Stock Price Prediction

Using Machine Learning models

Contents

- Problem Definition
- Data selection
- Exploratory analysis
- Feature Engineering
- KNN regression
- Decision tree regression
- Random forest regression
- SVM regression
- ARIMA model
- Summary

Problem Definition

• Stock market prediction and analysis are some of the most difficult jobs to complete. There are numerous causes for this, including market volatility and a variety of other dependent and independent variables that influence the value of a certain stock in the market. These variables make it extremely difficult for any stock market expert to anticipate the rise and fall of the market with great precision.

• However, with the introduction of Machine Learning and its strong algorithms, the most recent market research and Stock Market Prediction advancements have begun to include such approaches in analyzing stock market data.

•Machine Learning Algorithms are widely utilized by many organizations in Stock market prediction.

• Given the dataset with necessary parameters, upon feeding a new record we can predict the same through training the data with the appropriate machine learning algorithms.

Data selection

- Source: Dataset provided by Nasdaq's Historical Data
- Dataset Info: Top 10 Company's stock performance for past 10 years (2012-2022)
- List of companies: Amazon, AMD, Apple, Cisco, Meta, Microsoft, Netflix, Qualcomm, Starbucks, Tesla
- Total Number of rows: 2518
- Features Information:

Column Name	Datatype	Description
Close/Last	Float64	Closing price of Stock in a day
Volume	Int64	Total volume of stocks traded in a day
Open	Float64	Opening price of Stock in a day
High	Float64	Maximum share price reached in a day
Low	Float64	Minimum share price reached in a day

Exploratory analysis

- Null values: No null values were found in any of the columns
- Frequency: Day-wise;1132 non-working days are not included in the dataset
- Skewness: mostly positively skewed with outliers on the right end of the distribution indicating a rapid increase in price
- Distribution: Non-gaussian , generally multimodal distribution
- Pattern: Non-linear trend observed. No seasonality or cyclicity.
- Stationary? : Data is not stationary.

	Close	Volume	Open	High	Low
count	2518.000000	2.518000e+03	2518.000000	2518.000000	2518.000000
mean	61.229098	1.758535e+08	61.188815	61.879588	60.523350
std	48.277790	1.325883e+08	48.239987	48.874580	47.626046
min	13.950000	4.099995e+07	13.856100	14.271400	13.753600
25%	26.680000	9.296638e+07	26.665625	26.912500	26.395625
50%	40.220000	1.315068e+08	40.106250	40.554350	39.775000
75%	81.060000	2.085897e+08	80.878750	81.380000	80.162800
max	182.010000	1.457835e+09	182.630000	182.940000	179.120000

Summary statistics for Apple

Feature Engineering

- The Attributes High, Low, and Open are strongly correlated with the Close attribute. Hence they can be removed.
- The index is split into 3 attributes - day, month, and year to explore hidden patterns in the data and for resampling
- Volume attribute is log transformed to reduce the range and yield smaller values

Inference

- Resampling by month, we can infer that the stock price peaked in the spring-autumn season (August-October)
- Resampling by year, we can infer that there is a strong non-linear upward trend

KNN regression

MODEL USED :
from sklearn.neighbors import KNeighborsRegressor
INDEPENDENT VARIABLES:
7 parameters – Lag 1 , Lag 2 ,Lag 3 , Lag 4 ,
Lag 5 , Lag 6 and Lag 7 values
TEST SIZE :
The last 30 records from sample is taken for the test set
MODEL PARAMETERS:
N_neighbors => obtained via the GridSearchCV function
Which will return the optimised k value for each datatset

ERROR METRICS AND ONE-STEP FORECAST VALUES

	mse	mape	one-step forecast
Amazon	37.645840	0.041255	94.181429
AMD	20.372318	0.061313	76.954583
Apple	17.029862	0.022520	150.450000
Cisco	0.823653	0.016705	48.160000
Meta	140.626277	0.070396	112.347500
Microsoft	70.921567	0.027729	245.735714
Netflix	246.371372	0.046724	290.989600
Qualcomm	22.720673	0.033579	125.982500
Starbucks	10.449370	0.026904	98.020000
Tesla	210.803692	0.054612	198.866429

Decision tree regression

MODEL USED :
from sklearn.tree import DecisionTreeRegressor
MODEL PARAMETERS:
7 parameters – Lag 1 , Lag 2 ,Lag 3 , Lag 4 ,
Lag 5 , Lag 6 and Lag 7 values
TEST SIZE :
The last 30 records from sample is taken for the test set

