COMPARISON OF MACHINE LEARNING ALGORITHMS FOR BLOOD GLUCOSE PREDICTION ON AIDA SIMULATOR

by

Doğugün Özkaya

B.S. in Computer Engineering, Boğaziçi University, 2011

Submitted to the Institute of Biomedical Engineering
in partial fulfillment of the requirements
for the degree of
Master of Science
in
Biomedical Engineering

Boğaziçi University
2018

COMPARISON OF MACHINE LEARNING ALGORITHMS FOR BLOOD GLUCOSE PREDICTION ON AIDA SIMULATOR

APPROVED BY:

Assoc. Prof. Dr. Albert Güveniş .....
(Thesis Advisor)

Prof. Dr. Burak Güçlü .....

Assoc. Prof Dr. Murat Sarı .....

DATE OF APPROVAL:

ACKNOWLEDGMENTS

I would like to thank Assoc. Prof. Dr. Albert Güveniş for his expert advice, encouragement and guidance throughout this extensive project.

ACADEMIC ETHICS AND INTEGRITY STATEMENT

I, Doğugün Özkaya, hereby certify that I am aware of the Academic Ethics and Integrity Policy issued by the Council of Higher Education (YÖK) and I fully acknowledge all the consequences due to its violation by plagiarism or any other way.

Name :

Signature:

Date:

ABSTRACT

COMPARISON OF MACHINE LEARNING ALGORITHMS FOR BLOOD GLUCOSE PREDICTION ON AIDA SIMULATOR

Predicting blood glucose (BG) values has an increasing interest along with the recent progress in processing capacity of computers and spreading of mobile devices. Inspired from existing research studies, this study aims to use a BG simulator program, AIDA, to generate BG values and make predictions. Thus, comparing results to existing studies has directed this objective to provide an in-silico testing. Other points in using a simulator instead of real patient data is that it is easy to collect data, and it disregards external factors like pregnancy or stress. For estimates with prediction horizons (PH) with 15,30 and 60 minutes, support vector regression (SVR), decision tree regression, Gaussian process regression, k-NN regression, random forest regression and for neural networks: recurrent neural network (RNN) with long short-term memory (LSTM) unit and neuro-fuzzy network and feed-forward neural network (FFNN)have been employed. Among multiple algorithms neuro-fuzzy network (ANFIS) has the best results with RMSE values of 1.19 mg/dl, 2.53mg/dl and 5.81mg/dl for 15,30 and 60 minutes prediction horizons (PH). The audience for this paper is the research community who works on BG prediction and looking for ways to design a model for an algorithm for their selected set of inputs. This study presents a guide to selecting an algorithm and build a model for in silico simulation. This research can be extended to real world data or converted into a tool to create benchmark tests for models with given features and hyperparameters.

Keywords: Diabetes Mellitus, Machine Learning, AIDA, Blood Glucose Prediction.

ÖZET

AIDA SİMÜLATÖRÜ ÜZERİNDE KAN ŞEKERİ TAHMİNİ İÇİN MAKİNE ÖĞRENMESİ ALGORİTMALARININ KIYASLANMASI

Kan şekeri değeri tahmini, makine öğrenme tekniklerinin yaygınlaşması ve günümüz bilgisayarlarının ve akıllı cihazların kapasitelerinin artmasıyla ilgi çeken konulardan biri haline geldi. Bu çalışmada amaç, benzer çalışmalardan esinlenerek, AIDA simülatörü ile elde edilen verilerle kan şekeri tahmini yapmak. Bu doğrultuda hedef, sanal ortamda test yapılmasını sağlamak oldu. Simülatör kullanılmasının, veri toplamanın kolay ve hızlı olması, hamilelik, stres gibi çevresel etmenleri göz ardı etmesi gibi faydaları da mevcut. Tahminler, test anının 15, 30 ve 60 dakika sonrasındaki değerleri, destekçi vektör makinesi, karar ağacı, Gauss süreci, k-en yakın komşu, rastgele orman ve sinir ağları arasından da ileri ilerleyen, bulanık ve tekrarlayan yapay sinir ağları algoritmalarıyla gerçekleştirildi. Bu çalışmanın sonucunda bulanık sinir ağı, 15, 30 ve 60 dakikalık tahminlerde sırasıyla 1.19mg/dl, 2.53mg/dl ve 5.81mg/dl kök-ortalama karesel hata değerleri ile en iyi sonuca ulaştı. Bu makalenin hedef kitlesi, kan şekeri tahmini üzerinde çalışacak ve elinde girdilere ve performans kriterlerine göre etkili bir model seçmeye çalışan araştırma grupları olarak düşünülmektedir. Bu çalışma, bilgisayar ortamında test amaçlı model oluşturulması için bir rehber olma özelliği taşımaktadır. Aynı zamanda bağlamı gerçek hasta verisi özellikleri kullanılarak kıyaslama testleri yapılması doğrultusunda da genişletilebilir.

Anahtar Sözcükler: Diyabet, Makine Öğrenmesi, AIDA, Kan Şekeri Tahmini.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

a_{ij}	Description of a_{ij}
k	Covariance Function
x	Input Variable of First Observation
x'	Input Variable of Second Observation
σ	Variance of the Random Variable
l	Length Value for Covariance Function
y	
N	Noise Function
δ	Kroenecked Delta Function
K	Result of Kernel Function over Input Variable
K_*	Similarity of the Training Values to Test Values
K_{**}	Test Values among Each Other
y	Training Target Values
y_*	Test Target Values
$\overline{y}_*$	Estimation over Test Target Values

LIST OF ABBREVIATIONS

ANFIS	Adaptive Neuro-Fuzzy Inference System
ANN	Artificail Neural Network
AR	Accurate Readings
ARIMA	Autoregressive Model with Moving Average
BE	Benign Error
BG	Blood Glucose
BGL	Blood Glucose Level
CC	Correlation Coefficient
CGM	Continuous Glucose Meter
CH	Carbohydrates
DM	Diabetes Mellitus
DTR	Decision Tree Regression
EE	Erroneous Errors
EGA	Error Grid Analysis
FFNN	Feed-Forward neural Network
FIS	Fuzzy Inference System
GP	Gaussian Process
GPR	Gaussian Process Regression
HH	Hour
IDE	Integrated Development Environment
KNN	k-Nearest Neighbour
LSTM	Long Short-Term Memory
MARD	Mean Absolute Relative Difference
ML	Machine Learning
MM	Minute
MSE	Mean Square Error
NIR	Near-infrared spectroscopy
NN	Neural Network

k-NNR	k-Nearest Neighbour Regression
PH	Prediction Horizon
PPG	Photoplethysmography
RBF	Radial Basis Function
RFR	Random Forest Regression
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SOM	Self-Organizing Map
SVM	Support Vector Machine
SVR	Support Vector Regression
WFNN	Wavelet Fuzzy Neural Network
WHO	World Health Organization

1. INTRODUCTION

Blood glucose(BG) level prediction is a highly trending topic where researchers aim to find the BG values of a patient in the future based on external and internal inputs. This task is crucial for Type 1 Diabetes Mellitus(T1DM) patients who cannot produce insulin in their bodies and depend on external insulin. Since there is not a permanent solution for this disease, they try to regulate their lifestyles to manage their disease. For this purpose, predicting the BG value plays a key role in alerting them and helping them to take precautions. Also, it will be easier for both patient and clinician to decide insulin amount with a reliable prediction mechanism.

Continuous glucose meters(CGM) provide continuous readings of BG values and provide detailed insight about the BG variations. Using the data from a CGM device which collects data with a 5 minutes interval is efficient in prediction with time series analysis. The efficiency of linear models increases with the recursive approaches. Autoregressive model with moving average(ARIMA) is a useful and powerful estimator in this context [1]. Introducing exogenous variables such as blood pressure, cholesterol, low-density lipoprotein cholesterol and high-density lipoproteins produced results with increased accuracy [2].

Although exogenous variables are very helpful for increasing prediction accuracy, they cause high dimensionality and non-linearity. In this case a more flexible and accurate method is needed. Also, linear regression models need physiological responses of body to insulin and glucose intake, where it is hard to represent them in a mobile device. Overcoming this, SVR can be used with multivariate inputs such as BG value, plasma insulin, CH amount and physical activities. Including those inputs resulted with an increase in the accuracy both in short and long term [3]. SVR models can also be used for single input from CGM of a T1DM patient [4].

As the multivariate inputs are preferred for increased accuracy, more flexible

models are needed to make predictions. Artificial neural networks are very useful in that manner. Their flexible structure and lower spatial complexity compared to SVR, makes them accurate even in a wide range of patient cases. Neural networks can be used in both standalone prediction tools and CGMs with different architectures and multiple inputs. Their greatest advantage is that they make good predictions in hypoglycaemic and hyperglycaemic ranges better than other techniques [5] - [6]. Architectural structures of neural networks are not only limited to unidirectional networks. Recurrent neural networks make quite effective and accurate predictions on simulator data, prepared by AIDA simulator which have multivariate time series characteristics [7].

The high number of studies in the BG prediction also brings about the need to compare and review those studies. These will not only give a list of existing studies, but also will provide insights about algorithms, their input spaces and their performances. Kavakiotis et al. prepared a review study which does not only include only blood glucose prediction, but also diagnosis, genetic and environment related studies [8]. For the BG prediction topic, in particular, Oviedo et al. compiled prediction studies and classified them as data-oriented, physiological-oriented and control-oriented [9].

Comparing multiple algorithms is also important for researchers who want to conduct experiments on BG prediction. So far it is shown that SVR outperforms autoregressive model with physiological inputs like CH intake, insulin intake and glucose dynamics [10]. In another attempt to compare FFNN, wavelet fuzzy neural network(WFNN), self-organizing map(SOM) and linear regression, SOM has shown best result and FFNN and WFNN also outperformed linear regression [11].

These comparative studies carry out only a limited number of algorithms. Although they provide useful insights, some of them fail to use the same data or feature space in all of the methods. This brings about the need to develop a benchmark test with an extensive number of algorithms. This is what this study aims to accomplish. The comparison will be based on accuracy in root mean squared error(RMSE), prediction time and reliability metrics such as: Clarke‘s error grid analysis, mean absolute

relative difference(MARD) and correlation coefficient(CC).

During prediction studies, one of the greatest challenges is collecting error free and noise free data. In this study, to emphasize the effectiveness of simulated data, AIDA simulator is used. The AIDA is a tool that takes metabolic inputs which is specific to patient cases, along with behavioural inputs like food and insulin intake and produces a time series of BG values in 15 minutes intervals for 24 hours. With the synthetic data generated, we will be able to prepare a benchmark dataset for all parametric and non-parametric algorithms. The predictions will try to approximate BG values in 15, 30 and 60 minutes ahead of current time.

2. MATERIALS AND METHODS

2.1 Dataset

2.1.1 AIDA Simulator

The main tool for the data collection is AIDA blood glucose simulator. AIDA is a software, ultimately intended for the use of clinical personnel as a decision support system [12]. More specifically it can be described as a tool for simulating a BG distribution, based on insulin and CH intake along with patient specific conditions. The software first developed in 1991 and can be downloaded from internet. It is also available online since 1996 [13].

The system has a compartmental structure. A single extracellular structure, which receives glucose from intestines and produces in liver. And two compartments for active and plasma insulin activities which assume that patients cannot produce insulin internally. In this case external injection is the only available source of insulin [14] - [15].

The model runs on several parameters. Those parameters can be categorised into two, one is patient specific parameters and the other is input based parameters. Patient specific parameters are, renal threshold of glucose, renal function, hepatic and peripheral insulin sensitivities and body weight (kg). For the case scenarios, each patient has their specific parameters. Input based parameters are CH amount in the meals, short and intermediate acting effective insulin amounts in grams. Short acting insulins have their onset around 0.5-3 hours, and last up to 24 hours, and their peaks is experienced around 2-16 hours. The onset of intermediate acting insulins is 0.5-3 hours, and last 24 hours, while have their peak around 2-16 hours.

The model has its limitations also. The software is not intended for real world

patient simulation, therapy planning and glycaemic prediction [16]. Thus, the usage is limited to teaching and research purposes. Those limitations are brought out from several facts. The changes in renal threshold because of age and counter-regulatory hormones like Glucagon at low levels of BG were not included in the model. Physical activity and stress level are constant throughout simulation which may affect the blood glucose values [7]. Besides those limitations, AIDA is a powerful and popular simulation tool and has been used in several studies as a mean of data collection [4].

2.1.2 Data Collection via AIDA

Data collection is done through the online version of AIDA. The online version of AIDA has the same running principle as the downloadable version [14]. For online version there are 40 case studies of T1DM patients. We have selected a single patient case for the research. While selecting cases one of the main concerns was the use of insulin type. The effect of long acting insulin lasts up to 36 hours. Since AIDA produces 24 hours results, the effect of long acting insulin overflows to hours before the intake.

The main setback of AIDA is that it produces BG values for a single 24-hour period. This causes the evening and night activities to influence the BG in the morning. This assumes a day is a cycle of the same activities. To avoid that, the carbohydrate intake and insulin intake is limited between 13:00 and 19:00. So, in the mornings, the simulation will have a more euglycemic property. The meals are taken at 13:30 and 19:00 with 15 minutes variations. Short acting insulins are taken before the meals around 15 and 30 minutes. One of the long insulins are taken before the first meal, and the other one is after the second meal by 15 to 30 minutes. Thus, the fluctuations between midnight and morning BG values have been minimized, so simulating several days are seemed to be sequential.

To examine a stress test, an alternative dataset has been generated. The aim is to see how the models response to sharp changes and extreme BG values. As mentioned

above the CH and insulin intake amounts and times are regular. This helps AIDA to produce reasonable values throughout a day and values around beginning and end of the days are similar. So there is no sharp transitions around midnight. To prepare a dataset for the stress test, the simulated data should have extreme changes during transitions between days and extreme values throughout the day. For this purpose CH intake instances are increased to 6, and insulin amounts are modified and sometimes insulin intake is skipped. So sharp decreases like 100 mg/dl in 15 minutes or having BG value over 250 mg/dl over several hours have been observed. This dataset has been reordered with the same methodology and has the same features with the data previously described. In this case, since first CH and insulin intake starts at 07:15, the dataset reformatted in such a way that, day starts at 07:00 and ends in next day at 07:00.

Since the current study employs many different machine learning algorithms, a standard dataset is collected, then reshaped according to needs of each model. 30 days of data were collected in the form of vectors for each measurement in 15 minutes. Each point has the information of the CH and insulin amount and current BG values at that time. For real world experiments, data cleaning is a necessity to utilise sparse data and to avoid corrupted or null data. Since data is formed of simulation output, the need to clean data was no longer a concern.

In order to use the collected data as inputs, the data have to be modified into a time-step format. The initial data is formed of the columns: current time, CH amount, short acting insulin and intermediate acting insulin, and the current BG value. If there is a CH or insulin intake in any given time, the amount is used as the value, otherwise it is 0. Dataset was reordered into points of timesteps which contain properties of the previous readings, and the next BG value in the prediction horizon as the target value. Thus, the previous values of CH and insulin can affect the outcome in the model. The initial data and this structure can be visualized in the Table 2.1 and Table 2.2 respectively. For different prediction horizons: 15, 30 and 60 minutes, 3 different datasets have been created.

Table 2.1
The initial data extracted from AIDA simulator.

Columns of Initial Data Collected from AIDA	Column Unit
Hour	HH:mm
CH,amount	grams
Short Acting Insulin Amount	grams
Long Acting Insulin Amount	grams
Current BG	mg/dl

Table 2.2
The structure and example of the reordered dataset.

Columns	Time Window			Current Time	Target
Time	t-8 (12:00)	t-7 (12:15)	...	t (14:00)	t+4 (15:00)
BGL (mg/dl)	90.91	90.61	...	94.28	93.22
CH amount (grams)	0	25	...	0	-
Short Acting Insulin (grams)	4	0	...	0	-
Long Acting Insulin (grams)	12	0	...	0	-

2.2 Model Definitions

2.2.1 Support Vector Regression

Before starting to explain Support Vector Regression, it is compulsory to clarify the principle of support vector machines. SVMs were first introduced in 1995 by Cortes & Vapnik for binary classification purpose. The idea is basically looking for the optimal separating hyperplane between the two classes by maximizing the margin between the closest points of classes as in the Figure 2.1 [17].

The support vector regression technique is, to find a function that has the most deviation from the observed targets for the training data around a separating hyperplane. Since the data used in this study is nonlinear, the optimisation problem becomes finding the flattest function in feature set [18].

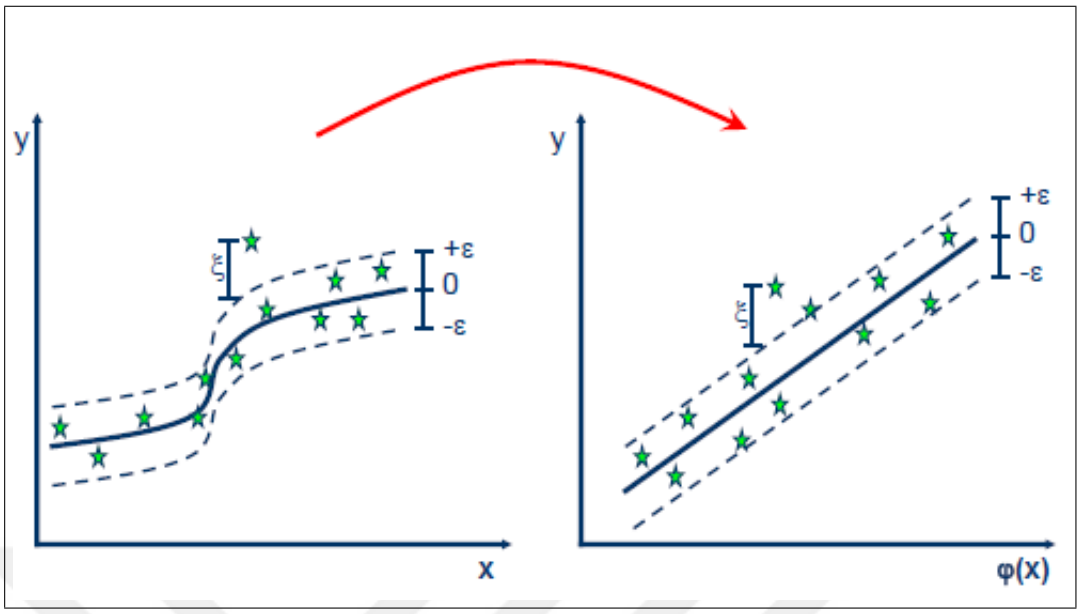

Figure 2.1 Support vector classification: the points lying on the boundaries are called support vectors, and the middle of the margin is the optimal separating hyperplane.

2.2.2 Feed Forward neural Network

FFNNs are specific types of Artificial neural networks (ANNs) which are inspired by real neurons in brain and used for computation and ML tasks [19]. These networks consist of several neurons (processing units), each generate an output from their activation function by recalculating their connection weights with incoming input. The neurons in input layer are activated from external data while neurons in hidden layers are given input data from previous layers. Output layers are fed with data from internal neurons and yield ultimate output.

Input neurons get activated by sensors perceiving the environment, other neurons get activated by weighted connections from previously active neurons. The learning is achieved by the objective of finding weights which make the NN generate the optimum output [20]. A FFNN is a special shaped NN that consists of three layers, input, hidden and output layers. In any given layer, each neuron is connected to a neuron in the next layer [21].

2.2.3 Recurrent Neural Network

Long Short-Term Memory: Recurrent neural networks have the topology in which connections coming from the neurons of output and intermediate layers return to previous layers to form a directed graph. This allows the network to process data that has temporal characteristics. This can be interpreted as a memory in an additional state [22]. For this study long-short term memory (LSTM) network is used as RNN model. This allows the model to keep the previous readings of BG in its memory. To describe shortly, an LSTM has memory cells that have looping connections onto them. It also has an additional unit to merge old and new data and has a multiplication unit to achieve to forget task [23].

2.2.4 Decision Tree Regression

Decision trees are structures that recursively divide data into partitions based on a set of rules that define each branch or node in the tree. A decision tree divides data into subsets from root to leaves to have the best possible data in the nodes in terms of homogeneity (purity) [24]. In this term, purity is the situation that each leaf has elements of the same class. Although this technique is generally used for classification tasks, it can be used for regression in order to make predictions on numerical values, also known as recursive partitioning. To define, decision tree regression is a method to predict numeric values of a dependant variable. Instead of classifying the data, here, objective is to minimize an optimization criterion to find the difference between target values and means of the 2 groups recursively. To avoid overfitting, the tree is pruned by criteria that uses tree size as a penalizing factor. So, unlike parametric models, that calculate coefficient for input, this algorithm calculates the relative importance of the input set, internally [25].

2.2.5 Random Forest Regression

Random forest classifier is an ensemble algorithm which basically reflects the collective decision of multiple different trees. The data is distributed over that collection of trees with bootstrapping aggregation, which randomly selects subsets of the data of which attributes are picked with attribute bagging. Then within these subtrees, the average of the decisions of related trees is calculated [26]. Random forests can also be extended for the purpose of regression. Since random forests are a collection of decision trees, the regression task runs on them. The methodology is same as explained in the decision tree regression: calculating relative importance of predictors by converging an optimization criterion to minimum [27].

2.2.6 k Nearest Neighbour Regression

k-nearest neighbour classification (k-NN), is based on the idea to find a target pattern to achieve clustering in terms of labels. k-NN assigns the majority vote of the data sample to target point [28]. This unsupervised non-parametric technique is also extended to regression for this study. A similar approach to classification, the sample mean (or trimmed mean, or some other statistics) of the numerical target of k nearest neighbours is calculated with some distance function, like Euclidian, or Minkowski [29].

2.2.7 Neuro-Fuzzy Networks

As the name suggests this hybrid approach combines fuzzy logic with neural networks. Essentially, it is a neural network whose hidden layer is a fuzzy inference system (FIS). Such a model is trained to determine the most appropriate membership function for the FIS [30]. The input is fuzzified by rules and mapped to a membership function. Then the output is given out as a fuzzy set and with defuzzification unit, converted into crisp outputs again [31]. There are two main approaches for this model, one is Mamdani FIS, which produces an output of fuzzy set and have a substantial

computational burden; the other one is Sugeno FIS, which is computationally efficient and produces linear or constant outputs [31].

2.2.8 Gaussian Process Regression

Gaussian process regression, is another non-parametric supervised algorithm. By definition it can be summarised such that it extends Gaussian distributed multivariate inputs to infinite dimensionality. Formally, a Gaussian process can be summarized such as: a posterior data generates data, located over some prior domain such that it follows multivariate Gaussian distribution. To describe the regression process, it is better to understand the role of Gaussian process in mapping input and output. To start with, the covariance of 2 observations can be shown as:

$$k(x, x') = \sigma_f^2 e^{\left[\frac{-(x-x')^2}{2l^2}\right]} \tag{2.1}$$

where l is the length value to give flexibility to this relation. And for a general representation of observation as a function of input, it is formulated as:

$$y = f(x) + N(0, n^2) \tag{2.2}$$

where N is the noise function. To add the noise in the covariance function, Kroenecked delta function sigma is used.

$$k(x, x') = \sigma_f^2 e^{\left[\frac{-(x-x^{'})^2}{2l^2}\right]} + \sigma_n^2 \delta(x, x^{'}) \tag{2.3}$$

And for regression, covariance function is calculated for all possible combinations

of these points.

