VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi-590018, Karnataka

Advanced Machine Learning (18AI72)

Mini Project Report on

"PREDICTING STOCK PRICE DIRECTION USING SUPPORT VECTOR MACHINE"

Submitted in partial fulfillment for the award of the degree of

Bachelor of Engineering
in
Artificial Intelligence & Machine Learning

Submitted by

USN	Name
1BI20AI025	Ms. Lahari G
1BI20AI026	Ms. Likitha M

for the academic year 2023-24

Under the Guidance of

Mrs. Subha Meenakshi
Guest Faculty
Dept. of AI&ML
BIT, Bangalore

Department of Artificial Intelligence & Machine Learning
Bangalore Institute of Technology
K.R. Road, V.V.Pura, Bengaluru-560 004

2023-24

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnana Sangama", Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY
Department of Artificial Intelligence & Machine Learning
K.R. Road, V.V.Pura, Bengaluru-560 004

Certificate

This is to certify that Advanced Machine Learning mini project work entitled **"Predicting Stock Price Direction using Support Vector Machine"** carried out by

USN	Name
1BI20AI025	**Ms. Lahari G**
1BI20AI026	**Ms. Likitha M**

bonafide students of **Bangalore Institute of Technology** in partial fulfillment for the award of degree of **Bachelor of Engineering** in **Artificial Intelligence & Machine Learning** under Visvesvaraya Technological University, Belagavi, during the academic year 2023-24 is true representation of mini project work completed satisfactorily.

Mrs. Subha Meenakshi Guest Faculty Dept. of AI&ML BIT, Bengaluru	Dr. Jyothi D. G. Professor & HoD Dept. of AI&ML BIT, Bengaluru.	Dr. Aswath M. U. Principal BIT Bengaluru.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the completion of a task would be incomplete without crediting the people who made it possible, whose constant guidance and encouragement crowned the efforts with success.

We would like to profoundly thank Management of Bangalore Institute of Technology for providing such a healthy environment for successful completion of Project work.

We would like to express our thanks to the Principal **Dr. Aswath M.U.** for his encouragement that motivates us for the successful completion of Project work.

It gives us immense pleasure to thank **Dr. Jyothi D.G.**, Professor & Head, Department of Artificial Intelligence & Machine Learning for her constant support and encouragement.

We would like to express our deepest gratitude to our mini project guide **Mrs. Subha Meenakshi.** for her constant support and guidance throughout the Mini Project work.

We are very much pleasured to express our sincere gratitude to the friendly co- operation showed by all the staff members of Artificial Intelligence and Machine Learning Department, BIT.

Last but not the least, we would here by acknowledge and thank our friends and family who have been our source of inspiration always instrumental in successful completion of the Project work.

Date : 19-12-2023 Ms. Lahari G

Place: Bengaluru Ms. Likitha M

ABSTRACT

In the world of finance, stock trading is the most essential activity. Predicting the stock market is an act of determining the value of a stock in near future and other financial instruments traded on the financial exchange such as NSE, BSE. The fundamental and technical analysis is being in use by the brokers of stock exchange when stocks are being predicted. Here in this report we proposed the method which is called as Machine learning (ML) which is made available by training the stock data, will then gain intelligence and thus finally uses the acquired knowledge for an appropriate prediction. We used Support Vector Machine to predict prices of a stock for small and large capitalizations and in the different markets, employing prices daily with the minute frequencies. In Support Vector Machine, when the data is spread then the line from where the most of the points pass is drawn and from there the vectors from the points to the line are drawn.

INDEX

Table of figures

CHAPTER – 1

INTRODUCTION

Chapter 1

INTRODUCTION

Stock price prediction is one of the most widely studied and challenging problems, attracting researchers from many fields including economics, history, finance, mathematics, and computer science. The volatile nature of the stock market makes it difficult to apply simple time-series or regression techniques. Financial institutions and traders have created various proprietary models to try and beat the market for themselves or their clients, but rarely has anyone achieved consistently higher-than-average returns on investment. Nevertheless, the challenge of stock forecasting is so appealing because an improvement of just a few percentage points can increase profit by millions of dollars for these institutions. Traditionally, many prediction models have focused on linear statistical time series models such as ARIMA [7]. However, the variance underlying the movement of stocks and other assets makes linear techniques suboptimal, and non-linear models like ARCH tend to have lower predictive error [17]. Recently, researchers have turned to techniques in the computer science fields of big data and machine learning for stock price forecasting. These apply computational power to extend theories in mathematics and statistics. Machine learning algorithms use given data to "figure out" the solution to a given problem. Big data and machine learning techniques are also the basis for algorithmic and high-frequency trading routines used by financial institutions. In this paper we focus on a specific machine learning technique known as Support Vector Machines (SVM). Our goal is to use SVM at time t to predict whether a given stock's price is higher or lower on day t+m. We look at the technology sector and 34 technology stocks. We input four parameters to the model - the recent price volatility and momentum of the individual stock and of the technology sector. These parameters are calculated using daily closing prices for each stock from the years 2007 through 2014. We analyze whether this historical data can help us predict price direction. If the Efficient Markets Hypothesis (EMH) holds true, prices should follow a random walk and be unpredictable based on historical data. We find that in the short-term this holds true, but in the long-term we can reach prediction accuracies between 55% and 60%. We conclude that our model can achieve significant prediction accuracies with some parameters in the long-term, but that we must look at more granular intra-day trading data to achieve prediction accuracies in the short-term.

