Computer Engineering Department
CMPE-255 | Data Mining | Professor David C. Anastasiu

Final Project Evaluation

STOCK PRICE PREDICTION USING HISTORICAL DATA AND TWITTER SENTIMENTS

Team 9

Manika Kapoor (011539203)
Masi Nazarian (011822044)
Sowmya Gowrishankar (005699733)

Fall 2017

Performance

Overall, ANN was the winner algorithm in terms of MSE performance, followed by ARIMA and SVR. Logistic Regression also performed relatively well, followed by FB Prophet in the list.

Table 1 shows the Mean-Squared Error values, comparing the performance of different algorithms on price forecasting of different cryptocurrencies.

	ARIMA	FB Prophet	ANN	SVR	LR
AAPL	1.493	1639.284	0.638	2.065	1.430
GOOGL	48.482	11647.023	43.5441	83.290	44.347
INTU	1.241	79.222	0.357	1.804	1.177
IBM	1.544	1143.301	3.77034	1.779	1.667
DWDP	0.265	19.465	0.160506	0.334	0.249
TOT	0.139	4.837	0.00597506	0.132	0.208

Table 1. The MSE of different algorithms on different stock data

Artificial Neural Network Graph

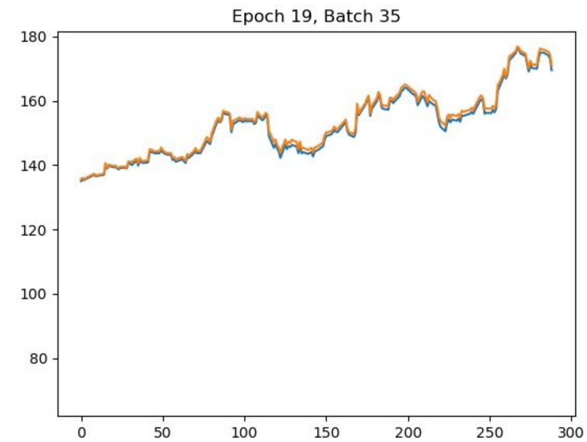
For ANN, the best results were achieved using 5 hidden layers with relu activation function. Figures below shows number of nodes in each layer, and the layer architecture.

```
# Model architecture parameters
numberOfFeatures = X_train.shape[1]
layer_nodes_1 = 1024
layer_nodes_2 = 512
layer_nodes_3 = 256
layer_nodes_4 = 128
layer_nodes_5 = 64
```

ANN number of nodes in each layer

```
# Define all the hidden layers
layer_1 = tf.nn.relu(tf.add(tf.matmul(X, weight_layer_1), bias_layer_1))
layer_2 = tf.nn.relu(tf.add(tf.matmul(layer_1, weight_layer_2), bias_layer_2))
layer_3 = tf.nn.relu(tf.add(tf.matmul(layer_2, weight_layer_3), bias_layer_3))
layer_4 = tf.nn.relu(tf.add(tf.matmul(layer_3, weight_layer_4), bias_layer_4))
layer_5 = tf.nn.relu(tf.add(tf.matmul(layer_4, weight_layer_5), bias_layer_5))
```

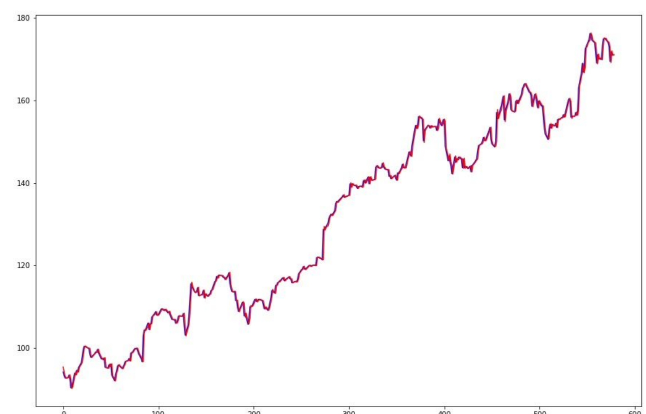
ANN layer architecture



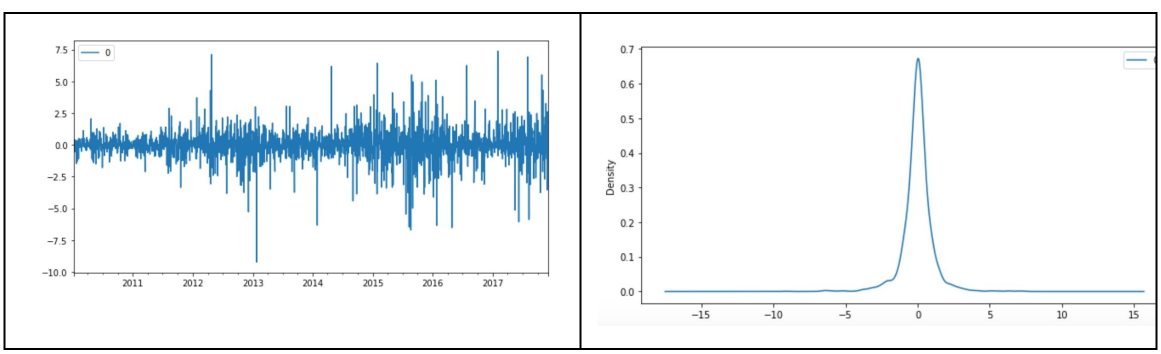
ANN prediction results for AAPL

ARIMA Graphs

```
model = ARIMA(history, order=(5,1,0))
model_fit = model.fit(disp=0)
output = model_fit.forecast()
```



