MACHINE LEARNING APPLIED TO INTRA-DAY PRICE MOVEMENT PREDICTION OF MEXICAN STOCKS

A dissertation submitted to the University of Manchester
for the degree of Master of Science
in the Faculty of Humanities

2022

Student ID: 10582162

Alliance Manchester Business School

Contents

Word count 8602 words

List of Tables

List of Figures

Abstract

As computers have taken on a more relevant role in decision-making in financial industry, particularly in stock trading, prediction models have become more accurate. However, forecasting accurately it is not an easy task due to the speculative nature of stock markets. The use of machine learning techniques is an increasingly frequent topic among researchers and investors, and the efficiency of the models has been growing over time. In this research, four supervised learning models are used to solve a binary classification problem, in which the aim is predicting the direction of movement of a stock in the next minute, taking as predictive variables the history of 30 days, in a 1-minute time resolution, of price, volume of operations and a set of 11 technical indicators. The same methodology was repeated for four assets: an Exchange Traded Fund that replicates the main index of the Mexican stock market, and the three most liquid shares listed in that same country. Logistic Regression model with variable selection through LASSO method broke the highest result, of 72.8%, when considering Accuracy as performance measure, while the Decision Tree Classifier model, also with variable selection, broke the highest level, with 52.1%, under the F1-Score metric.

Declaration

No portion of the work referred to in the dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Intellectual Property Statement

i. The author of this dissertation (including any appendices and/or schedules to this dissertation) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this dissertation, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has entered into. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the dissertation, for example graphs and tables ("Reproductions"), which may be described in this dissertation, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialisation of this dissertation, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy, in any relevant Dissertation restriction declarations deposited in the University Library, and the University Library's regulations.

Chapter 1

Introduction

1.1 Background

Algorithmic trading, understood as the activity where a computer program follows a defined set of instructions to place a trade in stock markets, is experimenting a massive adoption, and as it grows, the development of predictive models with greater precision becomes essential for financial firms. Nowadays, up to 7 out of each 10 dollars of the volume registered in the United States stock markets are carried out by a computer (Coherent Market Insights, 2019), and global trends are similar. In fact, in Europe and U.S., 10% of hedge funds used algorithms to trade over 80% of their value in 2020, according to fact-checked statistics by analysing Alpha website (Smigel, 2022).

In light of this, artificial intelligence tools, such as machine learning models, are becoming more popular and sophisticated, as they can help to make decisions at every step of the trading process, from stock picking to execution of orders. Among all its possible uses, stock price prediction is one of the most important and the ultimate goal of this research. Producing accurate results that contribute to profitable strategies is a huge challenge due to the volatile nature of stock markets. For that reason, exploring new techniques and methodologies to estimate the future behavior of this and other types of financial assets has become a topic of interest for both investors and researchers alike.

The two most common approaches used to forecast a stock's price are fundamental and technical analysis. In the first, investors use all the information available about a company, such as financial statements, news, competitive advantages, management

profiles, and sector data, among others, to judge the essential value of its stock. In the other hand, technical analysis is a methodological framework of analysing the historical evolution of financial assets' prices and inferring from this assessment future predictions (Tsinaslanidis, 2016).

Given the nature of the information used for fundamental analysis, that method is more useful for investors with medium and long-term investment horizons, while the changing dynamics of technical indicators make its analysis a better tool for short-term operations, such as intra-day trading. This last type of operations is the one that we are going to study in this dissertation.

Intra-day trading, also known as day trading, is described by Bulkowski (2013b) as the action of buying and selling securities during a session in which no open positions —stock in possession— remain when the session completes. The reason why day traders don't hold stocks overnight —between sessions— is usually to avoid risks and negative price gaps between one day's close and the next day's price at the open.

Even when day traders sell all their stocks before the day ends, the time frame in which each trader operates can be very different. Some trades can be executed in hours, but also in milli-seconds, depending on how long each trader estimates that a trend can last in a specific stock. In any case, direct-access day trading software is often needed to provide real-time data and execute strategies.

Despite the variety of time frames used by day traders and long-term traders, the use of machine learning to make predictions is quite similar, as in most of the cases they will predict the future price of a stock or direction of movement in the next unit of time, as we are doing it in this research.

Although we have been referring to day trading as a stock market activity, it is important to clarify that day trading applies to operations within the same day of all types of financial assets, that is, those investment instruments that are traded on public exchanges, such as bonds, derivatives, and currencies, among others.

1.2 Problem description

This research aims to solve with the highest possible accuracy a binary classification problem, which consists in predicting the direction of next-minute price movement

of a given stock by using technical indicators —calculated from its price and volume data— as features for a list of machine learning models.

Most of previous research founded on this topic (Henrique et al., 2019) is focused on make predictions about stocks and/or indexes from United States, but for originality, the assets analysed in this dissertation are from Mexico. The first one is an Exchange Traded Fund that replicates the main index of Mexican Stock Exchange, S&P/BMV IPC, while the other three are the stocks with heaviest weight on that index: América Móvil, Walmart de México, and Grupo Financiero Banorte. These stocks are also some of the most liquid stocks in that country.

To achieve our goal, it was necessary to divide the entire process into three stages, which are data extraction, calculation of technical indicators, and finally, implementation of machine learning models.

For real-world data extraction stage was used a specialized library for Python that allows to retrieve information from Yahoo! Finance[1], which is a platform that was launched in 1997 and is part of Yahoo! network.

For the second stage, a set of 11 technical indicators are calculated. This set is best suited for intra-day trading and gives the best features or predictor through which the accuracy can be maximized, according to Kumar and Haider (2021). The most known of them are simple moving average (SMA), Stochastic oscillator, and Relative Strength Index (RSI), but also Money Flow Index (MFI), Rate of change (ROC), and Average true range (ATR), among others.

Finally, the predictive stage was divided into two sub-stages. The first consists of a feature selection that allows us to optimize the subset of features that we are going to use. The second stage is the implementation of four machine learning models, which are Logistic Regression (LR), Support Vector Machines (SVM), XGBoostClassifier (XBC), and Decision Tree Classifier (DTC).

[1] https://finance.yahoo.com/

Chapter 2

Literature review

Despite the release and wide adoption of Efficient Markets Hypothesis (EMH) in 1970s (Fama, 1970), which asserts in essence that financial markets follow a random path and therefore are unpredictable, forecast models were constituting a whole branch of study with extensive literature and rapid advances in its results.

Over the years, evidence contrary to the EMH was found, summarized by Kumar et al. (2016), Malkiel (2003) and even by its own author Fama (1991), but what remains in its place is the non-stationary nature of financial markets behaviour, and the fact that they are chaotic, noisy and non-linear, and are influenced by general economy, characteristics of each industry, politics and even the psychology of investors. In the light of that, it has become more common to test different approaches and techniques to increase the accuracy of predictions.

As was mentioned in the introduction of this research, there is a variety of approaches for financial market forecast. One of them is technical analysis (TA) (Achelis, 2001), which consists in mathematical calculations based on market data, such as stock price and volume, that help to identify trend and momentum of financial assets. In the other hand is fundamental analysis (FA), which focuses on economic trends, public sentiments, financial statements and assets reported by companies, among other sources of information (Bulkowski, 2013a) to determine the future value of the stock. In general, TA is widely used for short-term trading, while FA is more adopted by long-term investors.

2.1 Intra-day trading

For this research, two important references were found, since both are focused on intra-day operations, that is, those in which a financial asset are bought and sold during the same session.

The first study was authored by Paspanthong et al. (2019), who analysed a time series that includes basic variables and technical indicators with information every minute, and then the effectiveness of nine machine learning models is compared to be able to predict the direction of movement in the next minute. As a final step, the output is used in a trading strategy to analyse how effective are these signals to generate profits.

To do this, authors acquired from specialized trading site IEX Trading a dataset that covers a period of almost three months, with which they trained their machine learning models. Subsequently, they calculated 48 different technical indicators, which were reduced to 14 after using a Lasso regularization method for feature selection. Finally, they used a logistic regression model as a base, to later fit different versions of the Support Vector Machine, Recurrent Neural Networks (RNN), and Convolutional Neural Networks (CNN) models. The accuracy of their models in the test sets fell in a range between 48.89% and 54.49%.

