Forecasting BIST100 Index with Neural Network Ensembles

Koray Beyaz[1]*, and Mehmet Önder Efe*[2]

[1]Department of Mechanical Engineering, Hacettepe University, Ankara 06800, Turkey
koraybeyaz@hacettepe.edu.tr
[2] Department of Computer Engineering, Hacettepe University, Ankara 06800, Turkey
onderefe@hacettepe.edu.tr

Abstract

This paper aims to provide a neural network-based approach to forecast the direction of movement of BIST 100 stock price index and investigates the difficulties of such an implementation. It is observed that a neural network implementation is highly sensitive to selection of features and optimization parameters such as learning rate. A methodology to overcome the difficulties of neural network implementations to financial time series is proposed in the paper. Several feature selection methods are employed to obtain a subset of the features that can be used in the training of any classification algorithm. The difficulties and benefits of using an ensemble of neural networks instead of a single neural network are also studied. Results have shown that the use of neural network ensembles yields promising results.

Keywords: Neural Networks, Ensemble, Bagging, Forecast.

1. Introduction

Time series analysis is the study of dependence between the future and the past observations in a sequence of observations. The objective of the time series analysis is to reveal this dependence and develop a predictive model to anticipate future behavior and trends. The model is then used for forecasting.

The fundamental tool for the time series analysis is the autoregressive integrated moving average model (ARIMA (p, d, q) where p, d and q stand for the number of autoregressive, integration and moving average parameters). The ARIMA model allows modeling nonstationary time series and can be extended further to model the seasonality of the series and exogenous inputs (SARIMAX). The famous Box-Jenkins approach is the most widely used procedure for parameter selection [1]. This methodology utilizes autocovariances to determine the dependencies between the future and the past observations of the time series and cross-covariances to determine the dependencies between the future observations and the exogenous variables. However, this method reveals only the linear relationships between the features and the output. Several studies have been devoted to modeling nonlinear time series such as financial time series. Kim shows that Support Vector Machines (SVM) can be used to predict financial time series [2]. Kaastra et. al. has used Artificial Neural Networks (ANN) for the modeling of financial time series and draw attention to the difficulties of neural network implementation in the context of financial time series analysis [3].

Zhang has proposed a hybrid ARIMA and ANN model for forecasting a time series where both linear and nonlinear dependencies exist [4]. Akbilgic et. al. has proposed a hybrid Radial Basis Function (RBF) neural network to forecast the direction of movement of Istanbul Stock Exchange National 100 (ISE100) [5].

The purpose of this study is to develop an ANN based architecture to forecast the direction of movement of a financial time series 20-days into the future (increase or decrease). The work of Akbilgic is chosen as a benchmark and the choice of the time series is BIST 100 (previously known as ISE100).

An important aspect in financial time series forecasting is the choice of features to build a predictive model. The choice of candidate features is a design decision and various choices are available such as stock market indices, news data, currency exchange rates and social media data. This study aims to provide a feature selection approach to choose the most of important features among the feature candidates. The candidate features are chosen to be several stock market indices.

Chen et. al. employs several feature selection methods in their study to improve the performance of support vector machines such as Random Forest and F-score [6]. In this study, random forest, extremely randomized trees and Gaussian radial basis kernels are applied for feature selection. The study also aims to investigate the usage of ensemble of neural networks to surpass the achievement of a predictor with a single neural network.

This paper is organized as follows. In the following section, an overview of feature selection methods is presented where mathematical concepts of each feature selection algorithm are introduced. In section 3, data preparation steps are explained, and the results of the feature selection methods are discussed. In section 4, different sampling methods are investigated in the training of neural networks. In the final section the results are presented with a discussion.