CHAPTER – 2

LITERATURE REVIEW

Chapter 2

LITERATURE REVIEW

S.NO.	Authors name & Year of Publication	Title name & Journal name	Abstract	Techniques used	Limitations
1	Kyoung-jae, Kim. 2003	Financial time series forecasting using support vector machines. ELSEVIER	This study applies Support Vector Machines (SVM) to predict stock prices, comparing its performance with Artificial Neural Networks (ANN) and Case-Based Reasoning (CBR) for financial forecasting.	SVM, ANN with CBR	SVM's prediction sensitivity to parameters like the upper bound C and kernel parameter δ2 is noted. Optimal parameter values are crucial. Challenges with overfitting and parameter selection in traditional ANN models are acknowledged.
2	Huseyin Ince, Theodore B. Trafalis 2004	Kernel Principal Component Analysis and Support Vector Machines for Stock Price Prediction. IEEE	This study compares stock price prediction using Kernel Principal Component Analysis (kPCA) and Support Vector Regression (SVR) against Multi-Layer Perceptron (MLP) neural networks. Objectives include influential input identification through kPCA and factor analysis, and assessing predictive capabilities of SVR and MLP models.	kPCA, SVR, MLP	The study highlights the impact of preprocessing techniques on input selection, influencing SVR and MLP model performance. It underscores the sensitivity of these techniques to preprocessing methods in stock prediction.

3	Lucas, K. C. Lai, James, N. K. Liu. 2010	Predicting Stock Market Price Using Support Vector Regression. IEEE	The paper proposes a new approach called Win-SVR, combining support vector regression (SVR) with different windowing operators for predicting stock market prices.	Win-SVR	The study is limited to one dataset from the Dhaka Stock Exchange.
4	Lu, Chi-Jie, Chang, Chih-Hsiang, Chen, Chien-Yu, Chiu, Chih-Chou, Lee, Tian-Shyug. 2009	Stock Index Prediction: A Comparison of MARS, BPN and SVR in an Emerging Market. IEEE	The study aims to compare the forecasting performance of Multivariate Adaptive Regression Splines (MARS), Backpropagation Neural Network (BPN), Support Vector Regression (SVR), and Multiple Linear Regression (MLR) models in predicting the Shanghai B-Share stock index.	MARS, BPN, SVR, MLR	The study focuses on the Shanghai B-Share stock index, limiting generalizability to other markets. Additionally, the choice of technical indicators and the specific time period may influence results. The study does not consider external factors or events that could impact stock prices.
5	Phayung Meesad, Risul Islam Rasel. 2013	Dhaka Stock Exchange Trend Analysis Using Support Vector Regression. SPRINGER	Develop a stock market trend prediction model using Support Vector Regression (SVM) with windowing operator.	Win-SVR with RBF kernel.	Limited to Dhaka Stock Exchange data, may not generalize to other markets; parameter tuning sensitivity in SVM may impact accuracy.
6	Kuan-Yu. Chen, Chia-Hui. Ho. 2005	An Improved Support Vector Regression Modeling	Develop an improved method for financial volatility forecasting using Support Vector	SVR GBC.	Limited to comparing results with GARCH (1, 1) and traditional grid search SVR methods; may face challenges in diverse

		for Taiwan Stock Exchange Market Weighted Index Forecasting. IEEE	Regression (SVR) with a hybrid genetic algorithm (SVR GBC) to simultaneously select optimal kernels and parameters.		market conditions or require additional validation across various financial instruments.
7	Yuling Lin, Haixiang Guo, Jinglu Hu. 2013	An SVM-based approach for stock market trend prediction. IJCNN	Develop an SVM-based approach for stock market trend prediction, focusing on feature selection and prediction modeling to improve generalization performance and hit ratio.	Quasi-linear SVM.	Limited to experimental results on Taiwan stock market datasets; potential challenges in adapting to diverse market conditions and requiring validation across various financial markets.
8	Jayant Gidwani, Parag Rangari, Shweta Bandewar, Sejal Barsinge, Prof.Vanita Buradkar. 2014	Visualizing and forecasting of stocks using support vector regression. JETIR	Develop a stock price prediction model using Support Vector Regression (SVR) and different windowing operators. Explore the performance of SVR with various kernel functions on NSE-listed firms' stock data.	SVR, EMA.	3. Linear kernel of SVR shows smaller prediction errors in the test set than in the training set; however, predictive power diminishes with fixed training periods and increased price frequency to minutes. SVR performs inferiorly to a random walk model for some stocks in up-to-the-minute prices with fixed training, irrespective of the kernel function used.
9	Lijuan. Cao. 2003	Support vector machines experts for time series forecasting. ELSEVIER	The paper proposes a Support Vector Machines (SVMs) expert model for time series forecasting using a two-stage neural network architecture,	SVM, SOM.	While demonstrating improved generalization performance, faster convergence, and reduced support vectors, the paper's approach may have computational

			combining self-organizing feature maps (SOM) for clustering and SVMs for modeling.		complexities due to the two-stage architecture and requires careful parameter tuning for optimal results.

2.1. EXISTING SYSTEM

The existing system for stock price prediction typically relies on a combination of traditional financial models and advanced machine learning techniques. Traditional models, such as ARIMA, have been widely used, but their effectiveness is limited due to the complex and dynamic nature of the stock market. In recent years, machine learning approaches, including Support Vector Machines (SVM) and Artificial Neural Networks (ANN), have gained popularity. SVM, as mentioned in the paper, is a prevalent choice for its ability to handle non-linear relationships and find optimal decision boundaries. These models utilize historical stock price data and various financial indicators as input features to predict future price movements. The challenge lies in achieving accurate short-term predictions, given the market's inherent volatility, while achieving more success in the long term. Ongoing research explores ways to improve model accuracy, feature selection, and the incorporation of additional factors for a more comprehensive understanding of stock price dynamics.

