Financial Statement Analysis

Business Computing Project

Robin, Johanna, Steve & Ziqiang

The following report will to analyze the current conditions of financial statement analysis and the opportunities our team saw to incorporate course concepts. The report will outline the source of our data, the code we use to analyze the financial statements, and the role of machine learning.

Shanghai Jiao Tong University

Table of Contents

Aim of the Project

In most countries, including China, Europe countries, and the US, quoted companies have to publicly provide accounting reports called financial statements. Investors use these reports to analyze the company's value and make decisions on whether they should invest in it or not. Currently this work is often done by hand. Accountants compute these ratios and analyze their computed data to determine investment decisions and a firm's place within its industry. In addition to the human capital costs associated with this manual process, there is the chance for computational errors.

The aim of our project is to construct software that analyzes and examines the financial statement of various companies in order to provide useful information for future investors and financial professionals conducting intrafirm analysis. Automating this process can give timely information and save company resources. The program uses components such as ratio analysis which are used to highlight if the company is a valuable investment target, as well as make future investment predictions. Furthermore, automating a part of the analysis eases the work of investors and financial analysts.

Advantages for Investors and Firms

This project might be very useful for investors because it helps to interpret the financial statements. This allows individuals to understand the strengths and weaknesses of a firm as well as its historical performance and current financial conditions. Key ratios such as the current ratio, return on assets, return on equity, and debt ratio analysis assesses and evaluates the quantitative information contained in a company's financial statement. Because these ratios and their derived financial statistics will be computed by a machine, human problems such as uncertainty, high frequency of errors, and analysis that is both time consuming and intense in nature will be solved through automation.

Therefore, these ratios provide information about various aspects of a company's operating and financial performance. The metrics the program can help to understand include its efficiency, liquidity, profitability, and solvency. This trend analysis may be useful for forecasting and planning future business activities while comparing industry peers. While

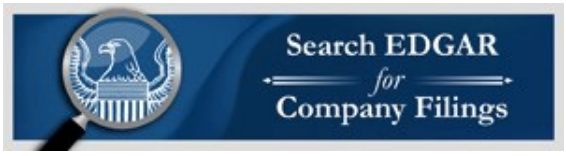
investors can use this tool to better understand the composition of the market, firms can use it to gain a greater understanding of their competitors. For example, if historical analysis of a smartphone manufacturer such as Oppo show high liquidity due to an above average current ratio, financial analysists working at similar companies such as Huawei can use this information in combination with Oppo's financial performance to determine if adopting a similar financial structure is a wise decision. Therefore, this tool is very helpful for future decision-making.

Data to Collect

One of the most important points of all research and analysis is the data that you use and the means by which you analyze it. In business, information is power, so it is very important that you can find information in both quantity and quality, as well as in a timely manner. Greater quantities of data in your sample will ensure the result of the analysis will be close to actual conditions. However, working with extensive data sets by hand is extremely time consuming, and through automation you can eliminate skimming and better understand the market.

Investors and firms must determine what information is relevant and necessary for their analysis, and then they can determine the best place to get it. In our case, because we want to analyze the US market, our platform that can give us the largest quantity of company information reported in an official and standard manner. Thus, we determined that the best option is to use the Security and Exchange Commission's (SEC) EDGAR database.

What is EDGAR?



The Electronic Data Gathering Analysis and Retrieval system (EDGAR) is a platform that allows firms and individuals access to more than 21 million financial filings. EDGAR has been available since 1934, however electronic publishing did not being until 1984, with the objective of helping investors.

EDGAR performs the automated collection, validation, indexing, acceptance, and forwarding of submissions by companies and others who are required by law to file forms with the U.S.

Securities and Exchange Commission (SEC). These include monthly filings, which often do not follow a standard format, 10-Q (quarterly) and 10-K documents, which are government mandate and more standard in nature.

Its primary objective is to increase the efficiency and fairness of the securities market for the benefit of investors, corporations, and the economy by accelerating the receipt, acceptance, dissemination, and analysis of time-sensitive corporate information filed with the agency.

