Pairs Selection and Trading in Indian Equities

Saket Joshi SAKET_JOSHI@G.HARVARD.EDU

Nishtha Sardana NISHTHASARDANA@G.HARVARD.EDU

Kareema Batool KAREEMABATOOL@G.HARVARD.EDU

Nikhil Nayak NNAYAK@G.HARVARD.EDU

Abstract

We investigate the state-of-the art methods and empirically evaluate different statistical and ML model for securities pairs selection discovery. We further describe ra baseline strategy which we use to benchmark the different models and implement such a strategy for trading on the Indian equity markets (NSE & BSE).

1. Introduction

Pairs trading is a market-neutral strategy where we use statistical techniques to identify two stocks that are historically highly correlated with each other. When there is a deviation in the price relationship of these stocks, we expect this to be mean reverting and buy the underperforming stock and simultaneously sell the outperforming one. If our mean-reversion assumption is valid then prices should converge to long term average and trade should benefit. However, if the price divergence is not temporary and it is due to structural reasons then there is a high risk of losing the money.

2. Motivation

Pair trading is a strategy for hedging risk by opening opposing positions in two related stocks, commodities, or other derivatives. The goal is to identify relationships between securities to guide trading strategies and help mitigate risks.

3. Updated Problem Statement

Investigate the application of machine learning/Artificial Intelligence methods to find statistical arbitrage opportunities in the stock market using pair trading strategy.

4. Data

We have taken NIFTY-100 data for the past 3 years on a day level basis for our analysis. However, we have considered the companies who have complete data for our selected tenure and escaped any that were listed. For each of these companies, we have taken the 'closing value' for our further analysis. However, we have eliminated businesses that lack data for the chosen tenure, leaving only the top 88 stocks. For these 88 equities, we use the closing price on a daily basis.

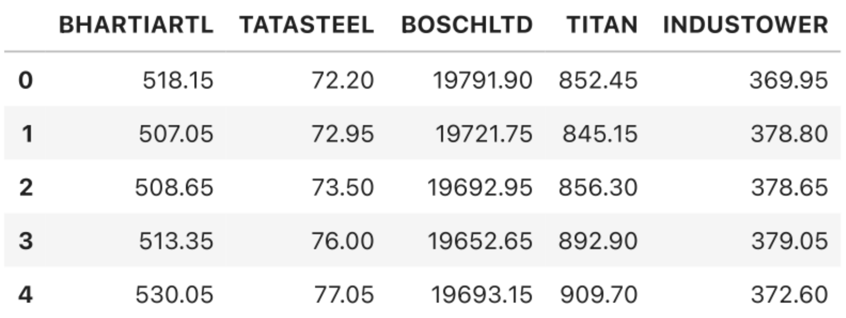
For each hourly-candle timestamps, we have the standard OHLC(Open-High-Low-Close) values for each stock. We show a sample of the raw data below. The final dataset has been shown in the table below:

4.1 Dataset collection methodolody

We have collected our data from NIFTY-100 data. Our collected data is from the past 3 years that we have compiled on daily basis. We had our raw data in form pf multiple 'csv' files that we later on compiled and merged using different data engineering methods in Python. After compiling our data in the form of one file we further applied other EDA methods.

4.2 Exploratory data analysis

Our data initially had stocks whose information was not provided for our selected tenure, so we dropped all such stocks and chose 88 stocks for which we have the data for our selected period. Furthermore, we have dealt with missing values in our dataset by imputing them with the mean. Finally, we have applied different exploratory data analysis methods such as the finding the correlation and cointegration between the pairs of stocks. A detailed description of these methods are provided in the following sub sections. Our final dataset is shown in the table below:



