University of Southampton
Faculty of Engineering and Physical Sciences
Electronics and Computer Science

Sentiment analysis of financial news to predict the direction of stock market and volatility

By

Harika Yadavalli
September 2020

Supervisor: Dr. Mahesan Niranjan
Second Examiner: Dr. Luis-Daniel Ibanez

A dissertation submitted in partial fulfilment of the degree
of **MSc Data Science**

Abstract

Forecasting the behaviour of time series data in Financial markets is one of the most challenging problems that has attracted attention from many researchers and traders. The behaviour is influenced by quantitative information of the system dynamics, captured in the past behaviour and qualitative information about the underlying fundamentals arriving via news in various forms.

From past research, historical data is the key and holds a strong signal that could predict the direction of the stock market better than random. Historical data aids in identifying the patterns observed in the financial markets but it is not sufficient as the markets never follow a particular trend over the time and observed to be highly volatile, thus making the market prediction weak. There has been extensive research done to predict the direction of the stock market using **sentiments** derived from the financial news, which has a high immediate impact on the stock markets, as the investor's behaviour is impacted by sentiments from the news.

The purpose of this study is to combine both quantitative and qualitative information to predict the stock market direction more accurately.

Using the latest technologies in natural language processing(NLP) to process the financial news includes pre-trained language models like Word2Vec by Google and advanced deep learning methods include LSTM models to train the sequential data, we achieved 55% mean validation accuracy consistently better than random, while that of historical stock data is 53%. We have combined the features from text and stock data using Keras to predict the direction of the market, we achieved 58% accuracy. We evaluated the result using daily news and stock data from 2008-2016 from the reliable Kaggle data source. High accuracy results are not expected in predicting the stock market, unlike other machine learning problems, since financial markets are non-stationary and depend on the forthcoming news data and too many variables. As it is not possible to foresee far in the future, we can predict an immediate impact on the market based on the current information.

We conclude with empirical results that the information exacted from news alone can predict the direction of the stock market better than historical data alone, and the prediction accuracy is further improved when combined both the financial news and historical stock data, together as an input to the machine learning algorithms.

Machine Learning, Deep Learning, Natural Language Processing, Sentiment Analysis, Financial markets and Volatility.

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

I have acknowledged all sources, and identified any content taken from elsewhere.

I have not used any resources produced by anyone else.

I did all the work myself, or with my allocated group, and have not helped anyone else.

The material in the report is genuine, and I have included all my data/code/designs.

I have not submitted any part of this work for another assessment.

My work did not involve human participants, their cells or data, or animals.

Contents

1 Introduction

In the Finance domain, Volatility[1] is a key parameter and gives information about how the markets are spread over the period. Forecasting the Volatility, also known as **implied Volatility**[2] and the direction of the stock market is the most challenging problem and plays a vital role in many financial applications, includes Risk Analysis, Portfolio management and Option-pricing models, and widely used by the most financial institutions like investment banks and even by the sole traders.

The data generated in the financial markets is enormous and ever-increasing day-to-day, so the Big data technologies and machine learning came along, to analyse the data to produce actionable insights into the business. Despite having enormous data, prediction results are not accurate enough to the expected level, because they are observed to be Volatile. Stock markets are highly volatile and sensitive to the information from the news because the investor's decisions are impacted by the sentiments from the news. Forecasting the direction of the stock market using the qualitative information derived from the financial news has attracted the attention of more researchers and traders.

Predicting the Volatility is a regression problem and it quantifies the fragility of the stock markets but doesn't convey which direction the market is going. So predicting the direction of the stock market along with volatility, together brings valuable information to the businesses to take meaningful decisions and lay out the plans accordingly. We predict the direction of the stock market, which has only two labels either up(label=1) or down(label=0), that is the market is expected to up with the positive sentiments and down with the negative sentiments from the news-derived information.

Atkins et al.'s research prove with empirical results that "news derived information can predict the Volatility better than close price"[1]. Using the data set from Kaggle[3], we predict the Volatility and the direction of the stock market, and evaluate whether the intuition from the Atkins et al.'s[1] research holds. This data set is at the day level, which differs from Atkins et al.'s data set at a minute level. This could have a big impact on the results but we validate whether the intuition still holds with the day level news data as well. We aim to improve the prediction accuracy by including the features derived from both news and stock data, as an input to the machine learning models.

1.1 Project Aims and Objectives

We start with the intuition based on Atkins et al.'s research[1] and evaluate the results whether the financial news predicts the stock market volatility better than the close price. By considering the key takeaways from Atkins et al.'s paper, we aim to predict the direction of the stock market and volatility using the financial news as well as historical data as input to the machine learning models, to achieve best plausible prediction results.

Our objective is to improve the prediction results to the next level, by combining both the input features, i.e news and historical stock data, and using the latest technologies in Natural Language Processing, includes pre-trained language models by Google.

2 Literature Review

Return on investment is an important factor in asset management, is an indicator of how profitable an asset is relative to the total assets. Typically, these returns are considered to be volatile in nature. **"Volatility is a statistical measure of the dispersion of returns for a given security or market index"**[1], plays an important role in financial applications.

2.1 Volatility

Volatility is widely used to estimate market risk and portfolio management applications. Volatility is used to derive many financial terms, for example, Sharp-ratio, which quantifies the excess return per unit of risk(volatility) and used for comparison of investment performance to achieve the optimal Portfolio. Volatility is an important parameter for pricing the financial derivatives, commonly seen in option-pricing models. The Black-Scholes model is widely used for option pricing because it is a closed-form solution for pricing European options under restrictive assumptions explained in section 2.1.1. It was hailed as a breakthrough in this topic[2], gaining the Nobel price for its inventors.

2.1.1 Black Scholes model[2]:

The Black Scholes model is a mathematical model for pricing an Option's contract. Following are the important assumptions with Black Scholes model:

- The option is an European Option, which can be exercised only at the expiration.
- Risk free rate and Volatility(σ) of the underlying securities are known and assumed to be constant.
- Returns of underlying assets are log-normally distributed.

$$C = S_0\mathcal{N}(d_1) - K\exp^{-rT}\mathcal{N}(d_2)$$

$$d_1 = \frac{\ln(S_0/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}$$

$$d_2 = \frac{\ln(S_0/K) + (r - \sigma^2/2)T}{\sigma\sqrt{T}}, d_2 = d_1 - \sigma\sqrt{T}$$

C = Call option price
S = Current stock price
K = Exercise or strike price
r = Risk free interest rate
T = Time to expire in years
$\mathcal{N}$ = Representing the normal distribution
σ = Standard Deviation of the Returns i.e Volatility

All the parameters in Black Scholes Model are observed directly except Volatility of the underlying stocks. Traders use the Option's price as a reference to forecast the volatility, known as **Implied Volatility**[2]. **Historical Volatility** is calculated based on the past data, which can be used to predict the future Volatility, also known as **Realised Volatility**.

2.1.2 Volatility Forecasting

It is not only important to know the present value of the Volatility, but also estimate the future values to realise the risk of the underlying assets/investments. Forecasting Volatility is vital for any financial institutions, for example, investment banks for trading and portfolio management.

In the past, Volatility has been predicted mainly using the historical data observing the trends with the time series data. But in the recent times, there has been extensive research using the information derived from the news and social media because of the dependency on the external factors, for example, natural disasters like Covid-19 pandemic which impacts the global economy and so the markets. Markets are dependent on investor's behaviour, is highly impacted by the sentiments derived from news and social media. So, Sentiment analysis of the financial news and social media plays an important role in predicting the volatility and direction of the stock market as well.

We are using the following standard formula from Hull Book[2] to derive the Volatility. Volatility is defined as the variance of log returns over the chosen time period.

$$r_i = \log(\frac{S_i}{S_i - 1})$$

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(r_i - r)^2$$

S_i = Stock Price at time i
r_i = Log Returns at time i
σ^2 = Variance or Volatility
r = Mean of Log Returns over time window N.

