WESTERN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE 9603/9063b – Data Analytics Foundations

Assignment 1: Forecasting

Student name: Jinliang Zhang
Student number: 25******8
Email address: jzhan964@uwo.ca

• Description of the selected forecasting problem (4 points)

In this assignment, I choose “Speed and Stopping Distances of Cars” dataset, used as “cars” in R datasets, to implement a regression forecasting problem. Through training from this dataset, the algorithms are going to forecast the distances taken to stop when the speed of cars is input.

• Description of available data (attributes, context, quantity…). Clearly indicate what attributes and/or parts you have used (4 points)

This cars dataset includes 50 observations on 2 variables. One is the speed of cars in miles per hour (mph) named “speed”, the other is the distances taken to stop in foot (ft) named “dist”. It is worth mentioning that the data were recorded in the 1920s.

Here are head 11 samples in “cars” dataset:

	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10
7	10	18
8	10	26
9	10	34
10	11	17
11	11	28

The numbers in the first column represent the serial number of the data. The numbers in the second column show the speed of cars, and the numbers in last column are the distances for cars taken to stop. In the experiment, all data is used to implement regression forecasting.

• *Short overview of the selected algorithms (4 points)*

Three algorithms are used in this experiment.

a. Linear regression.

Linear regression is a type of regression analysis that models the relationship between one or more independent and dependent variables using a least squares function called a linear regression equation. This function is a linear combination of one or more model parameters called regression coefficients. The case of only one independent variable is called simple regression, and the case of more than one independent variable is called multiple regression. In this experiment, simple regression has been used to forecast.

b. Support vector regression.

Support vector regression is a variant of support vector machine. SVM belongs to generalized linear classifiers. It maps the vectors into a higher-dimensional space in which an interval maximum hyperplane is established. Two classifications are separated by the hyperplane on both sides to maximize the distance between them. Assuming that the greater the distance between classification, the smaller total error of the classifier. For SVR, as it is a regression model, the goal of it is to find a hyperplane separating the data to minimize the distance between two classifications.

c. Neural network.

Neural networks are computational models which mimic the structure and

function of biological neural networks and are used to estimate or approximate functions. A typical neural network has three parts: architecture, activity rule, and learning rule. The structure specifies the variables in the network and their relations. Activity Rule is used to define how neurons change their stimuli based on the activity of other neurons. Learning rules specify how weights in the network adjust over time. There are many packages for neural networks in R. Package 'nnet' provides feed-forward neural network algorithm.

• *Specifics about how algorithms were applied and the evaluation procedure (4 points)*

Programming language: R.

Integrated Development Environment: Rstudio.

Steps:

1. Divide the data into 2 groups as training set and test set.
2. For linear regression, R has a function called "lm", the format is lm (formula, data) in which formula is the form of the model to fit, and data contains the data used to fit the model. Formula is represented as follows: $Y \sim X_1 + X_2 + ... + X_k$ (The left of ~ is the response variable, the right is the various variable separated by the plus sign).
3. Before applying support vector regression, a library named "e1071" need to be installed. There is a function call "svm" in this package, the format is svm (formula, data = NULL, ..., subset, na.action =na.omit, scale = TRUE). I used three parameters which are formula, data and type in the algorithm. Formula and data are used as same as linear regression function. There are five forms of type which are C-classification, nu-classification, one-classification, eps-regression, nu-regression five forms. I chose eps-regression to implement regression forecasting.
4. Before applying neural network, a package named "nnet" should be installed. The format of function "nnet" is nnet (formula, data, weights, ..., subset, na.action, contrasts = NULL). I used six parameters which are formula, data, size, decay, maxit and linout. Size is for the number of neurons in the hidden layer of neural network. Decay is for a modified bias parameter for neuron input weights. Maxit is for the maximum number of feedback iterations. I set linout as "T" for regression.
5. Then, function "predict" is used to predict the test set for these three algorithms. To evaluate these three algorithms, root mean square error (rmse) is used as method to evaluate the accuracy of these algorithm. Through comparing rmse, I can get which algorithm can be better fitted and used to forecast.