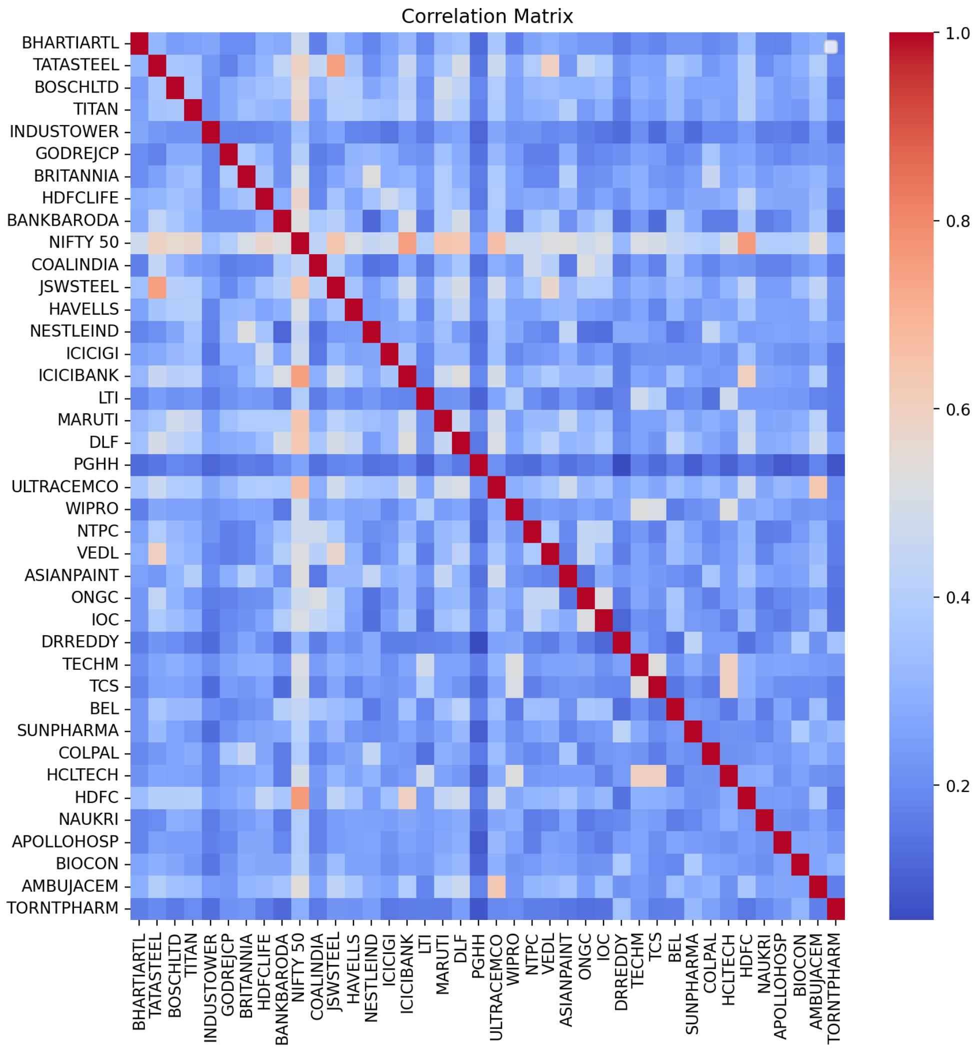
	BHARTIARTL	TATASTEEL	BOSCHLTD	TITAN	INDUSTOWER
0	518.15	72.20	19791.90	852.45	369.95
1	507.05	72.95	19721.75	845.15	378.80
2	508.65	73.50	19692.95	856.30	378.65
3	513.35	76.00	19652.65	892.90	379.05
4	530.05	77.05	19693.15	909.70	372.60

4.2.1 Correlation and co-integration between stock pairs

As a first step, we start by exploring the highly correlated pairs in our dataset of 88 companies.

We have leveraged the Pearson Correlation Coefficient to gain a general understanding of the relationship between these companies before trying to explore and locate cointegrated stocks.

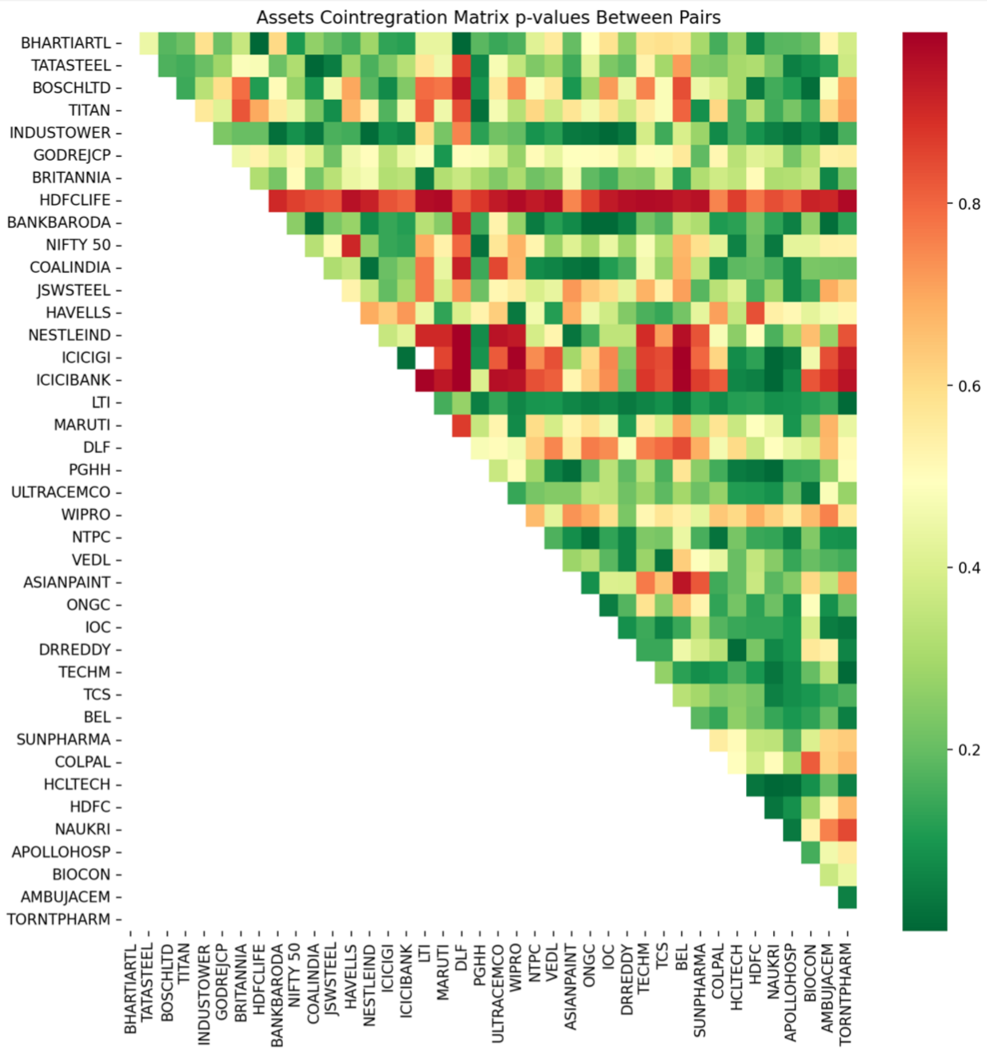
Pearson correlation coefficient varies between +1 to -1 and is a linear measure of the relationship between two variables. The value +1 indicates a strong positive correlation, zero indicates no relationship, and -1 indicates a strong negative relationship.



