Multiple Model-based Binary Classification for Forex Trading

By

LEUNG Cheuk Hei, Victor

Advised By

David Paul, ROSSITER

COMP 4971C - Independent Work

[2022-2023 Summer]

The Hong Kong University of Science and Technology

[Department of Computer Science and Engineering]

Date of submission

[August 12, 2023]

Table of Contents

1. Abstract

This document aims to juxtapose the efficacies of forex trading between fixed deposits and Multi-model-based Binary Classification. The project employs a rigorous trading strategy that leverages most machine-learning models to analyze daily trends. Swing and day trading strategies are implemented upon recognizing classical trends. The classification process intends to employ the entire spectrum of regressors and classifiers from sci-kit-learn. Both strategies intend to buy at a lower price and sell at a higher price. The outcomes are scrutinized to assess the viability of a buy-and-hold strategy. The study employs comprehensive 10-year data (2000/01/01 - 2010/01/01) for training. Trading will be conducted for 10 years (2010/01/02 - 2022/01/01). However, the average training and trading frames are reduced by half after dropping the NA data frame. The resulting trading behaviors, [Buy], [Sell], [Retain], [No Action], and [Buy & Sell] are constructed from the classification results. Assets, models, and metrics are integrated into arrays to assess their autonomous influences. The parameters of models remain untouched without additional tuning. The assessment criterion encompasses both classification metrics (accuracy, etc.) and trading metrics (Sharpe ratio, etc.), providing a holistic evaluation of the effectiveness of the trading strategy.

2. Objective

In the fast-paced and high-stakes world of trading, the ability to quickly and accurately process vast amounts of data is essential. Machine learning comprises unsupervised and supervised learning, data preprocessing, feature engineering, model selection, evaluation, and optimization techniques [1]. It has emerged as a critical tool for traders, providing advanced algorithms and models capable of analyzing data with incredible speed and precision. The integration of natural language processing (NLP) and chatbots powered by GPT-4 has transformed the way traders approach the market, enabling them to quickly parse and analyze news articles, social media feeds, and other sources of information to identify trends and patterns that can impact the market. For example, a chatbot powered by GPT-4 can be used to monitor social media feeds for mentions of a particular stock or company, analyzing sentiment and identifying potential risks or opportunities. By leveraging the latest advances in machine learning and NLP, traders can stay ahead of the curve

and make informed investment decisions that drive success.

3. Introduction

3.1. Asset Classes

3.1.1. Stock

It is commonly referred to as equities or shares, epitomize a form of ownership in a publicly traded company [2]. By purchasing a stock, an individual becomes a shareholder and can lay claim to a portion of the company's assets and earnings. Trading of stocks occurs on stock exchanges, where a convergence of buyers and sellers takes place. Market forces of supply and demand determine the price of a stock, and it is subject to fluctuation based on various factors such as the company's performance, prevailing economic conditions, and investor sentiment.

Top 10 Volume Stocks in 2010

1. Bank of America (BAC)
2. Citigroup (C)
3. Ford Motor Company (F)
4. General Electric (GE)
5. Cisco Systems (CSCO)
6. Pfizer Inc. (PFE)
7. Microsoft Corporation (MSFT)
8. Alcoa Inc. (AA)
9. AT&T Inc. (T)
10. Intel Corporation (INTC)

3.1.2. Commodity

It represents fundamental raw materials or primary products that are exchanged on commodity markets [3]. These products span a broad spectrum and include metals, such as gold and silver, energy sources, such as oil and natural gas, and agricultural products, such as wheat and corn. The pricing of commodities is established by the interplay between market forces of supply and demand, and they remain susceptible to fluctuation based on a wide array of factors such as meteorological conditions, political events, and global economic trends.

Top 10 Volume Commodities in 2010:

1. Crude Oil (CL)
2. Gold (GC)
3. Natural Gas (NG)
4. Corn (C)
5. Heating Oil (HO)
6. Brent Oil (CO)
7. Silver (SI)
8. Soybeans (S)
9. Copper (HG)
10. Wheat (W)

3.1.3. Forex

It is commonly referred to as foreign exchange or currency trading necessitates the astute buying and selling of diverse currencies to capitalize on profit derived from fluctuations in their exchange rates [4]. The Forex market functions as a platform for the union of buyers and sellers in the trading of currencies. The pricing of these currencies is subject to the multifarious interplay of market forces of supply and demand, which can be influenced by various factors including economic indicators, geopolitical events, and trends in the global market.

Top 10 Volume Forexes in 2010

1. US Dollar/Japanese Yen (USD/JPY)
2. Euro/US Dollar (EUR/USD)
3. British Pound/US Dollar (GBP/USD)
4. Australian Dollar/US Dollar (AUD/USD)
5. US Dollar/Swiss Franc (USD/CHF)
6. US Dollar/Canadian Dollar (USD/CAD)
7. Euro/Japanese Yen (EUR/JPY)
8. British Pound/Japanese Yen (GBP/JPY)
9. Euro/Swiss Franc (EUR/CHF)
10. New Zealand Dollar/US Dollar (NZD/USD)

3.2. Trading Terminologies

3.2.1. Bid-Ask Spread

Figure 1: Bid and Ask Price from MetaTrader 5

It is the difference between the highest price that a buyer is willing to pay for a security (the bid price) and the lowest price that a seller is willing to accept (the asking price) [5]. The bid-ask spread represents the transaction cost of buying and selling a security, and it can vary depending on factors such as market liquidity, trading volume, and volatility.

3.2.2. Limit Order

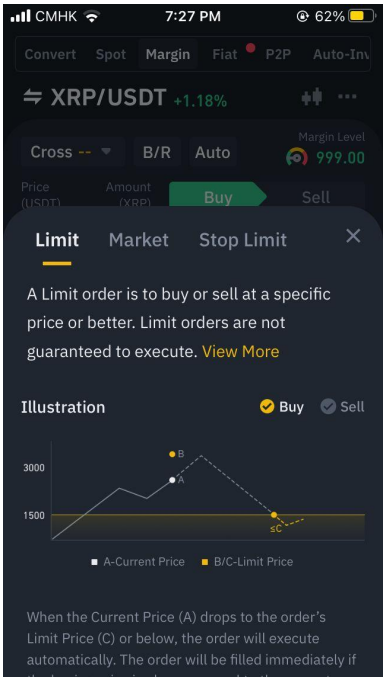

Figure 2: Principle of Limit Order from Binance

It is an order to buy or sell a security at a specific price or better [6]. A buy limit order is placed below the current market price, while a sell limit order is placed above the current market price. Limit orders are used to control the price at which a trade is executed and to avoid unfavorable price movements.

3.2.3. Stop Loss Order

It is an order to sell a security at a specific price or worse. It is used to limit potential losses on trade and to manage risk [7]. A sell-stop order is placed below the current market price, while a buy-stop order is placed above the current market price.

3.2.4. Margin

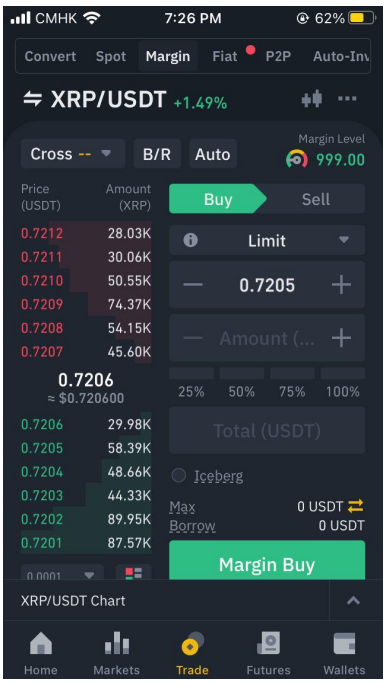

Figure 3: Margin on Binance

It denotes the guarantee that an investor must furnish to their broker or exchange as a safeguard against the credit risk that they may pose [8]. It allows traders to leverage their positions and potentially increase their profits, but it also increases their risk. Margin requirements can vary depending on the broker, the type of security being traded, and other factors.

3.2.5. Short Selling

It is the practice of selling a security that the trader does not own, intending to buy it back at a lower price in the future [9]. Short selling is used to profit from a decline in the price of a security. It can be risky, as losses can be unlimited if the price of the security rises instead of falling.

3.2.6. Liquidity

It refers to the ease with which a security can be bought or sold without significantly affecting its price [10]. Highly liquid securities can be traded quickly and with minimal impact on their price, while illiquid securities may be difficult to trade and can experience significant price movements as a result.

3.2.7. Volatility

It is a statistical metric that quantifies the extent of dispersion of returns for a particular security or market index [11]. High volatility can provide opportunities for traders to profit, but it also increases risk. Volatility can be influenced by a variety of factors, such as economic data releases, political events, and market sentiment.

3.2.8. Technical Analysis

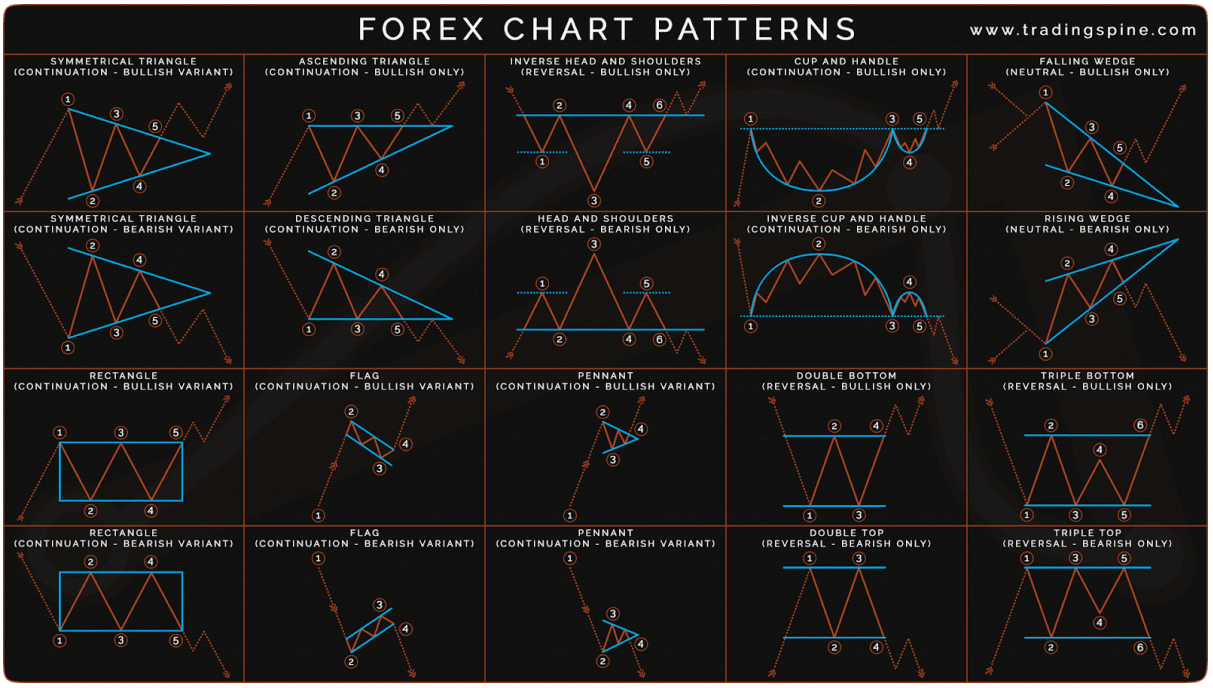

Figure 4: Chart Patterns from Trading Spine

It is a method of analyzing securities based on their price and volume movements [12]. It involves using charts and other tools to identify trends, patterns, and other signals that can help traders make trading decisions. Technical analysis can be used in conjunction with fundamental analysis, which involves analyzing a security's underlying financial and economic data.

