Machine Learning in Finance Groupwork Assignment
Submission 3: Modeling and Strategy Development

Solution Design

Date: 20 Nov 2018

Project Instructions

Using what you have learnt from Submissions 1 and 2, implement a trading strategy using machine learning. We recommend that students focus on classification – for example: trying to forecast if a stock will move up and down, above some threshold such as the 90-day standard deviation.

1. Decide on an algorithm or group of algorithms (for example, ensemble techniques).
2. Fit the model.
3. Show that it works out of sample, and use appropriate cross-validation techniques.
4. Provide the following performance metrics: (a) ROC curves, (b) Confusion Matrix, (c) Precision, Recall, F1-Score, Accuracy, and AUC.
5. Analysis of metrics and report.

Fund factsheet

Create a fund factsheet for your new investment strategy. Have a look at examples of popular funds found online and create a fact sheet with all the bells and whistles. It must at a minimum include (Pyfolio can be used):

1. Maximum Drawdown
2. Annualized Returns
3. Sharpe Ratio
4. Plot the Equity Curve

Python Libraries Used

The following libraries are used and can be installed using pip, e.g. “pip install pandas”:

fix-yahoo-finance 0.0.22 – see https://pypi.org/project/fix-yahoo-finance/
pandas 0.23.4 – see https://pandas.pydata.org/pandas-docs/stable/
numpy 1.15.0 – see https://docs.scipy.org/doc/numpy/reference/
backtrader 1.9.67.122 – see https://www.backtrader.com/docu/index.html
pyfolio 0.9.0 – see https://github.com/quantopian/pyfolio
matplotlib 3.0.0 – see https://matplotlib.org/contents.html
seaborn 0.9.0 – see https://seaborn.pydata.org/whatsnew.html
pandas-datareader 0.7.0 – see https://pandas-datareader.readthedocs.io/en/latest/
keras 2.2.4 – see https://keras.io/
scikit-learn 0.20.0 – see https://scikit-learn.org/

Solution Design

Step 1: Data Preparation and Processing

We use McDonald’s stock price data for 20 years from 1998-11-10 to 2018-11-10. Data are downloaded from yahoo finance API using python, we extracted information like Open, High, Low, Close, Adj Close and Volume.

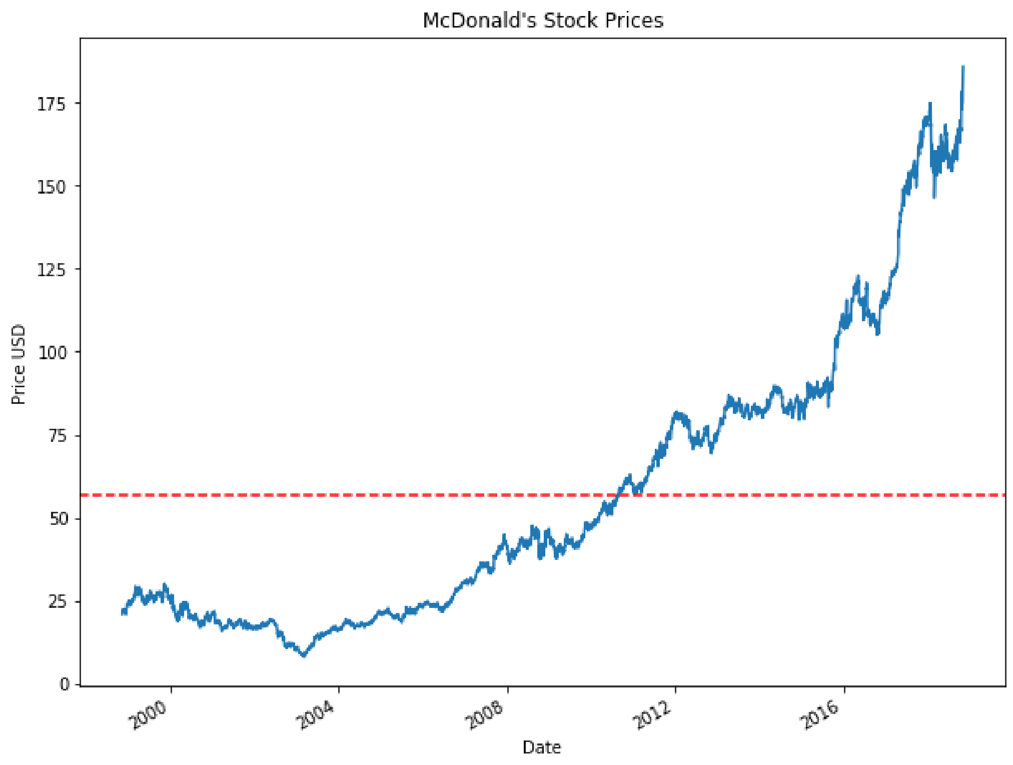


Figure 1: Daily Stock prices

Step 2: Data Labeling for Classification Model

We decided to detect the stock price movement and base our trading strategy on that movement for buy or sell. We defined the output value as price rise which is a binary variable storing 1 when the closing price of tomorrow is greater than the closing price of today. Otherwise the value will be stored as 0.

This is simple labeling method than meta-labeling with triple barrier method which immediately incorporate the trading strategy to labeling. We will consider this method for our further trading strategy improvement.

