DELHI TECHNOLOGICAL UNIVERSITY

Pairs Trading with Machine Learning

SANDESH JAIN - 2K18/IT/109

SOURABH BHARDWAJ - 2K18/IT/121

SUBMITTED TO - PROF. SEEMA SINGH

Contents

OVERVIEW

Brief Intro

Dataset

Data generation
PCA decomposition

Visualization

DBSCAN clustering
Matplotlib

Trading Logic

z-score

Introduction

Part 01

Background

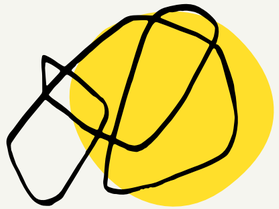
Overview of the project

Pairs Trading’ is an investment strategy used by many Hedge Funds. Consider two similar stocks which trade at some spread. If the spread widens short the high stock and buy the low stock. As the spread narrows again to some equilibrium value, a profit result. This project provides an analytical framework for such an investment strategy.

We propose a mean reverting Gaussian Markov chain model for the spread which is observed in Gaussian noise. Predictions from the calibrated model are then compared with subsequent observations of the spread to determine appropriate investment decisions.

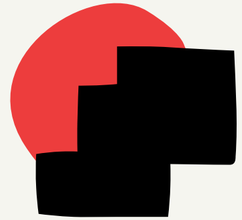
The methodology has potential applications to generating wealth from any quantities in financial markets which are observed to be out of equilibrium.

The Problem



Pair selection:

What are the characteristics of a good pair of stocks and how can we systematically quantify it?



Trading logic:

What are the optimal entry and exit levels? How can we limit the downside in the event if the trade goes wrong?

Objectives

What we want to achieve

Creation Data using yfinance making dataframe from it, visualization of data using clustering

Implementing existing pairs trading strategies (distance method, cointegration method) as baselines for comparison.

Implementing a profitable pairs trading strategy (trade logic only) which will tell us optimal entry and exit level of stocks by doing comparison among them to generate max profit.

Significance

In terms of Emotion

As trades are constrained within a set of predefined criteria. Why this is an advantage is because humans trading are susceptible to emotions that lead to irrational decisions

In terms of accuracy

If a computer is automatically executing a trade, you get to avoid the pitfalls of accidentally putting in the wrong trade associated with human trades

In terms of time

The Machine Learning Algorithm will save a lot of time visualization of stocks could be done in minutes, using this we could increase pace of market.

Project Timeline

01

Phase 01

Generating data using yahoo finance, and websites namely NSE etc.

02

Phase 02

Reducing Dimensions using PCA.
Adding a new factor which is industry that particular stock belongs to?

03

Phase 03

Clustering data using DBSCAN.

04

Phase 04

Integrating data using Cointegration.
Generating companies with similar behaviour

