Title

Predicting Apple Stock Price Using News Headlines and Other Features With Classical Time Series Models, Supervised Models, and Machine Learning Models

Permalink

https://escholarship.org/uc/item/41z973nx

Author

Jeong, Jaehui

Publication Date

2022

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Predicting Apple Stock Price Using News Headlines and Other Features
With Classical Time Series Models, Supervised Models, and Machine Learning Models

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Applied Statistics

by

Jaehui Jeong

2022

ABSTRACT OF THE THESIS

Predicting Apple Stock Price Using News Headlines and Other Features
With Classical Time Series Models, Supervised Models, and Machine Learning Models

by

Jaehui Jeong
Master of Applied Statistics
University of California, Los Angeles, 2022
Professor Yingnian Wu, Chair

This paper presents the idea of predicting Apple stock price using three different types of time series models based on many features including news headlines related to Apple. To collect data on if the news headlines have positive, neutral, or negative information, we used LSTM (Long Short Term Memory), GRU (Gated Recurrent Unit), BERT (Bidirectional Encoder Representations from Transformers) Sentimental Analysis, and BERT Fine-Tuning. The BERT Fine-Tuning model has the best result. For Apple stock price forecasting we used the classical time series models, ARIMA (Autoregressive Integrated Moving Average) and SARIMAX (Seasonal Autoregressive Integrated Moving Average with Exogenous Regressors), the supervised models, linear regression using PCA (Principal Component Analysis) and random forest, and the deep learning model, LSTM. The linear regression model using PCA performed the best. For further investigation, we could change the parameters of the LSTM model to get better results.

The thesis of Jaehui Jeong is approved.

Hongquan Xu

Nicolas Christou

Yingnian Wu, Committee Chair

University of California, Los Angeles

2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Investors always want to make more profit by buying a stock that does not already reflect new information. The prices in the stock market are determined by supply and demand. Based on the efficient-market hypothesis (EMH) [Wikc], consistently beating the market is impossible on a risk-adjusted basis since new information should be the only factor that affects market prices. The reason why consistently beating the market is impossible is that this hypothesis states that asset prices reflect all available information; however, it does not necessarily imply that stock prices are not predictable. Burton Malkiel wrote "A Random Walk Down Wall Street" [Mal73] using the random walk hypothesis [Wike]. Malkiel claimed that looking at price history could not help us accurately predict stock prices and this conclusion let him assert that hiring financial services people is actually not helpful for net portfolio return. [Wikf] For this reason, if we can provide a stock price prediction model, we will be able to help investors make more net portfolio return.

Apple Inc. is an American multinational technology company. Apple is the largest company in the world by market capitalization in 2021 in U.S. dollars. Also, since Apple is the largest company and was founded in 1976, there is a lot of accessible information. For these reasons, we decided to focus on the Apple stock price forecast. We used internal information from Apple such as earnings yield and price to book value as well as external information such as news headlines and interest rates to build predictive models.

We will cover two main parts which are text mining and forecasting. We used text mining to extract the information from news headlines as features for our predictive models. LSTM

(Long Short Term Memory), GRU (Gated Recurrent Unit), BERT (Bidirectional Encoder Representations from Transformers) Sentimental Analysis, and BERT Fine-Tuning are used for sentiment analysis. These models predict if news headlines have negative, neutral, or positive information. For forecasting, we used three different types of models; classical time series models, supervised models, and deep learning models. ARIMA (Autoregressive Integrated Moving Average) and SARIMAX (Seasonal Autoregressive Integrated Moving Average with Exogenous Regressors) are classical time series models, and linear regression using PCA(Principal Component Analysis) and random forest models are supervised models. The LSTM model is used again as a deep learning model for our forecasting.

CHAPTER 2

Methodology

2.1 Text Mining

2.1.1 RNNs

To understand what LSTM and GRU are, first we will discuss how Recurrent Neural Networks (RNNs) work. Recurrent neural networsk (RNNs) works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer. This method can be used for sequential data and Natural Language Processing (NLP). When we compare this neural networks to traditional feed-forward neural networks, RNN can take inputs in one at a time and in a sequence and generate many outputs. GRU and LSTM are also a type of RNN.

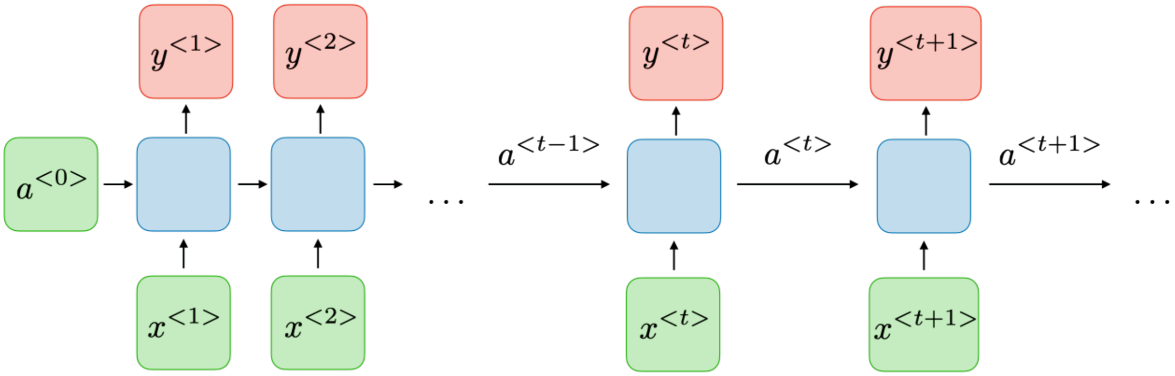


Figure 2.1: Architecture of a Traditional RNN [AA]

RNN takes one input at a time and in a sequence. The RNN produces an output, known as a hidden state, using the previous hidden state and a new input. This process will run until there is no more input or the model is programmed to finish.

Variables and functions

$$
\begin{aligned}
t &: \text{timestep} \\
a_t &: \text{activation(hidden layer vector)} \\
y_t &: \text{output vector} \\
g_1 \text{ and } g_2 &: \text{activation functions} \\
\text{W, U, and b} &: \text{parameter matrices and vector}
\end{aligned}
$$

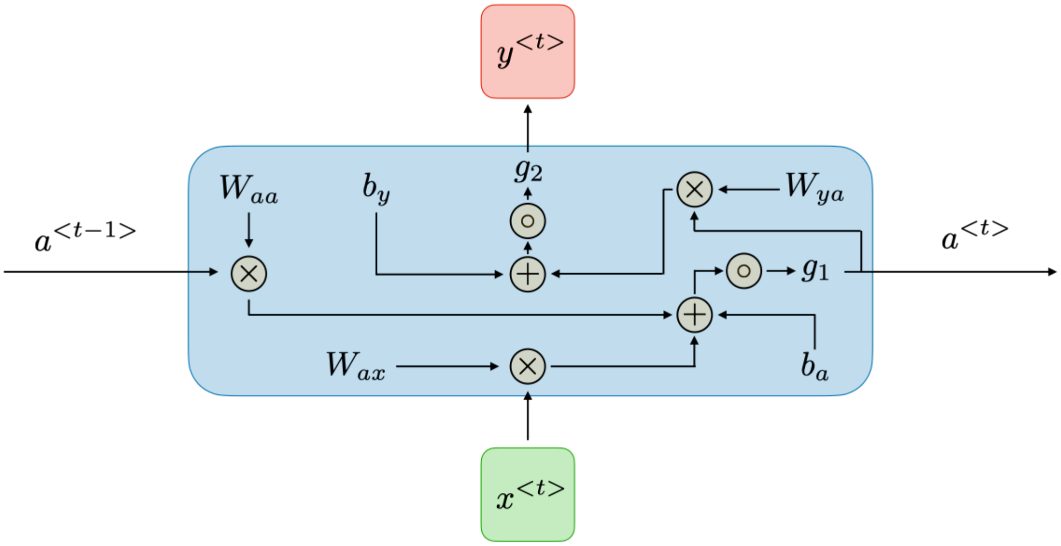


Figure 2.2: The Framework of RNN [AA]

Formulas

$$
\begin{aligned}
a_t &= g_1(W_{aa}a_{t-1} + W_{ax}x_t + b_a) \\
y_t &= g_2(W_{ya}a_t + b_y)
\end{aligned}
$$

Loss function

$$
L(\hat{y}, y) = \sum_{t=1}^{T_y} L(\hat{y}_t, y_t)
$$

Backpropagation through time

$$\frac{\partial L_t}{\partial W} = \sum_{t=1}^{T_y} \frac{\partial L_t}{\partial W}\Big|_t$$

Activation functions

$$\text{Sigmoid: } g(z) = \frac{1}{1+e^{-z}}$$

$$\text{Tanh: } g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

$$\text{RELU: } g(z) = max(0, z)$$

One of the biggest problems of RNNs is the vanishing and exploding gradient phenomena. It happens because capturing long-term dependencies is difficult due to multiplicative gradients. To deal with it, specific gates are used in LSTM and GRU. Gate is noted as Γ. Update gate Γ_u decides the weight of the previous value that will affect the current value. Relevance gate Γ_r decides if we need to drop previous information or not. These two are used both in GRU and LSTM. Forget gate Γ_f decides if we need to erase a cell or not and output gate Γ_o decides how much to reveal of a cell. These two gates are only used in LSTM.

2.1.2 LSTM

There are three gates, input gate (Γ_r and Γ_u), output gate (Γ_o), and forget gate (Γ_f) and four weights. The details to calculate all the values are provided in the equation section. Therefore, we will check the structure of LSTM here. To get a new cell state, we need to get the two main partitions; forget gate and input gate. Each part is independent to each other, which means adding the amount of new memory using input gate is totally independent of the information retained using forget gate. We will use output gate and the previous cell state to calculate a new hidden state. Figure 2.3 shows the framework of LSTM.

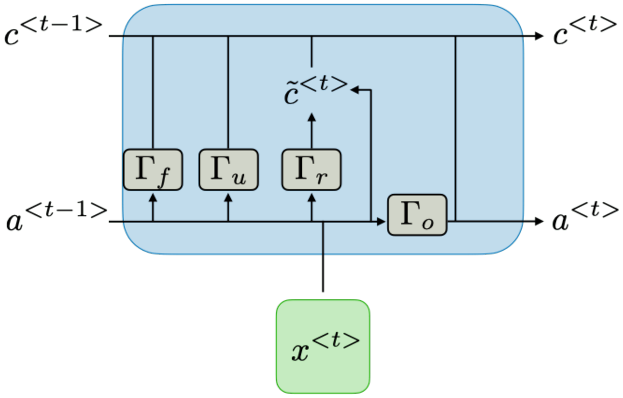


Figure 2.3: The Framework of LSTM [AA]

Formulas

$$\tilde{c}_t = \tanh(W_c[\Gamma_r \star a_{t-1}, x_t] + b_c)$$

$$c_t = \Gamma_u \star \tilde{c}_t + \Gamma_f \star c_{t-1}$$

$$a_t = \Gamma_0 \star c_t$$

$\star$: The element-wise multiplication between two vectors

2.1.3 GRU

The main difference between LSTM and GRU is that the addition of new information and the retention of previous memory are dependent. Another key difference is that GRU stores longer-term dependencies and short-term memory in a single hidden state, not like LSTM. There are two gates, update gate (Γ_u) and reset gate (Γ_r), and three weights. Figure 2.4 shows the framework of GRU.

Formulas

$$\tilde{c}_t = \tanh(W_c[\Gamma_r \star a_{t-1}, x_t] + b_c)$$

$$c_t = \Gamma_u \star \tilde{c}_t + (1 - \Gamma_u) \star c_{t-1}$$

$$a_t = c_t$$

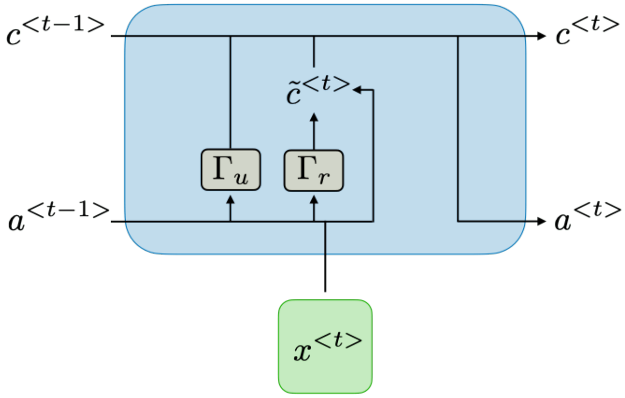


Figure 2.4: The Framework of GRU [AA]

2.1.4 BERT

BERT (Bidirectional Encoder Representations from Transformers) makes use of Transformer. Transformer includes two separate mechanisms, an encoder and a decoder. BERT is the encoder part of Transformer for natural language processing (NLP.) The main strategies are the Masked Language Model (MLM) and Next Sentence Prediction (NSP.) The transitional transformer was looking at a text sequence either from left to right to combined left-to-right and right-to-left training. BERT is applying the bidirectional training of Transformer. BERT can be explained in three parts embedding, transformers, and output.