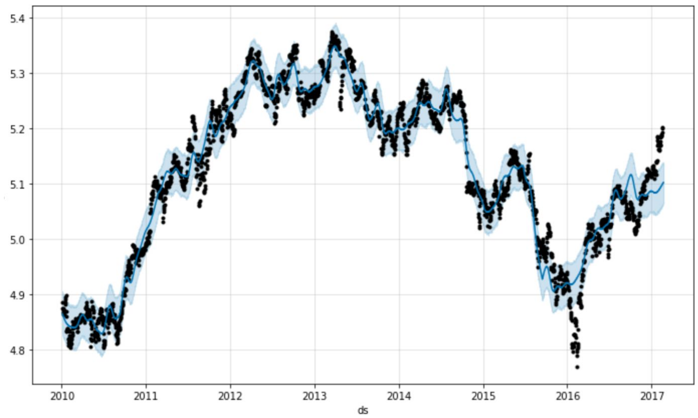
ARIMA prediction results for AAPL



The residual graphs produced by ARIMA

Facebook Prophet Graph

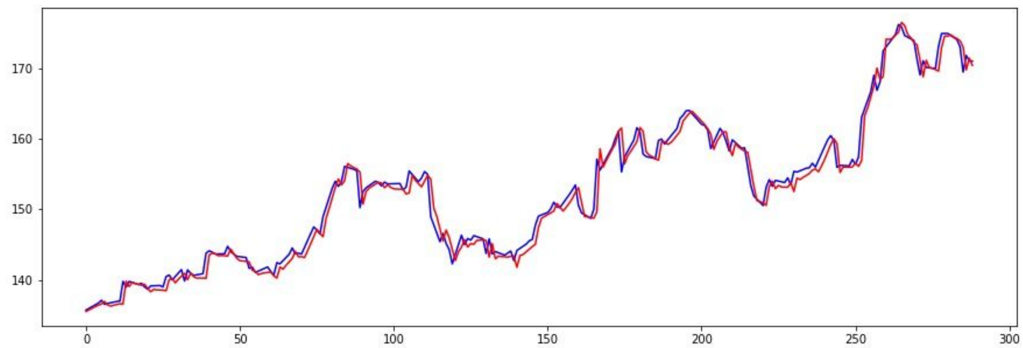
```
dataProphetRed = dataProphet.rename(columns={"index": "ds", "Close": "y"})
dataProphetRed['y_orig'] = dataProphetRed['y']

#log transform y
dataProphetRed['y'] = np.log(dataProphetRed['y'])

splitIndex = int(np.floor(dataProphetRed.shape[0]*0.95))
X_train_prophet, X_test_prophet = dataProphetRed[:splitIndex], dataProphetRed[splitIndex:]

model=Prophet(daily_seasonality=True)
model.fit(X_train_prophet)

future_data = model.make_future_dataframe(periods=30)
forecast_data = model.predict(future_data)
```



FB Prophet forecast for AAPL

SVR Graph

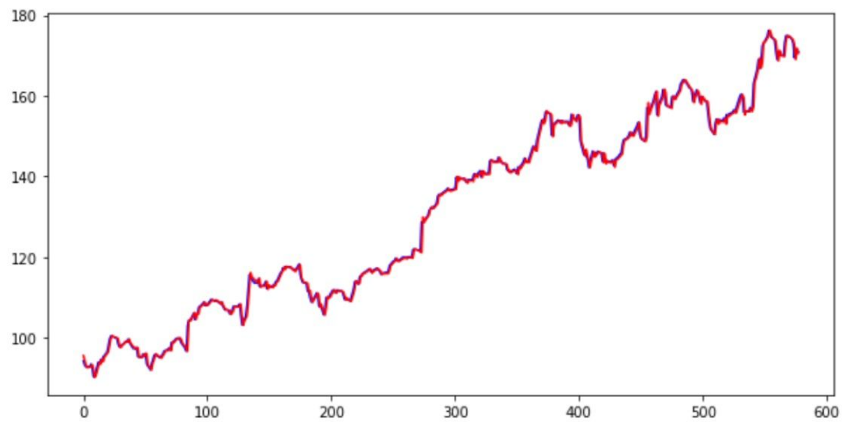
```
svm_linear = SVR(kernel='linear', C=2500)
```



SVR prediction results for AAPL

Linear Regression Graph

```
linearReg = LinearRegression(normalize=False)
linearReg.fit(X_train_transform, y_train_transform)
predictions = linearReg.predict(X_test_transform)
```



Linear Regression prediction results for AAPL

Stock Price Comparison with Twitter Sentiments

The tweets specific to 15 companies were collected for a month and then the python text analysis package, TextBlob was used to get the average polarity of the tweets. This average polarity is compared against the change in the stock price to see whether the twitter sentiments can be used as a feature to predict stock price.

It was observed that generally when the twitter sentiment was positive, there was a growth in stock price and vice-versa. The data cannot be used as a feature for stock price prediction as the twitter dataset was very small (only one month). So, we used it for inference.