We can observe from the below generated heatmap that there are multiple pairs with a strong positive correlation.

We have further defined a function to calculate p-values of the cointegration test for each pair, to generate a heatmap. If the p-value is less than 0.05, the null hypothesis can be rejected and the cointegration between the two time series with distinct symbols.

We can see in the below heatmap that there are many pairs with a p-value of less than 0.05. This means that for these pairs we can reject the null hypothesis and they can be cointegrated.



5. Baseline Model

5.1 Baseline Model

Pair trading is one of the most popular trading strategies since 1980 for finding statistical arbitrage opportunities in the stock market. The logic behind pairs trading is to trade pairs of stocks belonging to the same industry or having similar characteristics, such that their historical returns move together and are expected to continue to do so in the future. In this project, we will use machine learning/deep learning methods to find statistical arbitrage opportunities in the stock market using pair trading strategy. The baseline model recognizes correlated pairs using clustering methods.

Approach in the baseline model for pairs trading is to apply PCA for dimension reduction on a large set of features (different stocks) in order to ease computation of unsupervised clustering algorithms like DBSCAN/t-SNE for clustering of similar stocks. Following which, we use cointegration tests to extract all possible combinations of stocks in each cluster that are within 5% significance level (p-value is 0.05). These pairs of stocks are the final predictions of the model which are correlated enough for pair trading.

5.2 Specifications of the baseline model

1. PCA is a mathematical procedure that transforms a large number of variables into a smaller number of uncorrelated variables called principal components. In the baseline model, PCA is used to reduce daily stock prices of 88 companies to 50 variables while trying to keep as much variance as possible. Each of the resulting principal components can be seen as representing a risk factor, and the stocks will be clustered based on these components to find the stock pairs.

2. Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm. It is a density-based clustering non-parametric algorithm. We preferred DBSCAN over K-Means because it has a couple of important advantages. Specifically, DBSCAN does not cluster all stocks, i.e. it leaves out stocks which do not neatly fit into a cluster, and the number of clusters does not need to be specified.

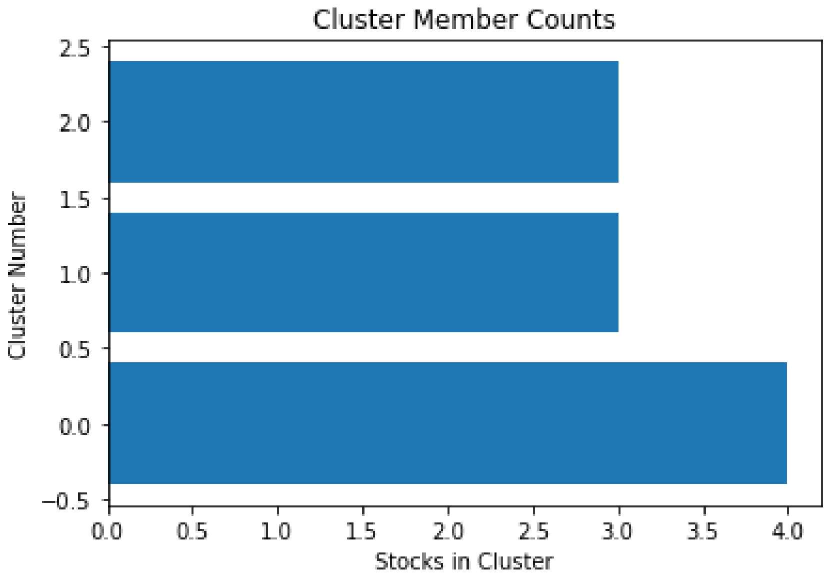
3. t-Distributed Stochastic Neighbor Embedding (t-SNE) is a nonlinear dimensionality reduction technique. Once the data is clustered using DBSCAN, we use t-SNE to visualize the high dimensional data and its clusters in a two-dimensional graph.

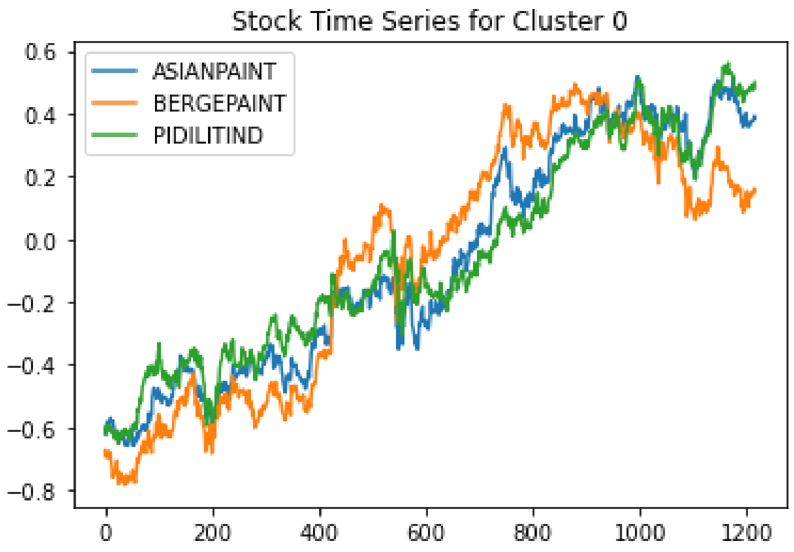
5.3 Results of the baseline model and updated problem statement

