Uncertainty Visualization for Stock Prediction

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Computer Science Engineering

by

Monika Yadav 16UCS109
Saloni Chauhan 16UCS165
Vrinda Goel 16UCS216

Under Guidance of
Dr. Subrat K Dash

Department of Computer Science Engineering
The LNM Institute of Information Technology, Jaipur

December 2019

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled "Uncertainty Visualization of Stock Prediction", submitted by Monika Yadav(16UCS109), Saloni Chauhan(16UCS165) and Vrinda Goel(16UCS216) in partial fulfillment of the requirement of the degree in Bachelors of Technology(B. Tech), is a bonafide record of work carried out by them at the Department of Computer Science Engineering, The LNM Institute of Information Technology, Jaipur, Rajasthan, India, during the academic session 2019-2020 under my supervision and guidance and the same has not been submitted elsewhere for the awardance of any other degree. In my/our opinion, this thesis is of standard required for the award of the degree of Bachelor of Technology(B. Tech).

Subrat Kumar Dash

Digitally signed by Subrat Kumar Dash
DN: cn=Subrat Kumar Dash, o=The LNM Institute of Information Technology, ou=CSE, email=subrat@lnmiit.ac.in, c=IN
Date: 2020.09.08 16:39:57 +05'30'

13th December 2019

Adviser: Subrat K Dash

Dedicated to My Family and Friends

Acknowledgments

We would like to express our gratitude to our instructor Dr.Subrat K Dash. He has been a constant source of inspiration and has guided us throughout our journey. Without his constant support, motivation and guidance this project would not have been successful.

Abstract

Stock Market is a highly appealing sphere that excites many cause of its unpredictable nature and scope of profits. Several trends govern a stock's price, ranging from time of the year to the parent company's performance, making it highly volatile. With the massive evolution in the field of Computer Science, especially neural networks, several attempts have been made to create algorithms and models that predict stock values- although many have good accuracy, the nature of the domain makes it nearly impossible to design models and algorithms as accurate as those available, for other applications.

Uncertainty is trivial in the real world, but attempting to predict a quantity without taking into account the probability of 'unknown-ness', may lead to highly differentiated and sometimes incorrect results. This project aims at analyzing the uncertainty behavior in stock prices, using the powerful tool of data visualization, which will also help us choose the model is best suited to make stock predictions.

Contents

Chapter *1*

Introduction

1.1 The Area of Work

The stock market has been around for centuries now and it is still one of the most unpredictable and changing fields today. Individuals and organisations, invest in companies, by purchasing the its stocks at a certain price, and then selling the stocks, ideally at a higher price to gain a profit. The stock prices are extremely volatile and vary even by the minute, as they depend on a large array of factors varying from the time of the day or year to the company's reputation. So, a big challenge in the field of Computer Science is to make a model that can predict stock prices accurately, so as to assist experienced and new investors to make the right investments at the right time. Our project aims to compare some existing prediction models and evaluate them on certain parameters, with the help of some tools, to choose the most suitable model.

1.2 Motivation

The Stock Market is a domain which can greatly benefit from an accurate prediction model, as it will encourage more people and organisations to invest and allow them to make more profit, thus boosting the global trade and economy. Also, a lot of research is going on in the field Neural Networks and its applications in the real world, and the creation of a model that can perform well in such a volatile field, will help gain insight into the scope of the field.

1.3 Problem Addressed

An individual or organisation invests in the stock market, with the hope of maximising their profit, by selling the stocks at the right moment. Being such a dynamic field, it takes a lot of experience and knowledge for one to make the right call to buy or sell a stock and when to do it- this makes it very risky to make big investments and also puts new investors at a disadvantage. So, a model that can assist with such decisions could be very useful- a standard algorithmic model, will not do justice to predicting the prices accurately, due to the plethora of dependencies- we need a model that can learn and adapt on its

own to the on-going market trends. This is why a neural network based model, could be ideal for this use case. Our project evaluates few neural network based prediction models on certain parameters and then visualises the uncertainty of the model with the best accuracy based on its error in predictions- this helps see which model is best suited for predictions in relative and absolute terms.

Chapter *2*

Literature Survey

2.1 Dataset

We have used daily timeseries data of stock prices. This doesn't include weekends which are non-working days in stock market. The data consists of 7 labelled features: Date, Open, Close, Low, High, Adj. Close, and Volume.

2.1.1 Source

The data is taken from Yahoo Finance website. In this study, we have worked on the stock prices of an e-commerce company Amazon (AMZN) spanning from 1st Jan 2009 to 25th March 2019, which gives a total of 2518 observations.

