Predicting stock returns with financial ratios and sentiment analysis

Kevin Zhao, Jesse Abrahams, George Kuliner, Qingxue Yin, Abdool Bhollah

December 2020

1 Abstract

This paper studies whether predicted stock returns projected from financial ratios and financial statement analysis can be used to create a market beating portfolio. Ten financial ratios were predicted using historical data from past earnings as well as qualitative information in financial statements. Using a textual analysis algorithm, informative variables were created to analyze twitter sentiment which were applied in a multiple machine learning algorithm to predict future financial ratios. The predicted ratios and twitter sentiment were then used to predict the stock returns of the firms.

2 Introduction

Financial ratios are fundamentals calculated from financial statements and are primarily used by investors to inform their decision making and for comparison purposes. Much literature has been written analysing and confirming the predictive power of these ratios with regards to stock returns. These range from Fama-French (1988) [1], Kothari and Schanken (1997) [2], and Pontiff and Schall (1998) [3]. These studies found that book-to-market ratio (B/M) and dividend yield (DY) can strongly predict stock returns.

For our paper we've chosen to expand on these studies by combining ten financial ratios based on five categories; profitability, valuation, liquidity, solvency, and efficiency. The ratio's are: return on capital (ROC), return on equity (ROE), return on assets (ROA), price to book ratio (P/B), price to earnings ratio (P/E), average dividend yield (DY), cash ratio (CR), quick ratio (QR), asset to equity (A/E), and inventory turnover (IT).

We have specifically identified these because we hypothesise that the predictive power of financial ratios increases when combined in a multiple linear regression model, with each ratio being complementary and providing specific information on firms.

This study used 10 years of historical data and includes 100 companies that are tracked by the S&P500. This is an index of the stock of 500 of the largest

U.S. corporations, and is divided into 11 sectors. Our paper focuses on the 10 largest companies in ten of these sectors. These are: communication services, consumer discretionary consumer staples, energy, financials, health care, industrials, materials, technology, and utilities. For the purpose of our paper, we decided to omit companies from real estate as we believe their stock returns are less responsive to financial ratios and more on macro economic factors. Changes in the composition of the S&P500 are made periodically to reflect changes in the size of companies and in the economy. For example, the recent inclusion of electric car maker TESLA.

2.1 Financial ratios

Profitability ratios assess a company's ability to generate earnings as compared to its expenses and costs. ROC, ROE, and ROA fall under this category. ROC looks at how well a company can turn capital i.e debt, equity. into profitable investments. ROE measures the profits generated by a company to the investments made by the company's stockholders. ROA measures how profitable a company is compared to its total assets.

Valuation ratios measure the attractiveness of an investment compared to its fundamentals such as book value, dividends,etc. P/B, P/E, and DY fall under this category. P/B compares a firm's market capitalization or market value relative to its book value. P/E is the ratio of the company's share price to earnings per share. DY is the ratio of dividends paid annually to current share price.

Liquidity ratios measure the company's ability to meet its short-term obligations [4]. It can also refer to the ability of the firm to raise liquid assets i.e. cash, Treasury bills, etc. CR and QR fall under this category. CR measures the ability of a firm to repay its immediate obligations with only cash or cash equivalents. QR is similar to the CR, it is the ability of a firm to pay its debts without needing to sell its inventory or source additional financing.

Solvency ratios measure the ability of a company to meet its long-term obligations. A/E measures of the share of the company's assets is from shareholder's equity rather than debt.

Efficiency ratios measure how effective a firm is at managing its assets and liabilities. IT falls under this category, and shows the amount of times a company has sold and replaced inventory during a given period.

2.2 Sentiment analysis

Sentiment analysis is used to determine if certain language is positive, negative or neutral, it is a natural language processing technique. It can be used to gauge opinion on brands, political events, and is often used by companies to understand feedback on products. In this study, the purpose of the sentiment analysis was to determine if a relationship existed between an investor's point of view, financial ratios, and returns.