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{bmatrix} \tag{2.4}$$

$$K_* = \begin{bmatrix} k(x_{*,x_1}) & k(x_{*,x_2}) & \dots & k(x_{*,x_n}) \end{bmatrix} \tag{2.5}$$

$$K_{**} = k(x_*, x_*) \tag{2.6}$$

In GP modelling, the data can be represented as a sample of multivariate Gaussian distribution which can be formulized as:

$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim N\big(0, \begin{bmatrix} K & K_*^T \\ K_* & K_{**} \end{bmatrix}\big) \tag{2.7}$$

In terms of conditional probability, it can be rewritten as:

$$y_*|y \sim (K_* K^{-1} y, K_{**} - K_* K_{-1} K_*^T) \tag{2.8}$$

Then the best estimation for target value is:

$$\overline{y}_* = K_* K^{-1} y \tag{2.9}$$

And its variance is:

$$var(y_*) = K_{**} - K_* K^{-1} K_*^T \tag{2.10}$$

For short, GP regression defines a prior over functions, which can be converted into a posterior over functions once we have seen some data. The Gaussian process regression uses this mentality to produce a distribution over functions. The functions

mentioned here are used to generate a posterior distribution with prior distribution and some observed training data [32]. The data is also converted via covariance functions and used in a normal distribution to generate those functions.

2.3 Implementation Details

The collected data is split into two parts, training and test sets. The first 20 days of the dataset is used for training the model. The remaining 10 days is used for testing. Thus, for each model the data have been split into train and test datasets with the ratio of 67% to 33%. The aim for testing is to evaluate how well the model has been trained with unseen data. The performances of test sets have been checked with the RMSE between observed simulation values and predicted values. RMSE is selected because it is more useful for the situations where larger differences are undesirable. This is mainly because of, the square is calculated at first. It is not steady like mean absolute error. Following testing and validations, predictions are performed over 12 and 24-hour subsets of data. Then the trained models predicted the BG values in 15, 30 and 60 minutes of PH. Prediction success is measured with root mean squared error (RMSE).

Following the testing, validations of the models have been implemented with K-fold cross validation technique. The dataset is split into k parts of which k-1 parts are used for training and remaining one is used for testing. This method is repeated for k times in a circular manner. K=3 has been selected for the number of folds. The success criteria of the models have been measured with coefficient of determination as the scoring parameter to observe the relation between observed values from simulation and predicted values. According to this criterion the result can be between 0 and 1. 1 is the perfect score, and as it gets closer to 0 it indicates that the relationship between two vectors gets weaker. If it has a negative value, it means, there is no connection between those compared vectors.

2.3.1 Support Vector Regression

For this algorithm, sci-kit learn 0.19.1 library with python 3.6 is used. For development Pycharm community edition was used as IDE. matplotlib library of python has been used for plotting results of the model. For Support Vector Regression, SVR class of scikit-learn library was used with the following hyper-parameters:

- kernel: RBF (radial basis function)
- C: 1000
- gamma: kernel coefficient for RBF: 0.1
- epsilon: 0.2

The data is split into 2 parts as input parameters and resulting BG value. For each prediction horizon, BG values for 15, 30 or 60 minutes after any given time, have been selected as the output. And for the previous 2 hours, all the features are used as input vectors. Then the data is scaled between 0 and 1 to avoid any of the features to suppress any other to affect the outcome significantly

2.3.2 Feed Forward Neural Network

For the implementation of the neural network, keras 2.1.1 library with python 3.6 is used. Keras is a machine learning library which runs on tensorflow, in order to provide an abstraction over the low-level tensorflow library. The implementation was conducted on pycharm IDE. Plotting for the visual control of predicted and actual blood glucose, matplotlib was used. For the cross validation, KerasRegressor class of the scikit-learn library was used.

For the sake of eliminating the effect of any of the features over any other, the input features are scaled between 0 and 1. Feature set includes CH intake amount,

short and intermediate acting insulin amounts and BG values for the 120 minutes time window. The target is the BG value in the prediction horizon.

The input layer of the neural network has 31 neutrons and has a single output. The hidden layers visualized in Figure 2.2 has the following structure:

- Dense layer, neuron count: 31, activation function: relu()
- Dense layer, neuron count: 155, activation function: relu()
- Dense layer, neuron count: 62, activation function: relu()

The loss function for the model has been selected as MSE, because the output has single output. Adam (Adaptive momentum) optimisation technique was used because of its superior performance over other techniques, during gradient descent [33].

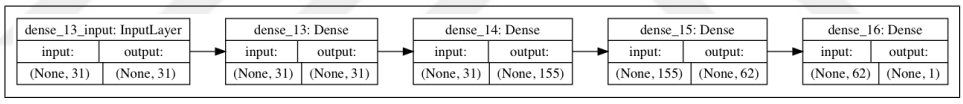

Figure 2.2 Schematic representation of the FFNN model layers.

2.3.3 Recurrent Neural Network (LSTM)

The implementation of this model is completed with keras 2.1.1 library with python 3.6. For visual confirmation during development matplotlib is used and the cross validation is completed with KerasRegressor class of the scikit-learn, as in the FFNN development.

The same procedure applies here also. The data is split into training and test sets and sliced into input and output with first 31 and the last columns, successively. A distinctive property of the LSTM is that it accepts input in a 3D shape in which features are shaped as “sequences”, “timesteps”, and “dimensions”. Sequences correspond to each data point in the dataset. Timesteps are the sequential states that data points are

stored. Dimensions correspond to the properties/columns of the data points [34]. The input features of dataset used in this study is 2D in nature. In order to convert into the 3D shape, a dimension is added in the second axis. Thus, the data has the same number of sequences, single time-step and same number of columns as before.

The input features picked for this algorithm are CH intake amount, short and intermediate acting insulin amounts and BG values in 120 minutes prior to the current timestep in the corresponding data point. In short, the same feature set with the FFNN experiment is used. This input is sent to LSTM network via an input layer with the shape of (1,31), corresponding to the dimensions (timesteps, dimensions). Following layers are an LSTM layer with 62 neurons and in the same shape with input layer. This layer propagates results with an output size of 62. The output layer has an input size of 62 and an output size of 1. This network is visualised in Figure 2.3 for more clear understanding.

The loss function of network is mean absolute error and Adam optimizer is used for the sake of both performance and decreased implementation complexity.

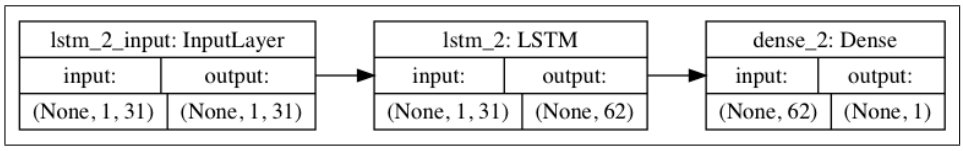

Figure 2.3 Schematic representation of the LSTM model layers.

2.3.4 Decision Tree Regression

The implementation of the model was conducted with python 3.6 and sci-kit learn 0.19.1. DecisionTreeRegressor class of sci-kit enables the development of this model on a higher level.

The CH intake amount, intermediate and short acting insulin amounts and previous BG values in 120 minutes are picked and scaled between 0 and 1. And the last column is spared as the output vector.

Prepared data is used to train DecisionTreeRegression model of which maximum depth is 60. The min_weight_fraction_leaf parameter which controls the ratio of the sum of all weights of all inputs to be at leaves, is set to be 0.02 to keep tree size under control. The criterion of split quality is determined by the mean squared error. This parameter is the determiner of the split in order to achieve purity in leaves. For classification tasks, purity means, having the same class of elements in one branch, but for regression, it separates elements according to the averages.

2.3.5 Random Forest Regression

Instead of implementing random forest regressor by explicitly combining multiple decision tree regressors, RandomForestRegressor class of sci-kit learn library was used and it was more compact, easy and convenient. The input scaling was between 0 and 1, slicing (first 31 columns as inputs and last column as output) and splitting were completed in the same way with other algorithm experiments.

The model is built with maximum depth of 10 and MSE as splitting criterion. Like in the regression trees, the critical parameter here is maximum depth. The performance increases as it increases, up to some point (10 for this case), because of the overfitting.

2.3.6 k-Nearest Neighbour Regression

The implementation environment and libraries are same with the Decision Tree Regression and Random Forest Regression. For this technique, KNeighborsRegressor class in scikit-learn is used. This model is built with number of neighbours parameter as 10. Other significant parameters are left default. Euclidean distance metric parameter, and uniform weights for predictions are selected. The data is scaled, sliced, and split in the same manner with the preceding algorithms.

2.3.7 Neuro-Fuzzy Network (Aadaptive Neuro-Fuzzy Inference System)

Different from other techniques, MATLAB R2017a and its ANFIS library is preferred for this part of the study. This is mainly due to lack of a reliable ANFIS library in python and the abstraction that MATLAB provides.

The data has the same pre-processing for training-test separation and input-output slicing as in all of the other algorithms. Grid partitioning was used as the clustering method for defining membership functions and fuzzy rules. This method generates membership functions for uniformly partitioned inputs and create a single-output fuzzy system.

The ANFIS model has been trained with training set with uniform radii of 0.5 for the optimum result, after a series of experiments, which specifies the influence of the input on data.

2.3.8 Gaussian Process Regression

This model was developed with python 3.6, sci-kit learn 0.19.1 and matplotlib libraries. Data slicing and splitting is same with the previous models. To build the model, GaussianProcessRegressor class of sci-kit library was used. The kernel function for the regressor was radial basis function (RBF). In fact, RBF is the squared exponential kernel. Since data is generated via simulation, a noise kernel has not been added. RBF is smooth in terms of distribution since it can have mean square derivatives in all orders, The RBF parameters are:

- length_scale: The length scale of the kernel which gives flexibility for calculating the covariance function: 10
- length_scale_bounds: sets lower and upper bound for length_scale.: 0.01, 100

This kernel is used as parameter for the GaussianProcessRegressor. Also, alpha value is set to 0.15 to define a subset for the model validation.

3. RESULTS

For each of the models, which produced results for this study, predictions were made over periods for 12 and 24 hours. 12 hours prediction period starts at 00:00 and ends at 12:00 while the period that lasts 24 hours starts at 12:00 and ends at 12:00 on the next day. For both periods, trained models predicted BG values for prediction horizons of 15, 30 and 60 minutes.

The evaluation of the models relies on predictive performances. To measure it, the error between prediction and observed BG values from simulation are calculated by RMSE.10 separate days of data, which is unseen by the models during training are used. This method also helped to compare the performance of the models among each other.

Table 3.1
Prediction errors in terms of RMSE(mg/dl) for 15 minutes prediction horizon.

Algorithm	12 hours	24 hours	R2
Decision Tree Regression	1.60 ± 0.35	4.04 ± 1.84	0.97
FFNN	1.53 ± 0.68	3.22 ± 1.63	0.98
Gaussian Process Regression	0.97 ± 0.83	4.54 ± 1.72	0.97
k-NN Regression	2.75 ± 0.59	7.20 ± 2.50	0.94
Random Forest Regression	0.53 ± 0.15	2.80 ± 0.96	0.98
Recurrent Neural Network: LSTM	1.14 ± 0.77	2.56 ± 096	0.96
Support Vector Regression	0.46 ± 0.19	3.00 ± 0.94	0.98
Neuro-Fuzzy Network	0.26 ± 0.06	1.19 ± 0.32	0.99

To get a brief idea about the results, it may be concluded that, for 12 hours which starts at midnight and ends at noon, all models have better results. That is due to the absence of noisy effect of the carbohydrate and insulin. Since the BG values do not quickly respond to those inputs, it is hard for models to predict their effects. This leads predictions that spread to 24 hours to have worse predictive power as expected.

Table 3.2
Performing times of predictions of 15 minutes in seconds during 12 and 24 hours of prediction periods.

Algorithm	12 hours	24 hours
Decision Tree Regression	0.00035	0.00042
FFNN	0.00096	0.00126
Gaussian Process Regression	0.26568	3.45250
k-NN Regression	0.00340	0.00583
Random Forest Regression	0.00138	0.00078
Recurrent Neural Network: LSTM	0.00109	0.00167
Support Vector Regression	0.00336	0.00559
Neuro-Fuzzy Network	0.00111	0.00129

For the 24 hours case, the prediction set starts at 12:00 and ends at 12:00, on the next day. The prediction power gets poorer during the active time between 13:00 and 20:00.

Analysing the effects of prediction horizons, it is shown that as the prediction horizon increases the RMSE increases and coefficient of determination decreases, from 15 minutes to 60 minutes. Those can be observed better in Table 3.1, 3.3 and 3.5.

To investigate results more closely, for all of the prediction horizons, ANFIS has the best results for the dataset prepared in this study. As a numerical performance, it scored with RMSE values of 0.26mg/dl, 2.19mg/dl for 15 minutes. The result pair for 30 minutes and 60 minutes of prediction horizons are 0.60mg/dl & 2.53mg/dl and 1.56mg/dl & 5.81mg/dl respectively. These are best results among all of the algorithms. This algorithm has the best r-squared error value of 0.99 for 15 and 30 minutes PH, and 0.95 for 60 minutes PH.

This carefully extracted data with certain conditions have been the basis for our study. To point the significance of the influence of the conditions, a random amount of carbohydrates and absence of insulin occasionally were used to create an alternative dataset. This distorted dataset has been subject to algorithms developed in this study.

Table 3.3
Prediction errors in terms of RMSE(mg/dl) for 30 minutes prediction horizon.

Algorithm	12 hours	24 hours	R2
Decision Tree Regression	2.24 ± 1.03	6.78 ± 2.42	0.93
FFNN	1.43 ± 0.77	4.97 ± 2.00	0.95
Gaussian Process Regression	1.35 ± 0.98	5.03 ± 2.08	0.95
k-NN Regression	2.79 ± 1.09	8.97 ± 2.82	0.91
Random Forest Regression	1.45 ± 0.71	3.81 ± 1.81	0.96
Recurrent Neural Network: LSTM	1.88 ± 0.50	3.99 ± 1.85	0.94
Support Vector Regression	0.84 ± 0.39	5.19 ± 2.15	0.96
Neuro-Fuzzy Network	0.60 ± 0.14	2.53 ± 0.45	0.99

The results of these can be viewed in Table 3.7.

Calculating RMSE to observe the performance of models is useful for evaluating them. In addition, certain statistics are used to measure the reliability of the results. The reliability refers to the compliance of predictions with blood glucose meters. For this purpose, mean absolute relative difference (MARD) and correlation coefficient are calculated. Also, Clarke's error grid analysis is performed to test this compliance. Clarke's error grid analysis is a tool produced to measure reliability of BG predictions. Table 3.8 - 3.13 gives information about EGA outputs in terms of accurate readings (AR), benign errors (BE), and erroneous errors (EE), MARD and CC.

The results of our study are compared to the results obtained from the stress test data to see the limitations of our models. For prediction duration of 12 hours, the accuracy order had minor changes. As seen in Figure 3.49, ANFIS still has the best accuracy, however random forest regression performed very poorly. This alteration lost its effect on 24-hour predictions as can be seen in Figure 3.50. ANFIS still has the best result and followed by FFNN, GPR and SVR.

Prediction times of algorithms were compared as another criterion of performance. As seen in Table 3.2, 3.4, and 3.6 DTR has the best results. It is notifiable

Table 3.4
Performing times of predictions of 30 minutes in seconds during 12 and 24 hours of prediction periods..

Algorithm	12 hours	24 hours
Decision Tree Regression	0.00036	0.00011
FFNN	0.00071	0.00113
Gaussian Process Regression	0.28525	0.24094
k-NN Regression	0.0031	0.00771
Random Forest Regression	0.00143	0.00161
Recurrent Neural Network: LSTM	0.00110	0.00164
Support Vector Regression	0.00331	0.00642
Neuro-Fuzzy Network	0.00099	0.00154

that ANFIS has the second-best time and combining this with its accuracy, this algorithm stands out. On the other hand, Gaussian process regression has the worst time in all prediction horizons for both 12 and 24 hours periods. This is due to the nature of the model: it tries to find the best distribution of functions, and it is also non-parametric.

Table 3.5
Prediction errors in terms of RMSE(mg/dl) for 60 minutes prediction horizon.

Algorithm	12 hours	12 hours	R2
Decision Tree Regression	3.73 ± 1.64	13.08 ± 3.38	0.83
FFNN	3.15 ± 0.98	8.31 ± 2.71	0.88
Gaussian Process Regression	2.12 ± 0.84	8.49 ± 2.72	0.89
k-NN Regression	2.84 ± 1.07	12.81 ± 3.40	0.82
Random Forest Regression	1.79 ± 1.18	10.06 ± 3.01	0.90
Recurrent Neural Network: LSTM	3.02 ± 1.28	7.69 ± 2.65	0.85
Support Vector Regression	1.75 ± 1.11	6.34 ± 2.03	0.89
Neuro-Fuzzy Network	1.56 ± 0.49	5.81 ± 2.01	0.95

Table 3.6
Performing times of predictions of 60 minutes in seconds during 12 and 24 hours of prediction periods.

Algorithm	12 hours	24 hours
Decision Tree Regression	0.00015	0.00012
FFNN	0.00078	0.00108
Gaussian Process Regression	0.25854	0.24542
k-NN Regression	0.00432	0.00655
Random Forest Regression	0.00299	0.00302
Recurrent Neural Network: LSTM	0.00112	0.00169
Support Vector Regression	0.00678	0.00350
Neuro-Fuzzy Network	0.00115	0.00134

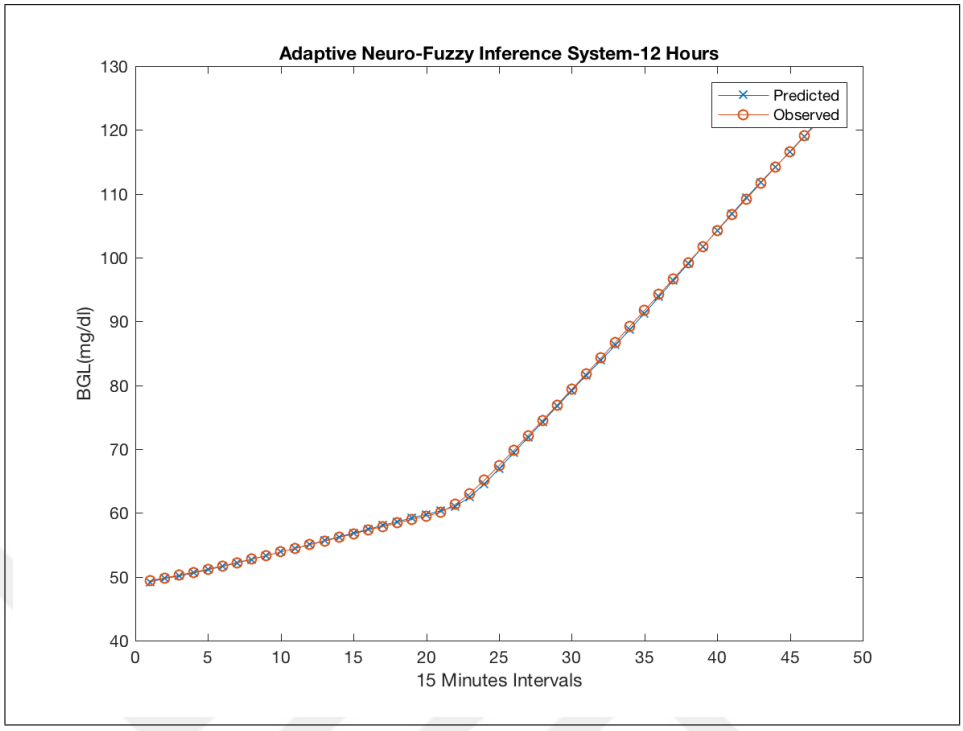

Figure 3.1 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using ANFIS.

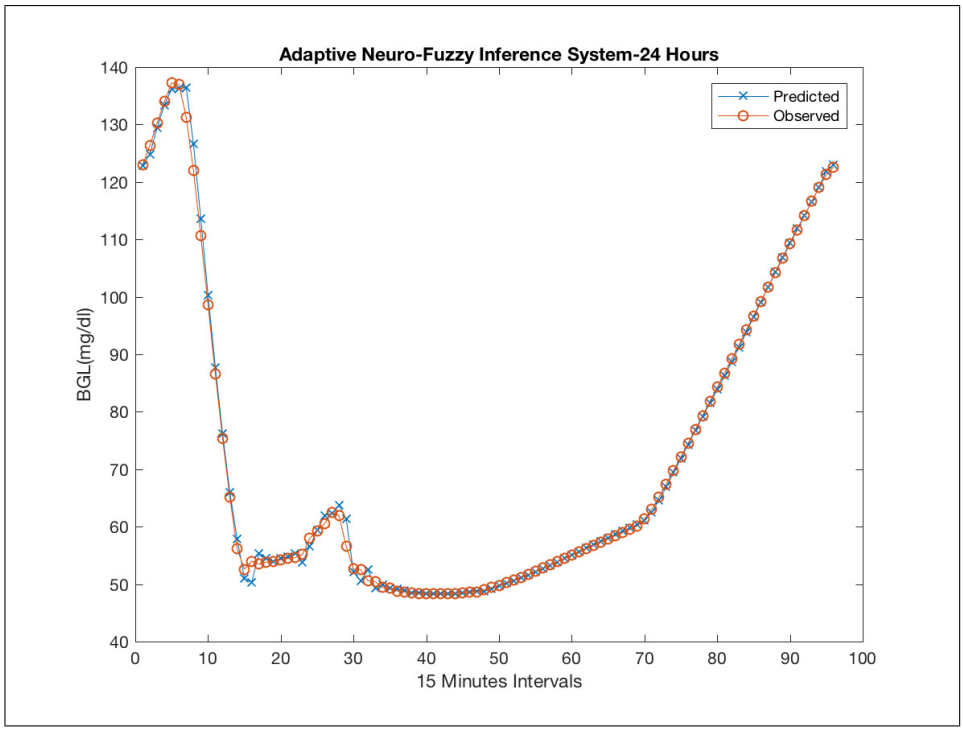

Figure 3.2 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using ANFIS.

