Machine Learning Engineer Nanodegree

Financial Time-Series Prediction via Machine Learning

Bryan Chik

15th August 2017

I. Definition

Project Overview

Throughout the past 50 years, there has been numerous attempts and a rich discussion on financial markets mechanics and looking drivers of asset returns. However, one drawback of classical methods in predicting asset returns are most models are linear in nature. Therefore, most models have bad prediction power, as they cannot cater any non-linearity with the markets. Machine learning techniques could overcome this problem and qualifies as a good candidate to try solving the asset price prediction problem.

In this project, I would like to explore the possibility of applying machine learning techniques to predict Hong Kong Hang Seng index's next day price movement, given closing prices of other 42 global exchange indices. A trading strategy will be designed based on the results of this research.

Initially I have obtained daily closing price of Hang Seng index and 42 other selected exchange indices globally from data provided by FactSet, starting from 1st January 2010 to 30th June 2017. A full list of indices can be found in the appendix.

The data is then trained to various model candidates and use for model performance evaluation and prediction.

Problem Statement

In this project, I am predicting Hang Seng Index's next day movement, which has 3 states: Up, Neutral and Down. This problem is a classification problem.

"Up" label is defined as the daily price movement compared to yesterday's close is greater than 0.5%. "Down" label is defined as daily price movement compared to yesterday's close is less than -0.5%. While "Neutral" label is defined as not "Up" or "Down".

4 classification algorithms will be utilized and see which one has the best predictability. Namely

- Logistic Regression
- AdaBoost
- Feed-Forward Neural Network
- LSTM

The performance of the above models are compared against a benchmark model, which in this case we are using Gaussian Naïve Bayes model, using 2 evaluation metrics for classification problems:

- F-beta Score (beta = 0.6)
- Precision

I anticipate there is at least 1 model would be better than the benchmark model, with higher precision and F-beta score. This in turn meaning the model does have higher prediction power, and can be used as an input to make trading decisions based on the prediction.

Metrics

2 specific metrics are chosen for evaluating the model performance:

- F-Beta Score (beta = 0.6)
- Precision

F-beta score is defined as

$$F_\beta = (1 + \beta^2)\frac{p \cdot r}{\beta^2 \cdot p + r}$$

where p = precision, r = recall

F-beta score is an overall measure that describes model accuracy, taking into account precision and recall. Note that I have placed special emphasize on precision with a beta value larger than 0.5. The reason behind is that in practical trading, it is easier to go long than short, so therefore, we want to predict better the “Up” labels.

For the same reason, precision is included in the evaluation metrics.

II. Analysis

Data Exploration & Visualization

The data that I used to study is sourced from FactSet Research System’s investment management workstation terminal. I have downloaded daily closing index levels of 43 global equity exchange indices, including Hang Seng index, in the form of a csv file. The first column is the date index, starting from earliest to closest. A snippet of the data is shown below.

	Date	Hang Seng Index	SSE Composite Index	ASX All Ordinaries	India S&P BSE SENSEX	TOPIX	KOSPI Composite Index	Taiwan TAIEX	FTSE Bursa Malaysia KLCI	FTSE Straits Times Index	...	S&P 500	DJ Industrial Average	Colombia IGBC	Canada S&P/TSX Composite	Bra Boves Ind
0	20100101	21872.50	3277.139	4882.7	17464.81	907.59	1682.77	8188.11	1272.78	2897.62	...	1115.100	10428.05	11602.14	11746.11	68588.
1	20100104	21823.28	3243.760	4889.8	17558.73	915.75	1696.14	8207.85	1275.75	2894.55	...	1132.990	10583.96	11641.37	11866.90	70045.
2	20100105	22279.58	3282.179	4939.5	17686.24	919.57	1690.62	8211.40	1288.24	2920.28	...	1136.520	10572.02	11628.92	11888.08	70239.
3	20100106	22416.67	3254.215	4946.8	17701.13	931.13	1705.32	8327.62	1293.17	2930.49	...	1137.140	10573.68	11626.59	11944.54	70729.
4	20100107	22269.45	3192.776	4930.5	17615.72	931.85	1683.45	8237.42	1291.42	2913.25	...	1141.695	10606.86	11594.89	11887.51	70451.

5 rows × 44 columns

Figure 1: First 5 rows of the raw index close data

Next, I have generated some basic descriptive statistics of the data set. This is an attempt to get some knowledge around the distributional properties and shape of the data, and with the aim to determine any data pre-processing is required or not. An excerpt of the output is shown below.

	Hang Seng Index	SSE Composite Index	ASX All Ordinaries	India S&P BSE SENSEX	TOPIX	KOSPI Composite Index	Taiwan TAIEX	FTSE Bursa Malaysia KLCI	FTSE Straits Times Index	Philippines PSE PSEi
count	1956	1956	1956	1956	1956	1956	1956	1956	1956	1956
mean	22276.82	2724.053	5041.916	22191.4	1139.882	1961.786	8439.77	1642.514	3061.694	5952.85563
std	1970.601	570.1801	503.8077	4533.157	296.9741	130.6422	783.3326	154.248	210.6159	1496.42738
min	16250.27	1950.013	3927.6	15175.08	695.51	1552.79	6633.33	1233.86	2528.71	2797.61
25%	20856.13	2235.232	4633.075	18200.95	849.33	1907.788	7829.51	1547.295	2883.243	4372.585
50%	22482	2686.009	5088.05	20365.1	1178.115	1976.71	8412.44	1657.094	3070.81	6463.375
75%	23458.26	3083.533	5436.925	26672.6	1376.413	2030.705	9040.78	1770.151	3226.84	7227.73
max	28442.75	5166.35	5976.4	31311.57	1691.29	2395.66	10513.96	1892.653	3539.95	8127.48

Figure 2: Descriptive Statistics of Asia Pacific indices

There are some important observations within the table:

1. We have around 1,956 sample data points
2. Different indices have different levels. For example, mean Hang Seng level is 22,276.82, while for Japan TOPIX index the mean level is around 1,139.88
3. Different indices have different volatility profiles and range

From the above observation, particularly 2 and 3, imply that the data is not directly usable as they are at different levels or scale. In order to preserve the structure of the data, the raw data has to go through a normalization process.

