Audit Procedures and Accounting Cycles: 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 apping 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")


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

mdy("4/1/17")


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


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

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. 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.

```
library(tidyverse)
library(ggplot2)

sox_stats <-
read.csv("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_data
_types.csv")
transaction_value <- sox_stats[,"audit_fee"]/3740000
transaction_value[is.na(transaction_value)] <- 0
qplot(transaction_value, geom="histogram")
```

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

1. the distribution is left-bounded at zero
2. the distribution may be zero-inflated, i.e., it may have a substantial number of zero measurements
3. the distribution is multi-modal, 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 9th 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 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.

```
library(tidyverse)
library(ggplot2)


price <-  rpois(1000, 2)
quantity <-   rpois(1000, 10000)
value <- price*quantity
qplot(value, geom="histogram")
qplot(value, geom="density")
```

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("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_AP_d
isbursements_journal.csv")

summary(disburse)

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

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

```
library(Hmisc)

bank_fin <-
read.csv("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_yaho
o_fin.csv")

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 packates 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 formated table, using kableExtra

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

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("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_AP_d
isbursements_journal.csv")


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

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

Density and Cumulative Distribution: Probability 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 extention of summary statistics computes estimates of density and cumulative distribution functions for data, as the following example using our disbursements data will show.

```
disburse <-
read.csv("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_AP_d
isbursements_journal.csv")


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")
```

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 occurance of insider breaches with audit fees and auditor decisions on section 404 audits,using the Sarbanes-Oxley data.

```
lr <-
```

```
read.csv("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_data
_types.csv") %>%
  lm(formula=insd~audit_fee+effective_404)
summary(lr)
```

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 <-

read.csv("/home/westland/audit_analytics_book/aaa_chapters/tables/ch_2_data
_types.csv") %>%
   glm(formula=insd~audit_fee+effective_404,  family = "binomial")
summary(lgt)
plot(lgt)
```

Machine Learning Methods

Over the past decade, very useful extensions to 20th century statistical 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()
```

```
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
```

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

Machine learning is a vast and rapidly evolving field. It is increasingly 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.