FORECASTING ISTANBUL STOCK EXCHANGE

by

Reşat Buğra Erkartal

Submitted to Graduate School of Natural and Applied Sciences
in Partial Fulfillment of the Requirements
for the Degree of Master of Science in
Systems Engineering

Yeditepe University
2018

FORECASTING ISTANBUL STOCK EXCHANGE

APPROVED BY:

Prof. Dr. Linet Özdamar (Thesis Supervisor) ....

Assist. Prof. Dr. Ahmet Yekta Kayman ....

Assist. Prof. Dr. Uğur Yıldıran ....

DATE OF APPROVAL: ..../..../2018

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor Professor Linet Özdamar for the continuous support of my MSc study and related research, for his patience, motivation, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. The door of her office was always open whenever I ran into a problem or had a question about my research.

Besides my advisor, I would like to thank the rest of my thesis committee: Assist. Prof. Uğur Yıldıran and Assist Prof Ahmet Yekta Kayman for their insightful comments and encouragement, but also for the hard question which incented me to widen my research from various perspectives.

Furthermore I would like to thank my wife Pınar Öktem Erkartal for her unlimited support and patience during my thesis.

I would also want to thank my collogue İsmail Kayahan for his great support with his knowledge as well as with his friendship.

Finally, I would like to thank my family for their endless love and support. I respect them for their love and care.

ABSTRACT

FORECASTING ISTANBUL STOCK EXCHANGE

Machine learning methods such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM) have many applications in the area of finance. The majority of these applications focus on the popular problems of share price or exchange rate movement prediction, however, applications on corporate bankruptcy prediction, corporate bond classification and other areas also arise in the literature. The reason behind using machine learning methods is that they are powerful classifiers that are flexible in terms of their assumptions as compared to statistical methods. The flexibility of machine learning classifiers makes them a useful tool to resort to for share price prediction.

This study proposes two novel models, the second being a restricted version of the first. The models generate 5 days ahead buy/sell signals for GARAN (Garanti Bankasi A.Ş.), an equity share that is the top traded stock in BIST100, Istanbul Stock Exchange -Turkey. The first model includes global macroeconomic indicators as well as local inputs whereas the second model is focused more on local inputs. The performances of the two models are tested using SVM, Neural Network with Back-Propagation (BPN), and Decision Tree (DT) algorithms. Though BPN and SVM have previously been used to predict BIST100 Index movement, DT has not been utilized before with this purpose. The performance of all proposed models/methods are tested for a time span of about 6 months. A simple trading strategy is implemented based on buy/sell signals to calculate the rate of return on investment during the testing period. All three algorithms are executed on a rolling horizon basis, that is, they are re-run weekly with updated data to generate daily buy/sell signals for the next week. The results illustrate that DT has about 80 percent prediction accuracy using both models and it outperforms BPN and SVM that have up to 60 percent prediction accuracy.

ÖZET

BORSA ISTANBUL'UN TAHMİNLENMESİ

Yapay Sinir Ağları (YSA) ve Destek Vektör Makineleri (SVM) gibi makine öğrenme yöntemlerinin finans alanında birçok uygulaması vardır. Bu uygulamaların çoğunluğu, hisse senedi fiyatının veya döviz kuru hareket tahmininin popüler sorunlarına odaklanmakta, ancak kurumsal iflas öngörüsü, kurumsal tahvil sınıflandırması ve diğer alanlardaki uygulamalar da literatürde ortaya çıkmaktadır. Makine öğrenme yöntemlerini kullanmanın ardındaki neden, istatistiksel yöntemlerle karşılaştırıldığında varsayımları açısından esnek olan güçlü sınıflandırıcılar olmasıdır. Makine öğrenimi sınıflandırıcılarının esnekliği, onları hisse fiyat tahmini için başvurmak için yararlı bir araç haline getirir.

Bu çalışma, iki yeni model önermektedir; ikincisi, ilkinin kısıtlı bir versiyonudur. Modeller, BIST100, İstanbul Menkul Kıymetler Borsası -Türkiye'de en çok işlem gören hisse senedi olan GARAN (Garanti Bankası A.Ş.) için 5 gün önceden alış / satış sinyalleri üretmektedir. Birinci model, küresel makroekonomik göstergelerin yanı sıra yerel girdileri de içermekte olup, ikinci model ise daha çok yerel girdilere odaklanmaktadır. İki modelin performansları, Destek Vektor Makineleri (SVM), Geri Yayılımlı Sinir Ağı (BPN) ve Karar Ağacı (DT) algoritmaları kullanılarak test edilmiştir. BPN ve SVM daha önce BIST100 Endeks hareketini tahmin etmek için kullanılmasına rağmen, DT bu amaçla daha önce kullanılmamıştır. Önerilen tüm modellerin performansı yaklaşık 6 aylık bir süre için test edilmiş ve test dönemi boyunca yatırım getirisini hesaplamak için alım / satım sinyallerine dayalı basit bir ticaret stratejisi uygulanmıştır. Her üç algoritma da bir sonraki hafta için günlük alım / satım sinyalleri oluşturmak için güncellenmiş verilerle haftalık olarak yeniden çalıştırılmıştır. Sonuçlar, DT'nin her iki modeli de kullanarak yaklaşık yüzde 80 tahmin doğruluğu olduğunu ve yüzde 60'a kadar tahmin doğruluğu olan BPN ve SVM'den daha iyi performans gösterdiğini göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

x_i	vector of p inputs for observation *i*
y_i	value of observation *i*
η	A constant that controls the learning rate
λ	Smoothing parameter for least square method
Φ	Non-linear transformation
AI	Artificial intelligence
ANFIS	Adaptive nero-fuzzy inference
ANN	Artificial neural networks
AORD	Australian all ordinary index
AR	Auto regressive
ARIMA	Autoregressive moving average
BPNN	The back propagation neural network
BSE	Bombay stock exchange
C	Candidate functions of svm
CAC	French stock exchange index
CART	Classification and regression tree
CHAID	Chi-squared automatic interaction detector
CISSE	Shanghai stock exchange
DAX	German stock exchange index
DBNN	Difference boosting neural network
DJIA	Dow Jones industrial average
DT	Decision trees
EBNN	Elman back propagation neural network
ECB	European central bank
ECBM	European central bank meeting anticipation
EGARCH	Exponential general autoregressive conditional heteroscedasticity
FEDI	Federal reserve bank interest rate
FOMC	The federal reserve bank open market committee

FRED	St.Louis federal reserve bank 5 year inflation anticipation
FSVMR	Fuzzy support vector machines regression
FTSE	UK stock exchange index
GA	Genetic algorithm
GARAN	Garanti bankası stock price in borsa istanbul
GARCH	Generalized autoregressive conditional heteroscedasticity
GCV	Generalized cross validation
GGAP-RBF	General growing and pruning radial basis function
GJR- GARCH	Generalized autoregressive conditional heteroscedasticity
HSI	Hang Seng index
ID3	Iterative dichotomiser 3
IGARCH	Integrated generalized autoregressive conditional heteroscedasticity
ISE	Istanbul stock exchange
KOSPI	Korea composite stock price index
KOSPI	Korea stock price index
LDA	Linear discriminate analysis
MAE	Mean absolute error
MAI	Market for alternative investment in thailand
MAPE	Mean absolute percentage error
MARS	Multi adaptive regression of splines
MLFF	Multilayer feed-forward
MLP	Multilayer perceptron
MSE	Mean square error
NLL	Negative log likelihood
NMI	Non-manufacturing indices
OPA	An optimal partition algorithm
PMI	Purchasing managers' indices
QDA	Quadratic discriminant analysis
RBF	Radial basis function
ROC	Rate of change
S&P	US stock exchange index

SENSEX	Bombay stock exchange market-weighted stock market index of 30 well-established 30 companies
SES	Singapore stock exchange index
SET	Stock exchange of Thailand
SOM	Self-organizing map
SRM	Structural risk minimization theory
SSM	Saudi stock market
SVM	Support vector machines
TEPIX	Iran stock exchange
TOPIX	Tokyo stock exchange prices index

1. INTRODUCTION

Stock market forecasting problems are very complicated which draws the attention of many researchers and investors in order give strategic business decisions. Predicting share prices or share price movement direction with high accuracy is almost impossible because they are too many external factors that affects the share price movement [1].

The estimation of the stock prices with its direction requires processing a significant amount of historical data. Machine learning methods are usually practical tools for predicting market states, because they do not rely on restrictive statistical assumptions. Furthermore, the factors affecting stock prices are usually unclear, and many anomalies exist between generally accepted inputs and the output.

Though Fundamental and Technical Analysis preserve their status among traders, the speed and efficiency of Artificial Intelligence (AI) techniques gained momentum due to fast developments in computational power during the last decades. Learning methods are heavily computational number crunching algorithms that manipulate vast historical data. A new trend in market prediction has evolved using learning algorithms and AI. Learning algorithms need human intelligence for guiding them in identifying the factors that are most effective in stock market prediction. 3 types of AI learning algorithms are suggested in this research while the methods used in ANN and DT are developed with new models, SVM model is taken from the literature [2].

This thesis proposes two novel models to predict the movement direction of the share price GARAN (Garanti Bankası A.Ş.) that is listed in the Istanbul Stock Exchange market BIST100. Garanti Bankası is the third largest bank in Turkey and GARAN has usually the highest trading volume in BIST100 (about 30 percent of the total transaction volume) and it is highly correlated with BIST100 index. The reason why we target GARAN share price direction in this study rather than the BIST100 index usually targeted in the previous literature is as follows. The average of GARAN share price's weekly rate of change is significantly higher than that of BIST100 contract price (traded on the VIOP platform, the Turkish derivatives market). Therefore, we expect to achieve a higher return on investment by targeting GARAN rather than BIST100.

The first model proposed here includes both global and local input variables whereas the second model includes only a subset of the first model's variables and it is more focused on local variables. Both proposed models are tested using three supervised learning methods: ANN with back-propagation (BPN), SVM and Decision Tree (DT) learning. In order to provide a good background of comparison, the performance of the two models are also compared against a recently proposed model [3] designed for generating buy/sell signals for the BIST 100 index. Öztekin et al.[3]'s model is adapted and reproduced in this study to target GARAN share price. Other previously proposed models are not reproduced here, but their prediction accuracies are reported as cited.

Performance measures used in the comparison of results include the number of correct share price movement predictions, the number of correct positive and negative movement predictions, as well as the return on investment calculated with a simple trading strategy. In this study, the parameters of all learning algorithms are re-optimized weekly on a rolling horizon basis in order to adapt to changing conditions. The re-optimization is carried out near the end of the trading session every Friday and the next 5 days' daily buy/sell signals are generated. All methods are tested during the first half of 2017 while validation phase covers the years 2010-2016.

1.1. BASICS OF LEARNING METHODS

1.1.1. Supervised Learning

Supervised learning is based on training a data set which is already labeled. In other word the true outcome is assigned. The techniques are used in feed forward or Multilayer Perceptron (MLP), in other words Hidden Layer Network, models. These MLP models have three distinctive characteristics:

i. One or more layers of hidden neurons that are not part of the input or output layers of the network that enable the network to learn and solve any complex problems;

ii. The nonlinearity reflected in the neuronal activity is differentiable;

iii. The interconnection model of the network exhibits a high degree of connectivity.

These characteristics along with learning through training help to solve difficult and diverse problems. Learning through training in a supervised ANN model is also called as error back-propagation algorithm. The error rate is calculated through the differences between the training output and the true outcome.

ANN adopts the weights of nodes in two passes:

Forward Pass: Each input is connected to a node that creates a temporary output signal that is transferred to another node in the next layer by a transfer function. The input signal transmitted to the next layer's node until the signal reaches the output node or nodes. After difference between the calculated output value and the desired level of the original value (data label) output, the error rate is found. that is calculated at the output layer is compared with the desired response and the difference defines the error for that node. The synaptic weights remain same during the forward pass.

Backward Pass: The error rate that is calculated is now propagated backwards. Each synaptic weight is updated by the "delta rule". After the update of the weights, feed forward propagation is starts again and this recursive computation is continues until the desired error rate is reached. [4-7].

Supervised learning paradigm of an ANN is used to find solutions to several linear and non-linear problems [8, 9].

1.1.2. Unsupervised Learning

Supervised learning is observed in Y output as observed in X input set. For unsupervised learning, only focus on the X gird cluster. Since there is no Y cluster associated with X, the Y output is not predictable. Unsupervised learning allows you to have an idea of the nature of the problems that you do not know about research results. The goal is to discover information about the measurements available. For example, the possibility of separating the observations at hand into subgroups is investigated, or the data are examined in an informative manner. The relationship between input variables are exploited to reveal this in problems. In other words the hidden patterns are investigated in order to find the desired outcome. Since there are no labels the error can not be calculated.

The lack of direction for the learning algorithm in unsupervised learning can sometimes be advantageous, since it lets the algorithm to look back for patterns that have not been previously considered [2]. The main characteristics of a Self-Organizing Map (SOM) are:

i. It transforms an incoming signal pattern of arbitrary dimension into one or 2 dimensional maps and performs this transformation adaptively;

ii. The network represents feed forward structure with a single computational layer consisting of neurons arranged in rows and columns;

iii. At each stage of representation, each input signal is kept in its proper context;

iv. Neurons dealing with closely related pieces of information are close together and they communicate through synaptic connections.