To further visualize the structure of the data, below is a plot of the data

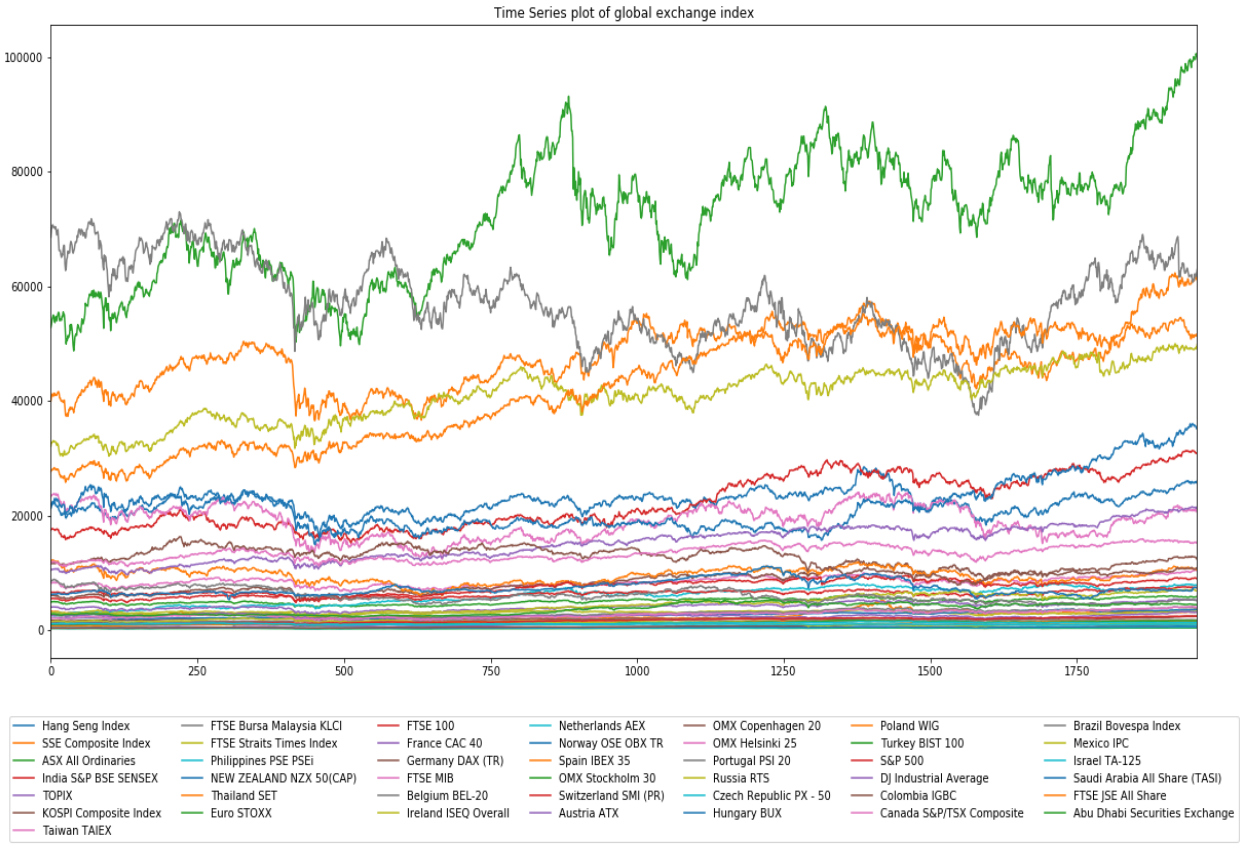

Figure 3: Time series plot of index data

From the above figure, the indices have various levels. In order to preserve their structure, normalization is required.

There is an extra point to note from the above plot. It looks like all the indices exhibit a trend. This trend component will cause issues during our fitting process, and therefore we need to "de-trend" the data. In statistics and time-series analysis context, the data has to be transformed into a stationary series before they can serve as an input in fitting our models.

A practical issue with the current data is that since the exchanges are in different time zones, we will need to apply a proper lag to indices that are in slower time zones. Based on each exchange's closing time, the below table lists out which indices need to be lagged. This step is essential and critical to avoid "look-ahead bias" which you predicting the past using the future.