The second widely used reference for this research is the study carried out by Kumar and Haider (2021), where several datasets with a set of technical indicators were analysed to predict the direction of movement of the asset in the next minute, but in a different way. They performed a mixture of models that included feature selection through a recursive model with three different machine learning methods, to subsequently adjust a neural network that increased the accuracy of predictions.

Data extraction in that study was done from Google Finance, a source that is no longer available, and considered the shares of 18 different companies from different stock exchanges. Subsequently, a new representation of data was made using 10-minute windows to obtain more information on possible behavior patterns. Once the new representation was obtained, 14 technical indicators were calculated, which were then selected with a Recursive Feature Elimination (RFE) method with different models, to finally adjust deep learning models. Accuracy ranged from 45% to 81%.

Table 2.1: References of literature about machine learning models applied to prediction for intra-day trading.

Reference	**Market/s**	**Asset/s**	**Predictive variable/s**	**Prediction/s**	**Main method/s**	**Performance Measure/s**
Kumar and Haider (2021)	Multiple	Stocks	TA	Direction	LSTM, LRC, SVM, DT	F1-Score
Li et al. (2014)	China	Stocks	News	Price	SVR	Return
Ghosh et al. (2022)	United States	Index	Prices	Direction	NN	Return
Naik and Mohan (2019)	India	Index	TA	Direction	NN	Accuracy
Kong et al. (2021)	China	Stocks	TA, liquidity measures	Jumps	NN	Accuracy, F1-Score
Labiad et al. (2016)	Morocco	Stocks	TA	Price	NN, Gradient Boosted Trees	Accuracy
Sun et al. (2019)	United States	Index	TA, prices	Shocks	Time Series, NN	Accuracy
Kumar and Haider (2020)	Multiple	Stocks	TA	Price	NN	Accuracy
Labiad et al. (2019)	Morocco	Stocks	TA	Direction	NN	Mean Absolute Percentage Error
Tanaka-Yamawaki and Tokuoka (2007)	United States	Stocks	TA	Direction	Genetic Algorithm	Accuracy
Taroon et al. (2020)	United States	Index	TA	Direction	Neural Network	Accuracy, F1-Score
Dutta et al. (2020)	India	Stocks	News, prices	Direction	SVM, RF, NN	Accuracy
Paspanthong et al. (2019)	United States	Index	TA	Direction	SVM, NN	Accuracy, Profit

TA: Technical Analysis, LSTM: Long-Short Term Memory, LRC: Logistic Regression Classifier, SVM: Support Vector Machine, DT: Decision Tree Classifier, SVR: Support Vector Regression, NN: Neural Networks, and RF: Random Forest

In addition to these two researches, at least a dozen extra studies were found that seek to carry out intra-day trading (Table 2.1), but with differences in several aspects, for example, the extension of temporary windows to forecast, predictive variables used, performance metrics and analysed assets.

Regarding predictive variables, the works of Dutta et al. (2020) and Li et al. (2014) stand out after they incorporate the publication of news to improve the accuracy. The first focuses on Chinese stock markets, while the second aims to analyse Indian markets.

On the other hand, when all the articles are grouped by type of variable to be predicted, most focus on directional movements, as in our investigation, but two particular cases stand out. They are the studies authored by Kong et al. (2021) and Sun et al. (2019), which seek to predict big jumps or "shocks" under volatility measures. The first study is focused on Chinese markets, while the second in U.S. Markets.

Whatever the classification is, the vast majority of authors highlight in their research how complicated forecasting can be in financial markets, given the randomness described in Fama's research (Fama, 1970).

2.2 Long-term trading

The aforementioned studies have foundations in previous literature used for long-term operations, and we considered relevant to mention them as they use data with the same structure as the one we utilised. This is opening price, closing price, high, low and volume of operation, but instead of getting it every minute, it is obtained for each complete trading session.

The oldest literature found dates back to early 1990s, a time that coincides with the findings of Henrique et al. (2019), who conducted a literature review on this subject using an automated bibliometric analysis system. According to his research, over the years, the number of articles that addressed this matter increased considerably, going from one article in 1991 to almost 70 publications in 2017, which was the last full year analysed by the author.

In that research, several classifications were made with 547 articles found. The most important was selecting only those publications whose final objective is predict

stock market behaviour, and eliminate those that only mention or quote an article about it. Under that condition, 57 articles were selected, and then sorted by country of stock market analysed, type of asset, type of predictive variables, variable to be predicted, machine learning models and performance metrics.

Among the most relevant results are that almost half of the studies used North American data (approximately 47%), and one sixth of them (approximately 17%) referred to data from Taiwan. Regarding the variables used as inputs in the models of financial market prediction, the TA indicators were the most popular, as they were used in approximately 37% of the studies, followed by fundamental information, which was used in 26%.

A last interesting result was that, regarding the predictive method, approximately 70% of the studies used at least some type of neural network; therefore, it was the classification method most used. The second was Support Vector Machine or Support Vector Regression, a more recent approach than neural networks, which was used in approximately 37% of the articles reviewed.

Chapter 3

Methodology

3.1 Data

Given the nature of the problem we are trying to solve, we need to get intra-day data, that means a time series that has as many days of operation as it is possible, with a time resolution of 1-minute. Naturally, it will be necessary to get one dataset for each analysed stock.

There are five necessary values on each time frame of 1- minute: open price, close price, maximum, minimum, and volume —number of traded shares—. The first four values are particularly useful because they help to create candlestick charts (Fig. 3.1), which are widely used by intra-day traders and will be shown in results chapter of this research, but also because many of the technical indicators use this information. On that type of charts, white candlesticks represent a period with a positive variation,

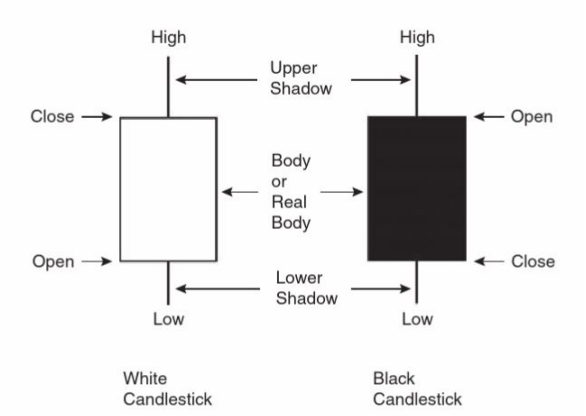

Figure 3.1: Candlestick chart (Bulkowski, 2008).

Table 3.1: Number of data points to train machine learning models by stock.

Stock	Number of data points
EWW	7,471
AMXL.MX	7,507
GFNORTEO.MX	7,345
WALMEX.MX	7,278

while blacks are for negative moves. The relevance to get the volume is to have a metric about the liquidity in the market.

As was mentioned in the introduction of this research, the stocks to be analysed are focused in Mexico. The first of them is an Exchange Traded Fund (ETF) listed with the code EWW in the New York Stock Exchange and that replicates the behaviour of the S&P/BMV IPC Index, the main Mexican index. Even when that asset is also listed in Mexico, the volume traded is much higher in the United States, and that is why we picked it. The other three are the biggest companies listed in Mexico in terms of market capitalization[1], which are also the most traded: América Móvil, with code AMXL.MX; Walmart de México, WALMEX.MX, and Grupo Financiero Banorte, GFNORTEO.MX. The period of data extracted was from August 4th to September 1st of 2022, and the size of datasets to train the machine learning models is described in Table 3.1

3.1.1 Data extraction

For this part of the process was used Yahoo! Finance as source. The procedure consisted in retrieving information from the platform through a Python library called yfinance[2], and save it as a CSV file with six columns, including the timestamp, open price, close price, maximum, minimum, and volume. The platform allows to pull 30 days of history, which means a bit more than 7,000 data points per stock.

Yahoo! Finance[3] is a media property that was launched in 1997 and is part of Yahoo! network. It provides financial news, data and commentary, including stock quotes, press releases, financial reports, and original content.

Other options were considered at this stage, such as The Bloomberg Terminal,

[1] *Market capitalization is calculated multiplying the outstanding number of stocks by its price.*
[2] https://pypi.org/project/yfinance/
[3] https://finance.yahoo.com/

because of its reliability. Nevertheless, it only allows to retrieve a history of 12 days, which is considerably less than what Yahoo! Finance offers.

