# Using Sentiment Analysis and a Hybrid SVM-HMM Model for Stock Price Prediction

**Bolanle Onifade**

**University of Nottingham**

# Abstract

Over the past few decades, many theories that describe the behaviour of the stock market have been proposed. These theories put forward ideas ranging from those claiming that the stock market cannot be predicted and to those claiming those that with the right assumptions, the stock market can be predicted. Regardless of this, many stock market prediction models (based on technical indicators or non-technical indicators) have been put forward. In this dissertation we aim to consider those non-technical factors that affect stock prices. These non-technical factors come in form of news articles. In the digital age of rapid response to ongoing situations, news articles about events tend to be released as soon as they occur and the assumption is that if we can track news sources, monitor them for the release of relevant news articles, we can use the sentiment orientation of the article (whether positive, negative or neutral) to predict the price of the stock market before the market has a chance to react to the article. This of course poses an important question on how the sentiment orientation of articles. There are several approaches that can be taken towards determining the class of articles but for the purposes of this dissertation, we will focus on the use of Support Vector Machines to perform classification of news articles. The output of classification will then be fed into a hybrid Support Vector Machines – Hidden Markov Model classifier to perform the price prediction.

# Introduction

A lot of past work has been done using technical and fundamental indicators such as the current investment, general economy, recessional periods, currency and industry – this is known as fundamental analysis. However, very few articles have attempted to go beyond that and while some of these methods perform reasonably well, very few articles in the literature have attempted to go beyond historical data. Even still, within the set of articles that attempt to use recent (external) information such as news articles, very few articles attempt to incorporate historical technical data. With this work, we aim to further fill in this gap by utilising both historical data (section 3.6.)that provide us with an indication of how well the stock price has done in the past as well as current news that provide us with a sense of what the general sentiment regarding a certain stock is. This is primarily because it stands to reason that we will make much higher profits with finding the middle ground between the two extremes. A few related works have attempted to use both methods to predict the stock market (Deng, Mitsubuchi, Shioda, Shimada, & Sakurai, 2011); however, the backbone of the current proposed model uses a Support Vector Machine and Hidden Markov Model (SVM-HMM) hybrid model for the predictions – which to our knowledge hasn’t been used previously.

# Literature Review

Often, it is the case that a useable corpus of financial news isn’t available hence authors are forced to generate their own data (Zhan, Cohen, & Atreya) either by manually classifying or automatically generating data (Fung, Yu, & Lu, 2005; Zhan, Cohen, & Atreya). Zhan et al. in their comparison of manual and automatically generated data highlight that the key to classifying news articles manually is the general information that is being conveyed by the articles. Mergers, lower interest rates are general considered *good news* while corruption, lawsuits, wars are considered *bad news.* Working with automatically generated data based on simply the stock price movement (if the stock price goes up after the article is released then the article is good otherwise, it’s bad). The classifier developed has F1-scores of 0.26, 0.38 and 0.36 for positive, neutral and negative articles respectively. They attributed the poor performance to the lack of data and poor article selection whilst also suggesting that news articles are better suited to long-term prediction as opposed to day-to-day prediction. It’s quite clear that there are while there are issues with automatic generation of data (section 5.3.2.), the method employed by Zhan et al. can be said to be too naïve.

Regardless of the criticisms of automatically labelling news article based on the direction of movement of the next day’s price movement, Kaya et al had a 60% accuracy using the method as well as -based feature selection. Fung et al. (2005) utilise piecewise linear segmentation to automatically classify news articles – a method that has been employed in the current work and will be explained in the next chapter.

Gidofalvi(2001), derive a unique method of automatically assigning labels to news articles by aligning news articles to the intraday stock data and although it doesn’t perform very well, we spend some time on it due to its contribution to the literature. A window of influence is defined which is used to evaluate the possible effect of a news article. The author defines the window of influence of an article with the timestamp as the lower boundary offset and the upper boundary offset from t. An offset is negative is is prior to. In addition, news articles that aren’t published within the opening and closing market times are filtered out as these are said to be ambiguous. Using set thresholds, news articles therefore are classified.

Perhaps, more unconventionally, is the use of tweets from twitter to predict the stock market (Bollen, Mao, & Zeng, 2011). They determine the correlation between the mood of the twitter feeds and the Dow Jones Industrial Average. The moods are determined using OpinionFinder (classifies into positive and negative) and Google-Profile of Mood States (GPOMS). A Granger Causality Analysis and Self-organising Fuzzy Neural Network are then used to determine the validity of the hypotheses that moods predict the stock market. The orientation provided by OpinionFinder is discovered to be less predictive than the GPOMS dimension *Calm*. They also claim that there exists a (3-4 days) time lag between the mood expressed on twitter and the changes in the DJIA values – hence stock price movements can be known well in advance.

Bar-Haim et al. (2011) propose a method of identifying expert investors from twitter feeds, which can then act as a basis for predicting the increase in stock prices. They compare two extreme methods: focusing on tweets that explicitly state transaction details as well as learning the correlation between the stock price and the tweet’s contents. The second approach removes restrictions on the applicable tweets but with the caveat that a lot of noise is likely to be introduced into the training process. They show that making the process user-sensitive improves the prediction accuracy. The algorithm involves a classifier which classifies a time-annotated set of tweets by each user and classifies each as bullish, bearish or neutral. Each tweet can then be evaluated for correctness by determining that the stock market behaves in accordance with the classification of the tweet. Tweets finally are ranked according to correctness. Another method utilised is unsupervised learning based on the timestamp of the tweet. Using several methods: joint-all model (a single SVM model trained on all tweets), transaction model (finds expert users in based on the correlation of their tweets to the movement of the stock price), per-user model (removes noise by unsupervised learning for each potential expert), joint-experts model (using the per-user model, train a single SVM model). They conclude that the most accurate models are the per-user and the joint-experts perform the best.

Zhang and Skiena (2010) compare blogs and news as basis for prediction, perform large-scale analysis of the stock market and propose a trading strategy based on sentiment data. Data from Dailies (an aggregator of news), twitter, Spinn3r RSS feeds and LiveJournal was processed by Lydia (a text processing system), resulting in time series consisting of a time series of words and their orientation. They discovered that the media exposure correlates more to the stock market of certain industries (Aerospace and Defence) and less so for others (Software and Computer Services).

AZFinText (Schumaker & Chen, 2005) works with the assumption that 20 minutes after a news article is released, the stock price reflects the effect of the news. As opposed to labelling with a polarity (up or down), it labels using the stock price 20 minutes after the news article is published. The system uses proper nouns as the features and filters the selected features by only further selecting proper nouns that occur three or more times. It uses support vector regression to try to predict the price of the stock in 20 minutes. For a period of 23 trading day, using 2809 articles and predicting for the S&P 500 Index, they achieved a 8.50% return on investment while the S&P 500 Index achieved only a 5.62%.

NewsCATS (Mittermayer & Knolmayer, 2006) categorises news articles into “good”, “bad” and “no movers”. Using a thesaurus as a curator, they classify press releases as good if the maximum gain in 15 minutes after its release is large () and the maximum loss is (). The inverse is the case for “bad” news. Neutral news are classified as those whose gain and loss do not exceed 3%. NewsCATS achieved an average of 0.29% over 2602 trades while a simulated random trader achieved a profit of 0.07% over 2599 trades. Mittermayer and Knolmayer conclude by cautioning that the results do not factor transaction cost and hence do not accurately reflect the potential profit.

# Background Knowledge

In order to make the documents useful to numeric classifiers such as the neural network or support vector machine, the terms need to be transformed into their weighted forms. The weighting of the terms is shown to be at least as important as their selection (Strzalkowski, 1994).

The vector space model introduced by Salton et al. (1975) is the preferred algebraic method of representing textual documents. The document is represented as a single vector where each dimension in the vector is a single feature. Features that do not exist in a particular document are simply given a value of 0. The weights are usually calculated using simply their presence (binary), counts (integer), frequency(float) or their term frequency inverse document frequency (TF-IDF) score (float). Hence we can say that in a corpus of documents and features, a document is represented as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.1) |

where . We described an improvement on TF-IDF in chapter 2 and now, we provide the mathematics behind the traditional TF-IDF (Salton, Wong, & Yang, 1975). Given, the document, the weight is calculated as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.2) |

where is the number focuments which contain the term . 1 is added to prevent divide-by-zero errors. is the term frequency and is given by the formula:

|  |  |  |
| --- | --- | --- |
|  |  | (3.3) |

where is the number of times feature occurs in document and is the total number of terms in the document . The higher the value assigned to a feature, the importance, it is given.

## T-Test Based Split-and-Merge Piecewise Linear Approximation

In their paper, The Predicting Power of Textual Information on Financial Markets, Fung et al, (2005) present a means of generating labelled data by detecting trends in a time series and aligning news with the trends; we present their algorithm here.

### The Splitting Phase

The split-phase of the algorithm handles discovering the trends in a time series while the merge phase helps avoid over-segmentation. Each time series can be represented as a list of tuples – each tuple containing the price and time:

|  |  |  |
| --- | --- | --- |
|  |  | (3.5) |

where is the price at for and the time series has a length .

The process starts by representing the trend with a line defined by the first and last points (a segment). In order to decide if the line is enough to represent , a one-tail t-test is defined:

|  |  |  |
| --- | --- | --- |
|  |  | (3.6) |

where is defined as the expected mean square error:

|  |  |  |
| --- | --- | --- |
|  |  | (3.7) |

where is the number of points in the segment. is the projected price (derived from the line ) at time. The t-statistic is defined by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.8) |

Where is the standard deviation. The t-statistic is compared with the t-distribution using a degree of freedom and an . Hence, there is a 0.05 probability of the null hypothesis being accepted given that is incorrect. If is accepted, then the mean-squared error is low and the projected trend line is very similar to the actual time series . If is accepted then, the line is not enough to represent . If is accepted, then the line is split where the error norm is maximum – – resulting in two segments. The process iterated by repeating the process on the two segments. The algorithm is represented in algorithm 3.1, as produced by Fung et al.

### The Merging Phase

Over-segmentation is the existence of adjacent two segments whose slopes bear enough similarity to not warrant two separate segments. To fix this issue, the merging phase merges these segments into one large one. The merging phase aims at combining adjacent segment whose merging would not result in being rejected.

Consider a time series (where ) , the result of splitting time series . is a segment in . Should potential merge of the segments and not result in being rejected then they are regarded as candidates for merging. Let the list contain all such pairs of candidates. The selected candidates are the pair that would result in the lest increase in . This process is repeated until the -test is rejected. Hence . The merge algorithm is provided in figure 3.2.

|  |
| --- |
| split(]) – split a time series of of length  from time to time where |
| 1:  2:  3:  4: **for**  to **do**  5:  6: **if**  **then**  7:  8:  9: **end if**  10:  11: **end for**  12:  13: **if** -test.reject **then**  14:  15:  16: **end if**  17: **return** |

Algorithm 3.1 – The Split Algorithm

|  |
| --- |
| merge() – merge two adjacent segments on time series T |
| 1: **while** *true* **do**  2:  3: **repeat**  4:  5:  6: **if**  **then**  7:  8:  9: **end if**  10: **until** end of time series  11:  **if -**test.accept() **then**  12: drop  13: **else**  14: break  15: **end if**  16: **end while**  17: **return** |

Algorithm 3.2 – The Merge Algorithm

## Support Vector Machines

Techniques in machine learning generally are classified into two main divisions: supervised and non-supervised learning. Support Vector Machines (SVMs) are classified under the former. This means that the process by which SVMs solve tasks is by first undergoing a training phase in which input data (referred to as a set of examples or instances) are used to learn a model that can then be used to classify new instances into a category of a set of categories. SVMs typically solve binary classification problems.

SVMs work by projecting and mapping (by using a kernel function) training data instances into a high-dimensional feature space, in order that the maximum gap between the two categories is created. Hence, new instances can be classified by side of the gap the point falls on. *Gap* is a vague term and is difficult to define. Instead, SVMs work by constructing a hyperplane that separates the two categories – this is referred to as the maximum margin hyperplane. The maximum margin hyperplane is sought as theoretically, such an hyperplane should lead to the least generalisation error.

A hyperplane which defines the decision boundary of the classes is in turn defined in terms of a set of points whose set of points satisfy the equation.

where is the weight vector of the hyperplane, is the kernel method that maps the data points to feature space and is the bias.

For linearly separable data, two hyperplanes can be defined which completely separate the data such that there are no points between them. The area in space that is bordered by the two hyperplanes is called a margin. The two hyperplanes are defined by the following equations:

In order to define and , vectors must be selected which just touch the margins – these vectors are referred to as support vectors as these are the only data instances required to represent the model. The margin can also be defined more formally in terms of and which are the shortest distance between the maximum margin hyperplane and the closest positive support vector and the shortest distance between the and the closest negative support vector, respectively.