	Index	Country	UTC Closing Time	UTC +8 Closing Time	Need Lag?
0	Hang Seng Index	Hong Kong	8:00:00 AM	4:00:00 PM	
1	SSE Composite Index	Shanghai	7:00:00 AM	3:00:00 PM	FALSE
2	ASX All Ordinaries	Australia	6:00:00 AM	2:00:00 PM	FALSE
3	India S&P BSE SENSEX	India	10:00:00 AM	6:00:00 PM	TRUE
4	TOPIX	Japan	6:00:00 AM	2:00:00 PM	FALSE
5	KOSPI Composite Index	South Korea	6:00:00 AM	2:00:00 PM	FALSE
6	Taiwan TAIEX	Taiwan	5:30:00 AM	1:30:00 PM	FALSE
7	FTSE Bursa Malaysia KLCI	Malaysia	9:00:00 AM	5:00:00 PM	TRUE
8	FTSE Straits Times Index	Singapore	9:00:00 AM	5:00:00 PM	TRUE
9	Philippines PSE PSEi	The Philippines	7:30:00 AM	3:30:00 PM	FALSE
10	NEW ZEALAND NZX 50(CAP)	New Zealand	5:00:00 AM	1:00:00 PM	FALSE
11	Thailand SET	Thailand	9:30:00 AM	5:30:00 PM	TRUE
12	Euro STOXX	London	5:00:00 PM	1:00:00 AM	TRUE
13	FTSE 100	London	4:30:00 PM	12:30:00 AM	TRUE
14	France CAC 40	France	4:30:00 PM	12:30:00 AM	TRUE
15	Germany DAX (TR)	Frankfurt	7:00:00 PM	3:00:00 AM	TRUE
16	FTSE MIB	Italy	4:25:00 PM	12:25:00 AM	TRUE
17	Belgium BEL-20	Belgium	4:30:00 PM	12:30:00 AM	TRUE
18	Ireland ISEQ Overall	Ireland	4:30:00 PM	12:30:00 AM	TRUE
19	Netherlands AEX	Netherlands	4:40:00 PM	12:40:00 AM	TRUE
20	Norway OSE OBX TR	Norway	3:30:00 PM	11:30:00 PM	TRUE
21	Spain IBEX 35	Spain	4:30:00 PM	12:30:00 AM	TRUE
22	OMX Stockholm 30	Sweden	4:30:00 PM	12:30:00 AM	TRUE
23	Switzerland SMI (PR)	Switzerland	4:30:00 PM	12:30:00 AM	TRUE
24	Austria ATX	Austria	4:35:00 PM	12:35:00 AM	TRUE
25	OMX Copenhagen 20	Denmark	4:00:00 PM	12:00:00 AM	TRUE
26	OMX Helsinki 25	Finland	4:30:00 PM	12:30:00 AM	TRUE
27	Portugal PSI 20	Portugal	4:30:00 PM	12:30:00 AM	TRUE
28	Russia RTS	Russia	3:45:00 PM	11:45:00 PM	TRUE
29	Czech Republic PX - 50	Czech	3:30:00 PM	11:30:00 PM	TRUE
30	Hungary BUX	Hungary	4:00:00 PM	12:00:00 AM	TRUE
31	Poland WIG	Poland	4:00:00 PM	12:00:00 AM	TRUE
32	Turkey BIST 100	Turkey	3:30:00 PM	11:30:00 PM	TRUE
33	S&P 500	United States	9:00:00 PM	5:00:00 AM	TRUE
34	DJ Industrial Average	United States	9:00:00 PM	5:00:00 AM	TRUE
35	Colombia IGBC	Colombia	7:00:00 PM	3:00:00 AM	TRUE
36	Canada S&P/TSX Composite	Canada	9:00:00 PM	5:00:00 AM	TRUE
37	Brazil Bovespa Index	Brazil	8:00:00 PM	4:00:00 AM	TRUE
38	Mexico IPC	Mexico	9:00:00 PM	5:00:00 AM	TRUE
39	Israel TA-125	Israel	3:30:00 PM	11:30:00 PM	TRUE
40	Saudi Arabia All Share (TASI)	Saudi Arabia	12:00:00 PM	8:00:00 PM	TRUE
41	FTSE JSE All Share	South Africa	3:00:00 PM	11:00:00 PM	TRUE
42	Abu Dhabi Securities Exchange	Abu Dhabi	11:00:00 AM	7:00:00 PM	TRUE

Figure 4: List of exchange closing time

Source: https://en.wikipedia.org/wiki/List_of_stock_exchange_opening_times

Finally, the selected indices would need to possess some kind of correlation to Hang Seng in order for them to qualify as a predictor. If the index is independent of Hang Seng, then it has no contribution to Hang Seng's movement information and can be dropped. To verify this, I have generated a sample correlation heat map below to investigate the correlation structure, using the lagged transformed data.