2. LITERATURE REVIEW

Stock markets can be described as public markets that exchange company stocks. All traders and transactions are registered with the stock market and regulatory body. Also, the stock markets can be expressed as the secondary markets that implicate the trading between two investors. If both of them are agreed on the same price than the transaction occurs. The prices of the shares will increase if the demand of that share increase and if the majority of the shareholders starts to sell there will be a decrease in price.

Companies can raise additional funds by public offerings of shares, allowing businesses to grow faster. Investors are keen on predicting share prices, especially those that are usually kept in pension fund portfolios.

Stock prices can be predicted with numerous approaches. One of the methods is based on the extraction of best features or selection of features in which reduces the complexity [9, 10]. This technique increases the accuracy of the predicting stock prices in the case of low complexity. Investigating relationships between companies in the same sector can also help building machine learning models. Here, we provide a brief survey on ANN, SVM and DT applications in share price prediction.

The stock markets which is an attractive study area among the researchers, reviewed in many studies frequently in order to find a certain pattern in the movement and forecasting the price of a share. While numerous scientific methods have been applied, but none of them is successful predicting the stock movement with 100 percent accuracy. In order to predict the behavior of a stock market movement, there exist various approaches and various forecasting techniques which can be used by stock market analysts.

This study is focused on machine learning and data mining techniques namely ANN, SVM and DT with fundamental analysis and macroeconomic indicators. In order to find a certain pattern in a time series data our algorithm has to work with large amount of data.

Stock prices are affected by many factors such as company related news, political events, natural disasters, etc. Since the information of stock price is changing too fast, algorithms should adapt themselves with the same speed. Also, these events needs to be processed rapidly through the assistance of a quality. Thus, in recent days many banks and large-scale investors prefer to trade stocks within the as soon as possible. On the other hand, behavioral science can also have an important effect in movement. According to Saad [11] prices have risen most and least on Friday and Monday, respectively.

Naeini et.al. [12] analyses on the cross-correlation between volume and price changes because it needs to be certain amount of volume in order to move a stock price. The results can be summarized as existence of "the power law cross-correlation between logarithmic price and logarithmic volume change" and the other is that "the logarithmic volume change follows the same cubic law as logarithmic price change".

2.1. ARTIFICIAL NEURAL NETWORKS

The traders and investors started to use ANN more often in their decision makings in the last decade. After the derivation of back-propagation in 1986, the modern era of neural networks was started [2]. NNs are implemented to different subjects, therefore, here, we concentrate on ANN applications developed for the stock exchange.

In many real life problems future needs to be forecasted depending on the historical data. Predicting the behavior of stock market indices is just one example. Though perfect prediction nearly impossible, ANN can be used to obtain reasonably good results. Short term (one-lag) and long term (multi-slag) predictions must be taken into account to forecast future price. On one side, the single lag approach forecasts the next upcoming value, on the other side some of the estimated values are also used to estimate the new values in multiple lag predictions.

The forecasting by ANN models consists of fitting of neural network (NN) and prediction. With the help of the forward and backwards passes ANN decides on the weights (until the desired error rate is reached) and builds a model that is ready for the prediction in the training step. The computation time for building the an ANN model is effected by the complexity of the model and it usually take hours. In the second phase the model built in the first step is used in order the forecast the future values. The prediction in time series models, the last desired number of data points are considered as inputs. Therefore, the computation time in predication phase is much more less as compared with the training phase.

Economies are effected by their stock market prices and vice versa. In addition, with the help of the technologic developments trading in the stock market is more accessible to public as well as the traders and investors. Due to that fact, stock market price are influenced by everyday life as well. This fact also effects characteristic of the stock market. Since more people have access to the market the uncertainty (noise) is also increases.

In order to predict the upcoming movement of the stock prices, the uncertainty must be reduced by adding more variables to the forecasting model. But unfortunately there are infinite many variable combinations to explain the whole market behavior completely.

On the other hand, nonlinear functions may describe the uncertainty better. Significant advantages can be achieved through the use of ANN, the most important of which is the ability to convert any non-linear function to a random degree of precision with the appropriate amount of hidden units [13].

As mentioned before, stock markets can be described as public markets that exchange company stocks and derivatives at an settled price that is determined by supply and demand. Companies can raise additional funds by public offerings of shares, allowing businesses to grow faster. Investors are keen on predicting share prices, especially those that are usually kept in pension fund portfolios.

Stock prices can be predicted with various methods. One of the methods is based on the extraction of best features or selection of features in which reduces the complexity [9,10]. This technique increases the accuracy of the predicting stock prices in the case of low complexity. Investigating relationships between companies in the same sector can also help building machine learning models. Here, we provide a brief survey on ANN, SVM and DT applications in share price prediction.

We first discuss ANN applications. A review of ANN applications in stock price prediction, exchange rates and crisis prediction is presented by [14].

Kimoto et al.[15] predict the TOPIX (Tokyo Stock Exchange Price Index) by using a model with five inputs in a BPN: price vector curve, turnover, interest rate, foreign exchange rate, DOW index and a lag of the output variable. McCluskey [16] targets the S&P 500 index using BPN and compares the results with the GA and hand coding approaches. Schierholt & Dagli [17] predict Thailand’s stock exchange index SET employing BPN with less than 2 percent error. Kim et al. [18] propose to adopt a rule based trading by classifying the rapid up/down direction change of the price for the Korean Stock Exchange index (KOSPI 200). The method first classifies price movement direction with several technical indicators using machine learning, and then, trading rules are mined to resolve conflicting outputs by inductive learning. Kutsurelis [19] targets the monthly returns of the TOPIX using the dollar index, S&P 500 price/earnings (P/E) ratio, U.S. market performance index and liquidity as inputs as well as one period lagged TOPIX index level. Chye and Suan [20] target the share price of Singapore Airlines (SIA) listed in Singapore Stock Exchange. The model predicts one-week ahead closing price of the share based on inputs of historical high, low and closing prices, and traded volume. Dutt and Ghosh [21] apply ANN to predict the 5 year US treasury. Burgess et al. [22] describe ANN model with 2 hidden layers for predicting Euro/Dollar futures using daily high, low, open and close prices. The ANN generates an average annual return of 47 percent. Yamashita et al. [23] utilize a multi-branch ANN with TOPIX as the target. Majumder and Hussain [24] present a computational approach for predicting the S&P CNX Nifty 50 index direction using lagged index attributes. Tilakaratne [5] predict one day ahead trading signals of the Australian All Ordinary (AORD) index using the current day's return of S&P 500 closing prices, FTSE 100 (UK) and CAC 40 (France) indices as inputs and lagged AORD index level. The authors find that feedforward ANN performs better than probabilistic networks. Charkha [25] uses BPN with early stopping and Radial Basis Function Neural Network (RBFNN) to predict the trend in stock price as well as the stock price itself with only past prices as inputs. Mandziuk and Jaruszewicz [26] propose ANN supported by genetic algorithm (GA) targeting the German DAX index with TOPIX and NYSE indices, EUR/USD and USD/JPY exchange rates as inputs. Here, the genetic algorithm catches the best input variables for daily forecasting and it is applied on a rolling horizon basis updating the inputs every 5 days. Mehrara et al. [27] simulate BPN and ANN with Genetic Algorithm to predict TEPIX (Iran Stock Exchange index) using technical analysis rules that are crossed over by the GA. Mantri et al. [9] apply various Auto Regressive methods and ANN to compute the volatility of the stock market at India, BSE SENSEX, to discover that both approaches are equally efficient in predicting volatility.

Now, we summarize the literature on the Istanbul Stock Exchange index (ISE) recently re-named as Borsa Istanbul Stock Exchange (BIST). Diler [28] present a model that uses technical indicators targeting the ISE index and implement BPN. Altay and Satman [29] also propose a similar BPN model. Both Diler [28] and Altay and Satman [29] achieve nearly 60 percent prediction accuracy with BPN. Yümlü et al. [30] target the ISE index using the USD\TRY exchange rate and two bond interest rates in a modular ANN model. Kara et al. [31] target the ISE index movement direction using another set of technical analysis indicators as inputs and compare BPN, OLS (Ordinary Least Squares) and SVM to discover that BPN outperforms SVM and OLS with 75.74 percent accuracy in predicting ISE's direction. OLS is the worst performer with 55 percent accuracy and SVM comes in second with 71.52 percent accuracy. The more recent model by Öztekin et al. [3] is also reported to have a good performance with a prediction accuracy of 72 percent for SVM during cross-validation and 60 percent accuracy for BPN. This model includes daily gold price in USD, USD/TRY and Euro/TRY exchange rates, Trend calculated for BIST100, NASDAQ Composite index, and previous BIST100 index closing price as inputs.

2.2. SUPPORT VECTOR MACHINES

Trafalis and İnce [32] compare SVM against BPN and RBFNN using the e-insensitive loss function and several different quadratic optimization algorithms. SVM is shown to be superior over BPN. Tay and Cao [33] propose a modified SVM for financial series prediction called C-ascending SVMs. The goal of this approach is to increase the weights of most current e-insensitive errors and de-weigh more distant ones – analogous to the discounted least squares approach. Both linear and exponential weight functions are tested against several stock indices including the S&P 500. Van Gestel et al. [34] propose SVM in a Bayesian evidence framework to predict US short term T-bill and the DAX30 index. Yang et al. [35] propose a non-fixed and asymmetrical margin, along with the use of momentum, to improve SVM's ability to predict financial time series. The e-insensitive loss function is modified to have different upside and downside margins based on the standard deviation of the input data. The authors show that the mean absolute error of SVM's one step ahead predictions of the Hang Seng Index (HSI) and Dow Jones Industrial Average (DJIA) index are lower than those of Auto Regressive and RBFNN. Liang and Sun [36] propose an adaptive method for modifying the RBF kernel function during the training of SVM. An optimal partitioning algorithm is used to modify the kernel, making the kernel data dependent. The authors target S&P 500 and Shanghai Stock Exchange (SSE) indices. Kim [37] proposes a model for predicting the KOSPI index with twelve "technical indicators" as inputs. In Yang et al. [38], a two phase SVM training method is proposed for detecting outliers in the data, thus reducing the mean absolute error in the predictions. Abraham et al. [39] compare the one-step ahead time series prediction performances of ANN, the Levenberg-Marquardt algorithm, SVM, Takagi-Sugeno neuro-fuzzy model and the Difference Boosting Neural Network. The targets are the NASDAQ100 index and the NIFTY index. The authors show that SVM performs marginally better. Bao et al. [40] propose SVM that uses the e- insensitive loss function and the RBF kernel function to predict five days ahead stock price direction of Haier, Inc. listed in SSE. Huang et al. [41] propose SVM for predicting the direction of the NIKKEI 225 index based on several inputs including interest rates, consumer price index, and other market data. Bao et al. [42] propose a Fuzzy Support Vector Machine Regression method (FSVMR) for predicting the SSE index. The FSVMR is trained during cross validation method to identify dynamic parameters. The latter approach is shown to be more effective than SVM. Jung and Reggia [43] use SVM to predict stock price index as a time series problem and observe that SVM performs better than BPN and case based reasoning. They argue that SVM implements the structural risk minimization principle, which leads to better generalization than conventional techniques.

2.3. DECISION TREES (DT)

Wu et al. [44] present a stock trading method by combining a filter rule and DT where the filter rule is used to generate candidate trading points that are subsequently clustered and screened by DT. This method is used to identify the right stocks in the Taiwan and NASDAQ stock exchange markets and the right purchase timing. Wang and Chan [45] propose a two-layer bias DT using technical indicators as inputs to create a decision rule that makes ‘buy’ or ‘don’t buy’ recommendations in the stock exchange market. Sun and Li [46] present a data mining method combining attribute-oriented induction, information gain, and DT for financial distress prediction using financial ratio attributes and an entropy-based discretization method. An empirical experiment with 35 financial ratios and 135 pairs of listed companies results in a satisfactory performance. Lu and Chen [47] employ DT based multi-learner mining techniques to explore classification rules of information transparency levels of listed firms in Taiwan’s stock market. The approach is able to provide explicit classification rules that are able to discriminate good information disclosure data from poor information disclosure data.

In Table 2.1, the literature is summarized indicating the specific stock exchange targets, the methods and input variables.