[1]https://www.investopedia.com/terms/v/volatility.asp
[2]https://www.macroption.com/implied-vs-realized-vs-historical-volatility/

2.2 Background Research

If we look at the research done a decade back in this field, there were some traditional and robust methods to predict the market Indices and Volatility. From one of the initial research papers on forecasting Volatility[4]:

> "There is no single perfect model when compared the existing models that are widely used across to predict Volatility, includes historical moving averages, Auto-Regressive models, conditional Heteroscedastic models and Implied volatility methods. Some methods like Historical average models are simple but the results are poor, while the other methods like Implied Volatility yields improved results but difficult to implement. Exponentially Weighted and Simple Moving Average methods are both efficient and easy to implement[4].

. This paper was one of the initial research paper that proved Neural Networks can be successfully used to forecast Volatility.

Support Vector Machine stood out for its outstanding classification results as the Machine Learning algorithms evolved to perform better than traditional statistics methods in the finance domain. It is because of the position of the hyperplane maximising the margin between the two classes and also its ability to handle the high dimensional data[5]. Later Neural Networks especially Multi-Layer Perceptron, Recurrent Neural Networks(RNN) had increased popularity along with the Support Vector Machines.

Recurrent Neural Networks(RNN) are a class of Neural Networks whose connection form a directed cycle, in contrast to the feed-forward Neural Networks, where the signals are sent only in one direction. This interesting structure of the RNN makes it suitable to model the Sequential data. RNNs has gained popularity due to its appropriateness to address the time series problems. Recurrent Neural Networks with sentiment analysis performs better than simple RNN[5]. Long Short Term Memory(LSTM) is a case of RNNs with special architecture to carry long and short term memory with input and forget gates. Recent research shows that LSTM outperforms to process the sequential data and so the time series problems in the Finance[10].

Since the markets have unexpected fluctuations and are highly volatile, it was hard to forecast the direction of the stock market and volatility only using historical information. Research had been done in the early 2000s, about how markets react to the information from the news and social Media[10].

Trained RNN Model with LSTM units predicts the direction of stock price movement more accurately from observing the sentiments from the news articles[6]. LSTM Models with the Attention mechanism is seen in Speech and Image recognition but not in Finance domain. Soft Attention mechanism assigns weights to the input information instead of the hard filtering, which can make use of all news articles whether it has a small or big impact on the market fluctuation.

> "Latest research on LSTM proves that attention based LSTM performs better than widely used standard LSTM, GRU and LSTM with wavelet transform"[7].

Using the information from the news and Latent Dirichlet Allocation (LDA)[3] for topic modelling and reducing features, Naive Bayes text classifier can predict the market Volatility

better than random and benchmark results[1]. With empirical results, Atkins et al.'s paper proved that the information derived from the news itself sufficed to predict the Volatility and better than the close price. This indicates the immediate impact of the news data on the stock market and further scope to explore the more advanced methods for processing text data.

2.2.1 Atkins et al.'s paper[1]

Review summary of Atkins et al.'s paper[1] is given below, from which we carry important information, and validate the results and to further improve the prediction results by modelling Ensemble methods using Financial news data and historical data and also using the latest technologies in Natural language processing.
Important takeaways:

- Intense research has been done on the subject, focusing on the relationship between news-derived information and the Implied volatility, computed by inversion of the option pricing formulae(Black-Scholes Model[2]).
- Log normal property of the Returns has been used for volatility calculation, which compliments the model as the idea is adopted from the Black-Scholes Merton model to exercise the Option value.
- Latent Dirichlet Allocation[3] has been used for topic modelling and feature reduction. Simple yet powerful Naive Bayes model has been used for the binary classification of the text data.
- Non-stationary time-variant data has been handled using the decay function to give the highest importance to the latest news data. Empirical results show that Bi-gram models are performing better than Uni-gram or Tri-gram, to give true weight to the semantics of the financial terms.

Atkins et al.'s Dataset:
News data from Reuters US news archive and the respective stock data is taken at minute granularity for a year. Data at the minute level, helped to see the immediate effect of the news data on the stock market volatility, as the prediction of the volatility is for the next hour.

Atkins et al.'s paper proves that financial news alone predicts volatility better than the close price. We would like to validate the intuition behind this research and improve the prediction accuracy by combining the news data with the historical data as an input to the Machine Learning Models.

We can leverage the power of Ensemble models, as it can perform better than any single model, one model predicting from the news articles and the other from the historical stock data.

3 Data and Machine Learning Models

3.1 Data

3.1.1 Data Sources

Dataset[3] is taken from the reliable Kaggle[11] data source. It consists of two channels of the data.

- News Data: Historical news headlines, consists of top 25 news headlines for a single day. This data was crawled from Reddit WorldNews Channel (/r/worldnews), which were ranked by reddit user's votes. Data Range: 2008-08-08 to 2016-07-01
- Stock Data: Historical Stock data was taken from Yahoo Finance. Dow Jones Industrial Average(DJIA) can be analysed to prove the concept. Data Range: 2008-08-08 to 2016-07-01

These data sources are used to predict the Volatility and direction of stock market, which can be calculated as shown in the Data Labelling section 3.1.2.

3.1.2 Data Labelling

Our aim is to predict the direction of stock market and Volatility using the financial news information. Labels are derived from the historical Stock data as shown in the Figure 12, which has labels at day level.

- **Volatility** is calculated as variance of the log returns over the last 10 days window, using the formula given in the section 2.1.2.
- **Direction of the market** is calculated as up(label=1) if the close price increases tomorrow compared to today's value and down(label=0) if the close prices reduces tomorrow.

To achieve Labelling the news data for each day, which is available as 25 news article per day. We combine the top 25 news articles into one single sequence of sentences per day. From the above labels at the day level, we join both data sets using the date. Combined data would have Top 25 news headlines in a single row with respective labels for that day.

3.1.3 NLP-Data Preprocessing

Data Pre-processing is vital and an important first step when analysing the data to bring meaningful insights or to train the Machine Learning Models to learn the business goals. Data needs to be cleansed, transformed and integrated with the right labels.

After pre-processing the data, it can be split into train and test data sets, and train the Machine Learning models to achieve the business needs. Financial text data needs to be cleaned according to the business needs as well, following pre-processing steps are done to process the news data.

[3]https://www.kaggle.com/aaron7sun/stocknews

Figure 1: Sample code for Data Labelling

```
import numpy as np
import pandas as pd
y= pd.read_csv("/content/Labels.csv")

#Volatility
y['log_returns'] = y['Close'].rolling(window=2) \
                             .apply(lambda x: np.log(x.iloc[1]/x.iloc[0]))
y['variance'] = y['log_returns'].rolling(window=10).var() # ==> Variance calculation
y['variance'].fillna(0, inplace=True)

# Direction of the market
y['market_direction'] = y['Close'].rolling(window=2) \
                                  .apply(lambda x: 1 if x.iloc[1] >= x.iloc[0] else 0)
y['market_direction'].fillna(0, inplace=True)

# Labelling
y_Volatility = y.reset_index()['variance'] *1000
y_Direction = y.reset_index()['market_direction']
```

- Remove the special characters including the digits as well.
- Remove the stop words, which are the most common words in a language, such as "the", "a", "an", "in" in English language.
- Stemming is a process of removing the suffix from a word, to represent its root word. It reduces the number of unique words in the data set but over stemming loses the contextual meaning of the word. We have used PorterStemmer as shown in Figure 2.
- Keras Tokenizer with maximum of 10000 words as input features.
- Padding: As some sentences are longer while others are shorter, we set the maximum sequence length to a fixed number of words while processing the row. Padding trims the very long sentences and appends zeros to the shorter ones. We chose the maximum sequence length as 350 based on the data distribution as shown in figure 3, which is appropriate to consider the majority of the data. Sample code is given the Figure 4.
- Integrate with the right labels, derived in the Data Labelling section 3.1.2. We have seen some missing values when Variance is calculated, which needs to be cleaned by filling the missing values with next variance value or mean of it. We have filled with zeros using fillna() method in the above code.