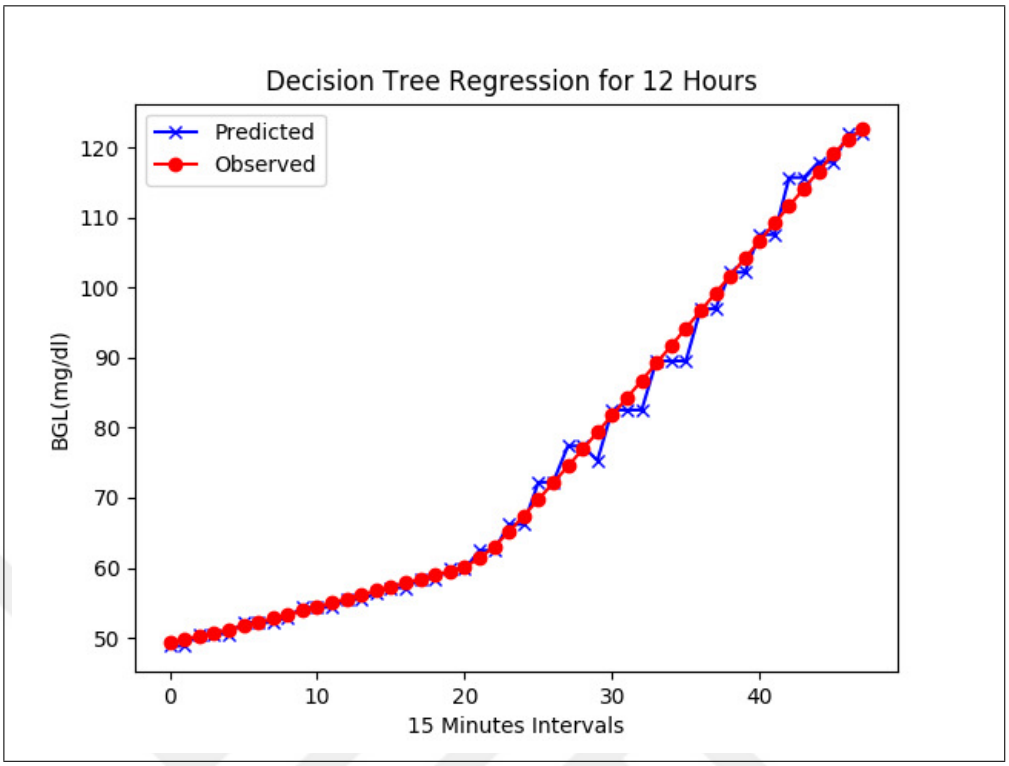

Figure 3.3 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using decision tree regression.

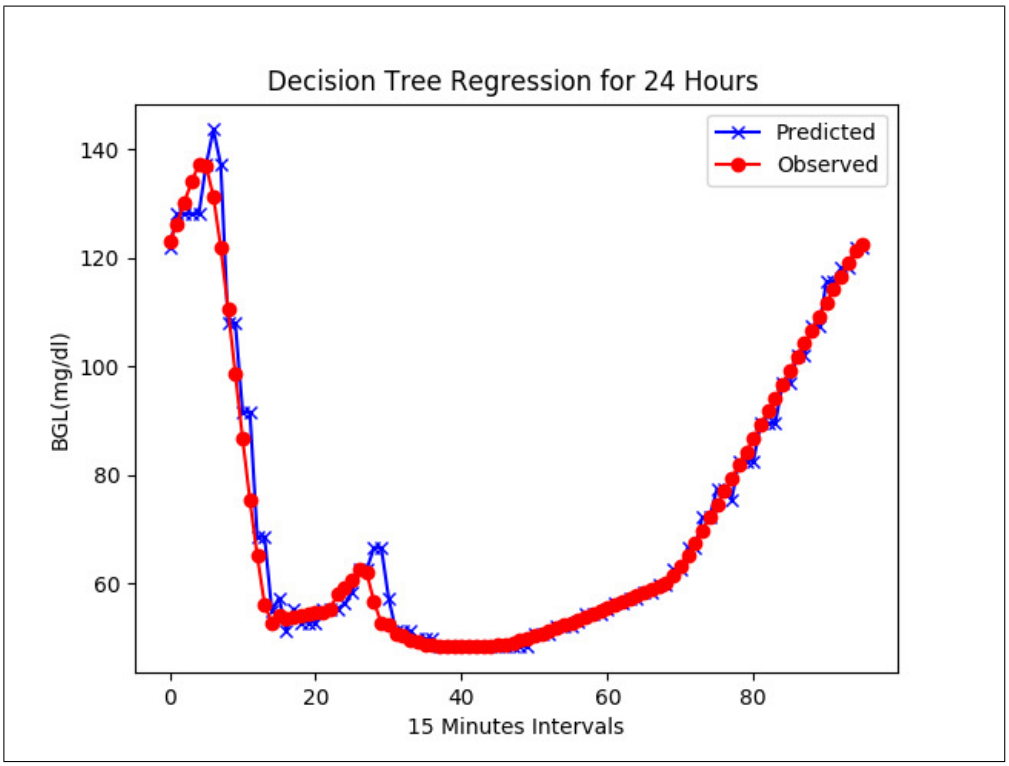

Figure 3.4 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using decision tree regression.

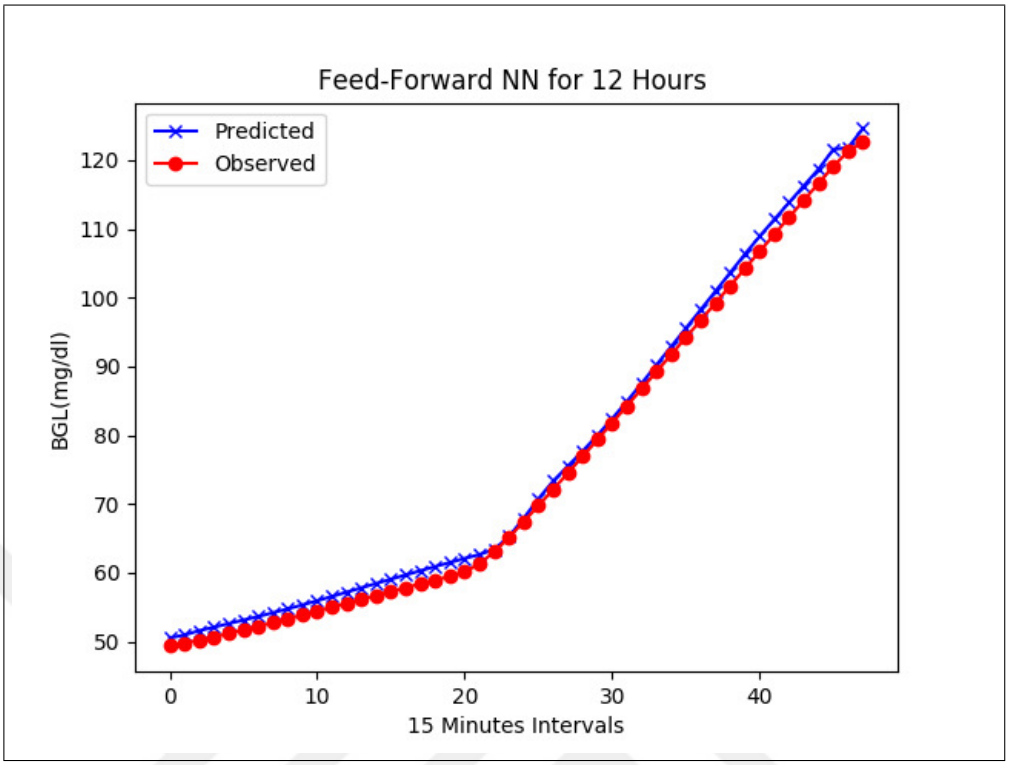

Figure 3.5 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using FFNN.

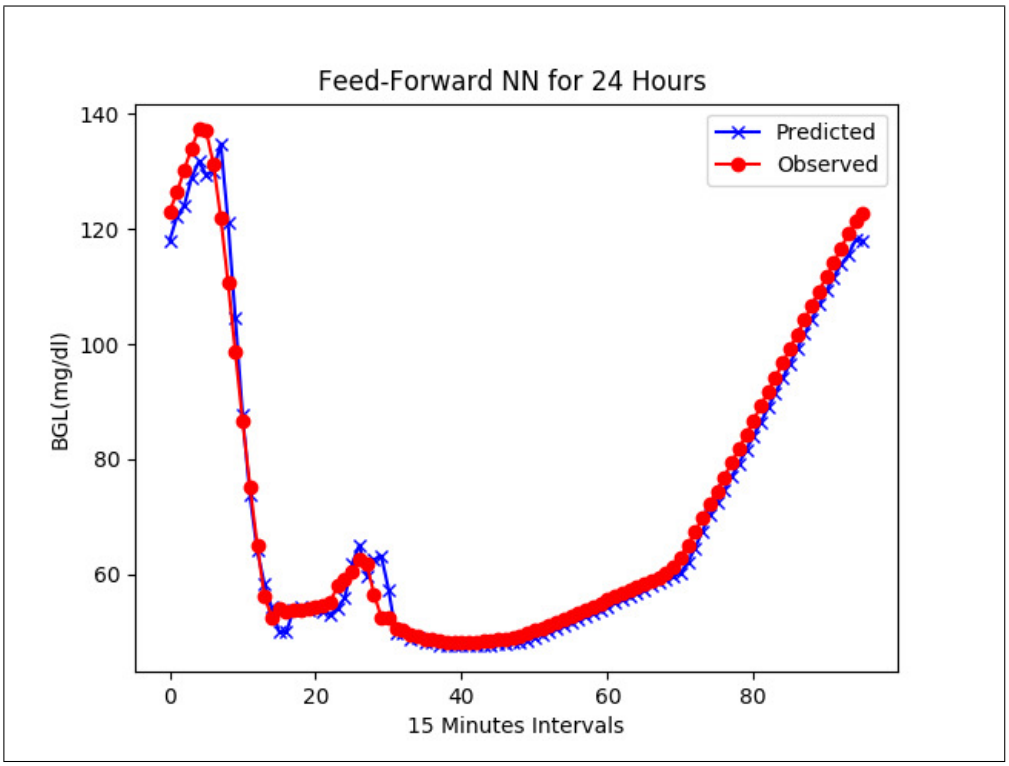

Figure 3.6 Predicting BG values 15 minutes ahead from 00:00 to 00:00 in next day using FFNN.

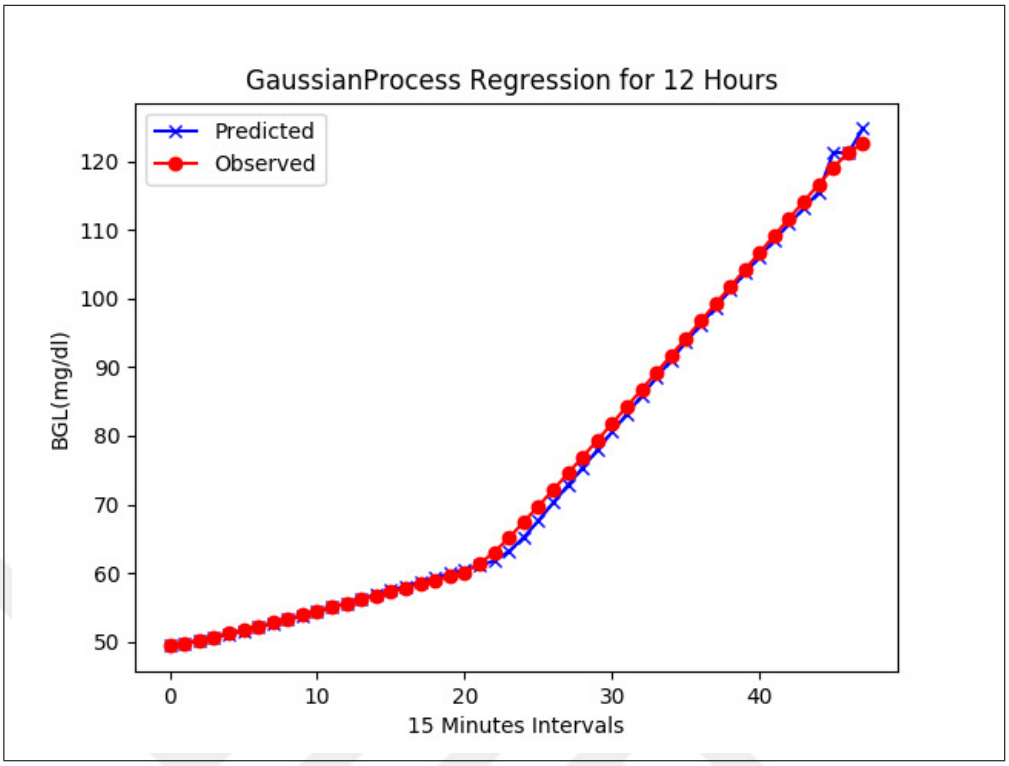

Figure 3.7 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using Gaussian process regression.

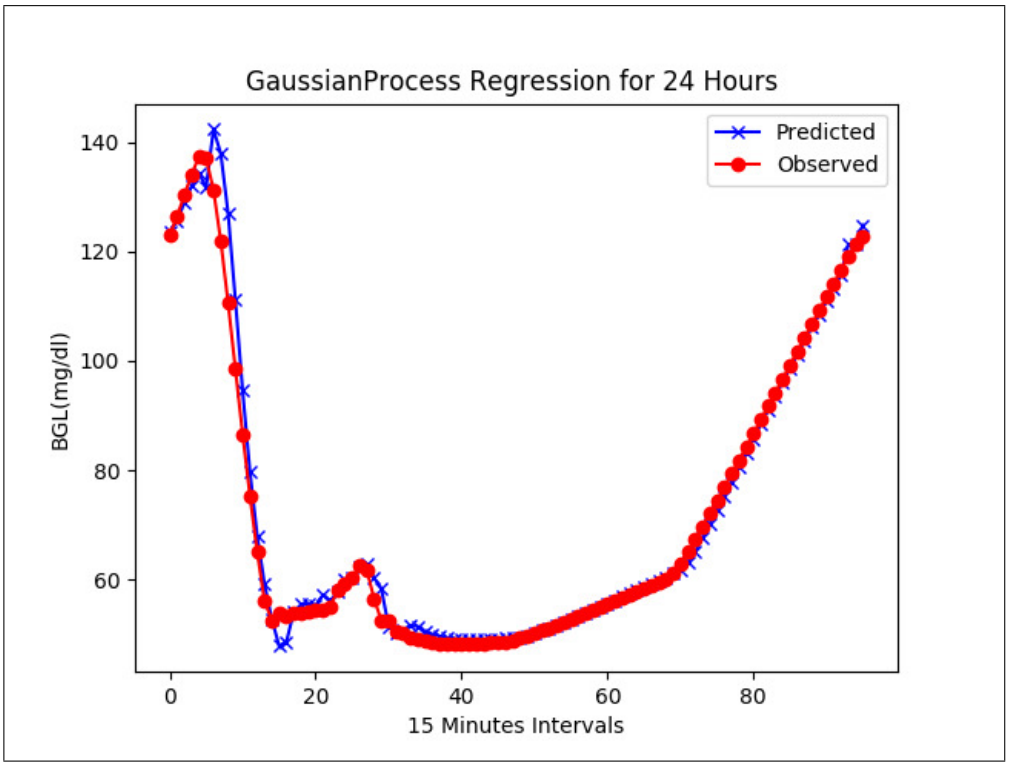

Figure 3.8 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using Gaussian process regression.

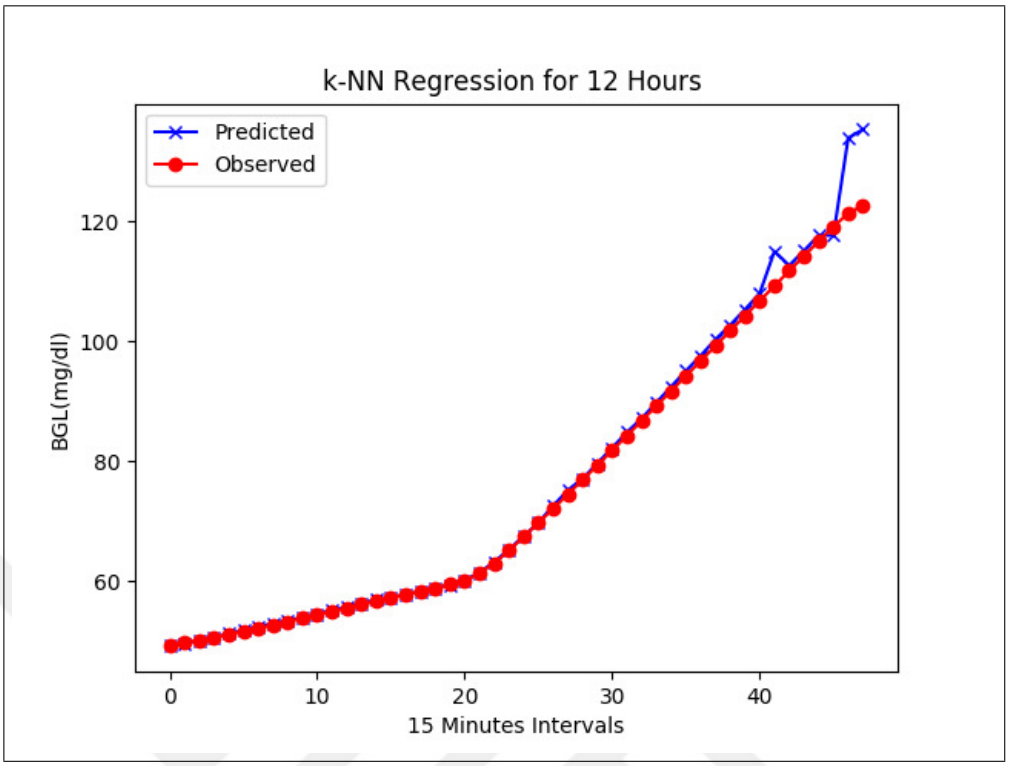

Figure 3.9 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using k-NN regression.

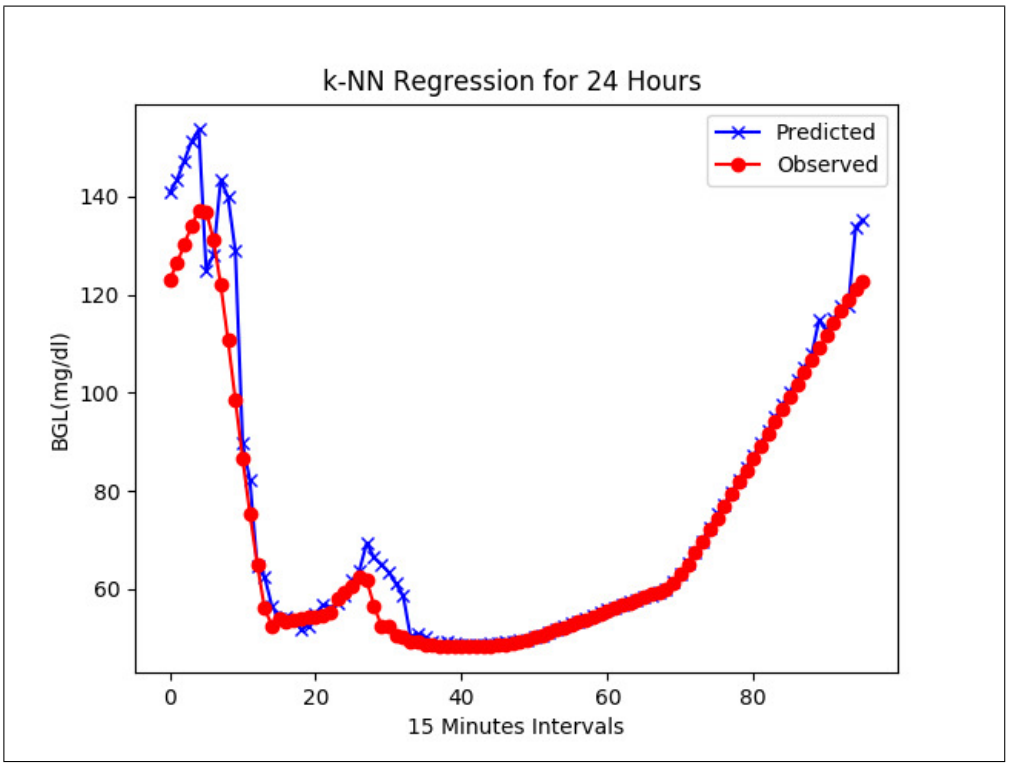

Figure 3.10 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using k-NN regression.

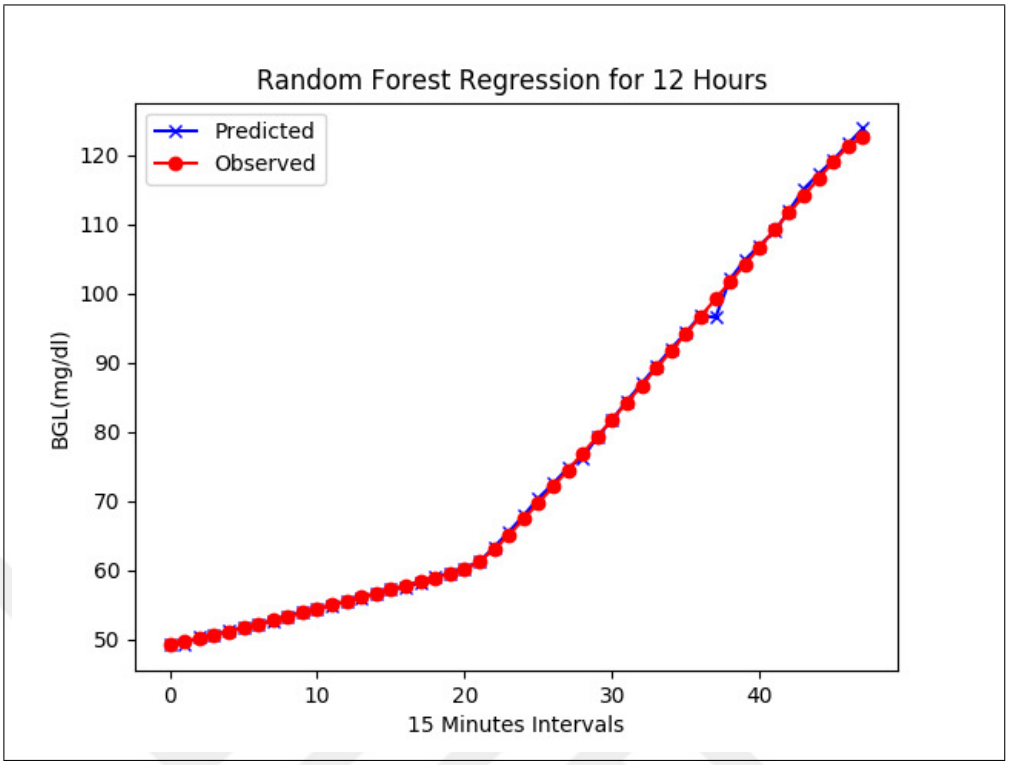

Figure 3.11 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using random forest regression.

