School of Computing, Engineering and Digital Technologies
Department of Computing and Games
Teesside University
Middlesbrough TS1 3BA

Resist the temptation to trade:

Lo “Buy-n-Hold" the power of passive investing

Submitted in partial requirements for the degree of MSc Data Science (advance practice)

Date: May 9, 2023

Ashish Kakran

Supervisor: Dr Ismail Kazmi

Acknowledgements

I would like to thank Dr Ismail Kazmi for providing me with his continued guidance and support throughout the development of this project.

And thanks to Mom and Dad, I graduated.

Resist the temptation to trade:

Lo “Buy-n-Hold" the power of passive investing

Ashish Kakran

Abstract

In recent years, the use of machine learning models to algorithmically trade financial assets has gained a lot of popularity among the quantitative research community and finance industry. Additionally, a lot of retail investors actively trading using technical analysis on asset pricing data. A lot of research papers for building predictive models for generating returns on financial assets have been published without understanding fundamentals of financial markets creating a gap (Hsu *et al.*, 2016) between machine learning and quantitative trading community.

A popular known result Efficient Market Hypothesis (EMH) developed by (Fama, 1965) argues against the predictability of stock prices in efficient markets. This generates doubts regarding the performance of these sophisticated models and the need for such complex techniques where a passive strategy of buying and holding assets to maturity might lead to better returns.

Therefore, the purpose of this project is to experimentally determine under the empirically known results of efficient financial markets, whether the returns obtained from actively trading financial assets based upon trading signals generated from artificial intelligence techniques are statistically significant compared to returns from passive buy and hold strategies on the same portfolio or the market index.

Contents

Introduction

1.1 Motivation

Participants in financial markets such as stock markets are always looking to maximizing their returns on investments. Their objective is to predict the market movements based on information such as participants' sentiments, news and historical trends and make the calls to buy or sell accordingly. Two ways to make investments are either to actively trade financial assets or to take a more passive approach where one would buy and hold the asset to maturity to meet their long-term financial targets. Moreover, participants can choose to trade individual stocks or choose to invest in mutual funds, ETFs and index funds that track the Market Index, a hypothetical portfolio of investment holdings that represents a segment of the financial market.

In recent years, a lot of papers have been published regarding prediction of stock prices using sophisticated machine learning and deep learning models. Earlier work such as (Fama, 1965) proposed that stock prices follow a random walk and trying to predict market movements is equivalent to predicting success, be it head or tail, on tossing a coin. This random walk hypothesis in stock market movements has been of high interest to researchers since no formal proof exists as such.

On the other hand, several researchers have come up with various claims rejecting random walk hypothesis. Recently published papers using state-of-the-art models claim to have high accuracy without any sort of benchmarking. From an investment perspective, it only makes sense to trade using algorithms if it can generate higher returns over the long term than a simple buy-and-hold strategy.

Thus, the motivation behind this project is to backtest different statistical and machine learning algorithms and compare their performance with returns from the buy-and-hold strategy to decide which of the trading strategies is superior.

1.2 Hypothesis formulation

The research hypothesis of this project is to answer whether actively trading using machine learning strategies generates higher returns or the good old school approach of passive investing.

The statistical hypothesis can be stated as follows:

Given mean excess returns $\mu_r$from random walk null model, do observed mean excess returns from the best machine learning model μ_l, exceeds the former or not.

More precisely,

Null Hypothesis $\boldsymbol{H_0}$: $\mu_l < \mu_r$ (mean excess returns on random walk model is higher than those of ML model)

Alternative Hypothesis: $\boldsymbol{H_a}$: $\mu_l \geq \mu_r$ (mean excess returns on best ML model is higher than those of random walk model)

The mean excess returns can either be calculated by differencing passive returns from the same asset that is being traded or returns on a market index fund such as Vanguard Total Stock Market Fund (VTI). The hypothesis is examined from both passive investing and indexing perspective.

1.3 Literature review

Earlier work in finance such as (Fama, 1970) hypothesized with empirical evidence that asset prices in efficient markets follow random walk. The study was updated later (Fama, 1991) to address additional research such as (Lo and MacKinlay, 1988) which claimed that stock market prices do not follow random walk.

However, there has been no concrete theoretical proof of efficient market hypothesis and researchers disagree on whether the stock prices follow random walk or not. Researchers have used various methods such as variance ratio test and surrogate method (Nakamura and Small, 2007) to test the random walk in the financial data. Some of the critiques have rejected the hypothesis to some extent (Jung and Shiller, 2005). Recent work such as (Andrew W. Lo, 2004) have developed what is called Adaptive Market Hypothesis which argues that whilst “rational” players in the market lead to an efficient market making stock prices unpredictable, the investors, however, can be irrational and decisions based on irrationality leads to inefficiencies in the market which could be capitalized on.

With the advent of Artificial Intelligence, machine learning and deep learning models are being used to algorithmically trade in the financial markets. (Ballings *et al.*, 2015) discuss multiple classifiers in isolation for market movements and (Kara, Acar Boyacioglu and Baykan, 2011) presents neural network model using historical stock prices in emerging market. (Tay and Cao, 2001) proposes SVM methodology to predict financial returns, later updated (Cao and Tay, 2003)

arguing that adapting parameters of SVM models can improve traditionally trained SVM models. (Thenmozhi, 2014) combines traditional time series model ARIMA to machine learning models SVM, neural network and random forest models. (Lin and Chen, 2018) uses long term short memory (LSTM) Neural network model to forecast short term stock prices. (Cho *et al.*, 2019) compares more sophisticated neural network models such as wavenet, Seq2Seq and LSTM models. (Li, 2022) improves LSTM incorporating self-attention mechanism from the data.

Some of the non-traditional approaches include models based on sentimental analysis from news and social media websites to capture inefficiencies based on irrationality of participants in the market. (Oliveira, Cortez and Areal, 2013) models stock market movements based on the investor sentiments on social media website twitter. Alternatively, cross sectional data has been utilized by (Abe and Nakagawa, 2020) as fundamental analyst would approach the problem instead of solely relying on the technical indicators. (Huang, 2018) models financial trading as Markov decision process and uses deep recurrent Q-learning network to derive over 6.4% return.

Whilst a full review of the machine learning based models being used to predict market movements and stock prices is impossible. A common theme that seems to emerge from quick survey is that only a very few models have been tested against the efficient market hypothesis and this has led to a lot of obscurity around the practical implications of these models in the real trading world. This gap between machine learning research and financial trading has been very clearly pointed out by (Hsu *et al.*, 2016) where they propose how current machine learning based approaches should be tested in the framework proposed by (Fama, 1970). (Hsu *et al.*, 2016) developed various models, however, these models are somewhat inconclusive due to the data unavailability back then. This thesis extends upon their work and deploy models based on some of the recent advances in machine learning and deep learning.

1.4 Thesis Outline

This thesis is divided into five chapters.

Chapter 1 introduces the problem as identified based upon literature review and then formulates the hypothesis this project aims to test.

Chapter 2 provides background on the technical analysis and algorithmic modelling while also explaining methods that were used in the project.

Chapter 3 explains experimental setup and modelling process in the context of process of standard data science-based projects.

Chapter 4 discusses the results observed from experimentation, their implications, limitations and model interpretation.

Finally, **chapter 5** concludes this thesis.

Background

2.1 Technical Analysis and Algorithmic trading

Technical analysis of financial securities and derivatives has been used since long ago as far as the stock market exists. The concept was popularized by Charles Dow when he analyzed American stock market data using patterns and other indicators such as rolling mean. In modern financial markets, the financial instruments pricing data such as stock prices data augmented with traditional technical indicators, behavioral indicators such as participants sentiments can be used to build machine learning models and thus these securities can be traded algorithmically.

The machine learning based trading strategy relies upon data sources that contain predictive trading signals for the target investment set. Following appropriate preprocessing and feature engineering, the datasets can be used to train ML models which can predict asset returns or other inputs influencing trading strategy in the market. These predictions, subsequently, can be turned into buy or sell signals based on human knowledge or automated rules to help derive long or short positions.

In this project, this signal generation problem has been translated into equivalent supervised classification problem where the prediction from machine learning models can be used to decide to take long or short positions.

Machine learning for trading workflow starts with collection of data from diverse sources such as end of the day pricing data, fundamental data and alternative data. After preprocessing of these data points, the features for building models to generate the signals can be engineered and their relationship could be explored. Given these set of features, the models are trained and validated on hold out sample and the predictions are made on test set. This strategy is backtested on a portfolio of assets. After the asset selection real time trades can be made by a downstream trading system.

This Machine learning model workflow for trading inspired from (Jansen, 2020) can be visualized as follows:

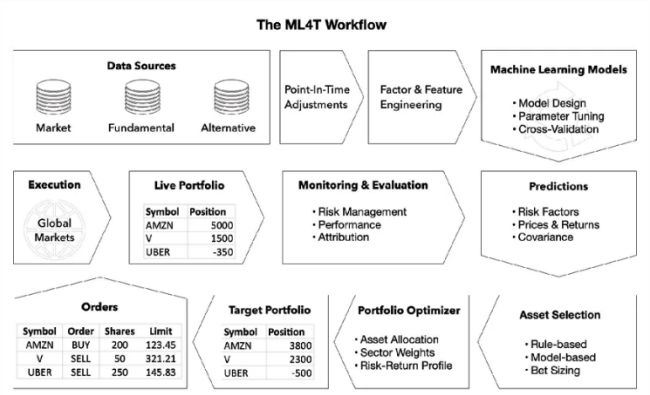

Figure 1.1: The machine learning for trading workflow

For the purpose of this project, only classification models for supervised learning have been experimented with in order to generate trading signals. These models are explained below as follows.

2.2 Logistic regression

A logistic regression model is a linear model (Geron, 2019) which estimates the probability that an instance belongs to a particular class.

For binary classification, the probabilities are calculated using sigmoidal function. For given feature vector $\theta \in R^d$ and training instance $x^{(i)} \in R^d$

$$\hat{p} = h_\theta(x) = \sigma(\theta^T x)$$

Where,

$$\theta^T x = \theta_0 + \Sigma_{j=1}^{d} \theta_j x_j$$

The logistic is a sigmoid function that outputs a number between 0 and 1.

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

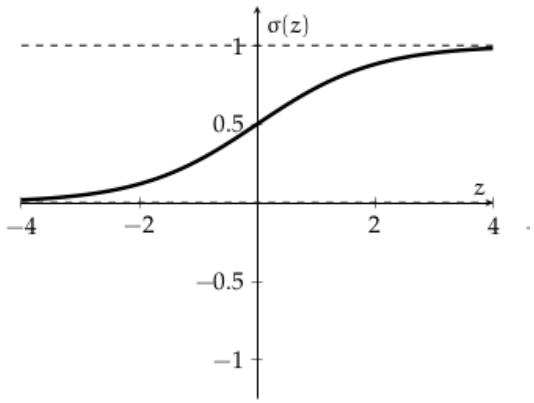

Figure 2.1: The sigmoid function

These probabilities can be used to classify an instance to a particular class based on threshold value. Scikit-learn by default uses 0.5 as the threshold.

