Stock Price Prediction on the New York Stock Exhange
CA684 Machine Learning

Tom Skehin - 16211700
tom.skehin2@mail.dcu.ie

19th April 2017

1 Introduction

Purchasing shares is one the most popular and intuitive forms of long term investment in the stock market, with 3 and 6 billion shares traded daily on the New York Stock Exchange (NYSE) alone. The fundamental strategy is to buy shares in a company at a particular price with the hope of selling them at a premium in the future. The percentage difference between the buying and selling price represents the return on investment to the market participant.

The value of savings, investments and pensions of many people depend on the movement of these prices so it would therefore be very useful for fund managers to have a means of predicting which stocks will increase in price over time.

In this assignment I'm hoping to produce a model that can predict an increase in price based on fundamental data about the stock and the underlying company.

2 Problem Definition

Under the rules of listing on the NYSE, a company is required to report financial information to shareholders [2]. The most significant of these is the Form 10-K, an annual report detailing the company's performance over the previous fiscal year. Amongst other information this filing includes the company's audited financial statements presented according to a set of accounting standards known as the Generally Accepted Accounting Principles or GAAP [4]. These standards allow for comparisons between companies and the derivation of standard performance related metrics.

I was able to source a dataset of this information, along with historical stock prices on `www.kaggle.com` [5]. The prices were fetched from Yahoo Finance and the fundamentals from NASDAQ Financials. The subset of fields I was interested are listed in the Appendix.

I am hoping that these descriptive features and their values at the time of filling the Form 10-K will be predictive of the change the price of the company stock in a year from then.

3 Review of Existing Methods

My primary source for this analysis was a paper written by Nikola Milosevic from University of Manchester [1]. He created a binary classifier for the change in price of an equity over a one year time period and attempted to predict it using data obtained from Bloomberg. This data was a mixture of fundamental information about the company for example Revenue and Sales Growth, but also information about the stock, Dividend Yield and Change in Net Stock Price over a one month period.

Milosevic used this data to train several classifier models using the open source machine learning tool Weka.

4 Proposed Method

Similarly to Milosevic's approach, I want to create a target feature based on the movement in stock price over the one year period. If p_0 is defined to be the stock price at the time of the 10-K filing and p_1 is the price one year later, then the target feature y is defined as follows:

$$y = \begin{cases} 1, & \text{if } \dfrac{p_1}{p_0} - 1 \geq 0.10 \\ 0, & \text{otherwise} \end{cases}$$

This represents a binary classifier identifying stocks that have increased by 10% or more in the previous year.

I'm focusing on four machine learning classification algorithms, Logistic Regression, Support Vector Machines (SVM), Naive Bayes and K-Nearest Neighbour. Three of these algorithms were looked at in Milosevic's analysis, but I'm hoping to expand on these methods by introducing cashflow descriptive features not included in that feature set. Cashflow models are commonly used by financial analysts as a means of determining the price of a company's stock so I'm hoping the inclusion of Net Cash Flow, Net Cash Flow-Operating, Net Cash Flows-Financing and Net Cash Flows-Investing will have a positive influence on my models.

I've also derived some standard company performance related metrics and ratios which should allow for some comparison of larger asset companies with some that are not as big.

Some sample metrics are given below:

$$\text{Book Value} = \frac{\text{Total Assets} - \text{Intangible Assets}}{\text{Total Liabilities}}$$

$$\text{Equity Turnover} = \frac{\text{Total Revenue}}{\text{Total Equity}}$$

$$\text{P/E Ratio} = \frac{\text{Share Price}}{\text{Earnings Per Share}}$$

$$\text{P/CF Ratio} = \frac{\text{Share Price} * \text{Estimated Shares Outstanding}}{\text{Net Cashflow}}$$

As I demonstrate in experiments in the next section, several of my features are exponentially distributed, I tackle this by taking the natural logarithm of these and introducing them new features. I'll also investigate different scaling and normalisation methods for my features.

5 Experiments

5.1 All features, rescaled between 0 and 1.

In order to establish a baseline for comparison, I first rescale my features to the range to $(0, 1)$. I've split the data 90/10 into training and test sets and fit each of the models using the training set. The performance metrics for the test set are given below.