The Bloomberg Terminal is a computer software system provided by financial data vendor Bloomberg L.P. launched in 1982 that enables professionals in financial service sector and other industries to access Bloomberg Professional Services through which users can monitor and analyse real-time financial market data and place trades on an electronic trading platform.[4]

Additionally, although The Bloomberg Terminal also has an Application Programming Interface (API) that allows data to be downloaded automatically, it is necessary to have a professional account to access this service, so it was not possible to access it that way for this research.

3.1.2 Data processing and technical indicators

This stage consists in preparing datasets that will be used by our machine learning models. To achieve this is necessary to calculate a selected set of technical indicators. These are numerical features based on market data that are useful to identify market behavior, such as trend and momentum.

To do this calculations, was used the library TA for Python (López Padial, 2018), which includes 84 different technical indicators, but we didn't use all of them. Instead, our selection were the following 11 indicators that are best suited for intra-day trading, according to Kumar and Haider (2021).

1. *Simple moving average (SMA)* is used to determine the trend of the market. In our study, we are computing the SMA in the time i, by finding the mean of Closing Prices (CP) in the last 14 minutes, i.e. $n = 14$.

$$\text{SMA}_i = \frac{CP_i + CP_{i-1} + CP_{i-2} + \cdots + CP_{i-n}}{n} \tag{3.1}$$

2. *Stochastic oscillator* is a momentum indicator that shows the relationship between interval close, to the interval highest and interval lowest price present in the chosen window size.

$$\%K = \frac{\text{(Current\_close-Min\_Low)}}{\text{(Max\_High-Min\_Low)}} \cdot 100, \tag{3.2}$$

[4] https://www.bloomberg.com/professional/

$$\%D = 3 \text{ - period of } \%K \tag{3.3}$$

where Min_low and Max_High are minimum and maximum for one period, and Current close is the Interval Close Price (ICP) of the 14th interval.

3. *Intra-day Momentum Index (IMI)* is a momentum indicator that shows the relationship between close price and open price. In our proposed work, the IMI is calculated using ICP of the previous interval and Interval Open Price (IOP) of the current interval for one period to find the gain and losses.

$$IMI = 100 \cdot \frac{\text{(Gains)}}{\text{(Gains+Losses)}} \tag{3.4}$$

4. *Relative Strength Index (RSI)* is a momentum indicator that measures the speed and change in price movements. In our study, we have calculated RSI using two consecutive intervals close price for one period to find the gain and losses.

$$RS = \frac{\text{Average Gain}}{\text{Average Loss}} \tag{3.5}$$

$$RSI = 100 - \frac{100}{1 + RS} \tag{3.6}$$

5. *Money Flow Index (MFI)* is an oscillator that was developed by Gene Quong and Avrum Soudack (Ghobadi, 2015) to measure the selling and buying pressure using both the stock price and volume.

$$MFR = \frac{\text{Positive Money Flow}}{\text{Negative Money Flow}} \tag{3.7}$$

$$MFI = 100 - \frac{100}{1 + MFR} \tag{3.8}$$

where typical price is the mean of the maximum, minimum and closing prices in the interval; money flow is the multiplication of typical price and volume; and positive money flow and negative money flow are the sum of gains and losses for the n interval present in the window.

6. *Rate of change (ROC)* indicator is a pure momentum oscillator that is used to measures the percent change in price between two intervals. In our study, we have computed ROC taking the difference between two intervals as 13.

$$ROC(5) = \frac{ICP_i - ICP_{i\text{-}13}}{ICP_{i\text{-}13}} \cdot 100 \tag{3.9}$$

where ICP_i = interval close price of current interval and $ICP_{i\text{-}13}$ = interval close price of the previous 13th interval present in the window.

7. *Commodity Channel Index (CCI)* is a technical indicator used to identify the market trend by visualizing the overbought and oversold conditions.

$$\text{CCI} = \frac{(\text{TP} - n\text{interval SMA of TP}])}{(\text{constant} \cdot \text{SD}} \cdot 100 \tag{3.10}$$

where typical price (TP) is the mean of the maximum, minimum and closing prices; constant = 0.015; and SD = standard deviation of typical price, n = 14.

8. *Force Index (FI)* is an indicator that uses price and volume to identify strong reasons or strength behind the change in market behavior with weighting (W) = 0.1818

$$RFI(i) = (ICP_i - ICP_{i\text{-}1}) * Volume, \tag{3.11}$$

$$FI = (RFI(14) - SMA_{13}) \cdot W + SMA_{14} \tag{3.12}$$

9. *Average true range (ATR)* is an indicator that uses the price volume to measure the volatility. This technical indicator is used to identify the gaps between the consecutive IOP such that it is above or below its maximum allowed limits for the session.

$$TR = [\max(IH, PC) - \min(IL, PC)] \tag{3.13}$$

$$ATR = \frac{TR}{n} \tag{3.14}$$

where IH = Interval High, IL = Interval low, and PC = Interval close price of previous interval.

10. *Directional Movement Index (DX)* is an indicator used to determine the directional trend and strength of the market. It is calculated by doing the technical analysis of both negative directional indicator (DI^-) and positive directional indicator (DI^+) that was proposed by Welles Wilder(Wilder, 1978).

$$DX = \frac{(DI^+) - (DI^-)}{(DI+) + (DI^-)} \cdot 100 \tag{3.15}$$

11. *Exponential moving average (EMA)* is an indicator used to determine the trend of the market with the weighting factor (W) (W = [2/ (selected interval + 1)]) to the most recent data points.

$$EMA_i = [(ICP_i - EMA_{(i-1)}) \cdot W] + EMA_{(i-1)} \tag{3.16}$$

Other authors used different sets of technical indicators, as Paspanthong et al. (2019), whose study used Simple Moving Average (SMA), Exponential Moving Average (EMA), Crossovers, consecutive price trends, with 5, 10, 12, 20, 26, 50, 100, 200 days lookback window, as features. However, the set used by Kumar and Haider (2021) achieved better results.

3.2 Feature Selection

Each phenomenon described in data can have many attributes. Census, for example, describes a population in several ways, such as age, gender, ethnic group, education, where they live, who they live with, among others, but not all attributes are important to study a specific topic. For instance, gender and age are relevant variables to study diseases, but political affiliation it is not important for that purpose.

In machine learning, attributes of data are also called features, and they are classified as relevant or irrelevant depending on what we are trying to predict. The process of selecting relevant features from a given set is called *feature selection*, and it is an important step of data processing that needs to be done before training the learners (Zhou, 2021).

There are two main reasons to do feature selection. Firstly, the "curse of dimensionality" is a common issue in practical learning problems due to the large number of features. If we can identify the relevant features, then the subsequent learning process will deal with a much lower dimensionality. From this point of view, feature selection shares a similar motivation with dimensionality reduction. Secondly, eliminating irrelevant features often reduces the difficulty of learning, because the learner is more likely to discover the truth without being distracted by irrelevant information.

It is important to note that during this process there is a risk of losing relevant information, and therefore affect its accuracy, although, as mentioned above, each process has different relevant variables. Another way to categorize features of a dataset is to determine if they are redundant, which is information that is relevant, but its dynamics is already included in another attribute. Zhou (2021) exemplifies this type of variable with the process of measuring the volume of a cube. If we have the height, we don't need to know the length or width, because we already know that information by definition of cube.

In this process, the key question is how to select the best possible subset of features. The answer becomes even more complex if you do not have prior knowledge of the subject studied. Although there is no single or optimal answer, the general idea is to generate subsets of variables and evaluate their performance under different metrics, so we can choose the ones that best suits for our purposes.

Visalakshi and Radha (2014) describes two ways to categorize feature evaluation: *Filter* and *Wrapper*. The main objective of evaluation is to compute the selective capability of a feature or subset to make out the class label.

Filter methods relies on general uniqueness of data to be evaluated and pick features, or subsets of features, without the help of any mining algorithm. The main advantage of using feature ranking is that it is computationally less complex and avoids over-fitting problems. Some of the ranking methods are Mutual Information, Pearson correlation criteria, Chi-square test, and Correlation coefficients.

On the other hand, there are the wrapper methods, which need mining algorithms and use their performance as evaluation criteria. These methods search for features which are suited for mining algorithm and aims to improve the mining performance. Generally speaking, wrapper methods are usually better than filter methods in terms

of the learner's final performance since the feature selection is optimized for the given learner.

