiosity that exceeds merely writing code. As I mentioned in the preface, the R language is not a typical language with a single core of developers and guidelines; rather it is a sharing platform for a wide range of data analytic functions—features which make it useful, dynamic, and sometimes messy. The effort put into familiarizing oneself with the R ecosystem will pay off many times over in access to the latest, most sophisticated algorithms that exist anywhere in industry. Because of its unique character, you can be assured that R's packages will always be at the forefront of financial analysis and analytical review, now and in the future.

Design of Audit Programs



Audit Programs as Natural Experiments

Audit programs can best be thought of as stylized *natural experiments*. A natural experiment is an empirical study in which the activities of firms, individuals or groups are exposed to the experimental and control conditions that are determined by nature, or by other factors outside the control of the auditors. The process governing the exposures resembles a “random” assignment. The concept of “randomness” is itself controversial, and I will briefly comment on this later in the chapter; it is intended to assure that samples are representative for some decision-making objective. Natural experiments are observational studies and are *not* controlled in the traditional sense of an experiment. The difference between a natural experiment and a non-experimental observational study is that the former includes a comparison of conditions that pave the way for causal inference, but the latter does not. Natural experiments are employed as study designs when controlled experimentation is extremely difficult to implement or unethical, such as in several research areas addressed by epidemiology and economics (Meyer 1995). Field and quasi-experiments are closely related, but are not appropriate for audit programs; here are the differences:

1. Natural experiments rely on an external force (e.g., a government, nonprofit, etc.) controlling the randomization treatment assignment and implementation,
2. Field experiments require researchers to retain control over randomization and implementation.
3. Quasi-experiments occur when treatments are administered as-if randomly (e.g., US Congressional districts where candidates win with slim-margins, weather patterns, natural disasters, etc.).

The perspective of audits as natural experiments motivates the methods applied to auditing in this book. It is also the reason that the R language is such an invaluable tool for auditing. At the core of this concept is the statistical analysis of *samples* to ultimately render the auditor’s opinion.

Collecting and Analyzing Audit Evidence: Sampling

A statistical sample is a subset of the population that is, for purposes of analysis, representative of that population. The population is all of the transactions relevant to a particular audit or procedure. For example, in confirmation of accounts receivable (A/R) the population is all of the accounts receivable from customers buying on credit for the year (assuming an annual audit). Audit samples are usually several orders of magnitude smaller than the population.

The sampling base or metric is the unit used to draw the sample. Monetary unit sampling (one dollar is a sample item) is applied at year end for substantive testing—determining whether the financial statement account balances contain material errors. Transaction or Record sampling (one accounting transaction is a sample item) is applied at year-end for internal control testing—determining whether the inherent level of transaction processing error risk exists in each accounting cycle.

AICPA Guidelines on audit sampling

Samples are the Achilles heel of statistical inference, and audit inference is no exception. Ideally, samples should be random—the sample should be representative, or every item in the population should have an equal chance of being included in the sample. In addition, a probability model is required to draw valid inferences—this model may have a significant impact on the choice of sampling method, as does the particular feature being measured. The more accurately the model and sampling depict reality, the more accurate, and less costly will be the audit inference.

Sampling is performed because it is typically infeasible to make a detailed inspection and audit of anything but the smallest portion of the population. Without sampling, audits would be economically unjustifiable.

In statistical hypothesis testing (termed Neyman–Pearson hypothesis tests after the Neyman–Pearson Lemma), the *p-value* (probability value or asymptotic significance) is the probability for a given statistical model that, when the null hypothesis is true the sample mean would be greater than or equal to the actual observed results. Hypotheses are ways of bifurcating the decision space of a particular statistical inference task. In auditing, this bifurcation is a stylized *accept* or *do not accept* that a transaction stream is *in-control* (interim tests) or an account balance is *fairly stated* or *is not fairly stated* (substantive tests).

Auditing makes decisions on control and fairness through searches for *material* or *intolerable errors* (the apportionment of *materiality* to individual accounts generates the set of *intolerable errors* for those accounts). Thus, in an auditing context, the p-value of a test is the probability that the monetary error in the account balance would be *intolerable*.

One of the auditing's idiosyncrasies is that audit tests consider only one-sided tests. Auditing, because of the *Conservatism Principle*, is concerned with overstating income and assets, or understating liabilities and expenses. As a consequence, all audit tests are one-sided.

There are three types of sampling commonly used in audits:

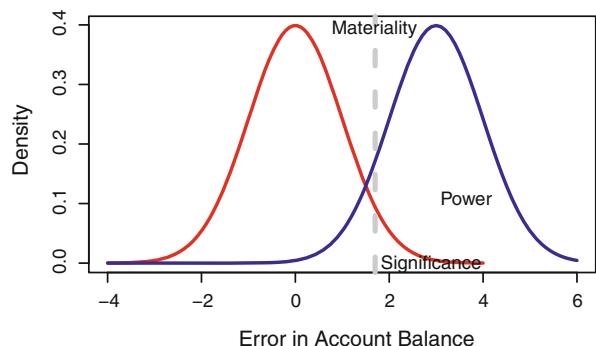
1. *discovery sampling* for interim tests: *Discovery sampling* for interim tests sets an estimation sample size for *transaction-unit* samples, so that we are likely to discover at least one error in the sample if the actual transaction error rate exceeds the *minimum acceptable error rate* (alternatively called the *out-of-control* rate of error). Discovery tests help the auditor decide whether the systems processing a particular transaction stream are in or out of control.
2. *attribute sampling* for interim tests: sets estimation sample size for *transaction-unit* samples to estimate the error rate in the entire transaction population with some confidence (e.g., 95%) that the estimate is within the *out-of-control* error rate cutoff for that transaction stream. If it is found that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Error estimates from attribute samples may either be *rates* or *amounts* or both.
3. *acceptance sampling* for substantive tests: *Discovery sample analysis* results in a decision of whether the internal control over a particular type of transaction error is “in-control” or “out of control.” If internal control is found to be insufficient, the auditor moves on to *attribute sampling* for interim tests, which sets estimation sample size for *transaction-unit* samples to estimate the error rate in the entire transaction population with some confidence (e.g., 95%) that the estimate is within the *out-of-control* error rate cutoff for that transaction stream. If it is found that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Errors estimates from attribute samples may either be *rates* or *amounts* or both.

The sample sizes (an important consideration in determining the scope and budget of an audit) can be determined using Cohen's power analysis (Cohen 1992) which is implemented in R's *pwr* package. Statistical “power” is the complement (i.e., $1 - \beta$) of the probability β of a type II error (type II error is the failure to reject a false null hypothesis, and is also known as a “false negative.”) Recall that, in statistical hypothesis testing, the probability α of a type I error is the “significance” of the test (a type I error is the rejection of a true null hypothesis, and is also known as a “false positive”) (Fig. 1).

To calculate the required sample size, you need to know four things:

- The size of the error to detect.
- The variance of the response.
- The desired significance level.
- The desired power.

Fig. 1 Probability model under null (red) and alternative (blue) hypotheses: materiality, power, and significance



Sampling for Interim Tests of Compliance

Discovery Sampling

Discovery sampling chooses a sample to determine whether an error rate does not exceed a designated percentage of the population. If the sample does not contain errors, then the actual error rate is assumed to be lower than the minimum unacceptable rate. The sampling calculation includes the following factors:

- Confidence level.
- Minimum unacceptable error rate.

Confidence level is a concept that originated in fiducial inference, an approach that has fallen out of fashion in favor of frequentist inference, Bayesian inference, and decision theory. It is still used in decision theoretic approaches such as auditing, and roughly reflects the confidence that the auditor has in a particular decision (e.g., the balance is “fairly stated”). Confidence is a concept that is intertwined with perceptions of “risk” associated with audit failures.

Unacceptable error rates are parameters that are fixed at the start of an audit by the audit manager. They may be set by the firm, by prior years experience, or by some other method. In general, their choice is idiosyncratic, determined by policies and perspectives of a particular firm or audit professional.

We can compute the discovery sample size using what mathematicians call an *urn model*. Urn models are typically stated as draws of colored balls from an urn. In our case, we can consider the urn to be the set of all transactions of a given type that the firm processes in a given accounting period.

Assume that the auditor determines that the *minimum acceptable error rate* for a particular transaction type (or alternately our *out-of-control* rate of error for that transaction type) is p . The discovery sample size needed for confidence c is $Pr[X \geq 0] = 1 - Pr[x = 0]$ the probability that x , the number of errors discovered, is anything but 0.

We can start by solving the probability of finding no errors in a sample of n draws from the urn:

$$Pr[X \geq 0] = 1 - \binom{n}{0} \times p^0 \times (1-p)^{n-0} = 1 - (1-p)^n$$

For confidence level c we want to choose n so that:

$$Pr[X \geq 0] = c \implies 1 - (1-p)^n = c \implies n = \frac{\log(1-c)}{\log(1-p)}$$

```
library(tidyverse)
library(reshape)

p <- seq(.0005,.015,.0005)
c <- .95
n <- log(1-c)/log(1-p)
samp <- data.frame(p,n)
ggplot(samp,aes(p,n)) +
  geom_line() +
```

```
labs(title="sample size for 95% confidence") +
  xlab("minimum acceptable error rate in population") +
  ylab("sample size")
```

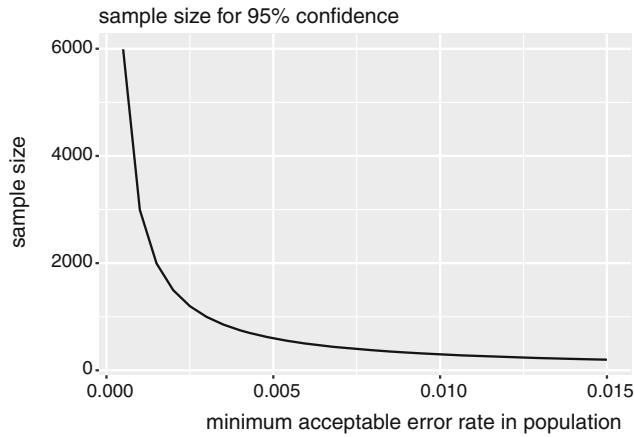


Fig. 2 Discovery sample size and minimum acceptable error rate = 0.1

Coefficient of Variation Formulas for Sample Size

If the auditor is not just concerned with the error rate, but also the stability of the error rate, then interim test sampling questions will involve both the error rate and the variability of that rate. If the *out-of-control* error rate is μ_1 and our goal is to detect an error of size Δ where the standard deviation of the sample is assumed to be σ then the sample size formula above becomes

$$n = \frac{8CV^2}{PE^2}[1 + (1 - PE)^2]$$

Where PE is the proportionate error $PE = \frac{\Delta}{\mu_1}$ and CV is the coefficient of variation $CV = \frac{\sigma}{\mu_1}$. Sometimes the auditor will not have any idea of the variability of the population, but will still wish to consider it in sample size. In this case, a variability of $\approx \frac{1}{3} \approx 35\%$ is typical. In this case:

$$n \approx \frac{1+(1-PE)^2}{PE^2} \approx \frac{2}{PE^2} \text{ for small error rates.}$$

Rules of Threes to Calculate 95% Upper Confidence Bounds

The rule of threes can be used to address the following type of question:

“The manager of the audit has told us that there have been no errors in the A/R account balance in the last 20 audits. Does this information give me an estimate whether there will be an error in this year’s audit of A/R?”

The answer is “yes.” Given no observed errors in the past n years, a 95% upper bound on the rate of occurrence is $\frac{3}{n}$.

The basis of this is that, by the law of rare errors, the observed events Y (i.e., no errors in the audit) follow a $Poisson(\lambda)$ distribution using n samples. The sum of Poisson random variables is Poisson, so the question of at least one Y not equal to zero is the probability that the sum $\sum Y_i$ is greater than zero. Let this probability be, for example, 0.95 so that $P[\sum Y_i = 0] = e^{-n\lambda} = 0.05$. Taking logarithms gives $n\lambda = -\ln(0.05) \approx 3$. Thus the rate $\lambda \approx \frac{3}{n}$.

Attribute Sampling

Each time a control is tested, the auditor examines the evidence (samples, etc.) to decide on the hypothesis that the account is fairly stated, or internal control is effective. Since even an “in-control” system will produce the occasional error, with an acceptable rate of error s , the auditor could make the assumption that $s = 0$ (which would give a small sample size), but preferably a realistic estimate of actual error should be used. The Risk Assessment Matrix and review of prior years’ audit papers for the control system will provide this rate s . The control decision takes the form of the following hypothesis test:

H_0 : The client’s system is in control, controls are effective, and error rate is approximately s .

H_a : The client’s controls are *not* effective, and the system is producing an intolerable number of errors $> r$.

Restating, if the expected error rate is s , the intolerable error rate is r , and the error rate in the control system is ϵ then restate the hypotheses as:

$H_0 : \epsilon = s$ where $\epsilon \in [0, 1]$

$H_a : \epsilon \geq r$ where $r, \epsilon \in [0, 1]$

This requires a one-sample proportion test to calculate power and sample size. The null hypothesis is $\epsilon = s$, and the alternative hypothesis is $\epsilon > r$ with the standard power and significance assumptions of $\alpha = 0.05$ and $\beta = 0.8$. Let us test a \$5% \$ intolerable error rate $r = 0.05$ and an expected error rate $s = 0.01$ (Fig. 3):

```
library(pwr)
sample <- pwr.p.test(h = ES.h(p1 = 0.05, p2 = 0.01),
                      sig.level = 0.05,
                      power = 0.80,
                      alternative = "greater")
sample

## proportion power calculation for binomial distribution (arcsine transformation)
##
##          h = 0.250692
##          n = 98.37558
##          sig.level = 0.05
##          power = 0.8
##          alternative = greater

plot(sample)
```

Acceptance Sampling

The basic idea of calculating power or sample size with the *pwr* package is to leave out the argument you want to calculate. If you want to calculate sample size, leave *n* out of the function.

To calculate power and sample size for one-sample t-tests, we need to set the type argument to “one.sample.”

Each time an account is audited or a control is tested, the auditor examines the evidence (samples, etc.) to decide on the hypothesis that the account is fairly stated, or internal control is effective. The decision takes the form of a hypothesis test. For an account balance, this is:

H_0 : The client’s balance for the account is correct (error is zero)

H_a : The client’s balance for the account contained a material error.

If intolerable error (i.e., the part of materiality allocated to this account) is M and the error in the account balance is ϵ then restate this as:

$H_0 : \epsilon = 0$

$H_a : \epsilon \geq M$

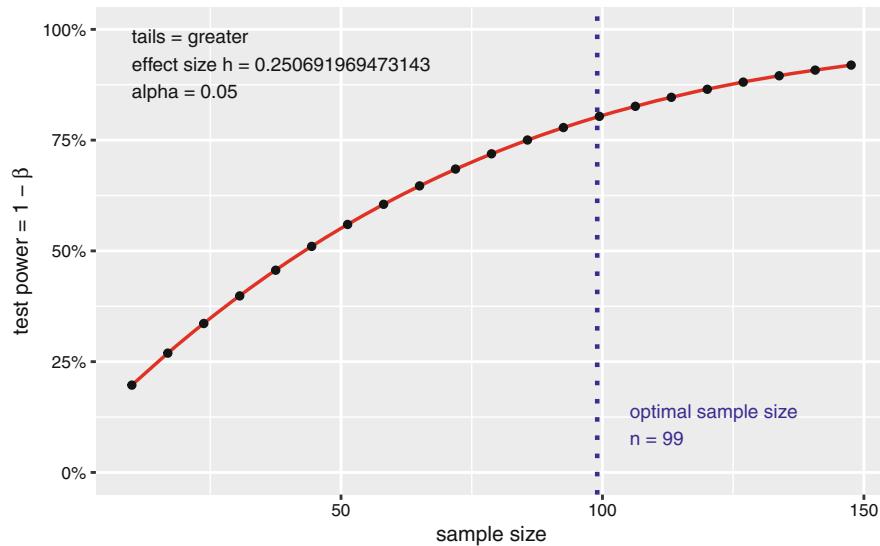


Fig. 3 Power of the test and sample size for a one sided test

In the alternative hypothesis, whether M is Dr or Cr is determined by the Conservatism Principle – the direction that will produce the lowest income.

Let us start with a basic formula for distinguishing between a zero error μ_0 and the actual error μ_1 with a one-sided test, a Normal distribution probability model having homogeneous variances $\sigma_0^2 = \sigma_1^2 = \sigma^2$ and single sample compared to a known (i.e., $\mu_0 = 0$) distribution. The “rule of thumb” for sample size is

$$n = \frac{8}{\Delta^2}$$

Where the error Δ we are trying to detect is:

$$\Delta = \frac{\mu_1 - \mu_0}{\sigma}$$

This is derived from the Normal sample formula:

$$n = \frac{2(z_{1-\alpha} + z_{1-\beta})^2}{(\frac{\mu_1 - \mu_0}{\sigma})^2}$$

For $\alpha = .05$, $\beta = .20$, $z_{1-\alpha} = 1.96$, and $z_{1-\beta} = .84$ the value $2(z_{1-\alpha} + z_{1-\beta})^2 \approx 8$.

This formula can also be used to calculate detectable error in the population using a given sample size n is $\Delta = \sqrt{\frac{8}{n}}$, the inversion of $n = \frac{8}{\Delta^2}$.

Suppose you are auditing a financial account, and want to compare the actual error in the account to an assumption (null hypothesis) that that account is fairly stated (has zero error). You will measure the actual expected error μ_2 with respect to an assumed error of μ_1 , which will typically be zero. We can then define $\Delta = \mu_1 - \mu_2$. The smaller the difference you want to detect, the larger the required sample size.

Of the four variables that go into the sample size calculation, the variance of the responses can be the most difficult to determine. Usually, before you do your experiment, you do not know what variance to expect. Investigators often conduct a pilot study to determine the expected variance, or information from a previously published study can be used.

The effect size combines the minimal relevant difference and the variability into one measurement $\frac{\Delta}{\sigma}$.

Significance α is often set at 0.05, for a 95% confidence, after a suggestion by RA Fisher in the 1920s (Stigler 2008).

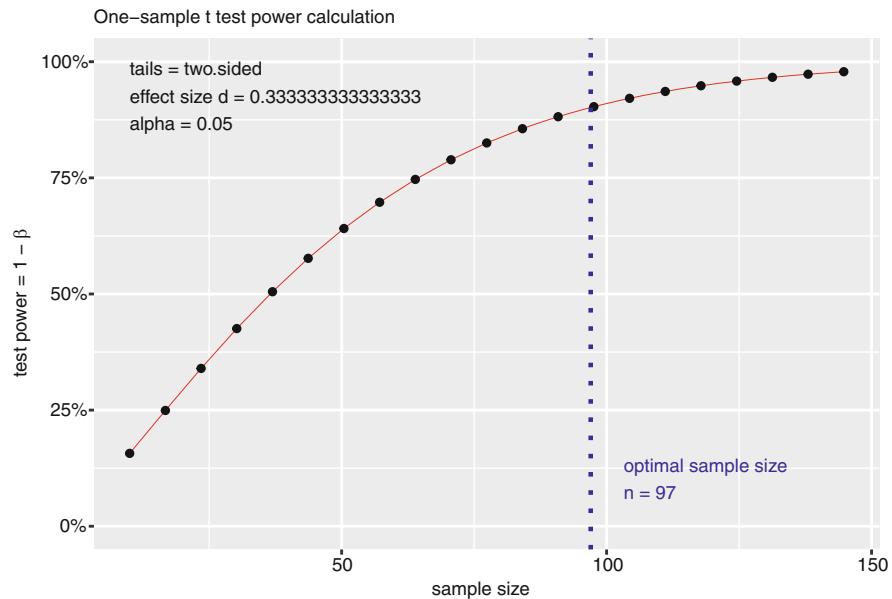


Fig. 4 Power of the test and sample size for a two sided test

$1 - \beta$, where β is the probability of a Type 2 error (failing to reject the null hypothesis when the alternative hypothesis is true). In other words, if you have a 20% chance of failing to detect a real difference, then the power of your test is .8, which was suggested in (Cohen 1992) (Fig. 4).

The calculation for the total sample size is

$$n = \frac{4 \times (z_\alpha + z_\beta)^2 \times \sigma^2}{\Delta^2}$$

```
library(pwr)

Delta <- 20
sigma <- 60
d <- Delta/sigma
sample <- pwr.t.test(d=d, sig.level=.05, power = .90, type = 'one.sample')
sample

## 
##      One-sample t test power calculation
## 
##           n = 96.50801
##           d = 0.3333333
##      sig.level = 0.05
##           power = 0.9
##      alternative = two.sided

plot(sample)
```

Acceptance Sampling with Poisson Data

Substantive testing often takes a “monetary unit” perspective of error, where each dollar in an account balance is assumed to be a sample-able unit. With monetary units, data can be assumed to be Poisson distributed, taking on discrete dollar-unit values greater than zero (i.e., being left truncated at zero). The Poisson distribution has one parameter, the rate λ . The required sample size to detect an error rate of λ is

$$n \approx \frac{4}{\lambda}$$

This is derived from the previous formula, noting that for a Poisson random variable Y , the transformed random variable $\sqrt{Y} \approx \text{Normal}(\mu = \sqrt{\lambda}, \sigma^2 = .25)$.

A Statistical Decision Framework for Auditing

A statistical decision framework for making audit decisions restructures these standards around a sequence of tasks:

1. *Objective of Audit Tasks:* Audit tasks must ultimately provide the basis for the auditor to make a decision to issue one of the four audit opinions: Unqualified, Qualified, Adverse, or Disclaimer.
 - (a) This decision is the implementation of reporting standard.
2. *True State of the Firm:* The audit report is a statement of the auditor’s decision after completing all audit tasks (thus an opinion, because it is not 100% certain) on the unobservable true state of the firm at the audit date; e.g., whether or not there is a material error in the accounts.
 - (a) Errors arise from non-compliance with generally accepted accounting principles (GAAP), inconsistencies from period to period, and inadequate, biased or improper information acquisition.
 - (b) Clearly, investors would like accurate information on firm operations and assets (this was the reason that auditing was originally mandated for public companies); this is the same as saying that they want to know the unobservable true state of the firm and depend on auditors to tell them whether or not the financial reports of the audited firm accurately represent that unobservable true state of the firm.
3. *Model:* The audit model is captured in the audit program which is a set of procedures that compare an “ideal” accounting of firm operations in compliance with GAAP to the actual procedures that were applied during the accounting period.
 - (a) The audit program presents the decision model that allows rendering an audit opinion.
 - (b) The auditor obtains a sufficient understanding of the entity and its environment, including its internal control, to assess the risk of material misstatement of the financial statements, whether due to error or fraud, and to design the nature, timing, and extent of further audit procedures.
4. *Evidence:* Acquisition of consistent, unbiased, complete, and adequate audit evidence is the major source of cost of an audit.
 - (a) The auditor must obtain sufficient appropriate audit evidence by performing audit procedures to afford a reasonable basis for an opinion regarding the financial statements under audit.
5. *Statistical decision theory:* These are mathematical tools that provide widely accepted (and defensible in a court of law) tools to support high quality and defensible audit opinions.
 - (a) Auditing standards suggest that auditors (among other things) must be proficient in statistical theory, and should obtain some computer programming skills.
 - (b) Auditors can assure that they maintain independence in mental attitude (free of bias) and exercise due professional care during the performance of the audit and the preparation of the report by invoking time-tested and widely respected research statistical decision models.
 - (c) Some components of audit decision-making may fall outside the purview of statistics when they involve unobservable constructs, subjective decisions, human behavior and judgment, and so forth. GAAP has focused on objectivizing accounting to the maximum extent; but the growth of the information economy has seen the growth of many intangible

components of firm value that have thwarted this objectification. Application of statistical tests are generally more challenging where subjective intangible assets are involved.

6. *Opinion (Audit Decision):* Material error in the audited financial statements exists or does not. Typically this decision requires a pooling of information from supporting decisions (e.g., the existence of tolerable error in individual year-end accounts).
7. *Assessment of decision quality:* statistical goodness-of-fit statistics determine how confident the auditor is that the opinion is defensible in court.
 - (a) Audit literature suggests a minimum 70% confidence level (without a full statistical explanation of the meaning of this), whereas statistical literature suggests confidence approaching 95% or greater for adequate decision-making. Ronald Fisher first proposed the 95% confidence standard in the early twentieth century, a standard that has withstood the test of time.
 - (b) AICPA and FASB guidance on “decision quality” can be somewhat equivocal, and unfortunately, too often, the adequacy of audit decision-making is decided in court.

Audit Tests as Learning Models

Auditing may be seen as a sequence of evidence collection and analyses that help the auditor learn enough to render an opinion. Conceptually, the tests involve a sequence of inferences:

1. Audit planning will have set the scope and sample size values for audit tasks that, in general, seek to discover (i.e., discovery sampling) whether or not transaction processing is or is not in control. This is usually stated as an assessment of error rates rather than dollar magnitudes of error.
2. If errors are discovered in a particular transaction flow, then sufficient additional evidence must be gathered to assess the rate of error occurrences in the transaction flow (i.e., attribute sampling)
3. Transaction flows with error rates that are sufficiently high to be deemed “out of control” (i.e., intolerable from the audit perspective) are listed on the Internal Control Memo generated at the end of the Interim compliance audit.
4. Financial statement account balances affected by any transaction flows with error rates that are sufficiently high to be deemed “out of control” will have their sample sizes in the audit program for year-end substantive tests adjusted to assure an accurate assessment of the existence of material error in the audited financial statements.

Various ad hoc and formal models have been proposed for the gradual accumulation of knowledge of the accounts throughout the audit. My own preference is for systems of audit task evidence accumulation modeled as Bayesian conjugate priors. The most general of these involves the Gamma distribution, but we will simplify some of the steps using the simpler one-parameter Poisson distribution. Both distributions take on values on the positive quadrant, and thus are more suitable for modeling accounting distributions than the Normal distribution, whose support spans the real line.

Auditing may be seen as a sequence of evidence collection and analyses that help the auditor learn enough to render an opinion. The following steps define the audit tasks in a statistical framework:

1. Set Decision Values for the Audit: Set “materiality” (maximum tolerable error in financial statements as a whole to determine “fairness” of presentation in the opinion) and allocate “tolerable Error” amounts to each of the financial statement accounts.
2. Interpret Decision Values into “In/Out” of Control Values: Set “tolerable error rate” for interim tests on a particular accounting cycle (i.e., the collective process of recording and processing the accounting events, usually a specific information system). These may be related to “materiality” by the following formula to the most significant transaction flow in the accounting cycle (e.g., sales transactions in the Sales & Collections Cycle):
Or alternately
3. Establish Prior Beliefs: Risk assessment establishes prior beliefs—based on prior years audit, analytical review, and other evidence—for particular error control processes in the accounting cycles being “in” or “out” of control—i.e., low or high rates of error
4. Conduct Interim Tests: Interim audit (organized around accounting processes) for each tested error type, set sample sizes based on “in” or “out” of control risk assessment. These tests may use “attribute” sampling to estimate the actual error rate for the accounting period; or “discovery” sampling to determine whether or not the accounting process is actually “in” or “out” of control for the current accounting cycle.

5. Complete Interim Compliance Tests: Additional “attribute” samples will be taken for “discovery” sample “out” of control conclusions to estimate the actual error rate during the accounting cycle.
6. Substantiate Account Balances: Year-end substantive testing (organized around financial statement account balances) sample size for each audited account is set to determine whether each account does or does not contain more than a “tolerable error.”
7. Aggregation: Reverse the mathematics in step 1, to aggregate (via a linear model) the error amounts discovered in each account into a determination of whether “material” error exists in the financial statements as a whole. The error found in the i th account is a multiplier for the significance of the i th account in the audit opinion (generally related to the dollar magnitude of the account balance). If the total error is less than materiality, then an unqualified opinion is warranted. The linear model would be:

If the auditor obtains sample values which provide the evidential “learning” from a particular audit task related to a particular audited value, then these may be aggregated into prior knowledge about the audited value through a sequence of Bayesian conjugate priors. In Bayesian probability theory, if the posterior and prior probability distributions are in the same family, then the prior and posterior are then called conjugate distributions. The prior is called a conjugate prior for the likelihood function that summarizes the data (Raiffa and Schlaifer 1961). A conjugate prior is an algebraic convenience, giving a closed-form expression for the posterior. Conjugate priors provide insight into how evidence updates prior belief.

The major hurdle in formalizing “learning” models of audit is that auditors must pool results of review, inference, and testing in a formal way, from four sequential sets of tasks:

1. review of prior years’ working papers,
2. analytical review,
3. interim tests of transactions, and
4. substantive testing.

The audit literature is generally mute on the procedure for pooling, and generally presumes the auditor pools information in some *ad hoc* and idiosyncratic manner. This section presents a formal Bayesian conjugate approach that lends to reproducibility and accuracy to inference, and can lower sample sizes needed in interim and substantive testing.

Bayesian pooling assumes that prior beliefs can be represented as Gamma distributions, and that evidence from sampling yields a monetary unit distribution that is Poisson. Both distributions’ supports are non-negative, which reflects accounting reality better than the Gaussian assumptions that are too often made. At each of the stages 2., 3., and 4. above, the following Bayesian revision is made:

$$\begin{aligned} \text{Gamma}(\text{prior belief} = \{\alpha, \beta\}) \times \text{Poisson}(\lambda = \text{evidence } \{x_i\}) \\ = \text{Gamma}\left(\text{posterior belief} = \{\alpha + \sum_{i=1}^n x_i, \beta + n\}\right) \end{aligned}$$

The pooling of interim decisions on control weakness and substantive evidence on the material error may involve the pooling of several transaction streams, via a particular arithmetic formula reflecting the transaction components of each audited account—e.g., *accounts receivable = credit sales – collections – uncollectable accounts*. Pooling can be accomplished by approximating the belief distribution from each component transaction streams for the account as a Gamma distribution rather than a Poisson (i.e., looking at interarrival times rather than counts, thus the α value is fixed, and only the rate β is set from the auditor beliefs) and using the following Bayesian conjugate revision to compute the Gamma prior for the substantive evidence:

$$\begin{aligned} & \text{Gamma}(\text{prior belief from 1st transaction stream} = \{\alpha, \beta\}) \\ & \quad \times \text{Gamma}(\text{belief from 2nd transaction stream} = \{\alpha_{\text{fixed}}, \beta_{\text{account}}\}) \\ & = \text{Gamma}(\text{posterior belief} = \{\alpha_{\text{fixed}} + n \cdot \alpha, \beta_{\text{account}} + n \cdot \beta\}) \end{aligned}$$

The substantive test audit decisions, based on confidence limits, on whether or not account balances contain *intolerable error* can be calculated from the posterior Gamma distributions

Materiality

Materiality is a concept or convention within auditing and accounting relating to the importance/significance of an amount, transaction, or discrepancy. The objective of an audit of financial statements is to enable the auditor to express an opinion whether the financial statements are prepared, in all material respects, in conformity with an identified financial reporting framework such as Generally Accepted Accounting Principles(GAAP). The assessment of what is material is a matter of professional judgment. “Information is material if its omission or misstatement could influence the economic decision of users taken on the basis of the financial statements. Materiality depends on the size of the item or error judged in the particular circumstances of its omission or misstatement. Thus, materiality provides a threshold or cutoff point rather than being a primary qualitative characteristic which information must have if it is to be useful.” The Financial Accounting Standards Board (FASB) has refrained from giving quantitative guidelines for determining materiality. This has resulted in confusion in the use of Auditing Standards No 47, “Audit Risk and Materiality in Conducting the Audit.” Several common rules that have appeared in practice and academia to quantify materiality include:

- Percentage of pre-tax income or net income (i.e., 5% of average pre-tax income (using a 3-year average));
- Percentage of gross profit;
- Percentage of total assets; (i.e., 1/3% of total assets);
- Percentage of total revenue; (1/2% of total revenues);
- Percentage of equity; (i.e., 1% of total equity).

During the planning phase of an audit, auditors establish tolerable error rate and materiality (both benchmark figures against which to compare the actual error rate and substantive error) to determine the nature, timing, and extent of audit procedures to perform.

Materiality is, in turn, allocated to individual financial statement elements or classes of transactions as described by AU Section 350 of the AICPA Professional Standards as “tolerable misstatement”—essentially the materiality for a single account on the financial statements. A tolerable misstatement is the amount an individual financial statement account can differ from its true amount without affecting the fair presentation of the financial statements taken as a whole. While the tolerable misstatement assigned to a particular account is less than planning materiality, the sum of tolerable misstatement assigned to all accounts is commonly greater than planning materiality.

Risk

The concepts of “fairness” of audited financial reports are most directly articulated in the AICPA’s professional standards AU Section 312 “Audit Risk and Materiality” which recognizes the inherent uncertainty of an auditor opinion on whether an entity’s financial statements are “fairly” presented—this uncertainty is referred to as audit risk. Audit risk is a function of three risks:

1. Inherent risk: Inherent risk is the risk that a material misstatement will occur with a management assertion assuming no internal control.
2. Control risk: Control risk is the risk that a material misstatement will occur or not be prevented by entity internal control.
3. Detection risk: Detection risk is the risk that a material misstatement with a management assertion will not be detected by the auditor’s substantive tests.

These risks are addressed in accounting systems through internal controls—preventive address inherent and control risk, and detective and corrective controls address detection risk. AICPA literature suggests that these risks are multiplicative, but this is, in general, a misleading assumption.

The Risk Assessment Matrix (RAM) The audit Risk Assessment Matrix (RAM) considers each of these three risks with respect to each particular procedure on each economic transaction. The RAM is a planning tool to assure that sample sizes and scope of tests are adequate to ensure that the overall risk of inappropriately issuing an unqualified opinion is limited to an appropriate level—essentially that Pearsonian Type I error is below some level which is set by the auditor, through decisions that simultaneously set “materiality” and “tolerable error.”

The auditors’ responsibility when conducting an audit is to provide reasonable assurance that the financial statements are fairly presented in all material respects. AU Section 312 of the AICPA Professional Standards states that financial statements

are materially misstated when they contain misstatements whose effect, individually or in the aggregate, results in financial statements that are not fairly presented.

AU Section 312 and 319 of the AICPA Professional Standards indicates that the auditor uses the assessed levels of control risk and inherent risk to determine the acceptable level for detection risk—that an unqualified opinion will be issued when a material misstatement exists. This risk is referred to as audit risk. These standards then set the sequence of steps taken by the auditor in applying statistical procedures to the audit. The Risk Assessment Matrix lists:

1. Rows: each major accounting process for each economically significant transaction.
2. Columns: dollar and transaction volume, the judgmental estimate of error occurrence rate, and the amount at risk for the yearly accounting period.

Judgments are based on prior year audit records, interviews, and other evidence of activities during the year.

The AICPA on Sampling Approaches and Risks

The AICPA's pronouncements on sampling and associated audit risk are somewhat idiosyncratic and ad hoc. In general, it is wise to take them as rough guides to the government body's intentions rather than practical tools for implementation. The AICPA's guidance tends to ignore both probability distributions as well as the role that data should take in inference. Thus rote applications of their formulas are liable to lead to errors in opinion formulation, and inefficient, ineffective use of evidence.

The AICPA distinguishes between “sampling risk”, which is the risk of incorrect decision-making when the sampled evidence is used in the inference; and non-sampling risk, which results from human error and thus allows any audit failures to be attributed to failures in the judgment of individual auditors. This differentiation opens up the potential for finger-pointing in lawsuits. It was at the root of the extensive scapegoating and cover-up of Arthur Andersen's activities in the Enron scandal.

The AICPA defines two aspects to sampling risk when performing tests of controls:

1. The risk of assessing control risk too low represents the risk that an audit sample supports the conclusion that the design and operation of an internal control is effective when, in fact, it is not.
2. The risk of assessing control risk too high represents the risk that an audit sample supports the conclusion that the design and operation of internal control is not effective when, in fact, it is effective.

The AICPA defines two aspects to sampling risk when performing substantive tests:

1. The risk of incorrect acceptance represents the risk that an audit sample supports the conclusion that a material misstatement does not exist when, in fact, a material misstatement does exist. This risk is similar to the risk of assessing control risk too low.
2. The risk of incorrect rejection represents the risk that an audit sample supports the conclusion that a material misstatement exists when, in fact, a material misstatement does not exist. This risk is similar to the risk of assessing control risk too high.
3. Random Sampling involves selecting items from the population so that each item has an equal chance of being selected. Random selection requires the use of random number tables or computer programs to guarantee that each population item has an equal chance of selection.
4. Systematic Sampling involves selecting every k th item from the population after a random start.
5. Haphazard Sampling involves selecting items from the population without consideration to known characteristics of the items in the population (i.e., any conscious bias in the selection of population items).
6. Block Sampling involves selecting items from the population in contiguous groups (or blocks).
7. Purposive or Directed Sampling involves selecting items from the population using some prespecified criteria (i.e., selecting accounts receivable for confirmation based on the outstanding balance).

The first two sample selection methods are referred to as probability (statistical) sample selection methods since every population item has a known probability of selection. The last three methods are referred to as non-probability (non-statistical) sample selection methods since every population item does not have a known probability of selection. The AICPA defines attribute sampling as a statistical approach used with tests of controls. It requires auditors to establish three factors:

1. Risk of assessing control risk too low;

2. Expected Population Deviation Rate;
3. Tolerable Deviation Rate represents the highest deviation rate the auditor could accept and still conclude that the design and operation of an internal control is effective.

If the computed maximum population deviation rate is less than or equal to the tolerable deviation rate, the auditor will conclude that the design and operation of the internal control is effective. The AICPA defines probability-proportionate-to-size sampling (PPS) sampling (also called monetary unit sampling or dollar-unit sampling) as a procedure for substantive tests where each dollar in the population is treated as a separate sampling unit (instead of each transaction-unit, e.g., customer, invoice, check, vendor, etc.). Factors considered when determining sample size for substantive tests are:

1. Risk of incorrect acceptance: type I error risk that the auditor concludes that a material misstatement does not exist when, in fact, a material misstatement does exist.
2. Expected Number of Misstated Dollars: a prior estimate of the dollar misstatement in the sample.
3. Tolerable Misstatement: the highest dollar misstatement that could occur before the population would be considered materially misstated.
4. Reported Monetary Balance: a point estimate of total recorded dollar balance in the population.

The AICPA defines the risk of incorrect acceptance, and the expected number of misstated dollars are used to determine the PPS Factor from the PPS table. The sample size is then determined using the following formula:

The substantive tests are not performed on individual dollars but rather are performed on “logical units” (transactions such as invoice, customer, check, vendor, etc.) containing individual dollars. This is simply a particular way of stratifying the sample. The AICPA defines a tainting as the percent misstatement for each misstated dollar and calculates it using the following formula:

The AICPA calculates an unadjusted upper and lower limit on misstatement using the AICPA’s PPS table. These calculations require the auditor to know the risk of incorrect acceptance, sample size, recorded dollars in the population (reported balance), sample dollars misstated, and tainting of misstated dollars. The unadjusted upper limit on misstatement is based on the observed overstated dollars and the lower limit on misstatement is based on the observed understated dollars. The unadjusted upper and lower limits on misstatement are calculated using the formula: The AICPA is careful to avoid using the term ‘confidence limit’ for its limits. Overall, it is difficult to determine exactly how the AICPA’s “adjusted limits” should be interpreted or used.

AICPA Pronouncements on Generally Accepted Auditing Standards

The American Institute of Certified Public Accountants (AICPA) has provided a set of Generally Accepted Auditing Standards (GAAS) summarized in AU Section 150 (these sections refer to the unified standards which you can find at the AICPA’s website). There are ten standards: three general standards, three fieldwork standards, and four reporting standards. Since 2002 in the USA, the Public Company Accounting Oversight Board has been responsible for Auditing Standards (AS) for publicly traded companies, but adheres broadly to the standards initially set by the AICPA.

The ten standards are:

1. General
 - (a) The auditor must have adequate technical training and proficiency to perform the audit.
 - (b) The auditor must maintain independence in mental attitude in all matters related to the audit.
 - (c) The auditor must exercise due professional care during the performance of the audit and the preparation of the report.
2. Field Work
 - (a) The auditor must adequately plan the work and must properly supervise any assistants.
 - (b) The auditor must obtain a sufficient understanding of the entity and its environment, including its internal control, to assess the risk of material misstatement of the financial statements, whether due to error or fraud, and to design the nature, timing, and extent of further audit procedures.
 - (c) The auditor must obtain sufficient appropriate audit evidence by performing audit procedures to afford a reasonable basis for an opinion regarding the financial statements under audit.

3. Reporting

- (a) The auditor must state in the auditor's report whether the financial statements are presented in accordance with generally accepted accounting principles.
- (b) The auditor must identify in the auditor's report those circumstances in which such principles have not been consistently observed in the current period in relation to the preceding period.
- (c) When the auditor determines that informative disclosures are not reasonably adequate, the auditor must state so in the auditor's report.
- (d) The auditor must either express an opinion regarding the financial statements, taken as a whole, or state that an opinion cannot be expressed in the auditor's report. When the auditor cannot express an overall opinion, the auditor should state the reasons, therefore, in the auditor's report. In all cases where an auditor's name is associated with financial statements, the auditor should clearly indicate the character of the auditor's work, if any, and the degree of responsibility the auditor is taking, in the auditor's report.

The field of accounting is comprised of a number of sub-disciplines, among them financial accounting and auditing. Financial accounting deals with the collecting, summarizing, and reporting of firm data on economic transactions relevant to government-regulated financial reporting. Auditing provides an independent verification of the "fairness" (a term with a unique definition in the context of accounting) of financial reports. Audit procedures are dictated by two objectives:

1. Cost efficiency in data collection and analysis, and
2. "Fairness" in reporting; this is generally interpreted as the absence of "material error" in the financial reports.

The primary tasks of auditing concerned with statistical tests fall into two broad categories, typically accomplished at different times of the year. The first set of audit tasks investigates whether the applied accounting procedures of the firm's individual transactions comply with firm procedures, government regulation, legal formality, and industry practice for internal control. Statistical tasks to determine the rate of procedural errors of a population of transactions are called compliance tests.

The second set of tasks—called substantive tests of details—verifies that reported monetary summarizations (i.e., the "account Balances") of economic transactions are not "materially" misstated. As with "fairness," "materiality" is an accounting term with a unique definition in the context of a specific audit. An audit considers an error to be material if its magnitude "is such that it is probable that the judgment of a reasonable person relying upon the report would have been changed or influenced by the inclusion or correction of the item" (Financial Accounting Standards Board, 1980). The Securities Exchange Act of 1934 gave the SEC the authority to promulgate financial reporting standards (generally accepted accounting principles or GAAP) for companies subject to the jurisdiction of the SEC. The SEC, in turn, delegated this authority to the FASB (paragraph 132 of SFAC2).

In the USA, Statements on Auditing Standards (SAS) provide guidance to external auditors on generally accepted auditing standards (GAAS). The SAS are codified in the sections of the AICPA's professional standards indexed with "AU" (for Audit). Financial auditing standards related to statistical analysis are described in Statement on Auditing Standards Number 39 (SAS #39) elaborated upon in SAS # 43, #45, and #111, and summarized in AU 350 (AICPA, 2013). Current auditing standards set by the American Institute of Certified Public Accounts do not mandate the use of statistical sampling when conducting audit tests.

However, there is wide agreement that statistical testing allows effective management and trade-offs in attaining the twin objectives of cost efficiency and reporting fairness. The utility of statistical tests was recognized even before the creation of the Securities and Exchange Commission in 1934 (Carman 1933).

Types of Sampling Allowed or Discussed by AICPA

AICPA pronouncements on sampling include risk assessment standards (SAS Nos. 104–111). This edition of the guide has been conformed to reflect the Defining Professional Requirements standard (SAS No. 102), and includes guidance on audit documentation (SAS 103), and communicating internal control related matters (SAS 112). They also include material from the AICPA Audit and Accounting Manual, published June 1, 2011, which emphasizes applications by practitioners when applying audit sampling including various tables and guidance on internal controls, illustrative confirmation letters, illustrative engagement and representation letters, and illustrative auditor's reports. The following subsections provide a brief introduction to the idiosyncratic terminology and the main statistical or quasi-statistical concepts introduced with these AICPA pronouncements.

Judgmental Sampling

Judgmental sampling is a selection process where the auditor decides which items should be audited. It involves a subjective selection of items for testing and a subjective evaluation of the results. It “is an acceptable method of selection provided the auditor is satisfied that the sample is not unrepresentative of the entire population” (APB, 1993). Unfortunately, differences in individual auditor’s ability, knowledge, experience, and prejudices; Pressure on the auditor to reduce the client’s cost of the audit, and; the auditor’s state of physical and mental health all influence judgmental sampling. Statistically valid sampling is also more influential in court.

Fixed-Interval Sampling

In fixed-interval sampling, you specify an interval and a random starting point, which must be greater than zero and less than or equal to the selected interval. When you use fixed-interval sampling, you must be conscious of patterns in the data. Because a fixed interval is used for sample selection, a decidedly nonrepresentative sample can be drawn if the data has a pattern that coincides with the interval you specify.

Cell or Random-Interval Sampling

Like fixed-interval sampling, cell or random-interval sampling is an interval selection method. In cell sampling, you specify an interval and a random seed, which is the basis for a series of pseudo-random numbers, the random seed can be any value.

Random Sampling

In random sampling, each item in the population has an equal, random chance of being selected. Typically the sample is small enough that the auditor need not worry about replacement. Depending on the sampling method, selected items might be set aside once they are drawn or put back into the stream to be drawn again. Because the size of audit samples is typically several orders of magnitude smaller than the population, there is in practice no difference between these two methods.

Conditional Sampling

You can apply a condition to subject only a portion of the population to selection. You might want to use conditional sampling to limit the scope of selections or exclude particular items. Conditions may exclude particular subdivisions of the firm, where those subdivisions are audited separately.

Stratified Sampling

Stratified sampling is a particular form of conditional sampling—conditioned on item size (in dollars, etc.). Especially in substantive testing of account balances, it is normal to test large dollar transactions more closely than smaller transactions. Typically a stratification of the sample will involve two or three strata or groupings of transactions—large to small. For example, large transactions (where each is over \$1000) might test a 100% sample; smaller transactions (less than \$1000) will only test 20% of the population. Monetary unit sampling is a way of continuously stratifying a sample by conditioning the dollar amount of the transaction.

Stratification is a process of dividing a population into subgroups, each of which is a set of sampling units with similar characteristics. Stratification of accounting populations is usually based on the recorded book value amounts of the line

items; the population is divided into groups (strata) according to their book values, and a sample is selected independently from each stratum.

The small samples used in audit contexts, combined with the idiosyncrasies of accounting data (greater than zero, often with a high number of zero balance transactions), may lead to unreliable confidence intervals when the populations have low error rates and when the distribution of transaction dollar values is highly skewed.

Monetary Unit Sampling

With monetary unit sampling, the data is treated as a stream of dollars. Each dollar has an equal chance of selection, and when a dollar is selected, the item containing the dollar is output. The population is the absolute value of all the dollars from the selected field in the table. The interval you specify is the number of dollars between each selection.

Given this sampling base, a \$1000 item is 1000 times more likely to be selected than a \$1 item because a \$1000 item contains 1000 times as many dollars, each of which is equally liable to be selected. This selection method biases the sample toward greater dollar items. MUS is most useful when the sampling units vary considerably in size (Tsui et al. 1985; Stringer 1979; Matsumura and Tsui 1982).

The idea behind “taintings” in Dollar-Unit Sampling (DUS)—also called Monetary Unit Sampling (MUS)—is that each transaction is a collection of discrete monetary units that can contain a percentage of the total error—it is an allocation of a total error to each monetary unit.

More generally, when an error impacts the account balance, we are concerned about the economic magnitude of that error. Accountants using the monetary unit sampling paradigm describe the economic magnitude of a transaction in terms of a count of dollars. If the transaction is X dollars in error, and the stated transaction contribution to the total account balance before auditing is Y dollars, then each of that transaction dollars is said to be ‘tainted’ by X/Y times 1/100 percent. This may sound convoluted, but actually makes sense when trying to construct an estimation model for the account balance in total when inferring from the sample.

In dollar-unit sampling, dollar amounts are typically considered to be only partially in error; the amount of error for a particular item is referred to as the error tainting. For example, a \$100 item that is totally in error has a 100% tainting, whereas a \$100 item that actually should have a value of \$93 has a 7% tainting. The maximum tolerable tainting is the sum of all the individual error taintings.

This formulation is a bit convoluted, and is a product of the MUS sampling approach that conditions on transaction value. It helps to remember that MUS is merely a continuous form of stratified sampling. Auditors conflate the sampling method with the analysis of sample results, leading to the creation of this new variable—tainting.

Current auditing standards set by the American Institute of Certified Public Accountants (AICPA) do not mandate the use of statistics in conducting audit tests. However, in an age in which nearly all accounting data is contained on computer databases, often out of control of firm management (e.g., on public clouds such as Salesforce.com offers for sales; or Amazon Web Services for logistics and inventory control), the use of automated data mining and statistical analysis packages provides the only route to cost-effective auditing.

Early applications of statistical analysis to audit were generally restricted to compliance tests, where transaction processing was found to either be in or out of compliance with accounting policy and procedures. Kenneth Stringer, an employee with Deloitte Haskins & Sells, began investigating audit applications of statistics with the objective of valuing accounts. Stringer consolidated his innovations into methodology recommendations (Stringer 1975, 1963, 1961; Bickel 1992; Pap and van Zuijlen 1995). The eponymous Stringer bound to determine confidence bounds for determining whether specific accounts were or were not “materially” in error. Stringer confidence limits were widely enough invoked to garner interest from the statistical community, which generally concluded that they tended to be much too conservative in the sense that they tended to find material error where accounts are fairly stated.

Perhaps the most important contribution of Kenneth Stringer to auditing was getting auditors to think of account valuations, and the errors in them, as discrete rather than continuous measures of wealth and obligations. It is natural to think in terms of the discrete occurrence of errors (an error either does or does not occur). Stringer’s emphasis on dollar-unit sampling (DUS) introduced a view that accounts are collections of individual dollars, some of which are “accurate” and some of which are “tainted” (i.e., in error and should not be in the account balance).

Stringer applies an urn model to account balances on the financial statements. Each account is an aggregation of individual transactions (think of an urn filled with balls), each with their own value. A transaction is either in error or it is not. The “error condition” of the transaction can be modeled as a [0,1] binary variable, and these errors should be comparatively rare after we complete our midyear compliance testing.

In the past auditors have adopted the Stringer bound to adjust confidence limits to account for accounting transaction idiosyncrasies. But this approach is seldom used, as it leads to conservative bounds with unknown small sample properties; it also tends to increase the needed sample size, and thus the cost of auditing.

Transaction or Record Sampling

A transaction sampling treats each recorded transaction equivalently. This results in a sample that is not biased by the values in a record—each transaction in the population has an equal chance of being selected. Transaction sampling expresses risk of error in terms of a percentage of a population for which a particular type of error exists—it is usually expressed as error rate. Transaction sampling is used in internal control tests because we are interested in determining “level of control” over particular types of errors—these will be expressed as error rates.

Internal control tests The purpose of internal control or compliance testing is to determine to what extent the system's internal controls are complying with the stated policies, plans, laws, and regulations.

Internal controls are a set of procedures that are designed to minimize the chance of errors in the operation of the accounting system. APB's auditing guideline defines “the whole system of controls, financial and otherwise, established by the management to carry on the business of the company in an orderly and efficient manner, ensure adherence to management policies, safeguard the assets, and secure, as far as possible, the accuracy and reliability of its records.”

Internal control compliance tests are designed to establish to what extent the controls can be relied on to detect material error and whether the internal controls were operating effectively throughout the period being audited.

Three approaches are taken to sampling and testing internal controls:

1. Estimation sampling: a random sample of items of a specified size is selected, and the proportion of errors of non-compliance is estimated to establish if it is less than some acceptable level. This is the most widely used approach to compliance testing.
2. Acceptance sampling: enables the auditor to reject or accept the population under certain conditions. A sample of a given size is drawn and if more than a certain amount of errors is found, the system is accepted, otherwise it is rejected. The auditor using acceptance sampling seeks to balance out the risks of rejecting “satisfactory” systems (frequently involving further audit costs) and of accepting “unsatisfactory” populations (exposing the auditor to the potential risk of giving an inaccurate clean audit opinion).
3. Discovery sampling: a sampling plan which selects a sample of a given size, accepts the population if the sample is error-free, and rejects the population if it contains at least one error. With discovery sampling the auditor may not be interested in determining how many errors there are in the population.

Non-statistical Sampling

The AICPA states that non-statistical sampling can be used with tests of controls or substantive tests. Non-statistical sampling does not require the use of a probabilistic selection method. The main advantage of non-statistical sampling is that it is less complex and less time consuming than statistical sampling. The main disadvantage is that sampling theory cannot be used to quantify sampling risk. The AICPA states that sample size for non-statistical sampling is left entirely to the auditor's professional judgment. The factors considered when determining sample size for tests of controls using a non-statistical approach are the same as those considered for attributes sampling. The factors considered when determining sample size for substantive tests using a non-statistical approach are:

1. Risk of incorrect acceptance represents the risk that the auditor concludes that a material misstatement does not exist when, in fact, a material misstatement does exist. The level used for this risk is based on the auditor's planned detection risk and other planned substantive tests. A higher risk of incorrect acceptance is used with a higher planned detection risk and/or other planned substantive tests. This risk is inversely related to sample size.
2. Risk of incorrect rejection represents the risk that the auditor concludes that a material misstatement exists when, in fact, a material misstatement does not exist. The level used for this risk is based on the cost and difficulty of obtaining additional evidence. A lower risk of incorrect acceptance is used when more costly or difficult evidence will be required if expanded testing is needed. This risk is inversely related to sample size.

3. Expected Misstatement represents the auditor's best estimate of the population misstatement. This estimate is normally based on prior experience with the client. This estimate is directly related to sample size.
4. Tolerable Misstatement represents the highest misstatement that could occur before the population would be considered materially misstated. This amount has an inverse relationship with the sample size.

Sample results for tests of controls are evaluated by comparing the sample deviation rate to the tolerable deviation rate and calculating an allowance for sampling risk. The sample deviation rate is calculated by dividing the number of sample deviations by the sample size. The allowance for sampling risk is calculated by subtracting the sample deviation rate from the tolerable deviation rate. If the allowance for sampling risk is large and positive, the auditor would most likely conclude that the design and operation of an internal control is effective. However, if the allowance for sampling risk is small or negative, the auditor would conclude that the design and operation of an internal control is not effective. What constitutes a large enough difference is a matter for professional judgment. Generally, smaller allowances for sampling risk are tolerated with higher risks of assessing control risk too low and larger sample sizes.

Sample results for substantive tests are evaluated in a similar manner. A projected population misstatement is calculated based on the sample results and compared to the tolerable misstatement. The projected population misstatement is computed by dividing the sample misstatement by the dollar value of the sample and multiplying this amount by the dollar value of the population. The difference between the projected population misstatement and tolerable misstatement is called the allowance for sampling risk. If the allowance for sampling risk is large and positive, the auditor would most likely conclude that a material misstatement does not exist. However, if the allowance for sampling risk is small or negative, the auditor would conclude that a material misstatement does exist. What constitutes a large enough difference is a matter for professional judgment. Generally, smaller allowances for sampling risk are tolerated with higher risks of incorrect acceptance and larger sample sizes.

Accounting Transaction Distributions

Probability distributions were proposed originally by Blaise Pascal as a way of understanding the mechanics of gambling. Pascal took uncertain measurements, i.e., random variables, and mapped them into two dimensions—the random variable's value (x-axis) and the probability of how frequently it takes on that value (y-axis). Probability distributions were the basis for statistical inference. When one wishes to infer conclusions from some set of data, the data are assumed to obey some probability distribution. Choice of the correct distribution for a specific problem is one of the most important aspects of effective inference from data.

In addition to choosing the right distribution, auditors need to determine how much data to collect. Data collection and processing in auditing is expensive—audit costs for a single sampled transaction may be hundreds or thousands of dollars. The total cost of an audit is extremely sensitive to the right choice of distributions, sample sizes, a methodology for inference, and decision models. Auditing that is cavalier with methods, relying too much on personal inference, will drive up audit costs with a high probability of rendering incorrect opinions. These problems have created huge and embarrassing failures for audit firms over the past two decades—such failures drove Arthur Andersen into bankruptcy.

Sample size determination is one of the most confusing areas of statistics. Ideally, we would not choose a sample size “up front,” but would continue sampling until a particular objective is met (e.g., until we determine that the system is either in or out of control). This is called “optimal stopping” and is often used in laboratory and factory sampling.

Where we need to do a “one-shot” sample, we need to know the distribution of the parameter we are estimating—this is usually impossible to do accurately, but that does not stop us from trying. Typically a sample distribution can be assumed to be Normal for systems that involve sums and linear operations as do accounting systems. This may often be a bad assumption in auditing, since all audited distributions should be censored at zero and are usually right-skewed and platykurtic, attributes will have a discrete distribution, often with a limited set of “valid” values. The auditor cannot assume that “one-size-fits-all” in sample size determination. There are three factors that need to be considered:

1. Decision effect size: for attribute error rates, this is “tolerable Error”; for value estimation, this is the proportion of materiality assigned to an account, also called “tolerable error.”
2. Assumed population distribution and parameters: if the assumption is normal, then it is expected that the auditor will choose values for mean and variance. Another good assumption is that the attribute errors or the dollar errors are Poisson distributed with mean, variance, skewness, and kurtosis all depending on one parameter and more closely resembling an accounting population distribution.

3. “Confidence” in our decision that, i.e., that the error rate observed in the sample would cause us to change our audit decision. Note that confidence is typically stated through two parameters, which are usually set to where (1) is called significance of the test, and is the probability of making a Type I error; and (2) is called the power of the test (do not blame me; this terminology was concocted by British polymath Karl Pearson in the early twentieth century).

Sample size for hypothesis testing is typically determined from a critical value that defines the boundary between the rejection and non-rejection regions. The minimum sample size that can differentiate between and occurs where the critical value that is exactly the same under the null and alternative hypotheses. The approach to computing sample size here is analogous to standard univariate calculations but using a formulation for variance customized to this problem (Cochran 2007; Kish and Frankel 1974; Snedecor and Cochran 1956; Westland and See-To 2007).

Compliance and substantive errors in transactions and financial accounts are relatively rare events—if they were not, then financial statements would be unusable, and the scope of auditing would be unmanageable. The “law of rare events” assumes an interval in time—e.g., an accounting period—where events happen at random with a known average number. If the interval is divided into subintervals of equal size, the probability that an event will fall in a particular subinterval converges to a Poisson distribution. The Poisson distribution is also sometimes called the law of small numbers (after the title of an 1896 book by Ladislaus Bortkiewicz) because it is the probability distribution of the number of occurrences of an event that happens rarely but has very many opportunities to happen (Fienberg et al. 1977; Repin et al. 2004; Keenan and Kotula 2004; Garstka 1977; Greene 1994; Watts and Leftwich 1977).

The Audit Cycle

The audit cycle encompasses the steps that an auditor will take to ensure that the company’s financial information is valid and accurate before releasing any financial statements. The audit cycle can call for different tasks to be performed at different times—for example, inventory can be counted in October, and account receivables will be determined in November.

The audit cycle typically involves several distinct steps and may include the identification process, where the company meets with auditors to identify the accounting areas that need to be reviewed; the audit methodology stage, where the auditors decide how the information will be collected for review; the audit fieldwork stage, where the auditors test and compare accounting samples; and the management review meeting stage, where the findings are presented by the auditors to the company’s management team.

The audit cycle is the accounting process auditors use to review a company’s important financial information. Companies use audits to verify their financial information is accurate and valid prior to releasing financial statements. Internal audits may be conducted by the company’s accounting staff for the purpose of management review. External audits are usually conducted by public accounting firms or private certified public accountants (CPA) to provide an objective opinion on the company’s financial and accounting processes. The audit cycle usually includes several steps, such as the identification process, audit methodology, fieldwork, management review meeting, and the remedial audit process. The identification process of the audit cycle determines the accounting processes needing to be reviewed per the company’s direction. This identification process usually involves company managers meeting with auditors and discussing the highest risk areas needing to be audited in the financial or accounting department. Company management will also discuss the goals needing to be achieved during the audit cycle. Once this step is complete, auditors will move into the audit methodology stage.

The audit methodology is stage helps auditors determine how they will collect information for review during the audit. Auditors may decide to interview company employees to determine how well they are trained and understand their role in the accounting process. Auditors may also request certain financial information from the accounting department that will be reviewed during the audit cycle’s fieldwork stage.

An audit is an accounting procedure under which the financial records of a company or individual are closely inspected to ensure that they are accurate. Many American taxpayers fear an Internal Revenue Service audit, while dishonest companies fear independent audits of their business practices which may reveal embezzlement and other misuses of funds. This review keeps a company honest and also reassures employees and investors as to the financial status of the organization. There are two primary types: internal and independent audits.

Regardless of the type of audit, it should be assumed that the procedure will be performed without bias. In the case of an internal audit, this can be difficult, because it is carried out by the accounting staff of the company concerned. Generally, this type can only successfully be carried out by a large accounting department, because auditors cannot audit records to which they contributed. Internal audits are usually carried out on a regular basis by large companies to ensure that their finances are in order, and if the company is publicly traded, the reports are available for inspection by stockholders.

An independent or external audit is carried out by a neutral third party, such as a professional accounting firm that specializes in the procedure. In both cases, all of the financial records of a company, including ledgers, bank statements, payroll, tax information, internal financial reports, official published reports, accounts payable, and accounts receivable, will be examined. During the audit, these records are closely inspected for any discrepancies, and if an inaccuracy is uncovered, it must be addressed and repaired.

Commonly, an audit will reveal a simple accounting mistake. In other cases, more sinister issues may come to light. Companies that are struggling financially may choose to make unsound financial decisions in an attempt to salvage the company, and these decisions will be revealed by a close audit. Sometimes the review will reveal that a company is on the brink of bankruptcy due to gross misuse of funds by high ranking personnel, as was the case with many American corporations in the early twenty-first century such as Enron and WorldCom.

When an inaccuracy is revealed by an independent audit, it is addressed by the auditors in the final report made to the company. In some cases, the review will be ordered by an external organization, such as the Securities and Exchange Commission, which will also receive a copy of the report. The issue must be repaired by the company. Common examples of repairable errors are failure to pay payroll taxes to the Internal Revenue Service, or misuse of pension plans. If the errors cannot be fixed because the company does not have the funds to address them, the company may face bankruptcy proceedings, and major creditors will be reimbursed after the company's assets are liquidated by an independent firm.

The Context of Auditing and Information Technology

An audit is the examination and statement of accounts and of other documents connected with accounts by third parties (individuals or firms) who have had no part in their preparation. Systems of financial inspection have long been used, especially in connection with public accounts. In Italy, the elaboration of commerce considerably increased the duties of an auditor in the late Middle Ages, but the auditing of business accounts did not become common until the 19th cent., when there was an increasing number of businesses continually growing in size and complexity. Corporate charters usually came to be granted only on the condition that licensed experts conduct annual audits. Such audits are particularly useful to the owners (partners or stockholders); executives (managers, officers, and directors); creditors or prospective creditors (investors, note brokers, and commercial and investment bankers); and receivers, trustees, and creditors' committees of a business. Audits are also useful to the vendors of a firm's merchandise, the owners of patents and other recipients of profit shares or royalties, governmental regulatory bodies, and prospective donors to institutions. An audit settles certain categories of questions. It must determine whether all assets and liabilities shown are actual, and that they are properly incurred, valued, and recorded. A check must be made of the surplus, income, and capital-stock accounts, verified by examining the authorizations for stock issues and comparing the amounts issued with the amounts authorized. Finally, auditing constitutes an independent check on the tendency to overstate assets and understate liabilities. The duties of auditors have even expanded into a comprehensive survey and analysis of the entire conduct of the financial and accounting branches of an enterprise. Thus the auditor needs, in addition to his knowledge of accounting, a broad understanding of business and finance. The accountant records the facts of a business; the auditor must determine whether or not such recording has been done accurately and honestly and then interpret and judge the facts, perhaps adding to his report recommendations for the future conduct of the business. In many countries, auditors are now established as a separate profession, requiring government licensing. In the USA, private audits are usually performed by certified public accountants; auditing of the federal government's accounts is conducted by Congress's Government Accountability Office (GAO). Formerly the General Accounting Office, it was established in 1921. The Internal Revenue Service periodically audits individual and corporate tax returns. The Public Company Accounting Oversight Board (established 2002) registers and regulates accountants and accounting firms that act as auditors.

It is quite unusual today, even with small and medium sized firms, to find accounting records that are not primarily retained on computer systems. Software and cloud platforms (e.g., SalesForce.com, for example, with sales accounting) have made it simple to automate, as well as uneconomic and risky, not to automate. The accounting systems comprise a subset of the firm's information system dedicated to processing financial transactions to provide:

1. internal reporting to managers for use in planning and controlling current and future operations and for non-routine decision-making; and
2. external reporting to outside parties such as to stockholders, creditors, and government agencies.

Methods, procedures, and standards followed in accumulating, classifying, recording, and reporting business events and transactions. The accounting system includes formal records and original source data. Regulatory requirements may exist on how a particular accounting system is to be maintained (e.g., insurance company).

From the auditor's perspective, the scope of the audit may be limited to the firm's internally implemented applications systems, and will depend on third parties to audit externally supported software and hardware. These external firms should publish a Statement on Auditing Standards No. 70 audit report (commonly abbreviated as SAS 70 audit report). SAS 70 provides guidance to third-party service auditors when assessing the internal control of a service organization and issuing a service auditor's report. SAS 70 also provides guidance to auditors of financial statements of an entity that uses one or more service organizations. Service organizations are typically entities that provide outsourcing services that impact the control environment of their customers. Examples of service organizations are insurance and medical claims processors, trust companies, hosted data centers, application service providers (ASPs), managed security providers, credit processing organizations, and clearinghouses.

There are two types of service auditor reports:

1. Type I service auditor's report includes the service auditor's opinion on the fairness of the presentation of the service organization's description of controls that had been placed in operation and the suitability of the design of the controls to achieve the specified control objectives.
2. Type II service auditor's report includes the information contained in a Type I service auditor's report and also includes the service auditor's opinion on whether the specific controls were operating effectively during the period under review.

Auditors' Opinion: The Product of an Audit

Report rendered by the independent CPA at the end of an audit investigation. The auditor reports on the nature of his or her work and the degree of responsibility assumed. In the audit opinion, the auditor states that he or she has examined the client's financial statements for the year then ended in accordance with generally accepted auditing standards (GAAS), including tests of the accounting records and other necessary auditing procedures. The auditor then indicates whether, in his or her opinion, the client's financial statements present fairly the financial position, results of operations, and changes in financial position for the year-ended in conformity with Generally Accepted Accounting Principles (GAAP) applied on a consistent basis. The four types of audit opinions are unqualified, qualified, adverse, and disclaimer:

1. Unqualified (aka "Clean")
 - (a) "We believe these financial statements are (1) fairly presented in (2) accordance with GAAP (3) consistently applied";
 - (b) Normal result of an audit;
 - (c) When problems are later found, there is a restatement and explanation.
2. Qualified
 - (a) "We believe these financial statements are (1) fairly presented in (2) accordance with GAAP (3) consistently applied; except for (1) {List Exceptions};"
 - (b) Qualified opinions may result from legitimate operations;
 - (c) Qualifications are often are a result of "contingent claims": e.g., large class-action lawsuits such as Monsanto dealt with for Asbestos.
3. Adverse
 - (a) "We DO NOT believe these financial statements are (1) fairly presented in (2) accordance with GAAP (3) consistently applied; THE AUDITORS DISAGREE WITH {List Exceptions};"
 - (b) Adverse opinions should be relatively rare and results from a breakdown of communications with the client.