3.3. Types of Machine Learning

3.3.1. Supervised Learning

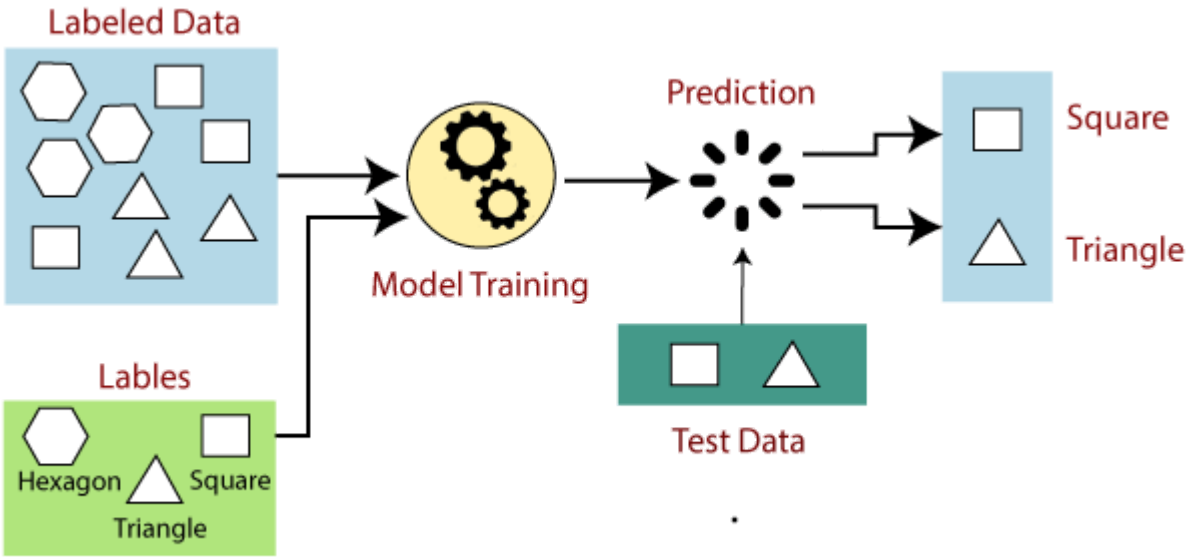

Figure 5: Supervised Learning from JavaTpoint

It is a type of machine learning where the algorithm is trained on a labeled dataset [13]. The input data is labeled with the correct output, and the algorithm learns to map the input to the correct output based on this labeled data.

Classification	Regression
The algorithm learns to classify input data into different categories based on the labeled data. For example, an algorithm might be trained to classify email messages as spam or not spam based on a labeled dataset of emails.	The algorithm learns to predict a continuous output value based on the labeled data. For example, an algorithm might be trained to predict the price of a house based on a labeled dataset of house prices and features.

Table 1: Types of Supervised Learning

3.3.2. Unsupervised Learning

It is a type of machine learning where the algorithm is trained on an unlabeled dataset [14]. The algorithm learns to find patterns and structure in the data on its own, without any explicit guidance.

Clustering	Anomaly Detection
The algorithm learns to group similar data points based on the underlying structure of the data. For example, an algorithm might be trained to cluster customers based on their purchasing habits.	The algorithm learns to identify data points that are significantly different from the rest of the data. For example, an algorithm might be trained to detect credit card fraud based on a dataset of credit card transactions.

Table 2: Types of Unsupervised Learning

3.3.3. Semi-Supervised Learning

It is a machine-learning paradigm that leverages a sparse set of labeled data and an abundance of unlabeled data to train a predictive model [15]. It involves training an algorithm with a combination of labeled and unlabeled data, to improve the accuracy of the algorithm's predictions.

Semi-Supervised Classification
The algorithm is trained on a small amount of labeled data and a large amount of unlabeled data. The labeled data provides some guidance on the classification task, while the unlabeled data helps the algorithm to learn the underlying structure in the data.

Table 3: Example of Semi-supervised Learning

3.3.4. Reinforcement Learning

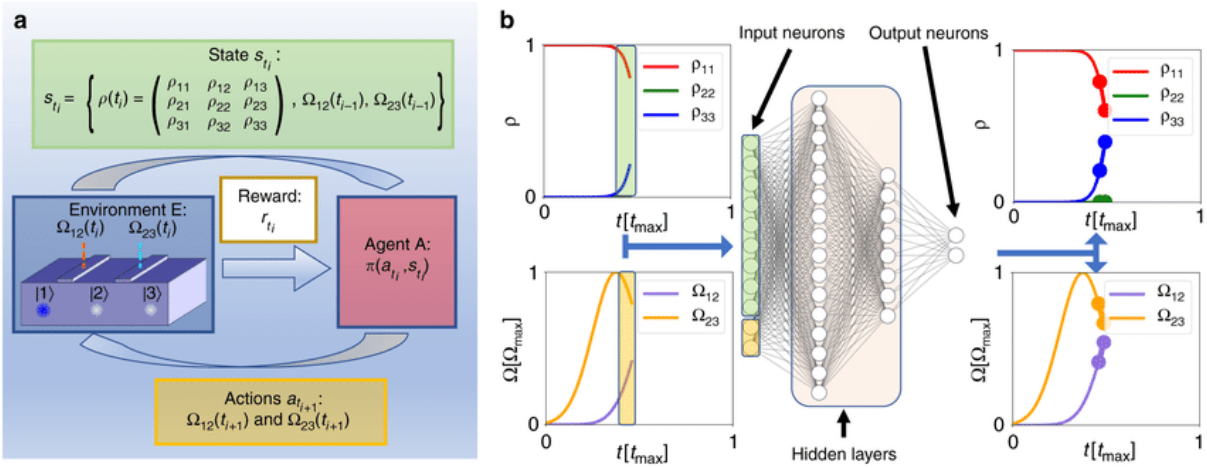

Figure 6: Deep Reinforcement Learning Architecture from Research Gate

It is a type of machine learning that involves training an agent to make decisions in a dynamic environment, based on a system of rewards and punishments [16].

Game Playing	Robotics
The algorithm learns to play a game by taking actions and receiving rewards or punishments based on the outcome of each action. The goal is to maximize the total reward throughout the game.	The algorithm learns to control a robot by taking actions and receiving feedback from the environment in the form of rewards or punishments. The goal is to perform a specific task, such as navigating a maze or manipulating objects.

Table 4: Types of Reinforcement Learning

3.3.5. Deep Learning

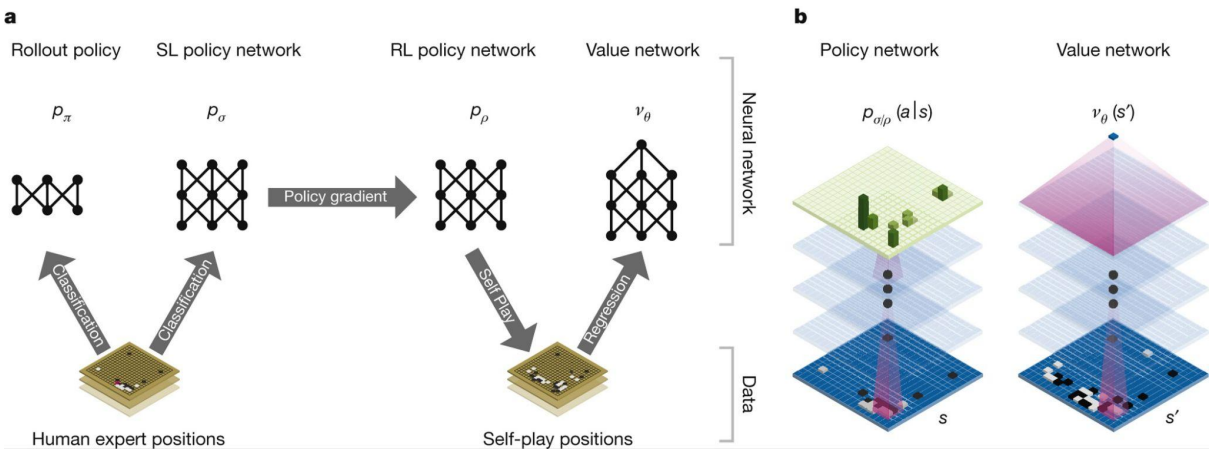

Figure 7: Deep Neural Networks and Tree Search of AlphaGo on Nature

It is a subset of machine learning that involves the use of neural networks with multiple layers [17]. These networks are designed to learn hierarchical representations of data, where each layer learns increasingly abstract features of the input data.

Image Recognition	Natural Language Processing
The network learns to identify objects in images based on a labeled dataset of images.	The network learns to process and understand natural language based on a labeled dataset of text data.

Table 5: Types of Deep Learning

4. Methodology

To facilitate machine trading for trading, the following library was employed.

Python Library
1. math 2. numpy 3. pandas 4. yfinance 5. warnings

6. statistics 7. matplotlib 8. IPython 9. scripy 10. google 11. sklearn

Table 6: Python Libraries Utilization

4.1. Data Collection

Historical prices of the top 10 assets in forexes were gathered from Yahoo Finance API with Python. The data were retrieved on a daily time frame basis, which compromises open, high, low, close, adjusted close, and volume. The data was gathered from January 1, 2000, to January 1, 2020.

Provider	Period	Dataset
Yahoo Finance	2000-01-01 - 2020-01-01	1. Open 2. High 3. Low 4. Close 5. Adjusted Close 6. Volume

Table 7: Context of Collected Data

4.2. Software Framework

Google Colab would be used to employ trading strategies. Python libraries - math, numpy, pandas, statistics, matplotlib, and IPython would be used for visualization and data analysis. Lastly, Classification will be conducted with the Scikit-learn library, as general machine learning models - calibration, cluster, ensemble, gaussian process, isotonic, and more are available.