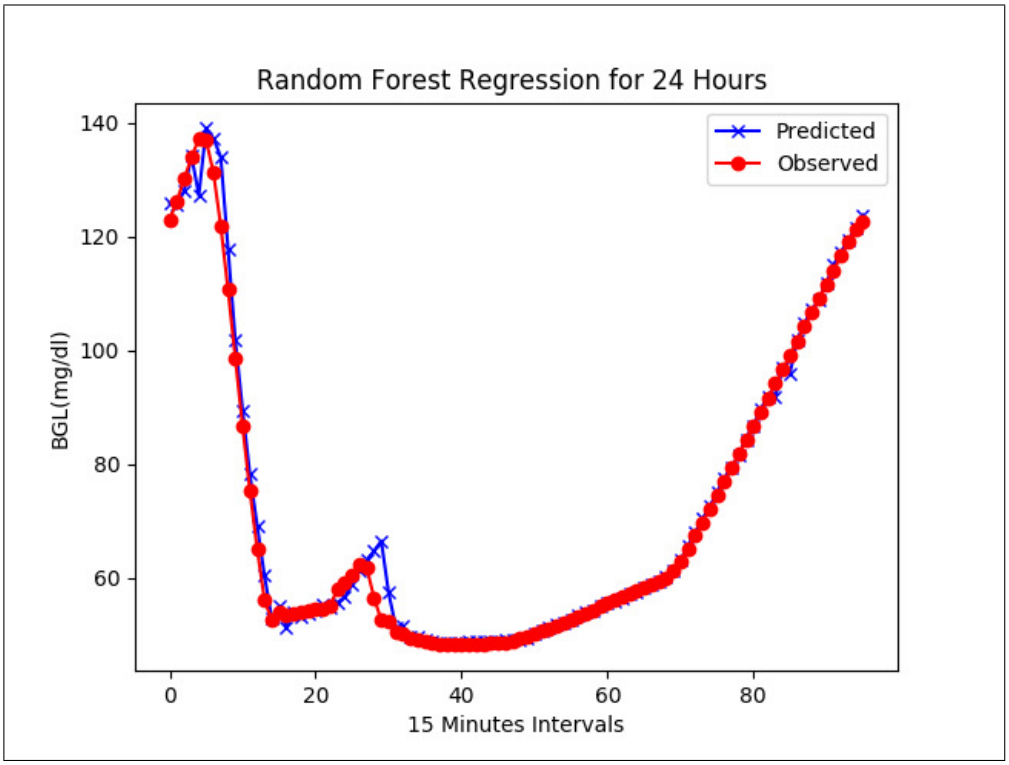

Figure 3.12 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using random forest regression.

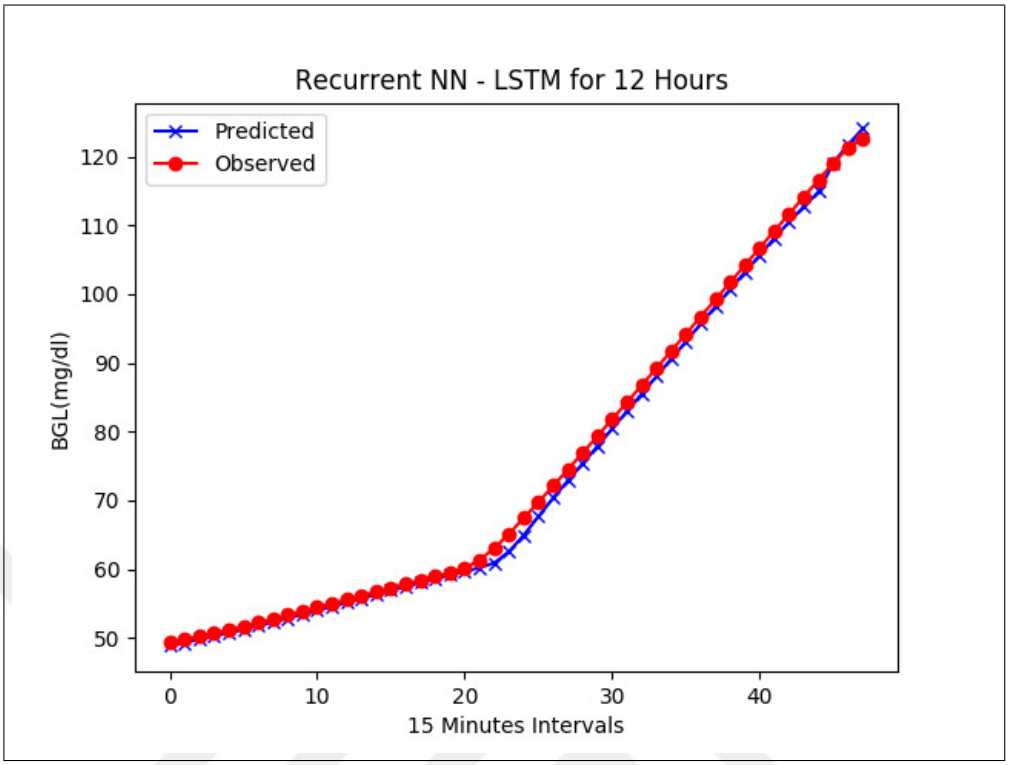

Figure 3.13 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using LSTM.

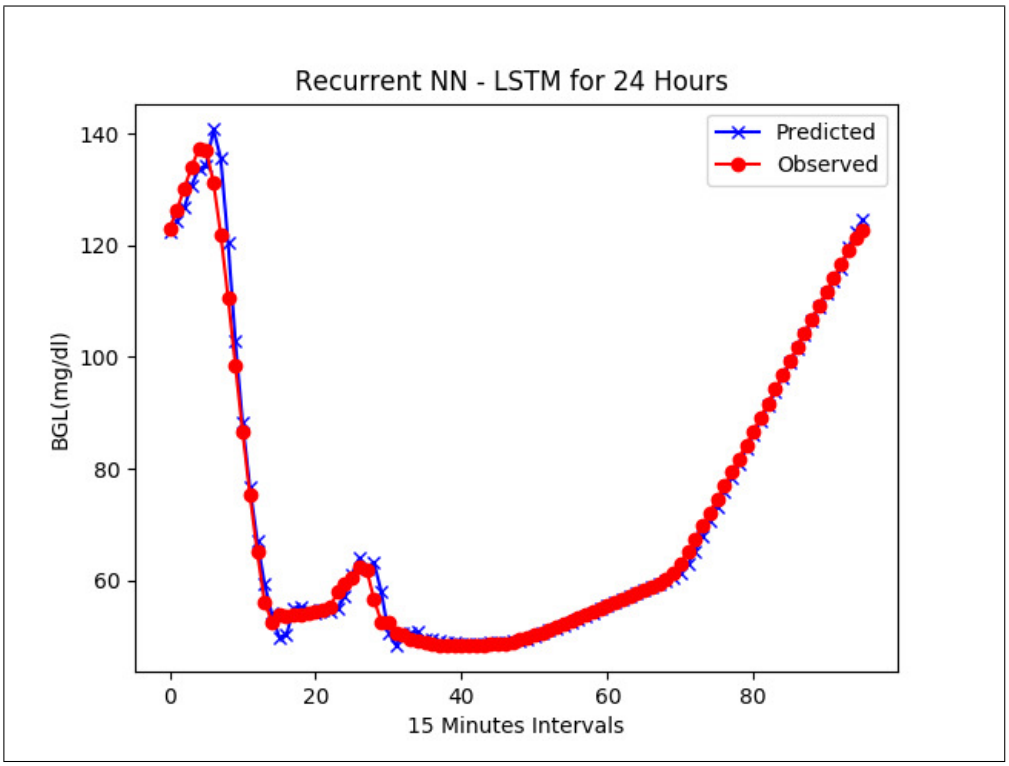

Figure 3.14 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using LSTM.

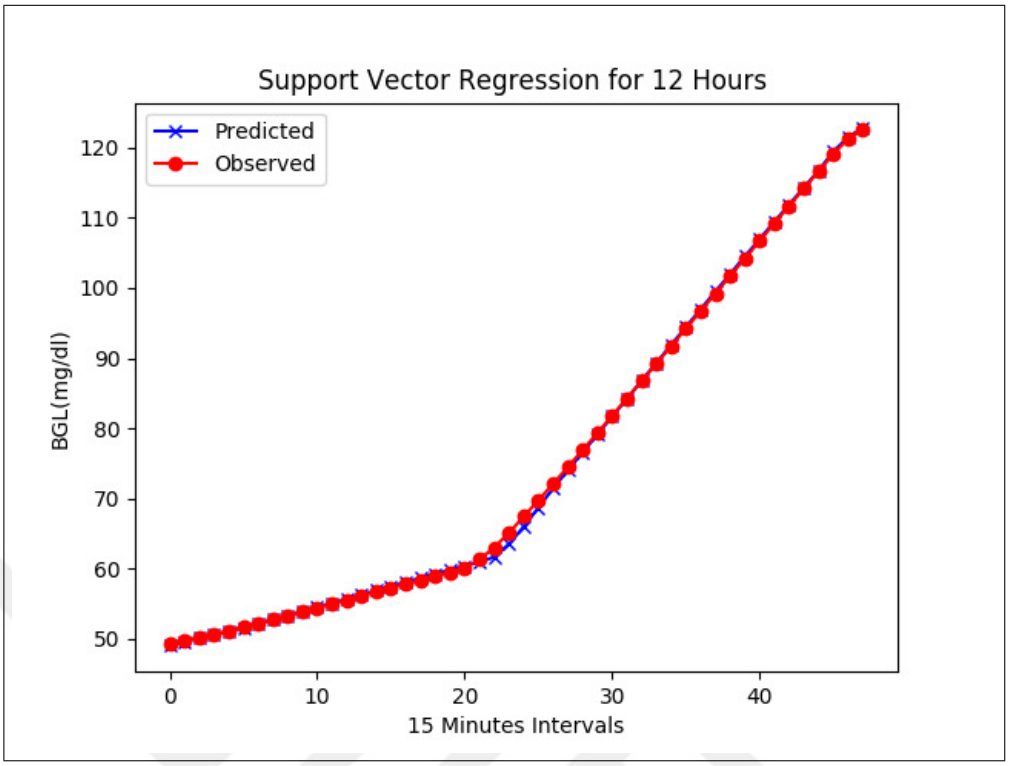

Figure 3.15 Predicting BG values 15 minutes ahead from 00:00 to 12:00 using support vector regression.

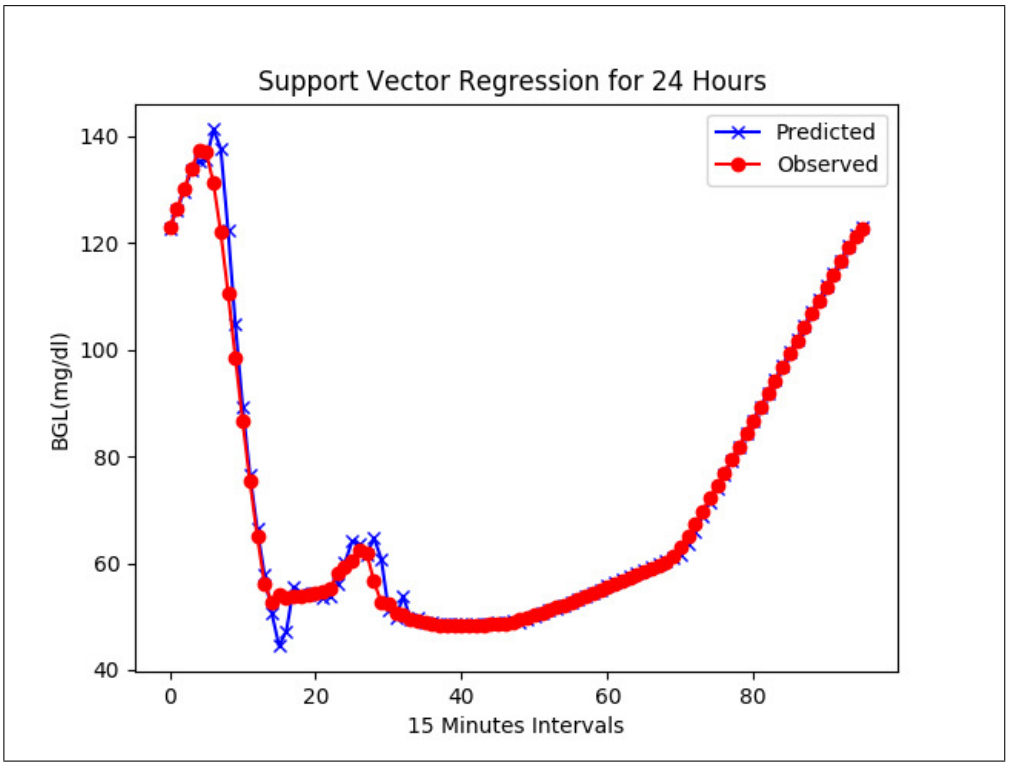

Figure 3.16 Predicting BG values 15 minutes ahead from 00:00 to 00:00 next day using support vector regression.

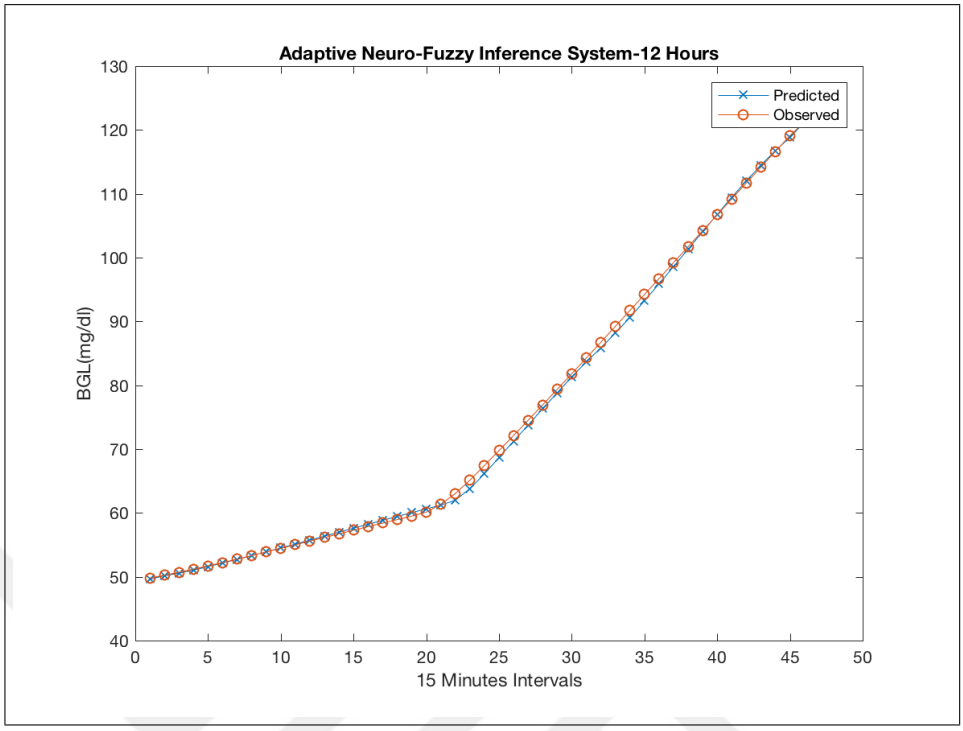

Figure 3.17 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using ANFIS.

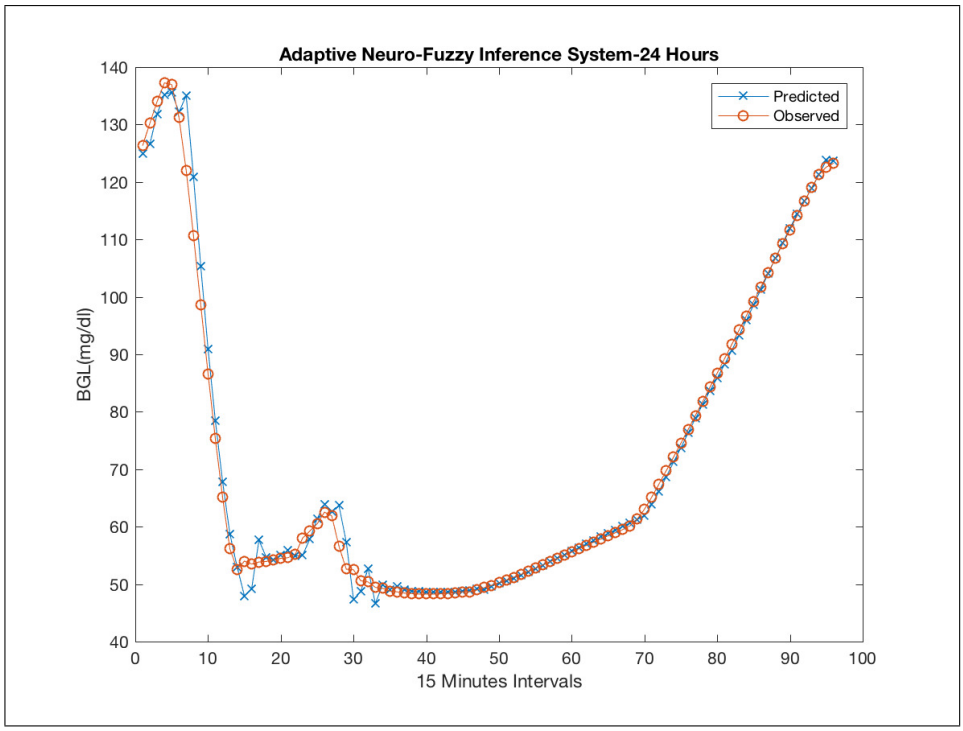

Figure 3.18 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using ANFIS.

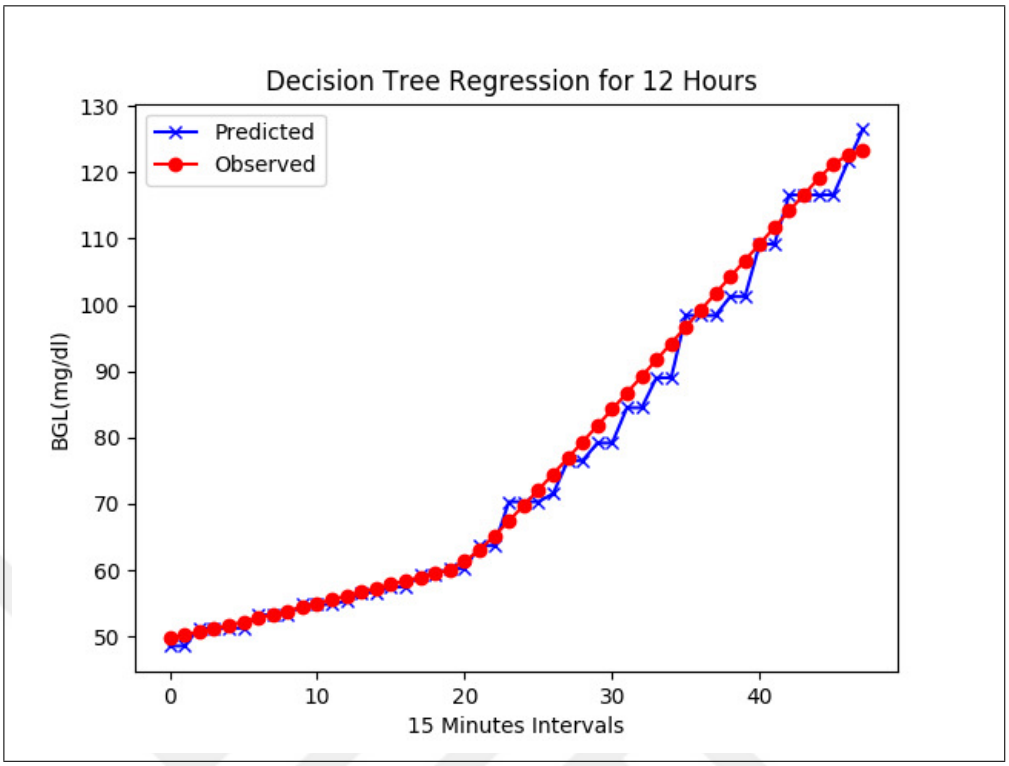

Figure 3.19 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using decision tree regression.

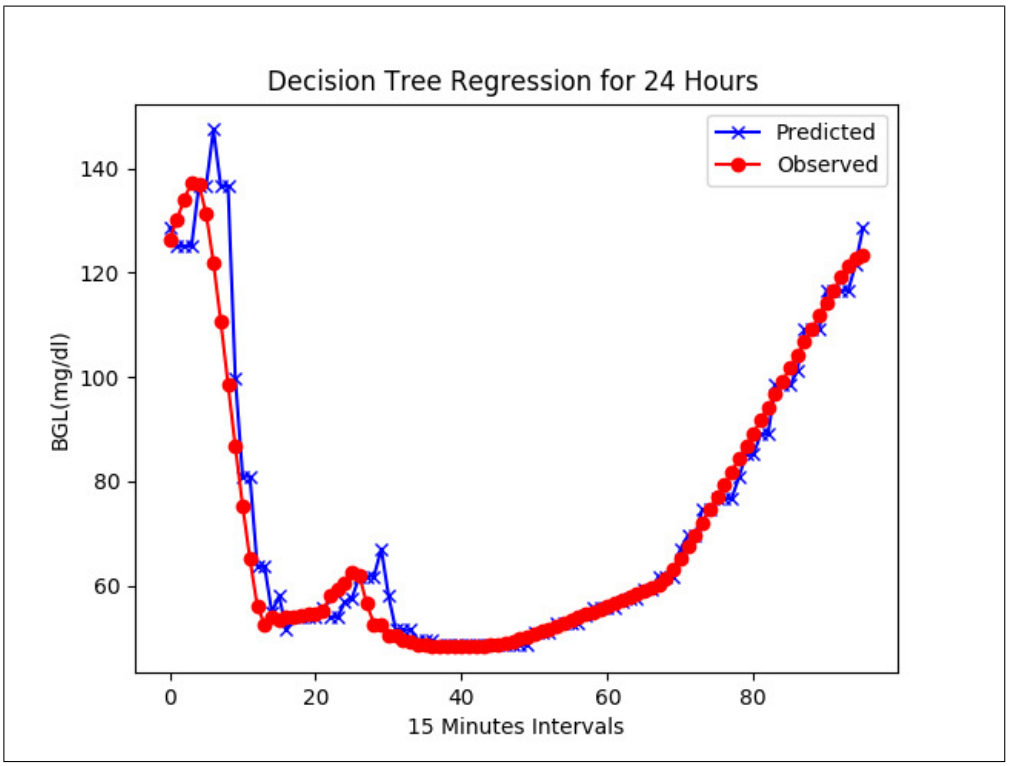

Figure 3.20 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using decision tree regression.

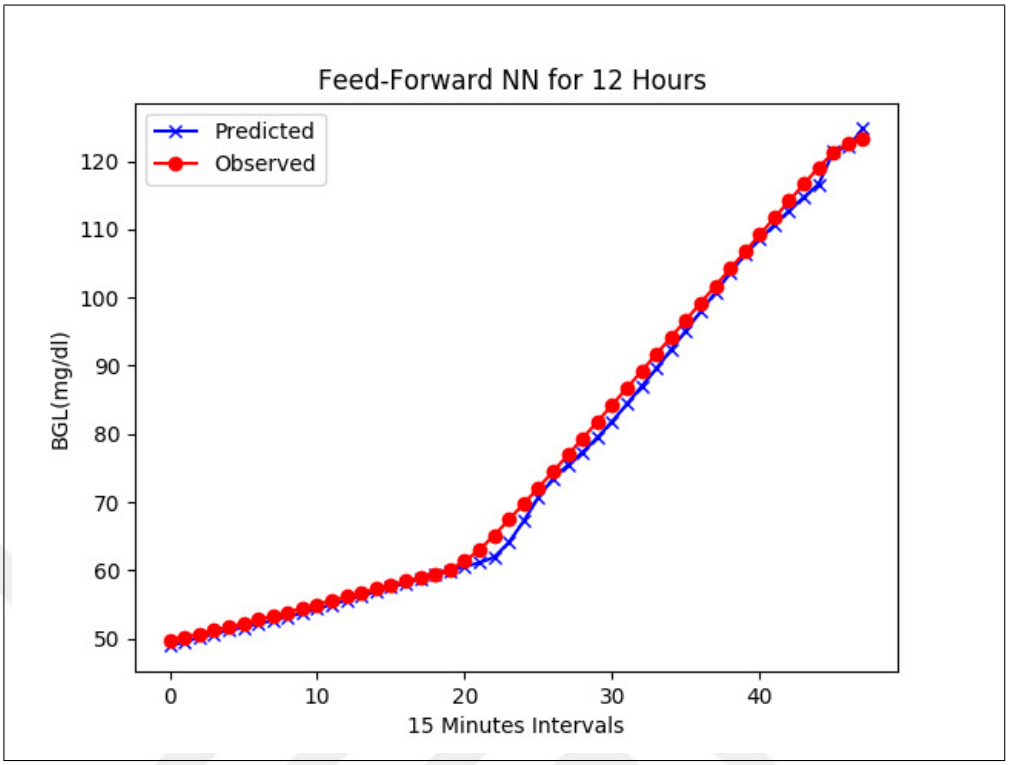

Figure 3.21 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using FFNN.