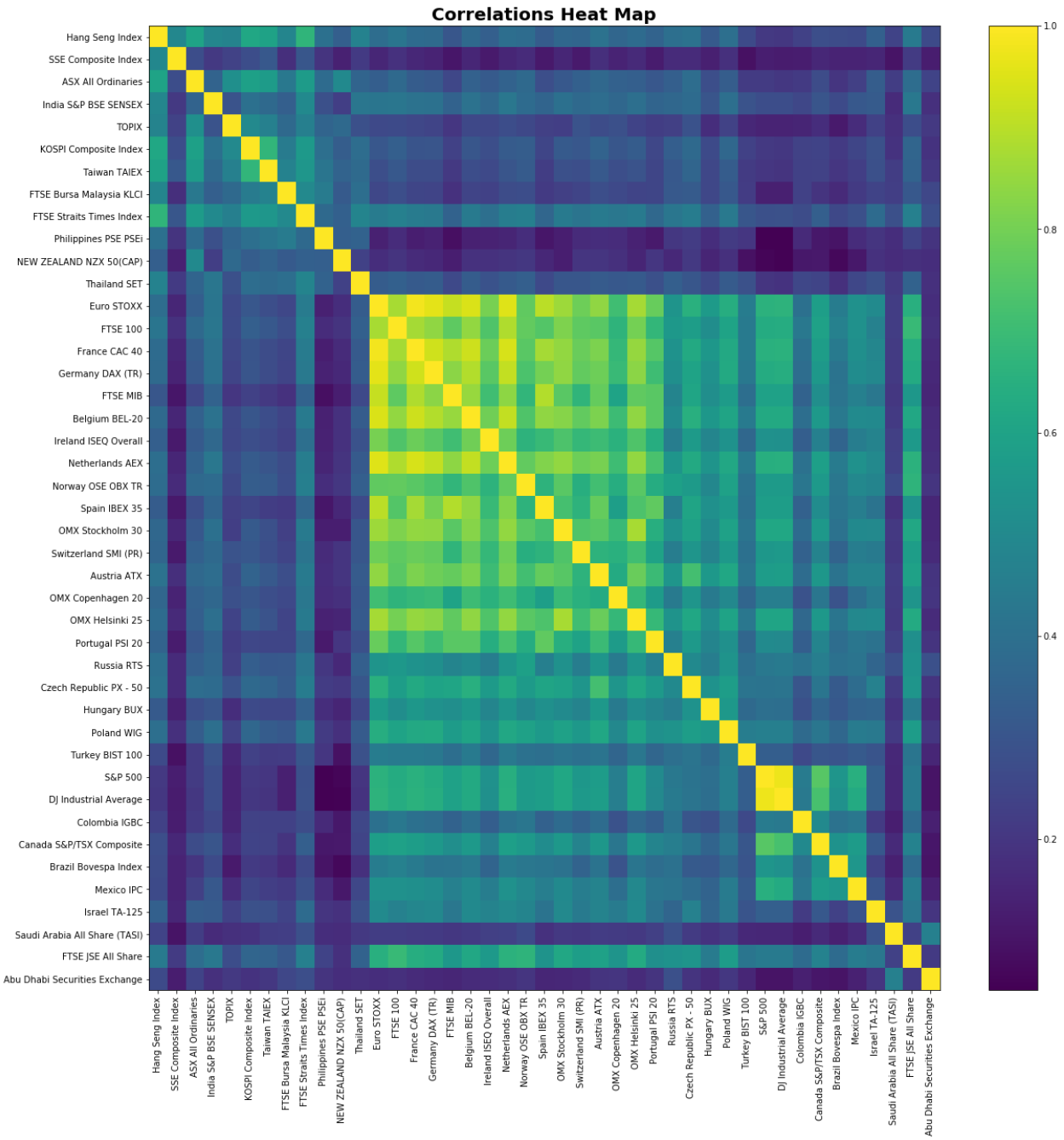

Figure 5: Sample correlation of the lagged index data

From the above chart, we can observe the following

1. Indices within the same geographical region tends to have higher correlation
2. Developed market indices are more correlated to each other compared to emerging market indices
3. Hang Seng index itself are more correlated to Asian indices, but indices from Eurozone and Americas also have positive correlation

From the observations above, we can initially confirm the indices should possess various degrees of prediction power towards Hang Seng index. Intuitively this makes sense as stocks listed in Hang Seng index are likely to have business running in different countries globally. Therefore, each stock should possess exposures to different countries, and its stock price will be affected by those markets.

Algorithms and Techniques

In this project, I have explored 4 different algorithms and compared their performance:

- Logistic Regression
- AdaBoost
- Feed-forward Neural Network
- LSTM

All 4 of the algorithm can be used as a classifier and able to take in continuous numerical feature inputs for prediction.

Logistic Regression

Logistic Regression served as a start as a lot of classical financial economics literature has been written based on a linear model form. We used the pre-processed data and fit the logistic regression model. Logistic regression aims at finding the minimizing the cross-entropy between the predicted labels and the true labels (which in our case, Up/Down/Neutral). The model finds the regression coefficient w that, for $L2$-penalized cost

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Or $L1$-penalized cost

$$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Source: http://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

During the fit, time-series cross-validation and grid-search is performed to find the optimal regularization hyper-parameter C and whether $L1$ or $L2$ penalization provides the best fit.

Special attention should be placed during the cross-validation. In this project, I have used a time-series cross-validation method which is an extension to K-fold cross-validation technique. Traditional K-fold cross-validation is not applicable in our context due to the fact that the data possess ordering. If we randomly shuffle the data, there is a possibility where the training set contain future time points and used to predict the past within the test set. Instead, the time series cross-validation will divide the data into K folds. Label K_i as the i-th fold, where $i \epsilon \{1,2, \dots, K\}$. At the i-th iteration during cross-validation, we will be using $K_1 \cup K_2 \cup \dots \cup K_i$ as the training set, and K_{i+1} as the test set. The chart below depicts

the cross validation process. This can ensure that the ordering of the time series can be preserved and no future time points will be used to predict the past.