2. Overview of Feature Selection Methods

2.1. Feature Selection with Random Forest (RF)

Random forest is an ensemble learning algorithm proposed by Breiman and it is based on decision trees [7]. The algorithm is an improvement over bootstrap aggregation (bagging) algorithm. In bagging, for each tree in the ensemble, a subset of observations is randomly selected with replacement. Consider a training set of n observations and features. A subset of observations is selected such that there are $n' < n$ training vectors $\left(x^1, x^2, \dots x^{n'}\right)$ of length k. The specified number of trees are trained with the corresponding subset of training samples. The decision of the ensemble is decided upon by votes taken by the trees in the ensemble. Thus, the variance is reduced even though the individual trees in the ensemble might be over trained. Random forest algorithm further decreases the variance by also randomly selecting a subset of features such that each training vector

$\left(x^1, x^2, \dots x^{n'}\right)$ is of size $k' < k$. This algorithm also provides a method to determine the importance of the features.

Importance of a feature is calculated by the mean decrease impurity, where the impurity measure is taken to be Gini index as suggested by Breiman [8]. Gini index is the probability of a misclassification given the distribution of labels. For a binary classification problem, it is calculated as follows.

$$i(t) = 2p(1|t)p(2|t) \tag{1}$$

The decrease in impurity for a node in a tree is then defined as the difference between the impurity of the parent node and the impurity of the target node. Then, according to [9], the importance of a feature x is calculated as a weighted sum of the impurity decrease caused by the feature, averaged over all the trees in the ensemble such that:

$$\Delta i = i_p - i_c \tag{1}$$

$$Ip(x) = \frac{1}{N_T} \sum_T \sum_n p(t) \Delta i(t) \tag{2}$$

where N_T is the number of trees in the ensemble, n represents the nodes that uses the feature x, $p(t)$ is the probability of reaching the target node and $\Delta i(t)$ is the corresponding decrease in impurity caused by the target node. The feature selection is then performed such that any desired number of features are selected according to their importance values.

2.2. Feature Selection with Extremely Randomized Trees (ET)

Extremely randomized trees algorithm is proposed by Geurts et. al. and it is another tree-based ensemble classification and regression algorithm [10]. The motivation be-hind the algorithm is to decrease the variance by further randomizing the generation of the trees in the ensemble. There are two main differences between Extra-Trees and Random Forest algorithms. The first one is that the Extra-Trees algorithm does not create a subsample of observations to train the trees in the ensemble and uses all the data set for each tree in the ensemble. The second difference is that Extra-Trees chooses the split (threshold) of a node randomly whereas Random Forest selects the best split (optimal) for each node. The importance of the features is again chosen by employing the mean impurity decrease explained previously.

2.3 Feature Selection with Kernel Matrix

Wang et. al. proposed kernel matrices to be employed instead of covariance matrices since covariance matrices cannot capture the nonlinear relationships between features [11]. Although the study is a response to the increasing popularity of covariance matrices in computer vision, the covariance matrices are also widely employed in time series analysis but fail to reveal the nonlinear relationships between the features and the output and cannot be used when the time series at hand is highly complex and requires non-linearities to be modelled. Financial time series are good examples of such time series. Moreover, linear analysis does not allow the full capacity of neural networks to be employed. The first of the proposed kernels in their study is the Bhattacharyya kernel which is defined as follows,

$$\kappa(f_i, f_j) = \sqrt{\frac{2\sigma_i\sigma_j}{\sigma_i^2+\sigma_j^2}} \exp\left[-\frac{(\mu_i-\mu_j)^2}{4\left(\sigma_i^2+\sigma_j^2\right)}\right] \tag{3}$$

where f_i and f_j are two vectors, and μ_i and σ_i^2 are the mean and the variance of f_i. In [11], it is suggested that the simple Gaussian RBF kernel might also be used if one is not interested in the type of the nonlinear relationship between the features. Since in this case we are only interested in revealing the important features, Gaussian RBF kernel is chosen to determine the important parameters. It is defined as follows.