Colab	Runtime	Hardware	GPU Type	Runtime

Version	Type	Accelerator		Shape
Pro	Python 3	TPU/GPU	v2/A100	High-RAM

Table 8: Setting of Google Colab Notebooks

4.3. Initial Account Setting

The initial capital in this study is HKD 10,000.

4.4. Ideation

Harnessing the collective power of all sci-kit-learn models for binary classification, this approach entails employing models that strive to fit the data through continuous regression, subsequently truncating the continuous predictions to yield discrete classifications. Intriguingly, linear regression, which constitutes the cornerstone of logistic regression. Thus, the logistic function inherently encapsulates a linear function enveloped within a sigmoid function. Despite possessing a narrower decision boundary, regressors possess the capacity to attain comparable levels of accuracy as classifiers. Therefore, our models embark on the classification endeavor by embracing regression as their modus operandi.

4.4.1. Most Scikit-learn Models

This model utilized most models from Probability Calibration, Dummy estimators, Ensemble Methods, Gaussian Processes, Linear classifiers, Classical linear regressors, Regressors with variable selection, Bayesian regressors, Outlier-robust regressors, Generalized linear models (GLM) for regression, Miscellaneous, Naive Bayes, Nearest Neighbors, Neural network models, Semi-Supervised Learning, Support Vector Machines, and Decision Trees to compare their effectiveness.

Regressors As Classifiers

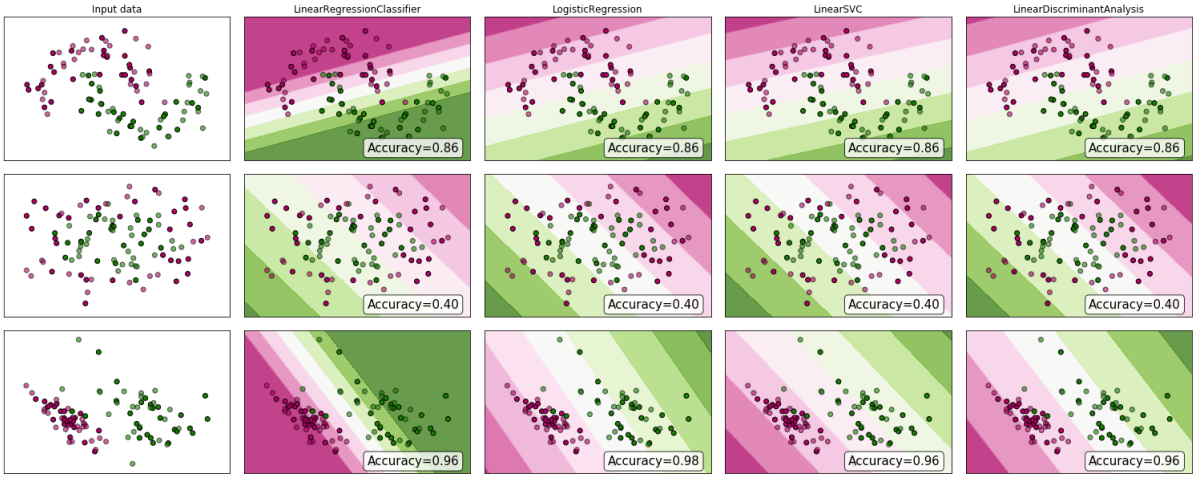

Figure 8: Accuracy of Classifiers and Regressors from Scikit-learn

Within the aforementioned feature space, it is noteworthy that a regression model can exhibit analogous behavior to classifiers, furnishing a hyperplane capable of partitioning the distinct classes. Consequently, regressors have the potential to achieve commensurate levels of accuracy as classifiers.

4.4.2. Feature Discretization

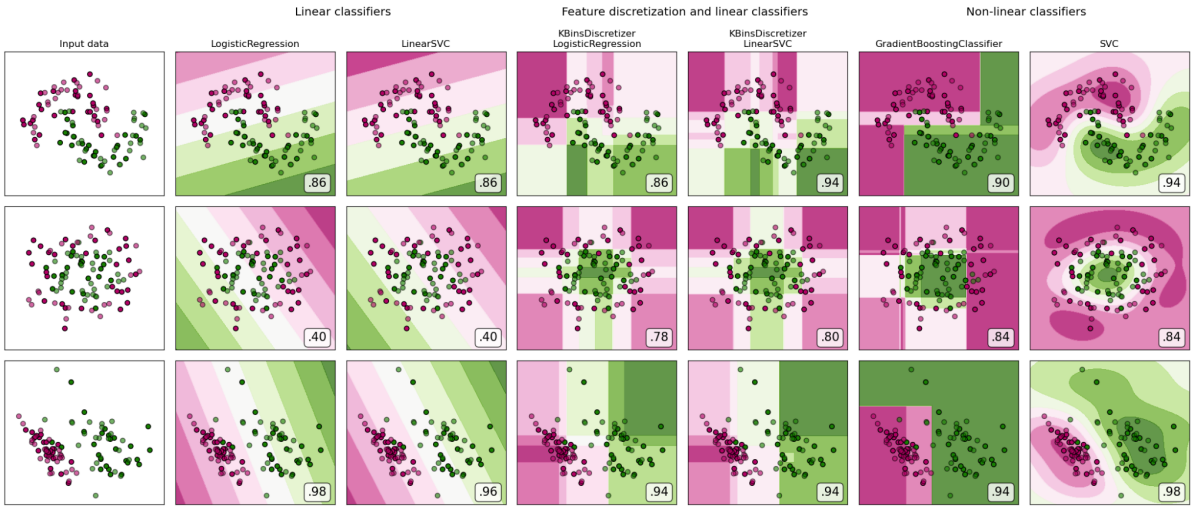

Figure 9: Feature Discretization from Scikit-learn

Dimension Reduction

The preliminary rows exhibit datasets characterized by non-linear separability, specifically exemplified by the intricate arrangements of the moons and concentric circles. In contrast, the third row portrays a dataset that manifests a semblance of

linearity in its separability. The judicious utilization of feature discretization yields a profound amplification in the effectiveness of linear classifiers when confronted with datasets that defy linear separability. Alas, when confronted with the linearly separable dataset, the efficacy of linear classifiers is woefully undermined by the application of feature discretization.

In this instance, price data is more analyzable when continuous price action is interpreted in daily batches with the same duration.

4.5. Strategies

After obtaining the data frame from yahoo finance, the data is processed to yield two new data frames (Volatility column, and Decision column). If the open price exceeds the close price, a decision of 1 will be generated; vice versa, a 0 will be generated. To associate trading with classification, trading behaviors, and profit, the Action column, and Profit column are created from the classification results. The trading rules for swing trading and day trading are derived from the same classification results (Decision column). Their only difference would be their final interpretation. No scaling would be used as the future prices of assets are unpredictable. To reduce complexity, no technical indicators will be employed in this study. All positions will be closed on the last day.

Action Type	
Swing Trading	**Day Trading**
1. Buy 2. Retain 3. Sell 4. No Action	1. Buy & Sell 2. No Action

Table 9: Actions Derived from Machine Learning Strategies

As shown in the table above, two distinctive action sets for swing trading and day trading are yielded from the same classification results.

Step - 0							
	Date	**Open**	**High**	**Low**	**Close**	**Adj Close**	**Volume**
0	2012-01-02 00:00:00	0.778089	0.780396	0.777122	0.778695	0.778695	0
1	2012-01-03 00:00:00	0.781006	0.790514	0.780518	0.781006	0.781006	0
2	2012-01-04 00:00:00	0.790326	0.790514	0.785731	0.790389	0.790389	0
3	2012-01-05 00:00:00	0.786596	0.787774	0.779727	0.786596	0.786596	0

Table 10: Original Data of [NZDUSD=X]

As shown in the above table, the original data is obtained from yahoo finance. No preprocessing is conducted yet.

Step - 1			
	...	**Volatility**	**Decision**
0	...	0.000000	0.000000
1	...	0.002968	1.000000

2	...	0.012014	0.000000
3	...	-0.004798	0.000000

Table 11: Generation of Volatility and Decision Columns

In the aforementioned table, two new columns - Volatility and Decision columns are created and appended to the original data sets. To infer today's action from the previous day, Decision columns are shifted one row up.

4.5.1. Strategy 1 - Swing Trading

Positions remain open from one to multiple days. A Decision of 1 from the previous day triggers a Buy action tomorrow. A Sell action will be triggered with a Decision of 0. Retain will be triggered if a Decision of 1 is detected with opened positions. The position will only be entered and exited at an open price.

Step - 2			
	...	**Swing Action**	**Swing Profit**
0	...	No Action	10000.000000
1	...	No Action	10000.000000
2	...	Buy	5000.395466
3	...	Sell	9976.404403

Table 12: Attachment of Action and Profit Columns for Swing Trading

As revealed in the above table, the profit of the swing trading strategy is calculated based on the corresponding actions. Both columns are appended for further analysis.

4.5.2. Strategy 2 - Day Trading

Positions remain open for only one day. A Decision of 1 from the previous day triggers a Buy & Sell action tomorrow. The position will only be entered at the open price and exited at the close price on the same day.

Step - 3			
	...	Day Action	Day Profit
0	...	No Action	10000.000000
1	...	No Action	10000.000000
2	...	Buy & Sell	10000.395535
3	...	No Action	10000.395535

Table 13: Attachment of Action and Profit Columns for Day Trading

As displayed in the above table, the profit of the day trading strategy is computed based on the actions taken, with both columns included for additional examination.