Zhou (2021) mentioned a third type of feature evaluation methods called embedded methods, which unify the feature selection process and the learner training process into a joint optimization process; that is, the features are automatically selected during the training. Regularization is part of this methods and exist two ways of do it, by using L_1 regularization or L_2. The first is called Ridge, while the second is called Lasso (Tibshirani, 1996), and is the method that we will use for this research.

3.2.1 Lasso

Lasso (Least Absolute Shrinkage and Selection Operator) is a regression analysis method that performs both feature selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model. It was presented by Tibshirani (1996), a researcher from Stanford University.

In general terms, Lasso method minimizes the residual sum of squares subject to the sum of the absolute value of the coefficients being less than a constant. It is stated as follows.

Given a linear regression with standardized predictors x_{ij} and centred response values y_i for $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, p$, the Lasso solves the l_1-penalized regression problem of finding $\beta = \beta_j$ to minimize

$$\sum_{n=1}^{N} \left(y_i - \sum_j x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{3.17}$$

This is equivalent to minimizing the sum of squares with a constraint of the form $\sum \beta_j \leq s$. It is similar to ridge regression, which has constraint $\sum_j \beta_j^2 \leq t$. Because of the form of the l_1-penalty, the Lasso does feature selection and shrinkage, whereas ridge regression, in contrast, only shrinks. If we consider a more general penalty of the form $(\sum_{j=1}^{p} \beta_j^q)^{1/q}$, then the lasso uses $q = 1$ and ridge regression has $q = 2$. Subset selection emerges as $q \to 0$, and the lasso uses the smallest value of q (i.e. closest to subset selection) that yields a convex problem. Convexity is very attractive for computational purposes (Tibshirani, 2011).

To utilise this feature selection method in our research we used scikit-learn library

for Python[5]. Before that, our data was scaled used *StandardScaler* method of the same library, which standardize features by removing the mean and scaling to unit variance. This process is necessary for some of machine learning models used.

3.3 Machine learning module

Once the optimal attributes have been selected under Lasso regularization method, we can implement four supervised machine learning models to our datasets. Three of them were used by Kumar and Haider (2021). To complete this process, we used Python language, and specifically XGBoost (Chen and Guestrin, 2016) and Scikit-learn (Pedregosa et al., 2011) libraries. We will use four different models which are described below.

3.3.1 Model training

1. Gradient Boosting Classifier (XGBClassifier)

   XGBoost is an implementation of gradient boosted decision tree algorithm designed for speed and performance (Brownlee, 2021) and stands for "Extreme Gradient Boosting", where the term "Gradient Boosting" originates from the paper *Greedy Function Approximation: A Gradient Boosting Machine* (Friedman, 2001).

   The tree ensemble model consists of a set of classification and regression trees (CART). Mathematically, we can write our model in the form

   $$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), (f_k) \in \mathcal{F} \tag{3.18}$$

   where K is the number of trees, f_k is a function in the functional space $\mathcal{F}$, and $\mathcal{F}$ is the set of all possible CARTs. The objective function to be optimized is given by

   $$\text{obj}(\theta) = \sum_{i}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \omega(f_k) \tag{3.19}$$

[5] https://scikit-learn.org/stable/

where $\omega(f_k)$ is the complexity of the tree f_k.

2. Logistic Regression Classifier (LRC)

   LRC, first introduced by Cox (1958), is a supervised machine learning classification algorithm mainly used for binary classification and the multinomial logistic regression model for multiple classification. Logistic regression is like a linear classifier that uses the sigmoid function for binary classification and softmax function for multiple classifications to find the probability of the output with their predicted class (Brownlee, 2020). This classification algorithm takes the real-valued as an input and makes a prediction with two possible outcomes of the default class 0 or class 1 by finding the probabilistic value of the input. If the probability is ($P < 0.5$) then it implies that the output as a prediction belongs to the class 1, otherwise if the probability is ($P > 0.5$) the prediction is for default class 0.

$$\text{Output} = b_0 + (b_1 \cdot x_1) + (b_2 \cdot x_2) + \ + (b_n \cdot x_n) = 0, \tag{3.20}$$

   where x_1, x_2, x_n is input vector X and $b_0, b_1, b2, \ldots, b_n$ are coefficient based on training data. The probability of the output for the binary class is calculated by using a sigmoid function where e is the constant Euler number.

$$P(\text{default class } 0) = \frac{1}{(1 + e^{(-\text{Output})})} \tag{3.21}$$

3. Decision Tree Classifier (DT)

   DT is a supervised machine learning algorithm used for solving classification problems (Wu et al., 2008). It is used to create a training model to do the prediction for the class of the target variables with a set of learning rules generated by training data. In other words, it is an algorithm that it is built to make decisions, similarly to how humans make decisions. (Bento, 2021) This algorithm solves the problem by using the tree-like structure for attribute selection with the best attribute as a root node and each internal node as an attribute and leaf node as the target value. The model use by default the Gini index approach to do the attribute selection and tree representation. It finds the best

attribute as the root node and further splits the training data in subsets. The subset is selected based on target variables, training data with labeled class 0 is present in one subset and labeled with class 1 is present in another class. Again, the best attribute is selected, and training data is splitted into subsets, likewise the process is repeated until the leaf node in all the branches of the tree is not achieved.

$$\text{Gini Index} = 1 - \sum_{i=1}^{n} (p_i)^2, \tag{3.22}$$

where n is the number of classes and p_i is the probability of each class present in training data.

4. Support Vector Machine Classifier (SVM)

   SVM is a supervised machine learning technique that is mainly used for classification problems (Cortes and Vapnik, 1995). In this algorithm, the input vector with n number of features is being plotted in the form of n-dimensional space. Furthermore, we perform the classification by finding the optimal hyper-plane line using the maximal margin approach that splits the input variable space in such a way that the hyper-plane line best separates the points in the input variable space by their class, either class 0 or class 1. The formula used to find the hyperplane line is shown below:

$$b_0 + (b_1 \cdot x_1) + (b_2 \cdot x_2) + \ + (b_n \cdot x_n) = 0, \tag{3.23}$$

   where x_1, x_2, x_n is the number of features, coefficients b_0, b_1, b_2, b_n that are used to calculate the slope of the line, and the intercept $b0$ are found by the learning algorithm. Now the classification can be performed by using the above line equation with the new input values. If the new point value returned by the equation is greater than 0, it belongs to class 0 and lies above the line. Similarly, if the new point value is less than 0, it belongs to class 1 and lies below the line.

3.3.2 Model evaluation

To evaluate the performance of the models, we have used three metrics: accuracy, cross validation score, and F1-score. These metrics are calculated with the help of the confusion matrix (Table 3.2).

Table 3.2: Confusion Matrix

Parameters	Predicted class: Yes	Predicted class: No
Actual Class: Yes	TP	FN
Actual Class: No	FP	TN

1. *Accuracy* is the ratio between all the correct predictions and all the data points, and it is calculated as

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{3.24}$$

2. *Cross validation score* is the accuracy calculated on each fold when the training set is divided in k folds to avoid over-fitting.

3. *F1-Score* is the hybrid measure of both precision and recall weighted averages. F1-score calculation takes both the values of FP and FN to find the model performance as an alternative to the accuracy.

$$\text{F1-Score} = 2 \cdot \frac{(\text{Precision} \cdot \text{Recall})}{(\text{Precision} + \text{Recall})} \text{ , where} \tag{3.25}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \text{ , and} \tag{3.26}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.27}$$

F1-Score is described in Pedregosa et al. (2011) as a harmonic mean of the precision and recall, where an F1-Score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1-Score are equal.