Table 2.1. Summary table of the models used in literature

Reference	Country	Model	Input	Target
Tan Sen Suan [30]	Singapore	BPNN	High	Closing Short Price
			Low	Of Singapore Airlines
			Closing	
			Volume	
			(last 3 days)	
Burgess [33]	Forex	BPNN	High	Future 3 month Euro/USD Contracts
			Low	
			Open	
			Close	
Dutt and Ghosh [34]	USA	BPNN	Liability	5 year Treasuries
			Sales/Net Worth	
			Financial Strength	
			Working Capital/Sales	
			Depth Proportion	
			Profit/Sales	
			Earning/ Fixed Cost	
			Projected 5 year Revenue Growth	
			Subjective Prospect of Company	

Table 2.1. Continued

Reference	Country	Model	Input	Target
Jibendu Kumar Mantri [8]	India	GARCH, EGARCH, GJR-GARCH, IGARCH, ANN	high low index levels	Volatilities
			low index levels	Volatilities BSE SENSEX INDEX (Sensex -30) and the NSE NIFTY (Nifty – 50)
M. Thenmozhi [2]	India	MLP using Back propagation	4 day return value	BSE (Bombay Stock Exchange) Sensex return
John S. Chandler	Korea Stock Price Index 200 (KOSPI 200)	ANN	Stochastic %K	price direction KOSPI 200
-Kyoung-jae Kim		CBR	Stochastic Slow %D	
-Ingoo Han [19]			Momentum	
			ROC	
			RSI	
			AD Oscillator	
			CCI	
			OSCP	
			Disparity5	
			Stochastic %D	
			Larry William's %R	

Table 2.1. Continued

Reference	Country	Model	Input	Target
Kimoto T. [18]	Japan	supplementary learning (proposed) for hierarchical networks	Price vector	TOPIX (Tokyo Stock Exchange Prices Index closing value)
			Turnover	
			Interest Rate	
			Foreign exchange rate	
			Dow-Jones Closing value	
Tag and Cao [46]	USA	C. ascending SVM	Bond prices	3months treasury
		e. Descending SVM		
Tony Van Gestel [50]	Germany	LS-SVM	Closing prices of DAX30,	DAX 30
			Germany 3-Month Middle Rate,	
			US 30-year bond,	
			S&P500,	
			FTSE	
			CAC40	
			(All inputs are used with 6 lags)	
Kim K.J. [57]	Korea	Regular SVM	Technical Indicators x 12= Huang	KOSPI Direction

3. METHODOLOGY

3.1. ANN STRUCTURE

The most important feature of the ANN model influenced by the biological nervous system is data processing. It is possible to perceive the artificial neural network as a whole organism consisting of a large number of computing units that interact with each other to solve any problem working together in harmony.

Although the process of brain information processing is not yet fully understood, there are various theories that try to explain this process. Accordingly, in the human brain, signals are gathered by each neuron from neighboring neurons via dendrites and sends electrical currents to other neurons through axons that divide by thousands. At the end of each branch, an organism called synapse transforms the immediate survival into electrical signals and thus sends "excitatory" or "inhibitory" effects to the connected neurons [48]. Each neuron can be seen as a processor that makes a very simple calculation, such as deciding whether to send signals to other neurons (see Figure 3.1).

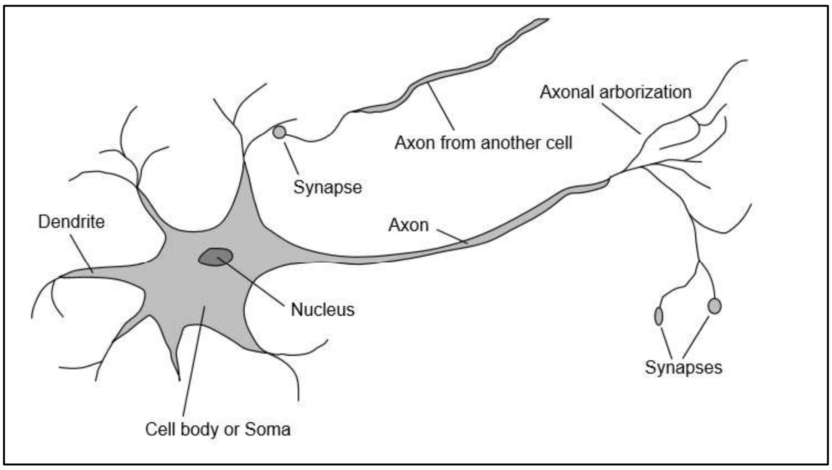

Figure 3.1. Human nerve cell structure

Learning occurs when the effects of synapses are altered, or in other words, the effect of a neuron on another neuron is changed [49].

The awareness that the complex learning system in the brain contains tightly connected neuron clusters has inspired ANN. Although structurally a neuron is simple, neurons that are intimately connected to each other can perform complex learning tasks. For example, the human being brain consists around 1011 neurons, which has approximately 1015 connections between these neurons. For this reason, the brain can be simulated as a parallel computer [50].

3.1.1. Artificial Neural Cell

In general, an ANN is consisted of a numerous simple computing units linked together by weighted links. By analogy, the processing units can be likened to the neurons in the brain. Each computing unit receives inputs from many other units and generates an output. The output is distributed in such a way that it enters other units in the network [51].

The correlation between a human nerve cell and ANN is provided in Table 3.1.

Table 3.1. Correlation between human nerve cell and ANN

Biological Neural Model	Artificial Neural Model
Synapse	Weight
Axon	Output
Dentrite	Input
Cell Body	Transfer Function

Computing units come together to form an ANN, but not randomly. Generally, input, hidden, and output layers form three layers, and each layer forms a network in a parallel fashion.

Information sent from a neuron to another neuron in the brain is very small. This shows that the brain does not carry critical information directly, that information is trapped and distributed in the interconnections between neurons. Another name of ANN, the "connectionist model" comes from this feature [52].

"Plasticity" is the replacement of synaptic connections between nerve cells or the creation of new links [53]. The basic concept for learning in a classroom is this flexibility. It is not yet known how all of the synapses in the brain change their weights.

Although ANN is designed with inspiration from biological nervous systems, it cannot model many complicated processes in its brain. For this reason, many of the features of ANN models are not compatible with biological neural networks.

3.1.2. Basic Components of the ANN

An exemplary ANN can be shown as in Figure 3.2.

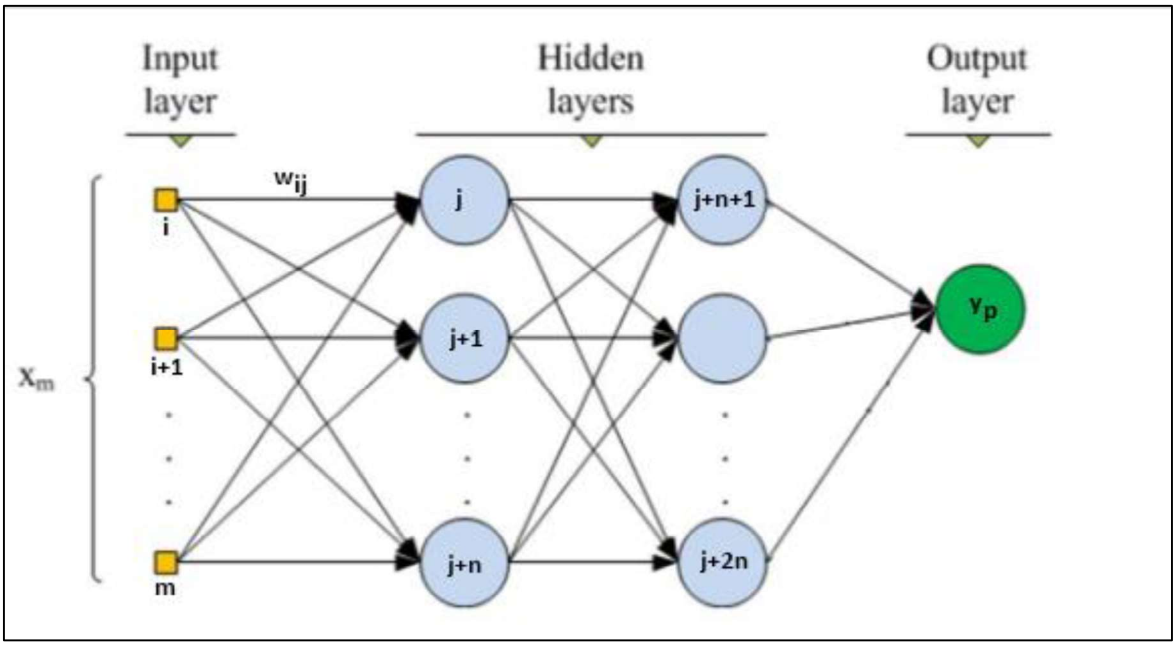

Figure 3.2. ANN structure

3.1.2.1. Layers

Most networks composes of three layers: input, hidden, and output layer (Figure 2). The amount of hidden layers may be higher, but only one hidden layer is sufficient for most applications. There is a weight indicated for each connection between two layers [50].

The number of input units in the input layer depends on the type and number of properties in the data set. The number of hidden layers and the number of units in each hidden layer can be determined by the user. Depending on the nature of the problem to be solved, there may be more than one unit in the output layer.

Input Layer: The input layer is a hypothetical layer. It retrieves the property values from the data set and passes these values to the hidden layer without any processing. Since there is no calculation on the units in the input layer, there are no faults in the units in this layer. Therefore, it does not have a detailed unit structure like the hidden and output layers.

Hidden Layer: In traditional feed-forward neural networks, hidden layers are used to connect the output units and input units. Since the number of neurons in the hidden layers and the total number of layers are unknown, researchers prefer to hide them in the scheme of the network.

In many applications the layers have structure of hundreds of nodes and at least 2 layers it is almost impossible to draw a full diagram with that may structures. If the target is selected as a binary variable (i.e direction of the share detection), the output layer is a single node. The output of this node is a threshold that provides a positive or negative indication of the target [54].

It may be desirable to use too many nodes in this layer, since more hidden layer units are expected to increase the network's ability to determine complex patterns and flexibility. On the other hand, an unnecessarily large number of hidden layers leads to over-learning, in which the training set is memorized and the ability to generalize fails. If over-learning occurs, the number of units in the hidden layer can be reduced, or vice versa, if the training results are not at the desired level, the number of units in the hidden layer can be increased [52]. Whenever a new hidden layer is added to the network, local minimum points to which the error function can be attached are added in the training process. For this reason, a single hidden layer network is created first. However, if this network is insufficient, then, a new hidden layer is added. In practice, the least number of hidden layers is used to meet the required learning accuracy [55].

There is no analytical method for determining the optimal number of units in the hidden layers. Hence, the only way to overcome the ambiguities in the number of hidden layers and number of units in these layers is the trial and error method [56].

There is no need to use a hidden layer for "linearly separable" problems, because, in such problems, multi-layer models have no advantage over single layer models [57].

Output Layer: For binary classification problems, it is common to use a single unit to which a "threshold" value is set priory to assign classes at the output layer. However, a single output unit can be used for problems with more than one class. In the same way, threshold values must be determined for each class to distinguish these classes from each other. Assessment of field experts is necessary to determine the threshold values correctly.

If the number of possible classes is not large, flag variables can also be used. Variables consisting of a class can be converted to a unit flag variable.

3.1.2.2. Weights

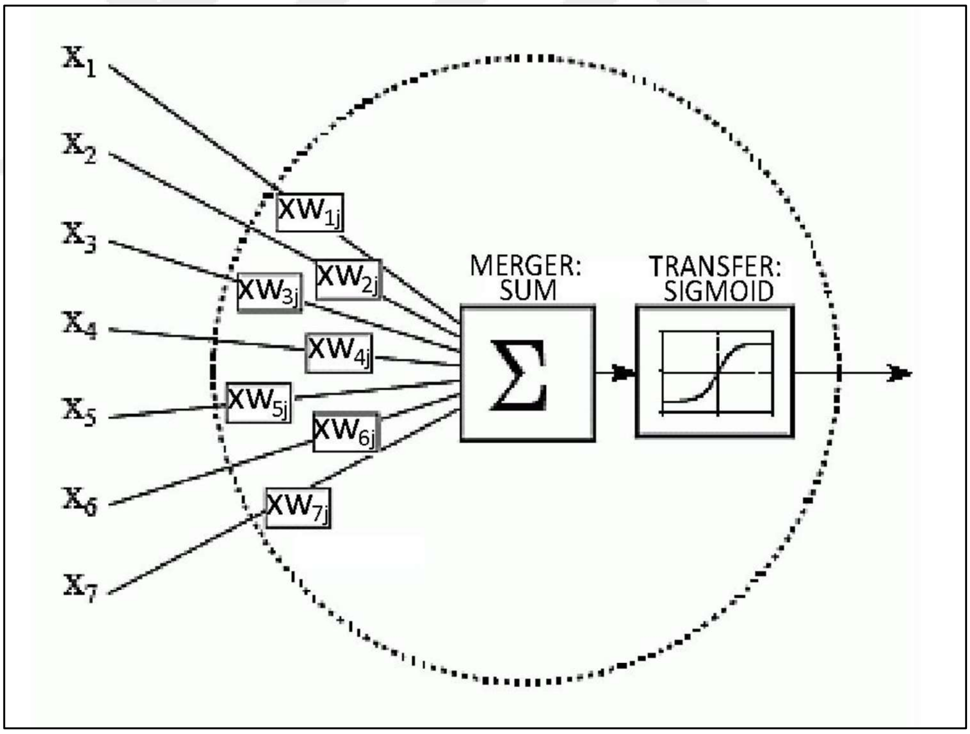

Figure 3.3. Neural network node activation

In ANN models, information is associated with weights w_{ij}. The display of information is an important factor affecting the design of the shaped network, and therefore, its performance.

Input signals x_i are multiplied with weights w_{ij} and combined linearly to calculate a neuron j's output y_j. Then, the output y_j is transformed into a new output by a transfer function g(·) (Eq 3.1) (See Figure 3.3).

$$y_j = g(\sum_i w_{ij} x_i) \quad \forall j \tag{3.1}$$

The transfer function usually takes a nonlinear shape that is limited and non-decreasing. [51].