ERROR METRICS AND ONE-STEP FORECAST VALUES

	mse	mape	one-step forecast
Amazon	29.720107	0.040724	93.08
AMD	72.712090	0.085019	69.50
Apple	24.539307	0.024484	150.02
Cisco	1.986807	0.025806	47.24
Meta	57.353713	0.037937	109.64
Microsoft	67.898913	0.026003	249.90
Netflix	167.495260	0.039131	288.59
Qualcomm	18.674473	0.031467	127.46
Starbucks	6.679680	0.023353	100.11
Tesla	181.285020	0.053361	197.79

Random forest regression

MODEL USED :
from sklearn.ensemble import RandomForestRegressor
INDEPENDENT VARIABLES:
7 parameters – Lag 1 , Lag 2 ,Lag 3 , Lag 4 ,
Lag 5 , Lag 6 and Lag 7 values
TEST SIZE :
The last 30 records from sample is taken for the test set
MODEL PARAMETERS:
N_estimators =100 (default)

ERROR METRICS AND ONE-STEP FORECAST VALUES

	mse	mape	one-step forecast
Amazon	18.472317	0.034199	93.5984
AMD	12.256803	0.040765	75.2949
Apple	19.350572	0.022927	149.4738
Cisco	0.676612	0.014555	47.9525
Meta	53.474874	0.035250	111.6307
Microsoft	43.233431	0.019695	247.4543
Netflix	118.792680	0.029304	286.2680
Qualcomm	16.045193	0.027952	123.8612
Starbucks	4.789583	0.017242	99.4303
Tesla	122.659639	0.040943	193.9471

SVM regression

MODEL USED :
from sklearn.svm import SVR
INDEPENDENT VARIABLES:
7 parameters – Lag 1 , Lag 2 ,Lag 3 , Lag 4 , Lag 5 , Lag 6 and Lag 7 values
TARGET VARIABLE:
1 parameter: Close
TEST SIZE :
The last 30 records from sample is taken for the test set
MODEL PARAMETERS:
Kernel=linear

ERROR METRICS AND ONE-STEP FORECAST VALUES

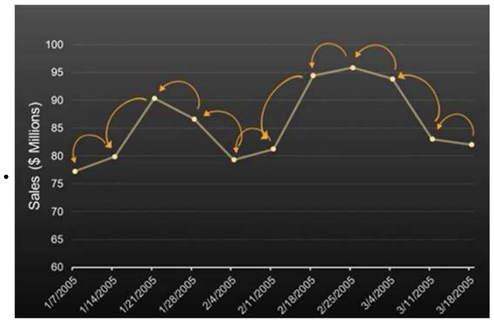
Company	MSE	MAE	R2 Score	One Step Forecast
Amazon	10.342230	171.356126	-0.524298	93.634674
AMD	4.833576	46.631366	0.061813	75.122035
Apple	7.370814	80.140053	-2.043974	148.920603
Cisco	1.894602	5.353249	-0.228144	48.509604
Meta	15.708168	463.791296	-1.096858	112.178676
Microsoft	12.533114	234.367941	-1.781537	249.261797
Netflix	30.069666	1239.832659	-2.305183	286.361297
Qualcomm	7.589718	82.348610	-1.555131	123.113220
Starbucks	3.693892	24.536437	0.157487	99.691707
Tesla	15.524446	384.297579	-0.130367	182.379342

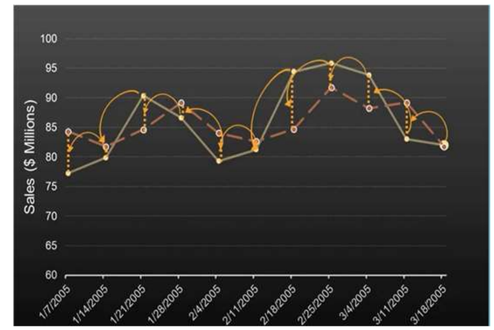
ARIMA Model

TERMS:

AR = autocorrelation(y_t comparing with yt-1 or yt-2..

MA = MOVING AVERAGE(y_t comparing with et-1or et-2..
(residual autocorelation)





WHY ARIMA in Stock price predicition?