$$\kappa(f_i, f_j) = \exp\left(-\beta \left\|f_i - f_j\right\|^2\right) \tag{4}$$

where $\beta = 1/2\sigma_i^2$ and σ_i is the variance of f_i

3. Data Preparation and Feature Selection

The stock indices data are collected from Yahoo Finance and have been compared with the available data at the UCI repository provided by [5]. The collected data set consists of daily values of stock price indices of several countries worldwide such as BIST 100 (Turkey), NIKKEI (Japan), BOVESPA (Brazil), ERUS (Russia), KOSPI (Korea), SP (U.S.), TA (Israel) and several Morgan Stanley Capital International (MSCI) combined indices such as EEM (Combined Emerging Markets) and URTH (Combined Developed Markets). The number of main features collected is 22. The first observation is at January 2012 and the final observations is at February 2018. In order to make the data stationary, logarithmic return difference is applied to all the 22 main features such that:

$$r = \log\left(\frac{x(t_{i+1})}{x(t_i)}\right) \tag{5}$$

where t_i represents the day x and is any feature vector. Other features are created from the 22 main features such that there are lagged features, exponential moving average features, rolling mean features, rolling median features, rolling minimum features and rolling maximum features. Lagged features are created starting from 1-day lag values up to 10-day lag values and rolling features are created starting from 2-day rolling values up to 10-day rolling values. In total, 1232 features are obtained from the original 22 features. Output is chosen to be the direction of movement of BIST 100 Index 20-days ahead (0: decrease, 1: increase). Then the training and the test data sets are separated such that the test data set begins from June 2017. Thus, the training data set contains 1402 days and the test data set contains 182 days. The training data set is then normalized between the interval $[-1, 1]$. The same normalization parameters are then used to normalize the test data set in order to prevent information leakage from the test data set to the train data set. Feature selection methods are then applied to select 32 features out of 264 candidates. The four most important features for each method are shown in Table 1.

Table 1. The most important six selected features for each method.

Method	Feature 1	Feature 2	Feature 3	Feature 4
RBF	EEM Max 10-day	TSEC Max 10-day	EEM Max 9-day	TSEC Max 9-day
RF	SSEC Max 9-day	SSEC Max 8-day	KOSPI Max 10-day	SSEC Max 7-day
ET	SSEC Max 10-day	KOSPI Max 10-day	SSEC Max 8-day	KOSPI Max 9-day

It is observed that selections of the Random Forest, Extra-Trees and Gaussian RBF kernel methods share several common features. Table 1 shows that the most important features are 7-day, 8-day, 9-day and 10-day rolling maximum features of EEM (Emerging Markets Index), SSEC (Shanghai Composite Index), TSEC (Taiwan Stock Exchange Corporation Index) and KOSPI (The Korea Composite Stock Price Index).

Most of the selected features with any method are rolling maximum features. Some of the rolling minimum, rolling median and moving average features also appear in the top 32 selected features. However, none of the original 22 features are in the top 32 features and many studies tend to only use lagged variables as the features. The number of shared features for each feature selection algorithm is shown in Table 2.

Table 2. Number of shared features between each method

Method	ET	RF	RBF
ET	32	20	1
RF	20	32	3
RBF	1	3	32

4. Neural Network Implementations

4.1. Neural Network without Ensembles

In this study a multilayer perceptron (MLP) has been employed to predict the direction of movement of the BIST 100 Index 20-days ahead. The procedure of the study is summarized in Figure 1.

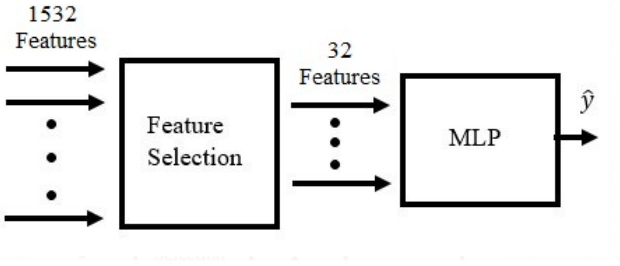

Fig. 1. Feature selection and neural network implementation

Two methods are employed in the training of the neural networks. For both methods, test data is selected as the final 15% of the data. In the first method, 30%-70% train/validation split has been performed. In the second method train/validation split (30%-70%) is again performed but stratification is also employed. Stratification allows that the training data to have the same label distribution as the original data.