Eq 4.1 is forward propagated recursively from hidden layer to hidden layer ending at the output node y_p. Thus, once Eq 3.1 determines y_j for a hidden node j, y_j becomes an input signal for the next layer.

At any node j, the combined output y_j can be obtained in different ways before being treated by the transfer function. Nonlinear combinations (e.g. geometric combination or minimum/maximum operators) can be used as merging functions. In Table 3.2, we list different combination functions that are transformed by g(·).

Table 3.2. Examples of merge functions

Merge Function	**Function Form**
Weighted Sum	$z = w_{ij}x_i$
Product	$z = \prod w_i x_i$
Maximum	$z = \max\{w_{ij}x_i\}$
Minimum	$z = \min\{w_{ij}x_i\}$

3.1.2.3. The Transfer Function

The transfer function g(·) generates a value for each computing using the linear combination its input signals.

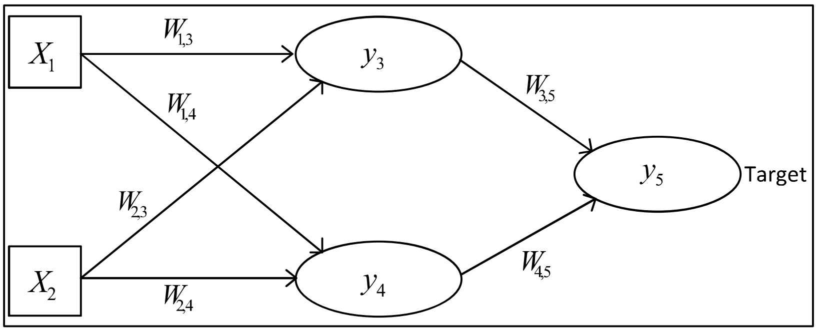

Figure 3.4. Illustrative ANN

In the example in Figure 3.4 the input signals coming to the 5th node is calculated as follows: Nodes 3 and 4 are hidden and their output values y_3 and y_4 are inputs for the output node 5. Hence, in feedforward networks, one can calculate y_5 as follows.

$$y_5 = g(W_{3,5} \cdot y_3 + W_{4,5} \cdot y_4) = g(W_{3,5} \cdot g(W_{1,3} \cdot y_1 + W_{2,3} \cdot y_2) + W_{4,5} \cdot g(W_{1,4} \cdot y_1 + W_{2,4} \cdot y_2)$$

There are several types of transfer functions used in the literature, however, selecting the appropriate one is by trial and error [83].

The sigmoid function $g(\cdot)$ in Eq (3.2) is an exemplary transfer function that transforms the weighted input calculated by the merge functions unto the (0,1) interval. If $g(\cdot)\geq 0.5$, then, it relays its value to the next layer nodes. Else, it suppresses the signal [58].

$$g(z) = \frac{1}{1 + e^{-\alpha z}} \tag{3.2}$$

where z is the outcome of the weighted input combination.

By changing the slope parameter α in Eq (3.2) sigmoid functions achieve different activation signals.

The sigmoid function is a function that combines linear, close, curvilinear and close-to-steady behavior depending on input. Near the center, the line is close to the center, and as you start to move away from the center, the curvature becomes almost constant as you approach the extreme values (Figure 3.5). For this reason, small increases in the input value, depending on the position in which it occurs, cause different magnitude increments in the value of the function. While small increments near the center bring large increments to the function, the increase of the input value at the extreme points creates very small increases in the value of the function [50].

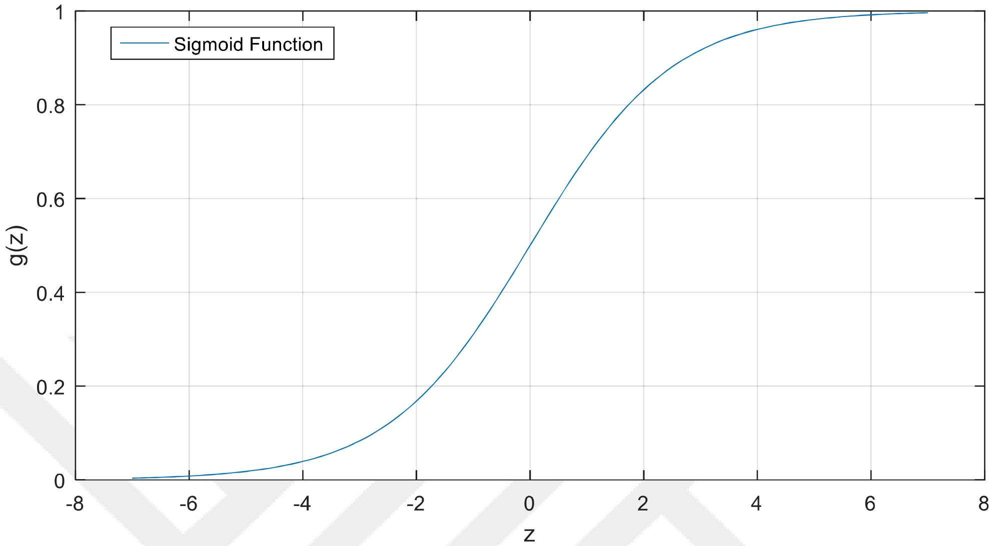

Figure 3.5. The sigmoid function

3.1.2.4. Error Function

The error function, "Error Sum of Squares" is used in many ANN models Equation (3.3). It is the sum of the squares of prediction errors in all output units of all observations in the training set [59].

$$SSE = \sum_{n=1}^{N} (y_n - \hat{y}_n)^2 \tag{3.3}$$

Where:

- $\{y_n\}$ is the actual nth observation in time series.
- $\{\hat{y}_n\}$ is the estimated or forecasted nth observation in time series.

Another error function is the cross entropy function that is used in cases where it is desired to give a probability estimate of the network output.

The advantages of these error functions are that they are easily derivable, that they are not affected by the trends and magnitudes of previous errors, and that the costs of equal errors are equal regardless of inputs. While the choice of error functions depends on the application, most of the applications use standard error functions [51].

3.1.3. A Numerical Example of ANN

The goal of back propagation is to optimize the weights so that the neural network can learn how to correctly map arbitrary inputs to outputs. The process starts by calculating an output with arbitrary weights.

Let us assume that nodes 1,2 are the inputs and node 5 is the target of the neural network illustrated in Figure 3.6. Nodes 3 and 4 are hidden nodes using the sigmoid transfer function.

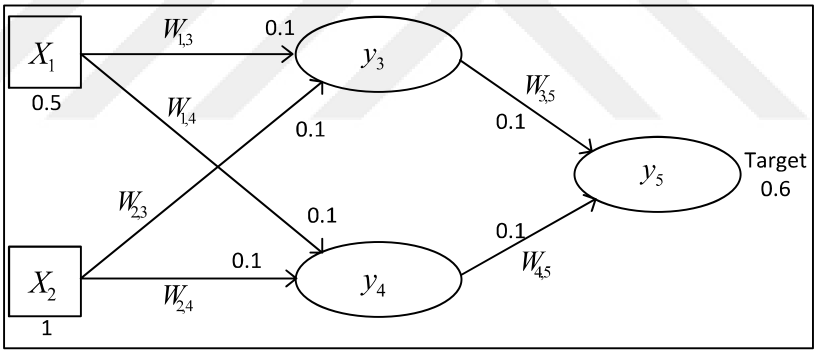

Figure 3.6. Example ANN

Initial weights are all assumed to be 0.1 and the values of input 1,input 2 and the target are 0.5, 1 and 0.6, respectively.

The net inputs coming to nodes 3 and 4 are calculated as follows:

$y_3 = g(w_{1,3} * x_1 + w_{2,3} * x_2) = g(0.1*0.5 + 0.1*1) = 0.517$

where g is the logistic transfer function.

y_3 is equal to y_4 because both nodes receive the same inputs.

Now the outputs of the nodes 3 and 4 will act like the inputs to node 5 such as follows:

$y_5=g(w_{3,5}*y_3+w_{4,5}*y_4)$

$=g(0,1*0.517+0,1*0.517)=0.103$

Remember that the actual target value is 0.6, therefore, the error is:

$SSE=(0.6-0.5235)^2=0.006$

The backward pass updates weights using the Delta Rule given Eq (3.4)

Delta Rule:

$$\Delta w = w - w_{old} = -\eta \frac{\partial SSE}{\partial w} = +\eta\delta x \tag{3.4}$$

where $\delta = y_{target} - \hat{y}$ and η is a constant that controls the learning rate. Backward propagation continues until either the error is small enough or the iteration limit is reached.

3.2. SUPPORT VECTOR MACHINES

SVMs are one of the commonly used machine learning technique that can classify and forecast the existing data. They have strong theoretical background and excellent empirical successes, their strength comes from their ability to separate data both linearly and nonlinearly. In order to run SVMs successfully the output should be categorical (preferably binary). SVM can be applied to multi-class problems as well using "one against one" approach combining several binary classifiers [27] or "one- against-all approach" training *m* SVM classifiers where each classifier distinguishes images in one category from all other m–1 categories [60].

We are given the training data $\{x_1 \ldots x_n\}$ that are vectors in some space X and their labels $\{y_1 \ldots y_n\}$ where y_i in {-1, 1}. SVMs as a binary classification method (their simplest form), the hyper planes are calculated in order to split the training data into half by a maximal margin. All vectors lying on one side of the hyper plane are labeled as -1, and all vectors lying on the other side are labelled as 1. If the vector lie very close to the hyperplane they are called support vectors. The training data might not be linearly separable and at that case non-linear hyperplane might be used for classification [61].

The basic idea is to transform input vectors into a high dimensional feature space using non-linear transformation Φ, and then to do a linear separation in feature space. To construct a non-linear SVM classifier, inner product $< x, y >$ is replaced by a kernel function K(*x, y*).A kernel is a function K(*x, y*) that given two vectors in input space, returns the dot product of their images in feature space [61].

A kernel is a function K(*x, y*) that given two vectors in input space, returns the dot product of their images in feature space [61].

$$K(x.y) = \langle \phi(x). \phi(y) \rangle \tag{3.5}$$

There are several different kernels; choosing one depends on the task at hand. One of the simplest is the polynomial kernel.

$$K(x.y) = \langle x.y \rangle^d \tag{3.6}$$

For example, taking d = 2 and x, y in RxR

$$\begin{aligned} \langle x.y \rangle^2 &= (x_1 y_1 + x_2 y_2)^2 \\ &= x_1^2 y_1^2 + x_2^2 y_2^2 + 2 x_1 y_1 x_2 y_2 \\ &= \left(x_1^2, x_2^2, \sqrt{2} x_1 x_2\right)\left(y_1^2, y_2^2, \sqrt{2} y_1 y_2\right) \\ &= \langle \phi(x). \phi(y) \rangle \end{aligned} \tag{3.7}$$

Defining:

$$\phi(x) = \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix} \tag{3.8}$$

The dual representation of the decision function is:

$$f(x) = sign\ (y_i \alpha_i {<} x.x_i {>} + b) \tag{3.9}$$

If the decision function is considered for the optimal hyper plane classifier in dual form and apply the mapping Φ to each vector it uses, we obtain:

$$f(x) = sign\left(\sum_{i=1}^{l} y_i \alpha_i K(x, x_i) + b\right) \tag{3.10}$$

3.3. DECISION TREE LEARNING

Although DT learning is commonly used in data mining, there are very few algorithms that is used in the finance area. Like in other models the main goal is to predict the future outcomes of the target variable depending on various input variables. Each node in between the root node and the end node represents to one of the input variables where those nodes may produce at least two child nodes that represents the class of the variables (See Figure 3.7). The process of separation is continued recursively until there are no variables are available to partition. In DT, different predictor functions such as majority class, or, naive Bayes classifier Kim et al. [18] are used to classify. This process is repeated on each derived subset recursively as illustrated in Figure 3.8. When the stopping criteria is reached (if the target=subset of the node or the subset are no longer available to split), then the recursion ends. This is a top-down induction method that is greedy in nature. Using this top down structure, DT partitions instances into separate classes leading to a generalized structure used to classify instances with unknown class values (See Figure 3.7).