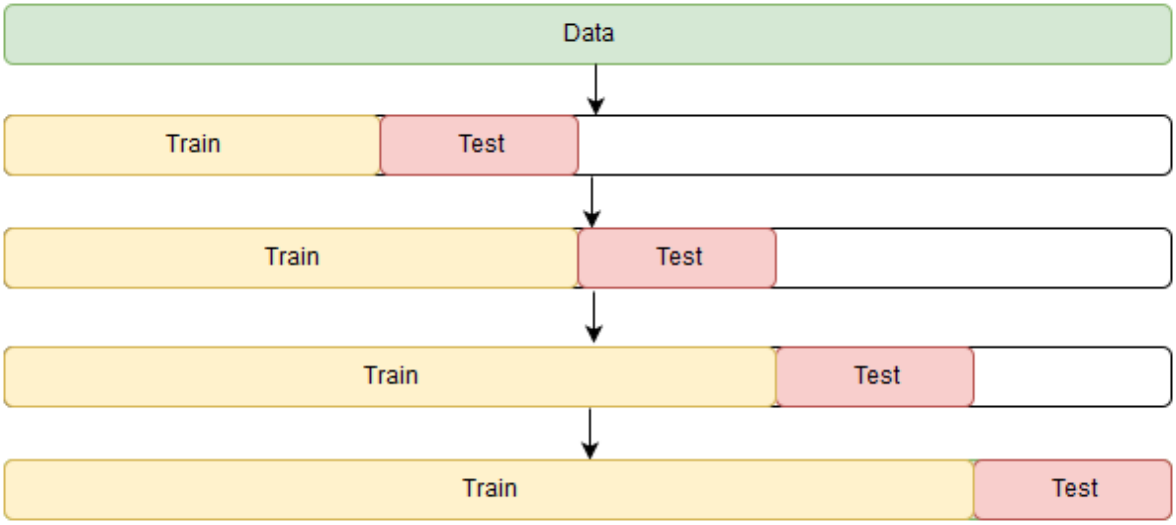

Figure 6: Diagram depicts the time-series cross validation method used in this project. Source: https://stats.stackexchange.com/questions/14099/using-k-fold-cross-validation-for-time-series-model-selection

Finally, C is a hyper-parameter that controls the proportion between the cross-entropy error and the regularization. A larger C puts more emphasize on minimizing the cross-entropy, and diminishing the regularization effect on the regression coefficients.

AdaBoost

The next model I have tried is the AdaBoost model, which is a model falls in a bigger class of ensemble learning models. Ensemble methods are well-known for its strong performance in real-world machine learning problems.

Ensemble methods rest on a principle that iteratively fitting a weak learner could converge to a strong learner and therefore gives better predictive power.

In this project, I have used the default weak learner from scikit-learn library, which is by using Decision Trees and also the default boosting algorithm, SAMME.R . Similar to the Logistic Regression case, a grid-search is performed on the below hyper-parameters

- Number of estimators: Maximum number of estimators that the boosting algorithm will stop. The grid search is performed in the discrete set $\{1, 10, 100, 300, 500\}$
- Learning Rate: Controls the shrinkage of the contribution to each classifier. The grid search is performed in the discrete set $\{0.001, 0.01, 0.1, 1, 10\}$

Same cross-validation methodology is used from logistic regression.

Feature importance metrics can also be generated in AdaBoost algorithms and cross check with intuition to see whether the model results make sense.

Feed-Forward Neural Network

Next I have turned to deep learning techniques, starting with a simple Feed-forward Neural Network. With a single hidden layer and 22 perceptrons. While there is no consensus on picking the number of perceptrons, a rule of thumb is to use the average number between the number of input features (i.e. 42) and number of outputs (i.e. 3).

Neutral Network is able to capture non-linearity exhibit with financial markets, and therefore I would expect the model should do better.

Other parameters of the Neural Network are as follows:

- Drop-out Probability = 0.6
- Activation: ReLU
- Epoch: 100
- Optimizer: Nesterrov Adam
- Learning Rate = 0.0003
- Batch Size = 1
- Loss function = Cross-Entropy

Instead of cross-validation, the sample is split into 80% training and 20% test set. 100 epoch runs are done during the training step, and predicts are run on the test set.

ReLU and drop-out are nowadays a standard feature to enhance the model performance. ReLU are known to provide better and more stable results, while drop-out can prevent overfitting.

Nesterov Adam optimizer is essentially Adam optimizer incorporated with momentum features. Nesterov momentum tends to have superior performance in locating the optimal point. (*Incorporating Nesterov Momentum in Adam* - http://cs229.stanford.edu/proj2015/054_report.pdf).

Learning rate is finely tuned such that when running 100 epochs the model loss is stabilized. More will be discussed in the next section.

LSTM Network

Finally, a type of recurrent network, long short-term memory network is used to fit the data. The LSTM network is capable of capturing long-term dependencies of the network states which traditional neural networks are not able to handle. There are well documented literatures describing financial time series possess memories, or “momentum” effects. As a result, LSTM is perhaps a very good candidate model such that this effect can be captured within the network architecture.

Similar to the Feed-forward Neural Network, I have only used 1 hidden layer and 22 percetrons.

Other parameters of the Neural Network are as follows:

- Drop-out Probability = 0.6
- Activation: ReLU
- Epoch: 100
- Optimizer: Nesterrov Adam
- Learning Rate = 0.0003
- Batch Size = 1
- Loss function = Cross-Entropy

Similar to the Feed-Forward Neural Network case, I have used 80% of the data as training set and 20% as test set to validate the model.

Benchmark

I have chosen the Naïve Gaussian Bayes as the benchmark model. Empirically, the returns distribution is approximately Gaussian. Therefore we can compare the models whether they are better off than computing the best likelihoods, under a Gaussian assumption.

The same evaluation metrics are computed (i.e. F-beta and precision) to compare across the models.

III. Methodology

Data Preprocessing

From the discussion in *Data Exploration* section, the following preprocessing steps are applied to the data

1. Transform the closing price data to returns data by calculating the 1-day continuous return
2. Apply suitable lag to exchanges that are in a slower timezone according to Figure 4
3. Generate labels according to Hang Seng's 1-day continuous return using the threshold previously described

The dataset then can be used to be trained in the algorithms mentioned.

Implementation

In this project, I have chosen scikit-learn's implementation for Logistic Regression and AdaBoost. For Neural Networks, I have used keras with tensorflow backend.

After data-preprocessing, the training data is then fed into the algorithms implemented within scikit-learn and keras. The chosen evaluation metrics, F-beta and precision are in-built. While for keras, I have

implemented a simple function mimicking the calculation to be used in the evaluation. The training process is a rather straightforward exercise of applying the functions within the libraries.