## Hidden Markov Models(HMM)

Hidden Markov Models (written as )are simply a set of parameters which explain a pattern for a known class or category. HMMs can be used to classify a test-pattern for which it has the highest posterior probability. Sequences can be considered as a series of states , written as . There are no restrictions on the number of states to be visited or the number of times a state can be visited. In order to define any sequence, transition probabilities –the probability of at ­ given at - must be evaluated. This is formulated mathematically as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.12) |

At each step in the sequence, the state emits a symbol . Symbols are otherwise referred to as observations as they are visible. Hence, we might have a sequence of observations **.** Given the states and observation, we can then make another deduction: the probability of observing a certain symbol given the state at time :

|  |  |  |
| --- | --- | --- |
|  |  | (3.13) |

is thus referred to as the emission probabilities. At each time step, a transition must occur and a symbol must be emitted, resulting the redundant formulations:

|  |  |  |
| --- | --- | --- |
|  | for all  for all | (3.14) |

There are three main problems addressed with the Hidden Markov Model: The Evaluation problem, the decoding problem and the learning problem.

The evaluation problem is described as: Given an HMM, the transition and emission probabilities, evaluate the probability of a sequence being generated by the model:

|  |  |  |
| --- | --- | --- |
|  |  | (3.15) |

The decoding problem: Suppose we have a observations sequence and a model **,** determine the most likely sequence of hidden states that generated the observations. The Viterbi algorithm is used to solve this problem (Algorithm 3.5.)

The learning problem: Given the number of states and the number of visible states and a set of training observations, determine the emission and transition probabilities.

The problems can be solved by employing the Forward algorithm, the Viterbi algorithm and the Baum-welch algorithm.

## Hybrid HMM-SVM Model

The variation in hidden markov models comes in the form of which the emission probabilities are calculated; hence, HMMs can be based on the Poisson distributions, Gaussian distributions or Gaussian mixture models and in this case SVMs. For continuous data, Gaussian mixture models are typically the preferred method for deriving the emission probabilities; however, they are often criticised for their poor discriminatory capabilities. Conversely, SVMs are popular for their great discriminatory capabilities.

SVMs do not directly output probabilities, instead they output the measured distance between the example data and the hyperplane: . In order to create a link between the posterior probability of a instance having a class : and the result of the SVM, we need to utilise Platt’s (1999) proposal for generating pseudo-probabilities by the use of a sigmoid function:

|  |  |  |
| --- | --- | --- |
|  |  | (3.21) |

where and are derived by using maximum likelihood estimation fron a training set. And is the unthresholded output of the SVM defined by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.22) |

In the binary case, we can thus compute the posterior conditional probabilities and of classes and given the symbol . Finally using Bayes’ rule, we can compute the emission probabilities from the outputs of the SVM using the prior probability calculated from the frequency of the class in the training data:

|  |  |  |
| --- | --- | --- |
|  |  | (3.23) |

# Scientific Method

The proposed system is comprised of two main parts: sentiment-based news classification and price prediction. Any developed system cannot completely rely on news as news is sporadic and there’s no guarantee that news regarding a certain entity will be released for every single trading day and thus, there needs to be a means by which prediction can still take place. This comes in the form of technical data. Hence, there are two phases of classification – the classification of pure news articles and the classification of technical data. Figure 4.1 shows the system overview, ignoring news labelling.

Unlike sentiment classification for domains such as films, cars or music, happiness, sadness and ambivalent news articles may not bear much information about the progress of the entity. While the sentiment of the article is what we aim to extract, a cursory look at any news article that bears financial information will show that news articles aren’t very sentimental. This indicates that classification based simply on human sentiment while might be accurate might not be as successful given that we aim to predict the stock market. Therefore, to supplement classification based on sentiment, we also classify based on the progression of the company. This means that the articles get classified into an additional set of categories (positive, negative and neutral).The aim with the progress classification is that articles get classified based on what the news article’s evaluator expects as regards to whether the entity’s stock price will go up, down or simply stay the same as a result of the article.

## Data Acquisition

### News Articles Acquisition

It’s clear that the very first task to be performed is the acquisition of news articles whether labelled or non-labelled. Although a fair bit of work have been done using this particular approach to stock price prediction, we were unable to find any publicly available datasets that fell in line with the purposes of this dissertation. Hence, a dataset was generated from online news sources.

Although selecting news sources seems like a trivial task, it requires careful consideration as the news sources has to be able to satisfy the following requirements:

1. Has to be popularly read, especially by traders. This is particularly important because a high level of trust needs to be placed in the news source, enough to determine that significant changes in stock price trend will be reflected in the news articles.
2. The news sources should have decent coverage of news– to ensure that we gather as much data as possible.

Given these requirements, investors on online forums (as well as individuals with knowledge of finance) were asked which news sources were read and the following sources were given: Reuters, Bloomberg, Financial Times, Market Watch, Yahoo Finance.

The table below shows the number of articles that were gathered for each company.

|  |  |  |
| --- | --- | --- |
| Company Name | Number of Articles | |
| Chevron | | 88 |
| Cocacola | | 52 |
| Disney | | 108 |
| Exxon | | 120 |
| Goldman | | 731 |
| IBM | | 118 |
| JP Morgan | | 613 |
| Microsoft | | 259 |
| Pfizer | | 115 |
| Visa | | 48 |

Figure 5.1 – Number of articles collected for each company

### Stock Data Acquisition

The price daily values were collected for the period from the 1st of January, 2013 to the 30th of September, 2014. Of course, the stock price data is only released for working days so this accounts for only 440 working days.

### Data Labelling

At the start of the project, the intention was to crowd source the labelling of the articles. This would be done by asking individuals with knowledge of economics and finance to evaluate the news articles. This involved uploading the corpus to a website for easier classification. (sentimentanalysis.bolanleonifade.me). However, the rate at which the articles were getting classified was very slow so the alternative approach taken was to use prior knowledge (of the current author) to label the article. This of course meant that experiments might suffer due to lack of enough financial knowledge. Therefore, in order to provide a baseline or at the very least, a means of evaluating the manually labelled data, a set of automatically labelled data was created as well.

#### Manual Labelling

Manual labelling of data is simply reading each news article and labelling them by hand. The evaluators are asked to estimate the company’s progression based on the news article. Their estimates can fall into the following categories: (up, down, neutral). The evaluators are also asked to provide the sentiment of the article (happy, sad, neutral). From henceforth, for clarity purposes, we shall refer to the former as “progress sentiment” and the latter as “feeling sentiment”

One might think that there is a perfect correlation between the two sets of categories. However, there can be differences between the two. For illustrative purposes, we will examine a couple of cases in which there are differences:

**Data Acquisition**

**Price Prediction**

News Articles Storage

Stock Price Data

External News Sources

External Stock Price Data

**Feature Generation and Selection**

Chi-square feature selection

Generate vector space model with TF-IDF as weights

SVM-based classifier

**Pre-processing**

Normalisation

Feature generation from stock price data and svm-classification of news

HMM-SVM-based classifier

Down Price Direction

Up Price Direction

News Classification into positive, Neutral or Negative

Figure 4.1 - Process Overview

**Price Prediction**

**Pre-processing**

Stemming

Tokenisation (into unigrams, bigrams)

Stop-word removal

The headline “Exxon Mobil reports Fire, oil spill at Nigerian terminal”, evokes a feeling of sadness but due to the established nature of Exxon, it’s unlikely that this event is going to lead to a massive dent in the stock price, we give the article a progress sentiment of neutral (because the article doesn’t go on to indicate that Exxon will suffer from this incident). Another article that wouldn’t be expected to change the stock price much is “JP Morgan employee falls to death from building roof in Hong Kong”. It’s however clear that the article is “sad”, but the effect on JP Morgan’s progress would virtually be nil.

If the previous two examples give the impression that from headlines, we can always tell the feeling sentiment of an article, it would be wrong. In fact, a seemingly neutral headline such as ”Coca-cola names Walter Finance Chief as Fayard Retires” goes on to discuss the recent struggles of coca-cola, therefore giving it a feeling sentiment of sad and a progress sentiment of neutral. In the same strain, we discovered articles can both be up for progress sentiment and feeling sentiment; this would be the case for articles that discuss an entity’s growing business.

#### Automatic Labelling

In order to perform automatic labelling of news articles, we need to generate projected trends, this gives us an idea of the overall outlook of the stock price – that is, for example, we can safely say that the overall projected trend of the stock price is an upwards movement if the price over a period of time has changed positively, ignoring every minor dip in the price trend. We can perform automatic labelling by using piecewise linear approximation. This allows us to align news articles with the projected stock price and simply labelling the articles based on the projected stock price.

There are obvious inaccuracies that can occur from the use of such a method – in fact, as shown in the literature review, classification based purely on price differences, tends not to be very accurate – this is because one cannot say for sure that all articles released during periods of overall upwards positive price movement are positive and vice versa. However, we are operating under the assumption that news articles strongly reflect the direction of movement of the stock market. We expect the price to move up when news articles discuss increases in sales, innovation, positive restructuring and we expect the price to move downwards when news articles discuss fines, bankruptcy, litigation, sanctions etc.

Automatic labelling, therefore provides us with a baseline. The higher the similarity between the results of automatic labelling and that of manual labelling, the more *trust*, we can place in the results of manual labelling.