2.1.2 Features

The data has the following labelled attributes:
• *Opening Price* is the first price of a stock at the start of a trading day.
• *Closing Price* is the price of the stock at the end of a trading day.
• *High and Low Prices* are high and low prices of the stock on a trading day. They are used to measure volatility of stock market for assessing the risk of a security.
• *Adj. Closed Price* is the true price of a stock that shows the stock's value after distributing dividends.
• *Volume* is the number of stocks or contracts traded for a security in all the markets during a given time period.

2.2 Stock Price Prediction and Uncertainty

Stock market prices have a volatile nature, and it is a challenge to forecast a financial time series. There are many factors involved in predicting stock market prices which have been studied to develop better models. The main purpose of the study is to help the investors make a profitable decision. Depending on the features of the market that are considered for making predictions, there are 2 ways to

analyse stock prices: (1) technical analysis, and (2) fundamental analysis.
In technical analysis, the historical prices are taken as the only factor for future predictions, whereas in fundamental analysis it is believed that the available factors are incorporated in the prices, so other internal and external features are important and viable for future price predictions. These additional factors can be the company turnover, features extracted from social media, financial news, etc.

2.2.1 Timeseries Analysis

In our work, we have performed a technical analysis by using the historical prices as the feature to train our models. This is useful for short term forecasting, in which case, the model can be used to predict prices for the next day to help an investor buy a company's stocks that are predicted to reach a price that is higher than the existing price. It is reasoned that the history of stock prices consists of trends and patterns that are repeated, which can be utilized for making predictions.

2.2.2 Uncertainty

In stock market, financial timeseries data has much uncertainty and noise which hinders the extraction of useful information. There are many studies which are focused on mining useful features that can be related factors for a timeseries, and using them can improve the accuracy of prediction. Although, it is challenging to achieve higher accuracy in price prediction, but we can utilize the information of uncertainty to help investors gain wider insights. The uncertainty associated with predictions can be an important factor in decision making for investors to strategise buy-and-sell actions.

2.3 Usefulness of Visualization

In a visual representation of stock prices, it is easier to interpret the movement of prices with the rise and fall of the graph. This is useful when someone wants to buy and sell stocks in a short period. In our study, we have experimented with uncertainty visualization using 3 types of plotting methods : (1) Scatter Plot, (2) Error Bars, and (3) Line Plot. For instance, when an investor is keen to buy stocks of a company, referring to these plots is a preferable option as compared to dealing with numbers of various stocks. The significance of plotting uncertainty attached to every prediction is that, it incorporates the probabilistic behaviour of stock market. It provides the information about the probability with which our model predicts the value of a closing price on a trading day and the margin of error that can be assumed for that value. So, we get an upper limit and a lower limit on our prediction of stock price, for example, using error bars. It reflects a risk factor in investment, which can be examined easily with visualization.

2.4 Forecasting Prices with Deep Learning

The traditional statistical methods, for e.g. ARIMA, have been used to predict financial time series in the past. Generally, they assume that the timeseries has a linear generation process which is not the case, hence, they have proven to be less reliable. There are many machine learning models which can incorporate the non-linearity of data to predict future stock prices. The most popular models for stock forecasts are deep learning models. These models are based on artificial neural networks which are mathematical models or computational models that take after biological neural networks, consisting of an interconnected group of artificial neurons. These are designed and coded to acquire knowledge via the network through a learning process and store the acquired knowledge based on the weights associated with the connections between neurons. Neural networks can work parallel with input variables which makes them suitable for handling large datasets efficiently. They don't require apriori assumptions related to the problem, and the regression models constituted in the neural networks are non-parametric. The artificial neurons, called units are assembled in layers in a deep learning model architecture. There is an input layer, a finite number of hidden layers, and an output layer. The hidden layers, at each level, transform its input data into a slightly more abstract and composite representation, consequently they can approximate the timeseries function. This enables the model to extract high level abstract-features from the data, and predict future stock prices. In our work, we have used *Multi-Layer Perceptron (MLP)*, *Convolutional Neural Network (CNN)*, and *Long Short Term Memory (LSTM)* models for the task of prediction.