The model parameters can be estimated using gradient ascent strategy.

$$\theta \coloneqq \theta + \alpha \nabla_\theta l(\theta)$$

Where

$$\ell(\theta) = \log L(\theta)$$

$$= \Sigma_{i=1}^{n} y^{(i)} \log h\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h\left(x^{(i)}\right)\right)$$

2.3 Support Vector Machine

A support vector machine is a numerical optimization-based machine learning algorithm (Hastie, Friedman and Tibshirani, 2009) which can perform linear or nonlinear classification.

For linear classification, the SVM model predicts the class of a new instance $x^{(i)}$ by computing the decision function $w^T x + b$.

If the decision function's value is positive, the classifier predicts class of $x^{(i)}$ to be positive (1) or else it is negative (-1), that is,

$$h_{w,b}(x) = g(w^T x + b)$$

$$g(z) = 1 \; if \; z \geq 0, \; and \; g(z) = -1 \; otherwise$$

The parameters w, b can be learned by solving following convex quadratic optimization problem with linear constraints:

$$\min_{w,b} \quad \frac{1}{2}\left|\left|w\right|\right|^2$$

$$s.t\, y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1,\ i = 1, \ldots, n$$

Geometrically, this is equivalent of solving for parameters w, b that would result in largest possible geometric margin with respect to the training set, where geometric margin γ with respect to the training set $S = \left\{\left(x^{(i)}, y^{(i)}\right);\ i = 1, \ldots, n\right\}$ is smallest of geometric margins on the individual training examples:

$$\gamma = \min_{i=1,\ldots,n} \quad \gamma^{(i)}$$

Where $\gamma^{(i)} = \frac{w^T x^{(i)} + b}{||w||} = \frac{w^T}{||w||} x^{(i)} + \frac{b}{||w||}$ (former term defines slope of the hyperplane in the space spanned by training vectors and latter term defines intercept of the hyperplane)

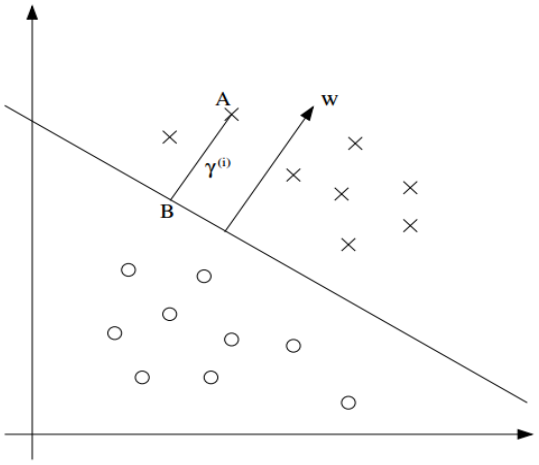

Figure 2.2: Optimal margin classification (Ma and Ng, 2007)

2.4 Random Forest

A Random Forest model is an ensemble of Decision trees (Tin Kam Ho, 1995), trained via bagging or pasting method.

Bagging (short for bootstrap aggregating) is a technique of ensembling the predictors of same training algorithm over different subsets of the training set chosen with replacement. Whereas in case of pasting, on the other hand, these subsets are chosen for training without replacement.

This sampling and training process is represented as follows:

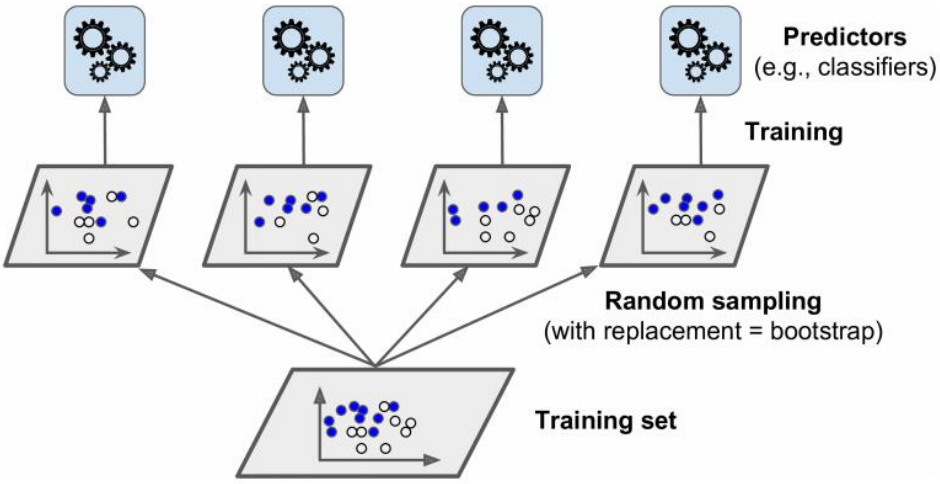

Figure 2.3: bagging/pasting for sampling and training (Geron, 2019)

2.5 Neural Network

A neural Network is a network of basic computation units called neurons, modeled after neural structure in human brains, which takes instance $x \epsilon R^m$as input maps it to a single output value $a \epsilon R$It is parameterized by a vector of weights $[w_1, \ldots., w_m] \epsilon R^m$ and a bias term $w_0 \epsilon R$.

For nonlinear output value, the parametrized value is activated by a non-linear differentiable activation function $f: \Re \mapsto \Re$.

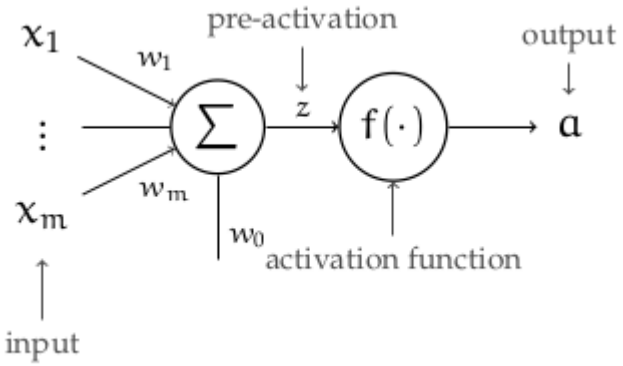

Figure 2.4: A single neuron function

The function represented by the neuron is expressed as:

$$a = f(z) = f\left(\Sigma_{j=1}^{m} x_j w_j + w_0\right) = f\left(w^T x^{(i)} + w_0\right)$$

A single layer is a set of such neurons, possibly fully connected where inputs to each unit in the layer are the same. In general, a layer has input $x \epsilon R^m$ and output $a \epsilon R^n$.

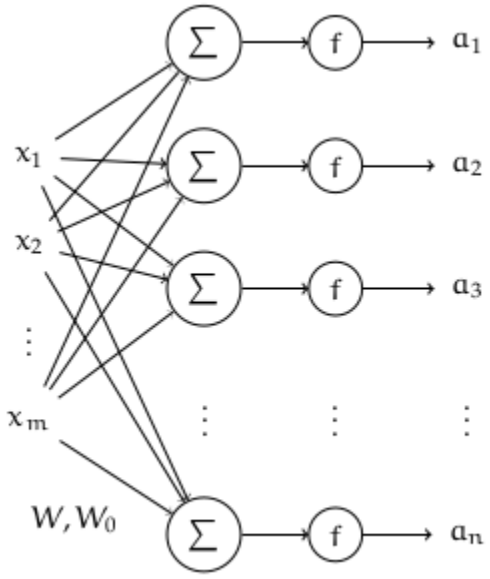

Figure 2.5 A network of neural units

Combining layers of non-linear neural units into complex architectures results in what is called a neural network.

There are several choices available when it comes to choosing activation functions of neurons. Scikit-learn by default uses Rectified linear unit (ReLu) function defined as $ReLu(z) = \max(0, z)$ for inner layers. The choice of activation function for outer layer depends on the problem for which the neural network algorithm is being used for. For regression no activation is required whilst for classification it could be Sigmoid function as defined earlier or SoftMax function in case of multiclass classification.

The SoftMax function takes a vector $z \epsilon R^n$ and generates an output vector $A \epsilon [0,1]^n$ such that $\Sigma_{i=1}^{n} A_i = 1$ which we can interpret as a probability distribution over n items (Geron, 2019)

$\widehat{p_k} = \sigma(z)_k = \frac{\exp(z)}{\Sigma_{j=1}^{K} \exp(z_j)}$, K is the number of classes.

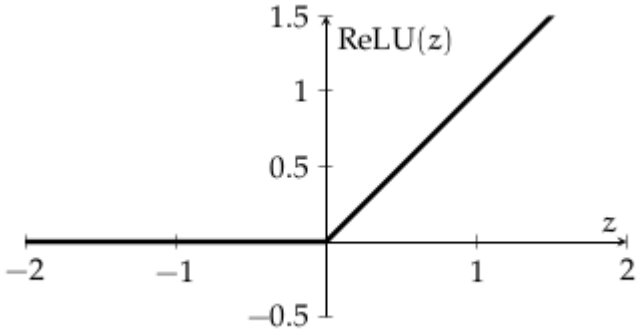

Figure 2.6: A Rectified linear unit activation function

The class prediction $A \epsilon [0,1]^n$ can then be determined by assigning the class associated with highest estimated probability, that is,

$$\hat{y} = \text{argmax}_{\text{k}}(\widehat{p^k})$$

The parameters of the neural network can be determined by following the gradient of loss function of the network in backward propagation order as gradients with respect to weights of neurons in inner layer can be expressed in terms of gradients with respect to layer forward to corresponding layer.

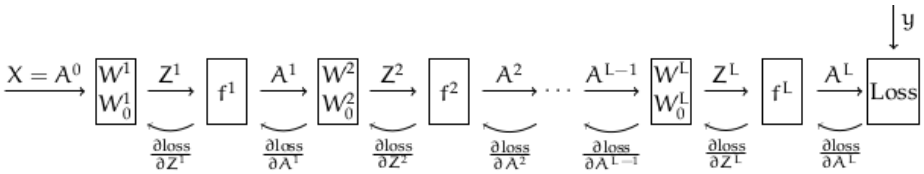

Figure 2.7: The Backpropagation process for estimating model parameters

2.6 Ensemble algorithms

For classification purposes, the algorithms can be combined into an ensemble classifier to take advantage of the type of error they make individually and combine these weak learners into a strong ensemble classifier. Some of the most common ensemble learners are discussed as follows:

2.6.1 Voting classifier

A voting classifier consists of several classifiers of different types, each predicting its own estimated class for instance in the training set. Depending upon the nature of output of these individual classifiers their predictions can be aggregated to output as single estimate.