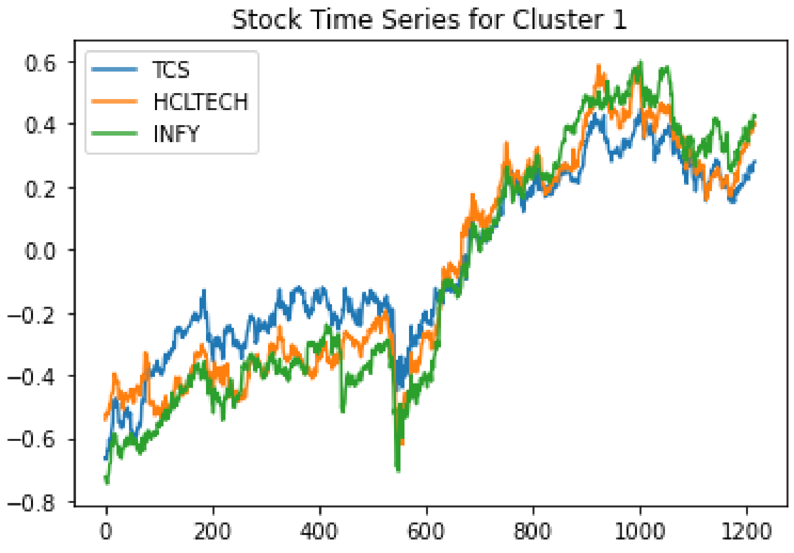
We used 1216 rows of data i.e. 1216 consecutive days of closing price for 88 companies. We used PCA with 50 components. Following which 3 clusters were discovered using DBSCAN algorithm. t-SNE is mainly used for visualization in 2 dimensions. Following 3 clusters were formed:

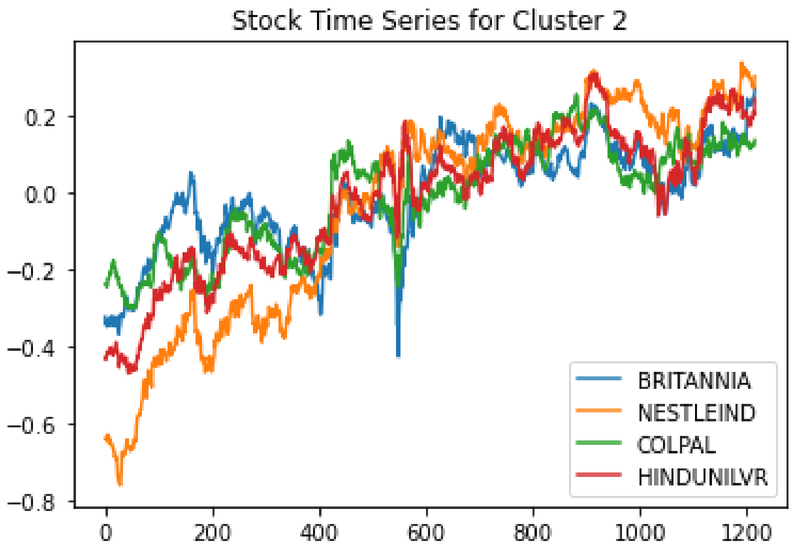
- Cluster 0 - BRITANNIA, NESTLEIND, COLPAL, HINDUNILVR
- Cluster 1 - ASIANPAINT, BERGEPAINT, PIDILITIND
- Cluster 2 - TCS, HCLTECH, INFY

Pairs within each cluster were investigated for correlation using cointegration. Cointegration tests identify scenarios where two or more non-stationary time series are integrated together in a way that they cannot deviate from equilibrium in the long term. The tests are used to identify the degree of sensitivity of two variables to the same average price over a specified period of time. Thus, it can be used for finding stock pairs that are correlated in the long term. Among the 88 companies 3 pairs were found to be significant using a p-value of 0.05. Pairs found were - ('TCS', 'HCLTECH'), ('TCS', 'INFY'), ('HCLTECH', 'INFY').


Cluster Member Counts
Cluster Number
Stocks in Cluster
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


Stock Time Series for Cluster 0
ASIANPAINT
BERGEPAINT
PIDILITIND
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
200
400
600
800
1000
1200





5.4 Updated Problem Statement

1. The baseline model tells if two stock prices are closely related, but in the next milestone we intend on predicting the value/gains of investing in these pair stocks in a specified long term interval.

2. We will investigate different base estimators and clustering based architecture to minimize hypothesis testing problem, which occurs when several tests are done due to a large number of stock pairs.

3. We will define and evaluate strategies to use the correlation / cointegration pairs identified for trading. One of such strategies will be mean-reversion. These will be baselined and compared against different pairs identification architectures.

6. Method and Approach

For constructing a Pairs Trading strategy, the first task is to find valid, eligible pairs which exhibit unconditional mean-reverting behavior. One simple approach is to iterate through all stock pairs in the universe of stocks, but this would not be computationally feasible. Instead, PCA can be used to compress the daily stock price data into several components, which will be regarded as price movement features for a stock. Then, we incorporate feature engineering for each stock by combining the price movement features generated by PCA with the features from economic prior knowledge. After we select the candidate pairs based on their correlation, we apply multiple machine learning algorithms to predict the spread between each pair, in an effort to catch the trading signal whenever the spread exceeds some predefined threshold. The particular machine learning methods we incorporate in our analysis are discussed below.