1) Input embedding

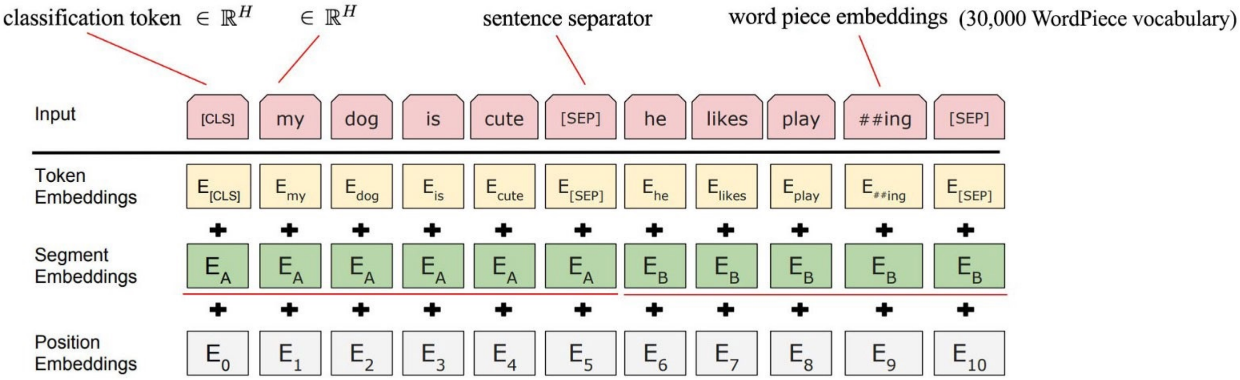

classification token $\in \mathbb{R}^H$
$\in \mathbb{R}^H$
sentence separator
word piece embeddings (30,000 WordPiece vocabulary)
Input
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2.5: BERT Input Embedding [DCL]

In BERT, the input embedding composes of word piece embedding, segment embeddings, and position embedding of the same dimension. We add them all together to get the final input embedding.

2) Transformers: Self-attention

Self-attention is the method Transformer uses to bake the "understanding" of other relevant words into the one we're currently processing. In BERT, multi-headed attention is used instead of self-attention.

The self-attention matrix for input matrices (Q, K, V) is calculated as [Ash17]:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}}) \cdot V$$

Mathematically multi-head attention can be represented by [Ash17]:

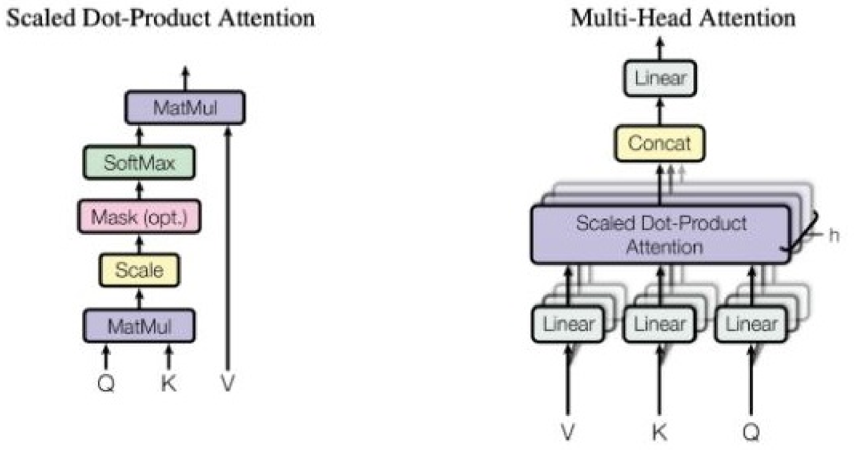


Figure 2.6: BERT Transformers: Self-attention [Ash17]

$$MultiHead(Q, K, V) = concat(head1, head2, \cdots, head_n) \cdot W_o$$

$$where, head_i = Attention(QW_i^Q, KW_i^k, VW_i^V)$$

2) Transformers: Feed-Forward Networks

We we check the dimension below. (The input is the 13 512-D word embedding vectors in this example.)

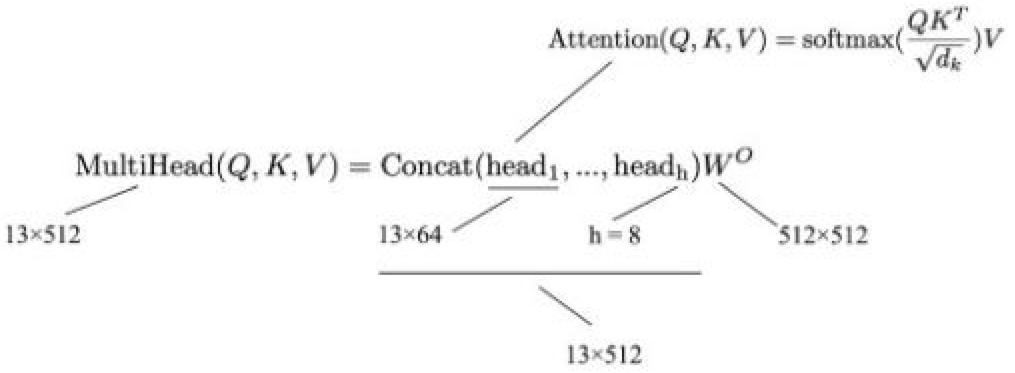

Attention(Q, K, V) = softmax($\frac{QK^T}{\sqrt{d_k}}$)V
MultiHead(Q, K, V) = Concat(head$_1$, ..., head$_h$)W^O
13×512
13×64
h = 8
512×512
13×512

Figure 2.7: BERT Transformers: Feed-Forward Networks1 [HUI]

Now we can apply a linear transformation to the concatenated result with W to get the linear layer. After we get the linear layer, we will apply the GELU (Gaussian Error Linear Unit) function and normalization. The GELU activation function is $x\Phi(x)$, where $\Phi(x)$ the standard Gaussian cumulative distribtuion fuction. This operation will be applied to each position separately and identically sharing the same weights. This operation is a position-wise feed-forward because the ith output depends on the ith attention of the attention layer only.

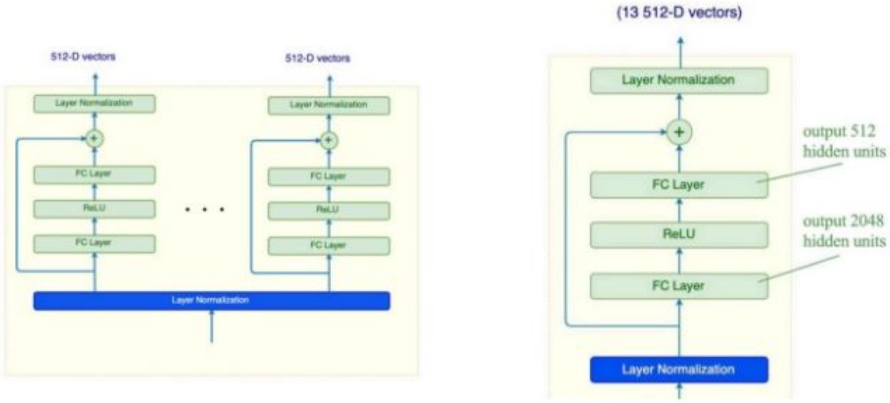


Figure 2.8: BERT Transformers: Feed-Forward Networks2 [HUI]

3) Output

We need to apply embedding to vocab (Embedding Inverse), and softmax to get the output, words.

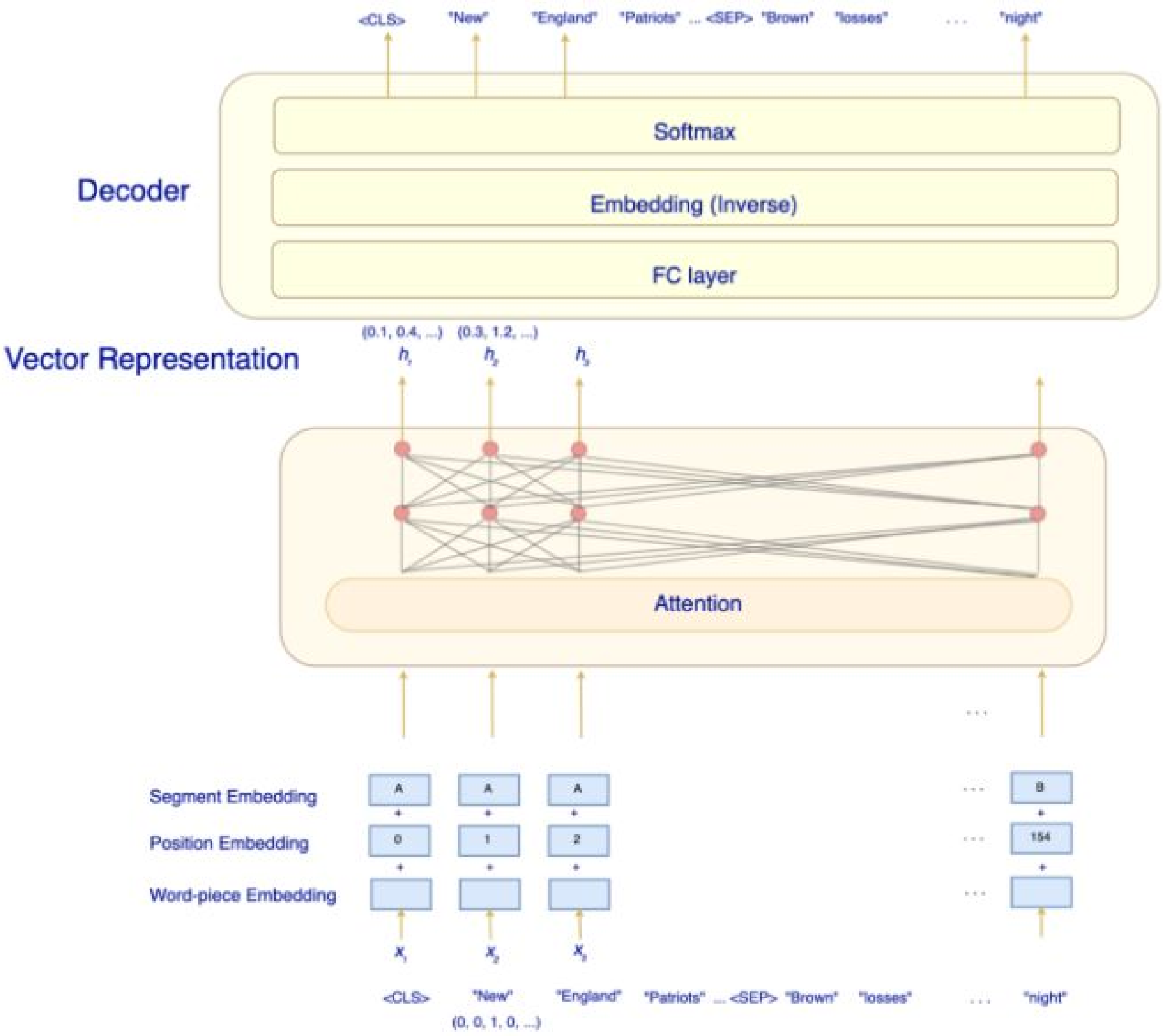


Figure 2.9: BERT Transformers: Feed-Forward Networks3 [HUI]

To train the model, we need to think about Masked LM(Masked Language Model), and NSP(Next Sentence Prediction). LM and NSP are the 2 NLP(Natural Language Processing) tasks to train BERT. We will look at how these two tasks can be done on the following pages.

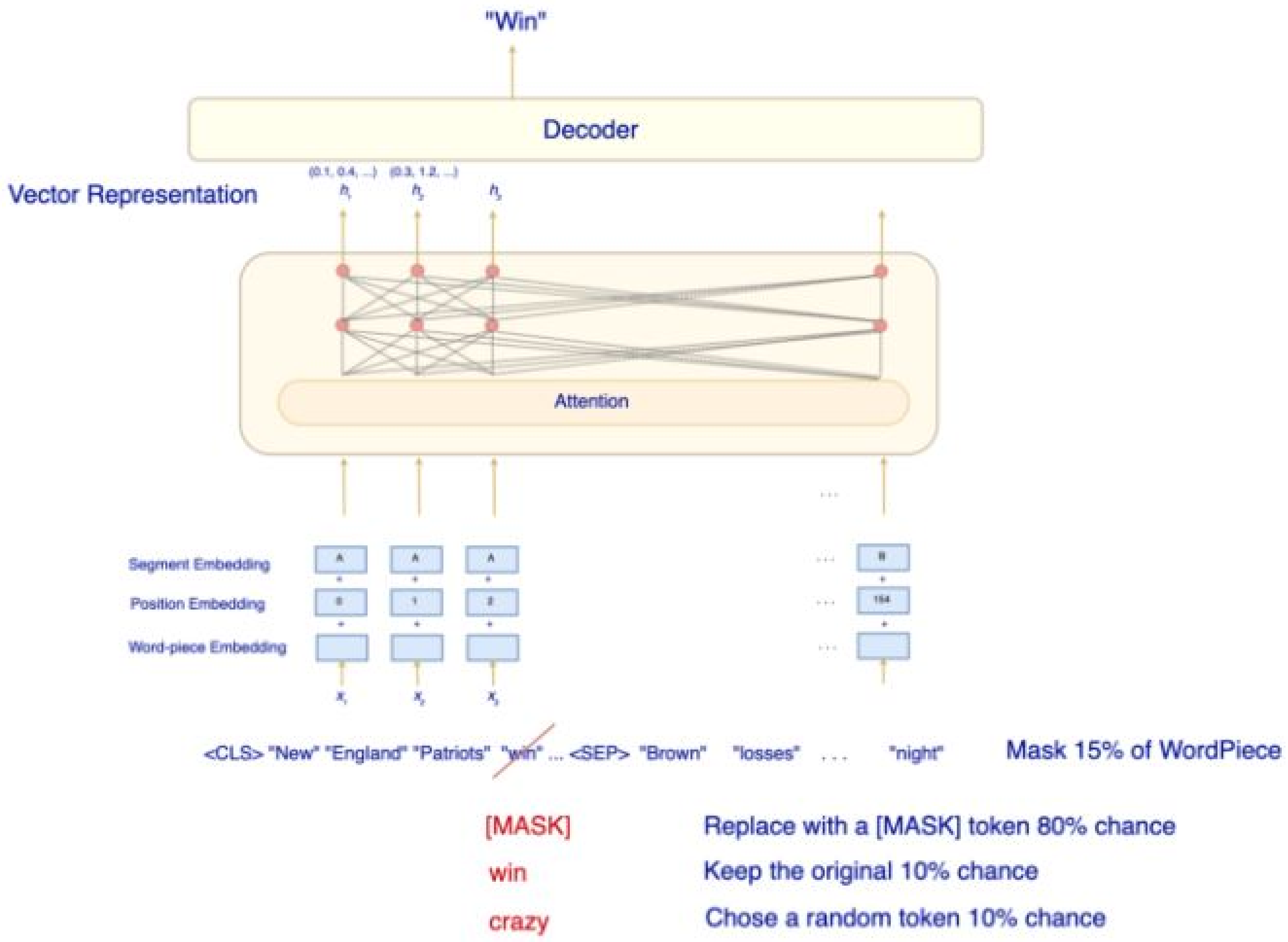


Figure 2.10: BERT Transformers: Feed-Forward Networks4 [HUI]

In the Masked LM, BERT masks out 15% of the WordPiece. 80% of the masked WordPiece will be replaced with a [MASK] token, 10% with a random token and 10% will keep the original word. The loss is defined as how well BERT predicts the missing word, [MASK], not the reconstruction error of the whole sequence.

