Stock Market Prediction

Applied Machine Learning-363

Instructor: Dr Madan Gopal

Team members:
Shobhit Tiwari - 1710110314
Utsav Raj|-1710110365

Submission Date:
05th April

INDEX

1.Introduction:

Predicting how the stock market will perform is one of the most difficult things to do. There are so many factors involved in the prediction – physical factors vs. Physiological, rational and irrational behaviour, etc. All these aspects combine to make share prices volatile and very difficult to predict with a high degree of accuracy. Machine learning techniques have the potential to unearth patterns and insights we didn't see before, and these can be used to make unerringly accurate predictions.

Broadly, stock market analysis is divided into two parts – Fundamental Analysis and Technical Analysis.

1) Fundamental Analysis involves analysing the company's future profitability on the basis of its current business environment and financial performance.

2) Technical Analysis includes reading the charts and using statistical figures to identify the trends in the stock market.

We will keep our focus on the technical analysis as it is fundamental

Analysis is not data related but more inference Analysis involving market not measurable quantities like feeling of consumer, government policies and etc.

2.Problem statement:

- The problem at hand is to predict the stock market prices in the Indian stock market
- The problem at various places on internet is mentioned as "Time-Series Forecasting". During our study we came across many methods to tackle this problem both Traditional Machine Learning Methods as well as Neural Networks but in out project we will use only Traditional Machine Learning Methods only.
- Our codes are implemented in Python Language using jupyter as IDE.

Explanation:

Time series forecasting:

The stock price predictor is a time series forecasting question

Predictions are made for new data when the actual outcome may not be known until some future date. The future is being predicted, but all prior observations are almost always treated equally. Perhaps with some very minor temporal dynamics to overcome the idea of "concept drift" such as only using the last year of observations rather than all data available.

A time series dataset is different.

Time series adds an explicit order dependence between observations: a time dimension.

This additional dimension is both a constraint and a structure that provides a source of additional information

Making predictions about the future is called extrapolation in the classical statistical handling of time series data.

Forecasting involves taking models fit on historical data and using them to predict future observations.

Descriptive models can borrow for the future (i.e. to smooth or remove noise), they only seek to best describe the data.

An important distinction in forecasting is that the future is completely unavailable and must only be estimated from what has already happened.

Time series analysis provides a body of techniques to better understand a data set.

Perhaps the most useful of these is the decomposition of a time series into 4 constituent parts:

1) Level. The baseline value for the series if it were a straight line.

2) Trend. The optional and often linear increasing or decreasing behaviour of the series over time.

3) Seasonality. The optional repeating patterns or cycles of behaviour over time.

4) Noise. The optional variability in the observations that cannot be explained by the model

Another important thing to note is that the market is closed on weekends and public holidays so it is not a continuous timer series

3.Algorithms Used
1.SVR Model

Abstract

In this model, support vector regression (SVR) analysis is used as a machine learning technique in order to predict the highest value of the target stock. In this model we compared the predicted values with our test sets.

Introduction

The main characteristic of SVR is that instead of minimizing the observed training error, SVR attempts the generalized error bound so as to achieve generalized performance. This generalization error bound is the combination of the training error and a regularization term that controls the complexity of the hypothesis space.

Theory

To evaluate the result from the SVR models is used to measure the error rate.

$$\text{MAPE} = \frac{\sum \frac{|A-F|}{A} \times 100}{N}$$

Here MAPE stands for mean average percentage error between actual prices(A) and predicted prices(P) and N is the number of days.

2.Linear Regression

Introduction

Linear Regression is a linear approach to modelling the relationship between a scalar response and one or more explanatory variables. The case of one explanatory variable is called simple linear regression. For more than one explanatory variable the process is called multiple regression.

Theory

Y= a+bX

Here X is the explanatory variable and Y is the dependent variable. The slope of the line is b and a is the intercept.

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

The first step in finding a linear regression equation is to determine if there is a relationship between the two variables. This is often a judgment call for the researcher. You'll also need a list of your data in x-y format (i.e. two columns of data—independent and dependent variables).

Assumptions:

When we use a linear regression model, we are implicitly making some assumptions about the variables in equation.

First, we assume that the model is a reasonable approximation to reality; that is, the relationship between the forecast variable and the predictor variables satisfies this linear equation.

Second, we make the following assumptions about the errors (ε1,...,εT)(ε1,…,εT):

- They have mean zero; otherwise the forecasts will be systematically biased.
- They are not autocorrelated; otherwise the forecasts will be inefficient, as there is more information in the data that can be exploited.
- They are unrelated to the predictor variables; otherwise there would be more information that should be included in the systematic part of the model.

It is also useful to have the errors being normally distributed with a constant variance σ2σ2 in order to easily produce prediction intervals.

Another important assumption in the linear regression model is that each predictor xx is not a random variable. If we were performing a controlled experiment in a laboratory, we could control the values of each xx (so they would not be random) and observe the resulting values of y. With observational data (including most data in business and economics), it is not possible to control the value of xx, we simply observe it. Hence we make this an assumption.

3.Random Forest Regressor

Introduction

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap Aggregation, commonly known as bagging .Random Forest is an improvement of Bagging ensemble learning method. It uses a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. This process is sometimes called "feature bagging".

Theory;

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favourably to Adaboost , but are more robust with respect to noise. Internal estimates monitor error, strength, and correlation and these are used to show the response to increasing the number of features used in the splitting. Internal estimates are also used to measure variable importance.

ExtraTrees

Adding one further step of randomization yields extremely randomized trees, or ExtraTrees. While similar to ordinary random forests in that they are an ensemble of individual trees, there are two main differences: first, each tree is trained using the whole learning sample (rather than a bootstrap sample), and second, the top-down splitting in the tree learner is randomized. Instead of computing the locally optimal cut-point for each feature under consideration (based on, e.g., information gain or the Gini impurity), a random cut-point is selected. This value is selected from a uniform distribution within the feature's empirical range (in the tree's training set). Then, of all the randomly generated splits, the split that yields the highest score is chosen to split the node. Similar to ordinary random forests, the number of randomly selected features to be considered at each node can be specified.

4. Flowchart of the Solution:

Model Designing:

1)Different libraries are imported

2)The data set file is loaded

3)The data is normalized

4)The profit or loss calculation is usually determined by the closing price of a stock for the day, hence we will consider the closing price as the target variable(regressor)

5)Setting date as index

6)Data is sorted in ascending order(date)

7)Another data set is created so that new features created by us don't change the original data set.

8)Then we identify the weekdays and set them as '1' whereas weekend as '0'. This is a new feature .We can add more features according to us.

9)The data is now divided for training, validation (

(already done till here-midterm)

Optimized by running the program for different datasets divisions

10) Then we implement linear regression algorithm

11) Then we implement support vector machine algorithm

12) Then we implement random forest algorithm

11) The predicted values and test values given are plotted on a graph having time on x-axis and stock price on y-axis (This algorithm can be adjusted to make it a long short term memory which will increase the accuracy of the model)

12) The Test values and predicted values are then evaluated at each point and it is found if the test predicted values are increasing or decreasing trend at that point.

13) The total no of points of predicted value trend following the test values trend are calculated

14) Then the percentage is calculated of the total point how many actually follow and how much of the prediction was correct

15) This allows the user of the said modelling design to predict the trend of a stock of company

16) We also found the test and predicted values accuracy

17) We also found the training and predicted values accuracy

The difference in training and testing accuracy is very high clearing showing overfitting of data in the above machine learning models.

5.Python Code

```
import numpy as np

import pandas as pd

df=pd.read_csv("/Users/shobh/ML_Project/NSE-TATAGLOBAL.csv")

rows = df.values.tolist()


from sklearn.model_selection import train_test_split

x_train = []

y_train = []

x_test = []

y_test = []

X = []

Y = []

for row in rows:

    X.append(int(''.join(row[0].split('-'))))

    Y.append(row[5])
```

```
max1=max(Y)
min1=min(Y)
for i in range(0,len(Y)):
    Y[i]=(Y[i]-max1)/(max1-min1)

x_train=X[:800]
x_test=X[800:]
y_train=Y[:800]
y_test=Y[800:]

x_train = np.array(x_train)
y_train = np.array(y_train)
x_test = np.array(x_test)
y_test = np.array(y_test)

x_train = x_train.reshape(-1,1)
x_test = x_test.reshape(-1,1)

from sklearn.svm import SVR
clf_svr = SVR(kernel='rbf', C=1e3, gamma=0.1)
```

```
clf_svr.fit(x_train,y_train)

y_pred_svr = clf_svr.predict(x_test)


from sklearn.linear_model import LinearRegression

clf_lr = LinearRegression()

clf_lr.fit(x_train,y_train)

y_pred_lr = clf_lr.predict(x_test)


from sklearn.ensemble import RandomForestRegressor

clf_rf = RandomForestRegressor(n_estimators=100)

clf_rf.fit(x_train,y_train)

y_pred_rf = clf_rf.predict(x_test)


x=({'test':y_test,'rf_val_pred':y_pred_rf,'svr_val_pred':y_pred_svr,'lr_val
_pred':y_pred_lr})

l=len(x['test'])



test_val=[];

rf_pred_val=[];

svr_pred_val=[];
```

```
lr_pred_val=[];
for i in range (l-1):
        if i==0:
            test_val.append('z')
        else:
            if x['test'][i+1]>=x['test'][i]:
                test_val.append(1)
            else:
                test_val.append(0)

for i in range (l-1):
        if i==0:
            rf_pred_val.append('z')
        else:
            if x['rf_val_pred'][i+1]>=x['rf_val_pred'][i]:
                rf_pred_val.append(1)
            else:
                rf_pred_val.append(0)

for i in range (l-1):
```

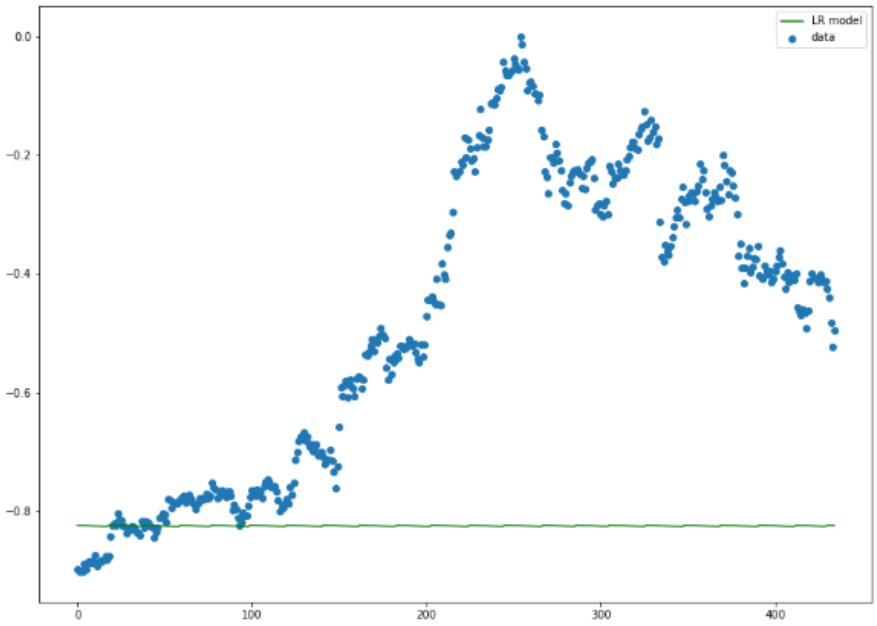
```
        if i==0:
            svr_pred_val.append('z')
        else:
            if x['svr_val_pred'][i+1]>=x['svr_val_pred'][i]:
                svr_pred_val.append(1)
            else:
                svr_pred_val.append(0)



for i in range (l-1):
        if i==0:
            lr_pred_val.append('z')
        else:
            if x['lr_val_pred'][i+1]>=x['lr_val_pred'][i]:
                lr_pred_val.append(1)
            else:
                lr_pred_val.append(0)

p=0
for i in range (l-1):
```

```
    if test_val[i]==rf_pred_val[i]:
        p+=1


n=0
for i in range (l-1):
    if test_val[i]==svr_pred_val[i]:
        n+=1


q=0
for i in range(l-1):
    if test_val[i]==lr_pred_val[i]:
        q+=1


print(l,p,n,q)
print("Trend percentage for Random Forest:",(p/l)*100)
print("Trend percentage for SVR: ",(n/l)*100)
print("Trend percentage for Linear Regression :",(q/l)*100)



import matplotlib.pyplot as plt
```

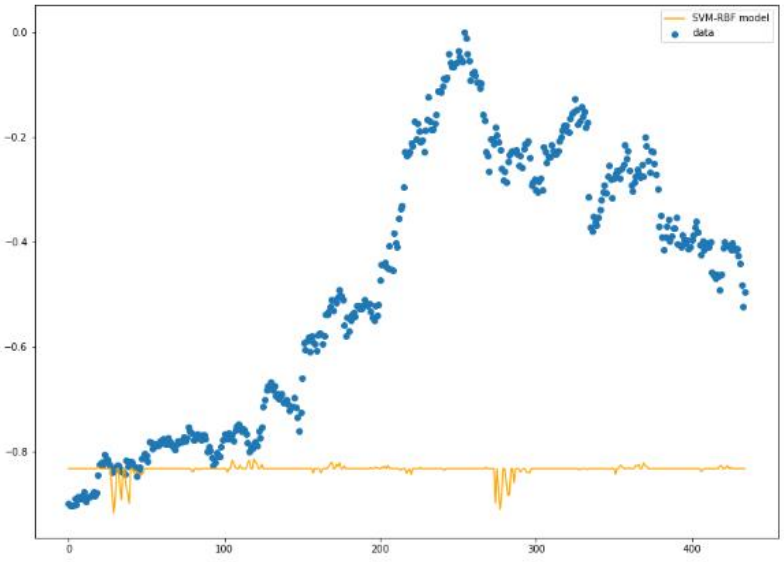
```
f,(ax1,ax2) = plt.subplots(1,2,figsize=(30,10))

# Linear Regression
ax1.scatter(range(len(y_test)),y_test,label='data')
ax1.plot(range(len(y_test)),y_pred_lr,color='green',label='LR model')
ax1.legend()

# Support Vector Machine
ax2.scatter(range(len(y_test)),y_test,label='data')
ax2.plot(range(len(y_test)),y_pred_svr,color='orange',label='SVM-RBF model')
ax2.legend()

f1,(ax3,ax4) = plt.subplots(1,2,figsize=(30,10))

# Random Forest Regressor
ax3.scatter(range(len(y_test)),y_test,label='data')
ax3.plot(range(len(y_test)),y_pred_rf,color='red',label='RF model')
ax3.legend()
```

```
print("Accuracy of Linear Regerssion
Model:",clf_lr.score(x_test,y_test))

print("Accuracy of SVM-RBF Model:",clf_svr.score(x_test,y_test))

print("Accuracy of Random Forest Model:",clf_rf.score(x_test,y_test))


print("Accuracy of Linear Regerssion
Model:",clf_lr.score(x_train,y_train))

print("Accuracy of SVM-RBF Model:",clf_svr.score(x_train,y_train))

print("Accuracy of Random Forest
Model:",clf_rf.score(x_train,y_train))
```

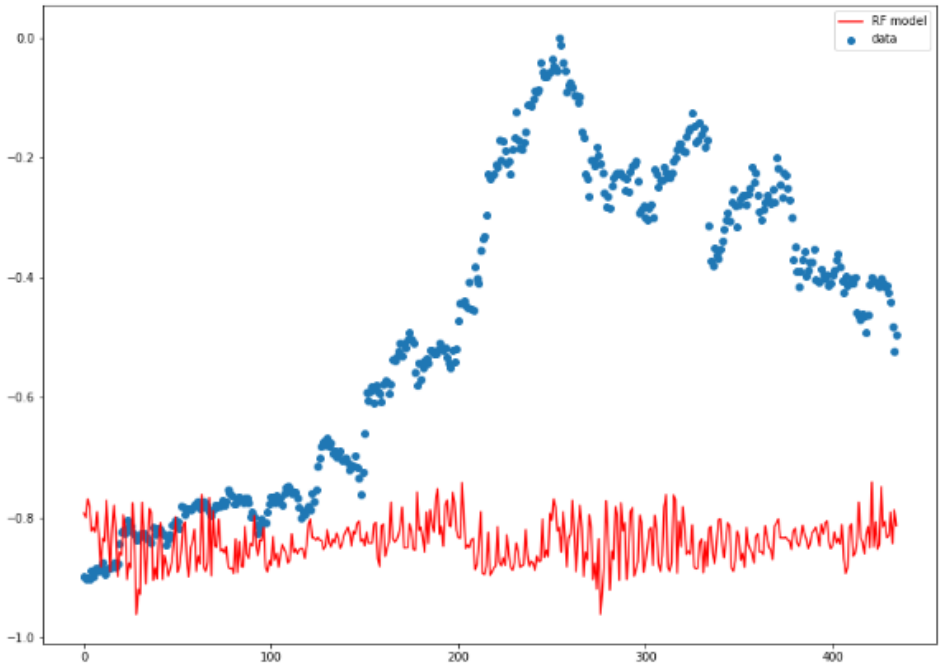
6.Graphs



Graph for the Linear Regression Model



Graph for the SVR model



Graph for the Random Forest Model

The graphs do not predict the market prices as we are using traditional machine learning methods. So we will analyse the market trend that is whether the prices falls or rises and compare it with our model.

7.Observation:

```
Trend percentage for Random Forest: 49.6551724137931
Trend percentage for SVR:  50.804597701149426
Trend percentage for Linear Regression : 48.275862068965516
```

```
Accuracy of Linear Regerssion Model for test data: -1.7897959899297353
Accuracy of SVM-RBF Model for test data: -1.8748120313217678
Accuracy of Random Forest Model for test data: -1.9844039706703707
```

```
Accuracy of Linear Regerssion Model for train data: 0.00011849609823744167
Accuracy of SVM-RBF Model for train data: 0.07552241400294402
Accuracy of Random Forest Model for train data : 0.667341136893099
```

8.Result:

We are able to predict the trend with approximately 50% for all the three algorithms.

Linear regression is a simple technique and quite easy to interpret, but there are a few obvious disadvantages. One problem in using regression algorithms is that the model overfits to the date and month column. Instead of taking into account the previous values from the point of prediction, the model will consider the value from the same date a month ago, or the same date/month a year ago. Results is that your model is **overfitting**. You can tell that from the large difference in accuracy between the test(-1.7) and train(0.001) accuracy.

Overfitting means that it learned rules specifically for the train set, those rules do not generalize well beyond the train set.

SVM suffers from the sample problem as it use long memory data to predict values and as stock market rarely depends on the long memory data the values predicted are highly inaccurate.

Random forest also has the same problem

Hence neural networks are being used today as they are able to predict non linear data much better then robust learning.

Due to the non linear property of stock market prices the above models are very inefficient.

Hence linear regression might be the best machine learning algorithm but it not even near the best for time series prediction.

Machine learning for time series data has following observation across all forecasting technique:

1.)You need to retrain your model every time you want to generate a new prediction:
2)The uncertainty of the forecast is just as important as, or even more so, than the forecast itself:
3)One more thing that distinguishes forecasting from other supervised learning tasks is that your forecasts are almost always going to be wrong.

4)They take this idea a step further in Financial Time series modelling, where they actually have classes of models built explicitly for modeling the uncertainty of a time series, as opposed to the time series itself

9.Some Advancement available:

1.)Auto ARIMA

Introduction

ARIMA is a very popular statistical method for time series forecasting. ARIMA models take into account the past values to predict the future values. There are three important parameters in ARIMA:

p (past values used for forecasting the next value)q (past forecast errors used to predict the future values)

d (order of differencing)

Parameter tuning for ARIMA consumes a lot of time. So we will use auto ARIMA which automatically selects the best combination of (p,q,d) that provides the least error

2.)Prophet

Introduction

There are a number of time series techniques that can be implemented on the stock prediction dataset, but most of these techniques require a lot of data preprocessing before fitting the model. Prophet, designed and pioneered by Facebook, is a time series forecasting library that requires no data preprocessing and is extremely simple to implement. The

input for Prophet is a data frame with two columns: date and target (ds and y).

Prophet tries to capture the seasonality in the past data and works well when the dataset is large

3.)LSTMS

Introduction

LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is because LSTM is able to store past information that is important, and forget the information that is not. LSTM has three gates:

The input gate: The input gate adds information to the cell state

The forget gate: It removes the information that is no longer required by the model

The output gate: Output Gate at LSTM selects the information to be shown as output

10.Bibliography

https://medium.com/@randerson112358/predict-stock-prices-using-python-machine-learning-53aa024da20a

https://towardsdatascience.com/time-series-machine-learning-regression-framework-9ea339290090a

https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6

https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-python/

https://machinelearningmastery.com/normalize-standardize-time-series-data-python/

https://towardsdatascience.com/3-facts-about-time-series-forecasting-that-surprise-experienced-machine-learning-practitioners-69c18ee89387