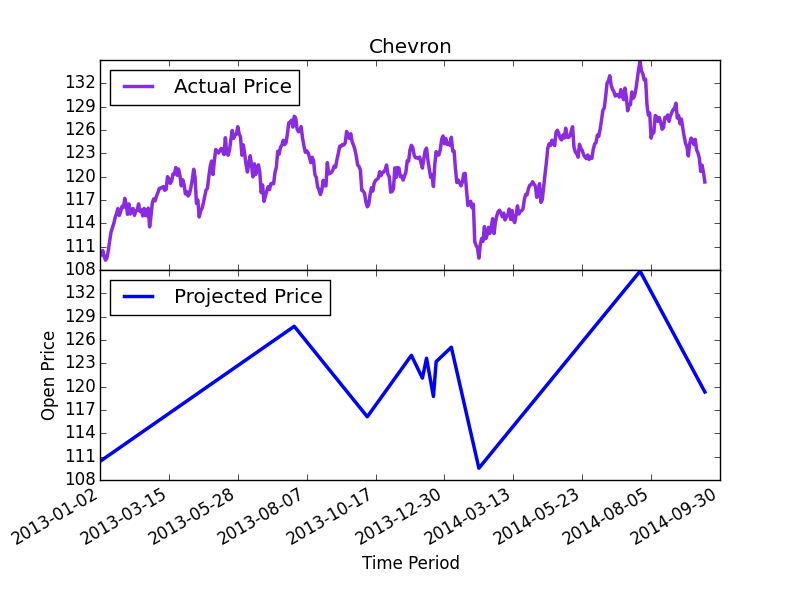


Figure 5.3 - Actual Stock Price and Projected Price of Chevron

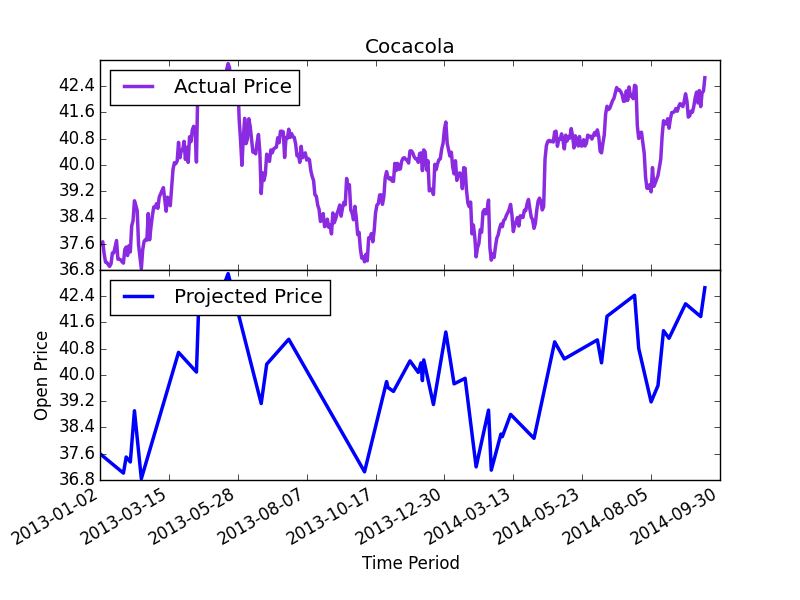


Figure 5.4 - Actual Stock Price and Projected Price of Cocacola

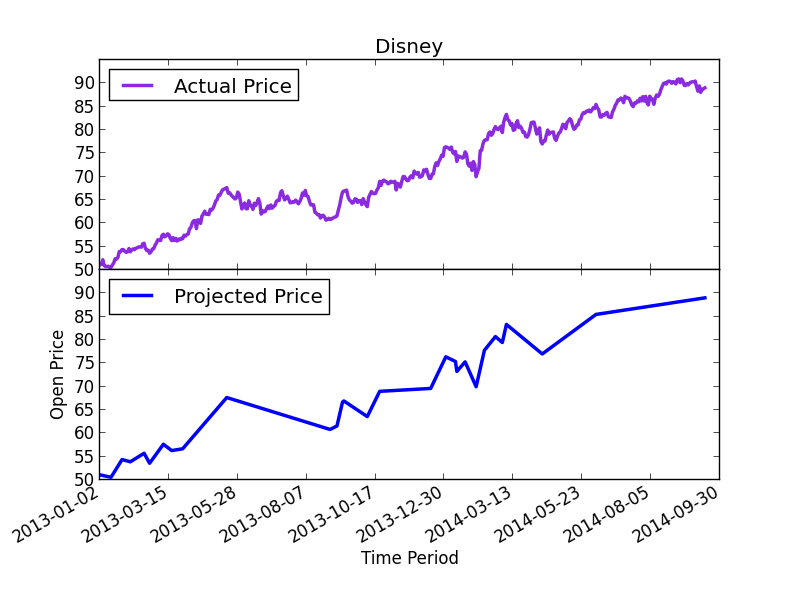


Figure 5.5 – Actual Stock Price and Projected Price of Disney

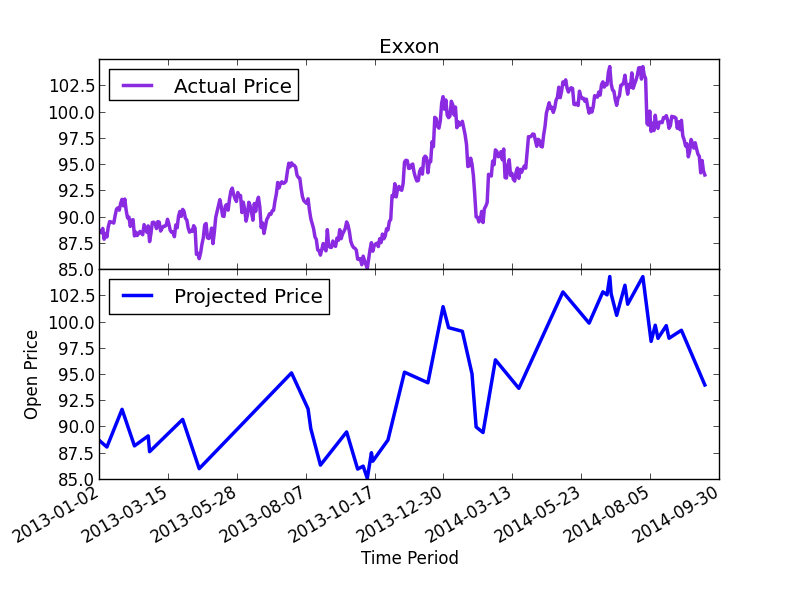


Figure 5.6 – Actual Stock Price and Projected Stock Price of Exxon

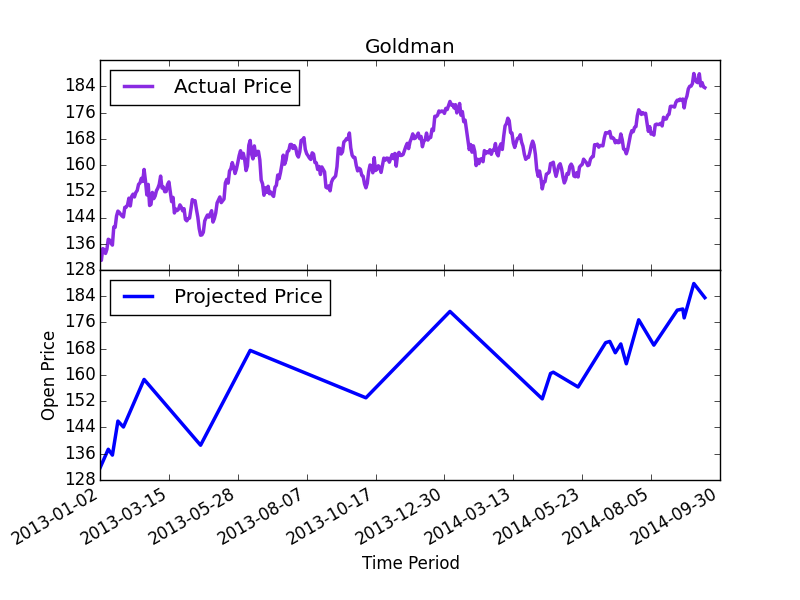


Figure 5.7 - Actual Stock Price and Projected Price of Goldman

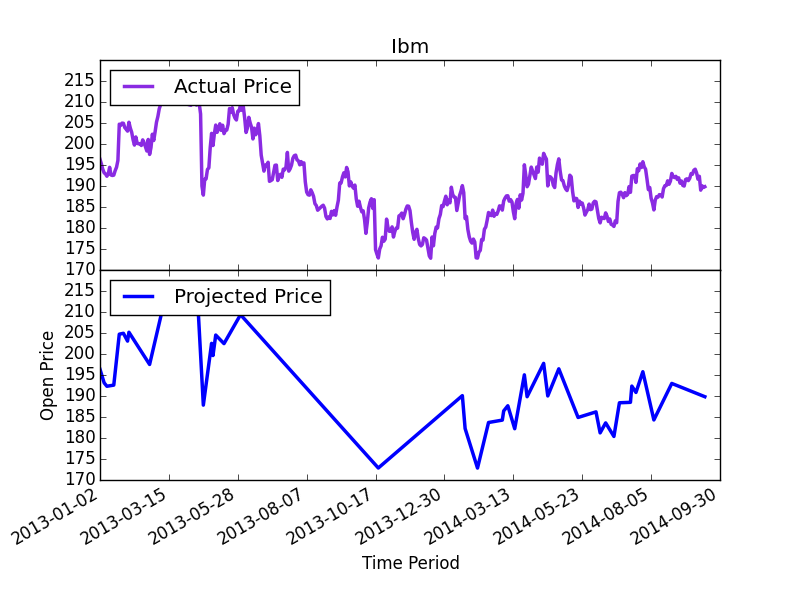


Figure 5.8 - Actual Stock Price and Projected Price of IBM

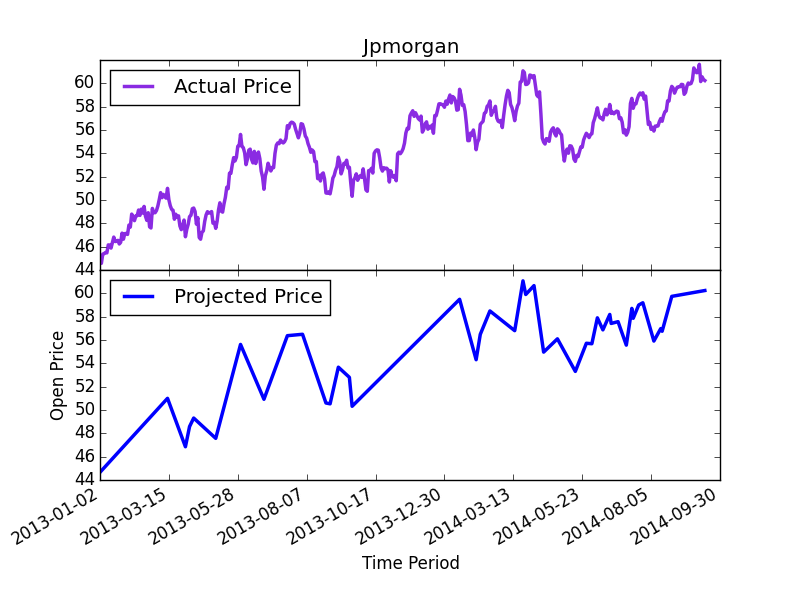


Figure 5.9 - Actual Stock Price and Projected Price of JPMorgan

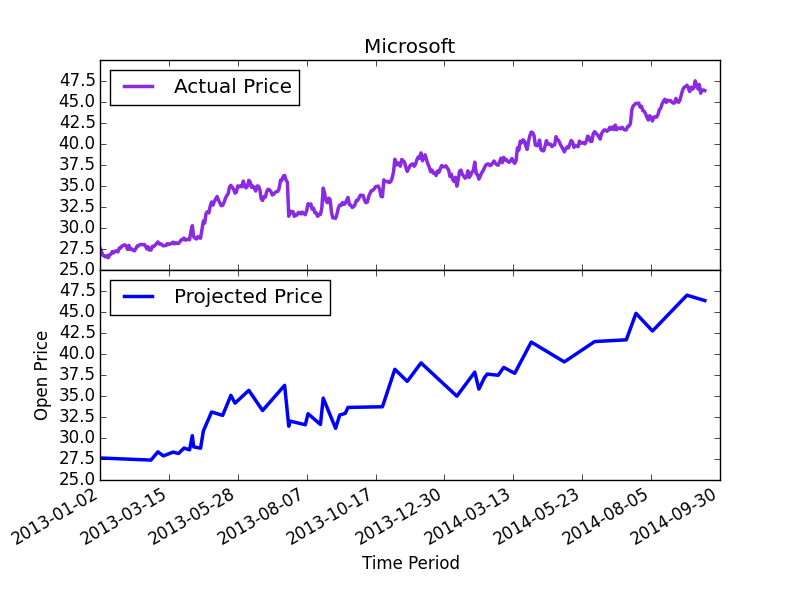


Figure 5.10 - Actual Stock Price and Projected Price of Microsoft

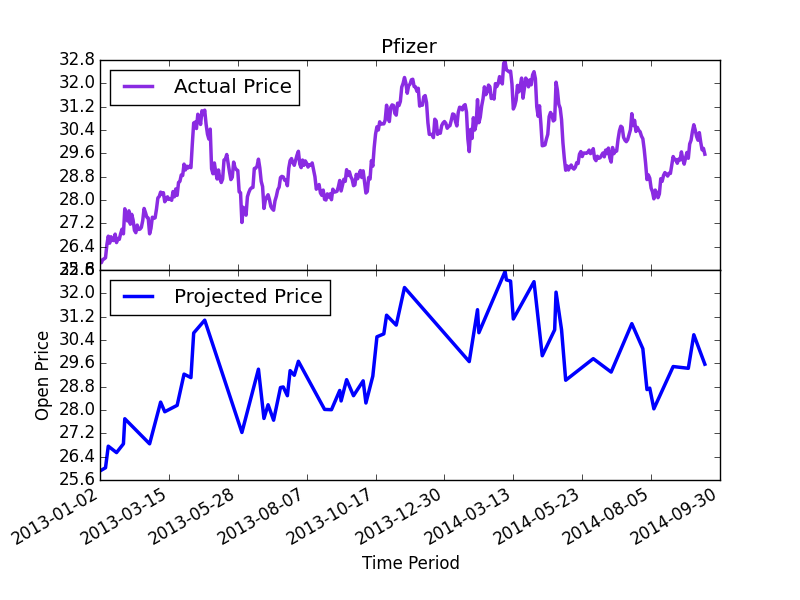


Figure 5.11 - Actual Stock Price and Projected Price of Pfizer

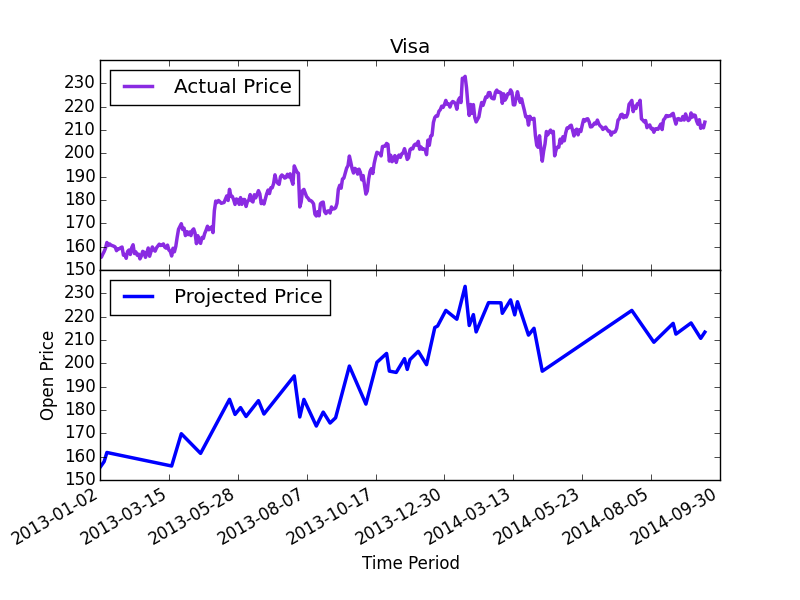


Figure 5.12 - Actual Stock Price and Projected Price of Visa

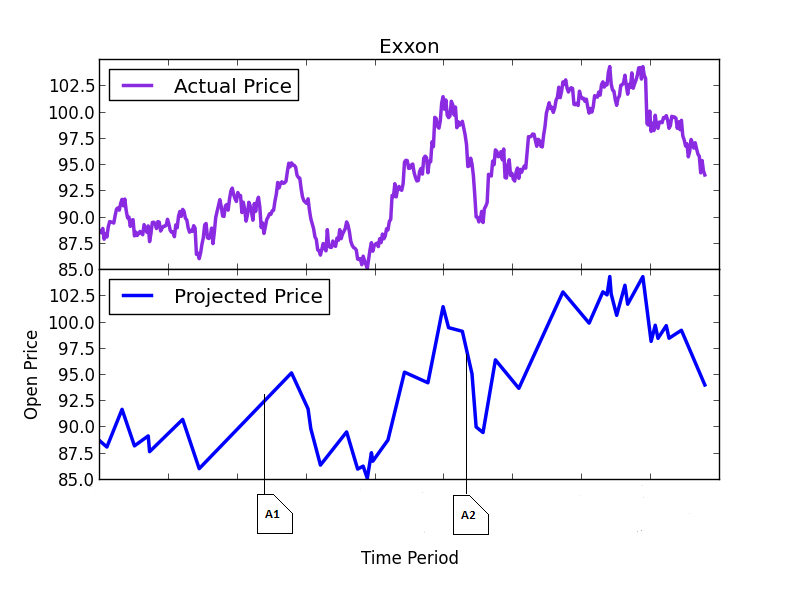
****

Figure 5.13 – Aligned News Articles with Trends

Using Figure 5.13 as a sample case, we would classify article A1 as “up” or positive while A2 would be classified as “down” or negative. After achieving this step, we can then proceed to calculate the correlation between the automatically calculated prices and the labelling. The table below shows the correlation results.