3 Data collection

Initially the sentiment analysis compared the results achieved using 10K/10Q documents and tweets. NLTK tokenizer and Textblob Natural Language Processing (NLP) libraries were used to analyze the text from SEC (Securities and Exchange Commission) EDGAR filings and tweets that were tagged with selected company tickers. Results from preliminary research on sentiment within 10Ks and through related articles, showed that sentiment extracted from these filings were positively biased, and thus not representative of the market. Overall, positively biased sentiment heavily skew results. This is because if the majority of 10Ks analyzed yield a positive score it is not then representative of the market since inherently some companies do experience loss [5].

To test the discrepancies or similarities between sentiment analysis through 10Ks and twitter, the study focused on providing scores to documents in order to determine if text is considered to be inherently "positive" or "negative". The scores applied to the provided text were derived from the NLTK tokenizer and Textblob Natural Language Processing libraries which have built functions to clean data, analyze it based on part of speech and content, and categorize it accordingly as positive or negative. Each part of text is then provided a score between -1 and 1. The negative values denote text with an overall "negative" tone attached to it whilst the opposite is true for a positively tagged text. The final score is calculated by averaging a set of -1 and 1s. The name of Textblob measurement used to calculate the sentiment scores is "polarity" which observes the tone of a phrase in any context independent of if the statement is considered to be an opinion or factual. The Textblob polarity score is derived from a deep learning model which used movie reviews as its training set; this provided the library with a means of providing scores to tokens in many different contexts since movie reviews are vast in the topics they cover and the tone they could be written in.

In order to demonstrate the differences that 10K and twitter sentiment analysis yield as well, we have provided data using the following companies from the technology sector; Apple (AAPL), Microsoft (MSFT), Amazon (AMZN) and Adobe (ADBE). The first 3 stocks are known as "FAANG" stocks which receive this distinction from their consistent success within the technology industry. The heralded status that the FAANG stocks have garnered is due to the fact that they make up a total of 15 percent of the S&P500. Specifically, Microsoft, Amazon, and Apple have each experienced share price increases that have exceeded 100 percent over the past 5 years accounting for the large rise in the index over the same period. On the other hand, Adobe's mark on the technology industry and the SP500 is not as readily apparent as the FAANG stocks. Its market cap is dwarfed by the over a trillion-dollar market caps of the largest companies within the index. Since investors in the market expect the performance of the FAANG stocks to be among the most successful within the index, it's likely that there wouldn't be any out of the ordinary response to growth within any of these companies since prosperity is to be expected with MSFT, AAPL, and AMZN. Moreover, Adobe's large increase in stock price in

Ticker	AAPL	MSFT	AMZN	ADBE
Score (10K)	34.0	43.0	39.0	27.0
Score (Twitter)	14.74	23.0	19.54	30.26
Change in Stock Price 2020	+72.53%	+38.61%	+73.26%	+52.50%
Change in Stock Price 2019	+86.16%	+55.26%	+23.03%	+45.78%
Change in Share Price Growth	-13.63%	-16.65%	+50.23%	+6.72%

Table 1: Discrepancies in twitter and 10K sentiment analysis

2020 will likely carry a more positive connotation along with it since there aren't nearly as high of expectations for Adobe's when compared to any of the FAANG securities.

The text observed were the 2020 10K filing for each and a sample of thousands of tweets over the same period. Since the amount of data received from twitter far outweighed the text from the 10K documents, the measurement for twitter used was to be a difference in the percentage of positive sentiment and negative sentiment tweets respectively. For 10K documents we took the difference between positive and negative tokens as a whole. The results of providing scores to each of these four companies is provided in Table 1.