Refinement

The initial model comparison has led to a best performing model, which is the LSTM model. It gave both the highest F-beta score equals 0.8235, a precision of 0.9352 and a model loss at 0.6937. It took my PC (Intel 4-core 3.2Ghz CPU, 24GB DDR3 RAM) 5 minutes and 20 seconds to finish 100 epochs.

As a step to further explore whether better performance can be achieved, I have tried to see whether feature space reduction can improve the results. I have applied 2 features selection method:

- Principal Component Analysis (PCA)
- Tree-based Selection

I have picked the number of principal components with cumulative variance explained to 90%, and reduced the features space, and re-fit the data using the best performing LSTM model. Since the number of features is reduced to 19, the number of perceptrons used are lowered to 11.

With the tree-based selection, I have re-used the feature importance from previous AdaBoost fit, and selected features with importance greater than the overall average. Using the reduced data, similar to PCA, it is used to re-fit with the LSTM model. Again, since the features number has reduced to 21 , the number of perceptrons have reduced to 9. Results are below

	F-Beta (beta = 0.6)	Precision	Training Time
LSTM	0.8244	0.9352	5 mins 24s
PCA – LSTM	0.8261	0.9352	5 mins 5s
Tree-based - LSTM	0.8244	0.9437	5 mins 17s

After apply feature selection, the F-beta and Precision scores have not significantly increased. However, there is a marginal improvement in training time.

IV. Results

Model Evaluation and Validation

The results of the model fitting are below

	F-Beta (beta = 0.6)	Precision
Benchmark	0.4922	0.5441
Logistic Regression	0.4697	0.4586
AdaBoost	0.6174	0.6827
Feed-Forward Neural Network	0.5806	0.8568
LSTM	0.8244	0.9352

From the above table, we can see the following

- All models except Logistic Regression perform better than our benchmark model
- LSTM model has the best performance, with F-beta 0.82 and 0.93 precision
- AdaBoost has a significant improvement in performance from logistic regression, and not a deep learning method
- Feed-Forward Neural network has a worse F-beta score than Adaboost, but a significantly higher precision score

Optimal parameters obtained by grid-search for Logistic Regress and Adaboost are as below:

- Logistic Regression
  - C = 1052631578.9474
  - Penalty = L1
- Adaboost
  - Number of estimators = 500
  - Learning Rate = 0.1

Cross-validation has applied to Logistic Regression and AdaBoost to obtain both the optimal parameter and the evaluation metrics. For the 2 neural networks, model loss is plotted against the epoch to make sure the data is fully trained and loss is minimized:

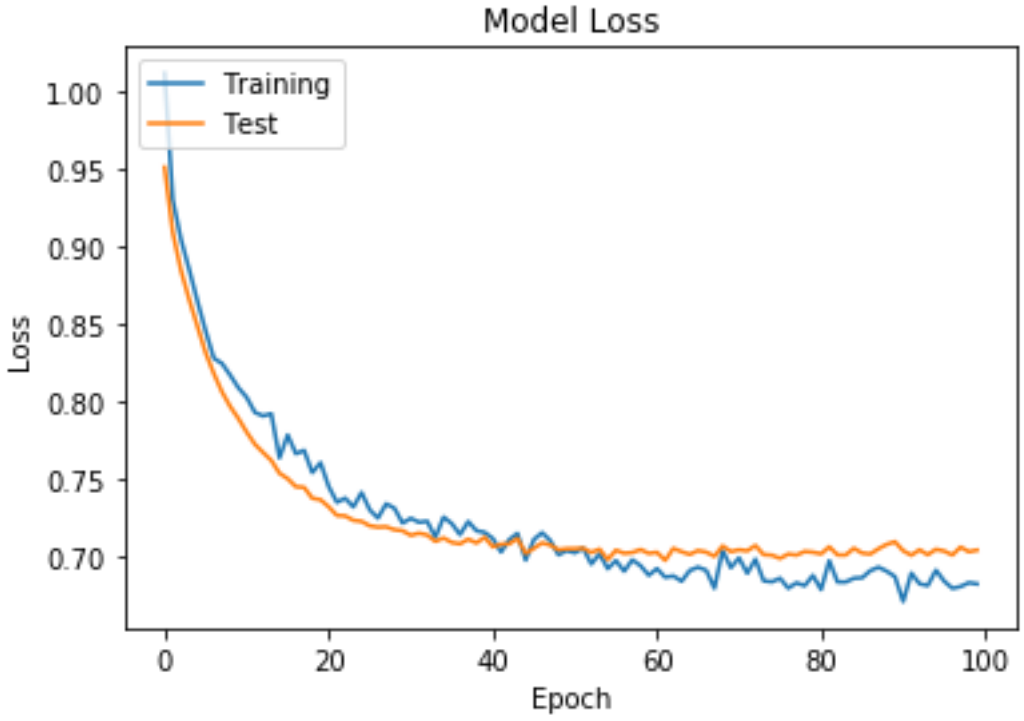

Figure 7: Model Loss for Feed-Forward Neural Network

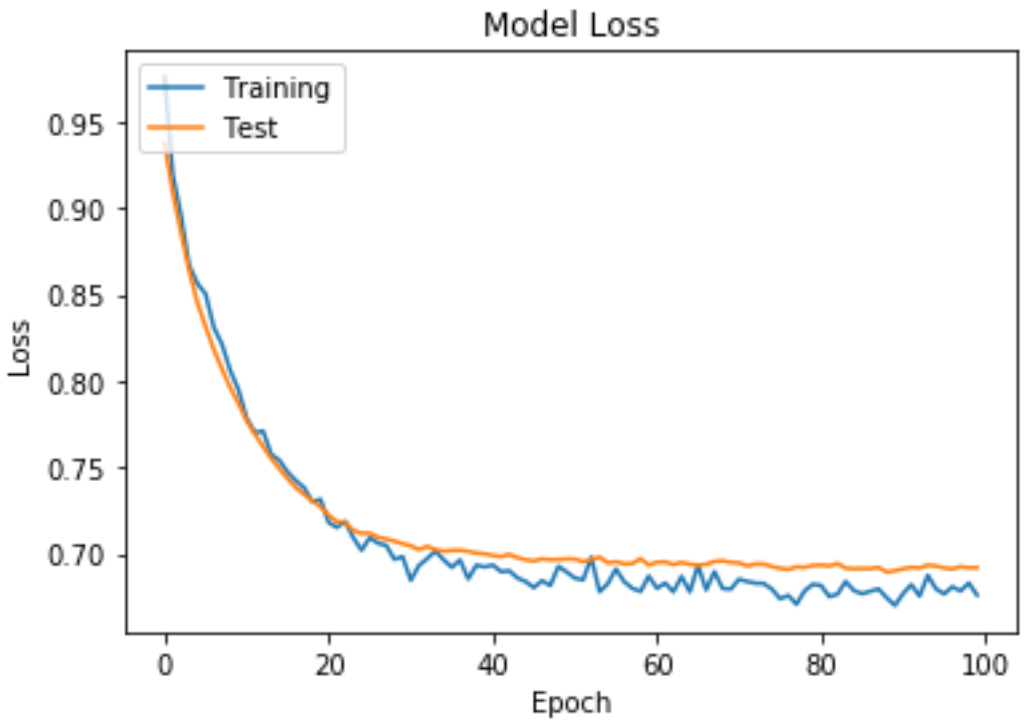

Figure 8: Model Loss for LSTM

The results are reasonably aligned to intuition due to the following reasons

- Logistic Regression is restricting the financial market to operate under a linear fashion. This is a rather strong assumption to the market dynamics, as non-linearity does exist under different market regimes. Therefore, the results in the prediction is worse than the benchmark model
- AdaBoost and Feed-forward Neural Network has better performance than the benchmark and logistic regression as it can capture extra complexity.
- Although AdaBoost has a higher F-beta score, Feed-forward Neural Network has a higher precision. This suggests Neural Network is better in predicting true positives within each label. This attribute is particularly important in developing a trading strategy as investable products within equities market are asymmetric. It is easier to go long than sell short. Therefore, a model with higher precision is more favorable.
- LSTM has the best performance out of the 4 models and the benchmark. Building on top of the feed-forward neural network, LSTM is able to further learn the time dependencies which is an important characteristic to be modeled within financial context.

Here are also some supplementary materials on the classification summary for Adaboost and Logistic Regression:

```
Split: 1
```

```
Training Size: 491; Test Size: 488
             precision    recall  f1-score   support

         Up       0.66      0.76      0.71       209
    Neutral       0.00      0.00      0.00        47
       Down       0.70      0.74      0.72       232

avg / total       0.61      0.68      0.64       488



Split: 2
Training Size: 979; Test Size: 488
             precision    recall  f1-score   support

         Up       0.66      0.69      0.68       213
    Neutral       0.00      0.00      0.00        51
       Down       0.64      0.75      0.69       224

avg / total       0.58      0.65      0.61       488



Split: 3
Training Size: 1467; Test Size: 488
             precision    recall  f1-score   support

         Up       0.72      0.78      0.75       216
    Neutral       0.00      0.00      0.00        46
       Down       0.74      0.84      0.79       226

avg / total       0.66      0.73      0.70       488
```

Figure 9: Characteristic Summary for Logistic Regression

From the above, table, the logistic regression is actually not too bad in predicting Up and Down labels. However, there are no predictions for Neutral label. This dragged the overall performance of the model. This has practical implications as we want to have Neutral labels where you do not need to trade every single day, which incur trading costs.

```
Split: 1
Training Size: 491; Test Size: 488
             precision    recall  f1-score   support

         Up       0.69      0.76      0.72       209
    Neutral       0.56      0.19      0.29        47
       Down       0.72      0.75      0.73       232

avg / total       0.69      0.70      0.68       488



Split: 2
```

```
Training Size: 979; Test Size: 488
             precision    recall  f1-score   support

         Up       0.67      0.69      0.68       213
    Neutral       0.69      0.22      0.33        51
       Down       0.67      0.75      0.71       224

avg / total       0.67      0.67      0.66       488


Split: 3
Training Size: 1467; Test Size: 488
             precision    recall  f1-score   support

         Up       0.71      0.75      0.73       216
    Neutral       0.71      0.26      0.38        46
       Down       0.74      0.78      0.76       226

avg / total       0.72      0.72      0.71       488
```

Figure 10: Characteristic Summary for AdaBoost

Similarly, for AdaBoost, the performance is boosted by having predictions over Neutral labels.