05

Phase 05

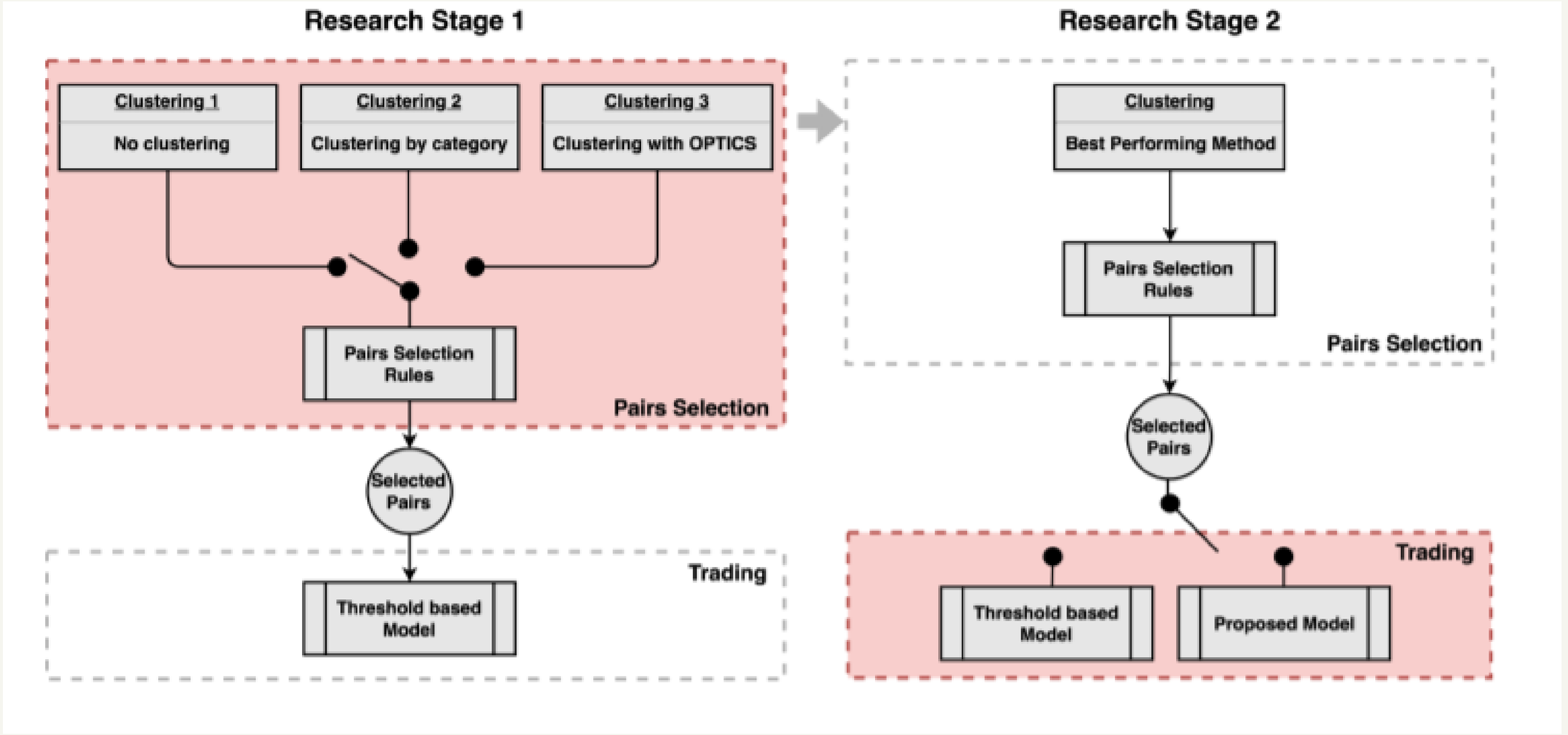
Getting data of companies that give higher profit than the threshold as NIFTY.

Requirements

Hardware Requirement:
Minimum 128 MB of RAM, 256 MB recommended.
110 MB of hard disk space required; 40 MB additional hard disk space required for installation (150 MB total).
Software Requirement:
This project has been designed for windows and other platforms. Python Language is used.
Important Library Used: -
• Yfinance
• Scipy
• Statsmodels
• Sklearn
Tools Used: -
• Jupyter Notebook
• MS Word
• MS PowerPoint

The requirement of the developer is to build the product that is suitable to form pair with similarity and the client wants to put money on the similar better performing stocks.