	Date	Open	High	Low	Close	Adj Close	Volume	Volatility	Decision	Swing Action	Swing Profit	Day Action	Day Profit
0	2000-01-03 00:00:00	102.070000	103.330002	101.309998	101.690002	101.690002	0	0.000000	0.000000	No Action	10000.000000	No Action	10000.000000
1	2000-01-04 00:00:00	101.639999	103.320000	101.470001	103.139999	103.139999	0	0.014259	1.000000	No Action	10000.000000	No Action	10000.000000
2	2000-01-05 00:00:00	103.129997	104.480003	102.750000	104.089996	104.089996	0	0.009211	1.000000	Buy	5049.760132	Buy & Sell	10185.279823
3	2000-01-06 00:00:00	104.089996	105.559998	103.900002	105.230003	105.230003	0	0.010952	1.000000	Retain	5049.760132	Buy & Sell	10407.581192
4	2000-01-07 00:00:00	105.260002	105.879997	104.830002	105.330002	105.330002	0	0.000950	1.000000	Retain	5049.760132	Buy & Sell	10421.371132
5	2000-01-10 00:00:00	105.099998	105.849998	104.279999	105.160004	105.160004	0	-0.001614	1.000000	Retain	5049.760132	Buy & Sell	10433.252159
6	2000-01-11 00:00:00	105.150002	106.309998	105.050003	106.019997	106.019997	0	0.008178	0.000000	Retain	5049.760132	Buy & Sell	10605.511192
7	2000-01-12 00:00:00	106.010002	107.309998	105.480003	105.709999	105.709999	0	-0.002924	1.000000	Sell	10138.240234	No Action	10605.511192
8	2000-01-13 00:00:00	105.720001	106.230003	105.389999	106.019997	106.019997	0	0.002933	0.000000	Buy	5169.400177	Buy & Sell	10665.510277
9	2000-01-14 00:00:00	106.080002	106.580002	105.410004	105.879997	105.879997	0	-0.001320	0.000000	Sell	10155.160263	No Action	10665.510277
10	2000-01-17 00:00:00	105.839996	105.860001	104.250000	104.889999	104.889999	0	-0.009350	1.000000	No Action	10155.160263	No Action	10665.510277

Figure 10: Data Frame after Preprocessing

```
[117] pd.DataFrame(ranking_x[0][0][0][0])
```

	0	1	2	3	4	5
0	USDJPY=X	USDJPY=X	USDJPY=X	USDJPY=X	USDJPY=X	USDJPY=X
1	CalibratedClassifierCV	DummyClassifier	DummyRegressor	AdaBoostClassifier	AdaBoostRegressor	BaggingClassifier
2	MinMaxScaler(feature_range=(0, 100))	MinMaxScaler(feature_range=(0, 100))	MinMaxScaler(feature_range=(0, 100))	MinMaxScaler(feature_range=(0, 100))	MinMaxScaler(feature_range=(0, 100))	MinMaxScaler(feature_range=(0, 100))
3	0.538551	0.471473	0.471473	0.510794	0.479954	0.509252
4	1907.281288	1907.281288	1907.281288	1907.281288	1907.281288	1907.281288
5	[10000, 9821.076705932617, 9686.319229125977, 9521.752197265625, 9222.883728027344, 9000.684036254883, 8799.892501831055, 8709.94497680664, 8542.39370727539, 8407.367309570312, 8246.47356414795, 7732.149375915527, 7858.29012298584, 7740.689353942871, 7675.285820007324, 7571.501182556152, 7213.12166595459,	[10000, 9972.71987915039, 10079.810531616211, 10084.673233032227, 9904.115936279297, 9768.116455078125, 9862.510925292969, 9694.867874145508, 9768.307083129883, 9728.813255310059, 9832.492332458496, 9940.992332458496, 9872.66510772705, 9959.863777160645, 9646.69538116455, 9615.091339111328, 9606.49114227295,	[10000, 9972.71987915039, 10079.810531616211, 10084.673233032227, 9904.115936279297, 9768.116455078125, 9862.510925292969, 9694.867874145508, 9768.307083129883, 9728.813255310059, 9832.492332458496, 9940.992332458496, 9872.66510772705, 9959.863777160645, 9646.69538116455, 9615.091339111328, 9606.49114227295,	[10000, 10107.579650878906, 9927.022354125977, 9791.022872924805, 9885.417343139648, 9960.216537475586, 10066.295593261719, 10177.795593261719, 10267.794219970703, 10234.13224029541, 10361.913932800293, 10394.113792419434, 10425.759269714355, 10421.601791381836, 10603.761749267578, 10348.562103271484,	[10000, 9972.71987915039, 10079.810531616211, 10084.673233032227, 9904.115936279297, 9768.116455078125, 9862.510925292969, 9694.867874145508, 9768.307083129883, 9728.813255310059, 9832.492332458496, 9940.992332458496, 9872.66510772705, 9959.863777160645, 9646.69538116455, 9615.091339111328, 9606.49114227295,	[10000, 10096.569458007812, 10173.068634033203, 10131.936813354492, 10243.936813354492, 10334.335433959961, 10300.3794631958, 10291.179252624512, 10419.518943786621, 10415.361465454102, 10597.521423339844, 10342.32177734375, 10371.862274169922, 10262.821990966797, 10225.861145019531, 10355.122085571289,

Figure 11: Evaluation Array for all Regressors and Classifiers

4.6. Analytical Metrics

To determine the profitability of the strategies, a collection of classification, and trading metrics have been utilized. Belows listed the contexts of each metric.

4.6.1. Classification Metrics

Most classification metrics have been embedded in this report. However, not all of them are analyzed because of the limitation of computing resources. Only the result of accuracy is further processed.

metrics.accuracy_score (The higher the better)	This metric measures the proportion of correctly classified samples out of the total number of samples and therefore higher values indicate better performance.
metrics.auc (The higher the better)	The area under the curve (AUC) measures the ability of a binary classifier to distinguish between positive and negative samples and values close to 1 indicate better performance.
metrics.average_precision_score (The higher the better)	This metric measures the weighted average of precision achieved at different threshold values and therefore higher values indicate better performance.
metrics.balanced_accuracy_score (The higher the better)	This metric is similar to accuracy but takes into account imbalanced datasets by computing the average of recall obtained for each class and therefore higher values indicate better

	performance.
metrics.brier_score_loss (The lower the better)	This metric measures the mean squared difference between the predicted probability and the real outcome and therefore lower values indicate better performance.
metrics.classification_report	This is not a metric but a function that generates a text report with various classification metrics including precision, recall, and F1 score for each class. The interpretation of these metrics is similar to what is described for metrics.precision_recall_fscore_support.
metrics.cohen_kappa_score (The higher the better)	This metric measures the agreement between two raters and values close to 1 indicate better agreement.
metrics.confusion_matrix	This is not a metric but a function that generates a matrix with the number of true positives, false positives, true negatives, and false negatives. The interpretation of this matrix depends on the context and the application.
metrics.dcg_score (The higher the better)	This metric measures the quality of a ranking of documents and therefore higher values indicate better performance.
metrics.det_curve	This is not a metric but a function that generates a curve of detection rate (i.e.,

	true positive rate) as a function of false positive rate for different threshold values. The interpretation of this curve is similar to what is described for metrics.roc_curve.
metrics.f1_score (The higher the better)	This metric is the harmonic mean of precision and recall and therefore higher values indicate better performance.
metrics.fbeta_score (The higher the better)	This metric is a weighted harmonic mean of precision and recall with a weight parameter beta and therefore higher values indicate better performance.
metrics.hamming_loss (The lower the better)	This metric measures the fraction of misclassified labels and therefore lower values indicate better performance.
metrics.hinge_loss (The lower the better)	This metric measures the margin between the predicted score and the true score and therefore lower values indicate better performance.
metrics.jaccard_score (The higher the better)	This metric measures the intersection over the union between two sets and therefore higher values indicate better similarity.
metrics.log_loss (The lower the better)	This metric measures the negative log-likelihood of the predicted probability and therefore lower values indicate better performance.

metrics.matthews_corrcoef (The lower the better)	This metric measures the correlation between the predicted and true binary labels and values close to 1 indicate better performance.
metrics.multilabel_confusion_matrix	This is not a metric but a function that generates a matrix of confusion matrices, one for each label. The interpretation of these matrices is similar to what is described for metrics.confusion_matrix.
metrics.ndcg_score (The higher the better)	This metric is a normalized version of DCG and therefore higher values indicate better performance.
metrics.precision_recall_curve	This is not a metric but a function that generates a curve of precision as a function of recall for different threshold values. The interpretation of this curve is similar to what is described for metrics.roc_curve.
metrics.precision_recall_fscore_support	This is not a metric but a function that generates arrays of precision, recall, and F1 scores for each class. The interpretation of these metrics is similar to what is described for metrics.f1_score.
metrics.precision_score (The higher the better)	This metric measures the fraction of true positives among the predicted positives and therefore higher values indicate better performance.
metrics.recall_score	This metric measures the fraction of

(The higher the better)	true positives among the actual positives and therefore higher values indicate better performance.
metrics.roc_auc_score (The higher the better)	This metric measures the ability of a binary classifier to rank positive samples higher than negative samples and values close to 1 indicate better performance.
metrics.roc_curve	This is not a metric but a function that generates a curve of true positive rate as a function of false positive rate for different threshold values. The interpretation of this curve is similar to what is described for metrics.det_curve.
metrics.top_k_accuracy_score (The higher the better)	This metric measures the proportion of samples for which the true label is among the top k predicted labels and therefore higher values indicate better performance.
metrics.zero_one_loss (The higher the better)	This metric measures the fraction of misclassified samples and therefore lower values indicate better performance.

Table 14: Description of Classification Metrics

Illustrated in the table, the classification report helps us to evaluate the effectiveness of trading strategies. A great strategy should show a directly proportional relationship between classification and profit.

4.6.2. Trading Metrics

To examine the profitability of models, major trading metrics have been added.

4.6.2.1. Net Profit

It is the total profit or loss made by a trader over a specific period after all expenses and commissions have been accounted for. It is a measure of the trader's profitability.

$$\text{Net Profit} = \text{Total Revenue} - \text{Total Expenses} - \text{Total Losses}$$

where

1. $TotalRevenue$ is the total amount earned from the investment or portfolio,
2. $TotalExpenses$ is the total amount spent on expenses related to the investment or portfolio, and
3. $TotalLosses$ is the total amount lost from the investment or portfolio.

4.6.2.2. Compounded Annual Growth Rate

It is the rate of return on an investment over a specific period, taking into account the effect of compounding on the investment returns. It is a measure of the trader's performance over time.

$$\text{CAGR} = \left(\frac{V_f}{V_i}\right)^{\frac{1}{n}} - 1$$

where

1. V_i is the initial value of the investment or portfolio,
2. V_f is the final value of the investment or portfolio after n years, and
3. $\frac{1}{n}$ is the annualization factor.

```
def cagr(account_money, years):
 initial_value = account_money[0]
 final_value = account_money[-1]
 cagr = (final_value / initial_value) ** (1 / years) - 1
 return cagr
```

Code snippet 1: Compounded Annual Growth Rate

4.6.2.3. Sharpe Ratio

It is a measure of the risk-adjusted return of an investment, taking into account the volatility of the investment returns and comparing them to the risk-free rate of return. It is a measure of the trader's ability to generate returns relative to the amount of risk taken.