6.1 PCA

A large number of variables are reduced to a smaller set of main components by the mathematical process known as principal component analysis (PCA). PCA will be used in this study to condense 500 daily stock prices into 50 variables while preserving as much variance as possible. The equities will be grouped based on each of the resulting primary components, which can be thought of as representing a risk factor.

6.2 DBSCAN

Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm proposed by Martin Ester, Hans-Peter Kriegel, Jorg Sander and Xiaowei Xu in 1996. It is a density-based clustering non-parametric algorithm which can be described in the following steps:

- 1. Find the points in the ϵ neighborhood of every point, and identify the core points with more than min P_{ts} neighbors, where min P_{ts} is a parameter to be tuned.
- 2. Find the connected components of core points on the neighbor graph, ignoring all non-core points.
- 3. Assign each non-core point to a nearby cluster if the cluster is an ϵ neighbor, otherwise assign it to noise.

Compared with K-Means, DBSCAN has advantages in our use case. Specifically, DBSCAN does not cluster all stocks,i.e. it leaves out stocks which do not neatly fit into a cluster, and the number of clusters does not need to be specified.

6.3 t-SNE

Once the data is clustered, we need a way to visualize the high dimensional data and its clusters into a two-dimensional graph. One approach to this visualization is using the nonlinear dimensionality technique known as t-Distributed Stochastic Neighbor Embedding (t-SNE).

We denote the original high dimensional data set as X, with each data point as x_j, and denote the mapped (transferred) low dimensional data set as Y , with each map point as y_i. Then, let $|x_i - x_j|$, $|y_i - y_j|$ be the distances among the original data points and map data points respectively. To maintain the relative structure of the data, if two data points in the original space are close, the corresponding map data points also need to be close. We can now define the conditional probability of the data points as,
$p_{j|i} = \frac{exp((-(|x_i - x_j|)^2)/(2\sigma_i^2))}{\sum exp((-|x_i - x_k|^2)/(2\sigma_i^2))}$ The above equation means that the probability distribution is constructed such that similar objects have a high probability of being picked, while non-similar objects have a low probability of being chosen. Then, we define the joint probabilities, $p_i j = \frac{p_{j|i} + p_{i|j}}{2N}$ and define a similar matrix of map points, $q_{ij} = \frac{exp(|y_i - y_j|^2)}{\sum exp(|y_i - y_k|^2)}$ The objective is to minimize the distance between the two probabilities. This process is done through the Kullback-Leiber divergence with a gradient descent, $KL(P||Q) = \sigma_{i,j} p_{ij} log(\frac{p_i j}{q_i j})$ The above equation tells us that if two map points are distant to each other, while the corresponding data points are not, they will be drawn together. The process iterates until a final mapping is procured and equilibrium is attained.

6.4 Logistic Regression

We have used Logistic regression as our one of the Machine learning models, which yields quite satisfactory results. We have used Sigmoid function since it works best for binary classification. The Loss function that we have used here is the Negative log likelihood which works best with Sigmoid function. We have used different hyperparameters with our Logistic regression classifier to finetune our hyperparameters according to our use case.

6.5 Random Forest

Before presenting the random forest, we first introduce the bagging algorithm. Suppose there are B independent training sets from the same distribution. The learning algorithm produces B decision functions: $f_1(x), f_2(x), f_3(x), ..., f_B(x)$, then the average prediction function is defined as $f_{avg} = \frac{1}{B} \sum f_b$. In practice, however, the training sets are usually not independent, so bootstrapping is used to generate samples $D^1, ..., D^B$ from original data D. The bagged decision function is then a combination of different functions. $f_{avg} = Combine(f_1(x), f_2(x), ..., f_B(x))$. The main idea behind random forest is to reduce the variance of the decision tree models without incresing bias by using bagging. Random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the prediction accuracy and control over-fitting.

6.6 Gradient Boosting

Gradient boosting is typically used with decision trees of a fixed size as base learners. Generic gradient boosting at the m-th step would fit a decision tree $h_m(x)$ to pseudo residuals. Let J_m be the number of leaves. The tree partitions the input space into J_m disjoint regions $R_{1m}, R_{2m} ... R_{jm}m$ and predicts a constant value in each region. We specify two hypothesis spaces of Gradient Boosting:

- base hypothesis space: H, which consists of real-valued functions

- combined hypothesis space: $F_M = \sum(v_m h_m(x)|v_m \in R, h_m \in H$

Using the indicator notation, the output of $h_m(x)$ for input x can be written as $h_m(x) = \sum b_{jm} 1_{Rjm}(x)$ where b_{jm} is the value predicted in the region R_{jm}. Given data D = ((x_1, y_1), ...,(x_n, y_n)), fitting a gradient boosting model is equivalent to choosing v_1, ..., v_M R which represent our lags of return changes, and h_1, ..., h_M H, which represents the decision tree basis functions. The ERM objective function is defined as $(J(v_1, ..., v_M, h_1, ..., h_M)) = \frac{1}{N} \sum l(y_i, h_1(x), ..., h_M(x))$