|  |  |  |  |
| --- | --- | --- | --- |
| Company Name | Correlation of Progress Sentiment(Actual) | | Correlation of Progress Sentiment(Actual) |
| Chevron | | 50.2 | 51.8 |
| Cocacola | | 32.1 | 32.0 |
| Disney | | 97.0 | 97.5 |
| Exxon | | 80.4 | 79.7 |
| Goldman | | 50.0 | 52.60 |
| IBM | | 52.12 | 53.41 |
| JP Morgan | | 85.37 | 85.14 |
| Microsoft | | 94.99 | 94.99 |
| Pfizer | | 53.08 | 53.33 |
| Visa | | 89.27 | 89.62 |

Figure 5.14 – Correlation values of the automatically generated data

Looking at the data above and comparing them, we see that the correlation values are quite similar to that of the manual data except for the correlation values of Goldman Sachs which is vastly different from the result of manual classification. The reason for this is that Goldman Sachs often is the source of the news (for example, Goldman Sachs often advises on buying and selling other companies) and the news is not about Goldman Sachs. This means that automatic labelling is blind to these issues as it labels both news *by* Goldman as well as news *about* Goldman without using any filter. However, when manually labelling, we ensure to label those articles are “neutral” in terms of progress of the entity and the relevant feeling sentiment.

#### Evaluating Labelled News Articles

Sentiment labelling generated via automatic categorisation is a reflection of the price movements, not a reflection of the articles themselves. However, since the articles themselves are manually labelled to reflect precisely the sentiment which they carry, we can conclude that if there exists a high similarity the results of manual labelling and the results of automatic labelling, then we can say confidently that the labelled articles can led to positive results in later classification.

We note that news articles labelled automatically cannot be classified based on feeling sentiment. Feeling sentiment by definition requires an evaluator to label articles based on what feelings are evoked by reading the article. However, since the method by which we automatically label stock data is based on the progression of the stock price, we can automatically label articles based on progress sentiment.

How therefore do we evaluate the results of labelling? An easy method of doing is by calculating the Pearson’s correlation coefficient.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Company Name | Correlation of Progress Sentiment(Actual) | | Correlation of Feeling Sentiment(Actual) | Correlation of Progress Sentiment(Projected) | Correlation of Feeling Sentiment(Projected) |
| Chevron | | 51.7 | 51.8 | 49.7 | 50.1 |
| Cocacola | | 32.8 | 32.6 | 32.99 | 32.84 |
| Disney | | 97.6 | 97.6 | 97.10 | 97.13 |
| Exxon | | 79.0 | 78.8 | 79.7 | 79.6 |
| Goldman | | 76.74 | 77.42 | 71.98 | 72.5 |
| IBM | | 53.2 | 53.3 | 51.7 | 51.8 |
| JP Morgan | | 84.2 | 84.4 | 84.5 | 84.7 |
| Microsoft | | 95.8 | 95.8 | 95.8 | 95.8 |
| Pfizer | | 51.1 | 51.6 | 50.9 | 51.3 |
| Visa | | 89.14 | 89.20 | 88.80 | 88.86 |

Figure 5.2 – Table of correlation between sentiment and stock price using manually labelled data

We finish off this section by pointing out that the more similar the correlation values are between the projected trend line (generated via piecewise linear approximation) and the actual stock price trend line, the higher the similarity is between the two trends.

### Data Pre-processing

Retrieved news articles are in HTML format. The news articles therefore need to be converted into plain text, tokenised (into unigrams and bigrams), stemmed and have stop words removed (discussed in chapter 3) before classification can take place.

### Document Representation

After the completion of all pre-processing steps, the documents are now ready to be transformed into vectors. The library Scikit-learn provides the TfidfVectorizer that converts the news articles into a TF-IDF-weighted document-term matrix. TF-IDF has been discussed in (chapter 3).

### Feature Selection or Reduction

In the literature, the chi-squared method for feature selection and the SVD method are popular and but we decided to use as SVD is a very expensive method.

### Classification

The final step is to train the SVM to predict the news article. In order to truly evaluate the SVM, cross validation over the data set was performed. 10-fold cross validation ensured we got the performance of the hyper-parameters of the SVM. The results of each fold were evaluated using the following metrics: confusion matrix, recall, precision and f-measure. These results can then be averaged over the number of folds to determine the overall performance of the hyper-parameters.

## Price Prediction

Price prediction is the actual step that combines the results of sentiment classification into a single result.

In order to make price predictions for each individual company, the news data have to be re-split into categories defined by the company the news is relevant to. Thus, we have 14 features for training the SVM-HMM model: 12 technical indicators(figure 4.3) and 2 news-based features (progress and feeling). For each company, 120 days of news classified by the SVM in section 4.2.6, combined with 120 days of technical data served as the test period while a definable and changeable number of days served as the training period. We say definable as we use a sliding window of training data to train the hybrid SVM-HMM model. This ensures that the model is as sensitive as permissible to the market as possible.

Manual analysis of the piecewise linear approximation-based trend line, gives an indication of how long the trends last for and thus the number of days for the training window. Once determined, we used a fixed training window width. Defining an appropriate window width has significant bearing price prediction; longer window width during times of stability is likely to lead to better results than shorter widths. Correspondingly, a shorter window width during times of rapid swings in the price might lead to better prediction. Keeping this in mind, it has to be noted that the SVM, in order to compute usable probability has to be trained on enough data, hence there is a trade-off between the number of training data that can be used for SVM training and the sensitivity to the market in the short-term. Having defined the window, the SVM is trained with data within the bounds of the window. The SVM can then provide the emission probabilities as explained in section 3.6.

Our definition of the HMM is almost complete as we have the number of states (two, each representing the upwards and downwards movement of the stock price) and the emission probabilities. The transition probability however, is still unknown so we train using the Baum-Welch algorithm for the transition probability. Finally, we can then run the Viterbi algorithm on the past days of data to predict the sequence of states for each of the past days. As we are only interested in predicting the next day’s price, we simply take the hidden state for the day as the prediction of the next day. We formalise the process of making a prediction in algorithm 4.1

|  |
| --- |
| price prediction |
| 1: **for** each trading day  2: select training window of length  3: using cross-validation and grid-search determine parameters for SVM  4: train SVM for emission probabilities using , using parameters  5: train HMM for transition probabilities using  6: hidden\_state of viterbi algorithm using past days of data  7: **yield** hidden\_state of |

Algorithm 4.1 – Price Prediction Algorithm

|  |  |
| --- | --- |
| Technical Indicator | Formula |
| Stochastic |  |
| Stochastic |  |
| Slow Stochastic |  |
| Momentum |  |
| Rate of Change |  |
| William’s %R |  |
| A/D Oscillator |  |
| Disparity5 |  |
| Disparity10 |  |
| Price Oscillator |  |
| Commodity Channel Index |  |
| Relative Strength Index |  |
|  |  |

|  |
| --- |
| where is the close price, is the highest high and is the lowest low for the last days, is the high price at time , is the low price at time t, is the 5 day moving average, is the 10 day moving average, , , , the upward price change at time and is the downward price change at time . |

Figure 3.2 – Technical Indicators and their formulas

# Experiments and Results

## Manual Classification

In this section, we detail the results of manual document classification and the settings used to achieve the results. As we use the linear classifier, the parameters that need to be set are the class weights and the cost. Other parameters to be set are default parameters by the classifier. First, we discuss the classification of progress sentiment and show the results, followed by the classification of feeling sentiment.

### Progress Sentiment Classification

In order to set the weights, we need to look at the support for each class. Figure 5.18 shows the number of articles supporting each class. The hyper-parameters used for configuring the SVM are as follows:

LinearSVC (the class used for classification) implements the one-versus-rest classifier for multiclass problems. We use a *C* value of 2.9. We use automatically set class weights (which modifies the C-values for each class) for the SVM as Figure 5.18 shows, the classes are not represented equally in the training sets. Using a StratifiedKFold cross validator, we can preserve the percentage of representation for each sample.

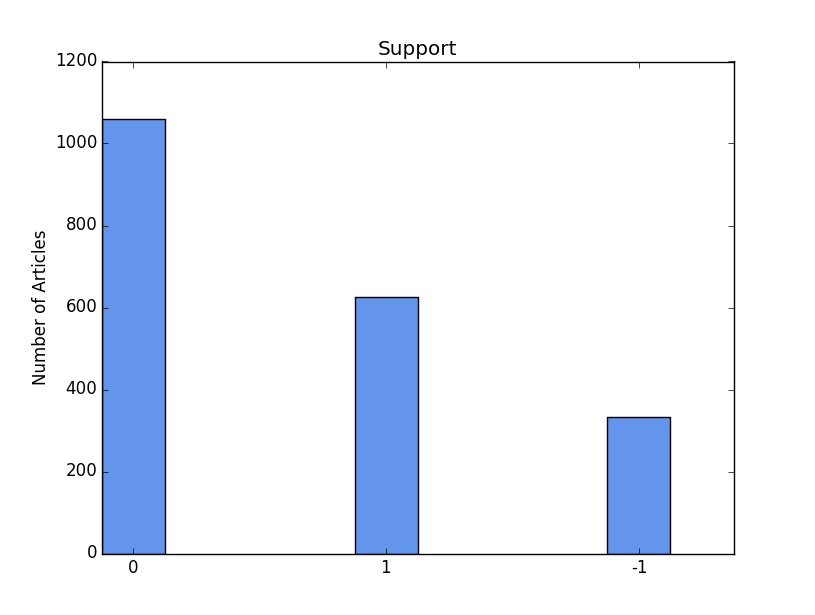
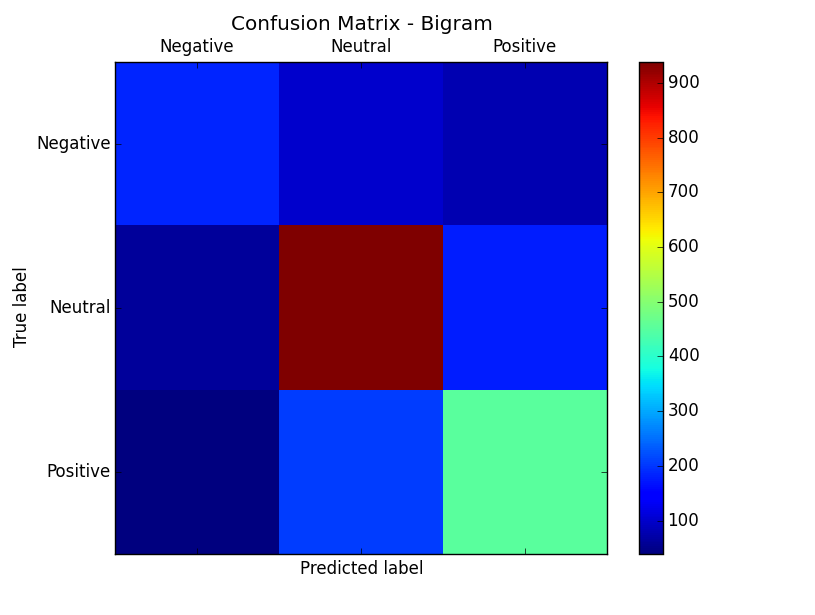
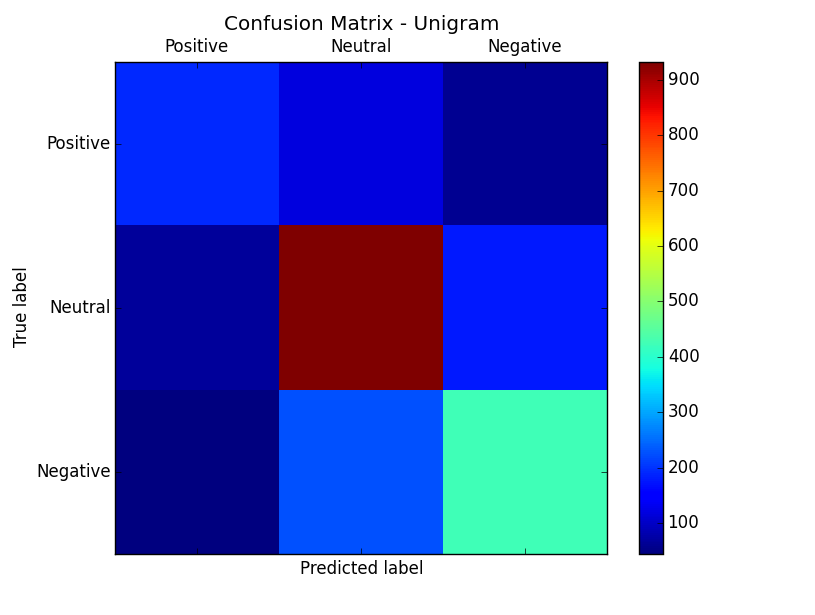


Figure 5.18 – support for the various classes (manual/progress)

Similar settings were used for bigrams, unigrams and combination experiments. In order to determine accuracy, we use cross validation and the following metrics: f-measure, recall, precision and confusion matrix. We compute the average of the scores of all the folds. The confusion matrixes for unigram, bigram and combination (Figure 5.19) show an overview of the accuracy for the three classes.