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p}$$

where

1. R_p is the portfolio return,
2. R_f is the risk-free rate of return, and
3. σ_p is the standard deviation of the portfolio return.

```
def sharpe_ratio(account_money, risk_free_rate, years):
    total_trades = len(account_money) - 1
    returns = np.diff(account_money) / account_money[:-1]
    avg_return = np.mean(returns) * total_trades / years
    std_dev = np.std(returns) * np.sqrt(total_trades / years)
    excess_return = avg_return - risk_free_rate
    if std_dev == 0 or np.isnan(std_dev):
        return 0.0
    sharpe_ratio = excess_return / std_dev
    sharpe_ratio = np.clip(sharpe_ratio, -3.0, 3.0)
    return sharpe_ratio
```

Code snippet 2: Sharpe Ratio

4.6.2.4. Maximum Drawdown

It is the maximum loss a trader has experienced during a specific period, from the highest peak to the lowest trough. It is a measure of the trader's risk tolerance and ability to manage drawdowns.

$$\text{Maximum Drawdown} = \max_{i,j:i<j} \left\{ \frac{V_i - V_j}{V_i} \right\}$$

where

1. V_i is the portfolio value at time i,
2. V_j is the portfolio value at time j, and
3. the maximum is taken over all pairs of indices i and j such that $i < j$.

```
def max_drawdown(prices):
    max_so_far = prices[0]
    max_drawdown = 0
    for price in prices:
        if price > max_so_far:
            max_so_far = price
        else:
            drawdown = 1 - price / max_so_far
            if drawdown > max_drawdown:
                max_drawdown = drawdown
    return max_drawdown
```

Code snippet 3: Maximum Drawdown

4.6.2.5. Recovery Rate

It is the percentage of the maximum drawdown that a trader has been able to recover. It is a measure of the trader's ability to recover from losses.

$$\text{Recovery Rate} = \frac{\text{Amount Recovered}}{\text{Amount Owed}} \times 100\%$$

where

1. $AmountRecovered$ is the total amount that is recovered after a default or other loss event, and
2. $AmountOwed$ is the total amount that was originally lent or invested.

```
def recovery_rate(prices, max_drawdown):
    max_so_far = prices[0]
    for price in prices:
        if price > max_so_far:
            max_so_far = price
```

```
    else:
        drawdown = 1 - price / max_so_far
        if drawdown > max_drawdown:
            max_drawdown = drawdown
recovery_rate = 1 - max_drawdown
return recovery_rate
```

Code snippet 4: Recovery Rate

4.6.2.6. Maximum Profit

It is the highest profit a trader has made on a single trade. It is a measure of the trader's ability to identify and capitalize on profitable opportunities.

$$\text{Maximum Profit} = \max_{i,j:i<j} V_j - V_i$$

where

1. V_i is the portfolio value at time i, and V_i is the portfolio value at time i, and
2. V_j is the portfolio value at time j, and V_j is the portfolio value at time j, and
3. the maximum is taken over all pairs of indices i and j such that $i < j$.

4.6.2.7. Maximum Loss

It is the highest loss a trader has experienced on a single trade. It is a measure of the trader's ability to manage risk and cut losses.

$$\text{Maximum Loss} = \max_{i,j:i<j} V_i - V_j$$

where

1. V_i is the portfolio value at time i, and V_i is the portfolio value at time i, and
2. V_j is the portfolio value at time j, and V_j is the portfolio value at time j, and
3. the maximum is taken over all pairs of indices i and j such that $i < j$.

4.6.2.8. Total Trades

It is the total number of trades executed by a trader over a specific period. It is a measure of the trader's activity and engagement in the market.

$$\text{Total Trades} = \sum_{i=1}^{n} I_i$$

where

1. n is the total number of time periods, and n is the total number of time periods, and
2. I_i is an indicator variable that takes a value of 1 if a trade was executed at time i, and 0 otherwise.

5. Performance Evaluation

Notice

1. The analysis is pruned due to the limitation of computing power and cost.
2. Only accuracy is analyzed.
3. Only forex with the top 10 market capitalization is tested.
1. Please notice regressors are used with normalization in the range of [0,1].
2. Use asset_report(, True) to see the detailed classification report.
3. No Scaling is conducted as the future range of assets is unpredictable.

Dataframes

[yFinance]

Training	Trading
1. 2000-01-01 - 2010-01-01 (Daily)	1. 2010-01-01 - 2020-01-01 (Daily)

Scalers

[Sklearn]

[Deprecated]

1. MinMaxScaler(feature_range=(0, 100)),
2. StandardScaler()

Metrics

[Sklearn]

Trading	Classification
1. Net Profit 2. Compound Annual Growth Rate 3. Sharpe ratio 4. Maximum Dropdown 5. Recovery Rate 6. Maximum Profit 7. Maximum Loss 8. Maximum Holding Time	1. metrics.accuracy_score,
	[Deprecated] 2. metrics.auc, 3. metrics.average_precision_score, 4. metrics.balanced_accuracy_score, 5. metrics.brier_score_loss, 6. metrics.classification_report, 7. metrics.cohen_kappa_score, 8. metrics.confusion_matrix, 9. metrics.dcg_score, 10. metrics.det_curve, 11. metrics.f1_score, 12. metrics.fbeta_score, 13. metrics.hamming_loss, 14. metrics.hinge_loss, 15. metrics.jaccard_score, 16. metrics.log_loss,

	17. metrics.matthews_corrcoef, 18. metrics.multilabel_confusion_matrix, 19. metrics.ndcg_score, 20. metrics.precision_recall_curve, 21. metrics.precision_recall_fscore_support, 22. metrics.precision_score, 23. metrics.recall_score, 24. metrics.roc_auc_score, 25. metrics.roc_curve, 26. metrics.top_k_accuracy_score, 27. metrics.zero_one_loss,

Assets

[GPT-3]

Stock	Commodity	Forex
[Deprecated] 1. 'BAC', # Bank of America 2. 'C', # Citigroup 3. 'F', # Ford Motor Company 4. 'GE', # General Electric 5. 'CSCO', # Cisco Systems 6. 'PFE', # Pfizer Inc. 7. 'MSFT', # Microsoft Corporation	[Deprecated] 11. 'CL=F', # Crude oil 12. 'GC=F', # Gold 13. 'NG=F', # Natural gas 14. 'ZC=F', # Corn 15. 'HO=F', # Heating Oil 16. 'BZ=F', # Brent Oil 17. 'SI=F', # Silver 18. 'ZS=F', # Soybeans	21. 'USDJPY=X', # US Dollar/Japanese Yen 22. 'EURUSD=X', # Euro/US Dollar 23. 'GBPUSD=X', # British Pound/US Dollar 24. 'AUDUSD=X', # Australian Dollar/US Dollar 25. 'USDCHF=X', # US Dollar/Swiss Franc

8. 'AA', # Alcoa Inc. 9. 'T', # AT&T Inc. 10. 'INTC' # Intel Corporation	19. 'HG=F', # Copper 20. 'ZW=F', # Wheat	26. 'USDCAD=X', # US Dollar/Canadian Dollar 27. 'EURJPY=X', # Euro/Japanese Yen 28. 'GBPJPY=X', # British Pound/Japanese Yen 29. 'EURCHF=X', # Euro/Swiss Franc 30. 'NZDUSD=X', # New Zealand Dollar/US Dollar

Models

[Sklearn]

Probability Calibration	**Dummy estimators**	**Ensemble Methods**
1. calibration.CalibratedClassifierCV(),	2. dummy.DummyClassifier(), 3. dummy.DummyRegressor(),	4. ensemble.AdaBoostClassifier(), 5. ensemble.AdaBoostRegressor(), 6. ensemble.BaggingClassifier(), 7. ensemble.BaggingRegressor(), 8. ensemble.ExtraTreesClassifier(),

		9. ensemble.ExtraTreesRegressor(), 10. ensemble.GradientBoostingClassifier(), 11. ensemble.GradientBoostingRegressor(), 12. ensemble.IsolationForest(), 13. ensemble.RandomForestClassifier(), 14. ensemble.RandomForestRegressor(), 15. ensemble.HistGradientBoostingRegressor(), 16. ensemble.HistGradientBoostingClassifier(),
Gaussian Processes	**Linear classifiers**	**Classical linear regressors**
17. gaussian_process.GaussianProcessClassifier(), 18. gaussian_process.GaussianProcessRegressor(),	19. linear_model.LogisticRegression(), 20. linear_model.LogisticRegressionCV(), 21. linear_model.PassiveAggressiveClassifier(),	27. linear_model.LinearRegression(), 28. linear_model.Ridge(), 29. linear_model.RidgeCV(), 30. linear_model.SGDRegressor(),

	22. linear_model.Perceptron(), 23. linear_model.RidgeClassifier(), 24. linear_model.RidgeClassifierCV(), 25. linear_model.SGDClassifier(), 26. linear_model.SGDOneClassSVM(),	
Regressors with variable selection	**Bayesian regressors**	**Outlier-robust regressors**
31. linear_model.ElasticNet(), 32. linear_model.ElasticNetCV(), 33. linear_model.Lars(), 34. linear_model.LarsCV(), 35. linear_model.Lasso(), 36. linear_model.LassoCV(), 37. linear_model.LassoLars(), 38. linear_model.LassoLarsCV(), 39. linear_model.LassoLarsIC(),	42. linear_model.ARDRegression(), 43. linear_model.BayesianRidge(),	44. linear_model.HuberRegressor(), 45. linear_model.TheilSenRegressor(),

40. linear_model.OrthogonalMatchingPursuit(), 41. linear_model.OrthogonalMatchingPursuitCV(),		
Generalized linear models (GLM) for regression	**Miscellaneous**	**Naive Bayes**
46. linear_model.PoissonRegressor(), 47. linear_model.TweedieRegressor(),	48. linear_model.PassiveAggressiveRegressor(),	49. naive_bayes.BernoulliNB(), 50. naive_bayes.GaussianNB(),
Nearest Neighbors	**Neural network models**	**Semi-Supervised Learning**
51. neighbors.KNeighborsClassifier(), 52. neighbors.KNeighborsRegressor(), 53. neighbors.RadiusNeighborsRegressor(), 54. neighbors.NearestCentroid(),	55. neural_network.MLPClassifier(), 56. neural_network.MLPRegressor(),	57. semi_supervised.LabelPropagation(), 58. semi_supervised.LabelSpreading(),
Support Vector Machines	**Decision Trees**	
59. svm.LinearSVC(), 60. svm.LinearSVR(),	65. tree.DecisionTreeClassifier(),	

61. svm.NuSVR(), 62. svm.OneClassSVM(), 63. svm.SVC(), 64. svm.SVR(),	66. tree.DecisionTreeRegressor(), 67. tree.ExtraTreeClassifier(), 68. tree.ExtraTreeRegressor(),	