2.2. PROBLEM STATEMENT

Developing an enhanced stock price prediction model using Support Vector Machines (SVM), this project aims to overcome short-term predictability challenges and assess the model's efficacy in the long run.

2.3. PROPOSED SYSTEM

Developing an advanced stock price prediction system using Support Vector Machines (SVM) with a focus on volatility and momentum parameters. The project aims to optimize short-term and long-term forecasting accuracy by leveraging SVM's capabilities, assessing the predictive power of historical data during the Great Recession, and offering valuable insights for more effective decision-making in the stock market.

CHAPTER – 3

SYSTEM REQUIREMENTS

Chapter 3

SYSTEM REQUIREMENTS

3.1. Hardware Requirement

Processor : Intel i5 10th gen or above

Monitor Screen : 14'' or above

RAM : 6 GB and above

Memory : 512 GB SSD / 1TB HDD and above

3.2. Software Requirement

Programming Languages : Python

Software required : jupyter notebook

Python Libraries : pandas, numpy, matplotlib, sklearn

Operating System: Windows

Terminal Window – To run the Python file using commands

CHAPTER – 4

SYSTEM DESIGN

Chapter 4

SYSTEM DESIGN

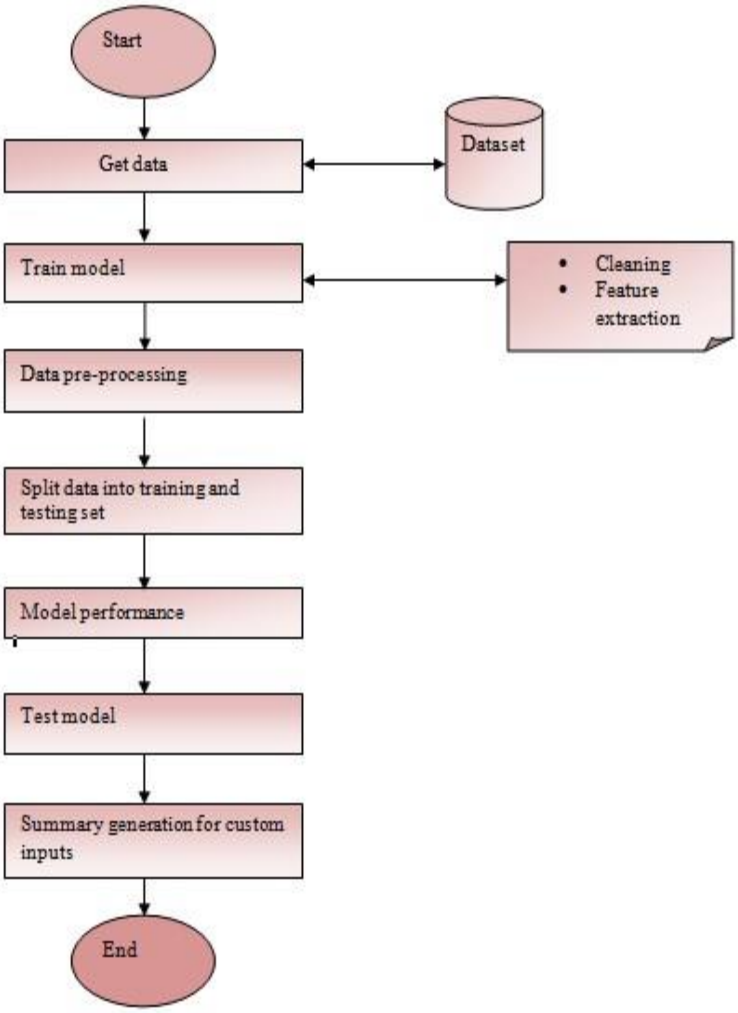

Fig 4.1. System Design

a. Preprocessing and Cleaning

Removing the redundant data and recovering and the missing data and. This step basically involves creation of useful features form the existing ones.

b. Feature Extraction

In this step searching is done with the space of possible feature subsets. We then picked up thesubset which is optimal or near-optimal with respect to some objective function. Overfitting and underfitting the dataset is major problem and hence, this is done to avoid the same.

c. Data Normalization

Information is should have been standardized for better exactness by guaranteeing that all highlights are not given over the top/low weight age.

d. Analysis of various supervised learning methods

Approach using Support Vector Machine (SVM)

We are going to use Support vector machines (SVMs) for supervised learning methods as for categorization, reverting and outliners detection. "Support Vector Machine" (SVM) is a directed AI calculation and we can utilize it for order and relapse issues yet we ordinarily use it for grouping issues. In SVM calculation, information things are plotted. Then, hyper-plane is identified by performing classification that differentiates the planes into two halves.