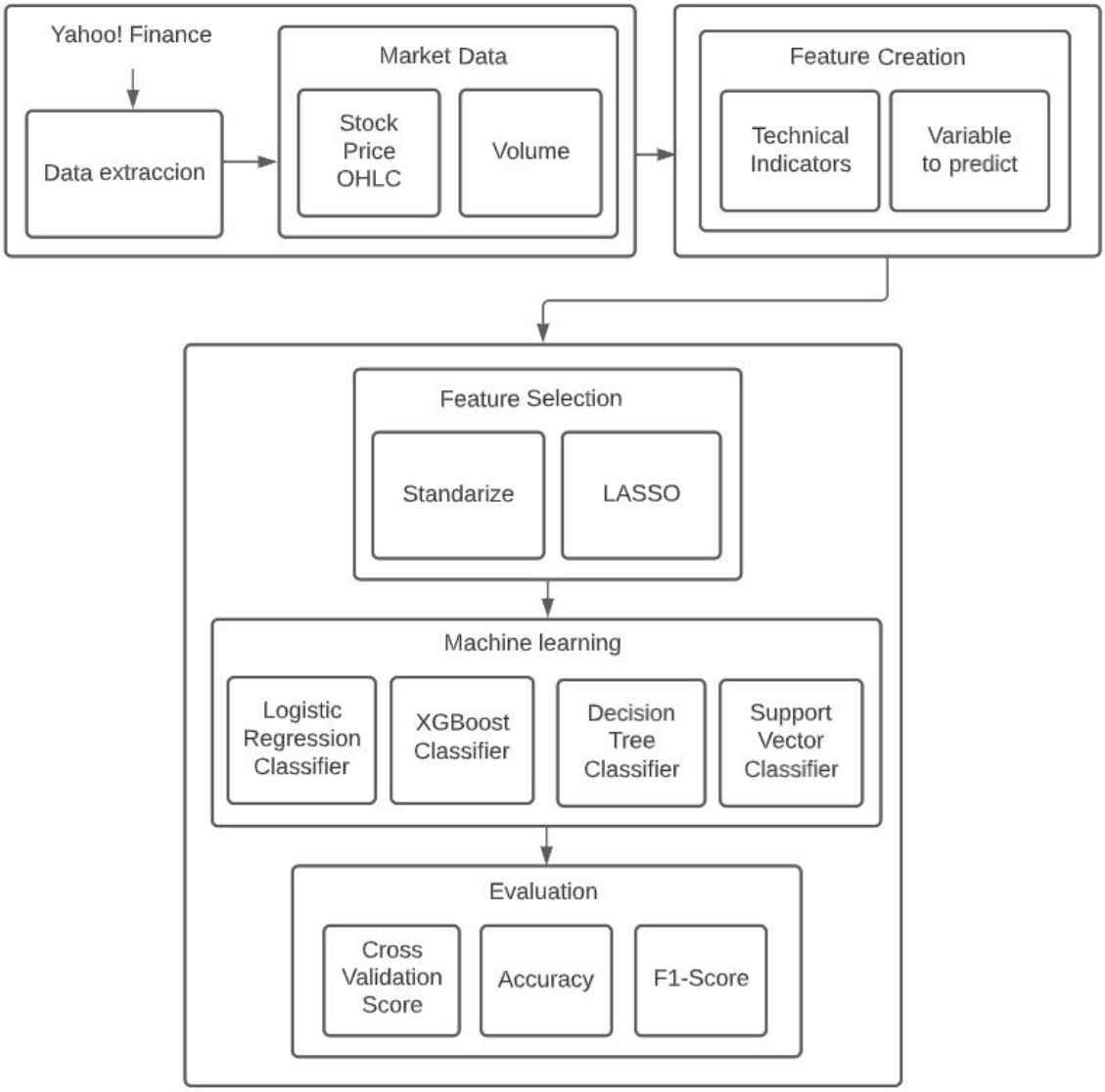

Figure 3.2: Proposed framework.

3.4 Proposed framework

Although the entire process is made up of several tasks, as we have described in previous sections, our Python code (See Appendix A) was built in three modules, one for data extraction, another for processing and calculating technical indicators, and third one to adjust machine learning models, make predictions and evaluate models. The framework is graphically described in Figure 3.2.

Chapter 4

Experimental results and analysis

In this section we will describe and analyse from different angles the results of 32 combinations derived from applying four different machine learning models to four stocks under two possible scenarios, which are with or without feature selection.

4.1 Performance by stock

4.1.1 EWW

We will start with results about the ETF that replicates the main index of Mexican stock market, which has the ticker EWW, and whose behavior in the analysed period is described in Figure 4.1.

In the same way that Paspanthong et al. (2019), we will take the results of logistic regression model as a baseline, due to its simplicity, and then make comparisons with the other methods (See Table 4.1). The first thing we can notice is that precisely our base model obtained the highest accuracy, with 58.5% without feature selection and slightly higher, with 58.8%, with feature selection. As we mentioned in Section 3.2.1, the selection was performed using the LASSO method (Tibshirani, 1996).

Although the accuracy results are higher than those obtained by Paspanthong et al. (2019), it is important to note the poor result under the F1-Score metric, since we obtained 38.2% without feature selection and 37 .3% with feature selection. A low F1-Score result is informative in the way that it may mean that our data set is unbalanced towards one of the two classes to be predicted.

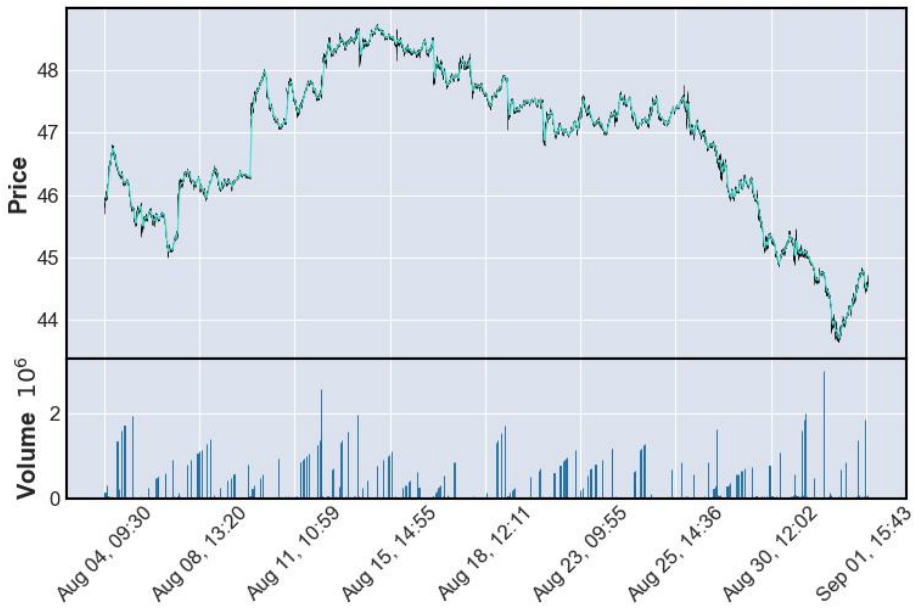

Figure 4.1: ETF replicating S&P/BMV IPC Index (EWW) Intra-day chart.

The second model with the best accuracy for EWW was the XGBClassifier, with 54.7% considering feature selection and 54.3% without feature selection. The DecisionTreeClassifier model had the third highest accuracy, with 53.7%, but the highest F1-Score (52.6%), while the SVC had the lowest accuracy, close to 50%.

Table 4.1: Performance metrics by model for EWW.

Feature selection	Model	Cross Val Score	Accuracy	F1-Score
No	XGBClassifier	54.3	54.3	**50.1**
No	LogisticRegression	58.5	**58.7**	38.2
No	DecisionTreeClassifier	52.6	50.7	49.4
No	SVC	50.8	50.2	48.7
Yes	XGBClassifier	54.2	54.7	47.7
Yes	LogisticRegression	**58.8**	58.4	37.3
Yes	DecisionTreeClassifier	51.0	53.7	52.1
Yes	SVC	52.8	50.9	49.2

4.1.2 AMXL.MX

The results obtained for this stock (See Table 4.2) were in general terms the best of our research, since with it we got a maximum accuracy of 72.8%, again with the Logistic Regression model with feature selection. Under the Cross Validation Score metric, the

result was similar, since we obtained 71.4% with the same model. Usually the first metric is slightly higher than the second, because in the first option occurs greater over-fitting. On the negative side, we could once again observe a very low F1-Score of 42.5%.

One explanation for a very high accuracy and a very low F1-Score is that the variable to be predicted, that is, the direction of movement, which can take the values 1 or 0, is very unbalanced towards one of the two classes. This possibility is easy to verify in Figure 4.2, in which there are notoriously more negative movements from the middle of the graph to the end. That means that this dataset contains more 0's than 1's.

Figure 4.2: América Móvil Stock (AMXL.MX) Intra-day chart.