The Challenges of Working with EDGAR Data

Databases are rarely without their quirks that make handling data difficult. For example, databases might not have all the information programmers seek (because it is not mandatory, or is not updated frequently). The SEC provides two ways to download and manipulate the data: compiled .txt database files & XBRL files, each with their own challenges. The .txt files are not real time company data, and are not industry standard like the XBRL files. However the plain text data is lighter easier to work with while coding. XBRL files are the opposite, increasing the timeliness of data by but increasing the complexity of its structure and increasing file sizes.

Additionally, some relevant information is not always published online. Some documents are either not required (they are voluntary, based on the discretion of the company) or not permitted to be filed electronically, so EDGAR is not a comprehensive resource. In order to move forward, our group had to acknowledge the limitations of each file format and EDGAR as a data source. We determined that the most relevant data was in the 10-Q and 10-K reports, with a more constant data structure and many of the key figures, this allowed for a constant environment to test and develop our code. Additionally, in order to create a more lightweight and development friendly process, we opted for the .txt formatted information.

The Development Process

By focusing on the .txt files in order to simplify the process and achieve a proof of concept build. The development process took about 4 weeks and was comprised of two main functional parts: Finding / cleaning the data, and computing / analyzing the data. Our code is broken into seven primary programs: The main, a program that handles downloads from the EDGAR database, one that unzips the data, a portion that sends this data to a database, a

section to compute various financial rations, another to analyze these ratios, and finally a section to publish a report in order to give data visualization. The structure of the two task-oriented parts and seven programs will be outlined below.

Part One: Finding and Cleaning the Data

The main begins by accepting inputs from users in order to determine what the program will be used for in this particular use case. Users can choose to alter the number of quarters, the ratios that will be required and computed later in the program, which SIC (reporting industry) code to use, whether the program needs to download and update new data, or if it needs to re-compute ratios. For example, in one use case, the user just wants to analyze data from Q4 of 2015, and the data is already downloaded. In this case they need only pull the ratios they want from the computed data.

When downloading new data, the main executes the EDGARDownload program. In this code, the URL is created by inserting user inputs into a predictable URL where only the year and quarter are variable. The program has error handling to ensure the whole file downloads, and provides user feedback in the event of success, timeouts, URL errors, HTTP errors, socket timeout errors, or incomplete reads of the data. After successfully reading a whole file, the program writes and saves it in a directory named after the target year. The file downloaded is a .zip, and is processed in the next section of the main.

The Unzip program simply unzips the file downloaded in the EDGARDownload program. The directory and target name are based on the previous information, and two functions are used. One unzips a given file while the other checks if files are unzipped and provides feedback, while feeding into the file unzipper if the files should, in fact, be unzipped.

Next the data is sent to a database for later analysis. In the DataToDatabase program, SQL is used in order to properly handle the data. The relevant fields and values are selected and because the program format is predictable, little formatting or other data manipulation is required in order to get the desired outputs. Three tables are generated for each data set: one for information on the report type and year, one providing the SIC and CIK (corporate identification key) information, and one containing the metrics for evaluation later. In the

ReportFilter program, the relevant information is extracted based on tags provided in the main.

Part Two: Computing and Analyzing the Data

Next ratios are computed in the ComputeRatios program. First, the tags input to the main determine which information will be computed. A dictionary stores the associated inputs for each ratio, and if a user inputs a ratio tag that the program cannot compute, the program handles the error by telling them they asked for an unsupported ratio and providing the list of supported ones. Next the program interfaces with the SQL database in order to convert the data back into something python can read. From there the appropriate ratio is computed using data from the database. This data is used in the second functional portion for analysis and reporting.

The data collected during the ratio computation is now fully accessible to Python for manipulation. The saved data from earlier is imported into the program and converted into a CSV during the Analysis program. We decided to use the KNN technique for our analysis. The program trains a model, performs Principal Component Analysis, and learns how to choose which data is desirable. The program returns the model of the KNN data and saves the input analysis after determining if the investment is classified as desirable or not. The graph output can make analysis very easy, as each point on the chart produced represents a particular company, and by tracking a company's position on the chart analysts can determine the desirability of the investment.

Machine Learning Techniques Used and Our Results

Many of the results of analysis tool are discussed above, including the outputs found in part two. Our program can predict with moderate accuracy (.66) the desirability of an investment. This is possible through the above Principle Component Analysis. An example output graph can be found in **Appendix A**. The bar on the graph represents the projection of a ratio's unit vector on the PCA plan, while each individual point on the graph represents a company. The program also outputs a list of desirable investments in text form.