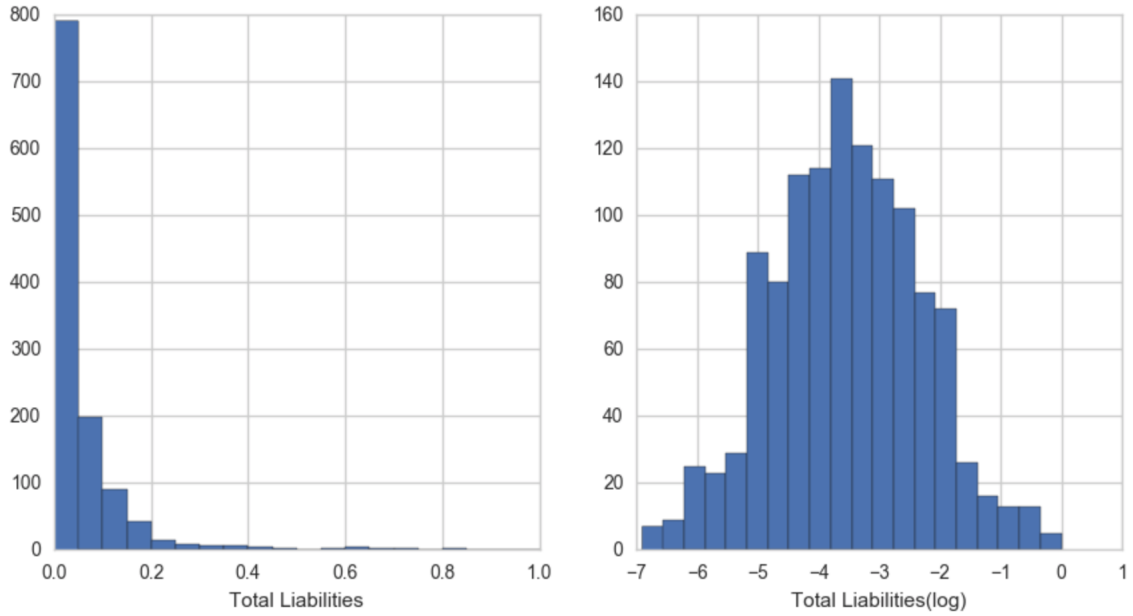
Algorithm	*Precision*	*Recall*	*F-Score*
Logistic Regression	0.52	0.55	0.50
SVM	0.47	0.55	0.43
Naive Bayes	0.55	0.57	0.50
K-Nearest Neighbour	0.47	0.48	0.48

5.2 Distributions of features

An analysis of the distributions revealed that a lot of the features appeared to be exponentially distributed. In an attempt to normalise the distribution I applied the following transformation to my matrix of features X:

$$\hat{X} = \log_e(X + 0.001)$$

The constant, added to X before taking the logarithm, is a smoothing factor to avoid undefined values where $x_{ij} = 0 \in X$. The histograms below show the distribution of my Total Liabilities feature before and after the transformation is applied.



The features in the matrix $\hat{X}$ were rescaled to the range $(0, 1)$ and split into training and test sets as before. The performance metrics for $\hat{X}$ are below, which show an improvement in F-Score from X across all models with the except of Naive Bayes.

Algorithm	*Precision*	*Recall*	*F-Score*
Logistic Regression	0.56	0.57	0.56
SVM	0.56	0.58	0.54
Naive Bayes	0.52	0.56	0.47
K-Nearest Neighbour	0.54	0.55	0.54

5.3 Choosing the 'most normal' features

I wanted to try and improve on the previous experiment where I applied the transformation to all features without bias. Instead, I wanted to apply the transformation where the resulting distribution was more normal than the original. Normality was tested using the normaltest() method in scipy.stats. The distribution with the higher p-score resulting from this test was deemed to be more normal of the two. This transformation, $\widetilde{X}$ was train/test split and the results are below:

This seemed to have a marginal impact on the models but if I can assume normality then I can use different scaling methods in the next experiment.