ARIMA(auto Regressive Integrated Moving Average) is popular statistical model used in time series forecasting. ARIMA models are particularly used in forecasting stock prices because they are often influenced by past trends and patterns

CONDITIONS:

1) The time series data should be stationary
2) No trend or patterns should exist
3) ACF and pacf should decay exponentially

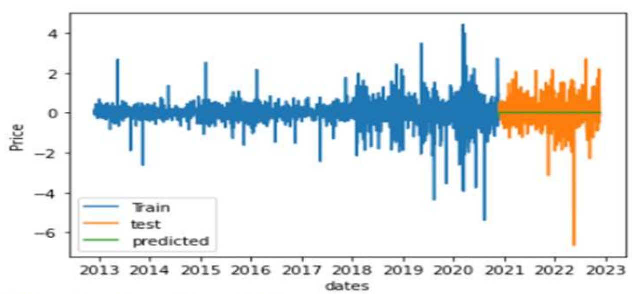
ARIMA Model

ERROR METRICS USED:

1. AIC(**Akaike's Information Criterion**) and BIC (**The Bayesian Information Criterion**)
2. MSE and Mape

CONCLUSION based on o/p:

AIC is less in cisco compared with rest company so arima
Fits good in cisco and error values is less

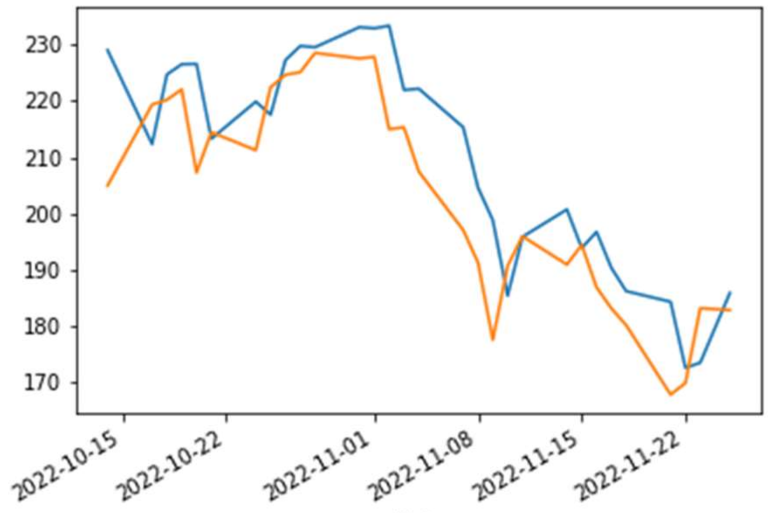



```
MAE 0.5506053129825531
MSE 0.6013821235335393
rmse 0.775488312957416
residual error -5.314511815418092e-05
```

	(1,1,1)	MSE(1,1,1)	MSE(5,1,0)
Amazon -(5,1,0)->10909.341	, 10520.101,	11.17677359515917	11.1505738
AMD -(5,1,0)->10231.028	, 9837.176	11.160798576761824	11.2294120
Apple -(5,1,0)->9802.419	,9414.979	8.098060453787479	8.975897
cisco -(5,1,0)->5285.489	,4868.836	0.6013821235335393	0.676479
meta -(5,1,0)-> 14885.638	,14436.574	51.8512578862905	53.8409642
microsoft -(5,1,0)-> 12674.490	,12234.598	23.12867488820738	23.21189
Netflix -(5,1,0)-> 18044.888	,17622.587	164.05368797507828	163.770975
Qualcomm -(5,1,0)->11385.310	,10968.451	14.078612705793349	14.110953
Starbucks -(5,1,0)->8422.909	, 7957.548	3.070390992845652	3.09749
Tesla -(5,1,0)->15517.896	, 15122.419	107.28741278353793	166.825

LSTM

MODEL USED :
from keras.models import Sequential
from keras.layers import Dense, LSTM
MODEL PARAMETERS:
Input layer –1,Sequential
Hidden layers – 2; 256 node LSTM layer; 128 node LSTM layer
Output layers- 2 ; 25 node Dense layer followed but 1 output node
Optimizer-Adam
INDEPENDENT VARIABLES:
7 parameters – Lag 1 , Lag 2 ,Lag 3 , Lag 4 , Lag 5 , Lag 6
and Lag 7 values
TARGET VARIABLE:
1 parameter: Close
TEST SIZE :
The last 30 records from sample is taken for the test set
ERROR METRICS: MSE:115 , MAPE:0.04



Conclusion

- The best performing model to predict the prices of stocks of various companies is the SVR(linear Kernel) model.It has been chosen based on the mean_squared_error and mean_absolute_perentage_error metrics . Though the SVR model proved to be the most time consuming out of all the models

- An important note to consider : The LSTM (Long Short Term Memory) model seems promising . The error values decrease significantly the more layers / more nodes are added to the model , though it is accompanied by an increase in runtime and complexity .

- Given enough time and resource , the LSTM model may prove to be better than the SVR model when it comes to predicting the stock prices

Thank You!