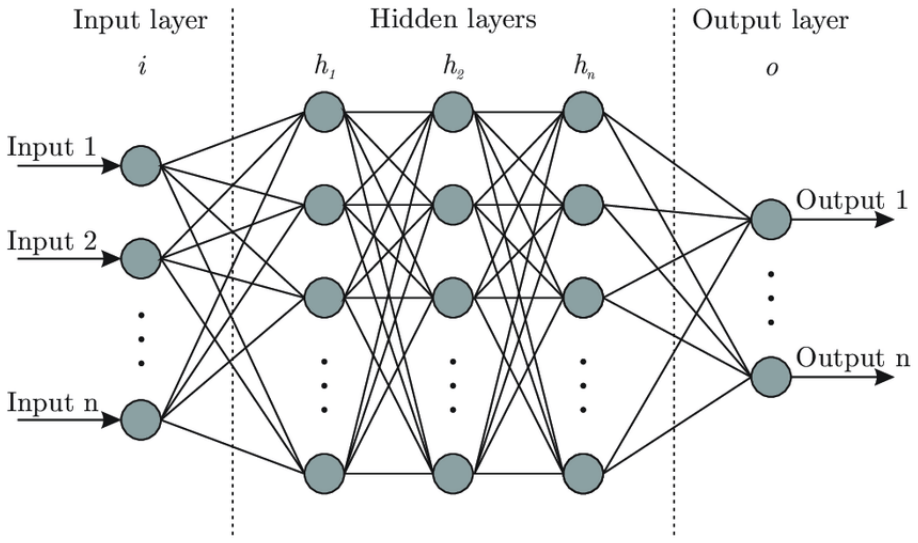

Figure 2.1 Neural Network

2.5 Work Flow

We designed the following workflow for our project:

2.5.1 Forming a problem statement

The problem statement as we have described, addresses the visualization techniques for plotting uncertainties values after predicting stock prices using different deep learning models.

2.5.2 Finding dataset

We found a datewise timeseries of stock prices to train our prediction models and also provide uncertainty calculations. Initially, we intended to use time indexed dataset, but we couldn't find a reliable source.

2.5.3 Data Analysis

This involves data pre-processing and feature extraction process. It deals with handling missing values, incorrect data, and data normalization.

2.5.4 Price Prediction

We have used 3 deep learning models for predicting stock prices. In several experiments, we have trained them with hyperparameter tuning for achieving best results.

2.5.5 Uncertainty Calculation

The next step is to measure uncertainty values using confidence intervals for each prediction. This gives a range of error for a prediction.

2.5.6 Visualization of Uncertainty

Our main task is to visualize the uncertainty of models to provide insights about our predictions and an easier way to interpret the values for decision-making.

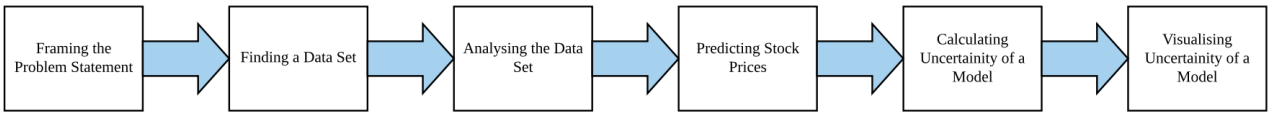

Figure 2.2 Process Flow

Chapter *3*

Methodology

3.1 Data Pre-processing

The pre-processing consisted of the following steps:

1. The data was already in the tabular form.
2. Traders and other stakeholders use closing price as a reference point to calculate performance over a time. In fact, investors and other stakeholders make decisions on closing stock prices. Institutional investors monitor a stock's closing price to make decisions regarding their investment portfolios. Hence, we will drop all columns except the date and closing price.
3. The data did not contain any missing or null values.
4. The sequence was then normalized by dividing each value with the maximum value in the sequence.
5. The sequence was then split into smaller sequences like a sliding window algorithm to obtain the x-

```
[ ] data.isnull().values.any()

    False

[ ] X_train.shape

    (1973, 50, 1)

[ ] data['Date'] = pd.to_datetime(data['Date'])

[ ] sequence = (np.asarray(data['Close']))
    norm = max(sequence)
    sequence = sequence/norm
```

Figure 3.1 Normalising the data

training values to make predictions. The value at the end of these smaller sequences was the y-training value.

```
[ ]  def split_sequence(sequence, n_steps):
       X, y = list(), list()
       for i in range(len(sequence)):
         # find the end of this pattern
         end_ix = i + n_steps
         # check if we are beyond the sequence
         if end_ix > len(sequence)-1:
           break
         # gather input and output parts of the pattern
         seq_x, seq_y = sequence[i:end_ix], sequence[end_ix]
         X.append(seq_x)
         y.append(seq_y)
       return X, y
```

Figure 3.2 Splitting the sequence into smaller sequences

3.2 Predicting values

The performances of the neural network models can be varied with changing and complicating/simplifying the architecture. For the sake of comparing the models, we have chosen to use the simplest forms of each neural network model, with minimum number of layers.