After pre-processing, we have the data with size $X_{1989\times350}$, i.e 1989 rows, and each row with 350 words, representing the sequence of top 25 headlines and respective labels y_{1989}, for Volatility as well as for the direction of stock market.

Figure 2: Sample code for Data pre-processing

```
import nltk
from nltk.corpus import stopwords
from nltk import stem
nltk.download('stopwords')

news_data =pd.read_csv("/content/drive/My Drive/Combined_News_DJIA.csv")
x= news_data.drop(['Label','Date'], axis=1)
x.columns = [str(c) for c in range(25)]

for i in x.columns:
    df = x[i].str.strip()
    df = df.str.replace(' ', '_')
    df= df.str.replace(r'b\'', '')
    df= df.str.replace(r'b\"', '')
    df = df.str.replace(r"[^0-9a-zA-Z\_]+", "")
    df = df.str.replace('_+', ' ')
    x[i]= df.str.lower()

x["combined_news"] = x.apply(lambda x: ''.join(str(x.values)), axis=1)
stopwords = nltk.corpus.stopwords.words('english')
ps = stem.PorterStemmer()

def clean_stem_stopWords(line: str ):
  return ' '.join(ps.stem(word) for word in line.split() if not word in stopwords)
x["combined_news"] = x["combined_news"].apply(lambda x: clean_stem_stopWords(x))
```

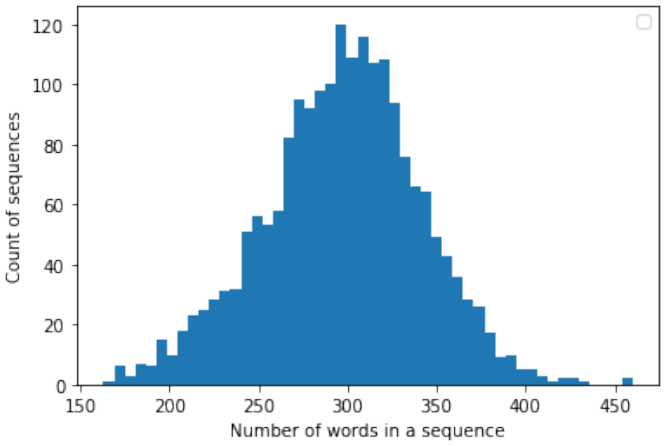


Figure 3: Histogram showing the distribution of sentence lengths in words of the data used over the period (2008-08-08 to 2016-07-01)

Figure 4: Code for word tokenization and padding

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences

MAX_SEQUENCE_LENGTH = 350
MAX_NUM_WORDS = 10000

tokenizer = Tokenizer(num_words=MAX_NUM_WORDS)
tokenizer.fit_on_texts(x["combined_news"])
data = tokenizer.texts_to_sequences(x["combined_news"])
data = pad_sequences(data, maxlen=MAX_SEQUENCE_LENGTH, padding='post')
```

4 Implementation methodologies

4.1 Word Representations

Text data(X) consists of word tokens after pre-processing as mentioned in the above Data Processing section 3.1.3. These words need to be represented as numerical features, which can be consumed by the machine Learning models. Some of the widely used word representations are given below.

- Bag of Words(BoW):
  Bag of words model is the very simplest form of the text representation in numbers, which can be in simply count or frequency of the words in the sequence. Unique words in the dataset act as features and occurrence of the word as the value for that feature. This is the quickest way to implement but has got drawbacks that the number of features increases as the new words appear, making the data matrix very sparse and also doesn't carry the semantic meaning of the words. We have implemented a simple bag of words with Bi-grams, which are good at carrying the contextual meaning and limiting the maximum number of features to 1000.

  $CountVectorizer(ngram\_range = (2, 2), stop\_words =' english', max\_features = 1000)$

  Even simple Logistic Regression has done a great job to predict the direction of the market, as shown in the table17.

- TF-IDF:
  "Term frequency-inverse document frequency is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus."[4] TF-IDF has some advantage over a simple bag of words, as it can tell the importance of a word in the document, but still carries the other drawbacks, lacking the semantic relationship between the words in the words dimensional space. Used the following sklearn class to convert the word tokens into a vector representation.

  $$sklearn.feature_extraction.text.TfidfVectorizer$$

- Word2Vec[13]:

  > "The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence. As the name implies, word2vec represents each distinct word with a particular list of numbers called a vector. The vectors are chosen carefully such that a simple mathematical function (the cosine similarity between the vectors) indicates the level of semantic similarity between the words represented by those vectors"[5].

We used google's pre-trained Word2Vec model, which carries the semantic meaning of the words in the language dimensional space.

4.2 Word Embedding

In natural language processing, word embedding is a collection of language modelling and feature learning techniques where words are mapped to vectors of real numbers. Word2Vec is one of the most popular pre-trained word embedding developed by Google[12], trained on Google news dataset. In our analysis, we considered to use Google's pre-trained Word2Vec model with 300-dimensional representation, since using the pre-trained vector representation, machine learning models can quickly learn as the word vectors carry the semantic meaning and simple mathematical operations works.

4.2.1 Word2Vec Implementation

We got the data $X_{1989\times350}$, which needs to be represented with pre-trained Word2Vec. For each unique word in our dataset, we capture the vector representation of 300 dimensions from pre-trained Word2Vec model and index it and save in a matrix named *embedding_matrix*. The data(X) carries the index of the word in the *embedding_matrix* instead of actual word. We use the *embedding_matrix* as initial weights to Embedding layer in keras, which could convert the input data(X) into respective 3-dimensional representation, that is each row with size of 350×300.

Sample Keras code to download Google's pre-trained Word2Vec model and Embedding matrix creation is given Figure 5.

Embedding layer in Keras needs 3-dimensional input.

- Total number of unique words
- Embedding dimension size(size of the word vector, i.e 300 in this case)
- Maximum sequence length(chosen as 350)

Figure 5: Code for Word2Vec *embedding_matrix*

```
#Download pre-trained Word2Vec model from google
from gensim.models import KeyedVectors

EMBEDDING_FILE = '/content/drive/My Drive/GoogleNews-vectors-negative300.bin'
word2vec = KeyedVectors.load_word2vec_format(EMBEDDING_FILE, binary=True)

# prepare the Embedding matrix
nb_words = min(MAX_NUM_WORDS, len(word_index))+1
print("num_words",nb_words)

embedding_matrix = np.zeros((nb_words, EMBEDDING_DIM))
for word, i in word_index.items():
    if i >= MAX_NUM_WORDS:
        continue
    if word in word2vec.vocab:
      embedding_vector =word2vec.word_vec(word)
      if embedding_vector is not None:
          # words not found in the word2vec vocabulary will be all-zeros.
          embedding_matrix[i] = embedding_vector
```

Finally, it needs the *embedding_matrix* as weights, as shown in Figure 6, which can be used by the machine learning models as an input layer and convert the input data with pre-defined sizes in the word embedding layer. Maximum sequence length is an optional parameter in Keras Embedding layer as Keras Sequential models can accept the variable length of sequences.

Figure 6: Code for Keras Embedding Layer