Table 15: Context of Multiple Model-based Trading Strategies

5.1. Visualization

5.1.1. Strategy 1 - Swing Trading

5.1.1.1. Accuracy

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
635	USDCHF=X	RidgeClassifierCV	0.645725	636.181269	-29.829384	-0.000515	2.998994	0.002983	0.997017	41.848689	0.000000	3
567	USDCHF=X	RidgeClassifierCV	0.645725	636.181269	-29.829384	-0.000515	2.998994	0.002983	0.997017	41.848689	0.000000	3
603	USDCHF=X	LinearSVR	0.645725	636.181269	-1.375756	-0.000024	nan	0.000138	0.999862	0.000000	-808.509035	1
561	USDCHF=X	GaussianProcessRegressor	0.645725	636.181269	-1.375756	-0.000024	nan	0.000138	0.999862	0.000000	-808.509035	1
671	USDCHF=X	LinearSVR	0.645725	636.181269	-1.375756	-0.000024	nan	0.000138	0.999862	0.000000	-808.509035	1
675	USDCHF=X	SVR	0.645725	636.181269	-1.375756	-0.000024	nan	0.000138	0.999862	0.000000	-808.509035	1
607	USDCHF=X	SVR	0.645725	636.181269	-1.375756	-0.000024	nan	0.000138	0.999862	0.000000	-808.509035	1
612	USDCHF=X	CalibratedClassifierCV	0.645725	636.181269	-2.809674	-0.000048	2.996980	0.000611	0.999389	38.388565	-787.903925	3
544	USDCHF=X	CalibratedClassifierCV	0.645725	636.181269	-2.809674	-0.000048	2.996980	0.000611	0.999389	38.388565	-787.903925	3
629	USDCHF=X	GaussianProcessRegressor	0.645725	636.181269	-1.375756	-0.000024	nan	0.000138	0.999862	0.000000	-808.509035	1

Figure 12: Accuracy Report [high to low]

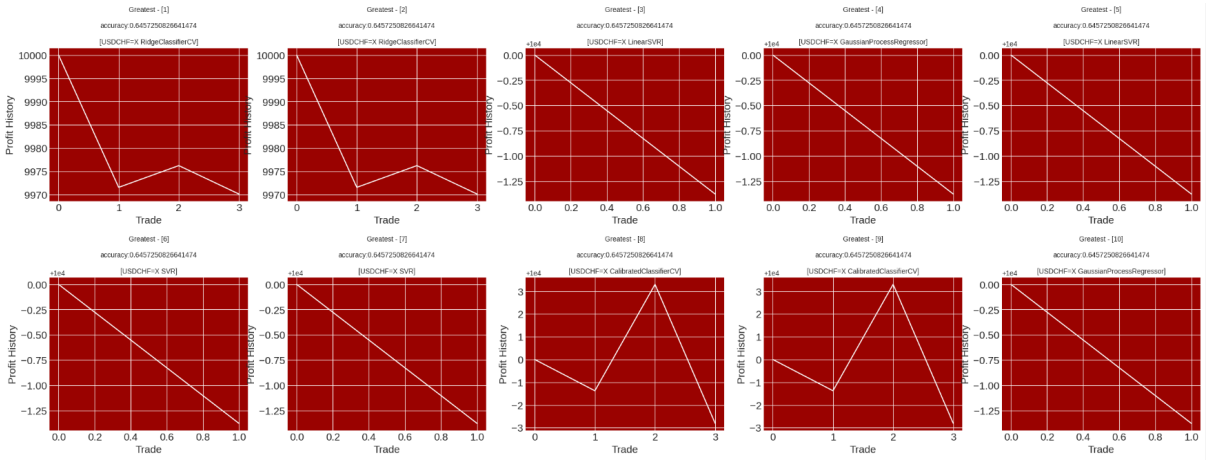

Figure 13: Accuracy Graph [high to low]

The fewer the trades, the higher the accuracy. Therefore, trading based on accuracy is

not reliable and profitable.

5.1.1.2. Compounded Annual Growth Rate

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
1322	NZDUSD=X	ElasticNet	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1244	NZDUSD=X	PassiveAggressiveClassifier	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1272	NZDUSD=X	BernoulliNB	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1340	NZDUSD=X	BernoulliNB	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1344	NZDUSD=X	RadiusNeighborsRegressor	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1328	NZDUSD=X	LassoLars	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1326	NZDUSD=X	Lasso	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1260	NZDUSD=X	LassoLars	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1258	NZDUSD=X	Lasso	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1254	NZDUSD=X	ElasticNet	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083

Figure 14: Compounded Annual Growth Rate Report [high to low]

Figure 15: Compounded Annual Growth Rate Graph [high to low]

Despite most different results, the same results are yielded on NZDUSD=X across the top-performing models. Therefore, a swing trading strategy may not be an efficient trading approach.

5.1.1.3. Sharpe Ratio

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
1293	NZDUSD=X	DummyClassifier	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1272	NZDUSD=X	BernoulliNB	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1226	NZDUSD=X	DummyRegressor	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1254	NZDUSD=X	ElasticNet	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1244	NZDUSD=X	PassiveAggressiveClassifier	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1225	NZDUSD=X	DummyClassifier	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1258	NZDUSD=X	Lasso	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1316	NZDUSD=X	SGDClassifier	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1340	NZDUSD=X	BernoulliNB	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083
1326	NZDUSD=X	Lasso	0.477218	-1224.100618	1640.929907	0.026956	3.000000	0.022668	0.977332	0.000000	-671.706950	2083

Figure 16: Sharpe Ratio Report [high to low]

Figure 17: Sharpe Ratio Graph [high to low]

It yields identical results as CAGR. Therefore, we may need to look for other trading metrics for comparison.

5.1.1.4. Maximum Drawdown

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
116	USDJPY=X	BernoulliNB	0.500771	1907.281288	-5452.733780	-0.104960	2.999848	0.545315	0.454685	174.929985	-188.748245	1278
48	USDJPY=X	BernoulliNB	0.500771	1907.281288	-5452.733780	-0.104960	2.999848	0.545315	0.454685	174.929985	-188.748245	1278
41	USDJPY=X	ARDRegression	0.482267	1907.281288	-4985.915359	-0.092568	2.999865	0.498830	0.501170	103.831161	-315.875702	1988
109	USDJPY=X	ARDRegression	0.482267	1907.281288	-4985.915359	-0.092568	2.999865	0.498830	0.501170	103.831161	-315.875702	1988
18	USDJPY=X	LogisticRegression	0.517733	1907.281288	-4791.089417	-0.087687	2.999869	0.479751	0.520249	723.748337	-145.139946	1040
86	USDJPY=X	LogisticRegression	0.517733	1907.281288	-4791.089417	-0.087687	2.999869	0.479751	0.520249	723.748337	-145.139946	1040
111	USDJPY=X	HuberRegressor	0.535852	1907.281288	-4558.073235	-0.082052	2.999868	0.455807	0.544193	454.271973	-137.080154	397
43	USDJPY=X	HuberRegressor	0.535852	1907.281288	-4558.073235	-0.082052	2.999868	0.455807	0.544193	454.271973	-137.080154	397
91	USDJPY=X	RidgeClassifierCV	0.507325	1907.281288	-4436.217216	-0.079187	2.999881	0.443895	0.556105	448.590591	-249.067009	1437
23	USDJPY=X	RidgeClassifierCV	0.507325	1907.281288	-4436.217216	-0.079187	2.999881	0.443895	0.556105	448.590591	-249.067009	1437

Figure 18: Maximum Drawdown Report [high to low]

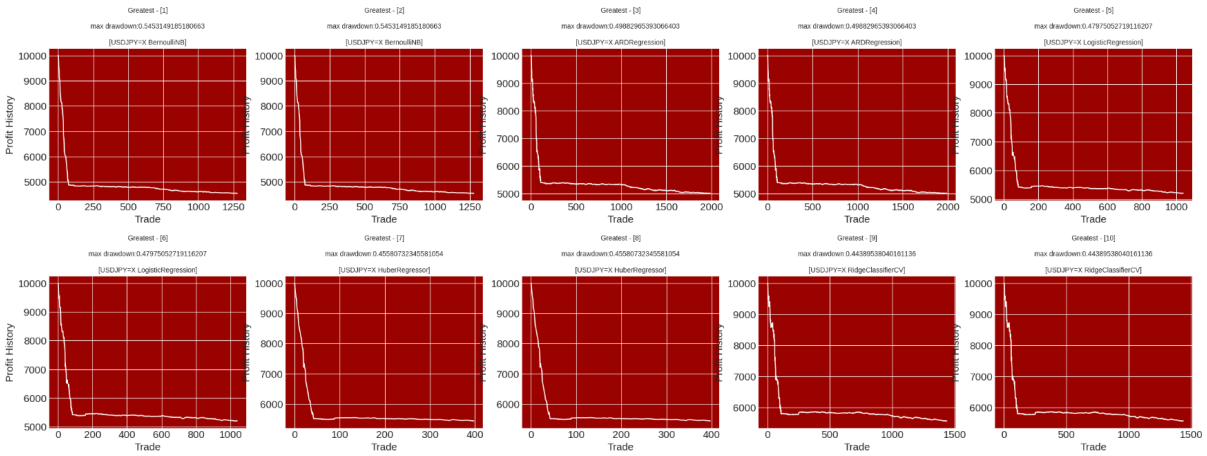

Figure 19: Maximum Drawdown Graph [high to low]

These are the worst-performing models based on MDD. Further loss is restricted as this strategy stops when the original capital is reduced by half.

5.1.2. Strategy 2 - Day Trading

5.1.2.1. Accuracy

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
635	USDCHF=X	RidgeClassifierCV	0.645725	636.181269	85.232508	0.001464	0.321749	0.000000	1.000000	4.668783	-28.399367	3
567	USDCHF=X	RidgeClassifierCV	0.645725	636.181269	85.232508	0.001464	0.321749	0.000000	1.000000	4.668783	-28.399367	3
603	USDCHF=X	LinearSVR	0.645725	636.181269	-808.509035	-0.014431	nan	0.080851	0.919149	0.000000	-1.375756	1
561	USDCHF=X	GaussianProcessRegressor	0.645725	636.181269	-808.509035	-0.014431	nan	0.080851	0.919149	0.000000	-1.375756	1
671	USDCHF=X	LinearSVR	0.645725	636.181269	-808.509035	-0.014431	nan	0.080851	0.919149	0.000000	-1.375756	1
675	USDCHF=X	SVR	0.645725	636.181269	-808.509035	-0.014431	nan	0.080851	0.919149	0.000000	-1.375756	1
607	USDCHF=X	SVR	0.645725	636.181269	-808.509035	-0.014431	nan	0.080851	0.919149	0.000000	-1.375756	1
612	USDCHF=X	CalibratedClassifierCV	0.645725	636.181269	-749.515360	-0.013343	1.988742	0.078790	0.921210	4.681382	-6.115300	2
544	USDCHF=X	CalibratedClassifierCV	0.645725	636.181269	-749.515360	-0.013343	1.988742	0.078790	0.921210	4.681382	-6.115300	2
629	USDCHF=X	GaussianProcessRegressor	0.645725	636.181269	-808.509035	-0.014431	nan	0.080851	0.919149	0.000000	-1.375756	1