System Design Overview



Data Collection Methods

Steps and Action Items

Step 01

Collecting data using yahoo finance

Step 02

Making data frame of collected data using pandas

Step 03

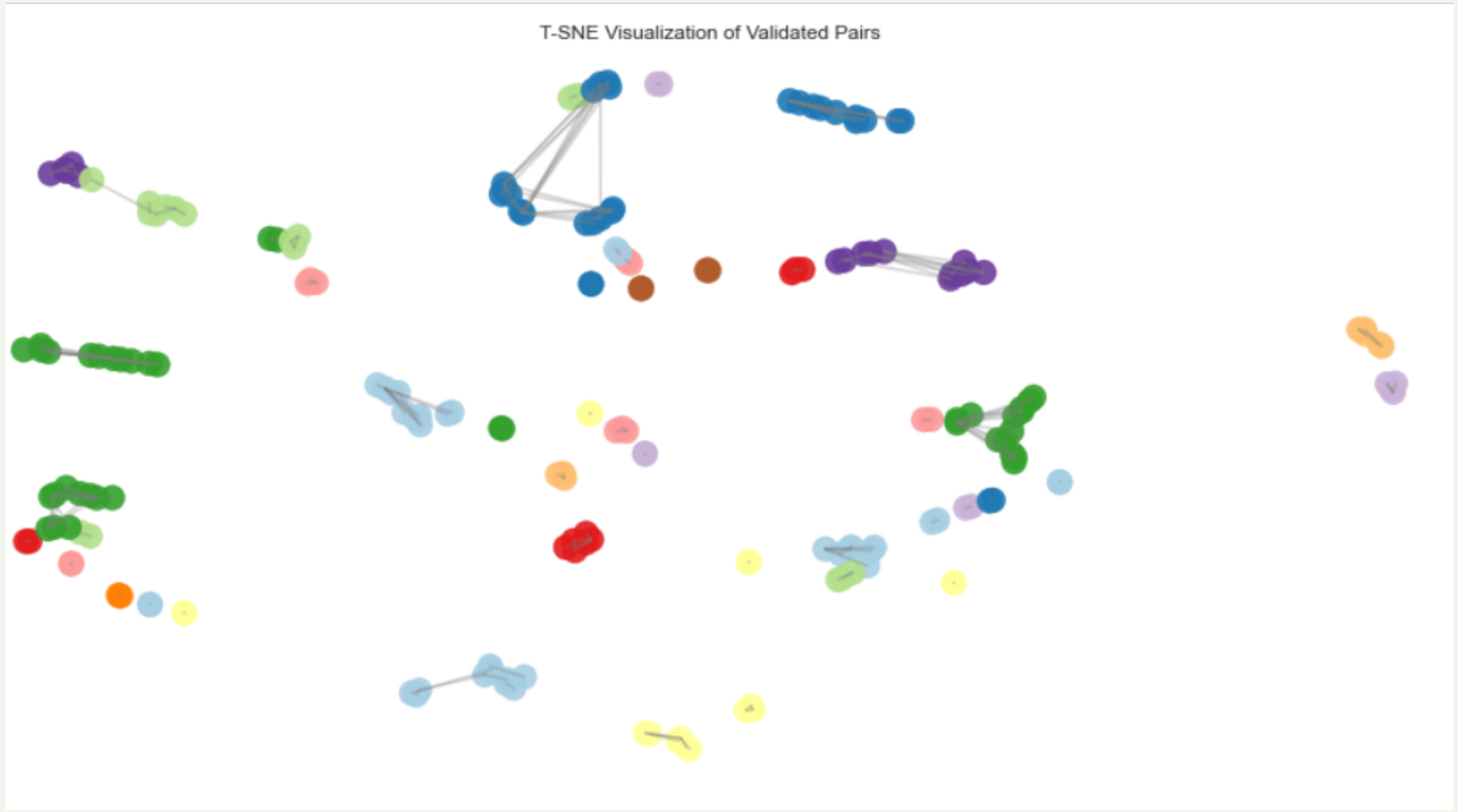
Collection of data from NSE as of company corresponding to its industry.

DIMENSION REDUCTION AND CLUSTERING

We use PCA to reduce the dimensionality of the returns data and extract the historical latent common factor loadings for each stock. For a nice visual introduction to what PCA is doing, take a look here.

We will take these features, add in the fundamental features, and then use the DBSCAN unsupervised clustering algorithm which is available in scikit-learn. Initially I looked at using KMeans but DBSCAN has advantages in this use case, specifically DBSCAN does not cluster all stocks; it leaves out stocks which do not neatly fit into a cluster; relatedly, you do not need to specify the number of clusters.