Algorithm	*Precision*	*Recall*	*F-Score*
Logistic Regression	0.55	0.56	0.55
SVM	0.53	0.55	0.51
Naive Bayes	0.56	0.58	0.52
K-Nearest Neighbour	0.52	0.52	0.52

5.4 Normalising features

I chose to use the RobustScaler class from the sklearn.preprocessing. This scales the data according to the interquartile range i.e. the range between the 1st and 3rd quartiles and as such is a lot less susceptible to outliers in the data than general standardisation. When $\widetilde{X}$ was scaled to this range and fitted, I got the following metrics for my models: Here SVM

Algorithm	*Precision*	*Recall*	*F-Score*
Logistic Regression	0.52	0.52	0.52
SVM	0.64	0.64	0.64
Naive Bayes	0.56	0.58	0.52
K-Nearest Neighbour	0.60	0.58	0.58

and K-Nearest Neighbour look to be the strongest performing of the models.

5.5 Recursive Feature Elimination on Linear Models

To evaluate the influence of my features on my SVM and Logistic Regression Models, I decided to use Recursive Feature Elimination with Cross Validation. It works by training a classifier on the initial set and assigning weights to each of the features. The lowest ranking features are then removed iteratively until the optimal the set remain. This automatic tuning of the number of features selected with cross-validation [3]. The results of the RFE are below

Algorithm	*No. Features*	*Precision*	*Recall*	*F-Score*
Logistic Regression	5	0.52	0.52	0.52
SVM	2	0.64	0.64	0.62

5.6 Optimal K for K-Nearest Neighbour

Using cross validation I was able to iterate through a reasonable number of K and find the optimal K-Nearest Neighbour model.

Algorithm	*K*	*Precision*	*Recall*	*F-Score*
K-Nearest Neighbour	57	0.57	0.58	0.58

6 Analysis

Judging by the F_1 measures, SVM seemed to provide the best performance across each of the classifier models. After applying the log transform and normalising it produced the best score of out of all of the experiments I conducted. This is broadly in line with the scores achieved by Milosevic for his SVM model.

I analysed the coefficients of the features in this model and tried to pick factors with large weights that I intuitively felt were fundamental to stock price evaluation, however, when retraining the model on this reduced set I was unable to produce model with as high an F_1 score.

7 Conclusion

Given more time I would like to investigate the relationships between the features themselves. A lot of these company fundamental values vary inline with each other, so it is quite possible that I am duplicating the same information across features.

In the future I would like to revisit this analysis and derive some more advanced company performance metrics to include in my feature set.

Finally, while an F_1 of 62% for my best model may be not be overly predictive in its own right, it should hopefully be useful to market participants in order to identify shares that would warrent further analysis.

8 Appendix

Table 1 – List of Features extracted from fundamental.csv on `kaggle.com`

Cash Ratio	Cash and Cash Equivalents	Current Ratio
Fixed Assets	Gross Profit	Intangible Assets
Long-Term Debt	Long-Term Investments	Net Cash Flow
Net Cash Flow-Operating	Net Cash Flows-Financing	Net Cash Flows-Investing
Net Income	Profit Margin	Quick Ratio
Total Assets	Total Equity	Total Liabilities
Total Revenue	Earnings Per Share	Depreciation
Inventory	Investments	Operating Margin
Research and Development	Retained Earnings	

References

[1] N. Milosevic. Equity forecast: Predicting long term sotck price movement using machine learning. https://arxiv.org/pdf/1603.00751.pdf, 2016. [Online; accessed 22-Apr-2017].

[2] New York Stock Exchange. NYSE Manual 203.00. http://nysemanual.nyse.com/, 2017. [Online; accessed 22-Apr-2017].

[3] scikit learn. 1.13. Feature selection. http://scikit-learn.org/stable/modules/feature_selection.html#rfe, 2017. [Online; accessed 22-Apr-2017].

[4] U.S Securities and Exchange Commission. How to Read a 10-K. https://www.sec.gov/fast-answers/answersreada10khtm.html, 2017. [Online; accessed 22-Apr-2017].

[5] www.kaggle.com. New York Stock Exhange. https://www.kaggle.com/dgawlik/nyse, 2017. [Online; accessed 22-Apr-2017].