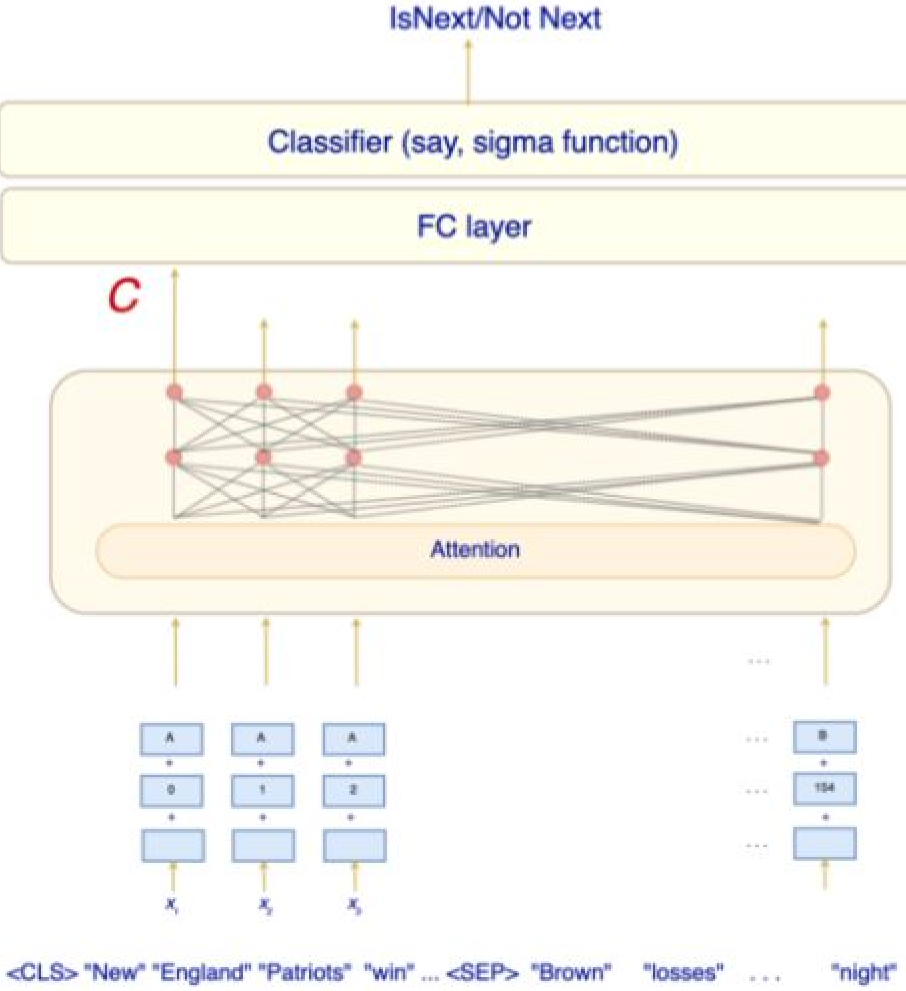

IsNext/Not Next
Classifier (say, sigma function)
FC layer
C
Attention
A
A
A
B
0
1
2
154
x_1
x_2
x_3
<CLS> "New" "England" "Patriots" "win" ... <SEP> "Brown" "losses" . . . "night"

Figure 2.11: BERT Transformers: Feed-Forward Networks5 [HUI]

The main purpose of NSP is to check the sequence between A and B. Please check the example above.

2.2 Forecasting

The stock price is a time series data. A time series is a series of data points indexed in time order. Forecasting a time series can be broadly determined by two types, univariate time series models and multivariate time series models. Univariate time series models use only one variable which is the target time series data. In this case, the stock price will be the only variable. One of the most popular univariate time series models is the ARIMA model. For multivariate time series models, it uses multiple variables including external or exogenous variables. All the models that we used except the ARIMA model fall into multivariate time series models. The models that we used can be divided into three different types; time series models, classical time series models, supervised models, and deep learning-based models. [Kor]

2.2.1 Classical Time Series

Time series data can be decomposed into seasonality, trend, and error. Seasonality refers to predictable changes that occur over a one-year period based on the seasons. Trend is a general tendency to move up or down, so trend will help us understand if the stock price has been increased or decreased. Error is the part of the variability in a time series that cannot be explained by seasonality and a trend.

Autocorrelation, sometimes known as serial correlation in the discrete-time case, is the correlation of a signal with a delayed copy of itself as a function of delay. [Wika] Autocorrelation can be positive or negative. Positive autocorrelation indicates that there is a higher chance to yield a high value when the present value is high. Negative autocorrelation is the opposite. If a value is high today, it will have a low-value tomorrow and vice versa.

2.2.1.1 ARIMA

ARIMA stands for Autoregressive Integrated Moving Average. The ARIMA model is a combination of three smaller models, Autoregression (AR), Moving Average(MA), and Integrated (I). The AR explains a variable's future value using its own lagged values. The order number of time lags of AR is p. The MA indicates that the regression error is a linear combination of error terms in previous time steps to predict the future. The order parameter denoted q for MA. The I is used to eliminate non-stationarity by replacing it with the difference between their values and the previous values. This part is also known as differencing process. The degree of differencing denoted d. For example, if we have an ARIMA(1, 1, 1) model, the first 1 is for the AR order, the second one is for the differencing, and the last 1 is for the MA order.

$ARMA(p', q)$ [Wikb]

$$\left(1 - \sum_{i=1}^{p'} \alpha_i L^i\right) X_t = \left(1 + \sum_{i=1}^{q} \theta_i L^i\right) \epsilon_t$$

Variables

X_t : time series data where t is an integer index (X_t are real numbers)

L : the lag operator

α_i : the parameters of the autoregressive part of the model

θ_i : the parameters of the moving average part

ϵ_t : $errorterms$

Assume that the polynomial term $\left(1 - \sum_{i=1}^{p'} \alpha_i L^i\right)$ has a unit root (a factor $(1 - L)$) of multiplicity d.

$$\left(1 - \sum_{i=1}^{p'} \alpha_i L^i\right) = \left(1 - \sum_{i=1}^{q'-d} \varphi_i L^i\right) (1 - L)^d$$

An $ARIMA(p, d, q)$ process expresses this polynomial factorisation property with $p = p'd$

$$\left(1 - \sum_{i=1}^{p} \varphi_i L^i\right) (1 - L)^d X_t = \left(1 + \sum_{i=1}^{q} \theta_i L^i\right) \epsilon_t$$

The above can be generalized as follows:

$$\left(1-\sum_{i=1}^{p}\varphi_i L^i\right)(1-L)^d X_t = \delta + \left(1+\sum_{i=1}^{q}\theta_i L^i\right)\epsilon_t$$

This defines an $ARIMA(p,d,q)$ process with drift$\frac{\delta}{1-\sum\varphi_i}$

Differencing

$$y_t' = y_t - y_{t-1}$$

2.2.1.2 SARIMAX

Seasonal autoregressive integrated moving-average with exogenous regressors (SARIMAX) is the most complex model in the ARIMA family of models. SARIMAX has AR, MA, differencing, seasonal effects, and external variables.

The SARIMA model is specified (p,d,q) X $(P,D,Q)s$ [sta].

$$\phi_p(L)\hat{\phi}_p(L^s)\triangle^d\triangle_s^D y_t = A(t) + \theta_q(L)\hat{\theta}_Q(L^s)\epsilon_t$$

Variables

$\phi_p(L)$: the non-seasonal autoregressive lag polynomial

$\hat{\phi}_p(L^s)$: the seasonal autoregressive lag polynomial

$\triangle^d\triangle_s^D y_t$: the time series, differenced d times, and seasonally differenced D times

$A(t)$: the trend polynomial (including the intercept)

$\theta_q(L)$: the non-seasonal moving average lag polynomial

$\hat{\theta}_Q(L^s)$: the seasonal moving average lag polynomial

2.2.2 Supervised Model

Supervised machine learning models learn how to map an input to an output based on example input-output pairs. The main difference between the classical machine learning models and supervised machine learning models is that variables are considered either dependent or independent variables which is not true for time series data. However, it is not difficult to convert the seasonality into independent variables. We can simply create year, month, week, and day columns using the timestamp.

2.2.2.1 Linear Regression Using PCA

Linear regression is the simplest supervised model. Simple linear regression uses only one independent variable to predict a dependent variable. In multiple linear regression, it uses multiple independent variables rather than using only one independent variable

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \epsilon$$

Variables

$$\begin{aligned} y &: \text{the dependent variable} \\ x_i &: \text{ith independent variables} \\ \beta_i &: \text{ith coefficients} \\ \epsilon &: \text{error} \end{aligned}$$

The predictor variables that we will use to train our models have a multicollinearity issue. To deal with this, we used PCA. The principal components of a collection of points in a real coordinate space are a sequence of p unit vectors, where the i-th vector is the direction of a line that best fits the data while being orthogonal to the first $i-1$ vectors. Here, a best-fitting line is defined as one that minimizes the average squared distance from the points to the line. [Wikd]

2.2.2.2 Random Forest

The linear model could be good but it cannot capture non-linear relationships. Since the linear model has limitations, we can use the Random Forest model which allows fitting nonlinear relationships. The Random Forest model is an ensemble learning method that is trained by constructing a multitude of decision trees. This model can be used for classification, regression, and other tasks as well. In this case, we will look at the Random Forest Regression model.

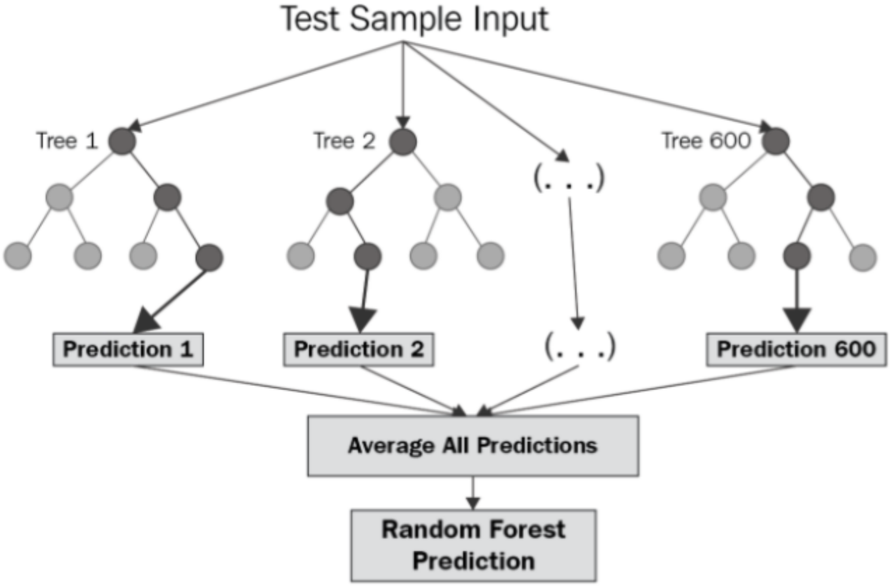


Figure 2.12: The structure of a Random Forest [Bak]

When we look at the diagram above, there is no interaction among trees. A Random Forest operates by constructing many decision trees at training time and getting the mean of the classes as the prediction of all the trees. To better understand how it works, we will check these steps [Bak]:

1. Picking k data points at random from the training set
2. Building a decision tree associated to these k data points
3. Choosing the number N of trees you want to build and repeat steps 1 and 2
4. For a new data point, making each one of your N-tree trees predict the value of y for the data point in question and assign the new data point to the average across all of the predicted y values.

2.2.3 Deep learning Model

Deep learning models are more complex than the other types of models. There are various deep learning models such as Prophet and DeepAR. However, we will use LSTM which we already mentioned earlier.