In addition, auditors may refuse to render an opinion, issuing a disclaimer of opinion. Audit opinion, especially interim testing, also provides the basis for the Sarbanes-Oxley disclosure required of publicly traded companies. The Public Company Accounting Reform and Investor Protection Act of 2002 (commonly called SOX or SarBox) is a US federal law passed in response to a number of major corporate and accounting scandals, including those affecting Enron, Tyco International, and WorldCom (now MCI). These scandals resulted in a decline of public trust in accounting and reporting practices. It was named after sponsors Senator Paul Sarbanes (D-Md.) and Representative Michael G. Oxley (R-Oh.). The act was approved by the House by a vote of 423-3 and by the Senate 99-0. The legislation is wide-ranging and establishes new or enhanced standards for all US public company Boards, Management, and public accounting firms. The Act contains 11 titles, or sections, ranging from additional Corporate Board responsibilities to criminal penalties, and requires the Securities and Exchange Commission (SEC) to implement rulings on requirements to comply with the new law. It established a new quasi-

public agency, the Public Company Accounting Oversight Board, which is charged with overseeing, regulating, inspecting, and disciplining accounting firms in their roles as auditors of public companies. The Act also covers issues such as auditor independence, corporate governance, and enhanced financial disclosure.

References

- Bickel, Peter J. 1992. Inference and Auditing: The Stringer Bound. *International Statistical Review/Revue Internationale de Statistique* 60 (2): 197–209.
- Carman, Lewis A. 1933. The Efficacy of Tests. *The American Accountant* 18: 360–366.
- Cochran, William G. 2007. *Sampling Techniques*. Hoboken: John Wiley & Sons.
- Cohen, Jacob. 1992. A Power Primer. *Psychological Bulletin* 112 (1): 155.
- Fienberg, Stephen E., John Neter, and Robert A. Leitch. 1977. Estimating the Total Overstatement Error in Accounting Populations. *Journal of the American Statistical Association* 72 (358): 295–302.
- Garstka, Stanley J. 1977. Models for Computing Upper Error Limits in Dollar-Unit Sampling. *Journal of Accounting Research* 15 (2): 179–192.
- Greene, William H., Accounting for Excess Zeros and Sample Selection in Poisson and Negative Binomial Regression Models (March 1994). NYU Working Paper No. EC-94-10, Available at SSRN: <https://ssrn.com/abstract=1293115>
- Keenan, Michael R., and Paul G. Kotula. 2004. Accounting for Poisson Noise in the Multivariate Analysis of Tof-Sims Spectrum Images. *Surface and Interface Analysis: An International Journal Devoted to the Development and Application of Techniques for the Analysis of Surfaces, Interfaces and Thin Films* 36 (3): 203–12.
- Kish, Leslie, and Martin Richard Frankel. 1974. Inference from Complex Samples. *Journal of the Royal Statistical Society: Series B (Methodological)* 36 (1): 1–22.
- Matsumura, Ella Mae, and Kam-Wah Tsui. 1982. Stein-Type Poisson Estimators in Audit Sampling. *Journal of Accounting Research* 20 (1): 162–170.
- Meyer, Breed D. 1995. Natural and Quasi-Experiments in Economics. *Journal of Business & Economic Statistics* 13 (2): 151–61.
- Pap, Gyula, and Martien C.A. van Zuijlen. 1995. The Stringer Bound in Case of Uniform Taintings. *Computers & Mathematics with Applications* 29 (10): 51–59.
- Raiffa, Howard, and Robert Schlaifer. 1961. *Applied Statistical Decision Theory*. Boston: Clinton Press, Inc.
- Repin, Sergey, Stefan Sauter, and Anton Smolianski. 2004. A Posteriori Error Estimation for the Poisson Equation with Mixed Dirichlet/Neumann Boundary Conditions. *Journal of Computational and Applied Mathematics* 164: 601–612.
- Snedecor, George W., and W.G. Cochran. 1956. *Statistical Methods Applied to Experiments in Agriculture and Biology*. Iowa: Iowa State College Press.
- Stigler, Stephen. 2008. Fisher and the 5% Level. *Chance* 21 (4): 12–12.
- Stringer, Kenneth W. 1961. Some Basic Concepts of Statistical Sampling in Auditing. *Journal of Accountancy (Pre-1986)* 112 (000005): 63.
- _____. 1963. Practical Aspects of Statistical Sampling in Auditing. In *Proceedings of the Business and Economic Statistics Section*, 405–411. Washington, DC: American Statistical Association.
- _____. 1975. A Statistical Technique for Analytical Review. *Journal of Accounting Research* 13: 1–9.
- _____. 1979. Statistical Sampling in Auditing. The State of Art. *The Accounting Review* 1: 113–27.
- Tsui, Kam-Wah, Ella Mae Matsumura, and Kwok-Leung Tsui. 1985. Multinomial-Dirichlet Bounds for Dollar-Unit Sampling in Auditing. *The Accounting Review* 60 (1): 76–96.
- Watts, Ross L., and Richard W. Leftwich. 1977. The Time Series of Annual Accounting Earnings. *Journal of Accounting Research* 15 (2): 253–271.
- Westland, J. Christopher, and Eric Wing Kuen See-To. 2007. The Short-Run Price-Performance Dynamics of Microcomputer Technologies. *Research Policy* 36 (5): 591–604.

Interim Compliance Tests



Interim Compliance Tests and the Management Letter

Compliance tests determine whether the firm's transaction processing is in compliance with generally accepted accounting principles (GAAP). They are usually performed about 9 months into the client's fiscal year because generally accepted auditing standards (GAAS) suggest that assessment of internal control be based on at least 9 months of transactions. They are therefore called *interim tests* (versus *year-end tests*).

Compliance tests are the bridge between the audit tasks prescribed by the risk assessment matrix (RAM) and the substantive tests that ultimately support the audit opinion. The audit tasks prescribe a certain level of evidence collection based on subjective and objective assessments of the risk associated with any given transaction processing through the accounting system. The interim compliance tests verify or refute the assumption that transaction processing is working as assessed in the RAM. If particular accounting control processes are found to be flawed, the compliance testing will:

1. Generate an entry in the SAS No. 115 letter to management, and generate a potential item for management response on the Sarbanes–Oxley letter.
2. Suggest non-audit systems consulting tasks where the audit firm might assist the client.
3. Increase the sample sizes and scope of the substantive year-end tests related to accounts that are affected by the transaction processing in which problems were identified.

The SAS 115 Letter to Management

The AICPA's (American Institute of CPAs) Statements on Auditing Standards require two major communications, the requirements of which are summarized in SAS 114 and SAS 115. This work, in turn, provides a basis for reporting under sections 302 and 404 of the Sarbanes–Oxley Act (SOX). I address SOX reporting in greater depth in Chap. “Sarbanes–Oxley Engagements.”

The SAS 114 letter communicates to those charged with governance, such as the Board of Directors, Audit Committee, President, or Management, the scope of audit procedures performed, significant findings, and other information, such as disagreements with management, audit adjustments, and significant estimates, that are not communicated in the audited financial statements.

The SAS 115 summarizes the auditor's understanding of the client's internal controls, how they are implemented, and their operational effectiveness. While any significant deficiencies or material weaknesses would have been discussed with management during the audit, the AICPA requires the auditor to communicate them to management in writing. Along with identifying deficiencies in the controls, the auditor may also offer recommendations for ways to improve these controls that will help to mitigate risk and strengthen client accounting processes.

The SAS No. 115 letter to management is known by several names: the Internal Control Letter, Letter of Recommendations, and so forth. The contents of the letter are described in AU Section 325 “Communicating Internal Control Related Matters Identified in an Audit.” The following excerpt the salient recommendations of SAS 115:

Deficiencies identified during the audit that upon evaluation are considered significant deficiencies or material weaknesses under this section should be communicated, in writing, to management and those charged with governance as a part of each audit, including significant deficiencies and material weaknesses that were communicated to management and those charged with governance in previous audits and

have not yet been remediated. Significant deficiencies and material weaknesses that previously were communicated and have not yet been remediated may be communicated, in writing, by referring to the previously issued written communication and the date of that communication.

The written communication is best made by the report release date, which is the date the auditor grants the entity permission to use the auditor's report in connection with the financial statements, but should be made no later than 60 days following the report release date.

In an audit of financial statements, the auditor is not required to perform procedures to identify deficiencies in internal control or to express an opinion on the effectiveness of the entity's internal control. Internal control is a process—effected by those charged with governance, management, and other personnel—designed to provide reasonable assurance about the achievement of the entity's objectives with regard to the reliability of financial reporting, effectiveness and efficiency of operations, and compliance with applicable laws and regulations. Internal control over the safeguarding of assets against unauthorized acquisition, use, or disposition may include controls related to financial reporting and operations objectives.

An illustrative written communication of recommendations of SAS 115:

In planning and performing our audit of the financial statements of ABC Company (the "Company") as of and for the year ended December 31, 20XX, in accordance with auditing standards generally accepted in the USA, we considered the Company's internal control over financial reporting (internal control) as a basis for designing our auditing procedures for the purpose of expressing our opinion on the financial statements, but not for the purpose of expressing an opinion on the effectiveness of the Company's internal control. Accordingly, we do not express an opinion on the effectiveness of the Company's internal control.

Our consideration of internal control was for the limited purpose described in the preceding paragraph and was not designed to identify all deficiencies in internal control that might be significant deficiencies or material weaknesses and therefore, there can be no assurance that all deficiencies, significant deficiencies, or material weaknesses have been identified. However, as discussed below, we identified certain deficiencies in internal control that we consider to be material weaknesses [and other deficiencies that we consider to be significant deficiencies].

A deficiency in internal control exists when the design or operation of a control does not allow management or employees, in the normal course of performing their assigned functions, to prevent, or detect and correct misstatements on a timely basis. A material weakness is a deficiency, or a combination of deficiencies, in internal control, such that there is a reasonable possibility that a material misstatement of the entity's financial statements will not be prevented, or detected and corrected on a timely basis.

We consider the following deficiencies in the Company's internal control to be material weaknesses:

[Describe the material weaknesses that were identified.]

A significant deficiency is a deficiency, or a combination of deficiencies, in internal control that is less severe than a material weakness, yet important enough to merit attention by those charged with governance.

We consider the following deficiencies in the Company's internal control to be significant deficiencies:

[Describe the significant deficiencies that were identified.]

This communication is intended solely for the information, and use of management, [identify the body or individuals charged with governance], others within the organization, and [identify any specified governmental authorities] and is not intended to be and should not be used by anyone other than these specified parties.

The auditor gains the understanding of the client's internal controls, how they are implemented, and their operational effectiveness, as required by SAS 115, during preliminary fieldwork while conducting walkthroughs of certain key areas, such as the cash receipt and cash disbursement processes. Where an internal control may not be operating as intended or may not be in place at all, then this deficiency will be conveyed in the SAS 115 letter. The remainder of this chapter describes audit processes during the interim tests and how efficiency and effectiveness can be improved through data science applications implemented in R.

Three Methods of Sampling

There are three broad methodologies for sampling in audits, each with a specific goal:

1. *Discovery* sampling for the discovery of *out-of-control* transaction streams.
2. *Attribute* sampling for estimating transaction error rate.
3. *Acceptance* sampling for determining whether an account balance is "fairly stated."

Acceptance sampling is germane to substantive testing, but generally not to interim testing, so discussion of acceptance testing will be deferred to the next chapter. The following explains how to apply our sample size formulas in the other two methodologies.

Discovery Sampling

One of the most important tools of cost control is “discovery sampling” to identify potential out-of-control situations that require additional compliance tests. Discovery sampling cost-effectively obtains the minimum sample size of transactions for which the occurrence of even a single error will indicate the likelihood of that transaction’s processing being “out of control.”

Compliance testing begins with the output from planning and risk assessment: the Risk Assessment Matrix (RAM) estimates the rate of a given compliance or accounting error in a particular population of accounting transactions. This estimate from the Planning and Analytical review stage is subjective and may often be zero or just a statement that the auditor does not know the error rate.

Compliance testing would be prohibitively expensive if every process control on every transaction flow needed to be tested. Fortunately, auditors can intelligently integrate the results of the prior year’s audits, of interviews with the client and publicly available information from news feeds, and auditors own judgment to control compliance auditing costs.

To understand the decisions involved in discovery sampling, consider a formal articulation of the problem. Let $\{x_i\}$ be an independent and representative sample of transactions from the population of all transactions that took place in the year. Assume that auditors find y errors in the sample and the maximum number of errors that can occur before we consider the process “out-of-control” *rate* is z . Then the “discovery” sample size is the value such that $f\{x_i|y\} \geq z$ for some statistical function f , which will be made explicit below. Note that since z is a rate, altering the accounting period, e.g., from 1 year to 9 months, changes the required sample size. This matters because interim testing often takes place with less than 1 year’s transaction data; audit guidelines require that interim testing audit at least 9 months of transactions before drawing conclusions on whether systems are “in-control.” This resulting decision will consequently determine if the associated transaction processing systems can be relied upon to generate accurate account balances at which are audited in year-end tests.

Discovery sampling for interim tests sets an estimation sample size for *transaction-unit* samples, so that we are likely to discover at least one error in the sample if the actual transaction error rate exceeds the *minimum acceptable error rate* (alternatively called the *out-of-control* rate of error). Discovery testing helps the auditor decide whether the systems processing a particular transaction stream are in or out of control. Figure 1 depicts sample size and statistical confidence in discovery sampling.

```
confidence <- seq(.99, .7, -.01)
n <- (log(1-confidence)) / log(1-.05)
plot(confidence, n, type="l")
```

So for a 5% intolerable error rate at 95% confidence we have

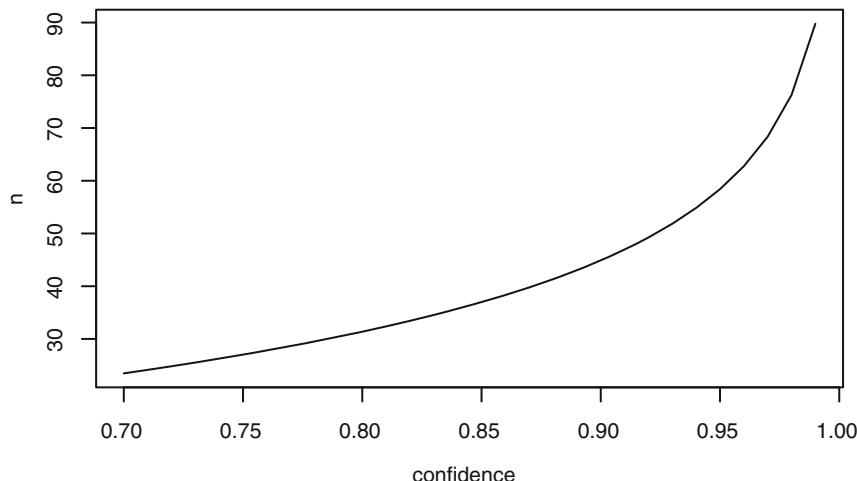


Fig. 1 Confidence and sample size in discovery sampling

```

confidence <- .95
n <- (log(1-confidence))/log(1-.05)
cat("\n Discovery sample size = ", ceiling(n))

##
## Discovery sample size = 59

```

Attribute Sampling

Attribute sampling for interim tests sets estimation sample size for *transaction-unit* samples to estimate the error rate in the entire transaction population, at some confidence level (e.g., 95%), that the estimate is within the *out-of-control* error rate cutoff for that particular transaction stream. If it is found that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Error estimates from attribute samples may either be *rates* or *amounts* or both.

The following code chunk demonstrates the use of the `pwr.t.test` t-test function in R's `pwr` package to compute sample size from the credit sales file generated by code in Chapter 13. Note that parameter inputs that would yield a sample size of less than 1.3 stop and further calculations by R, resulting in an `error` in `uniroot` function termination of the function. In this situation, the software is essentially telling you not to take any samples, and skip this test. This is probably not advisable from a risk perspective. The alternate conclusion would be that one or more of the parameter values chosen for (1) dataset size, (2) detection occurrence rate, (3) variability, or (4) effect size, do not make sense. They should be revised, generally in the following order: (1) variability, (2) detection occurrence rate, (3) effect size, which in interim testing will be the *intolerable* or *out-of-control* error rate.

```

library(readr)
library(pwr)
library(tidyverse)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

Client_sales_journal <- read_csv((system.file("extdata", "real_world_credit_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), col_types = cols(X1 = col_skip()))

size <- as.numeric(nrow(Client_sales_journal))                                ## data set size
Delta <- .05 * size                                                            ## detect 5% occurrence error
sigma <- .3 * size                                                             ## variability (guess ~1/3 size)
effect <- Delta / sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")                                         ## look for overstatement of earnings

cat("\n Attribute sample size for occurrence of error = ", ceiling(sample$n))

## data set size
size <- as.numeric(nrow(Client_sales_journal))  ## data set size
## average value of transaction
mu <- mean(Client_sales_journal$sales_unit_price * Client_sales_journal$sales_count)
## detect 5% amount intolerable error
Delta <- .05 * mu
## variability
sigma <- sd(Client_sales_journal$sales_unit_price * Client_sales_journal$sales_count)
effect <- Delta / sigma

```

```

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater") ## look for sales value too large

cat("\n Attribute sample size for amount of error = ", ceiling(sample$n))

```

Statistics for Interim Tests

Auditors use two types of sampling during interim compliance testing: (1) discovery sampling to discover an *out-of-control* system; and (2) attribute sampling to estimate the error rate in a system.

Discovery sampling sets an estimation sample size for *transaction-unit* samples, so that we are likely to discover at least one error in the sample if the actual transaction error rate exceeds the *minimum acceptable error rate* (alternatively called the out-of-control rate of error). Discovery tests help the auditor decide whether the systems processing a particular transaction stream are in or out of control.

To limit the cost of an audit, discovery samples are selected first to determine whether the client's accounting system is in or out of control. If the system is determined to be out of control, additional evidence is obtained until the auditor has a sample size sufficient for attribute sampling. Attribute sampling sets estimation sample size for *transaction-unit* samples to estimate the error rate in the entire transaction population with some confidence (e.g., 95%) that the estimate is within the out-of-control error rate cutoff for that transaction stream. If it is found that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream.

The profession has never been clear on the significance and power (type I and II error levels) of statistical decisions concerning whether a system is in or out of control. In the 1970s, Statement on Auditing Procedures 54 suggested that significance should probably be less than 0.3 (i.e., confidence limits should be set greater than 0.7) but otherwise has left the choice of type I and II error levels have been left to the auditor's judgment. Fisher suggested the 0.95 confidence level that we commonly used in scientific studies, and generally, levels above 0.9 are safe.

If the expected error rate is s , the intolerable error rate is r , and the error rate in the control system is ϵ , then restate the hypotheses as:

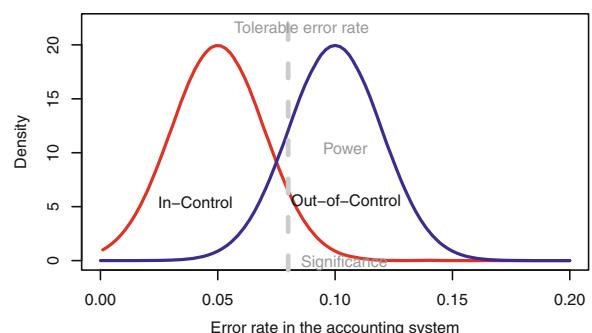
$$H_0 : \epsilon = s \text{ where } \epsilon \in [0, 1]$$

$$H_a : \epsilon \geq r \text{ where } r, \epsilon \in [0, 1].$$

This is depicted in the Fig. 2.

Discovery sampling chooses a sample to determine whether an error rate does not exceed a designated percentage. If the sample does not contain errors, then the actual error rate is assumed to be lower than the minimum unacceptable rate. The sampling calculation requires the auditor to specify the confidence level of the test and the minimum unacceptable error rate using an "urn model" where models are typical stated as draws of colored balls from an urn. Consider, for illustration, the audit of two sets of transactions: sales and subsequent collections for sales on credit. The decision is couched in terms of two assumptions of error distributions in a transaction stream: (1) errors are "in-control" and (2) errors are "out of control."

Fig. 2 Probability model under null (red) and alternative (blue) hypotheses: materiality, power, and significance



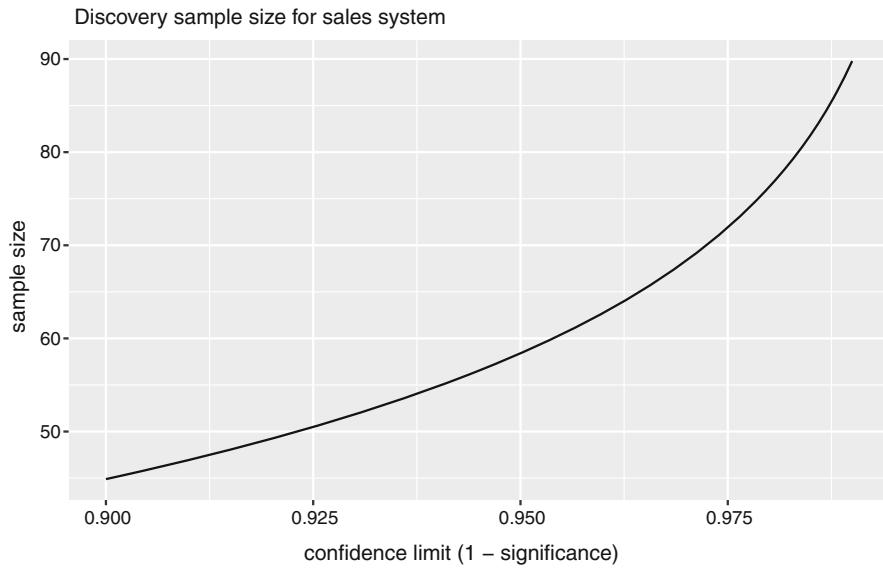


Fig. 3 Discovery sample size and minimum acceptable error rate = 0.05

Assume that the auditor determines that the minimum acceptable error rates are as follows:

- the sales accounting system is “in-control” when less than 0.05 of transactions are in error, and
- the A/R collections system is “in-control” when less than 0.1 of transactions are in error.

For confidence level c we want to choose $n = \frac{\log(1-c)}{\log(1-p)}$

```
library(tidyverse)

c <- seq(.9,.99, length.out = 100)
p <- .05
n <- log(1-c)/log(1-p)

samp <- data.frame(c,n)
ggplot(samp,aes(c,n))+
  geom_line()+
  labs(title="\n Discovery sample size for sales system") +
  xlab("confidence limit (1 - significance)") +
  ylab("sample size")

p <- .1
n <- log(1-c)/log(1-p)
samp <- data.frame(c,n)
ggplot(samp,aes(c,n))+
  geom_line()+
  labs(title="\n Discovery sample size for collections system") +
  xlab("confidence limit (1 - significance)") +
  ylab("sample size")
```

From the graphs, the sales system needs a sample size of 58, and the collections system needs 28 for 0.95 confidence. Start with the sales system to ascertain whether the system is in or out of control. The selection is from the client’s unaudited file, and audit evidence is obtained from the real world, which is reflected in “real_world_” prefixed files generated from the code presented in Chap. “Simulated Transactions for Auditing Service Organizations”. Note that unless you set a random seed, the values will change each time they are generated (Figs. 3 and 4).

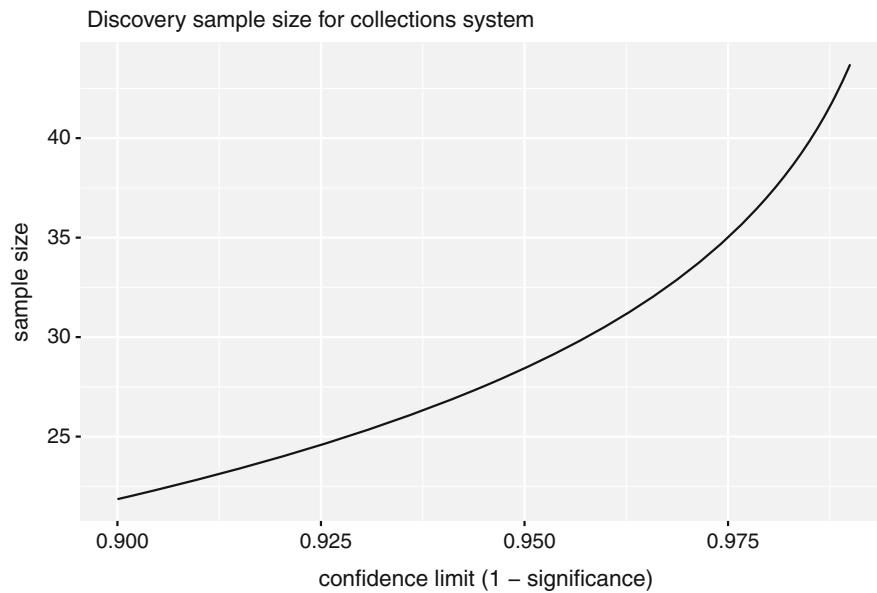


Fig. 4 Discovery sample size and minimum acceptable error rate = 0.01

```

library(tidyverse)
library(compareDF)
library(arsenal)
library(compare)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

sample_size = 58 # set the sample size based on the discovery sample algorithm

real_world_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package =
  "auditanalytics", mustWork = TRUE)),
  na.strings="0", stringsAsFactors=FALSE) %>%
  select(invoice_no,
         sales_unit_price,
         sales_count)

sales_journal <-
  read_csv(system.file("extdata", "sales_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

sales_temp <- split(sales_journal,sales_journal$cash_not_ar)
sales_journal <- sales_temp$`0` ## select only credit sales

discovery_sample_sales <-
  sales_journal[runif(sample_size,1,nrow(sales_journal)),] %>%
  select(invoice_no,
         sales_unit_price,
         sales_count)

```

```

audited_sample <-
  left_join(discovery_sample_sales,
             real_world_sales,
             by="invoice_no") %>%
  select(invoice_no,
         sales_unit_price.x,
         sales_count.x,
         sales_unit_price.y,
         sales_count.y)

exceptions <-
  audited_sample %>%
  filter(sales_unit_price.x != sales_unit_price.y |
         sales_count.x != sales_count.y
        )

error_rate_sales <- nrow(exceptions) / nrow(audited_sample)

cat("\n \n \n Discovery: error rate in sales:", error_rate_sales)

## 
## 
## 
## Discovery: error rate in sales: 0

error_sales_amt <-
  100 * sum(exceptions$sales_count.y * exceptions$sales_count.y) /
  sum(audited_sample$sales_unit_price.x * audited_sample$sales_count.x)

cat("\n \n Discovery: amount for sales sample is overstated (i.e., is in error):",
    prettyNum(error_sales_amt, big.mark=","),
    " %")

## 
## 
## Discovery: amount for sales sample is overstated (i.e., is in error): 0 %

```

If the error rate is above zero (i.e., if the auditor has found more than one exception), additional evidence needs to be collected for an attribute test to ascertain the true rate of errors. First, the auditor needs to consider prior years' audit outcomes (if they exist) in determining whether to proceed to attribute samples. This uses the “rule of threes” to calculate 95% upper confidence bounds. Assume that a review of the audit files shows that the sales transactions have been assessed to be “in-control” over the past 20 audits; then a 95% upper bound on the rate of occurrence is $\frac{3}{n}$. By the law of rare errors, the observed events Y (i.e., no errors in the audit) follow a $Poisson(\lambda)$ distribution using n samples. The sum of Poisson random variables is Poisson, so the question of at least one Y not equal to zero is the probability that the sum $\sum Y_i$ is greater than zero. Let this probability be, for example, 0.95 so that $P[\sum Y_i = 0] = e^{-n\lambda} = 0.05$. Taking logarithms gives $n\lambda = -\ln(0.05) \approx 3$. Thus the rate $\lambda \approx \frac{3}{n}$.

Attribute Sampling with t-Tests

For the sake of discussion, we will assume that we have determined an “out-of-control” sales system situation in discovery sampling. Attribute sampling size is determined using Cohen’s power analysis (Cohen 1992) which is implemented in R’s

pwr package. If discovery sampling suggests that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Since the auditor will have already collected evidence for the discovery sample, the cost of attribute sampling is only that of auditing the increment in sample size recommended by Cohen's pwr formulas. Transaction records already selected for discovery sampling should be combined with additional records drawn for attribute sampling. Error estimates from attribute samples may either be *rates of erroneous transactions* or from a monetary unit sampling perspective, which can be *rates of monetary error in the transaction stream*. I compute both in the following example. As in the prior example, the post-audit information is drawn from the real_world_credit_sales file, which is merely a device in this example; such information would not truly be available to the auditor, rather it is the information gained by analyzing audit evidence. Thus I use an inner_join of the real_world_credit_sales and sales_sample files to mimic the post-audit results for the sample of sales transactions.

```

library(readr)
library(pwr)
library(tidyverse)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

real_world_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package = "auditanalytics",
  mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

sales_journal <-
  read_csv(system.file("extdata", "sales_journal.csv", package = "auditanalytics",
  mustWork = TRUE),
  col_types = cols(X1 = col_skip()))

sales_temp <- split(sales_journal,sales_journal$cash_not_ar)
sales_journal <- sales_temp$`0`


size <- as.numeric(nrow(sales_journal))  ## data set size
Delta <- .05*size                                ## detect 5% occurrence error
sigma <- .3*size                                  ## variability (guess ~1/3 rd)
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")                           ## look for overstatement of earnings

cat("\n \n Attribute sample size for occurrence of error = ", ceiling(sample$n))

##
##
##  Attribute sample size for occurrence of error =  224

size <- as.numeric(sum(sales_journal$sales_unit_price*sales_journal$sales_count))  ## data set size
mu <- mean(sales_journal$sales_unit_price*sales_journal$sales_count)    ## average value of transaction
Delta <- .05*mu                                ## detect 5% amount intolerable error
sigma <- sd(sales_journal$sales_unit_price*sales_journal$sales_count)        ## variability
effect <- Delta/sigma

sample <- pwr.t.test(

```

```

d=effect,
sig.level = 0.05,
power = .8,
type="one.sample",
alternative="greater") ## look for sales value too large

cat("\n\n Attribute sample size for amount of error = ", ceiling(sample$n))

## 
## 
## Attribute sample size for amount of error = 572

```

Since the discovery sample size was 58 that auditor needs to obtain an additional $224 - 58 = 166$ records for attribute tests of occurrence rate, and $636 - 58 = 578$ records for attribute tests of amount rate (what monetary unit sampling field would call “taintings” rate).

```

discovery_sample_sales_add <-
  sales_journal[runif(166,1,nrow(sales_journal)),] %>%
  select(invoice_no,
         sales_unit_price,
         sales_count)

discovery_sample_sales_occ <- rbind(
  data.frame(discovery_sample_sales_add),
  data.frame(discovery_sample_sales))

audited_sample <-
  left_join(discovery_sample_sales_occ,
            real_world_sales,
            by="invoice_no") %>%
  select(invoice_no,
         sales_unit_price.x,
         sales_count.x,
         sales_unit_price.y,
         sales_count.y)

exceptions <-
  audited_sample %>%
  filter(sales_unit_price.x != sales_unit_price.y |
         sales_count.x != sales_count.y
        )

error_rate_sales <- nrow(exceptions) / nrow(audited_sample)

cat("\n \n \n Attribute: error rate in sales: ", error_rate_sales)

```

```

## 
## 
## 
## Attribute: error rate in sales: 0

discovery_sample_sales_add <-
  sales_journal[runif(578,1,nrow(sales_journal)),] %>%
  select(invoice_no,
         sales_unit_price,
         sales_count)

```

```

discovery_sample_sales_amt <- rbind(
  data.frame(discovery_sample_sales_add),
  data.frame(discovery_sample_sales))

audited_sample <-
  left_join(discovery_sample_sales_amt
             , real_world_sales,
             by = "invoice_no") %>%
  select(invoice_no,
         sales_unit_price.x,
         sales_count.x,
         sales_unit_price.y,
         sales_count.y)

exceptions <-
  audited_sample %>%
  filter(sales_unit_price.x != sales_unit_price.y |
         sales_count.x != sales_count.y
        )

error_sales_amt <-
  100 * sum(exceptions$sales_count.y * exceptions$sales_count.y) /
  sum(audited_sample$sales_unit_price.x * audited_sample$sales_count.x)

cat("\n \n \n Attribute: amount sales sample is overstated (i.e., is in error)  ",
    prettyNum(error_sales_amt, big.mark=","),
    " %")

```

```

## 
## 
## 
## Attribute: amount sales sample is overstated (i.e., is in error)  0 %

```

The “out-of-control” critical value is 0.05 error rate, and thus both in occurrence and in amount, the auditor determines the sales system to be “in-control.” Note that we added samples to the original discovery set in order to keep down audit costs, as investigating evidence is the major cost of an audit. The final error rate is based on a count of any differences in values in the client’s records versus the real world. If this exceeds the critical “intolerable” error rate, then the system is judged “out of control.” I repeat the same tests for the accounts receivable collection transactions below.

Audit of Collection Transactions

For the sake of discussion, we will assume that we have determined an “out-of-control” collections system and discovery sample size is determined using the “urn model” formula, and attribute sampling size is determined using Cohen’s power analysis (Cohen 1992) implemented in R’s *pwr* package. If discovery sampling suggests that the collections transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the collection systems that process this transaction stream. Errors estimates from attribute samples may either be *rates of erroneous transactions* or from a monetary unit sampling perspective, can be *rates of monetary error in the transaction stream*.

```

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

library(tidyverse)

```

```

real_world_collections <-
  read.csv(system.file("extdata", "real_world_collections.csv", package =
    "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

ar_collections_journal <-
  read.csv(system.file("extdata", "collections_journal.csv", package =
    "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

c <- .95
p <- .05
sample_size <- ceiling(log(1-c) / log(1-p))

discovery_sample_collections <-
  ar_collections_journal[
    ceiling(
      runif(
        sample_size, 1, nrow(ar_collections_journal))),]

audited_sample <-
  left_join(discovery_sample_collections,
             real_world_sales,
             by="invoice_no") %>%
  select(invoice_no,
         collection_amount.x,
         collection_amount.y)

audited_sample[is.na(audited_sample)] <- 0

exceptions <-
  audited_sample %>%
  filter(collection_amount.x != collection_amount.y)

error_rate_collections <- nrow(exceptions) / nrow(audited_sample)

cat("\n \n \n Discovery: error rate in collections: ", error_rate_collections)

## 
## 
## 
##   Discovery: error rate in collections:  0.1186441

error_collections_amt <-
  sum(exceptions$collection_amount.x - exceptions$collection_amount.y)

cat("\n \n \n Discovery: amount collections sample is overstated (i.e., is in error) $",
  prettyNum(error_collections_amt, big.mark=","))

## 
## 
## 
##   Discovery: amount collections sample is overstated (i.e., is in error)
##   $ 11,832

```

This error rate is above zero (i.e., the auditor has found more than one exception), so we need to attribute sample. Next set attribute sampling size using Cohen's power analysis (Cohen 1992), which is implemented in R's *pwr* package. Since

discovery sampling suggests that the collections transaction processing is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Errors estimates from attribute samples may either be *rates of erroneous transactions* or, from a monetary unit sampling perspective, which can be *rates of monetary error in the transaction stream*. We compute both in the following example.

```

library(readr)
library(pwr)
library(tidyverse)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

real_world_collections <-
  read.csv(system.file("extdata", "real_world_collections.csv", package =
  "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

ar_collections_journal <-
  read.csv(system.file("extdata", "collections_journal.csv", package =
  "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

size <- as.numeric(nrow(ar_collections_journal))    ## data set size
Delta <- .05*size                                     ## detect 5% occurrence error
sigma <- .3*size                                       ## variability (guess ~1/3 rd)
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")                                ## look for overstatement of earnings

cat("\n \n Attribute sample size for occurrence of error = ", ceiling(sample$n))

## 
## 
## Attribute sample size for occurrence of error =  224

size <- as.numeric(sum(
  ar_collections_journal$sales_extended))    ## data set size

mu <- mean(ar_collections_journal$sales_extended)    ## average value of transaction
Delta <- .05*mu                                       ## detect 5% amount intolerable error
sigma <- sd(ar_collections_journal$sales_extended)/3    ## variability
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")    ## look for sales value too large

cat("\n Attribute sample size for amount of error = ", ceiling(sample$n))

```

```
##  
## Attribute sample size for amount of error = 122
```

I mentioned earlier that accounting systems are highly multicolinear. This example illustrates this, as means and standard deviations (as well as the resulting samples) are very close to those of the sales transaction stream. Collections are just sales transactions that are delayed (assuming that customers eventually pay their bills). Since the discovery sample size was 59 that auditor needs to obtain an additional $224 - 59 = 165$ records for attribute tests of occurrence rate, and $122 - 59 = 63$ for records for attribute tests of amount rate (what the monetary unit sampling field would call “taintings” rate).

```
library(tidyverse)  
  
discovery_sample_collections_add <-  
  ar_collections_journal[  
    runif(165,1,nrow(ar_collections_journal)),]  
  
discovery_sample_collections_occ <- rbind(  
  data.frame(discovery_sample_collections_add),  
  data.frame(discovery_sample_collections))  
  
audited_sample <-  
  left_join(discovery_sample_collections_occ,  
            real_world_sales,  
            by="invoice_no") %>%  
  select(invoice_no,  
         collection_amount.x,  
         collection_amount.y)  
  
audited_sample[is.na(audited_sample)] <- 0  
  
exceptions <-  
  audited_sample %>%  
  filter(collection_amount.x != collection_amount.y)  
  
error_rate_collections <- nrow(exceptions) / nrow(audited_sample)  
  
cat("\n \n \n Attribute: error rate in collections: ", error_rate_collections)  
  
##  
##  
##  
## Attribute: error rate in collections: 0.1071429  
  
discovery_sample_sales_add <-  
  sales_journal[runif(63,1,nrow(sales_journal)),]  
  
discovery_sample_collections_amt <- rbind(  
  data.frame(discovery_sample_collections_add),  
  data.frame(discovery_sample_collections))  
  
audited_sample <-  
  left_join(discovery_sample_collections,  
            real_world_sales,  
            by="invoice_no") %>%  
  select(invoice_no,  
         collection_amount.x,  
         collection_amount.y)
```

```

audited_sample[is.na(audited_sample)] <- 0

exceptions <-
  audited_sample %>%
    filter(collection_amount.x != collection_amount.y)

error_collections_amt <-
  sum(exceptions$collection_amount.x - exceptions$collection_amount.y)

cat("\n \n \n Attribute: error amount in collections: $",
  prettyNum(error_collections_amt, big.mark=","))

```

```

## 
## 
## 
## Attribute: error amount in collections: $ 11,832

```

The “out-of-control” critical value is 0.10 error rate for the collections transaction stream, which can be compared to the occurrence and amount to determine the collections systems “out” or “in-control” status. Note that we added samples to the original discovery set in order to keep down audit costs, as investigating evidence is the major cost of interim testing.

Machine Learning Models for Audit of Controls

This section reviews the application of machine learning algorithms for the audit of internal control systems’ effectiveness. The models used here are a class of machine learning called “deep learning”—the “deep” adjective refers to there being multiple layers in the neural networks that comprise the model, not to some deeper esoteric concept of learning. Deep learning models grew out of a half-century of work starting with perceptron models that date to the 1950s, evolving into modern implementations in Tensorflow and PyTorch. Google’s TensorFlow is a leading deep-learning tool, and will be used in the example of deep learning used in this section. The TensorFlow framework is written in Python, and R communicates with it through R’s Python interface `reticulate`. TensorFlow itself relies heavily on BLAS, CUDA, and other code written in various languages, but the main application programming interface (API) is Keras, which is based in Python. R does not always have access to the latest updates to TensorFlow, but most important functions appear in the R implementation of Keras. We will be using R-Keras commands to construct a deep-learning model to predict the existence of errors in accounting control systems.

Machine learning tools are called artificial intelligence in much of the popular press, and their capabilities are too often misrepresented or overstated. Machine learning extends many of the models and concepts pioneered in mathematical statistics over the past two centuries, It’s functionality has leapfrogged statistical capabilities in two areas:

1. Data representations are much richer than the $2 \times 2 : observation \times measurement$ matrix representation that is standard in statistics. Data can be represented as multidimensional tensors (arrays in R terminology), and this allows complex datasets such as video and geographic information to be readily processed in machine learning.
2. Solution concepts and loss functions are much richer in machine learning because computationally intensive methods allow us to compute optima for response surfaces that may not have closed-form representations, and for which differential calculus has no first-order method of finding optima.

TensorFlow requires installation prior to using any of its functions in R. This and other aspects of TensorFlow use are beyond the scope of this text. The reader is directed toward the ample documentation at <https://www.tensorflow.org/> and <https://tensorflow.rstudio.com/> on installation and use.

This section develops a deep-learning autoencoder that assesses anomalies in the client’s transaction files based on pattern matching and dimensionality reduction. To demonstrate, I again use Chapter 13 simulations of a client’s accounting transactions with omissions, duplications, and monetary errors. In this section, I develop a deep- learning model to assess total errors and misstatements in the population (all of the transactions for an audit year) based on the post-audit results of a

sample of transactions. This is a highly unbalanced dataset – in most audits, the portion of transactions without an error will be several orders of magnitude larger than the number with errors.

```
library(readr)
library(tidyverse)
library(compare)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

real_world_cash_sales <-
  read.csv(system.file("extdata", "real_world_cash_sales.csv", package =
    "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

real_world_credit_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package =
    "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

sales_journal <-
  read.csv(system.file("extdata", "sales_journal.csv", package =
    "auditanalytics", mustWork = TRUE), na.strings="0", stringsAsFactors=FALSE)

sales_journal[is.na(sales_journal)] <- 0

sales_journal <- split(sales_journal, sales_journal$cash_not_ar)

credit_sales_journal <- sales_journal`0`
cash_sales_journal <- sales_journal`1`

real_world_credit_sales$sales_extended <- real_world_credit_sales$sales_unit_price *
  real_world_credit_sales$sales_count

credit_sales_journal$cost_extended <-
  credit_sales_journal$unit_cost *
  credit_sales_journal$sales_count

p1 <- hist(credit_sales_journal$sales_extended, breaks=20)
p2 <- hist(credit_sales_journal$cost_extended)

plot( p1, col=rgb(0,0,1,1/4), ylim=c(0,250),
      main = "Cost (orange) vs. Revenue (purple) of Sales")
plot( p2, col=rgb(1,0,0,1/4), add=T, ylim=c(0,250))
```

For an autoencoder to work well, we need a strong initial assumption: that the distribution of variables for normal transactions is different from the distribution for ones that are in error. A simple graphical example provides some insight into how the autoencoder can reveal control weaknesses at a systemic level. In the following code chunk, I create a ridgeplot of column values for the `left_joined` files for sales on the client's books and actual sales, with a factor field for exceptions. The ridgeplots highlight the errors in the client's sales records for `collection_date`, and `sales_unit_price`. Ridgeplots are not particularly refined, but they can provide a quick method for “eyeballing” systemic weaknesses.

```
library(tidyverse)
library(ggrridges)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

real_world_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package = "auditanalytics",
    mustWork = TRUE),
    na.strings="0", stringsAsFactors=FALSE) %>%
```

```

select(invoice_no,
       sales_unit_price,
       customer_no,
       sales_count,
       collection_amount,
       collection_date
     )

real_world_sales$sales_unit_price = real_world_sales$sales_unit_price*rbinom(nrow(real_world_sales), 1, .7)

sales_journal <-
  read_csv(system.file("extdata", "sales_journal.csv", package = "auditanalytics", mustWork = TRUE),
  col_types = cols(X1 = col_skip())) %>%
  filter(cash_not_ar == 0) %>%
  select(invoice_no,
         sales_unit_price,
         invoice_no,
         customer_no,
         sales_count,
         collection_amount,
         collection_date
       )

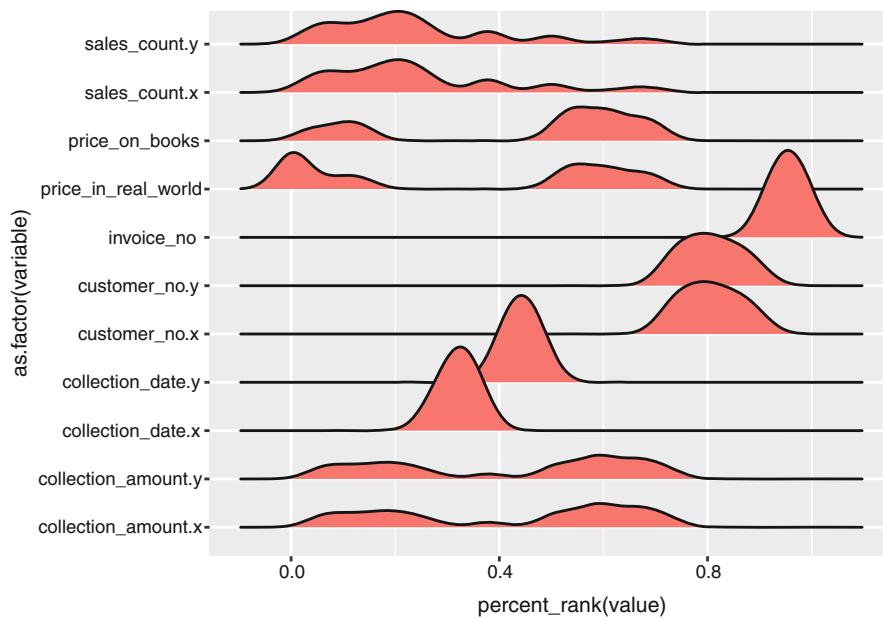
sales_except <-
  left_join(sales_journal,
            real_world_sales,
            by="invoice_no")

if(sales_except$sales_unit_price.x != sales_except$sales_unit_price.y) {
  sales_except$exception = 1
} else {
  sales_except$exception = 0
}

sales_except %>%
  select(exception,
         price_in_real_world = sales_unit_price.y,
         price_on_books = sales_unit_price.x,
         invoice_no,
         customer_no.y,
         sales_count.y,
         collection_amount.y,
         collection_date.y,
         customer_no.x,
         sales_count.x,
         collection_amount.x,
         collection_date.x
       ) %>%
gather(variable, value, -exception) %>%
ggplot(aes(y = as.factor(variable),
           fill = as.factor(exception),
           x = percent_rank(value))) +
  geom_density_ridges() +
  theme(legend.position = "none")

```

The plots reveal the simple strategy used to alter `sales_unit_price` values, while other variables had no change in their distribution. Real-world situations will generally be more complex, but the basic concepts still apply, and this should be sufficient to demonstrate the autoencoder.



Autoencoders and Unbalanced Datasets

Since only 5% of the observations are in error (set in Chap. “Simulated Transactions for Auditing Service Organizations” simulations), we have a highly unbalanced classification problem. Statistical approaches have difficulty with this kind of problem because there are only a small sample of error transactions. The alternative is to use an autoencoder, a neural network that is used to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. In unbalanced classification problems, traditional classification approaches tend not to work because we have only a very small sample of the rarer class. Similarly, traditional linear models and generalized linear regression models suffer from singular design matrices, and fail to converge.

Machine learning models are well suited for financial predictions. The high multicollinearity of accounting and financial data creates problems for traditional estimators based on statistical distributions. Machine learning weights essentially provide a very flexible, non-parametric basis for the extraction of information from large datasets. Given sufficient data, deep-learning models can achieve prediction accuracies that exceed that of statistical analysis. The following example shows how a simple autoencoder model can be used to substantially increase the efficiency of error prediction.

An autoencoder is an unsupervised neural network that is used to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. Since breaches should have a different distribution than non-breaches, we expect that our autoencoder will have higher reconstruction errors on breaches than on non-breaches. This means that we can use the reconstruction error as a quantity that indicates if a transaction is a breach or not. The autoencoder has an hourglass shape, with a large number of trainable weights on input and output sides, with a bottleneck (only a few trainable parameters) in the middle.

The ability to use a machine-generated indicator of error, such as the autoencoder’s reconstruction error level, is important in planning the current year’s interim control tests, because we do not have knowledge of the “errors” prior to audit tests (otherwise, we would not need to audit). We will have a large history of audited transaction files in the workpapers, and these can be aggregated to improve the results from machine learning over time. This ability to continually learn from each year’s audit is a significant advantage of using machine learning algorithms in auditing.

Audit working papers, combining current and past years’ audits, hold a wealth of detail about a client’s internal control weaknesses and control culture. These may involve interactions of many hidden factors, over time, and across the company. A branch of artificial intelligence known as deep learning, a.k.a. neural networks, offers the ability to elicit subtle, hidden causal factors in control failures. The advantage for the audit firm is that new evidence need not be collected. Instead, expenses can be minimized by mining existing audit files for predictors of control breaches.

Prior literature has investigated data mining of audit data to design preventive internal controls (Lee et al. 1998, 2000). Kotsiantis et al. (2006) and Kirkos et al. (2007) investigated data mining methods to identify fraudulent accounting reports. Gray and Debreceny (2014) and Sharma and Panigrahi (2013) have provided surveys of methods that have been applied, with

their industry context. Chye et al. (2004) provided a method to identify “going concern” alerts that have financial auditing implications.

Autoencoders map a layer with a large number of trainable weights into a layer with a smaller number of trainable weights (dimensionality reduction). That small layer is then mapped back to a layer with roughly the same number of weights as the original layer. It is akin to principal components analysis (PCA) in isolating the most information-rich predictors in a dataset, even when these predictors are combinations of measured predictors.

For this example, I constructed a simple deep-learning autoencoder with three fully-connected layers with [100, 10, 100] weights at levels 1 through 3. Since errors should be distributed differently than non-error transactions, the autoencoder should exhibit higher reconstruction errors from the error dataset, and the reconstruction error level can predict whether a transaction is in error.

Compiling the model requires three choices: (1) a loss function, (2) an optimization algorithm, and (3) a performance metric (objective function).

Traditional parametric statistics use a squared-error loss with first-order calculus optimization and F or R-statistics measuring performance; these were easy to compute in a time before computers.

- The current problem involves a binary choice (error or not-error); the most appropriate loss function is binary cross-entropy.
- Optimization involves a complex response surface, and we choose the computationally intensive, but fast and efficient Adam optimizer (Kingma and Ba 2014) for the task.
- Mean-squared-error (MSE) provides a straightforward performance metric for errors.

Real-world data can vary widely in magnitude: an individual sale’s quantity might be 2–3, but the inventory stock level for that same SKU might be 2000–3000 units. Deep-learning algorithms work best when the data is presented at roughly comparable scales. It is generally necessary to “normalize” the data; otherwise, convergence can be slow and unreliable. I split the sales dataset into train and test sets and then normalized the data to standardize on small input values of generally the same scale. Time data is also removed, as it is irrelevant for autoencoders (I discuss later the use of recurrent neural networks, which are commonly used for time-series forecasting).

I trained the model using Keras’ `fit()` function, and used `callback_model_checkpoint()` in order to save the model after each epoch. By passing the argument `save_best_only = TRUE`, I save only the epoch (i.e., pass of the dataset) with the smallest loss value on the test set. I also use `callback_early_stopping()` to stop training if the validation loss stops decreasing for 15 epochs (i.e., passes of the dataset).

```
## Creating virtualenv for TensorFlow at ~/.virtualenvs/r-tensorflow
## Installing TensorFlow ...
##
## Installation complete.
```

R will restart after the installation of keras. We can then proceed to build our deep-learning model.

```
library(tidyverse)
library(lubridate)
library(stringr)
library(keras)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

use_session_with_seed(123) ## make sure results are reproducible
## The use_session_with_seed() function establishes a common random seed for R, Python, NumPy, and
## TensorFlow. It furthermore disables hash randomization, GPU computations, and CPU parallelization,
## which can be additional sources of non-reproducibility.

real_world_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package = "auditAnalytics",
  mustWork = TRUE),
  na.strings="0", stringsAsFactors=FALSE) %>%
select(invoice_no,
       sales_unit_price,
       customer_no,
       sales_count,
```

```

        collection_amount,
        collection_date
    )

real_world_sales$sales_unit_price = real_world_sales$sales_unit_price*rbinom(nrow(real_world_sales),1,.9)

sales <-
  read_csv(system.file("extdata", "sales_journal.csv", package = "auditanalytics", mustWork = TRUE),
  col_types = cols(X1 = col_skip())) %>%
filter(cash_not_ar == 0) %>%
select(invoice_no,
       sales_unit_price,
       invoice_no,
       customer_no,
       sales_count,
       collection_amount,
       collection_date
      )

sales_w_error <-
  left_join(sales,
            real_world_sales,
            by="invoice_no")

if(sales_except$sales_unit_price.x != sales_except$sales_unit_price.y) {
  sales_w_error$error = TRUE
} else {
  sales_w_error$error = FALSE
}

sales_w_error[is.na(sales_w_error)] <- 0

sales_w_error$collection_date.x <- as.numeric(decimal_date(as.POSIXct(sales_w_error$collection_date.x)))
sales_w_error$collection_date.y <- as.numeric(decimal_date(as.POSIXct(sales_w_error$collection_date.y)))

y <- sales_w_error %>% select(error)

x <- sales_w_error %>%
  select (price = sales_unit_price.x,
         count = sales_count.x,
         amt_coll = collection_amount.x,
         date_coll = collection_date.x,
         real_price = sales_unit_price.y,
         real_count = sales_count.y,
         real_amt_coll = collection_amount.y,
         real_date_coll = collection_date.y,
         error)

c <- x$error

freq <- 100 * nrow(x[x$error!=0,])/nrow(x[x$error==0,])
cat("Error frequency = ", freq, "%")

## Error frequency = 0 %

bucket <- sample(2, nrow(sales), replace=T, prob = c(0.75, 0.25)) # 3:1 split

y_train <- y[bucket==1,]
y_test <- y[bucket==2,]

# don't use the error for ML
# the auditor doesn't know the errors,

```

```

# these are something to be found during the audit)
x_train <- x[bucket==1,1:8]
x_test <- x[bucket==2,1:8]
x_train_w_err <- x[bucket==1,9] # save error for later
x_test_w_err <- x[bucket==2,9]

col_means_train <- as.numeric(apply(x_train,2,mean)) ## invoice is an ID
col_stddevs_train <- as.numeric(apply(x_train,2,SD))

# Normalize training data
x_train <- as.matrix(scale(x_train, center = col_means_train, scale = col_stddevs_train))

# Test data is *not* used when calculating the mean and std.
x_test <- as.matrix(scale(x_test, center = col_means_train, scale = col_stddevs_train))

## The autoencoder model

x_train_model <- as.matrix(x_train) ## don't use the error to train

model <- keras_model_sequential()
model %>%
  layer_dense(units = 100, activation = "tanh", input_shape = ncol(x_train)) %>%
  layer_dense(units = 10, activation = "tanh") %>%
  layer_dense(units = 100, activation = "tanh") %>%
  layer_dense(units = ncol(x_train))

summary(model)

## -----
## Layer (type)          Output Shape         Param #
## =====
## dense_3 (Dense)      (None, 100)           900
## -----
## dense_2 (Dense)      (None, 10)            1010
## -----
## dense_1 (Dense)      (None, 100)           1100
## -----
## dense (Dense)         (None, 8)             808
## =====
## Total params: 3,818
## Trainable params: 3,818
## Non-trainable params: 0
## -----
```

Compile and fit the model to the data Use callback_model_checkpoint() in order to save the model after each epoch. By passing the argument save_best_only = TRUE I save only the epoch (i.e., pass of the dataset) with smallest loss value on the test set.

```

model %>% compile(
  loss = "binary_crossentropy",
  optimizer = "adam",
  metrics= 'mse'
)
```

```

checkpoint <- callback_model_checkpoint(
  filepath = "model.hdf5",          ## Keras model weights are saved to HDF5 format.
  save_best_only = TRUE,
  period = 10,
  verbose = 1
)

early_stopping <- callback_early_stopping(patience = 10)

history <-
  model %>%
  fit(
    x = x_train[y_train == 0,],
    y = x_train[y_train == 0,],
    epochs = 50,           ## potentially complex model so train for a long time
    batch_size = 8192,    ## adjust for your memory size
    validation_data = list(x_test[y_test == 0,], x_test[y_test == 0,]),
    callbacks = list(callbacks, early_stopping)
  )

plot(history)

```

After training we can get the final loss for the test set by using the evaluate() function.

```

loss <- evaluate(model, x = x_test[y_test == 0,], y = x_test[y_test == 0,])
loss

```

```

## $loss
## [1] -6.158478
##
## $mean_squared_error
## [1] 0.2640558

```

Once the model is trained and tuned, the autoencoder can be used to generate predictions. We assumed that mean-squared-error (MSE) of error transactions will be higher, and this can be used to predict errors in unaudited populations (Fig. 5).

A good measure of model performance in highly unbalanced datasets is the Area Under the ROC Curve (AUC). AUC has a straightforward interpretation for the audit task; it is the probability that an error transaction will have higher MSE than a non-error transaction. Calculate this using R's *Metrics* package, which implements a wide variety of common machine learning model performance metrics.

```

pred_train <- predict(model, x_train)
mse_train <- apply((x_train - pred_train)^2, 1, sum)

pred_test <- predict(model, x_test)
mse_test <- apply((x_test - pred_test)^2, 1, sum)

```

From the above analysis, the probability that errors have higher MSE is quite high, and the autoencoder is able to generate very informative predictors for the sales transaction data. Making predictions involves finding a threshold k such that if $MSE > k$ the auditor decides that the transaction is in error. To define this value, it is useful to look at the information science measures of precision and recall while varying the threshold k . The following code creates two plots that do just that, using an abbreviated form of R's ggplot called qplot, mapping precision and recall against the decision threshold for deciding whether or not to audit a transaction.

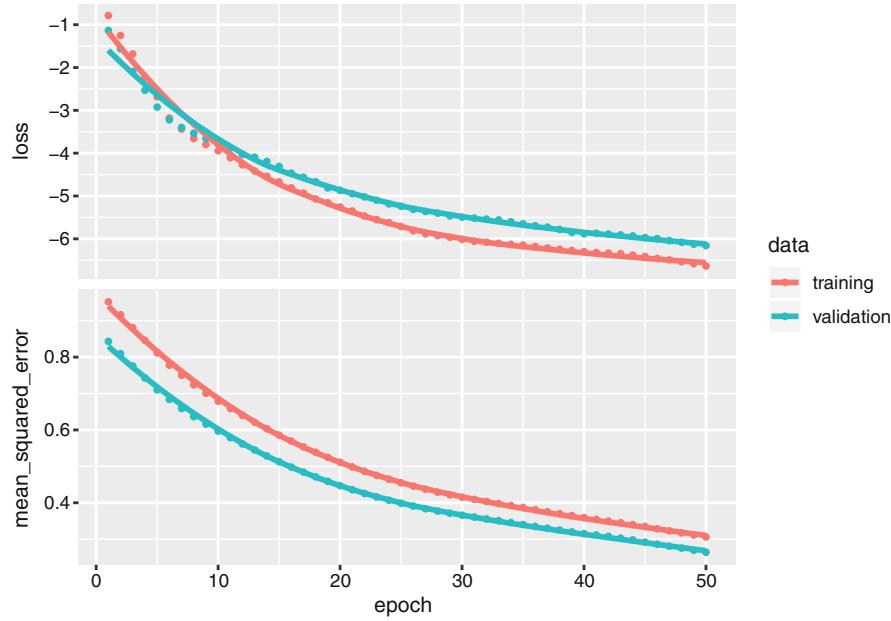


Fig. 5 Machine learning

```

possible_k <- seq(0, 0.5, length.out = 100)      # range of threshold values k
precision <- sapply(possible_k, function(k) {      # create a vector of precision x k
  predicted_class <- as.numeric(mse_test > k)
  sum(predicted_class == 1 & y_test == 1)/sum(predicted_class)
})

qplot(possible_k, precision, geom = "line") + labs(x = "Decision threshold k", y = "Precision")

recall <- sapply(possible_k, function(k) {        # create a vector of recall x k
  predicted_class <- as.numeric(mse_test > k)
  sum(predicted_class == 1 & y_test == 1)/sum(y_test)
})
qplot(possible_k, recall, geom = "line") + labs(x = "Decision threshold k", y = "Recall")

```

These two graphs provide insight into the trade-offs we make in threshold choice, but they are not directly interpretable into specific choices in the audit. For that, it is necessary to map the trade-off between:

1. audit cost, which will depend on sample size n , and
2. audit risk, i.e., the probability of accepting an account value α (in this case sales) containing an intolerable error. This in turn depends on the auditor's choice of intolerable error level M

The auditor starts with the assumption that the transaction system is “out of control” $X\%$ of the years, based on prior years’ workpapers. In the current year, the system is by default assumed to be “in control.” Use a one-sample proportion test to see if the probability of an “out-of-control” transaction system is significantly different from zero. Significance is measured by p -value; if this falls below the significance threshold, say 0.05, we can conclude that transaction processing is “out-of-control.” This can be determined using R’s `pwr.p.test` function in the `pwr` package.

Note that the thresholds for the following tests are for MSE of the model predictions, and will be different from those of precision and recall (which are simply ratios of counts). The prediction process identifies errors $\text{train_acc} \approx 87\%$ of the time; thus, sampling from these predictions can be thought of as a “distilled” sample of errors that are $\frac{1}{1-\text{train_acc}}$ as concentrated as in a sample from all of the population. The estimate of the population mean from the prediction sample multiplies the sample estimate of error by $\frac{(\text{population-size})(1-\text{train_acc})}{\text{sample-size}}$ for a 95% confidence (5% significance) this will be $z_{.95} \approx 1.96$

```

library(tidyverse)

# range of sample sizes
r <- seq(10,100,10)
nr <- length(r)

# thresholds
k <- seq(.5,20,.5)
nk <- length(k)

pred_mean <- array(numeric(nr*nk), dim=c(nr,nk))
full_mean <- array(numeric(nr*nk), dim=c(nr,nk))
pred_sd <- array(numeric(nr*nk), dim=c(nr,nk))
full_sd <- array(numeric(nr*nk), dim=c(nr,nk))

pop_error_pred <- array(numeric(nr*nk), dim=c(nr,nk))
pop_error_full <- array(numeric(nr*nk), dim=c(nr,nk))

## recover the error in training and test data to do sampling
x_train <- cbind(x_train,x_train_w_err)
x_test <- cbind(x_test,x_test_w_err)
x_test <- cbind(x_test,mse_test)

for(l in 1:nk) {

  predicted <- x_test[mse_test > k[l],]

  for (j in 1:nr){

    pred_sample <- dplyr::sample_n(as_tibble(predicted), r[j], replace=T)

    ## sample_n is part of tidyverse
    ## a tibble is tidyverse's version of a data.frame
    ## (a cute variation on 'table')

    pred_mean[j,l] <- mean(pred_sample$x_test_w_err)
    pred_sd[j,l] <- sd(pred_sample$x_test_w_err)

    ## for a 95% confidence (5% significance) this will be $z_{.95} \approx 1.96$ 

    pop_error_pred[j,l] <-
      (pred_mean[j,l] + pred_sd[j,l] * 1.96) * ## one-sided 95% CL
      nrow(sales)*(1-train_acc)/r[j]

    ## sample from all rows and compute sample mean-sd

    full_sample <- dplyr::sample_n(as_tibble(x_test), r[j])
    full_mean[j,l] <- mean(full_sample$x_test_w_err)
    full_sd[j,l] <- sd(full_sample$x_test_w_err)
    pop_error_full[j,l] <-
      (pred_mean[j,l] + pred_sd[j,l] * 1.96) *
      nrow(sales) / r[j]

  }

}

## save all of our computed data for the sales transaction audit;

```

```

# reload with load("audit_data.RData")
save(pop_error_full, pop_error_pred, pred_mean, pred_sd, full_mean, full_sd, file="audit_data.RData")

library(dplyr)
library(plotly)
library(reshape)
library(ggplot2)
library(scales) # needed for formatting y-axis labels to non-scientific type

# webshot::install_phantomjs() # phantomjs is required to run plotly, install webshot first

r <- seq(10,100,10)
nr <- length(r)

k <- seq(.5,20,.5)
nk <- length(k)

pred_scatter <- melt(pop_error_pred)
colnames(pred_scatter)=c("sample_size", "threshold", "error_est")
full_scatter <- melt(pop_error_full)
colnames(full_scatter)=c("sample_size", "threshold", "error_est")

full_scatter$sample_size <- full_scatter$sample_size * 20
full_scatter$threshold <- full_scatter$threshold * .5

pred_scatter$sample_size <- pred_scatter$sample_size * 20
pred_scatter$threshold <- pred_scatter$threshold * .5

pred_scatter$pred_or_full <- "predicted" ## indicator for plotting
full_scatter$pred_or_full <- "full"

full_pred_scatter <- rbind(full_scatter, pred_scatter)
grouped <- full_pred_scatter %>% dplyr::group_by(sample_size,pred_or_full)
full_pred_2D_plot <- summarize(grouped, mean=mean(error_est))

p <- full_pred_2D_plot %>%
  ggplot(aes(sample_size, mean, color=pred_or_full)) +
  geom_smooth(se=F) +
  xlab("Sample Size") +
  ylab("Estimated Population Error") +
  scale_y_continuous(labels = comma)

p

q <-
  pred_scatter %>%
  plot_ly(
    x = ~sample_size,
    y = ~threshold,
    z = ~error_est) %>%
  add_markers(size=10) %>%
  layout(scene = list(xaxis = list(title = 'sample size'),
                      yaxis = list(title = 'MSE decision threshold'),
                      zaxis = list(title = 'estimated error in sales account')))

# type "q" if you would like to generate an interactive 3D graph of the predicted population errors

```

In this example, using the deep-learning autoencoder to create error rich samples provided the auditor the ability, for 95% confidence (5% significance) level of audit risk, to estimate much tighter bounds for the population error for any sample size. This is possible because the deep-learning “distillation” process solves the problem of highly unbalanced error data—i.e., that there are many more correct transactions than errors.

From the 3D plotly graph, it is possible to see the effect of applying different MSE decision thresholds. Values from 5–10 generate the most stable estimates of population error at all sample sizes. Very high values perform badly for small samples, whereas small values tend not to provide enough information to correct the unbalanced dataset problem (Fig. 6).

The plotly package can create attractive, interactive 3D plots that are useful for exploring more complex relationships between variables. I am not enthused about the widespread application of plotly, though, because phantomjs (required for it to generate interactive HTML) can significantly slow the response of your computer; nor does plotly reliably render its graphics into image formats such as .png, that can be embedded in PDF files.

The 3D plot created by the plotly package shows that estimation based on the predicted errors provides much tighter bounds on the sample error (by an order of magnitude at least). Particularly for small sample sizes, estimation based on deep-learning choices for the sampled items is much less variable than estimation from taking a (uniform) random sample over the entire population. Using deep-learning predictive models offers the potential for cheaper and more effective audit risk-cost-scope trade-offs.

The R package plotly provides a rich set of 3D plotting tools that allow plots to be rotated, and individual points to be queried by placing the cursor over them. This can be very useful in planning audits, offering the potential for fine-tuning of the audit risk-cost trade-offs.