3.2.1 Convolutional Neural Network model

CNNs are shift invariant and space invariant artificial neural networks. They help in significantly reducing the number of learned parameters and giving similar results, if not better. The architecture we used in our project consists of 2 layers with activation function ReLu, followed by a max pooling layer(to further reduce the number of parameters), followed by a dense layer.

```
[ ]  def split_sequence(sequence, n_steps):
       X, y = list(), list()
       for i in range(len(sequence)):
         # find the end of this pattern
         end_ix = i + n_steps
         # check if we are beyond the sequence
         if end_ix > len(sequence)-1:
           break
         # gather input and output parts of the pattern
         seq_x, seq_y = sequence[i:end_ix], sequence[end_ix]
         X.append(seq_x)
         y.append(seq_y)
       return X, y
```

Figure 3.3 Architecture

3.2.1.1 Result

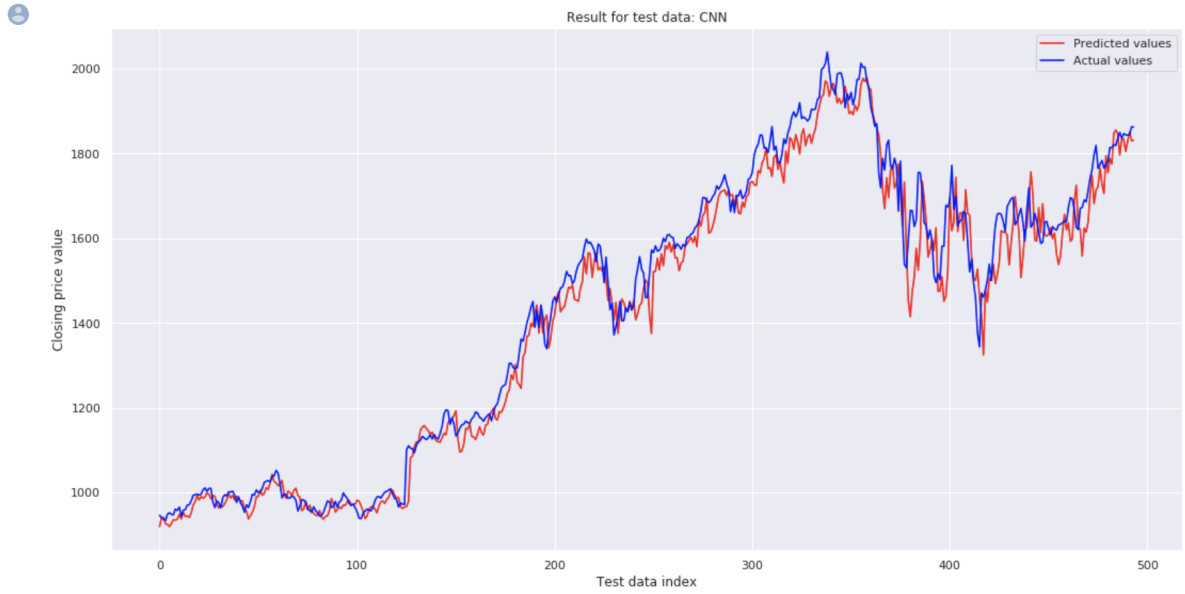

Figure 3.4 Result for CNN test data

3.2.2 Long-Short Term Memory model

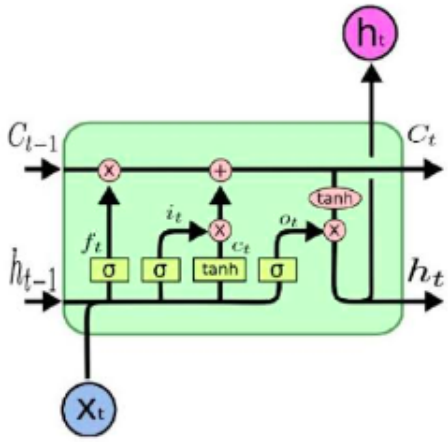

Figure 3.5 A cell in LSTM architecture

LSTM is a special kind of Recurrent Neural Network. It uses cell states for information flow. They can selectively remember or forget things. A cell state takes the following three inputs.