2.2.4 CAPM (Capital Asset Pricing Model)

In our thesis, we will not provide a trained CAPM (Capital Asset Pricing Model); however, since this model is one of the most popular models in finance, we will cover the concept of this model. The CAPM [FF04] is a model used to find an appropriate required rate of return of an asset. In other words, this model can be used for pricing an individual security or portfolio. [Wikd]

$$\frac{E(R_i) - R_f}{\beta_i} = E(R_m) - R_f$$
$$E(R_i) = R_f + \beta_i(E(R_m) - R_f)$$

where:

$E(R_i)$: the expected return on the capital asset

R_f : the risk-free rate of interest such as interest arising from government bonds

β_i : the sensitivity of the expected excess asset returns to the expected excess market returns

$E(R_m)$: the expected return of the market

$E(R_m) - R_f$: the market premium

$E(R_i) - R_f$: the risk premium

The formula to get the expected return is provided above. We will provide the β of Apple stock under the Exploratory Data Analysis section.

CHAPTER 3

Data

3.1 Data Source

Three methods are used to collect data; web scraping to collect Apple internal information such as net income and revenue from YCharts, using open source to get news headlines related to Apple from Wall Street Journal News and USA Today, and downloading some external information such as gold prices and inflation rates from various sources.

1) Web Scraping

Selenium package is used in python to perform web scraping from YCharts to collect Apple internal information from April 01, 2014 to December 31, 2021.

Table 3.1: Variables from YCharts

Variables	**Meaning**
Adjusted Closing Price	The closing price after adjustments for all applicable splits and dividend distributions.
Volumn	The number of shares traded in a particular stock, index, or other investment over a specific period of time
RS ratio	Relative strength is a ratio of a stock price performance to a market average (index) performance.

Table 3.2: Variables from YCharts

Variables	Meaning
Price to book	Book value is equal to the cost of carrying an asset on a company's balance sheet. It's calculated by dividing the company's stock price per share by its book value per share (BVPS).
Earnings yield	The earnings yield refers to the earnings per share for the most recent 12-month period divided by the current market price per share. The earnings yield (the inverse of the P/E ratio) shows the percentage of a company's earnings per share.
Dividend yield	The dividend yield, expressed as a percentage, is a financial ratio (dividend/price) that shows how much a company pays out in dividends each year relative to its stock price.
Net income	Net income indicates a company's profit after all of its expenses have been deducted from revenues.
Revenue	Revenue is how much money a business brings in by selling its goods or services at a certain price.
EPS diluted	Diluted EPS is a calculation used to gauge the quality of a company's earnings per share (EPS).
Return on assets	The term return on assets (ROA) refers to a financial ratio that indicates how profitable a company is in relation to its total assets.
Profit margin	Profit margin gauges the degree to which a company or a business activity makes money, essentially by dividing income by revenues.

2) Using open source

OpenBlender package is used in python to collect news headlines from Wall Street Journal News and USA Today that include 'apple', 'iphone', 'ipad', 'time cook', and 'mac store' from

April 01, 2014 to December 31, 2021. One day can have multiple news headlines and some days do not have any news headlines. This data is used to conduct sentiment analysis.

3) Downloading data

Financial News Headline dataset with a sentiment label is from Kaggle. [Bir] This dataset is used to train text mining models.

The variables that we downloaded are the factors that we expect may affect the stock price. Global gold price is from Kaggle. [Jis] US dollar exchange rates from 80 countries from BIS (Bank for International Settlements). [BIS] We will use the US dollar exchange rates from G20 countries only. Since Germany, Italy, and France use Euro, we will keep the US dollar exchange rates from France only. Gross Domestic Product(GDP) is from Fred (Federal Reserve Bank of St.Louis). [FREb] Inflation rates is from US Inflation Calculator. [USI] Unemployment rates is from data.bls.gov. [USB] Federal Funds Effective Rate is from Fred. [FREa]

3.2 Data Cleaning

3.2.1 Missing Values

Some predictor variables are available quarterly, but we want to predict daily adjusted close prices. For this reason, we had to deal with the missing values. Net come, revenue, EPS diluted, return on assets, and profit margin are the predictor variables that were provided quarterly only. Also, we had to deal with holidays because the stock price is not available on holidays. We filled the missing values with the previous values. For instance, if we get a value on March 31, 2018 and June 30, 2018, we will use the value on March 31, 2018 for the period from April 01, 2018 to June 29, 2018.

We can get a negative, neutral, and positive column using our text mining model; some days do not have any news headlines. For those days, we insert 1 into the neutral column, and 0 into the negative and positive columns because if there is no news at all, it indicates that there were no big events that may affect the stock price.

3.2.1.1 Text Cleaning

The Financial News Headline and Apple News Headline datasets needed to be cleaned. We kept the news headlines that were written in English, and cleaned punctuation, special characters, and stemming words. All the uppercase letters were converted to lowercase letters. Also, we dropped duplicated news headlines from the same date.

CHAPTER 4

Models

4.1 Text Mining Models

4.1.1 Exploratory Data Analysis

Before the modeling, we explored data using R as it is a perfect tool to tokenize and tidy the text. (This is the only part where we used R instead of Python.) We could explore the data at a glance by plotting word clouds for each of the three categories.

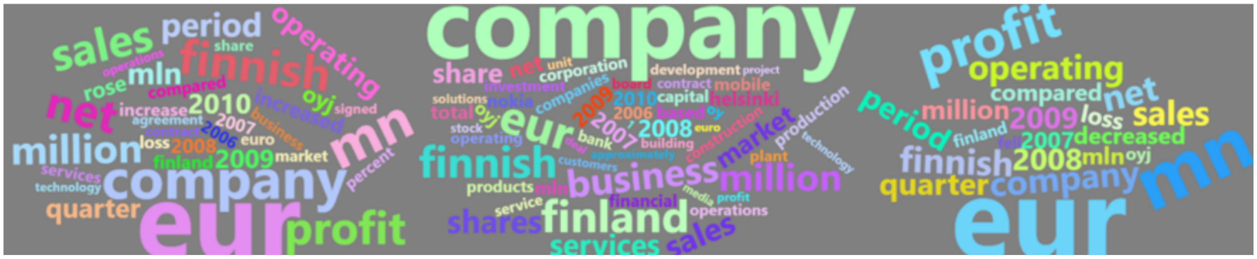


Figure 4.1: Word cloud for positive (left), neutral (middle), and negative (right)

Figure 4.1 shows three word clouds for positive (left), neutral (middle), and negative (right). Based on Figure 4.1, as the second most important currency in the world, "EUR" was the most prominent word over the three categories and the word "profit" appeared a lot in negative headlines. After we make tibble in R to tokenize each headline using the tidytext package, we were prepared to conduct sentiment analysis.

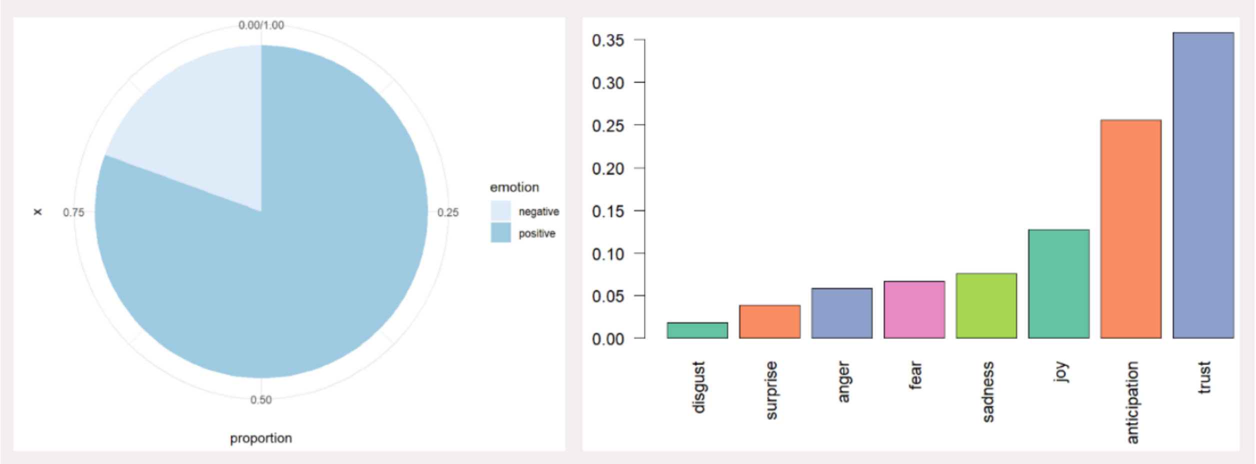


Figure 4.2: NRC sentiments for the whole headline data

We examined how ”NRC emotions” fall into each category using the Syuzhet package. Figure 4.2 shows NRC sentiments for the whole headline data. As can be seen in Figure 4.2, the whole tokens for headlines are composed of about 80 percent of positive words and 20 percent of negative words while trust and anticipation words were taking most part of NRC sentiments.

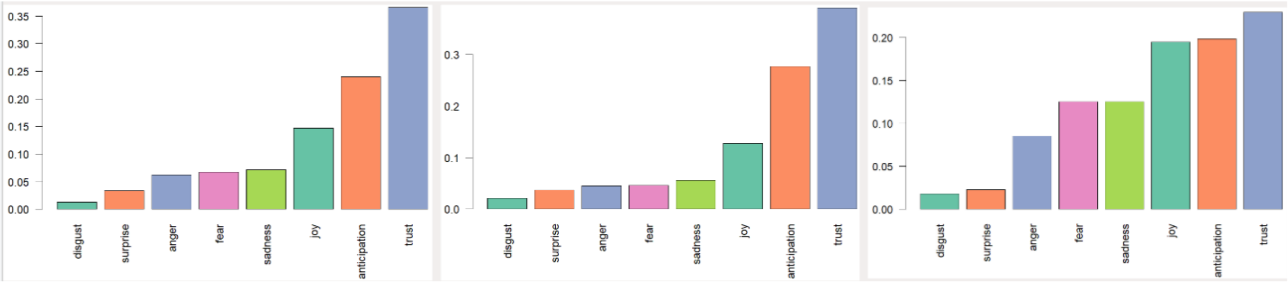


Figure 4.3: NRC sentiments for positive (left), neutral (middle), and negative (right)

We also conducted sentiment analysis for each category (positive, negative, and neutral). It was found that the composition of emotions was quite similar within positive and neutral categories while the tokenized headlines in a negative category consisted of more anger, fear, and sadness. It was interesting to find out that the positive sentiments such as joy and anticipation were taking more parts than those in positive or neutral, and we attribute this to negations.

4.1.2 Modeling

We decided to use four models: LSTM (Long Short Term Memory), GRU (Gated Recurrent Unit), BERT(Bidirectional Encoder Representations from Transformers) Sentimental Analysis, and BERT Fine-Tuning. We then tokenized and padded the texts, encoded the classification results, and split the data as training and testing datasets with a proportion of 8 to 2. We used Adam as the optimizer. For LSTM and GRU, we set the learning rate as 1e-4 and that of BERT Fine-Tuning was set at 1e-5. We put all the models' training processes with the EarlyStopping feature with three epochs for patience to save time. Table 5.1 shows results of text mining models and Figure 4.8 shows the architecture of BERT Fine-Tuning model which has the best result. Thus, we used the BERT Fine-Tuning model to get negative, neutral, and positive columns from Apple news headlines. We combined negative, neutral, and positive columns that we got using the BERT Fine-Tuning Model to the cleaned dataset to build predictive models.

4.1.2.1 LSTM

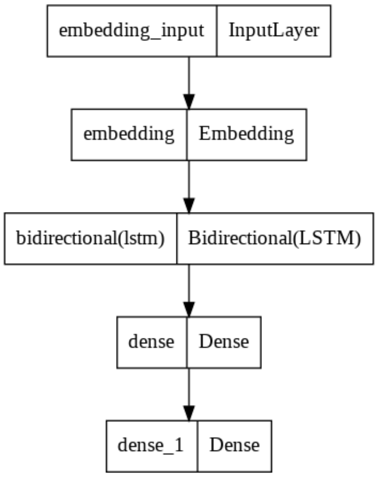
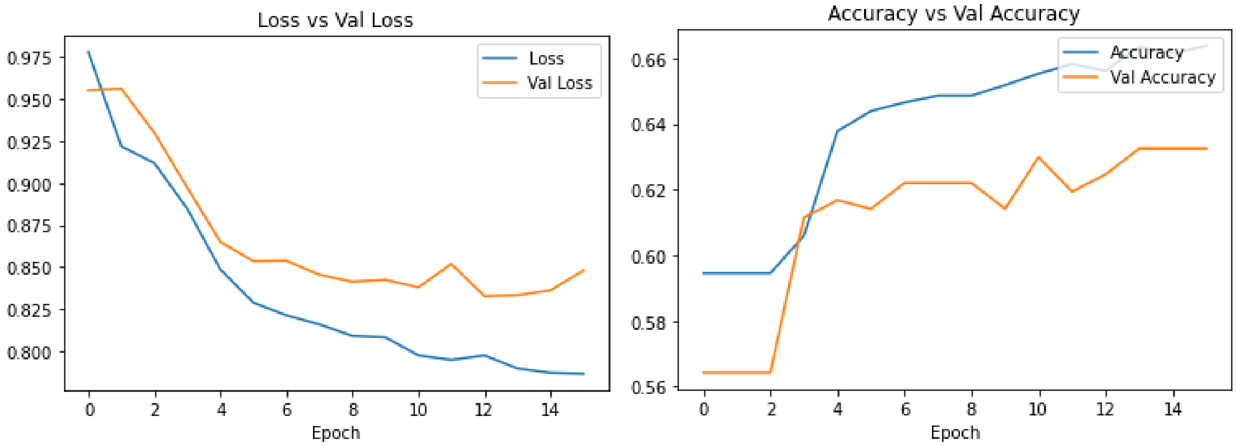

embedding_input | InputLayer
embedding | Embedding
bidirectional(lstm) | Bidirectional(LSTM)
dense | Dense
dense_1 | Dense

Figure 4.4: Architecture of LSTM Model



(a) Training and Validation Loss

(b) Training and Validation Accuracy

Figure 4.5: LSTM Model Performance Plots

The training and validation loss plot is one of the popular learning curves. A learning curve is a plot showing the progress over the experience of a specific metric related to learning during the training of a machine learning model. They are a mathematical representation of the learning process. [Bae] The training loss shows how well the trained model performs using the training data while the validation loss indicates how well the trained model fits new data. Figure 4.5-(a) shows the training loss and validation loss. Based on Figure 4.5-(a), we can find that the train and validation curves are improving; however, we can also find a big gap between the train and validation curves which means the train and validation set may have different distributions or the model is overfitted. The accuracy curves are also popular learning curves. We want to have a high value of accuracy. Figure 4.5-(b) shows the training accuracy and validation accuracy. In Figure 4.5-(b), the model performance is growing over time which indicates our model is learning per epoch. However, even though in the beginning, our model performance grows, it reaches a plateau over time. This indicates our model improves very slowly or may not learn anymore. We stopped training the LSTM model at epoch 16 and the value of loss for training set was 0.7867 and the value of accuracy was 0.6638. For the validation set, the value of loss was 0.8480 and the value of accuracy was 0.6325. The value of loss and accuracy for the test set will be provided under conclusion.