```
#Embedding layer
embedding_layer = Embedding(nb_words,
                            EMBEDDING_DIM,
                            weights=[embedding_matrix],
                            input_length=MAX_SEQUENCE_LENGTH,
                            trainable=False)
```

[4]https://en.wikipedia.org/wiki/Tf-idf
[5]https://en.wikipedia.org/wiki/Word2vec

4.2.2 Language Models[6]

Investigated the latest pre-trained language models for transfer learning and fine-tuning the models with the available financial news data. Bidirectional Encoder Representations from Transformers (BERT) is a pre-trained unsupervised natural language processing, a recent paper published by Google.

> "BERT is pre-trained using deep bi-directional representations by jointly conditioning on both left and right context in all layers in the network. So BERT can be fine-tuned with just one additional output to create state-of-the-art models for a wide range of machine learning problems"[13].

Due to time limitation, implementation was incomplete, but there is a future scope to analyse the news data and improve the prediction results further.

4.3 Machine Learning Models

Implemented the following machine learning models based on the objectives.

Volatility	**Direction of Stock market**
Predicting Volatility is a regression problem.	Predicting the direction of stock market is a classification problem.
Implemented sklearn models: Linear Regression, KernelRidge with rbf kernel, Support Vector Regressor(SVR), M	Logistic Regression, Support Vector Classifiers(SVC), Random Forest Classifier.
Implemented deep learning models: Multi Layer Perceptron (MLP with dropout and l2 Regularizer) and Long Short Term Memory(LSTM with dropout) using linear activation in the output layer with *mean_square_error* as the loss function for the regression problem.	Multi Layer Perceptron (MLP with dropout and l2 Regularizer) and Long Short Term Memory(LSTM with dropout) using sigmoid activation in the output layer with *binary_crossentropy* as the loss function for the classification problem.

4.3.1 Sklearn models

The segment of code given as an example below in Figure 7 gives the general setting of training and testing models with k-fold cross-validation. Sklearn offers the same structure to develop the various models used in this study, for example, regression problem to predict the volatility as well. K-fold cross-validation is implemented to compare the mean validation accuracy across the multiple Machine Learning algorithms.

[6]https://towardsdatascience.com/text-classification-with-nlp-tf-idf-vs-word2vec-vs-bert-41ff868d1794

Figure 7: Code for sklearn models

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import KFold, cross_val_score
from matplotlib import pyplot as plt
from sklearn.svm import SVC

# select the model
model1= LogisticRegression(random_state=0, max_iter=150)
model2 = SVC(kernel='rbf', max_iter=150)
model3 = RandomForestClassifier(n_estimators=200,criterion='entropy')

#Cross validation
cv = KFold(n_splits=10, random_state=1, shuffle=True)
scores = cross_val_score(model1, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
#Show box plot of K-fold validation accuracy
plt.boxplot(scores)
```

4.3.2 Keras deep learning models

The segment of code given as example below in Figure 8 gives the general setting of training and testing models of the deep learning models using Keras as it follows similar structure to develop various deep learning models used this study.

For **classification problem**, used *binary_crossentropy* as loss function and 'sigmoid' activation function in the final output layer.
For **regression problem**, *mean_squared_error* as loss function and 'linear' activation function in the final layer. Adam and Stochastic gradient descent(SGD) algorithms are used to optimise the models. Implemented Keras Dropout layer and l2 kernel regularizer to avoid the models being over-fitting.

Figure 8: Code for Deep Learning models using Keras

```
from keras.models import Sequential
from keras.layers import Dense, Embedding, BatchNormalization, Dropout

input_shape = ( 100,)

# Create the model
model = Sequential()
model.add(Dense(50, kernel_regularizer='l2', input_shape=input_shape, activation='relu')
model.add(Dropout(0.2))
model.add(Dense(10, kernel_regularizer='l2', activation='relu'))
model.add(Dropout(0.2))

# binary_crossentropy and sigmoid activation function for classification problem.
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# mean_square_error and linear activation function for regression problem.
model.add(Dense(1, activation='linear'))
model.compile(loss='mean_squared_error', optimizer='adam', metrics=['mse'])
#fit the model and statistics are stored in history for analysis
history = model.fit(x_train, y_train, epochs=50, batch_size=10, verbose=1, validation_sp
```

4.3.3 LSTM

"Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning"[16]. LSTM deals with the vanishing gradient problem and its special architecture have unique characteristics, making the LSTM cell to carry the long term dependency[17], and also the introduction of the forget gate[18] to remove the short term dependencies. These special characteristics making LSTMs to predict well with the sequential data, for example, time-series data and language modelling. For these reasons, LSTM is best suited for stock and news data analysis in our case.

Implemented simple LSTM models to learn the sequential text data along with Word2Vec word embedding layer. Segment of the implementation code in Keras is given in the Figure 9.

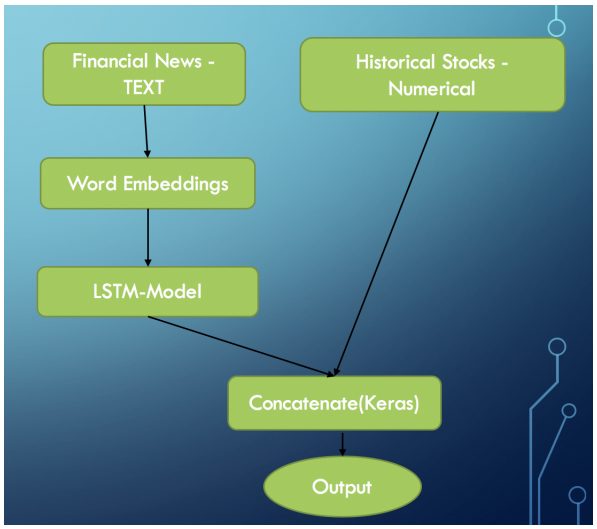
Figure 9: Sample Code for LSTM using Keras

```
# LSTM model
model = Sequential()
model.add(LSTM(30, dropout=0.15, activation ='relu', input_shape=input_shape))
model.add(Dense(1, activation='sigmoid'))
```

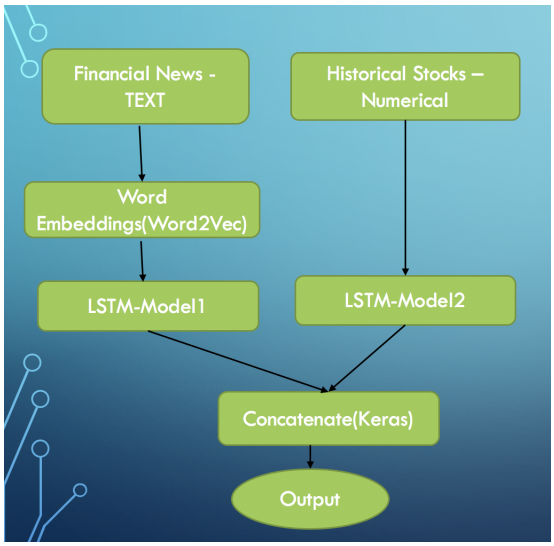
4.3.4 Ensemble Learning

"In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone"[15]. Adopting the idea behind the ensemble learning, trained one LSTM model with news data and another LSTM model with stock data alone and combined the outputs from these models as an input features to MLP layer to predict the objective, to achieve the better results.

The following diagram explains how the datasets are combined. Used powerful Keras "concatenate" layer to merge the features from a different type of sources.



(a) Concatenate LSTM features derived from TEXT directly with numerical features



(b) Concatenate two LSTM models, one trained by text and other by numerical features

Figure 10: Design diagram to understand how various input data types can be merged to provide an input to ML model.

Given the equal importance to the news data as well as stock data, by taking the equal number of features from both the LSTM models before concatenation. Implementation of this architecture in Keras code is given below.