The confusion matrixes show that there aren’t very big differences in the performances of the three methods of tokenisation (except when classifying positive articles). It’s very difficult to explain why this is the case except that all three methods carry similar levels of information for this problem.



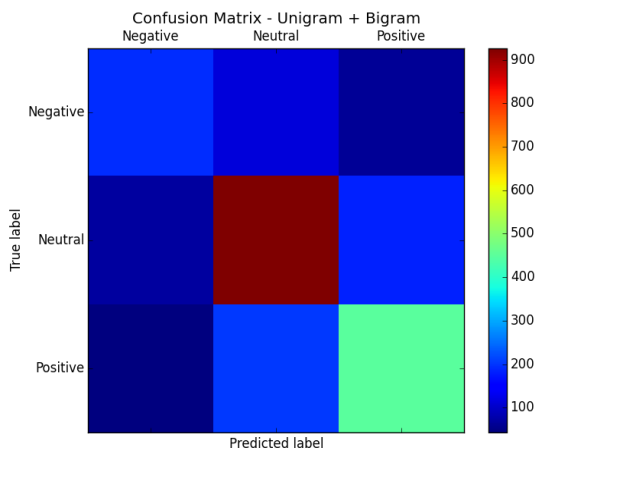


Figure 5.19 – Confusion matrices (Manual/Progress). Top left – Unigram, Top Right – Bigram, Bottom – Unigram + Bigram

|  |  |  |  |
| --- | --- | --- | --- |
|  | Unigram | Bigram | Unigram + Bigram |
| F-measure | 68.62 | 69.82 | 70.51 |
| Recall | 68.68 | 69.97 | 70.58 |
| Precision | 69.04 | 70.17 | 70.62 |

Figure 5.20 – Table of performance of linear SVM measured by cross validation (manual/progress)

Delving into the actual numbers, we see that overall, the bigram does better than the unigram and the combination of both does better than either of them singularly. Combining this information with the confusion matrix, we see that bigrams and the combination perform better due to being able to slightly classify positive news articles better.

Given the similarities in the values, T-tests were performed (with an alpha of 0.05 and a degree of freedom) to determine there are any significant differences between the results attained with the features. The t-tests confirmed that there are no statistically significant differences between the results.

### Feeling Sentiment Classification

Poorer results were achieved for the classification of feeling sentiment in general. This is contradictory to the initial belief that feeling sentiment would be easier than progress sentiment to classify. We performed classification using a linear SVM as before. The settings for feeling sentiment were quite different. In addition, classification performance for the feeling sentiment was quite poor overall. As per the previous section, we start by introducing the frequencies for the classes (Figure 5.21)

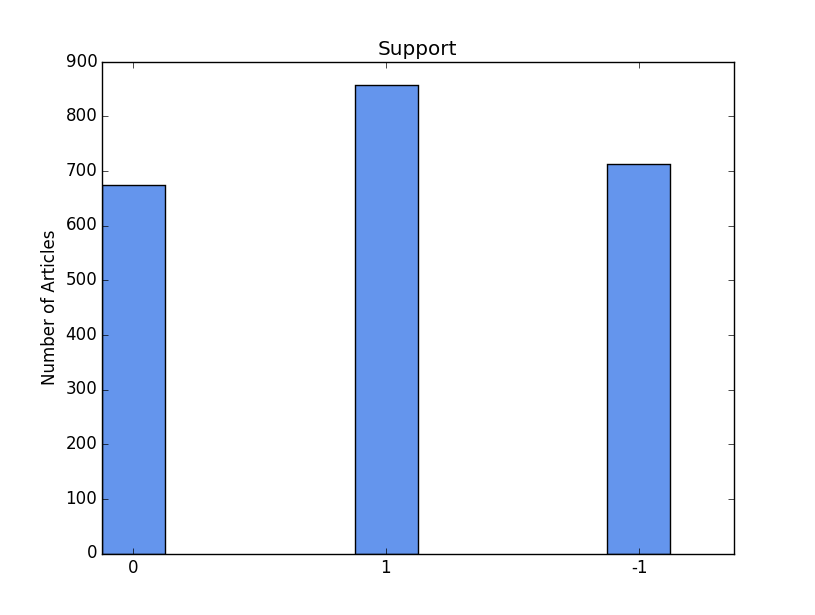
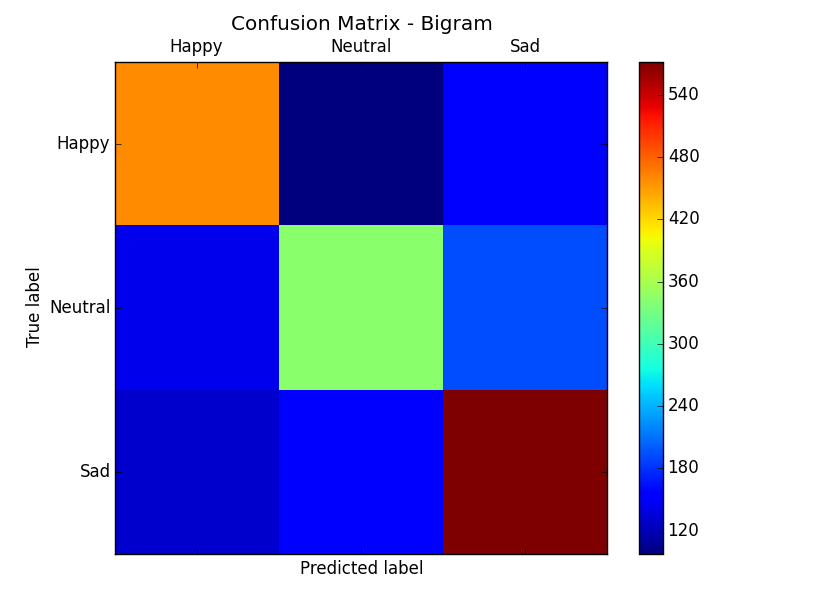
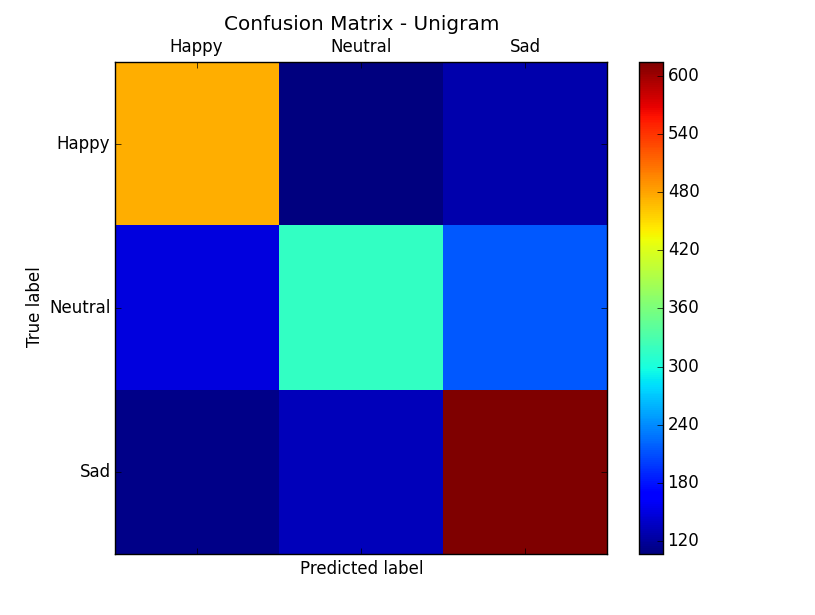


Figure 5.21 – Support for the classes (Manual/ Feeling)

Different hyper-parameter settings were used for classification. The parameter was set to higher levels with a value of. The other parameters, such as the class weight were also set automatically based on the class.

Considering the confusion matrices (Figure 5.15), we see that the all three methods of tokenising perform very similarly as before. A possible reason for this is that news articles often bear mixed feelings. On the surface, it may seem that news articles bear feeling sentiment orientations that lean towards one way or the other but this isn’t so. News articles often carry information that lean to both sides. A classic example of such news articles is articles that discuss “happy” sentiment. In a few of these articles, there’s also discussion of past “sad” sentiment that led to perhaps structural changes that result in improvement. Hence, while progress sentiment might be relatively clear, feeling sentiment can often be ambiguous when it comes to classifying neutral articles.



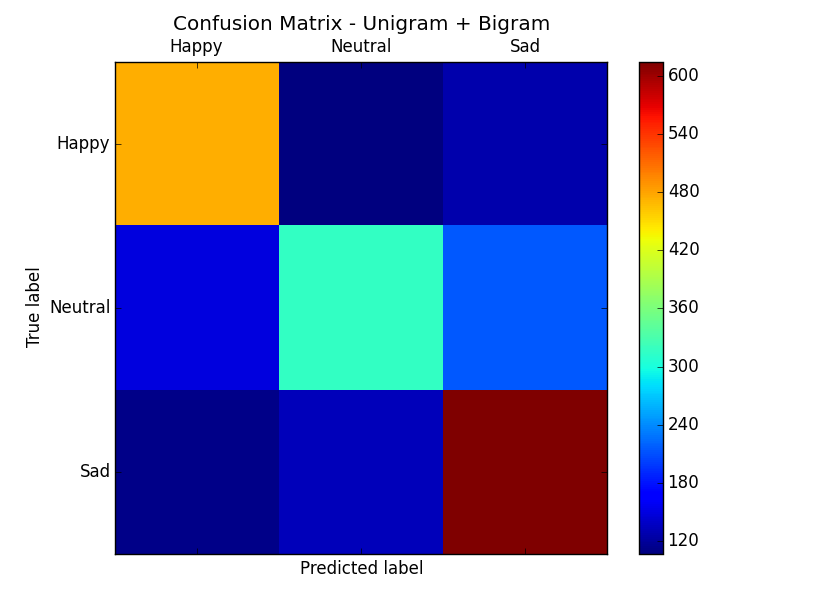


Figure 5.22 – Confusion matrices (Manual/ Feeling). Top left – Unigram, Top Right – Bigram, Bottom – Unigram + Bigram

|  |  |  |  |
| --- | --- | --- | --- |
|  | Unigram | Bigram | Unigram + Bigram |
| F-measure | 62.00 | 60.87 | 63.68 |
| Recall | 62.48 | 61.06 | 63.92 |
| Precision | 62.23 | 61.15 | 63.99 |

Figure 5.23 - Table of performance of linear SVM measured by cross validation (Manual/ Feeling)

Bigrams are typically expected to do better than unigram due to the fact they retain sentence structure but clearly, bigrams doesn’t do very well for this problem looking at the performance measures in Figure 5.16. However, combination of both performs better than either but not by much.

## Automatic Classification

In this section, we follow the same pattern as in section 5.6.1, with the exception that we only discuss progress sentiment. We use a value of 3.5. For automatic classification, neutral movements are severely underrepresented (only about 46 news articles were classified neutral); hence, we only consider positive and negative movements.