• *Accuracy comparison (4 points)*

The accuracy of the forecasting is shown as below:

rmseLR	10.3289727263006
rmseNN	10.3452587955458
rmseSVR	8.96947167703491

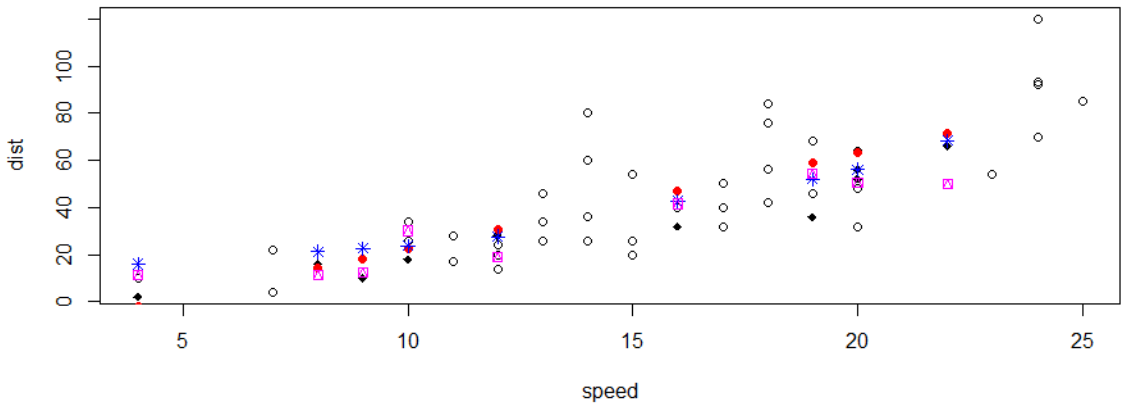
When rmse is used to evaluate, linear regression and neural network have similar values of rmse. Although, support vector regression has smaller value in once prediction. For once experiment, SVR algorithm gets better results than linear regression and neural network.

• ***Code.***

```
# read data

data <- cars


# Split data into training and testing

sample <- sample.int(n=nrow(data), size = floor(0.2*nrow(data)))

train <- data[-sample,]

test <- data[sample,]


#Linear regression

modelLR <- lm(dist    ~ speed, train)

modelLR


# make a prediction for each temperature

predictedLR <- predict(modelLR, test)

#original data

plot(train)

points(test, col = "black", pch=18)

#abline(modelLR)
```

```
points(test$speed, predictedLR, col = "red", pch=16)
rmseLR <- sqrt(mean((test$dist - predictedLR)^2))

#SVM
require (e1071)
modelSVR <- svm(dist ~ speed , train, type= "eps-regression")
modelSVR
predictedSVR <- predict(modelSVR, test)
rmseSVR <- sqrt(mean((test$dist - predictedSVR)^2))
points(test$speed, predictedSVR, col = "blue", pch=8)

#Neural Network
require (nnet)
modelNN <- nnet(dist~ speed, train,size = 20 ,decay =2e-4 ,maxit =
100 ,linout=T,trace=F)
modelNN
predictedNN <- predict(modelNN,test)
rmseNN <- sqrt(mean((test$dist-predictedNN)^2))
points(test$speed, predictedNN, col = "magenta", pch=14)
```

The screenshot for once regression forecasting:



The data frames and values generated from this forecast:

Data	
data	50 obs. of 2 variables
modelLR	List of 12
modelNN	List of 18
modelSVR	List of 31
predictedNN	num [1:10, 1] 50.7 19 11.5 41.6 49.8 ...
test	10 obs. of 2 variables
train	40 obs. of 2 variables
Values	
predictedLR	Named num [1:10] 63.1 30.6 14.3 46.8 71.2 ...
predictedSVR	Named num [1:10] 56.2 27.7 21.2 42.6 68.3 ...
rmseLR	10.3289727263006
rmseNN	10.3452587955458
rmseSVR	8.96947167703491
sample	int [1:10] 42 15 5 27 44 7 36 41 1 6