The training of the neural network has been performed with the features selected using all three feature selection algorithms. Adam optimizer is used for the training with 0.001 learning rate and 10^{-4} decay.

Hyperparameter tuning is performed with a grid search for different number of hidden layers, number of neurons per layer, batch sizes, learning rates, training sample sizes and ratio of dropouts.

Dropout is a technique that prevents overfitting proposed by Srivastava et. al. [12]. A percentage of the neurons and the neurons that are connected to them are dropped out at each training subset and the resulting smaller network is trained. Neurons to be dropped are selected randomly. Therefore, 2^n small neural networks are trained which have common parameters. In the testing period, one scaled neural network is used. For a neuron that is present in the training period with probability p, the weights leaving that neuron are multiplied by in the test period. Therefore, instead of calculating the average prediction of 2^n networks, one network is used to emulate the output of 2^n networks. This is a very useful technique that helps prevent overfitting and more information can be obtained from the seminal paper, [12].

The results indicate that small neural networks with 16-24 neurons and two layers are enough for the problem at hand and increasing the layers and neurons mostly leads to overfitting and thus, a decrease in validation accuracy.

It is observed that the training procedure is very cumbersome for the specific problem. Convergence of the network is very sensitive to the changes in optimizer parameters such as the learning rate and decay. For each feature set obtained from each feature selection method, previously tuned parameters become less useful and fine tuning is required. Out of all four feature selection methods, features obtained from Extra-Trees algorithm has been the one that is the easiest to train the network and the one that yields the highest accuracy. Unlike other methods, with the features of the Extra-Trees algorithm, successful networks are achieved with different number of layers and neurons. Thus, it has become the choice of feature selection method for this study.

4.2. Neural Network Ensemble with Bagging

After obtaining the parameters that result in a successful neural network, an ensemble approach is taken. For the same neural network configuration (2 layers, 24 neurons, 0.2 dropout probability), five neural networks are trained to create the voters in an ensemble. Three methods are applied in the training of the neural networks in the ensemble.

The first method aims to achieve variation between the neural networks with bagging and it will be referred as “bagging without label balancing”. This method is applied as follows.

- A sample set is generated from the training data set with repetition. Only 50% of the original training data is selected as the training sample for each neural network. Therefore, variation between neural networks are achieved.
- Validation data is selected as random 30% out of the bag observations without repetition.

- A neural network is trained with the prepared data and the epoch with the best validation accuracy is added to the ensemble.
- The procedure is applied to each neural network in the ensemble.
- After the training of each neural network is completed, the ensemble is configured as follows.
- Neural networks are sorted by their validation accuracy.
- The best neural networks are chosen such that ≤ 5. This selection is applied by comparing the specificity and sensitivity of the neural networks.
- A voting procedure is applied.

Two voting methods are applied in the study. The first is a democratic voting where all predictors in the ensemble have equal share in the decision.

The second voting method is to apply a weighted average of the votes of the predictors in the ensemble. Predictions of each neural network are calculated, and the weighted average of the prediction is used as the prediction of the ensemble. In the second voting scheme the "best vs. rest" method is implemented. The weight of the single best neural network is taken to be $(n-2)$ so that the best network will be overruled only if all the other neural networks vote against the best network.

The second training method is a variation of the first one. Again, the bootstrap aggregating procedure is applied to train the neural networks in the ensemble. However, the validation set is selected such that it contains the same number of observations for each label. However, the data set is shuffled before the selection of training and validation data in order to achieve variation between the neural networks in the ensemble. The purpose of this method is to prevent overfitting that might be caused by the randomness of the selection. This method will be referred as "bagging with label balancing".

