Stock Performance Predictor

Kaushal Patil
kaushal.p@ahduni.edu.in

Arpitsinh Vaghela
arpitsinh.v@ahduni.edu.in

Prachee Javiya
prachee.j@ahduni.edu.in

Vrunda Gadesha
vrunda.g@ahduni.edu.in

***Index Terms*—Financial Machine Learning, Quant Trading, Classification/Clustering, Boosting**

***Abstract*—This project aims to study and test fundamental and technical features to create a model that can classify/-cluster stock tickers as BUY SELL or HOLD signals. Using algorithms such as Logistic Regression, Linear Regression, K-nearest Neighbor, Decision Trees to create an accurate classifier/clustering model and use Bagging and Boosting techniques to achieve better performance.**

I. Introduction

Predicting the stock price trend by interpreting the seemly chaotic market data has always been an attractive topic to both investors and researchers. Among those popular methods that have been employed, Machine Learning techniques are very popular due to the capacity of identifying stock trend from massive amounts of data that capture the underlying stock price dynamics. In this project, we have applied numerous machine learning techniques to build a model for stock classification - BUY, SELL or HOLD.

Stock market decisions have a huge impact on investors. It is difficult to analyse every aspect manually in order to make the best prediction. There have been numerous researches in this field. [1] is based on the approach of predicting the share price using Long Short Term Memory (LSTM) and Recurrent Neural Networks (RNN) to predict the stock price on NSE data using various factors such as current market price, price-earning ratio, base value and some miscellaneous events.

II. Implementation

Libraries used : Matplotlib, Pandas, Seaborn, Scikit learn and numpy

A. Data Collection

The training data used was collected from US stock fundamentals database. The data consists of 63,547 entries and 30 fields. The data is divided into 4 quarters for each fiscal year.

	Ticker	SimFinId	Currency	Fiscal Year	Fiscal Period	Report Date	Publish Date	Restated Date	Shares (Basic)
0	A	45846	USD	2010	Q3	2010-07-31	2010-10-06	2010-10-06	347000000.0
1	A	45846	USD	2010	Q4	2010-10-31	2010-12-20	2011-12-16	344000000.0
2	A	45846	USD	2011	Q1	2011-01-31	2011-03-09	2011-03-09	347000000.0
3	A	45846	USD	2011	Q2	2011-04-30	2011-06-07	2011-06-07	347000000.0
4	A	45846	USD	2011	Q3	2011-07-31	2011-09-07	2011-09-07	348000000.0

Fig. 1: Dataset

B. Data Cleaning and pre-processing

The entries in the raw dataset that contained "bad" values were set to 0. The columns which did not contribute in learning were dropped.

```
clean_fundamen2=clean_fundamen.dropna(subset=['Price_Future','Price_Present'])
clean_fundamen=clean_fundamen.fillna(0)
```

For Data Visualization, correlation heatmap was plotted:

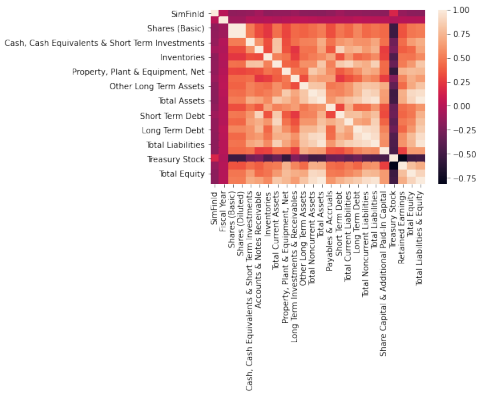

Fig. 2: Correlation heatmap using Seaborn

C. Data Labelling

The data entries were classified into 3 labels, 0,1 and 2 for Buy, Sell or Hold respectively. Present price and future price are compared for every quarter with a difference tolerance of 3%. Three additional columns - Price present, price future and labels are added into the dataset.

D. Algorithms

1) Principal Component Analysis: We performed 2 component PCA and plotted colorcoded datapoints based on their classes to see how the data is distributed.