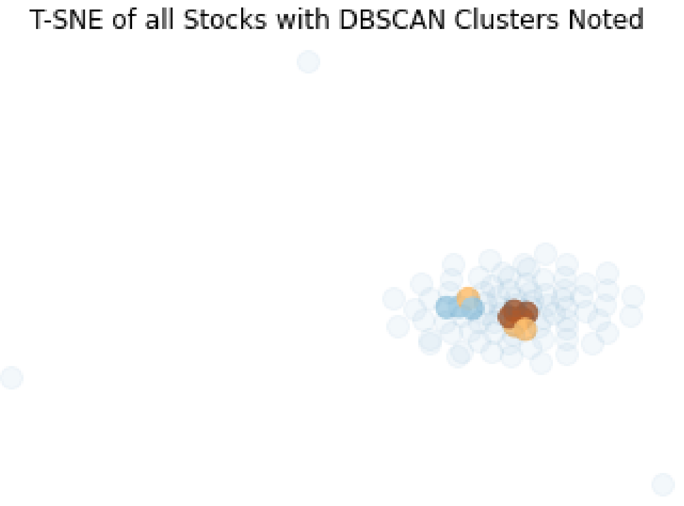
6.7 LSTM

RNN is usually used for training time series data, but a significant shortcoming of RNNs is their difficulty in handling "long-term dependencies.". Hochreiter and Schmidhuber introduce LSTM in 1997, a model that is explicitly designed to avoid this long-term dependency problem. The key to LSTM is the cell state, which is the main output. Next, h_t in this model can be seen as filtered version of the cell state. The first step for LSTM is the "forget gate layer", which determines the old information that should be kept. It is built by f_t, where f_t is defined as, $f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$ The next step is to determine the new information for the cell state. This part is built up by i(t) and C˜(t), $i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tilde{C_t} = tanh(W_c[h_{t-1}, x_t] + b_C)$ The new cell state is obtained by, $C_t = f_t C_{t-1} + i_t \tilde{C_t}$ The last step is to generate the new hidden layer, $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) h_t = o_t * tanh(C_t)$ In particular, we use LSTM as a time-series binary classifier, so the objective function is set to be the log-loss. Also, the more efficient ADAM algorithm is used for optimization.

7. Results

7.1 Pair Selection

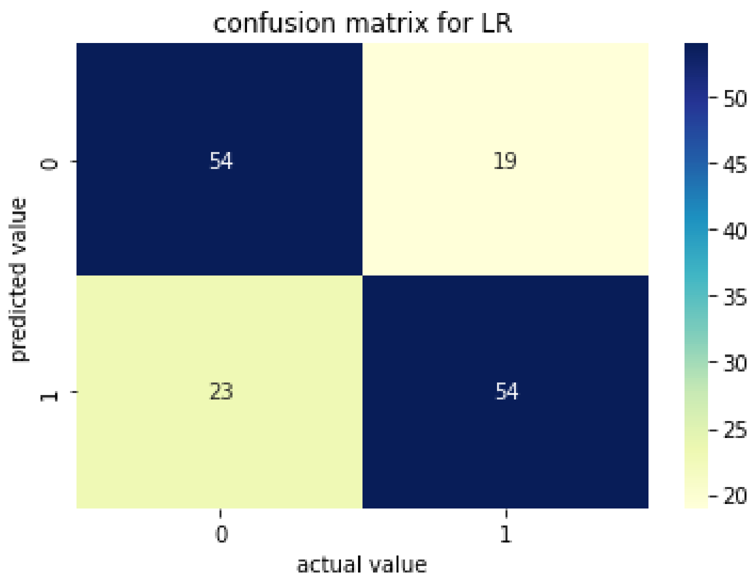
In our exploratory data analysis, we have used the concept of co-integration to find the correlation between different stock pairs. We have then used PCA, t$_SNEandDBSCANtoselectthecandidatepairsof$