4.1.2.2 GRU

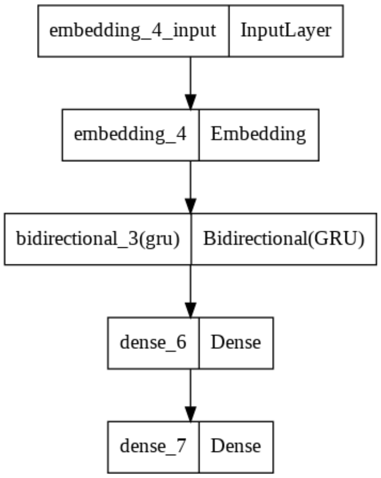
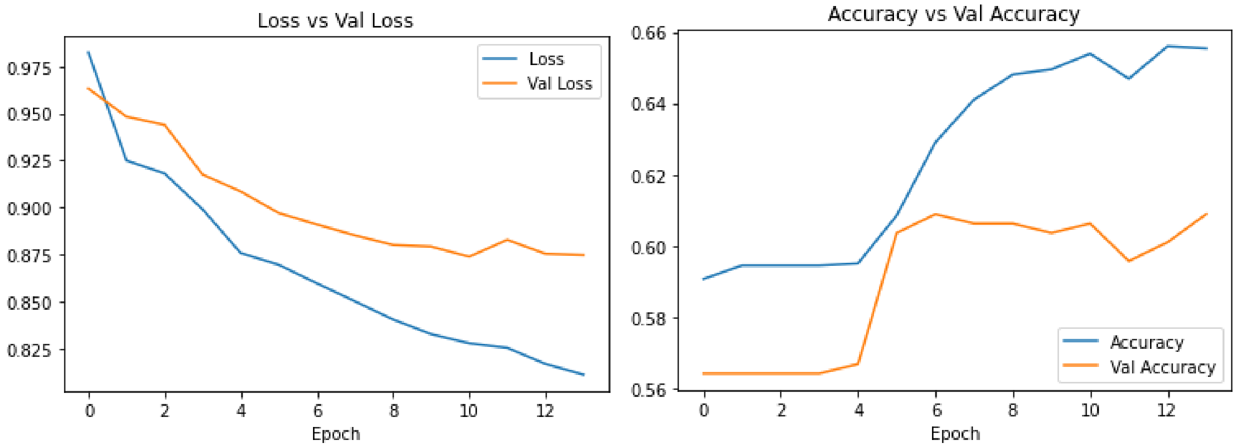


Figure 4.6: Architecture of GRU Model



(a) Training and Validation Loss

(b) Training and Validation Accuracy

Figure 4.7: GRU Model Performance Plots

Figure 4.7-(a) shows the training and validation loss. In Figure 4.7-(a), even though the training loss keeps decreasing, the validation loss does not decrease as much as the training loss. After epoch 11, we can see that there is a plateau which may indicate our GRU model is overfitted. Figure 4.7-(b) shows the training and validation accuracy. In Figure 4.7-(b), the traning accuracy and validation accuracy are very similar to the LSTM accuracy plot. The model performace improves over time in the beginning, but it reaches a plateau. We stopped training the GRU model at epoch 14 and the value of loss for training set was 0.8110 and the value of accuracy was 0.6554. For the validation set, the value of loss was 0.8746 and the value of accuracy was 0.6089. The value of loss and accuracy for the test set will be provided under conclusion.

4.1.2.3 BERT Fine Tuning

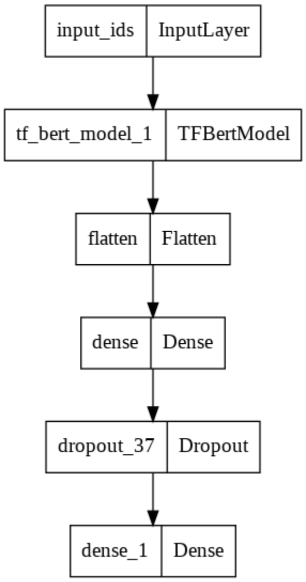
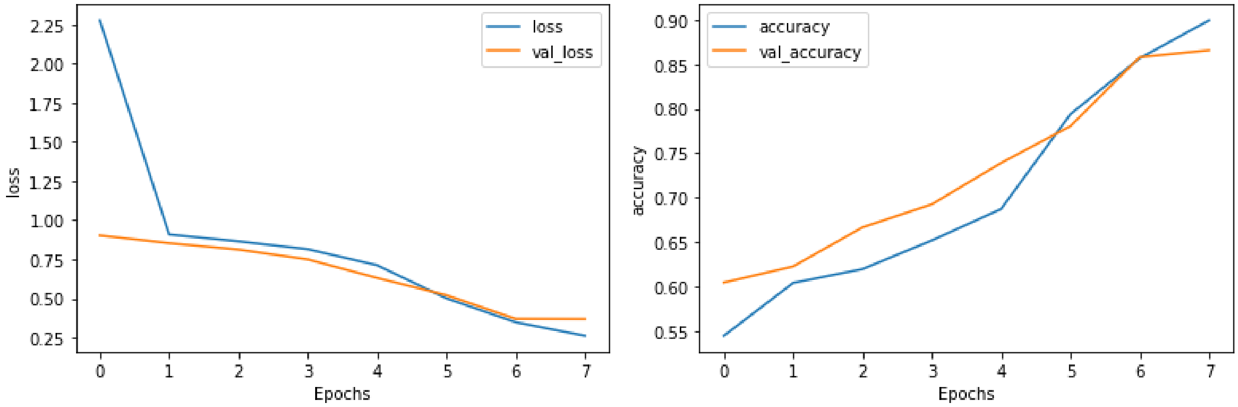


Figure 4.8: Architecture of BERT Fine-Tuning Model



(a) Training and Validation Loss

(b) Training and Validation Accuracy

Figure 4.9: BERT Fine-Tuning Model Performance Plots

The accuracy of the BERT sentimental analysis was only 0.263 and since we did not need to train this model at all, we will not cover the details about the BERT sentimental analysis model here. We will look at the BERT Fine-Tuning model. Figure 4.9-(a) shows the training and validation loss. Based on Figure 4.9-(a), the training loss and the validation loss have very similar values. Based on the training and validation loss plot, we can consider that this BERT Fine-Tuning model is a good fit model. Figure 4.9-(b) shows the training and validation accuracy. Even with Figure 4.9-(b), we can see that the accuracy for the training and validation set are very high and very similar. We stopped training the BERT Fine-Tuning model at epoch 8 and the value of loss for training set was 0.2613 and the value of accuracy was 0.8992. For the validation set, the value of loss was 0.3687 and the value of accuracy was 0.8656. The value of loss and accuracy for the test set will be provided under conclusion. However, even before we check the accuracy of the test set, we can already tell that this BERT Fine-Tuning model will work the best based on the loss and accuracy.

4.2 Predictive Models

4.2.1 Exploratory Data Analysis

In our final dataset, there are 35 variables including negative, neutral, and positive columns. As we mentioned, stock price data is time-series data that can be decomposed into trend, seasonality, and error.

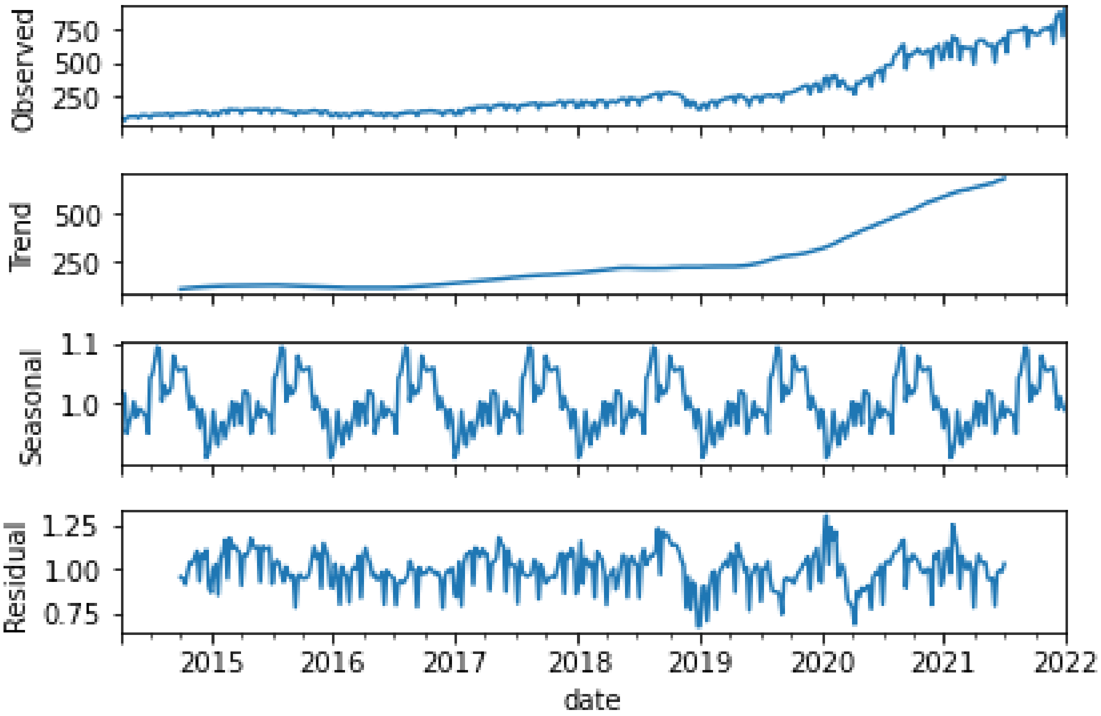


Figure 4.10: Adjusted Stock Price - Trend, Seasonality, and Error

To decompose the adjusted stock prices, we used a multiplicative model. Figure 4.10 shows the trend, seasonality, and error of the adjusted stock price. Based on Figure 4.10, we can find our target variable has trend and seasonality as well. The Apple stock price has increased over time and the price is more likely to increase in June and July and decrease in November and December. We also conducted the Dickey-Fuller test [DF79] to check if the data is non-stationarity or not. The Dicky-Fuller test is a statistical hypothesis test for it.

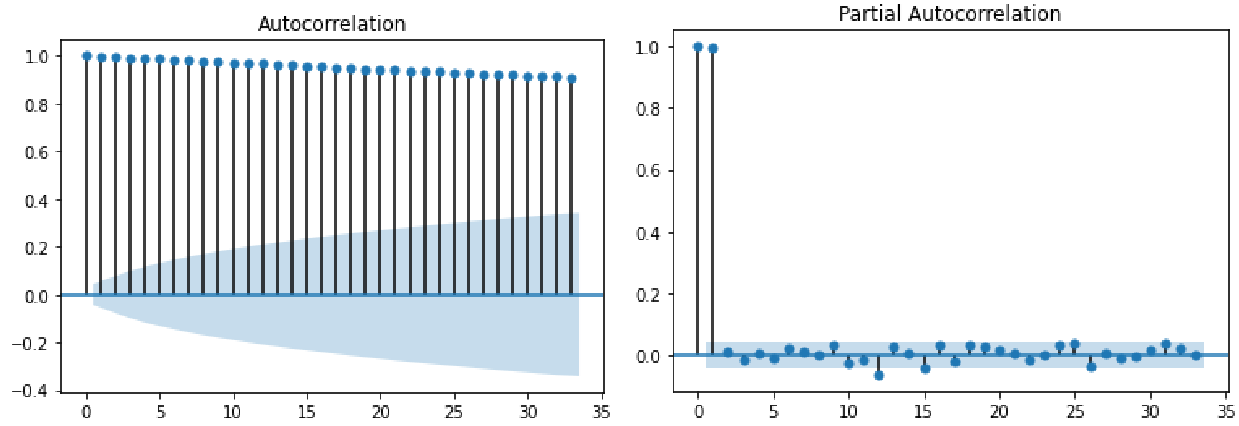
$$H_0 : \text{The data has non-stationairty}$$

$$H_a : \text{The data has stationarity}$$

The p-value of it was 0.999. Since the p-value is higher than 0.05, we can't reject the null hypothesis; therefore, this data is non-stationary. To build an ARIMA model, we need to deal with non-stationarity. We conducted the Dicky-Fuller test again on the differenced data and the p-value was very close to 0 which means we have enough evidence to reject the null hypothesis; therefore, this data is stationarity. We will use this differenced data later when we build an ARIMA model.