[7]https://towardsdatascience.com/combining-numerical-and-text-features-in-deep-neural-networks-e91f0237eea4

Figure 11: Code for merged models using Keras concatenate

```
from keras.models import Sequential, Model
from keras.layers import Dense, Embedding, LSTM, Input, concatenate

EMBEDDING_DIM=250
embedding_layer = Embedding(nb_words,
                            EMBEDDING_DIM,
                            weights=[embedding_matrix_pca],
                            input_length=MAX_SEQUENCE_LENGTH,
                            trainable=False)

#Model1 trained using the news data
nlp_input = Input(shape=(MAX_SEQUENCE_LENGTH, ), dtype='int32')
embedded_sequences = embedding_layer(nlp_input)
model1 =LSTM(10, recurrent_dropout=0.2)(embedded_sequences)


#Model2 trained from the stock data
meta_input = Input(shape=(4,1), name ='price')
model2 = LSTM(10, recurrent_dropout=0.2)(meta_input)

#Concatenate the two models
concat = concatenate([model1, model2])
classifier = Dense(10, activation='relu')(concat)
output = Dense(1,activation='sigmoid')(classifier)

#define the model with actual inputs and output
model = Model(
    inputs=[ nlp_input,meta_input],
    outputs=[output]
)

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

#print the model
print(model.summary())

# Fit the model
history = model.fit([X_train1, X_train2],y_train,validation_data= ([X_test1, X_test2],\
y_test),epochs=50,batch_size=10,verbose=1)
```

5 Results

Implemented various methods of word representations as discussed in the section 4.1, and trained the machine learning algorithms as discussed in the section 4.3. Finally, alidated the results by comparing the mean validation accuracy of the models.

We have categorised the results into three different sets, based on the Input features. Analysed the text data using bag of words and Word2Vec word representations and the results are compared.

Following code snippets are given in Figure 12 are used to capture the learning curves and box plots for the validation accuracy.

Figure 12: Code snippets used for the comparison of the results

```
#Plot the accuracy graph
from matplotlib import pyplot
pyplot.plot(hist.history['accuracy'])
pyplot.plot(hist.history['val_accuracy'])
pyplot.ylabel('Accuracy')
pyplot.xlabel('Number of epochs')
pyplot.legend(['train', 'validation'], loc='upper right')

#Boxplot - validation accuracy while training
plt.boxplot(history.history['val_accuracy'])
```

5.1 News data - Bag of Words

In this section, we processed only news data as input features to predict the objectives, i.e Volatility and the direction of the stock market. Bag of words and TF-IDF word representations are used, but both produce similar results and the differences are not noticeable with the prediction results. We have observed the following results when predicting the volatility.

5.1.1 Volatility

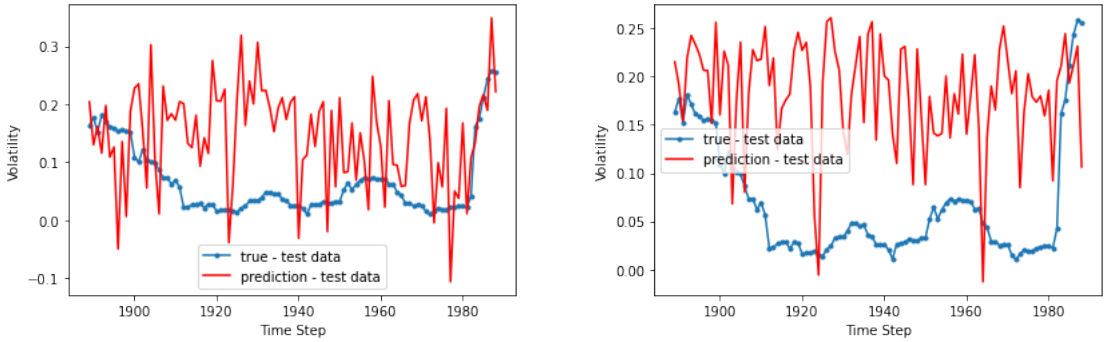
Comparison of the results for the various Machine Learning models are given below, to predict the Volatility based on news data alone. We have used standard scalar for the input data(X) and scaling for Volatility is done by multiplying with 1000. We used the same scaling to compare the results across various machine learning models. Comparison of the models with Mean square error and respective standard deviation with 10-fold cross-validation is given in the Figure 13.

Comparison of the actual with the predicted results for the Volatility test data is given the figure Figure 14.

Observation: From the figure Figure 14, MLP is able to learn that Volatility can not be negative, where as Linear Regression fails to learn this important information.

ML Algorithm	Train MSE	Train Std (+/-)	Validation MSE	Test Std (+/-)
Linear Regression(*fit_intercept=True*)	0.1149	0.0012	0.1526	0.1198
KernelRidge	0.1408	0.0016	0.2197	0.1491
SVR	0.9629	0.1146	1.2149	0.1198
MLP(layers-50-10-1, with dropout=0.2)	0.1453	0.0014	0.1472	0.1253
LSTM(layer-30-1, with dropout)	0.1453	0.0057	0.1558	0.0236

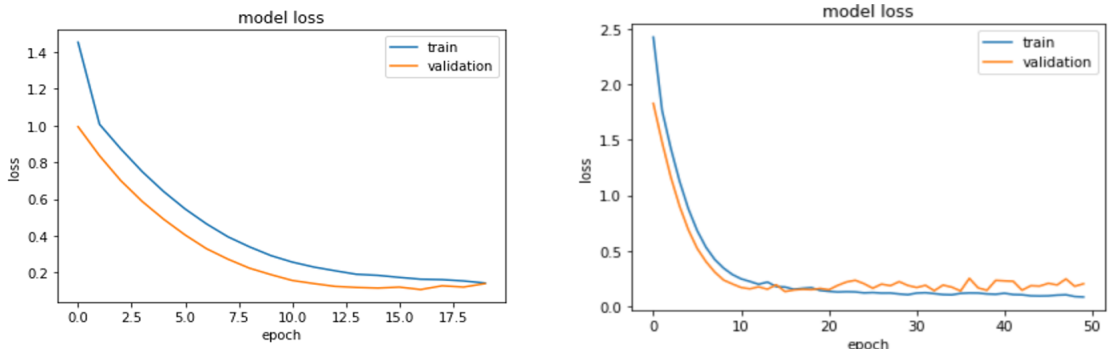
Figure 13: Comparison of the models to predict Volatility using only news data as an input



(a) Linear Regression with bias term (b) MLP

Figure 14: Comparison of the Actual Vs Predicted Volatility for test data using Bag of Words

Convergence results of the MLP model with a simple bag of words are shown in the figure Figure 15. As the number of epochs increases the model starts to overfit as shown in Figure 15(b).

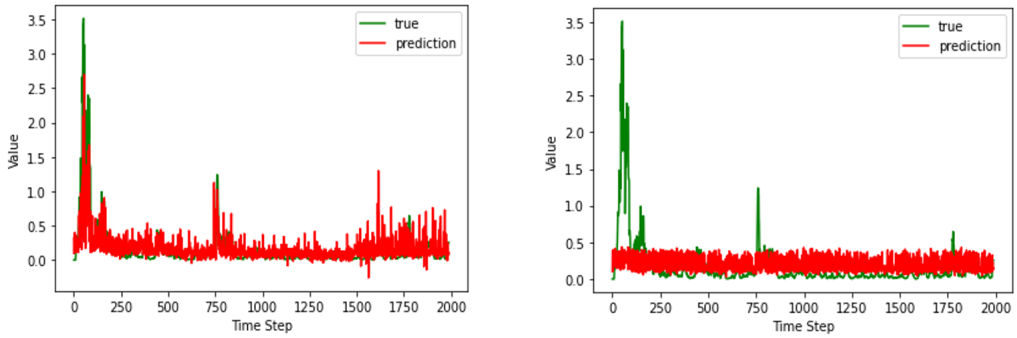


(a) MLP - Loss Convergence (b) MLP - Over-fitting

Figure 15: Convergence graphs when Predicting Volatility using Bag of Words and MLP

Comparison of the MLP prediction results for Volatility validation dataset with that of LSTM model is shown in the figure Figure 16.

Observation: From the figure Figure 16, MLP model is trained quickly to converge



(a) MLP - Actual Vs Predicted (b) LSTM - Actual Vs Predicted

Figure 16: Predict Volatility - Bag of Words

whereas LSTM model is unable to learn from the data as there is no sequential dependency between the input features with the CountVectorizer word representation.