Step 3: Feature Generation

We examined the features selection and run two algorithms viz the Artificial Neural Network (ANN) and Long Short-Term Memory (LSTM). We get the highest accuracy and lowest loss after several testing for both algorithms. The features that we used are technical indicators as listed below:

- Simple Moving Average (SMA)
- Exponential Moving Average (EMA)
- Moving Average Convergent Divergent (MACD)
- Momentum (MOM)
- Rate of Change (ROC)
- Relative Strength Index (RSI)
- Time Series Differentiated features (diff_10, diff_30, time-t, time-t+1)

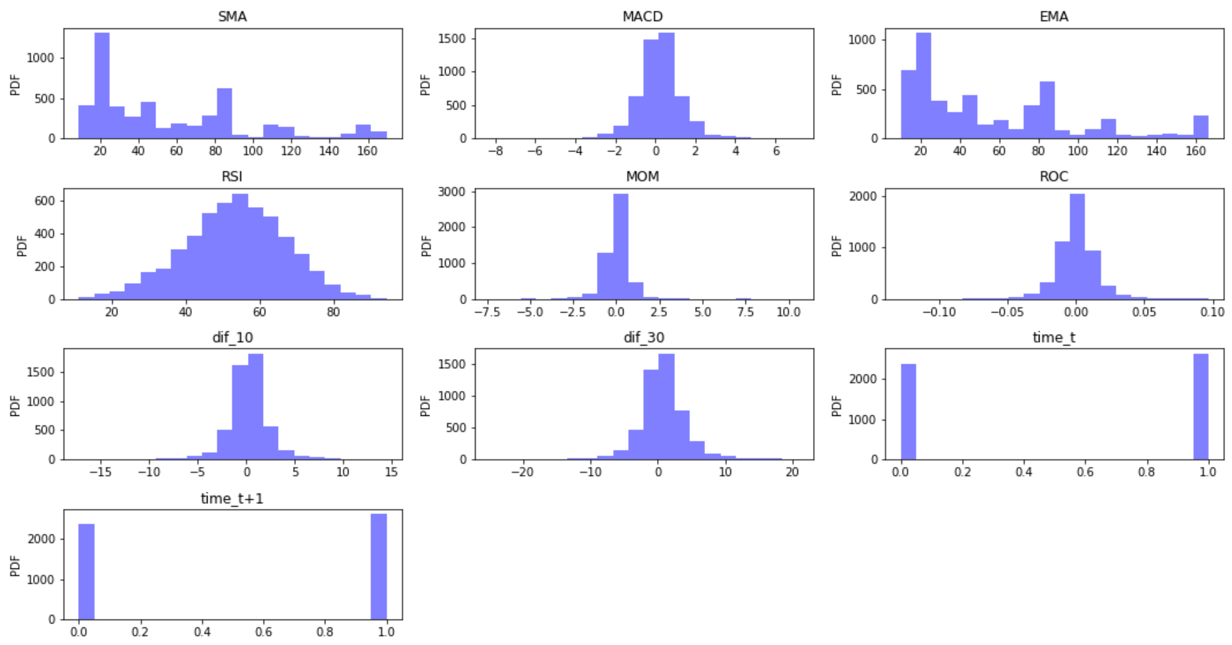


Figure 2: Features Distribution

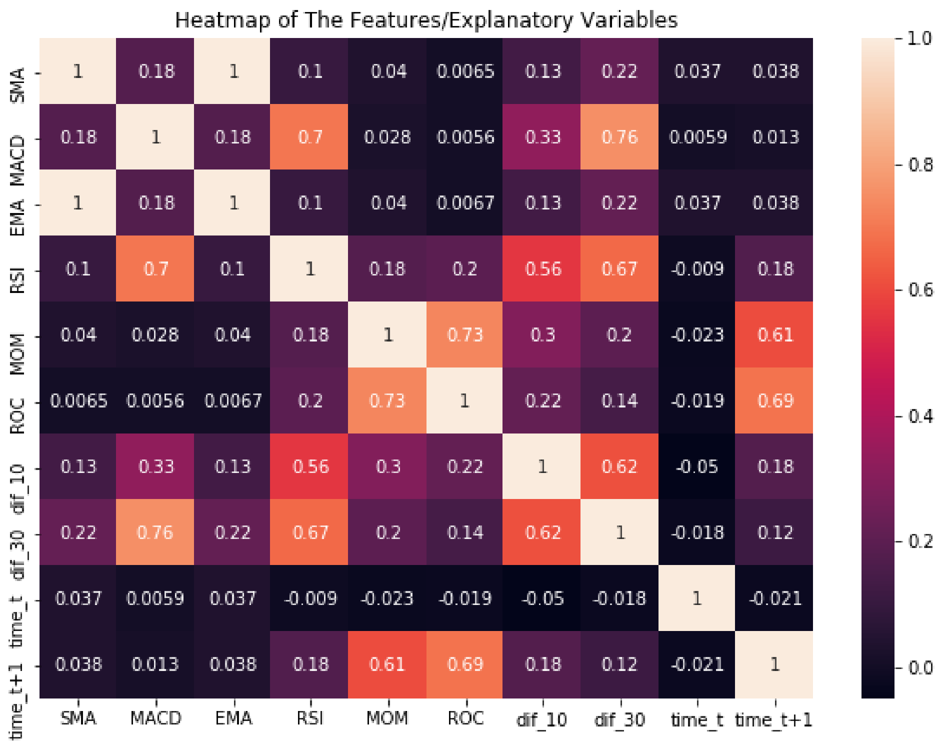


Figure 3: Features Correlation Heatmap

As we can see, EMA and SMA have the highest correlation. The next highly correlated pairs are MACD with dif_30 and MACD with RSI. We will see later if we should eliminate those features that have high correlation for better model performance.

Step 4: Apply Classification Model

We have selected four machine learning models in the process to pick the best or the most suitable algorithm for our research. The models are feed-forward Artificial Neural Network (ANN), Long Short-Term Memory (LSTM), Support Vector Machine (SVM) and Naive Bayes. The features or explanatory variables are then fitted into each of the models and performance metrics for all of them are calculated. We will then pick the model with the highest F1 score and use the signals predicted from the model to run our backtesting trades.