The second best model for AMXL.MX was the XGBClassifier, which had an accuracy of 69.8% and an F1-Score of 47.5% with feature selection, but the combination is better in the model without feature selection, since we obtained an F1-Score for the first time above 50%, when marking 50.5%.

Finally, with very similar results, the SVC model without feature selection obtained an accuracy of 62.1% and a F1-Score of 51.9%, while the DecisionTreeClassifier with feature selection got 60.9% and 51. 8%, respectively.

Table 4.2: Performance metrics by model for AMXL.MX.

Feature selection	Model	Cross Val Score	Accuracy	F1-Score
No	XGBClassifier	68.7	69.6	50.5
No	LogisticRegression	**71.4**	72.6	42.5
No	DecisionTreeClassifier	61.3	60.7	**52.0**
No	SVC	59.3	62.1	51.9
Yes	XGBClassifier	67.3	69.8	47.5
Yes	LogisticRegression	71.4	**72.8**	42.4
Yes	DecisionTreeClassifier	59.4	60.9	51.8
Yes	SVC	59.8	60.7	50.4

4.1.3 GFNORTEO.MX

The most relevant result after analysing GFNORTEO.MX stock is that with it we got the highest F1-Score, of 54.4%, when adjusting an XGBClassifier model without feature selection, and for this reason we are going to focus on that metric in this section.

Figure 4.3: Grupo Financiero Banorte Stock (GFNORTEO.MX) Intra-day chart.

The second highest F1-Score (See table 4.3 was obtained with the DecisionTreeClassifier without feature selection, scoring 53.2%, which was slightly reduced to 51.7%, when performing feature selection.

Finally, the SVC model and the Logistic Regression obtained F1-Scores of 49.5%

and 43.7%, respectively, with feature selection, although the accuracy in the latter was close to 60%.

As was mentioned in the section 3.3.2, F1-Score can be interpreted as a harmonic mean of the precision and recall, which means this other two metrics are more balanced on this stock.

Table 4.3: Performance metrics by model for GFNORTEO.MX.

Feature selection	Model	Cross Val Score	Accuracy	F1-Score
No	XGBClassifier	54.2	56.5	**54.4**
No	LogisticRegression	56.8	57.8	43.7
No	DecisionTreeClassifier	53.7	54.1	53.2
No	SVC	50.7	48.7	47.5
Yes	XGBClassifier	55.0	53.6	49.7
Yes	LogisticRegression	**57.1**	**58.1**	43.7
Yes	DecisionTreeClassifier	52.0	52.4	51.7
Yes	SVC	51.4	50.7	49.5

4.1.4 WALMEX.MX

Before analysing the results obtained by our list of machine learning models with WALMEX.MX stock, it is relevant to note in Figure 4.4 that the performance of this asset, at least in the second half of the graph it has more defined trends, since a marked bearish bias is observed as of August 18th, and then in the last session it is possible to see an accelerated recovery. This suggested that trend technical indicators would show clearer signs of direction, and thus might work as better predictive variables, but the results were not noticeably better than other stocks.

Although an accuracy of 60.2% was obtained in the LogisticRegression model without Feature Selection (See Table 4.4), the F1-Score of that model was 44.9%, a not very outstanding result. Under this last metric, the highest score was 52.9%, with the SVC model without feature selection.

4.2 Performance by model

Under this analysis we are going to focus on tables 4.5 and 4.6, where the maximum, minimum and average precision and F1-Score obtained were established by each of the

Figure 4.4: Walmart de México Stock (WALMEX.MX) Intra-day chart.

Table 4.4: Performance metrics by model for WALMEX.MX.

Feature selection	Model	Cross Val Score	Accuracy	F1-Score
No	XGBClassifier	55.1	54.9	50.4
No	LogisticRegression	58.5	**60.2**	44.9
No	DecisionTreeClassifier	53.1	53.0	50.9
No	SVC	52.8	54.9	**52.9**
Yes	XGBClassifier	54.3	55.3	49.9
Yes	LogisticRegression	**58.6**	60.2	44.0
Yes	DecisionTreeClassifier	51.8	53.4	51.2
Yes	SVC	52.1	52.1	50.0

four machine learning models used.

In terms of accuracy, we were able to observe that LogisticRegression performed highest; however, as previously mentioned, the result could be influenced by an imbalance of categories in our datasets. In second place was the XGBClassifier model, while the third and fourth places were obtained by the DecisionTreeClassifier and the SVC, respectively.

From this table, we can see that the effect of feature selection is not consistent across all models, because while it did improve the LogisticRegression and DecisionTreeClassifier results, it worsened them for XGBClassifier and SVC.

Now we are going to focus on the table 4.6, where the F1-Score was registered.

Table 4.5: Accuracy by model sorted by average.

Model	Feature Selection	Max.	Min.	Average
LogisticRegression	Yes	72.8	58.1	62.4
LogisticRegression	No	72.6	57.8	62.3
XGBClassifier	No	69.6	54.3	58.8
XGBClassifier	Yes	69.8	53.6	58.3
DecisionTreeClassifier	Yes	60.9	52.4	55.1
DecisionTreeClassifier	No	60.7	50.7	54.6
SVC	No	62.1	48.7	54.0
SVC	Yes	60.7	50.7	53.6

There we can see that the highest average was obtained by the DecisionTreeClassifier, followed in order by the XGBClassifier, the SVC and Logistic Regression. From this table we can observe a bias on the effect of the selection of variables, since only the DecisionTreeClassifier improved, while the score was reduced in the other models when doing this procedure.

Table 4.6: F1-Score by model sorted by average.

Model	Feature Selection	Max.	Min.	Average
DecisionTreeClassifier	Yes	52.1	51.2	51.7
DecisionTreeClassifier	No	53.2	49.4	51.4
XGBClassifier	No	54.4	50.1	51.3
SVC	No	52.9	47.5	50.2
SVC	Yes	50.4	49.2	49.8
XGBClassifier	Yes	49.9	47.5	48.7
LogisticRegression	No	44.9	38.2	42.3
LogisticRegression	Yes	44.0	37.3	41.9

Finally, we can see in the table 4.7 that performing the feature selection process worse, on average, the performance metrics, since while the average precision of all our models was 57.4% and the F1-Score of 48.8% without feature selection, both metrics dropped to 57.3% and 48.0%, respectively, with Lasso feature selection. This doesn't mean that feature selection in general is bad, but maybe a different method could perform better, such as Recursive Feature Elimination in combination with some machine learning model.

Table 4.7: Average performance by feature selection

Feature Selection	Average Accuracy	Average F1-Score
No	57.4	48.8
Yes	57.3	48.0

4.3 Comparative results

The comparative results of this research with similar works are shown in tables 4.8 and 4.9. Although more research on this topic was found in the literature review, not all of them are comparable, due to the type of problem they present, the variable to be predicted, the predictive variables, or the evaluation metrics.

In terms of accuracy, we were able to observe that our research registered an average rate of 57.4%, which places it second among the four comparable works under this metric. We can see other notable differences between our investigation and the others presented in the table 4.8. One of the most important is the country analysed, because while the other three works focus on U.S. stocks, ours used Mexicans. Another relevant difference is the number of days, since our research used the least number of days of operation. Although not reported in the table, it is worth mentioning that all models used some method of feature selection.

Comparing by F1-Score, our research was not so favored, since it was below the other analysed research. The maximum level reached in this dissertation was 54.4%, while the Kumar and Haider (2021) work reached 75.0%, and Taroon et al. (2020) 64.0%.

Table 4.8: Comparative results considering Accuracy with existing work.

Author	Market	Models	Max.	Min.	Avr.	Stocks	Trading Days
Tanaka-Yamawaki and Tokuoka (2007)	United States	Genetic Algorithm	82.0	55.0	66.0	8	365
This research	**Mexico**	**LRC, XGBC, SVC, DT**	**72.8**	**48.7**	**57.4**	**4**	**30**
Paspanthong et al. (2019)	United States	SVM, Neural Networks	54.5	48.9	51.3	1	84
Taroon et al. (2020)	United States	Neural Network	52	49.1	50.9	1	84

LRC: Linear Regression Classifier, XGBC: XGBoost Classifier, SVC: Support Vector Classifier, DT: Decision Tree Classifier, SVM: Support Vector Machine.

Table 4.9: Comparative results considering F1-Score with existing work.