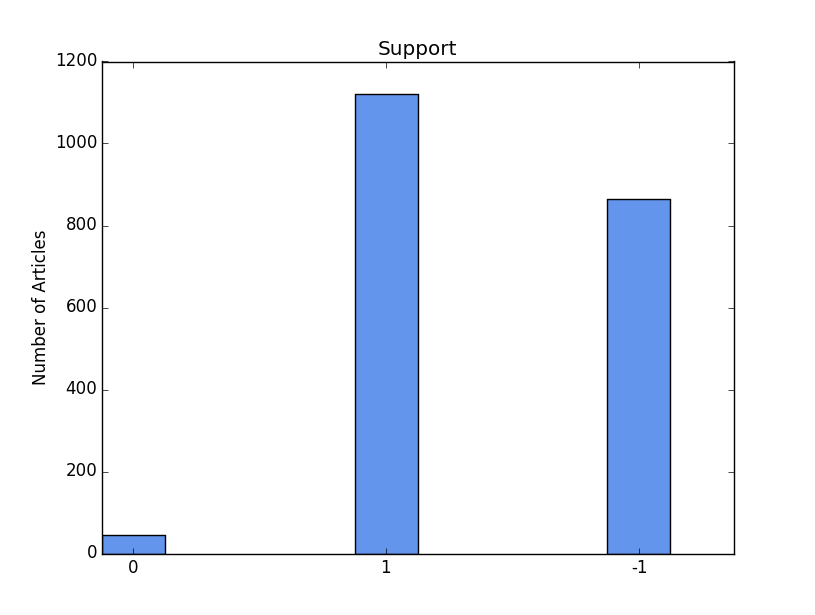


Figure 5.24 – Support for the classes (Automatic / Progress)

Here, we only present the numerical results (Figure 5.25) to restrict repetitiveness as well as the fact that the manually labelled data will be used. It would of course be interesting to consider how well automatically labelled data performs when used subsequently for price prediction, however time restraints prevent this. In addition, the intention was not to use automatic labelling for classification; instead, it provided an adequate benchmark for comparison with the results of manual labelling. We have extensively discussed the pitfalls with automatic labelling (section 5.3.) we believe that the results here can be explained by these.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Unigram | Bigram | Unigram + Bigram |
| F-measure | 64.50 | 66.30 | 64.49 |
| Recall | 67.97 | 72.55 | 61.36 |
| Precision | 61.43 | 61.12 | 68.05 |

Figure 5.25 - Table of performance of linear SVM measured by cross validation (Automatic/ Feeling)

## Price Prediction Results

We present summary statistics of the features in the appendix (excluding statistics) for each of the company analysed section 7.2.

An important aspect of the price prediction is setting the window width. For companies (Chevron, Disney, Microsoft, Visa, J.P. Morgan, Pfizer, Goldman) that show less frequent swings in their trends, we used the entire dataset available to us: That is, we train on 320 days, while predicting for 120 days. For companies that show slightly more rapid swings, we train on 120 days (6 months of data), predicting on 120 (Exxon), and finally, for the most unstable companies over the period we train on 60 days (Cocacola, IBM).

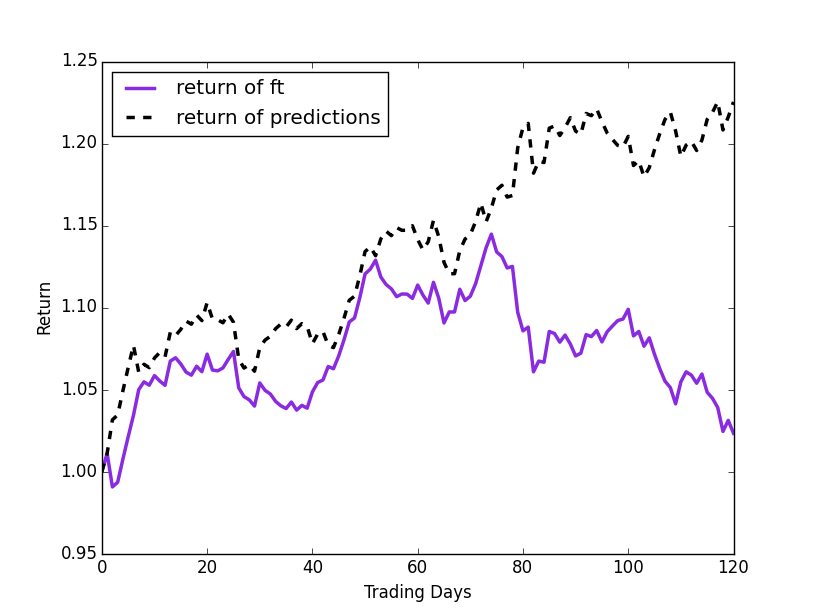
As with sentiment classification, we have opted to use a linear SVM. This again reduces the problem of setting parameters to finding the value. We employ a grid-search with cross-validation in finding the C-values. The process of finding the C value has to be repeated for every trading day. It’s easy to see how the process can be expensive during the experimental phase but daily predictions would be much faster as we’d be searching for one C-value – the value for the current day.

In order to present the results, we normalise the all close prices to a base of 1 (at the start of the 120-day period). The (financial times series) line is the return you achieve if you buy at the start of the period and hang on to stocks over the 120-day period. The simple return is calculated using the formula:

Where is the price at time . The cumulative return over a period of -days is thus calculated as:

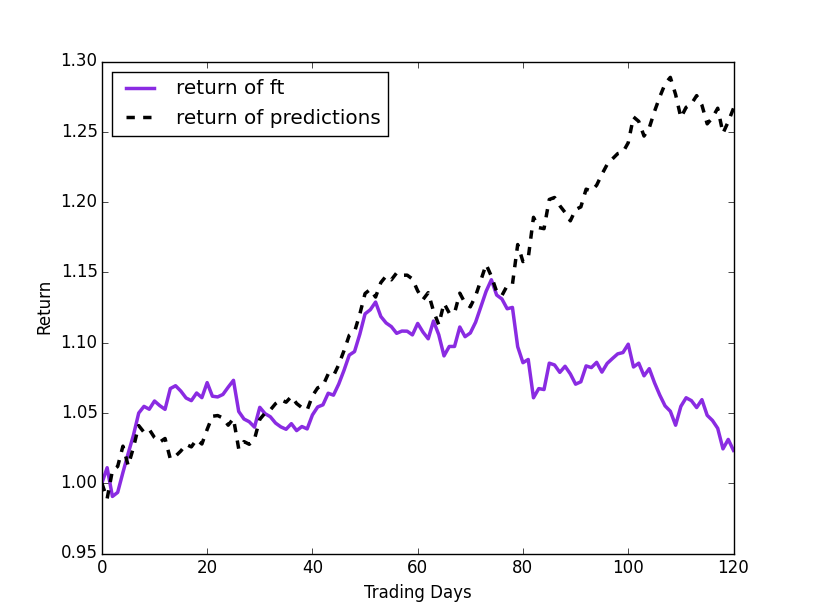
Hence, the return to be gained should one hang on to their stocks will change as the price changes (the trend line is simply a normalised version of the stock price trend line)In order to determine the return of a prediction system, we assume that trades are made based on the prediction. Should the prediction system make a false prediction, trades made based on that prediction result a negative return; however, should it make a correct prediction, trades made based on that prediction result in a positive return. We can thus use the cumulative return formula to generate the return over a period of -days.

Our experiment consists of two phases: using the SVM-HMM model inclusive of the sentiment data and without – this is in order to make a reasonable evaluation of the effect of sentiment data. Figures 5.26 to 5.45 show the results of both experiments.



**Chevron – News-based Model**

Figure 5.26 –Chevron return based on predictions vs return of time series (inclusive of news-based features)



**Chevron – Technical Indicators Only**

Figure 5.27 - Chevron return based on predictions vs return of time series

**Cocacola – News-based Model**

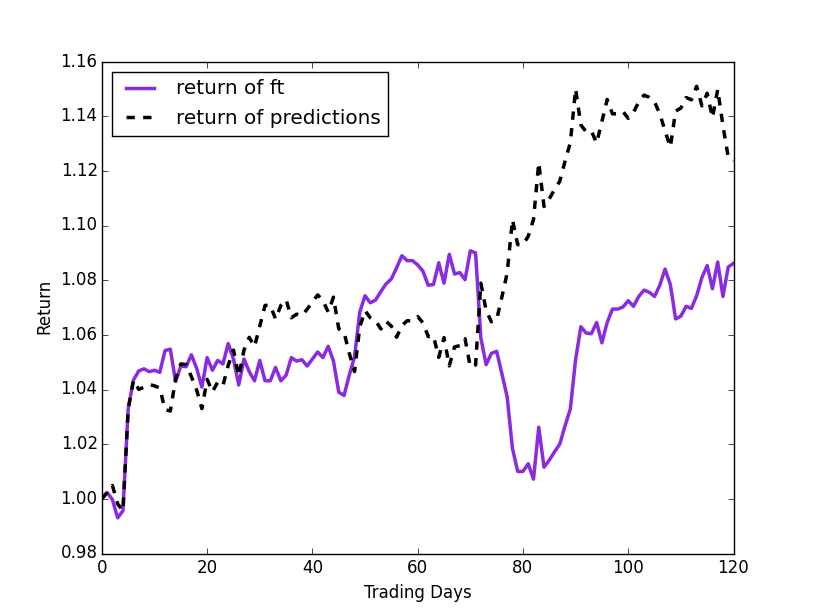
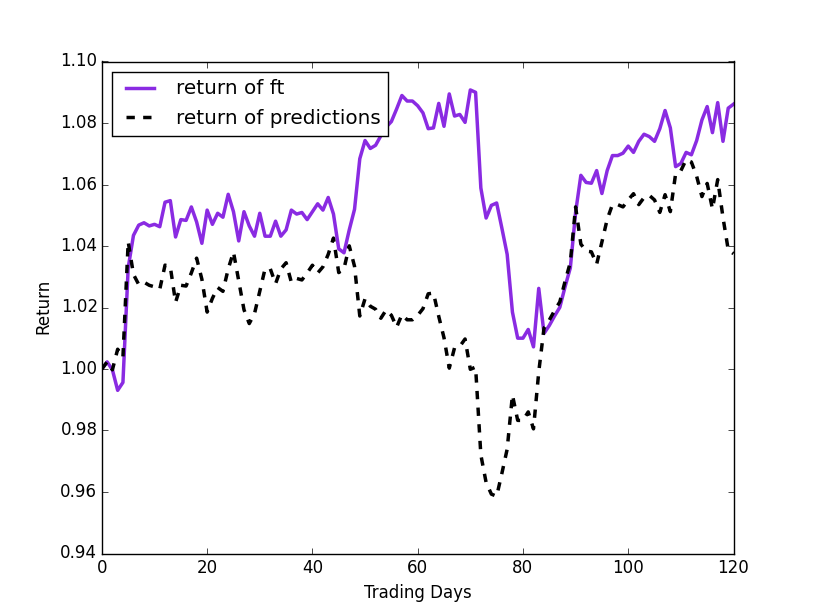
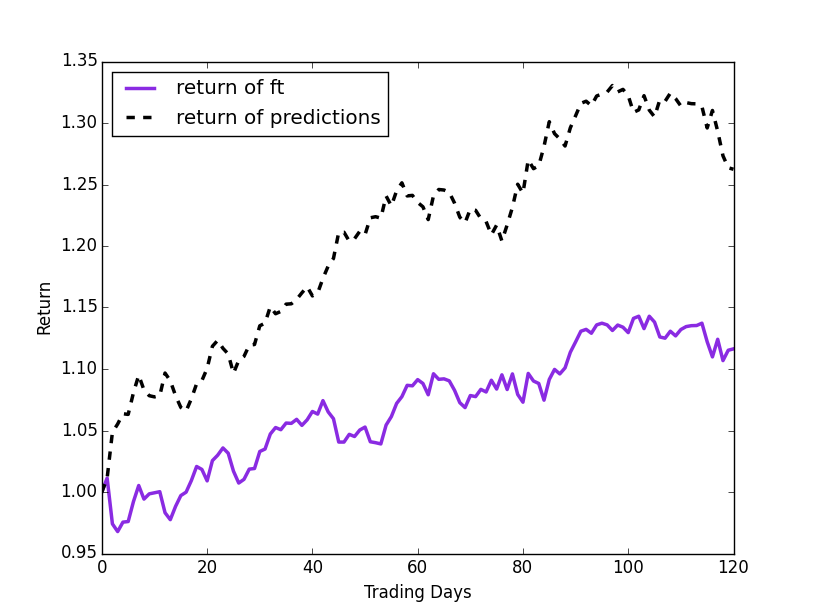


Figure 5.28 - Cocacola return based on predictions vs return of time series (inclusive of news-based features)



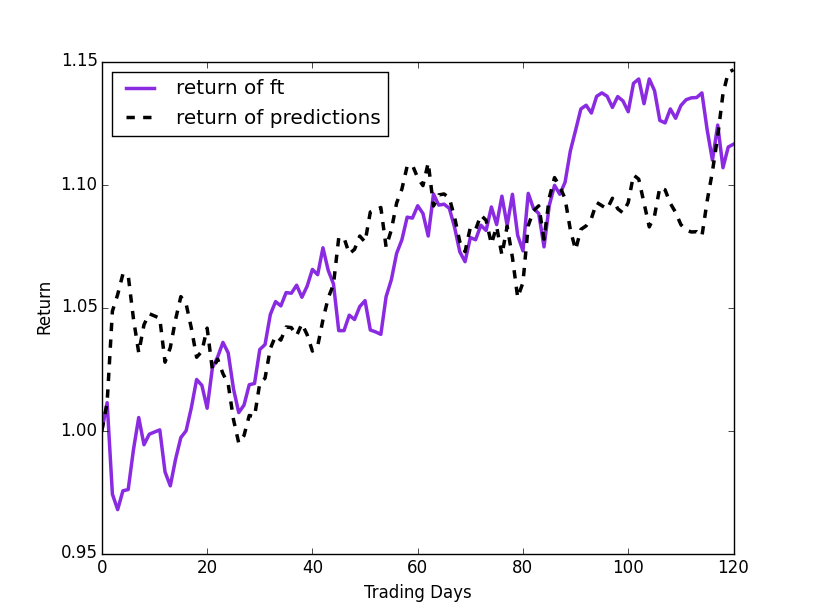
**Cocacola – Technical Indicators Only**