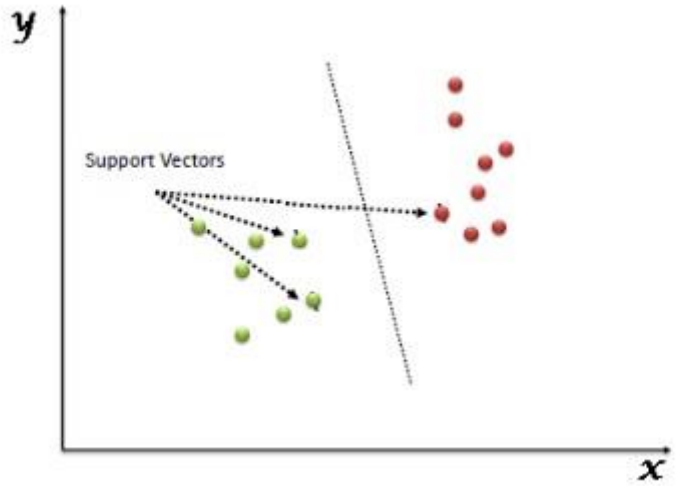

Fig 4.2. SVM Diagram

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/line).

SVM has the following advantages:

a. Effective in high dimensional spaces.
b. It is helpful in places where amount of tests are less that measure of measurements
c. Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
d. Versatile: SVM has the advantage of determining diverse piece capacities. We as a rule give normal bits however we can also specify custom kernels.

The disadvantages of support vector machines are:

a. If the features are greater than the samples, avoid over fitting in choosing Kernel functions and regularization term in crucial.

b. SVMs do not directly provide estimates of profitability, these are solved using an expensive five-fold cross-validation.

The support vector machines in scikit-learn support both dense and sparse sample vectors as input. Data must have been fit properly to fit predictions for sparse data. We have used C-ordered numpy.ndarray(dense) or scipy.sparse.csr_matrix(sparse) having dtype=float=64 for optimal performance.

Classification and Regression

The picture beneath shows SVC, NuSVC and LinearSVC are classes that are fit for performing multi-class characterization on a dataset

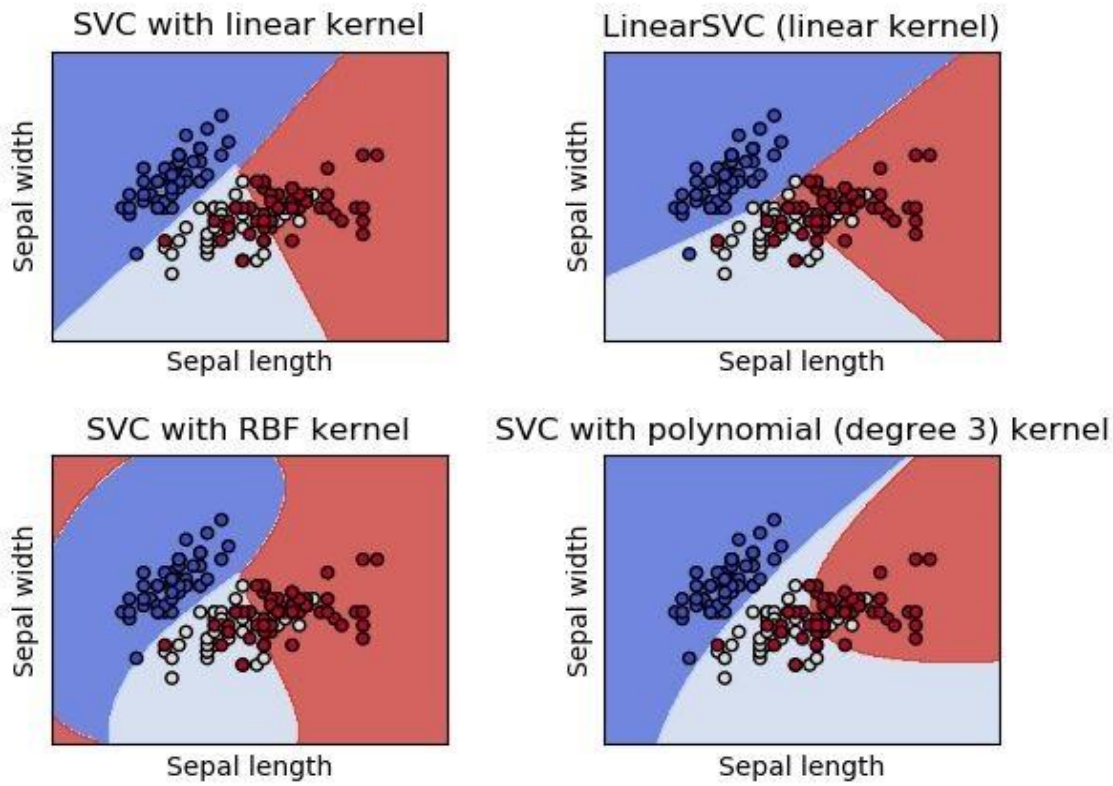

Fig 4.3. Classification of Data

SVC and NuSVC are comparable strategies, however they have slight contrast in the arrangement of parameters and furthermore, they have diverse numerical plans. On the other note, LinearSVC is some another execution of Support Vector machines for the instance of a straight part. In LinearSVC, Kernel is thought to be direct and that is the reason it does not acknowledge catchphrase piece.