4.3. Neural Network Ensemble without Bagging

In this method, several neural networks with the same configuration are trained without the bagging procedure. Instead of sampling from the training data set with replacement, the whole training set is used to train each of the neural networks after the validation set is separated from the training data set. However, the training data set is shuffled for each network in order to obtain variation between the different neural networks. The label balancing procedure is not applied and thus. The size of the validation set is taken as the 30% of the total training data set.

In this method, it is often useful to split the data into training and validation sets such that both sets are balanced in terms of the number of observations for each label.

5 Results and Discussion

In some of the studies reported in the past, test data set is selected as some portion of all the data from the end of the time series. Although this approach is useful in a sense that it allows a similar environment to the real time application of the algorithm, using the test data set accuracy as the final decision metric has some inherent problems. The reason is that the test data set does not have the same number of observations for each class.

In our study, the test data set contains 152 days of observations (after the removal of NA values) where most of the observations are indicating an increase in the BIST 100 price index. In this study, we propose to use sensitivity and specificity metrics in addition to the train accuracy and validation loss in determining the best predictor. When the specificity and sensitivity diverge from each other, it is observed that the predictor is becoming biased towards one label. In order to determine the true predictive power of a neural network or any other classifier, we also propose the preparation of another test data set.

The first test set is selected as 152 days at the end of the time series, and it will be referred as test 1. The second test set is a subset of test 1, which contains equal number of observations for each label and will be referred as test 2. It is observed that many neural networks that provide a train accuracy above 70% and a test 1 accuracy above 65% failed to succeed in test 2 and their accuracy dropped to around 50%. Although, many measures are taken to prevent this decrease in accuracy such as regularization, dropout and bootstrap aggregating, it is very difficult to differentiate between true good predictors from the lucky ones. A good solution might be to use a loss function that penalizes the difference between the sensitivity in specificity. However, in this study the decision is made by comparing the sensitivity and specificity of each neural network without employing such a loss function.

Three approaches are taken in the study in order to train the neural networks and the resulting ensemble. The first method employs bagging without label balancing, the second method employs bagging with label balancing and the third method makes use of all the training data set (no sampling).

Out of all four feature selection algorithms, only the Extra-Trees and the Random Forest algorithms resulted in neural networks displaying a test accuracy that is larger than 65%. Although the Gaussian RBF kernel method shares many features with the Extra-Trees and Random Forest algorithms, an acceptable network could not be obtained. One reason behind this result might be the ill nature of the problem. Changing features requires the hyper parameters to be tuned again and the tuning process is tedious. However, the shared features indicate the possibility of training an acceptable neural network with Gaussian RBF kernel method. Moreover, Gaussian RBF kernel method is computationally efficient, and it can be used as a first tool to reduce the number of features before employing the Extra-Trees and Random Forest algorithms which require higher computational effort. In this specific problem, the computational effort did not cause any problems since the data size is limited. By the nature of the financial forecasting problem, the data size is limited because the data is often collected with one day intervals. The dependencies between the features and the output change with time and collecting 50 years of data might lead to accurate predictive models.

In this study, the single best neural network yielded 75% train accuracy and 71% percent test 1 accuracy with the features of the Extra-Trees algorithm and an architecture with 24 neurons, two layers and a dropout probability of 0.2. The test 1 accuracy is higher than the accuracy of [5] which is 65%. However, when model is evaluated with the test 2 set, the accuracy dropped approximately to 50%. This result points out the possible misleading nature of using the test 1 accuracy as the final decision metric.

One of the important conclusions in the study of Akbilgic is that the 20-days ahead value of the BIST 100 (ISE 100) index is independent of its past and current values although many studies try to model the relationship only based on the past and current values of an index. Therefore, autoregressive approaches do not allow the problem to be modelled properly. The only significant linear relationship is found in one of the lag variables of

BOVESPA (Brazil) which is identified with cross-correlation matrices.

