APPENDIX 1

TITLE OF PROJECT

END TERM REPORT

*by*

NAME OF THE CANDIDATE(S)

Macha Sumanth Yadav

Yashwant Pratap Singh

Veesam Satya Teja

(Section: K18MS)

(Roll Number(s): 1,2,3)

A picture containing orange, drawing

Description automatically generated

Department of Intelligent Systems

School of Computer Science Engineering

Lovely Professional University, Jalandhar

April 2020

APPENDIX 2

Student Declaration

This is to declare that this report has been written by us. No part of the report is copied from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be copied, we are shall take full responsibility for it.

Macha Sumanth Yadav

Roll No. :- 01

Reg No. :- 11814751

Yashwant Pratap Singh

Roll No. :-02

Reg No. :- 11815738

Veesam Satya Teja

Roll No.: - 03

Reg No. :- 11807455

**Place :- Jalandhar**

**Date :-**

APPENDIX 4

BONAFIDE CERTIFICATE

Certified that this project report “Predict Stock Prices”is the bonafide work of “Macha Sumanth Yadav,Yashwant Pratap Singh & Veesam Satya Teja” who carried out the project work under my supervision.

Jasleen Kaur

Assistant Professor

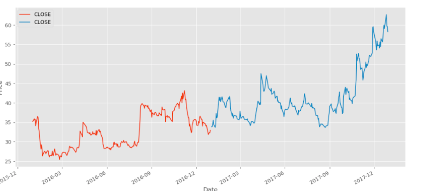
25340

Department of Intelligent Systems

PREDICTING STOCK PRICES

First, you have many types of data The stock market is like candy-land for any data scientists who are even remotely interested in finance. Hat you can choose from. You can find prices, fundamentals, global macroeconomic indicators, volatility indices, etc… the list goes on and on. Second, the data can be very granular. You can easily get time series data by day (or even minute) for each company, which allows you think creatively about trading strategies. Finally, the financial markets generally have short feedback cycles. Therefore, you can quickly validate your predictions on new data. Here is the equity data of stocks listed on *NSE*over 2016 and 2017.

Here is the plot of reports from NSE



## ****Steps we need to follow to build a model:****

1. Pre-Processing
2. Time series Analysis
3. Classification
4. Prediction

import pandas as pd  
import numpy as np

from datetime import datetime  
#to plot within notebook  
import matplotlib.pyplot as plt  
from matplotlib import style  
style.use('ggplot')  
##%matplotlib inline

#for normalizing data  
from sklearn.preprocessing import MinMaxScaler  
scaler = MinMaxScaler(feature\_range=(0, 1))  
from sklearn.feature\_extraction import DictVectorizer  
from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import

LinearRegression

from sklearn.metrics import recall\_score,

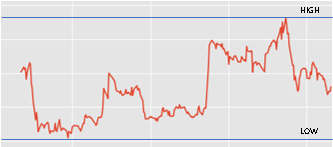
precision\_score  
from mlxtend.plotting

import plot\_decision\_regions

To begin with data processing, feature and target selection is important. In NSE data the given features are [ *‘OPEN’, ‘HIGH’, ‘LOW’, ‘TOTTRDQTY’, ‘Date’, ‘PREVCLOSE’, ‘TOTTRDVAL’, ‘TOTALTRADES’ ] and the labels are their corresponding [ ‘CLOSE’ ] values.*

#Read data  
stocks = pd.read\_csv('20microns.csv')  
print(stocks.head())  
#New Dataset  
stocks = stocks[[‘OPEN’, ‘HIGH’, ‘LOW’, ‘CLOSE’, ‘TOTTRDQTY’, ‘Date’, ‘PREVCLOSE’, ‘TOTTRDVAL’, ‘TOTALTRADES’]]

For dimensionality reduction, I took common feature from High and Low values which is



stocks with global max & min

HL\_PCT = ( [ HIGH — LOW ] / LOW ) \* 100

and replaced both HIGH and LOW features with HL\_PCT.

stocks[‘HL\_PCT’] = (stocks[‘HIGH’] — stocks[‘LOW’]) / stocks[‘LOW’] \* 100.0  
stocks = stocks[[‘OPEN’, ‘HL\_PCT’, ‘CLOSE’, ‘TOTTRDQTY’, ‘Date’, ‘PREVCLOSE’, ‘TOTTRDVAL’, ‘TOTALTRADES’]]

Next I did a time series analysis to separate test from train data.

#Time Series Analysis  
start16 = datetime(2016, 1, 1)  
end16 = datetime(2016, 12, 31)  
stamp16 = pd.date\_range(start16, end16)start17 = datetime(2017, 1, 1)  
end17 = datetime(2017, 12, 31)  
stamp17 = pd.date\_range(start17, end17)stocks['Date'] = pd.to\_datetime(stocks.TIMESTAMP,format='%Y-%m-%d')  
stocks.index = stocks['Date']

All the 2016 reports are placed in train dataset and 2017 reports are placed in test dataset.

train = []  
test = []  
for index, rows in stocks.iterrows():  
 if index in stamp16:  
 train.append(list(rows))  
 if index in stamp17:  
 test.append(list(rows))train = pd.DataFrame(train, columns = stocks.columns)  
test = pd.DataFrame(test, columns = stocks.columns)

Next step is to convert pandas Dataframe to Numpy array

#Pre-Processing the Train Data   
X\_train = train[['HL\_PCT', 'OPEN', 'TOTTRDQTY', 'TOTTRDVAL', 'TOTALTRADES']]  
x\_train = X\_train.to\_dict(orient='records')  
vec = DictVectorizer()  
X = vec.fit\_transform(x\_train).toarray()  
Y = np.asarray(train.CLOSE)  
Y = Y.astype('int')#Pre-Processing Test data  
X\_test = test[['HL\_PCT', 'OPEN', 'TOTTRDQTY', 'TOTTRDVAL', 'TOTALTRADES']]  
x\_test = X\_test.to\_dict(orient='records')  
vec = DictVectorizer()  
x = vec.fit\_transform(x\_test).toarray()  
y = np.asarray(test.CLOSE)  
y = y.astype('int')

We completed our processing part.

Let’s move to building our Machine Learning - regression model to predict those Closing values using training features. TheilSen Regressor is one of the best regression classifier for our data.

#Classifier  
from sklearn.linear\_model import TheilSenRegressor  
clf = TheilSenRegressor()  
clf.fit(X, Y)   
print("Accuracy of this Statistical Arbitrage model is: ",clf.score(x,y))  
predict = clf.predict(x)  
test['predict'] = predict

Accuracy score of our model is 98.2% which is decent for a yearly predictions. To plot the prediction results,

#Ploting   
train.index = train.Date  
test.index = test.Date  
train['CLOSE'].plot()  
test['CLOSE'].plot()  
test['predict'].plot()  
plt.legend(loc='best')  
plt.xlabel('Date')  
plt.ylabel('Price')  
plt.show()

A picture containing text, whiteboard

Description automatically generated

Accuracy score of our model is 98.2% which is decent for a yearly predictions. To plot the prediction results,

#Ploting   
train.index = train.Date  
test.index = test.Date  
train['CLOSE'].plot()  
test['CLOSE'].plot()  
test['predict'].plot()  
plt.legend(loc='best')  
plt.xlabel('Date')  
plt.ylabel('Price')  
plt.show()

The graph plotted between stock values and date along with prediction over 2017 dataset :

A picture containing text, whiteboard

Description automatically generated