1. The previous cell state: The trend of the previous days' sequence. This could be uptrend or a downtrend

2. The previous hidden state: The stock price value on the previous day.
3. The current input: The stock price on the day that we want the prediction for.

```
model = Sequential()
model.add(LSTM(10, activation='relu', input_shape=(n_steps, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

Figure 3.6 LSTM architecture used

3.2.2.1 Result

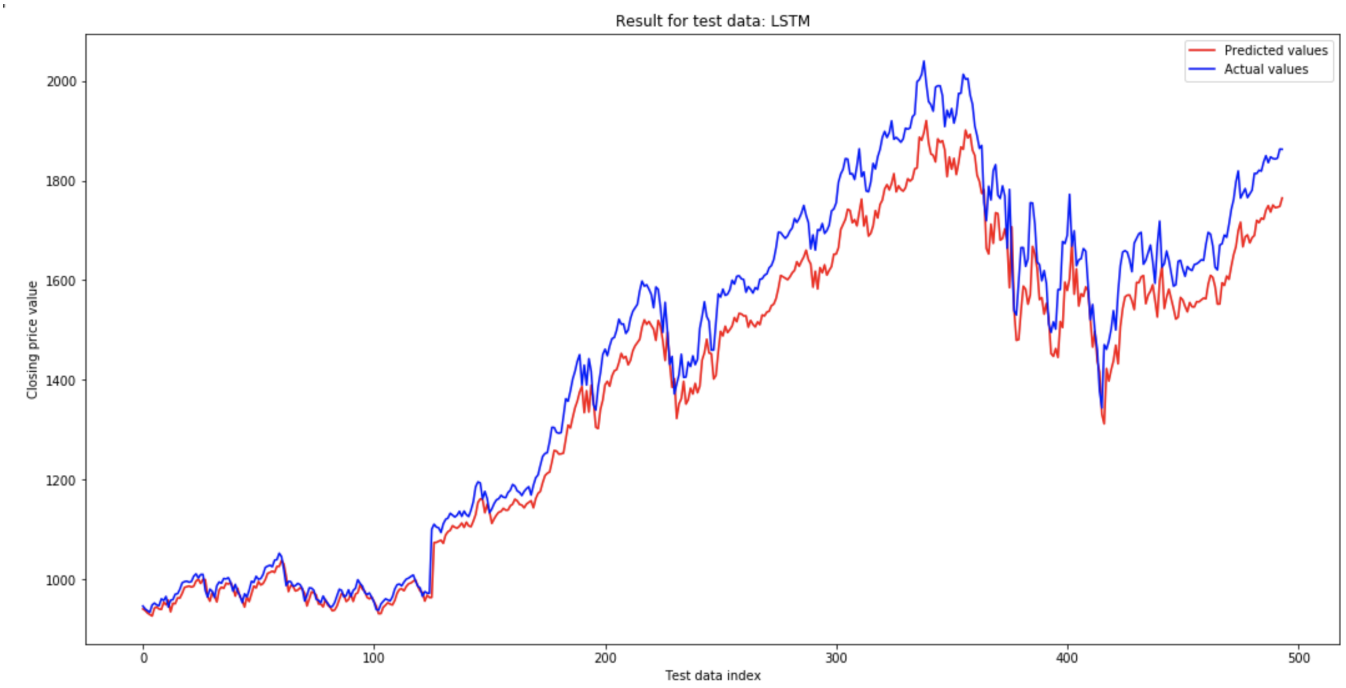

Figure 3.7 Result for LSTM test data

3.2.3 Multi Layer Perceptron model

A Multilayer Perceptron is a class of feed-forward artificial neural network. It is the most basic form of Neural network in which each layer is connected to only the layer immediately following it. The architecture used in this project is :

```
[ ]  # define model
     model = Sequential()
     model.add(Dense(100, activation='relu', input_dim=(n_steps)))
     model.add(Dense(50, activation='relu'))
     model.add(Dense(25, activation='relu'))
     model.add(Dense(1))
     model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

Figure 3.8 MLP architecture used

3.2.3.1 Result

Figure 3.9 Result for MLP test data

3.3 Calculating Error and Uncertainty

In order to measure the performance of the models, we have used the Root Mean Squared Error as one of the metrics. A lower value of RMSE indicates that the model provides more accurate predictions and is a better fit.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Figure 3.10 Formula used for calculating RMSE

In the above formula, y(hat) represents the predicted value of stock prices, y represents the actual stock prices and n is the total number of observations being considered.
Confidence Interval is used to find an interval of probable values instead of a specific value for a certain measurement according to a confidence level or Confidence. This confidence value is a probability that the value of the measurement will be in that interval. The higher the confidence value or probability, the larger the predicted range.
To take into consideration the trend in the graph of the closing values, the uncertainty of a day's value was calculated by taking into consideration only the values around that day. We use a 95% confidence level, which has a z-score of 1.96 used in the formula below.

$$Uncertainty = \hat{y} \pm \left(1.96 * \sqrt{\sigma / n}\right)$$

Figure 3.11 Formula used for calculating uncertainty

In the above formula, Sigma represents the standard deviation of the values around that day, n represents the number of these values being considered and y(hat) represents the predicted value.
Another performance metric used to evaluate the models is the Mean Uncertainty Error percentage which was calculated by taking the mean of all the uncertainty values for each value and finding the corresponding percentage.