Author	Market	Models	Max.	Min.	Avr.	Stocks	Trading Days
Kumar and Haider (2021)	Multiple	LSTM, LRC, SVM, DT	75.0	55.0	69.0	8	365
Taroon et al. (2020)	United States	Neural Network	64.0	36.0	52.8	1	84
This research	**Mexico**	**LRC, XGBC, SVC, DT**	**54.4**	**37.3**	**48.4**	**4**	**30**

LSTM: Long-Short Term Memory, LRC: Linear Regression Classifier, SVM: Support Vector Machine, DT: Decision Tree Classifier, XGBC: XGBoost Classifier, SVC: Support Vector Classifier.

4.4 CPU Time Analysis

As the objective of this research is to make accurate predictions about direction of movement of a stock in the next minute, it is important that execution times are minimal, since a model that takes more than a minute to make a prediction it would be useless. For this reason, we decided to include the table 4.10 with the time that each model took to produce a result.

The software utilised in this research was a laptop with Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz Processor, with 16 GB RAM memory.

Under this analysis, the best performance was obtained by Logistic Regression model, followed by Decision Tree Classifier. The other two models took considerably more time to predict a result.

Table 4.10: CPU Time by model.

Model	Time (sec)
Logistic Regression	0.11868
Decision Tree Classifier	0.48077
XGBoost Classifier	4.70576
Support Vector Classifier	6.35350

Chapter 5

Conclusions & Future Work

We will begin this chapter by explaining the main contributions of this research, then listing our three main conclusions, and lastly we will review the stages of our process in which the experimentation could be deepened to obtain better results.

We believe the greatest contribution of this research lies in the country to which the analyzed stocks belong, as Mexico is not a particularly active market in terms of algorithmic trading. As there is greater interest from researchers and investors in this matter, and these techniques are more widely used, the Mexican market could register greater liquidity, which could arouse greater interest from companies to finance themselves in the stock market and contribute in some way to the development of the country.

Another contribution is the use of Gradient Boost Classifiers for this kind of predictions, since we found limited use of this model in our literature review, despite the fact that its efficiency has been proven in other types of problems. Perhaps this research could arouse the interest of more people in experimenting with this model to find a better combination of hyper-parameters and obtain better results.

Having said that, our first conclusion is about the performance of the studied models. Although the Logistic Regression model, whose purpose is specifically to solve classification problems, gives high Accuracy results, it also shows us a low performance under F1-Score, which suggests that it is not a model that can be trusted to be reliable or to be incorporated into a trading strategy, since in certain cases they turned out to be less efficient even than a random prediction. Therefore, if both performance metrics are combined into one, as a sum, the highest score was obtained by the XGBoost

Classifier without Feature Selection, obtaining an accuracy of 69.6% and a F1-Score of 50.5 %.

The second conclusion is that, despite the advances in computational power in the last 30 years, a period in which literature on the subject we are studying could be found, forecasting in financial markets continues to be an extremely difficult task due to its behavior is multifactorial, with a large random component, as enunciated by the Efficient Markets Hypothesis of Fama (1970). Therefore, different approaches to this problem have been tried. The first is to define the variable to predict, which can be the price of a stock in the future or the direction of movement that the asset will have, from which other performance metrics are derived, such as return or Sharpe ratio, among others.

Although not much was addressed on this topic during the body of this dissertation, the reflection on the advances in the predictive power of the models also poses a challenge in terms of the computational power to execute buy or sell orders at high speed. For this reason, it has happened that the competition between large asset managers is now for infrastructure, since increasingly sophisticated equipment and even locations closer to the data centers of the (Narang, 2013) exchange centers are required.

The third conclusion, obtained from the literature review, is that the ensembled models, also called multi-level classifiers, have a better performance, because after using machine learning models to select the variables, other predictive methods are adjusted, usually neural networks. This is the case of the model built by Kumar and Haider (2021), whose results compared to simple classifiers grew by up to 12 percentage points.

Regarding future work, there are different experiments that could be carried out at each stage of our process with the aim of increasing performance metrics.

Let's start with data extraction. As was mentioned in the methodology section of this dissertation, other sources were considered to retrieve market data. Bloomberg was one of them, however, the amount of information available was small with a student account. IEX turned out to be another possible source of information, although it did not offer any advantage over the one we used. In most platforms there are paid versions with which you can access a longer period of information, which would be

useful in terms of better training a machine learning model. Nonetheless, there is a trade-off between the amount of data and the speed of predictive models that has to be considered.

Another element that certainly lends itself for experimentation is the selection of stocks that can be analysed. In the case of the Mexican market, the options are limited, because many of the securities listed there are not very liquid, that is, they do not register purchase and sale operations every minute, which could result in inaccurate or inadequate predictions, that is, with outdated time horizons. In addition, the real options to implement an algorithmic trading model in Mexico are more limited, because while most brokers in developed markets offer APIs to operate them, in Mexico there are not many alternatives.

In the second module in our proposed framework (See Fig. 3.2), which we call feature creation, there are too many options that could lead to different results. We will first focus on the calculation of technical indicators, because as will be highlighted in the methodology section, the library used has 84 possible technical indicators. Of course, the number of subsets (2^{84}) is too large, so we could choose from several criteria to make a selection. One of them, for instance, is to select certain types, such as momentum indicators, or perhaps trend indicators, but another possible option is the one carried out by Paspanthong et al. (2019), in which the authors calculate different types of moving metrics and then analyse the crossovers between them, to determine buy or sell signals. This also raises an infinity of possibilities that could also lead to better results.

In this module, the time resolution for the variable to be predicted is also an element to experiment with. The time window used in this investigation is one minute, but intra-day operations can be carried out with different time windows, for example, every 10 or 15 minutes, like Kumar and Haider (2021) or Labiad et al. (2016) did. Additionally, the variable to predict itself can change to price or return, instead of being a binary problem that predicts only direction. In fact, it can be kept as a classification problem, but with multiple classes, instead of just two. For example, predict small and large upswings as well as small and large downswings.

Finally, the experimentation in third module could be presented in each sub-stage. First, in the feature selection section, the possibilities are just too many. Recursive

Feature Elimination is one possibility, which in turn could be combined with many different models. Each of them could also require a different form of standardization, which would give us the possibility of improving performance.

The next sub-stage is itself the subject of a separate analysis, as there are a wide variety of classification models for supervised learning. Each one of them also has a huge variety of hyper-parameters that can be modified to obtain the best result, so we won't see more detail in this section.

Finally, the evaluation metrics are also very relevant, and varied, since we could evaluate our model only with the results of the prediction, but we can also evaluate its effect on trading strategies. That is, how profitable a strategy is by using one prediction model or another. We consider this last option very relevant for a more advanced study, because at the end of the day, the objective of trading is to generate profits.

Bibliography

Achelis, S. B. (2001). *Technical analysis from A to Z : covers every trading tool . . . from the absolute breadth index to the zig zag*, 2nd ed. edn, McGraw-Hill, New York.

Bento, C. (2021). Decision tree classifier explained in real-life: Picking a vacation destination. Accessed on 2022-08-10.
https://towardsdatascience.com/decision-tree-classifier-explained-in-real-life-picking-a-vacation-destination-6226b2b60575

Brownlee, J. (2020). Logistic regression for machine learning. Accessed on 2022-08-10.
https://machinelearningmastery.com/logistic-regression-for-machine-learning/

Brownlee, J. (2021). A gentle introduction to xgboost for applied machine learning. Accessed on 2022-08-10.
https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

Bulkowski, T. N. (2008). *Encyclopedia of candlestick charts*, J. Wiley & Sons, Hoboken, N.J.

Bulkowski, T. N. (2013a). *Fundamental analysis and position trading : evolution of a trader*, Wiley trading series, WILEY, Hoboken.

Bulkowski, T. N. (2013b). *Swing and day trading : evolution of a trader*, Wiley, Hoboken, N.J.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery*

and Data Mining, KDD '16, ACM, New York, NY, USA, pp. 785–794. http://doi.acm.org/10.1145/2939672.2939785

Coherent Market Insights (2019). Global algorithmic trading market to surpass us $ 21,685.53 million by 2026, *Technical report*. Accesed on 2022-10-08. https://www.bloomberg.com/press-releases/2019-02-05/global-algorithmic-trading-market-to-surpass-us-21-685-53-million-by-2026

Cortes, C. and Vapnik, V. (1995). Support-vector networks, *Machine learning* **20**(3): 273–297.