Autocorrelation

We created an ACF (Autocorrelation Function) plot a PACF (Partial Autocorrelation Function) plot using statsmodels package in Python.



(a) Autocorrelation

(b) Partial Autocorrelation

Figure 4.11: Autocorrelation and Partial Autocorrelation Plots

One of the alternatives to the ACF is the PACF. PACF gives us partial autocorrelation which means PACF only considers additional autocorrelation with each step back in the past. The Partial Autocorrelation plot has a much better representation of the autocorrelation. We can find that there is a strong positive autocorrelation with lag 1.

Table 4.1: Top 10 Variables That Have a Positive Correlation with Target Variable

Variable	**Correlation**
Price to book	0.99
RS ratio	0.96
US dollar exchange rates in Argentina	0.96
US dollar exchange rates in Turkey	0.93
Gold Price	0.91
US dollar exchange rates in Brazil	0.88
GDP	0.86
US dollar exchange rates in India	0.78
EPS diluted	0.74
Inflation Rates	0.65

Table 4.1 shows the top 10 Variables that have a positive correlation with the target variable. Based on Table 4.1, many variables that have a high positive correlation with the target variable are US dollar exchange rates. From this information, we can think that there could be a multicollinearity issue. The absolute value of a negative correlation between the target variable and predictor variables is relatively lower than the absolute value of a positive correlation. Dividend yield and Earnings yield are the only variables that have a high absolute value of a negative correlation.

To check if we have a multicollinearity issue, we will check the variance inflation factor (VIF). Generally, if a VIF is higher than 10, we can consider that we have high multicollinearity. There were only two variables that have a lower than 10 VIF value, and the rest 32 variables have a higher than 10 VIF value.

Beta of the stock

Beta is a measure of a stock's volatility in relation to the overall market (SP 500). The SP 500 index has Beta 1. [Run] High-beta stocks means the stocks could possibly provide higher return while it is riskier. Whereas, low-beta stocks provide lower returns with less risk. We will calculate the beta of the Apple stock to understand how volatile Apple stock is.[Ken]

$$\text{Best coefficient}(\beta) = \frac{Covariance(R_e, R_m)}{Variance(R_m)}$$

where:

$$\begin{aligned} R_e &= \text{the return on an individual stock} \\ R_m &= \text{the return on the overall market} \\ \text{Covariance} &= \text{how changes in a stock's returns are related to changes in the market's returns} \\ \text{Variance} &= \text{how far the market's data points spread out from their average value} \end{aligned}$$

We used SP 500 index as the overall market and got the SP 500 data from Yahoo!Finance. The beta of Apple stock is 1.17 which is slightly higher than 1. Therefore, we can conclude that even though Apple stock is a little bit risky, it will provide higher returns than those stocks that have a less value of beta.

4.2.2 Modeling

We split the data as training and testing datasets with the proportion of 8 to 2 and also, scaled the data.

4.2.2.1 Classical Time Series Models

1) ARIMA Model

The ARIMA model is a univariate time series model, so we will use the differenced data of the adjusted close price only rather than using all the external variables.

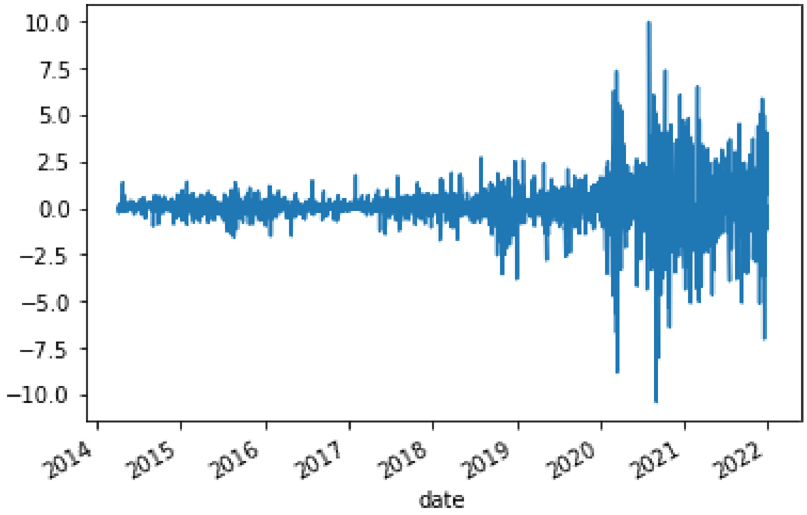
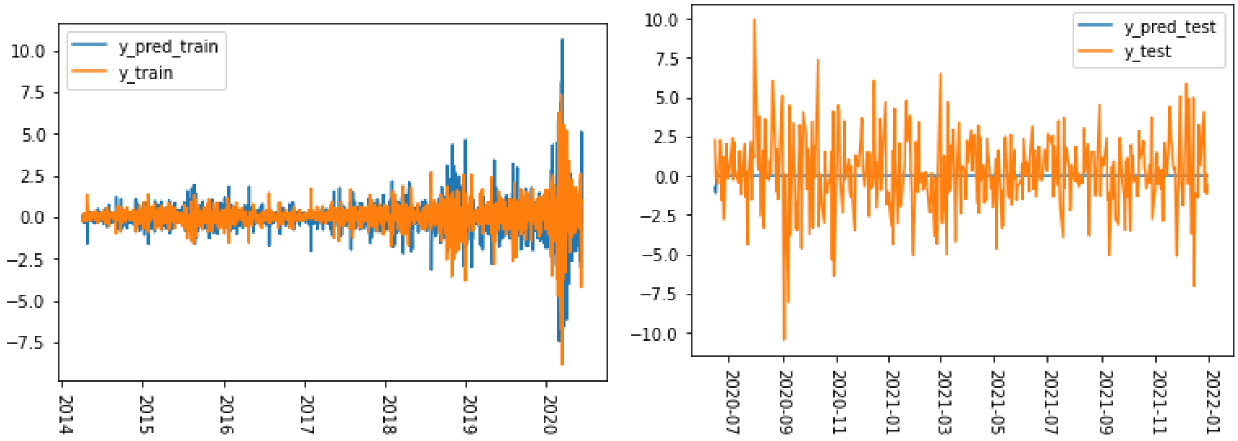


Figure 4.12: Differenced Adjusted Close Price Plot

We trained the differenced data using ARIMA(1, 1, 1).



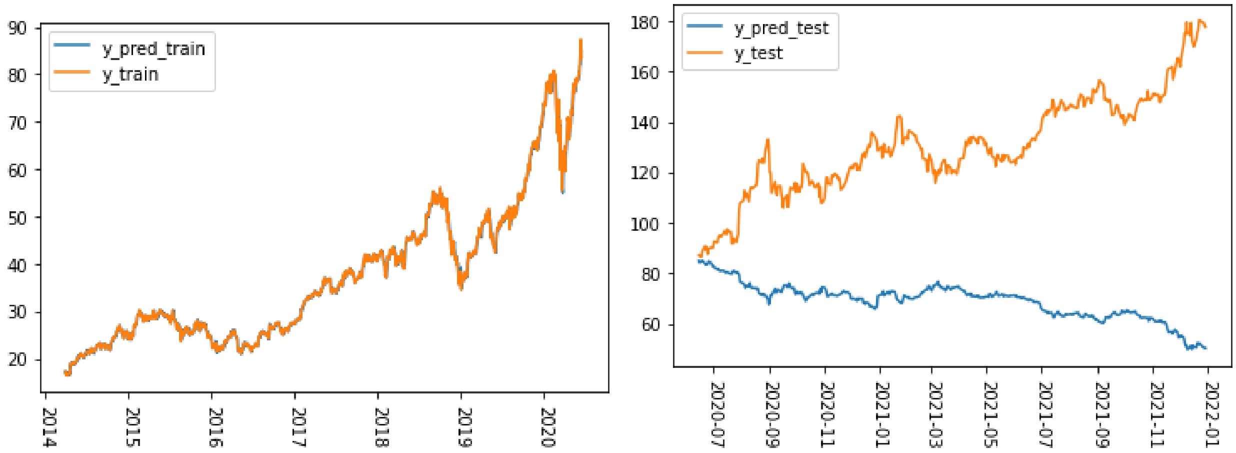
(a) Training Set

(b) Test Set

Figure 4.13: ARIMA - Predicted and Actual Values

2) SARIMAX Model

To find the best parameters for the SARIMAX model, we used pmdarima.arima package in python that performs a stepwise search to get AR and MA terms that has the lowest value of AIC. Based on this function, since SARIMAX(0,1,0) has the lowest value of AIC, we will use this parameter.



(a) Training Set

(b) Test Set

Figure 4.14: SARIMAX - Predicted and Actual Values

Table 4.2: ARIMA & SARIMAX - MSE

Dataset	**ARIMA - MSE**	**SARIMAX - MSE**
Training Set	1.35	0.63
Test Set	5.91	4458.57

Since the ARIMA model is built using the differenced data, we will not be able to compare the result to other models; however, when we look at the test set plot in Figure 4.13 which shows the predicted and actual values of ARIMA, we can see that this model does not perform well.

When we look at the test set plot in Figure 4.14 that shows the predicted and actual values of SARIMAX, we can see that this model cannot even predict the trend of price. Also, the value of MSE is extremely high.

4.2.2.2 Supervised Models

Since supervised models are not meant for time series data, we need to perform some data engineering. We created an additional three columns, month, year, and day. We extract these from the date column which is used as the index for other models.

1) Liner Regression Models using PCA

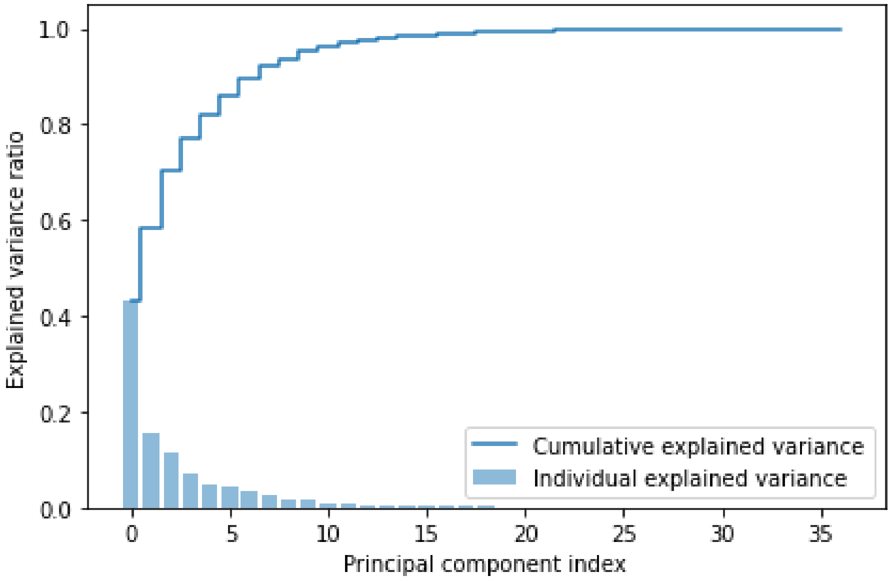
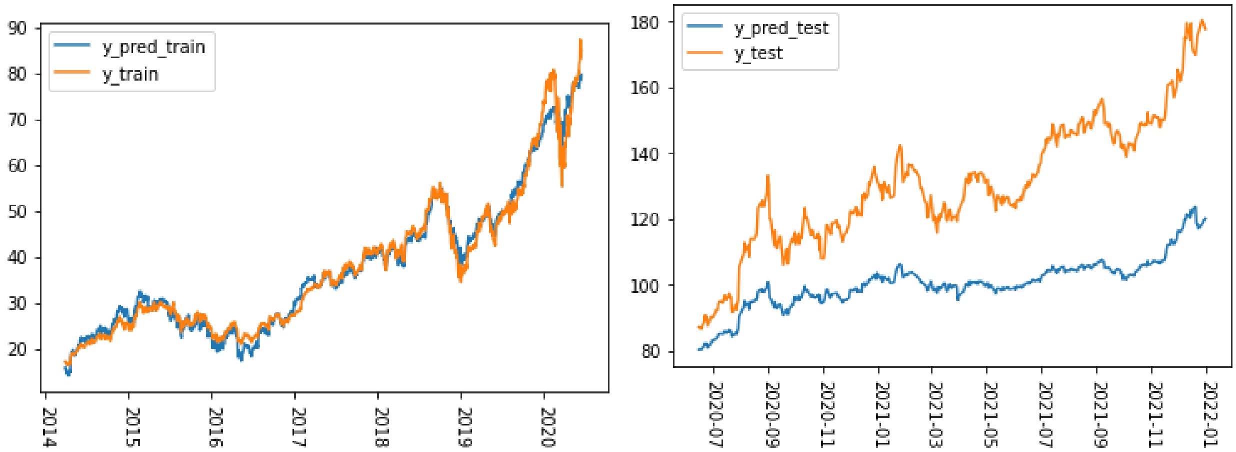