Figure 20: Accuracy Report [high to low]

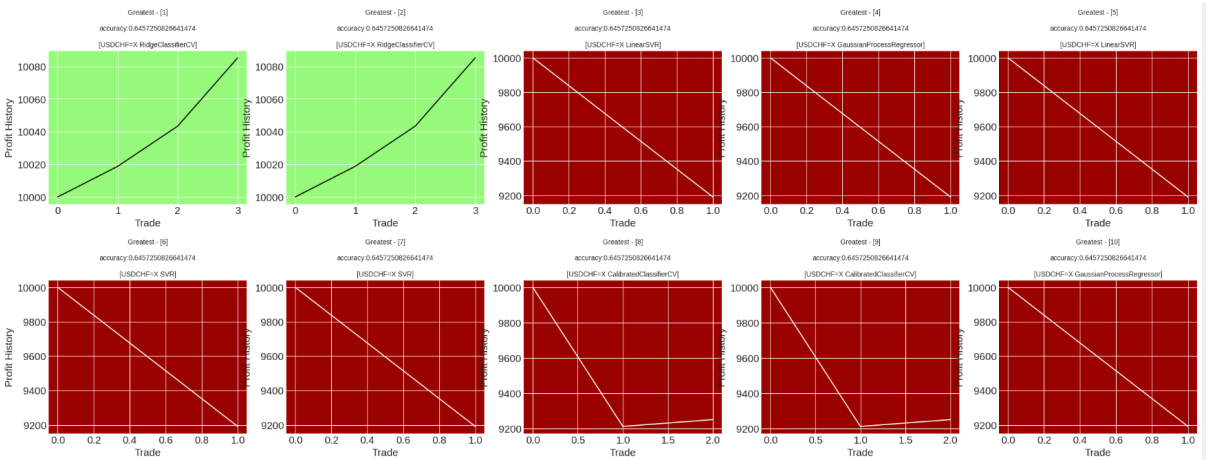

Figure 21: Accuracy Graph [high to low]

The fewer the trades, the lower the prediction error. Therefore, we may need to set a minimal trading threshold to truly evaluate the models' performance.

5.1.2.2. Compounded Annual Growth Rate

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
1042	GBPJPY=X	RidgeClassifier	0.507170	1683.827545	52985.301506	0.378626	2.483950	0.013219	0.986781	39.599648	-29.459274	404
974	GBPJPY=X	RidgeClassifier	0.507170	1683.827545	52985.301506	0.378626	2.483950	0.013219	0.986781	39.599648	-29.459274	404
875	EURJPY=X	LinearSVR	0.524807	884.481636	58524.969620	0.376779	2.642142	0.008871	0.991129	44.660141	-120.582230	483
1043	GBPJPY=X	RidgeClassifierCV	0.504302	1683.827545	52342.094276	0.376160	2.480290	0.013448	0.986552	39.599648	-29.459274	397
975	GBPJPY=X	RidgeClassifierCV	0.504302	1683.827545	52342.094276	0.376160	2.480290	0.013448	0.986552	39.599648	-29.459274	397
1038	GBPJPY=X	LogisticRegression	0.502868	1683.827545	42603.612122	0.335976	2.417668	0.013475	0.986525	39.599648	-29.665283	331
970	GBPJPY=X	LogisticRegression	0.502868	1683.827545	42603.612122	0.335976	2.417668	0.013475	0.986525	39.599648	-29.665283	331
519	AUDUSD=X	HuberRegressor	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463
451	AUDUSD=X	HuberRegressor	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463
434	AUDUSD=X	LinearRegression	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463

Figure 22: Compounded Annual Growth Rate Report [high to low]

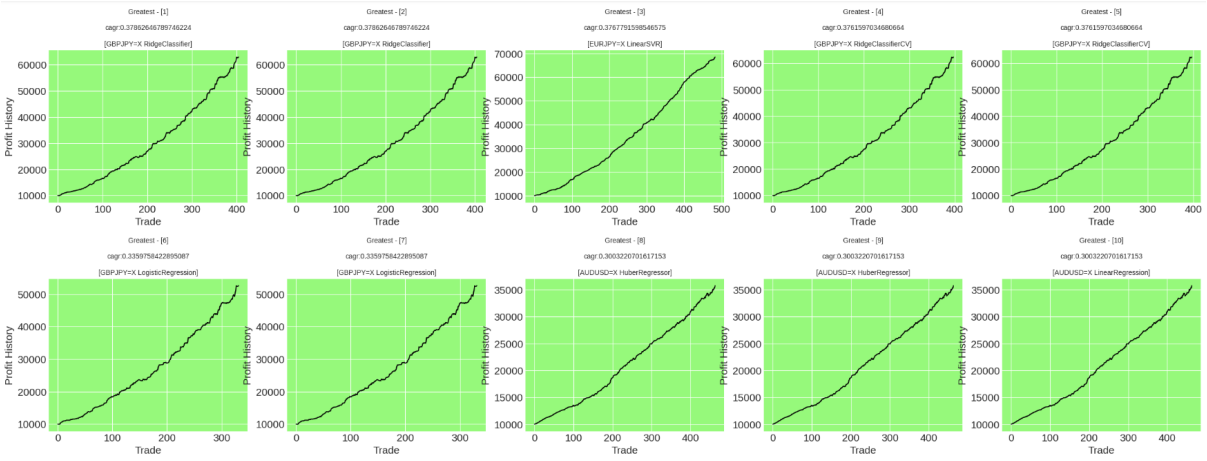

Figure 23: Compounded Annual Growth Rate Report [high to low]

These different winning results suggested that certain models can find patterns among price action. Moreover, day trading is more efficient than swing trading in terms of CAGR.

5.1.2.3. Sharpe Ratio

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
656	USDCHF=X	TheilSenRegressor	0.354275	636.181269	1377.845525	0.022505	3.000000	0.000000	1.000000	67.788522	-28.158172	2
875	EURJPY=X	LinearSVR	0.524807	884.481636	58524.969620	0.376779	2.642142	0.008871	0.991129	44.660141	-120.582230	483
519	AUDUSD=X	HuberRegressor	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463
434	AUDUSD=X	LinearRegression	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463
502	AUDUSD=X	LinearRegression	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463
451	AUDUSD=X	HuberRegressor	0.516366	-3186.365659	25784.408967	0.300322	2.620833	0.014615	0.985385	23.017211	-21.432681	463
18	USDJPY=X	LogisticRegression	0.517733	1907.281288	53664.750221	0.297526	2.548836	0.133610	0.866390	115.752686	-473.178253	624
86	USDJPY=X	LogisticRegression	0.517733	1907.281288	53664.750221	0.297526	2.548836	0.133610	0.866390	115.752686	-473.178253	624
974	GBPJPY=X	RidgeClassifier	0.507170	1683.827545	52985.301506	0.378626	2.483950	0.013219	0.986781	39.599648	-29.459274	404
1042	GBPJPY=X	RidgeClassifier	0.507170	1683.827545	52985.301506	0.378626	2.483950	0.013219	0.986781	39.599648	-29.459274	404

Figure 24: Sharpe Ratio Report [high to low]

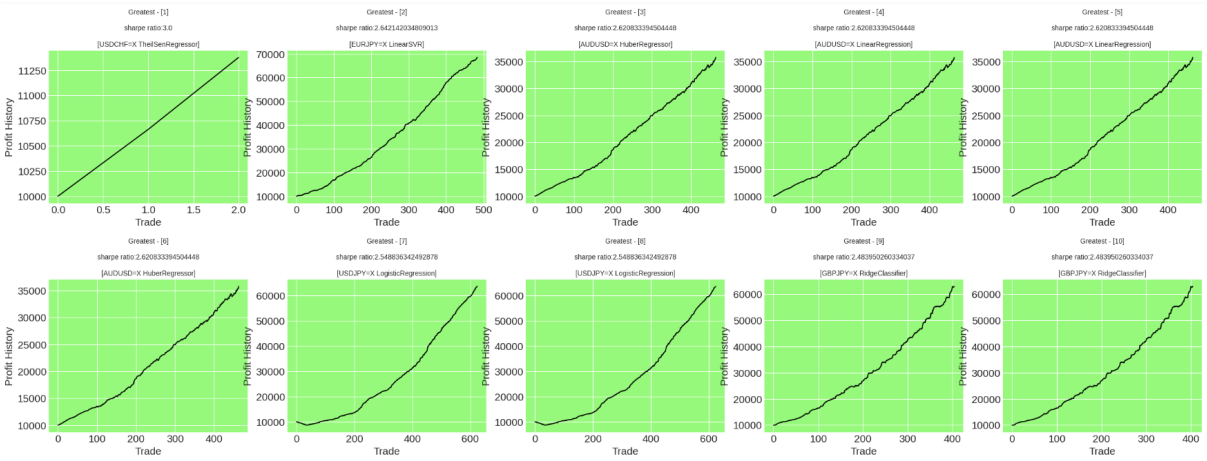

Figure 25: Sharpe Ratio Graph [high to low]

The results purposed that not always the model with the best Sharpe ratio could succeed significantly. Still, most top models have made a great profit.

5.1.2.4. Maximum Drawdown

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
551	USDCHF=X	ExtraTreesClassifier	0.545111	636.181269	-2307.953848	-0.044233	1.978855	0.239977	0.760023	67.717221	-28.537384	345
552	USDCHF=X	ExtraTreesRegressor	0.554558	636.181269	-2083.919427	-0.039490	1.988028	0.222628	0.777372	67.773021	-28.473065	362
556	USDCHF=X	RandomForestClassifier	0.565895	636.181269	-2092.326612	-0.039666	1.985023	0.217329	0.782671	67.745121	-28.477085	324
557	USDCHF=X	RandomForestRegressor	0.573925	636.181269	-2023.857795	-0.038238	1.985360	0.214553	0.785447	67.791622	-16.240170	346
624	USDCHF=X	RandomForestClassifier	0.560699	636.181269	-2032.817285	-0.038424	1.987558	0.208979	0.791021	67.773021	-28.544084	322
620	USDCHF=X	ExtraTreesRegressor	0.539443	636.181269	-1855.646262	-0.034771	1.993122	0.206991	0.793009	67.773021	-28.510585	372
412	AUDUSD=X	AdaBoostRegressor	0.490406	-3186.365659	-1847.146399	-0.041193	2.000584	0.202905	0.797095	23.582136	-46.151540	185
625	USDCHF=X	RandomForestRegressor	0.563533	636.181269	-1896.408776	-0.035605	1.988651	0.197933	0.802067	67.791622	-16.231040	345
1017	GBPJPY=X	DecisionTreeRegressor	0.483748	1683.827545	691.195801	0.011729	2.069393	0.189201	0.810799	39.599648	-27.387131	389
1019	GBPJPY=X	ExtraTreeRegressor	0.487572	1683.827545	21.178642	0.000369	2.054788	0.177498	0.822502	39.839645	-27.029129	388