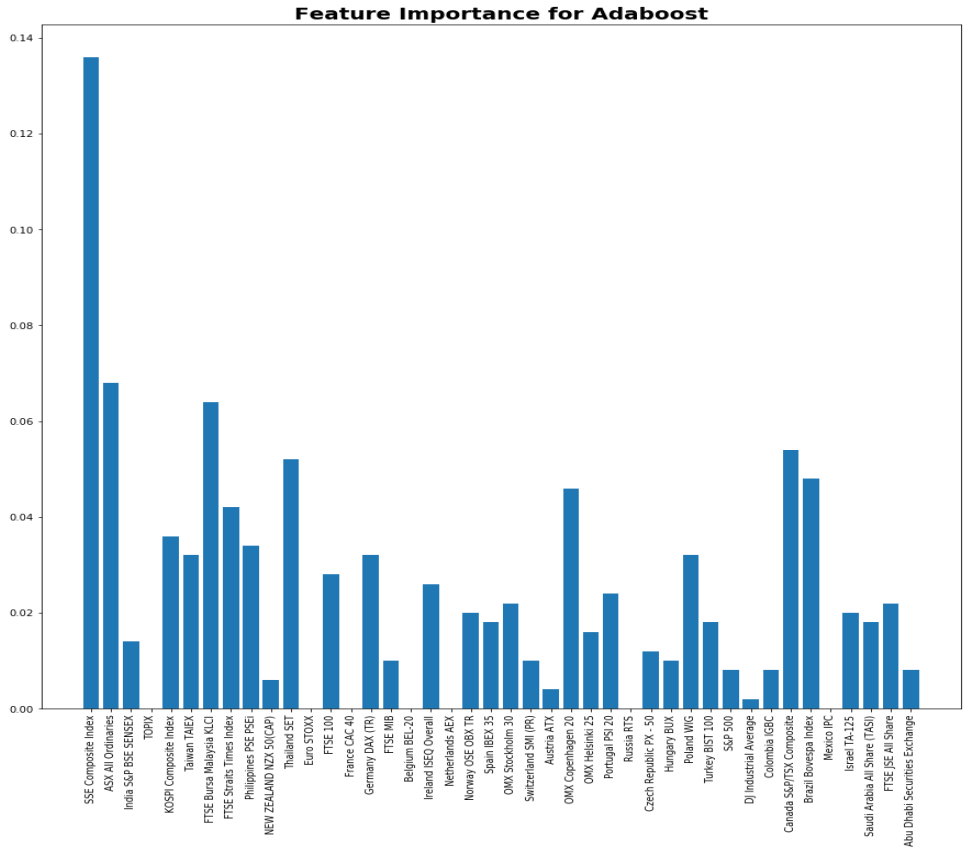

Figure 11: Feature Importance Summary for AdaBoost

As a cross-check for AdaBoost, I have generated the feature importance chart. From this chart, Asia Pacific indices have the most importance in predicting Hang Seng Index, with the most significant feature being the Shanghai Composite index. This rather makes sense as Hang Seng Index constituents contain a large number of Chinese H-shares, which are predominantly correlated to the China equity market performance. Also, companies within Hang Seng Index also have subsidiaries and investments within Asia. Therefore I should expect a higher importance is observed.

Japan has no significance, mainly due to the fact the Japan is a rather closed economy therefore not much importance over Hang Seng.

Europe has mild importance, with UK, Germany and Denmark being the top significant features. There are a lot of companies within Hang Seng Index has business or entities within UK (e.g. HSBC, Cheung Kong) so this is rather expected. Germany exports machines and industrial components to Hong Kong, so this is also expected. Demark stands out is rather interesting, but pursuing the reason is out of the scope of this project.

Finally, North America markets have lower significance than I expected. For example S&P 500 has only 0.01 importance. This is rather interesting because as the world's largest economy, it should exhibit more influence to Hong Kong. Also companies within Hang Seng do have business in US as well. Again, this remains an open question to further explore, but the overall result from this chart are in line with economic intuitions.

Justification

During the training, as described in other sections, cross-validation techniques are used for AdaBoost and Logistic Regression. For Neural Networks, 100 epochs are used to fully train the data. The results are reasonably stable, and the results are in-line with expectations, that all models except Logistic Regression are better than the benchmark model, while LSTM has the best performance due to the fact that it can capture time-dependencies within the financial time-series.

V. Conclusion

Reflection

Throughout this project, I have selected 4 machine learning algorithms, namely Logistic Regression, AdaBoost, Feed-Forward Neural Network and LSTM, to build a model to predict Hang Seng Index's next day movement direction, based on other 42 closing price of the global indices.

The data is first cleaned, preprocessed before serving as an input to be trained. Cross-validation techniques are used to obtain more stable and reliable results for AdaBoost and Logistic Regression, followed by a grid-search to tune the hyperparameters. 100 Epochs are run for the 2 Neural Network models to make sure the optimal model is obtained.

Main challenge of this project would be dealing with the time-series nature of the data. Time-series data often impose issues in classical statistical analysis as the data are often autocorrelated, ordered, and noisy. Special attention has to be made to avoid look-ahead bias in predicting the time-series data, and cross-validation techniques have to be carefully chosen.

The results obtained in this project are mainly in-line with my expectation, and what is observed based on my personal financial markets experience. The results could then be further carried into a production settings, where a trading strategy can be developed based on the model.

Improvements

There are several ideas that could potentially enhance further the models but have not explored within the scope of this project:

- **Extra Feature/ Feature Engineering** – in this project I have only used day close. Would adding more features could help in improvement? Perhaps adding daily high, low, mid, volume, or indices from other asset classes could help improving.
- **Cross-Validation techniques** – in this project I have only used 1 cross-validation technique. However, in practical settings, how could we incorporate streams of incoming new data is a key. Perhaps further can try a rolling-window cross-validation, where a fixed window of data is used to train (say 5 years) and predict on the next day's features. Or running extra epochs when a new data comes with to "append" with the existing model.
- **Batch Size Tuning** – In this project I have not tuned the batch size prediction as we are predicting only 1-day movement. However, we can perhaps try to predict the 1-week return, or 1-month return based on the daily index inputs.
- **Tuning labels threshold** – The current labels cutoff have no justification. What threshold level should be chosen is perhaps another question to ask before fitting the model. This would require some practical consideration as you might want to set a threshold such that the gain you have on a correct prediction covers your actual trading cost.