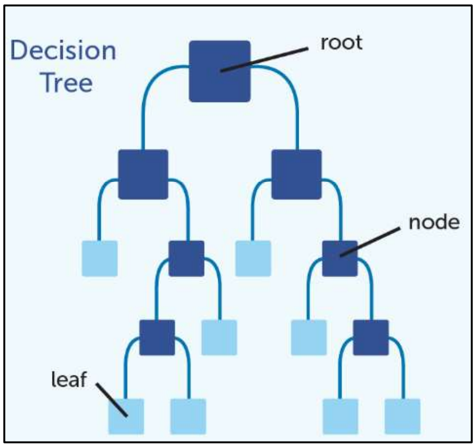

Figure 3.7. Ilustration of a decision tree

Unlike the intensive use of decision tree learning in data mining, the number of algorithms in finance is very limited. In this model, the goal is to estimate the target variable by taking the input variables as basis. Each node simultaneously creates two child nodes that determine the class partition of the input variable, which is an inner node. A leaf represents the class of the target variable given the domains of the input variables represented by the path from the root to the leaf (See Figure 3.7). Decision tree provides categorization with estimation tools such as majority class or Naïve Bayes classifier. Learning is based on the attribute test when the resource set is divided into sub-sets, which are repeated continually in each sub-set. Repetition continues until there is no effect on the prediction of the partitioning process or the subset and the target variable in a node are the same. This procedure comprises top-down orientation that is greedy in nature. DT creates separated sub-clusters that conducts a comprehensive structure which classifies the cases with unknown class values by the help of this top down structure. (See Figure 3.8)

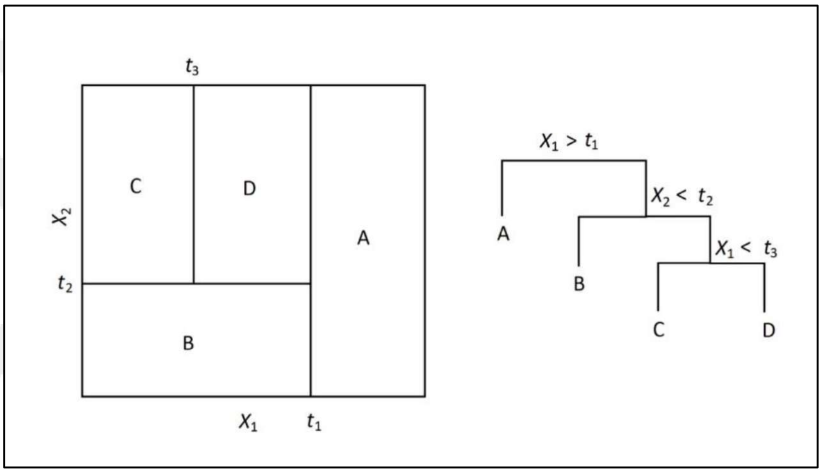

Figure 3.8. Recursive partioning of a decision tree

The notable algorithms are as follows:

ID3, C4.5, CART, CHAID, Decision Stream, MARS, Conditional Interface Trees.

Example of ID3

To illustrate DT learning we present an example with qualitative inputs using ID3 example:

Suppose we want ID3 to decide whether the quality of the movie is good enough to the theater. The necessary information needed to perform ID3 algorithm is given in the Table 3.3.

The target classification either can be YES or NO depending on the question "should we go to theater or not?"

The attributes are:

Quality of the movie (QOM) = {High Quality (HQ), Medium Quality (MQ), Low Quality (LQ)}

Location of the theatre (LOT) = {far, medium, near}

Money = {not enough, enough}

Time availability (TA) = {not available, available}

Examples of set S are given in Table 3.3.

Table 3.3. DT example set

Day	QOM	LOT	Money	TA	Go Theater
D1	HQ	Far	Enough	Not Available	No
D2	HQ	Far	Enough	Available	No
D3	MQ	Far	Enough	Not Available	Yes
D4	LQ	Medium	Enough	Not Available	Yes
D5	LQ	Near	Not Enough	Not Available	Yes
D6	LQ	Near	Not Enough	Available	No
D7	MQ	Near	Not Enough	Available	Yes
D8	HQ	Medium	Enough	Not Available	No
D9	HQ	Near	Not Enough	Not Available	Yes
D10	LQ	Medium	Not Enough	Not Available	Yes
D11	HQ	Medium	Not Enough	Available	Yes
D12	MQ	Medium	Enough	Available	Yes
D13	MQ	Far	Not Enough	Not Available	Yes
D14	LQ	Medium	Enough	Available	No

We need to find which attribute will be the root node in our decision tree. The gain is calculated for all four attributes:

Gain(S, QOM) = 0.246

Gain(S, LOT) = 0.029

Gain(S, Money) = 0.151

Gain(S, TA) = 0.048

In order to decide which of the attribute is root node the above calculation's gain values are taken into consideration, in which the maximum among them is the root node. Therefore QOM attribute is selected as root node.

Due to the fact that the QOM has three alternatives, the root node has three branches (HQ, MQ, LQ). Since the outlook is selected as the root node, the HQ branch nodes attributes should be tested. Because the outlook has already been used as root node the selection must be made between the remaining three attributes: Money, LOT, or TA.

S_{HQ} = {D1, D2, D8, D9, D11} = 5 examples from Table 3.3. with QOM = HQ

Gain(S_{HQ}, Money) = 0.970

Gain(S_{HQ}, LOC) = 0.570

Gain(S_{HQ}, TA) = 0.019

On the second iteration, we calculated the gain ratio again and Money has the highest gain; therefore, it is used as the decision node. This procedure continues recursively until all decision tree is completed.

The final decision tree is given in Figure 3.9.

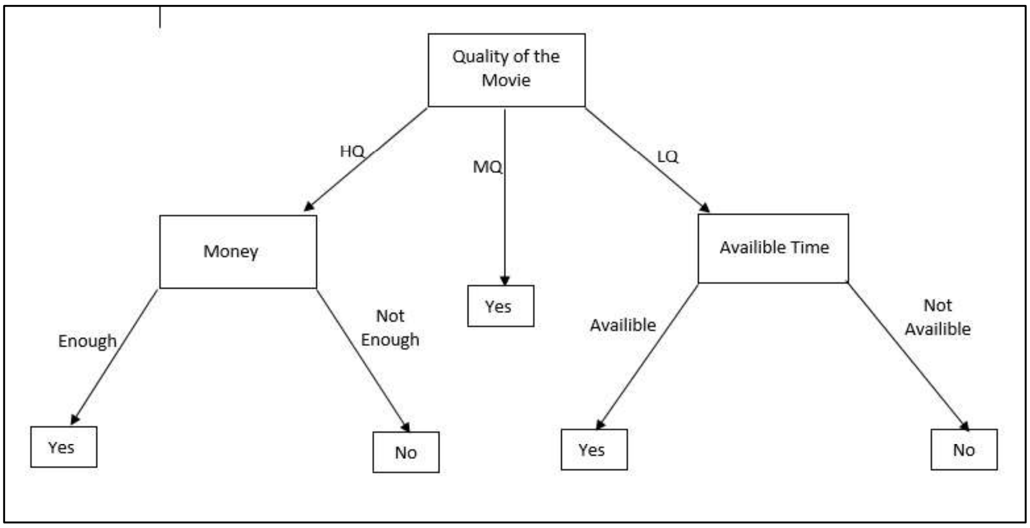

Figure 3.9. Final DT illustration

The decision tree can also be expressed in rule format:

IF QOM = HQ AND Money = Enough THEN Go Theater = yes

IF QOM = LQ AND Money = Enough THEN Go Theater = yes

IF QOM = LQ AND TA = Available THEN Go Theater = yes

IF QOM = MQ THEN Go Theater = yes

IF QOM = LQ AND TA = Not Available TA THEN Go Theater = no

In this study MARS algorithm is used for classification since the inputs are numerical.

3.3.1. Multivariate Adaptive Regression Spline (MARS) Algorithm

MARS algorithm stands for Multivariate Adaptive Regression Spline. MARS uses piecewise linear basis functions consisting of reflected differences for each pair of inputs. These functions are assigned weights and transformed into a regression function. MARS decides on the number of pairwise terms in the regression function by conducting a forward pass and adding new terms consecutively to the regression function to reduce residual error. Then, a backward pass is conducted by removing one term at a time from the model in order to improve validation. The algorithm removes the term which increases the residual error the least. It continues removing terms until cross validation is satisfied.

MARS uses piecewise linear basis functions of the form $(x\text{-}t)_+$ and $(t\text{-}x)_+$. The + means positive part only. Thus, the basis function is represented in Eq(3.11) below.

$$(x-t)_+ = \begin{cases} x-t, & if\ \ x>t \\ 0, & otherwise, \end{cases} \quad \text{and} \quad (t-x)_+ = \begin{cases} t-x, & if\ \ x<t \\ 0, & otherwise, \end{cases} \tag{3.12}$$

MARS uses a collection of functions comprised of reflected difference pairs for each input X_j with knots at each observed value x_{ij} of that input. In Eq(3.12) below, p is the number of input variables, and N is the number of instances in the data.

$$C=\left\{(X_t-t)_+, \left(t-X_j\right)_+\right\}_{\substack{t\in\{x_{1j},x_{2j},\ldots,x_{Nj}\} \\ j=1,2,3,\ldots,p}} \tag{3.13}$$

If all input values are distinct, then, the set C contains 2Np functions. Then the MARS model has the general form in Eq (3.13).

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X) \tag{3.14}$$

h_m (X) is a function from the set C candidate functions or a product of two or more such functions. β_m are the coefficients estimated by minimizing the residual sum of squares (as in linear regression). These coefficients can be considered as weights that represent the importance of the variable. Finally, M is the number of terms that the user allows in MARS execution. f(x) provides the prediction for a given input vector X.

4. MODELS BUILT FOR THE ANN, SVM AND THE DT METHODS

4.1. CHOOSING THE RIGHT TARGET

The ANN model developed in this study targets to generate daily buy/sell signals for the GARAN (Garanti Bankası) stock that is among the top traded stocks in terms of daily trading volume in BIST100, the Istanbul Stock Exchange. The reason for selecting GARAN's daily stock price direction in the ANN model is its high correlation with BIST100 index and its high trade volume. GARAN is easily bought and sold even in pre-holiday dormant market. The correlation of GARAN closing price with BIST100 index is 0.905 which is statistically significant at 1 percent using a sample of 1509 consecutive data. GARAN held about 30% of the transaction volume in BIST100 during the time interval (2010-2016). (See Figure 4.1)

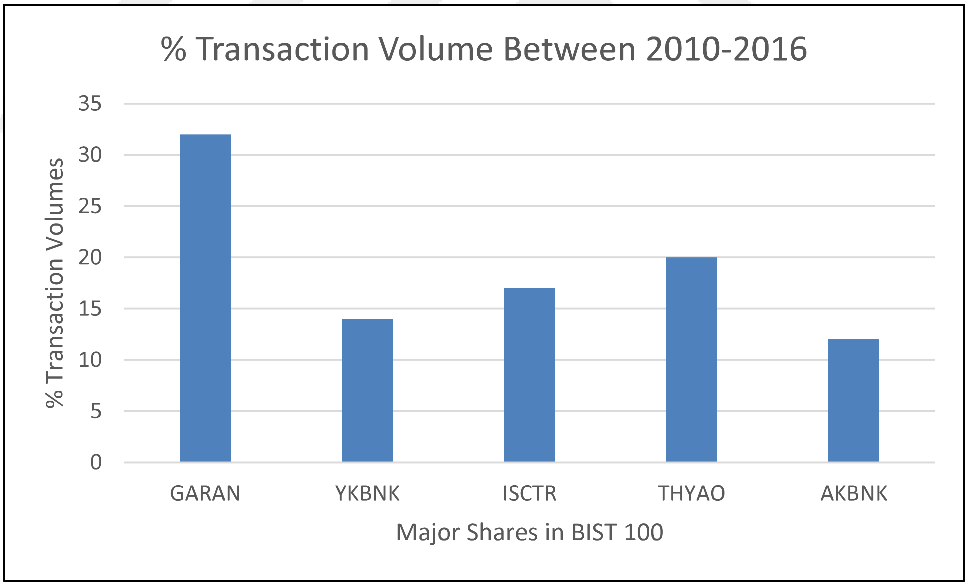

Figure 4.1. Transaction volumes of major shares in BIST100

Since more than 65 percent of the transactions in BIST100 is conducted by the foreign investors, we can safely say that BIST 100 is an integral part of the global financial system. Consequently, we include the 'Purchasing Managers' Indices (PMI) and Non-Manufacturing Indices (NMI) of countries such as USA, EU, China and Germany with prominent global GDP share. Japan NMI is excluded due to lack of statistically significant correlation at 0.01

level with GARAN closing price. Stock exchange index levels (Dow Jones, DAX, SHANGHAI, NIKKEI, SP500) pertaining to those countries with statistically significant correlation at 0.01 level to GARAN closing price are also included in the ANN model to accommodate factors other than PMI and NMI (See Table 4.1) . Another variable is the Fed Funds Rate that has particularly large impact on developing countries. All data used in the ANN model are acquired from Bloomberg server. [62]

Table 4.1. Correlation analysis between input variables and the target

Pearson Correlation Coefficient									
	German PMI	**EU PMI**	**China PMI**	**German NMI**	**USA NMI**	**NIKKEI**	**EU NMI**	**Japan NMI**	**USA PMI**
Correlation with GARAN	0.180	0.116	0.298	0.280	0.097	0.380	0.405	0.149	0.328
	SSE	DAX	DOW	S&P 500	BIST100	BOND	FRED	ECBM	
Correlation with GARAN	0.155	0.530	0.525	0.512	0.882	0.597	0.321	0.107	

Legend:*statistically significant at 1 percent.

All inputs are inserted in the model in terms of their daily rate of change. That is, the rate of change of a variable on day t is defined as; $100*(P_{t-1}-P_{t-2}/P_{t-2})$, where t is the index for days. Some of the inputs such as PMI, NMI and FED interest rate do not change daily. Their rates of change remain zero until the next time they are announced. On the other hand, all stock exchange indices change daily and their rates of change are placed in the ANN model. The Asian stock exchange index rate of change is calculated by the difference between days t and t-1, and the US index rate of change is calculated by the difference between days t-2 and t-3. As for the target, GARAN closing price, the rate of change is converted into a binary value: If rate of change is negative, the target value is -1, otherwise, the target value is set to 1.