As other classifiers, **SVC**, **NuSVC** and **LinearSVC** take two arrays as input: an array X of size [n_samples, n_features] which will hold the training samples, and second array y of class labels (strings or integers), size [n_samples]:

>>from sklearn import svm

x= [[0, 0], [1, 1]] y = [0, 1]

clf=svm.SVC(gamma='scale')

clf.fit(x, y)

SVC(C=1.0, cache_size=200, class_weight=None, coef=0.0, decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False)

clf.predict([[2., 2.]])

array([1])

 # get support vectors *clf.support_vectors_array([[0., 0.], [1., 1.]])*

 # get files of help vectors *clf.support_ array([0, 1]...)*

 # get number of help vectors for each class

clf.n_support_ array([1, 1]

4.1 Classification SVM Types

4.1.1 Classification SVM Type1

In this type of SVM, training involves error function minimization:

$$\frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$$

eq. 1

In limitations with:

$$y_i\left(w^T\phi(x_i)+b\right)\geq 1-\xi_i \text{ and } \xi_i \geq 0, i=1,\ldots,N$$

eq. 2

where C is the breaking point consistent, w is the vector of coefficients, b is a predictable, and addresses boundaries for managing non particular data (inputs). The rundown names the N getting ready cases. Note that addresses the class names and xi addresses the self-sufficient elements. Data input is moved to incorporate space using part. More misstep is rebuffed if C regard is greater.. Thus, C should be picked with care to keep up a key good way from over fitting.

4.1.2 Classification SVM Type2

The Classification SVM Type 2 involves model minimizes the error function:

$$\frac{1}{2}w^T w-v\rho+\frac{1}{N}\sum_{i=1}^{N}\xi_i$$

eq. 3

subject to the constraints:

$$y_i\left(w^T\phi(x_i)+b\right)\geq \rho-\xi_i, \xi_i \geq 0, i=1,\ldots,N \text{ and } \rho\geq 0$$

eq. 4

In a relapse SVM, subordinate variable and free factor reliance is evaluated. It expects as other relapse issues, the free and ward relationship is distinguished by deterministic capacity f(x) including a portion of the added substance commotion.

4.2 Regression SVM Types

4.2.1 Regression: SVM Type1

y = f(x) + noise eq.5

We need to find a useful structure for f(x)which will adequately foresee new cases that the SVM has not been given already. This can be cultivated by means of setting up the SVM model on a model set, i.e., getting ready set, a system that incorporates, like request the progressive improvement of a botch work.

4.2.2 Regression SVM Type2

The error function is given by:

$$\frac{1}{2} w^T w + C\sum_{i=1}^{N} \xi_i + C\sum_{i=1}^{N} \xi^* i$$

eq. 6

which we minimize subject to:

$$w^T \phi(x_i) + b - y_i \le \varepsilon + \xi_i^*$$
$$y_i - w^T \phi(x_i) - b_i \le \varepsilon + \xi_i$$
$$\xi_i, \xi_i^* \ge 0, i = 1,\ldots,N$$

eq. 7

Support Vector Machines models can use many numbers of kernels. These include polynomial, linear, radial basis function (RBF) and sigmoid.

Abstraction based extraction

Support Vector Machines has a basic idea dependent on recognizing of choice planes that characterize choice limits. A choice plane is one that isolates between various arrangement of articles with participations of various classes. The picture beneath delineates that. The picture, the items have a place either with class GREEN or RED. The limit that characterizes the different lines on the correct side of articles are GREEN and left side items are RED. Another article which is on the correct side is stamped, i.e., characterized, as GREEN and left one is delegated RED

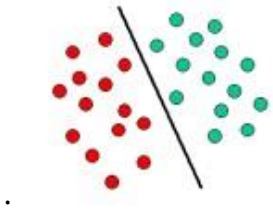

Fig 4.4. Abstraction

The above model is of a direct classifier, i.e., a classifier isolating two arrangements of articles as GREEN and RED. Most arrangement errands, although, are not unreasonably basic, and frequently progressively complex structures are required to make an ideal partition, i.e., accurately grouping new articles (test information) based on the models that are accessible (train information). This circumstance is appeared in the picture below. When past schematic is being analyzed, plainly a full partition of the GREEN and RED articles would require a bend (which is more perplexing than a line). Order undertakings that depend on attracting isolating lines to recognize objects of various class participations are known as hyperplane classifiers. Support Vector Machines are structured in such an away to deal with such undertakings.

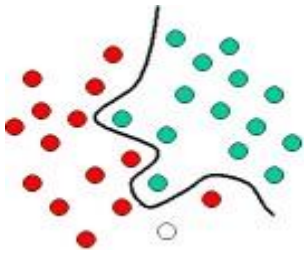

Fig 4.5. Abstraction

The picture underneath shows the fundamental structure behind Support Vector Machines. Here we can see the first items (left half of the schematic) mapped, that are revised, utilizing a lot of scientific capacities which are known as portions. The way toward reworking the items is known as mapping transformation. In the new setting, the mapped objects (right half of the outline) are straightly distinguishable along these lines, rather than building the unpredictable bend (left delineation), we need to do is to locate an ideal line which will isolate the GREEN and the RED items.

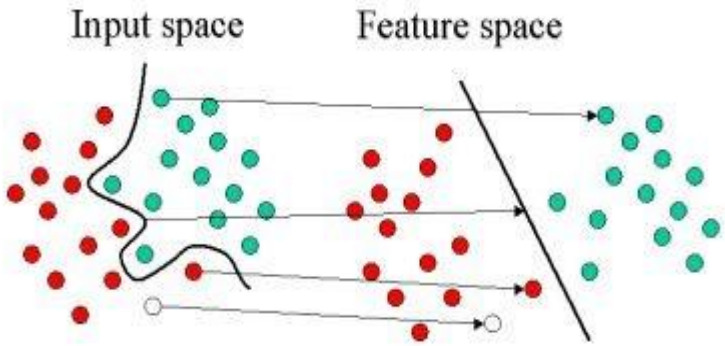

Fig 4.6. Abstraction

Parameters

There are two types of SVR: Linear and Multiple.