Challenges with the Development Process

The development process was not free from challenges, and our current iteration has important limitations that must be resolved in any future versions, but the version we currently have represents a functional prototype that demonstrates proof of concept.

Some of the issue our project faces are based on our decisions. The data used can only be updated quarterly rather than real time due to our choice to use .txt files instead of XBRL files. Even still, the program can take hours to run if computing new ratios on new data. There are bottlenecks based on internet and computer processing speed, some of which are server side from the SEC's website. The current iteration of the program cannot process huge date ranges, but luckily if investors are using the program with forward-looking objectives, this is of little consequence, as the latest quarter's data can be processed in a reasonable amount of time.

The program also faces some issues with design. In the current design, python hands of to SQL and vice versa often; future iterations should rely less on SQL queries and more on pandas to filter through the correct data. Finally, the accuracy of our results was marginal, at only 66% accuracy, our PCA could stand to find improvements as well.

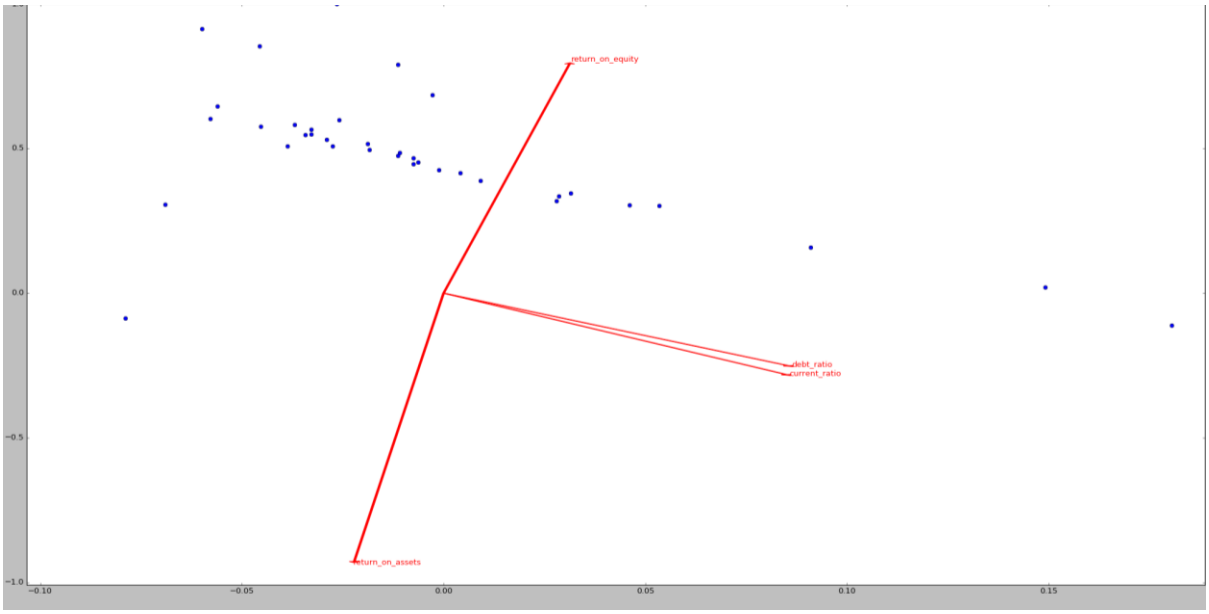
Moving Forward and Wrapping Up

Overall, the project is an important first step at automating a complex and tedious financial process. The business implications of an iteration of this program with tweaks to accuracy, structure, and design could help individuals and firms better analyze the desirability of an investment. As we move forward with this program, we plan to train it to increasingly large data sets and tweak some of the accuracy issues it currently suffers from. By improving the use of pandas and implementing a more verbose PCA, the next iteration of our program could be a potent predictor of firm performance.

This project has provided a valuable real world example of machine learning technique application. Moving forward as both professionals and programmers, the techniques employed in this project serve as a valuable baseline for making important financial impacts during our careers.

Appendices

Appendix A



References

U.S Securities and Exchange Commission ➔

- https://www.sec.gov/edgar/aboutedgar.htm