5.1.2 Direction of the Stock market

Principle Component Analysis(PCA) has been applied to reduce the number of features from 10000 to 50 to compare the performances of various models. We observed that validation accuracy of the models with 50 features is as good as with all the features.

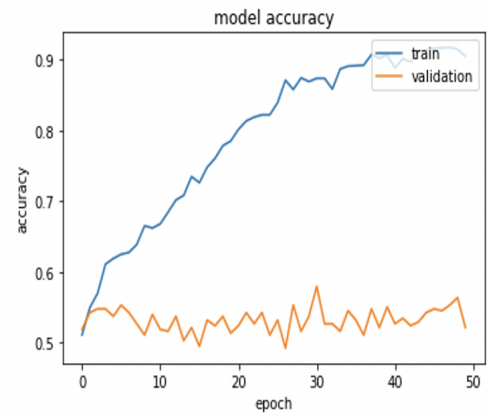
Comparison of the mean validation accuracy with respective uncertainty(σ) for various Machine Learning models are given in the figure Figure 17.

ML Algorithm	Validation Accuracy	Std (+ / -)
Logistic Regression	0.516	0.0407
SVC	0.53	0.0323
RandomForestClassifier	0.537	0.0461
MLP(layers-50-10-1, with dropout=0.2, & l2)	0.519	0.0141
LSTM(layers-30-1, dropout=0.2)	0.53	0.0099

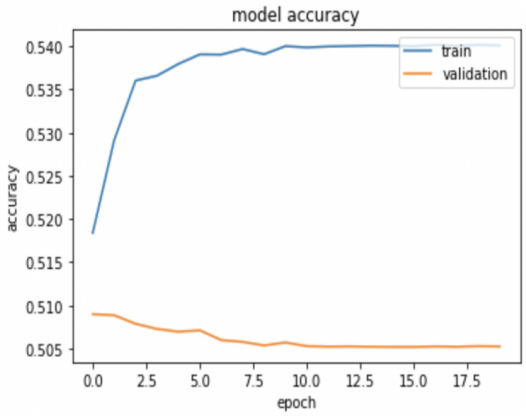
Figure 17: Accuracies on the validation set for various models when predicting the direction of the stock market using news data.

Comparison of the learning curves for both MLP and LSTM models are given in the figure Figure 18.

Observation: From the figure 16 and 18, simple MLP is able to learn the data with a bag of words better than LSTM for the same number of epochs. This shows that LSTM needs sequential dependency of the words, which is a drawback of a bag of words model and also requires huge dataset to learn the language.



(a) MLP



(b) LSTM

Figure 18: Predict direction of the stock market - Comparison of MLP and LSTM model's Learning Curves using Bag of words.

5.1.3 Direction of Volatility

Comparison of the results to predict the direction of Volatility for machine learning models are given in the figure Figure 19

ML Algorithm	Validation Accuracy	Std (+/-)
Logistic Regression	0.515	0.0407
SVC	0.53	.0323
RandomForestClassifier	0.534	0.0397
MLP(layer-50, layer-10, with dropout=0.2, l2)	0.502	0.0304
LSTM	0.527	0.0004

Figure 19: Comparison of mean validation accuracy of various machine learning models when predicting the direction of Volatility using news data.

Observation: Direction of the volatility prediction results are not good when compared to the direction of the stock market, which is opposed to the intuition taken from Atkins et al.'s paper. We compared our dataset with Atkins et al.'s experiments, both datasets are not at the same time granularity. This infers that news data has immediate impact on the market, which is unable to predict well when the volatility is defined over 10 days window. But news data at day level predicts the stock market direction far better than random and historical stock data, shown in figure 25.

5.2 News data - Word2Vec

Using Google's pre-trained Word2Vec[8] model, LSTM model learns from the first epoch itself and the prediction results are improved since pre-trained weights are carrying the semantic meanings of the words and context.

5.2.1 Volatility

Convergence graph of LSTM model with Word2Vec representation is shown in the figure Figure 20.

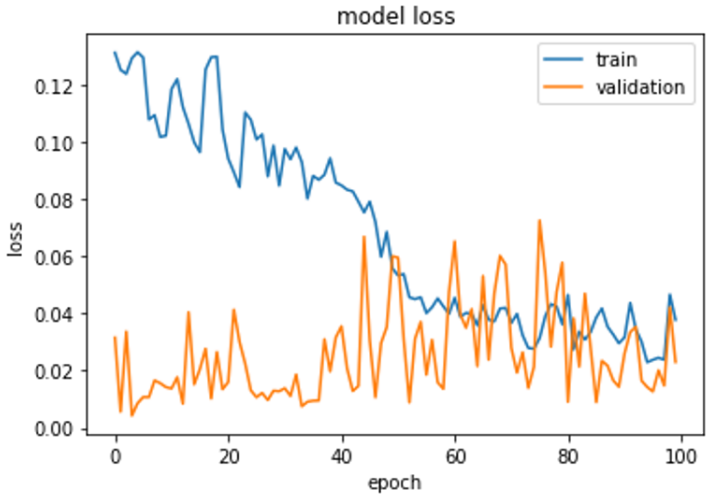
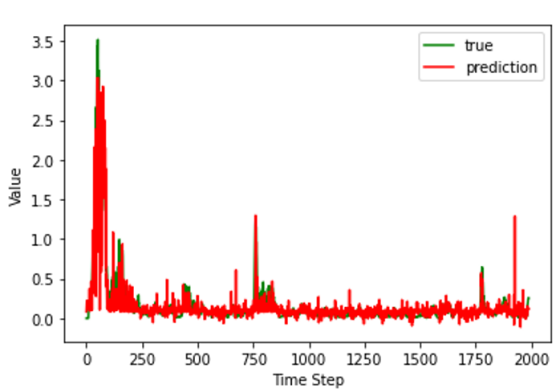
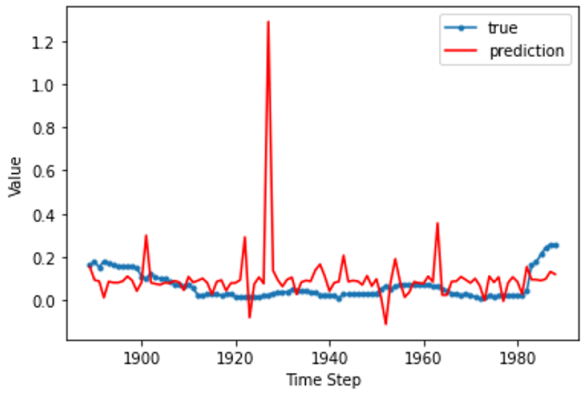


Figure 20: Predict Volatility - Convergence of LSTM model with Word2Vec

Comparison of the actual Volatility values with the prediction results is given in the below figure Figure 21.



(a) LSTM - Compare Train and test dataset



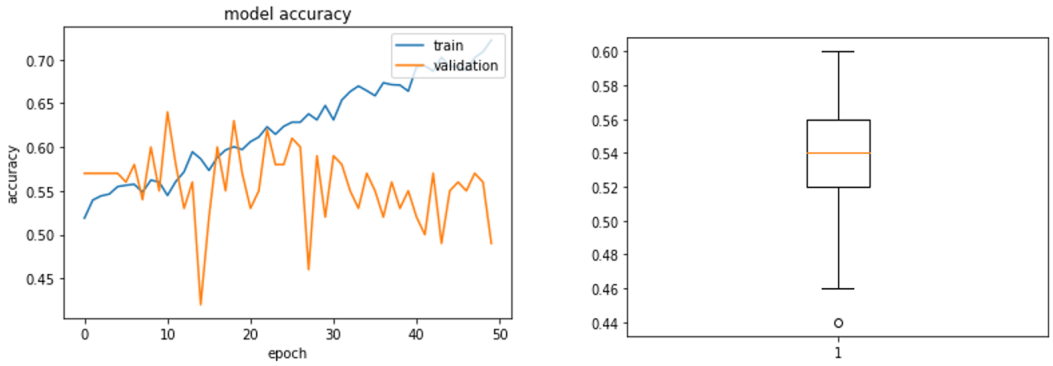
(b) LSTM - Compare test dataset

Figure 21: Compare Actual Volatility against prediction results using LSTM model with Word2Vec word representation.