The changes in stock price were introduced in order to provide perspective surrounding the differences in how the scores responded in relation to how we expected them to. Apple and Microsoft both recorded lower growth than in the previous year (2019), but still had largely positive 10K reports in the amount of positive tokens within the document. Microsoft had the lowest gain of 2020 and the largest difference in growth over the prior two years of the companies chosen, however its sentiment score from the 10Ks was highest of the four securities. As Adobe exceeded expectations the most among the group of stocks chosen, its sentiment should be higher than the rest of the securities since its performance was the most out of the ordinary.

Ultimately, the expected scores for the FAANG stocks were expected to be around the same values while Adobe would be expected to be higher since investor sentiment is directly influenced by the direction that a security is projected to move in. For this reason, the twitter sentiment was determined to likely be the better predictor of stock returns and financial ratios since there seems to be more no relation between any factors of the company and the scores provided by the 10Ks. The results from the 10Ks had MSFT with the highest sentiment score and ADBE with the lowest which had been the opposite of what had been desired due to the performances observed in other areas—here specifically highlighted in the change of their share prices.

The conclusion drawn from the sentiment analysis was that 10K documents beyond the accounting figures may not provide as much insight as originally thought. Companies have a tendency to use positive sounding vocabulary in their own reports that inevitably lead to similar sounding text, whereas the scope that a tweet can cover in relation to a ticker is much larger in its appli-

cability. This is more useful for analyzing investor point of view and thus likely more useful as a predictor of returns. A portion of the sentiment analysis code can be seen in Appendix 1.

For financial data, three data sources were considered: WRDS, Yahoo Financials, and Bloomberg Terminal. WRDS is a database hosted by the University of Pennsylvania, Wharton School of Business. It is easy to navigate but isn't compatible with python or any other library which would have made it hard if the models required additional data or modified data quickly.

Yahoo Finance is currently the most popular open-source financial data platform. It provides a variety of useful functions and an API that is compatible with python. We originally built a python script to retrieve information from yahoo API and convert the files into excel. However, after a few trials of the script, each run of the collection, which contained 10 years history of hundreds of tickers, roughly took at least one hour. Like WRDS, we predicted this would cause an issue later in the study, should the machine learning model require any modification of data input.

Bloomberg offers an add-in function that overcomes the downsides of the previous two methods. In addition, the terminal is one of the most authoritative and trustworthy sources of data on the market, and as such was chosen to be the source for our financial data retrieval. Bloomberg provides a convenient way of gathering huge data in a short time in exchange, through an excel add-in function that can retrieve data in real time. The function named "BDH" returns the historical data of a stock, and within it parameters of desired ratios and periods can be inputted. This process was repeated for all 100 stocks. A python script was then applied to the data retrieved from the Terminal to separate the dates and tickers. This was done so that the data could be run on R-Studio for the machine learning model. The script can be seen in Appendix 2.

4 Estimation and models

We can split the models used into two groups. The first is thinking that the parameter estimates for stock return models should be the same [5]. This means we treat the data as panel data. We treat the data in this way by using a dual indexing system. The first index is by the ticker name. Then followed by the time stamp. We can think of the data like this as a book where each page is dedicated to a ticker and its lines are differing time stamps. The advantage of treating the data like this, is that it allows us to take advantage of the greater amount of data points that can be fed into each model. This boosts the ability to generalize the model, as well as prevent over-fitting. The second group is by treating each stock with its own model, assuming that the parameter estimates for each stock should be different. Here the advantage is a boost in accuracy but this can be at the expense of over-fitting the in sample data.

Within the two groups we test data on quarterly and daily time-steps. Both predicting returns on the next time step. Due to the limits of the twitter API,

we used daily data to test the predictive power of the twitter sentiment. As discussed above with twitters constraint in mind, we chose many of the ratios to vary on daily data with the constraint on twitter data having We trained the daily models on the first 200 days of the past trading year, leaving the remaining for test data. For quarterly data, we test the predictive power of the 10-Q and 10-K sentiment. All of the models predict the returns of the stock in the next time step.