The threshold k is an audit planning decision parameter and in order to optimize audit planning, we want to find the value of k that minimizes audit cost (i.e., sample size) for a given audit risk of incorrectly accepting an account balance that is intolerably (materially) in error, where the intolerable error rate is ρ . Let $\rho = \$100,000$ be the intolerable error for the population. We want k for the minimum sample size such that $\text{population error} \geq \rho$

```
library(ggplot2)

audit_cost <- size_index <- 0
for (l in 1:nk) {
  size_index[l] <- which.min(pop_error_pred[, 1] >= 1e+05)
  audit_cost[l] <- r[size_index[l]]
  # cat('min size is = ', audit_cost[l], '\n\n')
}

qplot(k, audit_cost, geom = "line") + labs(x = "Decision threshold k", y = "Audit cost (sample size)")

# qplot is a simplified form for ggplot, which is handy for quick plots
```

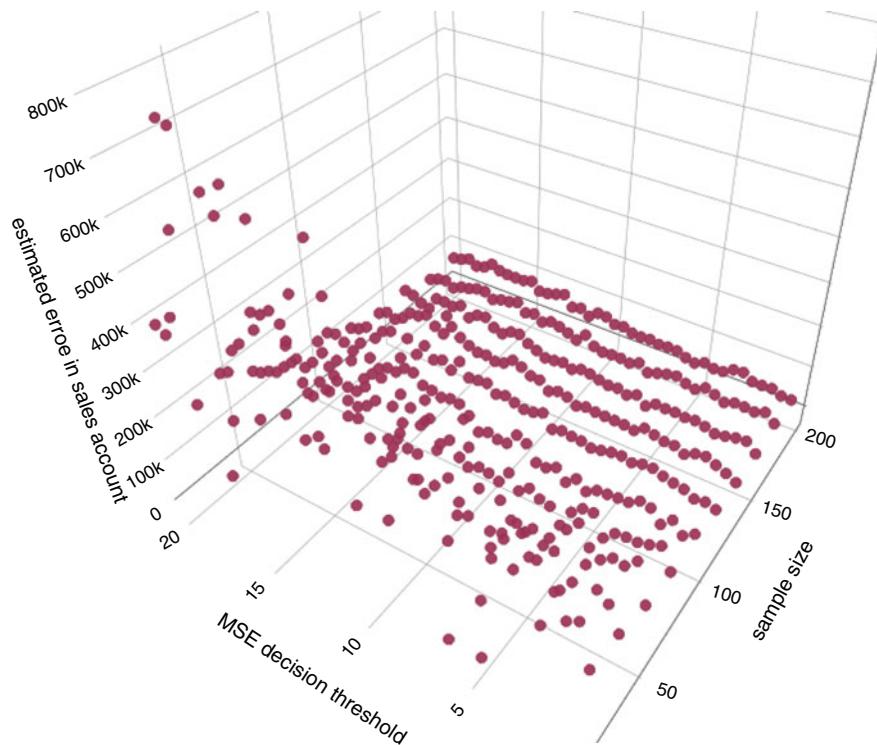


Fig. 6 Sample size and optimal MSE threshold values “ k ” for prediction of error rich datasets

Because we are computing decision thresholds using samples (as we would in a real audit) rather than having access to the full population statistics (the real state of the world) the curve is not a smooth monotonic curve, and the guidance provided by this graph is only approximate. The plot suggests that $k \approx 5\text{--}10$ will keep sample sizes under 50 transactions, while providing the desired level of resolution (i.e., such that we can detect any level of *population error* $\geq \rho = \$100,000$ with a confidence of greater than 95%).

The choice of suitable thresholds is clearly important in controlling audit costs and quality. A more detailed exploration of this audit choice can be accomplished with the `PresenceAbsence` package (Freeman and Moisen 2008) for R. It provides a set of functions useful when evaluating the results of the presence–absence analysis, for example, models of species distribution or the analysis of diagnostic tests. The package provides a toolkit for selecting the optimal threshold for translating a probability surface into presence–absence maps specifically tailored to their intended use. The package includes functions for calculating threshold dependent measures such as confusion matrices, percent correctly classified (PCC), sensitivity, specificity, and Kappa, and produces plots of each measure as the threshold is varied. It also includes functions to plot the Receiver Operator Characteristic (ROC) curve and calculates the associated area under the curve (AUC), a threshold independent measure of model quality. Finally, the package computes optimal thresholds by multiple criteria, and plots these optimized thresholds on the graphs.

Final Thoughts on Machine Learning Applications in Auditing

The model chosen for this demonstration was the autoencoder, an unsupervised machine learning algorithm that assumes that error and non-error transactions have different fingerprints (i.e., patterns distinguishable from the columns of data). Machine learning provides many other tools for prediction, some more sophisticated, with new discoveries being made and shared continually. Accounting datasets are often represented as sequence data, sequence order being given by identifiers like invoice number, or the transaction date. Recurrent neural networks are commonly used in machine learning to analyze sequence data, as these will identify subtle regularities over time. Where they have been applied, e.g., in reading electrocardiograms in medicine, they have significantly outperformed humans in diagnosis and predictions.

Speed of computation, and the ‘curse of dimensionality’ are often cited as limitations in the application of machine learning. But new models and faster hardware are continually pushing the bounds of what is possible. Neural Structured Learning (NSL) is the most recent learning paradigm to significantly speed up the training of neural networks by leveraging structured signals in addition to feature inputs. The structure can be explicit as represented by a graph or implicit as induced by adversarial perturbation. Modules for NSL have recently been incorporated into TensorFlow, though sadly at the time of this writing, they exist only in Python version. They should be accessible in R shortly.

By viewing machine learning as an advancement in statistical processing and decision-making, it is possible for the auditor to efficiently assess the value of new tools to a specific audit task. There are no “silver bullets” and auditors should be deeply skeptical of anyone pushing the latest vendor technology as a panacea. But there are exciting information technologies introduced every year, and some of these will be truly useful for accounting.

References

- Chye Koh, Hian, and Chan Kee Low. 2004. Going Concern Prediction Using Data Mining Techniques. *Managerial Auditing Journal* 19 (3): 462–476.
- Cohen, Jacob. 1992. A Power Primer. *Psychological Bulletin* 112 (1): 155.
- Freeman, Elizabeth A., and Gretchen Moisen. 2008. PresenceAbsence: An R Package for Presence Absence Analysis. *Journal of Statistical Software* 23 (11): 31.
- Gray, Glen L., and Roger S. Debreceny. 2014. A Taxonomy to Guide Research on the Application of Data Mining to Fraud Detection in Financial Statement Audits. *International Journal of Accounting Information Systems* 15 (4): 357–380.
- Kingma, Diederik P., and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. arXiv Preprint arXiv:1412.6980.
- Kirkos, Efstathios, Charalambos Spathis, and Yannis Manolopoulos. 2007. Data Mining Techniques for the Detection of Fraudulent Financial Statements. *Expert Systems with Applications* 32 (4): 995–1003.
- Kotsiantis, Sotiris, E. Koumanakos, D. Tzepelis, and V. Tampakas. 2006. Forecasting Fraudulent Financial Statements Using Data Mining. *International Journal of Computational Intelligence* 3 (2): 104–110.

- Lee, Wenke, Salvatore J. Stolfo, and Kui W. Mok. 1998. Mining Audit Data to Build Intrusion Detection Models. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, 66–72.
- Lee, Wenke, Salvatore J. Stolfo, and Kui W. Mok. 2000. Adaptive Intrusion Detection: A Data Mining Approach. *Artificial Intelligence Review* 14 (6): 533–567.
- Sharma, Anuj, and Prabin Kumar Panigrahi. 2013. A Review of Financial Accounting Fraud Detection Based on Data Mining Techniques. arXiv Preprint arXiv:1309.3944.



Substantive Tests

Substantive Tests

Objective of Substantive Tests

Auditing of financial accounts is the process of verifying that economic events in the real world are accurately summarized in the financial statements of a legal entity. An audit opinion—the product of such auditing—provides reasonable assurance supported by an independent 3rd party:

1. That the financial statements are presented fairly, in all material respects.
2. In accordance with some financial reporting framework (e.g., generally accepted accounting principles).
3. Applied consistently so that year-to-year trends and comparisons are possible.

In the USA, the auditing framework is dictated by generally accepted accounting principles (GAAP) and generally accepted accounting standards (GAAS). Internationally, the International Standards on Auditing (ISA) issued by the International Auditing and Assurance Standards Board (IAASB) is considered as the benchmark for the audit process. In 2004, a project was begun to clarify and converge the standards internationally by 2014. Under the US generally accepted accounting principles (GAAP), auditors must release one of three types of opinions of the overall financial statements:

1. An unqualified auditor's opinion is the opinion that the financial statements are presented fairly.
2. A qualified opinion is that the financial statements are presented fairly in all material respects in accordance with GAAP:
 - (i.) Except for a material misstatement that does not pervasively affect the user's ability to rely on the financial statements.
 - (ii.) Or with a scope limitation that is of limited significance.
3. An adverse audit opinion is issued when the financial statements do not present fairly due to departure from US GAAP with an explanation of the nature and size of the misstatement.

Additionally, an auditor can issue a disclaimer (which is considered a type of qualified opinion) because there is insufficient or biased data.

Exploratory Substantive Tests

The first step in substantive testing in an audit is to compare the client's transaction files to the Trial Balance. Traditionally this was accomplished by “footing” (i.e., calculating a total, perhaps using a vintage 10-key hand-crank adding machine) and “agree” (i.e., checking whether the total agrees with the Trial Balance account comprised by the transactions being totaled). The auditor has more sophisticated tools available today. This section explores particular tools that might be used to complete the exploratory analysis of client's transaction files.

As an example, consider the sales transactions generated by the code in the last chapter of this book. Sales will be the largest entry on the trial balance, thus a thorough exploration of the transactions generating this balance is demanded by the audit.

```

library(broom)
library(readr)
library(tidyverse)
library(compare)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

real_world_cash_sales <-
  read.csv(system.file("extdata", "real_world_cash_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0", stringsAsFactors=FALSE)

real_world_credit_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0", stringsAsFactors=FALSE)

real_world_sales <-
  rbind(real_world_cash_sales, real_world_credit_sales)

sales_journal <-
  read.csv(system.file("extdata", "sales_journal.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0", stringsAsFactors=FALSE)

credit_sales_journal <-
  sales_journal %>%
  filter(cash_not_ar == 1)

```

How many records are there? What fields exist? Of which type? Is there missing data? Is the data in a reasonable range? What sort of distribution does it have? Running a summary statistical function over the whole dataset is a great first step to understanding what you have, and whether it is valid. A simple `summary` tool exists in base R (in the following code chunk, which is not evaluated but left for the reader to explore).

```
summary(credit_sales_journal)
```

This highlights which fields have missing data. But it does not show the overall count of records, nor is there a standard deviation, nor does it reveal groupings that might be of use in stratification or error audits. Nor is it easy to convert this to an auditable report for the working papers.

Fortunately, several R functions provide more useful summarization tools for the auditor. Three of these will be discussed here, with examples of each for the sales transactions, and differing applications for reasons that I will articulate.

- `describe`, from the *Hmisc* package
- `describe` from *psych*
- `skim` from *skimr*

Here are comparisons of output from these different summarization tools that may or may not meet the specific needs of an audit or test. The output of *Hmisc*'s `describe` is significantly more informative for dates, omissions, and duplicated transactions than the other two packages, but the output tends to be rather crude and is left for the reader to explore. Here is a code chunk to `describe` the credit sales journal.

```

library(Hmisc)
Hmisc::describe(credit_sales_journal)

```

The *psych* `descript` function reports all the numeric and categorical fields in its output, but the categorical fields are recoded as numbers and summarized in a fashion normally associated with numeric fields. For numeric fields, the function reports a plethora of statistics: count, mean, standard deviation, median, trimmed mean (removing outliers), mean absolute

deviation, min, max, range, skewness, Kurtosis, and standard error. If any of these are meaningful for the auditor, this package will generate them for each numeric field in the dataset. Note that accounting transactions will often be right-skewed and kurtotic, and this function allows the auditor to investigate how far the transaction distribution differs from a Normal distribution.

```
library(kableExtra)
library(psych)
psych::describe(credit_sales_journal) %>%
  kable("latex", booktabs = T) %>%
  kable_styling(
    bootstrap_options = "striped",
    full_width = F,
    latex_options="scale_down")
```

Another useful feature is the availability of statistics for specific summary-by-group variation. In the following example, the auditors want to see sales statistics by customers.

```
library(broom)
library(kableExtra)
library(psych)

by_cust <- psych::describeBy(credit_sales_journal, credit_sales_journal$customer_no, mat = TRUE)

tidy(by_cust[,-c(1:2)]) %>%
  kable("latex", booktabs = T) %>%
  kable_styling(
    bootstrap_options = "striped",
    full_width = F,
    latex_options="scale_down")
```

The “mat=TRUE” parameter produces output in a matrix format, which can be used for creating tables and output for the work papers. The tidy function of broom allows saving the summary information in a table for later use in the audit.

The skim from skimr function provides an informative and compact summary, which can be saved as tables or reports in the working papers. I make the assumption that working papers are likely retained in computer format, even if there are separate physical documents retained. The output works with kable and tidyverse packages like dplyr. The histograms of field values can be useful in determining sample size and distributions that should be considered for tests. The skimr package works well with tidyverse packages.

```
library(tidyverse)
library(skimr)

skim_with(numeric = list(hist = NULL))
sales_rpt <-
  group_by(credit_sales_journal, customer_no) %>%
  skim()

head(sales_rpt)

## # A tibble: 6 x 7
## # Groups:   customer_no [1]
##   customer_no variable type     stat      level value formatted
##   <chr>        <chr>   <chr>    <chr>    <chr>  <dbl> <chr>
## 1 c00001       X       integer missing   .all      0    0
## 2 c00001       X       integer complete .all      5    5
## 3 c00001       X       integer n        .all      5    5
## 4 c00001       X       integer mean     .all    568. 568.4
## 5 c00001       X       integer sd        .all    602. 602.46
## 6 c00001       X       integer p0        .all     53    53
```

	Vars	n	Mean	Sd	Median	Trimmed	Mad	Min	Max	Range	Skew	Kurtosis	Se
X	1	179	682.899385	1118.954761	529	513.22759	437.3670	4	9041	9037	5.6132527	34.4838146	83.6346056
customer_no*	2	179	NaN	NA	NaN	NA	Inf	-Inf	NA	NA	NA	NA	NA
invoice_no*	3	179	NaN	NA	NA	NA	Inf	-Inf	NA	NA	NA	NA	NA
invoice_date*	4	179	NaN	NA	NA	NA	Inf	-Inf	NA	NA	NA	NA	NA
sku*	5	179	NaN	NA	NaN	NA	Inf	-Inf	NA	NA	NA	NA	NA
sales_count	6	179	14.94972	3.977851	15	14.83448	4.4478	5	28	23	0.3559026	0.5673765	0.2973186
cash_not_ar	7	179	1.00000	0.000000	1	1.00000	0.00000	1	1	0	NaN	NaN	0.0000000
sales_return	8	13	1.00000	0.000000	1	1.00000	0.00000	1	1	0	NaN	NaN	0.0000000
unit_cost	9	179	32.05028	5.128981	33	31.82759	4.4478	24	42	18	0.1614248	-0.7291953	0.3833580
sales_unit_price	10	179	70.77654	24.846615	72	71.44138	26.6868	27	108	81	-0.2062684	-0.9145744	1.8571232
sales_extended	11	179	1069.02235	514.443071	1053	1029.64138	484.8102	252	3024	2772	0.8198446	1.0088470	38.4512805
collection_amount	12	179	1069.02235	514.443071	1053	1029.64138	484.8102	252	3024	2772	0.8198446	1.0088470	38.4512805
collection_date*	13	179	NaN	NA	NA	NaN	Inf	-Inf	NA	NA	NA	NA	NA
collection_no*	14	179	NaN	NA	NA	NaN	Inf	-Inf	NA	NA	NA	NA	NA
shipper_date*	15	179	NaN	NA	NA	NaN	Inf	-Inf	NA	NA	NA	NA	NA
shipper_no*	16	179	NaN	NA	NA	NaN	Inf	-Inf	NA	NA	NA	NA	NA

Column	n	Mean	Sd	Median	Trimmed	Mad	Min	Max	Range	Skew	Kurtosis	Se
Vars	144	8.5000000	4.625862	8.5000000	8.5000000	4.0000000	1.0000000	16.000000	15.000000	0.000000	1.790588	0.3854885
n	144	18.7361111	11.309372	21.0000000	18.7068966	11.0000000	0.0000000	35.000000	35.000000	-0.055964	1.549810	0.9424477
Mean	69	392.1122490	489.954061	67.0937500	339.4469915	66.0937500	1.0000000	1393.800000	1392.800000	NaN	NaN	58.9835376
Sd	67	241.6645033	373.066561	19.7401219	172.1198018	19.7401219	0.0000000	1880.521779	1880.521779	NA	NA	45.5773344
Median	69	364.0724638	460.126559	68.0000000	313.0350877	67.0000000	1.0000000	1296.000000	1295.000000	NA	NA	55.3927284
Trimmed	69	369.9153524	471.612881	67.1153846	312.5770640	66.1153846	1.0000000	1393.800000	1392.800000	NaN	NaN	56.7755190
Mad	69	179.9640043	242.314642	19.2738000	146.2181737	19.2738000	0.0000000	836.186400	836.186400	NA	NA	29.1712549
Min	144	Inf	NaN	Inf	Inf	Inf	1.0000000	Inf	Inf	NaN	NaN	NaN
Max	144	-Inf	NaN	-Inf	-Inf	-Inf	-Inf	9041.000000	Inf	NaN	NaN	NaN
Range	144	-Inf	NaN	-Inf	-Inf	Inf	-Inf	9037.000000	Inf	NaN	NaN	NaN
Skew	54	0.6003240	1.118366	0.3577041	0.3753098	0.4298234	-0.5608848	4.552953	5.113837	NA	NA	0.1521904
Kurtosis	54	0.5509875	5.041150	-0.9599935	-0.7434573	0.5019336	-2.2130057	21.681755	23.894761	NA	NA	0.6860137
Se	67	58.4281545	85.212595	4.5446617	41.7175649	4.5446617	0.0000000	410.363495	410.363495	NA	NA	10.4103754

Creating Trial Balance Figures in One Step

Given the full set of transaction files that will be used in the audit, it should be possible to generate an independent auditor's set of trial balance figures to be compared to the client's. The following code shows how this might look for the transaction files generated in the last chapter of this book.

```
library(tidyverse)
library(broom)
library(kableExtra)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

sales_journal <-
  read_csv(system.file("extdata", "sales_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

purchase_journal <-
  read_csv(system.file("extdata", "purchase_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

perpetual_inventory_ledger <-
  read_csv(system.file("extdata", "perpetual_inventory_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

fyear_end_ar_ledger <-
  read_csv(system.file("extdata", "fyear_end_ar_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

fyear_begin_inventory_ledger <-
  read_csv(system.file("extdata", "fyear_begin_inventory_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

expenditures <-
  read_csv(system.file("extdata", "expenditures.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

disbursement_journal <-
  read_csv(system.file("extdata", "disbursement_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

deposit_daily <-
  read_csv(system.file("extdata", "deposit_daily.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

daily_ar_balance <-
```

```

read_csv(system.file("extdata", "daily_ar_balance.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

collections_journal <-
  read_csv(system.file("extdata", "collections_journal.csv", package =
    "auditanalytics", mustWork = TRUE)),
    col_types = cols(X1 = col_skip()))

Sales <-
  sum(sales_journal$sales_count *
    sales_journal$sales_unit_price)

Sales_Returns <-
  sum(sales_journal$sales_return *
    sales_journal$sales_unit_price)

Purchases <-
  sum(purchase_journal$po_count *
    purchase_journal$unit_cost)

Vendor_Disbursements <-
  sum(disbursement_journal$unit_cost * disbursement_journal$no_units_ordered)

YE_Inventory <-
  perpetual_inventory_ledger %>%
  group_by(sku) %>%
  inner_join(sales_journal, by="sku") %>%
  slice(n()) %>%
  mutate(inv_extended = unit_cost * stock_on_hand) %>%
  ungroup() %>%
  select(inv_extended) %>%
  sum()

Accounts_Receivable <- sum(fyear_end_ar_ledger$amount)

Accounts_R2 <-
  max(daily_ar_balance$ar_balance)

Begin_Inventory <-
  fyear_begin_inventory_ledger %>%
  group_by(sku) %>%
  mutate(inv_extended = unit_cost * stock_on_hand) %>%
  ungroup() %>%
  select(inv_extended) %>%
  sum()

Cost_of_Goods_Sold <-
  sum(sales_journal$sales_count *
    sales_journal$unit_cost) +
  YE_Inventory -
  Begin_Inventory

```

```

Misc_Expenses <-
  sum(expenditures$amount)

Collections_on_AR <-
  collections_journal %>%
  group_by(invoice_no) %>%
  inner_join(sales_journal, by="sku") %>%
  filter(cash_not_ar.x==0) %>%
  slice(n())

Uncollected_AR <-
  sum(Collections_on_AR$sales_extended.x) -
  sum(Collections_on_AR$collection_amount.x)

Collections_on_AR <-
  sum(Collections_on_AR$collection_amount.x)

Cash_Deposits <-
  sum(deposit_daily$deposit_amount)

Cash_in_Bank <-
  Cash_Deposits -
  Purchases -
  Misc_Expenses

Change_in_Equity <-
  Sales_Returns +
  Collections_on_AR+
  Purchases+
  Cost_of_Goods_Sold+
  Cash_in_Bank+
  Accounts_Receivable+
  YE_Inventory+
  Misc_Expenses-
  Sales-
  Uncollected_AR

## construct the trial balance and format this with knitr::kable and kableExtra

data.frame(
  "Account"=c("Sales",
             "Sales Returns",
             "Collections_on_AR",
             "Uncollected_AR",
             "Purchases",
             "Cost_of_Goods_Sold",
             "Cash_in_Bank (change from start of yr)",
             "Accounts_Receivable",
             "YE_Inventory",
             "Misc_Expenses",

```

```

    "Change_in_Equity",
    "Total"
  ) ,

"Dr" = c(0,
  Sales_Returns,
  Collections_on_AR,
  0,
  Purchases,
  Cost_of_Goods_Sold,
  Cash_in_Bank,
  Accounts_Receivable,
  YE_Inventory,
  Misc_Expenses,
  0,
  Sales_Returns+
  Collections_on_AR+
  Purchases+
  Cost_of_Goods_Sold+
  Cash_in_Bank+
  Accounts_Receivable+
  YE_Inventory+
  Misc_Expenses
),
"Cr" = c(
  Sales,
  0,
  0,
  Uncollected_AR,
  0,
  0,
  0,
  0,
  0,
  0,
  Sales_Returns +
  Collections_on_AR+
  Purchases+
  Cost_of_Goods_Sold+
  Cash_in_Bank+
  Accounts_Receivable+
  YE_Inventory+
  Misc_Expenses-
  Sales-
  Uncollected_AR,
  Sales+
  Uncollected_AR+
  Change_in_Equity
)
) %>%
  mutate_if(is.numeric, format, digits=4, nsmall = 2, big.mark=",") %>%
  kable(., align='r', caption = "Trial Balance", "latex", booktabs = T) %>%
  kable_styling(full_width = F)

```

Account	Dr	Cr
Sales	0.00	1,047,621.00
Sales Returns	3,454.00	0.00
Collections_on_AR	734,263.00	0.00
Uncollected_AR	0.00	26,006.17
Purchases	490,400.00	0.00
Cost_of_Goods_Sold	485,786.00	0.00
Cash_in_Bank (change from start of yr)	348,822.19	0.00
Accounts_Receivable	168,144.00	0.00
YE_Inventory	31,281.00	0.00
Misc_Expenses	211,890.81	0.00
Change_in_Equity	0.00	1,400,413.83
Total	2,474,041.00	2,474,041.00

Accounts Receivable Auditing

Accounts receivable are extensively audited during the substantive phase of the audit. Accounts receivable is frequently the largest current asset; thus, auditors allocate a significant budget to assure that accounts receivable are fairly stated. Accounts receivable audit procedures are designed to detect audit control risks, which include the following:

- That receivables do not exist
- That recorded receivable balances are inaccurate
- That it may not be possible to collect accounts receivable
- That the derivation of the allowance for doubtful accounts may not properly reflect bad debt experience
- That the sales transactions were not processed in the correct periods
- That revenue was incorrectly recognized

Debit accountability is an audit philosophy that recognizes double-entry's custom of recording and reporting accounting transactions twice—as a debit and as a credit. Cost-effective auditing verifies the debits and assumes that balancing will assure that corresponding credit entries are not materially misstated.

In annual audits, accounts receivable and inventory are the two most important current asset accounts—i.e., debit balance accounts that will expire within an annual business cycle. The audit of these two accounts represents a major expense during substantive year-end testing and is the focus of this chapter.

Accounts receivable are audited by obtaining external evidence, from inspecting the physical locations of the inventory, about the quantity, condition, obsolescence, pricing, and acquisition cost compared to their recorded value. This is the process of “confirmation.”

The following code chunks support standard accounts receivable audit procedures.

Footing and Agreeing to the Trial Balance

“Foot” means total the balance of the outstanding accounts receivable at year-end. “Agree” means that this balance should be traced to the trial balance. If these totals do not match, search for a journal entry that is incorrect. Investigate reconciling items. If the client has journal entries in the accounts receivable account in the general ledger, the auditors will likely want to review the justification for the larger amounts. This means that these journal entries should be fully documented.

Load the relevant client’s files into the R workspace.

```
rm(list=ls())
library(readr)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

# the real_world_ar_ledger file
```

```

# also contains the results of confirmation;
# the auditor takes a sample of client_ar_ledger records,
# and auditing obtains information from the real world
# which is simulated here by
# left_join(client_ar_ledger[{sample},], real_world_ar_ledger)

real_world_fyear_end_ar_ledger <-
  read.csv(system.file("extdata", "real_world_fyear_end_ar_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0",
  stringsAsFactors=FALSE)

fyear_end_ar_ledger <-
  read.csv(system.file("extdata", "fyear_end_ar_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0",
  stringsAsFactors=FALSE)

customer_credit_limits <-
  read.csv(system.file("extdata", "customer_credit_limits.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0",
  stringsAsFactors=FALSE)

sales_journal <-
  read.csv(system.file("extdata", "sales_journal.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0",
  stringsAsFactors=FALSE)

real_world_credit_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0",
  stringsAsFactors=FALSE)

real_world_cash_sales <-
  read.csv(system.file("extdata", "real_world_cash_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0",
  stringsAsFactors=FALSE)

```

Foot the client_ar_ledger file and aggregate the total to the Accounts Receivable entry on the Trial Balance. Compute the Accounts Receivable balances by customer in preparation for confirmations.

```

library(tidyverse)
library(lubridate)
library(kableExtra)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

# set the fiscal year end date to match the dataset
fyear_end <- paste0(format(Sys.Date(), "%Y"), "-12-31")

## create the unpaid_ar
sales_journal <-
  read.csv(system.file("extdata", "sales_journal.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0", stringsAsFactors=FALSE)

sales_journal[is.na(sales_journal)] <- 0

credit_sales_journal <-

```

```

sales_journal %>%
  filter(cash_not_ar == 0)

unpaid_ar <-
  credit_sales_journal[
    credit_sales_journal$collection_date >= fyear_end &
      credit_sales_journal$shipper_date <= fyear_end,]

foot <-
  unpaid_ar %>%
  select(sales_extended) %>%
  sum()

cat("\n\n A/R balance = ", foot)

## 
## 
## A/R balance = 150628

# create a list of unpaid AR by customer

unpaid_ar_by_cust <-
  unpaid_ar %>%
  select(customer_no, invoice_date, invoice_no, sales_extended) %>%
  group_by(customer_no) %>%
  select(sales_extended) %>%
  summarise(customer_ye_balance=sum(sales_extended))

unpaid_ar_by_cust %>%
  kable(caption="Year End A/R by Customer",
    "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = F)

```

customer_no	customer_ye_balance
c00001	3192
c00002	1956
c00003	12645
c00004	21407
c00005	23245
c00006	18294
c00007	22915
c00008	17666
c00009	13518
c00010	15790

Tests of Supporting Evidence

Sample invoices from the accounts receivable aging report and compare them to supporting documentation to see if they were billed in the correct amounts, to the correct customers, and on the correct dates-trace invoices to the shipping log. The auditors will match invoice dates to the shipment dates for those items in the shipping log to see if sales are being recorded

in the correct accounting period. This can include an examination of invoices issued after the period being audited to see if they should have been included in a prior period.

This can be accomplished using t-tests that were introduced in interim testing. Attribute sampling size is determined using Cohen's power analysis (Cohen 1992), which is implemented in R's *pwr* package. If discovery sampling suggests that a particular transaction stream is out of control, then attribute estimation will help us decide on the actual error rate of the systems that process this transaction stream. Errors estimates from attribute samples may either be *rates of erroneous transactions* or, from a monetary unit sampling perspective, can be *rates of monetary error in the transaction stream*. We compute both in the following code chunk

```

library(readr)
library(pwr)
library(tidyverse)

size <- as.numeric(nrow(credit_sales_journal))
## data set size
Delta <- .05*size
## detect 5% occurrence error
sigma <- .3*size
## variability (guess ~1/3 rd)
effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")                         ## look for overstatement of earnings

cat("\n \n Attribute sample size for occurrence of error = ", ceiling(sample$n))

##
##
## Attribute sample size for occurrence of error = 224

size <-
  as.numeric(
    sum(
      credit_sales_journal$sales_unit_price*
        credit_sales_journal$sales_count))
## data set size
mu <-
  mean(
    credit_sales_journal$sales_unit_price*
      credit_sales_journal$sales_count)
## average value of transaction
Delta <- .05*mu
## detect 5% amount intolerable error
sigma <-
  sd(credit_sales_journal$sales_unit_price*
    credit_sales_journal$sales_count)
## variability
effect <- Delta/sigma

```

```

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater")    ## look for sales value too large

cat("\n Attribute sample size for amount of error = ", ceiling(sample$n))

##
## Attribute sample size for amount of error =  572

```

Acceptance Sampling

Acceptance sampling for substantive tests results in a decision of whether or not to “accept” the account as either (1) being fairly presented, or (2) containing material (intolerable) error. Error estimates from acceptance samples are in *amounts* that may be viewed on a continuous scale of value for statistical tests or discrete *dollar-unit* amounts for dollar (monetary) unit sampling. If it is found that a particular account balance is materially in error, the most common step is to present this to the client, and request that accounts be adjusted to fairly present that client firm’s financial position.

Note that parameter inputs that would yield a sample size of less than 1.3 from the function below stop calculations and result in an *error* in *uniroot* function termination of the *pwr.t.test* function. In this situation, the software is essentially telling you not to take any samples and skip this test. This is probably not advisable from a risk perspective. The alternate conclusion would be that this implies one or more of the particular settings chosen for (1) dataset size, (2) detection occurrence rate, (3) variability, or (4) effect size do not make sense. They should be revised, generally in the following order: (1) variability, (2) detection occurrence rate, (3) effect size, which interim testing will be the *intolerable* or *out-of-control* error rate.

```

library(readr)
library(pwr)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

Client_ar_ledger <- read_csv((system.file("extdata", "fyyear_end_ar_ledger.csv", package =
"auditanalytics", mustWork = TRUE))
, col_types = cols(X1 = col_skip()))

## data set size
size <- as.numeric(nrow(Client_ar_ledger))

## average value of transaction
mu <- mean(Client_ar_ledger$amount)

Delta <- .1*mu                                # detect 10% (material) error in sample

sigma <- sd(Client_ar_ledger$amount)          # variability

effect <- Delta/sigma

sample <- pwr.t.test(
  d=effect,
  sig.level = 0.05,

```

```

power = .8,
type="one.sample",
alternative="greater") ## look for AR value too large

cat("\n Acceptance sample size = ", ceiling(sample$n))

```

The sample size then can be used in the following code chunk example to test for errors in the account. Assume that the sample size is 15.

```

library(broom)
library(readr)
library(tidyverse)
library(compare)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

Client_ar_ledger <- read_csv(system.file("extdata", "fyear_end_ar_ledger.csv",
  package = "auditanalytics", mustWork = TRUE),
, col_types = cols(X1 = col_skip()))

real_world_ar <-
  read_csv(system.file("extdata", "real_world_fyear_end_ar_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

acceptance_sample_ar <- Client_ar_ledger[runif(15,1,
  nrow(Client_ar_ledger)),]

audited_sample <-
  inner_join(
    acceptance_sample_ar,
    real_world_ar,
    by="invoice_no") %>%
  select(
    invoice_no,
    customer_no.x,
    amount.x,
    confirm_exception,
    confirm_response)

exceptions <- audited_sample[audited_sample$confirm_exception == 1,]

occ_error_rate_ar <- nrow(exceptions) / nrow(acceptance_sample_ar)
amt_error_rate_ar <- sum(exceptions$amount.x) / sum(acceptance_sample_ar$amount)

cat("\n\n Occurrence error in Accounts Receivable on the Trial Balance "
, occ_error_rate_ar)

## 
## 
## Occurrence error in Accounts Receivable on the Trial Balance  0.1333333

cat("\n\n
  Amount error in Accounts Receivable on the Trial Balance ", amt_error_rate_ar)

```

```
##
##
## Amount error in Accounts Receivable on the Trial Balance 0.2029926
```

Accounts Receivable Confirmation

A major concern with accounts receivable is whether the amount reflected in the customer's subsidiary ledger reconciles with the correct customer balance. When you audit accounts receivable, you typically will employ confirmations (i.e., letters to customers asking them to validate the amounts they owe) to make sure the amounts reflected in accounts receivable are accurate. Sending confirmations is not mandatory, but is typically a part of any full audit of the balance sheet. To confirm accounts receivable, you need to reconcile the accounts receivable subsidiary ledger to all the customers who owe the company money to the total amount shown on the balance sheet. Confirmation assesses whether:

1. transactions represented in accounts receivable are valid, authentic obligations of third parties.
2. customer balance is correct.
3. uncollectible accounts have been properly written off.

The accounts receivable subsidiary ledger report, also known as an aged accounts receivable report, shows all the amounts that customers owe to the company, grouped by the number of days outstanding as of a designated date. The normal grouping is as follows:

- Current (under 30 days outstanding)
- 30–60 days
- 61–90 days
- 91–120 days
- More than 120 days

The first two assessments—verifying customer existence and the correctness of customer balances—require confirmation requests.

These requests are form letters sent to customers listed in the accounts receivable subsidiary ledger to verify the facts and figures contained in the client's books. The confirmation form letter is short and lists the total amount that a customer owes at a certain date. Two types of confirmation requests are available:

1. Positive confirmation: A positive confirmation request asks that a customer sign and mail the form back to you, either attesting to the fact that the figure owed is correct or correcting it. A positive confirmation request can also show the dollar amount owed as blank, requiring the customer to fill it according to its records.
2. Negative confirmation: A negative confirmation request asks the customer to reply only if the figure shown on your letter is incorrect.

Additionally, under generally accepted accounting principles (GAAP), your client must allow for (i.e., estimate) uncollectible accounts. This typically implies comparing the previous year's uncollectible accounts with the amount estimated for the audit year to see whether uncollected debts are increasing or decreasing. You should peruse customer files to find out what action was taken by the firm to verify that accounts were indeed uncollectible.

The Role of Confirmations as Audit Evidence

Confirmations have been considered a strong form of evidence for nearly a century, largely because the procedure involves direct communication with independent sources outside the entity being audited. Accounts receivable confirmation was an optional audit procedure in the USA and not widely used before 1938. The McKesson–Robbins fraud used fraudulent receivables to cover up a bootlegging operation run by mobster Phillip Musica, and involved recording fictitious sales, receivables, and inventories. The Securities and Exchange Commission launched a major investigation to determine the adequacy of audit procedures generally in use. The accounting profession responded by issuing Statement on Auditing Procedures (SAP) no. 1, Extensions of Auditing Procedure, which recommended confirmation of receivables “wherever practicable and reasonable.”

In the early 1940s, a number of modifications were made to auditing standards involving confirmation. The most significant was SAP no. 12, Amendment to Extensions of Auditing Procedure. Auditors were required to disclose any situation in which confirmation of receivables with debtors was not performed, in effect making confirmation a mandatory procedure. The SAP no. 12 disclosure requirement was rescinded in 1974, but by then, confirmation of receivables was an ingrained procedure in US audit practice.

Statement on Auditing Standards no. 67, The Confirmation Process, issued in November 1991, dictates current US audit procedures on confirmations. The statement defines the confirmation process, relates confirmation evidence to risk assessment and financial statement assertions, and discusses the positive and negative confirmation forms and the situations in which they can be most helpful. It also discusses the problem of non-response to positive-form confirmations and the need to employ alternative procedures “to obtain the evidence necessary to reduce audit risk to an acceptably low level.”

The American Institute of CPAs auditing procedure study, Confirmation of Accounts Receivable, issued in 1984, provides additional guidance. The study, which discusses confirmation evidence in terms of financial statement assertions, maintains that confirmation is a primary source of evidence regarding assertion and a secondary source of evidence regarding the completeness and valuation assertions.

Confirmations and Experimental Design

Auditors have long suspected confirmation evidence is biased. According to a Canadian Institute of Chartered Accountants (CICA) 1980 audit technique study, “reliance on a test consisting only of replies means that the test is determined by the responding and non-responding debtors instead of the auditor, and it may no longer be representative or appropriate.” The CICA study also expressed concern with respondent apathy and “say yes” behavior (when customers confirm balances as correct without actually checking their records), which result in unreliable evidence: “The danger of using the communication technique in these circumstances is that the auditor may use it just to conform with professional standards and not to add audit assurance.”

The CICA study concluded that confirmations are best studied as a part of a carefully designed experiment—for several reasons:

1. The experimenter controls the size of the errors.
2. The experimenter controls which accounts are seeded with errors.
3. The experimenter controls the frequency of overstatement and understatement errors.
4. Other factors that may affect reliability can be studied, such as account balance size and age and transaction volume.
5. The underlying cause of any reporting bias can be isolated, making corrective action possible.

Audit Program Procedures for Accounts Receivable Confirmation

Timing of Confirmation Request

For both positive and negative confirmation requests, the debtor is provided with the balance as of a specified date. The date may be as follows:

1. Year-end date.
2. Date prior to the year-end. This date generally is one or two months prior to year-end.

It is recommended that confirmation requests be sent to debtors approximately a week before the date specified in the request. If the debtor is in a foreign country, the request should be mailed earlier.

Confirming Prior to Year-End

The auditor may decide to request that the debtor confirm the balance as of a date before the year-end. If the auditor follows this procedure, however, he or she should perform the following procedures during the year-end procedures:

1. Perform selective other substantive tests of transactions from the confirmation date to the balance sheet date. These tests would include the following:
 - a. Review subsequent sales invoices and related bills of lading.
 - b. Review subsequent customer cash receipts and related remittance advice.
2. If balances change significantly from confirmation date to year-end, it is recommended that the auditor reconfirm.
If the negative form of confirmation request is used, the auditor should normally do one of the following:
 1. Send out more requests than if the positive form is used.
 2. Apply other auditing procedures to a greater extent than if the positive form is used. Other auditing procedures include the examination of the following:
 - a. Subsequent cash receipts.
 - b. Subsequent cash remittance advices.
 - c. Sales and shipping documents.

Steps in Confirmation Process

The steps in the process of confirming receivables follow:

- Step-1. Obtain Aged Schedule of Accounts Receivable
 1. The auditor should obtain an aged schedule of accounts receivable as of the confirmation date.
 2. Determine that totals are correct.
 3. Compare all or a selected sample of account balances with the account balances in the accounts receivable subsidiary ledger.
 4. Investigate credit balances.
- Step-2. Select Accounts for Confirmation
 1. Auditors have used, and some continue to use, judgment in selecting accounts for confirmation. Statistical sampling methods, however, are ideal for the selection process.
 2. Whatever method of selection is used, the auditor generally considers the following accounts:
 - a. All accounts with a balance over a pre-determined amount. The pre-determined amount is based on the auditor's assessment of materiality.
 - b. Some or all accounts with zero balances.
 - c. Accounts with old unpaid items, especially when subsequent sales have been paid.
 - d. Accounts are written off during the year under review.
 - e. Accounts with entities related to the client but not audited by the auditor.
 - f. Certain accounts that appeared on the prior year's accounts receivable schedule but not on the current year's.
 3. Accounts with credit balances:
 - a. Occasionally, the client will not want confirmation requests sent to these accounts. If the amounts are material, it might result in a scope limitation; however, this is generally not the case.
 - b. If accounts with credit balances are not confirmed, alternative auditing procedures should be applied.
 4. Of the remaining accounts, a representative portion, both in dollar amount and number of accounts, should be selected.
- Step-3. Prepare and Mail Confirmation Requests: The auditor should observe the following procedures in preparing and mailing confirmation requests:

1. Prepare a schedule of accounts to be confirmed.
 - a. Assign each account a number. This number also should be placed on the confirmation request.
 - b. Total the dollar amount of receivables selected for confirmation and compute as a percentage of the total dollar amount of the receivables.
 - c. Determine the number of confirmation requests and compute as a percentage of the total number of accounts.
 - d. Leave sufficient blank columns after the customer's name to insert the following information when the confirmation reply is received: (1) Date reply received; (2) Amount confirmed; and (3) Explanation of the difference between the amount customer confirmed and client amount.
 - e. Leave a blank column for insertion of the date the second request was mailed.
 - f. Indicate at the bottom the date the first requests were mailed.
 2. Request that client address confirmation forms and prepare customer statements.
 - a. If the auditor desires that client does not know which accounts are to be confirmed, he or she should have his or her staff address confirmations.
 - b. If the auditor desires that client does not know which accounts are to be confirmed but wants the client to address confirmations, he or she should request the client to address confirmation to all accounts and then eliminate the accounts not selected for confirmation.
 3. When the auditor receives the addressed confirmation with the account balance and the customer statement, he or she should compare that balance with the balance on the schedule.
 4. Independently, some customer addresses should be checked. These tests can be made by comparing the address on confirmation with the address in the telephone book.
 5. After confirmations have been reviewed and numbered, the auditor should insert them and the customer statement in his or her firm's envelopes, that is, envelopes with the firm's return address.
 6. In addition to inserting the confirmation request in the envelope, insert a postage-paid return envelope bearing the auditor's address.
 7. When the requests have been stamped, the auditor should mail them.
 8. From the time the auditor receives the addressed confirmation requests containing the account balances, he or she should never lose control. The confirmation requests always should remain in the auditor's custody or under their supervision until mailed.
- Step-4. Process Responses to Confirmation Requests: When confirmation replies are received, the auditor should do the following:
 1. Enter for each account the following:
 - a. Date received.
 - b. Amount confirmed.
 2. If the amount confirmed differs from the account balance, the following should be done:
 - a. Photocopy confirmation reply.
 - b. Give photocopy to the client and request that the difference is reconciled and provide documentation for reconciling items.
 - c. Review documentation for reconciling items.
 - d. If documentation is satisfactory, enter reasons for the difference in receivable confirmation schedule.
 3. If the amount confirmed differs from the account balance and the client cannot satisfactorily reconcile the difference, the auditor should do the following:
 4. If the difference is small, the auditor may ignore it. If there are a significant number of small differences, however, the auditor should analyze them. If the analysis of the significant number of small differences indicates a deficiency in the receivable controls, the auditor may have to apply additional auditing procedures to satisfy himself or herself of the accounts receivable balance.
 5. If the difference is significant, request the client to correspond with the debtor. Make certain the correspondence states that the debtor response should be sent directly to the auditor.
 - Step-5. Summarize Confirmation Results: Near the conclusion of the engagement, the auditor should prepare a worksheet summarizing confirmation results. The worksheet should contain the following:

1. Number and dollar amount of confirmations sent and the percentage of these to the total receivables.
2. Number and dollar amount of confirmations received with no exceptions indicated and the percentage of these to the total confirmations requests.
3. Number and dollar amount of confirmations received with exceptions that were satisfactorily reconciled by the client. Compute the percentage of these to the total confirmations requested.
4. Number and dollar amount of confirmations received with exceptions that were not satisfactorily reconciled by the client.
5. Determine the total dollar amount of differences between client records and confirmation responses.
6. Determine the reasons for differences and materiality of differences.
7. Compute the percentage of these to the total confirmations requested.
8. Review statistics and determine if the results of the confirmation procedures provided sufficient competent evidential matter as to the existence of the receivables. If the auditor is not satisfied with the results of the confirmation procedures, he or she should perform other procedures such as the following:
9. Review subsequent cash receipts and accompanying remittance advice.
10. Review individual sales invoices and related shipping documents.

Non-response to Confirmation Requests

If a response to a confirmation request is not received within a reasonable period of time—two to three weeks—a second request should be sent. The auditor should note in the receivable confirmation worksheet the date the second request was mailed. If the non-response pertains to an account with a significant balance, the auditor should consider making a telephone call to the customer. If the auditor confirms by telephone, he or she should do the following:

1. Obtain the name and title of the person providing the information.
2. Request that the information provided be confirmed in writing.

If a confirmation request is returned to the auditor because it was not delivered, the auditor should do the following:

1. Determine customer's new address and mail confirmation request.
2. If customer went out of business, ascertain that client has established appropriate allowance.

Confirmation Responses Not Expected

Sometimes the auditor does not expect a response to a confirmation request based on past experience with the entity or with customers similar to those of the entity. When the auditor does not expect a response to a traditional confirmation request, he or she should do the following:

1. Request confirmation of specific items included in the account balance.
2. Review subsequent customer remittances. Where these amounts are significant, it is recommended that for a period of time subsequent to the balance sheet date, the auditor be present whenever the client receives mail. The auditor should open all mail from customers unable to confirm balances and compare remittance advices to ledger balances.
3. Undertake other procedures to validate the existence of the customer and sales to the customer. (For example, the customer could be looked up in a phone directory and called).
4. When fraud risk factors are present and confirmation of receivables is not possible, the auditor should employ unusual procedures if necessary to validate the existence of the customer and the sales to that customer.

```
library(pwr)
library(tidyverse)
library(kableExtra)

intolerable_error <- 5000

## Effect size (Cohen's d):
```

```

## difference between the means from H0 & Ha divided by the pooled standard deviation
cohen_d <- intolerable_error / sd(unpaid_ar_by_cust$customer_ye_balance)

sample_size <- pwr.t.test(
  d=cohen_d,
  sig.level = 0.05,
  power = .8,
  type="one.sample",
  alternative="greater") ## look for overstatement of AR = assets overstated

ar_sample <- sample_n(unpaid_ar_by_cust, ceiling(sample_size$n), replace=T)

kable(ar_sample,
      caption = "Customer Balances for Sampling and Confirmation",
      "latex",
      booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

```

customer_no	customer_ye_balance
c00004	21407
c00005	23245
c00006	18294
c00003	12645
c00009	13518
c00008	17666
c00003	12645
c00009	13518
c00006	18294
c00010	15790
c00001	3192
c00001	3192
c00004	21407
c00010	15790
c00001	3192
c00010	15790

Confirmation and Estimation of Error in Account

Once the auditor has completed the confirmation and follow-up for confirmation of accounts receivable, the sample results can be used to compute an upper bound on the accounts receivable balance. Were the client to post a balance in excess of this upper bound, the accounts receivable balance could be considered intolerably (materially) in error.

```

library(tidyverse)
library(lubridate)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

```

```

real_world_credit_sales <-
  read.csv(system.file("extdata", "real_world_credit_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), na.strings="0", stringsAsFactors=FALSE)
real_world_fyear_end_ar_ledger <-
  read_csv(system.file("extdata", "real_world_fyear_end_ar_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

## create real_world_unpaid_ar & real_world_ar_by_customer
## note that these 'real_world' values are not available to the auditor
## the auditor extracts information about them
## from a sample of obtained after the A/R audit confirmation process

real_world_unpaid_ar <-
  real_world_credit_sales[
    real_world_credit_sales$collection_date >= fyear_end &
      real_world_credit_sales$shipper_date <= fyear_end,]

real_world_ar_by_customer <-
  real_world_unpaid_ar %>%
  group_by(customer_no)

real_world_ar_by_customer$sales_extended <-
  real_world_ar_by_customer$sales_count *
  real_world_ar_by_customer$sales_unit_price

real_world_ar_by_customer %>%
  group_by(customer_no) %>%
  summarize(customer_ye_balance = sum(sales_extended))

## use the clients transaction files to compute the sample as a % of population

post_confirm_ar_balances <-
  left_join(ar_sample,real_world_fyear_end_ar_ledger, by="customer_no")

post_confirm_ar_balances[is.na(post_confirm_ar_balances)] <- 0
post_confirm_ar_balances <-
  post_confirm_ar_balances %>%
  mutate(balance_per_customer = customer_ye_balance * confirm_pct) %>%
  select(customer_no, customer_ye_balance, balance_per_customer) %>%
  mutate(overstatement=customer_ye_balance - balance_per_customer)

mean_trans_error <- mean(post_confirm_ar_balances$overstatement)
sd_trans_error <- sd(post_confirm_ar_balances$overstatement)
population_error_est_95_ub <-
  nrow(fyear_end_ar_ledger) *
  1.96 * sd_trans_error +
  mean_trans_error

cat("\n\n"
  Population error estimate 95% upper conf bound = ", population_error_est_95_ub)

```

Population error estimate 95% upper conf bound = 820351.3

Post-Confirmation Tests

After confirmations are received, and secondary follow-up procedures are pursued to confirm the balances, the following code chunk presents the results.

```
library(kableExtra)

ar_working_papers <-
  full_join(stratum_est,real_world_fyear_end_ar_ledger, by="customer_no")

kable(ar_working_papers, caption = "Audit Workpapers for A/R Confirmation","latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")
```

Post-confirmation, the figure of interest is the error in the population (and whether or not it is material).

Probability Proportional to Size (PPS) Sampling

I investigate the performance of stratified and PPS sampling to estimate population error, starting with a file of errors in the confirmed A/R balances. Clearly, this will not be available to the auditor; rather, only a sample of it will be. I provide it to benchmark the performance of a 3-stratum and PPS approach to sampling. I calculate overstatements of the A/R account because the Conservatism principle dictates that I want to find errors that overstate income (thus A/R balance overstated).

The R *survey* package supports PPS (i.e., arbitrary unequal probability) sampling with replacement, or using them with-replacement single-stage approximation to a multistage design. No special notation is required: just specify the correct sampling weights. To specify a PPS design, the sampling probabilities must be given in the prob argument of *svydesign*, or in the *fpc* argument, with prob and weight unspecified. In addition, it is necessary to specify which PPS computation should be used, with the *pps* argument. The optional variance argument species the Horvitz–Thompson (variance=“HT”) or Yates–Grundy (variance=“YG”) estimator, with the default being “HT.”

```
library(TeachingSampling)
library(tidyverse)
library(broom)
library(knitr)
library(kableExtra)

unpaid_ar <-
  sales_journal[1:15] %>%
  filter(cash_not_ar == 0 &
         collection_date >= fyear_end &
         shipper_date <= fyear_end) %>%
  select(-X)

confirmed_unpaid_ar <-
  real_world_credit_sales %>%
  filter(collection_date >= fyear_end &
         shipper_date <= fyear_end) %>%
  select(-X) %>%
  select(customer_no,
         invoice_no,
         invoice_date,
         sales_count,
         sales_return,
         sales_unit_price) %>%
  mutate(real_sales = sales_unit_price * (sales_count - sales_return)) %>%
  left_join(unpaid_ar, by="invoice_no") %>%
  mutate(overstatement = sales_extended - real_sales) %>%
  select(customer_no ~customer_no.x,
         invoice_no = invoice_no,
         invoice_date = invoice_date.x,
         overstatement)
confirmed_unpaid_ar[is.na(confirmed_unpaid_ar)] <- 0
```

```

## Separate the dataset into three strata and set the sample proportions
confirmed_unpaid_ar$stratum <-
  cut(confirmed_unpaid_ar$overstatement, breaks=3, labels=c("small", "medium", "large"))

sample_prop <- data.frame(prop = c(.05,.1,.2), stratum = c("small", "medium", "large"))

# Define the size of each stratum
N <- NULL
for(i in 1:3) N[i] <- summary(confirmed_unpaid_ar$stratum)[[i]]
Nh <- as.numeric(N[1:3]) ## vector of stratum sizes
nh = as.numeric(ceiling(N * sample_prop$prop)) ## vector of samples size by stratum

# Draw a stratified sample and estimate
samp <- S.STSI(confirmed_unpaid_ar$overstatement, Nh, nh)
ar_sample <- confirmed_unpaid_ar[samp,]

est <- data.frame(confirmed_unpaid_ar$overstatement)
estimate <- E.STSI(confirmed_unpaid_ar$stratum, Nh, nh, est)

## normalize the estimate to reflect the number of transactions in the client's file
normalized_estimate <- nrow(confirmed_unpaid_ar) * estimate[,2]/estimate[1,,1]
normalized_estimate <- round(normalized_estimate, digits=0)

tidy(normalized_estimate) %>%
kable(caption = "3-Stratum Horvitz-Thompson estimate of Total Error in A/R Balance") %>%
  kable_styling(bootstrap_options = "striped")

# Draw a PPS sample of size 'size' and estimate draw a random sample according to a
# PPS with replacement design
size <- 5
sample_prop <- S.PPS(size,confirmed_unpaid_ar$overstatement)
# returns a matrix of m rows and two columns.
# Each element of the first column indicates the unit that was selected.
# Each element of the second column indicates the selection probability of this unit

sample_ar <- confirmed_unpaid_ar[sample_prop[,1],]

est <- data.frame(confirmed_unpaid_ar$overstatement)
estimate <- E.PPS(est,sample_prop[,2])

## normalize the estimate to reflect the number of transactions in the client's file
normalized_estimate <- nrow(unpaid_ar) * estimate[,2]/estimate[1,1]
normalized_estimate <- round(normalized_estimate, digits=0)

tidy(normalized_estimate) %>%
kable(caption = "PPS Hansen-Hurwitz Estimate of Overstatement of A/R account balance","latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

actual_overstatement <- sum(confirmed_unpaid_ar$overstatement)
cat("\n\n Actual Overstatement of A/R balance in Population = ",actual_overstatement)

```

Estimation and Accrual of the Allowance for Doubtful Accounts

The Allowance for Doubtful (uncollectable) Accounts is a contra-account, i.e., it offsets the accounts receivable account balance to calculate the net revenues that have been earned. Some of the accounts receivable at year-end will not be paid, but we do not know which of these will not be paid (otherwise, the client probably would not have sold to this customer). These expenses must be reported as a contra-account (Cr balance) to the accounts receivable account (Dr balance). The estimation policy that the client applies will be based on historical payments and unpaid debt write-offs.

This section designs a predictive algorithm for credit defaults and slow payments—both necessary for the accurate calculation of customer credit limits and the total allowance for uncollectable accounts.

Aging of accounts receivable provides, along with estimates of default rates for each age-class of receivable, a crude method for estimating time to payment and the amount of accounts receivable, which will be unpaid. Traditional accruals ignore time-series trends, total experience with particular customers, and co-factors that influence payments. R provides statistical tools that can substantially improve on traditional methods for accruing the Allowance for Uncollectable Accounts and computing customer credit limits.

The following code chunks present examples of predictive algorithms to estimate the future collection of accounts receivable. Though there are various approaches to estimating when customers will pay their accounts, if at all. I adopt a “time to payment” estimate, where uncollectable accounts will theoretically have infinite time to payment. At year-end, the longer an outstanding receivable has remained unpaid, the more likely we are to decide that the time to payment will be infinite. Just as with traditional methods for estimating the Allowance for Uncollectable A/R, I apply a particular decision model for estimating uncollectible based on the predicted aging of A/R

```
library(tidyverse)
library(ggplot2)
library(lubridate)

sales_journal[is.na(sales_journal)] <- 0

credit_sales_journal <-
  sales_journal[, 1:15] %>%
  filter(cash_not_ar == 0) %>%      ## credit only
  mutate(collection_delay =
    .01 + time_length(          ## add .01 to get rid of 0's so logs can be computed
      ymd(collection_date) -
      ymd(invoice_date),
      unit = "day"),
    customer_no =
    as.character(customer_no)
  ) %>%
  as.data.frame()

## Simple OLS regression

l_mod <-
  lm(collection_delay ~
    customer_no +
    unit_cost +
    sales_unit_price +
    sales_count +
    sales_return,
    data=credit_sales_journal)

summary(l_mod)

##
## Call:
## lm(formula = collection_delay ~ customer_no + unit_cost + sales_unit_price +
##     sales_count + sales_return, data = credit_sales_journal)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -71.75 -39.56 -13.12  21.51 293.25 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  10.0000   10.0000  1.0000  0.3162    
## customer_no  0.0000    0.0000  0.0000  0.0000    
## unit_cost   -0.0000    0.0000  0.0000  0.0000    
## sales_unit_price  0.0000    0.0000  0.0000  0.0000    
## sales_count   0.0000    0.0000  0.0000  0.0000    
## sales_return  0.0000    0.0000  0.0000  0.0000
```

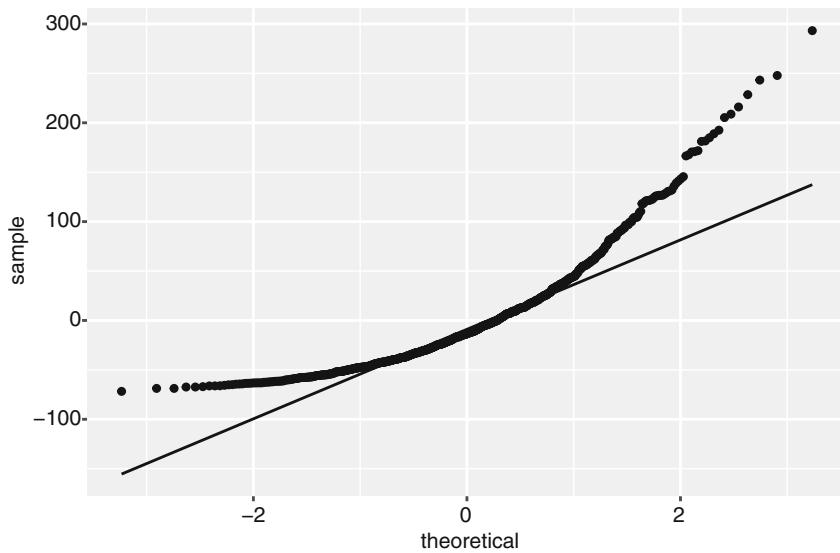


Fig. 1 Linear model of time to payment, residuals QQ plot

```

## (Intercept)      78.05837   21.29274   3.666  0.000263 *** 
## customer_noc00002 28.89028   26.88501   1.075  0.282881 
## customer_noc00003  2.49338   16.32973   0.153  0.878681 
## customer_noc00004  5.81833   15.63983   0.372  0.709976 
## customer_noc00005  7.50659   15.51915   0.484  0.628731 
## customer_noc00006  0.11012   15.93854   0.007  0.994489 
## customer_noc00007 -1.17515   15.51218  -0.076  0.939632 
## customer_noc00008 -2.39264   15.70583  -0.152  0.878956 
## customer_noc00009 -3.54437   15.41579  -0.230  0.818214 
## customer_noc00010  2.43218   16.74833   0.145  0.884574 
## unit_cost          0.04830   0.37271   0.130  0.896926 
## sales_unit_price  -0.10643   0.07275  -1.463  0.143860 
## sales_count         -0.27008   0.48400  -0.558  0.576988 
## sales_return        -15.33614   9.55476  -1.605  0.108868 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 55.05 on 807 degrees of freedom 
## Multiple R-squared:  0.01402,    Adjusted R-squared:  -0.001865 
## F-statistic: 0.8826 on 13 and 807 DF,  p-value: 0.5715

```

```

sq <- seq(1,length(l_mod$residuals))
res <- data.frame(sq,l_mod$residuals)

ggplot(res, aes(sample=l_mod$residuals)) +
  stat_qq() +
  stat_qq_line() +
  labs(title = "Linear Model of Time to Payment, Residuals QQ Plot")

```

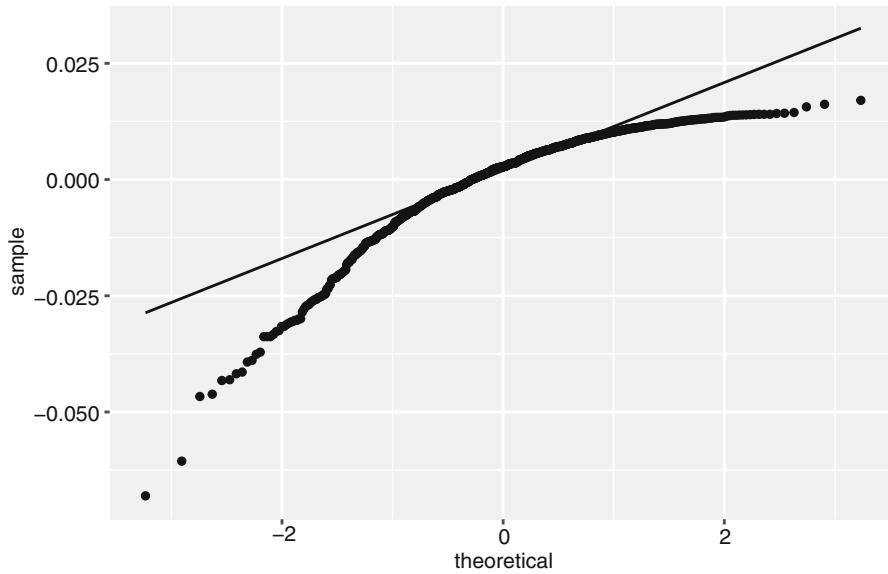


Fig. 2 GLM model of time to payment, residuals QQ plot

GLM-Exponential Regression

Time to payment can be better modeled using an exponential distribution, which is commonly used to model waiting times for Poisson processes. The exponential distribution is a specific Gamma distribution where the dispersion parameter (which is what distinguishes an exponential distribution from the more general Gamma distribution) does not affect the parameter estimates in a generalized linear model, only the standard errors of the parameters/confidence intervals/p-values, etc. The Gamma family is parameterized in `glm()` by two parameters: mean and dispersion; the “dispersion” regulates the shape. Therefore fit a GLM with the Gamma family, and then produce a “summary” with dispersion parameter set equal to 1, since this value corresponds to the exponential distribution in the Gamma family.

```
exp_mod <-  
  glm(collection_delay ~  
    customer_no +  
    unit_cost +  
    sales_unit_price +  
    sales_count +  
    sales_return,  
    data=credit_sales_journal,  
    family = Gamma(link = "inverse")  
)  
  
sq <- seq(1,length(exp_mod$residuals))  
res <- data.frame(sq,exp_mod$residuals)  
  
ggplot(res, aes(sample=exp_mod$residuals)) +  
  stat_qq() +  
  stat_qq_line() +  
  labs(title = "GLM Model of Time to Payment, Residuals QQ Plot")
```

Reviewing the Q-Q plot of residuals for the GLM model with Gaussian fit and an inverse link shows that it tends to overcorrect for the misfit in the linear model. Nonetheless, fit improves on a simple linear model. How well does the GLM model fit the collection time series? The following code chunk compares our predicted collections to actual collections. There is substantially more variance in the actual transaction flows than the predicted flows (which is to be expected), but

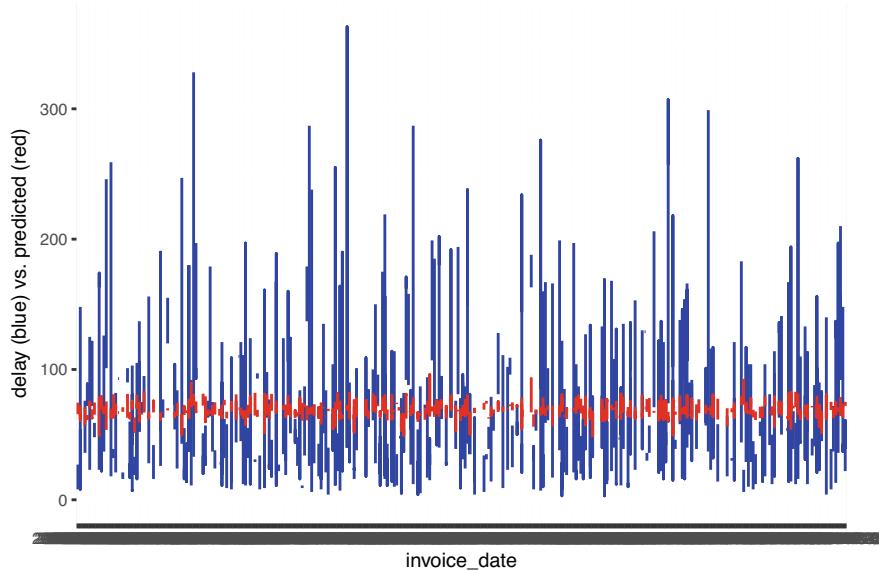


Fig. 3 GLM model predictions vs. original time series

the predictions are centered at the same place as the actual collections, and over time, this should be sufficient to accurately estimate the allowance for doubtful accounts.

```
## how well does the glm model predict?

predictors <-
  credit_sales_journal %>%
  select(customer_no,
         unit_cost,
         sales_unit_price,
         sales_count,
         sales_return
     )

pred_collection_delay <-
  1 / predict(exp_mod, dispersion=1, predictors)

plt <-
  credit_sales_journal %>%
  select(invoice_date, collection_delay) %>%
  cbind(pred_collection_delay)

ggplot() +
  geom_line(data = plt, aes(x = invoice_date, y = collection_delay), color = "blue") +
  geom_line(data = plt, aes(x = invoice_date, y = pred_collection_delay), color = "red") +
  xlab('invoice_date') +
  ylab('delay (blue) vs. predicted (red)')
```

Time-Series Forecasting

There are many methods for forecasting time series which can be broadly categorized as either (1) econometric methods, such as ARIMA; (2) signal processing methods, such as Fourier Analysis; or (3) machine learning methods which involve

recurrent neural networks (such as LSTM networks). In the current section, I demonstrate some standard ARIMA models that improve on existing methods for accruing the Allowance for Uncollectable A/R. Machine learning models show promise in more completely extracting information from client records. Forecasts from machine learning, or signal processing models can be used in the same fashion that is demonstrated here in the application of econometric models.