Tuning parameters esteem for calculations in Machine Learning emotionally improves the model execution. There are rundown of parameters accessible with SVR, which we will examine, so some significant parameters having higher effect on model execution, 'Part', 'Degree', 'Gamma', 'C'.

a. Kernel: It is a similarity function and requires two inputs and spits out how similar they are. It helps in representing the infinite set of discrete function in a family of constant function.
b. Gamma: It is used in RBF (Radial Basis Function) model to indicate variance. A small gamma means a Gaussian surface with large variance. Gamma controls the shape of peaks and height of pointed curves, higher the value of gamma, will try to exact fit the training datasets that is generalization error and cause overfitting problem.
c. C: It is a penalty parameter for error term. To have the best fit some points in regression can always be ignored this is indicated by c or C means low bias and high variance as you penalize a lot misclassification. It also controls the trade-off between smooth decision boundary and classifying the training points correctly.
d. Degree: Degree parameter specifies the degree of poly Kernel function. There is a trail and dealing with degree parameter. The more the degree parameter, more the accuracy but this also leads to more computational time and complexity.

CHAPTER – 5

IMPLEMENTATION

Chapter 5

IMPLEMENTATION

```
# Machine learning
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

# For data manipulation
import pandas as pd
import numpy as np

# To plot
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')

# To ignore warnings
import warnings
warnings.filterwarnings("ignore")

df = pd.read_csv('RELIANCE.csv')
df

# Changes The Date column as index columns
df.index = pd.to_datetime(df['Date'])
df

# drop The original date column
df = df.drop(['Date'], axis='columns')
df

# Create predictor variables
df['Open-Close'] = df.Open - df.Close
```

```
df['High-Low'] = df.High - df.Low

# Store all predictor variables in a variable X
X = df[['Open-Close', 'High-Low']]
X.head()

# Target variables
y = np.where(df['Close'].shift(-1) > df['Close'], 1, 0)
y

split_percentage = 0.8
split = int(split_percentage*len(df))

# Train data set
X_train = X[:split]
y_train = y[:split]

# Test data set
X_test = X[split:]
y_test = y[split:]

# Support vector classifier
cls = SVC().fit(X_train, y_train)

df['Predicted_Signal'] = cls.predict(X)

print(df['Predicted_Signal'])

# Calculate daily returns
df['Return'] = df.Close.pct_change()
```

```
# Calculate strategy returns
df['Strategy_Return'] = df.Return *df.Predicted_Signal.shift(1)

# Calculate Cumulutive returns
df['Cum_Ret'] = df['Return'].cumsum()
df

# Plot Strategy Cumulative returns
df['Cum_Strategy'] = df['Strategy_Return'].cumsum()
df

import matplotlib.pyplot as plt
%matplotlib inline

plt.plot(df['Cum_Ret'],color='red')
plt.plot(df['Cum_Strategy'],color='blue')
```

CHAPTER – 6

RESULTS WITH SCREENSHOTS

Chapter 6

RESULTS WITH SCREENSHOTS

Out[2]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2022-12-19	2581.000000	2604.000000	2566.699951	2599.300049	2590.150391	3567363
1	2022-12-20	2583.899902	2626.000000	2566.000000	2621.800049	2612.571289	3446291
2	2022-12-21	2621.000000	2633.000000	2576.100098	2584.500000	2575.402588	3935463
3	2022-12-22	2598.000000	2604.649902	2566.750000	2577.800049	2568.726074	3438692
4	2022-12-23	2563.300049	2590.500000	2492.250000	2502.199951	2493.392090	4733657
...	...	...	...	...	...	...	...
241	2023-12-11	2456.000000	2467.600098	2452.399902	2459.350098	2459.350098	3533069
242	2023-12-12	2460.000000	2464.949951	2420.149902	2424.050049	2424.050049	4598562
243	2023-12-13	2422.000000	2438.350098	2406.300049	2433.949951	2433.949951	5015591
244	2023-12-14	2454.000000	2474.949951	2442.649902	2464.149902	2464.149902	8486177
245	2023-12-15	2478.000000	2500.000000	2470.050049	2495.600098	2495.600098	7966076

246 rows × 7 columns

Fig 6.1. Reliance Dataset

Out[3]:

Date	Open	High	Low	Close	Adj Close	Volume
2022-12-19	2581.000000	2604.000000	2566.699951	2599.300049	2590.150391	3567363
2022-12-20	2583.899902	2626.000000	2566.000000	2621.800049	2612.571289	3446291
2022-12-21	2621.000000	2633.000000	2576.100098	2584.500000	2575.402588	3935463
2022-12-22	2598.000000	2604.649902	2566.750000	2577.800049	2568.726074	3438692
2022-12-23	2563.300049	2590.500000	2492.250000	2502.199951	2493.392090	4733657
...	...	...	...	...	...	...
2023-12-11	2456.000000	2467.600098	2452.399902	2459.350098	2459.350098	3533069
2023-12-12	2460.000000	2464.949951	2420.149902	2424.050049	2424.050049	4598562
2023-12-13	2422.000000	2438.350098	2406.300049	2433.949951	2433.949951	5015591
2023-12-14	2454.000000	2474.949951	2442.649902	2464.149902	2464.149902	8486177
2023-12-15	2478.000000	2500.000000	2470.050049	2495.600098	2495.600098	7966076

246 rows × 6 columns

Fig 6.2. Change of Date Column to Index Column

Out[4]:

Date	Open-Close	High-Low
2022-12-19	-18.300049	37.300049
2022-12-20	-37.900147	60.000000
2022-12-21	36.500000	56.899902
2022-12-22	20.199951	37.899902
2022-12-23	61.100098	98.250000

Fig 6.3. Predictor variables stored in a variable X

```
Out[5]: array([1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0,
               0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0,
               1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
               0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
               1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0,
               0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1,
               0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0,
               0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
               1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1,
               0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1,
               0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0,
               1, 1, 1, 0])
```

Fig 6.4. Target variable y

```
In [9]:  1  print(df['Predicted_Signal'])

Date
2022-12-19    1
2022-12-20    0
2022-12-21    0
2022-12-22    0
2022-12-23    0
             ..
2023-12-11    1
2023-12-12    0
2023-12-13    1
2023-12-14    1
2023-12-15    1
Name: Predicted_Signal, Length: 246, dtype: int32
```

Fig 6.5. Predicted signal

Out[12]:

Date	Open	High	Low	Close	Adj Close	Volume	Open-Close	High-Low	Predicted_Signal	Return	Strategy_Return	Cum_Re
2022-12-19	2581.000000	2604.000000	2566.699951	2599.300049	2590.150391	3567363	-18.300049	37.300049	1	NaN	NaN	NaN
2022-12-20	2583.899902	2626.000000	2566.000000	2621.800049	2612.571289	3446291	-37.900147	60.000000	0	0.008656	0.008656	0.00865
2022-12-21	2621.000000	2633.000000	2576.100098	2584.500000	2575.402588	3935463	36.500000	56.899902	0	-0.014227	-0.000000	-0.00557
2022-12-22	2598.000000	2604.649902	2566.750000	2577.800049	2568.726074	3438692	20.199951	37.899902	0	-0.002592	-0.000000	-0.00816
2022-12-23	2563.300049	2590.500000	2492.250000	2502.199951	2493.392090	4733657	61.100098	98.250000	0	-0.029327	-0.000000	-0.03749
...	...	...	...	...	...	...	...	...	...	...	...	...
2023-12-11	2456.000000	2467.600098	2452.399902	2459.350098	2459.350098	3533069	-3.350098	15.200196	1	0.001466	0.001466	-0.03733
2023-12-12	2460.000000	2464.949951	2420.149902	2424.050049	2424.050049	4598562	35.949951	44.800049	0	-0.014353	-0.014353	-0.05168
2023-12-13	2422.000000	2438.350098	2406.300049	2433.949951	2433.949951	5015591	-11.949951	32.050049	1	0.004084	0.000000	-0.04760
2023-12-14	2454.000000	2474.949951	2442.649902	2464.149902	2464.149902	8486177	-10.149902	32.300049	1	0.012408	0.012408	-0.03519

Fig 6.6. Calculated cumulative returns

Out[13]:

Date	Open	High	Low	Close	Adj Close	Volume	Open-Close	High-Low	Predicted_Signal	Return	Strategy_Return	Cum_Re
2022-12-19	2581.000000	2604.000000	2566.699951	2599.300049	2590.150391	3567363	-18.300049	37.300049	1	NaN	NaN	NaN
2022-12-20	2583.899902	2626.000000	2566.000000	2621.800049	2612.571289	3446291	-37.900147	60.000000	0	0.008656	0.008656	0.00865
2022-12-21	2621.000000	2633.000000	2576.100098	2584.500000	2575.402588	3935463	36.500000	56.899902	0	-0.014227	-0.000000	-0.00557
2022-12-22	2598.000000	2604.649902	2566.750000	2577.800049	2568.726074	3438692	20.199951	37.899902	0	-0.002592	-0.000000	-0.00816
2022-12-23	2563.300049	2590.500000	2492.250000	2502.199951	2493.392090	4733657	61.100098	98.250000	0	-0.029327	-0.000000	-0.03749
...	...	...	...	...	...	...	...	...	...	...	...	...
2023-12-11	2456.000000	2467.600098	2452.399902	2459.350098	2459.350098	3533069	-3.350098	15.200196	1	0.001466	0.001466	-0.03733
2023-12-12	2460.000000	2464.949951	2420.149902	2424.050049	2424.050049	4598562	35.949951	44.800049	0	-0.014353	-0.014353	-0.05168
2023-12-13	2422.000000	2438.350098	2406.300049	2433.949951	2433.949951	5015591	-11.949951	32.050049	1	0.004084	0.000000	-0.04760
2023-12-14	2454.000000	2474.949951	2442.649902	2464.149902	2464.149902	8486177	-10.149902	32.300049	1	0.012408	0.012408	-0.03519

Fig 6.7. Strategy cumulative returns

Out[14]: [<matplotlib.lines.Line2D at 0x1e4809bb9d0>]