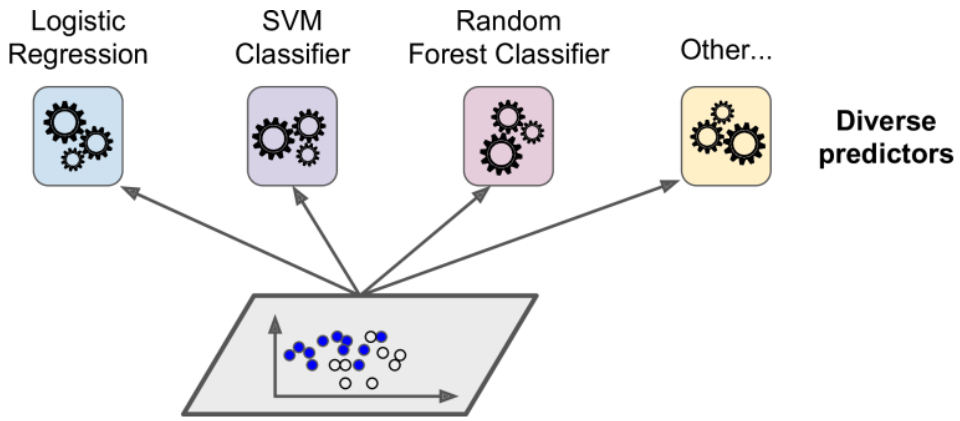

Figure 2.8: Combining diverse set of predictors into Voting classifier

In cases where the probability of an instance belonging to a class is maximized the ensembled voting classifier is said to be soft voting classifier. In the other case, where the most frequent class output by different predictors is aggregated as output, the voting classifier is said to be hard voting classifier. In this project, a soft voting approach is taken as it achieves higher accuracy.

2.6.2 Adaptive Boosting

Alternative to training different classifiers and ensembling them, a series of classifiers can be trained sequentially each correcting on the mistakes of the previous classifier's mistakes. To build an AdaBoost classifier, a base classifier such Decision Tree is trained and used to make predictions on the training set. The relative weight of misclassified instances is boosted, and subsequent classifiers are trained (or adapted) on misclassified instances by the previous classifier until no further improvement is observed.

The process of Adaptive boosting can be described as follows:

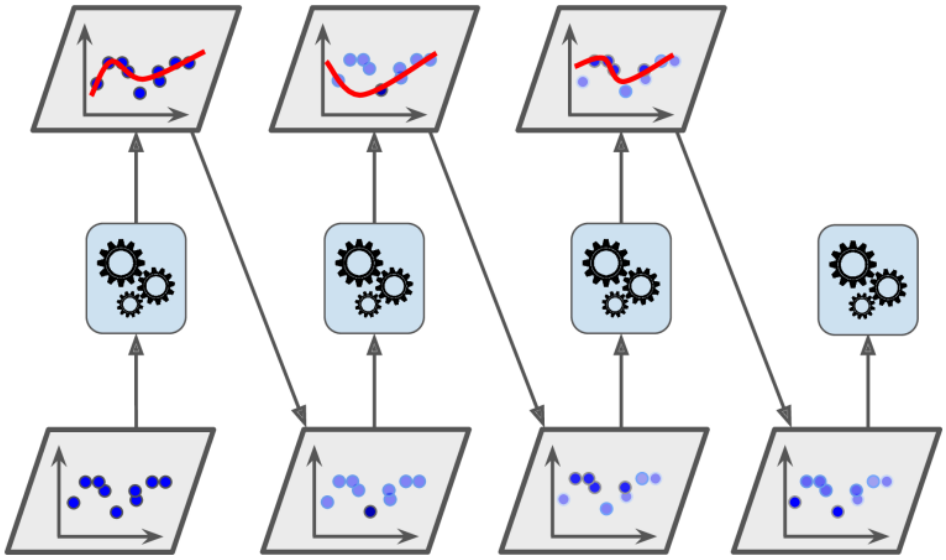

Figure 2.9: AdaBoost sequential training with instance weights penalized (Geron, 2019)

2.6.3 Gradient Boosting

The gradient Boosting algorithm (Friedman, 2001), like adaptive boosting, sequentially adds predictors to an ensemble, each one correcting its predecessor. However, in gradient boosting, these new learners are fit to residual errors made by the previous predictor.

Experimental setup and Modelling

3.1 Resources and tools

The project is implemented using python 3.x. The main libraries used for data modelling are numpy (Harris *et al.*, 2020), pandas (team, 2023) and yfinance for data collection. For building machine learning strategies scikit-learn (Pedregosa *et al.*, 2011) was used and for backtesting and portfolio performance evaluation, Backtrader and pyfolio were used respectively. Additionally, statmodels (Perktold *et al.*, 2023) , SciPy (Virtanen *et al.*, 2020), Matplotlib (Hunter, 2007) and seaborn (Waskom, 2021) were used for exploratory data analysis.

3.2 Overview of the dataset

The dataset consists of common stocks pricing data of 6 companies from yahoo finance, fetched using yfinance library. The choice of companies was decided by their market capitalization as listed on Nasdaq (*Stock screener, Nasdaq.*)

Symbol (ticker)	Name	Market Cap	Industry
AAPL	Apple Inc. Common stock	2.9 trillion (Mega)	Technology
MSFT	Microsoft Corporation Common stock	2.2 trillion (Mega)	Technology
DIS	Walt Disney Company Common Stock	184 billion (Large)	Movies/Entertainment
NKE	Nike Inc. Common Stock	194 billion (Large)	Shoe Manufacturing
AAL	American Airlines Group Inc. Common stock	8 billion (Medium)	Air Freight/Delivery services
ZG	Zillow Group, Inc. Class A Common stock	9 billion (Medium)	Real estate/business services
VTI	Vanguard Total Stock Market Index Fund	1.25 trillion (net assets)	finance

Table 3.1: A stock portfolio selection

In general, algorithmic trading utilizes four types of data as described as follows:

	Structured	Unstructured
Historical	End-of-day OHLCV data	Financial news
Real-time	Bid/ask prices for exchange (exchange order book)	Sentimental data from social media websites

Table 3.2: Type of financial data (Jansen, 2020)

Keeping consistency with the scope of the project, only end-of-day OHLCV data available from public websites like yahoo finance was used. The other types of data are hard to acquire without premium. Besides, as stated earlier according to Dow theory (Hayes, 2022) all the information about market expectations is already built into the stock prices. That's not to say, there aren't models being used based on fundamental and alternative data but due to constraints on data availability only pricing data is used as shown below.

Date	Open	High	Low	Close	Adj Close	Volume
2010-01-04	30.620001	31.100000	30.590000	30.950001	23.623901	38409100
2010-01-05	30.850000	31.100000	30.639999	30.959999	23.631538	49749600
2010-01-06	30.879999	31.080000	30.520000	30.770000	23.486511	58182400
2010-01-07	30.629999	30.700001	30.190001	30.450001	23.242250	50559700
2010-01-08	30.280001	30.879999	30.240000	30.660000	23.402550	51197400

Figure 3.1: end-of-day OHLCV price data on MSFT (fetched from finance.yahoo.com)

3.3 Data Exploration

As expected from any data science projects, a critical step of the data science process is exploratory data analysis (Tukey, 1977).

To that end, following were the aims of data exploration:

- Describe and summarize various statistical measures of the data for better understanding
- To understand distribution of daily returns and detect outliers
- Examining the influence of past data on future returns
- To understand relationship between OHLCV features in the data with returns
- Relationship of returns on individual stock/portfolio with market Index fund
- Gain possible insights about what models might be useful and decide about hypothesis testing and metrics required to evaluate models and results

The daily return on MSFT seems to be centered around 0 with bounded dispersion.

Figure 3.2: Daily returns on MSFT

The following figure shows yearly distributions of daily returns.

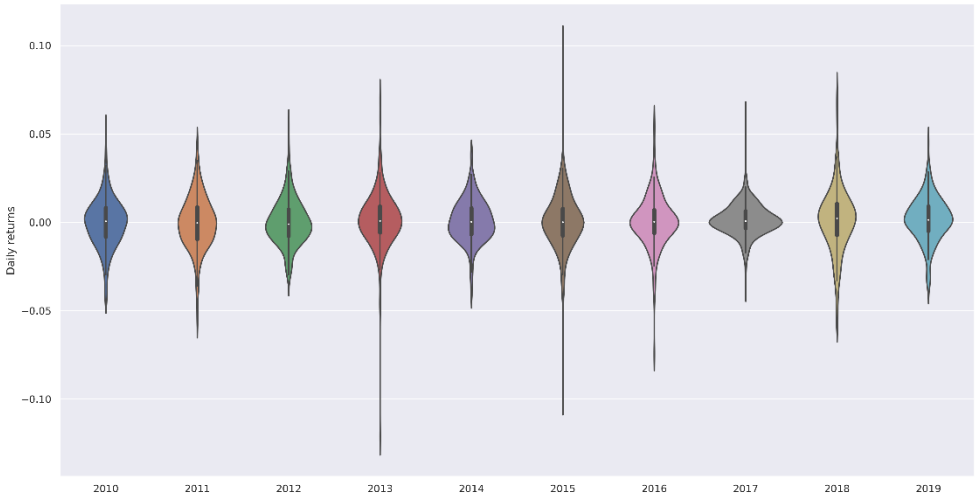

Figure 3.3: Violin plot of yearly distribution of daily returns on MSFT

There appears to be two major events in the distribution, the first one somewhere in 2013 and the second somewhere in 2015. A quick search indicates that Microsoft reported very poor earnings in 2013 with investors' confidence all time low (Dillet, 2013), eventually leading to ex CEO Steve Ballmer announcing resignation (Worstall, 2013). Whereas in 2015, Microsoft released windows 10 (Frank, 2015) and invested in various projects under CEO Satya Nadella regime. This shows how closely returns on a stock incorporate market sentiments.

A look at the histogram plot of the daily returns indicates thin tail (leptokurtosis) and departure from normal distribution.

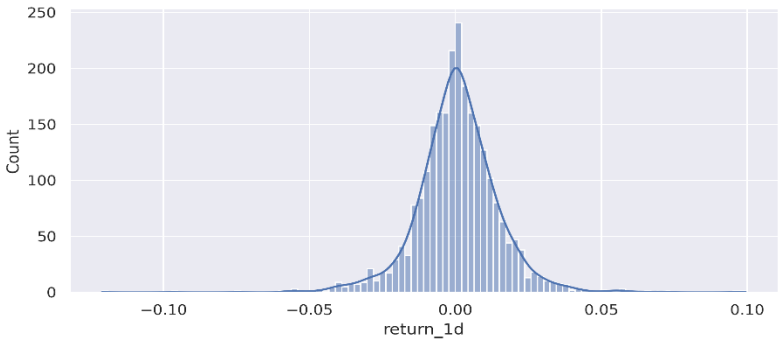

Figure 3.4: Thin tailed nature of daily returns on MSFT

The quantile-quantile plot confirms this non-normality.

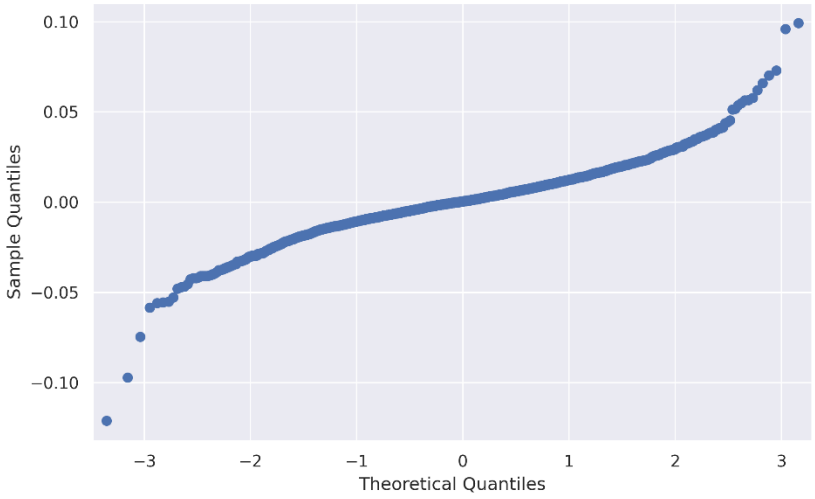

Figure3.5: QQ plot of daily returns fit to Normal distribution on MSFT

While this hypothesis could be more robust by performing a chi-square test, the exploratory graphs are enough for this project's purpose as the problem here is classification instead regression where one would explain the expected returns in future. The goal of the project is to simply generate the trading signals and compare the returns of using the model-based strategy to benchmark. This brings in the need to explain what might explain the future returns.