If we assume that there are time-dependent effects that influence the rate of payment, we could actually model payments history as an autoregressive integrated moving average (ARIMA) using R's time-series tools, as shown in the code chunk below.

```

library(ggplot2)
library(lubridate)
library(forecast)
library(tidyquant)
library(timetk)
library(sweep)
library(tidyverse)

sales_journal[is.na(sales_journal)] <- 0

## create a time-series object

credit_sales <-
  sales_journal[,1:15] %>%
  filter(cash_not_ar == 0) %>%      ## credit only
  mutate(collection_delay =
    time_length(
      ymd(collection_date) -
      ymd(invoice_date),
      unit = "day"),
    customer_no =
    as.character(customer_no)
  ) %>%
  as.data.frame() %>%
  group_by(collection_delay) %>%
  summarise_if(is.numeric, sum, na.rm = TRUE) %>%
  select(collection_amount) %>%
  ts(frequency=365, start = c(2019,1))

moving_sales <- ma(credit_sales , order=30) ## 1 month = 30 day moving average
#plot(moving_sales)

plot(moving_sales,
  xlab="Time to Pay",
  ylab="Amount Collected")

## Fit and forecast with auto.arima()
## The forecast package offers auto.arima() function to fit ARIMA models. It can also be
## manually fit using Arima(). A caveat with ARIMA models in R is that it does not have
## the functionality to fit long seasonality of more than 350 periods eg: 365 days for
## daily data or 24 hours for 15 sec data.

autoArimaFit <- auto.arima(credit_sales)
#plot(forecast(autoArimaFit, h=20))

plot(forecast(autoArimaFit, h=20),
  xlab="Time to Pay",
  ylab="Amount Collected")

sales_journal[is.na(sales_journal)] <- 0

## create a time-series object

```

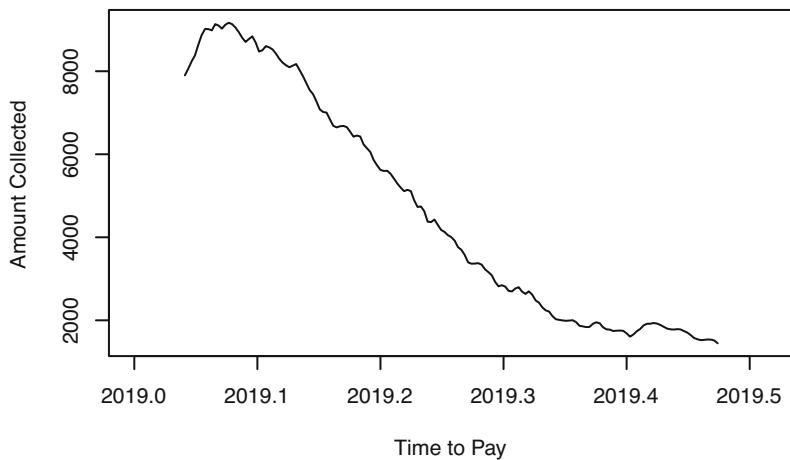
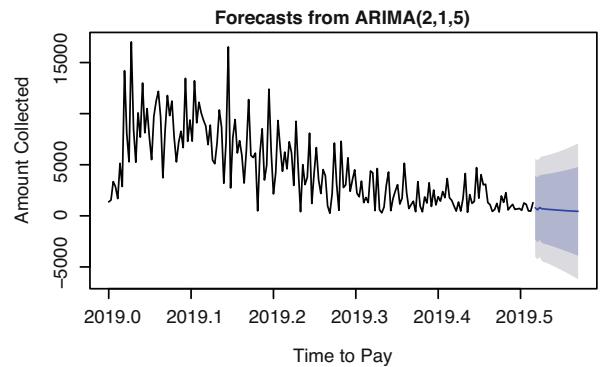


Fig. 4 Customer payment performance

Fig. 5 Customer payment performance



```
credit_sales <-
  sales_journal[,1:15] %>%
  filter(cash_not_ar == 0) %>%      ## credit only
  mutate(collection_delay =
    time_length(
      ymd(collection_date) -
      ymd(invoice_date),
      unit = "day"),
    customer_no =
      as.character(customer_no)
  ) %>%
  as.data.frame() %>%
  group_by(customer_no, collection_delay) %>%
  summarise(collections = sum(collection_amount)) %>%
  as.tibble()

credit_sales %>%
  ggplot(aes(x = collection_delay, y = collections, group = customer_no)) +
  geom_area(aes(fill = customer_no), position = "stack") +
  labs(title = "Customer Payment Performance", x = "Days to Pay", y = "Sales") +
  scale_y_continuous() +
  theme_tq()
```

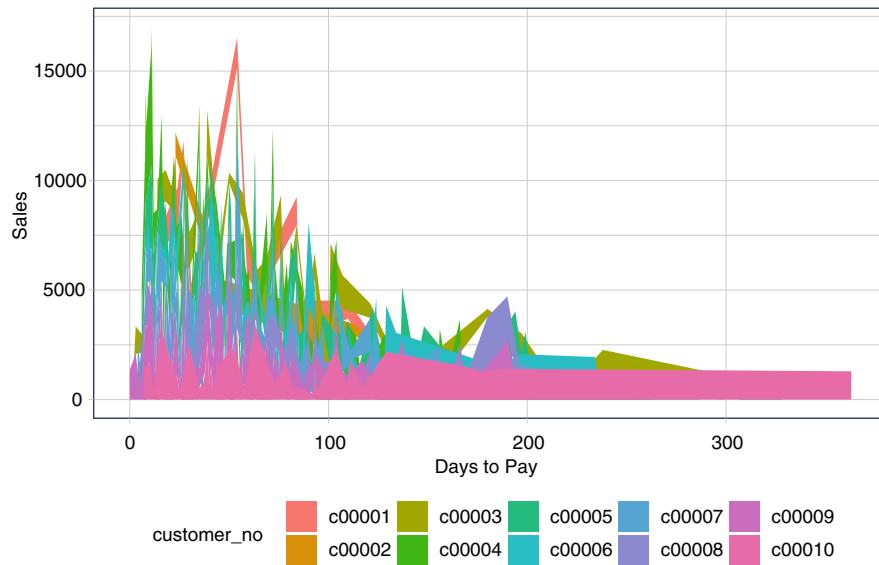


Fig. 6 GLM model of time to payment

Forecasting Accounts Receivable Collections

The forecasting workflow involves a few basic steps:

- Step 1: Coerce to a ts object class.
- Step 2: Apply a model (or set of models)
- Step 3: Forecast the models (similar to predict)
- Step 4: Tidy the forecast

First, I need to get the data organized into groups by months of the year.

```
daily_collections_by_customer <-
  sales_journal[,1:15] %>%
  filter(cash_not_ar == 0) %>% ## credit only
  mutate(invoice_day = decimal_date(as_date(invoice_date))) %>%
  group_by(customer_no, invoice_day) %>%
  summarise(daily_collections = sum(collection_amount)) %>%
  mutate(invoice_dt =
    round_date(date_decimal(invoice_day),
                unit = "day" ) ) %>%
  select(customer_no, invoice_date=invoice_dt, daily_collections)

# Next, we use the nest() function from the tidyverse package to consolidate each time series by
# group. The newly created list-column, 'data.tbl', contains the 'order.month'i and 'total.qty'
# columns by group from the previous step. The nest() function just bundles the data together
# which is very useful for iterative functional programming.

daily_collections_nest <-
  daily_collections_by_customer %>%
  group_by(customer_no) %>%
  nest()
```

```

### Step 1: Coerce to a ts object class

# In this step we map the tk_ts() function into a new column 'data.ts'. The procedure is
# performed using the combination of dplyr::mutate() and purrr::map(), which works really well
# for the data science workflow where analyses are built progressively. As a result, this
# combination will be used in many of the subsequent steps in this vignette as we build the
# analysis.

##### mutate and map

# The mutate() function adds a column, and the map() function maps the contents of a list-column
# (.x) to a function (.f). In our case, .x = data.tbl and .f = tk_ts. T

daily_collections_ts <-
  daily_collections_nest %>%
    mutate(data.ts = map(.x = data,
                         .f = tk_ts,
                         select = -invoice_date,
                         start = 2019,
                         freq = 365))

## Step 2: Modeling a time series

# Next, we map the Exponential Smoothing ETS (Error, Trend, Seasonal) model function, ets, from
# the forecast package. Use the combination of mutate to add a column and map to interactively
# apply a function rowwise to a list-column. In this instance, the function to map the ets
# function and the list-column is 'data.ts'. We rename the resultant column 'fit.ets' indicating
# an ETS model was fit to the time series data.

daily_collections_fit <-
  daily_collections_ts %>%
    mutate(fit.ets = map(data.ts, ets))




### sw_tidy -- do some model inspection with the sweep tidiers.
# To get the model parameters for each nested list, we can combine sw_tidy within the mutate
# and map combo.
# The only real difference is now we unnest the generated column (named 'tidy').
# because it's easier to compare the model parameters side by side, we add one additional call
# to spread() from the tidyverse package.

daily_collections_fit %>%
  mutate(tidy = map(fit.ets, sw_tidy)) %>%
  unnest(tidy) %>%
  spread(key = customer_no, value = estimate)

### sw_glance -- view the model accuracies also by mapping sw_glance within the mutate and
# map combo.

daily_collections_fit %>%
  mutate(glance = map(fit.ets, sw_glance)) %>%
  unnest(glance)




### sw_augment

augment_fit_ets <-
  daily_collections_fit %>%

```

```

  mutate(augment = map(fit.ets, sw_augment, timetk_idx = TRUE, rename_index = "date")) %>%
  unnest(augment)

# plot the residuals for the nine categories like so. Unfortunately we do see some very
# augment_fit_ets %>%
#   ggplot(aes(x = date, y = .resid, group = customer_no)) +
#   geom_hline(yintercept = 0, color = "grey40") +
#   geom_line(color = palette_light() [[2]]) +
#   geom_smooth(method = "loess") +
#   labs(title = "A/R Collection Delay by Customer",
#        subtitle = "ETS Model Residuals", x = "") +
#   theme_tq() +
#   theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
#   facet_wrap(~ customer_no, scale = "free_y", ncol = 3)

### sw_tidy_decomp -- create decompositions using the same procedure with sw_tidy_decomp() and
# the mutate and map combo.

daily_collections_fit %>%
  mutate(decomp = map(fit.ets, sw_tidy_decomp, timetk_idx = TRUE, rename_index = "date")) %>%
  unnest(decomp)

## Step 3: Forecasting the model

daily_collections_fcast <-
  daily_collections_fit %>%
  mutate(fcast.ets = map(fit.ets, forecast, h = 12))

## Step 4: Tidy the forecast -- apply sw_sweep to get the forecast in a nice 'tidy' idata frame.
# use the argument fitted = FALSE to remove the fitted values from the forecast

daily_collections_fcast_tidy <-
  daily_collections_fcast %>%
  mutate(sweep = map(fcast.ets, sw_sweep, fitted = FALSE, timetk_idx = TRUE)) %>%
  unnest(sweep)

## Visualization

daily_collections_fcast_tidy %>%
  ggplot(aes(x = index, y = daily_collections, color = key, group = customer_no)) +
  geom_ribbon(aes(ymin = lo.95, ymax = hi.95),
              fill = "#D5DBFF", color = NA, size = 0) +
  geom_ribbon(aes(ymin = lo.80, ymax = hi.80, fill = key),
              fill = "#596DD5", color = NA, size = 0, alpha = 0.8) +
  geom_line() +
  labs(title = "A/R Collection Delay by Customer",
       subtitle = "ETS Model Forecasts",
       x = "", y = "Units") +
  scale_color_tq() +
  scale_fill_tq() +
  facet_wrap(~ customer_no, scales = "free_y", ncol = 3) +
  theme_tq() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

```

Calculating Allowance for Uncollectable Accounts

Let the client's allowance for uncollectables accrual policy be: 1% of A/R under 45 days old are uncollectable; 2% of A/R between 45 days and 120 days old are uncollectable; 5% of A/R over 120 days old are uncollectable.

The statistical estimation from the time-series analysis previously completed provides confidence bounds for the forecast of optimistic, pessimistic, and expected uncollectables. This is “overkill” for audits, and I will restrict the application of the client's accrual policy for just the expected rate of collection of receivables, based on prior experience with each customer. Then we can apply the policy, using a forecast of daily collections and outstanding A/R using the following code chunk.

```

library(knitr)
library(kableExtra)
library(tidyverse)
library(lubridate)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

fyear_end <- paste0(format(Sys.Date(), '%Y'), '-12-31') # set the fiscal year end date to match the dataset

sales_journal <-
  read.csv(system.file("extdata", "sales_journal.csv", package = "auditanalytics", mustWork = TRUE)), na.strings="0", stringsAsFactors=FALSE)

credit_sales_journal <-
  sales_journal %>%
  filter(cash_not_ar == 1)

## calculate each customer's unpaid_ar

ar_by_customer <-
  credit_sales_journal[
    credit_sales_journal$collection_date >= fyear_end &
      credit_sales_journal$shipper_date <= fyear_end,] %>%
  mutate(age = ymd(fyear_end) - ymd(invoice_date)) %>%
  group_by(customer_no) %>%
  summarize(sum(sales_extended),
            average_age = mean(age)) %>%
  rename(customer_ye_balance='sum(sales_extended)')

## calculate each customer's forecasted collections
##(from the daily_collections_fcast_tidy data.frame)
##including 80 and 95% confidence limits

forecast_by_customer <-
  daily_collections_fcast_tidy %>%
  select(customer_no, key, daily_collections, lo.80, hi.80, lo.95, hi.95) %>%
  filter(key == "forecast") %>%
  group_by(customer_no) %>%
  summarize(f_collections = mean(daily_collections),
            f_lo.80 = mean(lo.80),
            f_lo.95 = mean(lo.95),
            f_hi.80 = mean(hi.80),
            f_hi.95 = mean(hi.95)
            ) %>%
  inner_join(ar_by_customer, by="customer_no") %>%
  mutate(days_to_pay = customer_ye_balance / f_collections,
        age_when_fully_paid = days_to_pay + average_age)

## apply the client's allowance for uncollectables estimation
##policy assuming the steady daily rate of payment from the ETS

allowance_for_uncollectable_ar <-
  forecast_by_customer %>%
  mutate(age_when_fully_paid =
            as.numeric(age_when_fully_paid),
        uc_lt_45 =
          ifelse(age_when_fully_paid <
                    45, customer_ye_balance * .01, 0),
        uc_45_120 =
          ifelse(age_when_fully_paid >
                    45 & age_when_fully_paid <= 120,
                (1 - 45 / age_when_fully_paid) *
                  customer_ye_balance * .02 +
                  (45 / age_when_fully_paid) *
                  customer_ye_balance * .01, 0),
        uc_gt_120 =
          ifelse(age_when_fully_paid >
                    120,
                (1 - 120 / age_when_fully_paid) *
                  customer_ye_balance * .05 +
                  (75 / age_when_fully_paid) * customer_ye_balance *
                  .02 + 45 *
                  f_collections * .01, 0),
        allowance = uc_lt_45 + uc_45_120 + uc_gt_120,
        )

```

```

    pct_of_ye_balance = allowance / customer_ye_balance
) %>%
select(customer_no, allowance, pct_of_ye_balance)

total_allowance <- sum(allowance_for_uncollectable_ar$allowance)
cat("Allowance for Uncollectable Accounts = ", total_allowance)

kable(allowance_for_uncollectable_ar, caption = "Accrual Calculations for Allowance for Uncollectables (by customer)", "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

```

Idiosyncratic Tests

Depending on the client's records, there may be specific transactions related to accounts receivable that will be audited:

- credit memos issued during the audit period will be audited for proper authorization, whether they were issued in the correct period, and whether the circumstances of their issuance may indicate other problems. They may also review credit memos issued after the period being audited to see if they relate to transactions from within the audit period.
- situations where the client is billing customers for sales despite still retaining the goods on-site (known as “bill and hold”), the auditors will examine your supporting documentation to determine whether a sale has actually taken place.
- review the receiving log to see if it records an inordinately large amount of customer returns after the audit period, which would suggest that the company may have shipped more goods near the end of the audit period than customers had authorized.
- related party receivables should be audited for collectibility, as well as whether they should instead be recorded as wages or dividends, and whether they were properly authorized.

Stratified Samples and Monetary Unit Sampling

In some cases, the sample designer has access to an “auxiliary variable” or “size measure,” believed to be correlated to the variable of interest, for each element in the population. These data can be used to improve accuracy in sample design. One option is to use the auxiliary variable as a basis for stratification.

Stratification has a long history in auditing as a method of audit cost control, focusing the auditing effort on higher value items. The standard approach to stratification involves simple partitionings of the transaction documents into low and high-value documents. During the 1960s and 1970s, sampling methods were developed around the idea of “monetary unit sampling,” which suggested that, for substantive auditing, individual dollars in an account balance should be sampled, rather than the transaction documents. Error estimation methods based on an assumption of Poisson distributions of monetary units were developed in Stringer (1961; 1963; 1975), but due to complexity and questions surrounding estimator characteristics, never saw widespread application.

Currently, monetary unit sampling uses methods developed for survey work called probability proportional to size (“PPS”) sampling, in which the selection probability for each element is set to be proportional to its size measure, up to a maximum of 1. Probability proportional to size (PPS) sampling is a method of sampling from a finite population in which a size measure is available for each population unit before sampling and where the probability of selecting a unit is proportional to its size. Its use arises in two particular contexts: (i) multistage sampling and (ii) single-stage sampling of establishments. Unbiased estimation is obtained using the Hansen–Hurvitz estimator in the case of PPS with replacement sampling and the Horvitz–Thompson estimator in the case of probability proportional to size without replacement (PPSWOR) sampling.

I provide examples below, using R’s *TeachingSampling* package for sampling and estimation with stratification and PPS. Note that the *TeachingSampling* package estimates both population size and amount from stratified and PPS samples. Since we already know the sample size from the client’s file, we can reduce uncertainty by converting to means and standard errors and then extrapolating to the population.

PPS and monetary unit sampling provide substantial increases in accuracy over stratified sampling with effective control of audit costs. They are also easier to set up (basically, no more difficult than using unstratified random sampling). The following code chunks create a 3-level stratification, and for comparison, a PPS sampling is based on the dollar value of the invoice amount.

```

library(TeachingSampling)
library(tidyverse)
library(broom)
library(knitr)

```

```

library(kableExtra)

unpaid_ar <-
  sales_journal[1:15] %>%
  filter(cash_not_ar == 0 &
         collection_date >= fyear_end &
         shipper_date <= fyear_end) %>%
  select(-X)

## Separate the dataset into three strata and set the sample proportions
unpaid_ar$stratum <-
  cut(unpaid_ar$sales_extended, breaks=3, labels=c("small", "medium", "large"))

sample_prop <- data.frame(prop = c(.05,.1,.2), stratum = c("small", "medium", "large"))

# Define the size of each stratum
N <- NULL
for(i in 1:3) N[i] <- summary(unpaid_ar$stratum)[[i]]
Nh <- as.numeric(N[1:3])                                ## vector of stratum sizes
nh = as.numeric(ceiling(N * sample_prop$prop))      ## vector of samples size by stratum

# Draw a stratified sample and estimate
ar_sample <- unpaid_ar[S.STSI(unpaid_ar$stratum, Nh, nh),]
est <- data.frame(unpaid_ar$sales_extended)
estimate <- E.STSI(unpaid_ar$stratum, Nh, nh, est)

## normalize the estimate to reflect the number of transactions in the client's file
normalized_estimate <- nrow(unpaid_ar) * estimate[,2]/estimate[1,,1]
normalized_estimate <- round(normalized_estimate, digits=0)

tidy(normalized_estimate) %>%
kable(caption = "3-Stratum Horvitz-Thompson estimate of Total Unpaid A/R", "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

```

Table 1 3-stratum
Horvitz-Thompson estimate of
total unpaid A/R

.rownames	Small	Medium	Large	Population
Estimation	91072	84362	14863	190297
Standard error	1886	1056	359	2191
CVE	2	1	2	1
DEFF	0	0	0	0

PPS

R contains other routines to implement probability proportional to scale sampling, which is used in population studies and other areas (Horvitz and Thompson 1952). The following code shows how this can be applied in an audit setting.

```

library(TeachingSampling)
library(tidyverse)
library(broom)
library(knitr)
library(kableExtra)

unpaid_ar <-
  sales_journal[1:15] %>%
  filter(cash_not_ar == 0 &
         collection_date >= fyear_end &
         shipper_date <= fyear_end) %>%

```

```

select(-X)

# Draw a PPS sample of size 'size' and estimate draw a random sample according to a
# PPS with replacement design
size <- 5
sample_prop <- S.PPS(size,unpaid_ar$sales_extended)
# returns a matrix of m rows and two columns.
# Each element of the first column indicates the unit that was selected.
# Each element of the second column indicates the selection probability of this unit

sample_ar <- unpaid_ar[sample_prop[,1],]

est <- data.frame(unpaid_ar$sales_extended)
estimate <- E.PPS(est,sample_prop[,2])

## normalize the estimate to reflect the number of transactions in the client's file
normalized_estimate <- nrow(unpaid_ar) * estimate[,2]/estimate[1,1]
normalized_estimate <- round(normalized_estimate, digits=0)

tidy(normalized_estimate) %>%
kable(caption = "PPS Hansen-Hurwitz estimate of Population Total ","latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

```

Table 2 PPS Hansen–Hurwitz estimate of population total

Names	x
Estimation	149647
Standard error	393390
CVE	11
DEFF	6212

In substantive year-end testing, we are no longer concerned with the rate of occurrence of errors, but in their absolute magnitude of error in the financial statement account balances. We want to assure that the financial statements do not contain an error equal to or of greater magnitude than our pre-determined “materiality.” We do this by auditing individual account balances and allocating materiality to individual accounts in what is called the “tolerable error” in that account.

Auditors are reductionist in their approach to materiality and year-end audits. They work up from single dollars (or whatever monetary unit prevails in a jurisdiction) to test the existence of material error in the accounts and financial statements as a whole. This is called monetary unit sampling (a specific use of probability-proportional-to-size (PPS) sampling).

Within a single sampled dollar, auditors perceive their sampled dollar as containing a proportion of error for each of the dollars making up the whole balance. In the audit literature, this proportion is colorfully referred to as “tainting”; a word which implies it is contaminated or polluted; affected with bad or undesirable quality as in “his administration was tainted by scandal.”

Stringer’s Perspective on Monetary Unit Sampling

Current auditing standards set by the American Institute of Certified Public Accountants (AICPA) do not mandate the use of statistics in conducting audit tests. However, in an age in which nearly all accounting data is contained on computer databases, often out of control of firm management (e.g., on public clouds such as Salesforce.com offers for sales; or Amazon Web Services for logistics and inventory control), the use of automated data mining and statistical analysis packages provide the only route to cost-effective auditing. Early applications of statistical analysis to audit were generally restricted to compliance tests, where transaction processing was found to either be in or out of compliance with accounting policy and procedures. Kenneth Stringer, an employee with Deloitte Haskins & Sells, began investigating audit applications of statistics with the objective of valuing accounts.

Stringer (1975) consolidated his innovations into methodology recommendations. The eponymous Stringer bound to determine confidence bounds for determining whether specific accounts were or were not “materially” in error. Stringer confidence limits were widely enough invoked to garner interest from the statistical community, which generally concluded that they tended to be much too conservative in the sense that they tended to find material error where accounts are fairly stated. Consequently, the application of the Stringer bound in practice has declined in popularity, as its use can substantially increase the cost of audits.

Perhaps the most important contribution of Kenneth Stringer to auditing was getting auditors to think of account valuations, and the errors in them, as discrete rather than continuous measures of wealth and obligations. It is natural to think in terms of the discrete occurrence of errors (an error either does or does not occur). Stringer’s emphasis on dollar-unit sampling (DUS) introduced a view that accounts are collections of individual dollars, some of which are “accurate” and some of which are “tainted” (i.e., in error and should not be in the account balance).

Stringer applies an urn model to account balances on the financial statements. Each account is an aggregation of individual transactions (think of an urn filled with balls), each with their own value. A transaction is either in error, or it is not. The “error condition” of the transaction can be modeled as a [0,1] binary variable, and these errors should be comparatively rare after we complete our midyear compliance testing.

Each transaction, in turn, can be considered to be an urn within the urn, holding n “accurate” dollars (white balls) and m “tainted” dollars (black balls). Stringer’s devoted much time to think through the relationships of distributions of each of the sets of urns and balls.

“Taintings” and the Poisson–Poisson Compound Distribution

The idea behind “taintings” in Dollar-Unit Sampling (DUS)—also called Monetary Unit Sampling (MUS)—is that each transaction is a collection of discrete monetary units (say \$1 in the USA, where any cents are rounded up to the nearest dollar). A transaction error is then allocated (divided by the total number of dollars), and a portion of that error is assigned to each \$1 in the transactions. For example, assume a Sale was recorded as \$100, but actually should have been recorded as \$10. In this case, there is an overstatement of \$90 Cr in the records (the record is \$90 Cr greater than reality). Each of the dollars in that particular Sales transaction is then considered to be “tainted” by a \$0.90 Cr = overstatement.

More generally, when an error impacts the account balance, we are concerned about the economic magnitude of that error. Accountants using the monetary-unit-sampling paradigm describe the economic magnitude of a transaction in terms of a count of dollars. This may sound convoluted, but actually makes sense when trying to construct an estimation model for the account balance in total when inferring from the sample.

The extent of auditing, size of the sample, and confidence in estimates all depend on the conclusions of interim tests. Interim tests are concerned with the rate of occurrence of errors in particular transaction cycles. The lower are these errors, the smaller will be the absolute magnitude of error in the financial statement account balances. Samples of transactions comprising a particular account balance can be relatively small if we can rely on the accounting systems, which is fundamentally what the interim tests are supposed to determine. It would be unusual if interim tests showed that internal controls were working, yet every sampled transaction had an error in it. We would usually expect most of our sample items to have zero error. In addition, the contributions of several different transaction cycle processes are likely to concentrate probabilities at multiple modes in the sampling distribution.

Let us consider the process that generates accounting errors more specifically. There are two distinct components—(1) the occurrence of an error in the transaction and (2) the economic magnitude of that error. Consider the error occurrence first. Interim tests are solely concerned with error rates, and thus at an individual transaction level whether or not an error has occurred or not. Errors should be rare—i.e., most transactions should have zero error, and there will be a spike in the probability distribution at zero that has most of the weight of the distribution. Errors may be compliance errors with no financial impact or may involve some misstatement of an accounting entry. The former will, during substantive testing of the account balances, be treated as a zero error, since we are only concerned with dollar errors.

If one has considerable information about the shape and scale of the distribution of error values, a distribution can be chosen with more than just one parameter. A good choice is the negative binomial distribution, which is a discrete probability distribution of k “error \$1”s in a sequence of Bernoulli trials with error probability p before a pre-specified number r of “actual \$1”s occurs.

A more tractable continuous counterpart for negative binomial random variables is the Gamma distribution with corresponding shape-rate parameterization. Kotz et al. (2004) method of moments’ scheme for estimators can be used to fit data to the Gamma distribution. The gamma distribution has been used to model the size of insurance claims, and can be credibly argued to provide good fits for the magnitude of errors in individual accounting transactions. The gamma distribution

is also used to model errors in multi-level Poisson regression models, because the combination of the Poisson distribution and a gamma distribution is a negative binomial distribution. In this case, the total distribution of taintings is given by a compound Poisson–Negative Binomial distribution. The class of mixed compound Poisson–gamma distributions which have positive mass at zero, but are otherwise continuous, are called Tweedie distributions. Tweedie distributions have been studied intensively over the past two decades and are particularly relevant to economic risk modeling. The Tweedie distribution has been applied to auditing by Westland (2017), where it has been shown to extract significantly more information than benchmark approaches.

The Audit of Physical Inventory

A physical inventory is the process of manually counting all of the items in your stock. Most distributors conduct a physical inventory once a year. It is an expensive process that few employees enjoy. But, accurate balance information is necessary for both efficient uses of sales resources and effective inventory management.

Periodic Inventory Systems

Under the periodic system, a company takes the physical inventory periodically and uses the resulting figure to adjust the balance sheet inventory asset account. Retail shops that use periodic inventory usually take inventory at their particular year-end. However, a business could take inventory more often, such as quarterly or at the end of every heavy sales season (like Valentine's Day, Mother's Day, and the December holidays).

Next, the company's accounting department subtracts ending inventory totals from the beginning inventory after adding in all inventory purchases made during the period. The resulting number is the cost of goods sold (COGS). The balance sheet inventory account is reduced, and the income statement expense account COGS is increased by that number to match revenue with expenses.

Perpetual Inventory Systems

The other inventory system used is the perpetual system. With this system, the inventory count is updated constantly, perpetually, by the electronic cash registers. If you have ever used the self-checkout, you have used one. The checkout features a glass window with a red beam of light. You run the bar code of a product over the red beam, and the price (updated for sales if necessary) is automatically recorded as a sale for which you are charged, and the business receives revenue.

This system takes the cost of the sold item out of the asset inventory account and moves it to the cost of goods sold. With point-of-sale inventory, cash register transactions update all purchases, inventory, COGS, and sales information throughout the system in real-time as the transactions occur.

Most large retail stores have point-of-sale inventory systems with item restock and reorder level alerts. The restock level alert advises purchasing employees when sales cause the number of an item in inventory to drop below the company's minimum quantity requirement. If the company also programs the reorder level (the maximum quantity of each item it wants to maintain on hand), the software can tell purchasing exactly how many of the items should be ordered.

Even if a client uses a point-of-sale system, taking a physical inventory at year-end is still important to verify that the perpetual system is working correctly. Taking a physical inventory is also the best way to identify breakage and employee theft issues.

Counting Inventory When Preparing Financial Statements

Inventory is a balance sheet asset account that needs to be adjusted for financial statements at the end of an accounting period. During the accounting period, your company buys inventory and records those purchases in a Purchases account without indicating any change to inventory.

When the products are sold, you record the sales in the Sales account but do not make any adjustments to the value of the inventory. Instead, you adjust the inventory value at the end of the accounting period because adjusting with each purchase and sale would be too time-consuming.

The steps for making proper adjustments to inventory in your books are as follows:

1. Determine the inventory remaining: In addition to calculating ending inventory using the purchases and sales numbers in the books, you should also do a physical count of inventory to be sure that what is on the shelves matches what is in the books.
2. Set a value for that inventory: The value of ending inventory varies depending on the method your company has chosen to use for valuing inventory.
3. Adjust the number of pieces remaining in inventory in the Inventory Account and adjust the value of that account based on the information collected in 1 and 2.

If you track inventory using your computerized accounting system, the system makes adjustments to inventory as you record sales. At the end of the accounting period, the value of your company's ending inventory should be adjusted in the books already.

Inventory Systems

Depending on the method your audit client uses for value ending inventory, the amount transferred from the balance sheet inventory account to the income statement cost of goods sold can vary wildly. Your first question for your client in this stage of the audit is how it values ending inventory. You need this information when recalculating your client's inventory valuation.

The three common accounting methods for inventory are used; depending on which you use, the same inventory items can result in different ending dollar amounts.

1. First-in, first-out (FIFO): Using the FIFO method, your client assumes that the oldest items in its inventory are the ones first sold. Consider buying milk in a grocery store. The cartons or bottles with the most current expiration date are pushed ahead of the cartons that have more time before they go bad. The oldest cartons of milk may not always be the first ones sold, but a client using the FIFO method bases its numbers on the oldest items being sold first.
2. Last-in, first-out (LIFO): Under this method, the client assumes that its newest items (the ones most recently purchased) are the first ones sold. Imagine a big jar of nails in a hardware store. As the jar empties, more nails are added on top of the old ones instead of redistributing the old nails to move to the top of the jar. Therefore, the newest nails are consistently sold to customers rather than the older ones. When a company uses the LIFO method, it may have to include a LIFO reserve amount in its notes to the financial statements. This reserve amount is the inventory cost difference between using FIFO and LIFO.
3. Weighted average: When a client uses this method, inventory and the cost of goods sold are based on the average cost of all units purchased during the period. This method is generally used when inventory is substantially the same, such as grains and fuel.

When a business uses or sells inventory, the cost moves from the inventory asset account to the income statement cost of goods sold expense account using the particular method the company has selected for its business.

Physical Inventory (Counting) Process

Companies generally apply two types of inventory counting procedures:

- Physical inventory (or full physical count or stock-taking)
- Cycle counts (or cycle counting)

A full physical count is usually performed once a year, closer to the end of the year. A physical inventory is needed to provide accurate accounting data and identify any differences between what is currently in the warehouse and what is reflected in the accounting system. If differences exist, then adjustments should be made to get the accounting counts to match the warehouse counts. A full physical inventory is usually performed when all inventory movements are stopped to ensure better accuracy.

Cycle counts represent a procedure to validate inventory quantity in the accounting system through regular partial counts. Cycle count frequency is determined so that every item is counted at least once a year, and a lot of companies perform counts to have all inventory items counted more than once during a year. Cycle counts may be performed on a daily, weekly, etc. basis depending on business needs.

Several methods of selecting inventory for cycle counting exist. One of the methods is to count inventory based on a geographical factor. For example, starting from one location of a warehouse and proceeding systematically to the next location during the year so that by the end of the year, the entire warehouse is counted at least once. If a company policy is to count all inventory multiple times a year, then this full cycle of counting a warehouse should take place that number of times, and the number of items counted each time should be adjusted accordingly.

Another method to select counts for cycle counting is to emphasize on items with more movement than on items with less movement. This is logical because items that move faster and have higher volume will have more chances of misplacements, etc. resulting in differences between what is in the warehouse and what is in the accounting system. Under this method, an inventory item is assigned to a particular category. Such categories are labeled with letters like A, B, and C (thus, the method is sometimes called the ABC method). For example, category A will include items that turn at least 5 times a year, category B items will include items that turn between 2 and 5 times, and category C items will include items that turn fewer than 2 times a year. Then, the company may decide that any items in category A should be counted 12 times a year, in category B—4 times a year, and in category C—once a year. Based on this information, the company would select items to count based on the established criteria. An alternative to assigning inventory items to categories would be a combination of how fast items turn and how much they cost.

Physical Inventory Count Versus Cycle Counts

There are some advantages that favor performing cycle counts rather than full physical counts. Cycle count advantages:

1. Ability to identify and correct inventory recording errors more frequently (e.g., daily)
2. Higher inventory quantity accuracy in the accounting system
3. Ability to cycle count without interruptions to normal operations
4. Fewer employees needed for cycle counting
5. Ability to eliminate annual physical inventory

Physical inventory disadvantages:

1. Difficult to have accurate inventory quantities in the accounting system throughout the year
2. Inventory counters may not be familiar with inventory, which may result in recording and counting errors
3. Temporary help is usually required to perform a physical inventory
4. Overtime costs may be incurred in order to fulfill a physical inventory
5. Operations are usually stopped during a physical inventory

Inventory Audit Procedures

Inventory is audited by obtaining external transaction evidence as well as internal observational evidence—from inspecting the physical locations of the inventory, about the quantity, condition, obsolescence, pricing, and acquisition cost compared to their recorded value.

In the following examples, the auditor is provided files that represent transactions in parts of the inventory system, but which in some cases do not contain key fields that are required to recreate all of the system's activity. In these cases, I have provided code that “guesses” at the correct key fields in order to recreate “master” files of inventory sales and procurement. The “guesses” rely on procedures in *dplyr* (a part of the *tidyverse* packages). The resulting master files will contain all of the information that will be required to perform the substantive part of the inventory audit.

Inventory audits are an essential component of most annual audits for manufacturing, extractive, and wholesale/retail firms. Given the significant size of some inventories, auditors may engage in a large and varied set of inventory audit procedures to ensure fairness of inventory valuation in the financial statements. The following are specific inventory audit procedures that are commonly applied in annual audits.

1. Cutoff analysis. The auditors will examine your procedures for halting any further receiving into the warehouse or shipments from it at the time of the physical inventory count, so that the extraneous inventory items are excluded. They typically test the last few receiving and shipping transactions prior to the physical count, as well as transactions immediately following it, to see if you are properly accounting for them.
2. Observe the physical inventory count. The auditors want to be comfortable with the procedures you use to count the inventory. This means that they will discuss the counting procedure with you, observe counts as they are being done, test count some of the inventory themselves and trace their counts to the amounts recorded by the company's counters, and verify that all inventory count tags were accounted for. If you have multiple inventory storage locations, they may test the inventory in those locations where there are significant amounts of inventory. They may also ask for confirmation of inventory from the custodian of any public warehouse where the company is storing inventory.
3. Reconcile the inventory count to the general ledger. They will trace the valuation compiled from the physical inventory count to the company's general ledger, to verify that the counted balance was carried forward into the company's accounting records.
4. Test high-value items. If there are items in the inventory that are of unusually high value, the auditors will likely spend extra time counting them in inventory, ensuring that they are valued correctly, and tracing them into the valuation report that carries forward into the inventory balance in the general ledger.
5. Test error-prone items. If the auditors have noticed an error trend in prior years for specific inventory items, they will be more likely to test these items again.
6. Test inventory in transit. There is a risk that you have inventory in transit from one storage location to another at the time of the physical count. Auditors test for this by reviewing your transfer documentation.
7. Test item costs. The auditors need to know where purchased costs in your accounting records come from, so they will compare the amounts in recent supplier invoices to the costs listed in your inventory valuation.
8. Review freight costs. You can either include freight costs in inventory or charge it to expense in the period incurred, but you need to be consistent in your treatment—so the auditors will trace a selection of freight invoices through your accounting system to see how they are handled.
9. Test for lower of cost or market. The auditors must follow the lower of cost or market rule, and will do so by comparing a selection of market prices to their recorded costs.
10. Finished goods cost analysis. If a significant proportion of the inventory valuation is comprised of finished goods, then the auditors will want to review the bill of materials for a selection of finished goods items, and test them to see if they show an accurate compilation of the components in the finished goods items, as well as correct costs.
11. Direct labor analysis. If direct labor is included in the cost of inventory, then the auditors will want to trace the labor charged during production on time cards or labor routings to the cost of the inventory. They will also investigate whether the labor costs listed in the valuation are supported by payroll records.
12. Overhead analysis. If you apply overhead costs to the inventory valuation, then the auditors will verify that you are consistently using the same general ledger accounts as the source for your overhead costs, whether overhead includes any abnormal costs (which should be charged to expense as incurred), and test the validity and consistency of the method used to apply overhead costs to inventory.
13. Work-in-process testing. If you have a significant amount of work-in-process (WIP) inventory, the auditors will test how you determine the percentage of completion for WIP items.
14. Inventory allowances. The auditors will determine whether the amounts you have recorded as allowances for obsolete inventory or scrap are adequate, based on your procedures for doing so, historical patterns, "where used" reports, and reports of inventory usage (as well as by physical observation during the physical count). If you do not have such allowances, they may require you to create them.
15. Inventory ownership. The auditors will review purchase records to ensure that the inventory in your warehouse is actually owned by the company (as opposed to customer-owned inventory or inventory on consignment from suppliers).
16. Inventory layers. If you are using a FIFO or LIFO inventory valuation system, the auditors will test the inventory layers that you have recorded to verify that they are valid.

Computer Analytic Workpaper Support

The first step in setting up work papers is the access to the client files. Here I bring in files germane to the inventory audit.

```
rm(list=ls())
library(readr)
library(tidyverse)
library(broom)
library(stringr)
library(data.table)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

## Set the directory for the new files (modify the path as needed)
default_dir <- "/home/westland/audit_analytics_book/audit_simulated_files/"

if (file.exists(default_dir)){
  setwd(default_dir)
} else {
  dir.create(default_dir)
  setwd(default_dir)
}

sales_journal <-
  read_csv(system.file("extdata", "sales_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

purchase_journal <-
  read_csv(system.file("extdata", "purchase_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

perpetual_inventory_ledger <-
  read_csv(system.file("extdata", "perpetual_inventory_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

fyear_end_ar_ledger <-
  read_csv(system.file("extdata", "fyear_end_ar_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

fyear_begin_inventory_ledger <-
  read_csv(system.file("extdata", "fyear_begin_inventory_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

expenditures <-
  read_csv(system.file("extdata", "expenditures.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

disbursement_journal <-
  read_csv(system.file("extdata", "disbursement_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))
```

```

deposit_daily <-
  read_csv(system.file("extdata", "deposit_daily.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

daily_ar_balance <-
  read_csv(system.file("extdata", "daily_ar_balance.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

collections_journal <-
  read_csv(system.file("extdata", "collections_journal.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

ap_ledger <-
  read_csv(system.file("extdata", "ap_ledger.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip()))

real_world_ye_inventory <-
  perpetual_inventory_ledger %>%
  group_by(sku) %>%
  inner_join(sales_journal, by="sku") %>%
  slice(n()) %>%
  mutate(inv_extended = unit_cost * stock_on_hand)

```

Footing, Reconcile the Inventory Count to the General Ledger

Trace the valuation compiled from the physical inventory count to the company's general ledger to verify that the counted balance was carried forward into the company's accounting records.

```

library(tidyverse)

## get the counts by SKU

foot_count <- perpetual_inventory_ledger %>%
  group_by(sku) %>%
  slice(n()) %>%
  select(stock_on_hand) %>%
  as.data.frame()

## get the unit cost by SKU

foot_sku <- purchase_journal %>%
  group_by(sku) %>%
  slice(n()) %>%
  select(unit_cost) %>%
  as.data.frame()

## extend

foot_value <- perpetual_inventory_ledger %>%
  group_by(sku) %>%

```

```

inner_join(sales_journal, by="sku") %>%
slice(n()) %>%
mutate(inv_value = stock_on_hand * unit_cost) %>%
ungroup() %>%
select(inv_value) %>%
sum()

cat("Value of inventory (per client books) at year end = $",
format(foot_value, big.mark=","))
## Value of inventory (per client books) at year end = $ 31,281

## compare this to the (unobservable) true value of year end inventory

true_foot_value <-
real_world_ye_inventory %>%
group_by(sku) %>%
inner_join(sales_journal, by="sku") %>%
slice(n()) %>%
mutate(inv_value = stock_on_hand * unit_cost.x) %>%
ungroup() %>%
select(inv_value) %>%
sum()

cat("\n Compare to the (unobservable)
true value of inventory at year end = $",
format(true_foot_value, big.mark=","))
## Compare to the (unobservable)
##      true value of inventory at year end = $ 31,281

```

Cutoff Analysis

The auditors will examine your procedures for halting any further receiving into the warehouse or shipments from it at the time of the physical inventory count (assuming they have a single year-end count), so that in-transit inventory is excluded. They typically test the last few receiving and shipping transactions prior to the physical count, as well as transactions immediately following it, to see if you are properly accounting for them.

```

library(tidyverse)
library(lubridate)
library(kableExtra)

fyear_begin <- paste0(format(Sys.Date(), '%Y'), '-01-01') # date can be hard-coded if needed
fyear_end <- paste0(format(Sys.Date(), '%Y'), '-12-31')

sales_journal$invoice_date <- as_date(sales_journal$invoice_date)
sales_journal$shipper_date <- as_date(sales_journal$shipper_date)

sales_cutoff_errors <- sales_journal %>%
filter(invoice_date <= fyear_end & shipper_date > fyear_end) %>%
select(customer_no, invoice_no, invoice_date, shipper_no, shipper_date, sales_extended)

```

```
# print the inventory receipts cutoff worksheet

perpetual_inventory_ledger %>%
  group_by(sku) %>%
  slice(n()) %>%
  left_join(ap_ledger, by="sku") %>%
  filter(receiver_date > fyyear_end) %>%
  kable(
    caption = "Inventory Receipt Cutoff Worksheet  
(verify that none of these receipts were included in Y/E inventory)",
    "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped",
  latex_options="scale_down")
```

Duplicates and Omissions

Companies index each high volume class of journal entry (e.g., sales, receipts, etc.) with an index number. Before the widespread use of computer entry for transactions, these were actually preprinted at the top of a multipart document, and auditors were expected to assure that preprinted transaction entry documents were being used, and the sequence of index numbers was fully accounted for. Omissions or duplicates of an index number might indicate an error in processing that would over or underestimate the account, causing other internal control problems. The following code finds duplicates and omissions (gaps in the indexing number on the journal entry transaction document)

```
library(tidyverse)
library(stringr)
library(dplyr)

# duplicated records
dup_purchase <- purchase_journal[duplicated(purchase_journal$po_no), ]
n <- nrow(dup_purchase)
cat("\n # of duplicate purchases = ", n)

dup_sales <- sales_journal[duplicated(sales_journal$invoice_no), ]
n <- nrow(dup_sales)
cat("\n # of duplicate sales = ", n)

receiver_journal <- perpetual_inventory_ledger %>% group_by(sku) %>% slice(n()) %>%
  left_join(ap_ledger, by = "sku")

dup_receiver <- receiver_journal[duplicated(receiver_journal$receiver_no), ]
n <- nrow(dup_receiver)
cat("\n # of duplicate receivers = ", n)

dup_shipment <- sales_journal[duplicated(sales_journal$shipper_no), ]
n <- nrow(dup_shipment)
cat("\n # of duplicate shipments = ", n)

## omissions

po <- as.numeric(substring(purchase_journal$po_no, 2))
po_min <- as.numeric(min(po))
po_max <- as.numeric(max(po))
```

Table 3 Inventory receipt cutoff worksheet (verify that none of these receipts were included in Y/E inventory)

sku	Date	stock_on_hand	ap_no	ap_date	no_units_ordered	extended_cost	receiver_no	receiver_date
BFQCM	2020-12-26	121	ap00138	2021-01-02	100	3400	rec00138	2021-01-01
CRBOW	2020-12-31	48	ap00147	2021-01-12	100	3300	rec00147	2021-01-11
CRBOW	2020-12-31	48	ap00150	2021-01-15	100	3400	rec00150	2021-01-14
IQFYO	2020-12-24	72	ap00143	2021-01-05	100	3500	rec00143	2021-01-04
ISIBO	2020-12-30	90	ap00145	2021-03-07	100	3000	rec00145	2021-03-06
OYJVB	2020-12-27	84	ap00142	2021-03-29	100	3600	rec00142	2021-03-28
QHBOX	2020-12-30	114	ap00139	2021-01-17	100	3300	rec00139	2021-01-16
QHVMN	2020-12-30	61	ap00144	2021-01-20	100	2700	rec00144	2021-01-19
YVQGO	2020-12-31	133	ap00140	2021-01-04	100	3600	rec00140	2021-01-03
ZAUZT	2020-12-31	91	ap00148	2021-01-25	100	3600	rec00148	2021-01-24

```

omit <- as.data.frame(setdiff(po_min:po_max, po))
n <- nrow(omit)
cat("\n # of omitted purchase records = ", n)

invoice <- as.numeric(substring(sales_journal$invoice_no, 2))
invoice_min <- as.numeric(min(invoice))
invoice_max <- as.numeric(max(invoice))

omit <- as.data.frame(setdiff(invoice_min:invoice_max, invoice))
n <- nrow(omit)
cat("\n # of omitted sales records = ", n)

receiver <- as.numeric(substring(receiver_journal$receiver_no, 4))
receiver_min <- as.numeric(min(receiver))
receiver_max <- as.numeric(max(receiver))

omit <- as.data.frame(setdiff(receiver_min:receiver_max, receiver))
n <- nrow(omit)
cat("\n # of omitted receiver records = ", n)

shipments <- as.numeric(substring(sales_journal$shipper_no, 2))
shipments_min <- as.numeric(min(shipments))
shipments_max <- as.numeric(max(shipments))

omit <- as.data.frame(setdiff(shipments_min:shipments_max, shipments))
n <- nrow(omit)
cat("\n # of omitted sales records = ", n)

## 
## # of duplicate purchases = 7
## # of duplicate sales = 44
## # of duplicate receivers = 0
## # of duplicate shipments = 44
## # of omitted purchase records = 6
## # of omitted sales records = 44
## # of omitted receiver records = 0
## # of omitted sales records = 44

```

Physical Count Exceptions to Perpetual Inventory Ledger

The perpetual inventory ledger maintains a running balance of inventory at cost and is updated from sales and purchase transaction documents. The actual balance of inventory will vary from the perpetual inventory ledger in both item count and value for several reasons, e.g.,

1. Obsolescence or other cause of a decline in market value
2. Damage to items in transit or storage
3. Misclassification or misplacement, often a problem with high volume retailers like Walmart
4. Markdown of value because a returned item was refurbished, or delivered with an open box
5. Quality control issues that reduce finished goods yield or require markdown of value.

The auditors want to be comfortable with the procedures you use to count the inventory. This means that they will discuss the counting procedure with you, observe counts as they are being done, test count some of the inventory themselves and trace their counts to the amounts recorded by the company's counters, and verify that all inventory count tags were accounted for. If you have multiple inventory storage locations, they may test the inventory in those locations where there are significant

amounts of inventory. They may also ask for confirmation of inventory from the custodian of any public warehouse where the company is storing inventory. The physical count of inventory is not just a count but offers employees and auditors the opportunity to inspect the condition and marketability of inventory. Items, where value has been significantly marked down, will be subject to Lower of Cost or Market (LOCOM) revaluation discussed in the next subsection.

```
library(kableExtra)
library(tidyverse)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

real_world_ye_inventory <-
  read_csv(system.file("extdata", "real_world_ye_inventory.csv", package = "auditAnalytics",
  mustWork = TRUE),
  col_types = cols(X1 = col_skip()))
real_world_ye_inventory[is.na(real_world_ye_inventory)] <- 0

inventory_count_differences <-
  perpetual_inventory_ledger %>%
  group_by(sku) %>%
  left_join(real_world_ye_inventory, by="sku") %>%
  slice(n()) %>% ## the final slice, by SKU, will be what is in-stock at year end
  filter(exception != "No exception, count is accurate") %>%
  mutate(err_perpetual = stock_on_hand - ye_stock_on_hand) %>%
  select(sku, err_perpetual, unit_cost, count_exception, exception, actual_unit_market) %>%
  as.data.frame()

kable(inventory_count_differences,
  caption = "Perpetual vs. Actual Overstatement
  (negative implies understatement of perpetual inventory)",
  "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")
```

Table 4 Perpetual vs. actual overstatement (negative implies understatement of perpetual inventory)

sku	err_perpetual	unit_cost	count_exception	exception	actual_unit_market
BFQCM	21	34	1	Obsolete_markdown	18.534077
QHBOX	14	34	1	Damaged	1.708307

Test for Lower of Cost or Market (LOCOM)

Inventory is typically valued at acquisition cost. The physical count of inventory is not just a count but offers employees and auditors the opportunity to inspect the condition and marketability of inventory. Items, where value has been significantly marked down, will be subject to Lower of Cost or Market (LOCOM) revaluation discussed previously. Auditors are required to apply the lower of cost or market rule to revalue inventory to its net realizable value and they will do so by comparing a selection of market prices to their recorded costs.

```
library(tidyverse)

real_world_ye_inventory[is.na(real_world_ye_inventory)] <- 0

inventory_count_differences <- perpetual_inventory_ledger %>%
  group_by(sku) %>%
  slice(n()) %>% ## the final slice, by SKU, will be what is in-stock at year end
  left_join(real_world_ye_inventory, by="sku") %>%
```

```

filter(exception != "No exception, count is accurate") %>%
mutate(err_perpetual = stock_on_hand - ye_stock_on_hand) %>%
select(sku,
       stock_on_hand,
       ye_stock_on_hand,
       unit_cost,
       actual_unit_market,
       err_perpetual,
       count_exception,
       exception) %>%
mutate(ye_cost = stock_on_hand * unit_cost,
       ye_market = ye_stock_on_hand * actual_unit_market,
       inv_markdown = ye_cost - ye_market) %>%
select(sku, ye_cost, ye_market, inv_markdown) %>%
as.data.frame()

```

total_inv_markdown <- sum(inventory_count_differences\$inv_markdown)

```

cat("\n Total Amount to Markdown
Client's Trial Balance Inventory Amount
(by LOCOM rule) = ",
total_inv_markdown)

```

```

## Total Amount to Markdown
## Client's Trial Balance Inventory Amount
## (by LOCOM rule) = 5965.762

```

```

library(knitr)
library(kableExtra)
kable(inventory_count_differences, caption = "Market and Cost of Inventory by SKU", "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped") %>%
  footnote(general = "",
            symbol = c(round(total_inv_markdown, 2)),
            general_title = "Total Markdown for LOCOM rule",
            title_format = c("italic", "underline"))
)

```

Table 5 Market and cost of inventory by SKU

sku	ye_cost	ye_market	inv_markdown
BFQCM	4114	1853.4077	2260.592
QHBOX	3876	170.8307	3705.169
<i>Total Markdown for LOCOM rule</i>			
* 5965.76			

Audit a Subset of High-Value Items

If there are items in the inventory that are of unusually high value, the auditors will likely spend extra time counting them in inventory, ensuring that they are valued correctly, and tracing them into the valuation report that carries forward into the inventory balance in the general ledger.

```

library(tidyverse)

high_value <- perpetual_inventory_ledger %>%
  left_join(purchase_journal, by="sku") %>%
  select(sku, stock_on_hand, unit_cost) %>%
  mutate(value = stock_on_hand * unit_cost) %>%
  arrange(desc(value)) %>%
  slice(1:5) %>% ## choose highest 5 in value
  as.data.frame()

library(knitr)
library(kableExtra)
kable(high_value, caption = "Top 5 inventory SKU by value", "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

```

Table 6 Top 5 inventory SKU by value

sku	stock_on_hand	unit_cost	value
ZAUZT	140	42	5880
ZAUZT	140	42	5880
ZAUZT	140	42	5880
BFQCM	136	42	5712
CRBOW	136	42	5712

Inventory Allowances

The auditors will determine whether the amounts you have recorded as allowances for obsolete inventory or scrap are adequate, based on your procedures for doing so, historical patterns, “where used” reports, and reports of inventory usage (as well as by physical observation during the physical count). If you do not have such allowances, they may require you to create them.

Obsolete inventory is not just that inventory that is identified as obsolete in the physical count, but also inventory on hand that is not expected to sell in some established period into the future. Management is often keen to control the holding cost (cost of warehouse space and management) of inventory, as well as the technological obsolescence (newer products arrive that are lower cost or better performing substitutes for the inventory). Allowances for obsolete inventory are often computed by predicting sales, and determining if there is excess inventory. Below is a code chunk to perform this calculation.

```

library(tidyverse)

## assume that the client maintains an inventory balance sufficient for 1 year of sales,
## and any excess balance is considered obsolete inventory that should be marked down
## to disposal value (which is determined to be 50% of original unit cost)

sales_of_sku_per_year <- sales_journal %>% ## sales_journal is only for this fiscal year, so this rate
                                         # is just the sum of sales in the sales_journal
  group_by(sku) %>%
  summarise(sales_rate = sum(sales_count, na.rm = T))

unit_costs <- purchase_journal %>%
  group_by(sku) %>%
  slice(n()) %>%
  select(sku, unit_cost) %>%
  as.data.frame()

allowance_for_obsolete_inv <- perpetual_inventory_ledger %>%
  group_by(sku) %>%
  slice(n()) %>% ## the final slice, by SKU, will be what is in-stock at year end
  left_join(sales_of_sku_per_year, by="sku") %>%
  mutate(years_on_hand = stock_on_hand / sales_rate) %>%
  left_join(sales_of_sku_per_year, by="sku") %>%

```

```

left_join(unit_costs, by="sku") %>%
  select(sku, years_on_hand, stock_on_hand, sales_rate = sales_rate.x, unit_cost) %>%
  filter(years_on_hand > 1) %>% ## more than 1 year stock on hand
  mutate(allowance = (years_on_hand - 1) * stock_on_hand * unit_cost) %>%
  select(sku, years_on_hand, stock_on_hand, sales_rate, unit_cost, allowance) %>%
  as.data.frame()

if(nrow(allowance_for_obsolete_inv) == 0){cat("\n\n No Obsolete Inventory for any SKU \n\n")}

## 
## 
## No Obsolete Inventory for any SKU

library(knitr)
library(kableExtra)
kable(allowance_for_obsolete_inv, caption = "Allowance for Obsolete Inventory by SKU",
      "latex", booktabs = T) %>%
  kable_styling(bootstrap_options = "striped")

```

Table 7 Allowance for obsolete inventory by SKU

sku	years_on_hand	stock_on_hand	sales_rate	unit_cost	Allowance
-----	---------------	---------------	------------	-----------	-----------

Other Inventory Tests

The test data generated by code in Chap. 13 is designed to test internal control over the processing of transactions through the client's systems. The setting is simple, with inventory warehousing and retailing, and no inflation. In practice, auditors encounter other challenges, in particular, where the client is manufacturing or refining complex products at multiple locations. The sort of audit working papers generated for these other inventory audit tasks will follow the same format as the above code chunks, using tidyverse commands to manipulate data into work papers that facilitate the straightforward completion of the audit tasks.

Both among companies and within companies, auditors will find huge variances in inventory type, value, perishability, and many other attributes. Each client's inventory will be different; there are larger differences in accounting for inventory than in any other financial account, as inventory fundamentally defines the business of the client. Thus the auditor needs to prepare to address particulars of a client's business, and each audit will be different. A list of common inventory audit tasks that the auditor might encounter are:

1. Test inventory in transit: There is a risk that you have inventory in transit from one storage location to another at the time of the physical count. Auditors test for this by reviewing your transfer documentation.
2. Test error-prone items: If the auditors have noticed an error trend in prior years for specific inventory items, they will be more likely to test them again.
3. Review freight costs: You can either include freight costs in inventory or charge it to expense in the period incurred, but you need to be consistent in your treatment—so the auditors will trace a selection of freight invoices through your accounting system to see how they are handled.
4. Finished goods cost analysis: If a significant proportion of the inventory valuation is comprised of finished goods, then the auditors will want to review the bill of materials for a selection of finished goods items, and test them to see if they show an accurate compilation of the components in the finished goods items, as well as correct costs.
5. Direct labor analysis: If direct labor is included in the cost of inventory, then the auditors will want to trace the labor charged during production on time cards or labor routings to the cost of the inventory. They will also investigate whether the labor costs listed in the valuation are supported by payroll records.
6. Overhead analysis: If you apply overhead costs to the inventory valuation, then the auditors will verify that you are consistently using the same general ledger accounts as the source for your overhead costs, whether overhead includes any abnormal costs (which should be charged to expense as incurred), and test the validity and consistency of the method used to apply overhead costs to inventory.

7. Work-in-process testing: If you have a significant amount of work-in-process (WIP) inventory, the auditors will test how you determine the percentage of completion for WIP items.
8. Test item costs: The auditors need to know where purchased costs in your accounting records come from, so they will compare the amounts in recent supplier invoices to the costs listed in your inventory valuation.
9. Inventory ownership: The auditors will review purchase records to ensure that the inventory in your warehouse is actually owned by the company (as opposed to customer-owned inventory or inventory on consignment from suppliers).
10. Inventory layers: If you are using a FIFO or LIFO inventory valuation system, the auditors will test the inventory layers that you have recorded to verify that they are valid.
11. Methods of Verifying Counts: If the company uses cycle counts instead of a physical count, the auditors can still use the procedures related to a physical count. They simply do so during one or more cycle counts and can do so at any time; there is no need to only observe a cycle count that occurs at the end of the reporting period. Their tests may also evaluate the frequency of cycle counts, as well as the quality of the investigations conducted by counters into any variances found. The extent of the procedures employed will decline if inventory constitutes a relatively small proportion of the assets listed on a company's balance sheet.

References

- Westland, J. Christopher. 2017. An Empirical Investigation of Analytical Procedures Using Mixture Distributions. *Intelligent Systems in Accounting, Finance and Management* 24 (4): 111–24. Wiley Online Library.
- Cohen, Jacob. 1992. A Power Primer. *Psychological Bulletin* 112 (1): 155. American Psychological Association.
- Horvitz, Daniel G, and Donovan J Thompson. 1952. A Generalization of Sampling Without Replacement from a Finite Universe. *Journal of the American Statistical Association* 47 (260): 663–85. Taylor & Francis Group.
- Kotz, Samuel, Narayanaswamy Balakrishnan, and Norman L Johnson. 2004. *Continuous Multivariate Distributions, Models and Applications*, Vol. 1. New York: John Wiley & Sons.
- Stringer, Kenneth W. 1961. Some Basic Concepts of Statistical Sampling in Auditing. *Journal of Accountancy (Pre-1986)* 112 (000005): 63. American Institute of Certified Public Accountants.
- . 1963. Practical Aspects of Statistical Sampling in Auditing. In *Proceedings of the Business and Economic Statistics Section*, 405–11. Washington, DC.: American Statistical Association.
- . 1975. A Statistical Technique for Analytical Review. *Journal of Accounting Research* 13: 1–9. JSTOR.

Sarbanes–Oxley Engagements



The Sarbanes–Oxley Act: Security, Privacy, and Fraud Threats to Firm Systems

Following the Enron and WorldCom collapses that marked the end of the dot-com bubble, U.S. legislators sought to better protect and inform investors through the passage of the Sarbanes–Oxley Act of 2002 (SOX). Section 404 of SOX requires companies to review their internal controls under the supervision of external auditors and declare whether their controls are effective. Section 404a prescribes the scope and rules for internal control assessments, and section 404b prescribes the reporting requirements. Section of SOX requires companies to self-report on effectiveness of internal controls and otherwise refers to the relevant sections of the Securities Exchange Act of 1934 for specifics. One significant reason for the focus on internal control is the potential for firms to suffer systems intrusion from external actors, commonly referred to as security breaches. These may lead to materially significant losses and misstatements which could result in financial results that are not fairly presented. Rice and Weber (2012); Rice et al. (2014), Ashbaugh-Skaife et al. (2008), Ge et al. (2016), Bedard and Graham (2011), Bedard et al. (2009), and Hoitash et al. (2009) have provided compelling evidence that firms with weak internal controls suffer increased numbers of privacy and security breaches.

Recent information systems breaches have grown costlier and more frequent; for example, Home Depot's 2015 breach has cost it \$232 million so far, an amount that they expect to reach billions. A 2015 breach of Ashley Madison stole 40 million accounts, including photos, details of sexual proclivities, and personal addresses. Target's 2013 breach affected the accounts of 70 million customers and so far has cost the firm \$162 million in added expense. In 2014 A Guardians of Peace breach of Sony Pictures stole over 100 terabytes of confidential data. 2014 also saw the theft of 360 million MySpace accounts, a LinkedIn hack that took more than 100 million accounts, a 500-million-account hack of Yahoo, 340 million AdultFriendFinder accounts, their second hack in a year, and numerous other breaches. Both frequency and scale of breaches have grown dramatically in the recent past.

Auditors have argued forcefully that they have no explicit responsibility for detecting fraud and external threats during audits. Nonetheless, audits were forced to embrace some responsibility for detecting fraud and threats after the Enron and WorldCom collapses. AICPA 2002 provides specific guidelines with respect to the auditor's responsibility for identifying external threats and fraud that may result in material misstatements. Security breaches and other external threats to the firm are subsumed under the category of fraud, which AU 316.05 clarified is:

...a broad legal concept and auditors do not make legal determinations of whether fraud has occurred. Rather the auditors interest specifically relates to acts that result in a material misstatement of the financial statements. The primary factor that distinguishes fraud from error is whether the underlying action that results in the misstatement of the financial statements is intentional or unintentional. Fraud is an intentional act that results in a material misstatement in financial statements that are the subject of an audit.

SOX sections 302 and 404, which are of most concern to auditors are implemented under the guidance of the Public Company Accounting Oversight Board (PCAOB), the regulator of public company auditors. When SOX 404 was put in place, the PCAOB issued Auditing Standard (AS) No. 2 to implement it. One regulation that AS 2 established was that the external auditor did attest to and report on the assessment of internal control made by management. When AS 5 (now AS 2201) was issued in 2007, it relaxed the requirements of AS 2, without substantially altering them.

Both AS 2 and AS 5 provide guidance to ensure that management's opinion and the auditor's opinion are based on the same evidence with the same standards and principles applied to derive those opinions. It is somewhat technical in indicating whether evidential procedures, decisions on materiality, etc., and evidence from control tests should defer to the standards in place at the client (management's) or the audit firm. These are details that would be needed for an unequivocal audit AS#5

para 1,2,3, and 5 provide a synopsis of the objectives and motivation for AS 5 which provides guidance for implementation of SOX audits.

1. This standard establishes requirements and provides direction that applies when an auditor is engaged to perform an audit of management's assessment of the effectiveness of internal control over financial reporting that is integrated with an audit of the financial statements.
2. Effective internal control over financial reporting provides reasonable assurance regarding the reliability of financial reporting and the preparation of financial statements for external purposes. If one or more material weaknesses exist, the company's internal control over financial reporting cannot be considered effective.
3. The auditor's objective in an audit of internal control over financial reporting is to express an opinion on the effectiveness of the company's internal control over financial reporting. Because a company's internal control cannot be considered effective if one or more material weaknesses exist, to form a basis for expressing an opinion, the auditor must plan and perform the audit to obtain appropriate evidence that is sufficient to obtain reasonable assurance about whether material weaknesses exist as of the date specified in management's assessment. A material weakness in internal control over financial reporting may exist even when financial statements are not materially misstated.
4. The auditor should use the same suitable, recognized control framework to perform his or her audit of internal control over financial reporting as management uses for its annual evaluation of the effectiveness of the company's internal control over financial reporting.

The PCAOBs pronouncements are important, but in practice, the AICPA and PCAOB focus on the “opinions” of auditors and management, which are based on evidence. But they are not “facts,” “actual real-world outcomes,” or “events” that cause real-world losses. In contrast, there are substantive events—security breaches—that are real-world events that cause loss (monetary, reputational, data). These events will occur whether or not an audit is performed. The opinions of auditors, ideally, should predict the probability of such adverse events occurring. The original intent of SOX was to prevent real failures such as occurred at Enron and Worldcom, and the consequent losses suffered by stakeholders.

The remainder of this chapter reviews the evidence for effectiveness of SOX in detecting or predicting the probability of control failures that result in real-world losses. After that, I develop an autoencoder machine learning model to determine whether there is statistically significant information content in SOX reports about subsequent control failures—specifically seven different classes of security breaches. Once we are assured that SOX really does contain statistically significant information about control failures, I use a predictive machine learning model that accommodates unbalanced datasets. Unbalanced datasets are a significant problem in investigating actual control failures. Criminals do not report their successful security breaches, and often client firms do not themselves know that they have errors, omissions, or systems intrusions.

Academic Research on SOX Effectiveness

Academic research on SOX reporting has, to date, focused on internal consistency, compliance, and accrual accounting assessments rather than management of external risks such as fraud and security breaches. We would expect a high correlation between auditor's assessment of controls and the external validation of their effectiveness through study of resulting breaches and other control failures. Prior research by Rice et al. (2014) has focused on the internal consistency of audit results and financial reporting, but not on external security threats.

Compliance cost and information content of SOX filings, with respect to other financial statistics reported by companies, have been studied almost since the inception of SOX. Rice and Weber (2012); Rice et al. (2014) studied SOX 404 reporting for a sample of firms who restated their financials, whose original misstatements are linked to underlying control weaknesses. They found that restatements, i.e., altering a prior year's financial statements due to auditor stakeholder or legal pressure, provided a great deal of information about control weaknesses that could lead to security breaches and management's awareness of these weaknesses. In general, restating firms did acknowledge their existing control weaknesses during their misstatement periods and the probability of reporting existing weaknesses is negatively associated with external capital needs, firm size, non-audit fees, and the presence of a large audit firm. Restatements are positively correlated with financial distress auditor effort previously reported control weaknesses, other restatements, and recent auditor or management changes. Rice and Weber (2012); Rice et al. (2014) found no evidence that penalties are more likely for firm managers or auditors that fail to report existing control weaknesses; rather, they concluded that class action lawsuits management turnover and auditor turnover are all more likely in the wake of a restatement when control weaknesses had previously been reported. They concluded that existing public and private enforcement mechanisms surrounding SOX 404 are unlikely to provide strong

incentives for compliance and offer a potential explanation for why most restatements are issued by firms that previously claimed to have effective internal controls.

Bedard et al. (2009), Hoitash et al. (2009) and Bedard and Graham (2011) examined the detection and severity classification of internal control deficiencies finding that external auditors, during their Section 404 audit, detect about three-fourths of unremediated internal control deficiencies. Ge et al. (2016) looked at a sample of 261 companies that disclosed at least one material weakness in internal control in their SOX filings finding that poor internal control is usually related to an insufficient commitment of resources for accounting controls with the most common account specific material weaknesses occurring in accounts receivable and inventory. SOX 302 disclosures, in contrast, tended to describe internal control problems in complex accounts such as the derivative and income tax accounts. They found that disclosing a material weakness is positively associated with business complexity, e.g., multiple segments and foreign currency, negatively associated with firm size, e.g. market capitalization; and negatively associated with firm profitability metrics, e.g. return on assets. Lin et al. (2011) investigated the role that a firm's internal audit function plays in the disclosure of material weaknesses reported under SOX 404 using data from 214 firms. They found that material weakness disclosures are negatively correlated with the education level of the internal auditors and positively correlated with the practice of grading audit engagements and external-internal auditor coordination. Ashbaugh-Skaife et al. (2008) reported that SOX disclosed internal control deficiencies were associated with more complex operations, recent organizational changes, greater accounting risk, more auditor resignations and have fewer resources available for internal control. They also found that firms with SOX disclosed internal control deficiencies had more prior SEC enforcement actions, and financial restatements were more likely to use a single dominant audit firm and had more concentrated institutional ownership. Feng et al. (2014, 2009) and Berger et al. (2005) found that internal control deficiencies were correlated with less accurate guidance. In particular, the impact of ineffective internal controls on forecast accuracy was found to be three times larger when the weakness was related to revenues or cost of goods sold. This finding reflects the importance of these two accounts in forecasting earnings. Ashbaugh-Skaife et al. (2008) found that firms that report internal control deficiencies have lower quality accruals as measured by accrual noise and absolute abnormal accruals compared to firms not reporting internal control problems. Additionally, firms whose auditors confirm remediation of previously reported internal control deficiencies exhibit an increase in accrual quality relative to firms that do not remediate their control problems. They found that material weaknesses are correlated with: (1) noise; (2) with accrual noise higher error term variance; and (3) with intentional misstatements that bias earnings upward.