This study reveals correlation between the BIST 100 index and the TSEC (Taiwan), MSCI EEM (emerging markets), KOSPI (Korea), MXX (Mexico) and several other stock market indices. The performances of the single neural networks and ensemble approaches are shown in Tables 3 and 4.

Table 3. Results of training three neural networks, each experiment is repeated 100 times.

	NN1: No bagging, No balancing	NN2: No bagging, Balancing	NN3: Bagging No balancing
Val. Loss	0.56 ± 0.01	**0.55** ± 0.02	0.56 ± 0.05
Val. Acc.	0.70 ± 0.02	**0.71** ± 0.02	0.69 ± 0.05
Test 1 Acc.	**0.71** ± 0.02	0.69 ± 0.03	0.66 ± 0.09
Test 2 Acc.	**0.63** ± 0.09	0.58 ± 0.09	0.57 ± 0.18

Table 4. Results of three neural network ensembles, each containing 10 neural networks. Each experiment is repeated 10 times.

	NN1 Ensemble	NN2 Ensemble	NN3 Ensemble
Test 1 Acc.	**0.71** ± 0.02	0.70 ± 0.01	0.68 ± 0.03
Test 2 Acc.	**0.62**, 0.07	0.55 ± 0.05	0.54 ± 0.14

In order to sort the neural networks in a decreasing importance, validation loss is used. However, as Table 3 clearly indicates, this approach does not always yield the best neural network.

If the test 2 accuracy is not considered, one may falsely believe that highly predictive models are obtained as all neural networks (NN1, NN2 and NN3) yield a test 1 accuracy above 65%.

It is also observed that one must be careful when trying to eliminate some of the neural networks to achieve higher ensemble performance.

Best performing model is chosen as NN1 (no bagging and no label balancing). It is expected for bagging to lower the neural network performance since it is decreasing the size of the training set. However, label balancing also results in a decreased performance which might be unexpected.

Table 4 shows that all the ensembles resulted in lower mean and standard deviation in test 2 accuracy. In general, the results are in the favor of neural networks without ensembles. Although it is difficult to decide between NN1 and NN1 ensemble, the ensemble requires much more training time and there is no incentive to prefer the ensemble.

6. References

[1] Box, G.E.P., Jenkins, G.: Time series analysis, forecasting and control. 5th edn. Wiley, New Jersey (2016).

[2] Kim, K.: Financial time series forecasting using support vector machines. Neurocomputing 5, 307-319 (2003).

[3] Kaastra, I., Boyd, M.: Designing a neural network for forecasting financial and economic time series. Neurocomputing 10, 215-236 (1995).

[4] Zhang, G.P.: Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing 50, 159-175 (2003).

[5] Balaban, E., Akbilgic, O., Bozdogan, H.: A novel hybrid RBF neural networks model as a forecaster. Statistics and Computing 24, 365-375 (2013).

[6] Chen, Y.W., Lin, C.J.: Combining SVMs with various feature selection strategies. Studies in Fuzziness and Soft Computing 207, 307-319 (2006).

[7] Breiman, L.: Random Forests, Machine Learning 45, 5-32 (2001).

[8] Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees, Chapman & Hall/CRC, New York (1984).

[9] Louppe, G., Wehenkel, L., Sutera, A., Geurtz, P.: Understanding variable importances in forests of randomized trees. Neural Information Processing Systems 26, 431-439 (2013).

[10] Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning 63, 3-42 (2006).

[11] Wang, L., Zhang, J., Zhou, L., Tang, C., Li, W.: Beyond covariance: feature representation with nonlinear kernel matrices. In: International Conference on Computer Vision, pp. 4570-4578.

[12] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a sim-ple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15, 1929-1958 (2014).