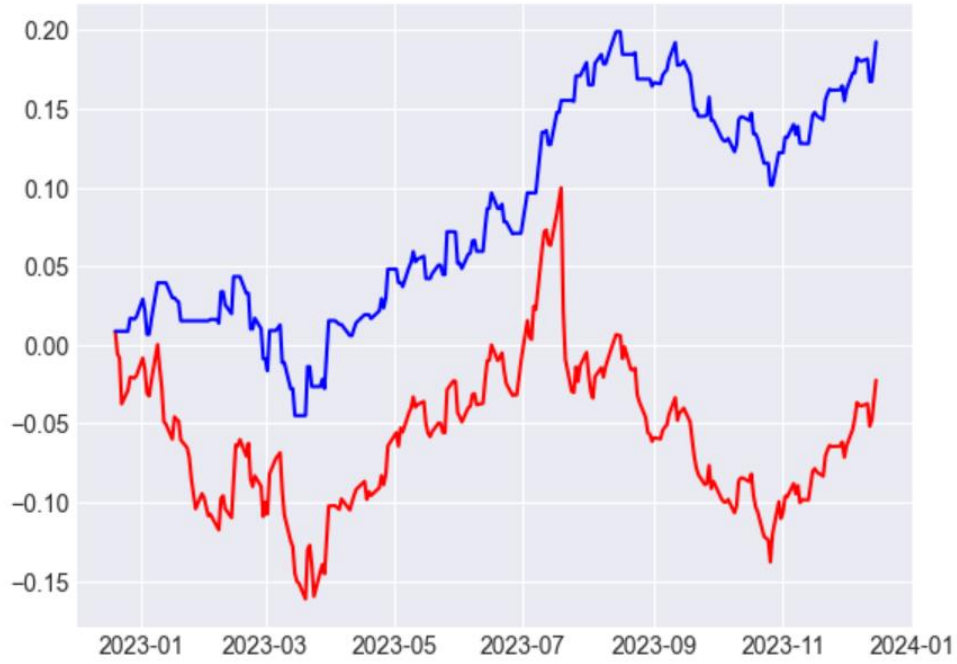

Fig 6.8. Plot between calculated cumulative returns and strategy cumulative returns

CONCLUSION

In conclusion, the examination of a Support Vector Machines (SVM) model for predicting stock price direction underscores the prevailing randomness in short-term price movements, aligning with the Efficient Markets Hypothesis (EMH). The model's limited success in short-term predictions suggests that relying solely on historical data may offer nominal advantages, while the heightened significance of input parameters in long-term forecasts emphasizes the influence of broader market trends. The observed superiority of sector-level historical data over short-term stock-specific trends implies that macroeconomic conditions play a pivotal role in predicting future stock price directions. The outlined avenues for future exploration, such as delving into intra-day trading data, expanding feature sets, and extending the model to diverse sectors and company sizes, offer potential enhancements to forecasting accuracy and market understanding.

References

[1] Kim, Kyoung-jae, “Financial time series forecasting using support vector machines,” In: Neurocomputing 55, pp.307 – 319, 2003.

[2] H. Ince, T.B. Trafalis, “Kernel Principal Component Analysis and Support Vector Machines for Stock Price Prediction,” In: 0-7803-8359- 1/04/ 2004 IEEE, pp. 2053-2058, 2004

[3] Lucas, K. C. Lai, James, N. K. Liu, “Stock Forecasting Using Support Vector Machine,” In: Proceedings of the Ninth International Conference on Machine Learning and Cybernetics, pp. 1607-1614, 2010

[4] Lu, Chi-Jie, Chang, Chih-Hsiang, Chen, Chien-Yu, Chiu, Chih-Chou, Lee, Tian-Shyug, “Stock Index Prediction: A Comparison of MARS, BPN and SVR in an Emerging Market,” In: Proceedings of the IEEE IEEM, pp. 2343-2347, 2009

[5] K. S. Kannan, P. S. Sekar, M. M. Sathik, P. Arumugam, “Financial Stock Market Forecast using Data Mining Techniques,” In: Proceedings of the International Multiconference of Engineers and computer scientists, pp. 555-559, 2010

[6] Yanjie. Hu, Juanjuan. Pang, “Financial crisis early warning based on support vector machine,” In: International Joint Conference on Neural Networks, pp. 2435-2440, 2008

[7] Kuan-Yu. Chen, Chia-Hui. Ho, “An Improved Support Vector Regression Modeling for Taiwan Stock Exchange Market Weighted Index Forecasting,” In: The IEEE International Conference on Neural Networks and Brain, pp. 1633-1638, 2005

[8] S. Xue-shen, Q. Zhong-ying, Yu. Da-ren, Qing-hua, Hu. Hui, Zhao, “A Novel Feature Selection Approach Using Classification Complexity for SVM of Stock Market Trend Prediction,” In: 14th International Conference on Management Science & Engineering, pp. 1654-1659, 2007

[9] B. Debasish, P. Srimanta, C.P. Dipak, “Support Vector Regression,” In: Neural Information Processing – Letters and Reviews Vol. 11, No. 10, pp. 203-224, 2007

[10] Chih-Wei. Hsu, Chih-Chung. Chang, Chih-Jen. Lin, “A Practical Guide to Support Vector Classification,” Initial version: 2003, Last updated version: 2010

[11] U. Thissena, R. van. Brakela, A.P. de. Weijerb, W.J. Melssena, L.M.C. Buydensa, “Using support vector machines for time series prediction,” In: Chemometrics and Intelligent Laboratory Systems 69, pp.35– 49, 2003

[12] Lijuan. Cao, “Support vector machines experts for time series forecasting,” In: Neurocomputing 51, pp. 321 – 339, 2003

[13] J. Smola. Alex, SchoLkop. Bernhard, “A tutorial on support vector regression,” In:

Statistics and Computing 14, pp.199–222, 2004

[14] Ahmad Kazema, Ebrahim Sharifia, Farookh Khadeer Hussainb, Morteza Saberic, Omar Khadeer Hussaind, “support vector regression with chaos-based firefly algorithm for stock market price forecasting ,” In: Applied Soft Computing 13 , pp. 947–958, 2013