# Comparing Profitability of different Machine Learning Algorithms in Financial Markets


| Akshat | Anurag Gulati | Ayush Madan | Kabir D. V. Baghel |
|---|---|---|---|
| 2020172 | 2020176 | 2020187 | 2020564 |
| IIITD ECE | IIITD ECE | IIITD ECE | IIITD CSAI |



## Abstract

*Until the late 1990s, trading decisions across all asset classes entirely depended on a person's analysis. Around the turn of the century, with the advent of faster machines, trading was automated with hard-coded strategies. Around the same time, machines learned to generate trading signals based on past data and strategies using learning algorithms. This project analyzes the trading signals generated by different machine learning algorithms and compares their potential profits to the original strategy given in the dataset and to each other. The learning and testing have been done specifically for the Indian stock market, but one can extrapolate them to other asset classes and markets.*


## 1. Introduction

Traders worldwide rely on fundamental and technical analysis while on the trading floor of an exchange. For instance, in stock markets, the former is based on understanding the financial records of a given company. In the forex market, it deals with the underlying economic and political policies of the two nations. In contrast, demand and supply analysis can be taken under the fundamental umbrella of the commodities market. On the other hand, technical analysis always deals with the patterns and statistics of price movement in any market. These statistical parameters are standard across different asset classes, making technical analysis consistent in them.

Over the past decades, researchers have employed machine learning algorithms to generate profitable trade signals. Through this project, the authors provide a comparative study on the profitability of learning algorithms in the arena of the Indian stock market, the stock choice for which is Tata Consultancy Services (TCS), listed on the Bombay Stock Exchange (BSE) as well as the National Stock Exchange (NSE) of India.

### 1.1. Related work

Since the mid-2000s, researchers have been developing intelligent trading systems using countless learning algorithms. In [1], the author proposes a hybrid model that uses decision trees to enhance the conventional filter rule. [2] proposes a better approach to [1] by incorporating future information into the criteria for clustering the trading points. The authors of [3] have used a nearest neighbors classifier built on conventional technical analysis.

The most recent developments in the field have been using deep learning for signal generation The authors of [4] study the use of artificial neural networks (ANNs) in this field, while [5] proposes the use of ensemble learning based on neural networks.

More advanced studies focus on using genetic algorithms and programming for trading. Incorporating news-based analysis using natural language processing is another approach that, when combined with one of the above methods, can give out a mix of fundamental and technical analysis. However, these advanced techniques are well beyond the scope of this project.

### 1.2. Overview

This project provides a comparative study of the below-mentioned machine learning algorithms and their profitability in the Indian stock market:

a. Logistic Regression
b. Naïve Bayes
c. Decision Tree
d. Random Forest
e. Multilayer Perceptron

The benchmark stock chosen for the same is Tata Consultancy Services (TCS), one of the largest companies by market cap on both BSE and NSE.

### 1.3. Technical indicators

"Technical indicators are heuristic or pattern-based signals produced by the price, volume, and/or open interest of a security or contract used by traders who follow technical analysis" (Investopedia [6]). Broadly, technical indicators can be divided into four main categories: trend, volume, volatility, and momentum indicators.

From a wide range of indicators available, this project utilizes a dataset that contains 15 of them.

a. *Trend indicators*

i. Commodities Channel Index (CCI)
ii. Exponential Moving Average (EMA)
iii. Moving Average Convergence Divergence
iv. Simple Moving Average (SMA)
v. Volume Weighted Moving Average (VWMA)

b. *Volume indicators*
i. Accumulation/Distribution Index (ADI)
ii. Ease of Movement (EMV)
iii. Money Flow Index (MFI)

c. *Volatility indicators*
i. Average True Range (ATR)

d. *Momentum indicators*
i. Awesome Oscillator (AO)
ii. Rate of Change (ROC)
iii. Relative Strength Index (RSI)
iv. Stochastic RSI %D
v. Stochastic RSI %K
vi. William's %R

The above indicators are calculated over a period of *n* days. Usually for short term, *n = 12 or 26*, while for mid-term and long-term *n = 50* and *200* respectively.