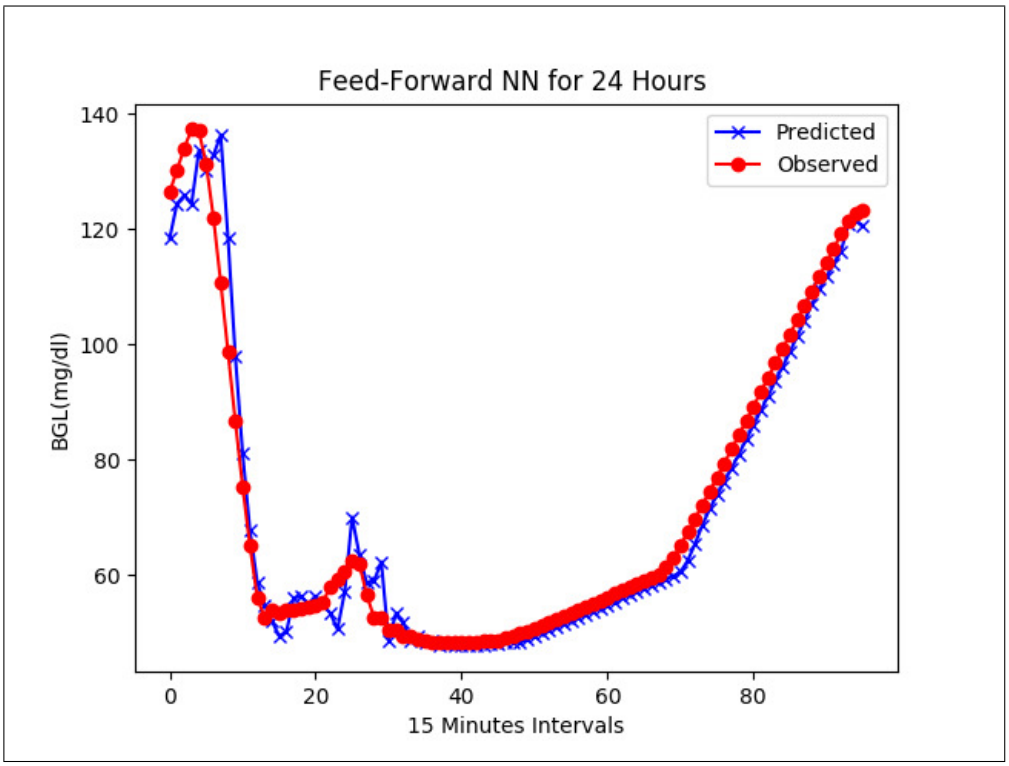

Figure 3.22 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using FFNN.

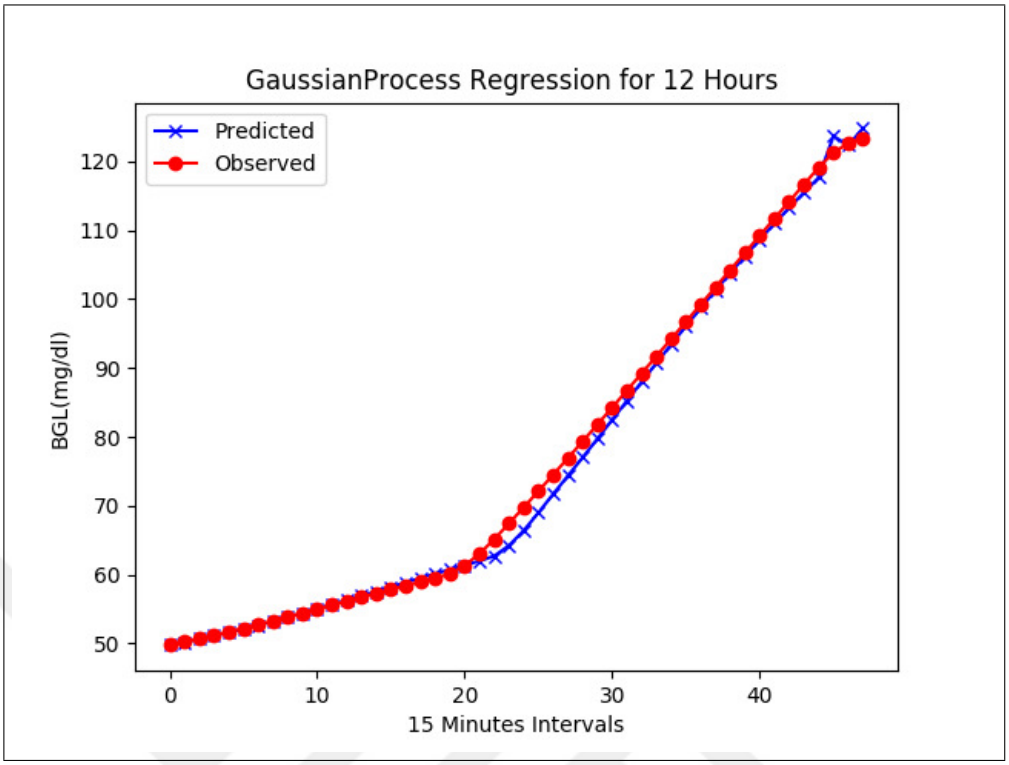

Figure 3.23 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using Gaussian processreg ression.

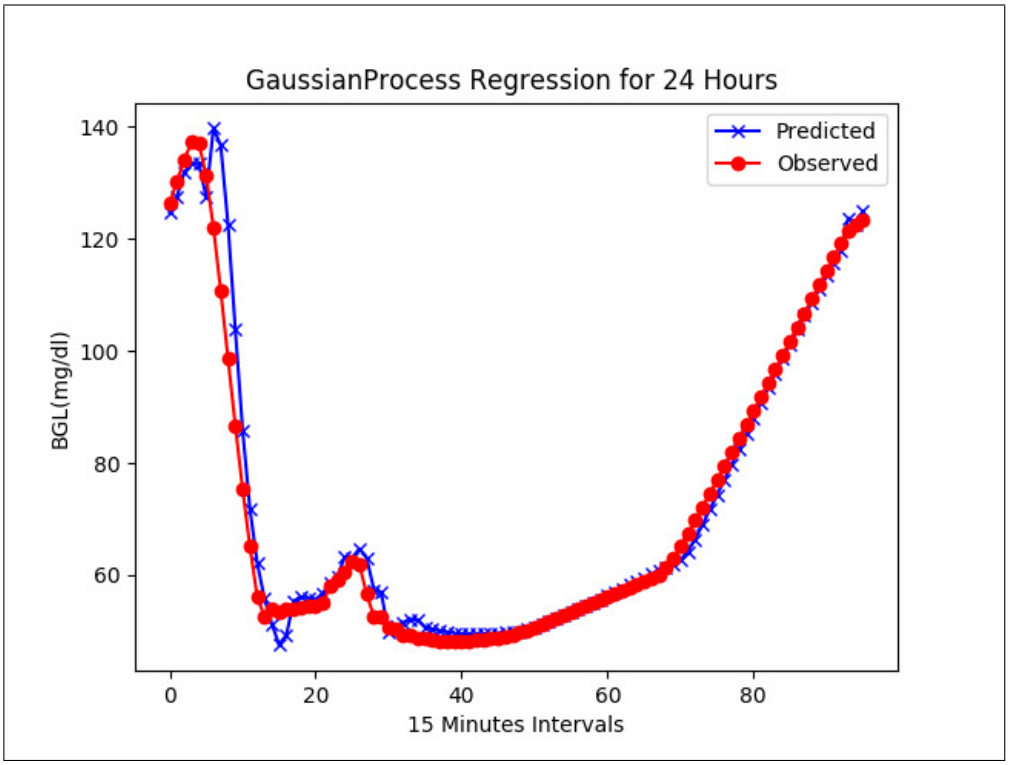

Figure 3.24 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using Gaussian process regression.

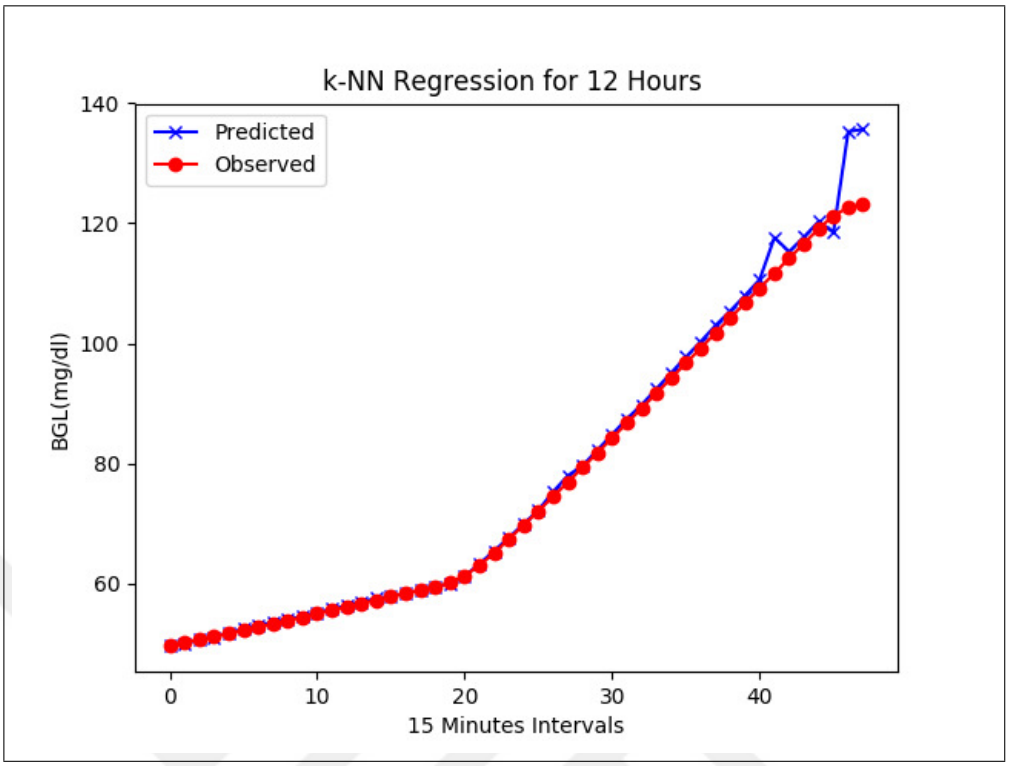

Figure 3.25 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using k-NN regression.

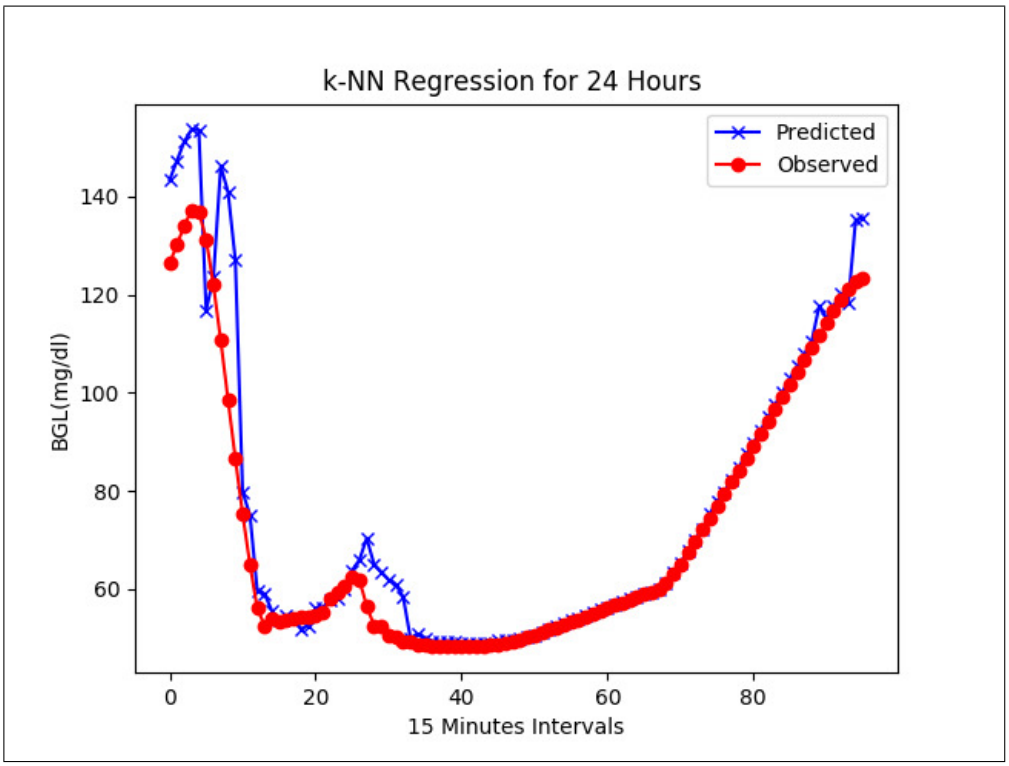

Figure 3.26 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using k-NN regression.

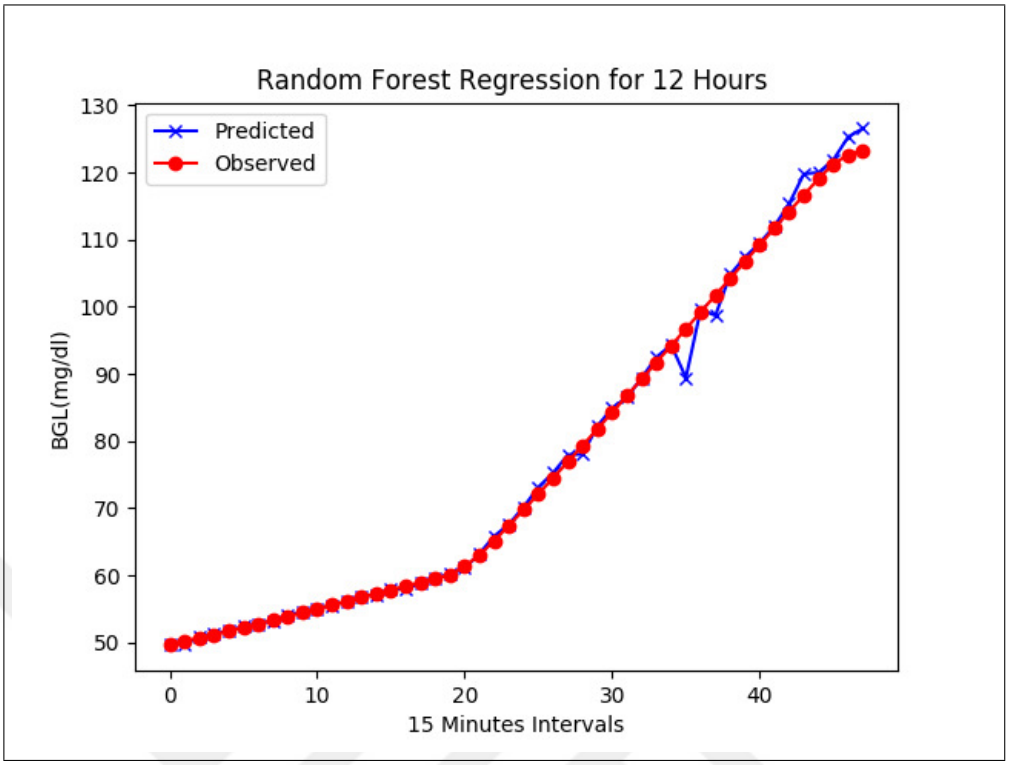

Figure 3.27 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using random forest regression.

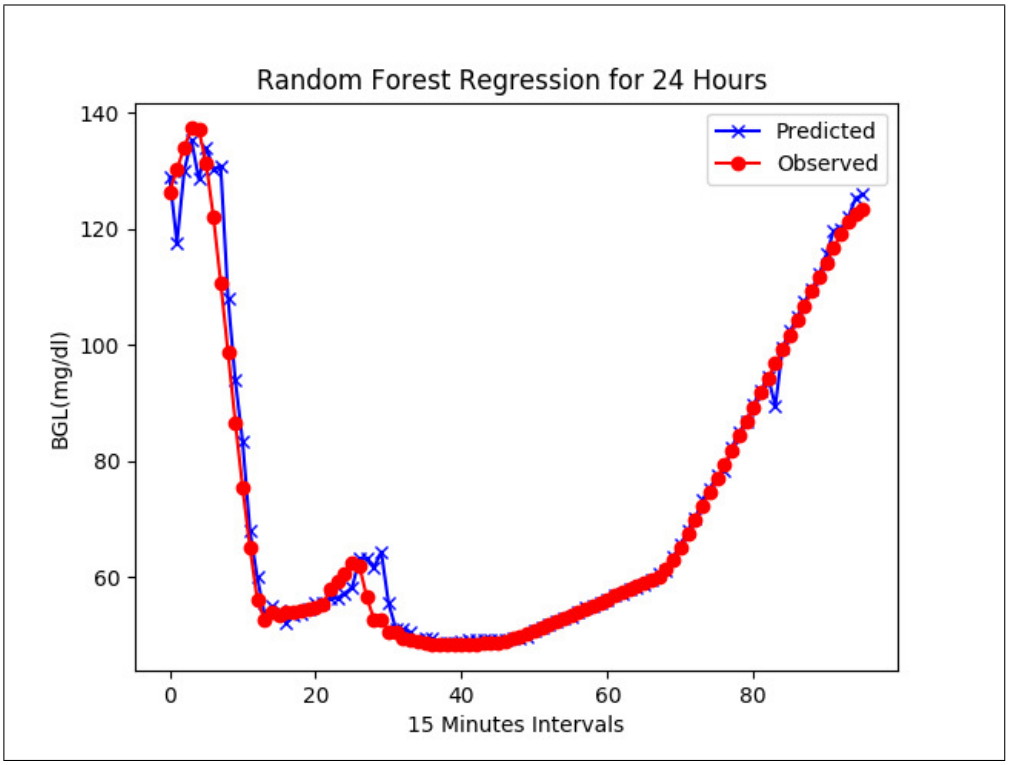

Figure 3.28 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using random forest regression.

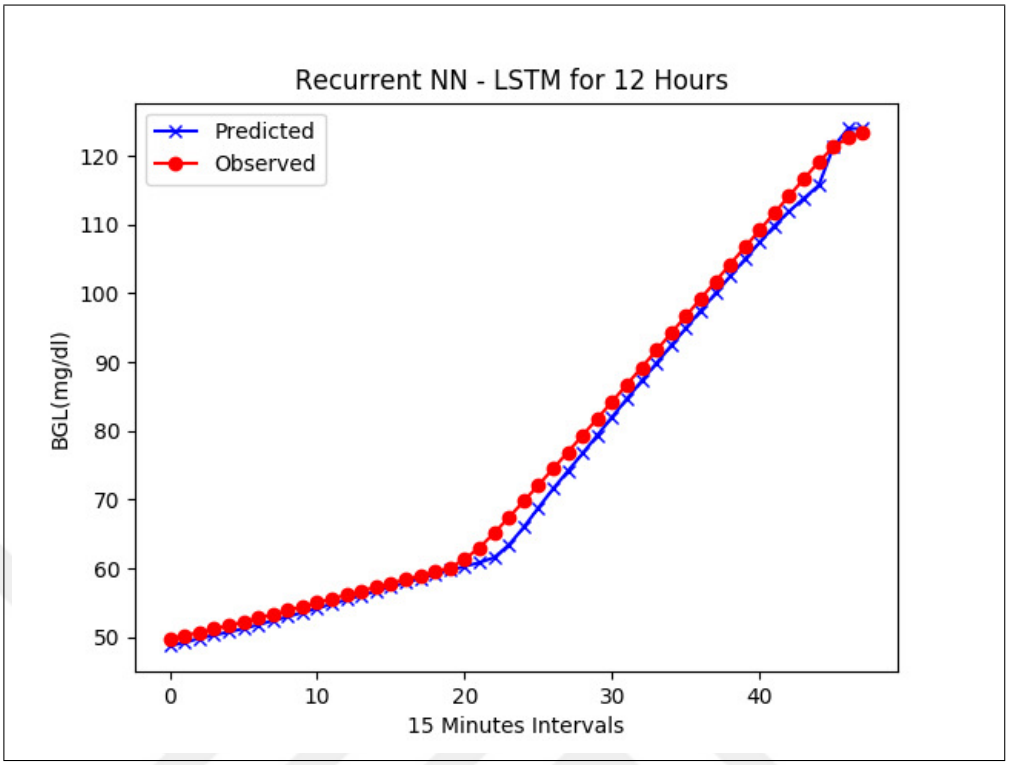

Figure 3.29 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using LSTM.

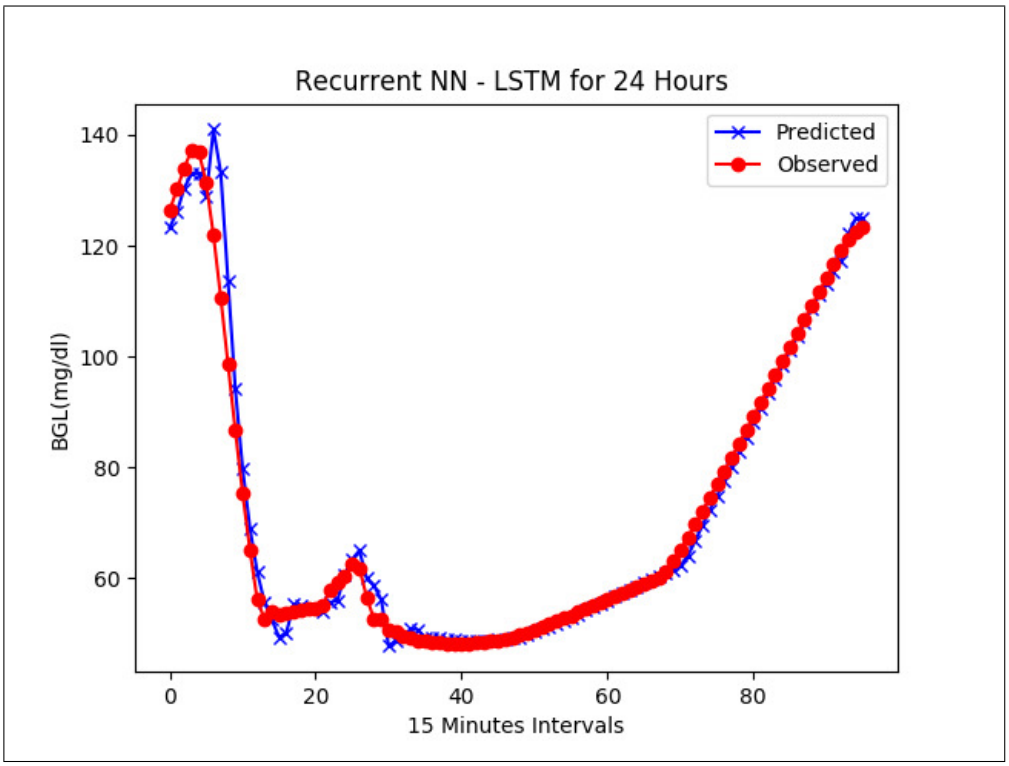

Figure 3.30 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using LSTM.