Figure 5.29 - Cocacola return based on predictions vs return of time series



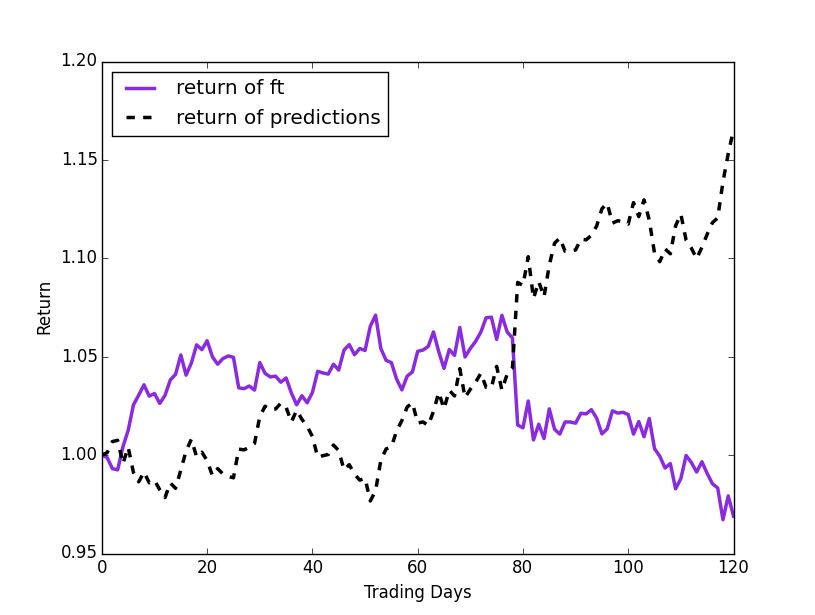
**Disney – News-based Model**

Figure 5.30 - Disney return based on predictions vs return of time series (inclusive of news-based features)



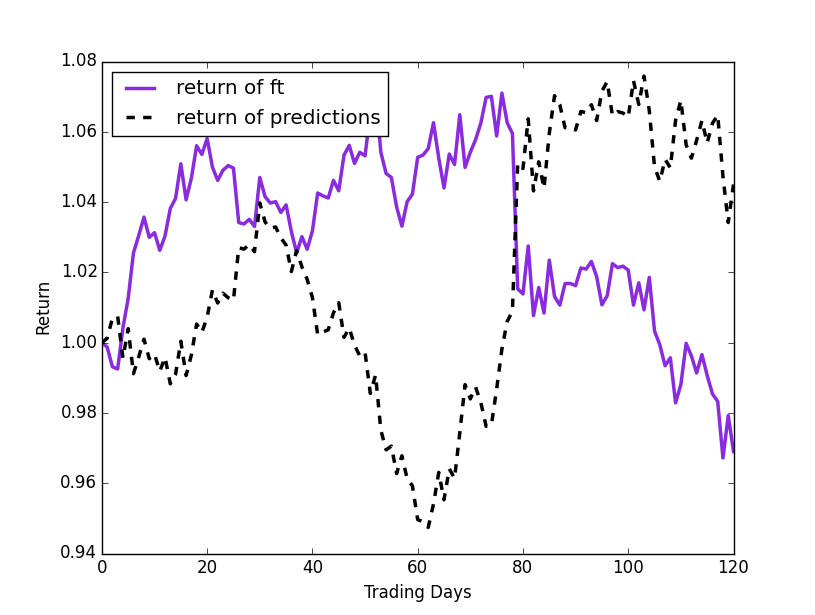
**Disney – Technical Indicators Only**

Figure 5.31 - Disney return based on predictions vs return of time series



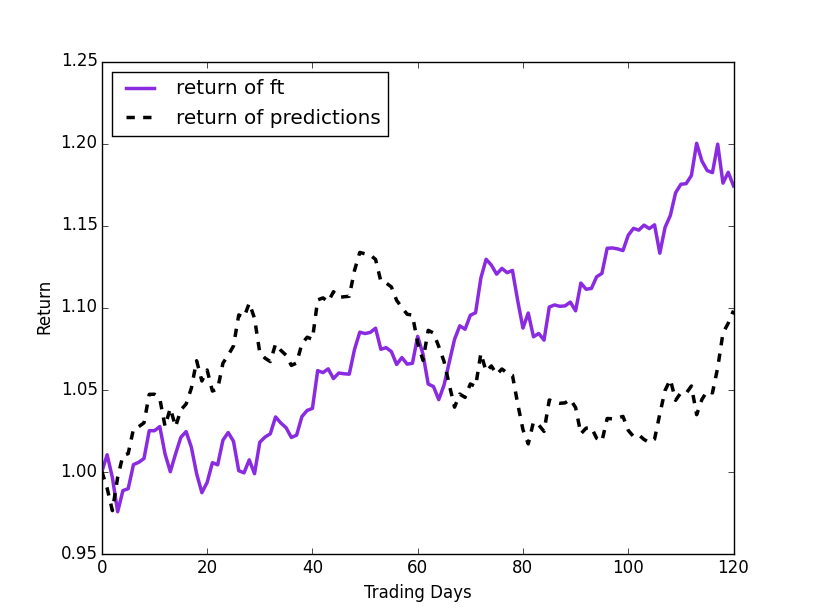
**Exxon – News-based Model**

Figure 5.32 - Exxon return based on predictions vs return of time series (inclusive of news-based features)



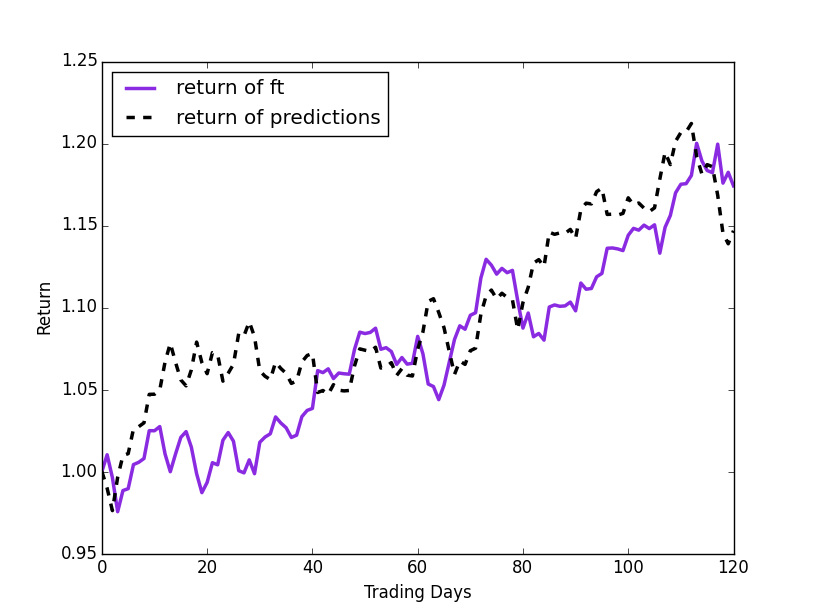
**Exxon – Technical Indicators Only**

Figure 5.33 - Exxon return based on predictions vs return of time series



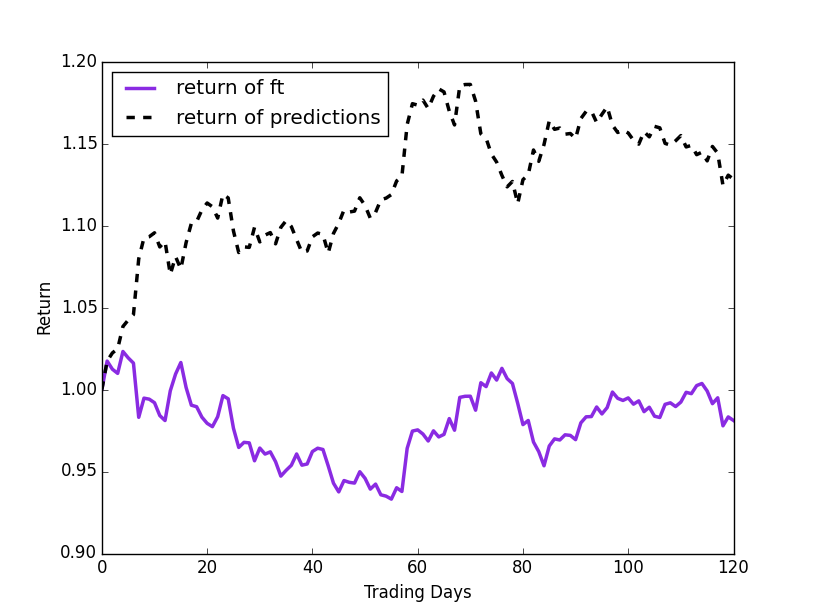
**Goldman Sachs– News-based Model**

Figure 5.34 – Goldman Sachs return based on predictions vs return of time series (inclusive of news-based features)



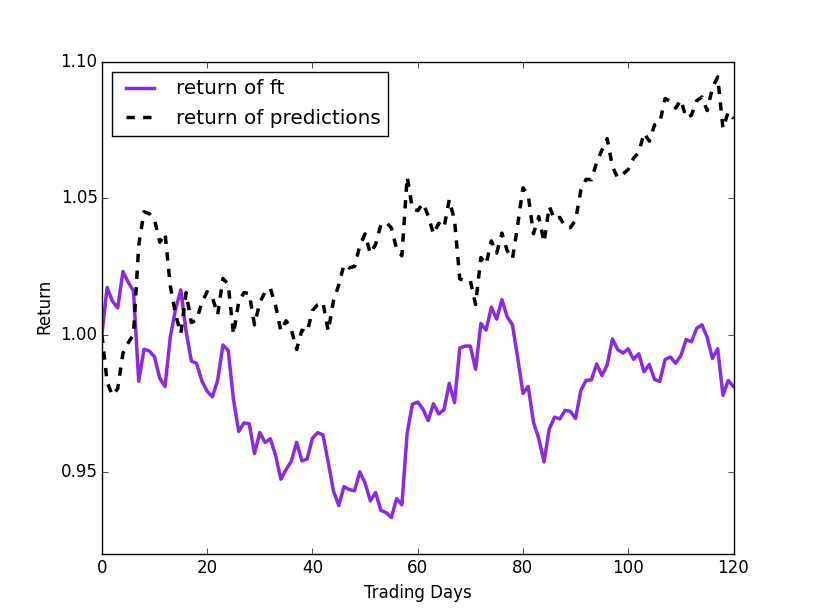
**Goldman Sachs – Technical Indicators Only**

Figure 5.35 – Goldman Sachs return based on predictions vs return of time series



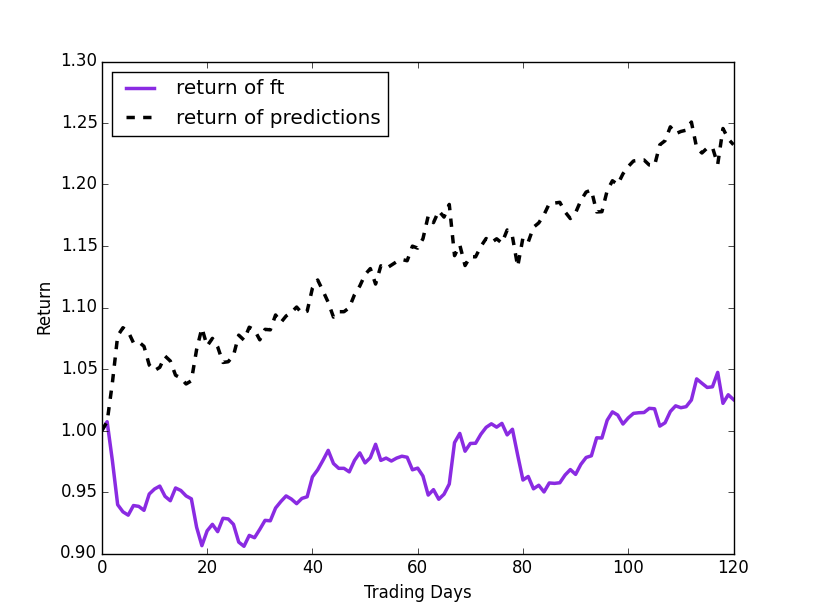
**IBM – News-based Model**

Figure 5.36 - IBM return based on predictions vs return of time series (inclusive of news-based features)