The following pair plot shows correlation between past weekly (5d), monthly (21d) and quarterly (63d) returns with forward one day return. Trading only takes place during working days hence lags for computing past returns are chosen to be 5,21 and 63 days. In total, there are roughly 252 trading days in a calendar year. As can be seen, there appears no correlation between past returns. The use of shifted lag returns while modelling the strategy might not capture any significant patterns and add noise to signal as confirmed later empirically.

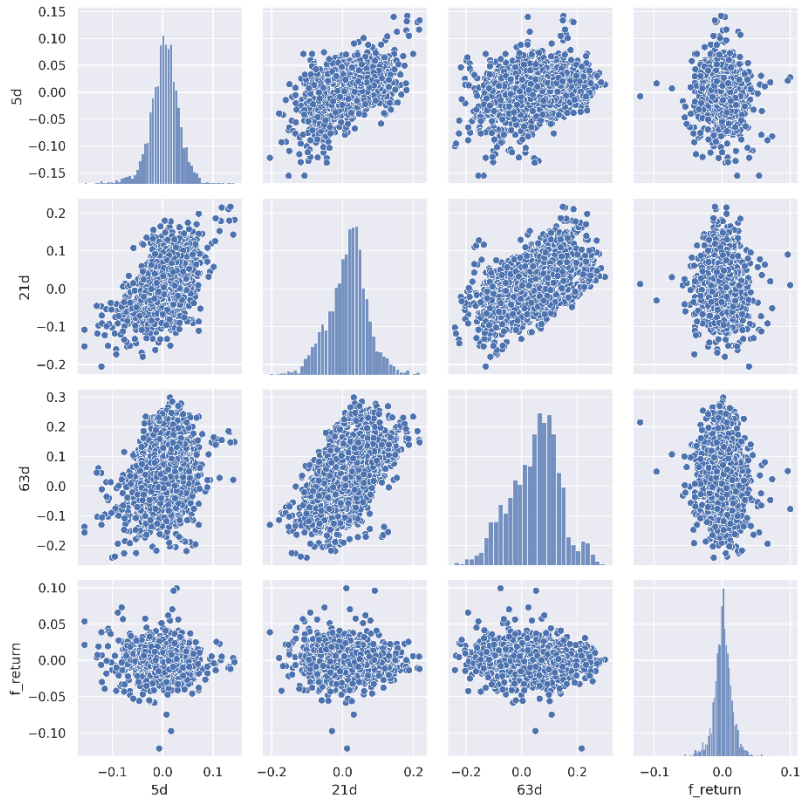

Figure 3.6: Correlation between future and lagged returns on MSFT

It can also be noticed that there appears to be some correlation between lagged quaterly returns and lagged monthly returns. However, it was confirmed that there is no correlation between previous quarter returns to forward monthly returns.

Indeed, the autocorrelation graph of returns confirms that there is no serial correlation among lagged returns.

The analysis so far necessitates the requirement of engineering features that might explain the returns. The first obvious choice is to look at how OHLCV data correlates with forward returns. As can be seen in the following heatmap, the open, high, low and close price are highly correlated making them redundant whilst there is no correlation between closing price but there is minor negative correlation between stock volume and forward returns.

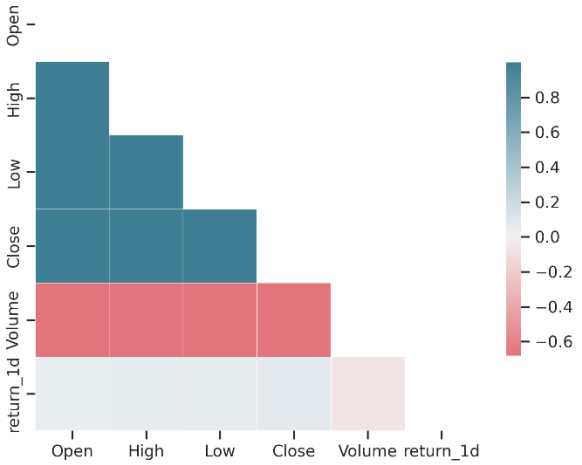

Figure 3.7: Correlation among OHLCV data and forward one day returns on MSFT

This indicates that stock returns are highly unpredictable, and any serious modelling strategy requires good feature engineering which will be examined in the following sections of this thesis.

Lastly, it warrants the need to determine how returns on a stock portfolio might correlate with returns on the market index since that may influence test metric for comparing returns on our strategy and the index. The following scatter plot shows that there is a positive correlation between market returns and portfolio returns.

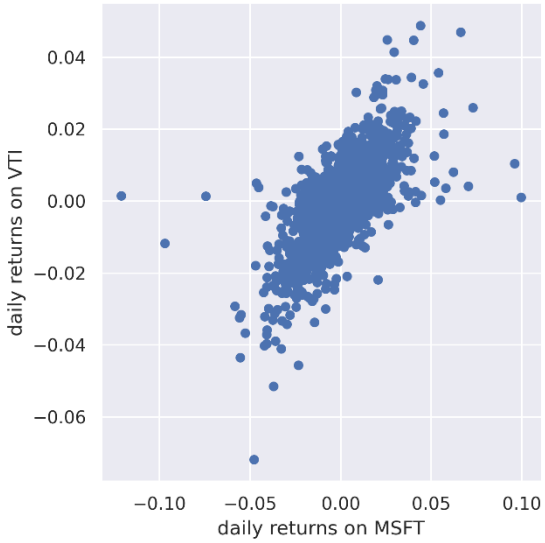

Figure 3.8: Correlation plot of daily returns on MSFT vs VTI

The exploratory analysis concludes the following:

- The data is clean and free of anomalies with need of some features to be scaled
- Returns on stock exhibits non-normality and no correlation with past returns
- Better features are required to predict forward returns
- Returns in portfolio used here are correlated with returns on market index since the latter includes the former
- The t-test could be used to establish the significance of mean excess returns from actively trading current portfolio with the mean excess returns of passive strategy.

3.4 Data Preparation

To avoid data snooping and data leakage, the OHLCV data was already split into a training set and test set prior to feature engineering because some of the technical indicator's functions consider the entire dataset to generate output values.

To expand a little more on the way prediction problem has been setup here, consider the OHLCV data obtained from a finance website such as finance.yahoo.com

Given OHLCV vector of a day i , $x^{(i)} = (open, high, close, low, volume)$, the corresponding label $y^{(i)}$ is the position of market on $(i+1)^{th}$ day. The machine learning algorithm takes $x^{(i)}$ as input and output predicted future position $\widehat{y^{(i)}}$. These predictions can then be compared to actual labels to evaluate performance. Of course, any machine learning algorithm would benefit from rich set of features in addition basic OHLCV pricing which is the part of next section.

3.5 Feature Engineering

Engineering features that could generate high quality trading signals is an active area of research.

The trading signals generated by any algorithmic trading strategy can be used to buy or sell assets with the purpose of gaining higher returns on a selected portfolio compared to a benchmark such as the market index. There are various models to price assets in order to generate alpha.

According to Fama-French three factor model (Fama and French, 1993), the returns on an asset can be explained as follows:

$$r = R_f + \beta(R_m - R_f) + b_s \cdot SMB + b_v \cdot HML + \alpha$$

Here r is the portfolio's expected rate of return, R_f is the risk-free rate of return and R_m is the return of the market Index. SMB (Stands for small minus big; market capitalization) measure past excess returns of small caps over big caps. And HML (stands for High minus low; High book-to-market ratio) measures past excess returns of value stocks compared to growth stocks. Estimating coefficients in the above equation can give an estimate of expected return on a

portfolio. While these three factors have already been statistically shown to explain returns, there is growing research in finance literature to identify new unknown factors (hidden in alpha) that can explain expected returns. The trading signals that try to produce excess returns are alternatively called alpha factors. This is also one of the grey areas of finance where financial institutions with profitable strategies have kept their factors secret.

However, from this project's scope, only commonly used alpha factors will be designed to be used in the models.

The design, evaluation and combination of alpha factors are important steps during design phase of algorithmic trading strategy workflow

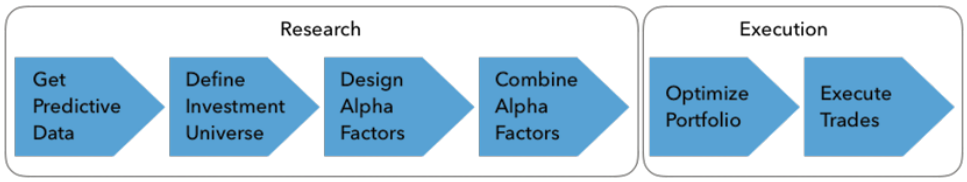

Figure 3.9: Alpha factor research and execution process (Jansen, 2020)

There are various ways to design alpha factors using transformations of market, fundamental and alternative data. Due to data availability constraints, this project only implements a subset of technical indicators as used by technical analysts to trade.

On broader level technical indicators could be categorized as follows:

a) Trend indicators

Trend indicators are one of the most basic types of indicators. These indicators measure the direction and strength of a trend based upon some transformations of price averages to establish baseline. The trend, either bullish or bearish, can then be judged depending upon if the short-term rolling average returns are higher or lower than long-term rolling average returns respectively. Simple moving average of returns is one of the examples of such indicators. Other more complex types of averages include exponential moving averages, kalman filter (a type of weighted average). The indicators are based upon time series analysis of OHLCV price data.

b) Momentum indicators

Momentum based factor strategies are among the most popular and often used in financial markets. Early works such as (Jegadeesh and Titman, 1993) provide empirical evidence for higher returns in US equity markets due to momentum. Value and momentum have been empirically shown repeatedly (Asness, Moskowitz and Pedersen, 2013). The logic behind using this factor is that asset prices exhibit a trend which is contrary to what the efficient markets hypothesis says. As confirmed in exploratory analysis, return prices on a stock exhibit no serial correlation.

