Trend Prediction of NIFTY using Machine Learning

Objective:

To predict the trend of Nifty by using previous days data of global assets.

Algorithm:

The algorithm for trend prediction is as described in the paper by **Shunrong Shen, Haomiao Jiang and Tongda Zhang.**

It first does calculate the feature matrix by taking the difference of current day price and previous day price and then normalize it. Then It checks the cross-correlation of Nifty with other assets.

SVM algorithm is used in the feature matrix and 10 fold cross validation is applied to optimize the SVM Parameters.

Dataset Used:

The data set used in this project is collected from *"www.investing.com"*. It contains 16 sources as listed in Table I and covers daily price from 02-Jan-2002 to 04-Mar-2018.
Due to holidays in different countries, missing days has been filled by using linear interpolation.

Stocks: NIfty_50, Nifty_500, NASDAQ, FTSE_100, Nikkei 225, SSEC, ASX, DAX, Hang_Seng
Currency: EUR_INR, GBP_INR, JPY_INR
Commodities: Crude_Oil, Gold, Silver, Natural_Gas

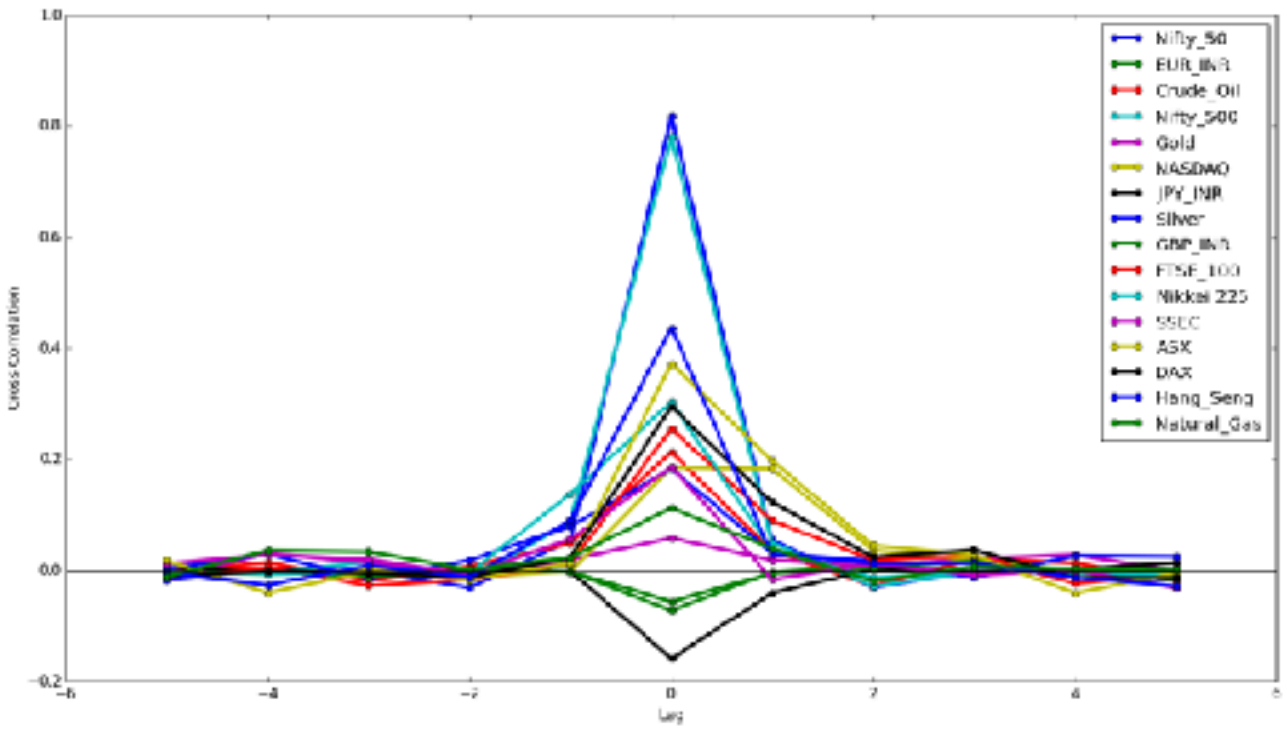
Training dataset: 2002 - 2016
Testing dataset: 2017 - 2018

Feature Selection:

$$F = (X_1, X_2, ..., X_n)^T$$

Where $X_t = (x_1(t), x_2(t), ..., x_{16}(t))$

$$\mathcal{N}(\nabla_\delta X(t)) = (X(t) - X(t-\delta)) / X(t-\delta) \quad \text{where } \delta = 1$$



Implementation:

From previous days Nifty price, A label vector is created with label [-1, 0, 1] where 1 signifies the positive movement, -1 for negative movement and 0 for steady price.

Used SKLEARN python library for SVM, model validation, confusion matrix (classification report) and accuracy report.

For tuning SVM hyperplane, following parameters are used:

kernel = [‘ linear ’, ‘ poly ’, ‘ rbf ’, ‘ sigmoid ’]
C = [0.1 , 1, 10, 100, 1000]
Gamma = [1e-3, 1e-4]
Degree = [1,2,3]
Cross validation = 5 fold

Best Model:

Kernel = sigmoid

On tuning cost:

Cost	**0.1**	**1**	**10**	**100**	**1000**	**10000**
Avg accuracy	54.00%	54.00%	54.00%	55.0%	57.8%	58.5%
Avg. error	0.001	0.001	0.001	0.024	0.035	0.036
						Best fit

On finer tuning:

Cost	Avg. Accuracy	Avg. error	
9500	58.5	0.037	
9800	58.6	0.035	**Best Fit**
10000	58.5	0.036	
10200	58.5	0.034	
10500	58.5	0.036	

Result:

Applying the model on test data:
Accuracy = 59.16 %

Class	Precision	Recall	f1-score	Support
-1	0.72	0.15	0.25	139
0	0.00	0.00	0.00	1
1	0.58	0.95	0.72	171
Avg / total	0.64	0.59	0.51	311

Based on grid search analysis the best model is :

Model	
Kernel	Sigmoid
Cost (C)	9800
Avg Accuracy (5 fold Cross validation)	58.6%
Avg error	0.035
Accuracy (test)	59.16%