We have found 166 clusters. The data are clustered in 4 dimensions. As an attempt to visualize what has happened in 2d, we can try with T-SNE. T-SNE is an algorithm for visualizing very high dimension data in 2d. We visualize the discovered pairs to help us gain confidence that the DBSCAN output is sensible; i.e., we want to see that T-SNE and DBSCAN both find our clusters.



STRATEGY

Cointegration between Stocks Because two cointegrated time series (such as X and Y above) drift towards and apart from each other, there will be times when the spread is high and times when the spread is low. We make a pairs trade by buying one security and selling another. This way, if both securities go down together or go up together, we neither make nor lose money—we are market neutral.

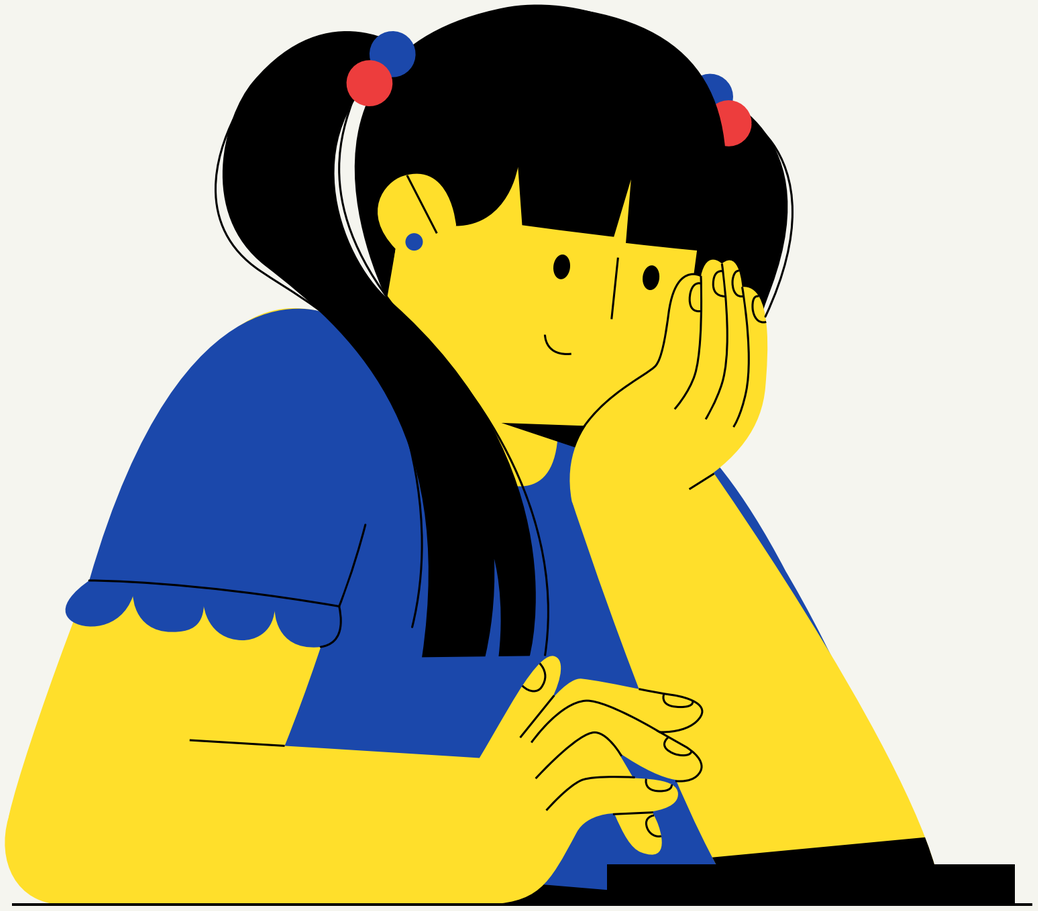
WE ALWAYS HAVE A “HEDGED POSITION”:

Going Long the Ratio This is when the ratio α is smaller than usual and we expect it to increase.

GO "LONG" THE RATIO WHENEVER THE Z-SCORE IS BELOW -1.5

Going Short the Ratio This is when the ratio α is large and we expect it to become smaller. In the above example, we place a bet on this by selling Y and buying X.

