-Kriti Garg (22118038)

REPORT
STOCK SENTIMENT ANALYSIS USING MACHINE LEARNING (from Twitter Data)

PROBLEM STATEMENT:
"Stock Price Prediction Using Twitter Sentiment Analysis" a method for predicting stock prices is developed using news articles. The changes in stock prices of a company, the rises and falls, are correlated with the public opinions being expressed in tweets about that company.

- Stock price prediction is one of the most important topic to be investigated in academic and financial researches.
- Various **Data mining techniques** are frequently involved in the studies. To solve this problem.
- But technique using machine learning/deep learning will give more accurate, precise and simple way to solve such issues related to stock and market prices.

Libraries used: Pandas, matplotlib.pyplot, numpy,nltk, sklearn

Processes Involved:

1. **Data Collection**:Tweets on **Microsoft, Google, AAPL**, are extracted .The tweets will have collected using **yFinance API** and filtered using keywords like $ MSFT, # Microsoft, #Windows etc.

```
!pip install yfinance
!pip install GetOldTweets3
!pip install treeinterpreter
```

2. : **Data Pre-Processing** :Stock prices data collected is not complete understandably because of weekends and public holidays when the stock market does not function. The missing data is approximated using a simple technique.
Stock data usually follows a **concave function**. So, if the stock value on a day is x and the next value present is y with some missing in between. The first missing value is approximated to be (y+x)/2 and the same method is followed to fill all the gaps.

```
def clean_tweets(self, tweets, is_bytes = False):
    test_tweet_list = []
    for tweet in tweets:
        if is_bytes:
            test_tweet_list.append(self.get_cleaned_text(ast.literal_eval(tweet).decode("UTF-8")))
        else:
            test_tweet_list.append(self.get_cleaned_text(tweet))
    return test_tweet_list

def clean_single_tweet(self, tweet, is_bytes = False):
    if is_bytes:
        return self.get_cleaned_text(ast.literal_eval(tweet).decode("UTF-8"))
    return self.get_cleaned_text(tweet)
```

3. **Tokenization:** Tweets are split into individual words based on the space and irrelevant symbols like emoticons are removed.

```
import nltk
nltk.download('vader_lexicon')
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('punkt')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import unicodedata
sentiment_i_a = SentimentIntensityAnalyzer()

from nltk.corpus import subjectivity
from nltk.sentiment import SentimentAnalyzer
from nltk.sentiment.util import *
```

4. **Stop word Removal:** Words that do not express any emotion are called Stop words. After splitting a tweet, words like a, is, the, with etc. are removed from the list of words.

5. **Sentiment Analysis:** Sentiment analysis task is very much field specific. Tweets are classified as **positive, negative and neutral** based on the sentiment present. Out of the total tweets are examined by humans and **annotated** as **1 for Positive, 0 for Neutral and 2 for Negative emotions.** For classification of nonhuman annotated tweets, a machine learning model is trained whose features are extracted from the human annotated tweets.

6.**Model Training**:The features extracted for the tweets are fed to the classifier and trained using classification methods like **Logistic Regression, Decision Tree, SVM** .to estimate the movement of the change in stock market price vs the volume as well as sentiment of news articles and tweets. Apply **Linear Regression to find relation between the change in stock market price vs the volume as well as sentiment of news articles and tweet**s.

```
from sklearn.model_selection import train_test_split
from treeinterpreter import treeinterpreter as ti
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor

from sklearn import svm
from sklearn.svm import SVR

from sklearn.metrics import mean_squared_error
from math import sqrt
```