Figure 4.15: The Cumulative Proportion of Variance Explained Plot

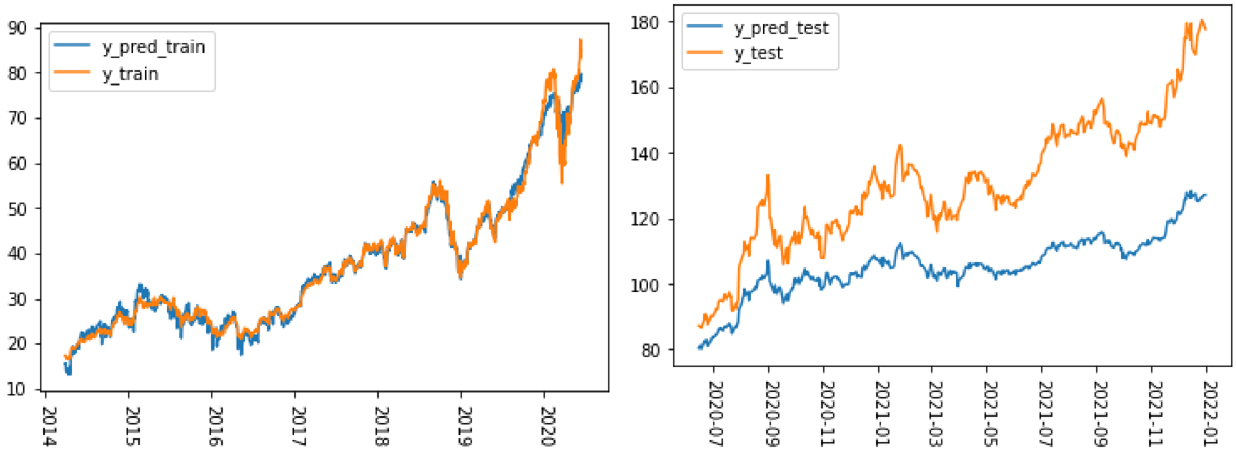
Figure 4.15 is a plot showing the cumulative proportion of variance. Based on Figure 4.15, the first ten principal components can explain more than 95% of variance and the first 18 principal components can explain more than 99%. We will build three linear regression models using 10 PCs, 18 PCs, and all 37 PCs.



(a) Training Set

(b) Test Set

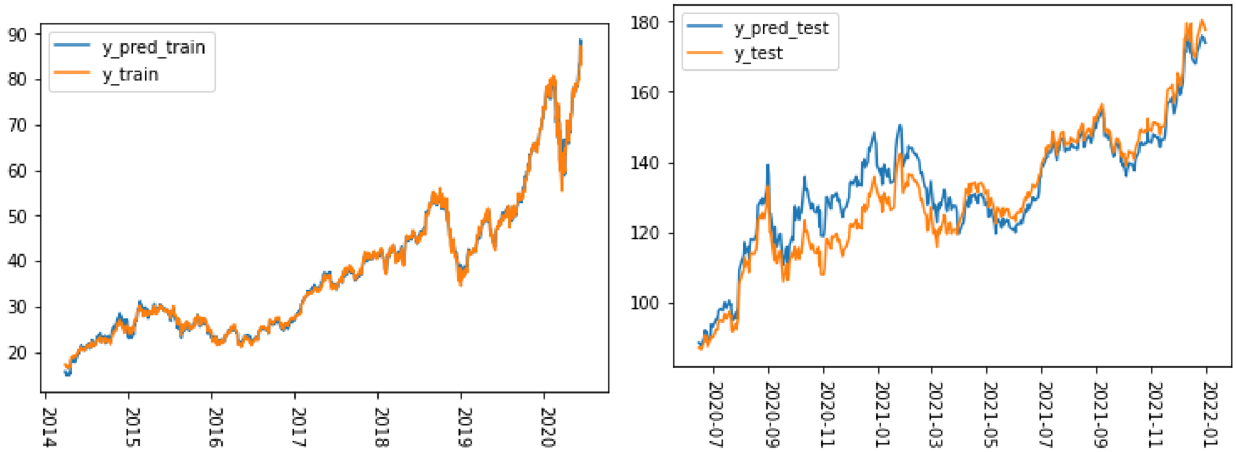
Figure 4.16: Liner Regression Using 10 PCs - Predicted and Actual Values



(a) Training Set

(b) Test Set

Figure 4.17: Liner Regression Using 18 PCs - Predicted and Actual Values



(a) Training Set

(b) Test Set

Figure 4.18: Liner Regression Using All 37 PCs - Predicted and Actual Values

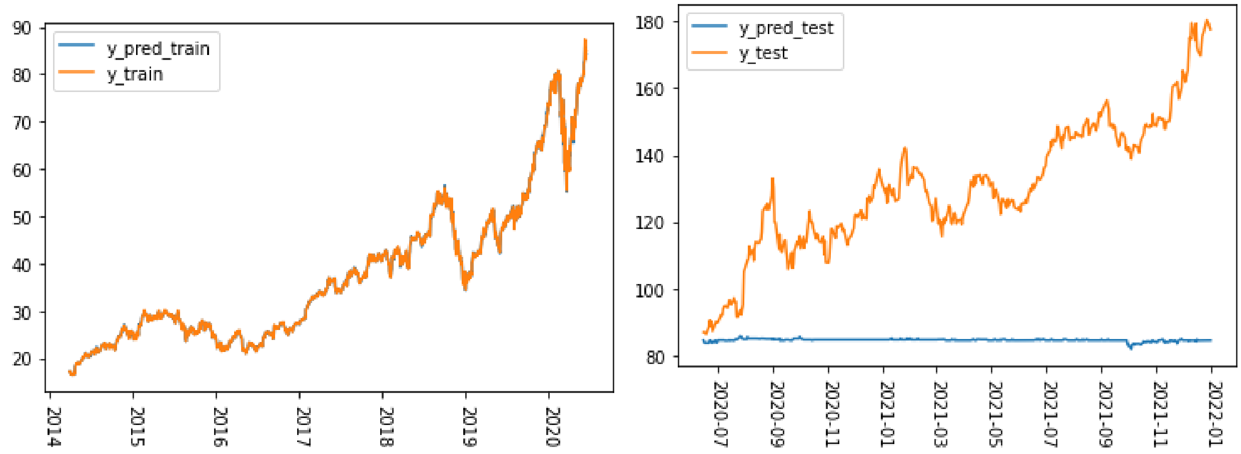
Table 4.3: Liner Regression Using PCA

Dataset	**10 PCs - MSE**	**18 PCs - MSE**	**37 PCs - MSE**
Training Set	4.32	2.65	0.28
Test Set	1044.69	736.62	38.75

Using all 37 PCs has the best result so far based on the value of MSE.

2) Random Forest

We also built a Random Forest model.



(a) Training Set

(b) Test Set

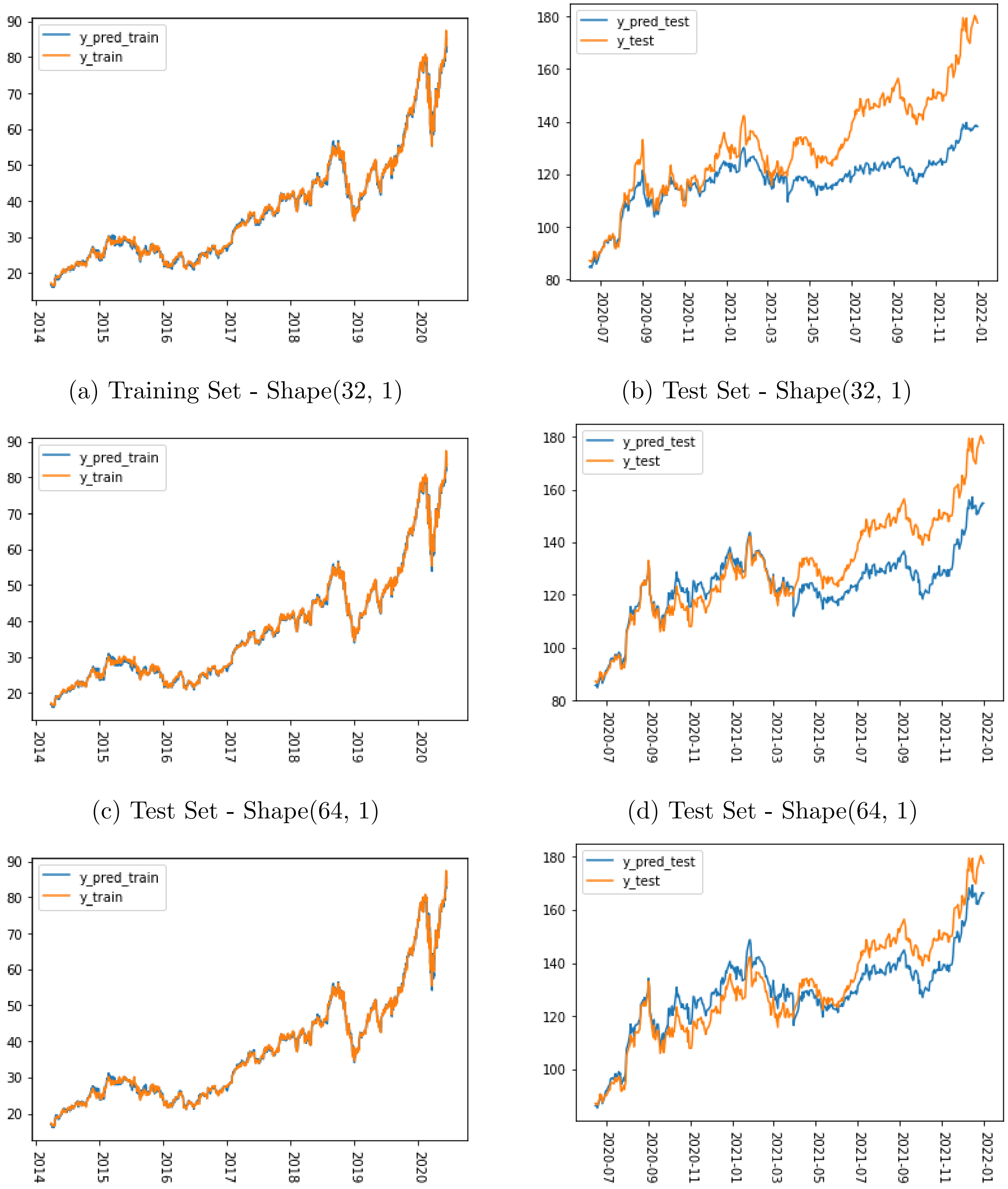
Figure 4.19: Random Forest - Predicted and Actual Values

Table 4.4: Random Forest - MSE

Dataset	**Mean Squared Error**
Training Set	0.02
Test Set	2461.25

One of the main problems with the Random Forest model is that if the prediction is not in the window where the model is trained, the model does not perform well.

4.2.2.3 Deep Learning model - LSTM



(a) Training Set - Shape(32, 1)

(b) Test Set - Shape(32, 1)

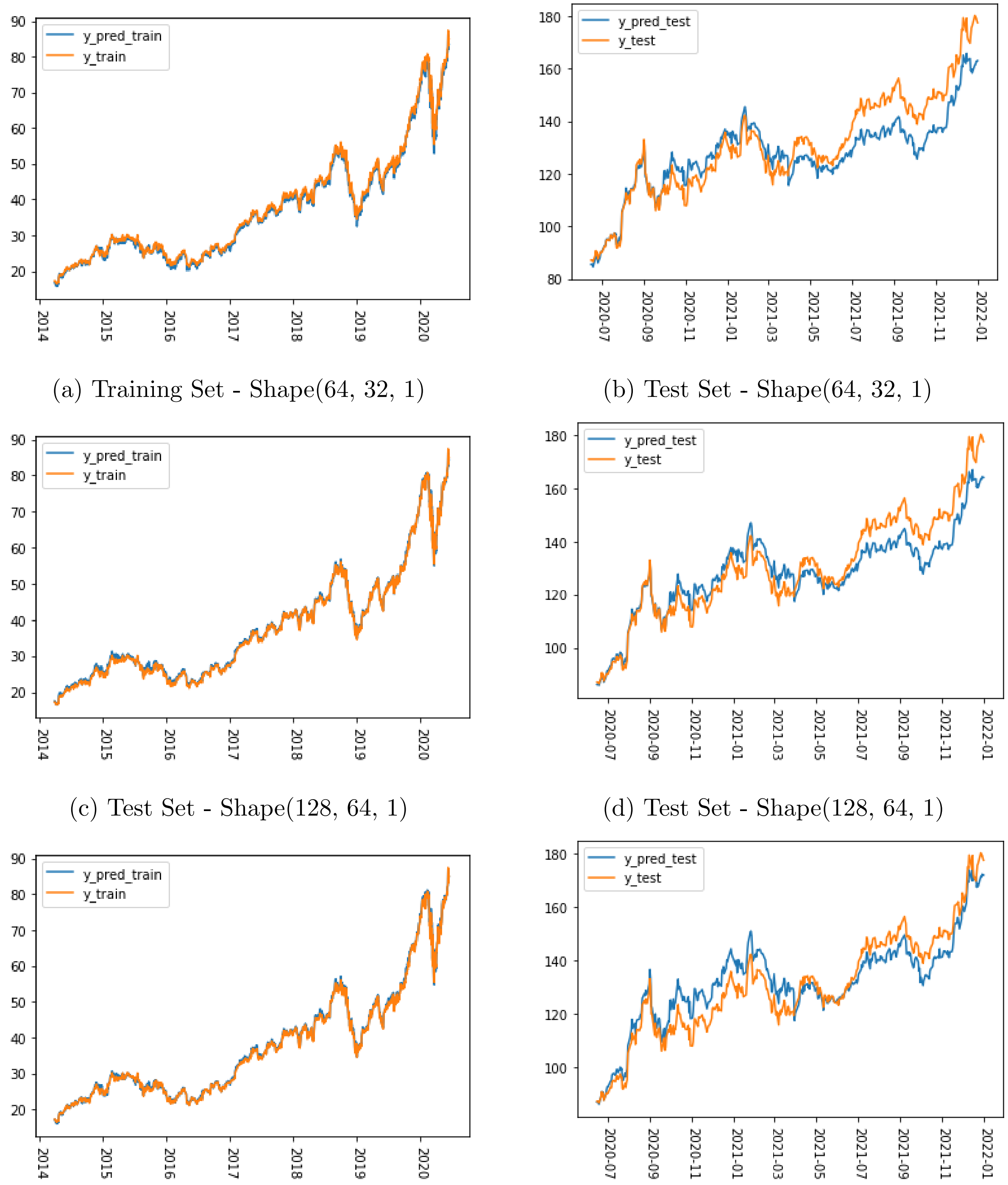
(c) Test Set - Shape(64, 1)