3.4 Result- Best Prediction Model

Based on the metrics explained above, the values of RMSE and Mean Error Percentage were calculated for the three models as shown below:

S.No	Model	Mean Uncertainty Error Percent	RMSE
1.	CNN	1.40%	0.055
2.	Multilayer Perceptron	1.20%	0.039
3.	LSTM	5.20%	0.073

Figure 3.12 RMSE and mean uncertainty error percent of each model

As can be seen from the above results, the Multilayer Perceptron model or MLP model gives the lowest error uncertainty and hence the most accurate results.
We will now use this model to visualise its uncertainty using different plots to help extract different kinds of data.

3.5 Uncertainty Visualization

Visualizing uncertainty has many practical uses and makes your predictions more safe and reliable.

3.5.1 Uncertainty visualization - Is it easy?

No, uncertainty as a concept is very difficult to grasp and imagine even if the possibility of it's existence being crucial to the final result is very high. The factors that affect uncertainty differ from problem to problem so there is no general way to measure it. The difficulty arises because :-
• The effective visualization of uncertainty, however, is not always possible through the simple application of traditional visualization techniques.
• Adding uncertainly to a visualization complicates the display for the human eye.
• The sources of uncertainty can sometimes be unknown and difficult to capture.

3.5.2 Sources of uncertainty

Uncertainty is different and due to different causes for every situation. But it can be classified into these two types:
• Aleatoric Uncertainty - It is regarded as statistical uncertainty, and is representative of unknowns that differ each time we run the same experiment. For example, the value of stocks for any company depends on real life events that affect the company (like a big decision made by a rival company). These events don't occur periodically and can be pretty random.
• Epistemic Uncertainty - Epistemic uncertainty is also known as systematic uncertainty, and is due to

things one could explain but isn't able to in practice. The model's accuracy plays a huge role. The dataset and how clean and trustworthy it is are also some causes. Eg. Value of gravitational acceleration on earths surface is equal to 9.8 m/s2. But when we take the height of the object, air density, etc which we tend to neglect, the uncertainty rises and can affect its value.

3.5.3 Plots

3.5.3.1 Scatter Plot

A colour coded scatter plot helps us to conveniently compare the amount of uncertainty for different values of x and y. Here, for each value of x, the position of the corresponding circle indicates the y-

Figure 3.13 Example of Scatter plot

value i.e. the prediction. The colour and size of the circle indicate the amount of uncertainty, higher the uncertainty, bigger the circle and darker the colour. It does not help with interpreting the exact value of the calculated error.

```
[ ]  plt.figure(figsize = (18,9))
     l = plt.scatter(x=np.asarray(dff.index), y=np.asarray(dff['N']),s = 8*np.asarray(dff['A']).astype(int)  ,c = cl, linewidths=2, edgecolors='black')
     plt.ylabel("Closing price value")
     plt.title("Scatter plot to visualize uncertainty")
     plt.xlabel("Test data index")
     plt.show()
```

Figure 3.14 Code to generate scatter plot

3.5.3.2 Error Bars

Error bars are plotted using the upper limit and lower limit of the probable values of the variable. It is used to display the variability of any measurement.

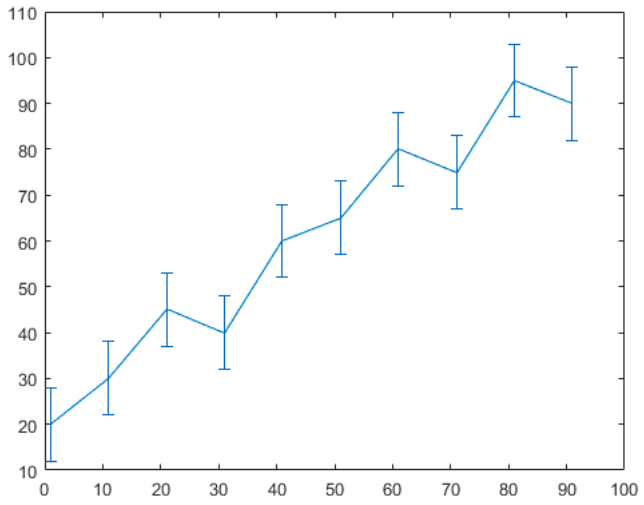

Figure 3.15 Example of Error Bar plot

Error bars can also be used to compare the uncertainty of different x values in some cases, but it is not as efficient as the scatter plot. It is best suited when you one needs to get the exact value of error for a x value from the graph.

```
[ ]  import numpy as np
     import matplotlib.pyplot as plt


     plt.figure(figsize = (25,10))
     x = np.arange(yhat1.shape[0])
     y = yhat1
     yerr = np.asarray(dff['A'])  # 1.96 = 95% confidence

     plt.errorbar(x, y, yerr=yerr, fmt='o', ecolor='black', elinewidth=2, capsize=2 , capthick=2, markersize = 8, markerEdgeColor='black', markerFaceColor='mediumspringgreen')
     plt.title("Error bar plot to visualize uncertainty")
     plt.xlabel("Test data index")
     plt.ylabel("Closing price value")
     plt.show()
```

Figure 3.16 Code to generate error bar plot

3.5.3.3 Line Plot

A combination of three line plots helps us get a good visualization of the upper limits and lower limits of the probable values of the measurement.