[8]https://code.google.com/archive/p/word2vec/

5.2.2 Direction of the Stock market

Prediction results for the direction of the stock market with LSTM model using news data alone are shown in the figure Figure 22.



(a) Accuracy Graph (b) Box plot - Validation Accuracy

Figure 22: Predict of direction of the Stock market with LSTM model using Word2Vec to represent news data.

Observation: Mean validation accuracy of Direction of the stock market is better than random and when using stock data only, shown in the figure 25 and the results are observed to be consistent across the multiple runs.

5.3 Stock data

Predict the Volatility and direction of the stock market only using the historical stock data. We used historical stock data for past 100 days as an input.

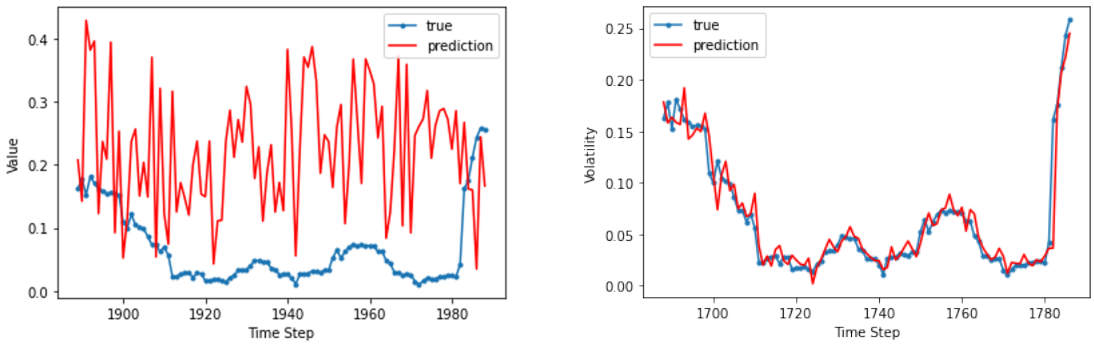
5.3.1 Volatility

Observation: Stock data is good at predicting the volatility and prediction results are better than the results from the news alone information, given in the section 13

ML Algorithm	Train MSE	Std (+/-)	Validation MSE	Std (+/ -)
Linear Regression	0.0007	9.4784e-09	0.0013	4.3011e-06
KernelRidge	0.0008	1.0921e-08	0.0014	4.4007e-06
SVR	0.0063	2.4985e-05	0.0090	8.0411e-05
MLP(layers-50-10-1, with dropout=0.2 & l2 Regularizer)	0.0048	0.0013	0.0104	0.0171
LSTM(layers-30-1, dropout)	0.0059	0.0017	0.0077	0.0046

Figure 23: Comparison of Volatility prediction results of various models when using historical Stock data

When compared the actual values with predicted results, only stock data is predicting the volatility values better than news. It infers strong correlation exists with the time series stock data compared to the information from the news data.



(a) Input - News data (b) Input - Historical stock data

Figure 24: Compare validation Volatility prediction results using historical stock data with that of news data alone.

5.3.2 Direction of the Stock market

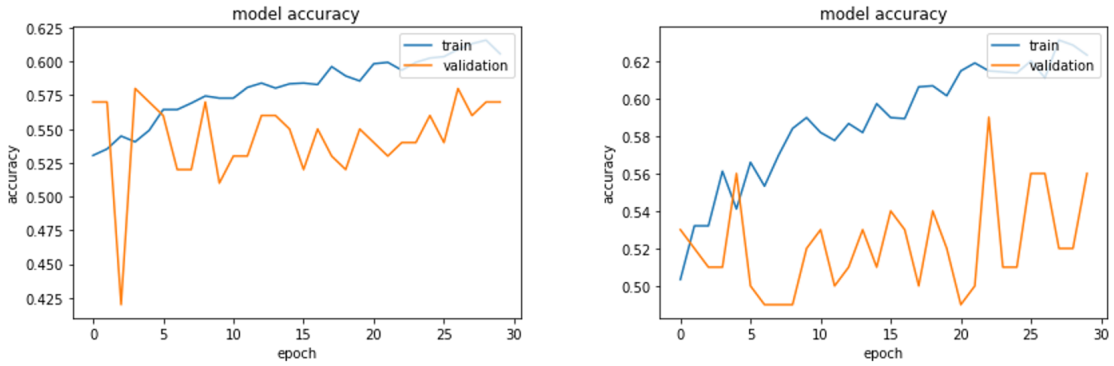
Direction of the stock market is better predicted with the news data with simple bag of words and Logistic regression, compared to the past 100 days worth of stock data, when compared the results from below figure Figure 25 with Figure 17.

ML Algorithm	Validation Accuracy	Std(+/-)
Logistic Regression	0.512	0.0323
SVC	0.513	0.0394
RandomForestClassifier	0.519	0.0476
MLP(layer-50, layer-10, with dropout=0.2, l2)	0.514	0.0226
LSTM(	0.540	0.0003

Figure 25: Comparison of the direction of the stock market prediction results of various models using historical Stock data

5.4 Combined news and Stock data

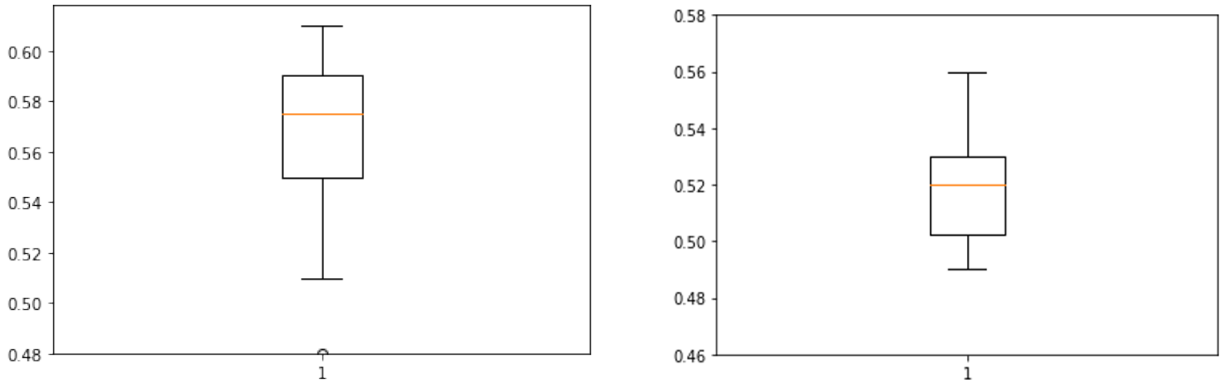
Combined the features from news and stock data using the architecture described in the ensemble Learning section 4.3.4. Prediction results for the direction of stock market and direction of Volatility are as shown in the figure 26. Merged model predicts the direction of stock market better than any single data source alone, proving the concept of ensemble Learning. And Volatility direction results also consistently better than random.



(a) Predict - Direction of Stock market (b) Predict - Direction of Volatility

Figure 26: Prediction of direction of the stock market and Volatility using news as well as historical data as an input features to the LSTM model

Respective box plots for the validation accuracy is as shown in the figure Figure 27.