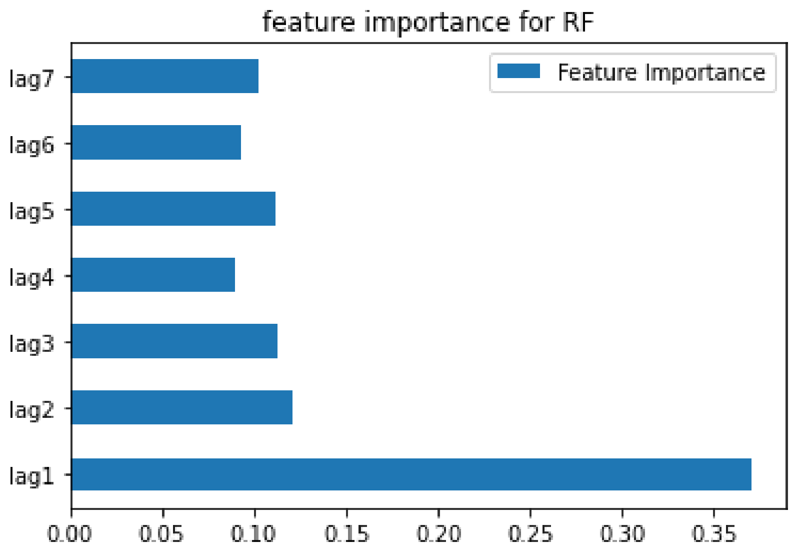
7.2 Prediction Models

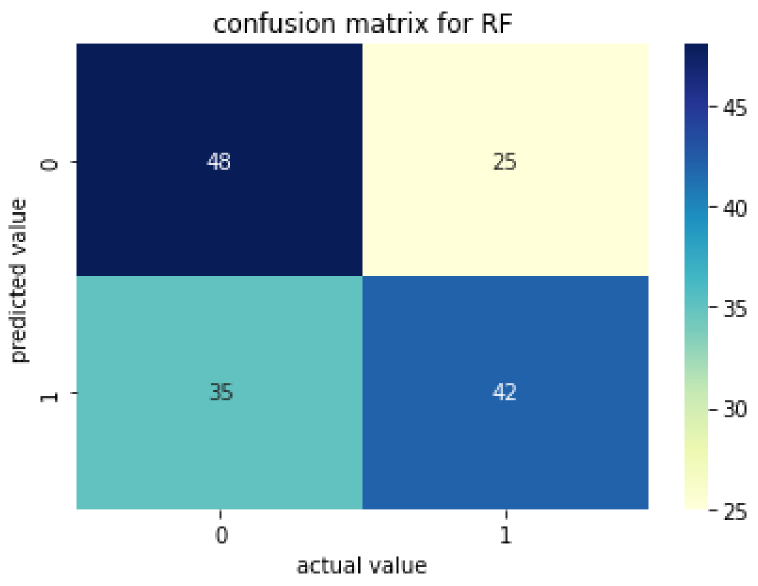
We have implemented different Machine learning models as listed above on our dataset. We first applied different exploratory data analysis methods and feature engineering, then we applied PCA and t-SNE methods to reduce the dimension of our dataset without compromising on capturing the most relevant features. Afterwards, we applied our different models to get the prediction.

1. Logistic Regression: We have used parameter grid to get the most fine tuned hyper parameter. The accuracy of our classifier with regularization strength of 0.7 is 0.74.

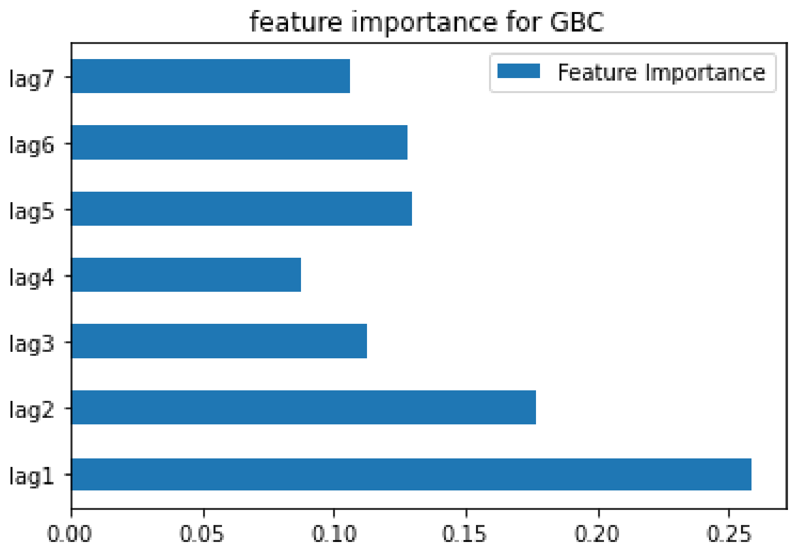


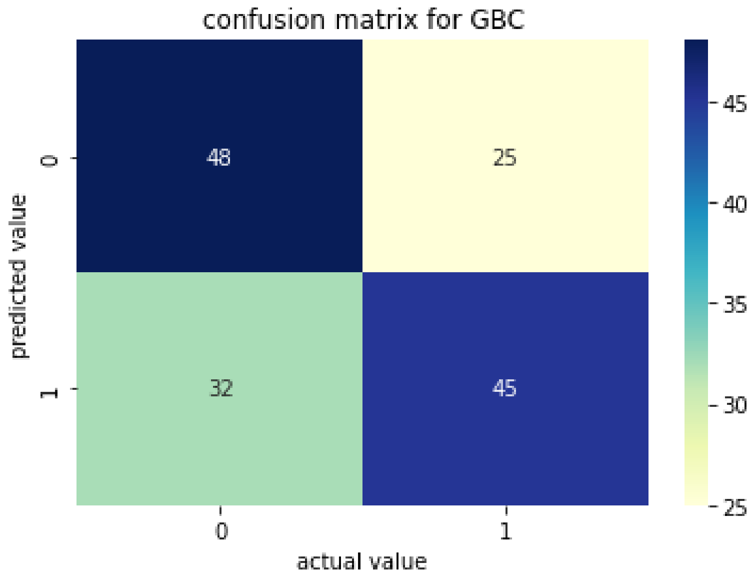
2. Random Forest: Our Random Forest classifier gave best results with splitting criterion 'Entropy'. The accuracy of our classifier is 0.61.



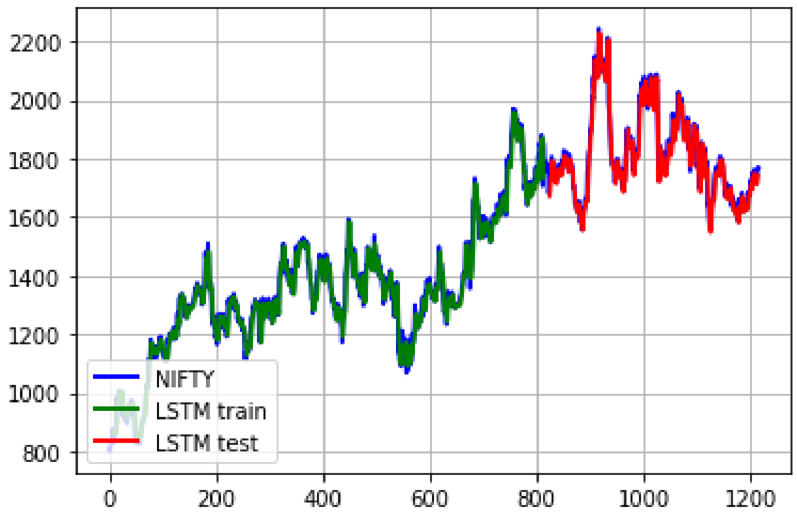


3. Gradient Boosting: Our Gradient Boosting classifier performed best with '50' estimators. The accuracy of our classifier is 0.67.