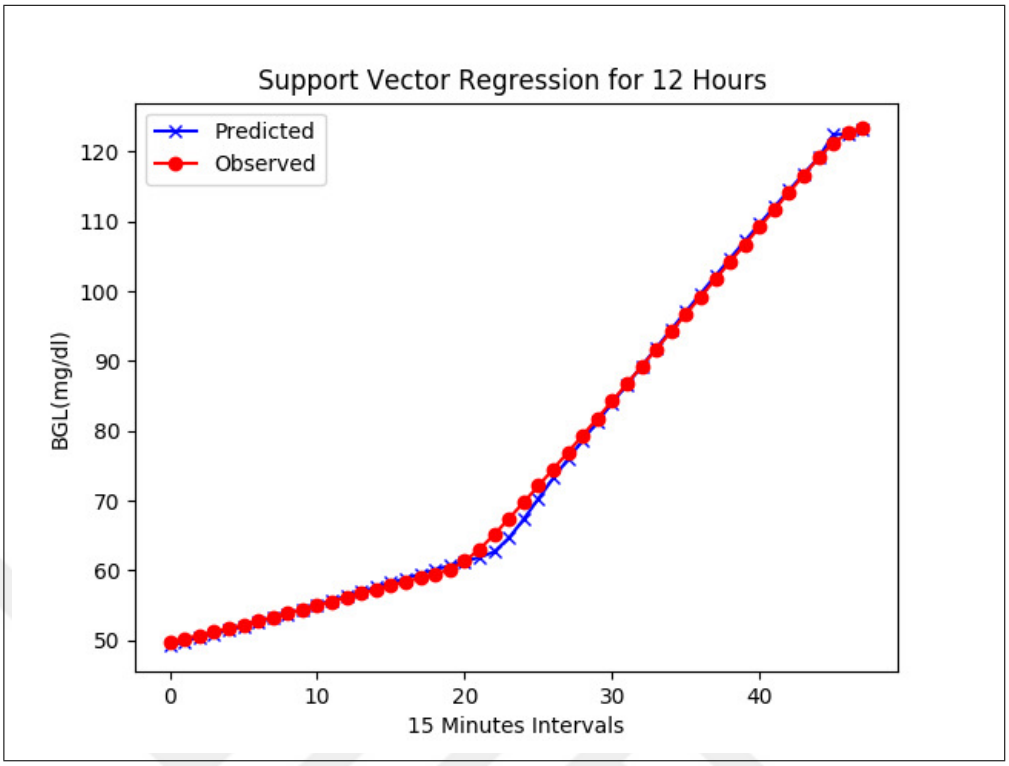

Figure 3.31 Predicting BG values 30 minutes ahead from 00:00 to 12:00 using support vector regression.

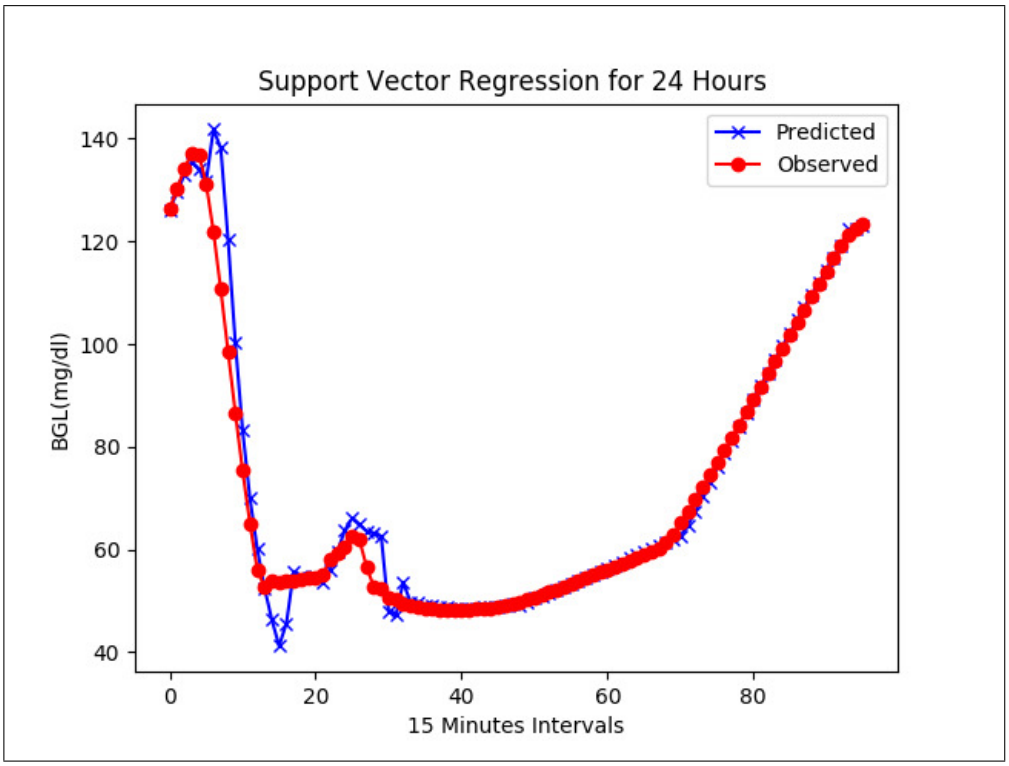

Figure 3.32 Predicting BG values 30 minutes ahead from 00:00 to 00:00 next day using support vector regression.

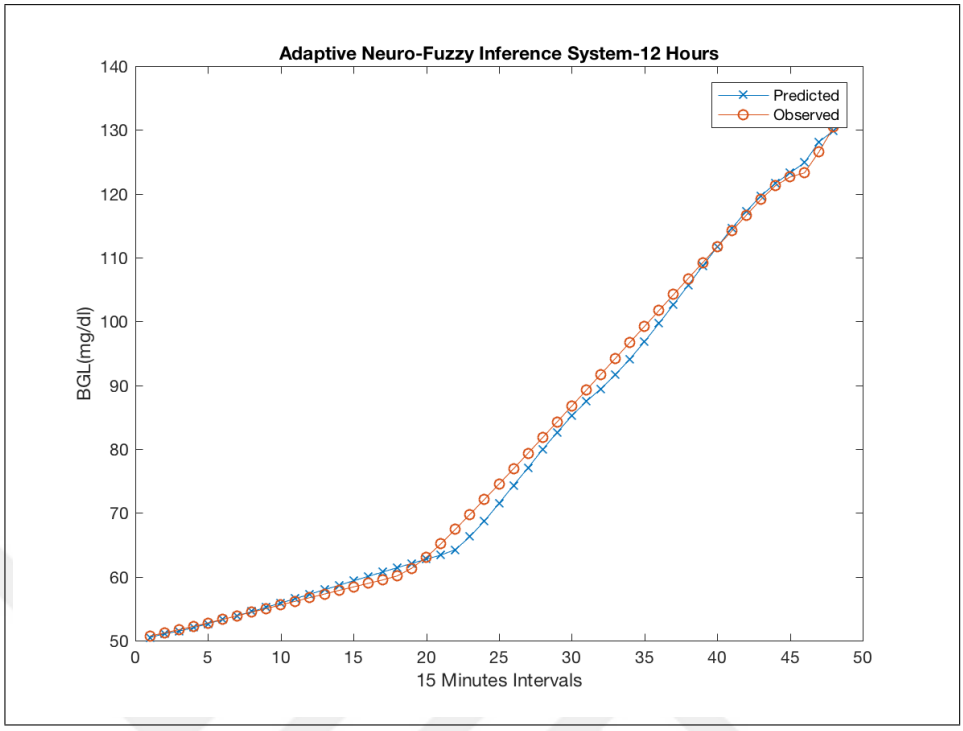

Figure 3.33 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using ANFIS.

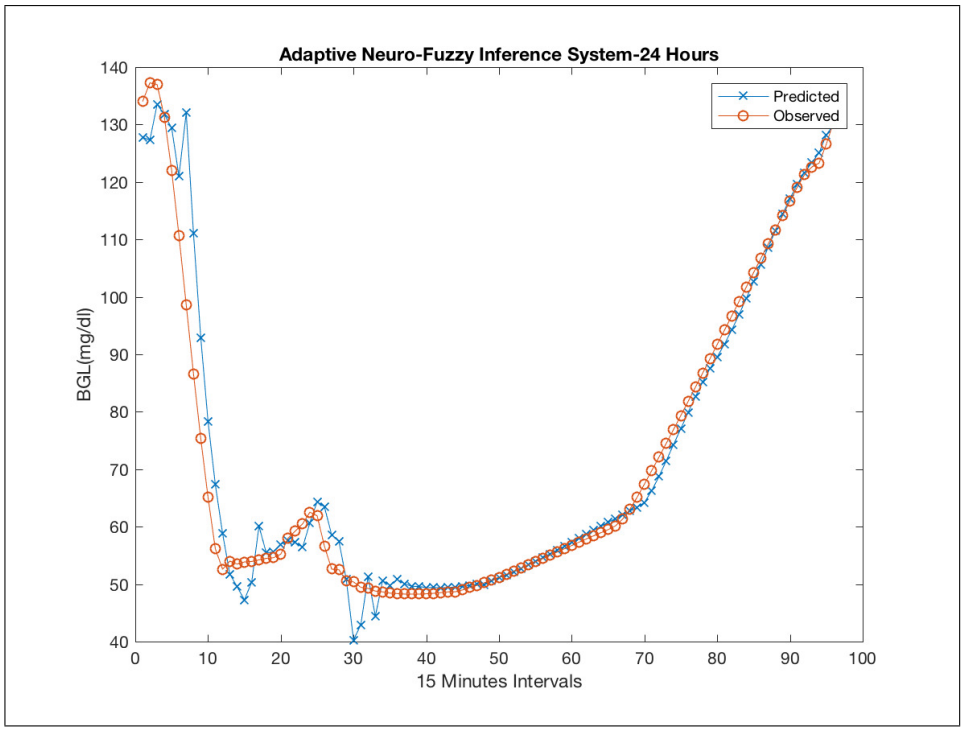

Figure 3.34 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using ANFIS.

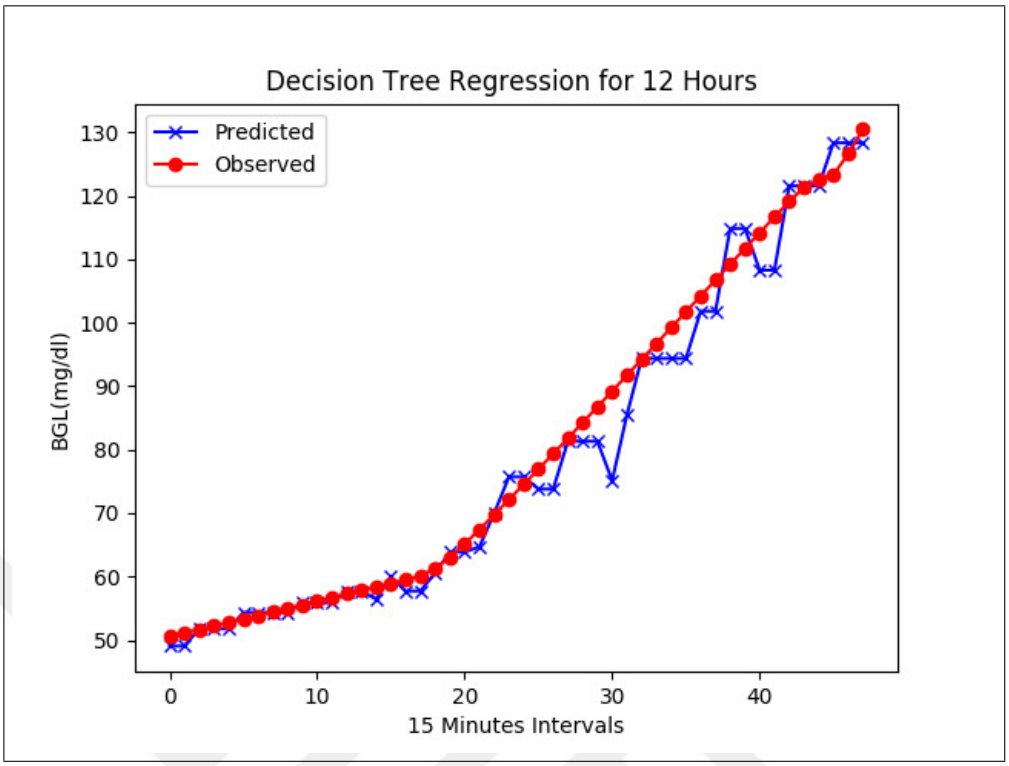

Figure 3.35 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using decision tree regression.

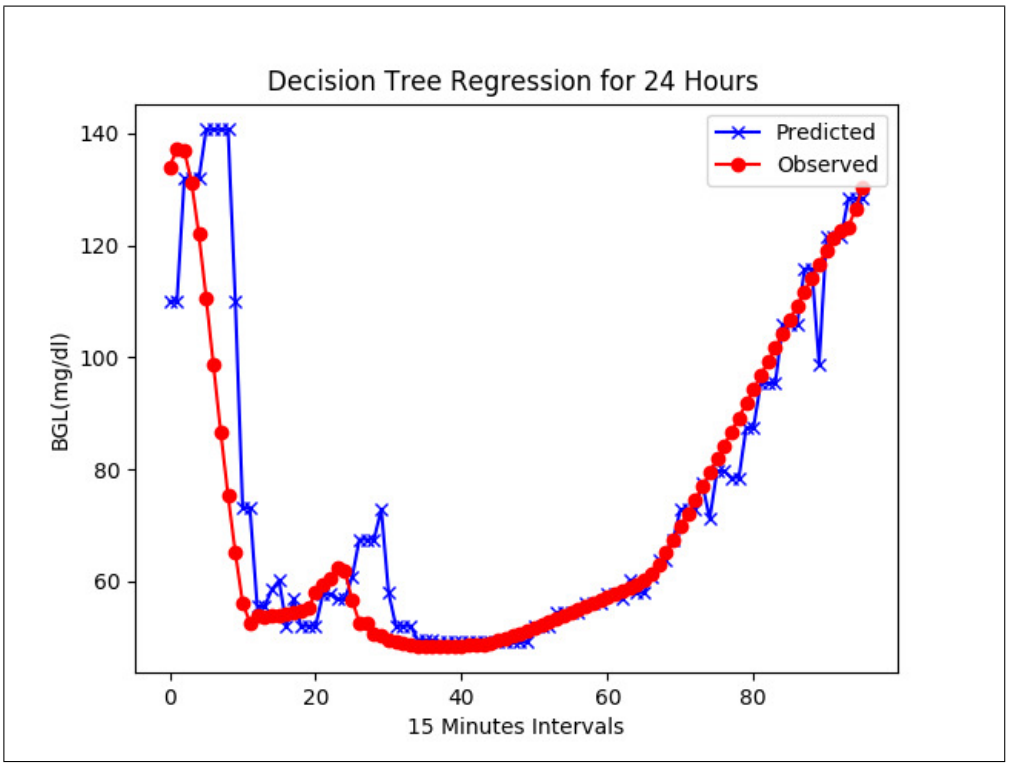

Figure 3.36 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using decision tree regression.

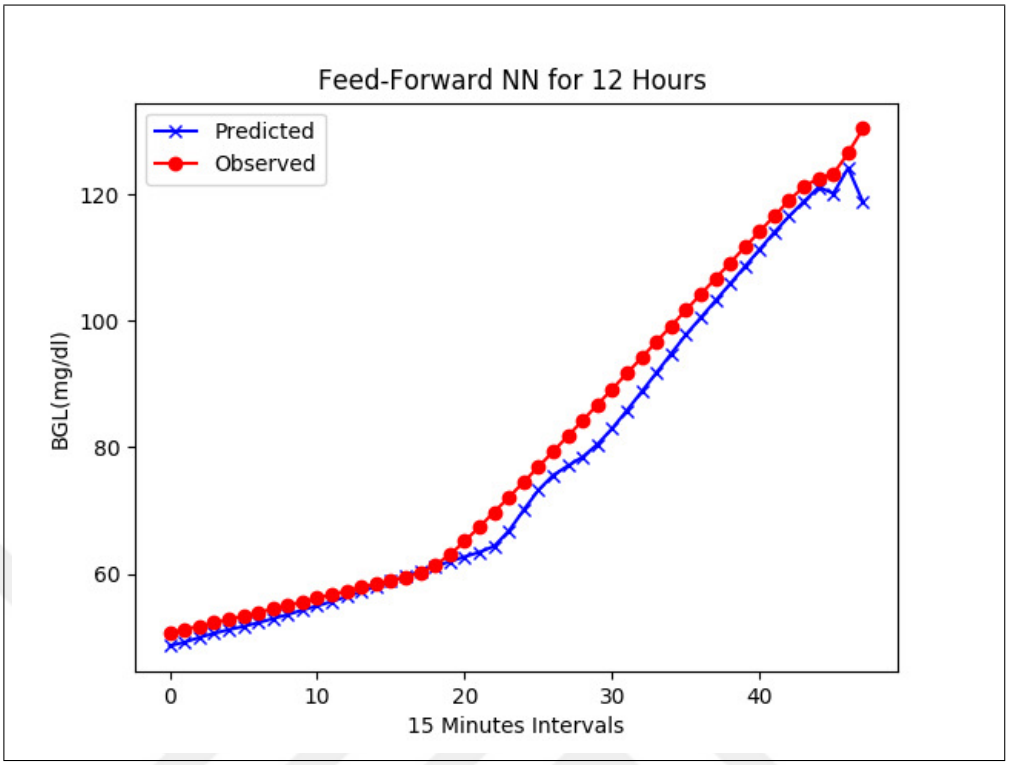

Figure 3.37 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using FFNN.

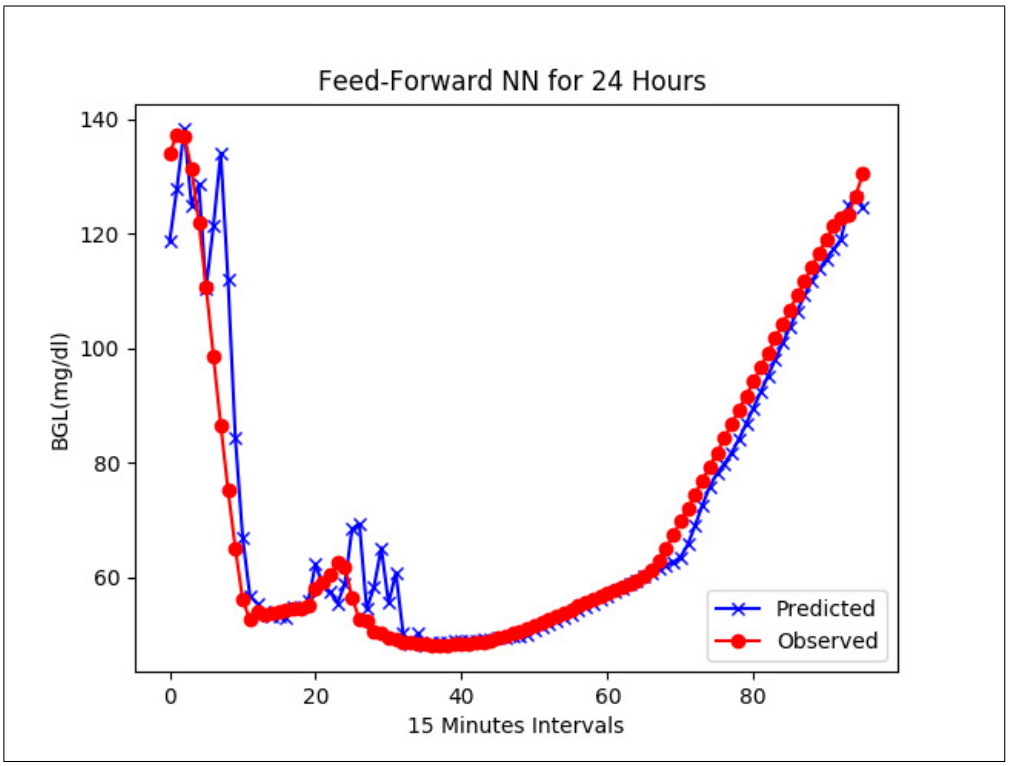

Figure 3.38 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using FFNN.

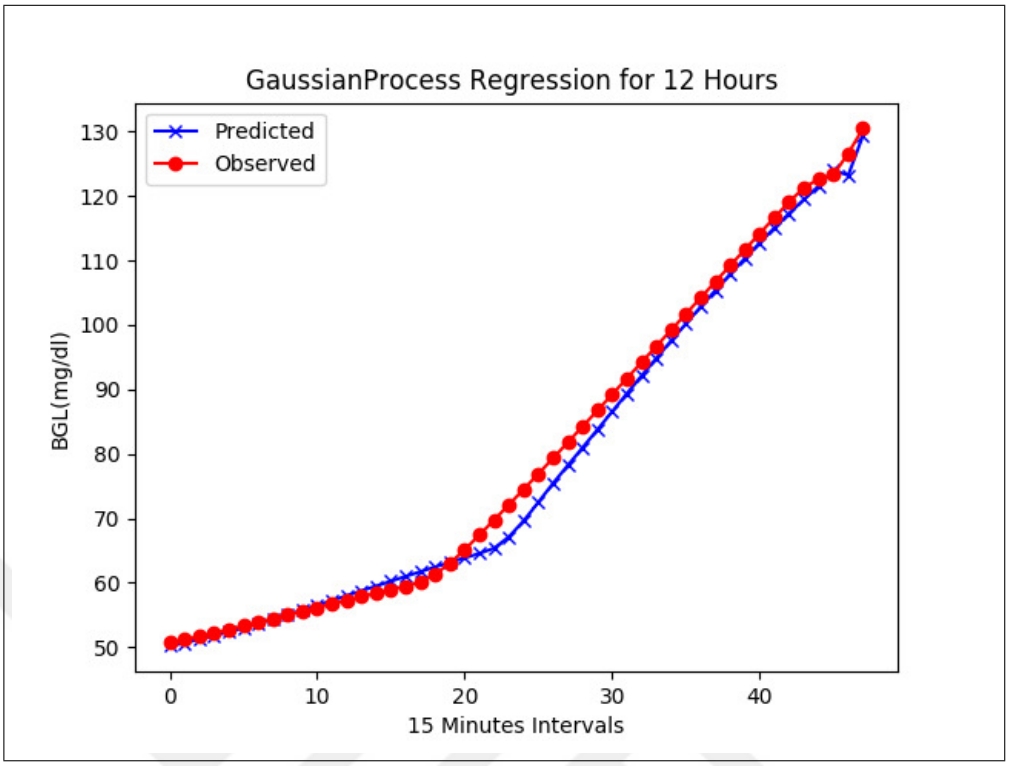

Figure 3.39 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using Gaussian process regression.