Due to time zone differences, on a given day t at session closure time in Turkey, we provide definitions of the input variable and output variable below.

4.2. MODEL A

In the proposed Model A, we concentrate on input variables that reflect macroeconomic outputs of leader countries and their stock exchange indices. We include the Purchasing Managers Indices (PMI) and Non-Manufacturing Indices (NMI) of countries leading the global economy, i.e., USA, the EU, China, Japan and Germany. Prominent stock exchange indices included in Model A are the DOW, DAX, SSE, NIKKEI, and S&P500. The reason behind our motivation is that more than 65 percent of the transactions in the BIST100 stock exchange are conducted by foreign investors, and the state of the global financial system certainly impacts GARAN share price. In Table 4.1, we present the correlation coefficients between Model A's input variables and GARAN closing price during the years 2010-2016 where all correlations are statistically significant. In Model A, we also include as an input the 5-year ahead expected USA inflation rate denoted here as FRED, because it is an indicator of the Federal Reserve Bank's potential tightening of monetary policy that affects liquidity and investment on stocks. In a similar fashion, the decisions made in the European Central Bank's (ECB) monetary policy meetings also impact European economies as well as the peripheral economies like Turkey. We note that Turkish stocks are affected by approaching ECB meetings even when ECB is not expected to make a change in its monetary policy. Therefore, we include an input variable that counts the number of days till the next ECB meeting, and denote it as ECBM.

As for local variables, we include the previous BIST100 index level, the trend in GARAN share price movement and the Turkish bond index. All data used in the proposed models are acquired from Bloomberg server whose address is https://www.bloomberg.com/professional /solution/bloomberg-terminal/".

Below, we provide brief definitions of the input variables and the target.

Purchasing Managers Index (PMI): PMI is a monthly indicator of the economic health of the manufacturing sector that is calculated using new orders, inventory and production levels, supplier deliveries and the employment environment. The PMIs for USA, Germany, EU and China are included in Model A. Japan's PMI is excluded due to lack of statistically significant correlation with GARAN's price.

Non-Manufacturing Index (NMI): NMI is an index based on surveys of a large number of non-manufacturing firms' purchasing and supply executives. NMI tracks economic data and it is a composite diffusion index based on national survey. NMI gets more attention than PMI due to its seasonally adjusted figures for several of its components.

Federal Reserve Bank interest rate: US Federal Reserve funds interest rate (denoted as FEDI here) is the interest rate at which banks lend reserve balances to other banks overnight. Banks use FEDI rate to determine all other short term interest rates. The Federal Reserve Bank Open Market Committee (FOMC) decides on the FEDI on pre-announced meeting dates.

The FOMC changes the FEDI to control inflation and maintain healthy economic growth. The FOMC members watch economic indicators and other signs of inflation or deflation/recession.

When the Federal Reserve Bank raises rates, it's called contractionary monetary policy. When FEDI current value goes higher, it means banks are less likely to borrow money in order to keep their reserves at a safe level. Thus, the credits and other economic contracts become more expensive.

When the Federal Reserve Bank lowers the rate, the opposite occurs. Credit card rates drop, so consumer expenditure rises. That's called expansionary monetary policy. The stock exchange is usually in favor of this policy.

Stock Exchange Indices: Stock Exchange Index levels are calculated as the sum of the current prices making up the index divided by a divisor. Stock exchange indices such as Nikkei, Shanghai, DAX, Dow Jones, S&P500 affect BIST100 index.

The Target of the ANN: The target of the ANN is the direction of the next day's closing price for GARAN stock.

In this study, ANN receives all input and target data during the years 2007-2016 for training. Prediction of GARAN price direction is carried out for the second half of 2016 and the first half of 2017 which have the characteristics of bear and bull markets, respectively.

4.3. MODEL B

The proposed Model B involves only a subset of the variables that take place in Model A. These are the Bond Index, Trend, BIST100, FRED and ECBM. These variables are explained in Model A.

Turkish Government Bond Index and BIST100 trend: The rate of change in the daily value of Bond index and the trend of BIST100 index are the input variables in the proposed DT model. These are explained in Öztekin et al.'s SVM model [3].

St Louis Federal Reserve Bank 5 year Inflation Anticipation (FRED): The Federal Reserve Bank of St. Louis maintains a database that has more than 421,000 economic time series from 81 sources. These include consumer price indices, employment rates, exchange rates, interest rates, GDP data, etc.

The time series that the DT model acquires as input is the 5-year ahead expected inflation rate that utilizes the 10 an d 5 year constant and inflation indexed treasury yields. Here, we call the 5 year ahead expected inflation rate as FRED. The calculation of FRED is provided in [63].

Rather than using the FED rate (as in the proposed ANN Model) that changes seldom throughout the year, we propose to use FRED expected 5-year ahead inflation rate announced by St Louis Federal Reserve Bank daily [63] FRED changes daily based on macroeconomic indicator announcements and market pricing.

European Central Bank Meeting Anticipation (ECBM): This input counts the remaining number of days to the next ECB monetary policy meeting.

European Central Bank monetary policy meeting impacts EU country economies when they result in expansion or contraction policies. Since Turkey's 50 percent of foreign trade is with the EU, Turkey's economy is also affected by ECB policies. Furthermore, the majority of foreign direct and indirect financial investments in Turkey come from EU countries. Thus, we include an input variable that counts the number of days left till the next ECB meeting. We expect that the model will show sensitivity to the coming ECB announcements. Another advantage to this kind of timer is that it is not only sensitive to the Euro interest rate, but to all kinds of announcements that will have an effect on the Turkish economy. After the meeting is over, the timer starts again by returning to 0 (See Table 4.2).

Table 4.2. Number of days left till the next ECB meeting

Date	ECBM
7.7.2010	27
8.7.2010	28
9.7.2010	1
12.7.2010	4
13.7.2010	5
14.7.2010	6
15.7.2010	7
16.7.2010	8

An illustrative input-target data set for DT is provided in Table 4.3.

Table 4.3. An illustrative input-target data set for DT

Date	Bond Rate of Change	Trend	ECBM	FRED	GARAN price direction
2.12.2010	-0,0690798	2	28	2,7	1
3.12.2010	-0,1564729	0	1	2,72	-1
6.12.2010	-0,0091899	0	4	2,72	1
7.12.2010	-0,0505398	1	5	2,79	1
8.12.2010	-0,0367376	0	6	2,7	-1
9.12.2010	0,0826294	-1	7	2,63	-1
10.12.2010	0,0872921	-2	8	2,67	-1
13.12.2010	-0,229916	0	11	2,7	1
14.12.2010	-0,1926870	1	12	2,75	-1
15.12.2010	-0,0228948	0	13	2,85	-1

4.3.1. Öztekin et.al’s Model Implemented for GARAN

In order to compare the performance of the proposed Models A and B with the literature, we reproduce the recent model presented by Oztekin et al. [3] that predicts the BIST100 index movement direction using BPN and SVM. Here, we denote this model the OZ_BIST model. We also adapt the model to target GARAN share price direction by calculating Trend variable for GARAN share price rather than the BIST100 index and by replacing the NASDAQ index with the S&P500 index. The latter change is made, because NASDAQ index has no correlation with GARAN, but S&P500 index has a statistically significant correlation. We denote this model OZ_GARAN. Similar to Model A and B, all model inputs except Trend are represented in the model in terms of rate of change.

In Table 4.4, we summarize the inputs of Models A and B proposed here as well as those of Oztekin et al. [3] adapted for GARAN (OZ_GARAN) and the original Oztekin et al. model targeting BIST100 (OZ_BIST).

4.4. SUMMARY OF THE MODELS

We provide a tabulated summary of the models proposed along with their specifics in Table 4.4..

Table 4.4. Summary of proposed learning models

MODEL A		**MODEL B**		**Öztekin et al. (2016) OZ_GARAN**		**Öztekin et al. (2016) OZ_BIST**	
INPUTS	**TARGET**	**INPUTS**	**TARGET**	**INPUTS**	**TARGET**	**INPUTS**	**TARGET**
USA PMI	GARAN	BOND INDEX	GARAN	S&P500 (new)	GARAN	NASDAQ	BIST100
GERMANY PMI		TREND		USD/XAU		USD/XAU	
EU PMI		BIST100		USD/TRY		USD/TRY	
CHINA PMI		FRED		EURO/TRY		EURO/TRY	
USA NMI		ECBM		TREND (GARAN)		TREND (BIST100)	
GERMANY NMI				BIST100		BIST100	
EU NMI				BOND INDEX		BOND INDEX	
JAPAN NMI							
SSE							
DAX							
DOW							
S&P500							
BOND INDEX							
TREND							
BIST100							
FRED							
ECBM							

5. COMPUTATIONAL RESULTS

5.1. VALIDATION OF THE MODELS

A K-fold cross-validation is a validation technique that allows the user to make fine tuning to parameters to the models. In this study, all three models are tested by using 5-fold cross-validation. The cross-validations take a subset of the training data set compare them with the training data output. Therefore when thee cross-validation results are not at the desired level, (below 50 percent) models assumed to be violated. By constantly checking the validation result and the parameter values, the models prediction accuracy might be improved without overfitting. Since our data set is time series the training data is divided into five equal groups consecutively, in which every one of them is compared with each other separately when one subset represents the validation set and the other four subsets represent the training data for the model. (See Figure 5.1).

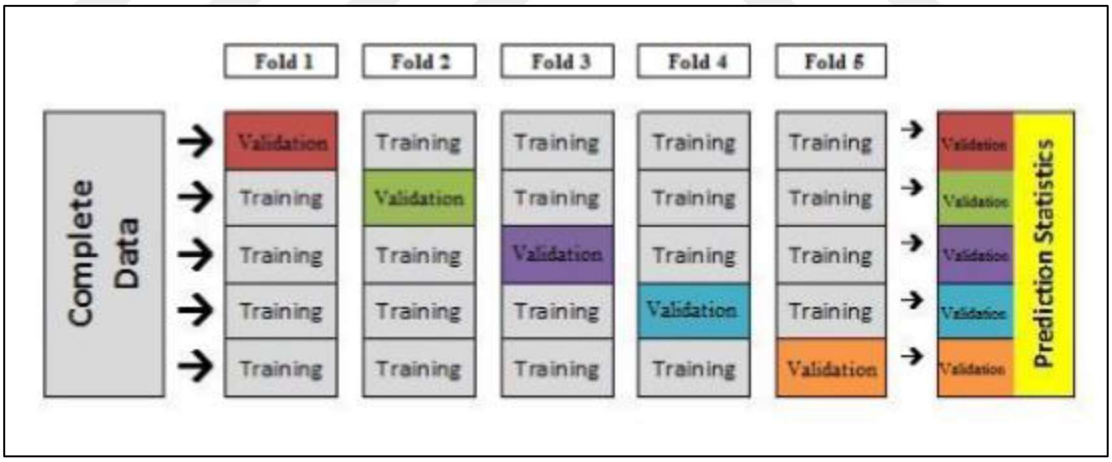

Figure 5.1. Consecutively separated data set for 5-fold validation

In the five-fold cross-validation, this process is repeated 5 times, each time using a different subset as validation set and the rest as training data. After all iterations are completed, the hit rates of all five iterations are considered and the average of the resulting weight values is stored.

5.2. PERFORMANCE MEASUREMENTS

The outcome of each model measured in terms of the hit rate that is the percentage of times. The direction of the next day's closing price is predicted correctly.

The following equation shows how the hit rate is calculated.

$$\text{Hit Rate} = \frac{\text{corrected no of predicted labels}}{\text{total number of testing labels}} \text{x} 100 \qquad \text{(Eq 5.1)}$$

Apart from the hit rate, other performance measurements investigate the bias of an algorithm, such as, True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). These measures are defined as follows (See Table 5.1).

Table 5.1. Classes of the performance measurements

Two class classification: Confusion Matrix		**Predicted Class**	
		Positive	**Negative**
Actual Class	Positive	True Positive TP	False Positive FP
	Negative	False Negative FN	True Negative TN

TP: Number of samples classified as true while they actually were true.

FP: Number of samples classified as true while they actually were false.

FN: Number of samples classified as false while they actually were true.

TN: Number of samples classified as false while they actually were false.

5.3. TRADING STRATEGY

In order to monetize the buy/sell signals generated by each model, we implement a trading strategy where at the end of each day we decide to buy stocks if the signal is up, and sell stocks at hand if the signal is down for the next day (See Figure 5.2).

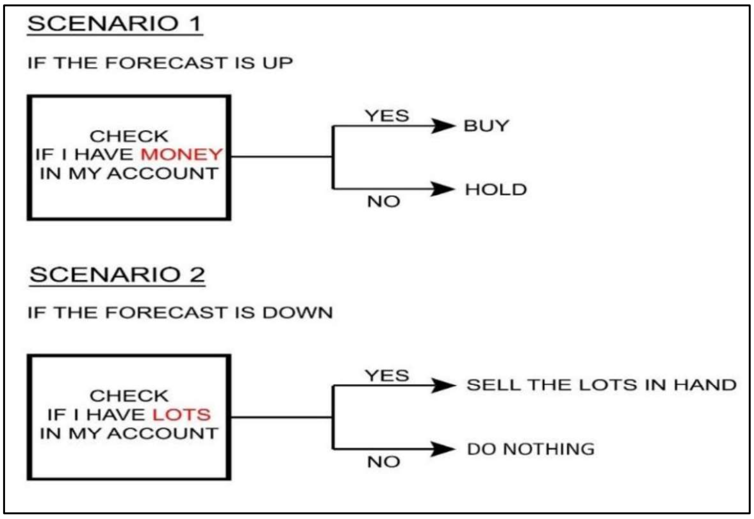

Figure 5.2. Trading strategy

The following assumpstion are made for trading:

-Bank account is 1600 TL.