Cox, D. R. (1958). The regression analysis of binary sequences, *Journal of the Royal Statistical Society: Series B (Methodological)* **20**(2): 215–232.

Dutta, A., Pooja, G., Jain, N., Panda, R. R. and Nagwani, N. K. (2020). A hybrid deep learning approach for stock price prediction, *Machine Learning for Predictive Analysis*, Lecture Notes in Networks and Systems, Springer Singapore, Singapore, pp. 1–10.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work, *The Journal of finance (New York)* **25**(2): 383–417.

Fama, E. F. (1991). Efficient capital markets: II, *The Journal of finance (New York)* **46**(5): 1575–1617.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine, *The Annals of statistics* **29**(5): 1189–1232.

Ghobadi, M. (2015). Profitability of technical analysis indicators to earn abnormal returns in international exchange markets, *Journal of Economics Finance and Accounting* **1**(4): 0 – 0.

Ghosh, P., Neufeld, A. and Sahoo, J. K. (2022). Forecasting directional movements of stock prices for intraday trading using lstm and random forests, *Finance research letters* **46**: 102280–.

Henrique, B. M., Sobreiro, V. A. and Kimura, H. (2019). Literature review: Machine learning techniques applied to financial market prediction, *Expert systems with applications* **124**: 226–251.

Kong, A., Zhu, H. and Azencott, R. (2021). Predicting intraday jumps in stock prices using liquidity measures and technical indicators, *Journal of forecasting* **40**(3): 416–438.

Kumar, D., Meghwani, S. S. and Thakur, M. (2016). Proximal support vector machine based hybrid prediction models for trend forecasting in financial markets, *Journal of computational science* **17**: 1–13.

Kumar, K. and Haider, M. T. U. (2020). Enhanced prediction of intra-day stock market using metaheuristic optimization on rnn–lstm network, *New generation computing* **39**(1): 231–272.

Kumar, K. and Haider, M. T. U. (2021). Blended computation of machine learning with the recurrent neural network for intra-day stock market movement prediction using a multi-level classifier, *International Journal of Computers and Applications* **43**(8): 733–749.

Labiad, B., Berrado, A. and Benabbou, L. (2016). Machine learning techniques for short term stock movements classification for moroccan stock exchange, *2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA)*, IEEE, pp. 1–6.

Labiad, B., Berrado, A. and Benabbou, L. (2019). Intelligent system for intraday stock market forecasting, *2019 5th International Conference on Optimization and Applications (ICOA)*, IEEE, pp. 1–6.

Li, X., Huang, X., Deng, X. and Zhu, S. (2014). Enhancing quantitative intra-day stock return prediction by integrating both market news and stock prices information, *Neurocomputing (Amsterdam)* **142**: 228–238.

López Padial, D. (2018). Welcome to technical analysis library in python's documentation!¶. Accessed on 2022-08-10.

https://technical-analysis-library-in-python.readthedocs.io/en/latest/

Malkiel, B. G. (2003). The efficient market hypothesis and its critics, *The Journal of economic perspectives* **17**(1): 59–82.

Naik, N. and Mohan, B. R. (2019). Intraday stock prediction based on deep neural network, *National Academy science letters* **43**(3): 241–246.

Narang, R. K. (2013). *Inside the black box : a simple guide to quantitative and high frequency trading*, Wiley finance series, second edition. edn, John Wiley & Sons, Inc., Hoboken, New Jersey.

Paspanthong, A., Tantivasadakarn, N. and Vithayapalert, W. (2019). Machine learning in intraday stock trading, *CS229: Machine Learning, Spring 2019, Stanford University, CA* .

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. et al. (2011). Scikit-learn: Machine learning in python, *Journal of machine learning research* **12**(Oct): 2825–2830. Accessed on 2022-08-08.

Smigel, L. (2022). 79+ amazing algorithmic trading statistics (2022). Accessed on 2022-07-10.
https://analyzingalpha.com/algorithmic-trading-statistics

Sun, J., Xiao, K., Liu, C., Zhou, W. and Xiong, H. (2019). Exploiting intra-day patterns for market shock prediction: A machine learning approach, *Expert systems with applications* **127**: 272–281.

Tanaka-Yamawaki, M. and Tokuoka, S. (2007). Adaptive use of technical indicators for the prediction of intra-day stock prices, *Physica A* **383**(1): 125–133.

Taroon, G., Tomar, A., Manjunath, C., Balamurugan, M., Ghosh, B. and Krishna, A. V. (2020). Employing deep learning in intraday stock trading, *2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, IEEE, pp. 209–214.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society. Series B, Methodological* **58**(1): 267–288.

Tibshirani, R. (2011). Regression shrinkage and selection via the lasso: a retrospective, *Journal of the Royal Statistical Society. Series B, Statistical methodology* **73**(3): 273–282.

Tsinaslanidis, P. E. (2016). *Technical analysis for algorithmic pattern recognition*, Springer, Cham.

Visalakshi, S. and Radha, V. (2014). A literature review of feature selection techniques and applications: Review of feature selection in data mining, *2014 IEEE International Conference on Computational Intelligence and Computing Research*, pp. 1–6.

Wilder, J. W. (1978). *New concepts in technical trading systems*, Trend Research.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y. et al. (2008). Top 10 algorithms in data mining, *Knowledge and information systems* **14**(1): 1–37.

Zhou, Z.-H. (2021). *Machine learning*, Springer, Singapore.

Appendix A

Additional material

A.1 Code

All the code used for this dissertation is available in

https://github.com/abrahamgonz/ml-stock-prediction-movement

A.2 Total results

Table A.1: List of total results

Stock	Feature selection	Model	Cross Val Score	Accuracy	F1-Score
EWW	No	XGBClassifier	54.3	54.3	50.1
EWW	No	LogisticRegression	58.5	58.7	38.2
EWW	No	DecisionTreeClassifier	52.6	50.7	49.4
EWW	No	SVC	50.8	50.2	48.7
EWW	Yes	XGBClassifier	54.2	54.7	47.7
EWW	Yes	LogisticRegression	58.8	58.4	37.3
EWW	Yes	DecisionTreeClassifier	51.0	53.7	52.1
EWW	Yes	SVC	52.8	50.9	49.2
AMXL.MX	No	XGBClassifier	68.7	69.6	50.5
AMXL.MX	No	LogisticRegression	71.4	72.6	42.5

Continuation of Table A.1					
AMXL.MX	No	DecisionTreeClassifier	61.3	60.7	52.0
AMXL.MX	No	SVC	59.3	62.1	51.9
AMXL.MX	Yes	XGBClassifier	67.3	69.8	47.5
AMXL.MX	Yes	LogisticRegression	71.4	72.8	42.4
AMXL.MX	Yes	DecisionTreeClassifier	59.4	60.9	51.8
AMXL.MX	Yes	SVC	59.8	60.7	50.4
WALMEX.MX	No	XGBClassifier	55.1	54.9	50.4
WALMEX.MX	No	LogisticRegression	58.5	60.2	44.9
WALMEX.MX	No	DecisionTreeClassifier	53.1	53.0	50.9
WALMEX.MX	No	SVC	52.8	54.9	52.9
WALMEX.MX	Yes	XGBClassifier	54.3	55.3	49.9
WALMEX.MX	Yes	LogisticRegression	58.6	60.2	44.0
WALMEX.MX	Yes	DecisionTreeClassifier	51.8	53.4	51.2
WALMEX.MX	Yes	SVC	52.1	52.1	50.0
GFNORTEO.MX	No	XGBClassifier	54.2	56.5	54.4
GFNORTEO.MX	No	LogisticRegression	56.8	57.8	43.7
GFNORTEO.MX	No	DecisionTreeClassifier	53.7	54.1	53.2
GFNORTEO.MX	No	SVC	50.7	48.7	47.5
GFNORTEO.MX	Yes	XGBClassifier	55.0	53.6	49.7
GFNORTEO.MX	Yes	LogisticRegression	57.1	58.1	43.7
GFNORTEO.MX	Yes	DecisionTreeClassifier	52.0	52.4	51.7
GFNORTEO.MX	Yes	SVC	51.4	50.7	49.5