	Ticker	Price_Present	Price_Future	Label
0	A	32.67	41.48	0
1	A	40.63	43.95	0
2	A	46.05	48.71	0
3	A	47.7	35.69	1
4	A	35.69	36.66	2
5	A	33.46	45.62	0

Fig. 3: Labelled dataset

2) Logistic Regression: We used logistic regression library from Scikit Learn. We normalised the data by creating our own function and with Standard Scalar library. We also tried training with unnormalized dataset.

```
clean_fundamen[features_to_normalize] =
    clean_fundamen[features_to_normalize].
    apply(lambda x:(x-x.min()) / (x.max()-x.
    min()))
```

3) Random Forest Classifier: RandomForestClassifier model is used from Scikit learn library. We reach upto a depth of 32, with 2 steps.

```
Max_Depth:  2  Score Train: 0.4718120464809459
Max_Depth:  2  Score Test: 0.4696095601112599
Max_Depth:  4  Score Train: 0.4781803420104531
Max_Depth:  4  Score Test: 0.4746574636859998
Max_Depth:  6  Score Train: 0.4894707464352768
Max_Depth:  6  Score Test: 0.4784691459771299
Max_Depth:  8  Score Train: 0.5114172629015071
Max_Depth:  8  Score Test: 0.48192026372720714
Max_Depth:  10  Score Train: 0.5507941340640382
Max_Depth:  10  Score Test: 0.4859379828989389
```

Fig. 4: Train and test scores for RFC

4) K means clustering with 2 component PCA: Partitioning n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

5) K nearest Neighbours: The KNN algorithm assumes that similar things exist in close proximity. For stock performance prediction KNN-Classification is suitable. We found MSE using KNN Algorithm and defined elbow function to find the appropriate value of K.

6) Model Stacking: This algorithm involves building a meta-model based on prediction outputs of other different models. In our implementation, we have stacked lasso, random forest and gradient boosting algorithms. Metrics for each algorithm:
Logistic Regression - Max iteration = 1000
Random Forest - max depth 18, step 1
Gradient Boosting - max depth 18, step 1

7) XG Boosting: XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. We have used XGBClassifier library and have applied this on numeric fields in our dataset. Input values are standardized using StandardScaler library. Test train split - 0.1

III. Results

Random Forest Classifier
Max depth 18
Score Overall: 0.738117903039369
Overfit
Max depth 30
Score Overall: 0.8245903311348337

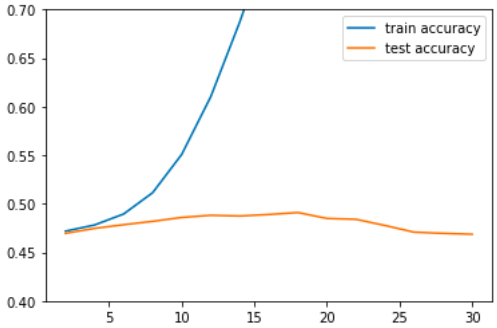

Fig. 5: Train and test accuracy graph for RFC

K Nearest Neighbours : From the graph, we can see the value of K = 11 is the most appropriate.

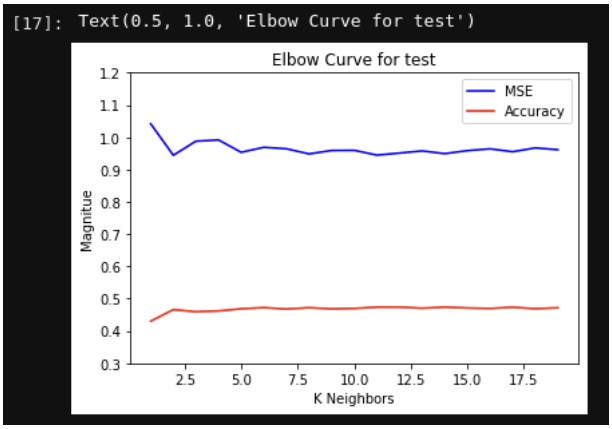

Fig. 6: Train Accuracy

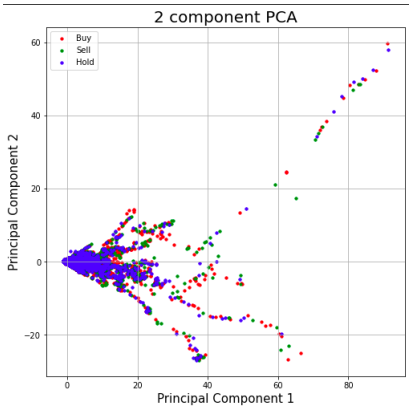

Fig. 7: 2 component PCA

IV. Conclusion

The max test accuracy we have obtained so far is 50%, in XGBoost and Random Forest, however the y labels are highly unbalanced i.e, there is a huge difference in the number of each label, Confusion matrix, precision, recall and f1 score could be better metrics to really understand the model performance.

References

[1] B. Jeevan, E. Naresh, B. P. V. kumar, and P. Kambli, "Share price prediction using machine learning technique," pp. 1–4, 2018.