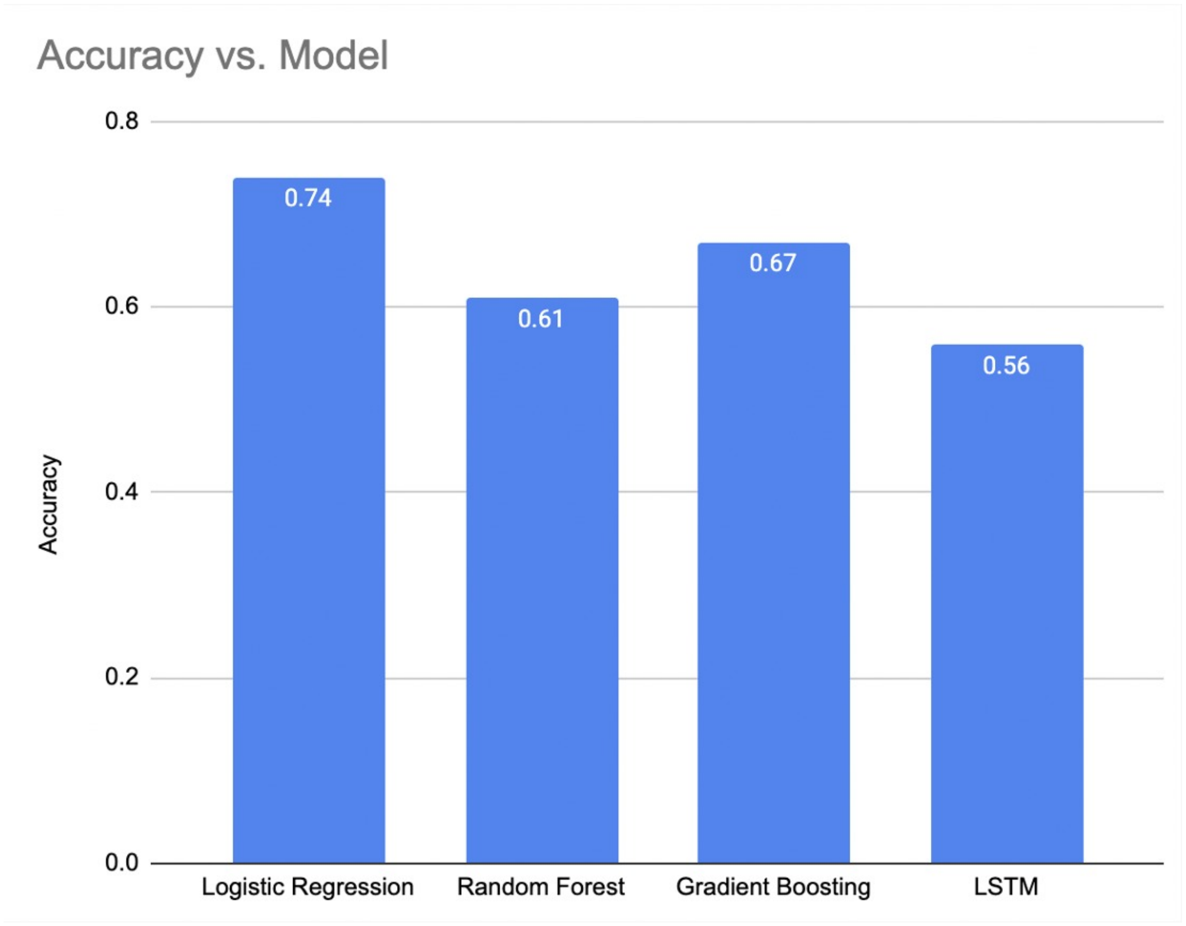


4. LSTM: We have used this Deep learning approach because our data has relative insensitivity for length of lag. The LSTM model gained the least accuracy of 0.56.



7.3 Comparison of different models

Among our implemented models, Logistic Regression outperforms all other models because it is less complex model thus it does not lead to overfitting. Random Forest classifier and Gradient boosting classifier yield quite similar F1- score in the range of 0.6, whereas LSTM yields the least accuracy among all. This comparison is given in the following graph.



8. Conclusions

According to our research, pair trading remains a viable trading strategy—but only when machine learning techniques are used. But prior to these techniques, we fine tuned our data according to our use case by various explotary data analysis methods. We also gained expertise in dimensionality reduction, supervised learning, and training neural networks on time-series data. Future research can study other neural networks with various layer counts, neuronal densities, and learning parameters to potentially produce better outcomes.

9. Future Work

Finally, there are numerous options to scale up our project. By feeding the model extra data from additional sources, such as current news, twitter feeds, etc using NLP, the model's long-term predictive power can be increased. As an alternative to the existing statistical method of stock selection, we might also investigate deep learning techniques for detecting the co-integrated pairs.

9.1 References

1. http://stat.wharton.upenn.edu/~steele/Courses/434/434Context/PairsTrading/PairsTradingGGR.pdf

2. Statistical Arbitrage by Pair Trading using Clustering and Machine Learning - https://israeldi.github.io/coursework/EECS545/545_Final_Project.pdf,