However, since momentum is often used by technical analysts in practice, to that end, this project includes momentum-based factors in order to really see if it works. Common momentum indicators include relative strength index (RSI), moving average convergence/divergence (MACD), Directional Index, average Directional movement index and other oscillator-based signals.

c) Volume indicators

Volume indicators measure the strength of a trend or establish confidence around trading direction based on some form of smoothing of trading volume which is very volatile. The intensity of trends, bullish or bearish correlated with market's expectations of future price direction, seems to be correlated with increase or decrease in volume. Indeed, the increase in trading volume can lead to a large movement in price. Common volume indicators include Chaikin oscillator, on-balance volume, Volume rate of change.

d) Volatility indicators

These types of indicators are based on standard deviation of price. Essentially, the purpose of volatility indicators is to measure the rate of price movement based upon change in highest and lowest historical prices. These provide insights into the amount of buying and selling taking place in each market which could generate signals about where the market might change direction. The following figure demonstrates how S&P500 Index (SPX), a market-capitalization weighted index of 500 leading publicly traded companies in the US, fluctuates based on implied volatility of the market as measured by Cboe's volatility index (VIX), a real-time index that represents the market's expectations for the relative strength of near-term prices changes of SPX. This inter-relationship between volatility and stock prices could be used as a signal to predict market direction. Common volatility indicators include average true range (ATR), normalized ATR, Bollinger bands (BB)

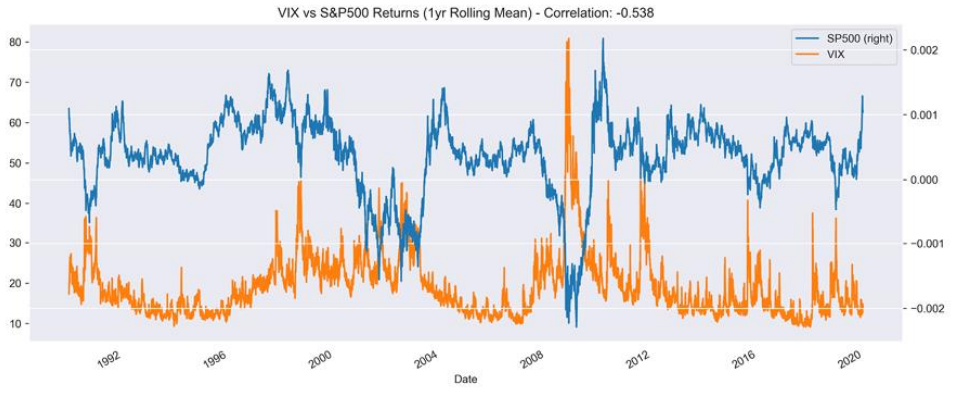

Figure 3.10: Comparison of VIX vs S&P 500 returns

Ideally, a machine learning based strategy would benefit from diverse uncorrelated alpha factors not just from technical analysis but also value indicators from fundamental pricing data, asset quality indicators from balance sheet and income statements of companies and factors from alternative data. Mining alpha factors automatically using machine learning and deep learning-based strategies is an active area of research, (Kakashadze, 2016) reports more than 100 such alpha factors.

Some of the basic features incorporated in machine learning models can be observed from following table:

Indicator	**calculation**	**Definition**
SMA	$SMA(N_t) = \frac{1}{N}\sum_{i=1}^{N} P_{t-N+i}$	Simple moving average simply calculates the roling average of Price $P_t$over specified window N. Standard lags used for trading are weekly (5 days), monthly (21 days) and quarterly (63 days).
HT_TRENDLINE	$H(u)(t) = \frac{1}{\pi} p.v. \int_{-\infty}^{+\infty} \frac{u(\tau)}{t-\tau} d\tau$	HIilbert transform generates in phase and quadrature components of a de-trended real valued signal like a price series to analyze variations of the instantaneous phase and amplitude.
BBANDS	$Bandwidth = \frac{UpperBB - lowerBB}{middleBB}$ Where $UpperBB = SMA(P_t, N_t) + m \cdot \sigma[P, N_t]$ $lowerBB = SMA(P_t, N_t) - m \cdot \sigma[P_t, N_t]$ $middleBB = SMA(P_t, N_t)$ $P_t = \frac{(High + low + Close)}{3}$ $m =$ No. of standard deviations $\sigma[P_t, N_t] =$ rolling standard dev. of P_t	A Bollinger band indicator consists of a middle band and two outer bands. The middle band is a simple moving average of at specific lag, typically 21. The outer bands are usually set 2 standard deviations. Falling Bandwidth reflects decreasing volatility and rising bandwidth reflects increasing volatility
RSI	$RSI_t = 100 - \frac{100}{1 + \frac{up_t}{down_t}}$	Relative Strength Index is a momentum oscillator that measures

	Where, $up_t = P_t^H - P_{t-T}^H$ $Down_t = P_{t-T}^L - P_t^L$	the speed and change of price movements. When RSI is greater than 70, the security is overbought while if it is less than 30, the security is oversold.
MACD	$MACD = EMA(N)_t - EMA(M)_t$ Where EMA(N) is the exponential moving average of price over lag N and M where N < M.	Moving average convergence and divergence is a trend-following momentum indicator that shows the relationship between to moving averages of a security's price. MACD offers the best of both worlds: trend following and momentum. It is not particularly useful for identifying overbought and oversold levels.
PLUS/MINUS DM	$+DM_t = \{^{up_t \ if \ up_t > down_t \ \& \ up_t > 0}_{0 \ otherwise}$ $-DM_t = \{^{Down_t \ if \ Down_t > Up_t \ and \ Down_t < 0}_{0 \ otherwise}$	Positive and negative directional movement form the backbone of the directional movement system. Combining these with the average directional index, one can determine both the direction and strength of a trend.
ADX	$ADX = 100 \cdot SMA(N)_t \left\|\frac{+DI_t - (-DI_t)}{+DI_t + (-DI_t)}\right\|$	Average directional movement index combines two other indicators, namely the positive and negative directional indicators to measure strength of the trend in signal. Its values range from 0 to 100. A signal with ADX value less than 25 is said to have weak trend and values above 50 indicate strong to extremely strong trend.

PPO	$PPO = \frac{EMA(12)_t - EMA(26)_t}{EMA(26)_t} \cdot 100$	The percentage Price oscillator is a momentum oscillator that measures the difference between two moving averages as a percentage of the larger moving average. It is like MACD- Histogram but it calculates percentages as opposed to absolute values.
STOCH	$K^{fast}(T_K) = \frac{P_t - P_{T_K}^L}{P_{T_K}^H - P_{T_K}^L} \cdot 100$	A stochastic oscillator is a momentum indicator comparing a particular closing price of a security to a range of its prices over a certain period. Stochastic oscillators are based on the idea that closing prices should confirm the trend. Bullish and Bearish divergences in the Stochastic Oscillator can be used to foreshadow momentum reversals.
MFI	$MFI = 100 - \frac{100}{1 + Money\ flow\ ratio}$	The Money Flow Index (MFI) incorporates price and volume information to identify overbought or oversold conditions. The indicator is typically calculated using 14 periods of data. An MFI reading above 80 is considered overbought and an MFI reading below 20 is considered oversold.
AD	$AD_t = AD_{t-1} + MFV_t$ Where $MFV_t = MFI_t \cdot V_t$ V is the period's volume	The Chaikin advance/decline (AD) line is a volume-based indicator which measures the cumulative cashflow into and out of an asset. The assumption here is that

		amount of buying or selling pressure could be estimated by the status of closing price relative to the high and low price for that period.
ATR	$TRANGE_t = \max\ (\lvert P_t^{high} - P_t^{low} \rvert, \lvert P_t^{high} - P_t^{close} \rvert, \lvert P_t^{low} - P_t^{close} \rvert)$ Then $ATR = SMA(TRANGE_t, t)$	The average true range indicators are a measure of volatility of the market. Originally introduced by Wilder in 1978, it has been used as a component of several other indicators since then. It tries to anticipate the changes in trend such that higher value implies higher probability of a trend change; the lower the indicator's value, the weaker the current trend.
Return_{t}d	$r_t = \frac{close - close_t}{close_t} \cdot 100$	Past t day returns are calculated using close price of today differenced with close price of day at lag t.

Table 3.3: A selection of technical indicators

To build better machine learning based trading model, it is desirable to engineer diverse set of features with little to no mutual correlation. This warrants the need for exploratory and correlation analysis on features created so far. Since the distribution of many of these features in the training set exhibits non-normality, spearman's correlation was the preferred choice to examine correlation over Pearson's correlation.

Spearman's correlation is a statistical measure of the strength of a monotonic relationship between paired data. The spearman correlation coefficient is defined as the pearson correlation coefficient between rank variables.

Given raw scores X_i, Y_i and their ranks $R(X_i)$, $R(Y_i)$, the spearman's correlation coefficient is calculated as:

$$r_s = \rho_{R(X),R(Y)} = \frac{cov(R(X), R(y))}{\sigma_{R(X)} \cdot \sigma_{R(Y)}},$$

Where,

ρ is Pearson's correlation coefficient applied to rank variables

$cov(R(X), R(Y))$ is the covariance of the rank variables

$\sigma_{R(X)}$, $\sigma_{R(Y)}$ are the standard deviations of the rank variables.

The following figure shows the spearman's correlation heatmap of the features implemented as discussed above.

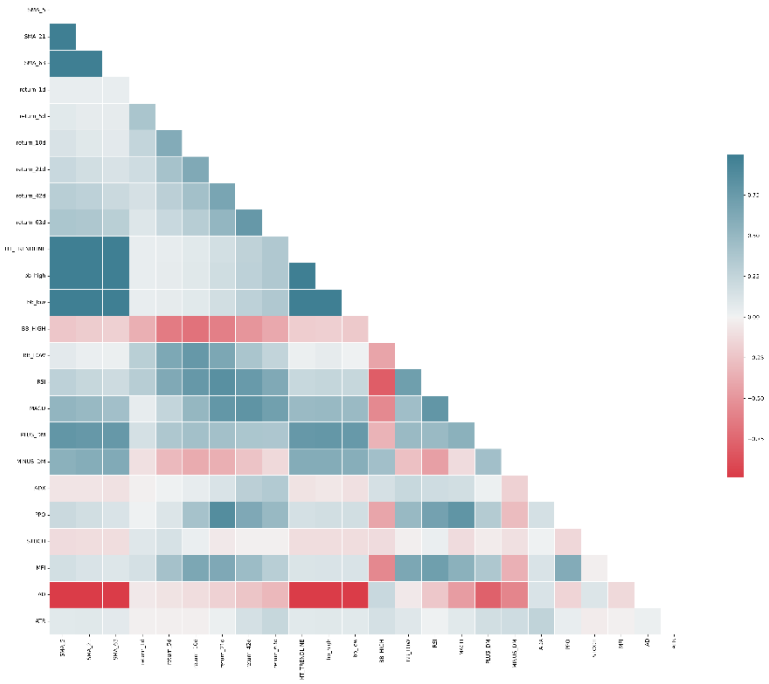

Figure 3.11: Spearman's Correlation heatmap of technical indicators on MSFT

An alternative approach is to use unsupervised learning algorithms like hierarchical agglomerative clustering (Müllner, 2013). The purpose of clustering here is to examine the relationship between alpha factors in order to choose factors that can keep the signal to noise ratio high for trading signals.

The algorithm begins with a forest of clusters that have yet to be used in the hierarchy being formed. When two clusters s and t from this forest are combined into a single cluster u, s andt t are removed from the forest and u is added to the forest. When only one cluster remains in the forest, the algorithm stops, and this cluster becomes the root.