-Each transaction is 100 lots.

-Short-Sell is not allowed.

-Commission for each transaction is 0,004.

-Share can be sold from it's closing price.

5.4. EXPERIMENTAL RESULTS

Each model is tested using SVM, BPN and DT methods. The results of Models A and B are presented in Table 5.2. We provide the percentage of correctly predicted labels, i.e., the hit rate, HR, the percentage of correct upward movement predictions, TP, and correct downward movement predictions, TN, as well as the percentile rate of return on investment and computation times. We also report the rate of return on investment for the Buy and Hold and fixed income investment options.

In Table 5.2, we observe that the best prediction accuracy is obtained by DT in both Models A and B. Model B validation results are slightly better than Model A's for all three methods. However**,** SVM and BPN's predictions are slightly better in Model A than in Model B affecting the return on investment. In DT's predictions and earnings, the reverse is true, Model B's results including validation, prediction and earnings are all better than those of Model A. Since DT is a tree based method and since Model B has a lower number of variables than Model A, DT performs better in the simpler model. DT's prediction accuracy is significantly superior to those of BPN and SVM in both models during both validation and prediction phases.

In Table 5.2, we observe that in both proposed models and in all methods, TN accuracy is higher than TP accuracy. This is important, because when TN accuracy is low, the investor actually loses money. On the other hand, when TP accuracy is lower, the investor loses a profit opportunity. Since DT has the best prediction accuracy, its rate of return on investment is also significantly higher than that of BPN and SVM. If the investor buys and holds GARAN share during the testing period, he/she makes a profit of 25.5 percent, however, DT's return on investment in both Model A and Model B outperforms the Buy & Hold option by about 13 percent and 20 percent, respectively. BPN's and SVM's return on investment rates are very poor due to their lower prediction accuracies despite the fact that their performance were not too inferior in validation phase. Even the fixed income return during the testing period provides better profit than BPN and SVM in Model B, but not in Model A. Using Model A, SVM and BPN can return almost triple the fixed income profit.

Table 5.2. Performance of model A and model B

	MODEL A						**MODEL B**					
	HR %	**TP %**	**TN %**	**Validation HR**	**Return Rate %**	**Comput. Time (CPUsec.)**	**HR %**	**TP %**	**TN %**	**Validation HR**	**Return Rate %**	**Comput. Time (CPUsec.)**
SVM	59	28	31	65	10	9	54	26	28	69	2.2	7
BPN	56	25	31	59	13	1011	54	24	30	61	2.3	354
DT	77	33	40	72	38	8	79	36	43	82	45	4
Buy & Hold					25.5						25.5	
Fixed Income					4.23						4.23	

An important point to note is that due to the extensive training phase, BPN requires a much higher amount of computation time (about 16 CPU minutes for Model A) than SVM and DT (less than 10 CPU seconds). These programs are re-run every Friday near the end of the session to make an investment decision for Monday. Therefore, BPN's longer computation time may become a liability.

In Table 5.3, we present the results of Oztekin et al.'s [3] OZ_BIST and OZ_GARAN models. Unfortunately, OZ_GARAN model performs poorly despite the fact that the NASDAQ index that is not correlated with GARAN has been replaced by the S&P500 index. The prediction accuracy of OZ_GARAN model is about 50 percent. Unlike Models A and B, the OZ_GARAN model is better in predicting positive movement (TP) than negative movement (TN). The same observation holds for the OZ_BIST model. This feature is also mentioned by Oztekin et al. (2016). Another surprising outcome observed in Table 5.3 is that BPN is not able to classify output labels when OZ_BIST model is utilized (all labels are decided as an upward movement). We could not find a good reason for this outcome. DT performs very poorly when used with OZ_BIST model. However, SVM's performance with OZ_BIST model is almost as good as our best result in Table 5.2, DT/Model B. Yet, the rate of return on investment achieved by SVM/OZ_BIST is lower than that of DT/Model B and compatible with our second best result, DT/Model A. The reason might be that GARAN share prices up/down movement range is higher than that of BIST 100 index. As mentioned previously, having a better negative movement prediction accuracy, TN, also reduces losses.

Table 5.3. Performance of OZ_BIST and OZ_GARAN models

	OZ_GARAN Model						**OZ_BIST Model**					
	HR %	**TP %**	**TN %**	**Validation HR %**	**Return Rate %**	**Comput. Time (CPUsec.)**	**HR %**	**TP %**	**TN %**	**Validation HR%**	**Return Rate %**	**Comput. Time (CPUsec.)**
SVM	50	31	19	65	4.2	7	78	44	34	81	38.9	8
BPN	44	34	10	53	2.1	667	FAILED TO CLASSIFY					1155
DT	53	35	18	70	1.7	6	36	26	10	62	-6.2	7
Buy and Hold					25.5						25.5	
Fixed Income Return					4.23						4.23	

In order to check if the differences of results between the three best performing combinations DT/Model A, DT/Model B and SVM/OZ_BIST are statistically significant, we ran McNemar's difference test on binary validation and prediction outputs and paired t-test for the rate of return on investment outputs. In Table 5.4, first we compare Model A and Model B. We find that there is no statistically significant difference between the predictions obtained by DT using Models A and B, but the differences between validation and earnings results are statistically significant. Next, we check for the differences between the best two performers, DT/Model B and SVM/OZ_BIST. Though no statistically significant difference exists between both validation and prediction results, the difference in the return on investments is statistically significant.

Table 5.4. Paired difference tests on validation, prediction and earnings data resulting from DT implemented on models A and B

Validation Data		**Prediction Data**		**Earnings Data**	
McNemar Test on DT Results Model A vs. Model B	Exact Significance (2-sided)	McNemar Test on DT Results Model A vs. Model B	Exact Significance (2-sided)	Paired t-test on DT Results Model A vs. Model B	Exact Significance (2-sided)
	0.028*		0.720		0.028*
Validation Data		**Prediction Data**		**Earnings Data**	
McNemar Test on DT/Model B vs. SVM/ OZ_BIST	Exact Significance (2-sided)	McNemar Test on DT/Model B vs. SVM/ OZ_BIST	Exact Significance (2-sided)	Paired t-test on DT/Model B vs. SVM/ OZ_BIST	Exact Significance (2-sided)
	0.719		0.706		0.029*

In Table 5.5, we present previously reported prediction accuracy results targeting BIST100 index direction by several authors mentioned in the literature survey: Kara et al. [31], Diler [28], Altay and Satman [29] and Oztekin et al. [3]. For comparison purposes, in Table 5.5, we also summarize the prediction accuracies illustrated in Tables 5.2 and 5.3 (Models A, B, OZ_BIST and OZ_GARAN). Among Oztekin et al.'s [3] cited results ANFIS (an adaptive neuro-fuzzy inference method that uses fuzzy if-then rules) is their least successful method.

Kara et al. [31].'s model performs worst under OLS. Among these previously cited results, the best prediction accuracy is obtained by Kara et al. [31] using BPN with almost 76 percent hit rate. SVM results of both Kara et al. [31] and Oztekin et al.[3] models are cited to be second best with 72 percent accuracy. As for the models OZ_BIST and OZ_GARAN that we reproduced here, SVM/OZ_BIST provides a better than cited prediction accuracy of 78 percent, however, the return on investment is about 38 percent and comparable to that of proposed DT/Model A, but inferior to that of DT/Model B. BPN/OZ_BIST did not return any result and it failed to classify the direction of movement of BIST100. DT/OZ_BIST resulted in a very inferior prediction accuracy of 36 percent. As illustrated in Table 5.5, OZ_GARAN model also results in an inferior prediction accuracy. When we compare the proposed DT/Model B results with SVM/OZ_BIST results (the two best achievements among all cited/proposed methods and models), we observe that though their prediction accuracies are similar, the return on investment is higher for Model B/DT and the difference is statistically significant.

Table 5.5. Summary of prediction accuracies (hr) of previous and proposed models. (ftc: failed to classify)

TARGET: BIST100 Direction										
Kara et al. (2011)			**Oztekin et al. (2016)**			**OZ_BIST**			**Diler (2003)**	**Altay & Satman (2005)**
OLS	**BPN**	**SVM**	**ANN**	**ANFIS**	**SVM**	**SVM**	**BPN**	**DT**	**BPN**	**BPN**
55%	76%	72%	60%	52%	72%	78%	FTC	36%	61%	58%
TARGET: GARAN Share Price Direction										
MODEL A			**MODEL B**			**OZ_GARAN**				
SVM	**BPN**	**DT**	**SVM**	**BPN**	**DT**	**SVM**	**BPN**	**DT**		
59%	56%	77%	54%	54%	79%	50%	44%	53%		

6. CONCLUSION

In this study, we develop two novel models that generate 5 days ahead buy/sell signals for GARAN equity share listed in Borsa Istanbul Stock Exchange, BIST100. GARAN is the top traded BIST100 share in terms of volume. The first model focuses on global macroeconomic indicators and leading stock exchange indices as well as local market attributes and USA and EU central banks' potential monetary policy actions. The second model includes only a subset of the first model's input variables and focuses on central bank influence and local market attributes, thereby reducing the number of variables.

Both models are tested using three machine learning methods, namely, BPN, SVM and DT. Among them, the best prediction accuracy and return on investment is obtained by DT algorithm. DT works more efficiently on the simplified second model, but there seems to be no statistically significant difference between the two models' prediction accuracies. However, their returns on investment differ.

The results of the proposed models are compared with other models cited in the literature and one such previous model reproduced here. The model reproduced here originally targets BIST100 index, however, here, we also adapt the model to target GARAN share price movement. We discover that this model works well for the BIST100 index but not for GARAN share. Furthermore, its return on investment achieved for the BIST100 index is lower than that of DT algorithm's achievement for GARAN using the second model.

REFERENCES

1. White H. Economic prediction using neural networks: the case of IBM daily stock returns. *Neural Networks, 1988 IEEE International Conference on*; 1988: IEEE.

2. Thenmozhi M. Prediction stock index returns using artificial neural networks. *Delhi Business Review X*. 2006; 7(2):59-69.

3. Öztekin A, Kızılaslan R, Freund S, Iseri A. A data analytic approach to forecasting daily stock returns in an emerging market. *European Journal of Operational Research*. 2016; 253(3):697-710.

4. Kim KJ, Oh KJ, Han I. Neural network forecasting of stock price index to integrate change- point detection with genetic algorithms.[cited 2017 10 December] Available from: http://koasas.kaist.ac.kr/bitstream/10203/5360/1/2000-160.pdf

5. Tilakaratne C, Morris SA, Mammadov MA, Hurst CP. Predicting stock market index trading signals using neural networks. [cited 2017 10 December] Available from: https://pdfs.semanticscholar.org/0604/4dd324e193bbc0e2fd9ddf8edfb21bb4a932.pdf.

6. Quah T. Using neural network for DJIA stock selection. *Engineering Letters*. 2007;15(1):126-133.

7. Majumder M, Hussian MD. Prediction of Indian stock market index using artificial neural network. [cited 2017 10 December] Available from: https://nseindia.com/ content/ research/FinalPaper206.pdf.

8. Yao JT, Tan CL. Guidelines for financial prediction with artificial neural networks. [cited 2017 10 December] Available from: http://citeseerx.ist.psu.edu/viewdoc/ download?doi=10.1.1.18.6874&rep=rep1&type=pdf.

9. Mantri JK, Gahan P, Nayak BB. Artificial neural networks- an application to stock market volatility. *International Journal of Engineering Science and Technology*. 2010;2(5):1451-1460.

10. Zhang D, Jiang Q, Li X. Application of neural networks in financial data mining, *International Journal of Computational Intelligence*. 2005;1(2):106-109.

11. Saad EW, Prokhorov DV, Wunsch DC. Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Transactions on Neural Networks.* 1998;9(6):1456-1470.

12. Naeini MP, Taremian H, Hashemi HB. Stock market value prediction using neural networks. *Computer Information Systems and Industrial Management Applications (CISIM), 2010 International Conference on*; 2010: IEEE.

13. Hornik K, Stinchcombe M, White H. Multilayer feed forward networks are universal approximators. *Neural Networks.* 1989; 2(5):359-366.

14. Li Y, Ma W. Applications of artificial neural networks in financial economics: a survey. *Computational Intelligence and Design (ISCID), 2010 International Symposium on*; 2010: IEEE.

15. Kimoto T, Asakawa K, Yoda M, Takeoka M. Stock market prediction system with modular neural networks. *Neural Networks, 1990 IJCNN International Joint Conference on;* 1990: IEEE.

16. McCluskey PC. Feedforward and recurrent neural networks and genetic programs for stock market and time series forecasting. [cited 2017 10 December] Available from: http://www.smartquant.com/references/NeuralNetworks/neural1.pdf

17. Schierholt K, Dagli C. Stock market prediction using different neural network classification architectures. *Computational Intelligence for Financial Engineering, Proceedings of the 1996 IEEE/IAFE Conference on;* 1996: IEEE.