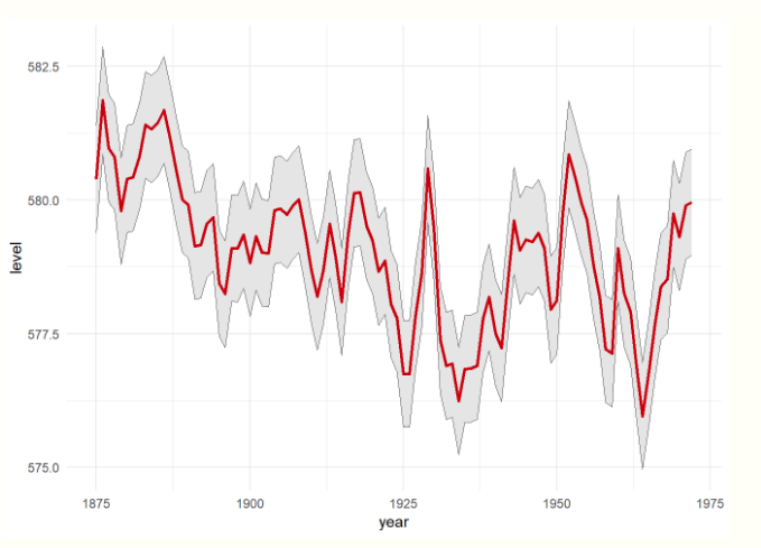

Figure 3.17 Example of Line plot

The line plot however, is not too efficient in comparing the uncertainties across the x-axis. It does give a good idea of the exact values.

```
[ ]  plt.figure(figsize = (30,12))
     line1, = plt.plot(range(yhat1.shape[0]), yhat1, linewidth = 1.5, color = 'red')
     line2, = plt.plot(range(yhat1.shape[0]), (np.squeeze(yhat1)+errors), linewidth = 1.5, color = 'blue')
     line3, = plt.plot(range(yhat1.shape[0]), (np.squeeze(yhat1)-errors), linewidth = 1.5, color = 'green')
     plt.ylabel("Closing price value")
     plt.title("Line plot to visualize uncertainty")
     plt.xlabel("Test data index")
     plt.legend(handles=[line1,line2,line3],labels=('Predicted values','Upper Limit','Lower Limit'),loc="upper left")
     plt.show()
```

Figure 3.18 Code to generate line plot

Chapter *4*

Final Result

We used the above discussed data visualisation methods to visualise uncertainty or error in the model that gave the best accuracy in the prediction .i.e. the Multilayer Perceptron model.
The following plots were created to visualise uncertainty and extract different information as mentioned above:
1. Line plot:

Figure 4.1 Line plot visualisation

In the above plot, the blue line represents the upper limit and the green line represents the lower limit of the closing stock prices which are plus-minus the uncertainty percent of the predicted stock price, represented by the red line e.g. If the uncertainty is 10% then the upper limit and lower limits will be 10% above and below the predicted line. This gives the user a good estimate of the probable stock price while also accommodating for uncertainty.