Evidence from Industry on SOX Effectiveness

Industry sources suggest that security breaches are substantially underreported by factors ranging from 3 to 30 (Menn 2012). Cybercrime is a low-risk high return vocation that (Lewis and Baker 2013) estimates annually costs the global economy between \$375 and \$575 billion in losses. Most breaches are not reported as they may signal to customers and investors a lack of internal control. This is especially true with financial institutions such as banks and insurance companies can avoid some public scrutiny of customer-facing retail and Internet firms. In addition, the best hackers are often state-sponsored security professionals using methods that are undetectable by current technologies and can subvert internal controls and compromise assets without ever being detected. Breaches, fraud compromised computer systems, as well as SOX compliance concerns, have motivated billions of dollars of expenditure on computers and information systems upgrades and replacements.

SOX statistics show that the adverse percentage rate for auditor attestations decreased rapidly from a high in 2004 of 16.9 to 10.3 in 2005 and by 2009, had leveled off at around 2, indicating either 1 management and internal staff were increasingly adept at convincing external auditors that their systems were secure and protected against fraud and external threats; or 2 there was a significant improvement in internal control immediately after implementation of SOX, at least in those areas that were included in the SOX audits. There were additional management-only assessments primarily applying to small private firms; although trends for these firms roughly followed those of larger firms, the rate of issuance of adverse opinions typically was around 30 higher than for publicly traded companies (Whalen et al. 2012) while their percentage of adverse opinions has declined in tandem with those of publicly traded firms.

Each of the Big 4 audit firms, which audit almost all of the listed firms on US exchanges, displayed unique biases in rendering adverse attestations:

- Ernst & Young focused on accounts receivables revenue recognition taxes and fixed assets;
- PricewaterhouseCoopers focused on accounts receivables revenue recognition taxes and payables;
- KPMG focused on accounts receivables revenue recognition taxes and inventory; and

- Deloitte & Touche focused on revenue recognition taxes, liabilities, inventory, and executive compensation (Whalen et al. 2012).

These biases are likely to reflect signature audit methods, internal forms and checklists, and audit histories that are unique to individual firms. Thus the firms will consequently allocate larger portions of the audit budget to certain accounts at the expense of others. Additionally, auditors tend to allocate more time to auditing debit balance accounts, assuming double-entry will assure the accuracy of the credit balance accounts. The specific accounts selected for audit depend on firm policy procedures and managing partners.

Information systems security vulnerabilities are addressed in auditor's assertions of whether specific material weaknesses have been found. These vary from year to year as old weaknesses are addressed, and new weaknesses surface. Eight financial reporting areas were cited as poorly controlled in an adverse SOX Opinion:

1. Accounts/Loans Receivable;
2. Investments or Cash;
3. Securities Debt Quasi-Debt Warrants and Equities; Revenue Recognition;
4. Tax Expense/Benefit/Deferral FAS109;
5. Liabilities Payables Reserves and Accrual Estimates;
6. Entity Foreign-Related Party Affiliated Subsidiary;
7. PPE Intangibles Fixed Assets; and
8. Inventory Vendor Cost of Sales.

Only 1, 2, and 8 tend to be control areas that will show up in reported security breaches.

Additionally, eight operational areas commonly appeared in SOX 404 attestations:

1. response to prior SOX 404 opinions;
2. response to material weakness;
3. personnel issues;
4. segregation of duties;
5. restatements of financials;
6. material year-end adjustments;
7. internal audit findings; and
8. IT Systems.

Only 3, 4, and 8 tend to be operational areas that will show up in reported security breaches although focused on revenue recognition taxes, liabilities, inventory, and executive compensation Whalen et al. (2012), Rice and Weber (2012), and Rice et al. (2014) found that 5 (he restatement of financials) was a significant indicator of internal control weaknesses that would lead to adverse SOX 302 and 404 assessments and potential security weaknesses.

Using R to Assess SOX Effectiveness in Predicting Breaches, and Identifying Control Weaknesses

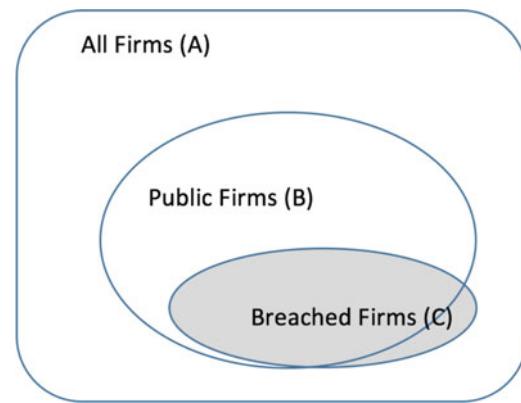
The Sarbanes–Oxley Act was a formal attempt to impose additional joint responsibility on auditors and management for the detection of fraud and external threats. SOX compliance has been contentious since the inception of the Sarbanes–Oxley Act in 2002. Questions have been raised concerning:

1. the effectiveness of expensive SOX reviews;
2. the external validity of SOX assessments for security breaches, errors, fraud, and other external financial threats to the firm; and
3. the usefulness of SOX reporting to investors and other stakeholders.

Unfortunately, these questions have been difficult to study as external breaches, frauds, and other crimes are significantly underreported in business, leading to a reporting bias in datasets that creates difficulties in conducting controlled studies.

Debates over the effectiveness of SOX audits have been continuous. Corporate executives, politicians, and lobbyists have argued that the Sarbanes–Oxley Act SOX of 2002 is a cumbersome and costly regulation that is not effective (Drawbaugh and Aubin 2012). Compliance with just section 404 has been estimated to average \$1.7 million per firm annually, and arguments have even been made before the US Supreme Court that SOX is unconstitutional de Mesa Graziano and Sinnett (2007).

Fig. 1 Overlap of SOX and breach datasets



Publications on the market and economy-wide effects of SOX implementation have opined with case studies and polemics as well as empirical studies. These have been studied in Bratton (2003), Romano (2004), Coates and John (2007), Engel et al. (2007), and Kang et al. (2010), but the evidence on whether SOX implementation has improved security and integrity of internal controls is equivocal.

In practice, the AICPA and PCAOB focus on the “opinions” of auditors and management, which are based on audit evidence. But they are not “facts,” “actual real-world outcomes,” or “events” that cause real-world losses. In contrast, there are substantive events—security breaches—that are real-world events that cause loss to the firm. The opinions of auditors, ideally, should predict the probability of such adverse events occurring.

I investigate SOX’s effectiveness, building R-code for an autoencoder, extracting data from SOX filings with the SEC, and security breach reports maintained by Privacy Clearinghouse. Westland (2019) collected SOX annual reports as well as publicly reported security breaches in the 10-year period 2005–2015. That dataset was used in the current chapter’s code, and is comprised of two overlapping groups of firms (Fig. 1):

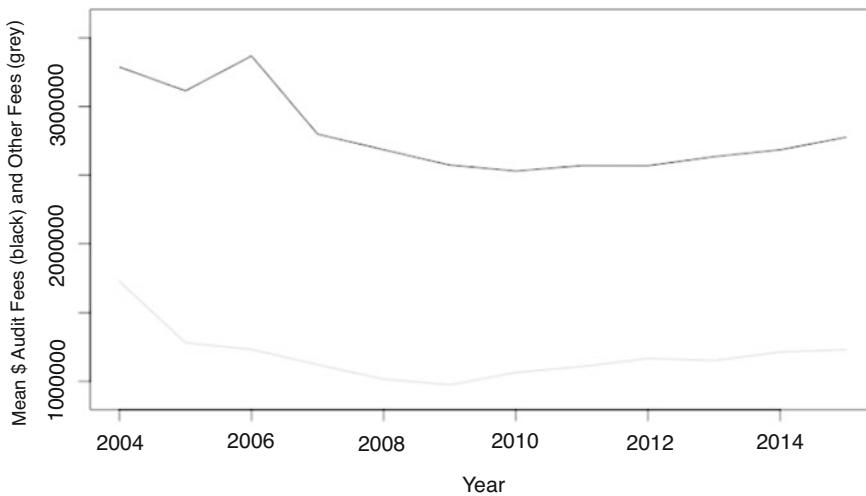
1. publicly traded firms that reported SOX assessments, and
2. publicly traded firms that had reported security breaches.

Figure 1 depicts the overlapping sets of: A all US firms, B publicly traded US firms, and C public and private US firms that have reported security breaches. The dataset that was used in the analysis in this section reports breaches for 213 of the 3398 listed firms which report SOX assessments between 2005 and 2015. Restriction of the analysis to subsets in Fig. 1, public firms that experienced a breach resolves the issue with firms by dropping the breached firms for which we have no SOX 404 reports. This yields a conceptual model for the panel regression of the form representing security breach classifications and other metrics from the Privacy Clearinghouse/Verizon dataset and represents SOX 302 and 404 reported assessments from the Audit Analytics dataset. Datasets were dichotomized into manual and automated control systems.

In any study of predictive capability, it is important to assure that the main effects are not coming from confounding factors unrelated to the main predictors in the SOX reporting. Figure 2 shows the trends in audit and non-audit fees during the investigated period, one in which adverse SOX reports showed a dramatic decline. Audit fees did not appear to be influencing the outcome of SOX reporting, which is what one would hope from independent auditors. The rapid initial decline in the SOX adverse percentage rate for auditor attestations from an initial 16.9 to a current level of around 2 might suggest that SOX 404 audit procedures, which are estimated to cost around \$2 million annually, are not adding significant information about systems security over that provided by management in internal auditors at the firm in their self-reported SOX 302 attestations. To investigate, we tested: SOX 404 attestations add new information to managements self-reported SOX 302 attestations. This tests the veracity of some of the critiques discussed in the introduction that SOX 404 attestations are costly without providing information not already available in SOX 302 attestations.

Table 1 reviews other factors that might influence the outcome of SOX audits, finding that firm capitalization, sales, income, or employee head count had no effect of SOX 404 adverse reports.

Table 2 summarizes the data items extracted from the SOX 302 and 404 reporting that were used in the analysis. Westland (2019) collected 170647 annual SOX 302 and SOX 404 SEC 10-K 10-KSB 10-Q 10-QSB 20-F and 40-F filings of U.S. firms publicly listed on the NYSE or NASDAQ. Firms began to report this data starting April 15, 2005, but advanced filings appeared as early as 2002. Out of these were selected the annual reporting starting with the period that SOX was active. These had the externally audited SOX information Section 404 as well as a comparable set of management reported SOX information Section 302. We acquired the Privacy Rights Clearinghouse/Verizon compilation of 388 major reported U.S. corporate security breaches since 2005, of which 213 applied to 184 NYSE and NASDAQ firms.

**Fig. 2** Audit fees**Table 1** SOX results vs. major firm statistics

Item	Capitalization		Sales		Income		Employees	
	Coefficient	t	Coefficient	t	Coefficient	t	Coefficient	t
Intercept	36,555	4.655	29,591	6.244	2664.76	3.279	74.164	6.029
CARD	5973	0.591	7754	1.167	396.35	0.348	20.796	1.214
DISC	-10,479	-1.228	11,702	2.194	-72.33	-0.079	26.028	1.891
HACK	-14,291	-1.697	-11,826	-2.205	-1432.22	-1.557	-11.49	-0.83
INSD	-6047	-0.698	13,329	2.377	-661.13	-0.688	64.556	4.474
PHYS	-6043	-0.576	4949	0.714	-913.76	-0.769	-6.953	-0.39
PORT	13,050	1.582	12,670	2.475	162.26	0.185	32.534	2.459
STAT	-22,686	-2.298	7116	1.045	-2604.27	-2.23	-8.22	-0.473
UNKN	NA	NA	NA	NA	NA	NA	NA	NA
R-squared	0.05429		0.04479		0.01104		0.04741	
F	10.43		10.74		2.556		11.28	

Confounding SOX attestations effects due to firm size is insignificant with respect to the research questions investigated in this chapter. This was not unexpected as neither Ge et al. (2016) nor Ashbaugh-Skaife et al. (2008) discovered size effects in SOX 404 disclosures though they did find that firm complexity did contribute to the number of adverse attestations under section 404. To assess the possibility of the size of confounding effects in SOX attestations about security breaches, four measures of firm size—sales, net income, market capitalization, and the number of employees were regressed against breaches. Table 2 summarizes the results: the fours are around 1–5, and the only significant relationship we found was that firms with more employees have more insider breaches INSD. This seems to be a logical consequence of larger, more labor-intensive operations. Other financial metrics are only weakly correlated with any type of security breaches. An insignificant amount of variance in breach occurrences depends on other structural parameters and fit statistics. Firm size is an insignificant influence on the *rate of occurrence* of security breaches.

The dataset used in the analysis in this section reports breaches for 213 of the 3398 listed firms that report SOX assessments between 2005 and 2015. Restriction of the analysis to subsets in Fig. 2 public firms that experienced a breach resolves the issue with firms by dropping the breached firms for which we have no SOX 404 reports. This yields a conceptual model for the panel regression of the form representing security breach classifications and other metrics from the Privacy Clearinghouse/Verizon dataset and represents SOX 302 and 404 reported assessments from the Audit Analytics dataset. Datasets were dichotomized into manual and automated systems data.

The rapid initial decline in the SOX adverse percentage rate for auditor attestations from an initial 16.9 to a current level of around 2 might suggest that SOX 404 audit procedures which are estimated to cost around \$2 million annually, are not adding significant information about systems security over that provided by management in internal auditors at the firm in their self-reported SOX 302 attestations. To investigate, we tested: SOX 404 attestations add new information to

Table 2 Data Items from SOX Reports

Item	Description	Values
SOX 302 metrics:		
IS_EFFECTIVE (302)	Self-reported management assertion concerning whether internal controls are effective in the firm	0 = no 1 = yes
MATERIAL_WEAKNESS (302)	Self-reported management assertion concerning whether there exist material weaknesses in internal control systems in the company	0 = controls are effective 1 = there are material weaknesses in the firm's controls
SIG_DEFICIENCY (302)	Self-reported management assertion concerning whether there exist significant deficiencies in internal control systems in the company	0 = controls are effective 1 = there are deficiencies in the firm's controls
SOX 404 metrics:		
IC_OP_TYPE	Manual system or Automated computer system	0 = Manual 1 = Automated
AUDITOR AGREES (404)	Auditors 'attestation' concerning management's SOX 404 assertions	0 = disagree 1 = agrees
COMBINED_IC_OP (404)	Combined opinion on the effectiveness of the firm's internal controls	0 = controls are generally effective 1 = there exists a significant control deficiency
IC_IS_EFFECTIVE (404)	Management asserts that internal controls are effective (Internal Control is Effective)	0 = no 1 = yes
AUDIT_FEE	Cost of SOX 404 audit	(numeric and monetary)

managements self-reported SOX 302 attestations. This tests the veracity of some of the critiques discussed in the introduction that SOX 404 attestations are costly without providing information not already available in SOX 302 attestations.

Confounding SOX attestations effects due to firm size is insignificant with respect to the research questions investigated in this chapter. This was not unexpected as neither Ge et al. (2016) nor Ashbaugh-Skaife et al. (2008) discovered size effects in SOX 404 disclosures though they did find that firm complexity did contribute to the number of adverse attestations under section 404. To assess the possibility of the size of confounding effects in SOX attestations about security breaches, four measures of firm size, sales, net income, market capitalization, and the number of employees were regressed against breaches. Table 1 summarizes the results: the four R^2 are small, and the only marginally significant relationship we found was that firms with more employees have more insider breaches INSD. This seems to be a logical consequence of larger, more labor-intensive operations. Other financial metrics are only weakly correlated with any type of security breaches. An insignificant amount of variance in breach occurrences depends on other structural parameters and fit statistics. Firm size is an insignificant influence on the rate of occurrence of security breaches.

Exploratory Analysis of the SOX-Privacy Clearninghouse Dataset

In the reports of both manual and automated systems, the estimators indicate that firms, on average, in SOX 302 reports attest that internal controls are effective six times as often as in SOX 404 reports. The signs for all the variables are in the expected direction, and there was no evidence that any of the fees charged to the firm influenced the auditor's opinion. This last test is particularly important as it validated the independence in practice of SOX auditors. Although the AICPA admonishes auditors in their demeanor and acts to maintain an independent perspective, there has been no guarantee that this actually happens in the practice. In our tests in both manual and automated systems the firms self-reported material weaknesses were associated with around 80 of SOX 404 negative control assessments while self-reported significant deficiencies appeared in only around 1–2 of SOX 404 negative control assessments. This suggests that managements self-reported material weaknesses are a significant indicator of control problems identified by auditors while significant deficiencies were not (Table 3).

I tested a panel of binomial models logit and probit (Fig. 3) using a General Linear Model but achieved a better fit using standard linear model panel regression. The reason that a binomial model may be preferred is to avoid estimator bias with smaller sample sizes but the current chapter's datasets are sufficiently large that there is no advantage to using probit or logit models, and linear model panel regression approaches will yield unbiased estimators. Test statistics for all the models indicate there are possible unobserved individual effects, and thus we used fixed effects panel regression. An example of a logit model fit, using the base-R `glm` function, appears in the following code chunk.

Table 3 Dataset contents and size

Item	Number of items
SOX reports:	
NASDAQ firms	1835
NYSE firms	1563
Total Firms	3398
SOX reports on automated systems	27,496
SOX reports on manual systems	31,441
Total SOX reports in NASDAQ & NYSE	58,937
Breach data:	
Payment card fraud (CARD)—Fraud involving debit and credit cards that is not accomplished via hacking. For example, skimming devices at point-of-service terminals	12
Unintended disclosure (DISC)—Sensitive information posted publicly on a website mishandled or sent to the wrong party via email fax or mail	70
Hacking or malware (HACK)—Electronic entry by an outside party malware and spyware	106
Insider (INSD)—Someone with legitimate access intentionally breaches information—such as an employee or contractor	51
Physical loss (PHYS)—Lost discarded or stolen non-electronic records such as paper documents	20
Portable device (PORT)—Lost discarded or stolen laptop PDA smartphone portable memory device CD hard drive data tape and so forth	103
Stationary device (STAT)—Lost discarded or stolen stationary electronic device such as a computer or server not designed for mobility	11
Unknown (UNKN)	15
Total number of breaches reported from 2005 to 2016	388
Total number of breaches applicable to NYSE and NASDAQ listed firms	213
Unique firms in breach dataset listed on NYSE and NASDAQ	184

```

library(readr)
library(tidyverse)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

sox_data <-
  read_csv(system.file("extdata", "innerjoinSOXBRCH.csv", package =
  "auditanalytics", mustWork = TRUE)),
  col_types = cols(X1 = col_skip())) %>%
  select(
    IC_IS_EFFECTIVE,
    MATERIAL_WEAKNESS,
    SIG_DEFICIENCY,
    AUDITOR AGREES,
    CARD.x
  )

sox_card_logit <- glm(CARD.x~., family=binomial(link=logit), data=sox_data)
summary(sox_card_logit)

## 
## Call:
## glm(formula = CARD.x ~ ., family = binomial(link = logit), data = sox_data)
## 
## Deviance Residuals:
##      Min        1Q     Median        3Q       Max 
## -0.2902   -0.2852   -0.2852   -0.2852    2.5389

```

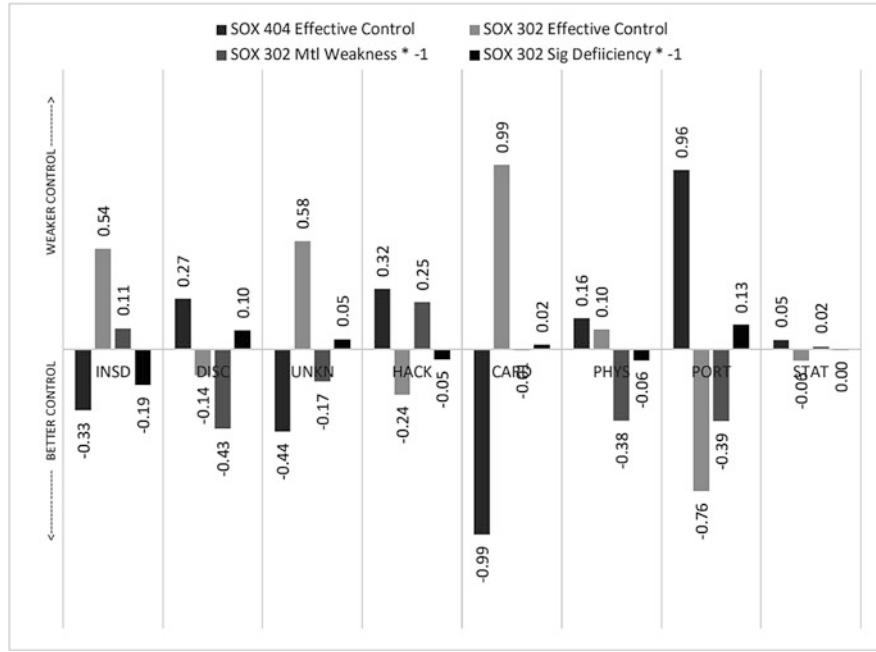


Fig. 3 Automated controls and breaches

```
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -18.02457 3486.50314 -0.005   0.996
## IC_IS_EFFECTIVE      14.84236 3486.50314  0.004   0.997
## MATERIAL_WEAKNESS   -14.84236 3486.50314 -0.004   0.997
## SIG_DEFICIENCY       -17.39277 2348.41165 -0.007   0.994
## AUDITOR AGREES        0.03591   0.60407   0.059   0.953
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 123.17 on 425 degrees of freedom
## Residual deviance: 117.39 on 421 degrees of freedom
## (1086 observations deleted due to missingness)
## AIC: 127.39
##
## Number of Fisher Scoring iterations: 19
```

Notice in the regression results that the signs of coefficients for “internal control effectiveness” and “material weakness” and “significant deficiency” are opposite, because the first is a statement about the effectiveness of controls, while the latter is about the ineffectiveness of controls. AIC values are small, suggesting a good fit from using the logit model.

We concluded that SOX 404 auditors maintain their independence consistent with Generally Accepted Auditing Standard (GAAS). With the investigation of potentially confounding factors of size and auditor independence effectively, completed we can move on to testing our main hypothesis: that SOX 404 and 302 attestations contain information about the future occurrence of specific types of real-world security breaches.

Figures 3 and 4 show the relative effectiveness of SOX assessments in controlling various types of security breaches in automated and manual systems, respectively. Looking more closely at these effects, we can see that SOX auditing has significantly varying effectiveness in predicting the various classes of a security breach, but the results were very similar for manual and automated systems. The model fit was nearly zero for hacking and malware (HACK), stationary devices (STAT), and situations where the cause of the breach was unknown (UNKN), suggesting that current SOX audit practice both inside and outside of the firm provide little information useful in controlling these threats. The estimators for unintended disclosures (DISC) and physical losses (PHYS) were small, which also suggests that SOX audits provide little information useful for

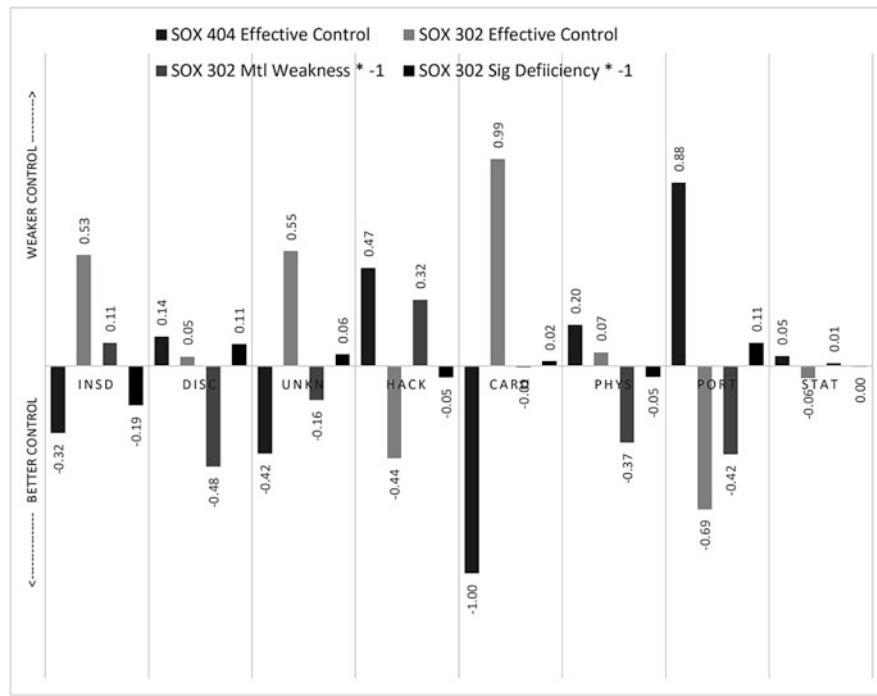


Fig. 4 Manual controls and breaches

controlling these types of breaches. Where SOX provided significant predictive power in identifying future security breaches, management's (SOX 302) assessments were generally much less informative than those provided by SOX 404 audits. This concurs with what we discovered in our previous tests that did not include the breach information, and is consistent with prior research by Bedard et al. (2009) and Bedard and Graham (2011) who found that management is overconfident in assessing internal control effectiveness reflected in self-reporting under SOX302. SOX 404 adverse decisions on effectiveness of controls occurred in 100% of the credit card (CARD) data breaches and around 33% of insider (INSD) breaches. But SOX 404 audits gave "effective" control decisions on 88% of situations where there was a controlled breach concerning a portable (PORT) device. This suggests that employees are subverting strict internal controls by using portable devices that can be carried outside the physical boundaries of the firm, and which can be used more freely than devices that are directly under the firm's control. Furthermore, the SOX auditors appear not to be detecting these subversions of internal control.

Using an Autoencoder to Detect Control Weaknesses

Autoencoders are self-supervised machine learning models, where the generated targets are the input, unmodified. Since only a small percentage of the transactions in the SOX-breach dataset represent credit card fraud at the audited corporations, we have a highly unbalanced classification problem. Traditional classification approaches are a poor choice for decision-making because the dataset contains such a small sample of frauds. Similarly, linear regression models suffer from singular design matrices, and may fail to capture non-linear relationships in real-world data, and predictors are multicolinear, which creates singularities.

To address these problems, I have designed a bottleneck autoencoder model for dimensionality reduction to explore the SOX feature space, similar to the use of principal component analysis. Because of the multicollinearity of accounting datasets, machine learning methods that train many weights in multiple layers are likely to extract this information more effectively to generate a single component that should provide a surrogate for "control weakness." An autoencoder is a neural network that is used to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. For an autoencoder to work well we have a strong initial assumption: that the distribution of variables for normal firms is different from the distribution for ones with control weaknesses. Since security breaches should have a different distribution than normal transactions (because their reporting is heavily manipulated by criminals and victimized firms) we expect that our autoencoder will have higher reconstruction errors on security breaches than on normal transactions. This

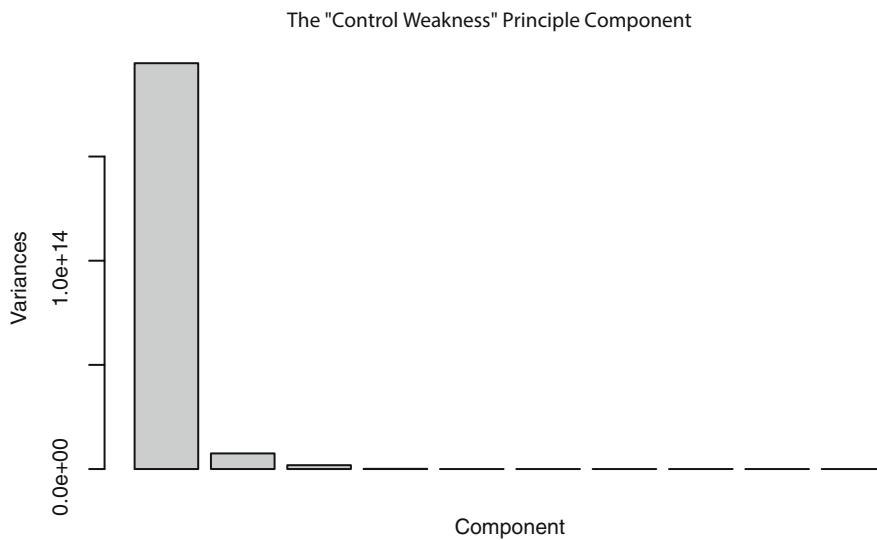


Fig. 5 Eigenvalues of the PCA decomposition

means that we can use the reconstruction error as a quantity that indicates whether a client’s systems are more likely to suffer security breaches. I use the autoencoder’s dimensionality reduction to explore the dataset’s feature space (similar to what we could do with a principal component analysis, as in Fig. 5).

I designed two autoencoders, one in Tensorflow/Keras for reference, and one in H2O for analysis of SOX control weaknesses, H2O (<https://www.h2o.ai/>) is a popular open source machine learning platform accessible in R which runs both in-memory and on H2O servers, offering an interface that is slightly simpler to use than Tensorflow’s. H2O supports the most widely used statistical and machine learning algorithms, including gradient boosted machines, generalized linear models, deep learning, and so forth, and contains an *AutoML* function that automatically runs through all the algorithms and their hyperparameters to produce a leaderboard of the best models. For larger models, H2O4GPU provides GPU-accelerated machine learning on hardware platforms with, e.g., NVidia’s GPUs.

As is typical with accounting data, predictors are multicolinear, which creates problems for traditional regression approaches because of the singularities. Figure 5’s cursory plot of principal components (a linear clustering algorithm) reveals that most of the variance is weighted on the first principal component. This does not mean that the data does not convey information; it is just that the information involves more complex multivariate non-linear relationships. Because of this, machine learning methods that train many weights in multiple layers are likely to extract this information more effectively. This single component is likely to be a surrogate for “control weakness,” suggesting covariance between SOX reports and the occurrence of security breaches (otherwise, we would likely see clustering around two large principle components).

Figure 5 shows the multicollinearity of the combined SOX and Privacy Clearinghouse information—most of the information in both datasets can be captured in one principal component. Given that SOX reports control weaknesses, and security breaches occur because of control weaknesses, this principal component is a strong surrogate for formal definitions of “control weakness.”

Preprocessing

Preprocessing splits the dataset into train and test sets and then *min-max* normalize the data (this is done because neural networks work much better with small input values). Based on the `date` variable, we will use the results before 2011 for training and the rest for testing. This is good practice because when using the model, we want to predict future breaches based on SOX reports issued previously.

Two helper functions are used to normalize the database (i.e., adjust the values to similar scale, which helps the machine learning algorithms converge more quickly). They are not really needed for this data, which is binary [0, 1] data, but if you wish to reuse this code on other data, they will be useful. The helper functions include:

1. a function to get descriptive statistics about the dataset that are used for min-max scaling (using the same normalization constants for training and test sets), and

2. a function to perform the min-max scaling.

```

library(tidyverse)
library(purrr)
library(keras)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

## function to get descriptive statistics

get_desc <- function(x) {
  map(x, ~list(
    min = min(.x),
    max = max(.x),
    mean = mean(.x),
    sd = sd(.x)
  ))
}

## function to normalize values

normalization_minmax <-
  function(x, desc) {
    map2_dfc(
      x,
      desc,
      ~(.x - .y$min) / (.y$max - .y$min))
  }

## get data
sox <-
  read_csv(
    system.file("extdata", "innerjoinSOXBRCH.csv", package = "auditAnalytics",
               mustWork = TRUE)),
    col_types = cols(X1 = col_skip())
  ) %>%
  select(
    date,
    CARD.x,
    IS_EFFECTIVE,
    MATERIAL_WEAKNESS,
    SIG_DEFICIENCY,
    IC_OP_TYPE,
    AUDITOR AGREES,
    COMBINED_IC_OP,
    IC_IS_EFFECTIVE,
    AUDIT_FEES
  )
sox[is.na(sox)] <- 0

## create normalized datasets for ML

```

```

desc <- sox %>% get_desc()

x_train <-
  sox %>%
  filter(date <= 2011) %>%
  normalization_minmax(desc) %>%
  select(-c(date))

x_test <-
  sox %>%
  filter(date > 2011) %>%
  normalization_minmax(desc) %>%
  select(-c(date))

y_train <-
  x_train[, "CARD.x"] %>% ## CARD.x the fraud indicator
  as.matrix()

y_test <-
  x_test[, "CARD.x"] %>%
  as.matrix()

x_train <-
  x_train %>%
  select(-c(CARD.x)) %>%
  as.matrix()

x_test <-
  x_test %>%
  select(-c(CARD.x)) %>%
  as.matrix()

```

Tensorflow Implementation of the Autoencoder

Here is the reference autoencoder in Tensorflow/Keras. The model converges quickly and accurately, suggesting that SOX control weakness reporting can be a successful predictor of vulnerability to credit card fraud at client firms (Fig. 6).

```

## assure that the Keras API is available

library(keras)
install_keras()

## Installation complete.

## define the model in Keras,
## this is a symmetric autoencoder with 4 fully connected layers.

model <- keras_model_sequential()
model %>%
  layer_dense(units = 100, activation = "tanh", input_shape = ncol(x_train)) %>%
  layer_dense(units = 10, activation = "tanh") %>%
  layer_dense(units = 100, activation = "tanh") %>%

```

```

layer_dense(units = ncol(x_train))

summary(model)

## Model: "sequential"
##
## Layer (type)                  Output Shape         Param #
## =====
## dense_3 (Dense)              (None, 100)          900
## dense_2 (Dense)              (None, 10)           1010
## dense_1 (Dense)              (None, 100)          1100
## dense (Dense)                (None, 8)            808
## Total params: 3,818
## Trainable params: 3,818
## Non-trainable params: 0
## 

## the model is a binary classifier, so we use an entropy loss

model %>% compile(
  loss = "binary_crossentropy",
  optimizer = "adam",
  metrics= 'accuracy'
)

checkpoint <- callback_model_checkpoint(
  filepath = "model.hdf5",
  save_best_only = TRUE,
  period = 1,
)

```

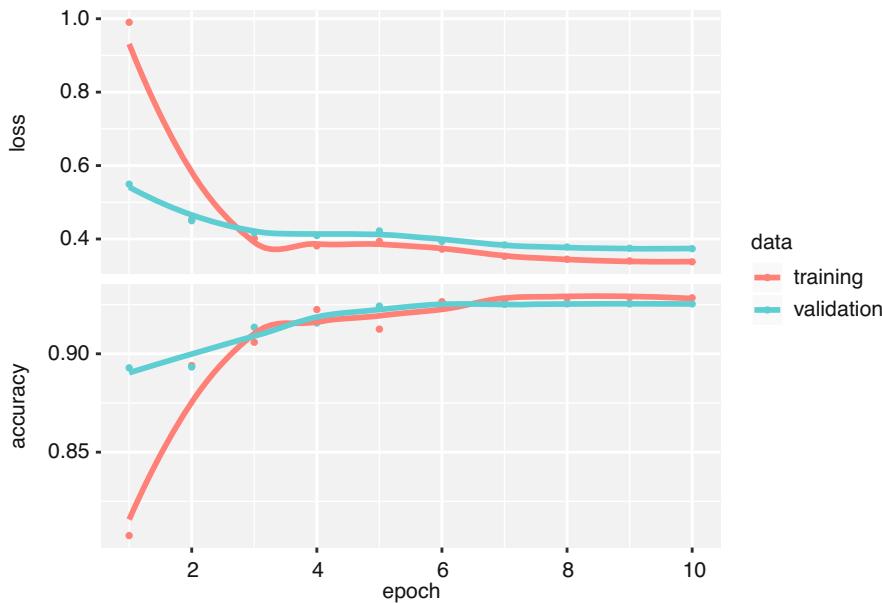


Fig. 6 Convergence of symmetric autoencoder with four fully-connected layers in detecting fraud with SOX data

```

    verbose = 1
)

## Warning in callback_model_checkpoint(filepath = "model.hdf5", save_best_only =
## TRUE, : The period argument is deprecated since TF v1.14 and will be ignored.
## Use save_freq instead.

early_stopping <- callback_early_stopping(patience = 5)

history <- model %>% fit(
  x = x_train[y_train == 0,],
  y = x_train[y_train == 0,],
  epochs = 10,
  batch_size = 32,
  validation_data = list(x_test[y_test == 0,], x_test[y_test == 0,]),
  callbacks = list(checkpoint, early_stopping)
)

plot(history)

```

The H2O Implementation of Autoencoders and Anomaly Detection for Fraud Analytics

The *h2o* package (<https://www.h2o.ai/>) provides the R interface to the H2O open-source machine learning platform, which supports gradient boosted machines, generalized linear models, deep learning, and models. It has become popular due to *AutoML*, which runs through algorithms and their hyperparameters to produce a leaderboard of the best models. It works on a wide range of big data infrastructures, Hadoop or Spark clusters and can directly read files from HDFS, Spark, S3, Azure Data Lake, and various other data sources into its in-memory distributed key-value store, and is relatively easy to productize. R also offers a GPU-accelerated H2O package *h2o4gpu* package. The following example applies H2O to industry-wide security assessment using SOX data. A more extensive implementation of a stochastic grid hyperparameter search for optimal fit can be found at Westland (2020).

The SOX-card fraud dataset needs special treatment when performing machine learning because they are severely unbalanced. Only ~3.44% of audits were for companies that experienced a credit card fraud, affirming that the database is highly unbalanced.

```

cat("% of SOX reports where client had credit fraud during the year = "
  , 100*sum(sox$CARD.x)/length(sox$CARD.x), "% of audits")

```

```

## % of SOX reports where client had credit fraud during
the year = 3.439153 % of audits

```

In such cases, the optimal predictor would predict in 100% of audits that there would be no credit card frauds, and that predictor would be right a respectable 96.56% of the time. A closer look at audit fee distributions and dates of frauds suggests that fraud occurrence is not random, and suggests there is information to be extracted from the dataset (Figs. 7 and 8).

```

library(tidyverse)

sox %>%
  ggplot(aes(x = AUDIT_FEES)) +
  geom_bar(color = "grey", fill = "lightgrey") +
  theme_bw() +
  scale_y_continuous(trans='log2') +
  facet_wrap(~ CARD.x, scales = "free", ncol = 2)

```

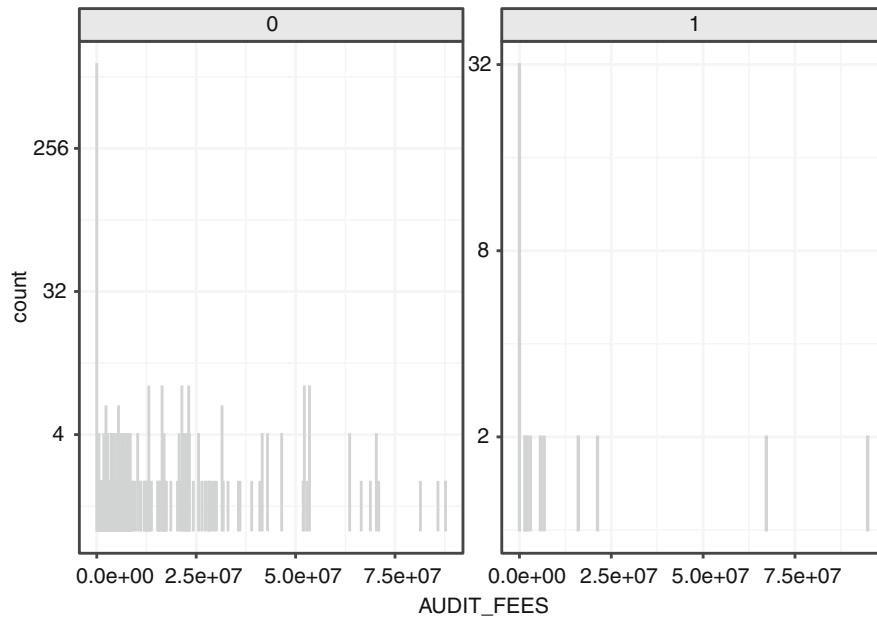


Fig. 7 Count profiles of fraud and non-fraud cases vs. audit fees

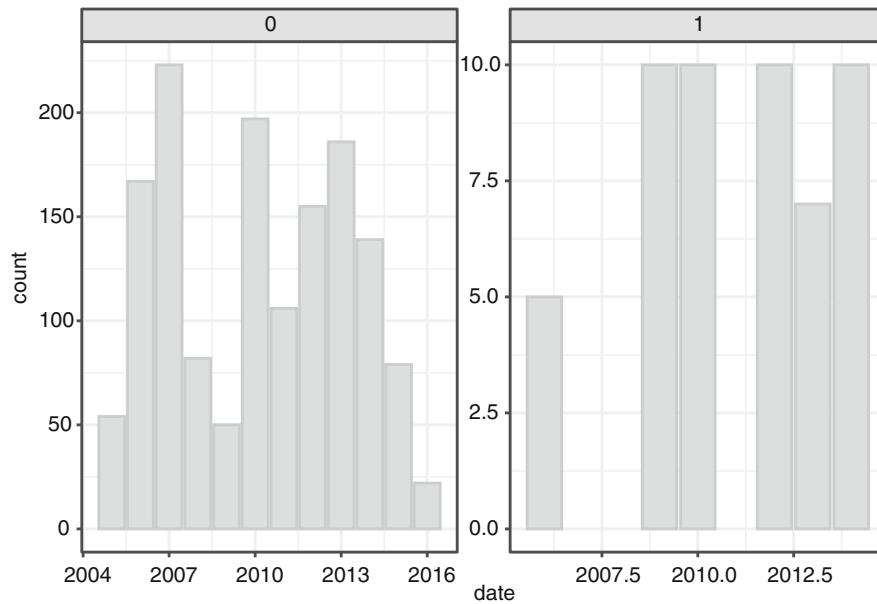


Fig. 8 Count profiles of fraud and non-fraud cases by reporting dates

```
SOX %>%
  ggplot(aes(x = date)) +
  geom_bar(color = "grey", fill = "lightgrey") +
  theme_bw() +
  facet_wrap(~ CARD.x, scales = "free", ncol = 2)
```

Interestingly, fraudulent credit card transactions had much more variability in the audit fees charged. This could have many causes, but it also seems to suggest that where fraud is a possibility, there may be greater uncertainty in the client's controls that necessitates additional audit steps.

For modeling, I used R's H2O implementation and converted the dataset to H2O format, splitting the dataset into training and test sets.

```
library(h2o)

## -----
## 
## Your next step is to start H2O:
##   > h2o.init()
## 
## For H2O package documentation, ask for help:
##   > ??h2o
## 
## After starting H2O, you can use the Web UI at http://localhost:54321
## For more information visit http://docs.h2o.ai
## 
## -----
## 
## Attaching package: 'h2o'

## The following objects are masked from 'package:stats':
## 
##   cor, sd, var

## The following objects are masked from 'package:base':
## 
##   &&, %*%, %in%, ||, apply, as.factor, as.numeric, colnames,
##   colnames<-, ifelse, is.character, is.factor, is.numeric, log,
##   log10, log1p, log2, round, signif, trunc
```

```
h2o.init(nthreads = -1)
```

```
## 
## H2O is not running yet, starting it now...
## 
## Note: In case of errors look at the following log files:
##       /tmp/Rtmpk1tm04/h2o_westland_started_from_r.out
##       /tmp/Rtmpk1tm04/h2o_westland_started_from_r.err
## 
## 
## Starting H2O JVM and connecting: . Connection successful!
## 
## R is connected to the H2O cluster:
##   H2O cluster uptime:      1 seconds 179 milliseconds
##   H2O cluster timezone:    America/Chicago
##   H2O data parsing timezone: UTC
##   H2O cluster version:     3.28.0.2
##   H2O cluster version age: 2 months and 10 days
##   H2O cluster name:        H2O_started_from_R_westland_cmp084
##   H2O cluster total nodes: 1
##   H2O cluster total memory: 29.97 GB
##   H2O cluster total cores: 20
##   H2O cluster allowed cores: 20
##   H2O cluster healthy:     TRUE
##   H2O Connection ip:       localhost
##   H2O Connection port:     54321
##   H2O Connection proxy:    NA
```

```

##      H2O Internal Security:      FALSE
##      H2O API Extensions:      Amazon S3, XGBoost, Algos, AutoML,
##                                Core V3, TargetEncoder, Core V4
##      R Version:      R version 3.6.1 (2019-07-05)

creditcard_hf <- as.h2o(sox)

##      |

splits <- h2o.splitFrame(creditcard_hf,
                        ratios = c(0.30, 0.30),
                        seed = 123)

train_unsupervised <- splits[[1]]
train_supervised <- splits[[2]]
test <- splits[[3]]

response <- "CARD.x"
features <- setdiff(colnames(train_unsupervised), response)

```

In H2O, I start by training an unsupervised autoencoder neural network by setting `autoencoder = TRUE`. Similar to the Tensorflow model, I use a bottleneck model that reduces the dimensionality of the data down to two nodes/dimensions. The autoencoder then + learns which credit card transactions are similar and which transactions are outliers or anomalies. Autoencoders can be sensitive to outliers, and thus may be prone to overfitting.

```

library(h2o)

model_nn <- h2o.deeplearning(x = features,
                               training_frame = train_unsupervised,
                               model_id = "model_nn",
                               autoencoder = TRUE,
                               reproducible = TRUE,
                               seed = 123,
                               hidden = c(10, 2, 10),
                               epochs = 100,
                               activation = "Tanh")

##      |

model_nn

## Model Details:
## =====
##
## H2OAutoEncoderModel: deeplearning
## Model ID: model_nn
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 251
## weights/biases, 8.2 KB, 47,100 training samples, mini-batch size 1
## layer units type dropout    11      12 mean_rate rate_rms momentum
## 1     1     9 Input   0.00 %     NA     NA     NA     NA     NA
## 2     2     10 Tanh   0.00 % 0.000000 0.000000  0.019855 0.016872 0.000000
## 3     3     2 Tanh   0.00 % 0.000000 0.000000  0.006878 0.003528 0.000000
## 4     4     10 Tanh   0.00 % 0.000000 0.000000  0.003076 0.001672 0.000000
## 5     5     9 Tanh     NA 0.000000 0.000000  0.004410 0.003630 0.000000
## mean_weight weight_rms mean_bias bias_rms
## 1          NA        NA       NA     NA
## 2     0.089231  0.398164  0.008491 0.157678
## 3    -0.046698  0.578193  0.252433 0.028225
## 4     0.062787  0.824127 -0.051472 0.199662
## 5     0.027984  0.399891  0.106388 0.141101

```

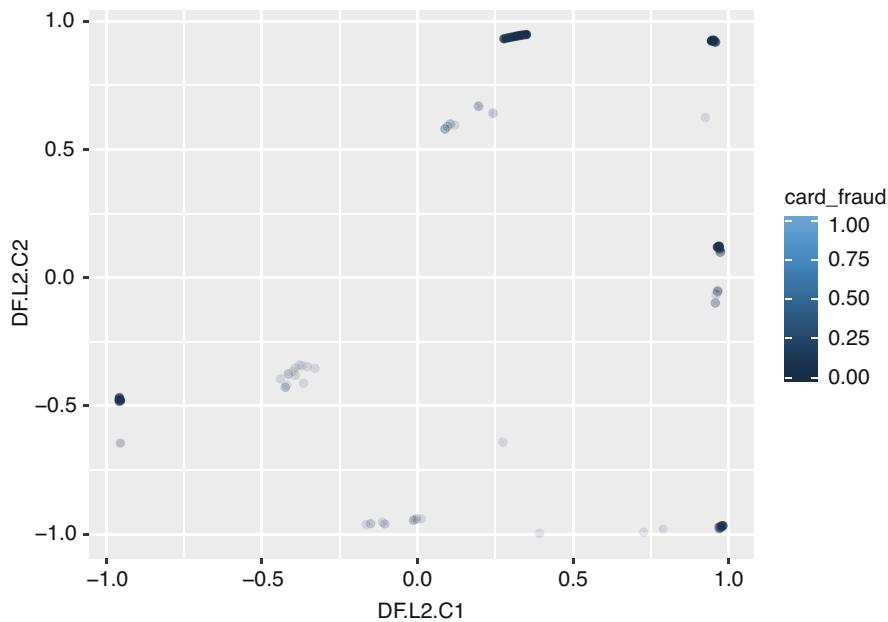


Fig. 9 Autoencoder layer-2 detection of fraud cases with SOX data

```
##  
##  
## H2OAutoEncoderMetrics: deeplearning  
## ** Reported on training data. **  
##  
## Training Set Metrics:  
## =====  
##  
## MSE: (Extract with 'h2o.mse') 0.0246203  
## RMSE: (Extract with 'h2o.rmse') 0.1569086
```

```
#Convert to autoencoded representation  
test_autoenc <- h2o.predict(model_nn, test)
```

Bottleneck autoencoders offer the ability to extract features captured in the middle layers, similar to factor analysis or PCA, but the features need not be linear, nor must they adhere to the simple optimization of traditional exploratory statistics. We can extract hidden features with the `h2o.deepfeatures()` function and plot to show the reduced representation of input data (Fig. 9).

```
library(tidyverse)  
  
train_features <- h2o.deepfeatures(  
  model_nn,  
  train_unsupervised,  
  layer = 2  
) %>%  
  as.data.frame() %>%  
  mutate(card_fraud = as.vector(train_unsupervised[, "CARD.x"]))
```

```
ggplot(  
  train_features,  
  aes(x = DF.L2.C1, y = DF.L2.C2, color = card_fraud)) +  
  geom_point(alpha = 0.1)
```

Graphing the two features extracted from layer 2 of the autoencoder shows that feature DF.L2.C1 clearly dichotomizes audits into those with high potential for card fraud, and those without. Dimensionality reduction through an autoencoder model alone can clearly identify fraud in this dataset. This concept can be carried forward further, using the reduced dimensionality representation of one of the hidden layers as features for model training. An example would be to use the ten features from the first or third hidden layer:

```
# Take the third hidden layer

train_features <-
  h2o.deepfeatures(
    model_nn,
    train_unsupervised,
    layer = 3
  ) %>%
  as.data.frame() %>%
  mutate(card_fraud = as.factor(as.vector(train_unsupervised[, "CARD.x"]))) %>%
  as.h2o()

response="card_fraud"

features_dim <- setdiff(colnames(train_features), response)

model_nn_dim <- h2o.deeplearning(y = response,
                                    x = features_dim,
                                    training_frame = train_features,
                                    reproducible = TRUE,
                                    balance_classes = TRUE,
                                    seed = 123,
                                    hidden = c(10, 2, 10),
                                    epochs = 100,
                                    activation = "Tanh")

h2o.saveModel(model_nn_dim, path="model_nn_dim", force = TRUE)

model_nn_dim

## Model Details:
## =====
##
## H2OBinomialModel: deeplearning
## Model ID: DeepLearning_model_R_1585689487935_1
## Status of Neuron Layers: predicting card_fraud, 2-class classification,
## bernoulli distribution, CrossEntropy loss, 184 weights/biases, 7.6 KB,
## 11,856 training samples, mini-batch size 1
##   layer units      type dropout      11      12 mean_rate rate_rms momentum
## 1     1    10    Input  0.00 %       NA       NA       NA       NA
## 2     2    10    Tanh  0.00 %  0.000000  0.000000  0.003902  0.003501  0.000000
## 3     3     2    Tanh  0.00 %  0.000000  0.000000  0.001949  0.000930  0.000000
## 4     4    10    Tanh  0.00 %  0.000000  0.000000  0.001565  0.000780  0.000000
## 5     5     2  Softmax       NA  0.000000  0.000000  0.006979  0.003106  0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1           NA          NA          NA
## 2     0.034443    0.335956  0.005633  0.047992
## 3    -0.053079    0.416090 -0.051265  0.013347
## 4     0.024471    0.476211  0.014101  0.024998
```

```

## 5      0.158343   1.464484 -0.000000 0.051659
##
##
## H2OBinomialMetrics: deeplearning
## ** Reported on training data. **
## ** Metrics reported on full training frame **
##
## MSE:  0.4347381
## RMSE: 0.6593467
## LogLoss: 1.382748
## Mean Per-Class Error: 0.3541667
## AUC: 0.6549106
## AUCPR: 0.5983136
## Gini: 0.3098213
##
## Confusion Matrix (vertical: actual; across: predicted) for
## F1-optimal threshold:
##          0    1    Error     Rate
## 0    163 293 0.642544 =293/456
## 1     30 426 0.065789 =30/456
## Totals 193 719 0.354167 =323/912
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                  metric threshold     value idx
## 1                 max f1  0.047196  0.725106 155
## 2                 max f2  0.041725  0.859404 158
## 3                 max f0point5 0.047196  0.639256 155
## 4                 max accuracy 0.047196  0.645833 155
## 5                 max precision 0.098203  0.857143 11
## 6                 max recall  0.041725  1.000000 158
## 7                 max specificity 0.098220  0.997807 0
## 8                 max absolute_mcc 0.047196  0.357033 155
## 9     max min_per_class_accuracy 0.054306  0.587719 150
## 10    max mean_per_class_accuracy 0.047196  0.645833 155
## 11                 max tns  0.098220 455.000000 0
## 12                 max fns  0.098220 456.000000 0
## 13                 max fps  0.000901 456.000000 205
## 14                 max tps  0.041725 456.000000 158
## 15                 max tnr  0.098220  0.997807 0
## 16                 max fnr  0.098220  1.000000 0
## 17                 max fpr  0.000901  1.000000 205
## 18                 max tpr  0.041725  1.000000 158
##
## Gains/Lift Table: Extract with 'h2o.gainsLift(<model>, <data>)'
## or 'h2o.gainsLift(<model>, valid=<T/F>, xval=<T/F>)'

```

One way to explore the performance of this model is to plot the “gains” chart. Gain at a given percentile is the ratio of the cumulative number of targets (e.g., frauds) up to that percentile to the total number of targets (e.g., frauds) in the entire dataset. Here is a gains chart of the model (Fig. 10).

```

library(tidyverse)
library(knitr)
library(kableExtra)
library(ggplot2)

## Gains/Lift Tables

```

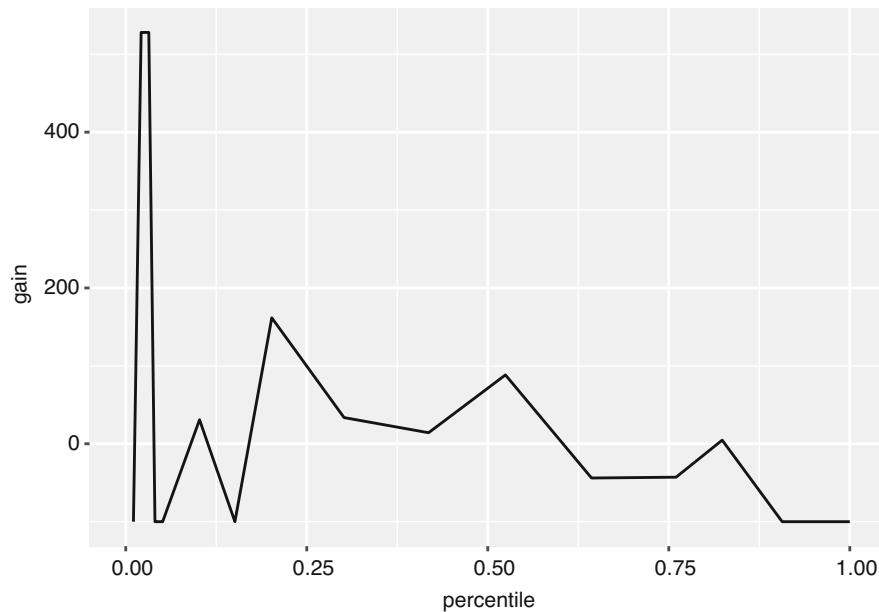


Fig. 10 Gains chart of autoencoder layer-3 in detecting fraud with SOX data

```

gains <- h2o.gainsLift(model_nn_dim, train_features, valid=F, xval=F)

gains %>%
  ggplot(aes(y=gains$gain, x=gains$cumulative_data_fraction)) +
  geom_line() +
  xlab("percentile") + ylab("gain") +
  xlim(0,1)
  
```

```

# For measuring model performance on test data, we need to
# convert the test data to the same reduced dimensions as the # trainings data:
  
```

```

test_dim <- h2o.deepfeatures(model_nn, test, layer = 3)
  
```

```

h2o.predict(model_nn_dim, test_dim) %>%
  as.data.frame() %>%
  mutate(actual = as.vector(test[, "CARD.x"])) %>%
  group_by(actual, predict) %>%
  summarise(n = n()) %>%
  mutate(freq = n / sum(n))
  
```

```

## # A tibble: 4 x 4
## # Groups:   actual [2]
##   actual predict     n   freq
##   <int> <fct>    <int>   <dbl>
## 1     0     0        165  0.292
## 2     0     1        400  0.708
## 3     1     0         2  0.0909
## 4     1     1        20  0.909
  
```

This looks quite good in terms of identifying fraud cases: 91% of fraud cases were identified! However, many non-fraud cases were also classified as fraud. In actual audit applications, this would not be a good model, and it is necessary to look further to improve the sensitivity and specificity of predictions.

Anomaly Detection

We can also ask which instances were considered outliers or anomalies within our test data, using the `h2o.anomaly()` function. Based on the autoencoder model that was trained before, the input data will be reconstructed, and for each instance, the mean squared error (MSE) between actual value and reconstruction is calculated for both class labels (Fig. 11).

```
anomaly <- h2o.anomaly(model_nn, test) %>%
  as.data.frame() %>%
  tibble::rownames_to_column() %>%
  mutate(
    card_fraud =
      as.factor(
        as.vector(
          test[, "CARD.x"]
        )
      )
  )

mean_mse <- anomaly %>%
  group_by(card_fraud) %>%
  summarise(mean = mean(Reconstruction.MSE))

## This, we can now plot:

ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE,
  color = as.factor(card_fraud))) +
  geom_point(alpha = 0.3) +
  geom_hline(data = mean_mse, aes(yintercept = mean, color = card_fraud)) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "instance number",
    color = "card_fraud")
```

The plot does not show a clear-cut classification into fraud and non-fraud cases but the mean MSE is definitely higher for fraudulent transactions than regular ones.

We can now identify outlier instances by applying an MSE threshold for what we consider outliers. We could, e.g., say that we consider every instance with an $\text{MSE} > 0.02$ (chosen according to the plot above) to be an anomaly/outlier.

```
anomaly <- anomaly %>%
  mutate(outlier = ifelse(Reconstruction.MSE > 0.02, "outlier", "no_outlier"))

anomaly %>%
  group_by(card_fraud, outlier) %>%
  summarise(n = n()) %>%
  mutate(freq = n / sum(n))

## # A tibble: 4 x 4
## # Groups:   card_fraud [2]
##   card_fraud outlier     n   freq
##   <fct>       <chr>     <int>  <dbl>
## 1 0           no_outlier 418  0.740
## 2 1           outlier    27   0.0530
```

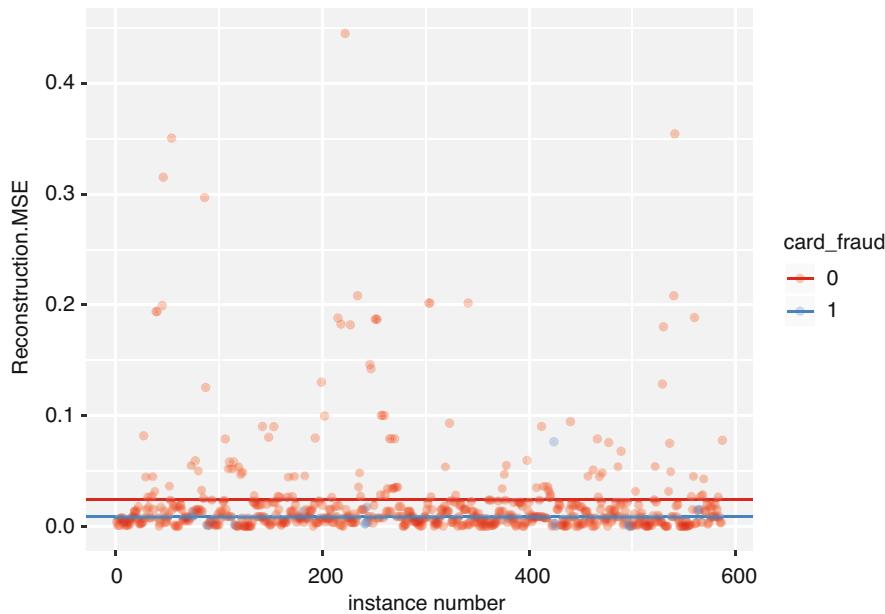


Fig. 11 Anomalies and outliers in detecting fraud with SOX data

```
## 2 0      outlier      147 0.260
## 3 1      no_outlier   21 0.955
## 4 1      outlier      1 0.0455
```

For this dataset, outlier detection is insufficient to correctly classify fraudulent credit card transactions either. This suggests that further analysis is necessary. In this example, we will continue with supervised learning to attempt to use SOX data to predict credit card fraud.

Pre-trained Supervised Model

In this section, the autoencoder model is used as a pre-training input for a supervised model. This model will use the weights from the autoencoder for model fitting.

```
train_features <- h2o.deepfeatures(
  model_nn,
  train_unsupervised,
  layer = 2
) %>%
  as.data.frame() %>%
  mutate(card_fraud = as.vector(train_unsupervised[, "CARD.x"]))

response <- "CARD.x"

features <- setdiff(colnames(train_supervised), response)

model_nn_2 <- h2o.deeplearning(y = response,
                                 x = features,
                                 training_frame = train_supervised,
                                 pretrained_autoencoder = "model_nn",
                                 reproducible = TRUE,
                                 balance_classes = F,
```

```

                    ignore_const_cols = TRUE,
                    seed = 123,
                    hidden = c(10, 2, 10),
                    epochs = 100,
                    activation = "Tanh")

model_nn_2

## Model Details:
## =====
##
## H2OResgressionModel: deeplearning
## Model ID: DeepLearning_model_R_1585689487935_17
## Status of Neuron Layers: predicting CARD.x, regression,
## gaussian distribution, Quadratic loss, 163 weights/biases,
## 7.3 KB, 45,400 training samples, mini-batch size 1
##   layer units type dropout      11      12 mean_rate rate_rms momentum
## 1     1     9 Input  0.00 %       NA       NA       NA       NA       NA
## 2     2    10 Tanh  0.00 % 0.000000 0.000000 0.006301 0.007721 0.000000
## 3     3     2 Tanh  0.00 % 0.000000 0.000000 0.002035 0.001171 0.000000
## 4     4    10 Tanh  0.00 % 0.000000 0.000000 0.002826 0.008509 0.000000
## 5     5     1 Linear NA 0.000000 0.000000 0.000234 0.000109 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1          NA          NA          NA          NA
## 2  0.090567  0.407879  0.031163  0.204913
## 3 -0.056527  0.640207  0.351327  0.060788
## 4  0.079290  0.937924 -0.047194  0.219902
## 5  0.109594  0.293195  0.009355  0.000000
##
##
## H2OResgressionMetrics: deeplearning
## ** Reported on training data. **
## ** Metrics reported on full training frame **
##
## MSE: 0.03092839
## RMSE: 0.1758647
## MAE: 0.06211831
## RMSLE: 0.1217686
## Mean Residual Deviance : 0.03092839

pred <- as.data.frame(h2o.predict(object = model_nn_2, newdata = test)) %>%
  mutate(actual = as.vector(test[, "CARD.x"]))

pred %>%
  group_by(actual, predict) %>%
  summarise(n = n()) %>%
  mutate(freq = n / sum(n))

## # A tibble: 258 x 4
## # Groups: actual [2]
##   actual predict     n   freq
##   <int>    <dbl> <int>   <dbl>
## 1 0 -0.0202     1 0.00177
## 2 0 -0.0201     1 0.00177
## 3 0 -0.0195     1 0.00177

```

```

## 4      0 -0.0154      2 0.00354
## 5      0 -0.0120      1 0.00177
## 6      0 -0.00903     1 0.00177
## 7      0 -0.00842     1 0.00177
## 8      0 -0.00832     4 0.00708
## 9      0 -0.00810     1 0.00177
## 10     0 -0.00611     1 0.00177
## # ... with 248 more rows

```

```
summary(pred)
```

```

##   predict      actual
## Min. :-0.02016  Min. :0.00000
## 1st Qu.: 0.02764 1st Qu.:0.00000
## Median : 0.03781 Median :0.00000
## Mean   : 0.03191 Mean  :0.03748
## 3rd Qu.: 0.04189 3rd Qu.:0.00000
## Max.   : 0.08967 Max.  :1.00000

```

This shows promise as the means are close (though the model tends to skew toward underprediction of fraud). This suggests that it is worthwhile to further improve the model by, e.g., performing a grid search for hyperparameter tuning, going back to the original features, selecting different engineered features, and perhaps exploring different algorithms.

Measuring Model Performance on Highly Unbalanced Data

Because of the severe bias toward non-fraud cases, we cannot use performance measures like accuracy or area under the curve (AUC), as they would give overly optimistic results based on the high percentage of correct classifications of the majority class. An alternative to AUC is to use the precision-recall curve or the sensitivity (recall)-specificity curve. To calculate and plot these metrics, we can use the ROCR package. There are different ways to calculate the area under a curve (e.g., see the PRROC package for details). In the next analysis, I create a simple line-integral function that calculates the area between every consecutive points-pair of x under the corresponding values of y .

```
library(ROCR)
```

```

## Loading required package: gplots

## 
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
## 
##     lowess

line_integral <- function(x, y) {
  dx <- diff(x)
  end <- length(y)
  my <- (y[1:(end - 1)] + y[2:end]) / 2
  sum(dx * my)
}

str(pred)

## 'data.frame': 587 obs. of 2 variables:
## $ predict: num 0.0206 0.0354 0.0444 0.0354 0.0378 ...
## $ actual : int 0 0 0 0 0 0 0 0 0 ...

```

```

summary(pred)

##      predict          actual
##  Min. :-0.02016   Min. :0.00000
##  1st Qu.: 0.02764  1st Qu.:0.00000
##  Median : 0.03781 Median :0.00000
##  Mean   : 0.03191 Mean  :0.03748
##  3rd Qu.: 0.04189  3rd Qu.:0.00000
##  Max.   : 0.08967  Max.  :1.00000

prediction_obj <- prediction(pred$predict, pred$actual)

par(mfrow = c(1, 2))
par(mar = c(5.1, 4.1, 4.1, 2.1))

# precision-recall curve
perf1 <- performance(prediction_obj, measure = "prec", x.measure = "rec")

x <- perf1@x.values[[1]]
y <- perf1@y.values[[1]]
y[1] <- 0

plot(perf1, main = paste("Area Under the\nPrecision-Recall Curve:
                           \n", round(abs(line_integral(x,y)), digits = 3)))

# sensitivity-specificity curve
perf2 <- performance(prediction_obj, measure = "sens", x.measure = "spec")

x <- perf2@x.values[[1]]
y <- perf2@y.values[[1]]
y[1] <- 0

plot(perf2, main = paste("Area Under the\nSensitivity-Specificity Curve:\n",
                           " ", round(abs(line_integral(x,y)), digits = 3)))

```

Precision is the proportion of test cases predicted to be a fraud that was indeed fraudulent (i.e., the true positive predictions), while recall or sensitivity is the proportion of fraud cases that were identified as fraud. And specificity is the proportion of non-fraud cases that are identified as non-fraud (Fig. 12).