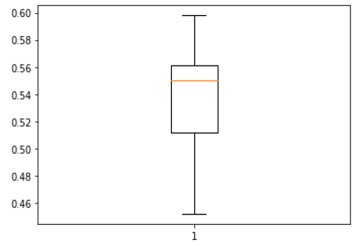


(a) Direction of Stock market (b) Direction of Volatility

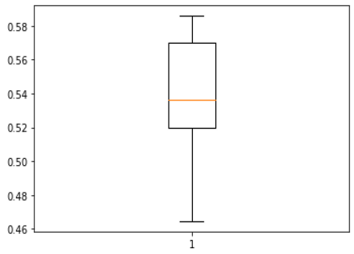
Figure 27: Validation accuracy Box plots - Combined Features

5.5 Comparison of the Models - Based on Input Features

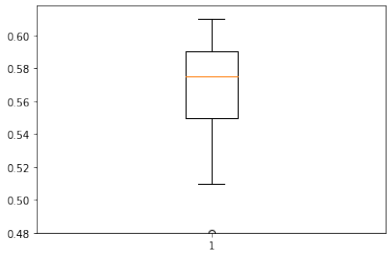
Box plots of validation accuracy of the Direction of Stock market and Volatility are as shown in the figure Figure 28, divided based on the input features to the models.



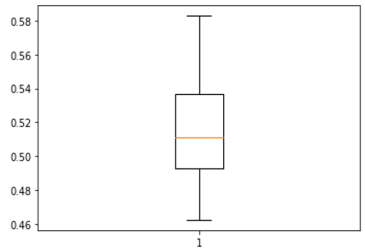
(a) TEXT - Direction of Stock market



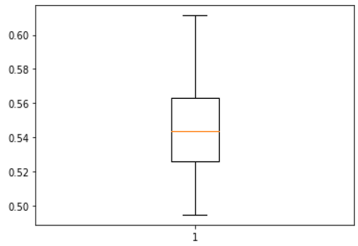
(b) STOCK - Direction of Stock market



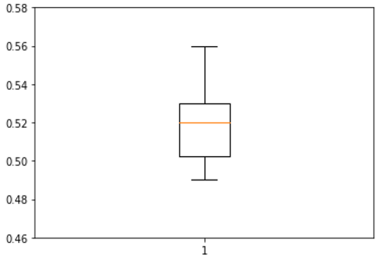
(c) TEXT and STOCK - Direction of Stock market



(a) TEXT - Direction of Stock market



(b) STOCK - Direction of Stock market



(c) TEXT and STOCK - Direction of Stock market

Figure 28: Comparison of validation accuracy of direction of the Stock market and Volatility based on the Input Features

Observations:

- Merged model performed better than the models with single data source.
- News data can predict the direction of the stock market better than stock data alone.
- Volatility is better predicted with the stock data compared to the news data. And news can predict Volatility better than random consistently.

6 Conclusion

From the results, news data alone can predict the direction of stock market better than using the historical stock data, proves that news-derived information contains a strong signal that impacts the financial markets in consistent with the previously published results. We proved with empirical results that prediction accuracy is improved when combined the features from news and stock data, accomplishing one of the objective.

We started with the research question whether Financial news data predicts stock market volatility better than close price, but the results did not reflect this intuition taken from Atkins et al.'s paper with our dataset at day level from 2008-2016. Since the results shows that the news data predicts the stock market direction better than volatility, as opposed to our objective of validating the intuition taken from Atkins et al.'s paper, we compared our dataset with Atkins et al.'s experiments, both the datasets are not at the same granularity. Due to this, both the experiments can not be compared directly as data granularity plays a major role. From this, We infer an important concept that the news data has immediate impact on the stock market, which is unable to predict well when the volatility is defined over 10 days window in our experiment, and predicted well when the Volatility is defined over 60 minutes window in Atkins et al.'s experiments.

We can conclude that Financial news at day level predicts the stock market direction better than historical stock data alone and the accuracy is further improved when both the features are combined to predict.

6.1 Future Work

After combining the news data and historical data, still the mean validation accuracy did not reach 60%, pave the way for the future work. To improve the accuracy, we may consider looking at the other parameters impacting the markets. These could be other financial parameters impacting the markets, example: Supply and demand ratio, Price to earnings ratios, inflation, Interest rates. In this work, we are predicting DJIA(Dow Jones Industrial Average) market, which consists of 30 large companies. We could experiment predicting the market for a single company/sector, to see improvement over accuracy, by including the company specific information(like company's financial performance) as an additional features. There is a technical scope to improve the accuracy, by using BERT without adding additional features. With BERT language model, the data is trained deeply to represent the contextual meaning of the words, we could use transfer learning to fine tune the pre-trained model. From the latest research, bidirectional encoders with attention mechanism is outperforming the LSTM models for the sequential data, which could be implemented to improve the prediction of the direction of the stock market. Inferences from the results, there is a scope to research to predict the duration of the impact that news could have on the stock market, which in turn could be an input feature to bring the sentiments out of the news, to predict the direction of the movement.

Acknowledgment

This paper is supported by the University of Southampton and I would like to thank Dr. Mahesan Niranjan for his guidance and patience during the entire work.

References

[1] Atkins, A., Niranjan, M., Gerding, E. (2018). Financial news predicts stock market volatility better than close price. The Journal of Finance and Data Science, 4, 120-137.

[2] Hull JC. Options, Futures, and Other Derivatives, Pearson. Upper Saddle River (N.J.): Prentice Hall; 2006.

[3] Blei DM, Ng AY, Jordan MI. Latent dirichlet allocation. J Mach Learn Res. 2003;3:993e1022.

[4] Ladokhin, S. (2009). Forecasting Volatility in the Stock Market. (BMI Paper). VU University Amsterdam, Faculty of Science.

[5] Rout, A.K., Dash, P.K., Dash, R., Bisoi, R.: Forecasting financial time series using a low complexity recurrent neural network and evolutionary learning approach. Journal of King Saud University - Computer and Information Sciences (2015)

[6] Souma, W., Vodenska, I. Aoyama, H. Enhanced news sentiment analysis using deep learning methods. J Comput Soc Sc 2, 33–46 (2019).

[7] Qiu, J., Wang, B., Zhou, C. (2020). Forecasting stock prices with long-short term memory neural network based on attention mechanism. PLoS ONE, 15.

[8] A. F. M, "Word2Vec model for sentiment analysis of product reviews in Indonesian language," International Journal of Electrical and Computer Engineering, vol. 9, (1), pp. 525-530, 2019.

[9] Gidófalvi, G. (2001). Using News Articles to Predict Stock Price Movements.

[10] Han, Jae Hyuk, "Comparing Models for Time Series Analysis" (2018). Wharton Research Scholars. 162.

[11] Sun, J. (2016, August). Daily News for Stock Market Prediction, Version 1. Retrieved [Date You Retrieved This Data] from https://www.kaggle.com/aaron7sun/stocknews.

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. In ICLR Workshop Papers.

[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In NIPS, pages 3111–3119

[14] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina (11 October 2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding".

[15] Opitz, D.; Maclin, R. (1999). "Popular ensemble methods: An empirical study". Journal of Artificial Intelligence Research.

[16] Sepp Hochreiter; Jürgen Schmidhuber (21 August 1995), Long Short Term Memory

[17] Sepp Hochreiter; Jürgen Schmidhuber (1997), "LSTM can Solve Hard Long Time Lag Problems", Advances in Neural Information Processing Systems 9

[18] Gers, F.A. (1999). "Learning to forget: Continual prediction with LSTM". 9th International Conference on Artificial Neural Networks: ICANN '99. 1999. pp. 850–855.

Appendix A Table of contents for the design archive

1. Read_me.txt
2. Labels.csv
3. Combined_News_DJIA.csv
4. News_Bag_of_Words_Volatility.ipynb
5. News_Bag_of_Words_Direction.ipynb
6. News_Word2Vec_Volatility.ipynb
7. STOCK_Volatility.ipynb
8. STOCK_Direction.ipynb
9. Merged_NEWS_and_STOCK_Classification.ipynb