(d) Test Set - Shape(64, 1)

(e) Test Set - Shape(128, 1)

(f) Test Set - Shape(128, 1)

Figure 4.20: LSTM with 2 Layers - Predicted and Actual Values



(a) Training Set - Shape(64, 32, 1)

(b) Test Set - Shape(64, 32, 1)

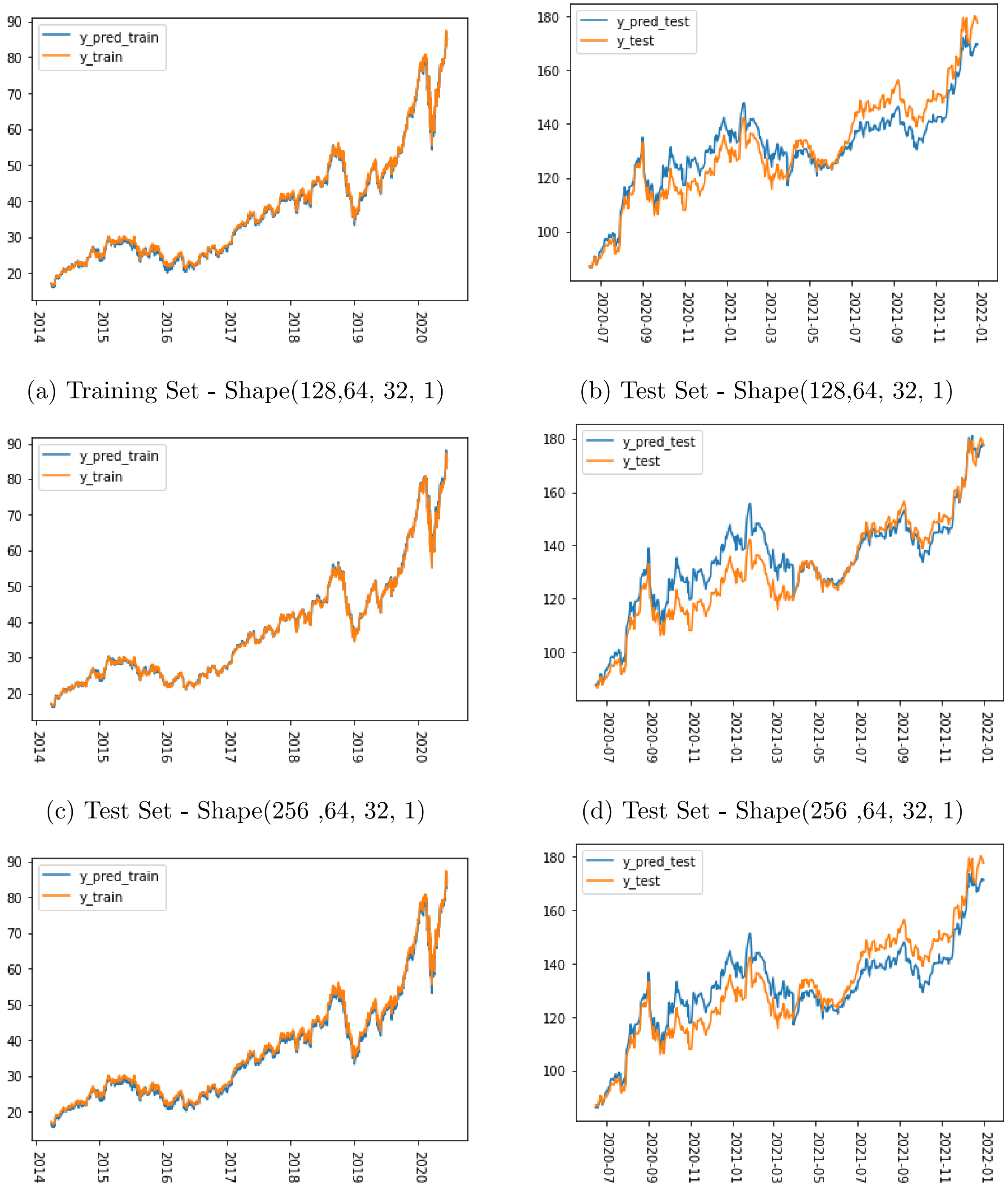
(c) Test Set - Shape(128, 64, 1)

(d) Test Set - Shape(128, 64, 1)

(e) Test Set - Shape(256 ,64, 1)

(f) Test Set - Shape(256 ,64, 1)

Figure 4.21: LSTM with 3 Layers - Predicted and Actual Values



(a) Training Set - Shape(128,64, 32, 1)

(b) Test Set - Shape(128,64, 32, 1)

(c) Test Set - Shape(256 ,64, 32, 1)

(d) Test Set - Shape(256 ,64, 32, 1)

(e) Test Set - Shape(256 ,128, 32, 1)

(f) Test Set - Shape(256 ,128, 32, 1)

Figure 4.22: LSTM with 4 Layers - Predicted and Actual Values

We built LSTM models with different shapes and the number of layers.

Table 4.5: LSTM with 2, 3, and 4 Layers - MSE

Part (a) 2 Layers

Dataset	**(32, 1)-MSE**	**(64, 1)-MSE**	**(128, 1)-MSE**
Training Set	0.34	0.23	0.21
Test Set	285.74	143.42	55.69

Part (b) 3 Layers

Dataset	**(64, 32, 1)-MSE**	**(128, 64, 1)-MSE**	**(256 , 64, 1)-MSE**
Training Set	0.77	0.28	0.27
Test Set	64.73	46.92	41.70

Part (c) 4 Layers

Dataset	**(128, 64, 32, 1)-MSE**	**(256 ,64, 32, 1)-MSE**	**(256 ,128, 32, 1)-MSE**
Training Set	0.36	0.15	0.95
Test Set	40.69	51.59	47.82

Table 4.5 shows the value of MSE for the LSTM models with 2, 3, and 4 layers. When we look at Table 4.5, we can conclude that when there are more layers, the results are more likely to be accurate.

CHAPTER 5

Conclusion & Further Discussion

5.1 Conclusion

5.1.1 Text Mining Models

Table 5.1: Results of Text Mining Models

Model	**Epoch**	**Loss**	**Test Accuracy**
LSTM	16	0.793	66.5%
GRU	14	0.807	65.1%
BERT Sentimental Analysis	-	-	26.3%
BERT Fine-Tuning	8	0.415	84.9%

As our exploratory data analysis suggested, our headlines were composed of mostly positive sentiments with some negative sentiments as well. We utilized multiple RNN models and BERT features. Our best model was BERT Fine-Tuning with eight epochs, 0.415 loss, and 84.9% accuracy.

5.1.2 Predictive Models

Table 5.2: Results of the Predictive Models

Model	Type	Details	MSE
SARIMAX	Classical Time Series Model	-	4458.57
LR Uisng PCA	Supervised Model	All 37 PCs	38.75
LSTM	Deep Learning Model	4Layers (128, 64, 32, 1)	40.69

Table 5.2 shows the results of the predictive models. Based on Table 5.2, the liner regression model using PCA has the best result. However, the value of LSTM's MSE is also very low. We could possibly find better parameters using the LSTM model.

5.2 Further Discussion

5.2.1 Text Mining Models

One of our biggest shortcomings was the lack of detail in the data. While we did not have that data at our disposal, we did a few literature reviews to find scientific articles written about the correlations between the sentiment of news headlines and stock price. Our project is useful in our functionality of predicting the sentiment of a potential headline and using it as a tool for journalists to give confidence in the reaction to the headline whether it be positive, negative, or neutral. One further extension is to create more labels that express sentiments in more detail rather than three vague categories. Is there a difference between a "sad" headline versus an "angry" headline? Does the use of inflammatory language have a more polarizing effect on one aspect of the market than another? These are all questions that could be further explored within the context of sentiment analysis.

5.2.2 Predictive Models

We expected that our LSTM model would work better since this model is the most recent model; however, the Linear Regression model using PCA worked the best. As we mentioned before, we could possibly change parameters to have better results for the LSTM model. Additionally, our SARIMAX can not predict stock prices at all. To solve this problem, we may need to investigate the parameters that we used.

REFERENCES

[AA] Afshine Amidi and Shervine Amidi. "Recurrent Neural Networks." `https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks`.

[Ash17] Niki Parmar Jakob Uszkoreit Llion Jones Aidan N. Gomez Lukasz Kaiser Illia Polosukhin Ashish Vaswani, Noam Shazeer. "Attention Is All You Need." *Computation and Language*, **15 pages**(1), 2017.

[Bae] Baeldung. "What is a Learning Curve in Machine Learning?" `https://www.baeldung.com/cs/learning-curve-ml#:~:text=One\%20of\%20the\%20most\%20widely,the\%20model\%20fits\%20new\%20data`.

[Bak] Chaya Bakshi. "Random Forest Regression." `https://levelup.gitconnected.com/random-forest-regression-209c0f354c84`.

[Bir] Adarsh Biradar. "Sentiment Analysis using bert." `https://www.kaggle.com/code/adarshbiradar/sentiment-analysis-using-bert/data`.

[BIS] BIS. "US dollar exchange rates." `https://www.bis.org/statistics/xrusd.htm`.

[DCL] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. `https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270`.

[DF79] David A Dickey and Wayne A Fuller. "Distribution of the estimators for autoregressive time series with a unit root." *Journal of the American statistical association*, **74**(366a):427–431, 1979.

[FF04] Eugene F. Fama and Kenneth R. French. "The Capital Asset Pricing Model: Theory and Evidence." *Journal of Economic Perspectives, American Economic Association*, **18**(3):25–46, 2004.

[FREa] FRED. "Federal Funds Effective Rate." `https://fred.stlouisfed.org/series/FEDFUNDS`.

[FREb] FRED. "Gross Domestic Product (GDP)." `https://fred.stlouisfed.org/series/GDP`.

[HUI] "NLP — BERT Transformer." `https://jonathan-hui.medium.com/nlp-bert-transformer-7f0ac397f524`, author=Jonathan Hui.

[Jis] Jishnu. "Global Gold price - Historical Data (1979-Present)." `https://www.kaggle.com/datasets/jishnukoliyadan/gold-price-1979-present`.

[Ken] Will Kenton. "Beta." https://www.investopedia.com/terms/b/beta.asp#:~:text=A\%20security's\%20beta\%20is\%20calculated,returns\%20over\%20a\%20specified\%20period.&text=The\%20beta\%20calculation\%20is\%20used,the\%20rest\%20of\%20the\%20market.

[Kor] Joos Korstanje. "How to Select a Model For Your Time Series Prediction Task [Guide]." https://neptune.ai/blog/select-model-for-time-series-prediction-task.

[Mal73] Burton Gordon Malkiel. *A random walk down Wall Street : the time-tested strategy for successful investing.* W. W. Norton Company, Inc., Apr 1973. https://en.wikipedia.org/wiki/A_Random_Walk_Down_Wall_Street.

[Run] Rune. "Calculate the market (SP 500) BETA with Python for any Stock." https://www.learnpythonwithrune.org/calculate-the-market-sp-500-beta-with-python-for-any-stock/.

[sta] statsmodels. "SARIMAX: Introduction." https://www.statsmodels.org/stable/examples/notebooks/generated/statespace_sarimax_stata.html.

[USB] USBureauofLaborStatistics. "Unemployment Rate." https://data.bls.gov/timeseries/LNS14000000.

[USI] USInflationCalculator. "Historical Inflation Rates: 1914-2022." https://www.usinflationcalculator.com/inflation/historical-inflation-rates/.

[Wika] Wikipedia. "Autocorrelation." https://en.wikipedia.org/wiki/Autocorrelation.

[Wikb] Wikipedia. "Autoregressive integrated moving average." https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average.

[Wikc] Wikipedia. "Efficient-market hypothesis (EMH)." https://en.wikipedia.org/wiki/Efficient-market_hypothesis.

[Wikd] Wikipedia. "Principal component analysis." https://en.wikipedia.org/wiki/Principal_component_analysis#:~:text=vectors.,the\%20data\%20are\%20linearly\%20uncorrelated.

[Wike] Wikipedia. "Random walk hypothesis." https://en.wikipedia.org/wiki/Random_walk_hypothesis.

[Wikf] Wikipedia. "Stock market prediction." https://en.wikipedia.org/wiki/Stock_market_prediction.