## 2. The Data

### 2.1. Dataset

The raw dataset obtained from [7] contains the price action data for TCS from January 14, 2002, to June 5, 2020. Using the raw data, the authors have calculated the technical indicators mentioned in section 1.3 and created a new dataset for analysis and learning, which contains a total of 40 features. Based on the general rule of thumb, buy low and sell high, the new dataset adds another column, Signal, that will serve as the label for all supervised learning algorithms.

### 2.2. EDA and feature selection

Exploratory data analysis (EDA) is a technique that helps extract insights from the data. Based on the mutual information gain of the features, the project selects the top 15 for further analysis (Figure 1). Out of these, the ones that are fully correlated (correlation coefficient = 1) are dropped, and only one of them is kept intact.

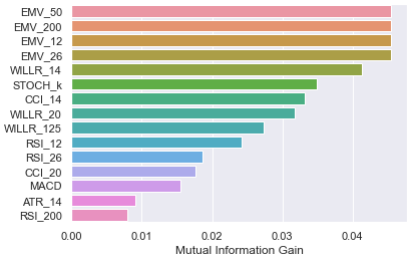



Figure 1: Top 15 features based on mutual information gain.

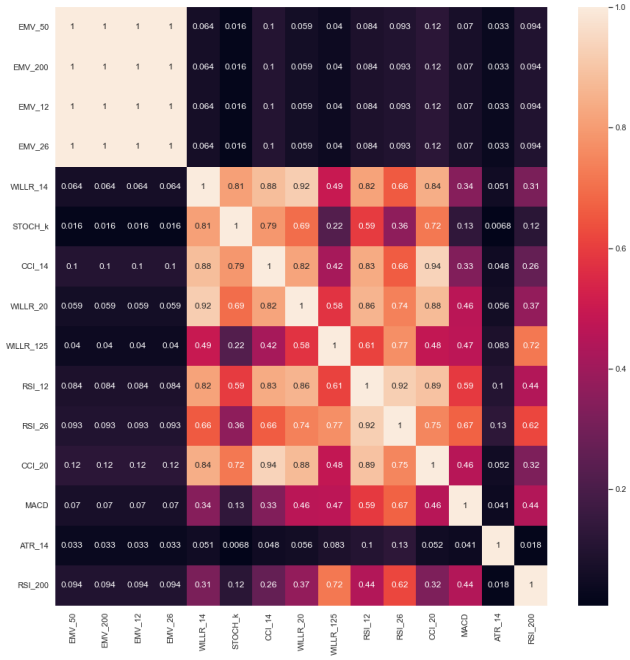



Figure 2: Observing the absolute correlation among the top 15 features. It is visible that EMV values over different time periods are fully correlated, hence selecting only one of these columns shall be sufficient for learning.

From the abovementioned analysis, the features finally selected for learning, in decreasing order of information gain, are as follows:

a. 50-days Ease of Movement
b. 14-days William's %R
c. Stochastic RSI %K
d. 14-days Commodities Channel Index
e. 20-days William's %R
f. 125-days William's %R
g. 12-days Relative Strength Index

## 3. Learning and Analysis

It is worth reiterating that this project aims to compare the profitability of different machine learning algorithms in the financial markets, and that the experimental focus is on the Indian stock market but the concepts can be extrapolated to other asset classes and markets. In the upcoming sections of this manuscript, the following learning algorithms have been deployed and analyzed for profits:

a. Logistic Regression
b. Decision Trees
c. Random Forests
d. AdaBoost
e. Naïve Bayes
f. Multilayer Perceptron

### 3.1. Logistic Regression

Logistic regression is one of the simplest classification algorithms, most deployed for binary classification problems. However, the same can be adjusted to deal with multiclass datasets. This project requires any model to classify a given price point as 'buy,' 'sell,' or 'wait.' A simpler explanation can be provided in binary classification. Say there are two classes to choose from: 0 (class A) and 1 (class B). In such a case, logistic regression yields the following decision boundary:

$$\sigma(\theta^T x) = 0.5 \quad (1)$$

where $\sigma(z)$ is the famous sigmoid function, defined as:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (2)$$

In addition, $\theta$ and $x$ are the weight and feature vectors respectively.