The precision-recall curve tells us the relationship between correct fraud predictions and the proportion of fraud cases that were detected (e.g., if all or most fraud cases were identified, we also have many non-fraud cases predicted as fraud and vice versa). The sensitivity-specificity curve thus tells us the relationship between correctly identified classes of both labels (e.g., if we have 100% correctly classified fraud cases, we will have no correctly classified non-fraud cases and vice versa).

We can also look at this a little bit differently, by manually going through different prediction thresholds and calculating how many cases were correctly classified in the two classes:

```

thresholds <- seq(from = 0, to = 1, by = 0.1)
pred_thresholds <- data.frame(actual = pred$actual)

for (threshold in thresholds) {

  prediction <- ifelse(pred$predict > threshold, 1, 0)
  prediction_true <- ifelse(pred_thresholds$actual == prediction, TRUE, FALSE)
  pred_thresholds <- cbind(pred_thresholds, prediction_true)
}

```

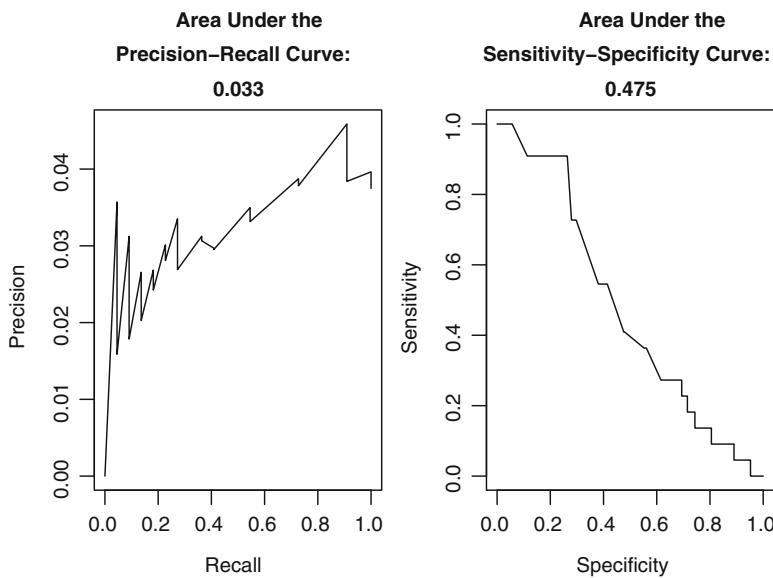


Fig. 12 Precision-recall and sensitivity-specificity curves in detecting fraud with SOX data

```

}

colnames(pred_thresholds)[-1] <- thresholds
pred_thresholds %>%
  gather(x, y, 2:ncol(pred_thresholds)) %>%
  group_by(actual, x, y) %>%
  summarise(n = n()) %>%
  ggplot(aes(x = as.numeric(x), y = n, color = actual)) +
  geom_vline(xintercept = 0.6, alpha = 0.5) +
  geom_line() +
  geom_point(alpha = 0.5) +
  theme_bw() +
  facet_wrap(actual ~ y, scales = "free", ncol = 2) +
  labs(x = "prediction threshold",
       y = "number of instances")

## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?

```

Figure 13's plots tell us that we can increase the number of correctly classified non-fraud cases without loosing correctly classified fraud cases when we increase the prediction threshold from the default 0.5–0.6:

```

pred %>%
  mutate(predict = ifelse(pred$predict > 0.6, 1, 0)) %>%
  group_by(actual, predict) %>%
  summarise(n = n()) %>%
  mutate(freq = n / sum(n))

## # A tibble: 2 x 4
## # Groups:   actual [2]
##   actual predict     n freq
##   <int>    <dbl> <int> <dbl>
## 1      0        0   565     1
## 2      1        0    22     1

```

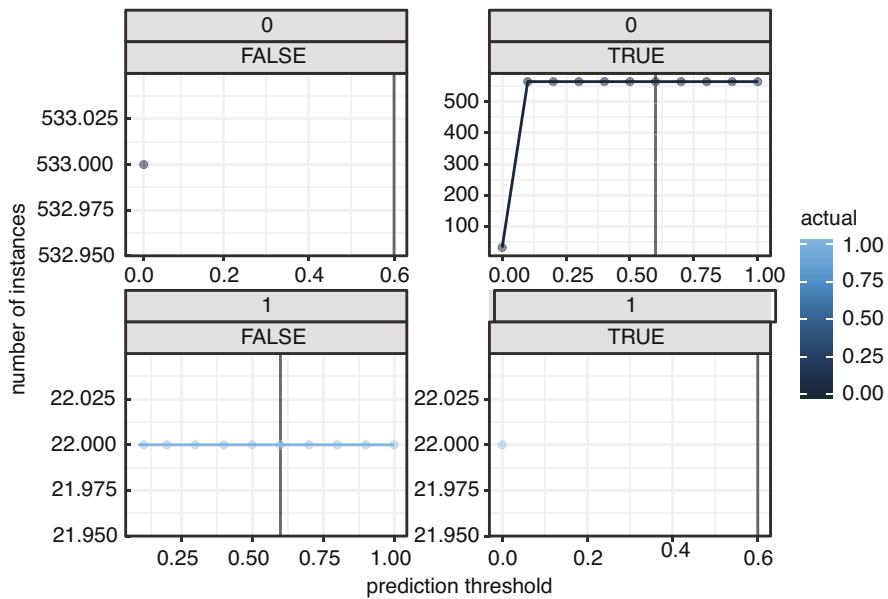


Fig. 13 Search for optimal prediction cutoff in predicting fraud with SOX data

Our final model does not greatly improve on the previous models, suggesting at this point that more information is needed for accurate prediction. Auditors, in particular, should have access to more extensive proprietary information than the publicly reported data in SOX. This can conceivably be merged with the SOX data, while using the previous analysis to provide additional specificity and sensitivity in credit card fraud detection.

Fama–French Risk Measures and Grid Search of Machine Learning Models to Augment Sarbanes–Oxley Information

Accounting and financial data are by their very nature non-linear and multicolinear, implying that cross-correlations and behavior of ensembles of variables can have a significant impact on predictions. Machine learning models can overcome such common problems with traditional statistics, because they can search for solutions in parameter spaces that are much larger. Interpretation can sometimes be more difficult, but we provide a set of “leverage” metrics for the influence of the input predictors that rank the importance of particular factors in the determination of credit card fraud. Though these cannot be interpreted in the directly causal manner in which predictors and regression statistics can be used, they do indicate where changes in the environment, management policy, and implementation will influence the occurrence of credit card fraud.

This research trained and cross-validated the following algorithms (in the following order): three pre-specified XGBoost GBM (Gradient Boosting Machine) models, a fixed grid of GLMs, a default Random Forest (DRF), five pre-specified H2O GBMs, a near-default Deep Neural Net, an Extremely Randomized Forest (XRT), a random grid of XGBoost GBMs, a random grid of H2O GBMs, and a random grid of Deep Neural Nets. From this base, two Stacked Ensemble models were trained, then a comprehensive grid search explored hyperparameters for each algorithm, optimizing the F1 classification statistic.

Classification performance is typically ranked on specificity-selectivity, or precision and recall performance. These are used in binary classification to study the output of some classifier (C). High precision means that an algorithm returned substantially more relevant results than irrelevant ones. High recall means that an algorithm returned most of the relevant results. These are related to the sensitivity-specificity framework. Sensitivity is the same as recall, and measures the proportion of actual positives that are correctly identified. Specificity measures the proportion of actual negatives that are correctly identified. All four are computed from the “confusion matrix”—a 2×2 matrix of accuracy in classification shown in Table 4.

Conditional on the classifier C , these are computed:

- $precision = \frac{TP}{TP+FP}|C.$
- $recall = sensitivity = \frac{TP}{TP+FN}|C.$
- $specificity = \frac{TN}{TP+FN}|C.$

The receiver operating characteristic (ROC) curve for either precision-recall or sensitivity-specificity summarizes the performance of a classifier over the range of C and sometimes plotted in a 2-dimensional graph. The integral of that curve between 0 and 1 is called the area under the curve (AUC) and summarizes performance of a model as a single number.

The $F1$ score (the harmonic mean of precision and sensitivity) is used to rank our classification models for whether or not credit card fraud has occurred, and is calculated as:

$$F1 = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

Table 5 provides the ranking of our models using the SOX-Privacy Clearinghouse (PCH) dataset. The best performing model (Table 5) based on the area under the sensitivity-specificity curve (AUC) was a gradient boosting machine listed on the first line of Table 6. This model had the following performance statistics.

For this model, the training confusion matrix is shown in Table 7 and validation in Table 8; the vertical identifiers represent actual outcomes (i.e., the response = 0/1), and horizontal identifiers represent predicted outcomes at the $F1$ -optimal threshold (refer to Table 4, which verbally describes each cell).

After validation, there is a slight trade-off of type I (0.008 dropping to 0.005) for type II error (0.13 rising to 0.20). The closeness of the training and validation matrices confirm that the model is robust. Furthermore, the overall statistics are very good, with sensitivity-specificity AUC of 99.3% and precision-recall AUC of 86% and Gini coefficient (i.e., AUC for the Lorenz curve) of 98.7%. All other statistics were excellent as well.

Table 9 presents the leverage of each predictor in the prediction of credit card fraud for the dataset. This is a list of the most important variables, rescaled and ranked to reflect different ranges of variables, and to make them comparable. Much of the predictive power is firm specific (as would be expected, since control weaknesses vary at the firm level). What is surprising is that the fee-based predictors are much more important in predicting credit card fraud than are the results of SOX testing (which demonstrate very little predictive power in this model). Unfortunately, the most important information seems to have been firm specific, with the next most important predictor having only half the influence of firm-specific effects. The most important non-firm specific predictors seem to be fees—this is to be an expected correlate, since companies with internal control problems are likely to need both more auditing as well as be more susceptible to security breaches. As generic predictors applicable to all firms in a set, audit fee information is more of a lagging indicator than a reliable predictor.

Fama–French Risk Factors

In asset pricing and portfolio management, the Fama–French models estimate stock returns from various risk factors (Fama and French 1993).

$$r = \alpha + R_f + \beta_{mkt} \cdot (R_m - R_f) + b_{smb} \cdot SMB + b_{hml} \cdot HML + \epsilon$$

Table 4 The confusion matrix

	Predicted condition positive	Predicted condition negative
Condition positive	(TP) True positive	(FN) False negative (type II error)
Condition negative	(FP) False positive (type I error)	(TN) True negative

Table 5 Best performing model statistics

Statistic	Value
AUC (sensitivity-specificity)	0.9934046
AUC (precision-recall)	0.8603881
R-squared	0.7071069
RMSE	0.1042045
MSE	0.0108586
LogLoss	0.0404258
Mean/class error	0.0346702
Gini	0.9868093

Table 6 Ranked machine learning models tested: SOX-PCH data

Model	AUC	AUC.PR	RMSE	MSE	log.Loss.
GBM_grid_1_AutoML_20200305_144328_model_1	0.993	0.860	0.104	0.011	0.040
GBM_4_AutoML_20200305_144328	0.989	0.778	0.117	0.014	0.049
GBM_3_AutoML_20200305_144328	0.988	0.753	0.119	0.014	0.052
GBM_2_AutoML_20200305_144328	0.988	0.801	0.109	0.012	0.045
XGBoost_grid_1_AutoML_20200305_144328_model_6	0.988	0.774	0.120	0.014	0.052
GBM_1_AutoML_20200305_144328	0.987	0.806	0.108	0.012	0.053
GBM_grid_1_AutoML_20200305_144328_model_4	0.977	0.579	0.148	0.022	0.077
XGBoost_grid_1_AutoML_20200305_144328_model_3	0.975	0.530	0.162	0.026	0.094
GBM_grid_1_AutoML_20200305_144328_model_3	0.973	0.815	0.117	0.014	0.053
DRF_1_AutoML_20200305_144328	0.961	0.731	0.126	0.016	0.070
GBM_5_AutoML_20200305_144328	0.956	0.653	0.137	0.019	0.066
StackedEnsemble_AllModels_AutoML_20200305_144328	0.953	0.781	0.125	0.016	0.068
StackedEnsemble_BestOfFamily_AutoML_20200305_144328	0.948	0.677	0.113	0.013	0.075
DeepLearning_grid_3_AutoML_20200305_144328_model_1	0.947	0.404	0.171	0.029	0.202
XRT_1_AutoML_20200305_144328	0.925	0.631	0.155	0.024	0.166
GBM_grid_1_AutoML_20200305_144328_model_2	0.914	0.443	0.186	0.035	0.141
DeepLearning_grid_1_AutoML_20200305_144328_model_1	0.909	0.438	0.174	0.030	0.213
DeepLearning_grid_2_AutoML_20200305_144328_model_1	0.903	0.442	0.165	0.027	0.220
XGBoost_3_AutoML_20200305_144328	0.891	0.208	0.183	0.034	0.129
XGBoost_grid_1_AutoML_20200305_144328_model_4	0.866	0.147	0.187	0.035	0.139
GLM_1_AutoML_20200305_144328	0.814	0.291	0.175	0.031	0.122
XGBoost_grid_1_AutoML_20200305_144328_model_2	0.741	0.078	0.191	0.037	0.154
XGBoost_1_AutoML_20200305_144328	0.703	0.066	0.192	0.037	0.159
XGBoost_grid_1_AutoML_20200305_144328_model_1	0.680	0.055	0.192	0.037	0.162
DeepLearning_1_AutoML_20200305_144328	0.673	0.143	0.186	0.034	0.153
XGBoost_2_AutoML_20200305_144328	0.621	0.050	0.192	0.037	0.162
XGBoost_grid_1_AutoML_20200305_144328_model_5	0.568	0.041	0.195	0.038	0.171

Table 7 Best performing model:
confusion matrix (train)

	0	1	Error	Rate
0	371	3	0.008021	0.0080214
1	2	13	0.133333	0.1333333
Totals	373	16	0.012853	0.0128535

Table 8 Best performing model:
confusion matrix (validate)

	0	1	Error	Rate
0	372	2	0.005348	0.0053476
1	3	12	0.200000	0.2000000
Totals	375	14	0.012853	0.0128535

- r is the portfolio's expected rate of return (**RET**);
- α is a market effect (**alpha**);
- R_f is the risk-free return rate;
- R_m is the return of the market portfolio. (coeff: **b_mkt**);
- *SMB* is “Small [market capitalization] Minus Big” (coeff: **b_smb**);
- *HML* is “High [book-to-market ratio] Minus Low” (coeff: **b_hml**);
- *ivol* is inferred volatility—volatility that can be inferred from the market price using calculations similar to an options pricing model and solving for the volatility input;
- *tvol* is theoretical volatility, inferred from calculations similar to options pricing using the Heston Model with a root-mean-square formula;
- ϵ is an error term with some presumed distribution (**exret**);
- R^2 measures the fit of the regression (**R2**).

Table 9 Variable leverage, best performing model on SOX-PCH data

Variable	Relative.importance	Scaled.importance	Proportion
Firm Specific (ticker)	38.886452	1.000000	0.317913
AUDIT RELATED_FEES	20.477877	0.526607	0.167415
TAX_FEES	17.072939	0.439046	0.139579
NON_AUDIT_FEES	15.740197	0.404773	0.128683
AUDIT_FEES	10.672934	0.274464	0.087256
BENEFITS_FEES	8.042397	0.206817	0.065750
Time-Specific (year)	4.881341	0.125528	0.039907
AUDITOR AGREES	1.726176	0.044390	0.014112
COMBINED_IC_OP	1.254117	0.032251	0.010253
IS_EFFECTIVE	1.041654	0.026787	0.008516
SIG_DEFICIENCY	0.829417	0.021329	0.006781
IC_OP_TYPE	0.770081	0.019803	0.006296
MATERIAL_WEAKNESS	0.645898	0.016610	0.005280
IC_IS_EFFECTIVE	0.276247	0.007104	0.002258

The Fama–French model used three predictors (with an additional two being added in later work)—(1) market risk, (2) the outperformance of small versus big companies, and (3) the outperformance of high book/market versus small book/market companies (Petkova 2006). Fama and French started with the observation that two classes of stocks have tended to do better than the market as a whole: (1) small caps and (2) stocks with a high book-to-market ratio (B/P, customarily called value stocks, contrasted with growth stocks). They then added two factors to CAPM to reflect a portfolio’s exposure to these two classes (Fama and French 1993). These factors are calculated with combinations of portfolios composed of ranked stocks and available historical market data.

The Fama–French three-factor model explains over 90% of the diversified portfolios returns, compared with the average 70% given by the CAPM (within a sample). They find positive returns from small size as well as value factors, high book-to-market ratio and related ratios. Examining β and size, they find that higher returns, small size, and higher β are all correlated. They then test returns for β , controlling for size, and find no relationship. Assuming stocks are first partitioned by size, the predictive power of β then disappears. They discuss whether β can be saved and the Sharpe–Lintner–Black model resuscitated by mistakes in their analysis, and find it unlikely (Fama and French 1992). In 2015, Fama and French extended the model, adding profitability and investment, and further suggesting volatility measures. These models tend to do well in predicting US returns, but fall short in other markets. For the purpose of this research, which is US-centric, the Fama–French metrics are relevant, and we adopted the three-factor model with two synthetic volatility metrics—inferred and theoretical volatility.

As we did previously, we trained and cross-validated algorithms in the following order: three pre-specified XGBoost GBMs (Gradient Boosting Machine) models, a fixed grid of GLMs, a default Random Forest (DRF), five pre-specified H2O GBMs, a near-default Deep Neural Net, an Extremely Randomized Forest (XRT), a random grid of XGBoost GBMs, a random grid of H2O GBMs, and a random grid of Deep Neural Nets. From this base, two Stacked Ensemble models were trained. A comprehensive grid search explored hyperparameters for each algorithm, optimizing the F_1 classification statistic.

Classification performance was ranked on F_1 score (the harmonic mean of precision and sensitivity) for whether or not credit card fraud has occurred is calculated as:

$$F_1 = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

Table 10 provides the ranking of our models using the previous dataset with added Fama–French risk factors. Because data was limited to publicly traded firms for the Fama–French risk factors, total firms in the database were reduced to around half the number available for the full SOX-Privacy Clearinghouse dataset.

Table 11 presents the performance of the best model with the added Fama–French risk factors. Prediction is perfect (100%) in the “trained” confusion matrix (Table 12), and makes only one mistake (a false negative) after validation (Table 13). The closeness of the training and validation matrices confirms that the model is robust and an improvement on the model without the Fama–French risk factors. Furthermore, the overall statistics are excellent, with sensitivity-specificity AUC of 99.8% (slightly improved), precision-recall AUC of 76% (slightly worse), and Gini coefficient (i.e., AUC for the Lorenz curve) of 99.7% (slightly improved). All other statistics were slightly improved.

Table 10 Machine learning models tested: SOX-PCH + Fama–French risk factors

Model	AUC	AUC.PR	RMSE	MSE	log.Loss.
GBM_grid_1_AutoML_20200305_140929_model_1	0.999	0.763	0.091	0.008	0.029
GBM_1_AutoML_20200305_140929	0.994	0.506	0.084	0.007	0.045
GBM_grid_1_AutoML_20200305_140929_model_2	0.994	0.706	0.162	0.026	0.098
GBM_2_AutoML_20200305_140929	0.994	0.675	0.111	0.012	0.042
GBM_4_AutoML_20200305_140929	0.994	0.675	0.121	0.015	0.046
XGBoost_grid_1_AutoML_20200305_140929_model_5	0.993	0.694	0.103	0.011	0.045
GBM_3_AutoML_20200305_140929	0.991	0.685	0.124	0.015	0.049
XGBoost_grid_1_AutoML_20200305_140929_model_6	0.991	0.602	0.134	0.018	0.065
XGBoost_grid_1_AutoML_20200305_140929_model_2	0.991	0.649	0.129	0.017	0.069
GBM_grid_1_AutoML_20200305_140929_model_3	0.990	0.640	0.108	0.012	0.048
XRT_1_AutoML_20200305_140929	0.987	0.664	0.106	0.011	0.046
GBM_grid_1_AutoML_20200305_140929_model_4	0.982	0.555	0.133	0.018	0.069
XGBoost_3_AutoML_20200305_140929	0.980	0.581	0.167	0.028	0.107
DRF_1_AutoML_20200305_140929	0.962	0.629	0.105	0.011	0.053
XGBoost_grid_1_AutoML_20200305_140929_model_1	0.956	0.395	0.174	0.030	0.125
GLM_1_AutoML_20200305_140929	0.952	0.489	0.134	0.018	0.074
XGBoost_grid_1_AutoML_20200305_140929_model_3	0.950	0.532	0.176	0.031	0.128
DeepLearning_grid_1_AutoML_20200305_140929_model_1	0.949	0.622	0.156	0.024	0.136
XGBoost_1_AutoML_20200305_140929	0.929	0.342	0.181	0.033	0.145
XGBoost_grid_1_AutoML_20200305_140929_model_7	0.923	0.571	0.179	0.032	0.138
StackedEnsemble_AllModels_AutoML_20200305_140929	0.855	0.608	0.117	0.014	0.065
StackedEnsemble_BestOfFamily_AutoML_20200305_140929	0.851	0.535	0.133	0.018	0.096
DeepLearning_grid_2_AutoML_20200305_140929_model_1	0.821	0.534	0.143	0.021	0.397
DeepLearning_grid_3_AutoML_20200305_140929_model_1	0.818	0.514	0.117	0.014	0.309
DeepLearning_1_AutoML_20200305_140929	0.713	0.060	0.174	0.030	0.139
XGBoost_grid_1_AutoML_20200305_140929_model_4	0.456	0.025	0.257	0.066	0.288
XGBoost_2_AutoML_20200305_140929	0.421	0.022	0.182	0.033	0.150

Table 11 Performance statistics of best model

Statistic	Value
AUC (sensitivity-specificity)	0.9985816
AUC (precision-recall)	0.7633333
R-squared	0.7477957
RMSE	0.0913309
MSE	0.0083413
LogLoss	0.0293261
Mean/class error	0.0035461
Gini	0.9971631

Table 12 Best model: confusion matrix (train)

	0	1	Error	Rate
0	141	0	0	0
1	0	5	0	0
Totals	141	5	0	0

Table 13 Best model: confusion matrix (validate)

	0	1	Error	Rate
0	140	1	0.007092	0.0070922
1	0	5	0.000000	0.0000000
Totals	140	6	0.006849	0.0068493

Table 14 Variable importance for best performing ML model: SOX-PCH + Fama–French risk factors

Variable	Relative_importance	Scaled_importance	Percentage
b_hml	11.0175772	1.0000000	0.3133020
Time-specific (year)	4.6752753	0.4243470	0.1329487
R2	4.5779834	0.4155163	0.1301821
NON_AUDIT_FEES	2.6473022	0.2402799	0.0752802
AUDIT_FEES	1.7185558	0.1559831	0.0488698
OTHER_FEES	1.6758081	0.1521031	0.0476542
alpha	1.3958076	0.1266892	0.0396920
b_mkt	1.2839173	0.1165335	0.0365102
b_smb	1.2726192	0.1155081	0.0361889
AUDIT RELATED FEES	1.1642095	0.1056684	0.0331061
TAX_FEES	0.9182994	0.0833486	0.0261133
exret	0.8094989	0.0734734	0.0230194
AUDITOR AGREES	0.6674294	0.0605786	0.0189794
tvol	0.4893094	0.0444117	0.0139143
ivol	0.4715100	0.0427962	0.0134081
COMBINED_IC_OP	0.2007296	0.0182190	0.0057081
IC_OP_TYPE	0.1744345	0.0158324	0.0049603
IC_IS_EFFECTIVE	0.0057306	0.0005201	0.0001630
IS_EFFECTIVE	0.0000030	0.0000003	0.0000001
SIG_DEFICIENCY	0.0000000	0.0000000	0.0000000
MATERIAL_WEAKNESS	0.0000000	0.0000000	0.0000000

Table 14 ranks the most important predictors of credit card fraud, rescaled, and ranked to make them comparable. The story is much more positive here. The most influential information comes from a generic predictor (i.e., can be used in predicting card fraud across companies). One Fama–French risk factor holds the most leverage in the prediction of credit card fraud—the coefficient β_{hml} for *HML*, the “High [book-to-market ratio] Minus Low.” Recall that Fama and French suggested that two classes of stocks have tended to do better than the market as a whole: (1) small caps and (2) stocks with a high book-to-market ratio—customarily called value stocks. The coefficient β_{hml} is positively correlated with the occurrence of security breaches involving credit card fraud. The second most influential variable is the year of reporting, with a scaled importance of 0.42. It is insightful to look at the distribution of security breaches involving credit card fraud (response variable = “1”) versus years when the fraud was not detected (response variable = “0”) where there were clusters of high fraud years after the crash of 2008 (Fig. 14). Reporting year has only 42% of the influence of β_{hml} ; while the Fama–French 3-factor model R^2 has about the same influence at 41%. After that comes non-audit fees (which are likely to be information systems-oriented) at 25%; and a number of predictors with ~10% of the influence of β_{hml} —these may or may not be robust in the long-run, as their influence is overwhelmed by the most influential predictors.

The high rank of the Fama–French 3-factor model’s β_{hml} and R^2 provide confidence that firms can use them effectively as generic predictors, applicable to all firms in a set, in analytical review and audit planning to assess potential credit card system control weaknesses that they may encounter.

Perhaps less assuring is the lackluster leverage, in models both with and without Fama–French risk factors, of the four major SOX specific opinions rendered by auditors and the signed SOX report:

- information systems are effective;
- internal control over information systems is effective;
- material weaknesses exist;
- significant deficiencies exist.

In both models, these four key audit and management opinions had only 1–2% of the leverage of the most significant predictor.

Variable leverage can be perceived as a type of correlation, but non-linear and more completely using all of the information in the dataset. Leverage may also tend to be model specific, since machine learning models learn more completely from the data inputs and consider non-linearities and covariances that linear models ignore. Our analysis shows that these variables offer considerable information for the prediction of fraud—the AUC of both models were ~100%; both models hold the potential to improve the monitoring of credit card security inside firms, during the audit and at the institutional and banking level.

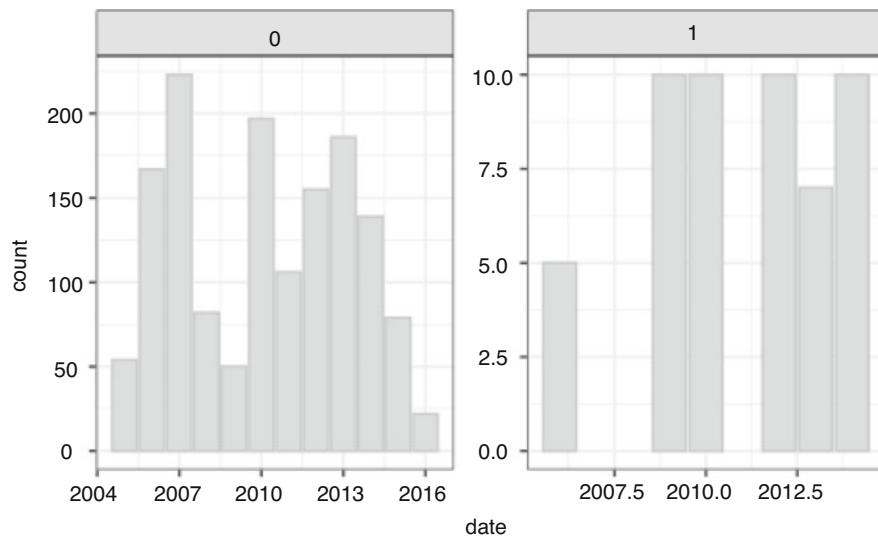


Fig. 14 Yearly distribution of card fraud security breaches (0 = no fraud; 1 = fraud)

Final Thoughts on Sarbanes–Oxley Reports

Because our SOX and credit card datasets are highly unbalanced, with the majority of observations not experiencing a security breach, our model can safely err on the side of deciding there is no breach (something management may be likely to do as well). In this situation, the level of type I errors (an incorrect prediction that there is no breach when there is one) is 6.8% for portable device breaches, 4% for credit card breaches, and 7.8% for insider breaches. If one thinks of 5% significance (or p-value) as being a criterion of good decision-making, then these predictions are not all that inaccurate by commonly applied standards.

But our predicament in this area highlights the need for more data. The amount of SOX data is substantial now, and approaching 15 years of reporting for public companies. A larger, more current database of breaches, though, is needed to improve the model. If we are to believe (Menn 2012), the size of our breach dataset should be 3–30 times as large—between 1000 and 10,000 reported breaches. A better funded, more diligent gathering of breach data (the current datasets breaches have been mostly self-reported) would dramatically improve the ability of machine learning models to predict future breaches. To explore this idea, an alternative test with a larger test dataset was created by splitting the training and test at 2010 rather than 2013, which placed 48% of the observations in the “test” set. Though this put more breach occurrences under performance measurement, the results were not significantly different. It is likely that accurate predictions of actual breach occurrence will require more observations.

We have case studies and qualitative research that suggest that strategies, approaches, and tools for systems breaches are constantly evolving. For example, after the Shadow Brokers in 2016 auctioned hacking tools from the NSA’s Equation Group, several security breaches were alleged to have occurred, which would not have been possible without the NSA tools (Burgess 2017). Thus, as time progresses, there may be a steady degradation of the ability of models to accurately predict future breaches. We do not know to what extent this may have impacted the current study. Nonetheless, it will be important to test this hypothesis as well in future research with more extensive data collection.

This chapter reviewed the requirements, implementation, and some research on Sarbanes–Oxley reporting. Much of the SOX audit work is done contemporaneously with the auditors’ work during interim tests of transactions. The chapter also looked into the more interesting potential of SOX opinions to provide useful information to auditors, clients, stockholders, and management to use SOX metrics to identify, predict and combat real-world security problems and losses. The models presented here derive from more extensive work published in Westland (2019), and the reader is directed to that article for additional insights. Additionally, it should be noted that TensorFlow added support for Neural Structured Learning (NLS) that trains neural networks with structured signals, and is particularly suited to unbalanced datasets. Unfortunately, at the time of this writing, NLS has yet to be incorporated in the R-Keras API for TensorFlow. Future applications of deep-learning to auditing, though, should seriously consider NLS to extend the methods used here in predicting the occurrence of errors, omissions, and duplications in the transaction population. The code chunks presented here can be modified and extended to benefit corporate security and improve controls in general. It is my hope that the ideas presented in this chapter are merely a start in extracting full value from SOX reporting.

References

- Ashbaugh-Skaife, Hollis, Daniel W. Collins, William R. Kinney Jr, and Ryan LaFond. 2008. The Effect of Sox Internal Control Deficiencies and Their Remediation on Accrual Quality. *The Accounting Review* 83(1):217–250.
- Bedard, Jean C., and Lynford Graham. 2011. Detection and Severity Classifications of Sarbanes-Oxley Section 404 Internal Control Deficiencies. *The Accounting Review* 86(3):825–855.
- Bedard, Jean C., Rani Hoitash, and Udi Hoitash. 2009. Evidence from the United States on the Effect of Auditor Involvement in Assessing Internal Control over Financial Reporting. *International Journal of Auditing* 13(2):105–125. Wiley Online Library.
- Berger, Philip G., Feng Li, and MH Franco Wong. 2005. The Impact of Sarbanes-Oxley on Cross-Listed Companies. *Unpublished Paper*.
- Bratton, William W. 2003. Enron, Sarbanes-Oxley and Accounting: Rules Versus Principles Versus Rents. *Vill. L. Rev.* 48:1023. HeinOnline.
- Burgess, Matt. 2017. Hacking the Hackers: Everything You Need to Know About Shadow Brokers' Attack on the NSA. *Wired*. <https://www.wired.co.uk/article/nsa-hacking-tools-stolen-hackers>.
- Coates, I.V., and C. John. 2007. The Goals and Promise of the Sarbanes-Oxley Act. *Journal of Economic Perspectives* 21(1):91–116.
- Drawbaugh, K., and D. Aubin. 2012. A decade on, is Sarbanes-Oxley working. *Reuters*, July 12
- Engel, Ellen, Rachel M. Hayes, and Xue Wang. 2007. The Sarbanes–Oxley Act and Firms’ Going-Private Decisions. *Journal of Accounting and Economics* 44(1–2):116–145. Elsevier.
- de Mesa Graziano, Cheryl, and William M. Sinnett. “How low can Sarbanes-Oxley Section 404 costs go?”, *Financial Executive* 23.6 (2007): 61–64.
- Fama, Eugene F., and Kenneth R. French. 1992. The Cross-Section of Expected Stock Returns. *The Journal of Finance* 47(2):427–465.
- _____. 1993. Common risk factors in the returns on stocks and bonds. *Journal of financial economics* 33(1):3–56.
- Feng, Mei, Chan Li, and Sarah McVay. 2009. Internal Control and Management Guidance. *Journal of Accounting and Economics* 48(2–3):190–209. Elsevier.
- Feng, Mei, Chan Li, Sarah E. McVay, and Hollis Skaife. 2014. Does Ineffective Internal Control over Financial Reporting Affect a Firm’s Operations? Evidence from Firms’ Inventory Management. *The Accounting Review* 90(2):529–557. American Accounting Association.
- Ge, W., A. Koester, and S. McVay. 2016. The Costs and Benefits of Section 404 (B) Exemption: Evidence from Small Firms’ Internal Control Disclosures. *Available at SSRN*.
- Hoitash, Udi, Rani Hoitash, and Jean C. Bedard. 2009. Corporate Governance and Internal Control over Financial Reporting: A Comparison of Regulatory Regimes. *The Accounting Review* 84(3):839–867.
- Kang, Qiang, Qiao Liu, and Rong Qi. 2010. The Sarbanes-Oxley Act and Corporate Investment: A Structural Assessment. *Journal of Financial Economics* 96(2):291–305. Elsevier.
- Lewis, James, and Stewart Baker. 2013. *The Economic Impact of Cybercrime and Cyber Espionage*. McAfee.
- Lin, Shu, Mina Pizzini, Mark Vargus, and Indranil R. Bardhan. 2011. The Role of the Internal Audit Function in the Disclosure of Material Weaknesses. *The Accounting Review* 86(1):287–323.
- Menn, J. (2012) Bank security: Thieves down the line, *Financial Times*, January 2, 2012, <https://www.ft.com/content/951f0efe-2d60-11e1-b985-00144feabdc0>, downloaded August 24, 2020
- Petkova, Ralitsa. 2006. Do the Fama–French Factors Proxy for Innovations in Predictive Variables? *The Journal of Finance* 61(2):581–612. Wiley Online Library.
- Rice, Sarah C., and David P. Weber. 2012. How Effective is Internal Control Reporting Under Sox 404? Determinants of the (Non-) Disclosure of Existing Material Weaknesses. *Journal of Accounting Research* 50(3):811–843. Wiley Online Library.
- Rice, Sarah C., David P. Weber, and Biyu Wu. 2014. Does Sox 404 have Teeth? Consequences of the Failure to Report Existing Internal Control Weaknesses. *The Accounting Review* 90(3):1169–1200. American Accounting Association.
- Romano, Roberta. 2004. The Sarbanes-Oxley Act and the Making of Quack Corporate Governance. *Yale LJ* 114:1521. HeinOnline.
- Westland, J. Christopher. 2019. The Information Content of Sarbanes-Oxley in Predicting Security Breaches. *Computers & Security*. <https://doi.org/10.1016/j.cose.2019.101687>.
- Westland, J. Christopher. 2020. Predicting Credit Card Fraud with Sarbanes-Oxley Assessments and Fama-French Risk Factors. *Intelligent Systems in Accounting, Finance and Management*. <https://doi.org/10.1002/isaf.1472>. Wiley Online Library.
- Whalen, D., M. Cheffers, and O. Usvyatsky. 2012. Financial Restatements: A Twelve Year Comparison. *Audit Analytics*.

Blockchains, Cybercrime, and Forensics



J. Christopher Westland

Blockchains for Securing Transactions

The concept of *blockchain* originated with the pseudonymous Satoshi Nakamoto (whose identity is hotly debated to this day) in a whitepaper “Bitcoin: A Peer-to-Peer Electronic Cash System” in 2008 (Nakamoto et al. 2008). This paper proposed a decentralized electronic currency (Bitcoin) that runs on the blockchain. The idea spread from this point and can be applied to many other areas—decentralized, incorruptible database of monetary transactions, contracts, un-hackable voting machines, and so forth. This section presents the technology underlying a blockchain through a simple sequence of R code chunks.

Assume your goal is to store some data in a secure way. You first store the data in a container—called a block. In the case of Bitcoin, each block contains several financial transactions. When there are new transactions (or there is new data), a new block will be created and will be combined with previous blocks to form a chain—the blockchain.

The “Block”

```
block_example <- list(index = 1,  
                      timestamp = "2020-01-01 00.00 CST",  
                      data = "a transaction",  
                      previous_hash = 0,  
                      proof = 9,  
                      new_hash = NULL)
```

Before you can start building the blockchain—chaining different containers with data together—you need two core concepts: *hashing* and *proof-of-work-algorithms*.

Hashing

A *hash function* is a term used to loosely refer to checksums, check digits, fingerprints, lossy compression, randomization functions, error-correcting codes, and ciphers. True hash functions are used to index a hash table for scatter storage addressing, and the term has come to apply in security as well. Although storage and security concepts overlap to some extent, each one has its own uses and requirements and is designed and optimized differently. It might be more correctly termed a fingerprinting algorithm—a procedure that maps an arbitrarily large data item (such as a computer file) to a much shorter bit string (its fingerprint) that uniquely identifies the original data.

A *hash value* (or just *hash*) generated by the hash function helps to ensure the integrity of a block by connecting it to the other blocks in the chain. A hash function takes input data and returns a unique, encrypted output. The hash guarantees the validity of a blockchain block by using a hashing algorithm (e.g., SHA256). The R *digest* package implements a function “*digest()*” for the creation of hash values (digests) of arbitrary R objects, using the “md5,” “sha-1,” “sha-256,” “crc32,”

“xxhash,” “murmurhash,” and “spookyhash” algorithms. This allows easy comparison of R language objects, and is applied here to secure the blockchain blocks.

```
library("digest")

# generate a unique hash value 'fingerprinting' the author's name
digest("j_christopher_westland", "sha256")

## [1] "1548f0f4ce0a02a03d8cde6034f89b5c88964217c84e813e241b0fc9f1cda608"
```

In the case of blockchains, the input data is the block (index, timestamp, data) to the hash function, but also the hash of the previous block. This means you can only calculate the valid hash if you know the hash of the previous block, which is created using the hash of the block before and so on and so forth—thus *blockchain*. This provides you with an immutable, sequential chain of blocks. If you altered one block afterwards you would have to calculate all the hashes for the sequential blocks again. Here is an implementation.

```
#Function that creates a hashed "block"
hash_block <- function(block) {
  block$new_hash <- digest(c(block$index,
                               block$timestamp,
                               block$data,
                               block$previous_hash), "sha256")
  return(block)
}
```

Proof-of-Work (PoW)

Where large amounts of information are stored in the blockchains, you need many blocks. But cryptocurrencies, in particular, need to control the number of new blocks created—otherwise, crypto-coins would lose their value were there an infinite number of coins created. Bitcoin, for example, is designed so that the maximum total amount of bitcoins that can ever exist is 21 million.

Blockchain works as a cryptocurrency because there is a valuable asset underlying each bitcoin that is mined—the work required to generate the hash. For cryptocurrencies like Bitcoin, this could be a problem as the time to create a new block should be more or less constant (around 10 min in the case of Bitcoin). Therefore the difficulty of PoW has to be adjusted continuously to account for increasing computational speed and varying numbers of miners in the network at a given time.

As a consequence, we add a so-called Proof-of-Work (PoW) algorithm, which controls the difficulty of creating a new block. “Proof” means that the computer has performed a certain amount of work. In practice the goal is to create an item that is hard to create but easy to verify. The following code chunk uses the following “task” as a PoW: find the next number that is divisible by 99 and divisible by the proof-number of the last block.

```
### Simple Proof of Work Alogrithm
proof_of_work <- function(last_proof) {

  proof <- last_proof + 1

  # Increment the proof number until a number is found that is divisible by
  # 99 and by the proof of the previous block
  while (! (proof %% 99 == 0 & proof %% last_proof == 0 )) {
    proof <- proof + 1
  }
  return(proof)
}
```

For blockchains like Bitcoin or Ethereum, the job of creating new blocks is done by so-called miners. When a new block has to be created, a computational problem is sent out to the network. The miner who solves the PoW problem first creates

the new block and is rewarded in bitcoins (this is the way new Bitcoins are actually created). This “lottery” of finding the new correct proof ensures that the power of creating new blocks is decentralized. When a new block is mined, it is sent out to everybody so that every node in the network has a copy of the latest blockchain. The idea that the longest blockchain in the network (the one which “the most work was put into”) is the valid version of the blockchain is called “decentralized consensus.”

In the case of Bitcoin, the PoW problem involves the problem of finding numbers that generate hashes with a certain amount of leading zeros. To account for increasing computational speed and varying numbers of miners in the network, the difficulty of the PoW can be adjusted to hold the time to create a new block constant at around 10 min.

Addin gTransactions (New Blocks) to the Blockchain

A blockchain is a growing list of transaction records (blocks) that are linked using cryptography (the hash). Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data (generally represented as a Merkle tree). By design, a blockchain is resistant to modification of the data, and provides an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way. In cryptocurrency and other applications, it is important that creating new blocks (such as the creation of a new Bitcoins) be expensive, so that the currency has some value. The following code chunk shows how blocks are chained together using hashes and how the cost of creating new blocks is being regulated by PoWs.

```
#A function that takes the previous block and normally some data
#(in our case the data is a string indicating which block in the chain it is)
gen_new_block <- function(previous_block){

  #Proof-of-Work
  new_proof <- proof_of_work(previous_block$proof)

  #Create new Block
  new_block <- list(index = previous_block$index + 1,
                    timestamp = Sys.time(),
                    data = paste0("this is block ", previous_block$index +1),
                    previous_hash = previous_block$new_hash,
                    proof = new_proof)

  #Hash the new Block
  new_block_hashed <- hash_block(new_block)

  return(new_block_hashed)
}
```

The first block in a blockchain is the so-called *genesis* block, containing no data and arbitrary values for proof and previous hash (as there is no previous block).

```
# Define Genesis Block (index 1 and arbitrary previous hash)
block_genesis <- list(index = 1,
                      timestamp = Sys.time(),
                      data = "Genesis Block",
                      previous_hash = "0",
                      proof = 1)
```

As each new transaction is processed, the blockchain grows by adding new blocks.

```
## Build the blockchain, starting with the Genesis
block and then add a few blocks using a loop.
```

```

blockchain <- list(block_genesis)
previous_block <- blockchain[[1]]

proof <- NULL
# How many blocks should we add to the chain after the genesis block
num_of_blocks_to_add <- 10

# Add blocks to the chain
for (i in 1: num_of_blocks_to_add) {
  block_to_add <- gen_new_block(previous_block)
  blockchain[i+1] <- list(block_to_add)
  previous_block <- block_to_add

  print(cat(paste0("Block ", block_to_add$index, " has been added", "\n",
    "\t", "Proof of Work: ", block_to_add$proof, "\n",
    "\t", "Hash Value: ", block_to_add$new_hash)))

  proof[i] <- block_to_add$proof
}

## Block 2 has been added
## Proof of Work: 99
## Hash Value: a4b7d41b8cc1d122d94b9ff013c74d76adbb94c3c2c5d44f92cfcd52f620a6c0
## NULL
## Block 3 has been added
## Proof of Work: 198
## Hash Value: 4a0dfa4ecea2453df38c8bbd4ec6d8c1f77fbefdc0664e5d0fad224d94fb06
## NULL
## Block 4 has been added
## Proof of Work: 396
## Hash Value: 847818c59adda3984f52adc8c7a970e2f0ffc38f3181f999a877581c56df1355
## NULL
## Block 5 has been added
## Proof of Work: 792
## Hash Value: 668443d7a1426fd56d90c6240dd84845a22773e54543d76911817141fbadbbe0
## NULL
## Block 6 has been added
## Proof of Work: 1584
## Hash Value: 34afb388202c30bb85ac2273560a682cda026fcc4869aac58696ec226335298
## NULL
## Block 7 has been added
## Proof of Work: 3168
## Hash Value: 4d233f2b37b5fda06f1c017919c50de33ce1e193a66528660a529e4406b8bbc5
## NULL
## Block 8 has been added
## Proof of Work: 6336
## Hash Value: a394dd66a129b626ae55ac5dfca7c99a74f46567a89a14778f4b324880d201e0
## NULL
## Block 9 has been added
## Proof of Work: 12672
## Hash Value: ae65234c984f12d686627e972f6535271ff8c829a5dadb4848ab2ef8c36a016a
## NULL
## Block 10 has been added
## Proof of Work: 25344

```

```

## Hash Value: 259fb5ffebaaee4ceea96eeb2c9934265f8034e5fcddc8fd9d92e568898349d8
      NULL
## Block 11 has been added
## Proof of Work: 50688
## Hash Value: a9f642ea924f5e6b0350e654fdd1db87af65472f2dfeaca340a0ab94b9e7a15d
      NULL

```

This is how one block in the chain looks.

```

blockchain[[10]]

## $index
## [1] 10
##
## $timestamp
## [1] "2019-12-15 11:38:19 MST"
##
## $data
## [1] "this is block 10"
##
## $previous_hash
## [1] "ae65234c984f12d686627e972f6535271ff8c829a5dad848ab2ef8c36a016a"
##
## $proof
## [1] 25344
##
## $new_hash
## [1] "259fb5ffebaaee4ceea96eeb2c9934265f8034e5fcddc8fd9d92e568898349d8"

```

Note that the cost of generating new blocks (the “Proof of Work”) grows exponentially (Fig 1).

```

library(tidyverse)

proof_work <- data.frame(1:length(proof), proof)
colnames(proof_work)[1] <- "cycle"

ggplot(proof_work, aes(cycle, proof)) +
  geom_line() +
  xlab("Block Number") +
  ylab("Cost of New Block Generation")

```

That is it! Reading vendor and industry hype surrounding blockchain would lead one to believe that the technology is much more complex. Indeed, implementing markets and commercial applications can become involved, but the underlying concepts are easy to implement and simple to understand.

Cybercrime and Forensics

The importance of crime by computer has motivated the passage of a number of laws addressing various parameters of use and abuse of computer technology, a.k.a. cybercrime.

1. The Electronic Communications Privacy Act of 1986 (amendment to Title III of the Omnibus Crime Control and Safe Streets Act of 1968) extended government restrictions on wiretaps from telephone calls to include transmissions of electronic data by computer, primarily designed to prevent unauthorized government access to private electronic communications.

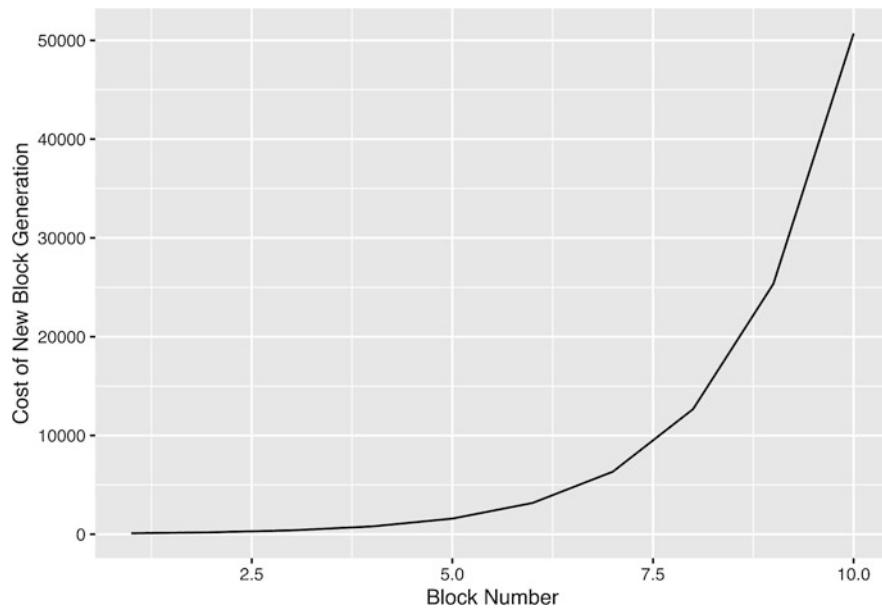


Fig. 1 Cost (Proof of Work) for each additional block added to blockchain

2. The Computer Fraud and Abuse Act (amendment to the Crime Control Act of 1984) clarified a number of the provisions in the original Act and also criminalized additional computer-related acts, addressing the distribution of malicious code and denial-of-service attacks, and included a provision criminalizing trafficking in passwords and similar items. The Act has been amended a number of times—in 1989, 1994, 1996, in 2001 by the USA PATRIOT Act, 2002, and in 2008 by the Identity Theft Enforcement and Restitution Act.
3. The Federal Information Security Management Act of 2002 recognized the importance of information security to the economic and national security interests of the USA. The act requires each federal agency to develop, document, and implement an agency-wide program to provide information security for the information and information systems that support the operations and assets of the agency, including those provided or managed by another agency, contractor, or other sources. It explicitly emphasizes a “risk-based policy for cost-effective security”.
4. The Cybersecurity Act of 2010 establishes greater collaboration between the public and the private sector on cybersecurity issues, especially those private entities that own infrastructures that are critical to national security interests. The most controversial parts of this bill grant the President the right to “order the limitation or shutdown of Internet traffic to and from any compromised Federal Government or United States critical infrastructure information system or network.”
5. International Cybercrime Reporting and Cooperation Act assures that the administration keeps Congress informed on information infrastructure, cybercrime, and end-user protection worldwide.

There are too many types of crime to enumerate, and criminals are constantly innovating, creating new classes of cybercrime. Generally, however, cybercrime may be divided into three categories: (1) crimes that target computers directly; (2) crimes facilitated by computer networks or devices; and (3) criminal business or large-scale activities that degrade the computing environment. Computer crime is consistently rewarding, with few penalties for criminals, and thus criminals are enticed to come up with new and innovative techniques every day. Indeed, there is even a name for computer-crime innovations—a zero-day attack exploits a previously unknown vulnerability in a computer application.

The following attempts to outline the major types of cybercrime of which auditors need to be aware of.

1. Crimes that primarily target computer networks or device.
 - (a) Computer viruses.
 - (b) Denial-of-service attacks.
 - (c) Malware (malicious code).
2. Crimes that use computer networks or devices to advance other ends.
 - (a) Cyberstalking.
 - (b) Fraud and identity theft.

- (c) Information warfare.
 - (d) Phishing scams.
3. Criminal business or large-scale activities that degrade the computing environment. These are sometimes difficult to differentiate from marketing, and indeed there is an ongoing debate over the ethics of many marketing approaches in an era of information technology.
- (a) Spam: the unsolicited sending of bulk email for commercial purposes.
 - (b) Fraud: computer fraud is any dishonest misrepresentation of fact intended to let another to do or refrain from doing something that causes loss.
 - (c) Obscene or offensive content: the content of websites and other electronic communications may be distasteful, obscene, or offensive for a variety of reasons. This is also controversial, as no two individuals have exactly the same taste. Nonetheless, there is wide-spread agreement that certain content, such as child pornography, should be prohibited from the Internet.
 - (d) Harassment: Whereas content may be offensive in a non-specific way, harassment directs obscenities and derogatory comments at a specific individual.
 - (e) Threats: although freedom of speech is protected by law in most democratic societies (in the US this is done by First Amendment), it does not include speech with “intent to harm or intimidate.”
 - (f) Markets in Illegal Products: marketing products on the Internet that would be illegal to trade physically is illegal. This might include drugs, pornography, national secrets, and so forth.
 - (g) Cyber terrorism: cyber terrorists use information technology to coerce a government or organization to advance their political or social objectives by launching a computer-based attack against computers, networks, and the information stored on them.
 - (h) Cyberextortion: cyberextortion is cyberterrorism directed to private firms, and typically intended for financial gain. Perpetrators typically use a distributed denial-of-service attack.

White-collar crimes, more specifically crimes involving flawed accounting, are financially motivated nonviolent crime committed by business and government professionals. Within criminology, it is defined as a crime committed by a person of respectability and high social status in the course of his occupation. Typical white-collar crimes include computer hacking, embezzlement through denial-of-service attacks, computer fraud, identity theft, bribery, Ponzi schemes, insider trading, embezzlement, cybercrime, copyright infringement, money laundering, identity theft, and forgery.

In the USA, sentences for white-collar crimes may include a combination of imprisonment, fines, restitution, community service, disgorgement, probation, or other alternative punishment. These punishments grew harsher after the Jeffrey Skilling and Enron Scandal, when the Sarbanes–Oxley Act of 2002 was passed defining new crimes and increasing the penalties for crimes such as mail and wire fraud. In other countries, such as China, white-collar criminals can be given the death penalty. Certain countries like Canada consider the relationship between the parties to be a significant feature in sentence when there is a breach of trust component involved.

White-collar crime, on average, pays well. As business moves away from physical assets, and toward information assets, the potential for gain steadily increases. This has fueled a growing epidemic of white-collar crime. The sentencing disparity in white-collar crime is a significant problem. It is estimated that less than 1% of white-collar criminals are ever incarcerated, and perhaps less than 10% are even detected (though such numbers are difficult to verify).

As Willie Sutton, the bank robber, observed when asked why he robbed banks “because that’s where the money is.” Sutton robbed banks and he was good at it. He made no bones about that. He usually packed a gun, either a pistol or a Thompson submachine gun “You can’t rob a bank on charm and personality.” Money is no longer held in physical form, but exists as bits and bytes in a growing array of remote servers and complex networks.

Computer forensics is a branch of forensic science pertaining to legal evidence found in computers and digital storage media. Computer forensics is also known as digital forensics. The goal of computer forensics is to explain the current state of a digital artifact. The term digital artifact can include a computer system, a storage medium (such as a hard disk or CD-ROM), an electronic document (e.g., an email message or JPEG image), or even a sequence of packets moving over a computer network. The explanation can be as straightforward as “what information is here?” and as detailed as “what is the sequence of events responsible for the present situation?” In legal cases, computer forensic techniques are frequently used to analyze computer systems belonging to defendants (in criminal cases) or litigants (in civil cases):

1. To recover data in the event of a hardware or software failure.
2. To analyze a computer system after a break-in, for example, to determine how the attacker gained access and what the attacker did.
3. To gather evidence against an employee that an organization wishes to terminate.

4. To gain information about how computer systems work for the purpose of debugging, performance optimization, or reverse-engineering.

Four principles dictate compliance with the need to maintain the integrity of digital evidence:

1. **Principle 1:** No action taken by law enforcement agencies or their agents should change data held on a computer storage media which may subsequently be relied upon in court.
2. **Principle 2:** In exceptional circumstances, where a person finds it necessary to access original data held on a computer on storage media, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions.
3. **Principle 3:** An audit trail or other record of all processes applied to computer-based electronic evidence should be created and preserved. An independent third party should be able to examine those processes and achieve the same result.
4. **Principle 4:** The person in charge of the investigation (the case officer) has overall responsibility for ensuring that the law and these principles are adhered to.

There are five basic steps to the computer forensics:

1. Preparation (of the investigator, not the data).
2. Collection (the data).
3. Examination.
4. Analysis.
5. Reporting.

Computer forensic software is used to secure evidence and generate reports for court should be validated. Digital evidence can be collected from computers, cell phones, digital cameras, hard drives, CD-ROM, USB memory devices, and so on. Non-obvious sources include settings of digital thermometers, black boxes inside automobiles, RFID tags, and web pages (which must be preserved as they are subject to change). Special care must be taken when handling computer evidence: most digital information is easily changed, and once changed it is usually impossible to detect that a change has taken place (or to revert the data back to its original state) unless other measures have been taken.

It is important, in building IT forensic cases, to maintain the chain of custody of evidence, documenting everything that has been done. In an investigation in which the owner of the digital evidence has not given consent to have his or her media examined (as in some criminal cases) special care must be taken to ensure that the forensic specialist has the legal authority to seize, copy, and examine the data.

The process of creating an exact duplicate of the original evidentiary media is called “imaging.” Using a standalone hard drive duplicator or software imaging tools such as EnCase or FTK Imager, the entire storage medium is completely duplicated. This is usually done at the sector level, making a bit-stream copy of every part of the user-accessible areas of the hard drive, which can physically store data, rather than duplicating the filesystem. The original drive media is then moved to secure storage to prevent tampering. During imaging of static data, a write protection device or application is normally used to prevent changes from being introduced to the evidentiary media during image acquisition. Hash verification is essential for evidence that will be presented in a courtroom.

There are two major competitors in the IT forensics marketspace—Guidance Software (EnCase) and AccessData (FTK). The companies generate revenue through sales, accreditation programs, and legal consulting services. They warrant the security of evidence gathered with their tools, and this creates barriers to entry for less established competitors. These auxiliary services are especially important, because the basic tasks of imaging and hashes can otherwise be accomplished with free or low-cost utilities.

AccessData Forensic Toolkit, or FTK, is a computer forensics software that scans a hard drive looking for evidence. It can, for example, locate deleted emails and scan a disk for text strings to use them as a password dictionary to crack encryption. The toolkit also includes a standalone disk imaging program called FTK Imager. It saves an image of a hard disk in one file or in segments that may be later on reconstructed. It calculates MD5 hash values and confirms the integrity of the data before closing the files. The result is a DD raw image.

Guidance Software’s EnCase is a suite of digital forensics products designed for forensic, cyber-security, and e-discovery use. The company also offers EnCase training and certification. EnCase contains tools for several areas of the digital forensic process: acquisition, analysis, and reporting. The software also includes a scripting facility called EnScript with various API’s for interacting with evidence.

No matter what law is invoked, the plaintiff and defendant need to build a legal case. This involves securing the evidence, which in the case of computer crimes is almost always some sort of computer file. That evidence, once secured, needs to

be tied to one or more aspects of the case that is argued in court. Because of the ephemeral character of data, the securing of the crime scene may involve specialized skills, and software designed for forensic work in order for the evidence to be admissible in court.

Forensic Analytics: Benford's Law

The first known reference to the Benford's Law was a two-page article in the American Journal of Mathematics in 1881 by Simon Newcomb. Newcomb explained that his discovery of the significant-digit law was motivated by an observation that the pages of a book of logarithms were dirtiest in the beginning and became progressively cleaner as one progressed through the book. A half century later, Frank Benford popularized a variation on Newcomb's distribution in his eponymous Benford's Law after spending years validating the relationship in datasets as different as atomic weights, baseball statistics, numerical data from Reader's Digest, and areas of rivers (Fig. 2).

Hal Varian (Varian 1972) suggested that Benford's Law could be used to detect possible fraud in lists of socio-economic data submitted in support of public planning decisions. Based on the plausible assumption that people who make up figures tend to distribute their digits fairly uniformly, a simple comparison of first-digit frequency distribution from the data with the expected distribution, according to Benford's law, ought to show up any anomalous results. In the USA, evidence based on Benford's law is legally admissible in criminal cases at the federal, state, and local levels. Benford analysis has subsequently been applied in a wide range of fields.

Most people have preconceived notions of randomness that often differ substantially from true randomness. The outcome of such preconceptions is systematic biases in the human generation of ostensibly "random" sequences. This is the justification behind Benford's analysis to infer fraud or other human manipulation of transaction datasets. An example of human clumsiness in assessing probabilities is the fact that, in a randomly selected group of 23 people, the probability is greater than half that at least two share a birthday—this comes as a surprise to many people. Or a more serious workplace example might involve "false-positives" in drug testing. Suppose that an employee is selected at random from a large population of which 1% are drug users; and that the drug test is 98% reliable (i.e., drug users test positive with probability 0.98 and those that do not use drugs test negative with probability 0.98). If the test result is positive, then the employee tested is more than twice as likely to be a nonuser rather than a drug user—again, most people will find this surprising.

Benford's Law concludes that the first digit in a large population of transactions (10,000 plus) will most often be a 1. Less frequently will the first digit be a 2; even less frequently a 3. In general, the probability the most significant digit of a number is equal to $d_1 \in [1, 2, \dots, 9]$ is $Pr(d_1) = \log_{10}(1 + \frac{1}{d_1})$

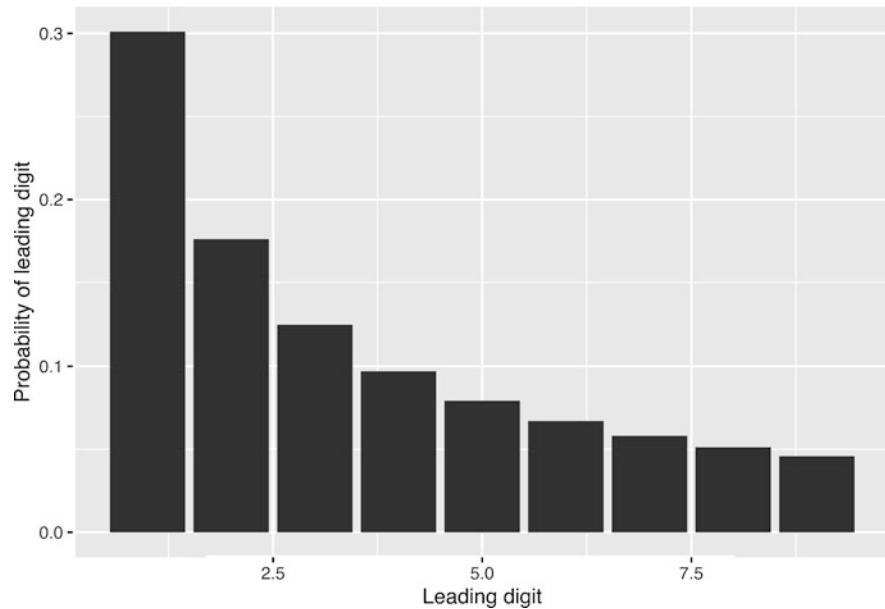


Fig. 2 Occurrence probabilities predicted by Benford's Law

```
library(tidyverse)
prd1 <- data.frame("digit"=1:9, "Pr_digit"=log10(1+(1/{1:9})))
ggplot(prd1, aes(digit, Pr_digit)) +
  geom_bar(stat="identity") +
  xlab("Leading digit") +
  ylab("Probability of leading digit")
```

An analysis of the frequency distribution of the first or second digits can detect abnormal patterns in the data and may identify possible fraud. An even more focused test can be used to examine the frequency distribution of the first two digits, the first three digits, and so forth. Some audit software programs can be used to determine the frequency distribution for the first digits, first two digits, and second digits.

This “law” has been found to apply to a wide variety of datasets, including electricity bills, street addresses, stock prices, population numbers, death rates, lengths of rivers, physical and mathematical constants, and processes described by power laws (which are very common in nature). It tends to be most accurate when values are distributed across multiple orders of magnitude. But not all data will have distributions as predicted by Benford’s Law. Sometimes there is a valid rationale for certain numbers occurring more frequently than expected. For example, if a company sends a large amount of correspondence via courier, and the cost is a standard rate (\$6.12) for sending a package of under one pound, then the first digit (6) or the first two digits (61) may occur more often than predicted by Benford’s Law.

Benford’s law can only be applied to data that are distributed across multiple orders of magnitude. Generally, if there is any cutoff which excludes a portion of the underlying data above a maximum value or below a minimum value, then the law will not apply. Thus, real-world distributions that span several orders of magnitude rather smoothly (e.g., populations of settlements, provided that there is no lower limit) are likely to satisfy Benford’s law to a very good approximation. On the other hand, a distribution covering only one or two orders of magnitude (e.g., heights of human adults, or IQ scores) is unlikely to satisfy Benford’s law (Fig. 3).

The following code chunks apply Benford’s Law analysis to the “expenditures” file generated in the last chapter of this book (Fig. 4).

```
library(tidyverse)
library(benford.analysis)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

# Read the employee expenditures file

expenditures <-
  read.csv(system.file("extdata", "expenditures.csv", package =
    "auditanalytics", mustWork = TRUE)),
  na.strings="0", stringsAsFactors=FALSE)

hist(expenditures$amount)

lead_digit <- extract.digits(expenditures$amount, number.of.digits=1)
lead_digit <- lead_digit[,2]

hist(lead_digit, breaks=9)

ben <- benford(expenditures$amount, number.of.digits=2)

plot(ben)
```

The results of this analysis reveal that the digits 49 were in the data more often than expected (Fig. 5).

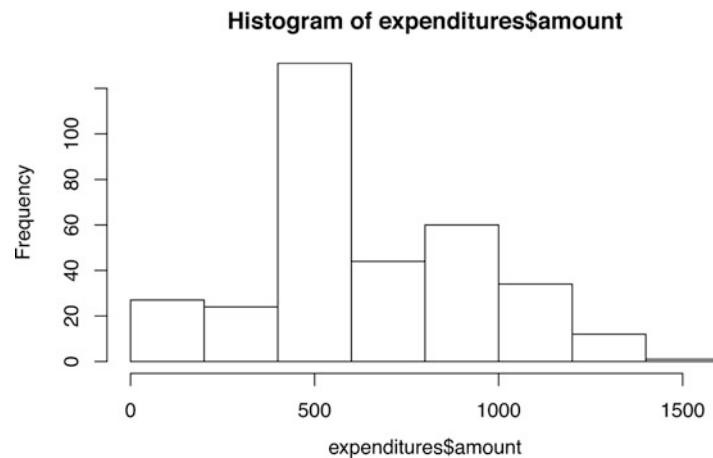


Fig. 3 Benford analysis profiles for the expenditures transaction file

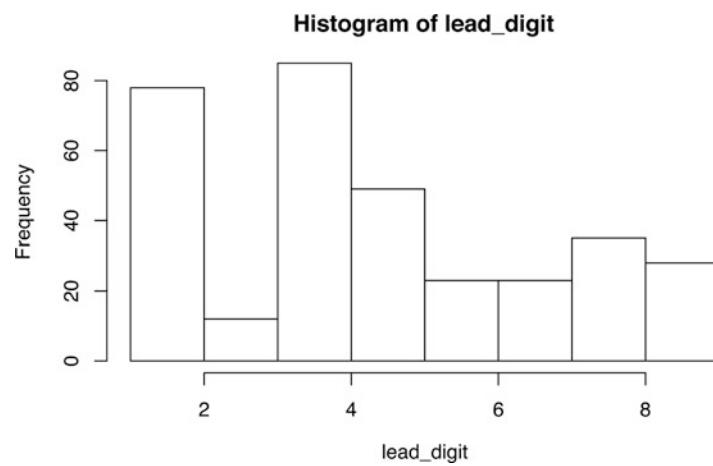


Fig. 4 Benford analysis profiles for the expenditures transaction file

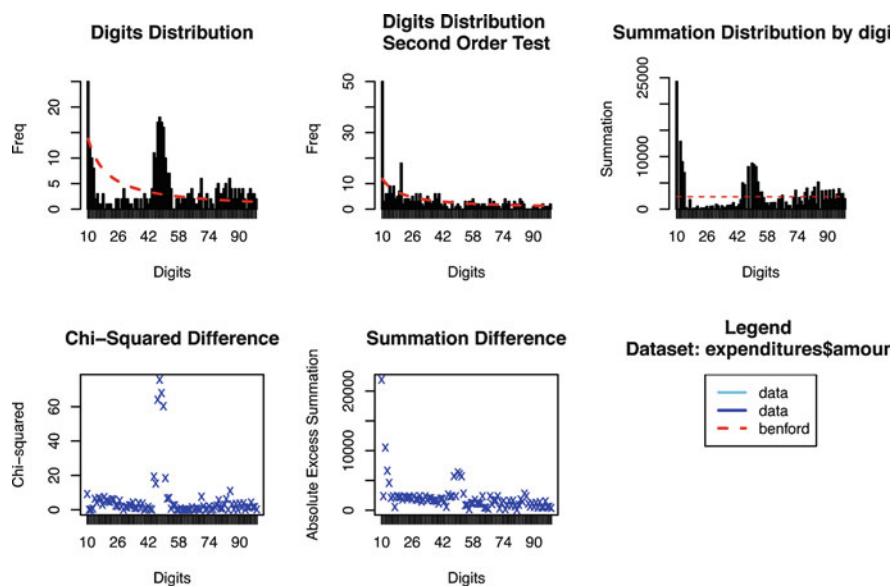


Fig. 5 Benford analysis profiles for the expenditures transaction file

```

library(tidyverse)
library(benford.analysis)

lead_digit <- extract.digits(expenditures$amount, number.of.digits=2)
employee_check <- cbind(expenditures,lead_digit)

employee_check[employee_check$data.digits ==49,] %>%
  arrange(employee_no) %>%
  select(employee_no, date, amount)

##   employee_no      date    amount
## 1      Emp0003 2019-05-25 495.0000
## 2      Emp0004 2019-05-19 491.0000
## 3      Emp0004 2019-05-24 499.0000
## 4      Emp0004 2019-07-18 496.0000
## 5      Emp0005 2019-09-01 499.0000
## 6      Emp0005 2019-11-23 499.8130
## 7      Emp0006 2019-05-17 498.0000
## 8      Emp0006 2019-05-31 495.0000
## 9      Emp0006 2019-07-27 498.0000
## 10     Emp0007 2019-05-13 495.0000
## 11     Emp0007 2019-06-15 497.0000
## 12     Emp0007 2019-08-28 496.0000
## 13     Emp0009 2019-07-14 491.0000
## 14     Emp0009 2019-08-21 493.0000
## 15     Emp0013 2019-07-31 496.0000
## 16     Emp0014 2019-06-11 492.0000
## 17     Emp0015 2019-04-24 497.9936

```

Classifying on the employee for all contracts with 49 as the first two digits determined several employees who submitted an excessive number of expense reports for \$499 (perhaps to avoid regulations required purchases over \$500 to have a supervisor's signature). The auditor could then use this information to initiate a further investigation of potential abuse of corporate control systems.