2. Scatter plot:

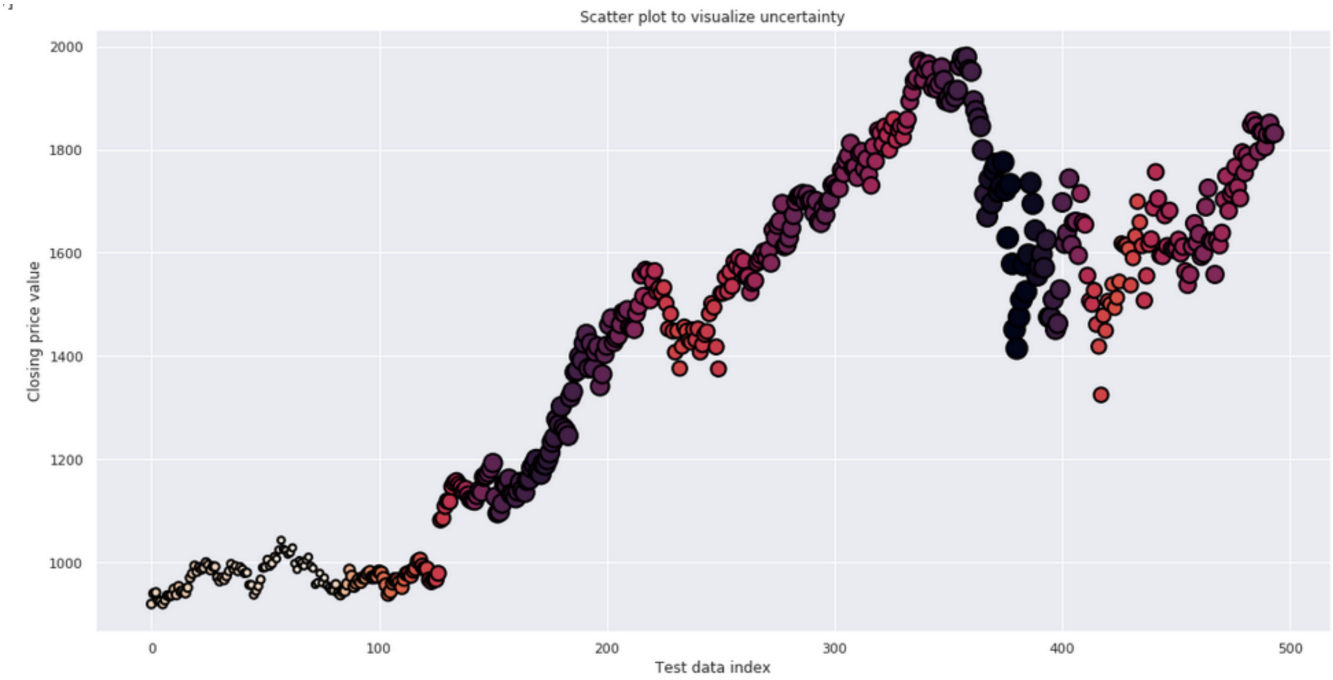

Figure 4.2 Scatter plot visualisation

In the above plot, the colour and the size of the circles or points represents the uncertainty- the almost black points, have the most uncertainty, while the orange to almost white points have least uncertainty. We cannot use this graph to get information about the exact predicted price, but it is easily readable, and very useful for comparing one data point with its neighbouring points. This is also a convenient tool to compare different models.

3. Error bar plot:

Figure 4.3 Error bar plot visualisation

In the above plot, similar to line plot, the circles represent the predicted values, while the bars extend above and below the point to represent the upper and lower limits of the predicted stock prices. However, unlike the line plot, the error bar plot is not a continuous graph and has discrete points, allowing it to be more readable and give more exact values of the prices.

Chapter *5*

Conclusions and Future Work

In this project, we used different neural-network based models to predict the closing prices of the stock values of Amazon, in an attempt to identify which model would be most suitable for this use case. We experimented with several models and approaches and chose the described three, due to their properties, diversity and performance. The uncertainty of the different models were calculated using certain experimented techniques and it was visualized to help select the most appropriate model. As was recognized by various plots and error values, we observed that the Multilayer Perceptron model gave the best results. This can be attributed to the property of the model to learn on the go, while not presuming any parameters or weights beforehand and the capability to accommodate to improbable events. Finally, we used different kinds of plots to visualize the uncertainty which can help us deduce different kinds of information about the data. According to the requirement of the problem, we can choose the most appropriate visualization.

5.1 Scope of further work

Our models were designed keeping in mind the specific project of which it is a part of therefore there is a lot of scope to extend the functionality and increase experimentation, to create a more generalised model fit for more use cases. For example:
1. Our current system and findings were based on the Amazon data set. Other data sets can be experimented with to see if similar results are attained.
2. Our primary focus was to visualise the uncertainty of a model and not on creating an optimal prediction model. More complex or tuned neural networks can be designed to obtain better prediction values.
3. Our project focused on three uncertainty visualisation methods. More visualisation techniques can be explored to help extract more information.
4. Our project is only a starting step in this domain. In the long run, this can be polished to be implemented in the the real world.

Bibliography

[1] Calculation of uncertainty referred from https://www.sciencedirect.com/topics/engineering/model-uncertainty.

[2] Creation of error bar retrieved from https://www.mathworks.com/help/matlab/ref/errorbar.html.

[3] Creation of various scatter plots retrieved from https://towardsdatascience.com/everything-you-need-to-know-about-scatter-plots-for-data-visualisation-924144c0bc5.

[4] Data set retrieved from https://in.finance.yahoo.com/quote/amzn/history?p=amzn.

[1] [3] [2] [4]