### 3.2. Decision Trees

A decision tree is a highly interpretable learning model that can be deployed for both regression and classification tasks. The algorithm uses some metric to determine the best feature around which a split of decision can be made at every level. One of the most popular metrics is the entropy measure, which is defined as follows:

$$H(Y) = -\sum_i \mathrm{P}(Y = y_i) \log_2 \mathrm{P}(Y = y_i) \quad (3)$$
$$H(Y|X) = -\sum_i P(x_i) \sum_j \mathrm{P}(y_j|x_i) \log_2 \mathrm{P}(y_j|x_i) \quad (4)$$

where $Y$ and $X$ are the label and feature vectors respectively. Here, (3) gives the entropy of the entire label vector while (4) gives the entropy of $Y$ given some feature $X$, i.e., it marks how important that feature at a given level.

The most important issue that arises while training a decision tree is overfitting, which can be resolved using multiple methods such a pruning or training random forests.

### 3.3. Random Forests

As a forest consists of many trees, Random Forests contains many decision trees. Every decision tree gives us a prediction, and the prediction with the highest count gets chosen as the final model's prediction.

The bagging-type ensemble is used for Random Forests. It chooses random samples from the data set with replacements. After that, we generate results using the models. Train-Test split isn't required in Random Forest as 30% of the data is always unseen.

### 3.4. AdaBoost

"AdaBoost also called Adaptive Boosting is an Ensemble Learning Method. Decision Trees of a single level are commonly used as base estimator for the AdaBoost Algorithm. It builds a model and gives equal weights to all the data points. It then assigns higher weights to points that are wrongly classified. Now all the points which have higher weights are given more importance in the next model. It will keep training models until and unless a lower error is received." - *[9]*

### 3.5. Naïve Bayes

Naïve Bayes is based on the Bayes' theorem. If there are multiple features (say n), and the independent and dependent feature vectors are X = ($x_1$, $x_2$, … $x_n$) and y respectively, then Bayes' theorem states:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \quad (5)$$

Or,

$$P(y|x_1, x_2 \dots x_n) = \frac{P(x_1, x_2 \dots x_n|y)P(y)}{P(x_1, x_2 \dots x_n)} \quad (6)$$

The defining property of Naïve Bayes is that it makes a 'naïve' assumption: all the features are independent of each other. This reduces the complexity of the equation given above as the term $P(x_1, x_2 \dots x_n|y)$ becomes $P(x_1|y)P(x_2|y)..P(x_n|y)$ due to independence. Hence, the simplified equation:

$$P(y|x_1, x_2 \dots x_n) = \frac{P(x_1|y)P(x_2|y)..P(x_n|y)\,P(y)}{P(x_1)P(x_2)..P(x_n)} \quad (7)$$

The above terms are easier to calculate, making this model a good choice for fast calculations.

### 3.6. Multilayer Perceptron

"A multilayer perceptron (MLP) is a fully connected class of artificial neural network (ANN)." *(Wikipedia)*

The fundamental building block of an MLP is a neuron, which gives out a non-linearized weighted sum of its inputs, a particular case of which is the perceptron learning algorithm, where the activation function is a step function. Three kinds of layers exist in an MLP, namely, input, hidden, and output layers. There can be any number of hidden layers, each of which can have any number of neurons. The most important feature must be noted is that each neuron in a hidden layer is connected to each neuron in the next and the previous layers, which is why MLPs fall under the umbrella of fully connected neural networks.

## 4. Methodology

The authors split the dataset into training and testing datasets in the ratio 70:30. Since the data is time-series, simply taking the first 70% of samples as training data and the rest 30% for testing would be a good choice. Scikit-learn library was used for different learning models.

In any trading system, the best metric to judge performance is the profit generated by the model. The authors run the trained model on testing data to calculate the profit over the next 100 trading days. It is assumed that the trading system buys a unit of stock at each 'buy' signal, and at each 'sell' signal, it sells a unit of stock. If the system buys stocks before a 'sell' signal, it should sell all the stocks and move on to the next signal and vice-versa. The authors also assume that any current position is squared off at the end of the 100 days.