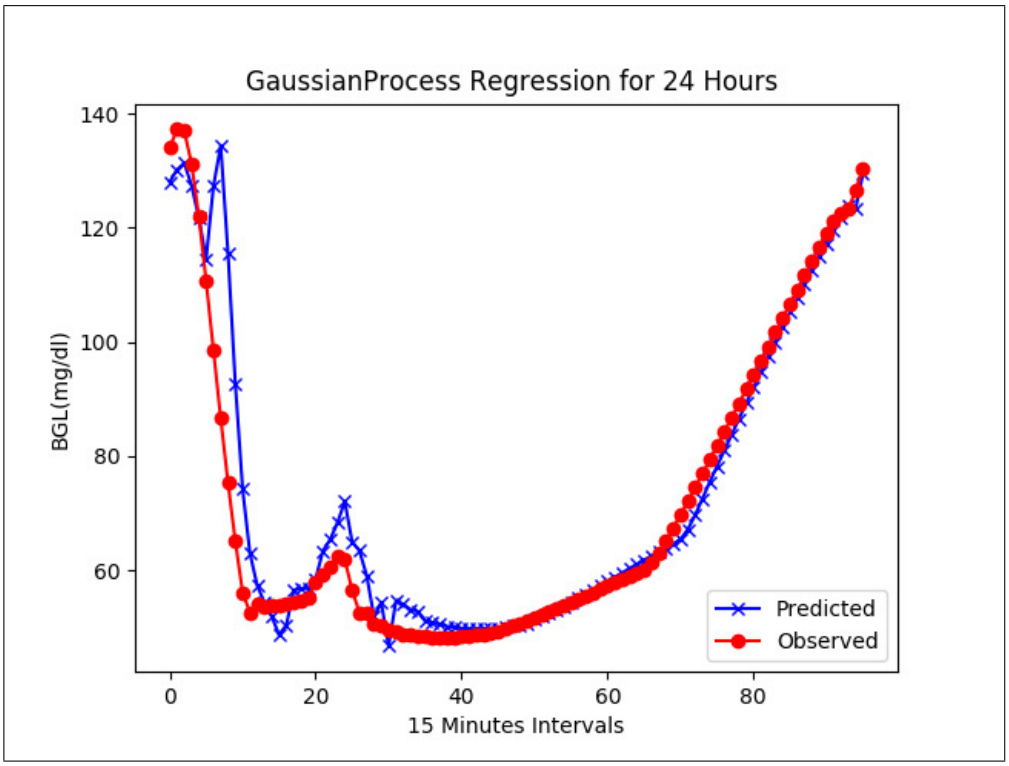

Figure 3.40 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using Gaussian process regression.

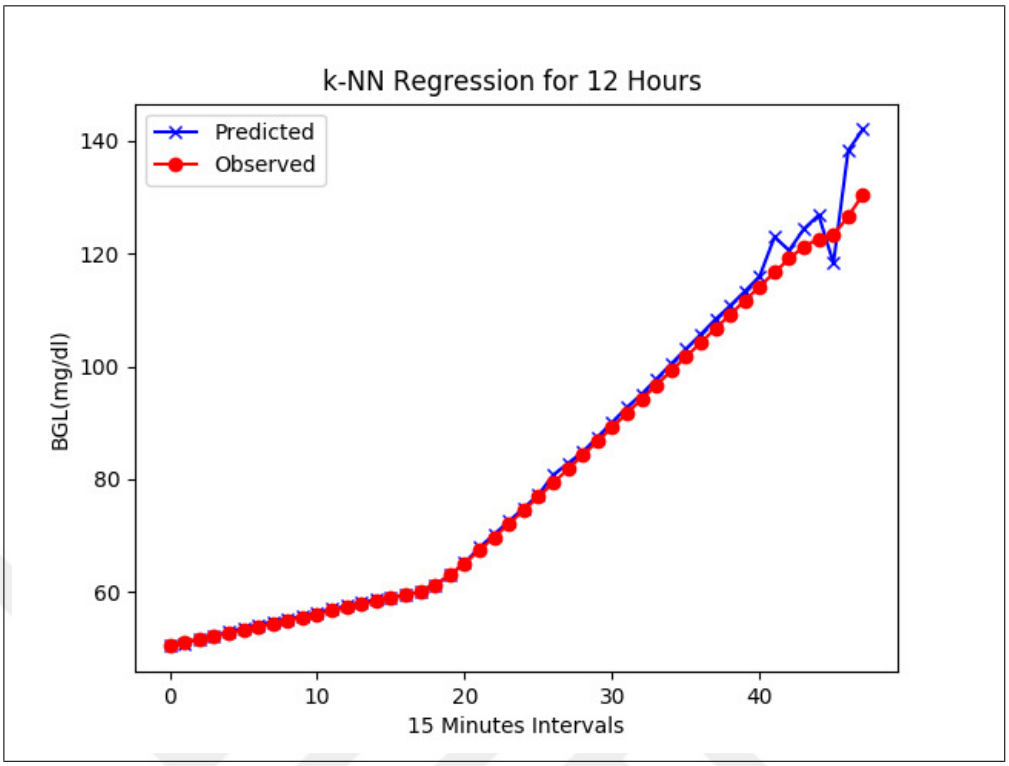

Figure 3.41 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using k-NN regression.

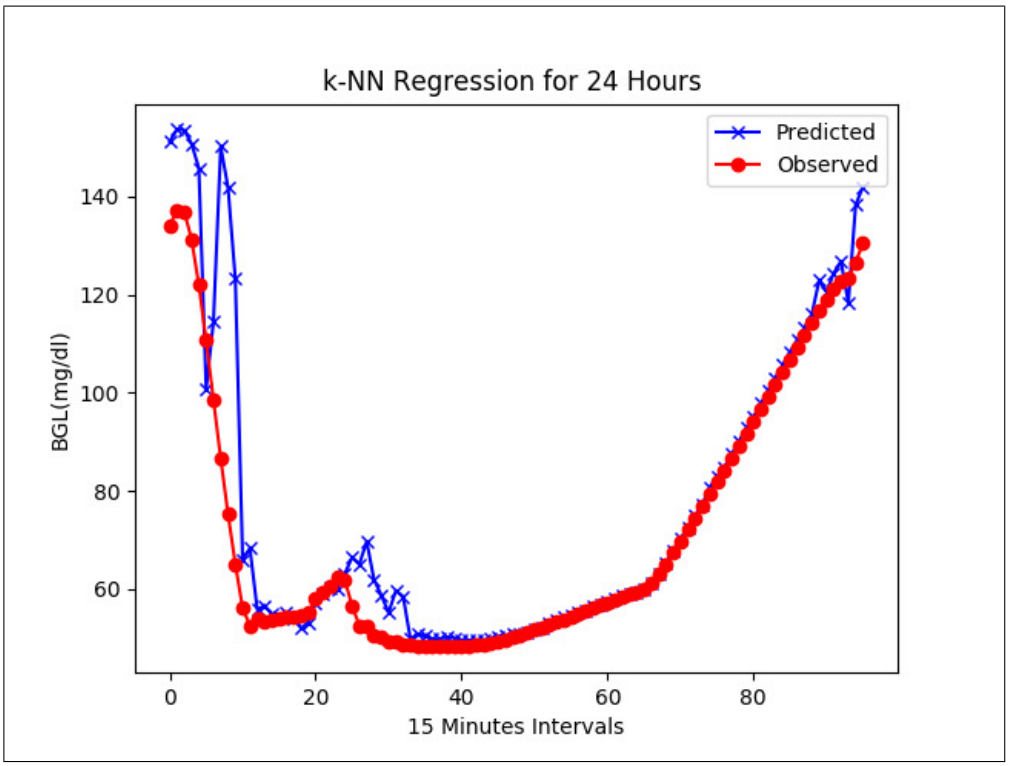

Figure 3.42 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using k-NN regression.

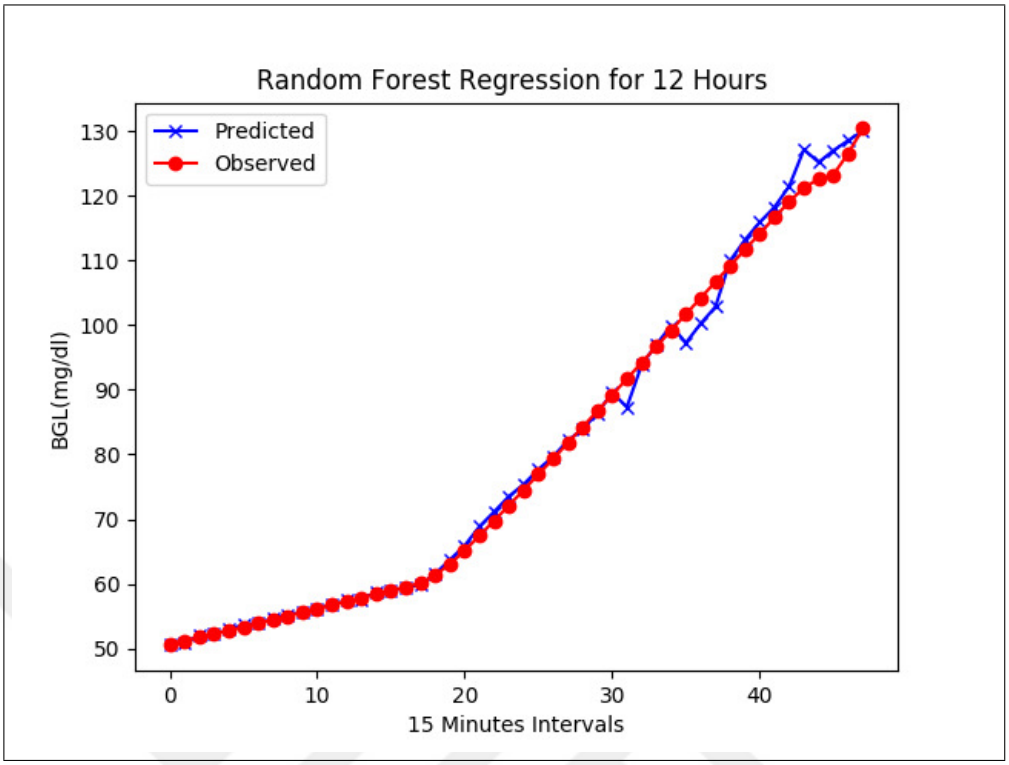

Figure 3.43 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using random forest regression.

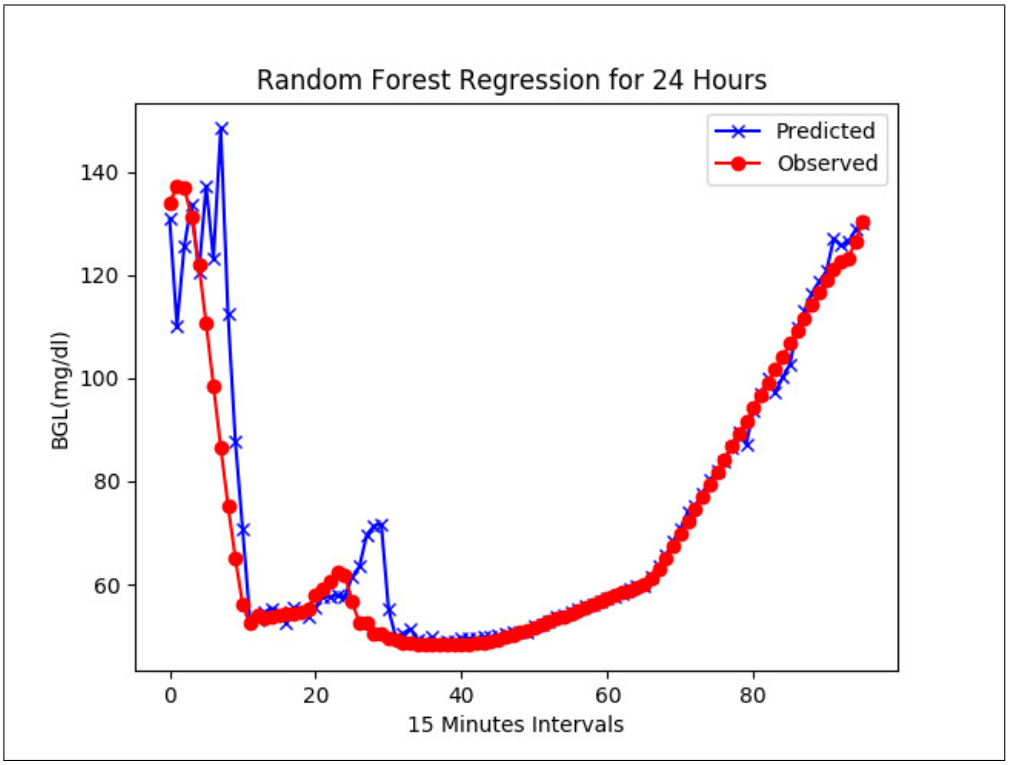

Figure 3.44 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using random forest regression.

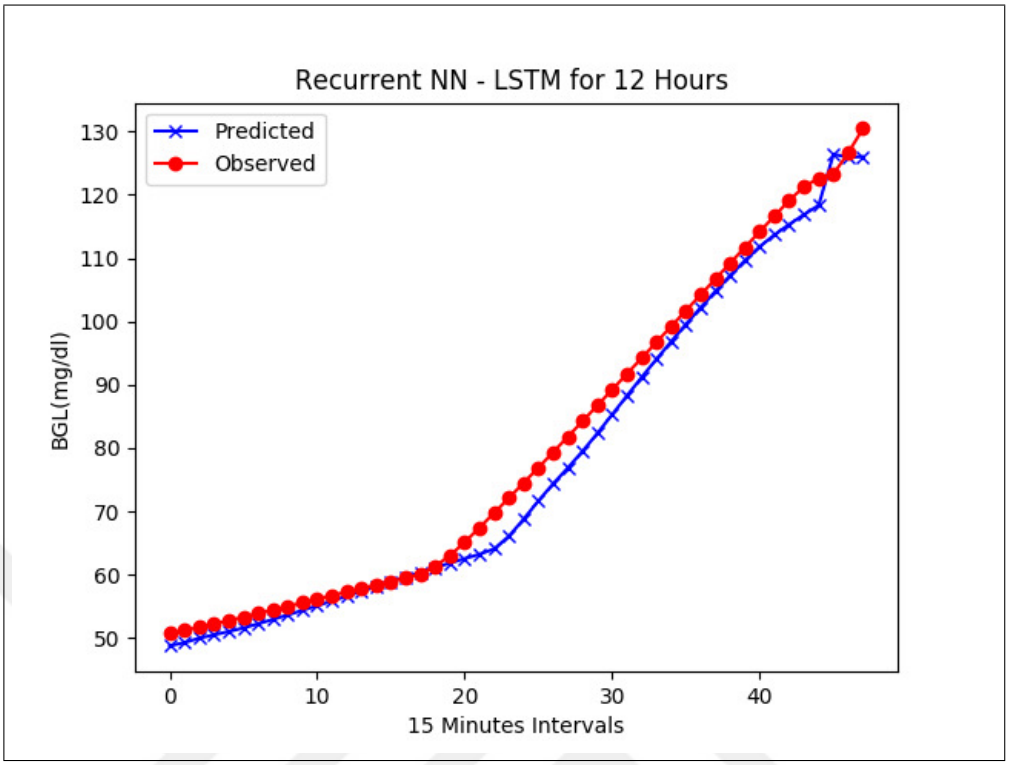

Figure 3.45 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using LSTM.

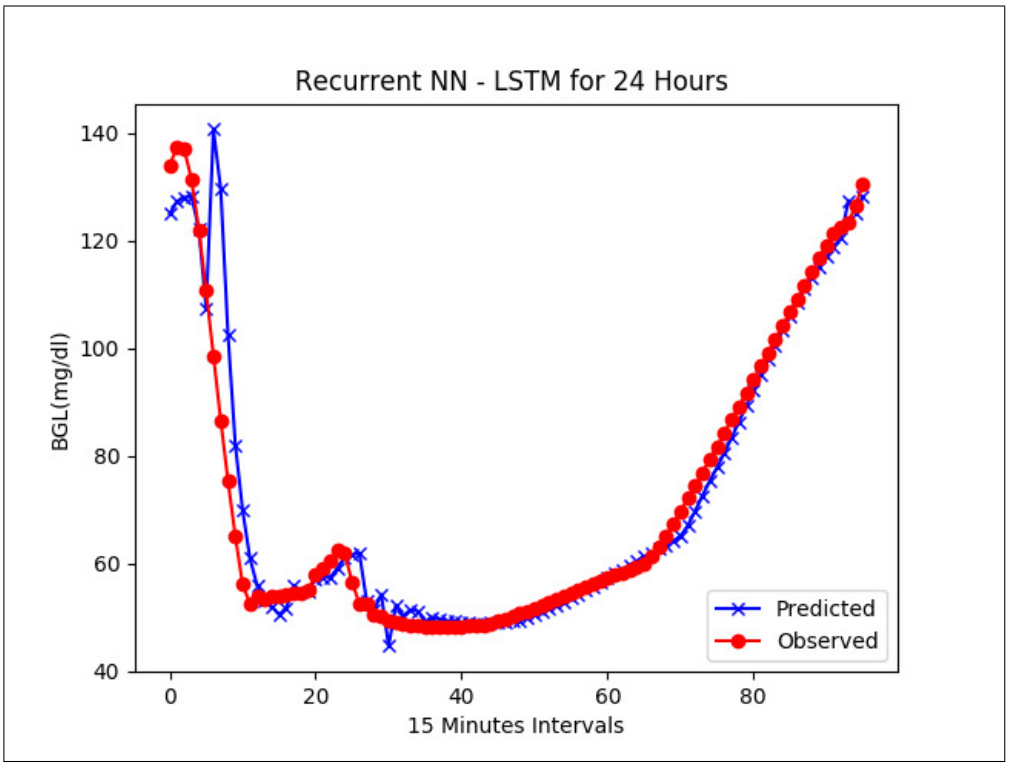

Figure 3.46 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using LSTM.

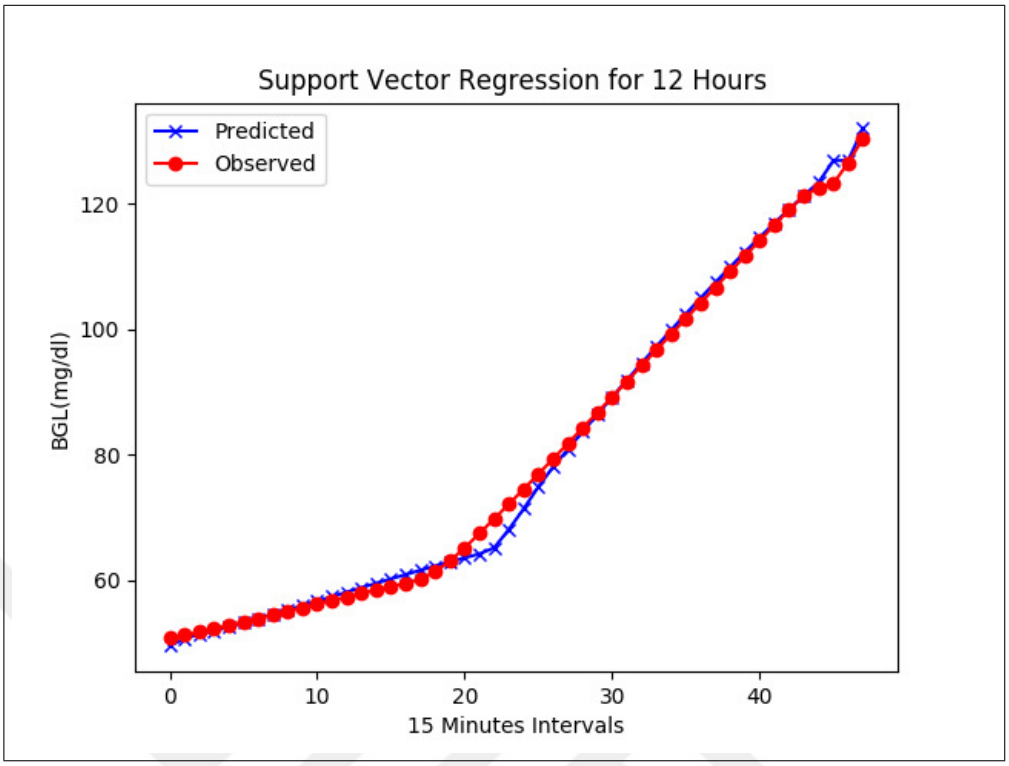

Figure 3.47 Predicting BG values 60 minutes ahead from 00:00 to 12:00 using support vector regression.

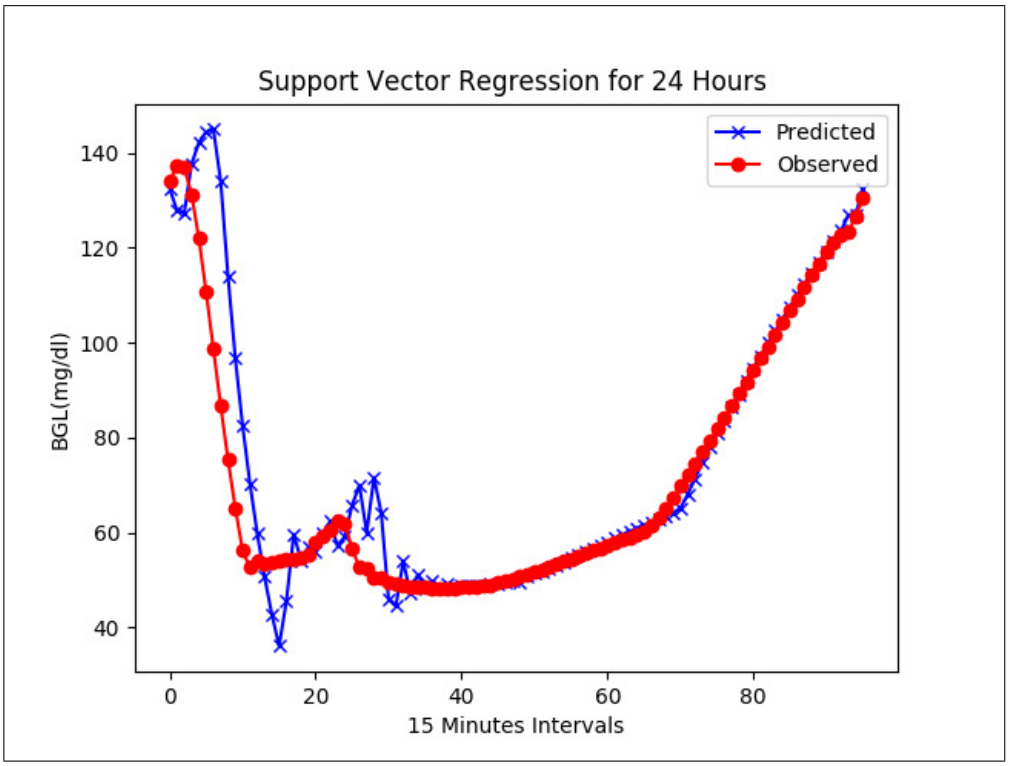

Figure 3.48 Predicting BG values 60 minutes ahead from 00:00 to 00:00 next day using support vector regression.