By default, Seaborn uses weighted linkage method to calculate distance matrix, that is,

$$d(u,v) = \frac{\big(dist(s,v) + dist(t,v)\big)}{2}$$

Where cluster u was formed with cluster s and t, and v is a remaining cluster in the forest.

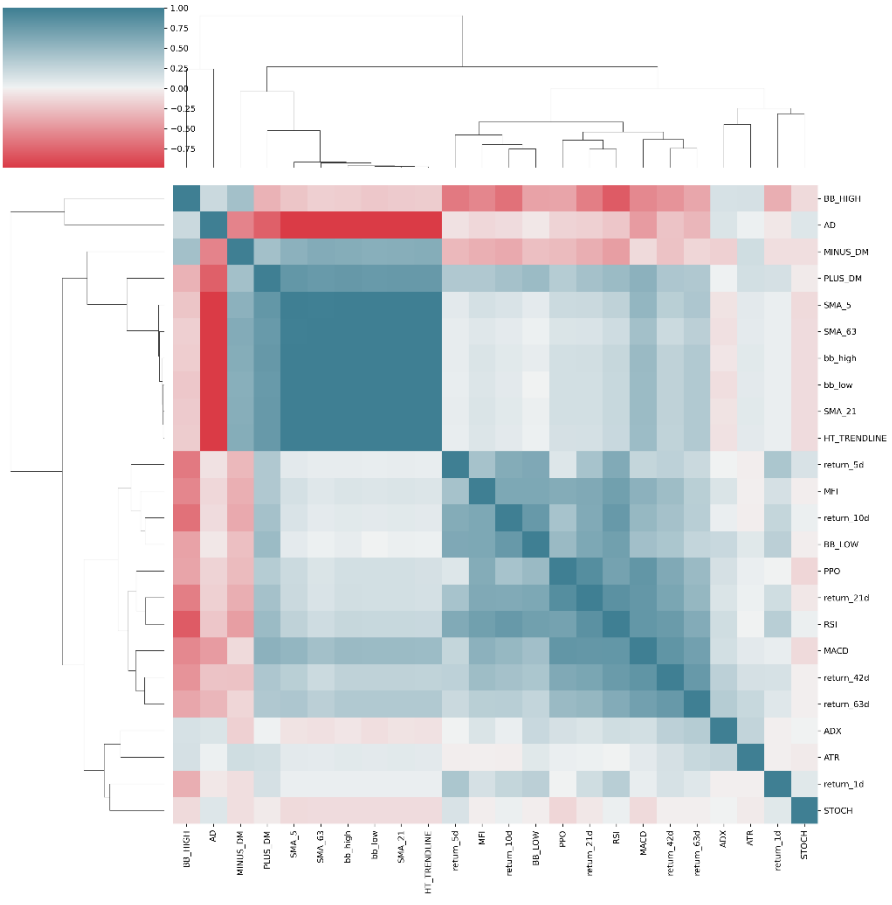

Figure 3.12: Cluster map of engineered technical features for MSFT

The simple moving average over weekly, monthly and quarterly lags are clustered together along with Hilbert transform trendline. This is not surprising since they convey the same information over different periods of time. The relationship of these factors with the target variable of forward one day return can be seen in the following correlations bar plot. Notice that the strength of correlation of these factors with the target variable is very weak compared to the correlation strength among themselves. This highlights the major challenge when it comes to designing better predictive models for financial markets movement prediction. The stochasticity of these features adds even error to predictions going forward in future.

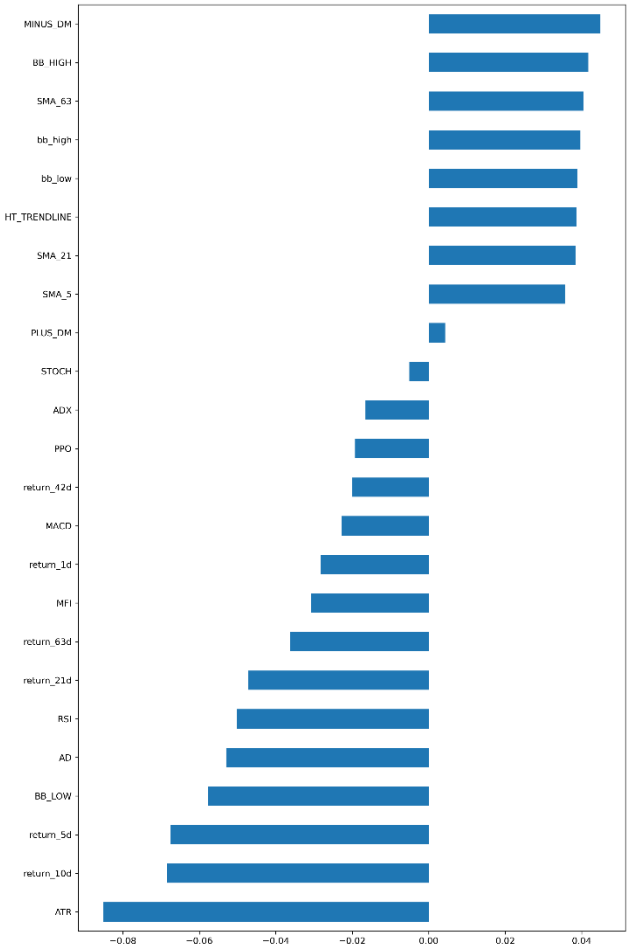

Figure 3.13: Correlation of technical indicators with one day forward returns on MSFT

3.6 Training and Hyperparameter tuning

After completion of exploratory analysis and feature engineering, the next step in the experimentation is to train different machine learning models in order to generate trading signals.

The training data is time-series indexed ranging from year 2010 to end of 2015 whereas the test data ranges from beginning of 2016 to end of 2018 right before covid-19 to avoid noise and black-swan events, this latter part will be discussed in more details in results and discussion sections.

The models were evaluated and their hyperparameters tuned via grid search method using cross-validation strategy with accuracy as the score metric.

For cross-validation, the splits were created considering time series nature (Bergmeir, Hyndman and Koo, 2018) of the index in context. For splitting, scikit-learn's TimeSeriesSplit method was used where the size of each split was set to 10 and train set size to 252 which is the number of trading days in a year. In other words, for cross validating the model performance, in each iteration, the algorithm trains the over past 252 days and performance is evaluated on the next 10 days with a gap of one day between train set and test size to safeguard from overlap.

This could be visualized as follows, the blue data points represent training set and red data points represent out-of-sample data points.

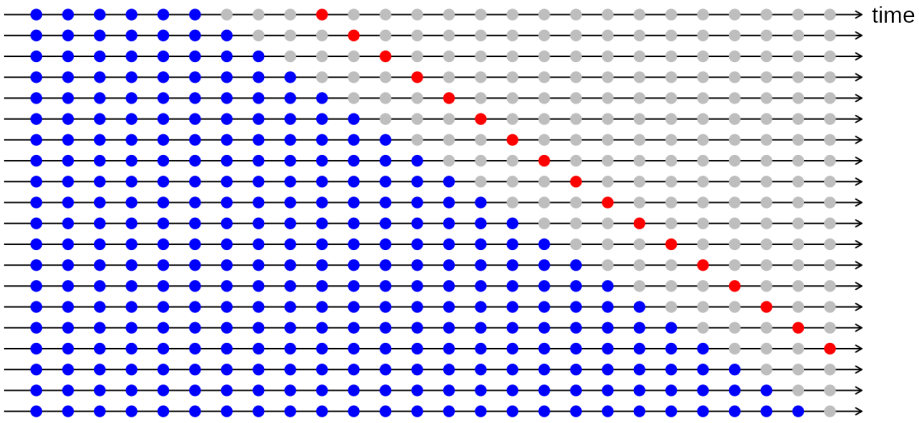

Figure 3.14: Time series cross validation with gaps (Hyndman and Athanasopoulos, 2021)

Once the models have been trained, they are compared to one another based upon the returns generated on test set using their prediction as trading signals, without considering real trading costs.

3. 7 Backtesting and portfolio evaluation

The next step in machine learning for trading workflow is to backtest the selected model for deployment in trading. Backtesting is an evaluation method (Chen, 2021) for assessing viability of a trading strategy by quantifying its performance on historical data. If backtesting passes desired trading criteria, then stakeholders can decide with confidence to deploy it in a downstream trading system.

From this project's perspective, there are no trading criteria set because the purpose here is to select whichever model is the best, evaluate the returns generated by that model and establish the significance of excess returns against those of a random walk model under the assumption that random walk hypothesis in stock prices is true.

Where this study differs from the ones listed in review is that to evaluate performance of machine learning models on a portfolio, event-driven backtesting adapted from (Jansen, 2020) is used over vectorized backtesting with the help of algorithms in backtrader library.

The difference between vectorized backtesting and event-driven backtesting is that the former simply evaluates trading signals from model with actual signals on past data whereas the latter adjusts for trading costs as one would trade in real world trading scenario.

Once the transactions and positions had been generated using backtrader, this data could be ingested into Pyfolio with the benchmark for portfolio evaluation.

3. 8 Hypothesis testing

Once cost adjusted returns from trading signals generated by backtested machine learning strategy have been obtained, these are compared with returns from passive strategy.

Specifically, the bootstrapped sample distribution of mean excess returns on the best strategy test is compared to the sample distribution of mean excess returns on the dummy strategy which produces trading signals randomly. The level of significance set to reject null hypothesis is 0.05. To compare sample means from null model and alternative model's population, Welch's t-test (Welch, 1951) was used to see which is more robust to populations with unequal variances under normality. The sample of means have been generated with the help of bootstrapping.

The bootstrap procedure could be described as follows. Assuming null hypothesis is true, a machine learning model is no better than a random model. Given the predicted returns from dummy classifier which randomly predicts long or short signals, this returns series is differenced to the actual series of returns. From this calculated excess returns series, we construct bootstrap samples by up sampling the series where sample size is set to 30. The Mean of this sample is noted, and the procedure is repeated over 10,000 times. The 95% percentile region on this distribution of mean excess returns is set as a critical region. If the mean excess returns from any

machine learning model are found to be in this critical region, then the returns generated are statistically significant and the alternative of random walk-in stock prices is true.

Mathematically, we want to compare population mean μ_l of excess returns from machine learning strategy and population mean μ_r of random walk strategy.

Where $H_0 : \mu_l < \mu_r$ is null hypothesis

And $H_a : \mu_l \geq \mu_r$ is alternative hypothesis

These means are estimated using bootstrap sampling as described above and p-value can be calculated using Welch's t-test since it is more robust to population with unequal variances.

3. 9 Interpretation

Finally, the results can be interpreted, and future implications of this study can be discussed.

The aim here are three-fold:

1. Interpretation of machine learning based active trading strategy with passive buy-n-hold strategy on same portfolio.

2. Interpretation of machine learning based active trading strategy with passive buy-n-hold strategy on same portfolio.

3. Interpretation of why ML-based strategies behave the way they do.

Results and Discussion

4.1 Outcome

On the test set, the Support Vector Classifier (SVC) achieved the highest cumulative returns over any other classifier as can be seen from the figure that follows. However, the returns are almost the same across all the classifiers including the dummy classifier which produces buy or sell signal randomly. The statistical significance of these returns is discussed next.