References

- Nakamoto, Satoshi, et al. 2008. *Bitcoin: A Peer-to-Peer Electronic Cash System*. White Paper.
 Varian, Hal R. 1972. Benfords Law. *The American Statistician* 26 (3): 65–66. American Statistical Association: 1429 Duke Street, Alexandria, VA 22314.

Special Engagements: Forecasts and Valuation



Special Engagements for Assurance and Valuation

Auditors have an intimate understanding of their client's finances. Because of that, they are likely to be called upon to assess the value of a new or existing investment. This chapter reviews the factors that go into performing a successful assurance and valuation assessment for a client.

Financial markets can be as challenging as consumer markets, but in different ways. Traditionally financial analysis has looked to the past, for precedents for a particular business, in the process of trying to predict future returns. Great strides have been made over the past two decades in the valuation and financing of new ventures. Financial markets offer a growing and increasingly complex array of investments, sold automatically with trading latencies measured in milliseconds. The complexity of methods required for the valuation of investments has grown even more rapidly. The time when ventures could be calculated on computer spreadsheets using published information and linear extrapolation has long since past, replaced by ever more mathematically sophisticated and computer-intensive approaches. One chapter is unfortunately insufficient for more than a very superficial review of these methods.

The Role of Valuations in the Market

Financial analysis and reporting play an essential role in a business' "reporting-control" cycle. A firm's or project's performance is realized on many dimensions—profit, competitive positioning, the strength of the workforce, and so forth. But only a few of these dimensions will capture the ongoing attention of management, and on even fewer of these dimensions, any particular stakeholder group possesses levers of control. For example, common stockholders of publicly traded firms are provided with audited financial reports each year, and quarterly unaudited financial reports. These are intended to provide them the information they need to for decisions about whether to buy-hold-divest from their inventory of the firm's stock. Stock sales and purchases are the only decisions shareholders can make unless they own enough shares to influence management. The specific financial "statistic" that fixates shareholder attention is the stock price, which is presumed to be correlated with earnings, assets, and other reported measures (though during the dot-com boom, curiously, this correlation was often negative). Time plays an important role, because shareholders' objectives are fixed on maximizing future returns—the percentage stock price increase per year.

All of this should suggest that, in practice, identifying the relevant financial information which will satisfy stakeholders' decision-making is not a simple task. Financial reporting directed just toward the simple task of buying and selling stock is extensive and costly. The more complex and nuanced decisions involved in managing a firm's operations require the sophistication of advanced managerial and budgetary accounting models—and ample quantities of subjective managerial "judgment."

Financial reporting is complex in a going concern; it grows even more complex in a new venture, because there are no ongoing operations, market outlets, or precedents to guide decision-makers. The need for financial information is even greater for new or untested ventures than for existing products and services.

Perhaps the most difficult task in selling any new or untested venture is providing a convincing story of what the new or untested venture is worth. Unfortunately, this task is not optional—if you expect someone to invest in your idea, they need to know where they will make their money; what will be their return on investment.

Hi-Tech, High Risk

Technology firms increasingly dominate the list of the largest firms. Even industries that have traditionally been asset-heavy—such as cars, aerospace, and chemicals—have steadily assumed larger and larger roles in firms. Several factors complicate an assessment of a new or untested venture’s value, especially where that value is heavily dependent on new and untested technology.

First, by definition, any new or untested venture is new, and thus there is no history of production or marketing to guide analysis; if there is, it will be incomplete and with very few data points.

Second, new or untested venture business can be several orders of magnitude riskier than a traditional business; the returns can commensurately be several orders of magnitude higher as well. Standard financial analysis methods tend to report a single “value” figure for a business or asset—e.g., saying that the business is worth \$10 million. But when ideas and businesses have never been tried before, high risk necessitates computing specific values for each potential outcome scenario—e.g., best case, worst case, the case if test marketing fails, and so forth.

Finally, the actual drivers of a new venture and its business model are too often things that never appear in the financial statement—e.g., the size of the network externality; the rapidity with which the technology’s performance improves; the appeal of the brand. For that reason alone, forward-thinking firms like eBay now dedicate significant portions of their financial statements to the non-financial measures which dictate their firm’s performance.

Assessment of a new venture’s value requires relevant data along with accurate models for assessing value. Since new or untested venture strategies are often driven by non-financial factors—e.g., time to market, design quality, and so forth—the acquisition of data that is not in the public domain (as, for example, the data in audited financial statements) can be costly and error-prone. Because, by definition, the business models of “new or untested ventures” are somehow new, they cannot rely on past relationships and dynamics in forecasting future earnings flows.

Assessing the value of an new or untested venture and its business model requires new sources of business intelligence and data mining to acquire relevant data. The assessment also requires forecasting methodologies with a level of sophistication not previously seen in the valuation of traditional investments. The following sections briefly survey how to identify and acquire relevant data for new or untested venture value assessments; it briefly covers the objectives and reporting requirements of forecasting methods required to support the innovator’s competition for venture funding. Finally, we discuss how the valuations demanded of new or untested ventures compare with existing valuation approaches and non-standard approaches such as brand-equity valuation and intangibles accounting.

A brief history of IBM’s risky gambles can provide greater insight into risk and return on new ventures. Arguably, the most radical financial ventures in history were not originally seen as much more than a technological oddity. On April 7, 1964, IBM revolutionized the industry with the announcement of its S/360 line of computers that became the mainstays of corporate accounting. IBM’s S/360 had been up to that time the largest private venture in American history, with \$5 billion spent on five new plants and an additional 60,000 employees. The subsequent decade was marked by a sea change in the structure of corporate accounting departments. Within a decade, corporate accounting departments at Sears, Ford, GM, DuPont, and other giants dwindled from huge “bullpens” accommodating hundreds of clerks to warrens of cubicle housing only 10–20% of their original number. Those who remained were dedicated to the care and feeding of IBM’s computerized accounting systems.

In many ways, IBM’s accounting revolution in the 1960s paralleled a similar gamble during 1930–1932, at the start of the Great Depression, to continue producing at full capacity and spend \$1 million on one of the first corporate research labs. In those days, IBM estimated that only 5% of business accounting functions were mechanized. IBM quickly introduced the Type 405 Alphabetic Accounting Machine, the 600 series punch card machines, and a system of machines designed for the banking industry. Continued production during the Depression assured that only IBM was capable of handling the massive volumes required of Social Security accounting that arose in the mid-1930s.

Strategy Drivers and Figures of Merit

Strategy drivers are the “levers of control” that management uses to direct operations. Since the actual strategy drivers may not be directly observable (e.g., customer satisfaction can be observed only indirectly), they are measured through figures of merit—observable, sometimes synthetic, measures that move in tandem with the strategy drivers. For example, when you buy a computer, you can seldom be assured that it will perform well for your particular computing tasks; but a synthetic

workload figure of merit, measuring the combined performance of CPU, memory, and compiler, can help you choose an appropriate price performance package.

Corporate Figures of Merit

When corporate executives are asked what the most important goal of their decision-making should be, they seldom say profits. Markets may demand fast product cycles; technological leadership requires new patents; stockholders are concerned about the share price. Profitability is low on the priorities of most executives. This alone should warn that value of a business model is not just about profits.

The business model—and there may be several of them for any given new or untested venture—is the basic unit of value analysis—one business model is given one set of valuations. For purposes of value assessment, we can ignore the larger strategic intricacies that motivate the choice of this particular business model, and consider only the value generation, the strategy drivers—the specific features that are actively managed in the business model—responsible for creating value. Then for purposes of valuation, the business model simplifies to:

$$\text{Value}_t = f_t(\text{strategy_drivers})$$

Both consumer markets and capital markets have their own distinct figures of merit that are used to assess the desirability and value of a particular investment proposition. You cannot manage what you cannot measure—and the success of a particular business model is ultimately measured by its performance on some “figure of merit.” The chosen figure of merit will reflect the objectives that got us into a particular business, which motivated the R&D for our new or untested venture to begin with. For example, the value of online auction software is likely to be judged on the size of the online community that it is able to develop; the value of a mail-order distribution channel might be measured on the percentage of the market that it is able to capture. Both of these measures of value are non-financial measures, but they ultimately have financial implications based on the cost–benefit structure of the business model. But management often considers such non-financial measures to be the primary objective of their investments.

The ROI Figure of Merit

Certain financial figures of merit are widely encountered, and we will review them here. The primary financial figure of merit is the annualized return on investment (ROI):

$$ROI = \frac{\text{value}_t - \text{value}_{t-1}}{\text{value}_{t-1}}$$

Several factors influence investors’ assessment of the hurdle rate—i.e., the minimum ROI required for them to be willing to invest money. Investors are a type of consumer, but rather than paying price p for something of value v received immediately (the product or service), they pay price p for a promise to return $(1 + i) \times p$ next year. If v is not high enough, the consumer does not contract to trade; if i is not high enough (i.e., higher than the hurdle rate), then the investor will not contract to invest. One significant difference between investors and consumers is the risk—the consumer’s transaction is over instantly, but many things can happen to the investor’s money in a year.

The Profit Figure of Merit

Central to ROI is the measurement of profit. Profit serves a crucial function in a free-enterprise economy—where firms are presumed to behave in a profit-maximizing fashion. High profits are the signal that consumers want more of the output of the industry. High profits provide the incentive for firms to expand output and for more firms to enter the industry in the long run. For a firm of above-average efficiency, profits represent the reward for greater efficiency. On the other hand, lower profits or losses are the signal that consumers want less of the commodity and/or that the production method is not efficient.

Profits provide the crucial signals for the reallocation of society's resources to reflect changes technology, liquidity, and in customers' tastes over time.

Accounting profit refers to the revenue of the firm minus the explicit or accounting costs of the firm. Explicit costs are the actual out-of-pocket expenditures of the firm to purchase or hire the inputs it requires in production. These expenditures include the wages to hire labor, interest on borrowed capital, rent on land and buildings, and the expenditures on raw materials. Economists extend this to an "economic profit" that equals the revenue of the firm minus its explicit costs and implicit costs. Implicit costs—including opportunity costs—refer to the value of the inputs owned and used by the firm in its own production processes.

The Sales Revenue Figure of Merit

Economist William Baumol suggested that, to satisfy stockholders, managers of modern corporations seek to maximize sales after an adequate rate of return has been earned. Baumol argued that a larger firm might feel more secure, may be able to get better deals in the purchase of inputs and lower rates in borrowing money, and may have a better image with consumers, employees, and suppliers. He also noted that some studies had shown a strong correlation between executives' salaries and sales, but not between sales and profits (this point is contentious, as other studies have found the opposite). As a consequence, he has suggested that at least some companies—notably those in oligopolistic markets—would seek sales growth over profitability.

Opportunity Costs

Opportunity costs include the salary that the entrepreneur could earn from working for someone else in a similar capacity (say, as the manager of another firm) and the return that the firm could earn from investing its capital and renting its land and other inputs to other firms. The inputs owned and used by the firm in its own production processes are not free to the firm, even though the firm can use them without any actual or explicit expenditures. Economic profit, rather than accounting profit, must be used in order to reach correct investment decisions.

Book value and accounting profit are less relevant to the valuation of assets than in the past—especially information assets. For that reason, analysts have adopted a varied set of alternatives; often, there is little agreement on which is best, and analysts are called upon to defend their particular approaches.

Additionally, the natural rate of profitability differs significantly between industries. Indeed, the greatest draw of a new or untested venture, despite its myriad headaches, is the profit that is significantly higher than average. Firms in such industries as steel, textiles, and railroads generally earn very low profits both absolutely and in relation to the profits of firms in the pharmaceutical, office equipment, and other high-technology industries. Several theories attempt to explain these differences.

Firms will vary in their assessment of an appropriate measure of profit for various reasons. Where operations are risky, firms require above-normal returns from economic profits to enter and remain in such fields as petroleum exploration with above-average risks. Similarly, the expected return on stocks has to be higher than on bonds because of the greater risk of the former.

Another economic theory suggests that profits arise as a result of friction or disturbances from long-run equilibrium. That is, in the long-run, perfectly competitive equilibrium, firms tend to earn only a normal return (adjusted for risk) or zero (economic) profit on their investment. At any time, however, firms are not likely to be in long-run equilibrium and may earn a profit or incur a loss. For example, at the time of the energy crisis in the early 1970s, firms producing insulating materials enjoyed a sharp increase in demand, which led to large profits. With the sharp decline in petroleum prices in the mid-1980s, many of these firms began to incur losses. When profits are made in an industry in the short run, more firms are attracted to the industry in the long run, and this tends to drive profits down to zero, and leads to firms earning only a normal return on investment.

In other cases, monopoly firms can restrict output and charge higher prices than under perfect competition, thereby earning a profit. Because of restricted entry into the industry, these firms can continue to earn profits even in the long run. Monopoly power may arise from the firm's owning and controlling the entire supply of a raw material required for the production of the commodity, from economies of large-scale production, from ownership of patents, or from government restrictions that prohibit competition.

The new or untested venture theory of profit postulates that (economic) profit is the reward for the introduction of a successful new or untested venture. The US patent system is, in fact, designed to protect the profits of a successful innovator—at least temporarily—in order to encourage the flow of new or untested ventures. Inevitably, as other firms imitate the new or untested venture, the profit of the innovator is reduced and eventually eliminated.

The managerial efficiency theory of profit relies on the observation that if the average firm (by definition) earns only a normal return on its investment in the long run, firms that are more efficient than the average would earn above-normal economic profits.

Valuation Models

The value of a new or untested venture lies in the future, and in the plans that management has for commercializing that new or untested venture. Consequently, the valuation of that new or untested venture demanded by potential investors and creditors must make a plausible case for its ability to generate future value.

The strategy model has two parts—a compelling narrative storyline about how, why, and when cash will be generated from the new or untested venture; and a forecasting model that ties this out to the numbers. The strategy model is an extension of the business model; it is focused only on the main drivers of value generation (whereas the business model is likely to be more complex and nuanced, dwelling on technological, social and market issues as well).

A valuation's strategy model consists of three submodels:

1. a behavioral model based on what we know about our prior behavior (or some similar firm's prior behavior);
2. a forecast model which lays out the future project scenario on a real options format for the new or untested venture project; and
3. a discounting model that articulates our attitude toward the time value of money. In this structure, the forecast model sets out our plans for new or untested venture R&D and market entry, the behavioral model assures that we make realistic assumptions about demand and costs, and the discounting model describes how aggressive we will be in demanding repayment of investments.

The last part—the discounting model—is often glossed over in traditional cash flow analysis; the rule of thumb is to use a risk-free rate. In venture capital, though, both risk and impatience may be embedded in the discount, and it is common to see hurdle rates of 40–70% in venture capital investments. There is also evidence that in practice, managers act as if they demand substantially higher discounts than the “risk-free rate.” Thus we require that assumptions concerning discounting be made explicit up front when dealing with risky and uncertain high technology and new or untested venture ventures (Diamond 1991; Conner et al. 1996).

The Behavioral (Historical) Model

Any pitch for funding of a new and innovative product or service will revolve around a single question—“Where do the investors make their money?” Venture capitalists, banks, shareholders, family, or whoever else is asked to put money into a new business are concerned about how money is to be generated in the future, and how much of it they will receive from their investment. The behavioral model tells the story that ties the salient operating and business characteristics of your new or untested venture to the generation of money in the future. If this linkage cannot be clearly articulated, it is unlikely that the new or untested venture will be of interest to investors. Advances in technology, especially the phenomenal expansion of the Internet over the past decade, have made possible a vastly expanded set of available strategies for new business. Just as important is the fact that “ideas” have become more important than “things” in driving value.

In building the behavioral model, our problem is to figure out how much our historical outcomes A from manipulating particular controllable strategy drivers can tell us about the potential for a given strategy B to generate value in the future. The strategy itself will be a given configuration of values of controllable strategy drivers, based on predictions of the exogenous strategy drivers. So assuming that these strategy drivers have yielded specific outcomes in the past, we can compute the set of coefficients and by setting particular values of the drivers predict the future generation of value. But the precision (i.e., the inverse of risk or variability) of the coefficients will only be as good as the information in the historical observations A. This is where Shannon Information comes in; it can predict how good is our dataset of historical observations (typically a set of financial statements augmented with a collection of non-financial data).

Shannon Information measures the amount of information provided by the behavioral model about value generated by the strategy model. This provides a general measure of the effectiveness of a strategy. The effectiveness that is ultimately “achievable” will vary between industries, contingent on the riskiness of those industries.

In general, measures of minimum achievable risk and information measures are inversely correlated.

Data: Transaction Stream Time Series

The R ecosystem contains a vast number of time-series standards and packages; this can make operations with time series confusing. Addressing the possibilities is beyond the scope of this book, though the auditor should be aware that there are many methods available for forecasting and analysis of the time series of events and transaction streams. The following code chunks provide examples of forecasting using the base R time-series class `ts` and packages that operate on that class of data. The package `tsbox` is brought in once to regularize (fill in missing dates) the time series with `ts_regular()` and `forecast` package for ARIMA forecasting. The following code chunk reads credit sales and collections data from the 2019 simulation data generated by code in the last chapter, and converts these to `ts` class for use in ARIMA forecasting of 2020 cash and collections transaction streams. These can then be differenced to forecast the change in accounts receivable balance (Figs. 1, 2, and 3).

```
library(lubridate)
library(tidyverse)
library(readr)
library(tsbox)

devtools::install_github("westland/auditanalytics")
library(auditanalytics)

credit_sales <-
  read_csv(system.file("extdata", "real_world_credit_sales.csv", package =
  "auditanalytics", mustWork = TRUE)), col_types = cols(X1 = col_skip())) %>%
  mutate(sales_amount = sales_count * sales_unit_price) %>%
  select(
    sales_date = invoice_date,
    sales_amount,
    collection_date,
    collection_amount
  )

ggplot(credit_sales) +
  geom_line(aes(x=sales_date, y=sales_amount, color="red")) +
  geom_line(aes(x=collection_date, y=collection_amount, color="blue"))

sales <-
  credit_sales[,1:2] %>%
  group_by(sales_date) %>%
  summarize(sales_amount = sum(sales_amount, na.rm = TRUE)) %>%
  arrange(sales_date)

sales_st <- sales[1,1]
sales_end <- sales[nrow(sales),1]

sales <- tsbox::ts_regular(sales)
## set dates with no sales to zero with tsbox package
sales$sales_amount [is.na(sales$sales_amount)] <- 0
```

```

## set dates with NA to zero
sales <- as.ts(sales)

collections <-
  credit_sales[,3:4] %>%
  group_by(collection_date) %>%
  summarize(collection_amount = sum(collection_amount, na.rm = TRUE)) %>%
  arrange(collection_date)

collections_st <- collections[1,1]
collections_end <- collections[nrow(sales),1]

collections <-
  ts_regular(collections)
## set dates with no sales to zero
collections$collection_amount[
  is.na(collections$collection_amount)] <- 0
## set dates with NA to zero
collections <- as.ts(collections)

```

How Much Information Is in a Dataset?

Accounting and most other financial information tend to be highly multicollinear, meaning that the data contained in the additional balance sheet or income statement accounts generally do not provide much new information about the business. This problem arises first because of the double-entry system, which replicates exactly the same journal information in two different accounts, one debit, and one credit. Second, it arises because different account entries will reflect the same transaction at different times in the transaction cycle. For example, a purchase will create journal entries for accounts payable, cash disbursements, and inventory increment and decrement—all of them for the same transaction but at different times in its life cycle. Thus the actual information content of a fixed-sized set of data items from a corporate financial dataset may

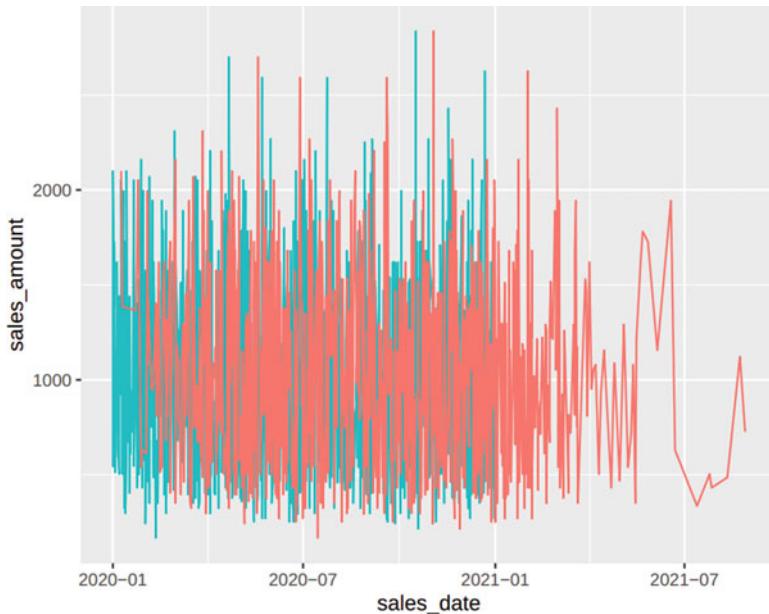


Fig. 1 Sales (red) and Collections (blue) Time Series

vary widely between firms or even between time periods. Being able to measure the actual amount of information in the data collected for forecasting provides guidance in collecting additional data needed for risk reduction, as well as assurance that existing datasets are adequate for the financial analysis at hand. Because of multicollinearity, financial analysts constantly question how much information is actually in their datasets. Yet measuring information content is elusive in practice.

In the mid-1930s, Claude Shannon established the field of information theory, counting among his many contributions some of the more successful early algorithms for casino gambling and computer chess as well as a formalization of the quantity of information in any collection of data. Based on the probability mass function of each source symbol to be communicated, the Shannon entropy H , in units of bits (per symbol), is given by

$$H = - \sum_i p_i \log_2(p_i) H = - \sum_i p_i \log_2(p_i)$$

where p_i is the probability of occurrence of the i^{th} possible value of the source symbol. Shannon information measures the “distance” between probability densities; there are other measures of the distance between two densities and these are also called “information” measures; e.g., Fisher, Wooters, Renyi, Tsallis, Hellinger, and Kullback–Leibler information measures, to name a few.

Forecast Models

Forecasting is the process of making predictions about the future based on past and present data and, most commonly, by analysis of trends. A commonplace example might be the estimation of some variable of interest at some specified future date. Prediction is a similar but more general term. Both might refer to formal statistical methods employing time series, cross-sectional or longitudinal data, or alternatively to less formal judgmental methods. Usage can differ between areas of application: for example, in hydrology, the terms “forecast” and “forecasting” are sometimes reserved for estimates of values at certain specific future times, while the term “prediction” is used for more general estimates, such as the number of times floods will occur over a long period.

Risk and uncertainty are central to forecasting and prediction; it is generally considered good practice to indicate the degree of uncertainty attaching to forecasts. In any case, the data must be up to date in order for the forecast to be as accurate as possible. In some cases, the data used to predict the variable of interest is itself forecast.

Physicist Nils Bohr, once quipped that “prediction is very difficult, especially if it is about the future.” Nearly all attempts at valuation need to rely on some guessing about the future—and because of that, it is very difficult. Even financial accounting, which is primarily concerned with tracking historical transactions, needs to make forecasts of the future life of assets in order to compute depreciation.

Good forecasting involves making educated guesses. Good forecasts satisfy four criteria:

1. Non-arbitrary: forecasting methodology defines a clear role for data, model drivers, assumptions, and hypotheses, which will be applied consistently from analysis to analysis. We can minimize these biases by using a standard formal methodology which encourages four traits of good research: (1) explains observed phenomena, and explains why; (2) is consistent with previously established knowledge; (3) is verifiable by other parties with access to the same data; and (4) stimulates further discussion, investigation, and revision as data become available.
2. Collective: forecasts should be easily understood by others, and their assumptions stated clearly enough to assure that others can replicate their conclusions.
3. Reliable: forecasts can be relied upon to make decisions. Reliability implies not only a specified degree of accuracy in financial reporting, but a clear idea of how accurate the reported numbers are through reporting of a dispersion statistic—for example, variance or standard deviation—to measure the reliability of the reported value.
4. Consistent and robust: forecasts will not change in the absence of fundamental information. Robust methods would limit and assure that different analysts using similar data would produce similar valuations, or where they were different, could clearly explain the assumptions which account for that difference.

This sort of valuation analysis is common in financial analysis—the representation is called a binomial variance lattice representation of a real options problem—and the problem is that of finding the value of management’s real option. In financial options analysis, there are multiple methodologies and approaches used to calculate an option’s value, including closed-form equations like the Black–Scholes model and its modifications, Monte Carlo path-dependent simulation methods, lattices, variance reduction and other numerical techniques, and partial-differential equations (of which the Black–Scholes

model is but one solution). Binomial lattices are easy to implement and easy to explain. In the limit, results obtained through the use of binomial lattices tend to approach those derived from closed-form solutions.

The expanding range of possible values between the confidence limits (the two lines extending into the future) is called a “value cone” and describes the risk profile of the particular business model that management designs to promote the new or untested venture. The value cone reflects the projection of the variability in our behavioral model into the future. It is, in general, expanding because our forecasting errors compound themselves moving further into the future. Any miscalculation of value generation next year will incorrectly forecast the path of development of the business model, and that misforecast will, in turn, create even greater forecast errors in the subsequent years. Forecast errors result because: (1) we have incomplete control over the drivers of value creation, and (2) we have imperfect knowledge about exogenous events that influence the success of our business.

The behavioral model, which encapsulates what is known about comparable firms, the industry’s “best practices” and prior years’ results, should be used to develop the forecasting model. In the code chunk below, it is assumed that the behavioral model suggests an ARIMA time-series model based on Holt’s linear method with additive errors. The three components ARIMA(p, d, q) are:

1. the AR order, typically chosen to minimize the Akaike information,
2. the degree of differencing. Differencing is a transformation applied to time-series data in order to make it stationary. A stationary time series’ properties do not depend on the time at which the series is observed, and
3. the moving average order.

```
library(forecast)

# fit an ARIMA (p, d, q) model
# order is the non-seasonal
#part of the ARIMA model:
#the three components (p, d, q) are
#the AR order, the degree of differencing, and the MA order.
# p is the order of the autoregressive part and
# q is the order of the moving average part.

# Holt's linear method with additive errors,
#or double exponential smoothing
fit <- arima(sales[,2], order=c(0,2,2))

# predictive accuracy, show the first few values
head(accuracy(fit))

## ME RMSE MAE MPE MAPE MASE ACF1
## Training set 55.4876 1695.92 1348.372 -Inf Inf 0.7076188 -0.001585686

# predict next year's observations
f_cast <- forecast(fit, 365)
plot(f_cast,
      main="ARIMA(0,2,2) Forecasts for Sales Transactions",
      xlab="Time Period (Days) starting 2019-1-1",
      ylab="Daily Transaction Total"
    )

fit <- arima(collections[,2], order=c(0,2,2))

# predictive accuracy, show the first few values
head(accuracy(fit))

## ME RMSE MAE MPE MAPE MASE ACF1
## Training set 1.613851 1395.177 979.6652 NaN Inf 0.7653655 0.04245381
```

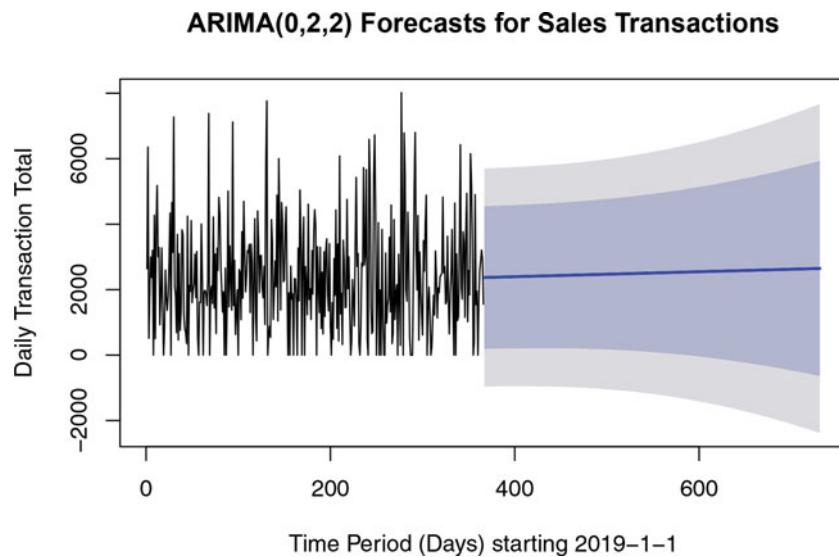


Fig. 2 Forecasts of future sales transaction streams

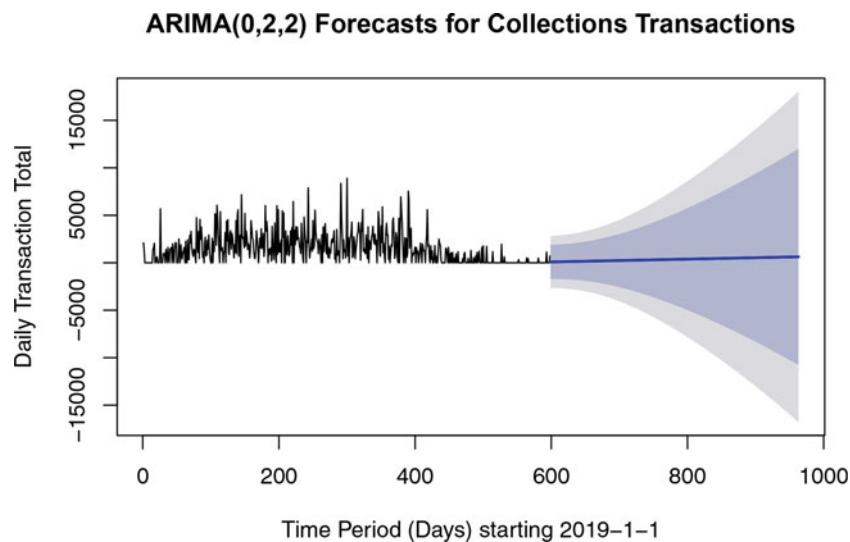


Fig. 3 Forecasts of future collections transaction streams

```
# predict next year's observations
f_cast <- forecast(fit, 365)
plot(f_cast,
  main="ARIMA(0,2,2) Forecasts for Collections Transactions",
  xlab="Time Period (Days) starting 2019-1-1",
  ylab="Daily Transaction Total"
)
```

```
forecast_sales <-
  arima(sales[,2], order=c(0,2,2)) %>%
  forecast(365)

upper_sales <-
  as.ts(forecast_sales$upper[,2] ) %>%
```

```

window(366, 365*2) %>%
sum()

lower_sales <-
as.ts(forecast_sales$lower[,2] ) %>%
window(366, 365*2) %>%
sum()

forecast_sales <-
as.ts(forecast_sales$mean ) %>%
window(366, 365*2) %>%
sum()

forecast_collections <-
arima(collections[,2], order=c(0,2,2)) %>%
forecast(365)

upper_collections <-
as.ts(forecast_collections$upper[,2] ) %>%
window(366, 365*2) %>%
sum()

lower_collections <-
as.ts(forecast_collections$lower[,2] ) %>%
window(366, 365*2) %>%
sum()

forecast_collections <-
as.ts(forecast_collections$mean ) %>%
window(366, 365*2) %>%
sum()

change_ar <- forecast_sales - forecast_collections
upper_ar <- upper_sales - upper_collections
lower_ar <- lower_sales - lower_collections

cat(
"\n\n The Expected 2020 Sales = $",
prettyNum(forecast_sales, big.mark=","))

##  

##  

## The Expected 2020 Sales = $ 912,686.1

cat(
"\n The Expected 2020 Collections = $",
prettyNum(forecast_collections, big.mark=","))

```

```

## 
## The Expected 2020 Collections = $ 23,726.63

cat(
  "\n The Expected 2020 Change in Accounts Receivable Balance = $",
  prettyNum(change_ar, big.mark=","))

## 
## The Expected 2020 Change in Accounts Receivable Balance = $ 888,959.5

cat(
  "\n 2020 AR Change : 95% Upper Bound = $",
  prettyNum(upper_ar, big.mark=","),
  " & 95% Lower Bound = $",
  prettyNum(lower_ar, big.mark=""))

## 
## 2020 AR Change : 95% Upper Bound = $ 1,815,473 & 95% Lower Bound = $ -37,553.94

```

The previous calculations estimate an expected increase in accounts receivable value from the beginning to the end of 2020 with a wide range of values within the confidence limits. Negative forecasts of collections are allowed, which is counterintuitive. Forecasted collections are understated, mistimed, or misforecasted in this rather crude analysis. The auditor would need to further investigate collection transaction flows to provide a more reliable forecast.

Discount Model

The prior forecast models generate estimates and confidence bounds for value-flows. In order to obtain a present value for the business (or more correctly, a pair of confidence limits for the present value), it is necessary to discount the future values back to the current value. This is done by summing all the future values generated in future periods t by a discount factor $\frac{1}{(1+r)^t}$ where r is the discount rate for each period. Discount rate r can be viewed either as:

1. a measure of management's impatience—how quickly they want to get their investment back; or
2. as an opportunity cost—a measure of potential return from other investment opportunities that were foregone to fund this new or untested venture business. From either perspective, deciding on the appropriate discount rate can be difficult. There is an extensive literature in finance that addresses the selection of discount rates. A detailed review is beyond the scope of this text, but we will briefly look at some of the more important considerations here.

Historical values may be found in the company's financial reports, or external benchmarks may be derived from treasury bill rates or other government debt. The historical discount rates simply provide a baseline for the choice of a discount rate or managerial target for the particular business model at hand. Discount rates for equity financing are a bit harder to assess, but convention uses the capital asset pricing model or its variant, the arbitrage pricing model, to compute yield based on historical correlation with the market.

Other approaches tend to try to quantify management's "impatience" and possibly also some measures of risk aversion, and then implant them into the discount rate applied to value a business model. Venture capitalists, for example, may apply an arbitrary rate perhaps as high as 40–60% annually for new startups. This high rate reflects both a desire for quick repayment on their investment, and the high risk of failure of startups. Evidence exists that managers may use similarly high rates for internal firm investments, though the reasons are different. In internal investment, it is common to periodically (e.g., annually during budget setting) rank proposed projects, and determine how much out of a fixed size fund might be allocated to the project. All projects considered need to offer returns on investment greater than some "hurdle rate" (often set as the return offered by external investments). But since only the highest return projects will be selected because of the fixed size of the budget, the return on projects actually implemented may be very high—again, above 40%. Another bias derives from upwardly mobile managers who may see themselves holding a particular divisional position for only a short period of time, and are looking for the maximum project payback in that short period. Thus they will be biased toward selecting projects that can generate a lot of near-term cash flow, as opposed to projects that have long-term strategic importance. In all of these cases, where new or untested ventures are developed into new businesses, it is likely that management will assign relatively high discount rates to assess their present value – discount rates that are often above 40% annually.

Terminal Dividend

Anyone used to traditional discounted cash flow analysis is familiar with the “terminal dividend” in that analysis setup. Discounted cash flow analysis usually assumes a forecasting horizon—a time after which forecasting is considered to be too error-prone or speculative to be useful. So speculation beyond this horizon is halted, and replaced with a terminal dividend that reflects not a cash payment, but rather an estimate of the market value. The terminal dividend was either: (1) the value of an annuity payable into perpetuity or (2) some price/earnings multiple. However computed, the horizon was typically far enough in the future that the terminal value presented only a small portion of total present value (though this assumption was widely abused during the dot-com boom). The approach was promoted in a popular finance text back in the 1960s, and called the Gordon growth model (Gordon 1962). The assumption of a terminal dividend is typically irrelevant in the analysis of new or untested ventures because of high risk, short product cycles, and demands for quick payback. All of these tend to raise managerial “impatience” and thus the required hurdle rate for investments in new or untested venture based business models. High discount rates reduce any value generated past the forecast horizon to negligible amounts. Consequently, there is no need to calculate a terminal dividend for most new or untested venture-based businesses.

Generating a Current Valuation

Once components are assembled for a valuation, the future cash flows may be forecasted and then discounted back to the “present value” at the selected discount rate. Use the standard net present value formula:

$$NPV = cf_0 + \sum_{k=1}^n \frac{cf_k}{(1+i)_k}$$

```
daily_discount <- .1/365 # 10% annual rate
NPV <- 0
for(i in 1:length(forecast_sales)) NPV <-
  NPV +
  as.numeric(forecast_sales[i] /
    ((daily_discount+1)^i))

cat(
  "\n\n The Jan. 1, 2020 NPV of the forecasted sales transactions for 2020 = $",
  prettyNum(NPV, big.mark=","))
## ## ## The Jan. 1, 2020 NPV of the forecasted sales transactions for 2020 = $ 912,436.1
```

Other Approaches to Valuation

Real Options

Real options assume that future performance will not result from a series of smooth future cash flows, as with, say, a traditional investment such as a bank’s mortgage contract. Rather cash flows will be contingent on future events or management decisions that may or may not occur and are uncertain in timing and amount. When properly constructed, real options models can help us make informed decisions about how much to invest, when to invest, and expected returns or losses on projects as a whole.

New or untested ventures distinguish themselves from more conventional investments in three ways: (1) technology development and marketing both need intense, adaptive management; (2) there is no history of investment performance which might yield insights in the value of the new or untested venture; and (3) the potential value of the new or untested venture depends on future events and managerial decisions which cannot be foreseen. This sets up impediments to commercializing an new or untested venture that can make resource planning and investment extremely difficult. New or untested ventures are inherently riskier, with added risk compensated for by high returns—research suggests, on average, about three to four times as great as traditional products and services, but significantly higher for “blockbusters.”

Successful innovators minimize their expenditures as they adapt their execution toward a successful launch of their product or service, much the same way that investors manage risk and expense via stock options. Options allow you to write a contract to execute a sale or purchase at a specific price within some future time frame. If the contract is profitable at a future date, then the option is executed; if not, it is not, and the investor bears a small out-of-pocket cost for writing the option contract.

Similarly, successful innovators rarely spend money when expenditures can be delayed or avoided. The problems facing innovators both inside and outside of the firm are similar—they are expected to maximize return on a highly risky investment, and tend to accomplish this by being stingy with cash. Corporations worry about the opportunity costs (the lost revenue from other more profitable uses of the investment money); banks, venture capitalists, and stock investors worry about the returns on their out-of-pocket investments.

With real options, both groups can realize the upside potential of an new or untested venture, while protecting themselves from the inherently higher risk of loss that new or untested ventures entail. Often the real options approach is imposed by corporations and investors themselves through tranches (i.e., portions of the total investment) that are made available after specific milestones—e.g., proof of concept—are reached. If an investment yields information that demonstrates the viability of the opportunity, you can proceed with further investments with more confidence than if the earlier investment had not been made. Because the insight and experience gained by investing in an option are captured only if you make the investment, learning from options investments can give you an edge over competitors, who will not privy to the knowledge because they have not experienced the same learning.

The behavioral model establishes the “levers of control” that are available to management to influence future firm value generation. But the future will be most strongly impelled by the actual plans that management implements, and the contingencies on which they pivot. In nearly every new product or change initiated by the new or untested venture, management is confronted with an array of scenarios, each demanding a unique response in structuring the business model.

Scenario Analysis and Decision Trees

The real options approach assumes that pivotal managerial decisions are made, and that an uncertain world subsequently unfolds, either rewarding or punishing that decision. The real options decision tree assumes that the manager makes a decision, and then finds out what happens. This fails when the manager finds out what happens (or at least guesses what might happen) prior to making a decision. In this case, scenario analysis and decision trees are better suited for the task of dealing with uncertainty. There are a number of easy-to-use programs that support scenario analysis and decision tree approaches to valuation.

Monte Carlo Simulations

Where decision trees become especially complex, perhaps growing into involved networks of options and decisions, simple decision tree software may meet its limits. The mathematics of mixing random variables in any degree approaching the real structure and transaction flows in business is daunting, if not impossible, to implement in practice. For this reason, analysts may choose to use Monte Carlo simulations to compute complex forecasts involving the interplay of multiple random variables. Unfortunately, each of these individual components can become a major source of error. Their combined effect can easily swamp the entire simulation.

There exists extensive empirical evidence on the statistical behavior of account balances, accounting errors, and accounting transactions that have shown that these are both right-skewed and highly kurtotic. Sums and differences of accounting errors are neither stationary nor well behaved. Interactions between skewed and kurtotic errors and accounts tend to multiply any uncertainties. Even though Monte Carlo simulation may, in theory, allow the solution of forecasting problems that are not mathematically tractable, they fail in practice because errors swamp out any valid forecasts. The forecast error

of a simulation is often orders of magnitude larger than the cash or income flows forecast, rendering these forecasts useless for valuation. In such situations, valuations become hyper-sensitive to new data, changing several hundreds of percent with even small changes in income or investment transactions.

Further Study

The preceding has merely sketched out the approach that must be taken to construct credible valuations of any business model that you might propose to commercialize an new or untested venture. Only the simplest calculations have been touched upon. The finance literature is extensive, well-developed, and mathematically complex. All of the approaches presented in this chapter can be extended and made much more accurate by the application of available techniques. The study of these techniques (and new ones appear in the academic journals every day) is a degree in itself, and is far beyond the scope of the current text. But the fundamental approaches, which are tailored to the character and availability of information useful for commercializing new or untested ventures, is going to be fundamentally as you have studied in this chapter.

References

- Conner, Kathleen R., and Coimbatore K. Prahalad. 1996. A Resource-Based Theory of the Firm: Knowledge Versus Opportunism. *Organization Science* 7 (5): 477–501. INFORMS.
- Diamond, Douglas W. 1991. Monitoring and Reputation: The Choice Between Bank Loans and Directly Placed Debt. *Journal of Political Economy* 99 (4): 689–721. The University of Chicago Press.
- Gordon, Myron J. 1962. *The Investment, Financing, and Valuation of the Corporation*. Homewood, IL: RD Irwin.

Simulated Transactions for Auditing Service Organizations



“Test Decks” and Their Progeny

In the early days of IBM 360 accounting systems (IBM’s advertising emphasized their 360-degree-full-circle-of-service) transactions were input on decks of 80-column Hollerith cards. Auditors would test automated controls by reading a deck of known transactions (i.e., a “test deck”) and comparing these with the accounting reports generated by a client’s automated system. In those days, this was termed “auditing *around* the computer.” Auditors contrasted this with “auditing *through* the computer” which followed transaction processing through the internal computer operations in order to verify that controls were functioning properly. It is hard for me to believe that any audit budget could survive auditors tracing a transaction’s journey through the code, yet this seemed to have been part of that period’s vernacular.

Though Hollerith cards are rarely seen these days outside a computer museum or eBay “vintage artifact” listing, the concept still exists of testing a client’s automated “black box” (i.e., a system where access to the code is limited by time, effort, or ownership). Simulations allow the auditor to compare known transaction inputs to computed output. Modern computing systems rely on code that is proprietary, with ownership by someone other than the client when systems are hosted on service bureaus and cloud platforms. Simulation of accounting transactions with known distributions and error occurrences provides useful tools for audit testing of black boxes. The current chapter details methods for creating simulated accounting transactions with known distribution and error characteristics.

Service Organization Audits

In the post-mainframe era, testing shared, proprietary third-party platforms of service organizations and cloud computing presents auditors with a significant challenge. The Statement on Standards for Attestation Engagements No. 18 (SSAE 18) is a set of auditing standards and guidance on using the standards, published by the Auditing Standards Board of the American Institute of Certified Public Accountants for redefining and updating how service companies report on compliance controls. It supersedes SSAE 16 and the Statement on Auditing Standards No. 70. Auditors may use transaction simulations for both compliance with SSAE 18 and with Sarbanes–Oxley’s requirement (section 404) to show effective internal controls covering financial reporting. SSAE 18 places the onus on the client and the service organization by requiring a formal “Third-Party Vendor Management Program” as well as a formal “Annual Risk Assessment” process. The risk assessment may also serve as an important resource for any future Risk Assessment Matrix (chapter “Analytical Review: Technical Analysis”) that the auditor may produce.

SSAE 18 auditing produces two Service Organization Control (SOC) reports. The SSAE 18 SOC 1 (SOC 1) Type 1 report focuses on a service provider’s processes and controls that could impact their client’s internal control over their financial reporting (ICFR). The examination helps ensure that both the system and personnel responsible for these controls at the third-party provider are doing their job in a manner that will not adversely affect their client’s ICFR. SOC 2 is a separate report that focuses on controls at a service provider relevant to security, availability, processing integrity, confidentiality, and privacy of a system. It ensures that your data is kept private and secure while in storage and in transit and that it is available for you to access at any time. The SOC 1 and SOC 2 reports come in two forms: Type I and Type II. Type I reports evaluate whether proper controls are in place at a specific point in time. Type II reports are done over a period of time to verify operational efficiency and effectiveness of the controls.

Accounting Cycles, Audit Tasks, and the Generation of Audit Data

Accounting cycles (also called transaction cycles or transaction processing subsystems) are the most basic way that we differentiate and cluster transaction processing in financial accounting operations. They serve an important role as an organizing principle for designing efficient and effective audit programs. Audit activities will be clustered around groups of related transactions, because errors, data location, and many other aspects of the audit are likely to be similarly clustered. An accounting cycle begins when accounting personnel create a transaction from a source document and ends with the completion of the financial reports and closing of temporary accounts in preparation for a new cycle. The actual implementation of accounting cycles varies with business type and company specific systems. In general, though, accountants recognize five archetypal accounting cycles. These are listed below along with the most important transactions in those cycles.

1. Sales (revenue) cycle: customer-facing activities that generate revenues
 - (a) sales orders
 - (b) cash receipts
2. Procurement (inventory, conversion, or production) cycle: activities to make available the company's products that generate cost of goods sold
 - (a) purchasing
 - (b) inventory replenishment
 - (c) production and cost accounting
3. Expenditure cycle: supporting activities that generate discretionary (fixed) costs
 - (a) accounts payable disbursements
 - (b) payroll/human resources
4. Treasury cycle: capital acquisition and repayment activities, and other activities that insure cash is available for uninterrupted operations
 - (a) borrowing/repayment
 - (b) issuing financial instruments
 - (c) dividends
 - (d) cash management
5. Fixed assets cycle: long-term investments and depreciation
 - (a) movable assets
 - (b) property
 - (c) land
 - (d) depreciation

Generation of Sales and Procurement Cycle Databases for A/B Testing of Audit Procedures

There are many situations where it is useful to be able to generate a simulated accounting dataset for audit tests as well as for examples and workouts for the classroom. I have constructed a generic dataset simulator here that assumes an e-commerce retailer buying and selling SKUs (stock keeping units, i.e., identifiers for the type of inventory).

The remainder of this chapter presents a set of R code for generating a set of simulated transactions from the sales and procurement cycles, along with their relevant financial reports (Fig. 1). This code is just one example of a particular subset of some hypothetical system we may be interested in auditing. Auditor's or instructors will, of course, want to create their own code directed toward specific objectives. But the methods presented here will serve as a rough boilerplate for the creation of simulated accounting transactions which will be useful for three purposes:

1. For audit firms, testing the power and validity of competing audit procedures in the firm (e.g., A/B testing of selected procedures).
2. For instructors, the code can be used to set up unique datasets for tests and workouts.

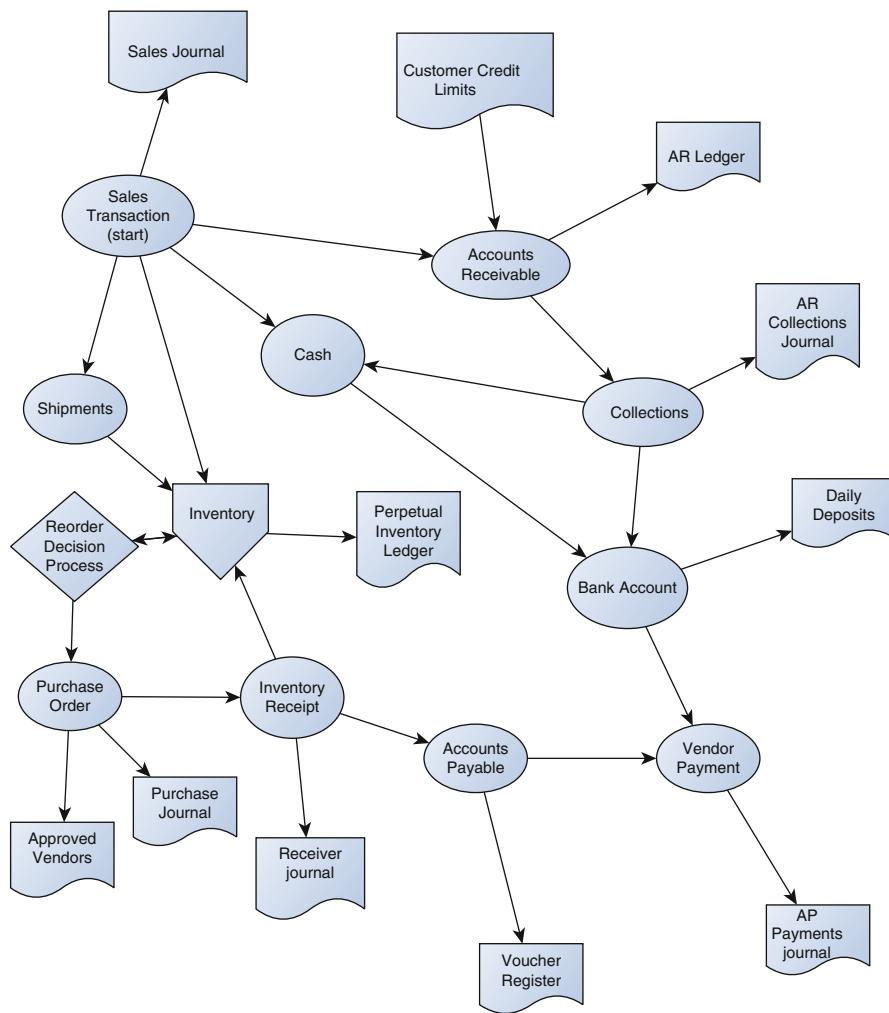


Fig. 1 'Flowchart of the sales and procurement cycle activities and records in the simulation

3. For regulatory bodies, simulated data can be used to assess the financial impact of new regulations.

The process of reviewing, modifying, and writing code to simulate real-world accounting transactions offers deep insight into the real-world properties of accounting data. Those who program realistic simulations will have a deeper understanding of real-world control processes that are the focus of the internal control review and Sarbanes–Oxley section 302 and 404 reporting. The subsequent code is only suggestive, but may be sufficient for the classroom or other purposes. I invite readers to modify and extend the modeling I have provided here to create even more nuanced and detailed simulations.

Setting Up the Simulation

Whatever the purpose, it is essential to generate realistic, large, replicable audit datasets. Our boilerplate example focuses on the two accounting cycles that generate the majority transactions analyzed in an audit—the Sales and collections cycle and the Procurement and payment cycle. These cycles are the main contributors to accounts for sales, cost of goods sold, cash, inventory, accounts receivable, and working capital.

With a few exceptions, all of our code uses R-base commands. The exceptions are Dirk Eddelbuettel’s “random” package; Hadley Wickham’s “tidyverse” and “plyr” packages; and Vitalie Spinu’s “lubridate” package for date-time formatting and arithmetic. Although they are used minimally in the subsequent code, they are four widely used packages in data science. The reader should become familiar with these packages as they make analysis and manipulation of accounting data simple and reliable.

I have used the “file\$variable” format rather than attaching and detaching files, which creates less confusion when you have many files. Variable names are descriptive, all lower case, connected by underlines—Hadley Wickham calls this “snake_case.”

Code and Data Repositories for *Audit Analytics*

Code chunks in many places in *Audit Analytics* text contain references to external .CSV formatted files. These files are hosted on GitHub in the `westland/auditanalytics` repository. A conscious decision was made, in writing the code in this book, to keep files in a .CSV format. Many sources of data that accountants encounter in practice retain data in Excel or other spreadsheet formats. Where they are not, as with database tables, they may be reformatted and presented to auditors as spreadsheets. In this vein, the last chapter creates simulated “test decks” of accounting transactions in .CSV formats because such formats are in their own way, the “native” formats for accounting and financial data. Presentation of *Audit Analytics* data in .CSV formats allows me to show examples of loading/parsing the kind of raw data encountered in practice.

This is not how files are typically stored and formatted in R. Nonetheless, for pedagogical purposes, I have chosen to retain .CSV formatting for the files that are downloaded from the book’s repositories. Otherwise GitHub’s `westland/auditanalytics` repository for this book follows the repository standards adopted by R and RStudio, and eventually I intend to release these as CRAN repositories.

Raw (.CSV) data files used in the *Audit Analytics* text are made available to the R workspace by installing the `auditanalytics` package from GitHub, and using `system.file()` to refer to specific files. The path to the file is returned by `system.file()` which can then simply be inserted into any other commands that need this path. The following code chunk provides a simple example.

```
devtools::install_github("westland/auditanalytics")
library(auditanalytics)

fyear_end_ar_ledger <- read.csv(
  system.file("extdata", "fyear_end_ar_ledger.csv", package = "auditanalytics", mustWork = TRUE))

head(fyear_end_ar_ledger)

##   X customer_no invoice_no amount shipper_no shipper_date
## 1 1      c00005     i00147    462      s00136 2020-03-01
## 2 2      c00007     i00302    504      s00313 2020-05-02
## 3 3      c00004     i00334   1122      s00318 2020-05-03
## 4 4      c00010     i00348   1292      s00345 2020-05-13
## 5 5      c00005     i00437    756      s00437 2020-06-15
## 6 6      c00005     i00461   1470      s00481 2020-07-01
```

Assumptions Used in the Generation of Simulated Data

The following assumptions are made in generating the accounting transaction simulations.

Document-Updating Events in the Accounting Cycle all Have: Identifier, Date, Number of Inventory Items, Unit Value

Entities

- stock_on_hand
- customer
- vendor
- bank

Events Associated with Entities

1. Sales cycle (amounts at sales price)

- sale
- sale_return
- shipment
- cash_receipt (cash sale)
- account_receivable (credit sale)
- collection (credit sale)
- deposit (bank)

2. Procurement Cycle (To Replace Sold Inventory; Amounts at Cost)

- inventory
- purchase_order
- account_payable
- receivers (for ordered inventory)
- disbursement (in payment to retire account_payable)

3. Reporting

- Balance Sheet
- Income Statement
- Statement of Changes in Working Capital
- Statement of Changes in Retained Earnings

Assumption There is no inflation in either buyer or seller markets throughout the year.

Assumption Documents are not grouped, rather one preceding transactions (e.g., Sales) triggers one and only one succeeding transaction (e.g., AR).

Document Generation

Accounting transaction information is highly multicolinear because many journal entries are triggered by other transactions. The subsystems simulated here are summarized in an influence chart (Fig. 2) that shows the cascade of triggered transactions.

```
library(DiagrammeR)

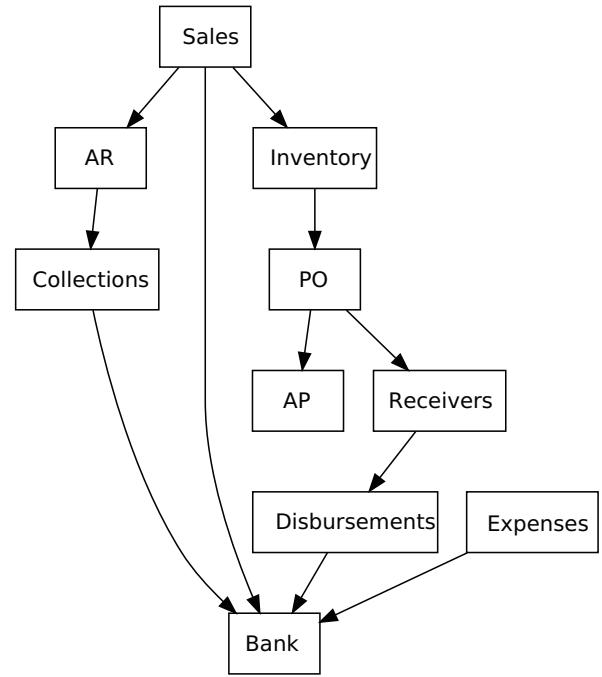
grViz("
digraph boxes_and_circles {
    # a 'graph' statement
    graph [overlap = true, fontsize = 10]

    # 'node' statements
    node [shape = box,
          fontname = Helvetica]
    Sales; AR; Collections; Bank; Inventory; PO; AP; Receivers; Disbursements; Expenses

    # 'edge' statements

    Sales -> AR
    AR -> Collections
    Sales -> Bank
    Collections -> Bank
    Sales -> Inventory
    Inventory -> PO
}
```

Fig. 2 Influence chart that showing cascade of triggered transactions



```

PO -> AP
PO -> Receivers
Receivers -> Disbursements
Disbursements -> Bank
Expenses -> Bank
}
")
  
```

Transactions are generated in the following order (items in {} means transfer data from other documents)

- sale_trigger event = {perpetual_count = perpetual_count - sale_count}
- inventory_reorder trigger event = {inventory/perpetual_count < reorder_count}
- inventory sku, unit_cost, unit sales price, count_beginning, reorder_count {= constant * sqrt(annual_sales)}
- sale invoice_no, invoice_date, sku, sale_count, sale_unit_price, sale_extended cash_or_ar, sale_return, perpetual_count {updated on sale_trigger}, cash_receipt {sale sub-classification}
- account_receivable {sale sub-classification}
- sale_return {sale sub-classification}
- shipment shipper_no, customer_no, invoice_no, shipper_date = {invoice_date + delay}
- collection receipt_no, receipt_amount = {sale_amount * unit_price}, receipt_date = {invoice_date + delay} or {invoice_date} for cash
- deposit deposit_no, deposit_amount = {sale_amount * unit_price}, deposit_date = {receipt_date + delay}
- purchase_order po_number, po_date, unit cost, reorder_amount (created on inventory_reorder trigger)
- receipt receiver_no, receiver_count, reorder_amount, receiver_date = {po_date + delay}
- account_payable ap_no, receiver_no, unit cost, reorder_count, ap_date = {receiver_date}
- disbursement disb_no, disb_amount, disb_date = {receiver_date + delay}

Strategy for Document Generation

1. The simulation creates two main files that represent starts of cascades of triggering events—(1) the *sales and inventory reduction* file and (2) the *procurement and inventory replenishment* file. Client's documents will be generated as a final step.
2. Amounts (prices, costs, counts) will be generated with a Poisson random number generator (positive integers).
3. Delays will be generated with an exponential random number generator (positive reals).
4. Classifiers will be generated using a Bernoulli random number generator (0/1).
5. Identifiers will be generated with sequential numbers ordered by date, with a transaction type prefix; e.g., s000123 will be a sale, with prefix "s."

Statistical Assumptions and the Distribution of Accounting Transactions

When one wishes to infer conclusions from some set of data—which is the objective of an audit—the data are a priori assumed to follow some probability distribution. Choice of the correct distribution for a specific situation or account is one of the most important aspects of effective inference from data, and predicates efficient and effective auditing.

In addition to choosing the right distribution, auditors need to determine how much data to collect—the appropriate sample size that balances the cost of an incorrect decision with the cost of auditing. Data collection and processing in auditing is expensive—audit costs for a single sampled transaction may run into hundreds or thousands of dollars. The total cost of an audit is extremely sensitive to the right choice of distributions, sample sizes, methodology for inference, and decision models. Auditing that is cavalier with methods, relying too much on personal inference, will drive up audit costs and raise the probability of an incorrect opinion.

Sample size determination is one of the most confusing areas of statistics. Ideally we would not choose a sample size “up front,” but would continue sampling until a particular objective is met (e.g., until we determine that the system is either in or out of control). This is called “optimal stopping” and is often used in laboratory and factory sampling. Where we need to do a “one-shot” sample, we need to know the distribution of the parameter we are estimating—this is usually impossible to do accurately, but that does not stop us from trying.

Many auditors assume they can simply assume client transactions to be Normally distributed, perhaps invoking but not understanding the Central Limit Theorem. This is usually a very bad assumption in auditing, since all audited distributions should be censored at zero and are usually right-skewed and highly kurtotic. It is also not uncommon to find that transaction distributions are multimodal (Westland 2017). Furthermore, attributes will have a discrete distribution, often with a limited set of “valid” values.

Thus auditor cannot assume that “one-size-fits-all” in sample size determination. There are three factors that need to be considered:

1. Effect size: for attribute error rates this is “tolerable error”; for value estimation this is the proportion of materiality assigned to an account, also called “tolerable error.”
2. Population distribution and parameters: A generally safe assumption is that the attribute errors or the dollar errors are Poisson distributed. When transactions are Poisson distributed, then the waiting time between transactions is typically assumed to be distributed exponentially when the transactions are independent. Compliance and substantive errors in transactions and financial accounts are relatively rare events—if they were not, then financial statements would be unusable and the scope of auditing would be unmanageable (a situation sometimes found in financial fraud cases). The law of rare events supports an assumption of Poisson distribution for monetary balances, and an exponential distribution for delays; e.g., the time a customer takes to pay their account receivable. The Poisson distribution is also sometimes called the law of small numbers (after the title of an 1896 book by Ladislaus Bortkiewicz) because it is the probability distribution of the number of occurrences of an event that happens rarely but has very many opportunities to happen. Numerous studies support the assumption of a Poisson process generating the distribution of accounting errors with (Ball and Watts 2012, Fienberg, Neter, and Leitch 1977, Garstka 1977, Greene 1994, Keenan and Kotula 2004, Repin, Sauter, and Smolianski 2004).

3. Confidence in our decision that: e.g., that the error rate observed in the sample would cause us to change our audit decision, or an investor to change their investment decision. Power and significance of audit tests set our formal definition of “confidence.” Computing sample size here is analogous to standard univariate calculations (Cochran 1977, Kish 1955, Lohr 1999, Snedecor and Cochran 1989, Westland and See-to 2007) but using a formulation for variance customized to this problem.

General Parameters of the Simulation

Based on our prior assumptions, we start by setting the simulation-wide parameters. These will include the transaction volumes, markups of costs to sales values, numbers of customers, inventory items, and so forth. Any of these parameters, as well as certain other parameters in the following code can be customized to the user’s own needs.

```
#####
## 1 - Simulation-wide parameters and general use functions
##
#####
## clear workspace (recommended)

rm(list=ls())

##
## your seed number is the starting point used in the
## generation of a sequence of random numbers
## assuring that you will obtain the same
## accounting datasets (given the same seed number)
## when you rerun the code
##

set.seed(123456) ## setting the seed assures that we can recreate the dataset by rerunning the code

#
# Fiscal year beginning and end dates
#
fyear_begin <- paste0(format(Sys.Date(), '%Y'), '-01-01') # date can be hard-coded if needed
fyear_end <- paste0(format(Sys.Date(), '%Y'), '-12-31')

#
# Inventory related parameters
#
no_of_sku <- 10      ## number of stock-keeping-units (sku), i.e., distinct inventory items
lngth_of_sku <- 5    ## number of alphanumeric characters in sku identifier
avg_cost <- 1000     ## ave. cost of sku from vendor
avg_markup <- 200     ## ave. markup of sales_price sku from vendor
avg_count <- 300      ## ave. number of items of sku
#
# Sales related Parameters
#
no_of_cust = 10       ## number of customer for sales
no_of_sales = 1000     ## number of sales transactions
avg_count_sold = 15    ## average number of items sold per transaction
#
# Shipping and A/R collection related parameters
# Starting balances have a date field of "1900-01-01" to avoid non-date contents
#
days_to_ship <- 10          ## average of 10 days to ship
days_to_collect_ar <- 60      ## average of 60 days to collect on A/R,
percent_ar_unpaid <- .02      ## 2% of AR are never paid (in default)
ar_in_default_cutoff <- 500    ## over 499 days late is in default
#
# Purchase related parameters
#
days_to_receive_inventory <- 15 # average of 15 days to receive ordered inventory
ap_record_created_delay <- 1    # account payable record created 1 day after receiver of inventory
ap_disbursement_delay <- 75      # account payable payment disbursement on average 75 days after receiver of inventory
```

```

#
##
## random dates
##

starting_inventory_budget <- 1000 # the approximate $ inventory on hand at the start of the simulation

rdate <- function(x,
                    min = paste0(format(Sys.Date(), '%Y'), '-01-01'),
                    max = paste0(format(Sys.Date(), '%Y'), '-12-31'),
                    sort = TRUE) {
  dates <- sample(seq(as.Date(min), as.Date(max), by = "day"), x, replace = TRUE)
  if (sort == TRUE) {
    sort(dates)
  } else {
    dates
  }
}

```

This last function helps in generating realistic dates and time intervals for the simulation.

The Sales Journal

Assuming we have inventory to sell, our transaction recording begins when a customer takes delivery of inventory in exchange for cash or an obligation to pay (i.e., and account receivable). The individual sales transactions are recorded in a sales journal. In the past, “journals” were physical books, where the original occurrence of a transaction was recorded, along with ad hoc notes indicating the nature of the transaction. “Ledgers,” on the other hand, were physical books that were used to track inventory and other stock items. Today, records are digital and descriptive fields are strictly defined in advance.