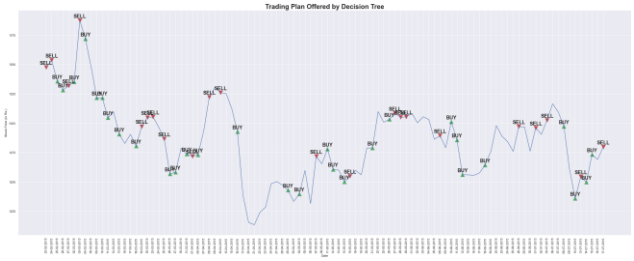
$$Profit = Selling\ Price - Buying\ Price \quad (8)$$

## 5. Simulation Results

Following are the trade signals obtained from different learning algorithms:

### 5.1. Logistic Regression


### 5.2. Decision Trees

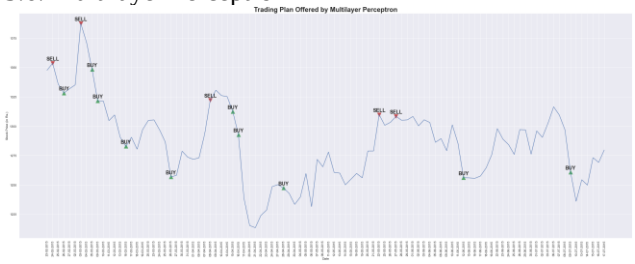



### 5.3. Random Forests


### 5.4. AdaBoost


### 5.5. Naïve Bayes


### 5.6. Multilayer Perceptron




## 6. Profits and Performance

| S.No. | Model/Algorithm | Profit (back testing) |
|---|---|---|
| 1 | Logistic Regression | 22.13% |
| 2 | Decision Tree | 27.44% |
| 3 | Random Forest | 29.01% |
| 4 | AdaBoost | 7.6% |
| **5** | **Naïve Bayes** | **120.39%** |
| 6 | Multilayer Perceptron | 21.2% |

## 7. Conclusion and Future Work

In this project, the authors have successfully tested six different learning algorithms on the Indian stock market, with the testbench stock being Tata Consultancy Services (TCS). Out of these, Naïve Bayes seems to be working the best, with a profit of roughly 120% over 100 days.

The authors would work in future on more algorithms on the dataset. As of the midterms, it can be safely concluded that Naïve Bayes performs as the best model.

References


[1] Lin, C. H. (2004). Profitability of a filter trading rule on the Taiwan stock exchange market. Master thesis, Department of Industrial Engineering and Management, National Chiao Tung University

[2] Wu, M. C., Lin, S. Y., & Lin, C. H. (2006). An effective application of decision tree to stock trading. *Expert Systems with applications*, *31*(2), 270-274.

[3] Teixeira, L. A., & De Oliveira, A. L. I. (2010). A method for automatic stock trading combining technical analysis and nearest neighbor classification. *Expert systems with applications*, *37*(10), 6885-6890.

[4] Fernandez-Rodrıguez, F., Gonzalez-Martel, C., & Sosvilla-Rivero, S. (2000). On the profitability of technical trading rules based on artificial neural networks: Evidence from the Madrid stock market. *Economics letters*, *69*(1), 89-94.

[5] Chang, P. C., Liu, C. H., Fan, C. Y., Lin, J. L., & Lai, C. M. (2009, September). An ensemble of neural networks for stock trading decision making. In *International conference on intelligent computing* (pp. 1-10). Springer, Berlin, Heidelberg.

[6] Investopedia: www.investopedia.com/terms/t/technicalindicator.asp

[7] Yahoo Finance: https://finance.yahoo.com/

[8] Technical Analysis Library: https://technical-analysis-library-in-python.readthedocs.io/en/latest/

[9] AdaBoost: https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/

[10] https://economictimes.indiatimes.com/definition/squaring-off


Appendix

1. Square off: Squaring off is a trading style used by traders, in which a trader buys or sells a particular quantity of an asset (mostly stocks) and later reverses the transaction. [10]
2. Shorting: When a trader sells a stock first and buys it later, it is known as shorting or a short position.
3. Position: Stocks owned or shorted by a trader.

Contributions:

1. Ayush Madan: AdaBoost
2. Akshat: Naïve Bayes
3. Anurag: Logistic Regression, MLP
4. Kabir: Decision Trees, Random Forests

Most of the work is done collaboratively with equal efforts from all the members of the group.