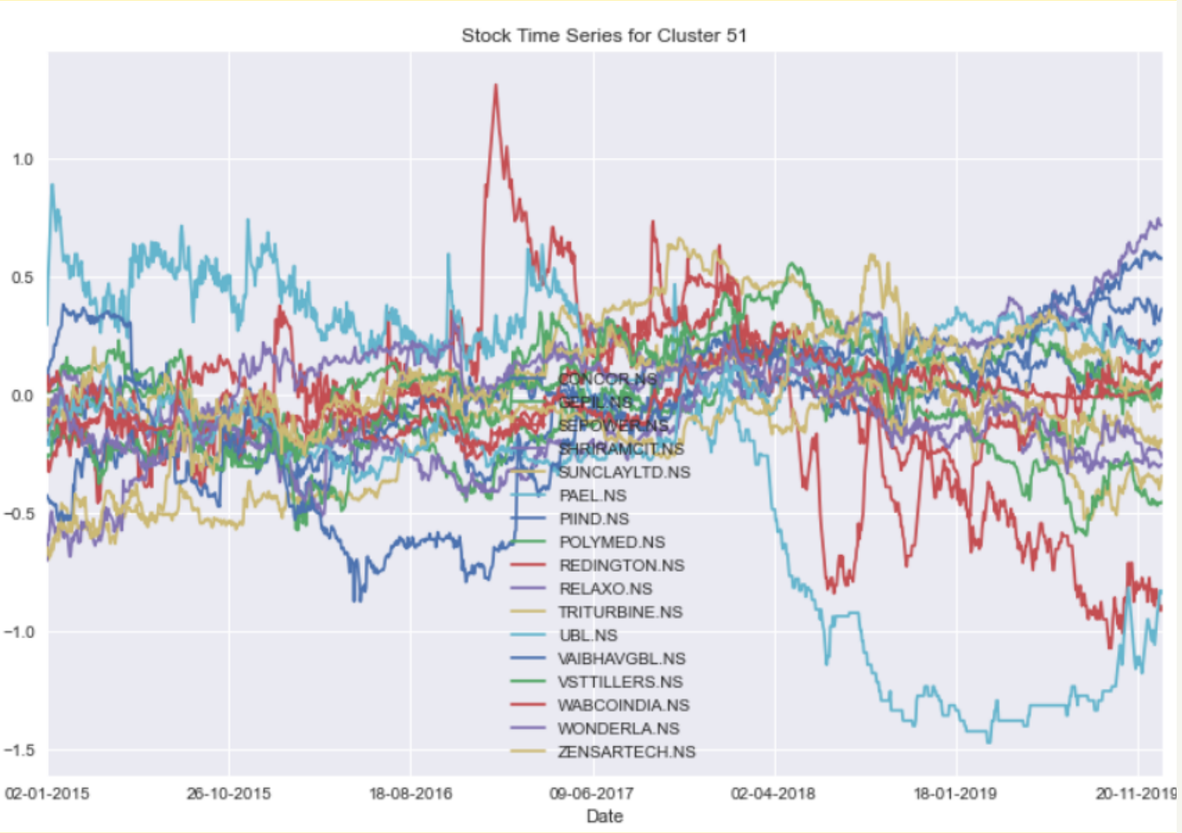
GO "SHORT" THE RATIO WHEN THE Z-SCORE IS ABOVE 1.5



Results

Part04

Qualitative Results



- Data

Creation of Data, Visualization of data, encoding of data resulting in visualization of Stock Time Series for cluster, Pair Visualization as we need prediction collectively rather of one.

- Cointegrating

182 cointegrated pairs were obtained after cointegration test which was based on indusrty, their market bahaviour in given data.

- Visualization

Visualization of Stock Time Series for cluster, Pair Visualization as we need prediction collectively rather of one.

Results

Z Score (Value) = (Value—Mean) / Standard Deviation

Why Z-score?

The absolute ratio isn't very useful in statistical terms. It is more helpful to normalize our signal by treating it as a z-score.