4.1 Panel Models

For Panel OLS we are seeking to solve for the parameters in the equation

$$y_{i,t} = \gamma_t + \beta x_{i,t}$$

Where i is the specific stock, t is the time step, β is the coefficients on the predictors, and γ is the time effect [6]. Looking at the results in *Table 2* We see that we have the highest error rates compared against all of the models.

For Random Effects, we seek to solve for the parameters of the same equation as Panel OLS, here however γ is the random effect. This is a slight variation to the model but not significant. We see the reflection of this in *Table 2* with the lower error rates overall.

The next model we used was long short term memory recurrent neural networks on the data being treated as panel data. is when the we are able to leverage the amount of data from a mix of using it as panel and with daily values. This allows us to take advantage of a specific type of neural networks, recurrent neural networks (RNN) and long short term memory RNN (LSTM). Both of these require thousands of data points for training properly. We get roughly 25,000 data points when using the daily panel data, allowing us to satisfy these requirements. With the quarterly panel data we are at a few thousand which leads to some caution when using the model on that data.

RNN work similar to that of neural networks, however previous hidden layers are fed back through the model allowing for a series effect [7]. Here we use this to take advantage of the time series nature of the data. The LSTM differs in that the data being fed back through the model starts to "be forgotten" in the sense that the weight of the data fed back from a longer time stamp has diminishing weight [7,8]. This allows time patterns at the beginning to not have influence over the entire time. Looking at the jump in the MSE in *table 2* for the quarterly data, we can see that the LSTM model suffered from over-fitting. We see that there was also a significant jump for the daily data.

Overall, the amount these models are off on average is more than enough to be off by the amount of gains that can be seen in the time period of a given stock.

4.2 Single Stock Models

For single stock evaluation we used two different models for quarterly and daily data, lasso and ridge regressions. We trained each stock using 5-fold cross

Model	Train MSE	Train MAE	Test MSE	Test MAE
Panel OLS (Quarterly)	10.3319%	37.1126%	14.7832%	38.6547 %
Random Effects (Quarterly)	10.3264%	38.3112%	12.6339%	45.6706 %
Panel OLS (Daily)	12.5971%	51.2615%	13.2428%	50.159 %
Random Effects (Daily)	11.0139%	53.5172%	11.2428%	51.0068 %
LSTM (Quarterly)	7.0131%	31.2615%	16.1632%	33.4121 %
LSTM (Daily)	6.7701%	33.653%	9.5157%	29.0264 %

Table 2: Panel Data Results

Model	λ	Train RMSE	Train MAE	Test RMSE	Test MAE
Lasso (Quarterly)	0.001	2.5302%	12.2534 %	8.5134%	13.0701%
Ridge (Quarterly)	0.1	2.5158%	12.1748%	14.6315%	13.3469%
Lasso (Daily)	0.001	8.7687%	12.3741%	5.2822%	6.3568 %
Ridge (Daily)	0.0001	8.5731%	12.489%	9.5157 %	7.9512%

Table 3: Single Stock (APPL Shown)

validation on the training data to choose an optimal tuning parameter. We chose optimal based on the lowest MSE on the left out fold. Once the optimal was found, we then fit the entirety of the train data on the model and predicted the returns on the test data.

Shown in *Table 3* we have the values when these models were run on AAPL. We chose to show AAPL as it performed consistently well compared to the other stocks on all the models tested.

The lasso regression on quarterly set all parameters to 0 except the return on cap, PE ratio, and the sentiment score. The model performed fairly well within the trained data, but it severely over-fit the training data. Even with the events of this year, especially with a company like Apple, did not play a significant role in their stock returns.

The ridge regression provided to be even more accurate than the lasso in regards to MSE and MAE, however it performed even worse than the lasso on the test data, providing more evidence of over-fitting.

The lasso regression on daily set all parameters to 0 except the PE ratio, inventory return ratio, the interest coverage ratio and the sentiment score. This makes sense as these were data that changed daily.