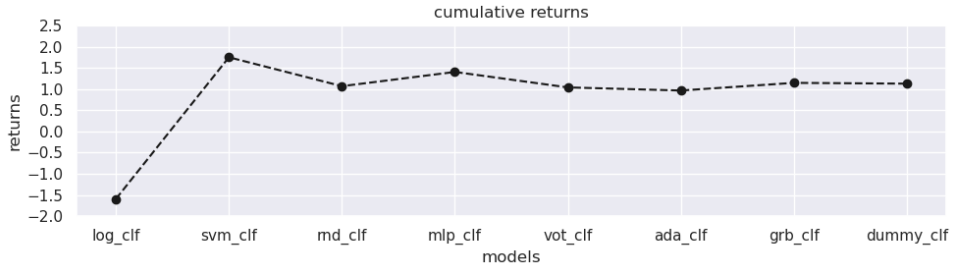

Figure 4.1: Cumulative returns comparison among ML models on portfolio in Table 3.1

Case 1: which is better: Actively trading with ML-based trading strategy OR passive investing on a portfolio?

For this case following hypothesis test was carried out using bootstrapping as described in previous section:

H_0: mean excess returns on SVC < mean excess returns on Dummy classifier

H_a: mean excess returns on SVC >= mean excess returns on Dummy classifier

The excess return here is the difference between returns on classifier and cumulative returns as one would gain in passive investing.

Here the p-value calculated using t-test on bootstrapped sample was 0.76, which means that current mean excess returns observed using SVC is likely to be seen 76% of the time if experiment is run repeatedly. Since this is not a rare event, we cannot reject null hypothesis. The excess returns generated by SVC are no better than a dummy classifier which generates buy or sell signals randomly.

Case 2: which is better: Actively trading with ML-based trading strategy OR passive investing on ETF index fund?

The hypothesis test was conducted the same way as in the previous case except here the difference was taken between cumulative returns on index fund VTI. The observed p-value here was 0.746. In this case as well, the null hypothesis could not have been rejected.

Case 3: which is better: passive investing in a stock portfolio OR passive investing on ETF index fund?

No hypothesis test was conducted for this case and the results are inconclusive because the data used for the purpose of this project wasn't large enough. However, a few observations were made during the experimentation. Consider the following figure which compares returns from passive and active strategies on MSFT with passive investing in market index funds VTI. The spread of returns on VTI is lower than passively investing on MSFT which in turn is lower than actively trading MSFT stock. This spread is due to lower volatility on index fund. This resonates with evidence mentioned (Malkiel, 2012) that investing in diverse portfolios such as market index fund reduces the risk exposure.

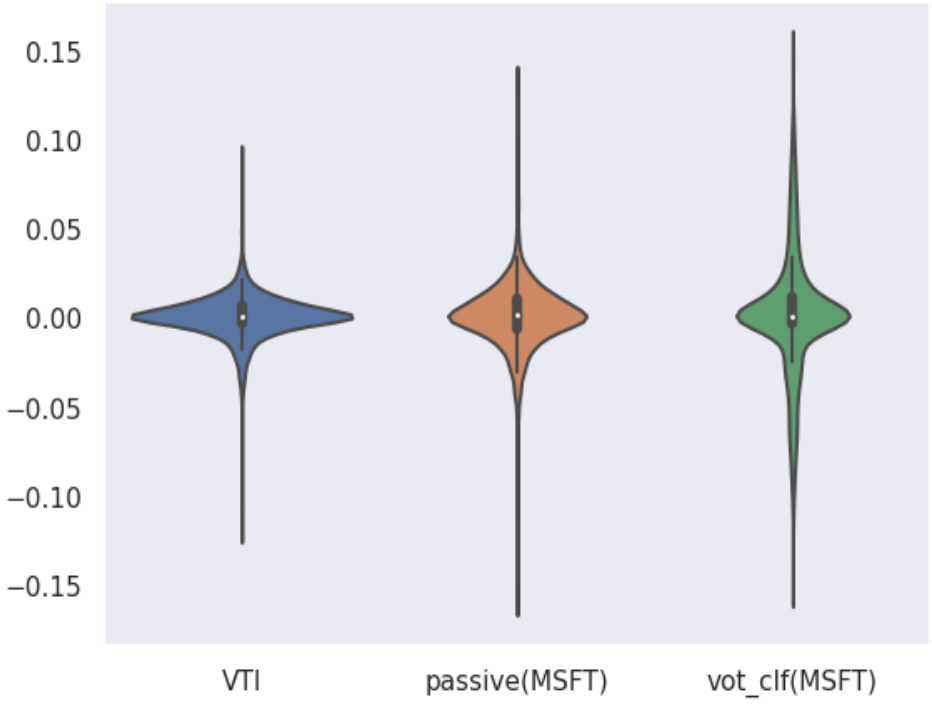

Figure 4.2 Distribution of returns on MSFT vs VTI

The overall returns on MSFT over the last 7 years are higher than the market index, this is due to incredible growth of Microsoft, which is only partially reflected in VTI. However, no company can keep growing forever at this rate as argued in (Graham and Zweig, 2003) and eventually the market is expected to catch up.

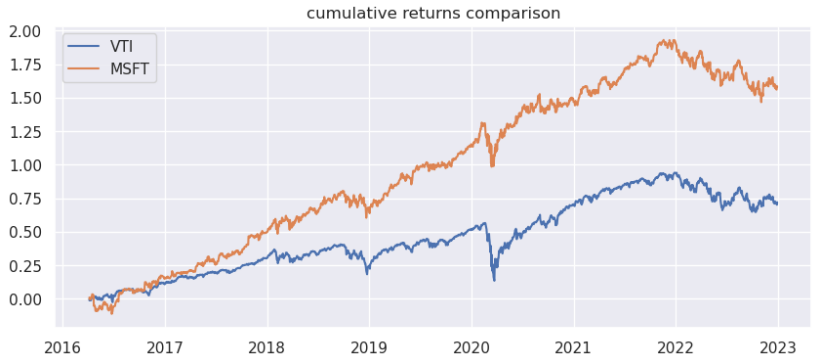

Figure 4.3 Cumulative return comparison MSFT vs VTI

4.2 Model Interpretation

Another observation that was made during experimentation is that accuracy of an algorithm doesn't always translate to higher returns. Consider the distribution of accuracy of models on test set and compare that with the cumulative returns shown in figure

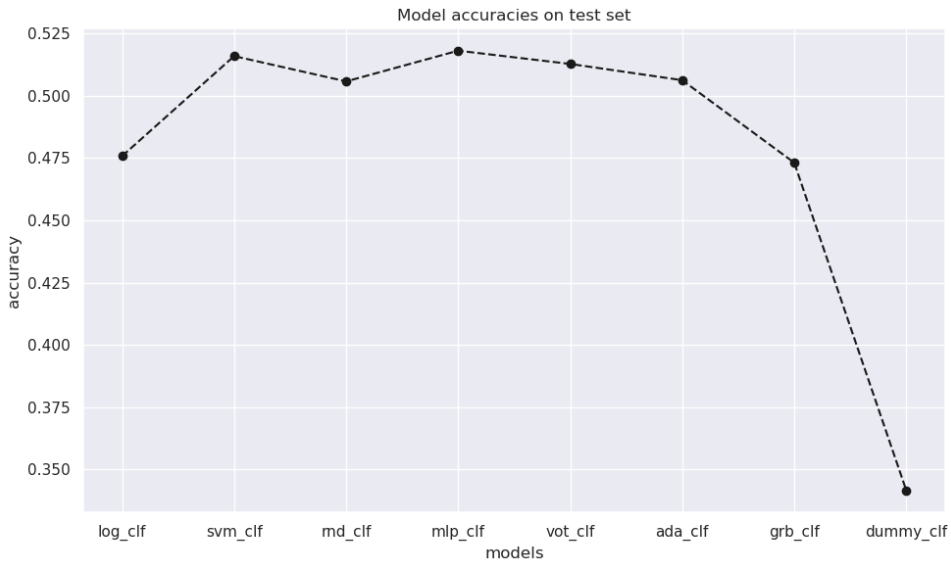

Figure 4.4: Comparison of models' accuracy on portfolio in table 3.1

Even though dummy classifier has very poor accuracy, the cumulative returns are almost the same as other classifiers. This is due to dispersion in data points, the data is noisy albeit represents the real-world scenario. The feature importance score assigned to different features by random forest model are given below:

Feature	**Score**
Chainkin A/D line	0.099
volume	0.082
ATR	0.077
Previous 2 months returns	0.066
Previous months return	0.062

Table 4.1 Feature Importance by Random Forest classification model

These simple trends, volume and volatility-based indicators seem to perform better than more complex momentum-based indicators.

The other ensemble models such Adaboost and gradient boost perform consistently poorly compared to other models. The generalization error among Adaboost and gradient boost models is higher compared to other models. None of the models, however, perform any better than a dummy classifier. It was also found that feeding more data to machine learning algorithms doesn't really improve their performance as the accuracy of model on the previous year's trading data is almost the same as the previous two years or three years trading data. The hyperparameters are suboptimal at best because as the searching algorithm doesn't converge well, some of the algorithms like logistic regression fail to converge at all.

4.3 Limitations

The limitation of this study includes selection bias. This selection bias comes from two uncorrelated factors. First the portfolio used in this project contains only six stocks and second, being the only type of data used in this study is end of the day pricing data. To limit selection bias in the first case, these stocks were chosen from diverse industries with different categories of market capitalization as one would invest in the real world. However, that's not to say there aren't stocks out there upon which machine learning algorithms can do reasonably well.

During one of the training rounds, it was also found that some of the learning algorithms like random forest and neural network can detect market inefficiencies due to black-swan events such as covid-19. The random forest and neural network models were able to take large number of short positions on the stocks and generate profit compared to as one would have got with passive investing. Given enough liquidity at one's disposal and considering someone's short term goals, profits can be generated. The caveat is that these profits do not seem to last long and eventually the buy-n-hold strategy catches up as the trading costs reduce the profits. This brings in the need for application of machine learning models in developing markets where inefficiency is expected to be higher than that in developed markets.

The second limitation is the choice of data. Institutional investors have been using data other than past prices to predict the market. The other type of data includes fundamental data, social media sentimental analytics and exchange order book real-time data. This study relies on Dow theory's argument that everything is already reflected in the price which may or may not be true as there's no concrete mathematical proof. How these types of data sources impact the performance of machine learning models need to be examined.

To address plaquing problem of overfitting in financial models, validation sets came in handy. However, even then these machine learning models seem to generalize poorly on test sets possibly due to their inherent complex nature. The dispersion in stock prices degrades the performance of these complex models as the algorithm tries to find the pattern when there is none.

Conclusion

With the advent of artificial intelligence, machine learning and deep learning models are transforming the way trading is being done in financial markets. This study explored the application of some of the machine learning models for trading against the claim of efficient market hypothesis which argues that stock prices follow random walk implying over the long run, across different markets, no model can consistently predict the market movements correctly.

To test this claim of efficient market hypothesis, this study was designed to empirically determine whether this is true. Diverse machine learning models such as logistic regression, support vector machine, random forest, neural networks and ensembling methods such as Voting classifier, Adaptive boosting and Gradient boosting were trained and tested against the Dummy classifier. The returns from these predictive models were found to be statistically insignificant compared to the Dummy classifier model. Moreover, the trading costs of actively trading diminish profits over the long horizon as found during backtesting.