WARNING

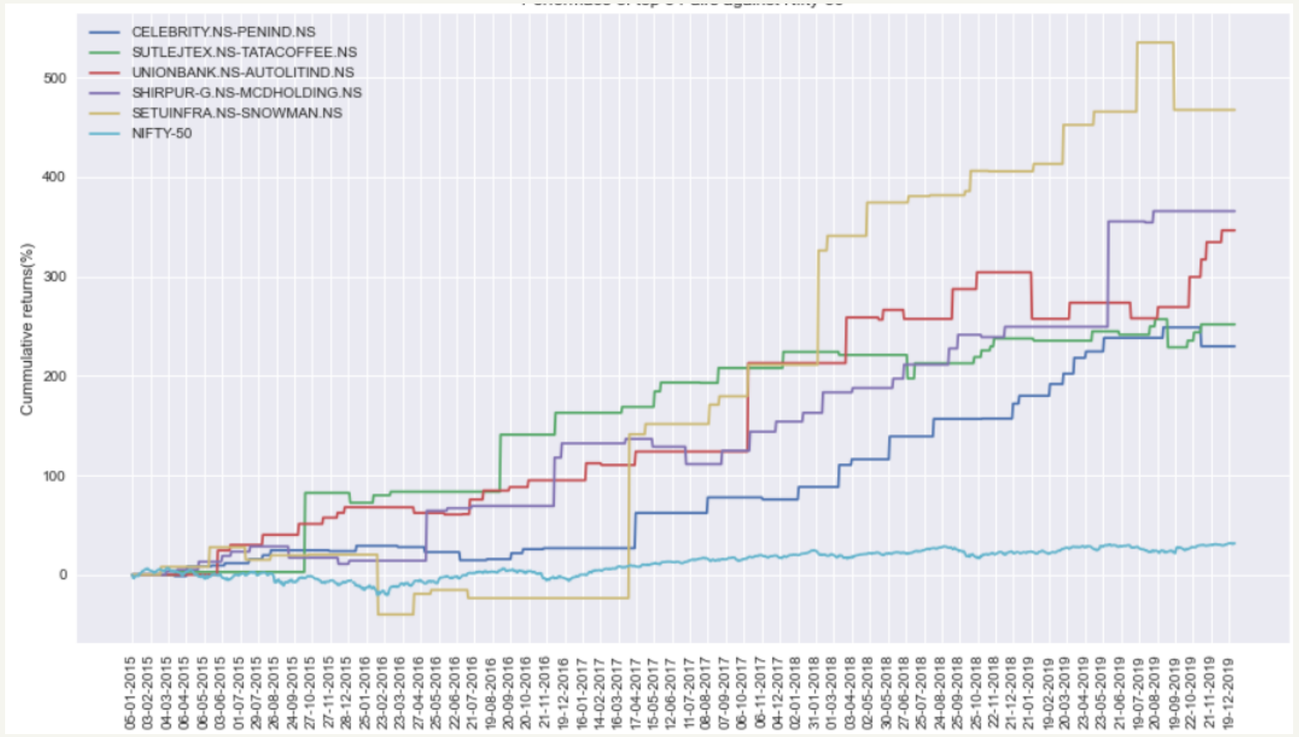
In practice this is usually done to try to give some scale to the data, but this assumes an underlying distribution. Usually normal. However, much financial data is not normally distributed, and we must be very careful not to simply assume normality, or any specific distribution when generating statistics. The true distribution of ratios could be very fat-tailed and prone to extreme values messing up our model and resulting in large losses

Z SCORE : PAIR TRADE

Here we Plot the prices and buy and sell signals from z score

Trading Algorithm

- We Consider Nifty data for Benchmarks
- We Trade all pairs listed by DBSCAN algorithm
- The above strategy trained on over 4 years of data and apply to later 1 year as testing data. Training data have a profit of over $350 Testing data have a profit of over $450