The ridge regression was interesting as all of the parameters were set to very small numbers excluding RD expenditure to market cap which has a coefficient of -1.8521. However it performed worse than lasso on the daily data across all of the measures.

The performance of AAPL on these models is not similar to most of the other stocks we selected. We believe that the biggest reason in this is AAPL fairly consistent returns over time.

5 Results and Discussion

Once the data was collected and processed through the machine learning models, an attempt was made at building a portfolio that would in theory create a risk-free arbitrage using the expected returns of the stocks. This is based on the idea that if two portfolios have the same beta but different expected returns, one can be sold and the other can be bought with the excess expected return being the risk-free arbitrage. In choosing the components of the portfolios, five stocks of varying expected returns and betas from various sectors were selected. This was done to minimize correlations between individual stocks while still maintaining the ability to diversify against individualized company risk.

After attempting variations of this portfolio to achieve a beta equivalent to the beta of the S&P 500 (ETF ticker SPY) we could not create a portfolio with higher expected return without the risk of high covariance within our portfolio or too high individual risk. The task of trying to achieve risk-free profits in an incredibly complex and massive market proved too complicated.

6 Conclusion

While our results may not have had the exact results we theorized, this attempt in creating a risk-free arbitrage shows how difficult it is to actually implement such a strategy. While the old adage states that "no battle plan survives first contact with the enemy" we gained valuable insights into some of the pitfalls of quantitative trading. The sentiment data proved to be a limitation because only data from 2020 was available. In addition, the market collapse from the onset of shelter-in-place orders and incredible whip-saw recovery was not reflected in financial ratios quarter to quarter or in sentiment derived from twitter. Our models were trained from a period between 2010 and 2019 which was one of the best bull markets in history and without a comparable data set to train our model our predicted returns did not match what actually happened. This was the case across the quantitative finance field whereby models of various quantitative funds (with much more complex models and much better risk management models) also suffered disastrous results in 2020.

This also highlights a need for robust risk management models, that can help mitigate for scenarios that are statistically low, but still possible, such as the first half of 2020. Moving forward, we plan on improving our models at every step, getting better and cleaner data, making our models robust and creating a more diversified portfolio that will hopefully predict better results. We also would like to further continue building a portfolio based on our models with an now evidently needed, risk management component. This project is the first of many steps in fully understanding the potential of data science and machine learning in quantitative finance. While at this current iteration we have not found any significant results, we hope to build upon what we have in place to build something to the point where we can derive more insight into the ever changing market landscape.

References

[1] E. F. Fama and K. French. *Dividend yields and expected stock return. Journal of Financial Economics*, 22:3–25, 1988.

[2] S. P. Kothari and J. Schanken. *Book-to-market, dividend yield, and expected market returns: A time-series analysis. Journal of Financial Economics*, 44:169–205, 1997.

[3] J. Pontiff and L. D. Schall. *Book-to-market ratios as predictors of market returns. Journal of Financial Economics*, 49:141–160, 1998.

[4] T. Yuz. A sentiment analysis approach to predicting stock returns. `https://medium.com/@tomyuz/a-sentiment-analysis-approach-to-predicting-stock-returns-d5ca8b75a42`.

[5] T. Arkan. *The Importance of Financial Ratios in Predicting Stock Price Trends: A Case Study in Emerging Markets*. 2016.

[6] Kevin Sheppard. Models for panel data. `https://bashtage.github.io/linearmodels/doc/panel/models.html`.

[7] Mahendran Venkatachalam. Recurrent neural networks. `https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce`.