The following code generates the original transaction from a sale to a customer, as well as those transactions that are triggered by a subsequent event (e.g., collection) related to that particular sales transaction.

```

#####
# 2 - Populate the Sales Journal file
#####

library(tidyverse)
library(lubridate)
library(random)

#
# Create a sales_journal and then populate it
# Sales prefix = "i"; invoice_no is sequenced by date)
# order by invoice date (use lubridate and plyr)
#

invoice_no <- seq(from = 1, to = no_of_sales)
invoice_no <- sprintf("i%05d", invoice_no)

invoice_date <- ymd(rdate(no_of_sales, min = fyear_begin, max = fyear_end))

sales_journal <-
  data.frame(invoice_no, invoice_date) %>%
  arrange(invoice_date)

#
# create customers and tie them to sales
#

```

```

customer_string <- seq(from = 1, to = no_of_cust)
customer_string <- sprintf("c%05d", customer_string)
customer_no = sample(customer_string, no_of_sales, replace = T, prob=runif(no_of_cust,0,1))
sales_journal = data.frame(customer_no, sales_journal)

#
#  create skus and their vendors
#

sku <- randomStrings(n=no_of_sku, len=lnghth_of_sku, digits=F, upperalpha=TRUE,
                      loweralpha=F, unique=TRUE, check=TRUE)
vendor_string <- seq(from = 1, to = no_of_sku / 3)
vendor_string <- sprintf("v%05d", vendor_string)
vendor_no = sample(vendor_string, no_of_sku, replace = T, prob=runif(no_of_sku/3,0,1))
sku = data.frame(vendor_no, sku) %>%
  rename(sku = V1)

sales_count <- rpois(no_of_sales,avg_count_sold)
sku_of_sale <- sample(sku$sku, no_of_sales, replace=TRUE) ## sku from 1.1
sales_return <- rbinom(no_of_sales, 1, .05) ## 1 if there was a return on this sale
cash_not_ar <- rbinom(no_of_sales, 1, .2) ## 1 is a cash sale (20%), 0 is credit (80%)

unit_cost <- rpois(nrow(sku), 30)
sales_unit_price <- unit_cost * ceiling(abs(rnorm(nrow(sku), 2, 1)))

#
#  Write sales_journal
#

sales_journal <- do.call(
  cbind.data.frame,
  list(sales_journal,
       sku_of_sale,
       sales_count,
       cash_not_ar,
       sales_return,
       unit_cost,
       sales_unit_price
     )
)

colnames(sales_journal) <-
  c("customer_no",
    "invoice_no",
    "invoice_date",
    "sku",
    "sales_count",
    "cash_not_ar",
    "sales_return",
    "unit_cost",
    "sales_unit_price")

#####
## 3 - Add inventory and other information to Sales Journal
##      shipments & collections,
##      which are defined by unique IDs and date delays
##      2% of AR are never paid (in default)
#####

```

```

library(lubridate)

sales_journal$sales_extended <- sales_journal$sales_count * sales_journal$sales_unit_price

shipper_date <- invoice_date + round(rexp(no_of_sales, 1/days_to_ship), 0)

collection_amount <- sales_journal$sales_extended ## set amount for cash sales
collection_date <- sales_journal$invoice_date ## set date for cash sales
for(i in 1:no_of_sales) {
  if(cash_not_ar[i] == 0) { ## collect only on credit sales
    if(rbinom(1, 1, percent_ar_unpaid) == 1) {collection_date[i] <- shipper_date[i] + ar_in_default_cutoff}
    ## 1 is a default (percent_ar_unpaid %)
    collection_date[i] <- shipper_date[i] + round(rexp(1, 1/days_to_collect_ar), 0)
    collection_amount[i] <- sales_journal$sales_extended[i]
  }
}

##
## collection_no prefix = "r"; is sequenced by date)
# (lubridate and plyr)
#
sales_journal$collection_amount <- sales_journal$collection_amount <- as.numeric(collection_amount)
sales_journal$collection_date <- sales_journal$collection_date <- ymd(collection_date)
sales_journal <- arrange(sales_journal, collection_date)
sales_journal$collection_no <- sprintf("r%05d", seq(from = 1, to = no_of_sales))

sales_journal <- arrange(sales_journal, invoice_date)
sales_journal <- arrange(sales_journal, invoice_date)
sales_journal$collection_no <- sales_journal$collection_no

#
## shipper_no prefix = "s"; is sequenced by date)
# (use lubridate and plyr)
#
sales_journal$shipper_date <- ymd(shipper_date)
sales_journal <- arrange(sales_journal, shipper_date)
sales_journal$shipper_no <- sprintf("s%05d", seq(from = 1, to = no_of_sales))

sales_journal <- arrange(sales_journal, invoice_date)
sales_journal <- arrange(sales_journal, invoice_date)
sales_journal$shipper_no <- sales_journal$shipper_no
sales_journal$shipper_date <- sales_journal$shipper_date

collections <- sales_journal %>%
  select(customer_no,
         invoice_date,
         invoice_no,
         sales_extended,
         collection_amount,
         collection_date,
         collection_no,
         cash_not_ar,
         sku)

collections <- split(collections, cash_not_ar)$'0' ## credit only

shipments_journal <-
  sales_journal %>%
  select(shipper_no, date=shipper_date, customer_no, sku, invoice_no, invoice_date, quantity = sales_count)

```

Cash and Bank Deposits

Cash sales and collections on accounts receivable generate cash, which is deposited to the bank. Inventory procurement and other expenditures will use cash from the bank account. We assume that deposits are made daily in this code.

```
#####
# 4 - Generate daily bank deposit records
##
#####

library(tidyverse)

##
## Daily deposits
##

deposit_daily <-
  sales_journal %>%
  group_by(invoice_date) %>%
  summarize(deposit_amount = sum(sales_extended)) %>%
  as.data.frame()

deposit_no <- sprintf("d%05d", seq(from = 1, to = nrow(deposit_daily)))
deposit_daily <-
  cbind(deposit_daily, deposit_no) %>%
  rename(deposit_date = invoice_date)
```

Inventory and Purchase Orders for Inventory Replenishment

Eventually after selling enough inventory, that inventory will need to be replenished. In practice, inventory levels for any particular stock keeping unit (SKU) are monitored, and a purchase order is generated when inventory falls below a certain point. In this code we used a standard economic order quantity model to determine when a purchase order is triggered. The purchase order is sent to the vendor, who after a certain “lead time” delivers the inventory to your client. Delivery is recorded as a “receipt” and a receiver (transaction document) is written up, as well as an account payable (voucher).

Generator functions allow you to declare a function that behaves like an iterator, but in a fast, easy, and clean way. An iterator is an object that can be iterated (looped) upon. It is used to abstract a container of data to make it behave like an iterable object. Simply speaking, a generator is a function that returns an object (iterator) which we can iterate over (one value at a time).

A generator function is a special type of function that you call repeatedly to obtain a sequence of values. Often, generators need to maintain internal state, so they are typically constructed by calling another function that returns the generator function (the environment of the function that returns the generator is then used to track state). For example, the following `sequence_generator()` function returns a generator function that yields an infinite sequence of numbers:

```
sequence_generator <- function(start) {
  value <- start - 1
  function() {
    value <-> value + 1
    value
  }
}
> gen <- sequence_generator(10)
> gen() # 10
> gen() # 11
```

Generator functions can signal completion by returning the value NULL. The current state of the generator is the value variable that is defined outside of the function. Note that superassignment (`<->`) operator is used to update this state from within the function.

My simulation will use a fixed 50 unit order quantity, and fixed 50 unit reorder point for generation of purchase orders. A more sophisticated approach would use economic order quantities (EOQ) to set the size of purchase orders; this could be implemented via R's `plyr` package as shown below.

```
#####
# 5 - Generate inventory EOQs (economic order quantities)
##
#####

library(plyr)
library(kableExtra)

###  

## EOQ = sqrt(2SD / H) where S = Setup costs, D = Demand rate, H = Holding costs  

## so reorder_count should be proportional to sqrt(D)  

###  

###  

reorder <-  

  ddply(sales_journal, .(sku), summarize,  

    reorder=round((sum(sales_count))^.5,0))  

kable(reorder, caption = "Economic Order Quantities for Each SKU") %>%  

  kable_styling(bootstrap_options = "striped")
```

Inventory

Sales and procurement cycles represent activities that consume and replenish inventory. Thus our starting point for simulation is inventory —its identifiers, vendors, pricing, and so forth.

```
reorder_stock_level <- 50
reorder_count <- 100

#####
# 6- Generate the Inventory (by SKU) file
#####

library(lubridate)
library(tidyverse)

## GENERATOR FUNCTIONS FOR THE SEQUENTIAL TRANSACTION NUMBERING
#
## Purchase order prefix = "p";
## voucher_no prefix = "ap";
## disbursement_no prefix = "dis";
## receiver_no prefix = "rec";
#  

#  

po_no_generator <- function(){
  i <- 0
  function(){
    i <- i + 1
    sprintf("p%05d", i)
```

```

    }
}

new_po <- po_no_generator()

rec_no_generator <- function() {
  i <- 0
  function() {
    i <= i + 1
    sprintf("rec%05d", i)
  }
}
new_rec <- rec_no_generator()

ap_no_generator <- function() {
  i <- 0
  function() {
    i <= i + 1
    sprintf("ap%05d", i)
  }
}
new_ap <- ap_no_generator()

disb_no_generator <- function() {
  i <- 0
  function() {
    i <= i + 1
    sprintf("dis%05d", i)
  }
}
new_disb <- disb_no_generator()

## compute the starting inventory for each stock-keeping unit (sku)
ave_starting_inventory <- 100

fyear_begin_inventory_ledger <-
  sales_journal %>%
  as.data.frame() %>%
  group_by(sku) %>%
  slice(1) %>%
  select(sku, unit_cost, sales_unit_price) %>%
  as.data.frame() %>%
  mutate(stock_on_hand = ave_starting_inventory + floor(rnorm(no_of_sku, 0, 10)),
         invoice_no=0,
         invoice_date=as_date("1900-01-01"),
         sales_count=0,
         sales_return=0
  )
  #####
  ##

## Create the purchase journal
## based on the reorder_stock_level &
## reorder_count for each sku inventory
## #####
#####

sku_stock <-
  fyear_begin_inventory_ledger %>%

```

```
select(sku, stock_on_hand)

reorder_stock_level <- 50
reorder_count <- 100

for(i in 1:nrow(sales_journal)){
  for(j in 1:nrow(sku_stock)){
    if(sku_stock$sku[j] == sales_journal$sku[i]){

      sales_journal$running[i] <-
        sku_stock$stock_on_hand[j] -
        sales_journal$sales_count[i] +
        sales_journal$sales_return[i]

      sku_stock$stock_on_hand[j] <-
        ifelse(sales_journal$running[i] < reorder_stock_level,
               sales_journal$running[i] + reorder_count,
               sales_journal$running[i])

      sales_journal$po_count[i] <-
        ifelse(sales_journal$running[i] < reorder_stock_level,
               reorder_count,
               0)
      sales_journal$po_no[i] <-
        ifelse(sales_journal$running[i] < reorder_stock_level,
               new_po(),
               0)
      sales_journal$receiver_no[i] <-
        ifelse(sales_journal$running[i] < reorder_stock_level,
               new_rec(),
               0)

      sales_journal$ap_no[i] <-
        ifelse(sales_journal$running[i] < reorder_stock_level,
               new_ap(),
               0)
      sales_journal$disbursement_no[i] <-
        ifelse(sales_journal$running[i] < reorder_stock_level,
               new_disb(),
               0)

## note that ifelse fails to parse dates, and
## I use a difference approach for the dates that doesn't use 'if'
## rather it prunes out the non-PO rows to create the purchase journal

## Also note that I am updating inventory on the po_date, but the correct
## date will be the receiver_date. This will create inventory count (cutoff)
## discrepancies; the number of these can be modulated
## by changing the days_to_receive_inventory parameter at the beginning of the code

sales_journal$receiver_date[i] <-
  sales_journal$invoice_date[i] + round(rexp(1,1/days_to_receive_inventory))

sales_journal$po_date[i] <-
  sales_journal$invoice_date[i]

sales_journal$ap_date[i] <-
  sales_journal$receiver_date[i] + ap_record_created_delay
```

```

sales_journal$disbursement_date[i]      <-
  sales_journal$receiver_date[i]  + round(rexp(1,1/75),0)

}

sales_journal$receiver_date <- as_date(sales_journal$receiver_date)
sales_journal$po_date <- as_date(sales_journal$po_date)
sales_journal$ap_date <- as_date(sales_journal$ap_date)
sales_journal$disbursement_date <- as_date(sales_journal$disbursement_date)

#####
## Create the various purchase related journals
##
#####

purchase_journal <-
  sales_journal %>%
  filter(po_count != 0) %>%
  select(sku, po_count, unit_cost,
         po_no, po_date,
         receiver_no, receiver_date,
         ap_no, ap_date,
         disbursement_no, disbursement_date)

perpetual_inventory_ledger <-
  sales_journal %>%
  arrange(sku, invoice_date) %>%
  select(sku, date = invoice_date, stock_on_hand = running)

sales_journal <- sales_journal[,1:15]

receiver_journal <-
  purchase_journal %>%
  select(receiver_no, receiver_date,
         sku, unit_cost, received = po_count, po_no, po_date)

ap_ledger <-
  purchase_journal %>%
  mutate(extended_cost = po_count * unit_cost) %>%
  select(ap_no, ap_date, sku, no_units_ordered = po_count, extended_cost,
         receiver_no, receiver_date)

disbursement_journal <-
  purchase_journal %>%
  select(disbursement_no, disbursement_date, sku,
         no_units_ordered = po_count,
         unit_cost, ap_no, ap_date)

purchase_journal <-
  purchase_journal %>%
  select(po_no, po_date, sku, po_count, unit_cost)

#####
## Cost of Goods Sold
## After sales, the most significant account on the income statement is

```

```

## cost of goods sold (essentially sales restated
## at their procurement / manufacturing cost).
#####
# daily_sales_cgs_margin <-
# sales_journal %>%
# group_by(invoice_date) %>%
# mutate(cgs = sum(sales_count * unit_cost),
#        gross_margin = sum((sales_unit_price - unit_cost) * sales_count)
# )%>%
# summarize(total_sales = sum(sales_extended),
#           cgs=sum(cgs),
#           gross_margin = sum(gross_margin),
#           date = first(invoice_date)) %>%
# ungroup() %>%
# as.data.frame() %>%
# select(date , total_sales, cgs, gross_margin)

```

Perpetual Inventory, Accounts Payable, and Other Inventory Related Accounts

Before the computerization of accounting in the 1970s, most inventory tracking was performed with periodic counts. Cycle counts of subsets of SKUs would update the inventory ledger. There were two significant problems with periodic inventory: (1) stockouts and obsolescence were common due to the disconnect between transactions and actual balances; and (2) thefts and shortages were difficult to control. Computerization made perpetual (i.e., up to date) inventory balances possible, while reducing the labor cost associated with counts. By the 1990s, most companies were relying on perpetual inventory systems. A significant part of the audit is spent on validating the accuracy of perpetual inventory balances.

The prior code chunk computed inventory reorders, and the associated accounts in response to sales demand. I provide a code chunk here to break out the separate accounting documents that would appear in an accounting system.

The purchase order is sent to the vendor, who after a certain “lead time” delivers the inventory to your client. Delivery is recorded as a “receipt” and a receiver (transaction document) is written up, as well as an account payable (voucher). The term “voucher” refers to an old paper system that was used throughout the nineteenth and twentieth centuries, where paperwork from purchase orders, receivers, debit memos, disbursements, and so forth were stapled together and filed. You will still find that term used in digital accounting systems.

Customer Credit Limits and Outstanding Accounts Receivable

The amount of credit extended to customers is typically monitored carefully to limit the risk of non-payment of accounts receivable. The risk of non-payment and suggestions for credit limits to particular customers may be obtained from service companies such as Dun & Bradstreet. Typically a customer is given a credit limit based on their payment history with the company as well as external assessments of risk.

```

#####
## 7 - customers' credit limits
## cr limit proportional to 10 x {annual sales / sku} / {time_to_pay} + {random}
## we would like to have some customers exceeding their credit limit
#####

library(tidyverse)
library(lubridate)
library(kableExtra)

cr_limit <-

```

```

sales_journal %>%
  group_by(customer_no) %>%
  mutate(ann_sales = sum(sales_extended),
        time_to_pay = mean(as.numeric(collection_date - invoice_date)),
        credit_limit = 10 * round(ann_sales / time_to_pay, digits=-2)) %>%
  slice(1) %>%
  select(customer_no, credit_limit)

kable(cr_limit, caption = "Customer Credit Limits") %>%
  kable_styling(bootstrap_options = "striped")

```

Accounts Receivable

I assume that the business has just started this year. Were we to assume a beginning accounts receivable balance, I would also need to generate a set of sales transactions from the prior year, with shipments and collections, which would add complexity without any accompanying insights. This can be left as an exercise for the interested reader. The sales would be generated by replicating the mechanisms that I have already used to generate sales in the audit year.

Credit sales generate accounts receivable, which may (or may not) ultimately be collected, but are sometimes left unpaid for various reasons. The confirmation of accounts receivable and assessment of uncollectable accounts receivable are considered two of the most important tasks in an audit.

```

#####
# 8 - accounts receivable (updated daily)
## follows the pattern to compute daily perpetual ar
## you will need this for the tests of whether customers exceed credit limits
#####

library(tidyverse)
library(lubridate)

real_world_credit_sales <- 
  credit_sales_journal <-
  sales_journal[,1:15] %>%
  filter(cash_not_ar == 0)    ## credit only

real_world_cash_sales <-
  cash_sales_journal <-
  sales_journal[,1:15] %>%
  filter(cash_not_ar == 1)    ## cash only

## put daily AR, plus an aging ...

fyear_dates <-  data.frame(f_date=seq(as.Date(fyear_begin), as.Date(fyear_end), "day"))
# sales_journal <-  full_join(sales_journal,fyear_dates, by=c("invoice_date"="f_date"))

#%>%  full_join(fyear_dates, by=c("date"="f_date"))

daily_cum_ar_balance <-
  sales_journal %>%
  filter(cash_not_ar == 0) %>%      ## credit sales only
  arrange(invoice_date) %>%
  group_by(invoice_date) %>%
  summarize(sales_sum = sum(sales_extended)) %>%

```

```

ungroup(invoice_date) %>%
right_join(fyear_dates, by=c("invoice_date"="f_date"))

daily_cum_ar_balance[is.na(daily_cum_ar_balance)] <- 0

daily_cum_ar_balance <-
  daily_cum_ar_balance %>%
  mutate(ar_cum = cumsum(sales_sum)) %>%
  select(date=invoice_date, ar_cum) %>%
  arrange(date)

#daily_cum_ar_balance[is.na(daily_cum_ar_balance)] <- 0

daily_cum_collections <-
  sales_journal %>%
  filter(cash_not_ar == 0) %>%      ## credit sales only
  arrange(collection_date) %>%
  group_by(collection_date) %>%
  summarize(collect_sum = sum(collection_amount)) %>%
  ungroup(collection_date) %>%
  right_join(fyear_dates, by=c("collection_date"="f_date"))

daily_cum_collections[is.na(daily_cum_collections)] <- 0

daily_cum_collections <-
  daily_cum_collections %>%
  as.data.frame() %>%
  filter(collection_date <= fyear_end) %>%
  mutate(collect_cum = cumsum(collect_sum)) %>%
  select(date = collection_date, collect_cum) %>%
  arrange(date)

daily_ar_balance <- inner_join(daily_cum_ar_balance, daily_cum_collections, "date")

daily_ar_balance <-
  daily_ar_balance %>%
  mutate(ar_balance = ar_cum - collect_cum) %>%
  select(date, ar_cum , collect_cum, ar_balance) %>%
  arrange(date)

```

Accounts Receivable Aging

The accounts receivable aging may be either automatically compiled by the client, as it is here; or the auditor may compile it from the work papers, as would be done during substantive testing. The accounts receivable aging provides a basis for revising the amounts in the allowance for uncollectable accounts. Typically there are “rules of thumb” for the collectability of accounts that have been outstanding and unpaid for a certain amount of time; this will vary by client and by customer. As was argued in chapter “Substantive Tests” on substantive testing, a more formal basis can be created for estimation and allowance for uncollectable accounts using time-series forecasting. For this chapter, I simply print the aging that would be the basis for traditional ways of calculating the allowance for doubtful accounts.

```

#####
# 9 - accounts receivable aging
##
#####
```

```

library(tidyverse)
library(lubridate)
library(kableExtra)

## create age breaks at <45 days; 45 - 120 days; >120 days

i_date <- yday(ymd(as_date(sales_journal$invoice_date)))
ye_date <- yday(ymd(as_date(fyear_end)))

age_ar <-
  sales_journal %>%
  filter(cash_not_ar == 0) %>%
  filter(collection_date > fyear_end) %>%
  mutate(i_date <- yday(ymd(as_date(invoice_date))),
         ye_date <- yday(ymd(as_date(fyear_end))),
         age =ye_date - i_date,
         age_lt_45 = ifelse(age<45, sales_extended, 0),
         age_45_120 = ifelse(age<=120 & age>=45,sales_extended,0),
         age_gt_120 = ifelse(age>120,sales_extended,0)    ## redundant but easier to read
    ) %>%
  select(customer_no, invoice_no, age, sales_extended,age_lt_45, age_45_120, age_gt_120)

## aging by customer

customer_age_ar <-
  age_ar %>%
  group_by(customer_no) %>%
  summarize(total = sum(sales_extended),
            lt_45 = sum(age_lt_45),
            between_45_120 = sum(age_45_120),
            gt_120 = sum(age_gt_120)
  ) %>%
  select(customer_no, total,lt_45, between_45_120, gt_120)

knitr::kable(customer_age_ar, caption = "Customer A/R Aging") %>%
  kable_styling(bootstrap_options = "striped")

## aging of all A/R

age_ar <-
  age_ar %>%
  summarize(total = sum(sales_extended),
            lt_45 = sum(age_lt_45),
            between_45_120 = sum(age_45_120),
            gt_120 = sum(age_gt_120)
  ) %>%
  select(total,lt_45, between_45_120, gt_120)

knitr::kable(age_ar, caption = "A/R Aging") %>%
  kable_styling(bootstrap_options = "striped")

```

Employee Expenditures

The expense file is not related to anything else. I make this interesting for fraud testing (Benford tests).

```
#####
## 10 - Employee expenditures
#####

no_exp_records <- floor(no_of_sales/3)
exp_string <- seq(1, no_exp_records)
exp_string <- sprintf("E%05d", exp_string)

exp_date <- rdate(no_exp_records, min = fyear_begin, max = fyear_end)

## create some odd behavior for the Benford test

exp_set_1 <- runif(no_exp_records/3, 1, 1000)
exp_set_2 <- rpois(no_exp_records/3, 490)
exp_set_3 <- abs(rnorm(no_exp_records/3, 900, 200))

no_emp_records <- floor(no_exp_records/20)
emp_string <- seq(1, no_emp_records)
emp_string <- sprintf("Emp%04d", emp_string)

emp_string <- sample(
  emp_string,
  no_exp_records,
  replace = T,
  prob=runif(no_emp_records, 0,1))

expenditures <- data.frame("exp_no"=exp_string ,
                            "employee_no"=emp_string ,
                            "date"= exp_date,
                            "amount"= c(exp_set_1,exp_set_2,exp_set_3)
                           )
```

Omissions, Duplications, and Monetary Errors in Transactions

Test for omissions and duplication of transactions is standard in the interim tests of internal controls. The prior generation of “real-world” AR and Inventory records is complete, but in this section, we remove and duplicate transactions to simulate a failure of controls in the client’s accounting systems.

```
#####
## 11 - Omitted, Duplicate and In-Error Records
## (errors are called "taintings" in the monetary unit sampling vernacular)
#####

# set.seed(123456)
library(tidyverse)

## create the real world transaction journals

credit_sales_journal <-
  sales_journal[,1:15] %>%
  filter(cash_not_ar == 0) %>%
```

```

select(customer_no,
       invoice_no,
       invoice_date,
       sku,
       sales_count,
       sales_return,
       unit_cost,
       sales_unit_price,
       collection_no,
       collection_date,
       collection_amount,
       shipper_no,
       shipper_date
     )

real_world_credit_sales <- credit_sales_journal

## omitted records

omit_row <- as.data.frame(rbinom(nrow(real_world_credit_sales), 1, .05))
colnames(omit_row) = "omitted"

test <- cbind(real_world_credit_sales, omit_row)
test1 <- split(test, omit_row)
real_world_credit_sales_new <- test1$`0`

n <- nrow(test1$`1`)
cat("\n\n # omitted credit sales = ", n)

## duplicated records

dup_row <- as.data.frame(rbinom(nrow(real_world_credit_sales_new), 1, .05))
colnames(dup_row) = "duplicated"

test <- cbind(real_world_credit_sales_new, dup_row)
test1 <- split(test, dup_row)
real_world_credit_sales_new <- rbind(test, test1$`1`)
real_world_credit_sales_new$dup_row = NULL

n <- nrow(test1$`1`)
cat("\n\n # duplicated credit sales = ", n)

## 'taint' 5% the values in monetary fields

real_world_credit_sales_new$taint_ed <-
  rbinom(nrow(real_world_credit_sales_new), 1, .05) ## 5%

real_world_credit_sales_new$taint_ing <-
  (real_world_credit_sales_new$sales_count / mean(real_world_credit_sales_new$sales_count)) *
  abs(rnorm(nrow(real_world_credit_sales_new), .5, .5))

n <- nrow(real_world_credit_sales_new)
cat("\n\n # omitted credit sales = ", n)

## Cash sales - Note that these share the invoice number sequence with credit sales

cash_sales_journal <

```

```

sales_journal[,1:15] %>%
  filter(cash_not_ar == 1) %>%
  select(customer_no,
         invoice_no,
         invoice_date,
         sku,
         sales_count,
         sales_return,
         unit_cost,
         sales_unit_price,
         collection_no,
         collection_date,
         collection_amount,
         shipper_no,
         shipper_date
        )

real_world_cash_sales <- cash_sales_journal

## omitted records

omit_row <- as.data.frame(rbinom(nrow(real_world_cash_sales), 1, .05))
colnames(omit_row) = "omitted"

test <- cbind(real_world_cash_sales, omit_row)
test1 <- split(test, omit_row)
real_world_cash_sales_new <- test1$`0`

n <- nrow(test1$`1`)
cat("\n\n # omitted cash sales = ", n)

## duplicated records

dup_row <- as.data.frame(rbinom(nrow(real_world_cash_sales_new), 1, .05))
colnames(dup_row) = "duplicated"

test <- cbind(real_world_cash_sales_new, dup_row)
test1 <- split(test, dup_row)
n <- nrow(test1$`1`)
cat("\n\n # duplicated cash sales = ", n)

real_world_cash_sales_new <- rbind(test, test1$`1`)

n <- nrow(test1$`1`)
cat("\n\n # duplicated cash sales = ", n)

## 'taint' 5% the values in monetary fields

real_world_cash_sales_new$taint_ed <-
  rbinom(nrow(real_world_cash_sales_new), 1, .05) ## 5%

real_world_cash_sales_new$taint_ing <-
  (real_world_cash_sales_new$sales_count / mean(real_world_cash_sales_new$sales_count)) *
    abs(rnorm(nrow(real_world_cash_sales_new), .5, .5))

n <- nrow(real_world_cash_sales_new)
cat("\n\n # tainted cash sales = ", n, "\n\n")

```

```

## collections

real_world_collections <- collections

## omitted records

omit_row <- rbinom(nrow(real_world_collections), 1, .05)

test <- cbind(real_world_collections, omit_row)
test1 <- split(test, omit_row, drop=F)
real_world_collections_new <- test1$`0`

n <- nrow(test1$`1`)
cat("\n\n # omitted collections = ", n)

## duplicated records

dup_row <- rbinom(nrow(real_world_collections_new), 1, .05)

test <- cbind(real_world_collections_new, dup_row)
test1 <- split(test, dup_row, drop=F)
real_world_collections_new <- rbind(test, test1$`1`)
real_world_collections_new$dup_row = NULL

n <- nrow(test1$`1`)
cat("\n\n # duplicated collections = ", n)

## 'taint' 5% the values in monetary fields

real_world_collections_new$taint_ed <-
  rbinom(nrow(real_world_collections_new), 1, .05) ## 5%

real_world_collections_new$taint_ing <-
  rexp(nrow(real_world_collections_new), .5) -.5 ## exponential -50%-inf% taint

split_cash <-
  split(real_world_collections_new,
        real_world_collections_new$taint_ed)

split_cash$`1`$sales_extended <-
  split_cash$`1`$sales_extended * split_cash$`1`$taint_ing

collections <- rbind(split_cash$`1`, split_cash$`0`)

n <- nrow(split_cash$`1`)
cat("\n\n # tainted collections = ", n, "\n\n")

## AR

## Sales-inv for the AR records

real_world_sales_journal <- sales_journal

## omitted records

omit_row <- rbinom(nrow(real_world_sales_journal), 1, .05)

```

```

test <- cbind(real_world_sales_journal,omit_row)
test1 <- split(test,omit_row, drop=F)
real_world_sales_journal_new <- test1$`0`

n <- nrow(test1$`1`)
cat("\n\n # omitted s_i = ",n )

## duplicated records

dup_row <- rbinom(nrow(real_world_sales_journal_new),1,.05)

test <- cbind(real_world_sales_journal_new,dup_row)
test1 <- split(test,dup_row, drop=F)
real_world_sales_journal_new <- rbind(test, test1$`1`)
real_world_sales_journal_new$dup_row = NULL

n <- nrow(test1$`1`)
cat("\n\n # duplicated s_i = ",n )

## 'taint' 5% the values in monetary fields

real_world_sales_journal_new$taint_ed <-
  rbinom(nrow(real_world_sales_journal_new),1,.05) ## 5%

real_world_sales_journal_new$taint_ing <-
  rexp(nrow(real_world_sales_journal_new),.5)-.5 ## exponential -50%-inf% taint

split_cash <-
  split(real_world_sales_journal_new,
        real_world_sales_journal_new$taint_ed)

split_cash$`1`$sales_extended <-
  split_cash$`1`$sales_extended * split_cash$`1`$taint_ing

sales_journal <- rbind(split_cash$`1`,split_cash$`0`)

n <- nrow(split_cash$`1`)
cat("\n\n # tainted s_i = ",n, "\n\n ")

sales_journal <- real_world_sales_journal_new

## Inventory
## generate omitted inv'y records

real_world_po_journal <- purchase_journal

## omitted records

omit_row <- rbinom(nrow(real_world_po_journal),1,.05)

test <- cbind(real_world_po_journal,omit_row)
test1 <- split(test,omit_row, drop=F)
real_world_po_journal_new <- test1$`0`

n <- nrow(test1$`1`)
cat("\n\n # omitted po_inv = ",n )

```

```
## duplicated records

dup_row <- rbinom(nrow(real_world_po_journal_new), 1, .05)

test <- cbind(real_world_po_journal_new, dup_row)
test1 <- split(test, dup_row, drop=F)
real_world_po_journal_new <- rbind(test, test1$`1`)
real_world_po_journal_new$dup_row = NULL

n <- nrow(test1$`1`)
cat("\n\n # duplicated po_inv = ", n)

purchase_journal <- real_world_po_journal_new
```

Audit Tasks: Inventory and Accounts Receivable

The two major audit tasks in the sales and procurement cycle are (1) confirmation of accounts receivable, conducted by contacting customers and third parties, and (2) count of physical inventory, which I assume is conducted through a one-shot physical count conducted at fiscal year-end (12–31). In both cases, a full dataset of all “real-world” records is generated, with the assumption that the auditor will take a sample of the clients recorded transactions and obtain, through confirmation or counting, these “real-world” values. This will require sampling of client records to select a subset of transactions to confirm, or a set of SKUs to count, and then left_joining the sample with these “real-world” values, which simulates the audit work that was done.

```
#####
## 12 - Simulation of completed audit work on Physical Inventory Counts
##

library(tidyverse)
library(lubridate)

perpetual_inventory_ledger$date <- as_date(perpetual_inventory_ledger$date)

working_perpetual <- ## get the unit_cost from the purchase journal
  left_join(perpetual_inventory_ledger, purchase_journal[,1:5], by=c("sku")) %>%
  group_by(sku) %>%
  arrange(date) %>%
  filter(date<=fyear_end)

real_world_ye_inventory <-
  year_end_inventory <-
    working_perpetual %>%
    as.data.frame() %>%
    filter(date<=fyear_end) %>%
    group_by(sku) %>%
    arrange(date) %>%
    slice(n()) %>%
    ungroup %>%
    select(last_transaction_date=date, sku, ye_stock_on_hand=po_count, unit_cost)

real_world_ye_inventory$count_exception <-
```

```

real_world_ye_inventory$exception <-
  real_world_ye_inventory$count_exception <-
    rbinom(nrow(real_world_ye_inventory), 1, .1) # 10% error rate

exception <- c("Obsolete_markdown" ,
               "Damaged",
               "Misclassified",
               "Refurbished_discount",
               "Open_box")

real_world_ye_inventory$actual_unit_market <- real_world_ye_inventory$unit_cost
real_world_ye_inventory$exception <- "No exception, count is accurate"

for(i in 1:nrow(real_world_ye_inventory)){
  if(real_world_ye_inventory$count_exception[i] == 1) {
    real_world_ye_inventory$actual_unit_market[i] <-
      real_world_ye_inventory$unit_cost[i] *
        (1 - runif(1,0,1))

    real_world_ye_inventory$exception[i] <- sample(exception,
                                                    size = 1,
                                                    replace=T)
  }
}

real_world_ye_inventory <- as.data.frame(real_world_ye_inventory)

```

Accounts Receivable Confirmations

```

#####
## 13 - Simulation of completed audit work on
## Accounts Receivable Confirmations
#####

library(tidyverse)

library(tidyverse)
library(lubridate)

fyear_end_ar_ledger <-
  sales_journal %>%
  filter(cash_not_ar == 0) %>%
  filter(collection_date > fyear_end) %>%
  select(customer_no, invoice_no, amount = sales_extended, shipper_no, shipper_date)

neg_confirm <-
  c("No Response",
    "Unable to confirm balance",
    "Customer will not pay shipping",
    "Customer rejects charge",
    "Customer claims to have returned the goods",
    "Customer has paid in full")

```

```

"Customer cannot pay",
"Customer claims fraction of the amount charged")

real_world_fyear_end_ar_ledger <- fyear_end_ar_ledger
real_world_fyear_end_ar_ledger$confirm_exception <- rbinom(nrow(fyear_end_ar_ledger), 1, .1) # 10% error rate
real_world_fyear_end_ar_ledger$confirm_response <- "Confirmed, Balance OK"
real_world_fyear_end_ar_ledger$confirm_pct <- 1

for(i in 1:nrow(fyear_end_ar_ledger)){
  if(real_world_fyear_end_ar_ledger$confirm_exception[i] == 1) {
    real_world_fyear_end_ar_ledger$confirm_response[i] <-
      sample(neg_confirm, size = 1, replace=T)
    real_world_fyear_end_ar_ledger$confirm_pct[i] <-
      runif(1,.5,.99)
  }
}

real_world_fyear_end_ar_ledger <- as.data.frame(real_world_fyear_end_ar_ledger)

```

Accounting Files for Audit

The prior code generated a set of files that contain a complete set of records of transactions, triggers, and subsequent events for our simulated sales and procurement cycle. These do not appear in this form in the real world. Rather the accounting statements that auditors will find at the client's site will be organized differently. The following code reorganized the data we generated previously into files that would be typical of those found in an audit. These are written to .CSV files.

```

#####
## 14 - separate po_inv and sales_journal files into component ledgers and journals
##       write these into CSV files in the 'audit_files' directory
##
#####

library(tidyverse)

## Set the directory for the new files (modify the path as needed)
default_dir <- "/home/westland/audit_analytics_book/audit_simulated_files/"

if (file.exists(default_dir)){
  setwd(default_dir)
} else {
  dir.create(default_dir)
  setwd(default_dir)
}

# sales

write.csv(real_world_cash_sales, "real_world_cash_sales.csv")
write.csv(real_world_credit_sales, "real_world_credit_sales.csv")
write.csv(sales_journal[,1:15], "sales_journal.csv")

write.csv(real_world_collections[,1:9], "real_world_collections.csv")
write.csv(collections[,1:9], "collections_journal.csv")
write.csv(deposit_daily, "deposit_daily.csv")

```

```

write.csv(fyear_end_ar_ledger, "fyear_end_ar_ledger.csv")
write.csv(real_world_fyear_end_ar_ledger, "real_world_fyear_end_ar_ledger.csv")
write.csv(daily_ar_balance, "daily_ar_balance.csv")

cr_limit %>%
  as.data.frame() %>%
write.csv("customer_credit_limits.csv")

write.csv(shipments_journal, "shipments_journal.csv")

# expenses

write.csv(expenditures, "expenditures.csv")

# inventory

write.csv(fyear_begin_inventory_ledger[,1:4], "fyear_begin_inventory_ledger.csv")
write.csv(real_world_ye_inventory, "real_world_ye_inventory.csv")
write.csv(perpetual_inventory_ledger, "perpetual_inventory_ledger")

# purchases

write.csv(purchase_journal[,1:5], "purchase_journal.csv")
write.csv(ap_ledger, "ap_ledger.csv")
write.csv(receiver_journal, "receiver_journal.csv")
write.csv(disbursement_journal, "disbursement_journal.csv")

#####
##      END OF PROGRAM CODE
##
#####

```

Auditing with Simulated Accounting Transactions

Once the auditor has generated datasets using the previous code, these can be entered to existing accounting software and platforms for testing. Clearly these systems need to be “sandboxed” (i.e., isolated from critical system resources and software) during audit testing. Effective use of simulated transactions demands some prediction of risks and faults that might occur.

Computer systems are most reliable when transactions conform to formal, expected parameters. This is because the programmers of these systems are given specifications based on formal definitions of the system and expected inputs. Exceptions and errors will test the internal control subsystems that prevent or detect divergence from formal specifications. Since error correction is a highly error-prone process, generating two or more errors for every faulty correction, the ability to repeatedly test control systems is beneficial.

References

- Ball, Ray, and Ross Watts. 1972. Some Time Series Properties of Accounting Income. *The Journal of Finance* 27 (3): 663–681.
 Cochran, William G. 1977. The Estimation of Sample Size. *Sampling Techniques* 3: 72–90.
 Fienberg, Stephen E., John Neter, and Robert A. Leitch. 1977. Estimating the Total Overstatement Error in Accounting Populations. *Journal of the American Statistical Association* 72 (358): 295–302.
 Garstka, Stanley J. 1977. Models for Computing Upper Error Limits in Dollar-Unit Sampling. *Journal of Accounting Research* 179–192.

- Greene, William H. 1994. Accounting for Excess Zeros and Sample Selection in Poisson and Negative Binomial Regression Models.
- Keenan, Michael R., and Paul G. Kotula. 2004. Accounting for Poisson Noise in the Multivariate Analysis of Tof-Sims Spectrum Images. *Surface and Interface Analysis: An International Journal Devoted to the Development and Application of Techniques for the Analysis of Surfaces, Interfaces and Thin Films* 36(3): 203–212.
- Kish, Leslie. 1995. *Survey Sampling*, Vol. 60. New York: Wiley-Interscience.
- Lohr, Sharon L. 1999. *Sampling: Design and Analysis*. Brooks.
- Repin, Sergey, Stefan Sauter, and Anton Smolianski. 2004. A Posteriori Error Estimation for the Poisson Equation with Mixed Dirichlet/Neumann Boundary Conditions. *Journal of Computational and Applied Mathematics* 164: 601–612.
- Snedecor, George W., and Witiam G. Cochran. 1989. *Statistical Methods*, 8th ed. Ames: Iowa State University Press Iowa 54: 71–82.
- Westland, J.C. 2017. An Empirical Investigation of Analytical Procedures Using Mixture Distributions. *Intelligent Systems in Accounting, Finance and Management* 24(4): 111–124.
- Westland, J. Christopher, and Eric Wing Kuen See-To. 2007. The Short-Run Price-Performance Dynamics of Microcomputer Technologies. *Research Policy* 36(5): 591–604.

Index

A

Abbreviated balance sheet, 101

Acceptance sampling, 49, 84, 142, 145, 148, 157, 164, 204

Accomptants, 7

Accountancy, 12

Accounting

Al-Khwarizmi's Algebra, 4–5

assumptions, 14

Chart of Accounts, 14

commodities, 3

computers, 3

constraints, 15

data types, 24–25

definition, 42

entries and document files, 15

financial, 12–13

GAAP, 14

industrial revolution, 6–7

marketplaces, 3

and modern auditing, 7

principles, 14–15

public, 8–9

renaissance, 6

signs, 3

theory, 14

Accounting cycles

accounting transactions, 64

audit activities, 310

audit procedures, 41

audit tasks, 149

conversion cycle, 49

definition, 41

expenditure cycle, 49, 310

financial reports, 49

financing, 49

fixed assets, 49, 310

revenue cycle, 49

role, 310

sales and procurement cycles, 311

treasury cycle, 310

Accounting errors, 228

Accounting Principles Board (APB), 9, 12

Accounting Research Bulletins (ARBs), 8

Accounting standards, 14

Accounting transactions, 309

accounting cycles, 41, 49

accounting systems by design, 45

assumptions

document generation, 313–314

document-updating events, 312–313

parameters, 316–317

procurement cycle, 313

reporting, 313

audit evidence, 60–61

auditing, 337

audit procedures, 41

audit tests, 42–43

cash and bank deposits, 319–320

characteristics, 44–45

Chart of Accounts, 42

CLT, 46

components, 42

convenience transactions, 45

couching institutional language, 47–48

distribution of transaction values, 45

dollar-unit sampling, 44

and double-entry bookkeeping, 44

FTG theorem, 46

GAAP, 42

inventory and purchase order, 320–321

journal entries, 44

machine learning, 58–60

material errors, 46

metrics and estimates (*see Metrics and estimates (accounting transactions)*)

raw data, 44

real-world events, 42

real-world properties, 311

sales journal, 317–319

samples and populations, 48–49

single monetary unit, 44

stakeholders, 42

substantive testing, 50

transaction streams, 42, 45

value × quantity, 46–47

working with dates, 43–44

Account payable, 49, 320, 3320

Accounts balance, 15

Accounts receivable (A/R), 48, 325–328

aging, 327–328

balance, 211

Accounts receivable auditing

acceptance sampling, 204–206

annual audits, 200

confirmation, 200, 206

debit accountability, 200

- Accounts receivable auditing (*cont.*)
 experimental design, 207
 footing and agreeing, 200–202
 procedures (*see* Audit programs)
 supporting documentation, 202–204
- Accounts receivable confirmation
 aged accounts receivable report, 206
 assessment, 206
 auditing standards, 207
 CICA, 207
 confirmation evidence, 207
 customer's subsidiary ledger, 206
 GAAP, 206
 McKesson–Robbins fraud, 206
 non-response, 210
 normal grouping, 206
 requests, 206
- Accurate dollars, 228
- Activities, 13
- Agree to trial balance, 200
- Al-Khwarizmi's Algebra
 double-entry, 4–5
- Allowance for doubtful accounts, 214, 224
- American Institute of Accountants (AIA), 8
- American Institute of Certified Public Accountants (AICPA), 8, 12, 47, 49, 227
 fixed-interval sampling, 155
 GAAAS, 153–154
 judgmental sampling, 155
 random sampling, 155
 risks, 152–153
 sampling approaches, 152–153
- Analytical review
 AICPA, 89
 institutional context
 AICPA, 89–90
 guidelines, 91
 risk assessment procedures, 90
 statistical terms, 90
 internet resources, 102–103
 procedures, 89
- Annual audits procedures, 231–232
- “Annual Risk Assessment” process, 309
- Application programming interface (API), 106–107, 123
- Arabic numerals, 5
- A/R balance, 213
- ARIMA model, 218, 219
- Arrays, 35
- Artificial intelligence, 3, 22
- Athenian silver drachma, 3
- Attribute sampling, 142, 145
 factors, 152
 interim compliance testing, 167
 for transaction error rate, 164
 transaction-unit samples, 166
 with t-tests, 170–173
 “urn model” formula, 170
- Audit analytics, 1–3
 accounting data types, 24–25
 binary data, 28–30
 business and data analytics, 19–24
 categorical data, 26–28
 data storage and retrieval, 33–38
 numerical vs. categorical, 25–26
 ordinal (ordered factor) data, 30–32
 R packages, 38
- Audit budget
 sampling, 81
 technical metrics, 81
 technical sampling structure, 81
- Audit cycle, 159–160
- Audit evidence
 Fisher information, 61
 log-likelihood, 60–61
 the score, 61
- Auditing
 and accounting, 3–9, 85
 around the computer, 309
 budgets and risk reduction, 77
 characteristics, 85
 component, 43
 computers, 1–3
 computer systems, 337
 data collection and processing, 315
 emerging technologies (*see* Information technology and intangible assets)
 failures, 85
 financial statements, 1, 63
 governmental accounting, 1
 planning and budgets, 86
 risk assessment, 63
 risk assessment matrix dashboard, 78
 risk-based, 1
 R packages, 17–18
 sample size determination, 315
 service organizations (*see* Service organizations)
 with simulated accounting transactions, 337
 staffing and budgets, 76
 testing methods and reporting practices, 1
 through the computer, 309
 in USA, 63
- Auditing framework, 191
- Auditing service organizations
 accounting cycles, 310
 accounting files, 336–337
 accounting transactions, 315–317
 accounts payable, 325
 accounts receivable, 325–328
 accounts receivable confirmations, 335–336
 audit tasks, 310, 334–335
 cash and bank deposits, 319–320
 customer credit limits, 325
 document generation, 313–314
 document-updating events, 312–313
 duplication, 321–334
 employee expenditures, 328–329
 entities, 313
 generation, 310
 inventory, 321–325
 inventory related accounts, 325
 inventory replenishment, 320–321
 monetary errors, 329–334
 omissions, 329–334
 outstanding accounts, 325
 perpetual inventory, 325
 procurement cycle, 313
 progeny, 309
 reporting, 313
 sales and procurement cycle databases, 310
 sales journal, 317–319
 simulated accounting transactions, 337
 simulation, 311

test decks, 309
transactions, 329–334
Auditing Standards (AS), 1453
Auditors opinion, 191
Audit procedures, 48
 interim compliance tests, 41
 PCAOB's AS, 41
 planning and risk assessment phase, 41
 substantive tests, 41
Audit programs
 accounting cycles, 159–160
 accounting transaction distributions, 158–159
 AICPA, 142–143, 152–154
 allowance for doubtful accounts, 214–217
 audit cycle, 159–160
 confirmation process steps, 208–210
 confirmation requests, 207
 context of auditing, 160–161
 error estimation, 211–212
 field and quasi-experiments, 141
 forecasting accounts receivable collections, 221–223
 GLM-exponential regression, 217–218
 information technology, 160–161
 learning models, 149–150
 materiality, 151
 natural experiments, 141
 non-response to confirmation, 210
 non-statistical sampling, 157–158
 not expected to confirmation, 210
 poisson data, 148
 post-confirmation tests, 213–214
 product, 161–162
 “random” assignment, 141
 randomness, 141
 risk, 151–152
 sample size, 142
 sampling base/metric, 141
 statistical decision framework, 148–149
 statistical framework, 149
 statistical sample, 141
 time-series forecasting, 218–221
 transaction/record sampling, 141
 types, 142
 year-end procedures, 208
Audit risk, 48, 64
Audit tasks
 accounting files, 336–337
 accounts receivable, 336
 accounts receivable confirmations, 334–335
 physical inventory, 334
 “real-world” values, 334
 sales and procurement cycle, 334
Autoencoders
 H2O for analysis, 255
h2o package, 259
 neural network, 254
 PCA decomposition, 255
 pre-trained supervised model, 268–270
 security breaches, 254
 self-supervised machine learning models, 254
 tensorflow implementation, 257–259
 Tensorflow/Keras for reference, 255

B

Balance sheet, 13, 67–69, 72, 101
The Bankruptcy Act of 1831, 7
Benford's Law, 289
 code chunks, 290
 expenditures transaction file, 291
 occurrence probabilities, 289
Bernoulli distribution, 53
Bernoulli process, 53
Bernstein-von Mises theorem, 61
Big Data, 11
Binary data, 25, 28–29
Binomial distribution, 53
Blockchains
 concept, 281
 cost, 286
 hash function, 282
 transaction records, 2983–285
 validity, 281
Bootstrap and resampling, 54
British Companies Acts, 7, 8
Business analytics, 19–24

C

Canadian Institute of Chartered Accountants (CICA), 207
Capital, 16
Capital Asset Pricing Model (CAPM), 48
Capital/net worth, 13
Cash and bank deposits, 319–320
Cash flow statements, 13
Cell/random-interval sampling, 155
Cells in Excel, 2
Central Limit Theorem (CLT), 46, 48, 54
Certified public accountants (CPA), 159
Chart of Accounts, 14, 16, 24, 42
The Clayton Act of 1913, 8
Cloud computing, 1
Cloud servers, 12
Cohen's power analysis, 203
Collection transactions, 170–173
Committee on Accounting Procedure (CAP), 8–9
Commodities, 3
Common accounting methods, 230
Common inventory audit task, 242–243
Common technical metrics, 93–94
The Companies Act of 1862, 7
Company's financial health
 financial ratio, 92–93
 technical metrics, 92
Comprehensive R Archive Network (CRAN), 38
Compute-intensive search algorithms, 58
Computer analytic workpaper support, 232–234
Computer forensics, 287
 IT forensics marketspace, 288
 software, 288
 steps, 288
Computerized accounting system, 230
Conditional sampling, 155
Confidence level, 143
Confirmation process (audit program)
 accounts selection, 208

- Confirmation process (audit program) (*cont.*)
 accounts with credit balances, 208
 aged schedule, 208
 prepare and mail confirmation requests, 208–209
 responses to confirmation requests, 209
- Confirmation request, 210
- Congress's Government Accountability Office (GAO), 160
- Conservatism principle, 15
- Consistency principle, 15
- Continuous (interval, float, numeric) data, 26
- Continuous variable, 20
- Control risk (CR), 47
- Conversion cycle, 49
- Copper coins, 3
- Correlation coefficient, 52
- Cost accounting, 7
- Cost efficiency, 47
- Cost of goods sold (COGS), 229
- Cost principle, 14
- Couching institutional language, 47–48
- Credit entry, 15
- Credit limits, 325
- CSV file, 2
- Customer payment performance, 220
- Cutoff analysis, 235
- Cybercrime, 247
 criminal business, 287
 and forensics, 285
 types, 286–287
 White-collar crimes, 287
- Cycle counts, 230
- statisticians and data scientists, 33
 tensor, 33
 tibbles, 36
 vectors, 34
- Data tables, 36
 Data types, 24–25
 Date-time data, 43
 Debit accountability, 200
 Debit entry, 15
 Deep-learning, 22, 58
 Density and cumulative distribution, 53–54
 Descriptive information, 24
 Design of Audit Programs, 49
 Detection risk (DR), 48
 Direct labor analysis, 242
 Discovery sampling, 143–144
 compliance testing, 165
 cost control, 165
 for interim tests, 165
 minimum unacceptable rate, 167
 sample size
 and minimum acceptable error rate, 168
 and statistical confidence, 165
 sets, 82
 transaction-unit samples, 167
- Discrete (integer, count) data, 26
- Discrete *dollar-unit* amounts, 204
- Dividends, 13
- Document generation, 313–314
- Dollar-unit sampling (DUS), 44, 156, 228
- Double-entry, 4, 5, 16, 44
- Duplicates and omissions, 236–238

D

- Data analytics, 2
 and business, 24
 continuous variable, 20
- EDA (*see* Exploratory data analysis (EDA))
- exploratory statistics, 19
- games of chance, 19
- governance, 19
- ImageNet, 24
- income distribution over time, 20–21
- income, employees and sales, 21, 22
- income *vs.* other variables, 20–21
- industry statistics, 19
- learning, 22–23
- machine learning, 19, 24
- nested set, 22
- R's plotluck package, 19–20
- Data frames, 36–37
- Data management, 12
- Dataset inspection and manipulation, 37–38
- Data storage and retrieval
 archival storage, 33
 arrays, 35–36
 data frames, 36
 dataset inspection and manipulation, 37–38
 data tables, 36
 factors, 37
 growth, 33
 lists, 36–37
 machine learning, 33
 matrices, 34–35
 matrix data, 33
 R language, 34

E

- Earnings per share (EPS), 13
- Earnings power, 8
- Eclipse IDE, 2
- Economic activities, 24
- EDGAR database, 94–99
- Edge computers, 12
- Efficient markets hypothesis (EMH), 48
- Employee expenditures, 328329
- Entity's contractual activities, 24
- Equal-field system, 4
- Error checking, 16
- Expenditure cycle, 49
- Exploratory data analysis (EDA)
 complements, 19
 computing power and software, 19
 model testing and prediction, 19
- Exploratory statistics, 19
- Exploratory substantive tests
 credit sales journal, 192
 kable and tidyverse packages, 193
 "mat=TRUE" parameter, 193
psych, 192
 R functions, 192
 sales statistics, 193
 sales transactions, 191
 skimr function, 193
 statistical function, 192
 summarization tools, 192
- Trial Balance, 191
- Exponential distribution, 53

F

- Factors, 37
- Fama–French risk factors, 274–279
- Federal Trade Commission (FTC), 8
- Fibonacci's *Liber Abbaci*, 5, 6
- Financial accounting
 - AICPA, 12
 - balance sheet, 13
 - cash flow statements, 13
 - communication, 12
 - FASB, 12
 - GAAP, 12
 - GASB, 12
 - IFRS, 12
 - income statement, 13
 - language of business, 12
 - management accounting, 12
 - PCAOB, 12
 - presentation, 12
 - principles of accountancy, 12
- Financial Accounting Standards Board (FASB), 12, 151, 154
- Financial accounts auditing, 191
- Financial audits, 9, 10
- Financial engineering, 10
- Financial information, 65–73
 - ad hoc analyse, 71
 - balance sheet, 69, 71
 - EDGAR database, 94
 - SEC's XBRL databases, 65
 - XBRL file, 65
- Financial information sector, 9
- Financing activities, 13
- Finished goods cost analysis, 242
- First-in, first-out (FIFO), 230
- Fisher information, 61
- Fisher–Tippett–Gnedenko (FTG) theorem, 46
- Fixed assets, 13, 49
- Fixed-interval sampling, 155
- Flying cash, 5
- Foot, 200, 201
- Forecasting accounts receivable collections, 221–223
- Foreign Corrupt Practices Act, 9
- Forensic analytics, 289
- FTK Imager, 288
- Full disclosure principle, 15

G

- Gamma distribution, 217, 228
- Gamma family, 217
- Gaussian distributions, 47
- General Accounting Office, 160
- Generalized linear model, 217
- Generally accepted accounting principles (GAAPs), 8, 12, 14, 42, 148, 151, 163, 191
- Generally accepted accounting standards (GAASs), 153–154, 161, 163, 191, 253
- The Glass–Steagall Act, 8
- GLM-exponential regression, 217–218
- GLM model, 217, 218
- Google's TensorFlow, 177
- Governmental Accounting Standards Board (GASB), 12
- Graph (or network) data structures, 38
- Guesses, 231

H

- Hash function, 281, 282
- Hash value, 281, 282, 284, 285
- The Hawley–Smoot Tariff Act, 8
- High-value items audits, 240–241
- Hindu–Arabic numerals, 4
- Hindu–Arabic system, 5
- Horvitz–Thompson estimator, 213
- Hypothesis tests
 - significance tests, 54
 - transaction/record sampling, 157
 - types of errors, 157

I

- Idiosyncratic tests, 225
- ImageNet, 24
- Income statement, 13
- Indian numerals, 4
- Industrial revolution, 6–7
- Information technology and intangible assets
 - auditors, 10
 - Big Data, 11
 - 'Big Four' accounting firms, 9
 - data management, 12
 - enrollments in audit degree programs, 9
 - financial accounting, 12
 - financial audits, 9, 10
 - financial control, 10
 - financial information sector, 9
 - global stock of data assets, 11
 - graduate degrees, 9
 - growth of data, 11
 - stock market, 10
 - Tobin's q (market cap/replacement cost), 10
- Inherent risk (IR), 48

Intelligence scanning

- business intelligence, 115
- cloud-based information services, 127
- financial and non-financial data, 115
- newsAPI package, 130–131
- qualitative data, 115
- R language, 139
- stream, 128–129, 131–133
- tidy data
 - advantage, 121
 - bind_rows(), 122
 - bing and NEC lexicons, 121
 - dictionary-based methods, 117
 - domain-specific sentiment lexicons, 117
 - emotional content, 115
 - get_sentiments(), 117–118
 - human readers approach, 115
 - integer division, 119–121
 - one-token-per-row structure, 115
 - positive and negative words, 118–119
 - sentiments dataset, 116
 - unigrams, 116
 - XBRl format, 117–118
- Interactive development environments (IDEs), 2
- Interim compliance tests, 41
 - collection transactions, 173–177
 - discovery sampling, 167
 - GAAP, 163

- Interim compliance tests (*cont.*)
 machine learning models (*see* Machine learning)
 RAM and substantive tests, 163
 sampling, methodologies, 164
 SAS 114 letter, 163–164
- Interim tests, 163
- Internal Revenue Service, 160
- International Financial Reporting Standards (IFRS), 12
- International Standards on Auditing (ISA), 191
- Interquartile range (IQR), 52
- Interstate Commerce Commission (ICC) Act of 1887, 8
- Inventory, 229
 allowances, 241–242
 layers, 243
 perpetual, 325
 and purchase orders, 320–321
 replenishment, 320
 sales and procurement cycles, 321–325
- Inventory audit procedures
 company's general ledger, 234–235
 computer analytic workpaper support, 232–234
 cutoff analysis, 235–236
 duplicates and omissions, 236–238
 high-value items, 240–241
 inventory allowances, 241–242
 perpetual inventory ledger, 238–239
- Investing activities, 13
- J**
 Journal entries, 44
- Judgmental sampling, 155
- Jupyter notebooks, 2
- K**
 Kraft–McMillan theorem, 58
- L**
 The language of business, 12
 Last-in, first-out (LIFO), 230
 Learning models, 42–43, 149–150
 Liabilities, 13
Liber Abbaci (book), 5
 Linear regression, 55
 LOCOM test, 239–240
 Logical units, 153
 Logit model, 55
 London-based International Financial Reporting Standards, 12
 Lower of Cost or Market (LOCOM), 239
Lubridate package, 43–44
- M**
 Machine learning, 19, 22–24
 accounting datasets, 189
 artificial intelligence, 177
 autoencoders and unbalanced datasets, 180
 compute-intensive search algorithms, 58
 computing decision thresholds, 189
 data storage and retrieval, 33
 decision strategies, 58
 and deep-learning, 58
 deep-learning autoencoder, 177
 deep-learning “distillation” process, 187
 “eyeballing” systemic weaknesses, 178
- Google's TensorFlow, 177
 inference, 58
 internal control systems effectiveness, 177
 Kraft–McMillan theorem, 58
 mean-squared-error (MSE), 184–186
 MNIST data, 60
 models, 219
 National Institute of Standards dataset, 58–60
 recurrent neural networks, 181
 R package, 187, 188
 squared-error loss function, 58
 statistics, 24
 TensorFlow, 177
 traditional parametric statistics, 181
- Mainframe computers, 1
- Management accounting, 12
- Management letter, 163
- Market efficiency, 48
- Matching principle, 15
- Material errors, 46
- Materiality, 15, 151
- Matrices, 34–35
- Measurements, 24
- Metrics and estimates (accounting transactions)
 Bernoulli distribution, 53
 bootstrap and resampling, 54
 CLT, 54
 count, 51
 dataset, 51
 degrees of freedom (df), 52
 density and cumulative distribution, 53–54
 deviations, 52
 hypothesis tests (significance tests), 54
 information, 51
 linear regression, 55–58
 mean, 51
 mean absolute deviation (mad), 52
 median, 51
 order statistics, 52
 outlier, 51
 overfitting, 55
 Poisson distribution, 53
 random sampling and sample bias, 54
 range, 52
 Skew and Kurtosis, 52
 standard deviation, 52
 standard error, 54
 statistics *vs.* data science, 50
 “sum” command in R, 50
 tidy function, 51
 trimmed mean, 51
 variance, 51
- Microcomputers, 1
- Microsoft Excel, 2
- Minimum acceptable error rate, 143
- Modern computing systems, 309
- Monetary unit sampling (MUS), 48, 156–157, 227–228
- N**
 National Institute of Standards dataset, 58–60
 Natural language processing (NLP) modeling, 107–108
 Near-cash, 13
 NetBeans IDE, 2
 Neural Structured Learning (NSL), 189
 Nominal accounts, 4, 16

Noncurrent assets, 13
 Normal (Gaussian) distribution, 52
 Normal distribution probability model, 146

O
 Objectivity principle, 15
 Obsolete inventory, 241
 Operating activities, 13
 Optimal stopping, 158
 Order statistics, 52

Ordinal (ordered factor) data, 30–32
 “Out-of-control” collections system, 177
 Out-of-control error rate, 142, 144

P
 Package installation, 17
 Periodic inventory, 229, 325
 Permanent accounts, 4
 Perpetual inventory ledger, 238
 Perpetual inventory systems, 229
 Physical counts *vs.* cycle counts, 231
 Physical inventory, 230–231
 Physical inventory auditing
 counting inventory, 229–231
 count *vs.* cycle counts, 231
 distributors, 229
 inventory audit (*see* Inventory audit procedures)
 inventory systems, 230
 periodic inventory systems, 229
 perpetual inventory systems, 229
 Physical inventory (counting) process, 230–231
 Point-of-sale inventory systems, 229
 Poisson distribution, 53
 Poisson–Negative Binomial distribution, 229
 Poisson regression models, 229
 Power-driven machines, 6
 PPS sampling, 213
 Principal components analysis (PCA), 181
 Probability distribution, 52, 158
 Probability proportional to size (PPS)
 auditors, 227
 audit setting, 226
 design and computation, 213
 Hansen–Hurvitz estimator, 225, 226
 materiality, 227
 monetary unit sampling, 225
 population, 225
 PPSWOR, 225
 R’s *TeachingSampling*, 225
 R *survey* package, 213
 sampling, 152, 225
 tainting, 227
 tolerable error, 227

Proof-of-work (PoW), 282–283
 Public accounting, 8–9
 Public Company Accounting Oversight Board (PCAOB), 12, 41, 160, 246
 Python, 2

Q
 Qin coins, 3
 Qin Dynasty period, 3
 QQ-plot, 52, 216

R
 Random sampling, 155
 and sample bias, 54
 Rates of erroneous transactions, 84, 171, 173, 175
 Rates of monetary error, 171, 173, 175
 Real accounts, 4, 16
 Remote storage, 1
 Renaissance, 6
 Revenue cycle, 49
 Revenue recognition principle, 14
 R functions, 192
 Risk, 151–152
 Risk assessment matrix (RAM), 64, 77, 151–152
 client–server systems, 78
 KPMG, 77
 reactive programming, 78
 R Studio, 81
 server side, 79
 Shiny, 78
 and substantive tests, 163
 Risk-based auditing, 1
 Royal Society, 6
 R packages, 38
 auditing, 16–17
 foundations of audit analytics, 38
 R’s *plotluck* package, 19–20
 R’s *pwr* package, 203
 R statistical language, 106
 R-Studio, 2
 R *survey* package, 213
 R workspace, 200

S
 Sales and procurement cycles, 310, 311, 321
 Sales journal, 317–319
 Sample size determination, 158, 315
 Sarbanes–Oxley (SOX) Act
 anomaly detection, 267–268
 audit opinion, 162
 auditor responsibility, 42
 autoencoders, 254–255
 and credit card datasets, 279
 effectiveness (*see* SOX effectiveness)
 Fama–French risk measures, machine learning, 273–274
 Foreign Corrupt Practices Act, 9
 fraud and external threats, 245
 guidance for implementation, 246
 information systems, 245
 internal control, 245
 linear regression model, 55
 management-only assessments, 247
 management’s and auditor’s opinion, 245–246
 precision-recall/sensitivity (recall)-specificity curves, 270–273
 preprocessing, 255–257
 R’s *H2O* implementation, 259–267
 Sarbanes–Oxley data, 55–58
 SAS (statistical software), 2
 SAS 114 letter, 163–164
 Scatterplot, 52
 Scientific Management, 7
 The score (or informant), 61
 Securities Act of 1933, 8
 Securities and Exchange Commission (SEC) Act, 8
 Securities Exchange Act of 1934, 154
 Service Organization Control (SOC), 309

- Service organizations
 accounting cycles (*see* Accounting cycles)
 client's ICFR, 309
 R-base commands, 311
 sales and procurement cycles, 310, 311
 sales journal, 317–319
 SSAE 18 auditing, 309
- Shareholders' equity, 13
 Sherman Act of 1890, 8
 Shiny, 78
 Signal processing models, 219
 Single monetary unit, 44
 S language, 2
 Social networks
 motors and auto industry, 125–126
 Roland Musical Instruments and Industry, 126–127
 Twitter, 123–125
- The Social Security Act, 8
 Software as a service (SaaS) infrastructure, 41
 SOX effectiveness
 academic research, 246–247
 attestations effects, 250
 audit and non-audit fees, 249
 auditor attestations, 247
 and breach datasets, 249
 building R-code, 249
 compliance cost and information content, 246
 financial reporting areas, 248
 industry sources, 247
 operational areas, 248
 rate of occurrence, security breaches, 250
 security breaches, 247
- SOX-Privacy Clearinghouse (PCH) dataset
 assessments, in automated and manual systems, 253
 automated controls and breaches, 253
 binomial models logit and probit, 251
 Fama–French risk factors, 274–279
 GAAS, 253
 internal control effectiveness, 254
 multicollinearity, 255
 self-reported material weaknesses, 251
- Spatial data structures, 38
 Spreadsheet programs, 2
 Squared-error loss function, 58
 Standard error, 54
 Statement on Standards for Attestation Engagements No. 18 (SSAE 18), 309
- Stock, 6
 Stock market, 10
 Stratification, 225
 Stratified samples/monetary unit sampling
 auxiliary variable/size measure, 225
 PPS, 225–227
 standard approach, 225
 taintings, 226–227
- Stratified sampling, 156
 Stringer's perspective, MUS, 227–228
 Substantive tests, 41, 50
 accounts receivable (*see* Accounts receivable auditing)
 audit program procedures (*see* Audit programs)
 exploratory, 191–195
 objectives, 191
 physical inventory audit (*see* Physical inventory audit)
 stratified samples/MUS, 225–229
 trial balance figures, 196–200
- Substantive year-end testing, 227
- T**
 Tainted dollars, 228
 Tainting, 227, 228
 Tang Dynasty, 3, 5
 Technical metrics
 computing, 99–100
 visualizations, 100–102
- Temporary accounts, 4
 Tensorflow, 177
 Tensorflow installation, 17
 Testamentary disposition, 4
 “Test decks”, 309
 Test error-prone items, 242
 Third-Party Vendor Management Program, 309
 Tibbles, 36
 Time (accounting), 24
 Time-series analysis, 224
 Time series data, 38
 Time-series forecasting, 218–221
 “Time to payment”, 215
 Tobin's q (market cap/replacement cost), 10
 Tolerable error, 227
 Touchstone, 3
 Trade flourished, 4
 Traditional accruals, 215
 Transaction cycle, 228
 Transaction/record sampling, 48
 Transactions
 omissions, duplications and monetary errors, 329–334
 statistical sampling, 1
- Trial balance, 16, 191, 196–200
 Tweedie distributions, 229
- U**
 Urm models, 143, 167, 173
 US Census data, 103–106
 Use Of Negative Form Of Confirmation Request, 207
- V**
 Vectors, 33
 Verifying Counts methods, 243
 Vocabulary-based vectorization, 108–114
- W**
 Web scraping
 movie reviews, 135–136
 product user forums, 137–139
 rvest package, 133
SelectorGadget, 134
 simple sentiment analysis, 134–135
 tabular data, 137
 Xpaths, 137
- Wedgwood's cost accounting, 7
 Weighted average, 230
 Wharton Research Data Service, 19
 Work-in-process (WIP)
 inventory, 232
 testing, 232
- Y**
 Yates–Grundy estimator, 213
- Z**
 Zero balance, 4