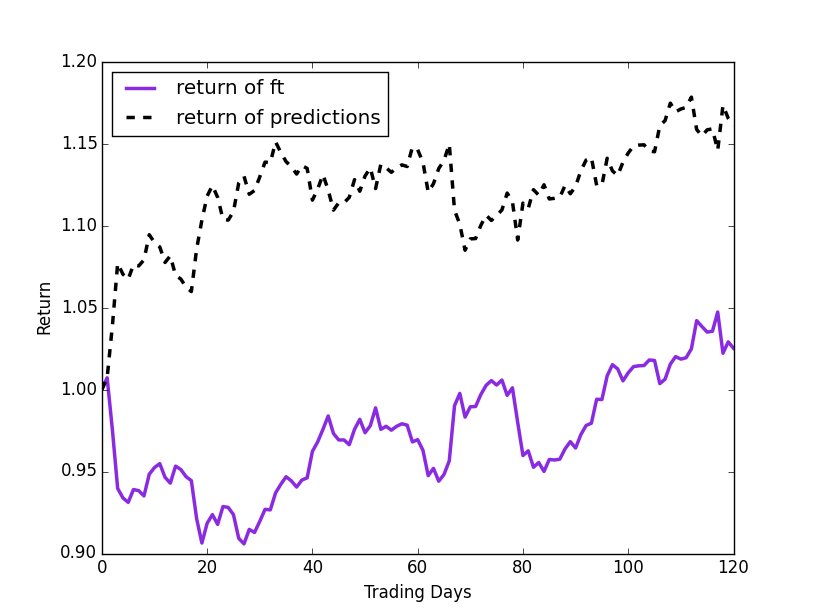
**IBM – Technical Indicators Only**

Figure 5.37 - IBM return based on predictions vs return of time series



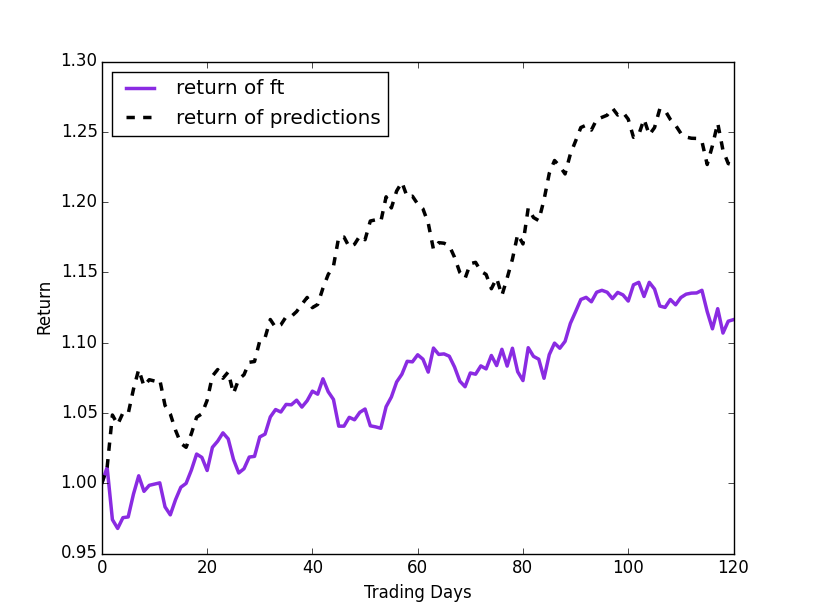
**J.P. Morgan – News-based Model**

Figure 5.38 – J.P. Morgan return based on predictions vs return of time series (inclusive of news-based features)



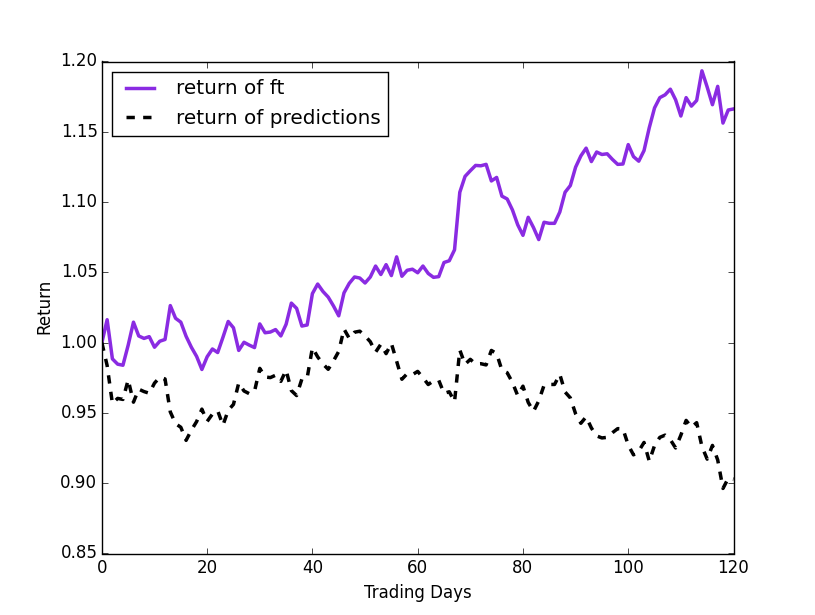
**J.P. Morgan – Technical Indicators Only**

Figure 5.39 – J.P. Morgan return based on predictions vs return of time series



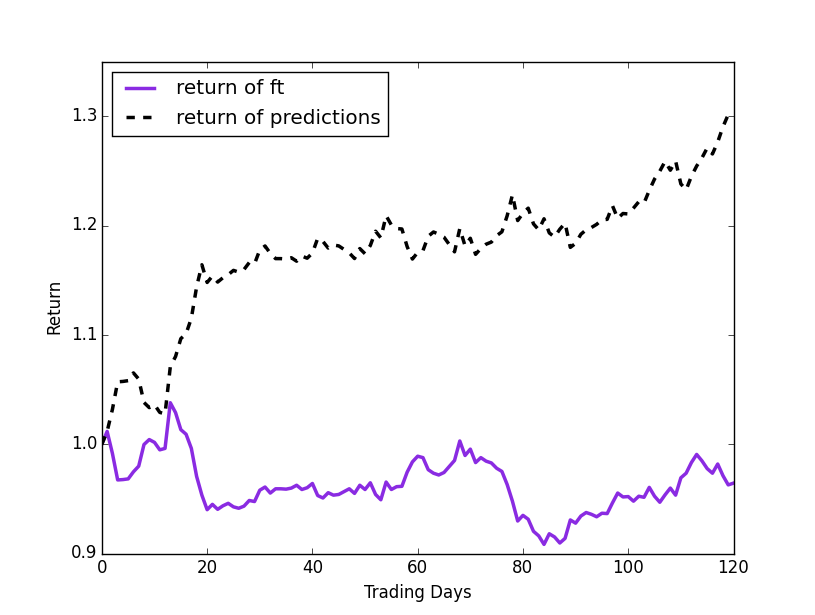
**Microsoft – News-based Model**

Figure 5.40 - Microsoft return based on predictions vs return of time series (inclusive of news-based features)



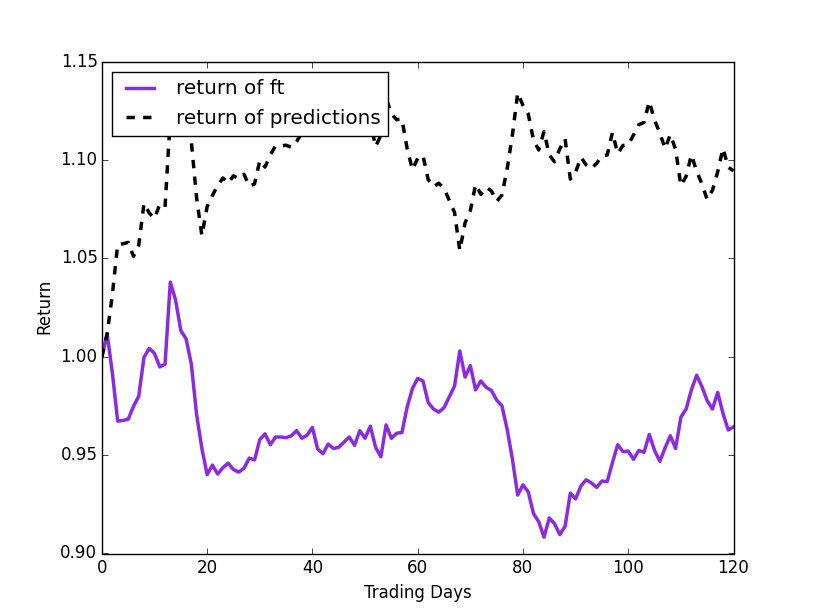
**Microsoft – Technical Indicators Only**

Figure 5.41 - Microsoft return based on predictions vs return of time series



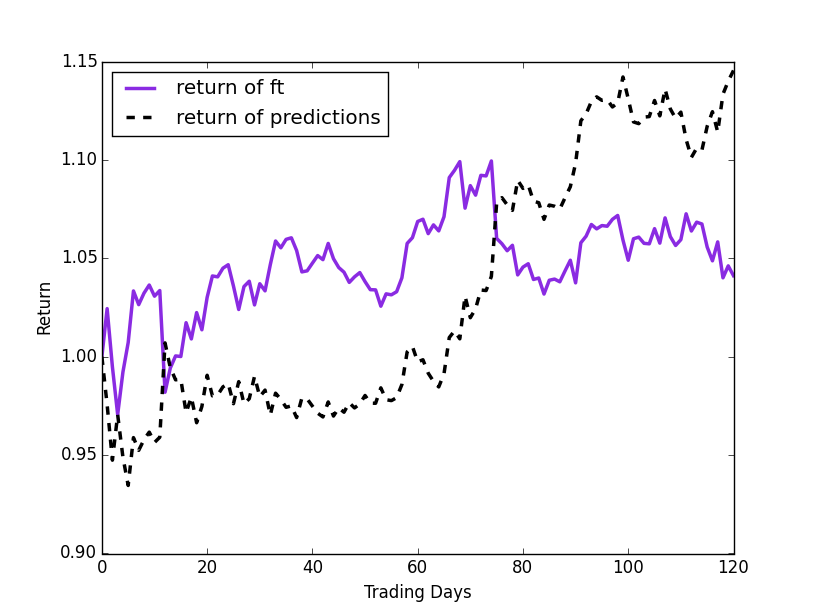
**Pfizer – News-based Model**

Figure 5.42 - Pfizer return based on predictions vs return of time series (inclusive of news-based features)



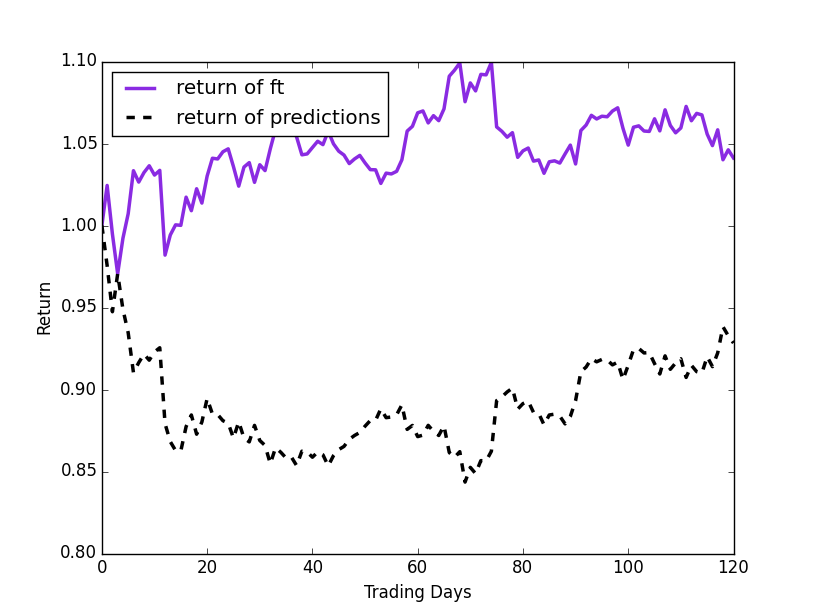
**Pfizer – Technical Indicators Only**

Figure 5.43 - Pfizer return based on predictions vs return of time series



**Visa – News-based Model**

Figure 5.44 - Visa return based on predictions vs return of time series (inclusive of news-based features)



**Visa – Technical Indicators Only**

Figure 5.45 - Visa return based on predictions vs return of time series

Examining, all the preceding images, it’s quite clear that the addition of news-based features has a real and tangible effect on the overall prediction accuracy of the model. We attempt at this juncture to provide reasons and possible justification for our results. It’s important to note that each trend line is different and we have treated each company differently, and thus each of the models have different results.

Chevron and Goldman’s news based models, experienced poorer performance than the technical based models. We propose that the reason for this is the less than perfect classification of sentiment as the other news based models experienced a better performance both than the market and the technical-only models.

We highlight Cocacola seems to be a special case of (relative) positivity given that the very little news, that the classifier has to work with. We also note that it performs better than the technical indicators only model.

In addition, we note that though the graphs seem to say otherwise, the technical-only indicators had higher than random hit ratios, ranging from 51.67% (Visa) – 61.67 Chevron), which correlates with the results in the literature. Our graphical way of showing the results, however is a better way of showing the effects of the models over time.

In general, we see that both technical indicator only models and news-based model tend to suffer a loss in performance at the same time, however, the news based model tends to suffer less and recover faster. We believe that the results aren’t only dependent on news, but also on explicitly stating the sentiment for days that there isn’t any news (as discussed in section 5.3.1.)

In conclusion, taking the hit ratios, into account, we note that adding news seems to have a stabilising effect on predictions so that highly fluctuating prices have less adverse effect on prices. Given, our 80% success rate (in terms of predictability of news vs technical indicators), we can say that our results correlate with the results of the literature review: in general news can be a reasonable basis for stock price prediction.