[8] Hua Shi. Covid-19 global data -time series panel data with lstm recurrent neural networks. `https://melaniesoek0120.medium.com/covid-19-global-data-time-series-prediction-with-lstm-recurrent-neural-networks-f7825c4a`

[1] [2] [3] [4] [5] [6] [7] [8]

Appendices

A Appendix 1

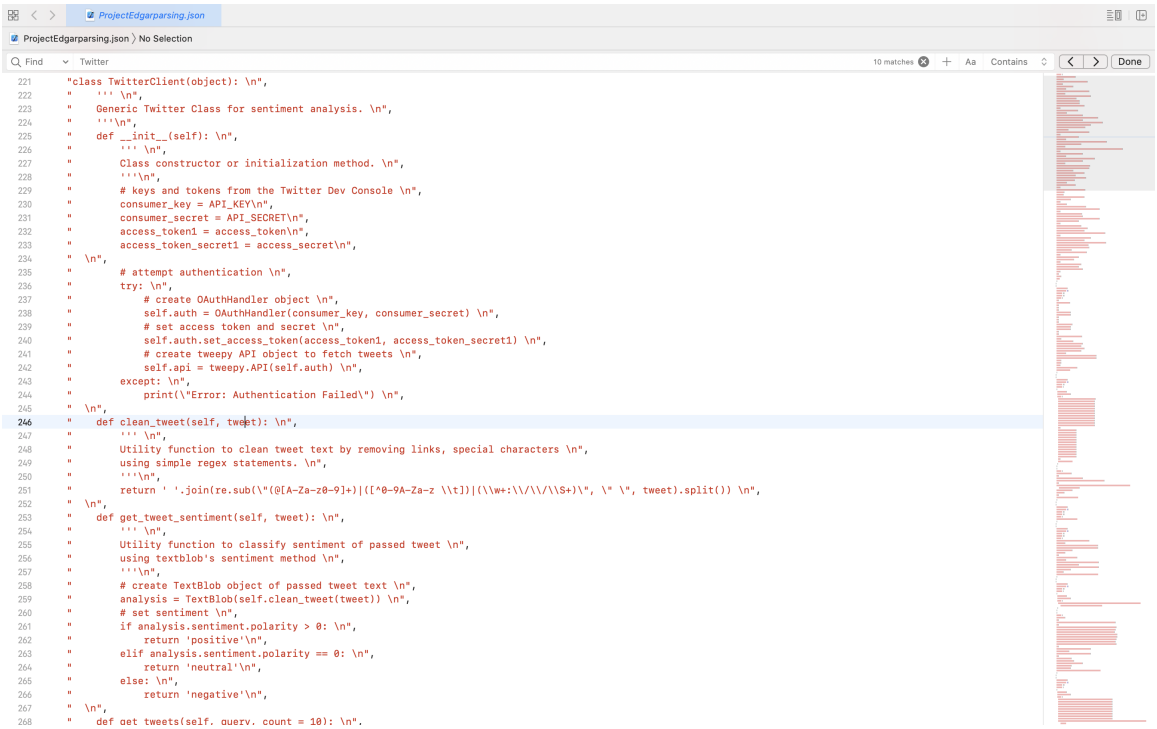


Figure 1: Portion of sentiment analysis code

B Appendix 2

```
@author: CAT
"""

import pandas as pd
import os
import xlwt
os.chdir("C:/Users/CAT/OneDrive/MSQF")
reader=pd.read_excel("team data version 1.0.xlsx")
header=reader.columns.tolist()

# for index,element in enumerate(header):
#     if element[:3]=='Unn':
#         header.pop(index)

for i in range(len(header)-1,-1,-1):
    if header[i][:3]=='Unn':
        del header[i]

reader1=reader.iloc[1:]

writer=xlwt.Workbook()
sheet=writer.add_sheet("main")

for i in range(2):#2 for testing purpose, actual is len(header)
    for j in range(len(reader1)):
        sheet.write(i*2852+j+1,0,header[i])
        for k in range(14):
            sheet.write(i*2852+j,k+1,reader1.iloc[j,k])
writer.save("pyteam.xls")
```

Figure 2: Data cleaning python script