Figure 26: Maximum Drawdown Report [high to low]

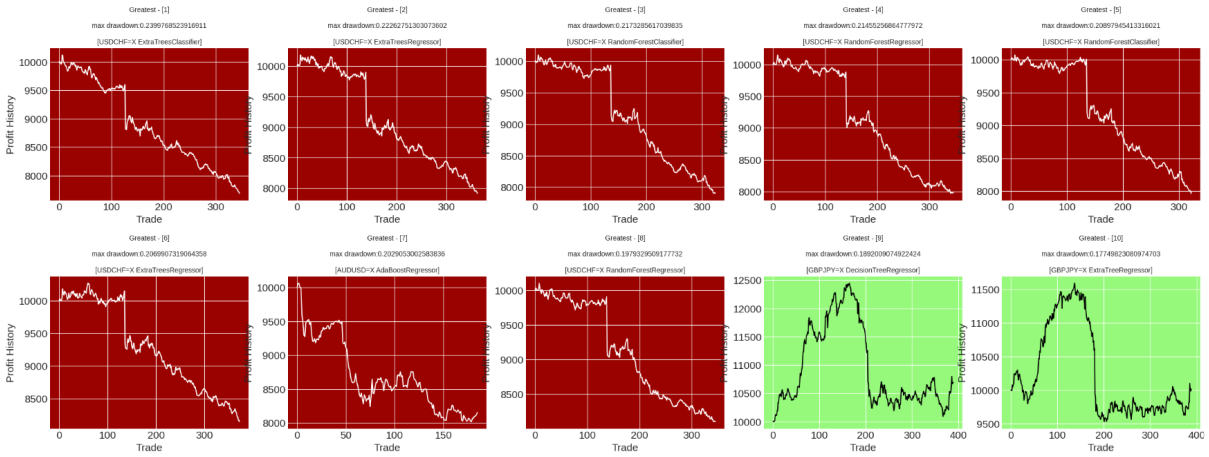

Figure 27: Maximum Drawdown Graph [high to low]

The higher the MDD, the lower the potential profit. Still, day trading is performing better than swing trading with the same MDD metrics.

5.2. Evaluation

Total Results = 10 forexes * 2 scalers * 68 models = 1360 results

Although this approach has included the scaler in the array. After trial and error, It is proven to be ineffective and deprecated for trading. To obtain the number of unique results, the total results should be divided in half.

Unique Results = 1360 * 0.5 = 680 results

The presented metric-based optimization is hardly perfect regarding the different trading situations. For instance, the accuracy, and maximum drawdown could be maximized and minimized due to the lack of trading for the strategy. The fewer the trades, the higher the accuracy. Therefore, an additional filter has to be employed to filter out trading approach that seldom trades.

5.3. Comparison

5.3.1. Filtered Models VS All Models

For numerical comparison, it is recommended to view the Colab notebooks. It can

even run multiple assets for analysis. With the exhaustion of computation resources, it only analyses forex data with accuracy metrics.

```
Gain of diversify investment in swing trading:

net profit ---> -13683392.978001952 ( -1.0061318366177907 % )
```

Figure 28: Dummy Investment with All Models in Swing Trading Strategy

```
Gain of diversify investment in day trading:

net profit ---> -12078918.743174851 ( -0.8881557899393273 % )
```

Figure 29: Dummy Investment with All Models in Day Trading Strategy

Despite the superiority of the performance of an individual model, the overall return of using all models to trade is not great. Therefore, we should always select the best few models for the sake of profit maximization.

5.3.1. Filtered models VS Buy-and-Hold Approach

	asset	model	accuracy	profit threshold	net profit	cagr	sharpe ratio	max drawdown	recovery rate	max profit	max loss	total trade
680	USDCAD=X	CalibratedClassifierCV	0.497167	2847.411386	462.221408	0.007817	nan	0.000000	1.000000	21.056642	-13.167994	1
723	USDCAD=X	HuberRegressor	0.513220	2847.411386	1050.005814	0.017356	2.074429	0.042565	0.957435	21.160768	-13.202031	178
729	USDCAD=X	GaussianNB	0.507082	2847.411386	800.953337	0.013367	2.064271	0.041632	0.958368	21.156657	-13.192832	117
728	USDCAD=X	BernoulliNB	0.380076	2847.411386	1421.504364	0.023170	nan	0.000000	1.000000	61.641415	-38.334748	1
727	USDCAD=X	PassiveAggressiveRegressor	0.499056	2847.411386	1072.774777	0.017717	2.075249	0.000000	1.000000	61.641415	-38.334748	6
726	USDCAD=X	TweedieRegressor	0.500472	2847.411386	289.328157	0.004927	2.008775	0.007055	0.992945	21.090894	-13.189152	5
725	USDCAD=X	PoissonRegressor	0.500472	2847.411386	289.328157	0.004927	2.008775	0.007055	0.992945	21.090894	-13.189152	5
724	USDCAD=X	TheilSenRegressor	0.559962	2847.411386	9191.712558	0.118896	2.301055	0.024639	0.975361	21.107335	-13.170754	288
722	USDCAD=X	BayesianRidge	0.500472	2847.411386	289.328157	0.004927	2.008775	0.007055	0.992945	21.090894	-13.189152	5
748	USDCAD=X	CalibratedClassifierCV	0.497167	2847.411386	462.221408	0.007817	nan	0.000000	1.000000	21.056642	-13.167994	1

Figure 30: Maximum Profit for the Buy-and-Hold Approach [USDCAD=X]

The profit threshold represents the profit of a Buy-and-Hold approach. Illustrated in the above figure, the profit of the buy-and-hold approach is surprisingly low compared to ML approaches. Simple supervised learning models - logistic regression or linear regression can already achieve great results. Also, regressors could have a similar effect with classifiers for classification.

5.4. Abnormality

Dummy Regressor/Classifier is performing great in certain strategies. It has implied a great model should significantly outperform not only buy-and-hold approaches but also the dummy models.

Its occupancy is an implication of successful bad trade. As dummy models fail to deliver constancy, the result is valid.

6. Improvement

6.1. Hardware

The temporal constraints imposed by Google Colab, which may be as restrictive as 12 hours per session, can pose a significant challenge for complex, long-running machine-learning tasks that necessitate days or even weeks of uninterrupted processing. Furthermore, the notebook's limited functionality and inability to generate comprehensive reports for all classification metrics underscore the necessity of leveraging a more potent TPU/GPU to unlock the full potential of machine learning models.

6.2. Data Source

Despite the vast quantity of data provided by Yahoo Finance's Python API, its quality is subject to variability, with potential discrepancies arising from the absence of adjustments for corporate actions such as dividends, stock splits, and other related events. This may impede the accuracy and reliability of any ensuing analysis. Moreover, the API's coverage may be incomplete, with less prominent or smaller stocks omitted from the dataset. Such limitations may restrict the scope of analysis, rendering Yahoo Finance's stock data an unsuitable option for professional trading analysis, particularly given its lack of inclusion of spreads.

6.3. Technical Analysis

The machine learning models solely deduce trading action based on rudimentary upward or downward trends in the market, which may prove inadequate for professional traders who typically rely on a range of technical indicators to inform their decisions. As such, the incorporation of classical trading patterns, such as

double tops, higher highs, and cup-and-handle formations, is imperative for a more nuanced and sophisticated approach to trading.

7. Conclusion

To conclude, machine learning models have yielded auspicious outcomes, with supervised learning paradigms proving effective in the classification of market trends and patterns. Such models have empowered traders to cultivate intricate strategies for the anticipation of forthcoming market movements, furnishing them with a marked competitive advantage over their counterparts.

Machine learning models have evinced noteworthy potential for the recognition of market trends, the prognostication of future movements, and the establishment of remunerative trading strategies. Despite the substantial demands of resources and proficiency requisite for their implementation, the potential advantages of these models render them a compelling toolset for traders striving to acquire a competitive edge in the mercurial and constantly evolving realm of finance.

8. References

[1] V. K. Pandey et al., “Machine learning algorithms and fundamentals as emerging safety tools in preservation of fruits and vegetables: A review,” MDPI, https://www.mdpi.com/2227-9717/11/6/1720.

[2] A. Hayes, “Stocks: What they are, main types, how they differ from bonds,” Investopedia, https://www.investopedia.com/terms/s/stock.asp.

[3] J. Fernando, “What is a commodity and understanding its role in the stock market,” Investopedia, https://www.investopedia.com/terms/c/commodity.asp.

[4] C. Mitchell, “Forex (FX): Definition, how to trade currencies, and examples,” Investopedia, https://www.investopedia.com/terms/f/forex.asp.

[5] A. Ganti, “What is a bid-ask spread, and how does it work in trading?,” Investopedia, https://www.investopedia.com/terms/b/bid-askspread.asp.

[6] M. J. Kramer, “What is a limit order in trading, and how does it work?,” Investopedia, https://www.investopedia.com/terms/l/limitorder.asp.

[7] M. J. Kramer, “Stop-loss orders: One way to limit losses and reduce risk,” Investopedia, https://www.investopedia.com/terms/s/stop-lossorder.asp.

[8] J. Fernando, “Margin and margin trading explained plus advantages and disadvantages,” Investopedia, https://www.investopedia.com/terms/m/margin.asp.

[9] A. Hayes, “Short selling: Definition, Pros, Cons, and examples,” Investopedia, https://www.investopedia.com/terms/s/shortselling.asp.

[10] A. Hayes, “Understanding liquidity and how to measure it,” Investopedia, https://www.investopedia.com/terms/l/liquidity.asp.

[11] A. Hayes, “Volatility: Meaning in finance and how it works with stocks,” Investopedia, https://www.investopedia.com/terms/v/volatility.asp.

[12] S. Seth, “Technical analysis for stocks: Beginners overview,” Investopedia, https://www.investopedia.com/articles/active-trading/102914/technical-analysis-strategies-beginners.asp.

[13] A. S. Gillis and D. Petersson, “What is supervised learning?: Definition from TechTarget,” TechTarget, https://www.techtarget.com/searchenterpriseai/definition/supervised-learning.

[14] A. Ross, “What is unsupervised learning?,” Unsupervised, https://unsupervised.com/resources/blogs/what-is-unsupervised-learning/.

[15] P. Potrimba, “What is semi-supervised learning? A guide for beginners.,” Roboflow, https://blog.roboflow.com/what-is-semi-supervised-learning/.
[16] P. Bajaj, “Reinforcement learning,” GeeksforGeeks, https://www.geeksforgeeks.org/what-is-reinforcement-learning/.
[17] K. Reyes, “What is Deep Learning and How Does It Works [Explained],” Simplilearn, https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-deep-learning.