We referred to the paper of Sreelekshmy Selvin, Vinayakumar R, Gopalakrishnan E.A, Vijay Krishna Menon, Soman K.P (2017) *"STOCK PRICE PREDICTION USING LSTM, RNN AND CNN-SLIDING WINDOW MODEL"* [2] for the ideas of picking ANN and LSTM for our research. The paper highlighted the performance of each model. The author used the quantifier method as percentage error and used three deep learning algorithm CNN, RNN and LSTM. The author slided the window to the size of 100 minutes with an overlap of 90 minutes from period of July-2014 to Jun-2015. The result showed that CNN performs better than the other two. However, ANN and CNN have some limitations in learning the patterns because stock market data has tremendous noise and complex dimensionality.

Long short-term memory (LSTM), a kind of Recurrent Neural Networks (RNN) has been researched and regarded as a better model than ANN for time series prediction as it cares about the time order and can decide which information should be forgotten and which inputs are used as features. For both feed forward ANN and LSTM, we used 'Adam' optimizer, 'Mean Squared Error' for loss and 'accuracy' for metrics. SVM and Naïve Bayes have also been used. They showed good performance in Jigar Patel paper (2014) *'Predicting stock and stock price index movement using Trend Deterministic Data Preparation and Machine learning'* [4]

Step 5: Out of sample Test and Cross Validation Techniques

We performed cross-validation for all the algorithms used by using TimeSeriesSplit. TimeSeriesSplit is the popular cross validation for time series. As we need to keep the sequence of time series, we cannot split randomly. Instead, the algorithm works like the steps below [8]:

1. Folds for time series cross validation are created in a forward chaining fashion.
2. Suppose we have a time series for yearly consumer demand for a product during a period of n years.

We split the data into training set and validation set. We performed training on training set while leave out the test set for prediction.

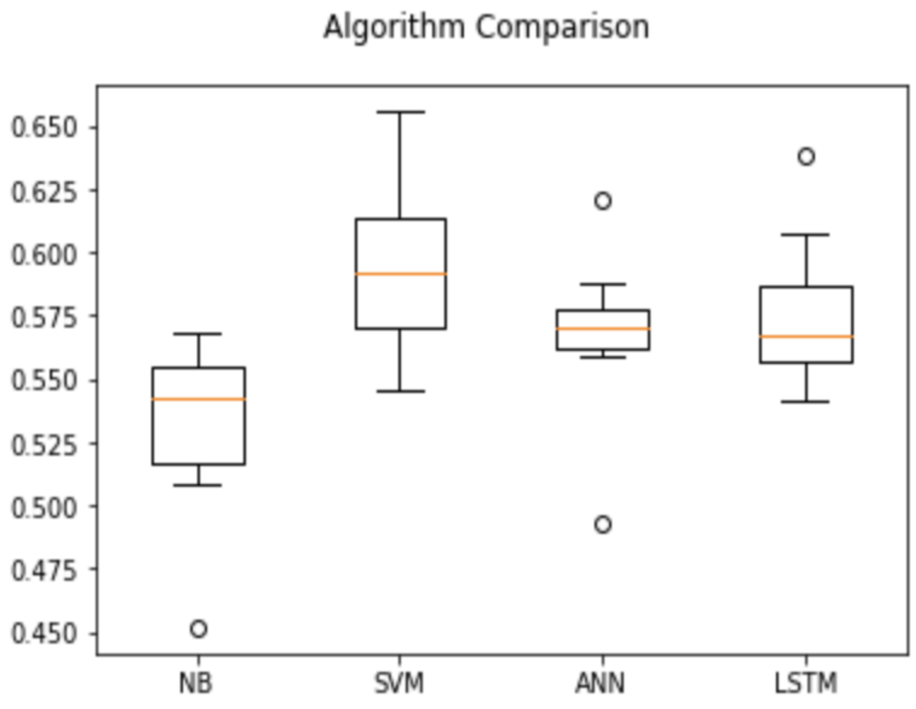
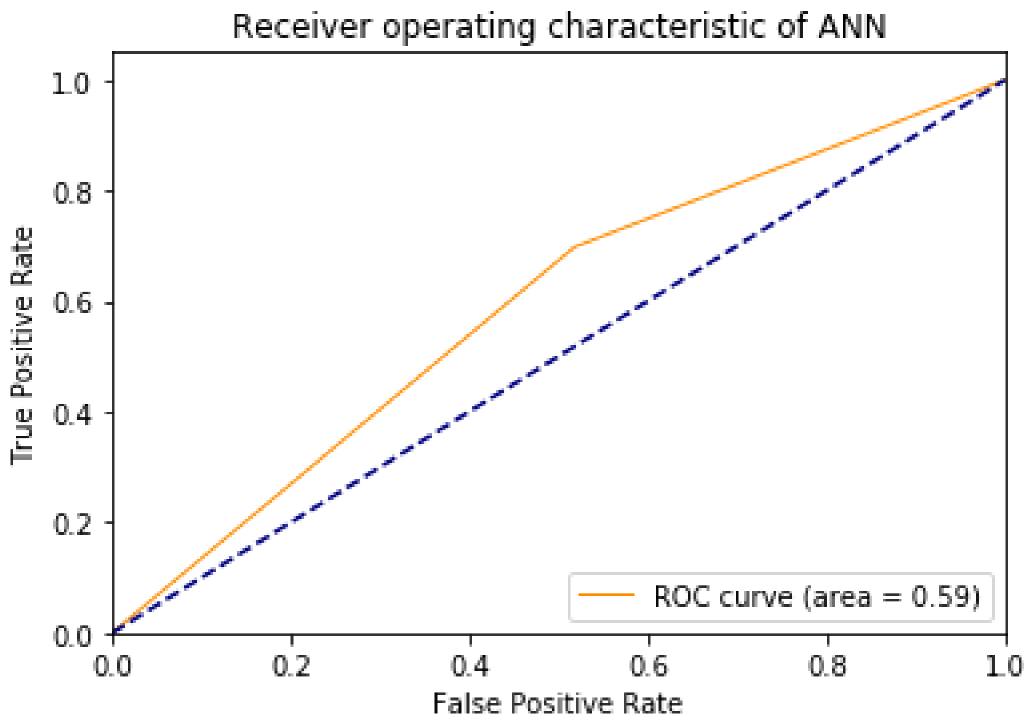
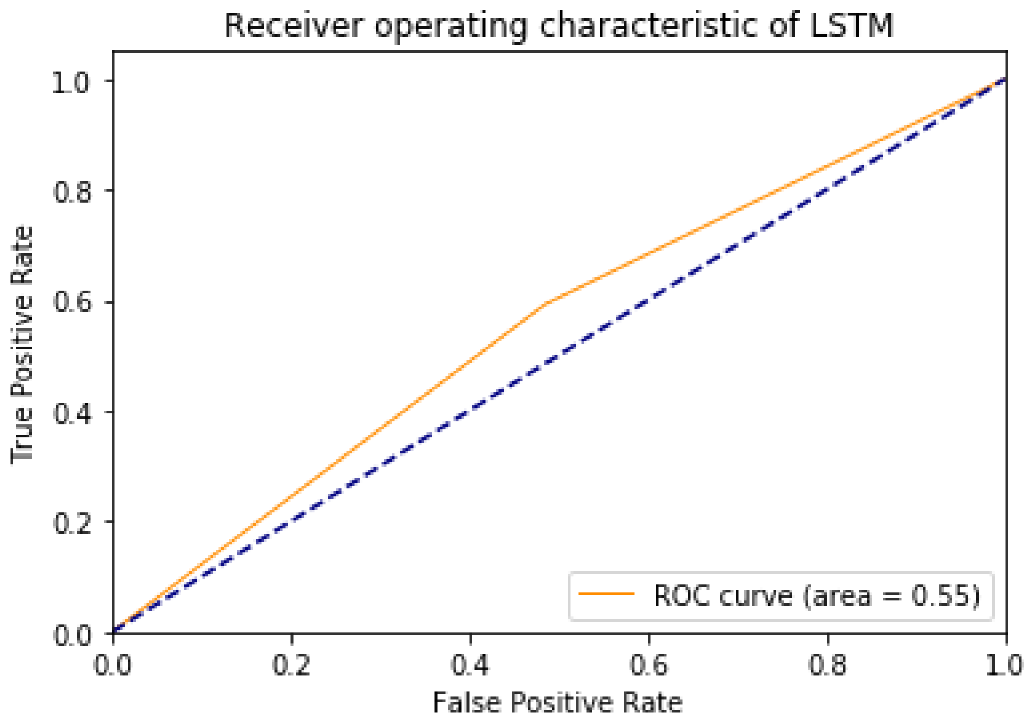
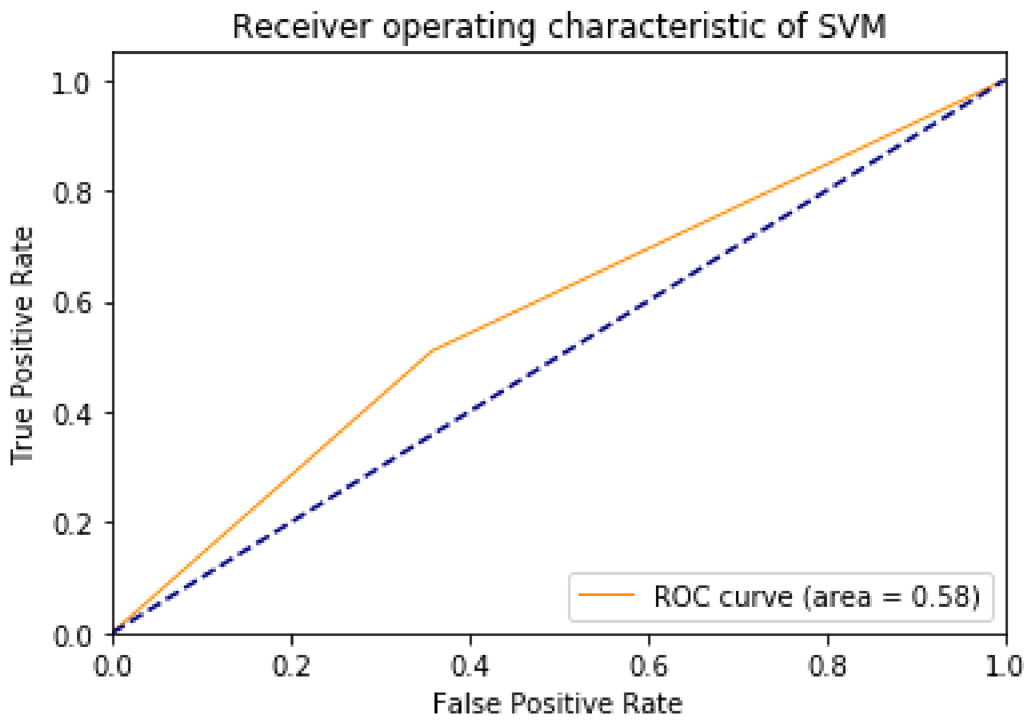


Figure 4: Algorithm after cross_validation performance comparison

From the cross-validation result, SVM and LSTM showed better performances. However, Naive Bayes stands out of the crowd when it came to test prediction performances. Naive Bayes gave the highest F1_score of 0.685 and the recall score is also high which is 0.805 on the test set. The figure shown below are the ROC curves of each model:







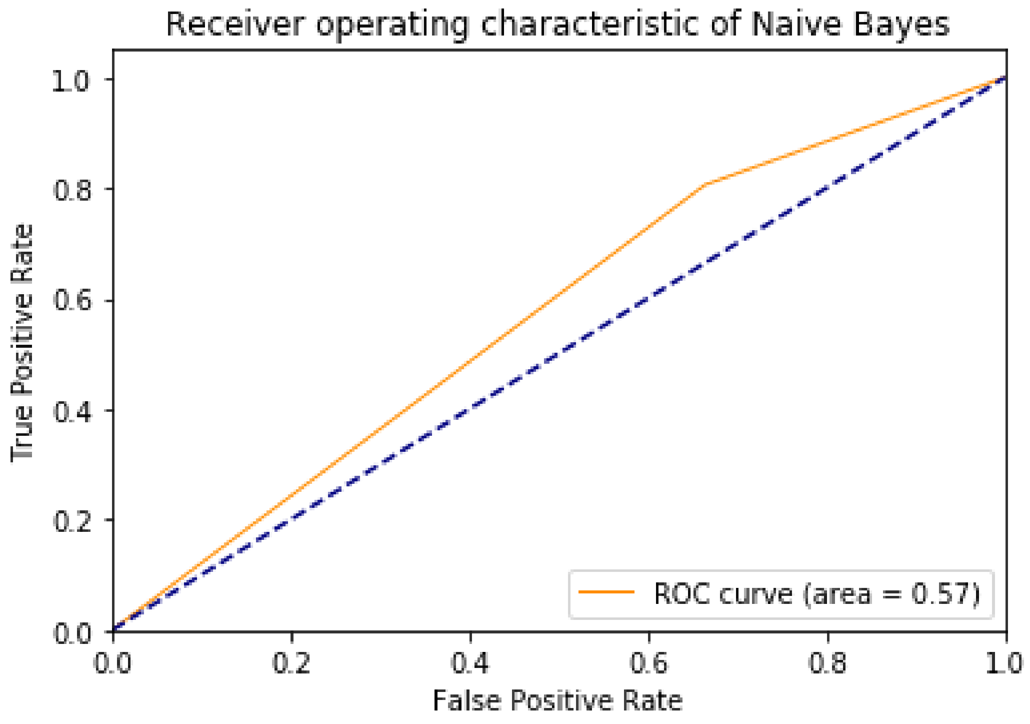


Figure 5: ROC curve of each model

Step 6: Performance Metrics

Based on the performance metrics above, Naive Bayes gave the highest F1 score of 0.685 and recall score of 0.805 on the test set. Naive Bayes gave the best performance. The second-best model is the feed-forward ANN model with F1 score of 0.646. So the predictions from Naive Bayes model will be used as signals for backtesting.

Before going further to understand performance metrics, we need to understand the definition of binary prediction outcomes:

- **True Negative:** predict 0 while should have the class is actually 0
- **False Negative:** predict 0 while should have the class is actually 1
- **False Positive:** predict 1 while should have the class is actually 0
- **True Positive:** predict 1 while should have the class is actually 1

Below are the performance metrics of Naive Bayes model:

1. ROC curve is the curve plots TPR vs. FPR at different classification threshold. The probability to correctly predict true positive case is 0.8 which is acceptable. However, the false positive rate at 0.66 or proportion of actual negative cases which are not correctly identified is quite low.
2. Confusion Matrix: [[152 300, 107 442]] which we can see the number of positive and negative cases are quite similar and balance.
3. Precision: 0.59 the proportion of positive cases that were correctly identified. We also need to improve this rate.
4. Recall or Sensitivity: 0.8 represent the proportion of actual positive cases which are correctly identified.
5. F1 score: 0.68, similar accuracy but considers both the precision and the recall of the test to compute the score, reach best at 1 and worse at 0. In our case, 68% is still low but at least show positive sign.
6. Accuracy: 0.59, the proportion of the total number of predictions that were correct is considered low.
7. AUC score: 0.57, is the area under the curve which measures the entire two-dimensional area underneath the entire ROC curve. There are many ways to interpret AUC, but one can say it is the probability that the model ranks a random positive example more highly than a random negative example. The perfect AUC will be 1 which it should rank positive and negative the same.

Step 7: Features Selection for better Performance

We used PCA with 3 components to select features in order to improve the model performance.

```
[0.32753046 0.22306156 0.19715589]
```

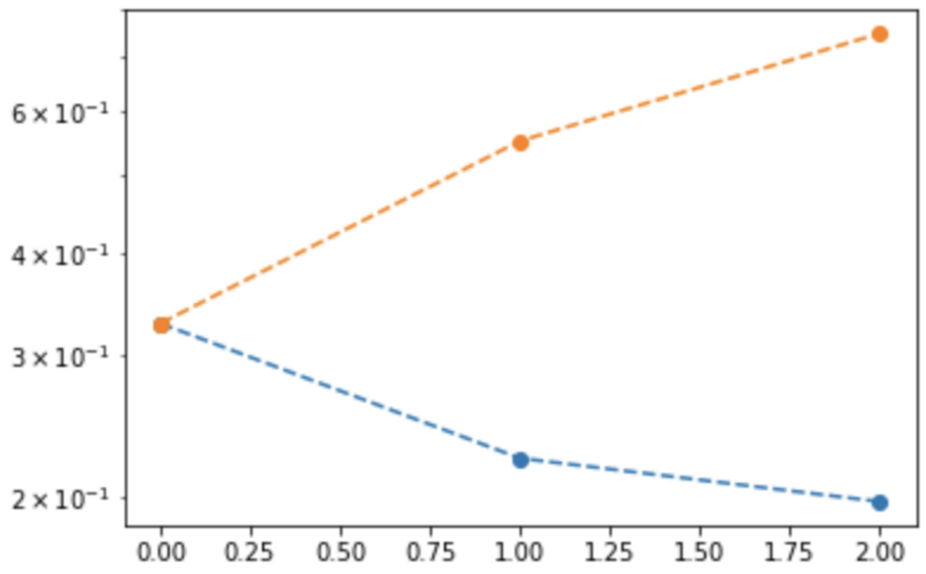
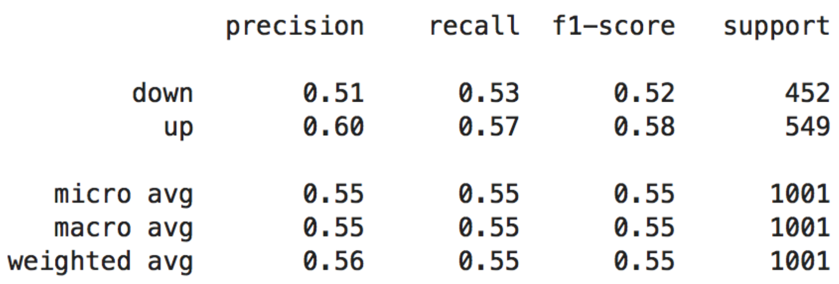


Figure 6: Components importance

As we can see, the first three components already explained 73% of the model which is quite impressive. The additional components have less importance on the model. However, when we fitted the new features after applying PCA in, the f1 score performance is much less than without feature reduction. It might be good as it might avoid bias and overfitting. However, it does not mean feature reduction is a good solution for better performance in this case. Therefore, we decided to leave the features as it is and will optimize in further research.



```
              precision    recall  f1-score   support

        down       0.51      0.53      0.52       452
          up       0.60      0.57      0.58       549

   micro avg       0.55      0.55      0.55      1001
   macro avg       0.55      0.55      0.55      1001
weighted avg       0.56      0.55      0.55      1001
```

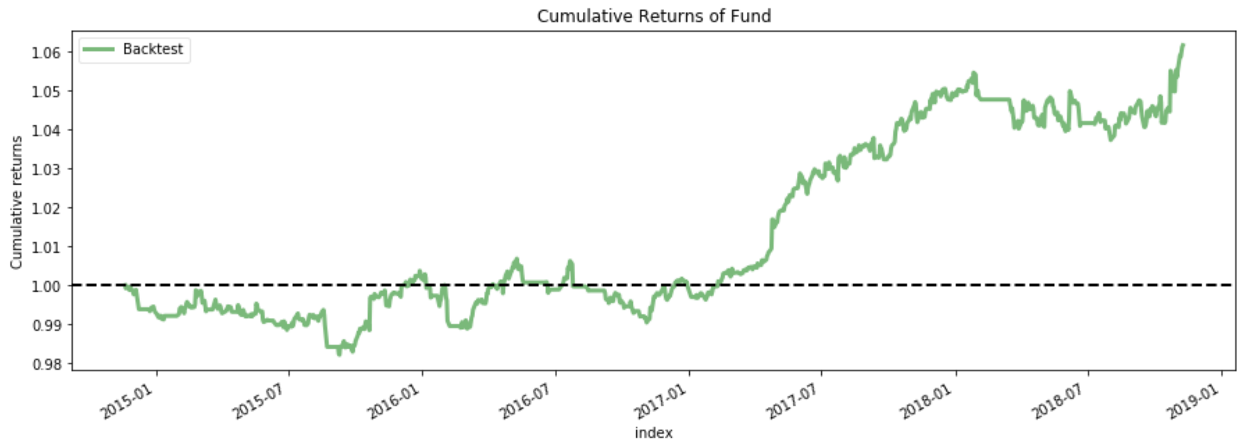
Figure 7: Classification report

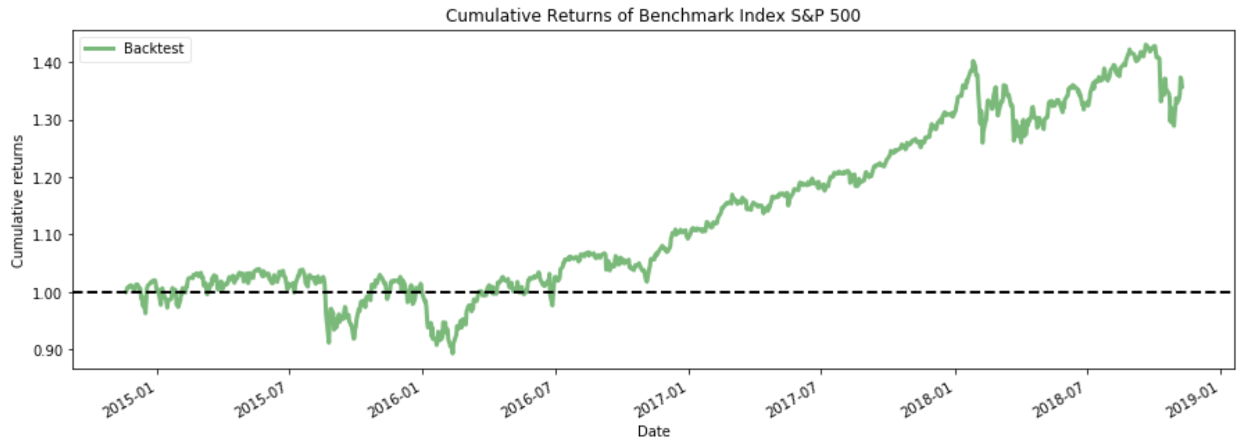
Step 8: Fund Factsheet

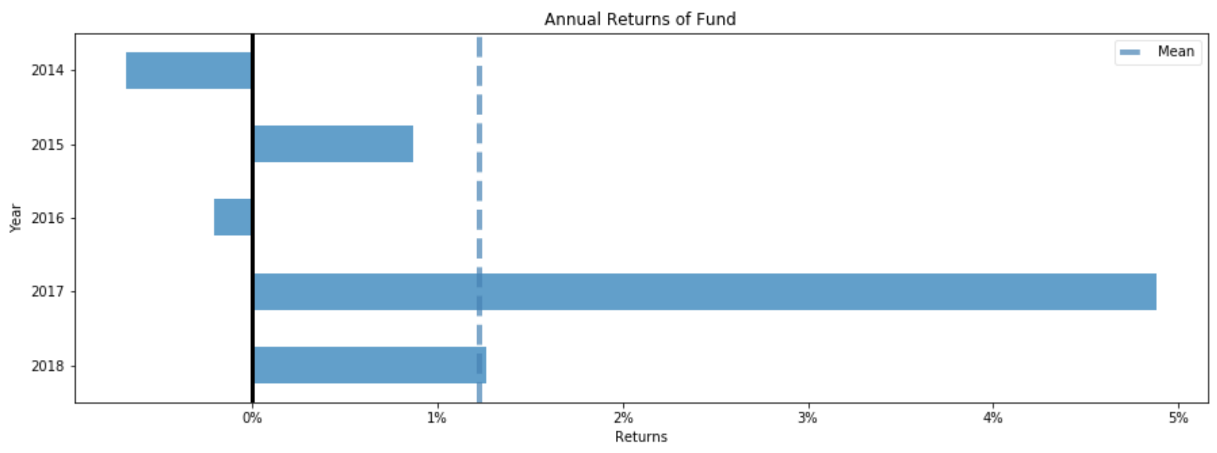
Our trading strategy was simple: go long when the prediction result > 0.5, go short when prediction result < 0.5. The backtesting traded a total of 67 trades and earn USD 5,879 for an initial capital of USD 100,000 for 4 years with annual return of 1.5%. The cumulative return was 1.06%. It was less than index benchmark return of 1.3%.

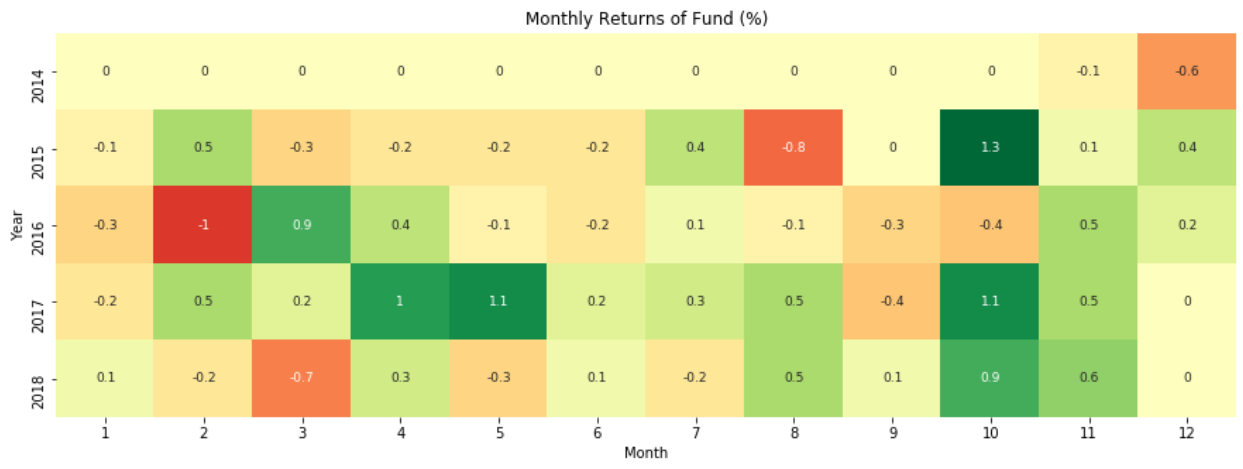
```
Total number of trades: 67
Total number of closed trades: 66
Total number of opening trades: 1

Gross profit/loss: USD 5879.007000000019
Net profit/loss: USD 5879.007000000019
Paper gain/loss: USD 294.0002000000468
Total gain/loss: USD 6173.0072000000655

Total trades won: 30 (Won amount: USD 13302.004200000008 )
Total trades lost: 36 (Lost amount: USD -7422.997199999988 )
```

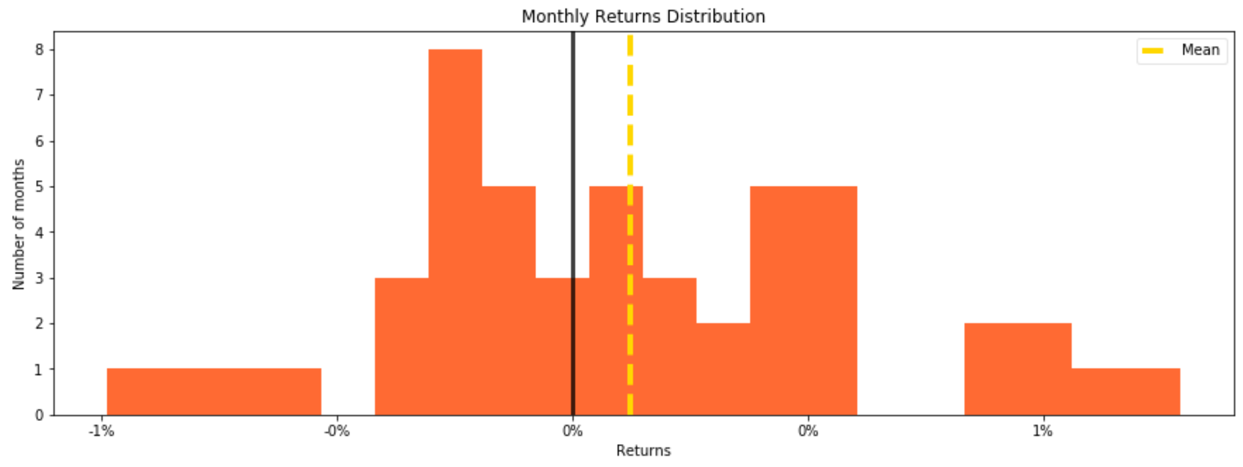
Figure 8: Fund fact sheet 1






Annual Returns of Fund
Mean
2014
2015
2016
2017
2018
Year
0%
1%
2%
3%
4%
5%
Returns


Monthly Returns of Fund (%)
2014: 0 0 0 0 0 0 0 0 0 0 -0.1 -0.6
2015: -0.1 0.5 -0.3 -0.2 -0.2 -0.2 0.4 -0.8 0 1.3 0.1 0.4
2016: -0.3 -1 0.9 0.4 -0.1 -0.2 0.1 -0.1 -0.3 -0.4 0.5 0.2
2017: -0.2 0.5 0.2 1 1.1 0.2 0.3 0.5 -0.4 1.1 0.5 0
2018: 0.1 -0.2 -0.7 0.3 -0.3 0.1 -0.2 0.5 0.1 0.9 0.6 0
Year
1 2 3 4 5 6 7 8 9 10 11 12
Month


Monthly Returns Distribution
Mean
Number of months
0 1 2 3 4 5 6 7 8
-1%
-0%
0%
0%
1%
Returns

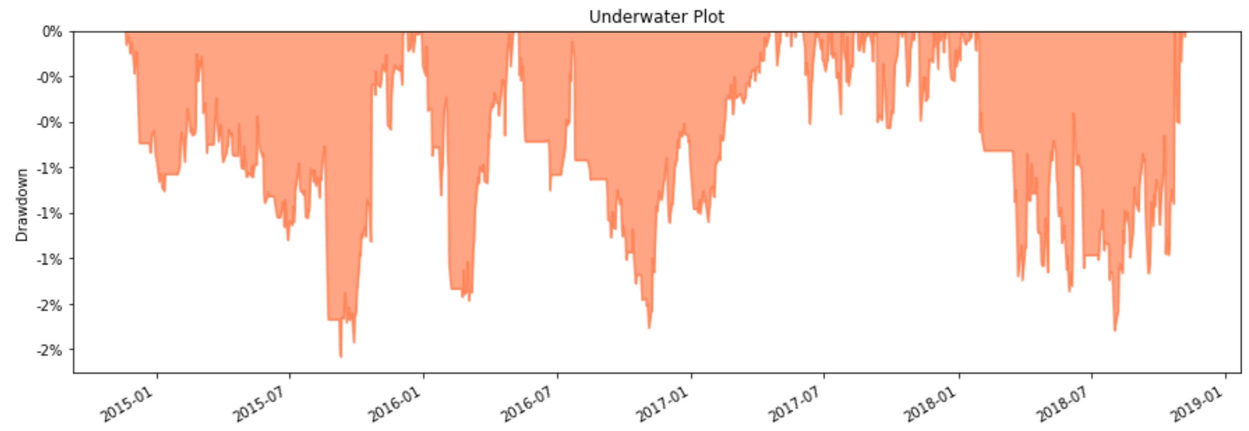


Figure 9: Fund fact sheet 2

For further improvement, we will need to consider the width range of stock price movement as the movement up or down is not enough. In real life, we would like to give more bet to the trades which we think have high probability of bringing return and ignore those downward movements that resulted in small losses which would not be wise to sell it immediately.

We also can try to incorporate the trading strategy directly to the labeling algorithm such as to incorporate stop-out thresholds in the labeling algorithm or the 90 days rolling standard deviation threshold. By creating two algorithms to make trading decisions, it will somehow avoid the problem of overfitting of one algorithm.

References:

[1]. Klaus Greff, Rupesh K. Srivastava, Jan Koutn´ık, Bas R. Steunebrink, Ju¨rgen Schmidhuber, (2016) *"LSTM: A Search Space Odyssey"*.

[2]. Sreelekshmy Selvin, Vinayakumar R, Gopalakrishnan E.A, Vijay Krishna Menon, Soman K.P (2017) *"STOCK PRICE PREDICTION USING LSTM, RNN AND CNN-SLIDING WINDOW MODEL".*

[3]. Jason Brownlee (2016) *Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras.* Available at :https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/

[4]. Jigar Patel (2014), *'Predicting stock and stock price index movement using Trend Deterministic Data Preparation and Machine learning'*

[5]. Kyoung-jae Kim (2003) *'Financial time series forecasting using support vector machines'.* Available at http://www.computerscienceweb.com.

[6]. Pyfolio. Available at: https://github.com/quantopian/pyfolio

[7]. Backtrader. Available at: https://github.com/backtrader/backtrader

[8]. SUNIL RAY (2018) *Improve Your Model Performance using Cross Validation (in Python and R)* Available at: https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r.

[9]. *Important model evaluation error metrics*. Available at: https://www.analyticsvidhya.com/blog/2016/02/7-important-model-evaluation-error-metrics/