18. Kim K, Han I, Chandler JS. Extracting trading rules from the multiple classifiers and technical indicators in stock market. *KMIS '98 International Conference on;* 1998.

19. Kutsurelis EJ. Prediction financial markets using neural networks: an analysis of methods and accuracy. *Master of Science Thesis.* California: Naval Postgraduate School; 1998.

20. Suan ST, Chye KH. Neural network applications in accounting and business. *Accounting and Business Review*, 1997; 4(2): 297-317.

21. Dutt SD, Ghosh D. A note on the foreign exchange market efficiency hypothesis. *Journal of Economics and Finance.* 1999; 23(2): 157-161.

22. Burgess AN. A computational methodology for modelling the dynamics of statistical arbitrage. *Phd thesis.* London: Decision Technology centre- London Business School, 2000.

23. Yamashita T, Hirasawa K, Hu J. Application of multi-branch artificial neural network stock market prediction. *Neural Networks, IJCNN '05, Proceedings. 2005 IEEE International Joint Conference on*; 2005: IEEE.

24. Majumder M, Hussian A. Prediction of indian stock market index using artificial neural network. [cited 2017 12 December] Available from: https://nseindia.com/content /research/FinalPaper206.pdf.

25. Charkha P. Stock price prediction and trend prediction using neural networks. *First International Conference on Emerging Trends in Engineering and Technology*; 2008: IEEE.

26. Mandziuk J, Jaruszewicz M. Neuro-evolutionary approach to stock market prediction. *Neural Networks, IJCNN 2007, International Joint Conference on;* 2007: IEEE.

27. Mehrara M, Moeini A, Ahrari M, Ghafari A. Using technical analysis with neural network for prediction stock price index in Tehran stock exchange. *Middle Eastern Finance and Economics- London EuroJournals*. 2010; 6(6):50-61.

28. Diler AI. Predicting direction of ISE national-100 index with back propagation trained neural network. *Journal of Istanbul Stock Exchange.* 2003;7(25-26):65-81.

29. Altay E. , Satman MH. Stock market forecasting: artificial neural network and linear regression comparison in an emerging market. *Journal of Financial Management \& Analysis*. 2005;18(2):18-33.

30. Yümlü S, Gürgen F, Okay N. A comparison of global, recurrent and smoothed-piecewise neural models for Istanbul stock exchange (ISE) prediction. *Pattern Recognition Letters*. 2005;26(13): 2093-2103.

31. Kara Y, Boyacioglu MA, Baykan ÖK. Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange, *Expert Systems with Applications*. 2011;38(5):5311-5319.

32. Trafalis TB, Ince H. Support vector machine for regression and applications to financial forecasting. *Neural Networks, IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*; 2000: IEEE.

33. Tay FEH, Cao LJ. Application of support vector machines in financial time series forecasting. *Omega*. 2001; 29(4):309–317.

34. Van Gestel T, Suykens JAK, Baestaens DE, Lambrechts A, Lanckriet G, Vandaele B, DeMoor B, Vandewall J. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on Neural Networks*. 2001; 12(4):1398–1402.

35. Yang H, King I, Chan L. Non-fixed and asymmetrical margin approach to stock market prediction using support vector regression. *Neural Information Processing, ICONIP '02. Proceedings of the 9th International Conference on;* 2002: IEEE.

36. Liang Y, Sun Y. An improved method of support vector machine and its application to financial time series forecasting. *Progress in Natural Science*. 2003; 13(9):696–700.

37. Kim KJ. Financial time series forecasting using support vector machines. *Neurocomputing*. 2003;55(1-2):307–319.

38. Yang H, Huang K, Chan L, King I, Lyu MR. Outliers treatment in support vector regression for financial time series prediction. *Lecture Notes in Computer Science, Proceedings of the International Conference on Neural Information Processing;* 2004: ICONIP.

39. Abraham A, Philip NS, Saratchandran P. Modeling chaotic behavior of stock indices using intelligent paradigms. *Neural, Parallel & Scientific Computations*. 2003: 11(1/2): 143–160.

40. Bao Y, Lu Y, Zhang J, Forecasting stock price by SVMS regression, *Lect. Notes Comput. Sci.* 2004;3192:295–303.

41. Huang W, Nakamori Y, Wang SY. Forecasting stock market movement direction with support vector machine. *Computers & Operations Research*. 2005; 32 (10): 2513– 2522.

42. Bao YK, Liu ZT, Guo L, Wang W. Forecasting stock composite index by fuzzy support vector machines regression. *Machine Learning and Cybernetics. 2005. Proceedings of 2005 International Conference on*; 2005: IEEE.

43. Jung J, Reggia JA. The automated design of artificial neural networks using evolutionary computation, *Success in Evolutionary Computation*. 2008:19-41.

44. Muh-Cherng W, Sheng-Yu L, Chia-Hsin L. An effective application of decision tree to stock trading. *Expert Systems with Applications.* 2006; 31(2): 270-274.

45. Wang JL, Chan SH. Stock market trading rule discovery using two-layer bias decision tree. *Expert Systems with Applications.* 2006; 30(4): 605-611.

46. Sun J, Li H. Data mining method for listed companies' financial distress prediction. *Knowledge-Based Systems*. 2008; 21(1):1-5.

47. Lu CL, Chen TC. A study of applying data mining approach to the information disclosure for Taiwan's stock market investors. *Expert Systems with Applications*. 2009; 36(2): 3536–3542.

48. Gorunescu F. *Data mining: concepts, models and techniques*. Berlin: Springer, 2011.

49. Tunçhan C. *Modern sezgisel teknikler ve uygulamaları*, İstanbul: Papatya Yayıncılık, 2008.

50. Larose DT. *Discovering knowledge in data: An introduction to data mining*, New Jersey: Wiley-Interscience, 2005.

51. Reed R, Marks RJ. *Neural smithing: Supervised learning in feedforward artificial neural networks*, Massachusetts: The MIT Press, 7, 1999.

52. Mehrotra K, Mohan C, Ranka S. *Elements of artificial neural networks(complex adaptive systems series)*, Massachusetts: The MIT Press, 1997.

53. Negnevitsky M. *Artificial intelligence a guide to intelligent systems.* London: Pearson Education Limited; 2005.

54. Baker HK, Smith DM. In search of a residual dividend policy. *Review of Financial Economics*. 2006;15(1):1-18.

55. Kriesel D. A brief introduction to neural networks, [cited 2017 10 December] Available from: http://www.dkriesel.com/_media/science/neuronalenetze-en-zeta2-2col-dkriesel com.pdf.

56. Munakata T. *Fundamentals of the new artificial intelligence: neural, evolutionary, fuzzy and more*. 2. Edition, London: Springer; 2008.

57. Kecman V. *Learning and soft computing: support vector machines, neural networks, and fuzzy logic models,* London: MIT press; 2001.

58. Öztemel E. *Yapay sinir ağları*, 2. Baskı, İstanbul: Papatya Yayıncılık; 2006.

59. Zurada MZ. *Introduction to artificial neural systems*. New-York:West publishing company; 1992.

60. Şenol D. Prediction of stock price direction by artificial neural network approach. *Master of Science Thesis.* İstanbul: Boğaziçi University; 2008.

61. Jung JY. Evolutionary Design of Artificial Neural Networks, *Ph.D. Thesis.* Washington DC: University of Maryland, 2007.

62. Bloomberg Terminal, [cited 2017 10 December] Available from: https://www.bloomberg.com/professional/solution/bloomberg-terminal/.

63. FRED Index, [cited 2017 10 December] Available from: https://fred.stlouisfed.org/series/T5YIFR.

APPENDIX A: OUTPUT OF MODEL B DT

Date	Closing Price	Garan Earning	Garan Real Dir	Garan Forecast DT	Money on Hand Model B
1.2.2017	8,11			77	1800
2.2.2017	8,28	0,17	1	1	1817
3.2.2017	8,56	0,28	1	1	1845
6.2.2017	8,34	-0,22	0	0	1867
7.2.2017	8,35	0,01	1	1	1868
8.2.2017	8,47	0,12	1	1	1880
9.2.2017	8,59	0,12	1	1	1892
10.2.2017	8,36	-0,23	0	0	1915
13.2.2017	8,55	0,19	1	1	1934
14.2.2017	8,53	-0,02	0	1	1932
15.2.2017	8,5	-0,03	0	0	1935
16.2.2017	8,62	0,12	1	1	1947
17.2.2017	8,75	0,13	1	0	1934
20.2.2017	8,65	-0,1	0	0	1944
21.2.2017	8,74	0,09	1	0	1935
22.2.2017	8,52	-0,22	0	0	1957
23.2.2017	8,55	0,03	1	0	1954
24.2.2017	8,42	-0,13	0	0	1967
27.2.2017	8,37	-0,05	0	0	1972
28.2.2017	8,35	-0,02	0	1	1974
1.3.2017	8,63	0,28	1	1	2002
2.3.2017	8,68	0,05	1	0	1997
3.3.2017	8,61	-0,07	0	1	1990
6.3.2017	8,76	0,15	1	1	2005
7.3.2017	8,76	0	1	0	2005
8.3.2017	8,58	-0,18	0	0	2023
9.3.2017	8,49	-0,09	0	0	2032
10.3.2017	8,67	0,18	1	1	2050
13.3.2017	8,63	-0,04	0	1	2046
14.3.2017	8,55	-0,08	0	0	2054
15.3.2017	8,6	0,05	1	1	2059
16.3.2017	8,67	0,07	1	1	2066
17.3.2017	8,65	-0,02	0	0	2068
20.3.2017	8,79	0,14	1	1	2082
21.3.2017	8,65	-0,14	0	0	2096
22.3.2017	8,57	-0,08	0	0	2104
23.3.2017	8,46	-0,11	0	0	2115
24.3.2017	8,65	0,19	1	1	2134

27.3.2017	8,59	-0,06	0	1	2128
28.3.2017	8,64	0,05	1	0	2123
29.3.2017	8,63	-0,01	0	0	2124
30.3.2017	8,66	0,03	1	1	2127
31.3.2017	8,59	-0,07	0	0	2134
3.4.2017	8,58	-0,01	0	1	2133
4.4.2017	8,71	0,13	1	1	2146
5.4.2017	8,75	0,04	1	0	2142
6.4.2017	8,7	-0,05	0	0	2147
7.4.2017	8,78	0,08	1	1	2155
10.4.2017	9,33	0,55	1	1	2210
11.4.2017	9,21	-0,12	0	0	2222
12.4.2017	9,25	0,04	1	0	2218
13.4.2017	9,13	-0,12	0	0	2230
14.4.2017	9,4	0,27	1	1	2257
17.4.2017	9,38	-0,02	0	1	2255
18.4.2017	9,43	0,05	1	1	2260
19.4.2017	9,22	-0,21	0	0	2281
20.4.2017	9,27	0,05	1	1	2286
21.4.2017	9,3	0,03	1	1	2289
24.4.2017	9,54	0,24	1	1	2313
25.4.2017	9,66	0,12	1	1	2325
26.4.2017	9,66	0	1	0	2325
27.4.2017	9,47	-0,19	0	0	2344
28.4.2017	9,59	0,12	1	1	2356
2.5.2017	9,61	0,02	1	0	2354
3.5.2017	9,47	-0,14	0	0	2368
4.5.2017	9,37	-0,1	0	1	2358
5.5.2017	9,45	0,08	1	1	2366
8.5.2017	9,36	-0,09	0	0	2375
9.5.2017	9,56	0,2	1	1	2395
10.5.2017	9,58	0,02	1	1	2397
11.5.2017	9,35	-0,23	0	0	2420
12.5.2017	9,3	-0,05	0	0	2425
15.5.2017	9,35	0,05	1	1	2430
16.5.2017	9,42	0,07	1	1	2437
17.5.2017	9,49	0,07	1	1	2444
18.5.2017	9,45	-0,04	0	0	2448
22.5.2017	9,54	0,09	1	1	2457
23.5.2017	9,69	0,15	1	1	2472
24.5.2017	9,77	0,08	1	1	2480
25.5.2017	9,71	-0,06	0	0	2486
26.5.2017	9,62	-0,09	0	0	2495

29.5.2017	9,64	0,02	1	0	2493
30.5.2017	9,53	-0,11	0	0	2504
31.5.2017	9,6	0,07	1	1	2511
1.6.2017	9,5	-0,1	0	0	2521
2.6.2017	9,64	0,14	1	1	2535
5.6.2017	9,54	-0,1	0	1	2525
6.6.2017	9,61	0,07	1	1	2532
7.6.2017	9,58	-0,03	0	0	2535
8.6.2017	9,52	-0,06	0	0	2541
9.6.2017	9,6	0,08	1	1	2549
12.6.2017	9,71	0,11	1	1	2560
13.6.2017	9,62	-0,09	0	0	2569
14.6.2017	9,7	0,08	1	1	2577
15.6.2017	9,63	-0,07	0	0	2584
16.6.2017	9,56	-0,07	0	0	2591
19.6.2017	9,75	0,19	1	1	2610
20.6.2017	9,75	0	1	0	2610
21.6.2017	9,75	0	1	1	2610
				% Earning	45%