Table 3.7
The prediction errors of algorithms in RMSE (mg/dl) on 60 minutes prediction horizon with stress test data.

Algorithm	12 hours	24 hours	R2
Decision Tree Regression	22.05	27.13	0.79
FFNN	11.27	18.64	0.77
Gaussian Process Regression	11.89	19.30	0.87
k-NN Regression	26.02	32.87	0.78
Random Forest Regression	18.36	26.10	0.91
Recurrent Neural Network: LSTM	12.60	19.80	0.85
Support Vector Regression	12.49	20.37	0.88
Neuro-Fuzzy Network	9.78	14.88	0.90

Table 3.8
Clarke's error grid analysis, MARD AND CC values of algorithms for predictions of 15 minutes ahead, during 12 hours period.

Algorithms	AR	BE	MARD	CC
DTR	100%	0%	1.37%	0.99774
FFNN	100%	0%	0.92%	0.99957
GPR	100%	0%	0.86%	0.99929
k-NNR	100%	0%	0.98%	0.99610
RFR	100%	0%	0.37%	0.99976
RNN	100%	0%	1.31%	0.99953
SVR	100%	0%	0.42%	0.99982
ANFIS	100%	0%	0.28%	0.99994

Table 3.9
Clarke's error grid analysis, MARD AND CC values of algorithms for predictions of 15 minutes ahead, during 24 hours period.

Algorithms	AR	BE	MARD	CC
DTR	100%	0%	2.92%	0.98944
FFNN	100%	0%	3.43%	0.99527
GPR	100%	0%	2.27%	0.99268
k-NNR	100%	0%	4.03%	0.98088
RFR	100%	0%	1.80%	0.99508
RNN	100%	0%	1.82%	0.99563
SVR	100%	0%	1.81%	0.99471
ANFIS	100%	0%	0.91%	0.99908

Table 3.10
Clarke's error grid analysis, MARD AND CC values of algorithms for predictions of 30 minutes ahead, during 12 hours period.

Algorithms	AR	BE	MARD	CC
DTR	100%	0%	1.96%	0.99680
FFNN	100%	0%	1.06%	0.99926
GPR	100%	0%	1.17%	0.99876
k-NNR	100%	0%	1.09%	0.99615
RFR	100%	0%	0.78%	0.99835
RNN	100%	0%	2.02%	0.99885
SVR	100%	0%	0.76%	0.99947
ANFIS	100%	0%	0.60%	0.99973

Table 3.11
Clarke's error grid analysis, MARD AND CC values of algorithms for predictions of 30 minutes ahead, during 24 hours period.

Algorithms	AR	BE	MARD	CC
DTR	98.958%	1.041%	4.39%	0.97163
FFNN	100%	0%	3.08%	0.98702
GPR	100%	0%	3.40%	0.98426
k-NNR	98.958%	1.041%	4.98%	0.96824
RFR	100%	0%	2.51%	0.99072
RNN	100%	0%	2.77%	0.98920
SVR	100%	0%	3.22%	0.98444
ANFIS	100%	0%	1.90%	0.99585

Table 3.12
Clarke's error grid analysis, MARD AND CC values of algorithms for predictions of 30 minutes ahead, during 12 hours period.

Algorithms	AR	BE	MARD	CC
DTR	100%	0%	2.95%	0.99083
FFNN	100%	0%	1.91%	0.99684
GPR	100%	0%	2.05%	0.99791
k-NNR	100%	0%	1.32%	0.99679
RFR	100%	0%	1.14%	0.99777
RNN	100%	0%	3.19%	0.99811
SVR	100%	0%	1.60%	0.99803
ANFIS	100%	0%	0.28%	0.99994

Table 3.13
Clarke's error grid analysis, MARD AND CC values of algorithms for predictions of 30 minutes ahead, during 24 hours period.

Algorithms	AR	BE	MARD	CC
DTR	95.833%	4.166%	8.62%	0.89537
FFNN	96.875%	3.125%	6.67%	0.94366
GPR	97.916%	2.083%	5.97%	0.95364
k-NNR	98.958%	1.041%	7.26%	0.92988
RFR	95.833%	4.166%	5.77%	0.93892
RNN	97.916%	2.083%	4.65%	0.95983
SVR	97.916%	2.083%	6.34%	0.94135
ANFIS	98.958%	1.041%	0.91%	0.99908

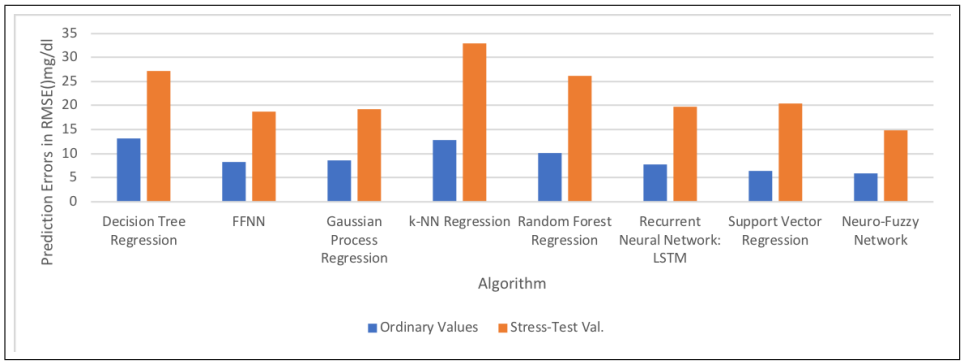

Figure 3.49 Comparison of Prediction Error During 12 Hours of Prediction for Ordinary Data vs Stress-Test Data.

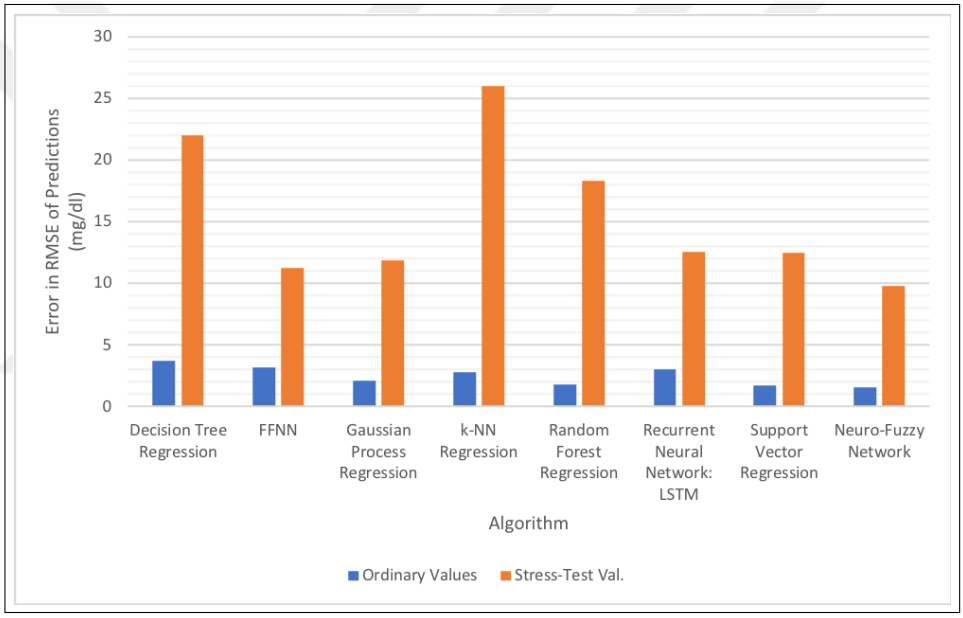

Figure 3.50 Comparison of Prediction Error During 24 Hours of Prediction for Ordinary Data vs Stress-Test Data.

4. DISCUSSION

In this study, the recent literature and machine learning applications on blood glucose prediction have been reviewed. We compared those machine learning algorithms among themselves with the input, fed from simulation program AIDA. The data consisted of CH intake amount, insulin intake amount, and blood glucose measurements from the simulator.

All models have good scores for nocturnal periods, however as the results are further investigated, ANFIS, support vector regression and random forest regression have the best results among others. And as the prediction period spreads out to 24 hours, the prediction power decreases. This decrease also occurs in the increase of prediction horizon.

In order to see the comparison of the algorithms among each other, grouping will provide a lot of help. Structurally, they can be divided into parametric and, non-parametric algorithms. Parametric algorithms are neural network type of algorithms, RNN, FFNN and ANFIS. k-NN regression, decision tree regression, random forest regression, Gaussian process regression and SVR are non-parametric algorithms. Parametric algorithms are briefly the ones that have finite number of parameters, or parameters are not influenced by the distribution of data. On the contrary, non-parametric models do not meet those conditions.

Among the parametric models, neural networks, ANFIS have the best result because of the learning method. Instead of optimising learning parameters on nodes, using fuzzy logic rules to learn, leads to faster convergence of model on training data. Although this model has an asymptotic $O(n)$ time complexity, it has an exponential spatial complexity, which makes it very demanding in terms of memory. LSTM and FFNN has better spatial complexity than ANFIS, however worse time complexity $O(n^2)$ where n roughly denotes to number of nodes.

Non-parametric models have their best results with SVR followed by random forest regression. Although random forest regression provides a powerful prediction performance, it loses its effectiveness on unseen target values and inputs. This is why it was outperformed by SVR. SVR also had better accuracy than another method that employs RBF as kernel, Gaussian prediction regressor. This is mainly because of that the Gaussian process regression produces some randomisation over the model parameters given the input data, which is not suitable for the simulation data which has no noise. Comparing SVR to k-NN regression, k-NN had higher errors in predictions. Although it is a powerful regression method, this poor accuracy is caused by the dimensionality of the data which is high for k-NN to fit well. The reason that SVM performed better in prediction times is because of its computational complexity compared to Gaussian process regression, $O(n^2)$ and $O(n^3)$ respectively.

We presented a way to compare algorithms on a unified simulation dataset for specific patients. The good sides of using simulation data can be listed as follows:

1. Provides simplicity and increases accuracy by eliminating external factors like: stress, exercise, illness, pregnancy and glucose detector noise
2. AIDA allows fast and easy BGL data extraction
3. Helps to encourage perform in-silico testing with different patient cases, and clinical trial with real patient data will be warranted.

Besides the simulation advantages, this study has its strengths in:

1. Providing selection criteria for ML algorithms by generating accuracy and runtime analytics of several algorithms.
2. Applying different prediction horizons, prediction periods, reliability metrics on an extended number of algorithms will provide a management mechanism for the patients using CGMs

3. Comparison of parametric, non-parametric and neural network algorithms for patient specific cases.

The standardization of the data has its advantage in providing a distribution of BG values with a pattern. To investigate the effect of this standardization and pattern on the algorithms, we also made a sample run with stress test data with a 60 minutes PH. According to this experiment, as can be seen in Table 3.7, the order of the performance of the algorithms have not changed except for random forest regression. This is due to the tree based algorithms are not accurate with outlier data. Also there has been a visible loss in prediction accuracy. The decrease in accuracy indicate that using simulation data for patient specific cases, makes it easier to monitor BG fluctuations and make better predictions.

In order to prove that the results produced by the models comply with the blood glucose meter standards, MARD, and EGA results are investigated and reflected in Table 3.8 - 3.13. According to these results, accurate readings must be over 95% and MARD values must not exceed 10% [35] - [36]. Predictions of 15 minutes ahead for all models have AR of 100%. Decision tree regression and k-NN regression has the worst AR for 30 and 60 minutes predictions as expected. Same applies to MARD also.

However, it is a setback that this study could not be extended to real world data and extra features (physical activity, stress level, etc.), although this is not the scope for simulation dataset. Another point might be missed through this study is that the frequency of the blood glucose measurements. Conventional CGMs take measurements with 5 minutes, and our simulation produces measurements in every 15 minutes. Also, the simulation cannot generate data with effects like physiological factors, pregnancy etc.

We aimed to keep development platforms same for all of the algorithms. However, absence of a useful library for ANFIS in python, has directed us to use MATLAB and its neuro-fuzzy network library to accomplish this task. Since libraries are ab-

straction over the actual algorithms, developer can only fine-tune hyperparameters. Thus, there remains a blank point in equalizing the computing conditions on which algorithms run.

Comparing our study to its inspiring pioneers [4] - [7] - [11], we have expanded the covering area with number and diversity of algorithms. We also improved the results in terms of accuracy. Our study has better results compared to real world patient data studies as expected. For support vector machine regression, in the 60 minutes prediction horizon, best acquired result is 7.14mg/dl in terms of RMSE [3]. For this algorithm our study has RMSE of 6.34 mg/dl. However, our study had worse performance compared to a similar study over AIDA with recurrent neural networks, with 3.02mg/dl to 2.70mg/dl in terms of RMSE. And the ANFIS model as a neuro-fuzzy network has better performance than the wavelet fuzzy neural network for 30 minutes PH where we predicted 2.53 mg/dl and Pappada et al. predicted with RMSE of 15.64mg/dl. For 60 minutes PH, the RMSE results are 5.81 mg/dl (ANFIS) to 25.5 mg/dl (WFNN) [11]. For FFNN, our study also has the better result than real world examples [5] - [6] - [11].

Our comparison study has been able to produce outcomes for the blood glucose prediction along with the increasing interest in machine learning. The findings can be listed as:

1. Running machine learning algorithms on data generated from AIDA is applicable for in silico studies. This will work as a preliminary test for the real-world cases.

2. This paper suggested a lookup mechanism for the feature selection and model building guide for the predictive studies.

3. Proving the fact that using Gaussian process regression as a supervised machine learning model with multivariate inputs. Showing that it can compete with other algorithms in terms of accuracy is also encouraging for future use.

We are also able to make a contribution to clinicians who are working over T1DM patients by providing a BG prediction tool to foresee the situation of the patients. In a practical approach, the results of this study help patients by predicting their BG values given their food and insulin intakes. This will alert patients before having serious consequences caused by hypoglycaemia or hyperglycaemia, and also direct them to manage their lifestyles better.

For the future studies, models we investigated might be extended to real world data, as it is the biggest disadvantage of this study. Thus, the effect of new features on models could be observed. Since our main concern is making a comparison with common machine learning techniques, a platform which will present every model and their results, according to prediction horizon and feature set would be useful for the researchers in this domain. Also future researchers who want to benefit from this study should use it not only as a look-up table but also a way to use synthetic data for this domain.

Similar review studies are either comparing certain algorithms or compilation of several studies on blood glucose prediction. Combining and getting beyond both approaches, we both reproduced techniques used on recent studies with simulation dataset and compared them to original results for short term predictions. For the practical use of the patients, they can make predictions over given prediction horizons and take precautions for hypoglycaemic and hyperglycaemic situations before occurring.

5. CONCLUSION

Following our experiments and results we reached multiple conclusions. One of them is that using AIDA for blood glucose prediction gave out similar results to studies that use real world values. Comparing the machine learning algorithms in this study, we have found out that ANFIS is the best algorithm in terms of accuracy performance and decision tree regression is the fastest in predicting time due to its structure that holds predefined averages in leaves. In order to test the limits of the models built, we trained them with a dataset which has immediate changes and contains extreme cases, and observed that, although the prediction errors have increased, the order of performance did not change. Considering these we get to the point that using such a simulator with regression models is an appropriate approach for blood glucose prediction domain. Furthermore the outcomes of this study has been narrowed to comparison of GPR and SVR and submitted to ICIETS2018 conference to be held in Karnaraka India [37].

REFERENCES

1. Eren-Oruklu, M., A. Cinar, L. Quinn, and D. Smith, "Estimation of future glucose concentrations with subject-specific recursive linear models," *Diabetes Technology & Therapeutics*, Vol. 11, no. 4, pp. 243–253, 2009.

2. Huzooree, G., K. K. Khedo, and N. Joonas, "Glucose prediction data analytics for diabetic patients monitoring," in *Next Generation Computing Applications (NextComp), 2017 1st International Conference on*, pp. 188–195, IEEE, 2017.

3. Georga, E. I., V. C. Protopappas, D. Ardigò, M. Marina, I. Zavaroni, D. Polyzos, and D. I. Fotiadis, "Multivariate prediction of subcutaneous glucose concentration in type 1 diabetes patients based on support vector regression," *IEEE Journal of Biomedical and Health Informatics*, Vol. 17, no. 1, pp. 71–81, 2013.

4. Reymann, M. P., E. Dorschky, B. H. Groh, C. Martindale, P. Blank, and B. M. Eskofier, "Blood glucose level prediction based on support vector regression using mobile platforms," in *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*, pp. 2990–2993, IEEE, 2016.

5. Pappada, S. M., B. D. Cameron, P. M. Rosman, R. E. Bourey, T. J. Papadimos, W. Olorunto, and M. J. Borst, "Neural network-based real-time prediction of glucose in patients with insulin-dependent diabetes," *Diabetes Technology & Therapeutics*, Vol. 13, no. 2, pp. 135–141, 2011.

6. Zecchin, C., A. Facchinetti, G. Sparacino, and C. Cobelli, "How much is short-term glucose prediction in type 1 diabetes improved by adding insulin delivery and meal content information to cgm data? a proof-of-concept study," *Journal of Diabetes Science and Technology*, Vol. 10, no. 5, pp. 1149–1160, 2016.

7. Robertson, G., E. D. Lehmann, W. Sandham, and D. Hamilton, "Blood glucose prediction using artificial neural networks trained with the aida diabetes simulator: a proof-of-concept pilot study," *Journal of Electrical and Computer Engineering*, Vol. 2011, p. 2, 2011.

8. Kavakiotis, I., O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda, "Machine learning and data mining methods in diabetes research," *Computational and Structural Biotechnology Journal*, Vol. 15, pp. 104–116, 2017.

9. Oviedo, S., J. Vehí, R. Calm, and J. Armengol, "A review of personalized blood glucose prediction strategies for t1dm patients," *International Journal for Numerical Methods in Biomedical Engineering*, Vol. 33, no. 6, 2017.

10. Plis, K., R. C. Bunescu, C. Marling, J. Shubrook, and F. Schwartz, "A machine learning approach to predicting blood glucose levels for diabetes management.," in *AAAI Workshop: Modern Artificial Intelligence for Health Analytics*, no. 31, pp. 35–39, 2014.

11. Zarkogianni, K., K. Mitsis, E. Litsa, M.-T. Arredondo, G. Fico, A. Fioravanti, and K. S. Nikita, "Comparative assessment of glucose prediction models for patients with type 1 diabetes mellitus applying sensors for glucose and physical activity monitoring," *Medical & Biological Engineering & Computing*, Vol. 53, no. 12, pp. 1333–1343, 2015.

12. Lehmann, E., and T. Deutsch, "Aida2: A mk. ii automated insulin dosage advisor," *Journal of Biomedical Engineering*, Vol. 15, no. 3, pp. 201–211, 1993.

13. Lehmann, E., and T. Deutsch, "A physiological model of glucose-insulin interaction in type 1 diabetes mellitus," *Journal of Biomedical Engineering*, Vol. 14, no. 3, pp. 235–242, 1992.

14. Lehmann, E. D., "Experience with the internet release of aida v4. 0-http://www. diabetic. org. uk/aida. htm-an interactive educational diabetes simulator," *Diabetes Technology & Therapeutics*, Vol. 1, no. 1, pp. 41–54, 1999.

15. Lehmann, E., and T. Deutsch, "A physiological model of glucose-insulin interaction," in *Engineering in Medicine and Biology Society, 1991. Vol. 13: 1991., Proceedings of the Annual International Conference of the IEEE*, pp. 2274–2275, IEEE, 1991.

16. Lehmann, E., I. Hermanyi, and T. Deutsch, "Retrospective validation of a physiological model of glucose-insulin interaction in type 1 diabetes mellitus," *Medical Engineering & Physics*, Vol. 16, no. 3, pp. 193–202, 1994.

17. Hornik, K., D. Meyer, and A. Karatzoglou, "Support vector machines in r," *Journal of statistical software*, Vol. 15, no. 9, pp. 1–28, 2006.

18. Smola, A. J., and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, Vol. 14, no. 3, pp. 199–222, 2004.

19. vanGerven, M., and S. Bohte, "Artificial neural networks as models of neural information processing: Editorial on the research topic artificial neural networks as models of neural information processing," *Frontiers in Computational Neuroscience*, Vol. 11, p. 114, 2017.

20. Schmidhuber, J., "Deep learning in neural networks: An overview," *Neural Networks*, Vol. 61, pp. 85–117, 2015.

21. Svozil, D., V. Kvasnicka, and J. Pospichal, "Introduction to multi-layer feed-forward neural networks," *Chemometrics and Intelligent Laboratory Systems*, Vol. 39, no. 1, pp. 43–62, 1997.

22. Graves, A., M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, no. 5, pp. 855–868, 2009.

23. Gers, F. A., J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Computation*, Vol. 12, no. 10, pp. 2451–2471, 2000.

24. Friedl, M. A., and C. E. Brodley, "Decision tree classification of land cover from remotely sensed data," *Remote Sensing of Environment*, Vol. 61, no. 3, pp. 399–409, 1997.

25. Rathore, S. S., and S. Kumar, "A decision tree regression based approach for the number of software faults prediction," *ACM SIGSOFT Software Engineering Notes*, Vol. 41, no. 1, pp. 1–6, 2016.

26. Pal, M., "Random forest classifier for remote sensing classification," *International Journal of Remote Sensing*, Vol. 26, no. 1, pp. 217–222, 2005.

27. Adusumilli, S., D. Bhatt, H. Wang, P. Bhattacharya, and V. Devabhaktuni, "A low-cost ins/gps integration methodology based on random forest regression," *Expert Systems with Applications*, Vol. 40, no. 11, pp. 4653–4659, 2013.

28. Kramer, O., "K-nearest neighbors," in *Dimensionality Reduction with Unsupervised Nearest Neighbors*, pp. 13–23, Springer, 2013.

29. Altman, N. S., "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, Vol. 46, no. 3, pp. 175–185, 1992.

30. Abraham, A., "Adaptation of fuzzy inference system using neural learning," in *Fuzzy Systems Engineering*, pp. 53–83, Springer, 2005.

31. Jang, J.-S., "Anfis: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, no. 3, pp. 665–685, 1993.

32. Ebden, M., *et al.*, "Gaussian processes for regression: A quick introduction," *The Website of Robotics Research Group in Department on Engineering Science, University of Oxford*, 2008.

33. Kingma, D. P., and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

34. Greff, K., R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 28, no. 10, pp. 2222–2232, 2017.

35. Parkes, J. L., S. L. Slatin, S. Pardo, and B. H. Ginsberg, "A new consensus error grid to evaluate the clinical significance of inaccuracies in the measurement of blood glucose.," *Diabetes Care*, Vol. 23, no. 8, pp. 1143–1148, 2000.

36. Bailey, T. S., "Clinical implications of accuracy measurements of continuous glucose sensors," *Diabetes Technology & Therapeutics*, Vol. 19, no. S2, pp. S–51, 2017.

37. Doğugün Özkaya, A. G., "Using simulation generated synthetic data for benchmark testing of blood glucose prediction algorithms," *Submitted to IEEE Sponsored Innovations in Engineering, Technology and Sciences, ICIETS2018 conference to be held in Karnaraka India, on dates 20th-21st September 2018*, 2018.