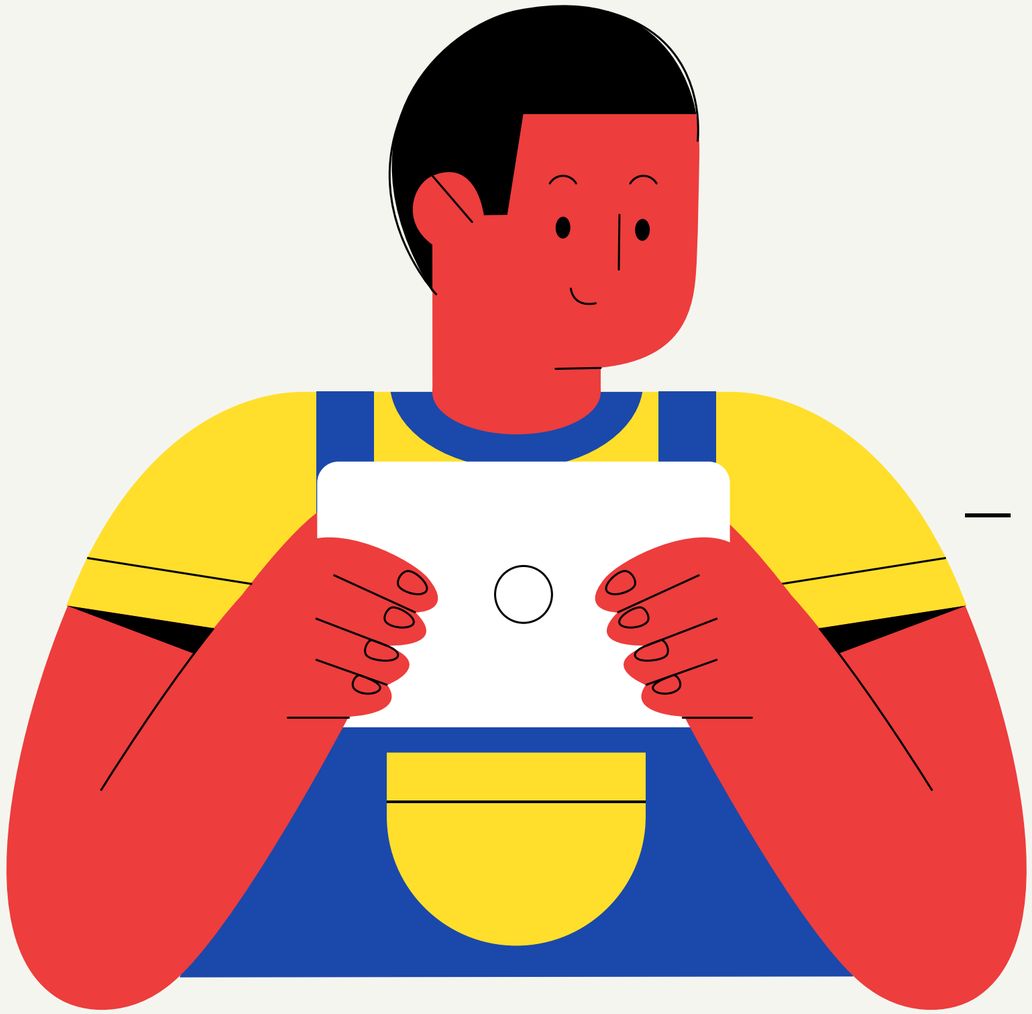


48 PAIRS OUTPERFORMS NIFTY-50 WITH MAXIMUM PROFIT OF 467% ON SINGLE PAIR OVER SPAN OF 5 YEARS. Some of them are

UNIONBANK.NS-AUTOLITIND.NS 346.256000
SHIRPUR-G.NS-MCDHOLDING.NS 365.599711
SETUINFRA.NS-SNOWMAN.NS 467.436900

Future Work

Part 05



We can optimize further by changing and check for performance improvements on validation data. We could also try more sophisticated models like Logistic Regression, SVM etc to make our 1/-1 predictions.

There is need to optimised the algorithm with additional information on financial health or future expansion or quaterly results to cluster down the stocks more accurately.