The limitation of this study included selection bias when it comes to choosing portfolio of stocks and choice of data being used. This was addressed by considering the diverse set of stocks in the portfolio as one would invest in the real-world scenario.

The outlook for future research is that rather than predicting the market movements on a given set of days, the machine learning models should be used to detect anomalies instead and possibly gain advantage of these black-swan events like covid-19.

The implication of this study for investors and technical analysts is that trying to predict stock market based on pricing data is a waste of resources as none of sophisticated machine learning based trading models seem to outperform random walk models. Most investors with low-risk appetite and long-term goals would benefit from investing in diverse portfolios and holding it to maturity. The gains made by these models are nullified by the losses. To that end, “Buy-n-hold the power of passive indexing”.

Bibliography

Abe, M. and Nakagawa, K. (2020) 'Cross-Sectional Stock Price Prediction Using Deep Learning for Actual Investment Management', *Proceedings of the 2020 Asia Service Sciences and Software Engineering Conference.* Nagoya, Japan, Association for Computing Machinery. Available at: https://doi.org/10.1145/3399871.3399889 doi: 10.1145/3399871.3399889.

Andrew W. Lo (2004) 'The Adaptive Markets Hypothesis', *The Journal of Portfolio Management,* 30(5), pp. 15-29. Available at: https://doi.org/10.3905/jpm.2004.442611.

Asness, C.S., Moskowitz, T.J. and Pedersen, L.H. (2013) 'Value and Momentum Everywhere', *The Journal of Finance (New York),* 68(3), pp. 929-985. Available at: https://doi.org/10.1111/jofi.12021.

Ballings, M. *et al.* (2015) 'Evaluating multiple classifiers for stock price direction prediction', *Expert Systems with Applications,* 42(20), pp. 7046-7056. Available at: https://doi.org/10.1016/j.eswa.2015.05.013.

Bergmeir, C., Hyndman, R.J. and Koo, B. (2018) 'A note on the validity of cross-validation for evaluating autoregressive time series prediction', *Computational Statistics & Data Analysis,* 120, pp. 70-83. Available at: https://doi.org/10.1016/j.csda.2017.11.003.

Cao, L.J. and Tay, F.E.H. (2003) 'Support vector machine with adaptive parameters in financial time series forecasting', *IEEE Transactions on Neural Networks,* 14(6), pp. 1506-1518. Available at: https://doi.org/10.1109/TNN.2003.820556.

Chen, J. (2021) *Backtesting: Definition, How It Works, and Downsides.* Available at: https://www.investopedia.com/terms/b/backtesting.asp .

Cho, C. *et al.* (2019) 'Toward Stock Price Prediction Using Deep Learning', *Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing Companion.* Auckland, New Zealand, Association for Computing Machinery. Available at: https://doi.org/10.1145/3368235.3369367 doi: 10.1145/3368235.3369367.

Dillet, R. (2013) *Microsoft Experiences Its Biggest Drop Of The Century As Shares Fall 12 Percent.* Available at: https://techcrunch.com/2013/07/19/as-shares-fall-12-percent-microsoft-experiences-its-biggest-drop-since-2000/ .

Fama, E. (1965) 'Random Walks in Stock-Market Prices', *Financial Analysts Journal,* Available at: https://doi.org/10.2469/faj.v21.n5.55.

Fama, E.F. (1970) 'Efficient Capital Markets: A Review of Theory and Empirical Work', *The Journal of Finance,* 25(2), pp. 383-417. Available at: https://doi.org/10.2307/2325486.

Fama, E.F. (1991) 'Efficient Capital Markets: II', *The Journal of Finance,* 46(5), pp. 1575-1617. Available at: https://doi.org/10.2307/2328565.

Fama, E.F. and French, K.R. (1993) 'Common risk factors in the returns on stocks and bonds', *Journal of Financial Economics,* 33(1), pp. 3-56. Available at: https://doi.org/10.1016/0304-405X(93)90023-5.

Frank, B. (2015) *Looking back on 2015, Microsoft's biggest year ever.* Available at: https://www.infoworld.com/article/3018035/looking-back-on-2015-microsofts-biggest-year-ever.html .

Friedman, J.H. (2001) '1999 REITZ LECTURE GREEDY FUNCTION APPROXIMATION: A GRADIENT BOOSTING MACHINE'', .

Geron, A. (2019) *Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: concepts, tools, and techniques to build intelligent systems.* Third edn. Sebastopol, CA: O'Reilly.

Graham, B. and Zweig, J. (2003) *The intelligent investor: a book of practical counsel, revised edition.* Rev. edn. Pymble, N.S.W., Australia; New York, N.Y: HarperCollins.

Harris, C.R. *et al.* (2020) 'Array programming with NumPy', *Nature,* 585(7825), pp. 357-362. Available at: https://doi.org/10.1038/s41586-020-2649-2.

Hastie, T., Friedman, J.H. and Tibshirani, R.J. (2009) *The elements of statistical learning: data mining, inference, and prediction.* London; New York: Springer.

Hayes, A. (2022) *Dow Theory Explained: What It is and How it works.* Available at: https://www.investopedia.com/terms/d/dowtheory.asp .

Hsu, M. *et al.* (2016) 'Bridging the divide in financial market forecasting: machine learners vs. financial economists', *Expert Systems with Applications,* 61, pp. 215-234. Available at: https://doi.org/10.1016/j.eswa.2016.05.033.

Huang, C.Y. (2018) *Financial Trading as a Game: A Deep Reinforcement Learning Approach.*

Hunter, J.D. (2007) 'Matplotlib: A 2D graphics environment', *Computing in Science & Engineering,* 9(3), pp. 90-95.

Hyndman, R.J. & Athanasopoulos, G. (2021) *Forecasting: principles and practice, 3rd edition, OTexts: Melbourne, Australia.* Available at: https://otexts.com/fpp3/tscv.html .

Jansen, S. (2020) *Machine Learning for Algorithmic Trading: Predictive Models to Extract Signals from Market and Alternative Data for Systematic Trading Strategies with Python, 2nd Edition.* Birmingham: Packt Publishing, Limited.

Jegadeesh, N. and Titman, S. (1993) 'Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency', *The Journal of Finance,* 48(1), pp. 65-91. Available at: https://doi.org/10.2307/2328882.

Jung, J. and Shiller, R.J. (2005) 'SAMUELSON'S DICTUM AND THE STOCK MARKET', *Economic Inquiry,* 43(2), pp. 221-228. Available at: https://doi.org/10.1093/ei/cbi015.

Kakashadze, Z. (2016) '101 Formulaic Alphas', *Wilmott (London, England),* 2016(84), pp. 72-81. Available at: https://doi.org/10.1002/wilm.10525.

Kara, Y., Acar Boyacioglu, M. and Baykan, ÖK. (2011) 'Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange', *Expert Systems with Applications,* 38(5), pp. 5311-5319. Available at: https://doi.org/10.1016/j.eswa.2010.10.027.

Li, Y. (2022) 'Stock Quantitative Prediction Analysis Method Based on Deep Learning Transformer Self-Attention Mechanism', *Proceedings of the 7th International Conference on Cyber Security and Information Engineering.* Brisbane, QLD, Australia, Association for Computing Machinery. Available at: https://doi.org/10.1145/3558819.3565216 doi: 10.1145/3558819.3565216.

Lin, M. and Chen, C. (2018) 'Short-Term Prediction of Stock Market Price Based on GA Optimization LSTM Neurons', *Proceedings of the 2018 2nd International Conference on Deep Learning Technologies.* Chongqing, China, Association for Computing Machinery. Available at: https://doi.org/10.1145/3234804.3234818 doi: 10.1145/3234804.3234818.

Lo, A.W. and MacKinlay, A.C. (1988) 'Stock Market Prices do not Follow Random Walks: Evidence from a Simple Specification Test', *The Review of Financial Studies,* 1(1), pp. 41-66.

Ma, T. and Ng, A. (2007) 'CS229 Lecture notes', .

Malkiel, B.G. (2012) *The Elements of Investing: Easy Lessons for Every Investor.* 2. Aufl. edn. New York: Wiley.

Müllner, D. (2013) 'fastcluster : Fast Hierarchical, Agglomerative Clustering Routines for R and Python', *Journal of Statistical Software,* 53(9) Available at: https://doi.org/10.18637/jss.v053.i09.

Nakamura, T. and Small, M. (2007) 'Tests of the random walk hypothesis for financial data', *Physica A: Statistical Mechanics and its Applications,* 377(2), pp. 599-615. Available at: https://doi.org/10.1016/j.physa.2006.10.073.

Oliveira, N., Cortez, P. and Areal, N. (2013) 'Some Experiments on Modeling Stock Market Behavior Using Investor Sentiment Analysis and Posting Volume from Twitter', *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics.* Madrid, Spain, Association for Computing Machinery. Available at: https://doi.org/10.1145/2479787.2479811 doi: 10.1145/2479787.2479811.

Pedregosa, F. *et al.* (2011) 'Scikit-learn: Machine Learning in Python', *Journal of Machine Learning Research,* 12, pp. 2825-2830.

Perktold, J. *et al.* (2023) 'statsmodels/statsmodels: Release 0.14.0', .

Stockscreener,Nasdaq. Available at: https://www.nasdaq.com/market-activity/stocks/screener .

Tay, F.E.H. and Cao, L. (2001) 'Application of support vector machines in financial time series forecasting', *Omega,* 29(4), pp. 309-317. Available at: https://doi.org/10.1016/S0305-0483(01)00026-3.

team,T.p.d.(2023)'pandasdev/pandas:Pandas',Availableat: https://doi.org/10.5281/ZENODO.7857418.

Thenmozhi, M. (2014) 'Forecasting stock index returns using ARIMA-SVM, ARIMA-ANN, and ARIMA-random forest hybrid models', *International Journal of Banking, Accounting and Finance,* 5(3), pp. 284. Available at: https://doi.org/10.1504/IJBAAF.2014.064307.

Tin Kam Ho (1995) 'Random decision forests', - *Proceedings of 3rd International Conference on Document Analysis and Recognition.* doi: 10.1109/ICDAR.1995.598994.

Tukey, J.W. (1977) *Exploratory data analysis.* London (etc.); Reading, Mass: Addison-Wesley.

Virtanen, P. *et al.* (2020) 'SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python', *Nature Methods,* 17, pp. 261-272.

Waskom, M.L. (2021) 'seaborn: statistical data visualization', *Journal of Open Source Software,* 6(60), pp. 3021.

Welch, B.L. (1951) 'On the Comparison of Several Mean Values: An Alternative Approach', *Biometrika,* 38(3), pp. 330-336. Available at: https://doi.org/10.2307/2332579.

Worstall, T. (2013) *Links 23 Aug. Steve Ballmer Announces Resignation, Microsoft Stock Soars And No It Won't Be Bill Gates.* Available at: https://www.forbes.com/sites/timworstall/2013/08/23/links-23-aug-steve-ballmer-announces-resignation-microsoft-stock-soars/ .

List of figures:

List of tables: