REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
FACULTY OF MECHANICAL ENGINEERING
INDUSTRIAL ENGINEERING

COMPARISON OF MACHINE LEARNING ALGORITHMS: COVID-19 DATA SET

GAYE NUR ONUR **YAĞMUR ECEM PEHLEVAN**
17061049 **16061047**

GRADUATION THESIS

ADVISOR
PROF. DR. SELİN SONER KARA

İSTANBUL, 2021

PREFACE

In this thesis, different machine learning algorithms are used to apply regression, classification and clustering methods with COVID-19 data set. The data set consists of features of patients such as sex, pregnancy, age. To find the algorithms that give more successful results, the algorithms applied are compared. The applications of algorithms are done by using the R programming language.

We would like to express our heartfelt thanks to our precious advisor Prof. Dr. Selin Soner Kara, who has a big knowledge and gives us a lot of support. Also, we want to thank our families that help us every time we need.

TABLE OF CONTENTS

5. SECTION

LIST OF SYMBOL

Y	dependent variable
X	independent variable
y_i	ith dependent variable
$\hat{y}_i$	ith predicted independent variable
α	alpha
λ	lambda
N	number of observations
p	number of independent variables
q_i	ith observation
p_i	ith observation
X_p	pth independent variable
β_0	intercept
β_p	coefficient of pth independent variable
e	estimation error
P_O	observed agreement
P_e	expected agreement
LL	log-likelihood of the model on the training dataset
s	number of parameters in the model
P(Y)	probability of dependent variable
K	kappa
R^2	R-Squared
k	number of neighbors, clusters

ABBREVERATION LIST

RSS	Residual Sum of Squares
RMSE	Root Mean Squared Error
MAE	Mean Absolute Error
MSE	Mean Squared Error
MSEP	Mean Squared Error of Prediction
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
MLR	Multiple Linear Regression
L1	Lasso Regression
L2	Ridge Regression
Enet	Elasticnet
PC	Principal Component
PCR	Principal Component Regression
PCA	Principal Component Analysis
PLSR	Particial Least Square Regression
EDV	Exploratory Data Visualization
ANN	Artificial Neural Networks
RF	Random Forest
SVR	Support Vector Regression
SVM	Support Vector Machine
CART	Classification and Regression Trees
Bagging	Bootstrap Aggregating
GBM	Gradient Boosting Machines
Cp	Complexity Parameter
RBS	Radial Basis Function
CV	Cross Validation
L	Loss function

LIST OF FIGURES

Page

LIST OF TABLES

ABSTRACT

COMPARISON OF MACHINE LEARNING ALGORITHMS: COVID-19 DATA SET

Gaye Nur Onur Yağmur Ecem Pehlevan

Department of Industrial Engineering

Graduation Thesis

Advisor: Prof. Dr. Selin Soner Kara

The whole world is under the influence of coronavirus (COVID-19). It has various destructive effects on the people and countries. This virus can be fatal for a lot of people. 3.782.490 deaths that resulted from COVID-19 and 174.918.667 infected cases have been reported up to 12 June 2021 [1]. However, there is not any particular knowledge about for which people COVID-19 can be fatal. If it is known, the protection of more risky people can be increased.

Vaccination of all people is not an easy and fast process, so analysis and prioritization of people is a significant step. In this way, more risky people can have priority to be vaccinated, and although the number of cases continues to increase, the number of deaths can decrease.

Machine learning algorithms can be the solution to this problem. There was no data at the beginning of the epidemic. But now, there is a large pool of information. Machine learning algorithms can use this data and find patterns.

In this study, several machine learning algorithms were applied to the data set consisting of COVID-19 cases. It is aimed to find the best algorithms that minimize error and maximize prediction success. 16 different methods were applied for regression, classification, clustering by using R programming language, and then these methods

were compared. R programming language is one of the well-known programming languages used for data science and machine learning projects.

Keywords: Artificial Intelligence, Machine Learning, Data Science, Covid-19, R Programming Language

ÖZET

MAKİNE ÖĞRENMESİ ALGORİTMALARININ KARŞILAŞTIRILMASI: COVID-19 VERİ SETİ

Gaye Nur Onur Yağmur Ecem Pehlevan

Endüstri Mühendisliği Bölümü

Lisans Bitirme Tezi

Adviser: Prof. Dr. Selin Soner Kara

Tüm dünya koronavirüsün (COVID-19) etkisi altındadır. İnsanlar ve ülkeler üzerinde çeşitli yıkıcı etkileri vardır. Bu virüs birçok insan için ölümcül olabilir. 12 Haziran 2021'e kadar COVID-19 kaynaklı 3.782.490 ölüm ve 174.918.667 enfekte vaka bildirilmiştir [1]. Ancak, COVID-19'un hangi insanlar için ölümcül olabileceğine dair belirli bir bilgi yoktur. Bu bilgi sayesinde, daha riskli kişilerin korunması arttırılabilir.

Tüm insanların aşılanması kolay ve hızlı bir süreç değildir, bu nedenle kişilerin analiz edilmesi ve önceliklendirilmesi önemli bir adımdır. Bu sayede daha riskli kişiler aşı olma önceliğine sahip olabilmekte ve vaka sayısı artmaya devam etse de ölüm sayısı düşebilmektedir.

Makine öğrenmesi algoritmaları bu soruna çözüm olabilir. Salgının başlangıcında herhangi bir veri yoktu. Ama şimdi, büyük bir bilgi havuzu var. Makine öğrenimi algoritmaları bu verileri kullanabilir ve kalıpları bulabilir.

Bu çalışmada, COVID-19 vakalarından oluşan veri setine çeşitli makine öğrenmesi algoritmaları uygulanmıştır. Hatayı en aza indiren ve tahmin başarısını en üst düzeye çıkaran en iyi algoritmaların bulunması amaçlanmaktadır. R programlama dili kullanılarak regresyon, sınıflandırma, kümeleme için 16 farklı yöntem uygulanmış ve

daha sonra bu yöntemler karşılaştırılmıştır. R programlama dili, veri bilimi ve makine öğrenimi projeleri için kullanılan iyi bilinen programlama dillerinden biridir.

Keywords: Yapay Zeka, Makine Öğrenmesi, Veri Bilimi, Covid-19, R Programlama Dili

1. SECTION

INTRODUCTION

1.1 Literature Review

Machine learning algorithms are used for several medical studies, like COVID-19. Nowadays, It can be said that the biggest problem in the world is COVID-19. When COVID-19 started to spread all over the world, there was not any specific knowledge about it. Therefore, researches have been increased to find effective treatment methods and understand the disease better. There are different kinds of researches. 419 articles were published and made available between December 1, 2019, and June 27, 2020 according to the study titled “Role of Machine Learning Techniques to Tackle the COVID-19 Crisis: Systematic Review”. Hafsa Bareen Syeda et al selected and investigated the 130 of those articles. They classified articles into 3 themes: Computational Epidemiology, Early Detection and Diagnosis, and Disease Progression. They determined that 71 of 130 articles are about Computational Epidemiology that covers predicting the COVID-19 outbreak, the impact of containment policies, and potential drug discoveries topics. 40 articles are classified under Early Detection and Diagnosis class that about applied AI techniques to detect COVID-19 by using patients’ radiological images or laboratory test results and 19 studies classified under Disease Progression that focused on predicting disease progression, outcomes (ie, recovery and mortality), length of hospital stay, and the number of days spent in the intensive care unit for patients with COVID-19 [2].

In the research titled “Predicting in-Hospital Mortality of Patients with COVID-19 Using Machine Learning Techniques” of Fabiana Tezza et al, machine learning methods such as recursive partition tree, support vector machine, gradient boosting machine, random forest were used to predict in-hospital mortality [3].

In the study of Patricia Melin et al, they analyze the spatial spread of COVID-19 in the world by using self organizing maps. Self organizing maps is an artificial neural network method. And in this way, they tried to learn which countries are behaving and can use similar strategies [4].

Also, machine learning algorithms can be used to predict the number of cases. In the study of Narinder Singh Punn et al, the trend of the COVID-19 is predicted by using support vector regression, polynomial regression, and deep learning regression models such as a standard deep neural network and recurrent neural networks using long short-term memory cells. In the study, methods are compared according to performances of prediction for the case number of the next 10 days [5].

1.2 The Aim of the Thesis

In the thesis, six methods were used for linear regression, and seven methods were used for non-linear regression to predict the value of C-reactive protein. C-reactive protein (CRP) is a protein made by the liver [6]. Six methods were used for classification to predict the patient's case status, and three methods were used to cluster cases. The aim is to find the best methods for the data set by comparing the methods applied.

1.3 Hypothesis

Machine learning algorithms can be used in prediction for data set about COVID-19 cases. The appropriate algorithms for each data set are different, and some of them give better results.

2. SECTION

DATA SCIENCE

The concept of data and what it means must be understood before starting data analysis and machine learning studies. Data is collection of data objects and their attributes. Attributes are properties or categories of the data. Object is defined by collecting attributes.

Large and complex data structures reveal a different concept: Big Data. It will no longer be correct to interpret this concept in a way that only indicates the size of the data amount. Big Data also refers to the size of the processes from obtaining this data to producing results.

The work of obtaining results/inferences with data that is ready for use or not yet is called data science. Certain algorithms and statistical science can be used in this inference process. The amount and variety of data increasing day by day and technology advances has made data mining an inevitable tool. Targeted inferences can be made by extracting usable and meaningful data from these large data collections.

The components of data science are listed as follows: statistics, computer science, domain knowledge and intuition. The problems to be solved in data science are analyzed by turning them into a statistical model. While the model is being built, constants and parameters are determined. It is tried to obtain minimum error so that optimum outputs can be obtained. The process called machine learning uses this algorithm. It provides a rule to be put forward by making use of historical data and tries to reach the desired output by applying this rule. In the second component, computer science, programming

is involved. The collection of data, making it usable, visualizing it, extracting certain patterns from the data, transferring the methods to be used to the codes and making all these available for production are realized through programming. Domain knowledge is essential in data science. Domain knowledge is definitely needed to infer information such as which data will work and how. Otherwise, the available data may be misleading for studies. Finally, intuition. While processing the available data and using certain methods in these processes, it is desired to reach the optimum result. In line with this request, intuition is applied while eliminating question marks such as which algorithms to use [7].

Data science, artificial intelligence, machine learning, and deep learning are interconnected concepts. As seen in Figure 2.1, data science and artificial intelligence have a common area. Machine learning is a subset of artificial intelligence and deep learning is a subset of machine learning.

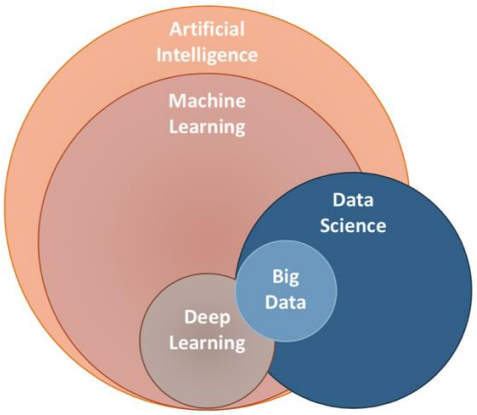


Figure 2.1 Interconnected Concepts [8]

As mentioned above, data science is a systematic process that includes the processing, collection, preprocessing and analysis of data so that a meaningful interpretation can be made by looking at the data. As a result of the analysis, an estimate is made and these results are presented. It includes many disciplines such as data science, statistics, mathematics and information technology.

Artificial intelligence gives machines the ability to reason based on past experience by mimicking human intelligence. It is aimed that the machines understand the data, analyze it, establish relationships, create models and make decisions by making

predictions according to this model. Since machines can store much more data than human memory, it is quite useful in situations where human intelligence is not enough. Insufficient amount of data used by machines can cause wrong decisions with artificial intelligence.

Machine Learning, the developing technology of recent years, enables algorithms to be structures that can learn. It allows the creation and operation of algorithms that make predictions within the scope of data. Machine learning is a subset of artificial intelligence. Machine learning algorithms are applied to make artificial intelligence. Algorithms that make predictions using historical data instead of following the rules enable studies on large quantities of data to be done faster. Statistics are used a lot when trying to give machines the ability to learn. More detailed information about machine learning will be given in the next section.

Deep learning is a subset of machine learning. Artificial neural networks have been developed with inspiration from the human brain and nervous system. An artificial nervous system was created by examining and imitating the biological nervous system.

3. SECTION

DATA PRE-PROCESSING

Before starting the data analysis and studies, the preliminary preparation process must be performed. There may be missing values or outliers in the available data. If the study is carried out without cleaning and correcting these data, it may lead to results such as erroneous results and estimations.

3.1 Missing Values

Missing value is that some values of the observations in the dataset are missing. Missing data may be encountered for various reasons. How to solve the missing data problem is related to the formation of missing data. There are many methods to solve the missing value problem. The first method is that when the number of observations is high, observations and variable with missing values can be deleted. However, if the majority of the available data consists of lost data, deletion may cause erroneous and misleading results. In another method, a suitable value is determined and assigned for the missing value. There are several methods for determining the value to assign. The important point is to be completely impartial when filling in these missing data. The mean or median of the variable with the missing value can be assigned, or the assignment can be made by examining other observations similar to that observation. The third method is predictive assignment methods. In this method, the model is created without including the missing-valued observations, and then the missing-valued observations are estimated using this model.

3.2 Outliers

Observations that have obvious differences from other observations are called outliers. Continuing with these data leads to misleading and erroneous results. Incorrect data entry, a truly contradictory situation, etc. situations are effective in the formation of outliers. The methods to be used to detect outliers can be divided into different groups as univariate, multivariate, parametric (a distribution or non-parametric tests (independent of any distribution) according to the distribution) and statistical or graphical methods according to the number of variables [9].

4. SECTION

DATA VISUALIZATION

Exploratory data visualizations (EDVs) are types of visualizations that you put together when you don't have a clue of what information is in your data. The data mining team is made up of data scientists who specialize in qualitative research [10].

Techniques used in exploratory data analysis:

- Descriptive statistics
- Inferential statistics
- Statistical visualization
- Data visualization
- Hierarchical clustering
- K means
- Principal component analysis

When a data set is displayed in a multidimensional manner, it has six layers, as seen in Figure 4.1. The first dimension shows the distributions (histogram) of the second-dimension variables, the third-dimension densities (red lines), the fourth dimension shows the relations of the variables with each other, the fifth dimension shows the p-values of the correlations, and the sixth dimension shows the scatter plots.

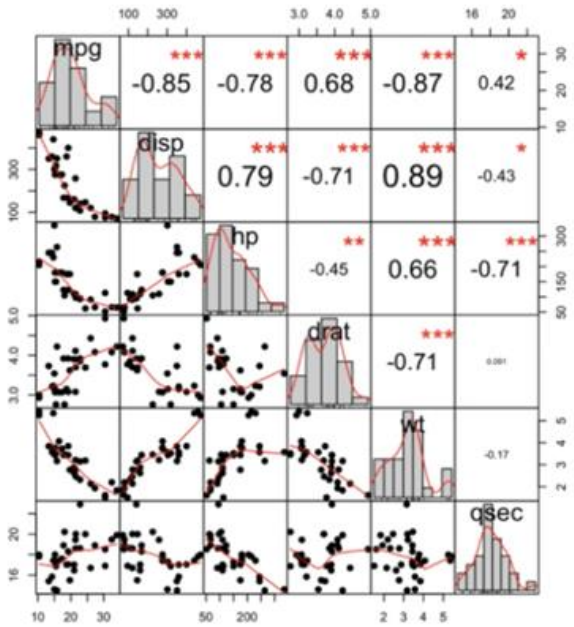


Figure 4.1 Six layers of data [11]

What is ggplot2?

It is a data visualization library by Hadley Wickham adapted from the work "Grammar of Graphics" by Leland Wilkinson et al. The basis of Grammar of Graphics is the connection between data and the aesthetic relations of visuals.

ggplot2 has different layers as can be seen in Figure 4.2 [12]:

- Data
- Aesthetics: variables, color, size, shape ...
- Geoms: point, line, barplot ...
- Facets: Dividing data into subsets
- Stats: Statistical transformation and data summary Coordinate system
- Scales: color, size shape ...
- Theme

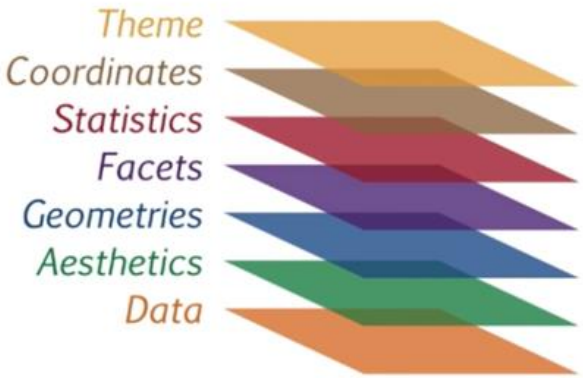


Figure 4.2 Layers of ggplot [12]

4.1 Examining distributions

Bar plot (column chart) : A chart that presents categorical data with rectangular bars of height or length proportional to the values they represent as can be seen in Figure 4.3.

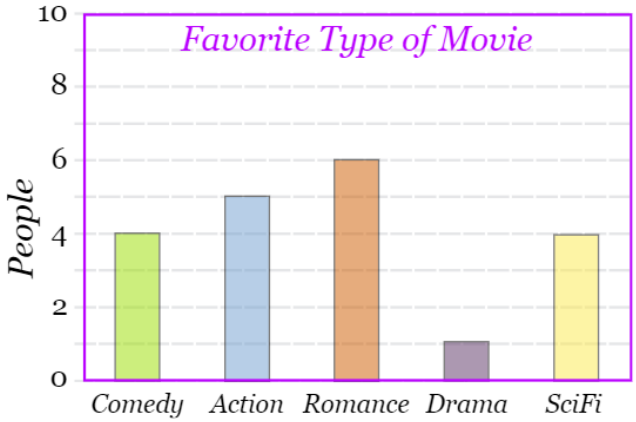


Figure 4.3 Barplot Example [13]

Histogram : A column chart representation of a grouped data distribution. It can be seen with an example like Figure 4.4.

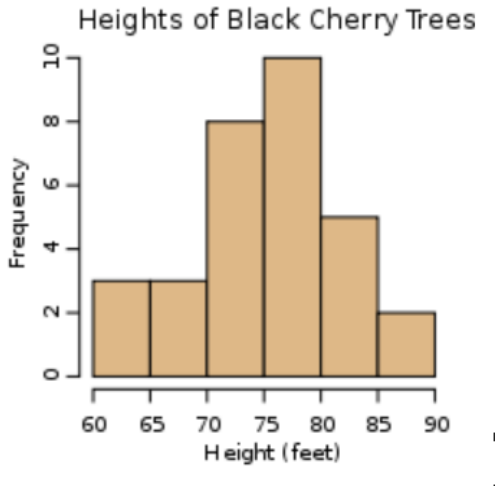


Figure 4.4 Histogram Example [14]

Density : As shown in Figure 4.5, the density graph visualizes the distribution of data across a continuous period or a specific time period [15].

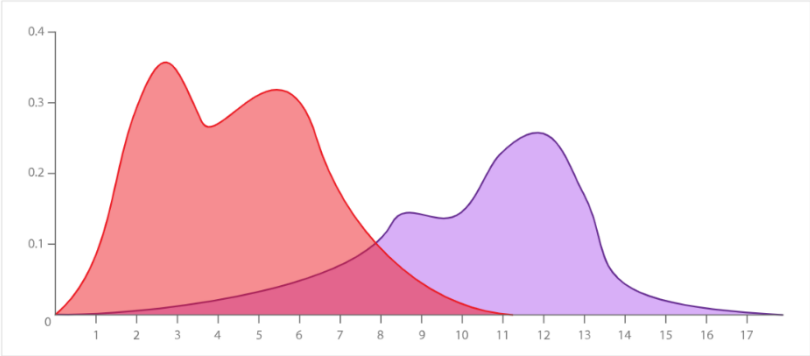


Figure 4.5 Density Graph Example [15]

Ridgeline : The distribution of a numeric variable for numerous groups can be studied using a Ridgelineplot. In this example, graph shows the price distribution of diamonds based on their quality, as shown in Figure 4.6 [16].

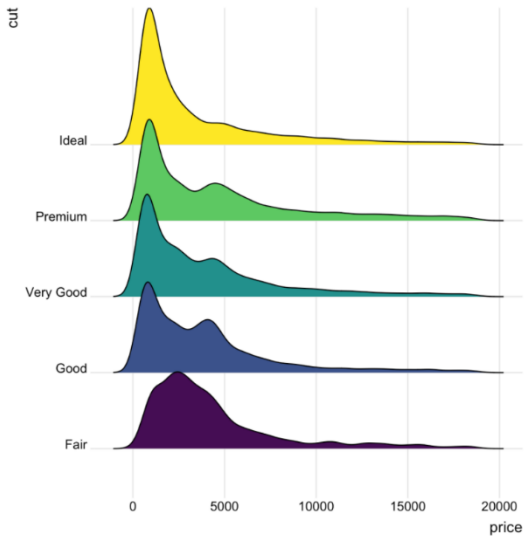


Figure 4.6 Ridgeline Example [16]

Interactive charts : An interactive charts allows the user to perform actions: zooming, hovering a marker to get a tooltip, choosing a variable to display and more [16].

Multi Frequency Graphs : They showing the distribution of categorical and continuous variables according to each other as in the Figure 4.7.

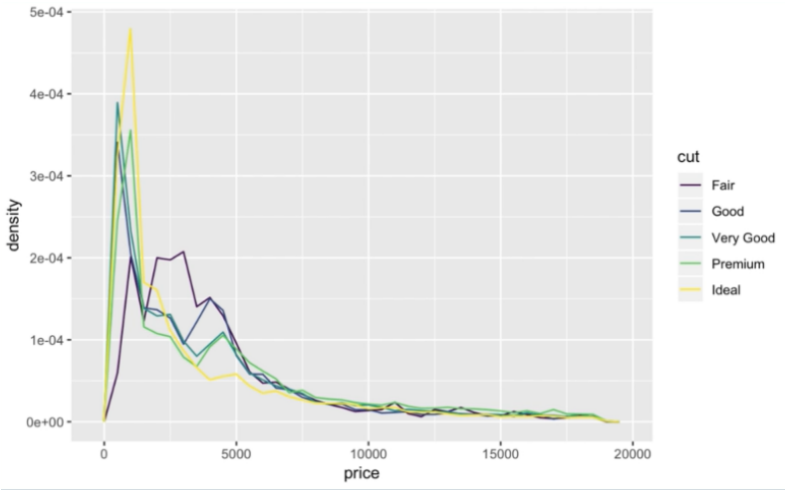


Figure 4.7 Multi Frequency Graph Example [11]

Box Plot : Box Plot graphs can be used to compare distribution of several groups as it can be seen in Figure 4.8 . It should be known that, the data distribution is hidden behind each box.

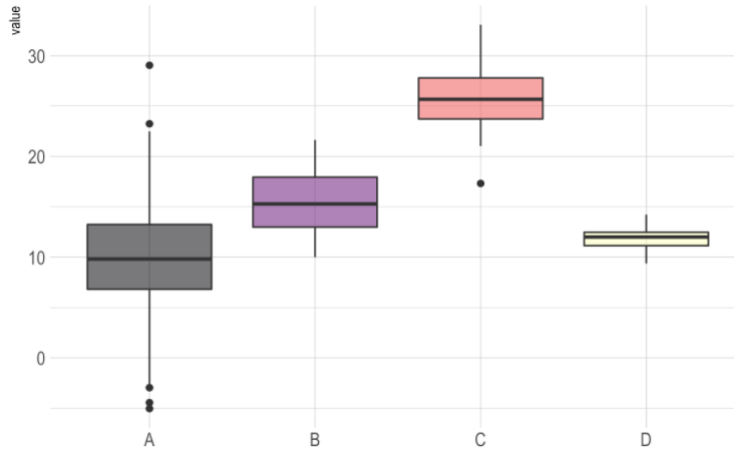


Figure 4.8 Box Plot [16]

Violin : In this type of charts, a plotting numeric data and a combination of the box plot with a kernel density plot can be seen in Figure 4.9. This charts gives the information about median, interquartile range and the lower/upper adjacent values [17].

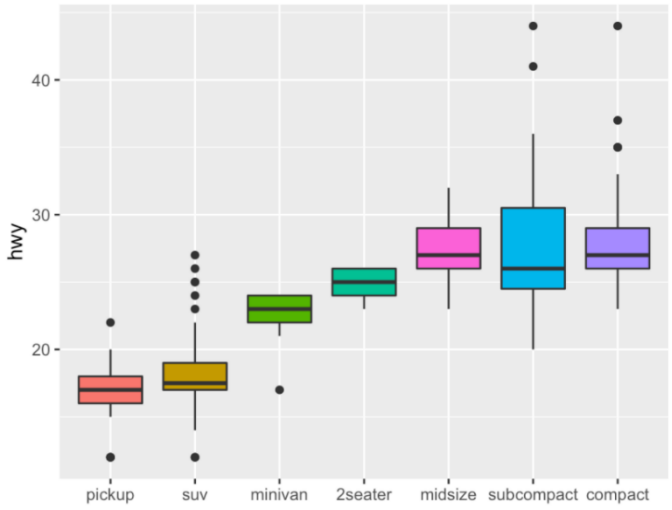


Figure 4.9 Violin Graph Example [16]

4.2 Examining correlations

Scatter Plot : A Scatterplot displays the relationship between 2 numeric variables. Each dot represents an observation. Their position on the X (horizontal) and Y (vertical) axis represents the values of the 2 variables. An example can be seen in Figure 4.10 [16].

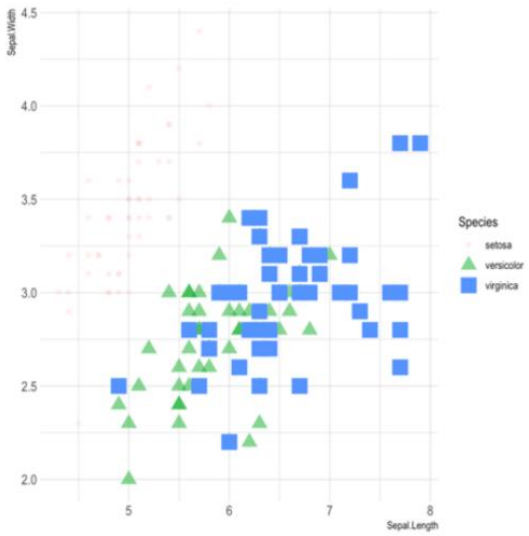


Figure 4.10 Scatter Plot Example [16]

Heatmap : The heat map shows the values for the respective main variable (as it can be seen in Figure 4.11) in the two axis variables as a grid of colored squares.

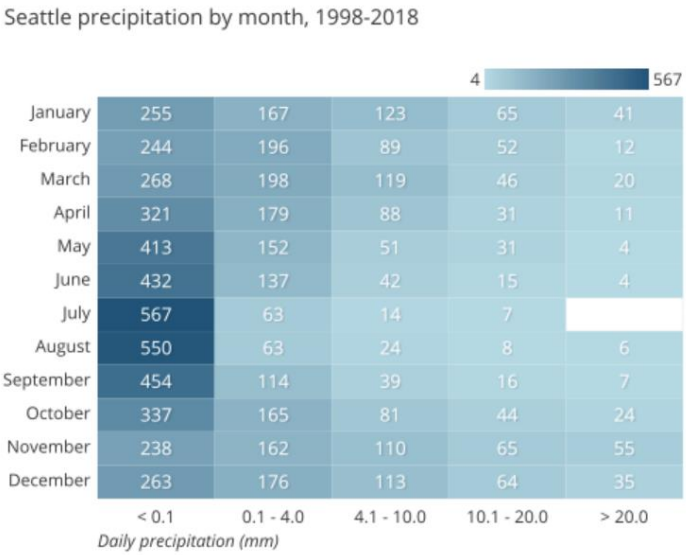


Figure 4.11 Heatmap Example [18]

Correlation Matrices : It shows the relationship between more than two variables. An example can be seen in Figure 4.12 for a dataset about cars.

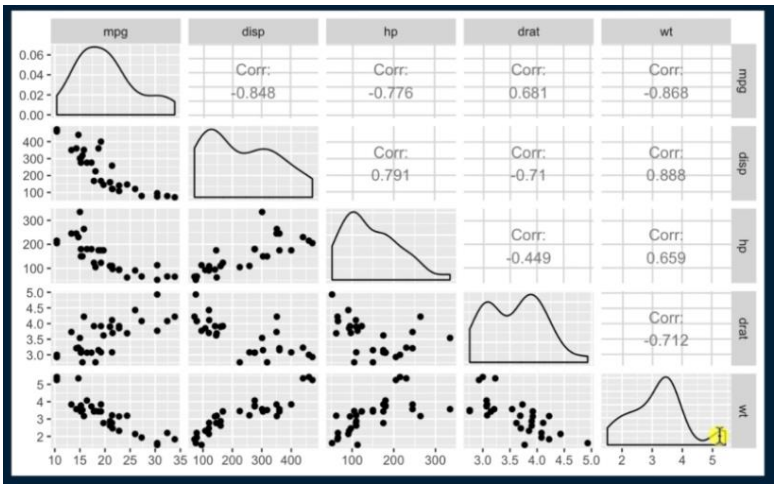


Figure 4.12 Correlation Matrices Examples [11]

5. SECTION

MACHINE LEARNING

5.1 What is Machine Learning?

Machine learning is creating artificial intelligence algorithms that are trained with data and can make inferences (predictions) as a result. It makes use of statistics and data science while doing this estimation and learning process. Different algorithms and methods can be seen in Figure 5.1.

Machine learning systems use a statistical learning algorithm that automatically learns and improves without human help. Machine learning is one of the most important subfields of artificial intelligence and makes a great and continuous contribution to the development of today's technologies. It has benefited greatly from its application in many areas and has facilitated the processes it is in [19].

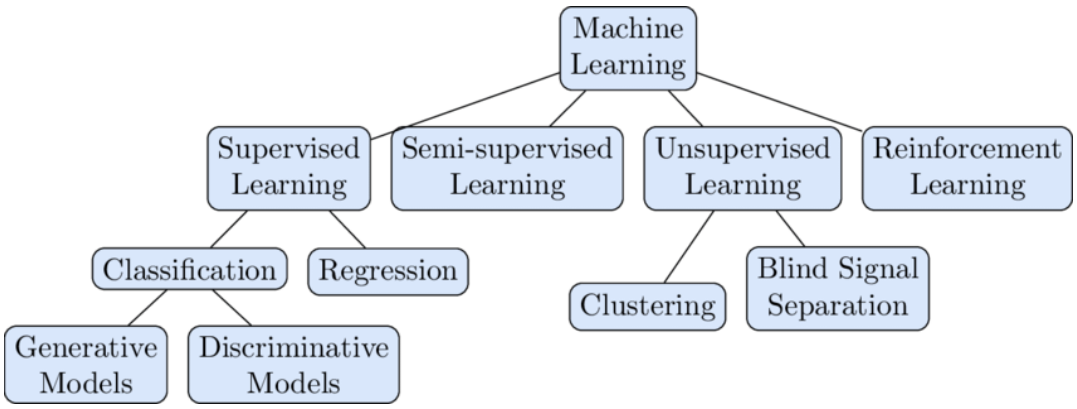


Figure 5.1 Machine Learning Categories [20]

5.1.1 Supervised Learning Algorithms / Methods

In the supervised learning method, inputs and outputs are found in the data using, labeled data is used. A model is created by analyzing the inputs and outputs. Using this model, the outputs of new observations are estimated based on their inputs. Regression and classification problems are supervised learning applications.

Classification: A classification problem is when the output variable is a category, such as “hot” or “cold” or “dangerous” and “not dangerous”. [21]

Regression: A regression problem is when the output variable is a real value, such as “TL” or “height” [22].

5.1.2 Unsupervised learning algorithms / Methods

In the unsupervised learning method, there are inputs but no outputs, unlabeled data is used. In this method, there is no correct answer as in supervised learning, as a structure is tried to be created using the inputs. Clustering and association problems are applications of unsupervised learning.

5.1.3 Semi-supervised learning algorithms / Methods

In the semi-supervised learning method, the data contains both labeled and unlabeled data, so it is similar to both supervised learning and unsupervised learning.

5.1.4 Reinforcement learning algorithms / Methods

The training of machine learning models to make a series of judgments is known as reinforcement learning. In an uncertain, potentially complex environment, the agent learns to achieve a goal. An artificial intelligence meets a game-like circumstance in reinforcement learning. To find a solution to the problem, the computer uses trial and error. Artificial intelligence is given either rewards or penalties for the acts it takes in order to get it to accomplish what the programmer desires [23].

5.2 Model Validation

Model validation methods are used to determine how well the model is working. Overfitting (High Variance) and Underfitting (High Bias) are tried to be prevented.

The model memorizes the data set in the event of overfitting. While the model works very well in the data set, it fails to predict a new observation. Underfitting occurs when the developed model fails to predict the data set's observations. As shown in Figure 5.2, there is a significant disparity between the actual and predicted values of the observations in the event of underfitting in regression applications. The observations seemed to have been successfully predicted in the case of overfitting. For new observations, however, this model will fail. Figure 5.3 shows how overfitting and underfitting occur in classification applications. In the case of underfitting, classification is ineffective. Although it appears to be effective at overfitting, it is deceiving.

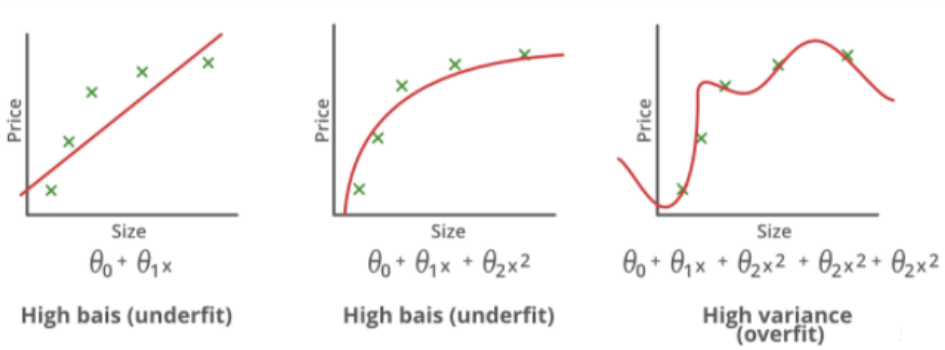


Figure 5.2 Underfitting and Overfitting for Regression [24]

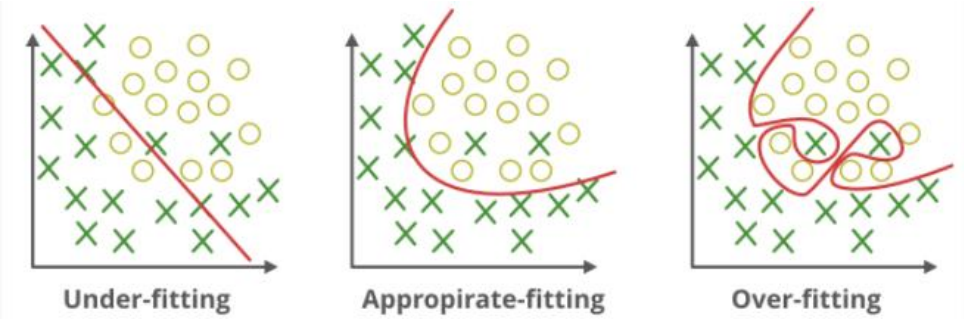


Figure 5.3 Underfitting and Overfitting for Classification [24]

The validation methods used to prevent these situations are described below.

5.2.1 Holdout Method

The data set is split into two sections: training and test. The model that was developed with the training set is subsequently put to the test with the test set.

5.2.2 K-Fold Cross Validation

It is the most widely used machine learning method. A train set and a test set are created from the dataset. The train set is then split into k subgroups. Each of these k groups is eliminated one at a time, and the model is built using the k-1 group and tested with the removed group. Then the average of errors calculated k times is calculated. The test error is calculated using the validated model and the test set.

5.2.3 Leave-One-Out Cross Validation

Leave-One-Out Cross Validation works in a similar way to K-Fold Cross Validation. In K-Fold Cross Validation, groups are excluded sequentially, while in this method, each observation is left out in order and the model is created and tested with the observation that is excluded.

5.2.4 Bootstrap Method

In the Bootstrap Method, train set and test set created by selecting random observations from the data set are used.

5.3 Model Evaluation

These are the ways for assessing the model's success and determining its performance. For regression problems with a continuous variable output and classification problems with a categorical variable output, different methods are utilized.

5.3.1 Model Evaluation Methods for Regression

5.3.1.1 Mean Squared Error

It is obtained by dividing the sum of the squares of the difference between the estimated value obtained by using the model created and the actual value by the number of observations.

5.3.1.2 Root Mean Squared Error

The sum of the squares of the difference between the estimated value obtained by using the model created and the actual value is divided by the number of observations. It is obtained by taking the square root of the result. Root Mean Squared Error equals the square root of Mean Squared Error.

5.3.1.3 Mean Absolute Error

It is obtained by dividing the sum of the absolute values of the difference between the estimated value obtained by using the model created and the actual value by the number of observations.

5.3.1.4 Mean Absolute Percentage Error

The difference between the estimated value obtained by using the model created and the actual value is divided by the real values and then summed. The obtained value is divided by the number of observations and multiplied by 100.

5.3.1.5 R-Squared

On a handy 0–100 percent scale, R-squared measures the strength of the link between the model and the dependent variable. The equation that yields the smallest difference between all of the observed values and their fitted values is identified using linear regression. To be more specific, linear regression determines the dataset's least sum of squared residuals. It is defined as (5.1).

$$R^2 = Variance\ explained\ by\ the\ model\ /\ Total\ variance \quad (5.1)$$

In most cases, the higher the R^2, the better the regression model matches your data [25].

5.3.2 Model Evaluation Methods for Classification

5.3.2.1 Confusion Matrix

Confusion Matrix is a method used for classification algorithms. If the predicted class is 1 and the real class is 1, it is True Positive (TP). If the predicted class is 1 but the real class is 0, it is False Positive (FP). If the predicted class is 0 and the real class is 0, it is True Negative (TN). If the predicted class is 0 but the actual class is 1, it is False Negative (FN). This matrix is visualized in Figure 5.4. Using Confusion Matrix, accuracy, precision, recall, true positive rate, false positive rate can be calculated. The synonym of sensitivity is recall. The formulas are as in (5.2) for Accuracy, (5.3) for Precission, (5.4) for Recall, (5.5) for F1 Score and (5.6) for Specifity.

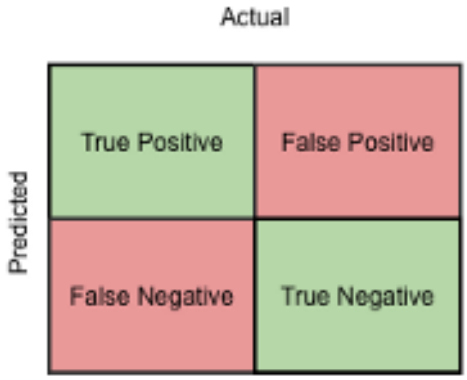


Figure 5.4 Confusion Matrix

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FN + FP)} \tag{5.2}$$

$$Precission = \frac{TP}{(TP + FP)} \tag{5.3}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{5.4}$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{5.5}$$

$$Specificity = \frac{TN}{(TN + FP)} \tag{5.6}$$

5.3.2.2 Receiver Operating Characteristics (ROC) Curve

The receiver operating characteristics curve (ROC) is the plot of true positive rate versus false positive rate. The ROC curve is interpreted by evaluating the area under the curve. The area under the ROC curve is called AUC. A large area means that the curve is near the upper left. This situation is positive in terms of evaluation. When the curve seen in Figure 5.5 approaches the area indicated by points, it means that the model has failed.

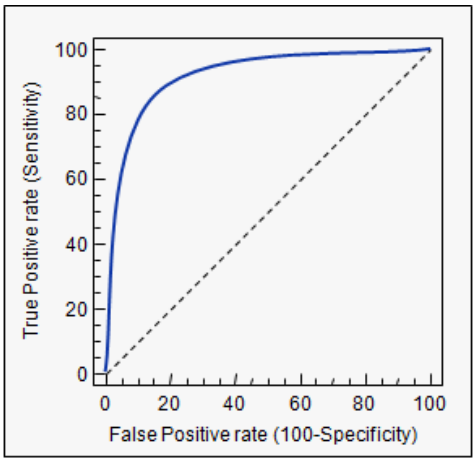


Figure 5.5 Receiver Operating Curve [26]

5.3.2.3 Kappa

Cohen's Kappa statistic is a very valuable metric that is underutilized. Measures like accuracy and precision/recall do not provide a complete view of our classifier's performance in multi-class classification situations. In rare circumstances, we may encounter an issue with unequally distributed classes. It's a great tool for dealing with both multi-class and imbalanced class problems. It defined as in (5.7).

$$K = P_O - P_e \,/\, 1 - P_e \tag{5.7}$$

The P_O is the observed agreement, and P_e is the expected agreement. It essentially informs you how much better your classifier performs compared to a classifier that guesses at random based on the frequency of each class. Cohen's kappa is always less than or equal to 1. The classifier is useless if it has a value of 0 or less. Its values cannot be interpreted in a uniform fashion. Landis and Koch (1977) propose a method for describing values [27].

5.3.2.4 AIC and BIC

AIC and BIC are both penalized probability criteria, they are used for regression to select the best predictor subsets. AIC defined as in (5.8).

$$AIC = -2/N * LL + 2 * s/N \tag{5.8}$$

The *N* is the number of examples in the training dataset, *LL* is the log-likelihood of the model on the training dataset, and *s* is the number of parameters in the model. Model that give smallest AIC over the set of models is can be selected [28].

BIC formula is defined as in (5.9).

$$BIC = -2 * LL + log(N) * s \tag{5.9}$$

The *log()* has the base-e called the natural logarithm, *LL* is the log-likelihood of the model, *N* is the number of examples in the training dataset, and *s* is the number of parameters in the model. Model that give smallest BIC over the set of models is can be selected. [28].

5.4 Linear Regression

5.4.1 Simple Linear Regression

Simple linear regression is applied for two variables. One of these variables is the dependent variable, another one is the independent variable. By using these variables, the simple linear regression model is set up. While setting up the model, it is aimed to

create a linear line that minimizes the prediction error. This linear line provides to understand the relationship between two variables. For the new observations, estimation of the dependent variable is done by using the independent variable and model. (5.10) represents the simple linear regression model. "Y" parameter indicates the dependent variable, "X" parameter indicates the independent variable, "β_0" is the intercept of y-axis, "β_1" is the coefficient of the independent variable, and "e" indicates estimation error in (5.10).

$$Y = \beta_0 + \beta_1 X + e \tag{5.10}$$

A simple linear regression application is visualized in Figure 5.6.

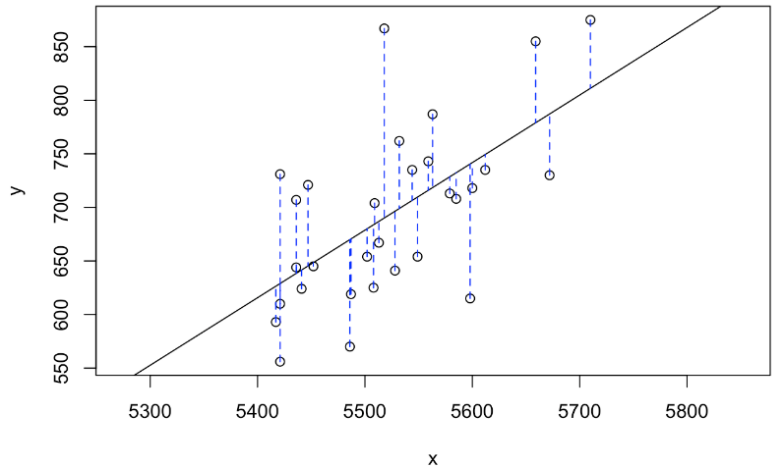


Figure 5.6 Simple Linear Regression [29]

5.4.2 Multiple Linear Regression (MLR)

Multiple linear regression is a similar method to simple linear regression. The difference of multiple linear regression is that it is applied with more than two variables. One of these variables is the dependent variable, other ones are independent variables. The aim of the method is to create a linear line that minimizes the residual sum of squares. So, it is tried to find the best coefficients that minimize the residual sum of squares. Thanks to this linear line, the dependent variable can be predicted by using independent variables. The formula of the residual sum of squares is given in (5.11). Formulation of multiple linear regression is given in (5.12). In terms of the confidence of the model,

there should be no multicollinearity problem between the variables. The correlation coefficient of the two variables should not be higher than 0.90.

$$\sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{5.11}$$

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 ... + \beta_N X_N + e \tag{5.12}$$

5.4.3 Ridge Regression (L2)

Ridge regression is a similar method to multiple linear regression. The aim of the model is minimizing residual sum of squared and finding the appropriate coefficients, like in MLR. However, it has a difference from MLR. "lambda" parameter (λ) is used to regularize the coefficients. Regularization is done by multiplying squares of the coefficients with λ. So, regularization for coefficients is done. This method is effective when multicollinearity problem exists. In the result of ridge regression, coefficients of the model does not be equal 0, but it approaches to 0. Because of coefficients are never equal 0 in ridge regression, elimination of variables can not be done. (5.13) is the formula of the residual sum of square in ridge regression. "N" represents the number of observations, "p" represent the number of independet variables. Model tuning should be done to find optimal λ parameter. Parameter α is always 0 for ridge regression. This parameter is going to explain with more detail in elastic regression section.

$$RSS_{L2} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{J=1}^{p} {\beta_j}^2 \tag{5.13}$$

5.4.4 Lasso Regression (L1)

Lasso regression is a regularization method like ridge regression. It is aimed to find optimal coefficients which minimizes residual sum of squared. Regularization is done by multiplying absolute of coefficients with λ. In lasso regression, coefficients can be regularized as equal to 0 unlike ridge regression. In ridge regression, it never happens. So, elimination of some variables can be done. (5.14) is the formula of the residual sum

of square in lasso regression. Model tuning should be done to find optimal λ parameter too, like in ridge regression. Parameter α is always 1 for lasso regression.

$$RSS_{L1} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{J=1}^{p} |\beta_j| \tag{5.14}$$

5.4.5 Elastic Regression

Elastic regression is a combination of ridge and lasso regression. Regularization is done with using penalty of both of them, so both methods are used. “alpha” parameter (α) is used to represent mixing rate of these methods. This parameter is used for ridge and lasso regressions too. But for those method, the value of α is predetermined. So, model tuning for α is done for just elastic regression. Also, model tuning is done for λ parameter. Hereby, two parameters were tuned in elastic regression. The formula of the loss function for elastic regression is given in (5.15).

$$L_{Enet} = \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{2N} + \lambda \left(\frac{1-\alpha}{2} \sum_{J=1}^{p} {\beta_j}^2 + \alpha \sum_{J=1}^{p} |\beta_j| \right) \tag{5.15}$$

5.4.6 Principal Component Regression (PCR)

In the principal component regression method, the model is set up by decreasing the number of variables instead of using all variables. After the number of variables is decreased, the multiple linear regression method is applied with principal components. While applying PCR, correlations among variables are considered. For example, a data set has 6 variables, so it has 6 dimensions. PCR reduces the dimension of the data to less than 6. In the PCR, PCR is useful when there are highly correlated variables. It avoids the multicollinearity problem. Also, it is useful when there are too many variables in the data set. Figure 5.7 shows an example of PCR. In the original data, there are three variables. After applying principal component analysis dimension of data reduced to 2 as PC1 and PC2.

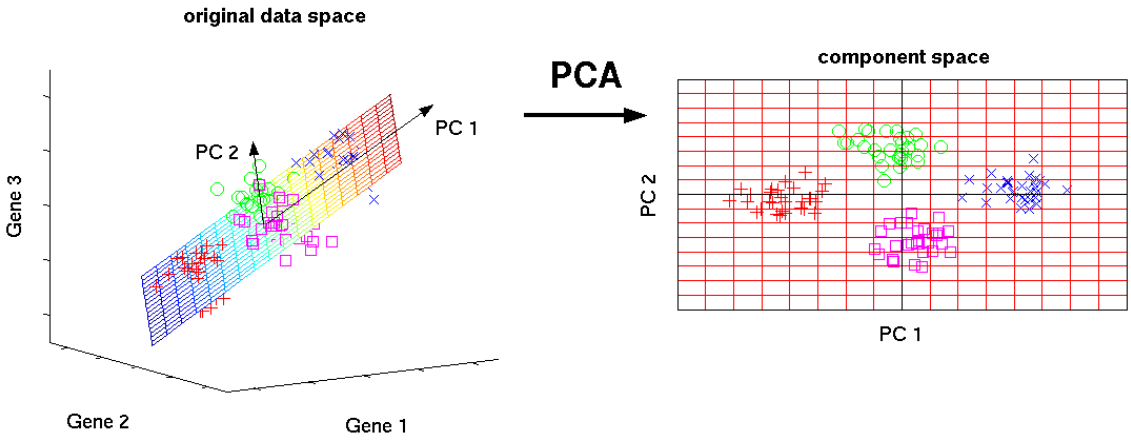


Figure 5.7 Dimension reduction [30]

The difference between MLR and PCR is visualized in Figure 5.8. It can be seen that MLR is applied to the data set directly. But in the PCR, after principal components are determined MLR is applied.

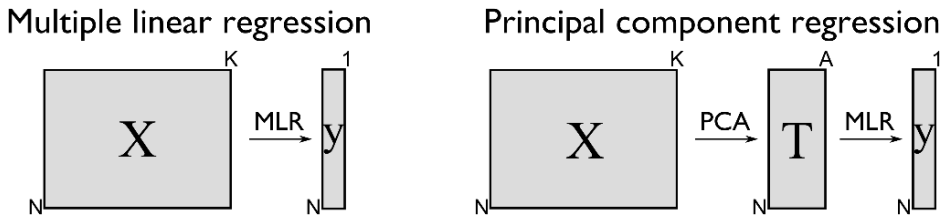


Figure 5.8 Multiple Linear Regression and Principal Component Regression [31]

5.4.7 Particial Least Squares Regression (PLSR)

PLSR is a similar method to PCR. It deals with the multicollinearity problem like PCR when among variables there is high correlation. PLSR is a useful method when the observation number is not very high. In PCR, correlations among variables are considered, but in PLSR variables are considered according to correlation with the independent variable. In this way, the dimension of the data set is reduced. Figure 5.9 visualizes the logic of PLSR.

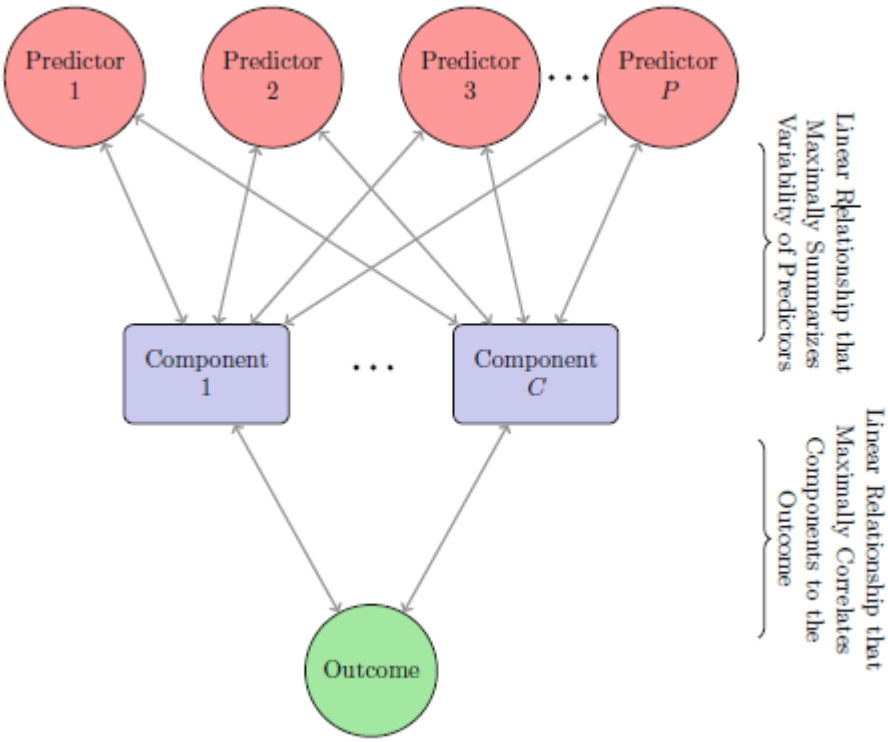


Figure 5.9 Structure of PLS Model [32]

5.5 Non-Linear Regression

5.5.1 K-Nearest Neighbor Algorithm

K-Nearest Neighbor is a method that can be used for non-linear regression and classification. The logic of the algorithm is similar for both of them, it will be examined for non-linear regression in this section. The first step is determining the number of neighbors (k) which will be used, in this method. When a new observation occurred which independent variables are known but the dependent variable is unknown, it is looked at old observations and the new observation. The k nearest observations to the new observation are determined. Figure 5.10 shows the k nearest observation of the new observation when k is accepted as 3. The observation shown in red is the new observation, other ones are old.

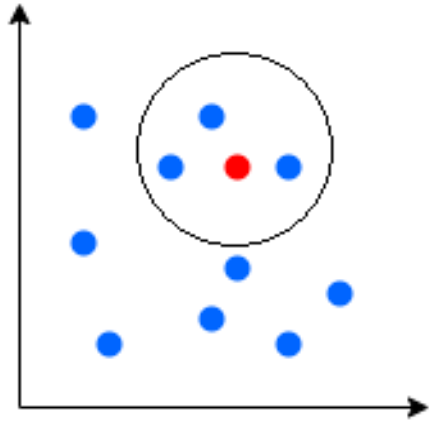


Figure 5.10 KNN Algorithm

The mean of these nearest observations' dependent variables is accepted as the new observation's dependent variable. There are different methods to calculate distances among observations, such as: Euclidean Distance, Manhattan Distance, Minkowski Distance. Formulas of distance calculating methods are given in (5.15), (5.16), (5.17).

$$Euclidean\ Distance = \left(\sum_{i=1}^{n}(p_i - q_i)^2\right)^{1/2} \tag{5.15}$$

$$Manhattan\ Distance = \sum_{i=1}^{n}|p_i - q_i| \tag{5.16}$$

$$Minkowski\ Distance = \left(\sum_{i=1}^{n}|p_i - q_i|^p\right)^{1/p} \tag{5.17}$$

5.5.2 Support Vector Regression (SVR)

Support vector regression is a support vector machine method for regression application. Support vector machine can be used for regression and classification. In this part, we will examine for regression. In SVR, a line or a curve is determined which is called hyperlane. For linear regression it is a line, for non-linear regression it is a curve. Hyperlane has decision boundaries. SVR application is visualized for linear regression on the left side of Figure 5.11, for non-linear regression on the right side of Figure 5.11.

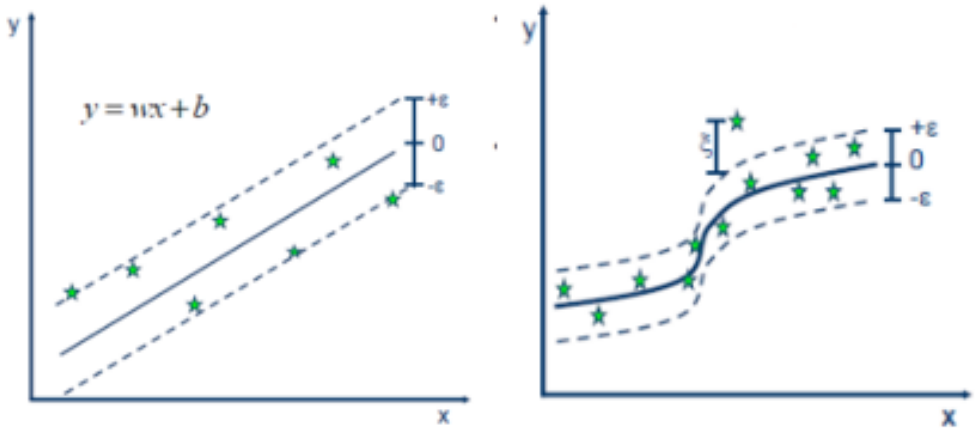


Figure 5.11 SVR application for linear regression and non-linear regression [33]

It is aimed to contain the maximum amount of the data inside of the decision boundaries with minimum error. Epsilon represents the acceptable error value. Decision boundary leaves outliers on the outside because they increase the error. While maximizing margin, it is not possible to minimize the prediction error. Therefore, cost parameter (C) is used. If C is selected large value, margin tends to be small, otherwise, it tends to be large.

5.5.3 Classification and Regression Trees (CART)

CART is one of the oldest decision tree methods. Decision trees can be used for both regression and classification. In this section, CART will be explained for regression. CART is applied by using all observations and all variables. The importance levels of variables are significant for decision trees. At the beginning of the tree there is only one node and the variable in that node are split two. Then splitting is continued, until the end of the decision tree which there are leaf nodes. There are independent variables in the nodes until the leaf nodes, but there are dependent variables in the leaf nodes. Prediction is done according to the decision tree.

If the decision tree is too branched, the tree needs to prune because too many branches can cause some problems such as overfitting. Pruning means stopping branching at a specific level. An example of a decision tree of a dataset related to pollution consist of numerical variables is given in Figure 5.12.

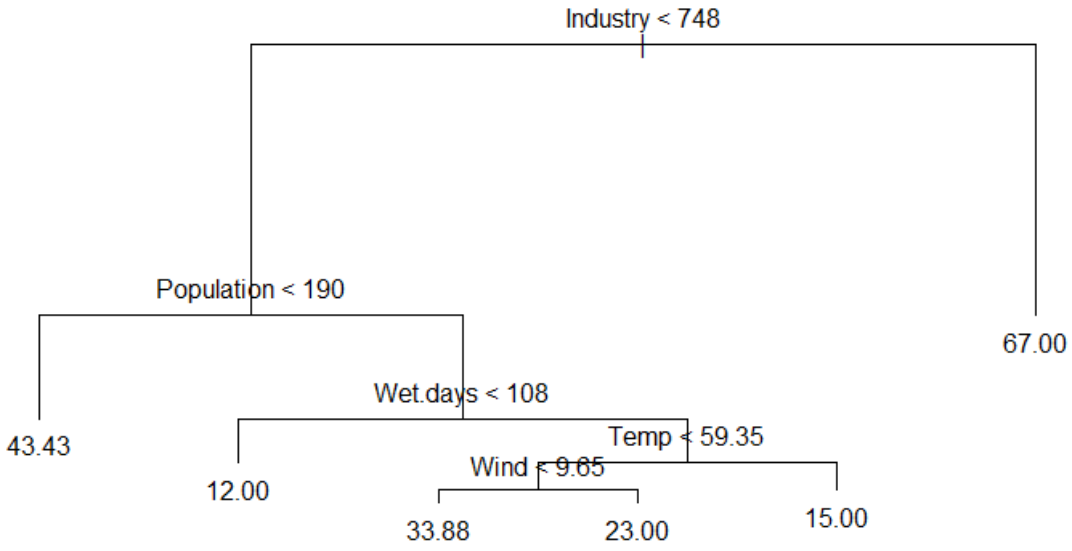


Figure 5.12 Decision Tree of Pollution Dataset [34]

The cost complexity criterion is used to determine where to prune. We find the optimal subtree by using a cost complexity parameter that penalizes our residual sum of square function for the number of terminal nodes of the tree. For a given value of cost complexity parameter, we find the smallest pruned tree that has the lowest penalized error. [35]

5.5.4 Bagging Trees Regression

Bagging trees regression is a different method of decision trees. This method is more advanced than CART. Bagged trees can be used for both regression and classification applications. It will be examined for regression in this section. In the bagging trees method, multiple decision trees are created by using different random observation samples, but all variables are included. This process is called bootstrapping. The bootstrap method is a resampling technique used to estimate statistics on a population by sampling a dataset with replacement [21]. Every decision tree is created like in the CART. When predicting a new observation is wanted, the prediction results are determined for each decision tree created. Then the average of these results is calculated. In this way, the final prediction is done.

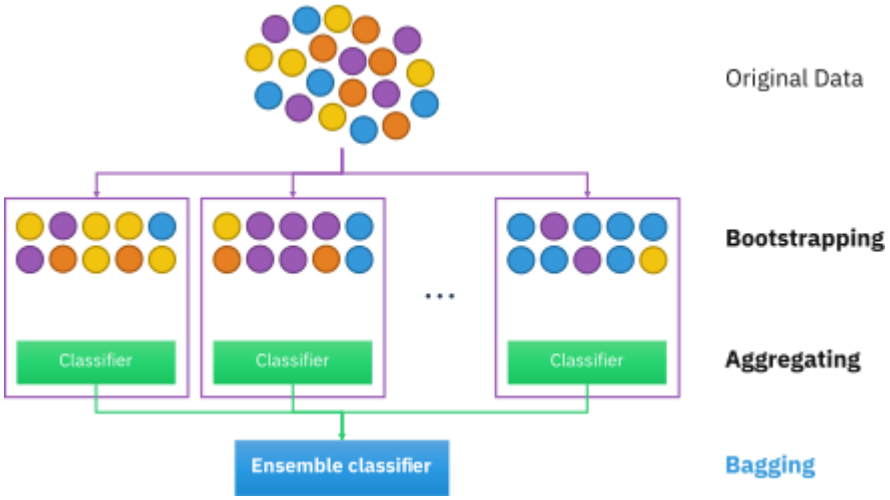


Figure 5.13 Bagging Trees [36]

Figure 5.13 shows the application of bagging trees. Each different data sets created with different observations is called bootstrap.

5.5.5 Random Forest

Random forest is one of the decision tree methods. The random forest can be applied for regression and classification. In the random forest method, different trees are created like in bagging trees. But it is a difference between random forest and bagging trees that while resampling, not just different observations are used, also different random variables are used in random forest. The weight value is given to the trees by evaluating the error rate of the trees that are formed as a result of the application of the method. When predicting the result, the weight of the trees is taken into account.

5.5.6 Gradient Boosting Machines (GBM)

Gradient boosting machines is used for regression and classification. This method is generated is based on the idea of Breiman that boosting can be seen as optimization [11]. In the first step of GBM, a model is set up and residuals of the target variable are calculated. Then a new model is set up by using residuals and errors of residuals are calculated. These errors calculated are added to the predicted target variable and a new model is set up with the calculated variable. These iterations are repeated until achieving the stopping criteria. With the result extracted from each model, the next

model is developed and the prediction is made gradually and at the end, the final model is reached. The application logic of this method is visualized in Figure 5.14.

Figure 5.14 Gradient Boosting Machines [37]

5.5.7 Artificial Neural Networks for Non-Linear Regression

Artificial neural networks are quite similar to natural nerve systems in the human brain in terms of idea. It may be thought of as a series of parallel adaptive neuron networks. Neurons are basic nonlinear computer components that are cognitive. The only purpose of neural networks is to undertake both analysis and the creation of such significant parallel computing systems. Input, hidden, and output neurons are the three kinds of neurons. As a stimulant to the network, input neurons receive input from external sources. The network's output signals are generated by the output of neurons. Hidden neurons calculate the intermediate functions, and these neurons are not visible from the outside. Format of Figure 5.15 contains a schematic representation of a layered neural network [38].

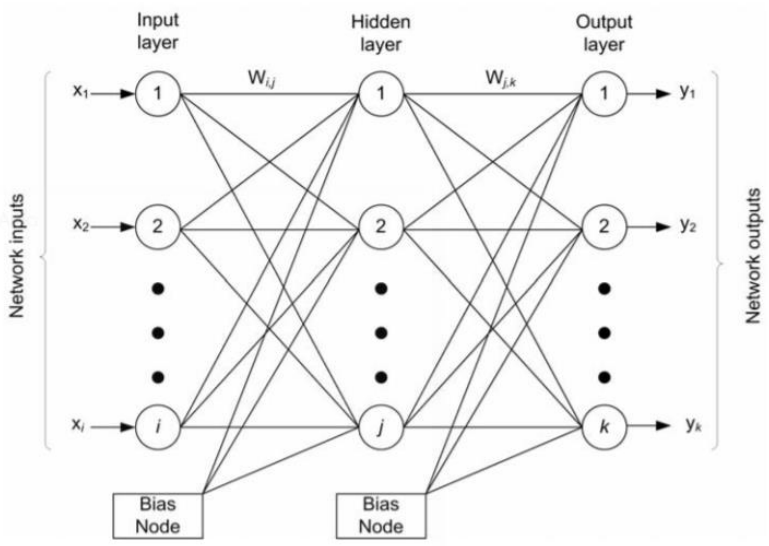


Figure 5.15 Layered Neural Network [39]

5.6 Classification

Classification methods provide to predict dependent variable when the dependent variable is categorical.

5.6.1 Logistic Regression/ Multinomial Logistic Regression

Despite this method is a regression, it is used for classification. Logistic regression deals with predicting binary dependent variables such as 1/0 and Yes/No. Multinomial Logistic Regression deals with predicting dependent variable which has more than two classes such as 2/1/0 and Good/Medium/Bad. Like in MLR, it is aimed to build a model that prediction performance is the highest.

At the end of logistic regression, the final class is not given directly, probability is given. For the two-class prediction, if the probability is greater than 0.5, it is assigned to class 1. Otherwise, it is assigned to class 0.

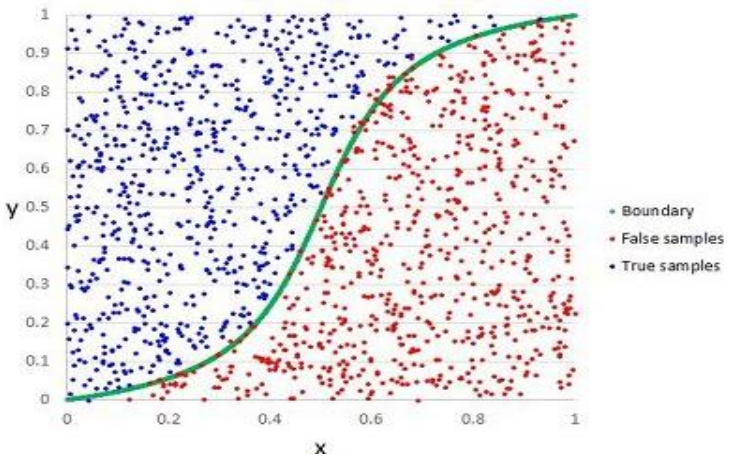


Figure 5.16 Logistic Regression [40]

Figure 5.16 visualizes the logistic regression application for dependent variable prediction with two classes.

The logistic regression model called as Sigmoid Function is given for a data set consists of one independent variable and one dependent variable in (5.18). Logistic regression model for a data set consists of more than two variables is given in (5.19). As the result of these equations, the probability of the dependent variable is calculated.

$$P(Y) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}} = \frac{1}{1+e^{-(\beta_0+\beta_1 X)}} \tag{5.18}$$

$$P(Y) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p}} \tag{5.19}$$

In multinomial logistic regression, more than one model is set up. The number of models that are set up is 1 less than the number of classes of the dependent variables. For each class, the probability of belonging to that class is calculated with models set up. The prediction is done according to probability values. The class that has the highest probability is the class that the observation belongs.

5.6.2 K-Nearest Neighbor Algorithm

K-nearest neighbor algorithm was explained for regression, it will be explained for classification now. Like in regression, the first step is defining the number of neighbors (k). The k nearest neighbors of the observations are determined. Distance among

observations can be calculated with different methods such as Euclidean Distance, Manhattan Distance, Minkowski Distance, like in regression. Whichever class has the greatest amount among the dependent variable of the neighbors is estimated as the class of the new observation.

5.6.3 Classification and Regression Trees (CART)

CART application is pretty similar for classification and regression. The difference between them is that there are categorical variables in the leaf nodes for classification, while there are continuous variables for regression in CART. An example of CART for sex prediction is visualized in Figure 5.17. According to the tree, if the height of someone is higher than 180 cm, sex is predicted as male. If the height is less than 180 cm and the weight is higher than 80 kg, sex is predicted as male. Otherwise, it is predicted as female.

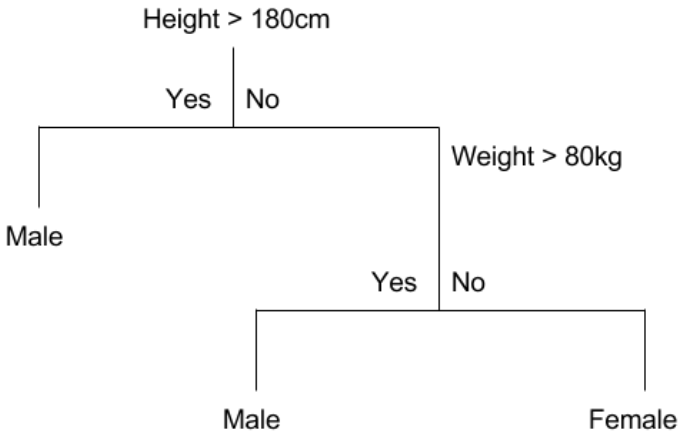


Figure 5.17 Classification And Regression Trees for Classification [41]

5.6.4 Random Forest

In random forest application for classification, different trees are generated by using random and different observations and variables, like in regression. The result of the regression application is a numerical value, but it is a categorical value for classification application. Trees are evaluated according to error rate and weighted. Then, the final result is determined by considering the weights of trees.

5.6.5 Artificial Neural Networks

Artificial Neural Networks, which are used in non-linear regression methods, are also a method used for classification. This method, which is used in the estimation of a numerical dependent variable in regression analysis, is used in the estimation of the classes in which the observations are included in this title.

5.6.6 Support Machine Vectors

Support Vector Networks, often known as SVMs (Support Vector Machines), are supervised learning algorithms that evaluate labeled training data. SVM may utilize either a linear or non-linear model to classify characteristics in a training set into categories. The data set's kernel function determines the classifier's linearity such as linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid. [42].

SVM is an excellent technique in high-dimensional domains, and it's especially useful in document classification and sentiment analysis, where the dimensionality might be exceedingly high. Because only a subset of the training points is used in the actual decision-making process of allocating new members, only these points must be remembered (and computed) when making judgments. The division of classes is frequently non-linear. The ability to apply fresh kernels gives the decision boundaries a lot more flexibility, which leads to better categorization results [43].

5.7 Clustering

Unsupervised learning is the approach used when one has little or no idea of what the desired output from the data looks like. A model can be created from data where the effect of the variables is unknown. In unsupervised learning, there is only data and no information is given about them. Inferences have been made from these data. Since no information is given about the data from the very beginning, it cannot be said that the results obtained are absolutely correct. Various models and structures can be created by clustering the data based on the relationships between the variables. There is no feedback based on prediction results in unsupervised learning [23].

5.7.1 K-Means

The aim is to divide the data into clusters according to their similarity.

K-Means implementation steps for unsupervised learning can be shown as follows:

1. The number of clusters is determined.
2. Random center is selected.
3. For each observation, distances to k centers are calculated and each observation is assigned to the nearest k centers.
4. Each observation is assigned to the closest center, the cluster.
5. Center calculations are made again for the clusters formed after the assignment process.
6. This process is repeated for the specified number of iterations and the clustering structure of the observations in the case where the intra-cluster error-squares sum is minimum is selected as the final clustering.

5.7.2 How is the optimum number of clusters determined?

By comparing the overall error sum of squares generated for each of the clustering studies generated with different numbers of k values, the optimum k is established. There are various approaches for determining this. The following are a few of them:

Elbow

1. Determine the K number (centers to be tried)
2. For each number of trials determined, the total of errors within the cluster is taken.
3. Errors are shown on the graph.
4. The elbow point where the slope changes is determined as the optimum cluster.

Average Silhouette

This approach measures how similar each observation is to the cluster it is in compared to other clusters. An in-cluster quality score is determined. From here, the answer to the question of how many optimum clusters should be is obtained. It is used by changing the method of the function used for Elbow. The method indicates the number of sets it recommends on the chart.

Gap Method

First, the sample reference data set is created with Monte-Carlo simulation. The minimum and maximum values of each variable are taken and the remaining values are filled with the help of simulation. It then compares the total sums of the squares of the clusters found according to different k values with the expected values (for the reference data set).

5.7.3 Hierarchical Clustering

The k-centered clustering method has one drawback. You must first identify the number of clusters. To overcome this limitation, hierarchical clustering was devised. The hierarchical clustering algorithm's underlying logic is based on the combination or division of related features. There are two primary techniques, according to this working logic: agglomerative and divisive. All objects are originally separated from one another in the agglomerative technique, also known as bottom up. Each piece of information is thought to be a different set. After that, clusters with comparable properties are combined to form a single cluster. Unlike the agglomerative method, the top-bottom technique favors a separation strategy. There is only one cluster at the start of this method. Objects are split from the main set according to the distance / similarity matrix at each stage, resulting in different subsets. As a result of the process, each data becomes a cluster [23].

5.7.3.1 Agglomerative Hierarchical Clustering:

Initially there are as many sets of data as there are.

Step 1. The data set contains two observations that are closest to each other.

Step 2. These two points are brought together to create a new observation. In other words, the data set now consists of observations in the first combination.

Step 3. The same process is repeated and ascends upwards. In other words, the combined sets are combined according to their similarities in the same way and proceed upward. The process continues until all observations are collected in a single cluster.

Points close to each other are determined by distance criteria.

5.7.3.2 Divisive Hierarchical Clustering:

Initially there is one cluster, the whole data set.

Step 1. The cluster of all observations is divided into two.

Step 2. The newly formed clusters are divided into dissimilar subsets.

Step 3. The same process is repeated until the number of observations is reached

The number of clusters in hierarchical approaches is determined by examining the generated dendograms. The number of clusters is calculated before adopting non-hierarchical approaches. In hierarchical approaches, both observations and variables can be clustered, whereas in non-hierarchical approaches, only observations are clustered [11].

5.7.4 Principal Component Analysis (PCA)

It is based on a minimal number of variables being used to indicate the main aspects of multivariate data. The size of the variable is lowered in this way (with a minor amount of information loss). The amount of data that may be lost is expected to be minor, equivalent to mistake and noise. There is no correlation between components. Multiple linear connection problem is eliminated. These uncorrelated linear combinations are prime components. The important point in this method is to find the eigenvalues and eigenvectors of the covariance or correlation matrices [11].

A simple algorithm for PCA is as follows:

1. Standardization / Scaling. When variable variances are very different from each other, it is necessary to bring the values to close scales by standardizing.

2. Calculate the covariance matrix.

3. Compute the eigen values and vectors.

4. Re-orient the data.

5. Plot the data (PC1 against PC2)

6. SECTION

R APPLICATIONS OF MACHINE LEARNING ALGORITHMS IN COVID-19 DATASET

6.1 Data Preprocessing

First of all, Missing Data Analysis was performed. Initially, there were 2,975 case observations available. The names of the variables are: Hospital, Sex, Patient_Followup_Status, Case_Status, Contact, Pregnancy, Notification_District, Intensive_care, Intubation, Age , CT_Result, CT, Pneumonia, Process_Status, Province, District, Neighborhood, Ferritin, CRP, Oxygen.

After the missing Pregnancy cases and CT_Result observations were filled in with the "highest frequency result", the remaining observations with missing data were deleted to avoid errors in data processing, resulting in 2,438 non-missing observations.

In addition, the effect levels of the variables on the data were determined by leveling the Case_Status, Pregnancy, Intensive_care, Pneumonia and CT variables.

6.2 Explanatory Data Analysis And Data Visualization

6.2.1 First Look at Variable Types and Data

The mean, standard deviation and variance values of the numerical variables in the data were calculated as in Figure 6.1.

variable <chr>	mean <dbl>	std_dev <dbl>	variation_coef <dbl>
Age	39.800656	15.844820	0.39810449
Ferritin	209.787348	125.998789	0.60060242
CRP	1.029251	1.414659	1.37445446
Oxygen	97.780558	2.038972	0.02085253

Figure 6.1 First Look at Numeric Variable

The graph consisting of the frequencies of these numerical variables can be accessed from the Figure 6.2 below.

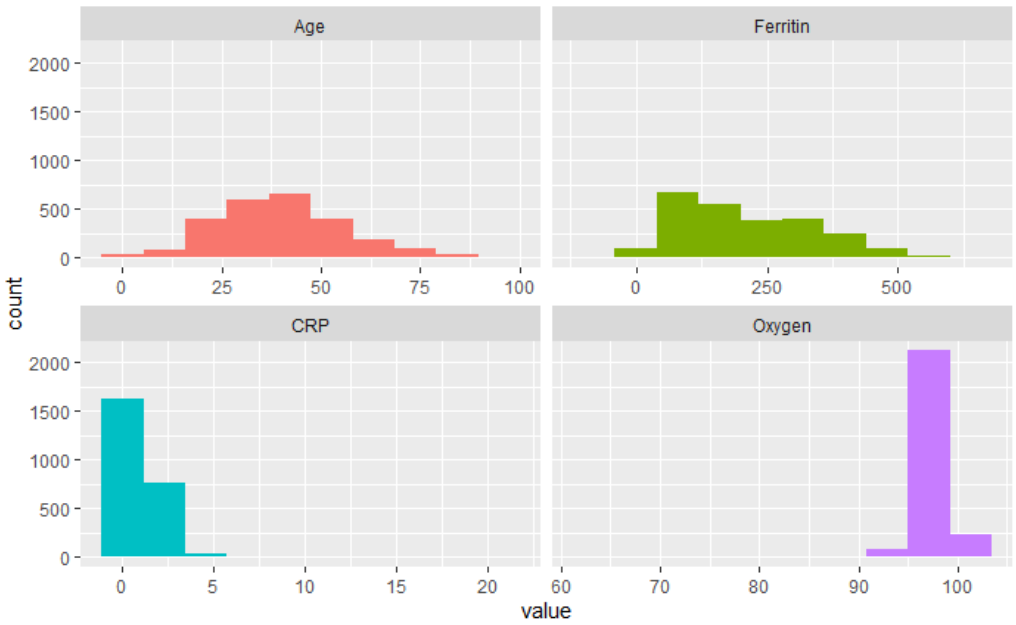


Figure 6.2 Frequences of Numeric Variables

The frequency distributions of the categorical variables in the data set are given in the graphs below. As seen in Figure 6.3, the number of male patients in the data set in the study is more than female patients. In Figure 6.4, it is seen that the patients in isolation at home are more than the patients followed in the hospital. The district distribution of the cases is shown in Figure 6.5.

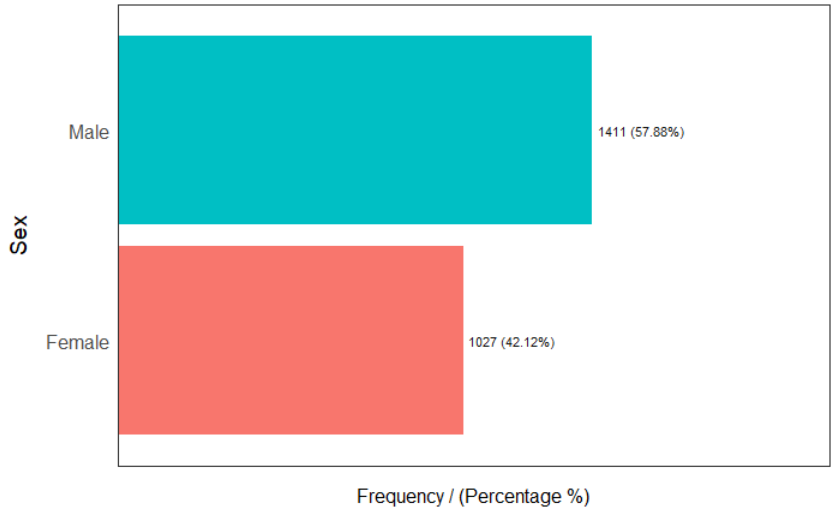


Figure 6.3 Sex Distribution

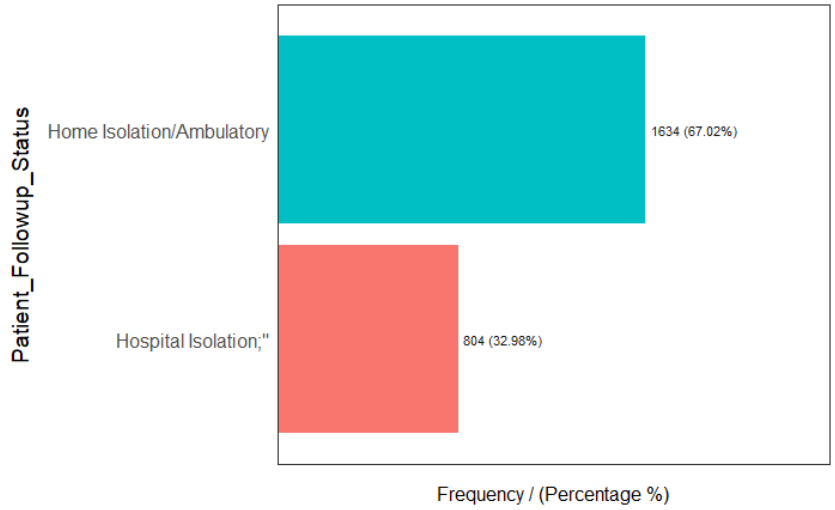


Figure 6.4 Patient Follow Up Distribution

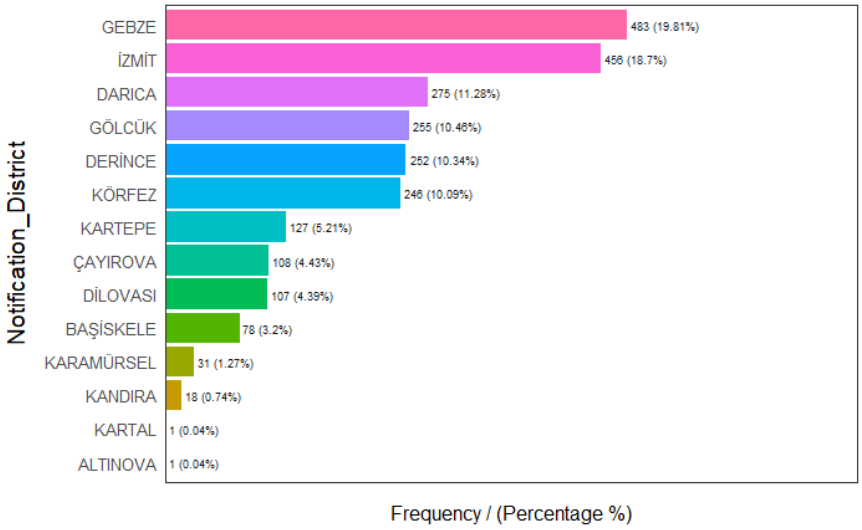


Figure 6.5 Notification District Distribution

The main focus, the Case_Status variable, has three classes: Bad, Medium, and Good. The number of cases included in these classes can be seen in Figure 6.6.

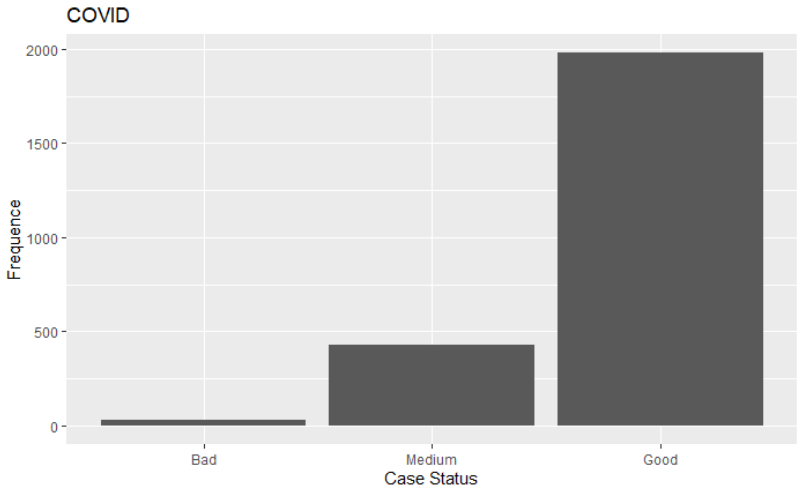


Figure 6.6 Case Status Distribution

The relationship between Case_Status and Oxygen is given in Figure 6.7 below. As can be seen, cases where the Oxygen level is at the desired level are more likely to be Medium and Good.

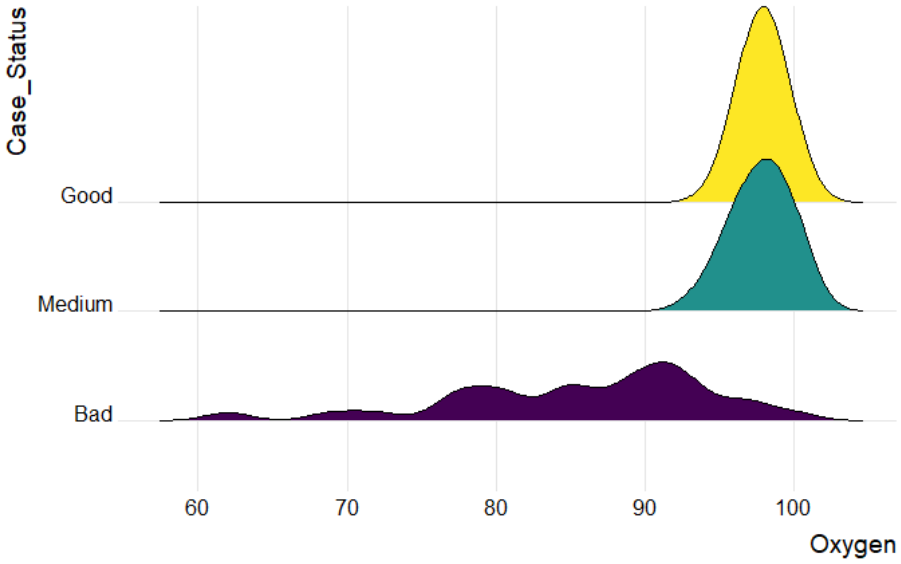


Figure 6.7 Relationship Between Case Status and Oxygen

The following Figure 6.8 Scatter Plot chart shows the positioning of Bad, Medium and Good cases according to their values in the Oxygen and CRP numerical variables.

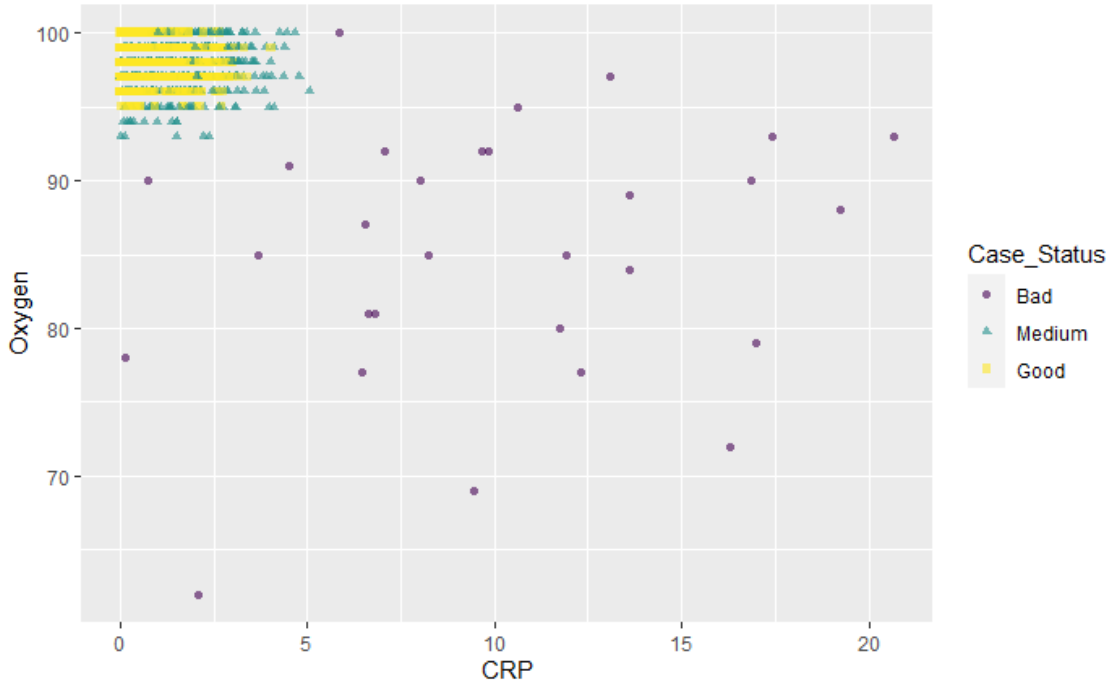


Figure 6.8 Positionings of Cases according to Oxygen and CRP

6.2.2 Clustered Heatmaps For Observations

With a clustered heatmap in Figure 6.9, it is seen that the values of the observations (each case) in the variables and in which cluster in the data set they are located. The first 6 observation (head()) functions were used in order not to spoil the visuality.

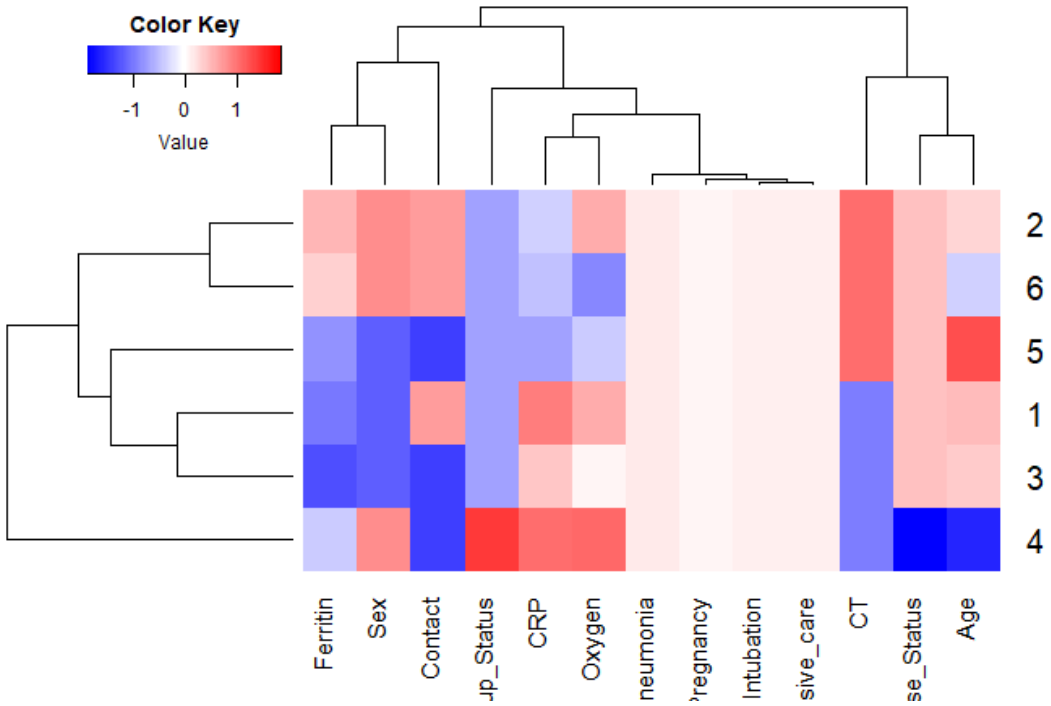


Figure 6.9 Clustered Heatmap

6.2.3 Correlation Matrices for Variables

The strength and direction of the linear relationship between the variables are shown with the correlation matrix in Figure 6.10. Dark blue dots indicate strong positive correlations, and dark red dots indicate strong negative correlations. Other circles indicate the strength and direction of the correlation according to their color and size.

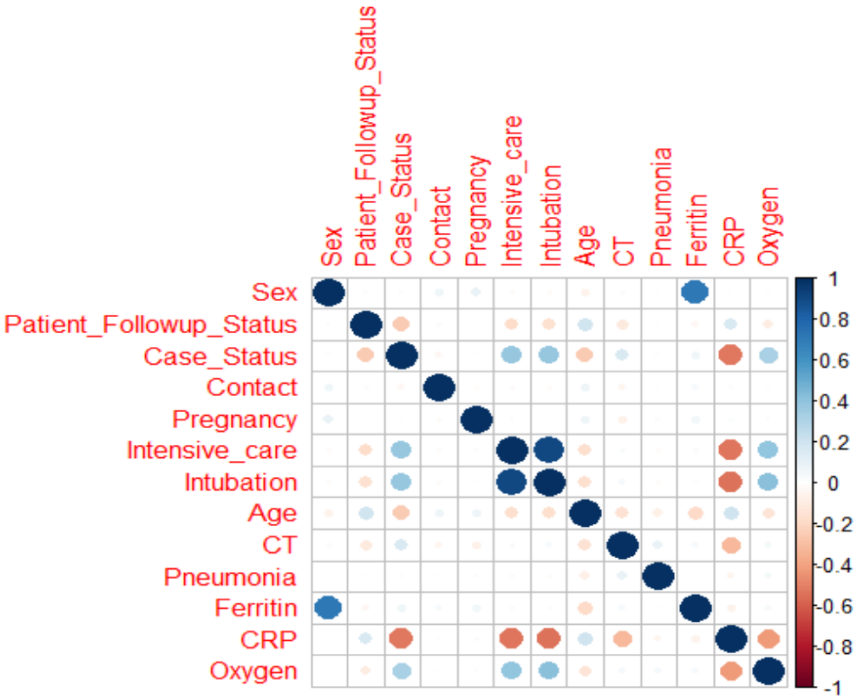


Figure 6.10 Correlations of Variables

In the Figure 6.11 below, strong positive correlations were collected on one diagonal and negative correlations on the other diagonal. Circles denoting weak correlations were crossed out.

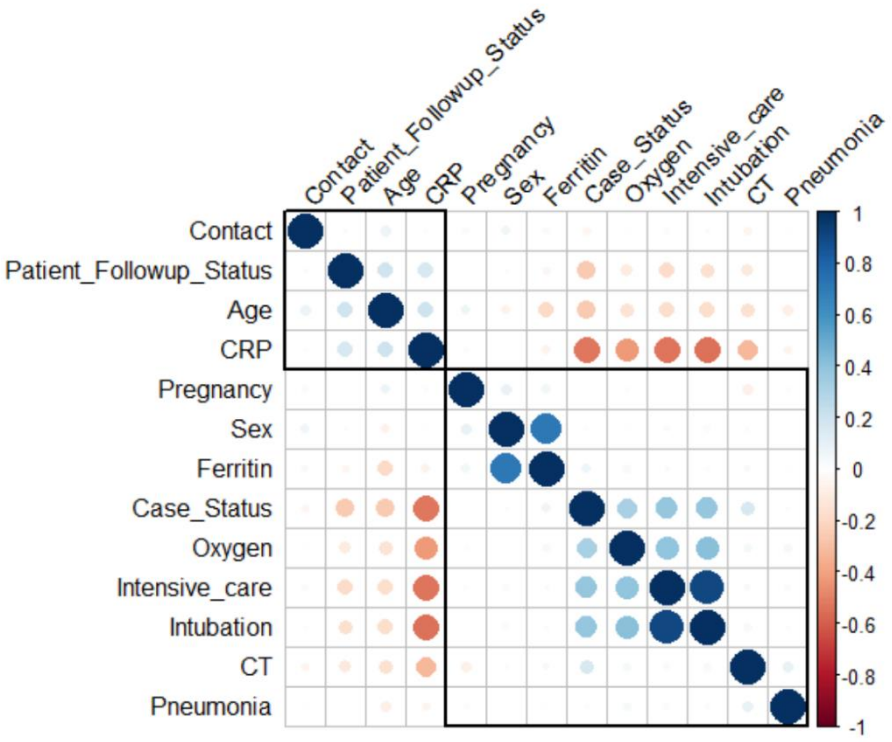


Figure 6.11 Fixed Correlations of Variables

6.3 Linear Regression

In this section, linear regression methods will be used for the prediction of “CRP” variable. So, the dependent variable is going to be “CRP” variable.

The first step of all linear regression methods is dividing the data set into two pieces as train set and test set. Train set consists of %80 of the data set.

6.3.1 Multiple Linear Regression

Before starting to apply MLR, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables which the number of categories is too many are separated.

6.3.1.1 Modeling

Firstly, the model is established with all variables by using “lm” function. In Figure 6.12, coefficients of variables, R-squared, and p-value are given. R-squared is 0.5798. And p-value is less than 0.05. On the right side of some variables, there are “*” signs. If there

is this sign, it means that the variable is significant. “Case_Status” variable has the most influence on the model.

```
Coefficients: (1 not defined because of singularities)
                                                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                                                    0.5472756  1.2827003   0.427  0.66968
SexMale                                                        0.0566113  0.0582298   0.972  0.33107
Patient_Followup_StatusHospital Isolation;"                   -0.0846640  0.0584322  -1.449  0.14752
Case_Status.L                                                 -4.9311350  0.2292489 -21.510  < 2e-16 ***
Case_Status.Q                                                  2.2915269  0.1341408  17.083  < 2e-16 ***
ContactYes                                                     0.0465522  0.0425608   1.094  0.27419
Pregnancy.L                                                    0.0344973  0.1735591   0.199  0.84247
Intensive_care.L                                              -0.3534631  0.2423328  -1.459  0.14484
IntubationYes                                                  1.2416402  0.3868252   3.210  0.00135 **
Age                                                           -0.0002292  0.0013689  -0.167  0.86705
CT_ResultCompatible with Mix Infection                         0.5061380  0.5299337   0.955  0.33965
CT_ResultCompatible with Viral Pneumonia                      -0.6817256  0.4036600  -1.689  0.09141 .
CT_ResultCOVID 19                                             -0.6327318  0.3929931  -1.610  0.10755
CT_ResultNon-Infectious CT Sign                               -0.2151759  0.7310430  -0.294  0.76853
CT_ResultNormal                                               -0.5882142  0.3930783  -1.496  0.13471
CT.L                                                          -0.5755674  0.0333231 -17.272  < 2e-16 ***
Pneumonia.L                                                   -0.0800705  0.0815484  -0.982  0.32628
Process_StatusContinues with Referral to the Hospital          0.0034781  0.0925508   0.038  0.97003
Process_StatusCurrently Discharged/Monitoring At Home Continues 0.0022704  0.0523708   0.043  0.96542
Process_StatusDischarged with Healing / Monitoring at Home Continues 0.0400614  0.1021167   0.392  0.69487
Process_StatusHome Monitoring Continues                              NA         NA      NA       NA
Process_StatusHospital Monitoring Continues                    0.2563608  0.0859842   2.981  0.00290 **
Ferritin                                                      -0.0003655  0.0002298  -1.590  0.11192
Oxygen                                                         0.0377928  0.0128102   2.950  0.00321 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.871 on 1927 degrees of freedom
Multiple R-squared:  0.5846,    Adjusted R-squared:  0.5798
F-statistic: 123.2 on 22 and 1927 DF,  p-value: < 2.2e-16
```

Figure 6.12 MLR model output

To establish a model that consists of only significant variables, “dropterm” function was used. “dropterm” function tries fitting all models that differ from the current model by dropping a single term, maintaining marginality [44]. Figure 6.13 shows which variables are significant.

```
Model:
CRP ~ Sex + Patient_Followup_Status + Case_Status + Contact +
    Pregnancy + Intensive_care + Intubation + Age + CT_Result +
    CT + Pneumonia + Process_Status + Ferritin + Oxygen
                      Df Sum of Sq    RSS     AIC F Value     Pr(F)
<none>                             1462.0 -515.61
Sex                    1      0.72 1462.8 -516.65   0.945  0.331071
Patient_Followup_Status 0     0.00 1462.0 -515.61
Case_Status            2    416.56 1878.6  -30.75 274.517 < 2.2e-16 ***
Contact                1      0.91 1462.9 -516.40   1.196  0.274187
Pregnancy              1      0.03 1462.1 -517.57   0.040  0.842468
Intensive_care         1      1.61 1463.7 -515.45   2.127  0.144842
Intubation             1      7.82 1469.8 -507.21  10.303  0.001350 **
Age                    1      0.02 1462.0 -517.58   0.028  0.867052
CT_Result              5     10.29 1472.3 -511.93   2.711  0.018965 *
CT                     1    226.35 1688.4 -236.92 298.333 < 2.2e-16 ***
Pneumonia              1      0.73 1462.8 -516.63   0.964  0.326284
Process_Status         4      7.37 1469.4 -513.80   2.429  0.045894 *
Ferritin               1      1.92 1464.0 -515.05   2.529  0.111924
Oxygen                 1      6.60 1468.6 -508.82   8.704  0.003214 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figure 6.13 Significance variables of MLR model

In Figure 6.14, coefficients of the new model which is established according to the result of “dropterm” function that is applied to the previous model are given. The new model’s R-squared is 0.5798 and p-value is less than 0.05.

```
Coefficients:
                                                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                                                        0.15838    1.24654   0.127  0.89891
Case_Status.L                                                     -5.00895    0.22310 -22.452  < 2e-16 ***
Case_Status.Q                                                      2.33657    0.13112  17.819  < 2e-16 ***
IntubationYes                                                      1.66282    0.25122   6.619 4.67e-11 ***
CT_ResultCompatible with Mix Infection                             0.47939    0.52930   0.906  0.36521
CT_ResultCompatible with Viral Pneumonia                          -0.69621    0.40310  -1.727  0.08431 .
CT_ResultCOVID 19                                                 -0.65310    0.39255  -1.664  0.09633 .
CT_ResultNon-Infectious CT Sign                                   -0.24369    0.73002  -0.334  0.73856
CT_ResultNormal                                                   -0.61489    0.39192  -1.569  0.11683
CT.L                                                              -0.57335    0.03316 -17.292  < 2e-16 ***
Process_StatusContinues with Referral to the Hospital              0.00664    0.09223   0.072  0.94261
Process_StatusCurrently Discharged/Monitoring At Home Continues    0.08966    0.06060   1.480  0.13914
Process_StatusDischarged with Healing / Monitoring at Home Continues 0.12183  0.10631   1.146  0.25195
Process_StatusHome Monitoring Continues                            0.07889    0.05810   1.358  0.17468
Process_StatusHospital Monitoring Continues                        0.27440    0.08548   3.210  0.00135 **
Oxygen                                                             0.03831    0.01276   3.002  0.00271 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.871 on 1934 degrees of freedom
Multiple R-squared:  0.5831,    Adjusted R-squared:  0.5798
F-statistic: 180.3 on 15 and 1934 DF,  p-value: < 2.2e-16
```

Figure 6.14 MLR model output after “dropterm” function

When “dropterm” function is applied to the new model, it can be seen that all variables used are significant. In Figure 6.15, the new model and variables’ significance statuses are given.

```
Model:
CRP ~ Case_Status + Intubation + CT_Result + CT + Process_Status +
    Oxygen
               Df Sum of Sq    RSS     AIC F Value     Pr(F)
<none>                      1467.3 -522.57
Case_Status     2    449.93 1917.2   -5.03 296.517 < 2.2e-16 ***
Intubation      1     33.24 1500.6 -480.89  43.810 4.673e-11 ***
CT_Result       5     10.20 1477.5 -519.06   2.689  0.019840 *
CT              1    226.86 1694.2 -244.23 299.016 < 2.2e-16 ***
Process_Status  5      8.67 1476.0 -521.07   2.286  0.043882 *
Oxygen          1      6.84 1474.2 -515.50   9.015  0.002713 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 6.15 Significance variables of MLR model after "dropterm" function

6.3.1.2 Prediction

Train errors are given in Figure 6.16. R-squared is 0.5166.

```
     RMSE  Rsquared       MAE
0.8674511 0.5830528 0.5166427
```

Figure 6.16 MLR train error

Test errors are given in Figure 6.17. R-squared is 0.6779.

```
     RMSE  Rsquared       MAE
1.0158795 0.6778781 0.5263543
```

Figure 6.17 MLR test error

6.3.1.3 Detection of Outliers

For detection of outliers, Cook's Distance was used. Cook's distance is the scaled change in fitted values, which is useful for identifying outliers in the X values (observations for predictor variables). Cook's distance shows the influence of each observation on the fitted response values. An observation with Cook's distance larger than three times the mean Cook's distance might be an outlier [45]. In Figure 6.18, observations are seen and observations on the upper side of the red line are detected as outliers.

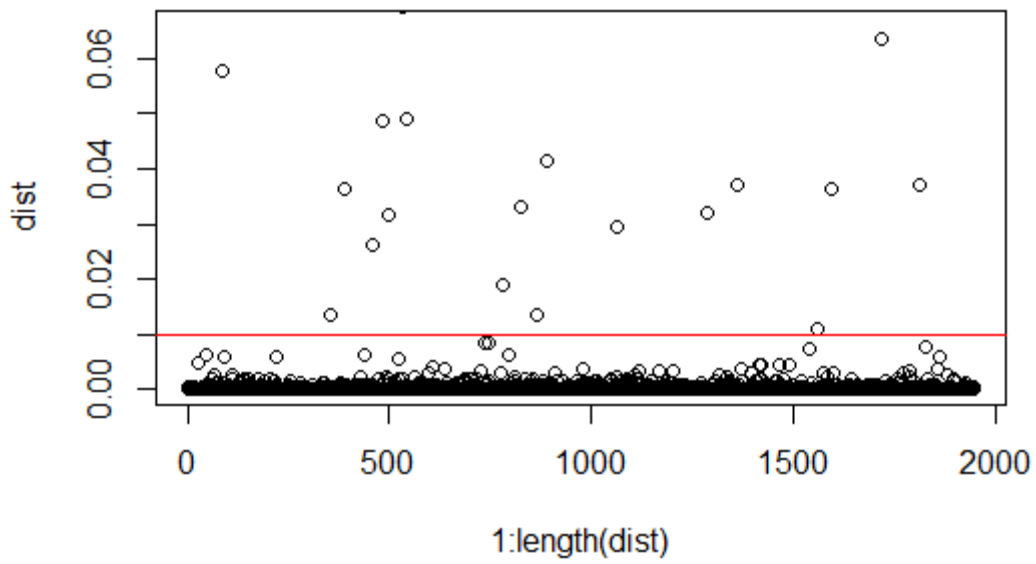


Figure 6.18 Outliers of MLR model

After outliers are removed from the data set, the model is set up with new dataset and variables that are the same variables used in Figure 6.15. Coefficients of the model are given in Figure 6.19. R-squared is 0.4639 and p-value is less than 0.05.

```
Coefficients:
                                                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                                                        2.52463    1.26010   2.004 0.045264 *
Case_Status.L                                                     -4.97902    0.28233 -17.636  < 2e-16 ***
Case_Status.Q                                                      2.31479    0.16366  14.144  < 2e-16 ***
IntubationYes                                                      1.31546    0.38736   3.396 0.000698 ***
CT_ResultCompatible with Mix Infection                             0.03154    0.60718   0.052 0.958579
CT_ResultCompatible with Viral Pneumonia                          -0.62953    0.39135  -1.609 0.107871
CT_ResultCOVID 19                                                 -0.75584    0.38498  -1.963 0.049752 *
CT_ResultNormal                                                   -0.66781    0.38444  -1.737 0.082535 .
CT.L                                                              -0.57602    0.02537 -22.705  < 2e-16 ***
Process_StatusContinues with Referral to the Hospital              0.07576    0.07222   1.049 0.294304
Process_StatusCurrently Discharged/Monitoring At Home Continues    0.08729    0.04655   1.875 0.060909 .
Process_StatusDischarged with Healing / Monitoring at Home Continues 0.09334  0.08117   1.150 0.250338
Process_StatusHome Monitoring Continues                            0.05462    0.04459   1.225 0.220827
Process_StatusHospital Monitoring Continues                        0.15843    0.06715   2.359 0.018415 *
Oxygen                                                             0.01483    0.01258   1.180 0.238322
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6638 on 1901 degrees of freedom
Multiple R-squared:  0.4678,    Adjusted R-squared:  0.4639
F-statistic: 119.4 on 14 and 1901 DF,  p-value: < 2.2e-16
```

Figure 6.19 MLR model output after outliers are removed

Train errors of the model that outliers are removed are given in Figure 6.20. R-squared is 0.4678.

```
     RMSE  Rsquared       MAE
0.6611859 0.4678333 0.4613263
```

Figure 6.20 MLR train error of the final model

Test errors of the model that outliers are removed are given in Figure 6.21. R-squared is 0.6925.

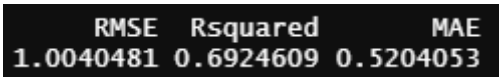


Figure 6.21 MLR test error of the final model

6.3.2 Ridge Regression

Before the application of ridge regression, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables were separated from the data set. Numeric variables were scaled and the data set converted into matrix format. Also, One Hot Encoding method was applied to convert categorical variables into dummy variables. Because categorical variables should be binary. One Hot Encoding method converts factors of categoric variables into columns. After applying this method, each factor will have its own column. If the factor is the factor of the observation, it is 1. Otherwise, it is 0. The new variables which are created by One Hot Encoding are called dummy variables. Additionally, numeric variables were scaled.

6.3.2.1 Modeling

Model is set up by using “glmnet” function. Lambda parameter was accepted as 0 in this model. The alpha parameter is 0 in ridge regression.

In Figure 6.22, the changes of the coefficients according to the lambda parameter are seen. No coefficient has been 0 but they tend to decrease.

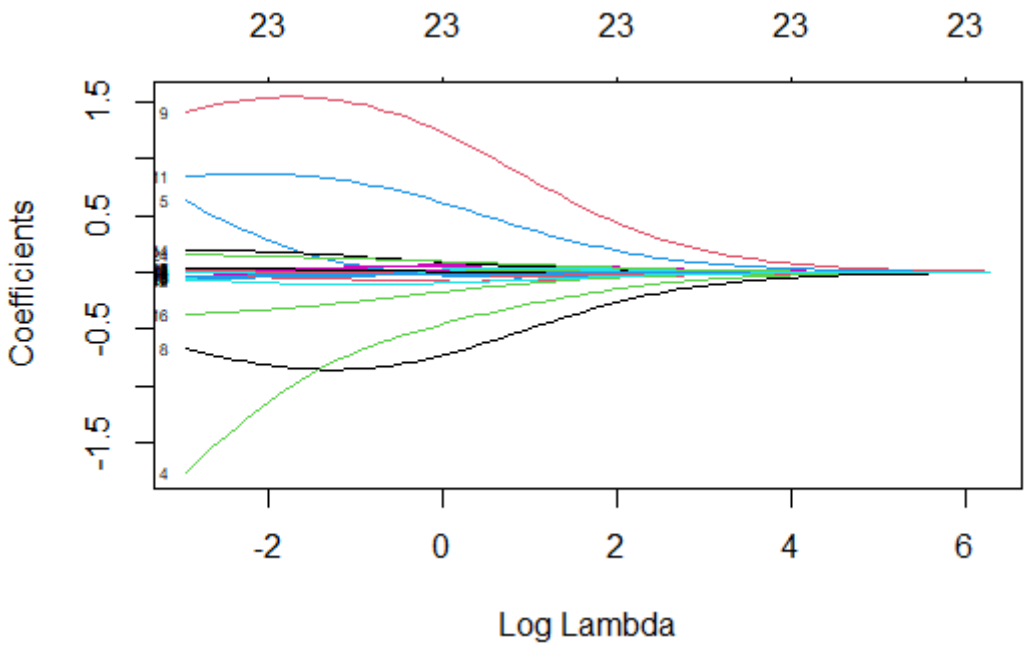


Figure 6.22 L2 model - The change of the coefficients

In the result of this model, R-squared for the train set is 0.3006. Results are given in Figure 6.23.

```
     RMSE  Rsquared       MAE
0.8088439 0.2944159 0.4800343
```

Figure 6.23 L2 model train error

Test errors are given in Figure 6.24. The results of the train and test sets are similar.

```
     RMSE  Rsquared       MAE
1.0321068 0.2996448 0.4983362
```

Figure 6.24 L2 model test error

6.3.2.2 Model Tuning

The success of the model is dependent on the lambda parameter. Thus, it is trying to find the appropriate lambda parameter in this section.

All lambda parameters were tried from 1000 to 0.01 by decreasing 0.01 parameter in every step. “cv.glmnet” function was used. “nfold” parameter was 10. Figure 6.25, shows the relation of lambda parameter and model success.

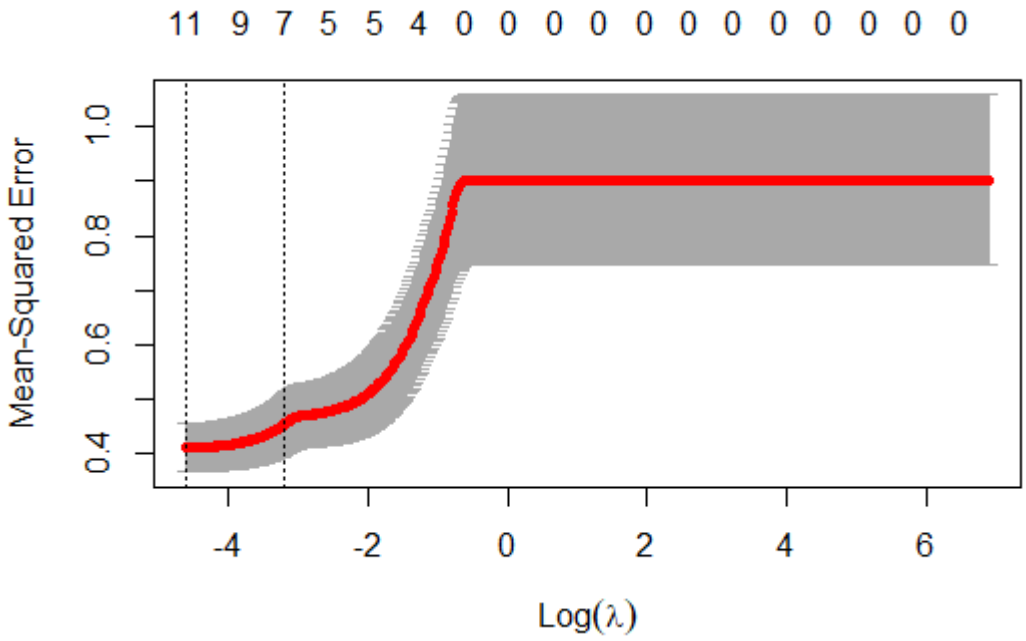


Figure 6.25 L2 model - The change of the MSE according to λ

According to model tuning, the ideal lambda parameter is determined as 0.01.

6.3.2.3 Model Prediction Performance After Model Tuning

The new model is established with the ideal lambda parameter which is 0.01. Train set results are given in Figure 6.26.

```
     RMSE  Rsquared       MAE
0.6144196 0.5817617 0.3668818
```

Figure 6.26 L2 model train error after model tuning

The test set results are given in Figure 6.27.

```
     RMSE  Rsquared       MAE
0.7444305 0.6579241 0.3773829
```

Figure 6.27 L2 model test error after model tuning

6.3.3 Lasso Regression

Before starting to apply lasso regression, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables were separated from the data set. Then, the data set is converted into matrix format. Categoric variables were converted into dummy variables with One Hot Encoding. The alpha parameter is equal to 1 in lasso regression.

6.3.3.1 Modeling

In this section, modeling was done for each alpha parameters which are decreasing from 100 to 0.01 by 0.01. Figure 6.28 shows the change of coefficients according to lambda parameters. In general, the coefficients decrease as the lambda value increases. When the logarithm of lambda is almost equal to 0, all coefficients change to 0.

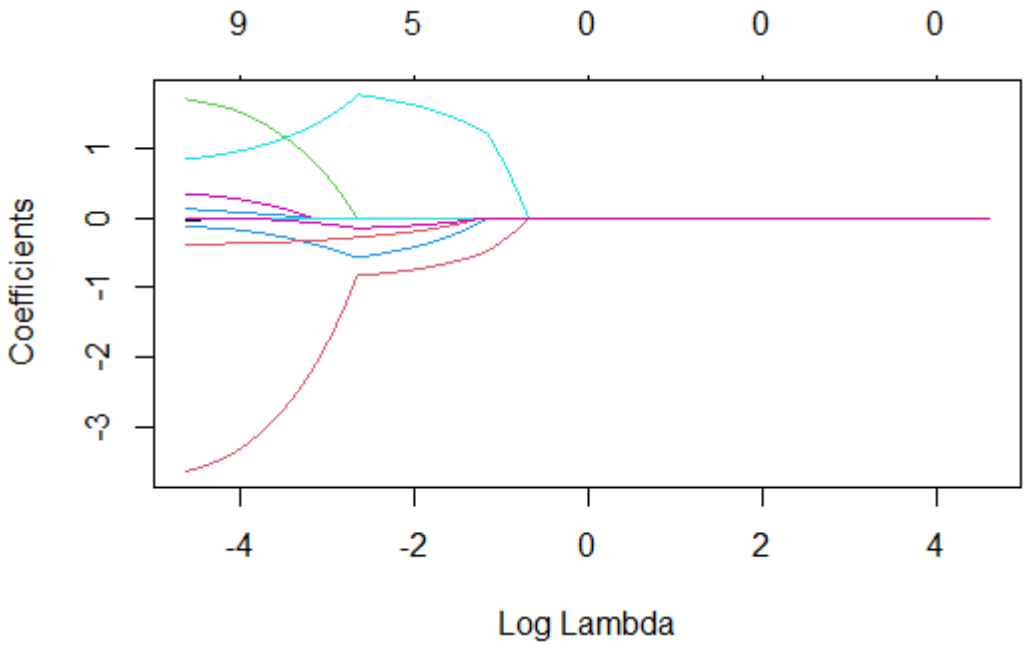


Figure 6.28 L1 model - The change of the coefficients

6.3.3.2 Model Tuning

Model tuning is done for lambda parameters used in the modeling section. The relation between mean squared error and logarithm of lambda is given in Figure 6.29. The best lambda that is the result of model tuning is determined as 0.01.

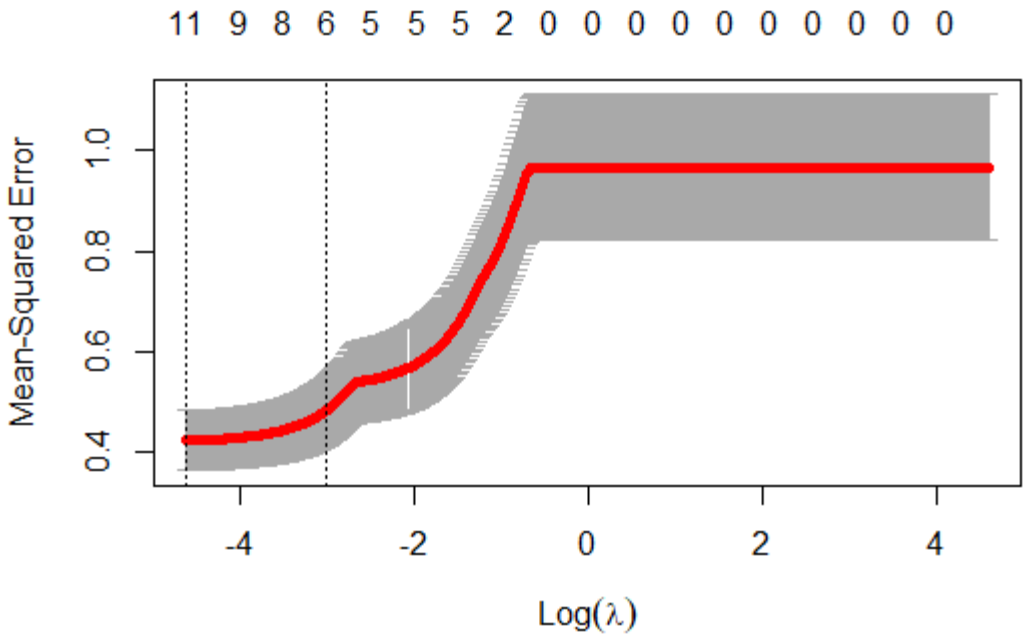


Figure 6.29 L1 model - The change of the MSE according to λ

6.3.3.3 Model Prediction Performance Evaluation

The determined lambda parameter is used for the new model. The result of the new model for the train set is given in Figure 6.30.

```
     RMSE  Rsquared       MAE
0.6219810 0.6009716 0.3639204
```

Figure 6.30 L1 model train error after model tuning

Test errors are given in Figure 6.31.

```
     RMSE  Rsquared       MAE
0.6647013 0.5981432 0.3716379
```

Figure 6.31 L1 model test error after model tuning

6.3.4 Elastic Regression

In the first step, the data preprocessing, which is the same as what is done in ridge regression and lasso regression, was done.

6.3.4.1 Modeling

Modeling was done for each alpha parameter which is decreasing from 100 to 0.01 by 0.01. The alpha parameter is accepted as 0.5 for all models established. In Figure 6.32,

the changes of coefficients according to the logarithm of lambda are seen. When the logarithm of lambda is almost equal to 2, all coefficients are 0.

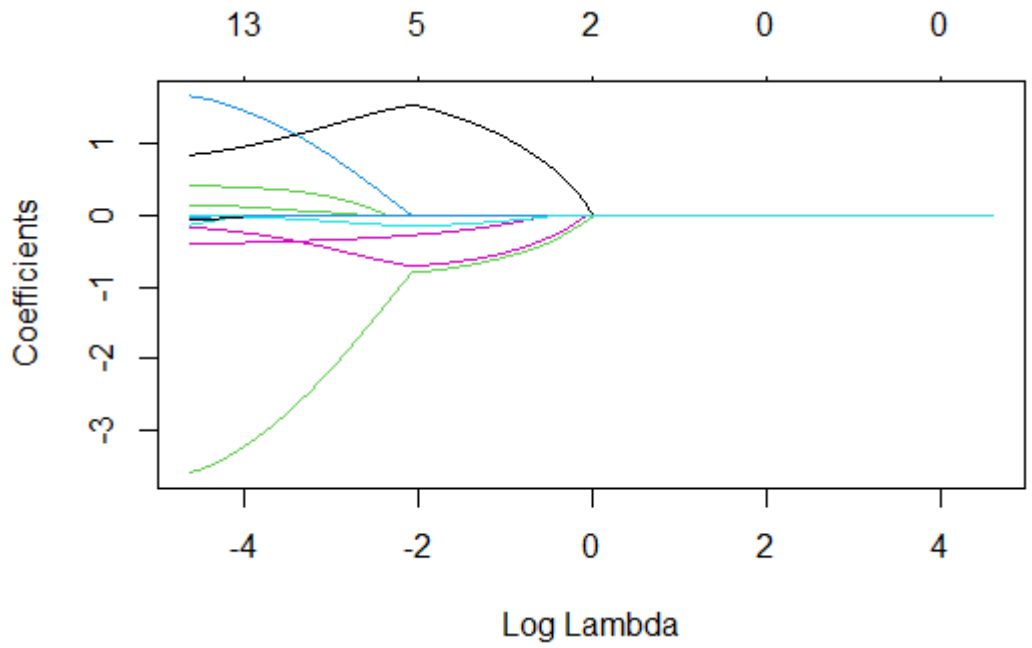


Figure 6.32 Elasticnet Regression model - The change of the coefficients

6.3.4.2 Model Tuning with Random Search

It is necessary that model tuning will be done for alpha and lambda parameters. Tune length was accepted as 30. RMSE, R-squared and MAE were calculated for 30 randomly selected alpha and lambda parameters, as seen in Figure 6.33. The best alpha value was found as 0.9625183 and the best lambda value was found as 0.006940991. When these parameters were used in the model, RMSE and MAE were minimized, R-squared was maximized.

alpha	lambda	RMSE	Rsquared	MAE
0.003819999	0.126076974	0.6869227	0.4995976	0.3962588
0.038894081	3.952455033	0.8566569	0.4503963	0.5176932
0.039905200	0.070865859	0.6719291	0.5196313	0.3883192
0.045927682	0.001218694	0.6444077	0.5596620	0.3743283
0.093213422	0.108524856	0.6831069	0.5057945	0.3938857
0.152251354	0.001287175	0.6442854	0.5598173	0.3741698
0.167353947	0.548720551	0.7384148	0.4666120	0.4393163
0.173980918	1.511765543	0.8544285	0.4119935	0.5214088
0.192765232	0.091108205	0.6798840	0.5106299	0.3922625
0.193049215	0.270218590	0.7168695	0.4676996	0.4155618
0.241640392	0.121939623	0.6920344	0.4955934	0.3986766
0.247812303	0.001813794	0.6440162	0.5599479	0.3739094
0.321009526	0.698857813	0.8067552	0.4320098	0.4929202
0.356563486	0.071986387	0.6757496	0.5167430	0.3902715
0.367566071	0.051103743	0.6643178	0.5310483	0.3845454
0.377854486	0.001954405	0.6438169	0.5601789	0.3736893
0.384349870	0.106391883	0.6947881	0.4927409	0.4000241
0.385277187	4.820964377	0.9536256	NaN	0.5651594
0.395038988	5.548606222	0.9536256	NaN	0.5651594
0.435767065	5.235123082	0.9536256	NaN	0.5651594
0.450040051	0.017307768	0.6449853	0.5555065	0.3741562
0.520878227	7.185448447	0.9536256	NaN	0.5651594
0.560668888	0.228728644	0.7361767	0.4608961	0.4341185
0.561984668	2.205065921	0.9536256	NaN	0.5651594
0.585401208	0.008692617	0.6422232	0.5602016	0.3724057
0.601590991	0.186582744	0.7286697	0.4631391	0.4262646
0.718185942	0.004806622	0.6425931	0.5608676	0.3724639
0.802335471	0.005571152	0.6421950	0.5611790	0.3721296
0.894285213	0.031917096	0.6540698	0.5455360	0.3797240
0.962518321	0.006940991	0.6413602	0.5619217	0.3716296

RMSE was used to select the optimal model using the
 smallest value.
The final values used for the model were alpha = 0.9625183
 and lambda = 0.006940991.

Figure 6.33 Elasticnet Regression model - Prediction errors according to α and λ in random search

Train errors of the model that set up with parameters came from random search is given Figure 6.34.

RMSE	Rsquared	MAE
0.6205230	0.6023002	0.3632341

Figure 6.34 Elasticnet Regression model train error after random search

Test errors were given in Figure 6.35.

RMSE	Rsquared	MAE
0.6655228	0.5967628	0.3723671

Figure 6.35 Elasticnet Regression model test error after random search

6.3.4.3 Model Tuning with Grid Search

In the grid search, lambda parameters that were used are from 0 to 0.5, and alpha parameters that were used are from 0 to 3. In the grid search result, the final values used for the model were alpha = 1 and lambda = 0.01. Figure 6.36 shows the output of model tuning.

alpha <dbl>	lambda <dbl>
1	0.01

Figure 6.36 The best α and λ according to grid search

For the train set, the prediction performance of the model that is set up with the parameters are seen in Figure 6.36 are given in Figure 6.37.

RMSE	Rsquared	MAE
0.6219763	0.6009767	0.3639131

Figure 6.37 Elasticnet Regression model train error after grid search

The prediction performance of the model for the test set is given in Figure 6.38.

RMSE	Rsquared	MAE
0.6647016	0.5981398	0.3716422

Figure 6.38 Elasticnet Regression model test error after grid search

6.3.5 Principal Component Regression

In the first step, "Hospital", "Notification_District", “Process_Status”, "Province", "District", "Neighborhood" variables are separated from the data set because they have too many categories.

6.3.5.1 Modeling

For modeling, “pcr” function was used. Figure 6.39 gives MSEP values according to the number of components.

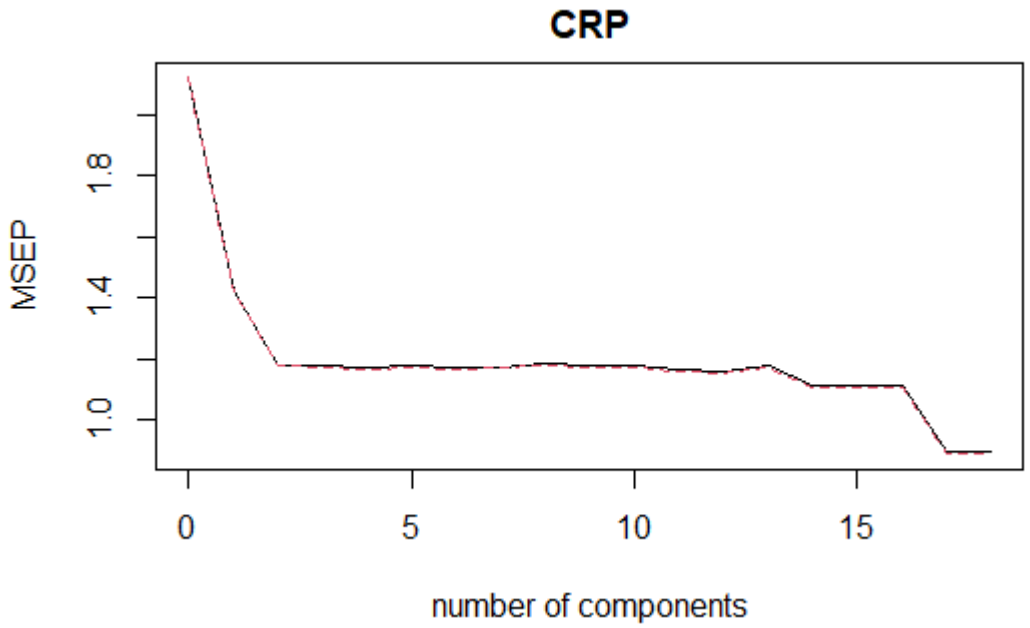


Figure 6.39 PCR - The change of MSEP according to the number of components

6.3.5.2 Model Prediction Performance

Errors for the prediction of the train set are given in Figure 6.40.

```
     RMSE  Rsquared       MAE
1.0210972 0.5080140 0.6126317
```

Figure 6.40 PCR model train error

Errors for the prediction of the test set are given in Figure 6.41. R-squared is pretty less than R-squared value obtained from the train set. So, it can be said that there is overfitting.

```
     RMSE  Rsquared       MAE
1.1139000 0.3123042 0.6806450
```

Figure 6.41 PCR model test error

6.3.5.3 Model Tuning

Model tuning is done for finding the optimal number of components. The output of model tuning is given in Figure 6.42. Prediction errors according to component numbers can be examined. The final value of the component number is 10.

```
Principal Component Analysis

1952 samples
  14 predictor

Pre-processing: centered (3), scaled (3), ignore (11)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1756, 1757, 1757, 1756, 1757, 1758, ...
Resampling results across tuning parameters:

  ncomp  RMSE      Rsquared   MAE
   1     1.296216  0.1547411  0.7888277
   2     1.216364  0.2763662  0.7579632
   3     1.214863  0.2884890  0.7620086
   4     1.159682  0.3552357  0.6682965
   5     1.160259  0.3540892  0.6740313
   6     1.159237  0.3545087  0.6720997
   7     1.154006  0.3620009  0.6653344
   8     1.154474  0.3616318  0.6655179
   9     1.139988  0.3784393  0.6459359
  10     1.138989  0.3791944  0.6468892
  11     1.140024  0.3785251  0.6483035
  12     1.139257  0.3801384  0.6493587
  13     1.139425  0.3798655  0.6502139

RMSE was used to select the optimal model using the
 smallest value.
The final value used for the model was ncomp = 10.
```

Figure 6.42 PCR - Prediction performances for different number of components

The relation of the RMSE and component numbers was visualized in Figure 6.43.

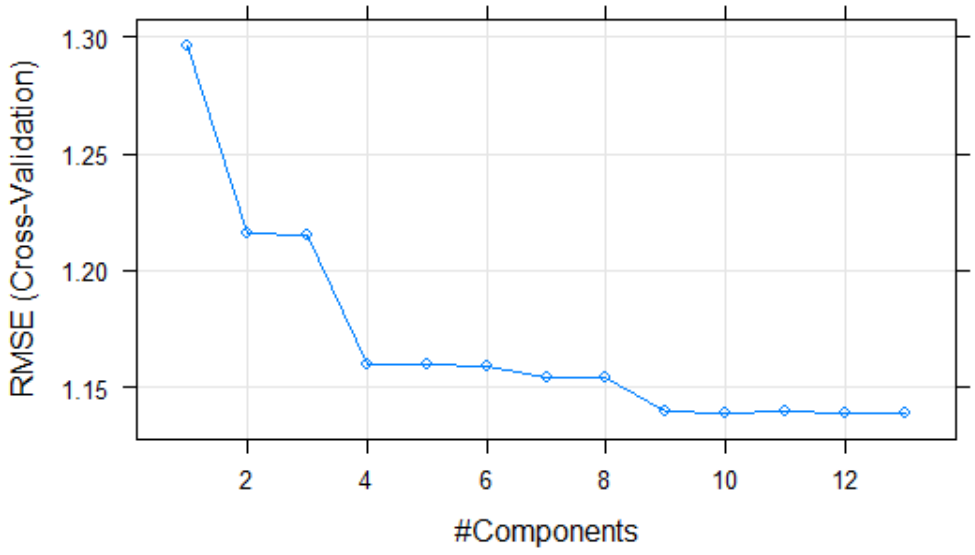


Figure 6.43 The RMSE values for different number of components

6.3.5.4 Model Prediction Performance After Model Tuning

Prediction success for the train set after model tuning is given in Figure 6.44.

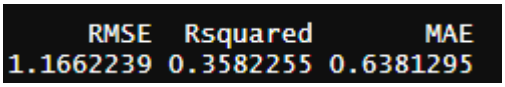


Figure 6.44 PCR model train error after model tuning

Prediction success for the test set after model tuning is given in Figure 6.45.

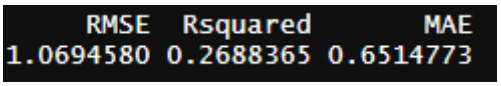


Figure 6.45 PCR model test error after model tuning

Prediction success is nearly %27. After model tuning, RMSE increased and R-squared decreased. It can be said that PCR is not an appropriate method for the data set.

6.3.6 Partial Least Squares Regression

For data preprocessing, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables are removed from the data set.

6.3.6.1 Modeling

Modeling for the PLS method is done with “plsr” function. According to Figure 6.46, it can be said that MSEP is decreasing with the increase in the number of components. But, after the number of components is almost equal to 15, the change of MSEP stops.

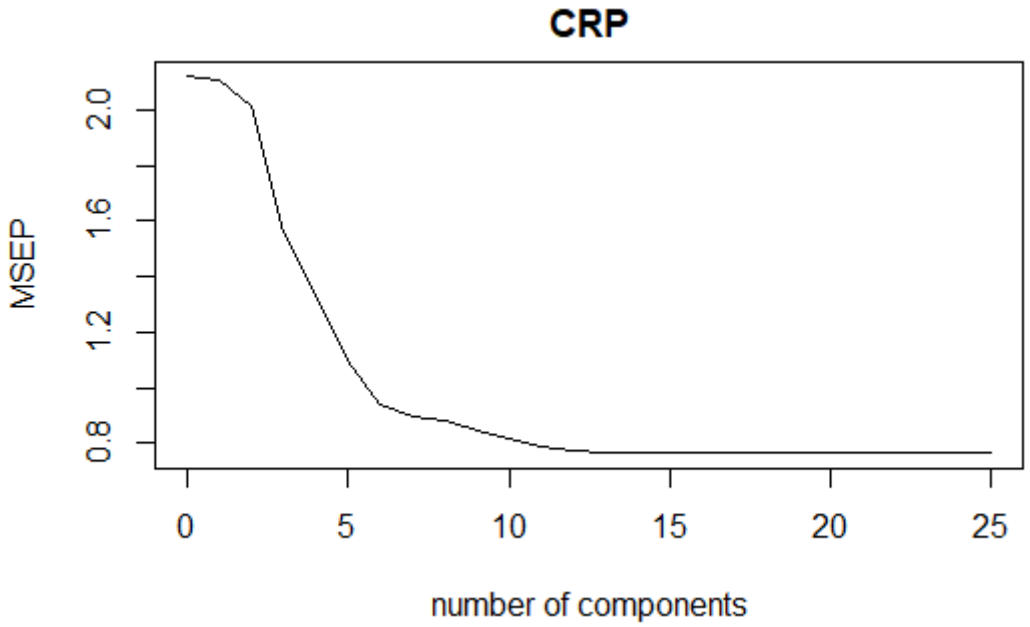


Figure 6.46 PLSR - The change of MSEP according to number of components

6.3.6.2 Prediction Performance

Prediction results for the train set are given in Figure 6.47.

```
     RMSE  Rsquared       MAE
0.9802449 0.5465935 0.5662494
```

Figure 6.47 PLSR model train error

Prediction results for the test set are given in Figure 6.48. R-squared was 0.55 for train set prediction. However, it decreased to 0.42 in test set prediction. It can be said that there is an overfitting problem.

```
     RMSE  Rsquared       MAE
1.0245338 0.4181848 0.5736433
```

Figure 6.48 PLSR model test error

6.3.6.3 Model Tuning

The aim of the model tuning is to find the ideal number of components which maximizes the prediction success. Figure 6.49 gives information about prediction successes according to the number of components. In the result of model tuning, the final number of components value used for the model was determined as 13.

```
Partial Least Squares

1952 samples
  14 predictor

Pre-processing: centered (3), scaled (3), ignore (11)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1757, 1756, 1756, 1757, 1758, 1757, ...
Resampling results across tuning parameters:

  ncomp  RMSE       Rsquared   MAE
   1     1.1464619  0.3232744  0.7036310
   2     1.1022369  0.3738227  0.6541536
   3     1.0684539  0.4292816  0.6201599
   4     1.0125412  0.4847395  0.6004350
   5     0.9856958  0.5012866  0.5989310
   6     0.9595592  0.5345595  0.5614260
   7     0.9530957  0.5433000  0.5560110
   8     0.9466231  0.5481955  0.5511062
   9     0.9328398  0.5585658  0.5477906
  10     0.9212920  0.5640589  0.5404549
  11     0.9107394  0.5738106  0.5413877
  12     0.9039248  0.5791402  0.5345898
  13     0.9014150  0.5810552  0.5316907

RMSE was used to select the optimal model using the
 smallest value.
The final value used for the model was ncomp = 13.
```

Figure 6.49 PLSR - Prediction performances for different number of components

Figure 6.50 visualizes the change of RMSE according to the number of components.

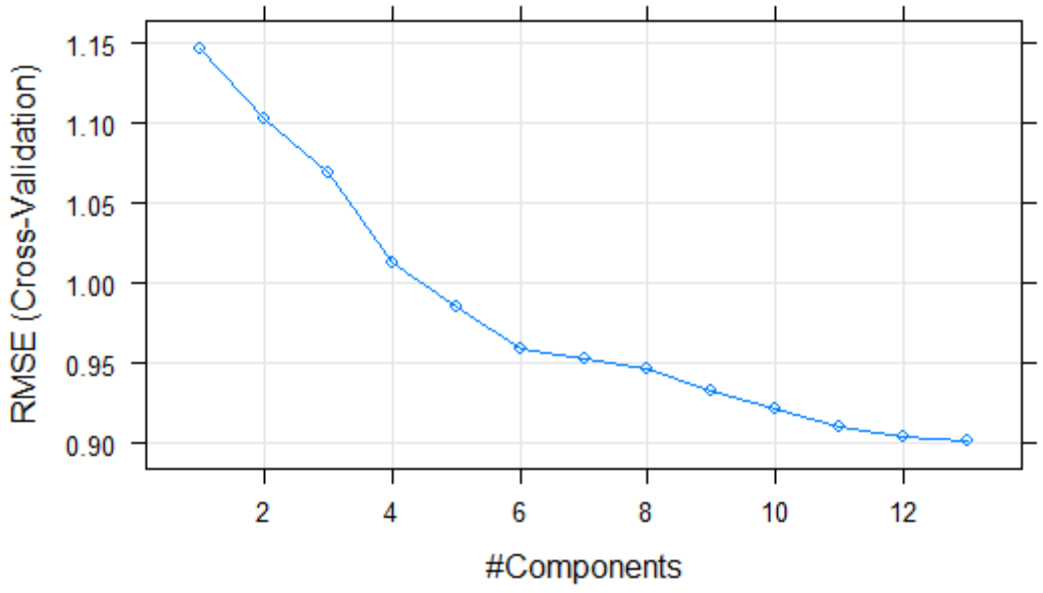


Figure 6.50 PLSR – The RMSE value for different number of components

6.3.6.4 Prediction Performance After Model Tuning

Performances of the prediction which is done with the tuned model for the train set are given in Figure 6.51.

```
     RMSE  Rsquared       MAE
0.8760623 0.6378499 0.5191051
```

Figure 6.51 PLSR model train error after model tuning

Prediction performances for the test set prediction are given in Figure 6.52. R-squared dropped from 0.64 to 0.49, so an overfitting problem can exist.

```
     RMSE  Rsquared       MAE
0.9813841 0.4924394 0.5279559
```

Figure 6.52 PLSR model test error after model tuning

6.3.7 Comparison the Prediction Performances of the Methods

Table 6.1 Comparison of the linear regression methods

Linear Regression	**Test Set**		
	RMSE	**Rsquared**	**MAE**
Multiple Linear Regression	1,004048	0,692461	0,520405
Ridge Regression	0,744431	0,657924	0,377383
Lasso Regression	0,664701	0,598143	0,371638
Elasticnet Regression	0,664702	0,59814	0,371642
Principal Component Regression	1,069458	0,268837	0,651477
Partial Least Squares	0,981384	0,492439	0,527956

Table 6.1 summarizes the prediction performances of the methods. The highest R-squared belongs to the multiple linear regression method. But its RMSE and MAE are not the lowest. The R-squared of ridge regression is the second highest and RMSE and MAE values are smaller than multiple linear regression.

6.4 Non-Linear Regression

Different non-linear regression methods are applied to the data set, in this section. It will be aimed to predict “CRP” variable. In data preprocessing of all methods, the data set was divided as train set and test set. Train set consists of %80 of the data set.

6.4.1 K-Nearest Neighbor Algorithm

In the first step of the KNN "Hospital", "Notification_District", "CT_Result", "Process_Status", "Province", "District", "Neighborhood" were removed from data set. Because these variables have too many categories. Remained categoric variables are converted into numerical categoric variables such as “Case_Status”. For “Case_Status”, bad cases were converted to 0, medium cases were converted to 1, and good cases were converted to 2.

6.4.1.1 Modeling

Firstly, the model was set up by accepting the number of neighbors is 10. “knn.reg” function was used. The prediction success of this model for the test set is given in Figure 6.53. It can be said that prediction success is really low. But the first model was set up with the randomly selected number of neighbors. Thus, prediction performance after model tuning is important to evaluate.

```
     RMSE  Rsquared       MAE
1.2187063 0.0400541 0.7697546
```

Figure 6.53 KNN model test error

6.4.1.2 Model Tuning

In this section, prediction performances of all models that their number of neighbors is from 1 to 20 are calculated and compared. Prediction performances of all models are given in Figure 6.54. The best prediction result belongs to the model that is set up with 3 neighbors.

```
k-Nearest Neighbors

1952 samples
  12 predictor

Pre-processing: centered (12), scaled (12)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1757, 1756, 1757, 1756, 1757, 1757, ...
Resampling results across tuning parameters:

  k   RMSE       Rsquared   MAE
   1  1.0974923  0.4580874  0.6554690
   2  0.9894025  0.5407532  0.5800648
   3  0.9833352  0.5274009  0.5651746
   4  1.0355591  0.4892122  0.5687354
   5  1.0131750  0.5028085  0.5563972
   6  1.0296504  0.4941078  0.5597380
   7  1.0232926  0.5020830  0.5534203
   8  1.0289840  0.5048195  0.5535585
   9  1.0513563  0.4984258  0.5572318
  10  1.0552066  0.5055050  0.5524222
  11  1.0583164  0.5012757  0.5540765
  12  1.0600214  0.4992248  0.5522434
  13  1.0614365  0.4983188  0.5510605
  14  1.0524138  0.5002454  0.5497063
  15  1.0464144  0.5042759  0.5487744
  16  1.0337599  0.5137350  0.5460136
  17  1.0249994  0.5188856  0.5457704
  18  1.0202994  0.5210754  0.5427658
  19  1.0118054  0.5260443  0.5405227
  20  1.0109164  0.5253641  0.5400533

RMSE was used to select the optimal model using
 the smallest value.
The final value used for the model was k = 3.
```

Figure 6.54 Prediction performances according to different k values

Figure 6.55 visualizes the relation between RMSE and the number of neighbors. When the number of neighbors is equal to 3, RMSE is in the lowest value.

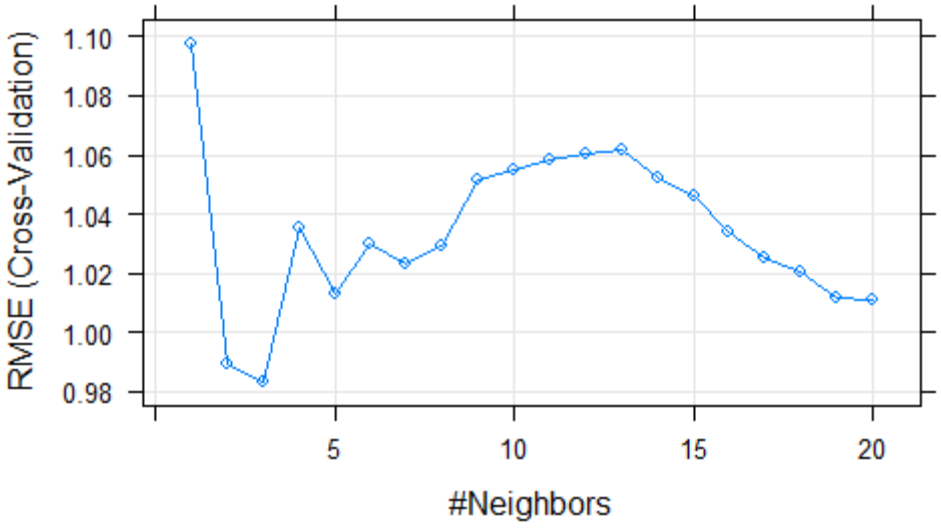


Figure 6.55 The change of the RMSE according to the k values

6.4.1.3 Model Prediction Performance After Model Tuning

```
     RMSE  Rsquared       MAE
1.1147843 0.4428158 0.5859948
```

Figure 6.56 KNN model test error after model tuning

Tuned model prediction performance for the test set is given in Figure 6.56. When the number of neighbors was equal to 10, R-squared was pretty low. After model tuning, the best number of neighbors was determined as 3. As a result that, the R-squared increased to 0.44.

6.4.2 Support Vector Regression

In the data preprocessing of SVR, the same data preprocessing done for KNN method in the previous section was done. Also, the scaling process was done.

6.4.2.1 Modeling

Modeling for the SVR method is done with “svm” function. Prediction errors of the train set are given in Figure 6.57.

```
     RMSE  Rsquared       MAE
0.9456111 0.6540598 0.4695889
```

Figure 6.57 SVR model train error

Prediction errors of the test set are given in Figure 6.58. The R-squared value decreased. Overfitting problem can be exist.

```
     RMSE  Rsquared       MAE
0.8787145 0.4926536 0.5131237
```

Figure 6.58 SVR model test error

6.4.2.2 Model Tuning

Model tuning is done for cost parameter (C) in SVR. Tune length is accepted as 14 while model tuning. In Figure 6.59, prediction errors for 14 different cost parameters are given. The best cost parameter is found as 8.

```
Support Vector Machines with Radial Basis Function Kernel

1952 samples
  12 predictor

Pre-processing: centered (12), scaled (12)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1757, 1756, 1757, 1756, 1757, 1757, ...
Resampling results across tuning parameters:

  C        RMSE       Rsquared   MAE
     0.25  1.2195230  0.3116791  0.5554628
     0.50  1.1623713  0.3876933  0.5495759
     1.00  1.0883219  0.4685900  0.5441153
     2.00  1.0183011  0.5187760  0.5371904
     4.00  1.0043751  0.5260774  0.5428229
     8.00  0.9987812  0.5273912  0.5480000
    16.00  1.0070338  0.5147569  0.5567512
    32.00  1.0261711  0.4935372  0.5716901
    64.00  1.0541182  0.4636512  0.5881170
   128.00  1.0842846  0.4347639  0.6069733
   256.00  1.1233895  0.4017698  0.6283385
   512.00  1.1713737  0.3673597  0.6547577
  1024.00  1.2450895  0.3289381  0.6940621
  2048.00  1.3331641  0.2960703  0.7357043

Tuning parameter 'sigma' was held constant at a value
 of 0.09453079
RMSE was used to select the optimal model using the
 smallest value.
The final values used for the model were sigma =
 0.09453079 and C = 8.
```

Figure 6.59 The prediction performances for different C values

Figure 6.60 visualizes the relation between RMSE and cost parameters.

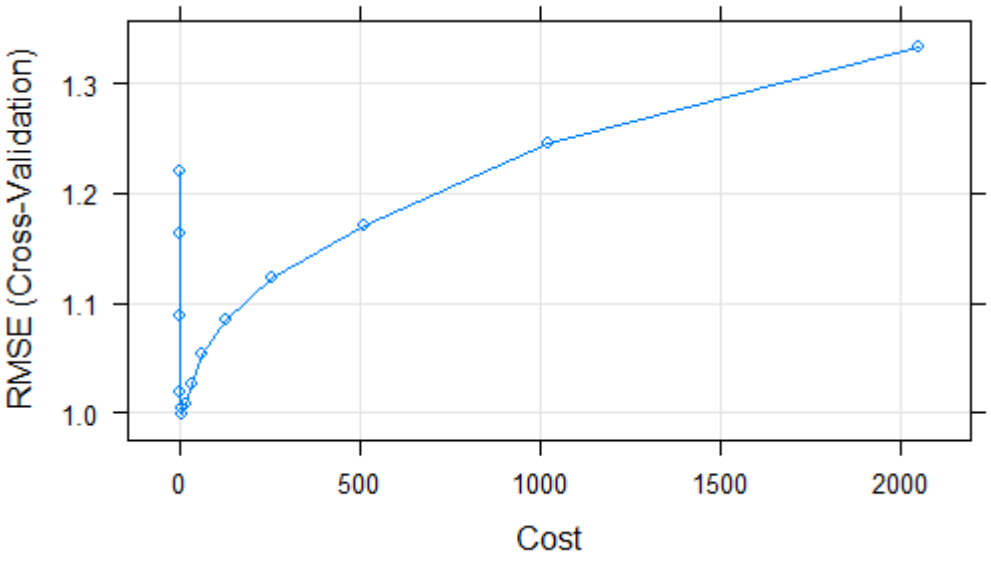


Figure 6.60 The change of RMSE values according to different C values

6.4.2.3 Model Prediction Performance After Model Tuning

Prediction performances when prediction is done with the tuned model for the train set are seen in Figure 6.61. R-squared is equal to 0.84.

```
     RMSE  Rsquared       MAE
0.5831563 0.8417350 0.3832566
```

Figure 6.61 SVR model train error

Prediction performances for the test set are seen in Figure 6.62. R-squared decreased to 0.42. It can be said that there is an overfitting problem.

```
     RMSE  Rsquared       MAE
1.0799213 0.4182440 0.5597518
```

Figure 6.62 SVR model test error

6.4.3 Classification and Regression Trees (CART)

Before the application of CART, "Hospital", "Notification_District", "Process_Status", "Province", "District", "CT", "Neighborhood" are removed from the data set.

6.4.3.1 Modeling

While modeling, “rpart” function is used. In Figure 6.63, the importances of variables are given. It can be seen that the most important variable is “Case_Status” and the next one is “Oxygen”.

Case_Status	Oxygen	Intubation
2292.762896	1675.118900	997.595129
Intensive_care	Age	CT_Result
759.002210	124.382256	119.010600
Ferritin	Sex	Pneumonia
69.676173	22.614956	7.319412

Figure 6.63 CART – Importance of variables

The decision rules resulting from the establishment of the model are given in Figure 6.64.

```
n= 1952

node), split, n, deviance, yval
      * denotes terminal node

1) root 1952 4136.7690  1.0402400
  2) Case_Status=Medium,Good 1929 1366.7120  0.9273978
    4) Case_Status=Good 1588  759.1446  0.7751739
      8) CT_Result=Normal 1309  540.3754  0.6683074 *
      9) CT_Result=Compatible with Bacterial Infection,Compatible with Mix Infection,Compatible
with Viral Pneumonia,COVID 19,Non-Infectious CT Sign 279  133.6810  1.2765650 *
    5) Case_Status=Bad,Medium 341  399.4087  1.6362880 *
  3) Case_Status=Bad 23  685.4525 10.5042300
    6) Age< 64 12  257.7608  8.2777350 *
    7) Age>=64 11  303.3095 12.9331300 *
```

Figure 6.64 CART model output

Figure 6.65 visualizes the tree created.

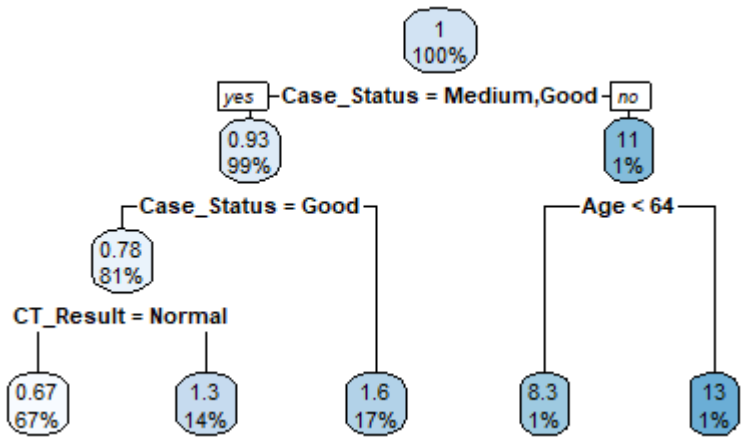


Figure 6.65 Structure of regression tree

When pruning is done, branching did not change. The pruned tree is the same as the tree in Figure 6.65.

6.4.3.2 Prediction Performance

Prediction performances of the tree for the train set are given in Figure 6.66.

RMSE	Rsquared	MAE
0.9150762	0.6048763	0.6160662

Figure 6.66 CART model train error

Prediction performances of the tree for the test set are given in Figure 6.67. R-squared decreased to 0.41.

RMSE	Rsquared	MAE
1.0695453	0.4129919	0.6192376

Figure 6.67 CART model test error

6.4.3.3 Model Tuning

Model tuning was done for complexity parameters. In Figure 6.68, RMSE, R-squared and MAE values are given for 20 models that are established with different complexity parameters (cp). The model that minimizes RMSE is established with the complexity parameter that is 0.0479.

```
CART

2438 samples
  12 predictor

Pre-processing: centered (17), scaled (17)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2194, 2195, 2194, 2194, 2195, 2194, ...
Resampling results across tuning parameters:

  cp           RMSE       Rsquared   MAE
  0.000000000  1.1180515  0.4347905  0.6972424
  0.002083333  1.0451066  0.5061530  0.6353148
  0.004166667  1.0448384  0.5065878  0.6347735
  0.006250000  1.0448384  0.5065878  0.6347735
  0.008333333  1.0448384  0.5065878  0.6347735
  0.010416667  1.0448384  0.5065878  0.6347735
  0.012500000  1.0448384  0.5065878  0.6347735
  0.014583333  1.0457084  0.5066037  0.6353676
  0.016666667  1.0457084  0.5066037  0.6353676
  0.018750000  1.0457084  0.5066037  0.6353676
  0.020833333  1.0646577  0.4853237  0.6712987
  0.022916667  1.0616638  0.4859819  0.6792436
  0.025000000  1.0441722  0.5011981  0.6771409
  0.027083333  1.0445317  0.5019120  0.6794179
  0.029166667  1.0118455  0.5120344  0.6741729
  0.031250000  1.0118455  0.5120344  0.6741729
  0.033333333  1.0118455  0.5120344  0.6741729
  0.035416667  1.0118455  0.5120344  0.6741729
  0.037500000  1.0118455  0.5120344  0.6741729
  0.039583333  1.0118455  0.5120344  0.6741729
  0.041666667  0.9937488  0.5201866  0.6718108
  0.043750000  0.9937488  0.5201866  0.6718108
  0.045833333  0.9937488  0.5201866  0.6718108
  0.047916667  0.9937488  0.5201866  0.6718108
  0.050000000  1.0095464  0.4981310  0.6864880

RMSE was used to select the optimal model
 using the smallest value.
The final value used for the model was cp
 = 0.04791667.
```

Figure 6.68 Prediction performances for different cp values

The change of RMSE according to complexity parameters is given in Figure 6.69. It can be seen that the smallest value of RMSE is between 0.04 and 0.05 nearly.

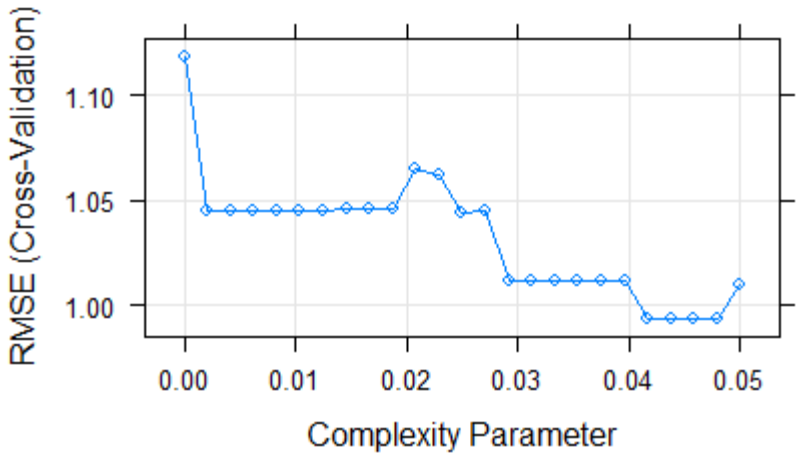


Figure 6.69 The RMSE values according to different cp values

6.4.3.4 Model Prediction Performance After Model Tuning

The tree of the model established with tuned complexity parameter is visualized in Figure 6.70. The branching of the tree was decreased.

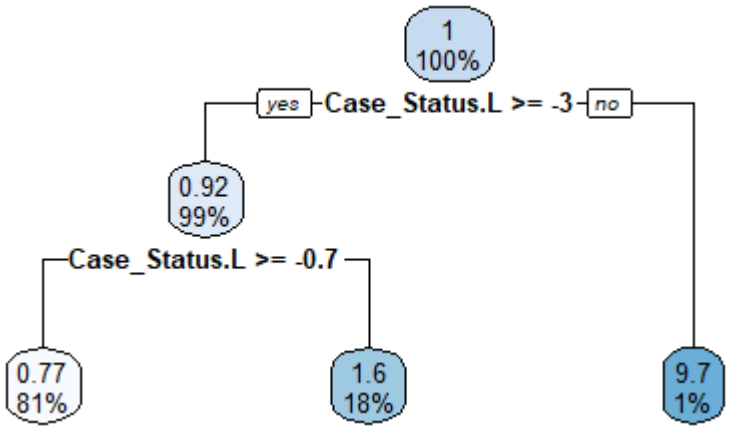


Figure 6.70 The structure of the regression tree after model tuning

Prediction performances of the model for the train set are seen in Figure 6.71.

```
     RMSE  Rsquared       MAE
0.9761253 0.5538946 0.6706258
```

Figure 6.71 CART model train error after model tuning

Figure 6.72 shows the prediction performances of the model for the test set. Prediction success was decreased in the test set prediction, so it can be said that an overfitting problem may exist.

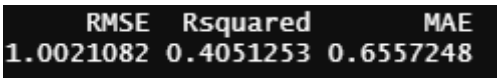


Figure 6.72 CART model test error after model tuning

6.4.4 Bagged Trees Regression

In the first step of the application bagged trees regression, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables were separated from the data set.

6.4.4.1 Modeling

There are 3 different methods to set up bagged trees regression model. The first one is done with “ipredbagg” function. Prediction performances for the train set are given in Figure 6.73 when the first model was set up with train set by using “ipredbagg” function.

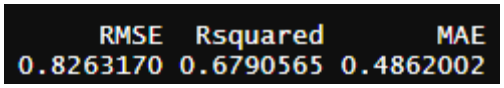


Figure 6.73 Bagged trees regression first model train error

And the prediction errors of the first model for the test set are given in Figure 6.74. Prediction success decreased in the prediction of the test set for the first model.

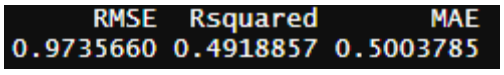


Figure 6.74 Bagged trees regression first model test error

The second method is using the “bagging” function to set up the model. Figure 6.75 shows the prediction errors of the second model for the train set. Prediction errors for the test set prediction are given in Figure 6.76. Prediction success decreased in the prediction of the test set for the second model, like the first model.

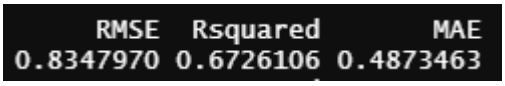


Figure 6.75 Bagged trees regression second model train error

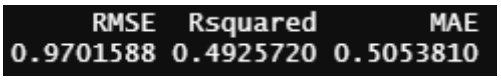


Figure 6.76 Bagged trees regression second model test error

And in the last model, the third model, “randomForest” function is used. In this section, “randomForest” function can be used for bagged trees regression if the “mtry” parameter is accepted as 1 less than the train set column number. “mtry” parameter is the number of variables randomly sampled as candidates at each split [46]. The number of trees is accepted as 2000. In Figure 6.77, importance levels of variables are given according to MSE and Node Purity.

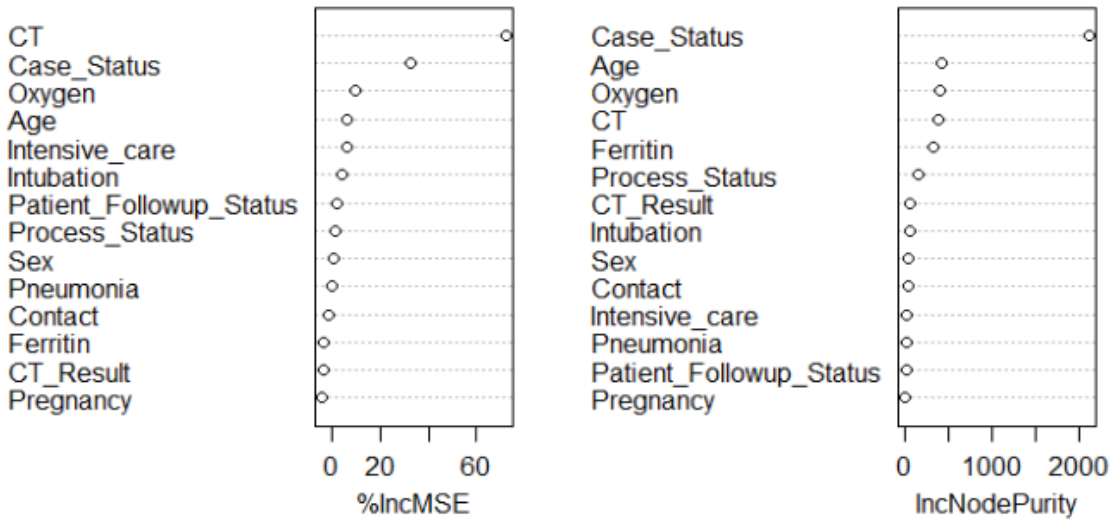


Figure 6.77 Importance levels of variables

For the third model, prediction errors of the train set prediction are given in Figure 6.78.

```
     RMSE  Rsquared       MAE
0.4229402 0.9224821 0.2334674
```

Figure 6.78 Bagged trees regression third model train error

Prediction errors of the test set prediction are seen in Figure 6.79. Prediction of the train set is really successful. However, the prediction of the test set is not as successful as the train set prediction. It can be said that an overfitting problem exists.

```
     RMSE  Rsquared       MAE
1.1024866 0.4404897 0.5463967
```

Figure 6.79 Bagged trees regression third model test error

According to the number of trees, the change of error is given for the third model is given in Figure 6.80. After the number of trees exceeds 100, the error is almost the same.

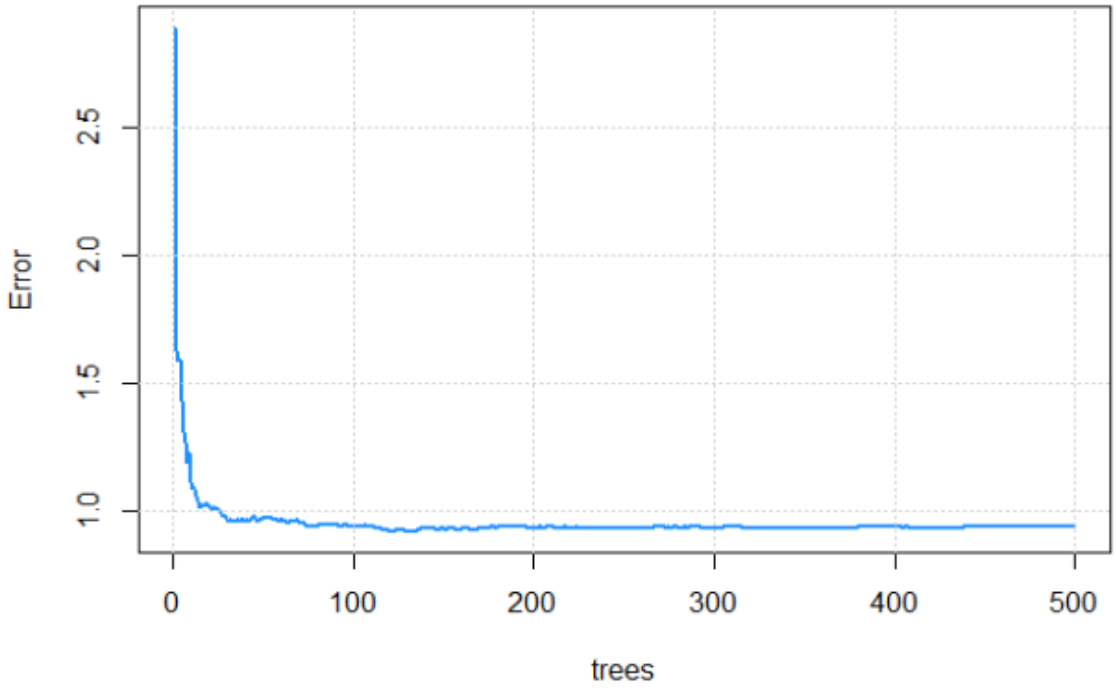


Figure 6.80 The change of the error by number of trees

6.4.4.2 Model Validation

Model tuning was done for the third model. It is aimed to find the optimum "mtry" parameter. The output is given in Figure 6.81.

```
Random Forest

1952 samples
  14 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1756, 1758, 1757, 1756, 1757, 1758,
...
Resampling results:

  RMSE       Rsquared   MAE
  0.9395354  0.5834884  0.5293239

Tuning parameter 'mtry' was held constant at a value of 14
```

Figure 6.81 Validation output

Prediction errors of the model after model validation for the train set are given in Figure 6.82.

```
     RMSE  Rsquared       MAE
0.4191176 0.9232208 0.2327995
```

Figure 6.82 Bagged trees regression third model train error after model tuning

Prediction errors for the test set are seen in Figure 6.83. R-squared of the test set prediction is too smaller than the train set prediction. An overfitting problem may exist. The prediction performance of the third model after model validation is still almost the same as before.

```
     RMSE  Rsquared       MAE
1.1138138 0.4379914 0.5485276
```

Figure 6.83 Bagged trees regression third model test error after model tuning

6.4.5 Random Forest

Before the application, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables were removed from the data set.

6.4.5.1 Modeling

For modeling in the random forest method, “randomForest” function is used. Figure 6.84 shows some information about the model. In this function, the default parameter “mtry” is considered as the number of predictors divided by 3. 500 trees were used and 4 variables were tried each split.

```
Call:
 randomForest(x = train_x, y = train_y, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 4

          Mean of squared residuals: 0.8440214
                    % Var explained: 60.17
```

Figure 6.84 Random forest model output

Importance levels of the variables that are used in the modeling are given in Figure 6.85. The most important variable is “CT” according to MSE. Also, the most important variable is “Case-Status” according to node purity.

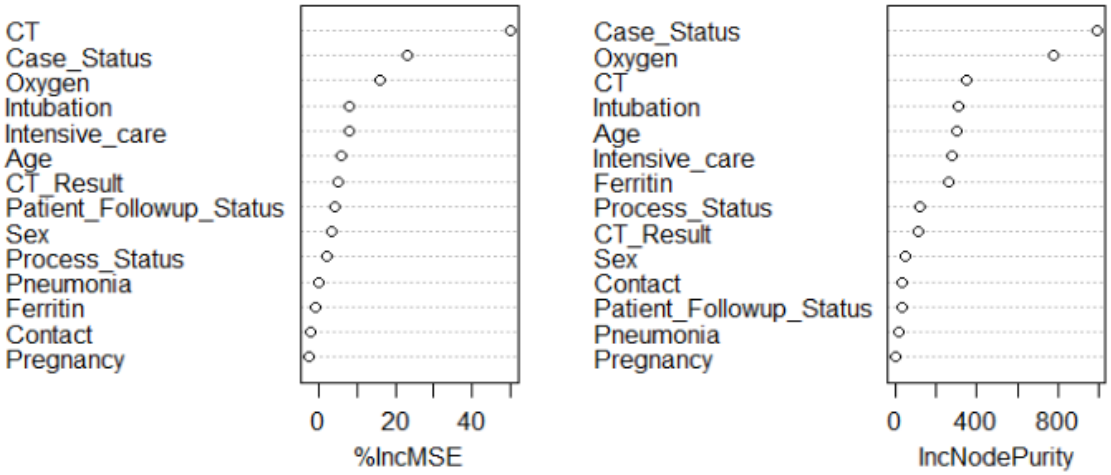


Figure 6.85 Importance levels of variables

6.4.5.2 Model Prediction Performance

The prediction success of the random forest model for the train set is given in Figure 6.86.

```
     RMSE  Rsquared       MAE
0.5357169 0.8774672 0.3246403
```

Figure 6.86 Random forest model train error

The prediction success of the model for the test set is given in Figure 6.87. Errors increased in the test set prediction.

```
     RMSE  Rsquared       MAE
1.0142056 0.4606666 0.5252563
```

Figure 6.87 Random forest model test error

6.4.5.3 Model Tuning

Model tuning is done to find ideal “mtry” and “ntree” parameters in the random forest. In the grid search, the grid contains numbers from 1 to 10 for “mtry” parameter. The best “mtry” parameter is selected between these numbers that are in the grid. “ntree” is the number of trees.

```
Random Forest

1952 samples
  14 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1757, 1756, 1757, 1756, 1757, 1756,
...
Resampling results across tuning parameters:

  mtry  RMSE       Rsquared   MAE
   1    1.0144517  0.5323467  0.6352884
   2    0.9087260  0.5758026  0.5375686
   3    0.8915264  0.5866549  0.5146876
   4    0.8952789  0.5815477  0.5166445
   5    0.8990280  0.5779534  0.5174924
   6    0.9022157  0.5733170  0.5205693
   7    0.9050292  0.5727095  0.5216723
   8    0.9035886  0.5708353  0.5218466
   9    0.9107034  0.5667926  0.5231285
  10    0.9095578  0.5691055  0.5244333

RMSE was used to select the optimal model using
 the smallest value.
The final value used for the model was mtry = 3.
```

Figure 6.88 Prediction performances for different mtry values

Figure 6.88 shows the output of the grid search. There are errors according to different "mtry" parameters. When the parameter was equal to 3, R-squared was maximized and MAE was minimized. So, the optimal "mtry" parameter is found as 3.

To find the best number of trees, values that were tried are 100, 200, 300, 500, 1000 and 2000. Figure 6.89 shows the results according to the number of trees. When 300 trees were used, the best MAE and R-squared were found. The best RMSE was found when 100 trees were used.

```
MAE
          Min.   1st Qu.    Median      Mean   3rd Qu.
100  0.4545943 0.4727968 0.5057762 0.5156757 0.5579588
200  0.4568400 0.4707936 0.5041159 0.5157286 0.5578945
300  0.4568293 0.4690761 0.5035667 0.5145302 0.5550353
500  0.4569119 0.4691092 0.5043522 0.5147591 0.5541921
1000 0.4551179 0.4741247 0.5043437 0.5155984 0.5554246
2000 0.4550685 0.4756465 0.5037654 0.5156739 0.5563066
          Max. NA's
100  0.6132554    0
200  0.6265181    0
300  0.6238746    0
500  0.6254147    0
1000 0.6179378    0
2000 0.6183228    0

RMSE
          Min.   1st Qu.    Median      Mean   3rd Qu.
100  0.6499900 0.7438514 0.8104181 0.8832115 1.0177112
200  0.6533620 0.7451241 0.8182024 0.8786656 0.9826694
300  0.6482986 0.7439992 0.8116198 0.8776715 0.9903211
500  0.6484850 0.7465918 0.8109785 0.8796467 0.9936705
1000 0.6496238 0.7473176 0.8150584 0.8806139 0.9892178
2000 0.6476081 0.7470266 0.8222601 0.8824167 0.9908736
          Max. NA's
100  1.317675    0
200  1.307534    0
300  1.315325    0
500  1.331367    0
1000 1.304289    0
2000 1.306020    0

Rsquared
          Min.   1st Qu.    Median      Mean   3rd Qu.
100  0.3577389 0.5211718 0.6234428 0.5980196 0.7144191
200  0.3799887 0.5295466 0.6406382 0.6079811 0.7089282
300  0.3747468 0.5275346 0.6419440 0.6099011 0.7141420
500  0.3734733 0.5269879 0.6406663 0.6089074 0.7122774
1000 0.3506672 0.5280136 0.6307179 0.6009973 0.7111910
2000 0.3532230 0.5274460 0.6288572 0.6004506 0.7129496
          Max. NA's
100  0.7456948    0
200  0.7650444    0
300  0.7621491    0
500  0.7568083    0
1000 0.7541568    0
2000 0.7563542    0
```

Figure 6.89 Prediction performances for different number of trees

6.4.5.4 Model Prediction Performance After Model Tuning

After model tuning, the new model was set up with optimal parameters which “mtry” parameter is 3 and “ntree” parameter is 300. Model prediction performance for the train set is given in Figure 6.90.

RMSE	Rsquared	MAE
0.6021774	0.8445928	0.3772899

Figure 6.90 Random forest model train error after model tuning

The performance of prediction the test set is given in Figure 6.91. R-squared for test set prediction increased a little bit according to the first model. Test set prediction performance is smaller than train set prediction. So, it can be said that the overfitting problem exists.

RMSE	Rsquared	MAE
0.9816893	0.4740011	0.5202573

Figure 6.91 Random forest model test error after model tuning

6.4.6 Gradient Boosting Machine

6.4.6.1 Modeling

The gradient boosting machine model is set up with “gbm” function. The model was set up by using the train set. While establishing the model, the number of trees was accepted as 5000, interaction depth was accepted as 1, shrinkage was accepted as 0.01, and the number of cross-validation folds was 5. Figure 6.92 shows the importance levels of variables. It is seen that “Case_Status” is the most important variable.

var <chr>	rel.inf <dbl>
Case_Status	48.932643386
Oxygen	25.822223199
Intubation	8.220664998
CT	5.895004673
Intensive_care	5.293049714
Age	4.301624162
Ferritin	0.643437076
Process_Status	0.564025497
CT_Result	0.243638392
Pneumonia	0.053583235
Contact	0.028425808

Figure 6.92 GBM - Importance levels of variables

In Figure 6.93, the change of error according to the number of iterations is given. As the number of iterations increased, the amount of error decreased. After the iteration count rose to just over 1000, the decline of error slowed down.

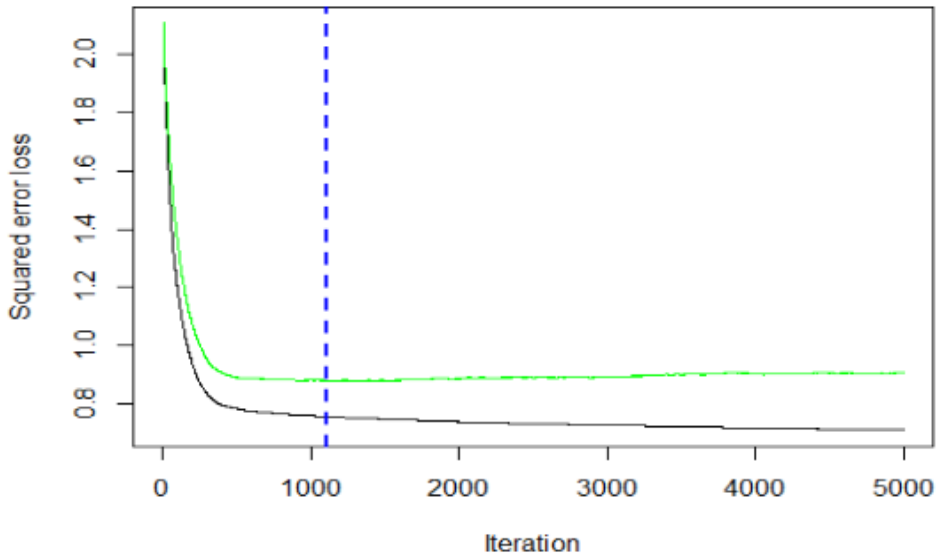


Figure 6.93 The change of squared error loss by number of iterations

6.4.6.2 Model Prediction Performance

The performance of the prediction to the train set is given in Figure 6.94.

```
     RMSE  Rsquared       MAE
0.8438343 0.6642638 0.5154587
```

Figure 6.94 GBM model train error

The performance of the model for the test set is given in Figure 6.95. R-squared is smaller than train error and RMSE and MAE are higher than train error.

```
     RMSE  Rsquared       MAE
0.9932999 0.4779419 0.5339439
```

Figure 6.95 GBM model test error

6.4.6.3 Model Tuning

In the model tuning part of GBM, it was done for the number of trees, interaction depth, shrinkage and minimum terminal node size parameters.

Model tuning was done by increasing the number of trees from 0 to 1000 by 200, the interaction depth parameters by increasing from 1 to 4 by 2, the shrinkage parameters being 0.01 and 0.1, the minimum terminal node size parameters by increasing from 10 to 20 by 1.

Figure 6.96 shows the RMSE values for the different numbers of trees, interaction depth, shrinkage and minimum terminal node size parameters were used. It can be said that when the shrinkage parameter is 0.1, the interaction depth parameter is 1, and minimum terminal node size parameter is 10 and the number of trees parameter (number of iterations) is almost 900, RMSE is at the highest level.

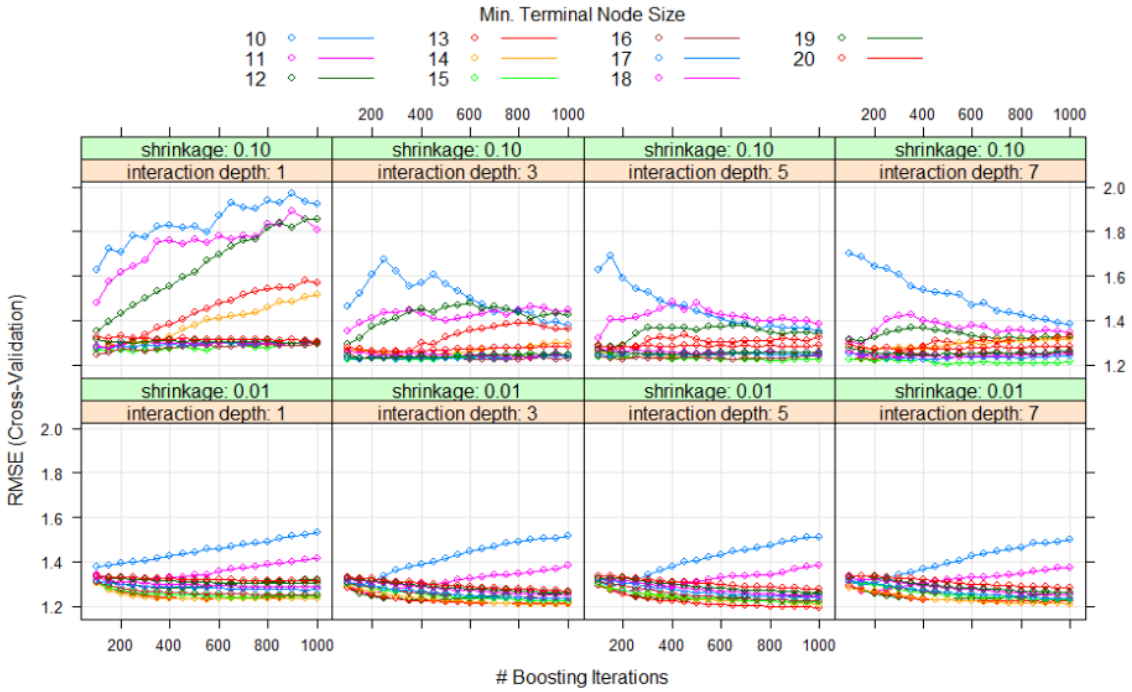


Figure 6.96 RMSe values for different parameters tuned

In Figure 6.97, the best parameters are shown. When the shrinkage parameter is 0.01, the interaction depth parameter is 5, the minimum terminal node size parameter is 13, and the number of trees parameter (number of iterations) is almost 1000, the best result is achieved. Train errors of the model for the train set are given in Figure 6.98. RMSE is 1.21, R-squared is 0.49 and MAE is 0.78 for the train set when these parameters were used.

shrinkage <dbl>	interaction.depth <dbl>	n.minobsinnode <int>	n.trees <dbl>
0.01	5	13	1000

Figure 6.97 The final values of tuned parameters

RMSE	Rsquared	MAE
1.2115204	0.4900635	0.7792932

Figure 6.98 GBM model train error after model tuning

After model tuning, model prediction performance for the test set is given in Figure 6.99. Prediction performance decreased so much. The first model which was set up before model tuning gave better results. In that model, the number of trees parameter was accepted as 5000, tuning was made between 0 and 1000. Because of that, the model tuning results did not have better prediction performance.

RMSE	Rsquared	MAE
1.1097268	0.2154662	0.7783155

Figure 6.99 GBM model test error after model tuning

6.4.7 Artificial Neural Networks

Statistical values of the variables in the data set are given in Figure 6.100. For the Artificial Neural Networks method, firstly the variables were scaled. To estimate the CRP value, the data set is divided into train and test.

variable <chr>	mean <dbl>	std_dev <dbl>	variation_coef <dbl>	p_01 <dbl>	p_05 <dbl>	p_25 <dbl>
Sex	0.5787531	0.49386030	0.85331779	0.00000000	0.00000000	0.0000000
Patient_Followup_Status	0.3297785	0.47022903	1.42589350	0.00000000	0.00000000	0.0000000
Case_Status	1.7994258	0.43015165	0.23904940	0.00000000	1.00000000	2.0000000
Contact	0.6657096	0.47183858	0.70877539	0.00000000	0.00000000	0.0000000
Pregnancy	0.9930271	0.08322949	0.08381392	1.00000000	1.00000000	1.0000000
Intensive_care	0.9840033	0.12548818	0.12752821	0.00000000	1.00000000	1.0000000
Intubation	0.9868745	0.11383562	0.11534965	0.00000000	1.00000000	1.0000000
Age	39.8006563	15.84481988	0.39810449	5.00000000	16.85000000	29.0000000
CT	0.4786710	0.49964735	1.04382197	0.00000000	0.00000000	0.0000000
Pneumonia	0.9667760	0.17925764	0.18541796	0.00000000	1.00000000	1.0000000
Ferritin	209.7873479	125.99878860	0.60060242	8.48243072	44.56878652	104.5292105
CRP	1.0292513	1.41465898	1.37445446	0.01319177	0.06814893	0.2878649
Oxygen	97.7805578	2.03897180	0.02085253	93.00000000	96.00000000	97.0000000

Figure 6.100 Statistic Look of Data

6.4.7.1 Modeling

In order to establish a 1-layer 1-neuron model with the neuralnet function, it was necessary to enter the variables into the function one by one. The resulting network structure is shown in Figure 6.101.

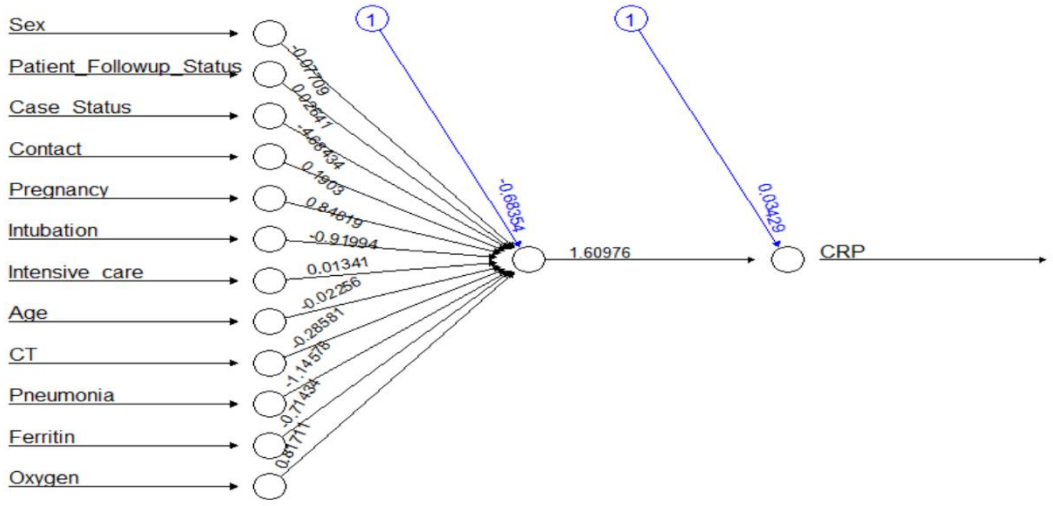


Figure 6.101 First Network Model

Figure 6.102 shows the network structure consisting of two layers with 1 and 3 neurons in each layer, respectively.

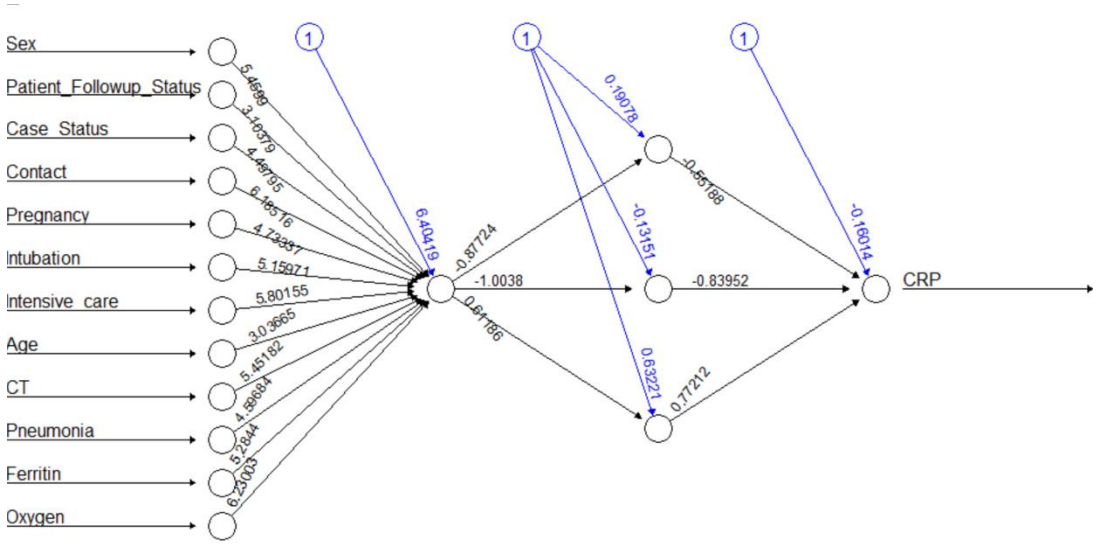


Figure 6.102 Network Model with Two Layers

6.4.7.2 Variable Impact and Significance Levels

The significance levels of the variables are as seen in Figure 6.103. The importance levels of the variables were listed with the waiter function. As can be seen, the most important variable in this data set is the Case_Status variable.

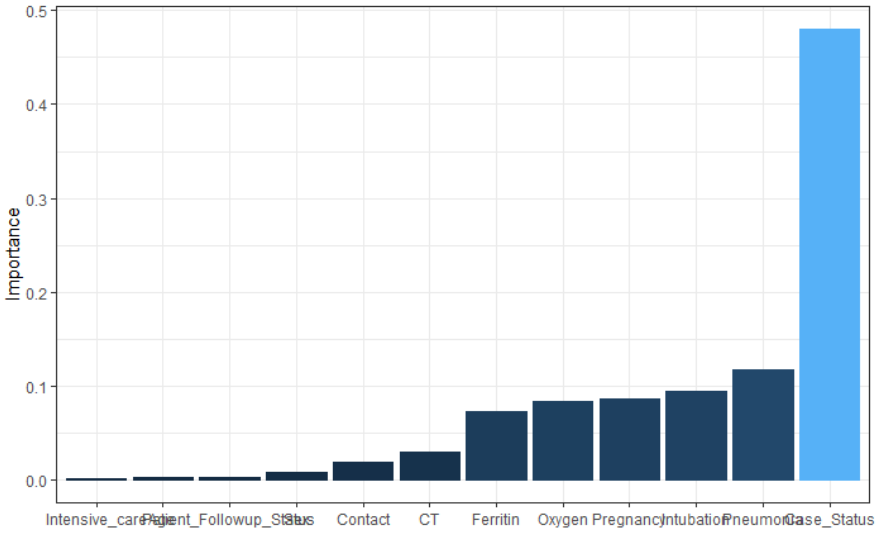


Figure 6.103 Significance Levels of the Variables

Before the estimation step, standardization was undone. The train error is like Figure 6.104. The Rsquared is 0.5816.

```
      RMSE   Rsquared        MAE
0.04564929 0.58156729 0.03242725
```

Figure 6.104 Train Error of ANN

The test error is as Figure 6.105.

```
      RMSE   Rsquared        MAE
0.05161807 0.41849022 0.03226691
```

Figure 6.105 Test Error of ANN

6.4.7.3 Model Tuning

Model tuning was performed with mlpWeightDecay, which represents a multi-layer perceptron. Figure 6.106 shows how the train error changes in response to different layer numbers.

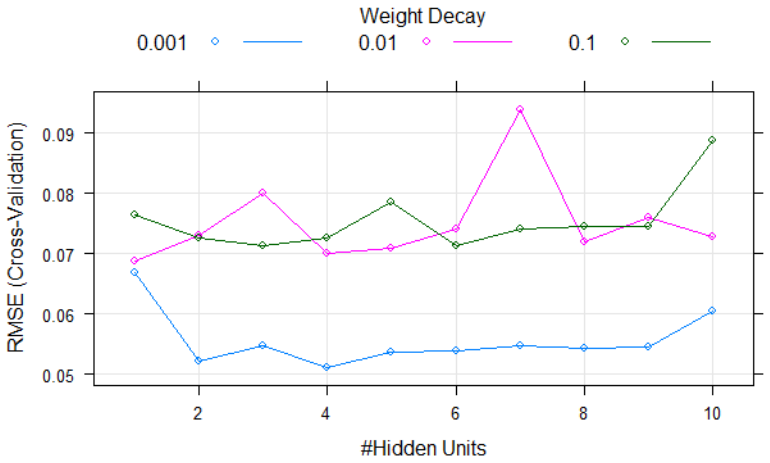


Figure 6.106 Model Tuning Graph of ANN

Estimation was made on the parameters chosen by the model as the best tune. The best tune values are shown in the Figure 6.107 below.

	size <int>	decay <dbl>
4	4	0.001

Figure 6.107 Best Parameters of Tuning ANN

Train Rsquared value of tuned model is as can be seen in Figure 6.108.

RMSE	Rsquared	MAE
0.05059983	0.52912960	0.03046142

Figure 6.108 Train Error of Tuned ANN

The final test error is as in Figure 6.109 below. Rsquared is 0.370919.

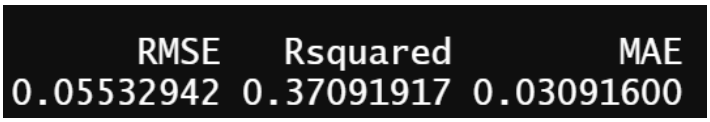


Figure 6.109 Test Error of Tuned ANN

6.4.8 Comparison the Prediction Performances of the Methods

Table 6.2 Comparison of the non-linear regression methods

Non-Linear Regression	Test Set		
	RMSE	Rsquared	MAE
K-Nearest Neighbor Algorithm	1,114784	0,442816	0,585995
Support Vector Regression	0,878715	0,492654	0,513124
Classification and Regression Trees	1,002108	0,405125	0,655725
Bagged Trees Regression	0,970159	0,492572	0,505381
Random Forest	0,981689	0,474001	0,520257
Gradient Boosting Machines	0,9933	0,477942	0,533944
Artificial Neural Networks	0,055329	0,370919	0,030916

Results at the end of all applications are given in Table 6.2. The highest R-squared value is 0.492654 and it belongs to the support vector regression method. This method has also the second lowest RMSE value. The lowest RMSE and MAE are in the artificial neural networks model. However, R-squared of artificial neural networks is the smallest of all methods. Difference of R-squared between support vector regression and bagged trees regression is pretty small, but bagged trees regression has higher RMSE than support vector regression. So, despite there are differences in prediction performances, it can be said that the best result is given by support vector regression.

6.5 Classification

In the classification part, patients' situation is going to predict. The primary goal is to estimate the probability that people may be in bad situation when their test positive for coronavirus.

"Case Status" is a categorical variable that shows a case's status. So, " Case Status" is going to use as dependent categorical variable in the classification methods.

In the first step of all classification methods applied, the data set was separated into a train set and a test set. %80 of the data set is the train set and the rest of the data set is the test set.

6.5.1 Multinomial Logistic Regression

6.5.1.1 Exploratory Data Analysis

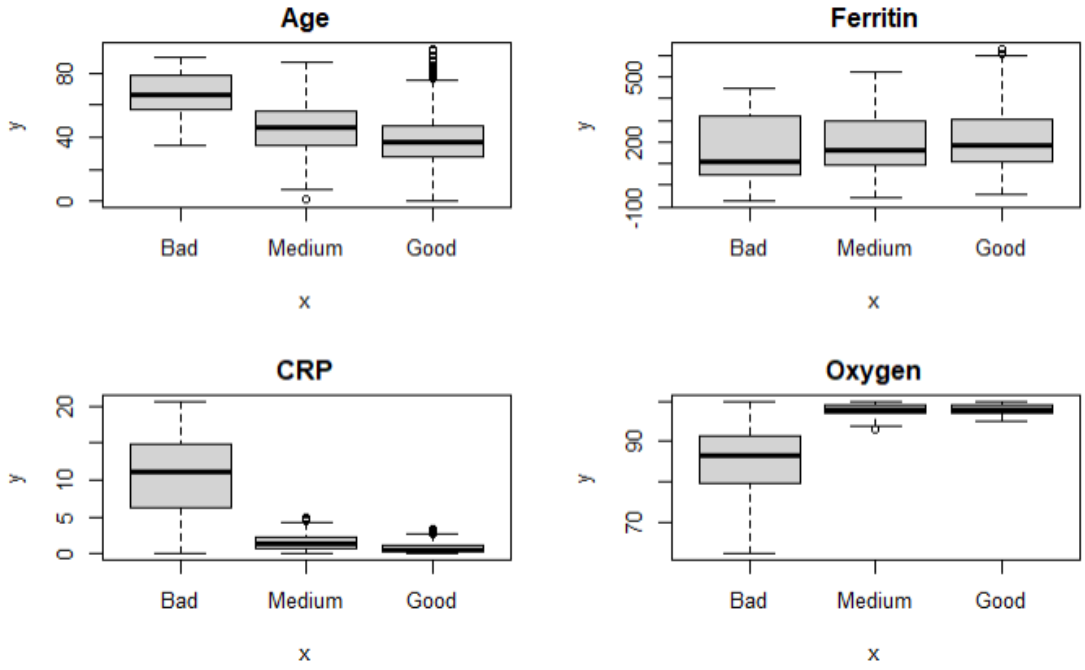


Figure 6.110 Boxplots for Numerical Variables

Boxplots are plots that visualize the minimum, maximum values, median and range of the numerical variable. In Figure 6.110, boxplots are given for Age, Ferritin, CRP and Oxygen variables. By examining the boxplots, the relationships between the numerical variables and the categorical "Case Status" variable were tried to be understood.

For variable “Age”, medians are dissimilar in the different case situations. By looking at the first boxplot, we can say that generally the case situation is bad for old people. At the more young ages, the case situation is medium or good.

For variable “Ferritin”, medians seem close. In bad situations, the median is a little bit low and the range is slightly wide.

The median and range for the “CRP” variable are quite different from the others in the bad case condition.

For variable “Oxygen”, median and range dissimilar in bad case condition from other case conditions.

Thus, we can say that “Age”, “CRP”, “Oxygen” variables change according to case conditions, but “Ferritin” variable does not differ much according to case situations.

Up to this point, analysis has been made for numerical values. Now analysis will be done for categorical variables. The p-value is obtained from the chi-square test was evaluated to analyze the categorical variables. Generally, p-value less than 0.05 for a variable is considered statistically significant. But it is not certain, according to the researcher p-value limit can be changed.

Sometimes, the distribution of variable factors can be unbalanced. That is, the difference between the number of observations of a variable in different factors can be very large. In the situations like this, “Chi-squared approximation may be incorrect” warning appears and p-value can be obtained as NA in the result of Chi-Square Test.

```
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Sex)
X-squared = 1.8094, df = 2, p-value = 0.4047
```

Figure 6.111 Chi-Square Test for Case Status and Sex

When chi-squared test was applied for “Case Status” and “Sex” variables, p-value obtained as 0.4047, as seen in Figure 6.111. It means the “Sex” variable can not be significant.

```
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Patient_Followup_Status)
X-squared = 123.89, df = 2, p-value < 2.2e-16
```

Figure 6.112 Chi-Square Test for Case Status and Patient Follow up Status

When chi-squared test was applied for “Case Status” and “Patient Follow up Status” variables, the test result is given in Figure 6.112. The p-value obtained as 2.2e-16, means “Patient Follow up Status” variable can be significant.

```
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Contact)
X-squared = 3.4913, df = 2, p-value = 0.1745
```

Figure 6.113 Chi-Square Test for Case Status and Contact

When chi-squared test was applied for “Case Status” and “Contact” variables. Chi-squared test results for “Case Status” and “Contact” variables are given in Figure 6.113. The p-value was obtained as 0.1745. It means the “Contact” variable can not be significant.

```
Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Pregnancy)
X-squared = 1.6734, df = 2, p-value = 0.4331
```

Figure 6.114 Chi-Square Test for Case Status and Pregnancy

When chi-squared test was applied for “Case Status” and “Pregnancy” variables, p-value was obtained as 0.4331, as the results are seen in Figure 6.114. It means “Pregnancy” variable can not be significant. But also, “Chi-squared approximation may be incorrect” warning appeared, so we checked the distribution of case numbers according to people are pregnant or not.

	Yes	No
Bad	0	24
Medium	4	340
Good	9	1575

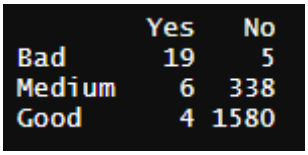
Figure 6.115 Case Status and Pregnancy

In Figure 6.115, we can see the number of cases pregnant is too few. So we can not say anything certain about significance.

```
Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Intensive_care)
X-squared = 1006.1, df = 2, p-value < 2.2e-16
```

Figure 6.116 Chi-Square Test for Case Status and Intensive Care

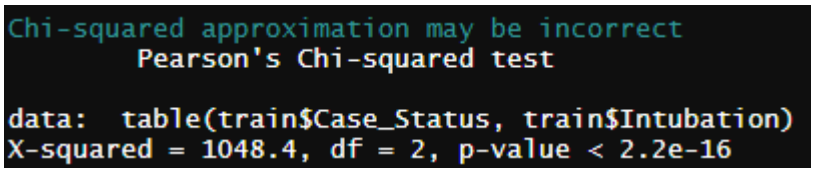
When chi-squared test was applied for “Case Status” and “Intensive Care” variables, the results are given in Figure 6.116, p-value was obtained as 2.2e-16. It means “Pregnancy” variable can be significant. However, the warning appeared again.



```
        Yes   No
Bad      19    5
Medium    6  338
Good      4 1580
```

Figure 6.117 Case Status and Intensive Care

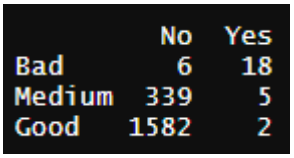
We can observe that the number of cases in which intensive care is “Yes” is pretty fewer than others in Figure 6.117.



```
Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Intubation)
X-squared = 1048.4, df = 2, p-value < 2.2e-16
```

Figure 6.118 Chi-Square Test for Case Status and Intubation

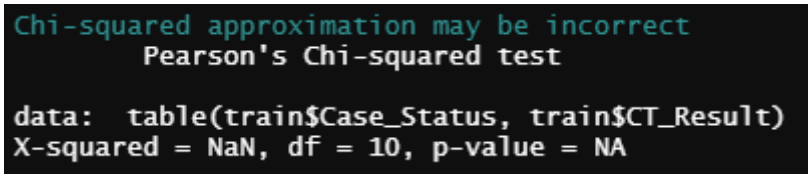
Chi-squared test results are given for “Case Status” and “Intubation” variables in Figure 6.118, the p-value was obtained as 2.2e-16. It means “Intubation” variable can be significant.



```
          No  Yes
Bad        6   18
Medium   339    5
Good    1582    2
```

Figure 6.119 Case Status and Intubation

Because of warning appeared again, the distribution of case numbers was checked. The number of cases with intubation “Yes” is less than the number of cases with intubation “No”, as seen in Figure 6.119.



```
Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$CT_Result)
X-squared = NaN, df = 10, p-value = NA
```

Figure 6.120 Chi-Square Test for Case Status and CT Result

The results of chi-squared test that was applied for “Case Status” and “CT Result” variables are given in Figure 6.120, the p-value was obtained as NA. There are more than two categories.

```
       Compatible with Bacterial Infection
Bad                                      0
Medium                                   2
Good                                     4

       Compatible with Mix Infection
Bad                                    1
Medium                                 0
Good                                   6

       Compatible with Viral Pneumonia COVID 19
Bad                                    2       11
Medium                                19      148
Good                                  54      210

       Non-Infectious CT Sign Normal
Bad                           0     10
Medium                        0    175
Good                          0   1310
```

Figure 6.121 Case Status and CT Result

Because of obtained p-value is NA for “CT Result”, we looked at the distribution table given in Figure 6.121. And we can see that number of cases whose result is “Normal” is quite more than other results.

```
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$CT)
X-squared = 71.605, df = 2, p-value = 2.826e-16
```

Figure 6.122 Chi-Square Test for Case Status and CT

When chi-squared test was applied for “Case Status” and “CT” variables, p-value was obtained as 2.826e-16, as seen in Figure 6.122. It means “CT” variable can be significant.

```
Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Pneumonia)
X-squared = 2.3766, df = 2, p-value = 0.3047
```

Figure 6.123 Chi-Square Test for Case Status and Pneumonia

When chi-squared test was applied for “Case Status” and “Pneumonia”, p-value was obtained as 0.3047, as seen in Figure 6.123. It means “Pneumonia” variable can not be significant.

```
        Yes   No
Bad       2   22
Medium   13  331
Good     49 1535
```

Figure 6.124 Case Status and Pneumonia

The warning appeared for “Case Status” and “Pneumonia” variables. When the table, seen in Figure 6.124, is created for “Case Status” and “Pneumonia” variables we can say there is unbalance between the numbers of cases which “Pneumania” variable is “Yes” and “No”.

```
Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  table(train$Case_Status, train$Process_Status)
X-squared = NaN, df = 14, p-value = NA
```

Figure 6.125 Chi-Square Test for Case Status and Process Status

When chi-squared test was applied for “Case Status” and “Process Status” variables, p-value was obtained as NA. The results are given in Figure 6.125. And there are more than two categories.

```
       Continues in Hospital After Home Monitoring
Bad                                              6
Medium                                         116
Good                                           259

       Continues with Referral to the Hospital
Bad                                            8
Medium                                        46
Good                                          65

       Currently Discharged/Monitoring At Home Continues
Bad                                                    2
Medium                                                94
Good                                                 380

       Death
Bad        0
Medium     0
Good       0

       Discharged with Healing / Monitoring at Home Continues
Bad                                                        0
Medium                                                    19
Good                                                      59

       Healed Home Monitoring Continues
Bad         0                          0
Medium      0                         42
Good        0                        696

       Hospital Monitoring Continues
Bad                                 8
Medium                             27
Good                              125
```

Figure 6.126 Case Status and Process Status

Because of obtained p-value is NA for “Process Status”, we looked at the distribution of cases given in Figure 6.126. We can see that there is unbalance between the numbers of cases among different categories.

6.5.1.2 Modeling

In the modeling part, 4 different models were set up according to significance, and they are compared. To set up the models, “multinom” function is used.

The first model is modelBase that is set up with all variables by using the train set. ModelBase is shown in Figure 6.127.

```
modelBase <- multinom(Case_Status ~ . , data = train)
```

Figure 6.127 The first Multinomial Logistic Regression model

```
Residual Deviance: 1327.458
AIC: 1415.458
```

Figure 6.128 Residual deviance and AIC of the first model

In Figure 6.128, the residual deviance of modelBase is given as 1327.458, and AIC is given as 1415.458.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Medium Good
    Bad      5      0    0
    Medium   0     32   14
    Good     1     53  381

Overall Statistics

               Accuracy : 0.8601
                 95% CI : (0.826, 0.8897)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.003495

                  Kappa : 0.4531

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Medium Class: Good
Precision               1.00000       0.69565      0.8759
Recall                  0.83333       0.37647      0.9646
F1                      0.90909       0.48855      0.9181
Prevalence              0.01235       0.17490      0.8128
Detection Rate          0.01029       0.06584      0.7840
Detection Prevalence    0.01029       0.09465      0.8951
Balanced Accuracy       0.91667       0.67078      0.6856
```

Figure 6.129 Confusion matrix and statistics of the first model for the test set

After prediction according to modelBase, confusion matrix was established and statistics calculated, by using “caret” library for the test set. We can see that accuracy is 0.8601, and the kappa coefficient is 0.4531 in Figure 6.129. The p-value is less than 0.05. It seems a successful prediction. However, we want to predict especially bad case statuses. So, precision, recall, and F1 values are important to evaluate. Precision is 1 for bad case prediction. This means every predicted bad case statuses are really bad cases. Recall is 0.833 for bad case prediction and this means %83.3 of real bad cases statuses are predicted as bad cases. And F1 score is 0.91 for bad case prediction.

The second model is model2 that is set up with numerical variables that are significant according to exploratory data analysis and all categorical variables. Model2 seems in Figure 6.130.

```
model2 <-  multinom(Case_Status ~  Age + CRP + Oxygen + Sex +
Patient_Followup_Status + Contact + Pregnancy +Intensive_care + Intensive_care
+ Intubation + CT_Result + CT + Pneumonia + Process_Status  , data = train)
```

Figure 6.130 The second Multinomial Logistic Regression model

```
Residual Deviance: 1327.468
AIC: 1411.468
```

Figure 6.131 Residual deviance and AIC of the second model

In Figure 6.131, the residual deviance of model2 is given as 1327.468, and AIC is given as 1411.468.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Medium Good
    Bad      5      1    0
    Medium   0     31   14
    Good     1     53  381

Overall Statistics

               Accuracy : 0.858
                 95% CI : (0.8238, 0.8878)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.00505

                  Kappa : 0.4458

 Mcnemar's Test P-Value : 1.783e-05

Statistics by Class:

                     Class: Bad Class: Medium Class: Good
Precision               0.83333       0.68889      0.8759
Recall                  0.83333       0.36471      0.9646
F1                      0.83333       0.47692      0.9181
Prevalence              0.01235       0.17490      0.8128
Detection Rate          0.01029       0.06379      0.7840
Detection Prevalence    0.01235       0.09259      0.8951
Balanced Accuracy       0.91563       0.66490      0.6856
```

Figure 6.132 Confusion matrix and statistics of the second model for the test set

Confusion matrix and statistics of model2 for the test set can be seen in Figure 6.132. Accuracy is 0.858, and the kappa coefficient is 0.4458. The p-value is less than 0.05.

Precision is 0.833 for bad case prediction, so we can say %83.3 of predicted bad case statuses are really bad cases. Recall is 0.833 for bad case prediction. It means %83.3 of real bad cases statuses are predicted as bad cases. According to these results, F1 score is 0.833.

The third model is model3 that is set up with numerical variables that are significant according to the exploratory data analysis and categorical variables that their p-value is less than 0,05 and the warning “Chi-squared approximation may be incorrect” does not appear. Model3 seems in Figure 6.133.

```
model3 <- multinom(Case_Status ~ Age + CRP + Oxygen + Patient_Followup_Status
+ Intensive_care + Intubation  , data = train)
```

Figure 6.133 The third Multinomial Logistic Regression model

```
Residual Deviance: 1449.575
AIC: 1477.575
```

Figure 6.134 Residual deviance and AIC of the third model

In Figure 6.134, the residual deviance of model3 is given as 1449.575, and AIC is given as 1477.575.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Medium Good
    Bad      5      1    0
    Medium   0     23   12
    Good     1     61  383

Overall Statistics

               Accuracy : 0.8457
                 95% CI : (0.8104, 0.8766)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.03344

                  Kappa : 0.3651

 Mcnemar's Test P-Value : 1.285e-07

Statistics by Class:

                     Class: Bad Class: Medium Class: Good
Precision               0.83333       0.65714      0.8607
Recall                  0.83333       0.27059      0.9696
F1                      0.83333       0.38333      0.9119
Prevalence              0.01235       0.17490      0.8128
Detection Rate          0.01029       0.04733      0.7881
Detection Prevalence    0.01235       0.07202      0.9156
Balanced Accuracy       0.91563       0.62033      0.6442
```

Figure 6.135 Confusion matrix and statistics of the third model for the test set

Confusion matrix and statistics of model3 for the test set can be seen in Figure 6.135. Accuracy is 0.8457, and the kappa coefficient is 0.3651. The kappa coefficient is less than 0.4, so it can be interpreted as slightly low. The p-value is less than 0.05 for model3. For bad case prediction, precision is 0.833 and recall is 0.833. These mean that %83.3 of predicted bad cases are really bad cases, and %83.3 of really bad cases are predicted as a bad case. F1 score is 0.833.

And the last model is model4 that is set up with numerical variables that are significant according to the exploratory data analysis and categorical variables that p-value is less than 0.05 and obtained as NA. Model4 seems in Figure 6.135.

```
model4 <- multinom(Case_Status ~ Age + CRP + Oxygen + Patient_Followup_Status
+ Intensive_care + Intubation + CT_Result + Process_Status  , data = train)
```

Figure 6.136 The fourth Multinomial Logistic Regression model

```
Residual Deviance: 1346.692
AIC: 1410.692
```

Figure 6.137 Residual deviance and AIC of the fourth model

In Figure 6.137, the residual deviance of model4 is given as 1346.692, and AIC is given as 1410.692.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Medium Good
    Bad      5      1    0
    Medium   0     25   12
    Good     1     59  383

Overall Statistics

               Accuracy : 0.8498
                 95% CI : (0.8149, 0.8804)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.01889

                  Kappa : 0.3886

 Mcnemar's Test P-Value : 3.049e-07

Statistics by Class:

                     Class: Bad Class: Medium Class: Good
Precision               0.83333       0.67568      0.8646
Recall                  0.83333       0.29412      0.9696
F1                      0.83333       0.40984      0.9141
Prevalence              0.01235       0.17490      0.8128
Detection Rate          0.01029       0.05144      0.7881
Detection Prevalence    0.01235       0.07613      0.9115
Balanced Accuracy       0.91563       0.63210      0.6551
```

Figure 6.138 Confusion matrix and statistics of the fourth model for the test set

Confusion matrix and statistics of model4 for the test set can be seen in Figure 6.138. Accuracy is 0.8498, and the kappa coefficient is 0.3886. The kappa coefficient is slightly low. The p-value is less than 0.05 for model4. Precision, recall value and F1 score are the same with model3.

	F1 Score	Accuracy	Kappa
Model Base	0,90909	0,8601	0,4531
Model 2	0,83333	0,858	0,4458
Model 3	0,83333	0,8457	0,3651
Model 4	0,83333	0,8498	0,3886

Table 6.3 Comparison of Multinomial Logistic Regression models

Table 6.3 summarizes the prediction performance results of the models. When all models are compared, the F1 score of modelBase is more than other models. Also, its accuracy and the kappa coefficient are the highest of all of them. So model tuning will be done for modelBase.

6.5.1.3 Model Tuning

In this step, model tuning is done for the decay parameter. When model tuning is done, the best decay parameter was determined as 0.0001.

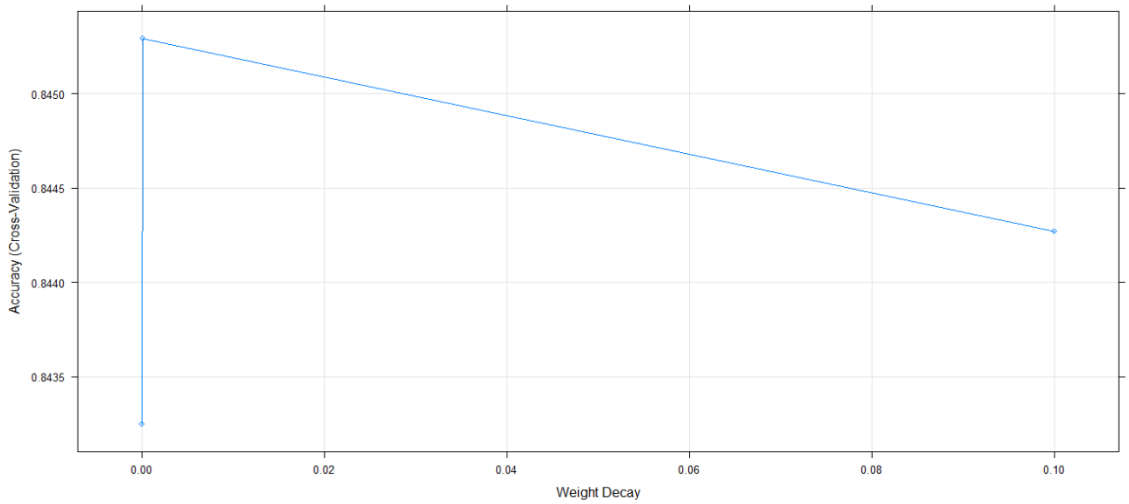


Figure 6.139 The change of accuracy according to decay parameter

We can see the change of accuracy according to the decay parameter in Figure 6.139. The highest accuracy is achieved where the decay parameter is 0.001. Then, accuracy continuous to decrease until the decay parameter is 0.1.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Medium Good
    Bad      5      0    0
    Medium   0     32   14
    Good     1     53  381

Overall Statistics

               Accuracy : 0.8601
                 95% CI : (0.826, 0.8897)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.003495

                  Kappa : 0.4531

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Medium Class: Good
Precision               1.00000       0.69565      0.8759
Recall                  0.83333       0.37647      0.9646
F1                      0.90909       0.48855      0.9181
Prevalence              0.01235       0.17490      0.8128
Detection Rate          0.01029       0.06584      0.7840
Detection Prevalence    0.01029       0.09465      0.8951
Balanced Accuracy       0.91667       0.67078      0.6856
```

Figure 6.140 Confusion matrix and statistics of the model for the test set after model tuning

In Figure 6.140, confusion matrix and statistics of the tuned model are given. Because the best decay parameter is too small, no change occurred.

6.5.2 K-Nearest Neighbor Algorithm (KNN)

Before starting the KNN application, variables, whose number of categories are too high such as "Hospital", "Notification_District", "CT_Result", "Process_Status", "Province", "District", "Neighborhood", were eliminated. And other categorical variables converted to numeric categorical variables. For example; if case status is bad, it is 0, if case status is medium, it is 1, else it is 2. Then train set and test set were separated.

6.5.2.1 Modeling

In the first step, KNN was applied for three different conditions that numbers of the neighbors are equal to 3, 5 and 10 by using “knn” function.

When the number of neighbors is equal to 3, accuracy is 0.7639, precision for bad class is 1, and recall for bad class is 0.125 for the test set. Its confusion matrix is given in Figure 6.141.

	test_y		
knn_fit3	0	1	2
0	1	0	0
1	5	9	30
2	2	78	362

Figure 6.141 Confusion matrix for the test set when k=3

When the number of neighbors is equal to 5, accuracy is 0.7885, and recall for bad class is 0 for the test set. Its confusion matrix is given in Figure 6.142.

	test_y		
knn_fit5	0	1	2
0	0	0	0
1	1	7	15
2	7	80	377

Figure 6.142 Confusion matrix for the test set when k=5

When the number of neighbors is equal to 10, accuracy is 0. 8029, and recall for bad class is 0 for the test set. Its confusion matrix is given in Figure 6.143.

	test_y		
knn_fit10	0	1	2
0	0	0	0
1	0	4	5
2	8	83	387

Figure 6.143 Confusion matrix for the test set when k=10

6.5.2.2 Model Tuning

With model tuning, it is aimed to find the optimum number of neighbors. In the result of model tuning, the optimal number of neighbors is determined as 9. In Figure 6.144, the change of the RMSE of KNN models is given. It can be seen that the least RMSE is in the model with 9 neighbors.

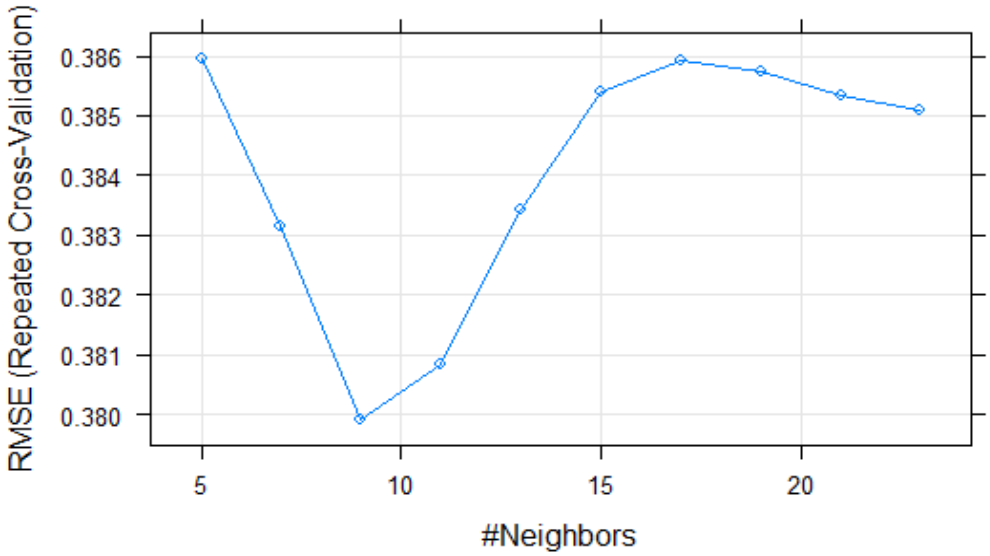


Figure 6.144 The change of RMSE by different k values

Confusion matrix and statistics of the tuned KNN model are given in Figure 6.145 for the test set.

```
Confusion Matrix and Statistics

          test_y
knn_tuned   0   1   2
        0   0   0   0
        1   0   5   5
        2   8  82 387

Overall Statistics

               Accuracy : 0.8049
                 95% CI : (0.7669, 0.8392)
    No Information Rate : 0.8049
    P-Value [Acc > NIR] : 0.5274

                  Kappa : 0.0618

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity           0.00000  0.05747  0.98724
Specificity           1.00000  0.98750  0.05263
Pos Pred Value            NaN  0.50000  0.81132
Neg Pred Value        0.98357  0.82809  0.50000
Prevalence            0.01643  0.17864  0.80493
Detection Rate        0.00000  0.01027  0.79466
Detection Prevalence  0.00000  0.02053  0.97947
Balanced Accuracy     0.50000  0.52249  0.51994
```

Figure 6.145 Confusion matrix and statistics of the model for the test set after model tuning

Accuracy of the KNN model is 0.8049. It is a high rate but only interpreting to accuracy is not enough alone. When we look at p-value and kappa coefficient, p-value is 0.5274, and the kappa coefficient is 0.0618. The p-value is too high and the kappa coefficient is too small. Sensitivity for bad cases is 0, this means model could not predict any real bad cases as bad cases. And specificity for bad cases is 1. Precision can not calculate because the number of predicted bad cases is 0. It can be said that KNN is not an appropriate classification method for our data set.

6.5.3 Classification and Regression Trees (CART)

In the first step of applying the CART method, variables that have too many categories are separated from the data set, such as "Hospital", "Notification_District", "Province", "District", "Neighborhood".

6.5.3.1 Modeling with “tree” Funciton

By using the train set, the tree is created with “tree” function. We can look at the summary of the tree in Figure 6.146. In the tree that is created, there are 11 nodes. Residual mean deviance is 0.5808, and the misclassification error rate is 0.1096. “CRP”, "CT_Result", "CT", "Process_Status", "Oxygen" variables were used in the tree.

```
Number of terminal nodes:  11
Residual mean deviance:  0.5805 = 1127 / 1941
Misclassification error rate: 0.1096 = 214 / 1952
```

Figure 6.146 CART model output

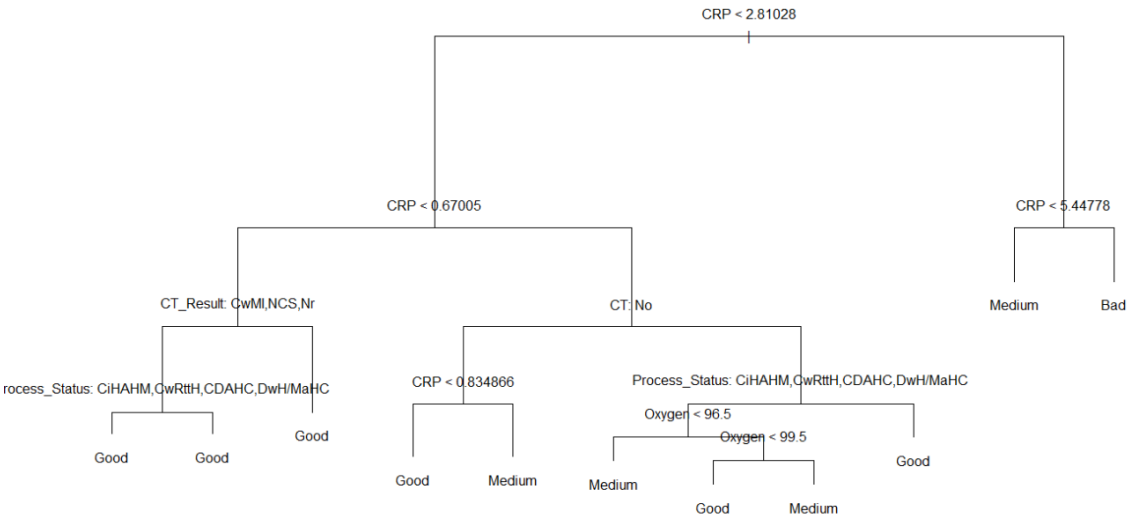


Figure 6.147 The structure of CART tree

In Figure 6.147, the plot of the tree is given.

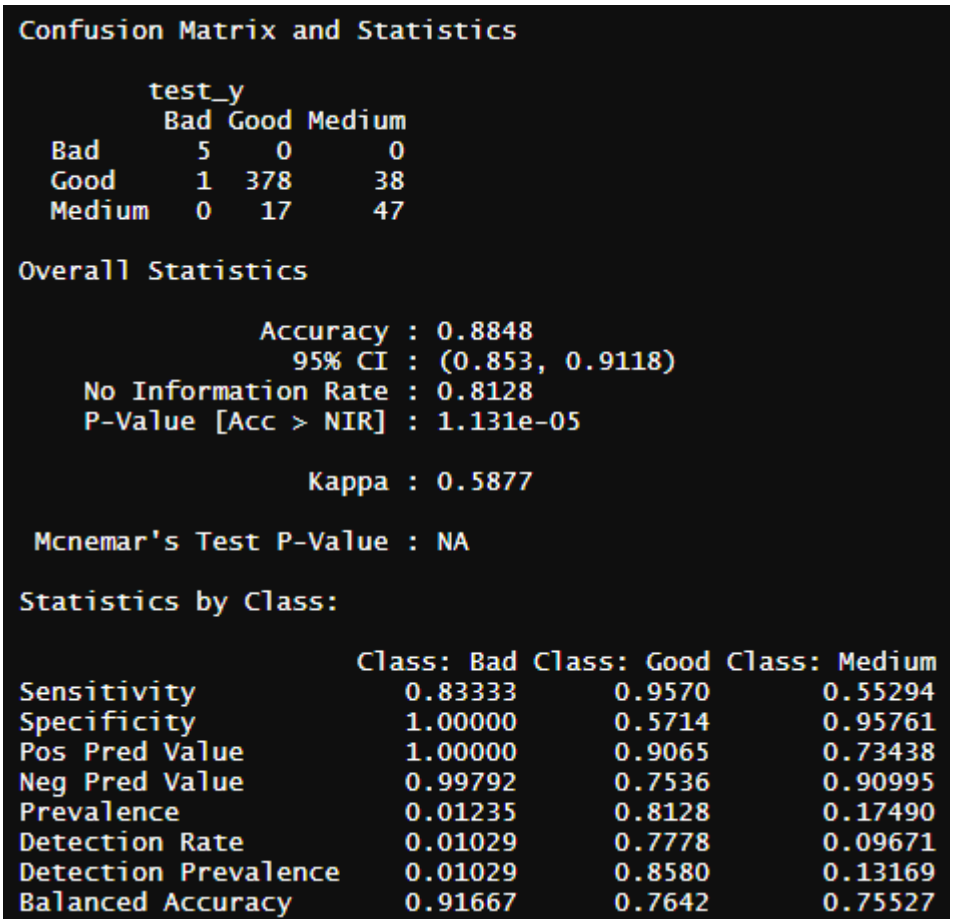
6.5.3.2 Prediction for Model with "tree" Function

Confusion matrix and statistics for the train set are given in Figure 6.148. Accuracy is 0.8904. The kappa coefficient is higher than 0.4 and p-value is less than 0.05. Sensitivity is 0.7917. It means %79.17 of real bad cases were predicted as bad case status. Precision is 1, and F1 score is 0.88374 for bad cases.

```
Confusion Matrix and Statistics

          train_y
            Bad Good Medium
  Bad        19    0      0
  Good        2 1535    160
  Medium      3   49    184

Overall Statistics

                Accuracy : 0.8904
                  95% CI : (0.8757, 0.9039)
    No Information Rate : 0.8115
    P-Value [Acc > NIR] : < 2.2e-16

                   Kappa : 0.5986

 Mcnemar's Test P-Value : 8.404e-14

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity            0.791667      0.9691       0.53488
Specificity            1.000000      0.5598       0.96766
Pos Pred Value         1.000000      0.9045       0.77966
Neg Pred Value         0.997413      0.8078       0.90676
Prevalence             0.012295      0.8115       0.17623
Detection Rate         0.009734      0.7864       0.09426
Detection Prevalence   0.009734      0.8694       0.12090
Balanced Accuracy      0.895833      0.7644       0.75127
```

Figure 6.148 Confusion matrix and statistics of the model for the train set

Confusion matrix and statistics for the test set are given in Figure 6.149. Accuracy of the tree is 0.8848, and the kappa coefficient is 0.5877. The p-value is less than 0.05. Sensitivity for bad class is 0.8333. And precision is 1 for bad cases. F1 score is 0.90907.



```
Confusion Matrix and Statistics

          test_y
           Bad Good Medium
  Bad        5    0      0
  Good       1  378     38
  Medium     0   17     47

Overall Statistics

                Accuracy : 0.8848
                  95% CI : (0.853, 0.9118)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 1.131e-05

                   Kappa : 0.5877

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9570       0.55294
Specificity             1.00000      0.5714       0.95761
Pos Pred Value          1.00000      0.9065       0.73438
Neg Pred Value          0.99792      0.7536       0.90995
Prevalence              0.01235      0.8128       0.17490
Detection Rate          0.01029      0.7778       0.09671
Detection Prevalence    0.01029      0.8580       0.13169
Balanced Accuracy       0.91667      0.7642       0.75527
```

Figure 6.149 Confusion matrix and statistics of the model for the test set

6.5.3.3 Model Tuning with CV for the Model with "tree" Function

In this part, CV is done to find the optimum number of nodes. According to the optimum node number determined, the pruning process will be done.

CV is applied to find the optimum number of nodes, and it is determined as 11. The relation of misclassification rate and the number of nodes is given in Figure 6.150. Misclassification rate decreases, while the number of nodes increasing until the number of nodes is 11.

But when we established the first tree before the CV, it had the same node number. So pruning is unnecessary and results did not change.

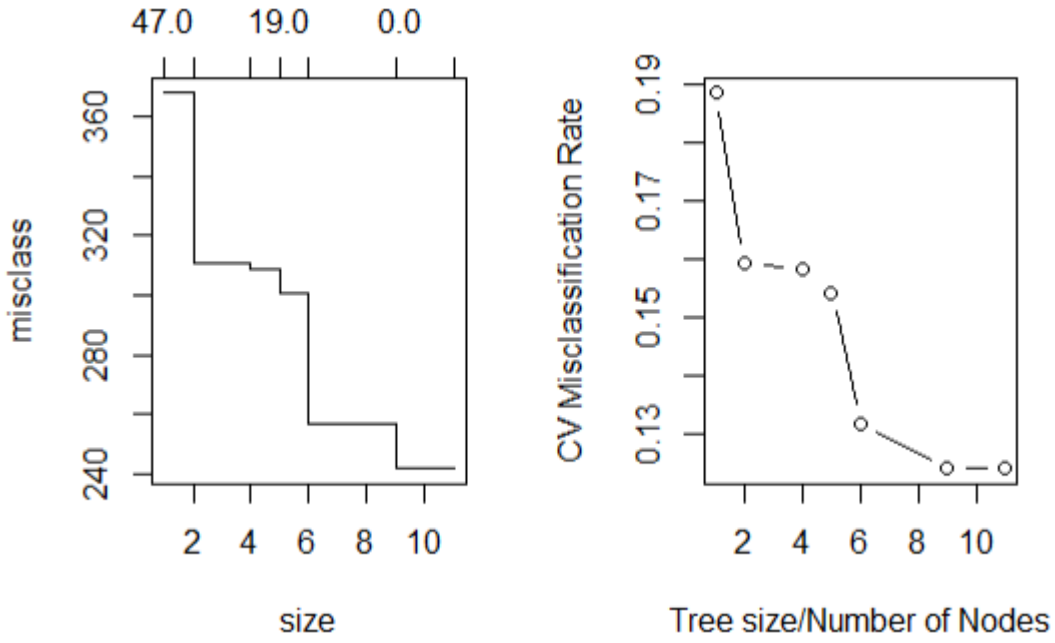


Figure 6.150 The change of misclassification by number of nodes

6.5.3.4 Modeling with "rpart" Function

By using the train set, the tree is created with "rpart" function.

In Figure 6.151, the relationship between complexity parameter and error is given. The minimum complexity parameter is determined as 0.01

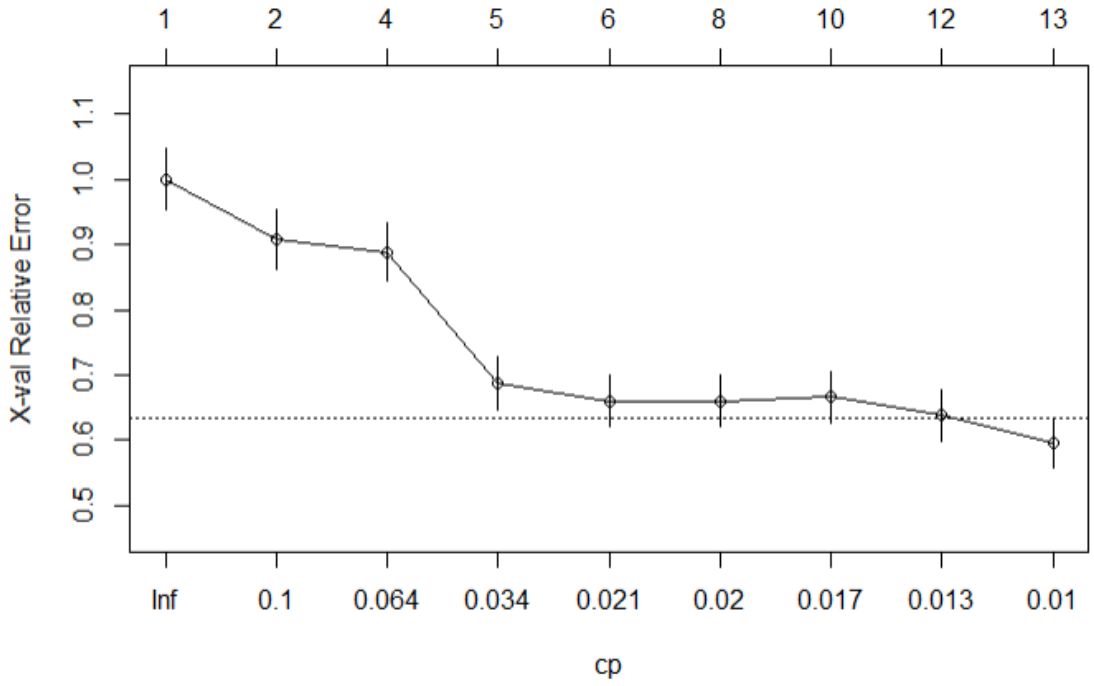


Figure 6.151 The relation of complexity parameter and error

Then, pruning the tree is done with the complexity parameter determined, as seen in Figure 6.152.

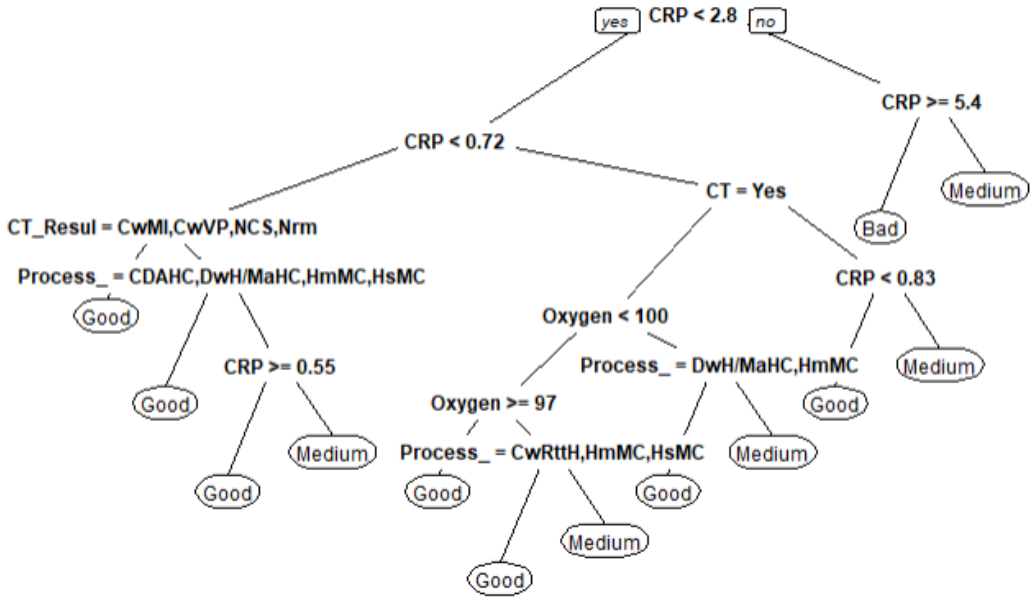


Figure 6.152 The tree after pruning

6.5.3.5 Prediction for Model with "rpart" Function

Confusion matrix and statistics for the train set are given in Figure 6.153. Accuracy of the tree is 0.9016, and the kappa coefficient is 0.6539. Also, the p-value is less than 0.05. Sensitivity for bad cases was calculated as 0.7917. That is, %79.17 of the bad cases were predicted right. Precision is calculated as 1 for bad cases. F1 score is calculated as 0.88374 for bad cases.

```
Confusion Matrix and Statistics

          train_y
            Bad Good Medium
  Bad        19    0      0
  Good        2 1529    132
  Medium      3   55    212

Overall Statistics

               Accuracy : 0.9016
                 95% CI : (0.8876, 0.9145)
    No Information Rate : 0.8115
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6539

 Mcnemar's Test P-Value : 5.31e-08

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity            0.791667      0.9653        0.6163
Specificity            1.000000      0.6359        0.9639
Pos Pred Value         1.000000      0.9194        0.7852
Neg Pred Value         0.997413      0.8097        0.9215
Prevalence             0.012295      0.8115        0.1762
Detection Rate         0.009734      0.7833        0.1086
Detection Prevalence   0.009734      0.8519        0.1383
Balanced Accuracy      0.895833      0.8006        0.7901
```

Figure 6.153 Confusion matrix and statistics of model used “rpart” for the train set

Confusion matrix and statistics for the test set are given in Figure 6.154. Accuracy is 0.8868. The kappa coefficient is 0.6115, and the p-value is less than 0.05. Sensitivity is 0.833 for bad cases. Precision is calculated as 1 for bad cases. F1 score is calculated as 0.90907 for bad cases.

```
Confusion Matrix and Statistics

          test_y
           Bad Good Medium
  Bad        5    0      0
  Good       1  374     33
  Medium     0   21     52

Overall Statistics

                Accuracy : 0.8868
                  95% CI : (0.8553, 0.9136)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 6.228e-06

                   Kappa : 0.6115

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9468        0.6118
Specificity             1.00000      0.6264        0.9476
Pos Pred Value          1.00000      0.9167        0.7123
Neg Pred Value          0.99792      0.7308        0.9201
Prevalence              0.01235      0.8128        0.1749
Detection Rate          0.01029      0.7695        0.1070
Detection Prevalence    0.01029      0.8395        0.1502
Balanced Accuracy       0.91667      0.7866        0.7797
```

Figure 6.154 Confusion matrix and statistics of model used “rpart” for the test set

6.5.3.6 Model Tuning with “caret” Library

In Figure 6.155, the relation of complexity parameter and logLoss is given. The best complexity parameter is determined as 0.0234.

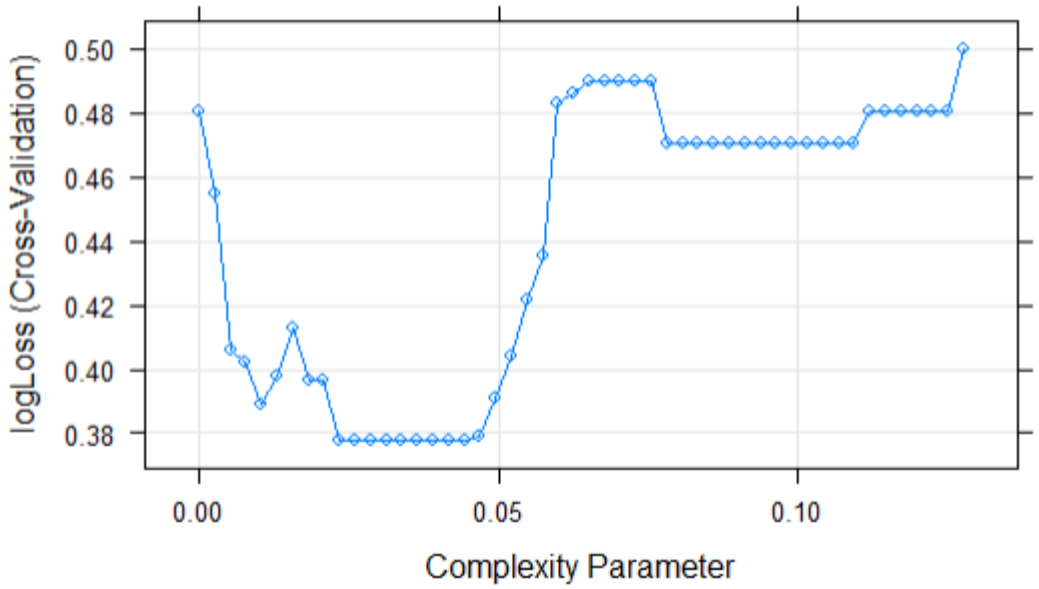


Figure 6.155 The relation of complexity parameter and logLoss

```
Confusion Matrix and Statistics

          train_y
            Bad Good Medium
  Bad        19    0      0
  Good        2 1563    218
  Medium      3   21    126

Overall Statistics

                Accuracy : 0.875
                  95% CI : (0.8595, 0.8894)
    No Information Rate : 0.8115
    P-Value [Acc > NIR] : 3.03e-14

                   Kappa : 0.49

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity            0.791667      0.9867       0.36628
Specificity            1.000000      0.4022       0.98507
Pos Pred Value         1.000000      0.8766       0.84000
Neg Pred Value         0.997413      0.8757       0.87902
Prevalence             0.012295      0.8115       0.17623
Detection Rate         0.009734      0.8007       0.06455
Detection Prevalence   0.009734      0.9134       0.07684
Balanced Accuracy      0.895833      0.6945       0.67568
```

Figure 6.156 Confusion matrix and statistics of model used caret" library for the train set

In Figure 6.156, confusion matrix and statistics of the tuned model for the test set are given. Accuracy is 0.875, and the p-value is less than 0.05. The kappa coefficient is 0.49. Sensitivity is 0.7917, and precision is 1. F1 score is 0.88374 for bad cases.

```
Confusion Matrix and Statistics

          test_y
           Bad Good Medium
  Bad        5    0      0
  Good       1  387     54
  Medium     0    8     31

Overall Statistics

                Accuracy : 0.8704
                  95% CI : (0.8372, 0.8989)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.000435

                   Kappa : 0.4745

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9797       0.36471
Specificity             1.00000      0.3956       0.98005
Pos Pred Value          1.00000      0.8756       0.79487
Neg Pred Value          0.99792      0.8182       0.87919
Prevalence              0.01235      0.8128       0.17490
Detection Rate          0.01029      0.7963       0.06379
Detection Prevalence    0.01029      0.9095       0.08025
Balanced Accuracy       0.91667      0.6877       0.67238
```

Figure 6.157 Confusion matrix and statistics of model used “caret” library for the test set

Figure 6.157 shows the confusion matrix and statistics of the tuned model for the test set. Accuracy is 0.8704, and p-value is less than 0.05. Also, the kappa coefficient is 0.4745. Sensitivity is 0.833 for bad case class. Precision is calculated as 1 for bad cases. And F1 score is 0.90907 for bad cases.

6.5.3.7 CART Application Results

In Table 6.4, the results of CART applications are given. According to Table 6.4, the results of all CART methods are very close. For the test set, the highest accuracy belongs to the model with “rpart” function.

Table 6.4 Results of CART models

CART		
Modelling with tree() function	**Train Set**	**Test Set**
Node number=11		
Accuracy	0,8904	0,8848
Sensitivity	0,7917	0,8333
Precision	1	1
F1	0,8837	0,9091
Modelling with rpart() function	**Train Set**	**Test Set**
Accuracy	0,9016	0,8868
Sensitivity	0,7917	0,8333
Precision	1	1
F1	0,8837	0,9091
After Model Tuning with "caret" Library		
Best cp=0,0234	**Train Set**	**Test Set**
Accuracy	0,875	0,8704
Sensitivity	0,7917	0,8333
Precision	1	1
F1	0,8837	0,9091

6.5.4 Random Forest

Before starting to implement Random Forest, "Hospital", "Notification_District", "Province", "District", "Neighborhood" variables are separated from the data set. Because they have too many categories.

6.5.4.1 Modeling

The model was established with the train set by using “randomForest” function. In this function default “mtry” parameter is accepted as the square root of the number of predictors. The number of variables tried at each split is 3 and the number of trees is 3. In Figure 6.158, the importance levels of variables are given. The most important variable is determined as “CRP”.

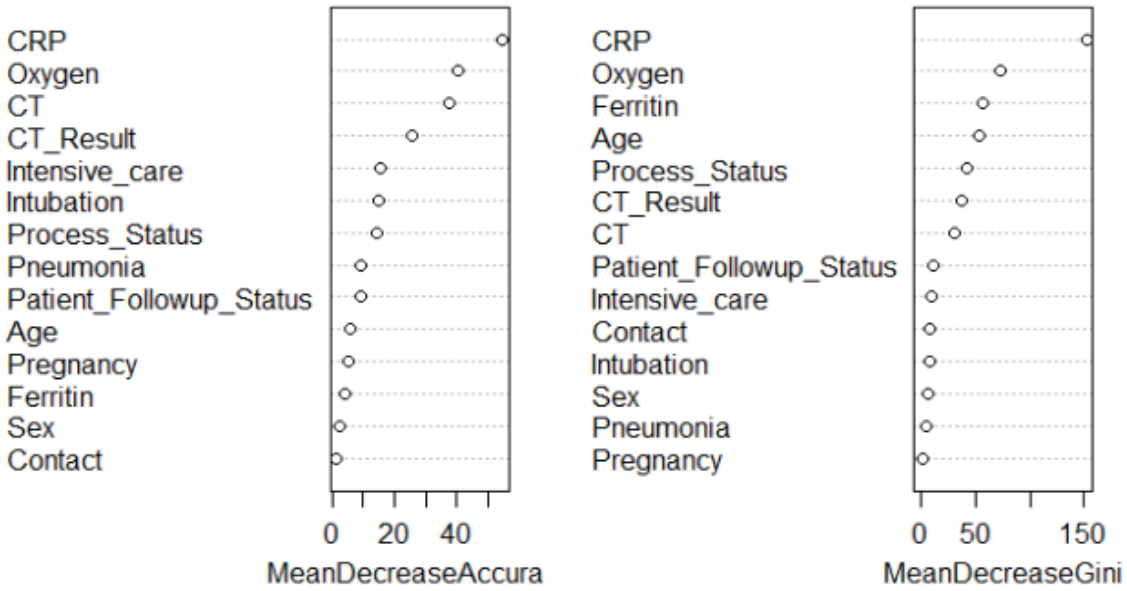


Figure 6.158 RF - Importance levels of variables

6.5.4.2 Prediction

When the prediction is done for the train set, confusion matrix and statistics are as in Figure 6.159. Accuracy is 0.9713 and p-value is less than 0.05. Also, the kappa coefficient is 0.9018 which is really high rate. Sensitivity is 1 and precision is 1 for bad cases. So F1 score is 1.

```
Confusion Matrix and Statistics

          Reference
Prediction  Bad Good Medium
    Bad      24    0      0
    Good      0 1584     56
    Medium    0    0    288

Overall Statistics

               Accuracy : 0.9713
                 95% CI : (0.9629, 0.9783)
    No Information Rate : 0.8115
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9018

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity              1.0000      1.0000        0.8372
Specificity              1.0000      0.8478        1.0000
Pos Pred Value           1.0000      0.9659        1.0000
Neg Pred Value           1.0000      1.0000        0.9663
Prevalence               0.0123      0.8115        0.1762
Detection Rate           0.0123      0.8115        0.1475
Detection Prevalence     0.0123      0.8402        0.1475
Balanced Accuracy        1.0000      0.9239        0.9186
```

Figure 6.159 Confusion matrix and statistics of RF model for the train set

In Figure 6.160, confusion matrix and statistics for the test set are given. Accuracy is 0.9012. The p-value is less than 0.05, and the kappa coefficient is 0.6292. Sensitivity is 0.8333 and precision is 1 for bad cases. In this situation, F1 score is 0.90907.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Good Medium
    Bad      5    0      0
    Good     1  387     39
    Medium   0    8     46

Overall Statistics

               Accuracy : 0.9012
                 95% CI : (0.8712, 0.9263)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 5.4e-08

                  Kappa : 0.6292

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9797       0.54118
Specificity             1.00000      0.5604       0.98005
Pos Pred Value          1.00000      0.9063       0.85185
Neg Pred Value          0.99792      0.8644       0.90972
Prevalence              0.01235      0.8128       0.17490
Detection Rate          0.01029      0.7963       0.09465
Detection Prevalence    0.01029      0.8786       0.11111
Balanced Accuracy       0.91667      0.7701       0.76061
```

Figure 6.160 Confusion matrix and statistics of RF model for the test set

6.5.4.3 Model Tuning with Random Search

In random forest method, it is aimed to find the best “mtry” parameter. In model tuning, the tune length parameter was used as 15. Accuracy rates were calculated for 15 different and random mtry values.

```
Random Forest

1952 samples
  14 predictor
   3 classes: 'Bad', 'Good', 'Medium'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 1756, 1758, 1757, 1756, 1757, 1757, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   1    0.8114790  0.0000000
   2    0.8273635  0.1409884
   7    0.8908964  0.6040864
  10    0.8883349  0.6025545
  11    0.8883296  0.6017033
  14    0.8842217  0.5927541
  18    0.8867806  0.5992866
  19    0.8878010  0.6062812
  21    0.8831934  0.5900775
  22    0.8862625  0.5997890
  23    0.8857576  0.5984574

Accuracy was used to select the optimal model using the
 largest value.
The final value used for the model was mtry = 7.
```

Figure 6.161 The prediction performances of the models tuned with random search by different mtry values

In the result of the model tuning with random search, the mtry value that maximizes accuracy is 7.

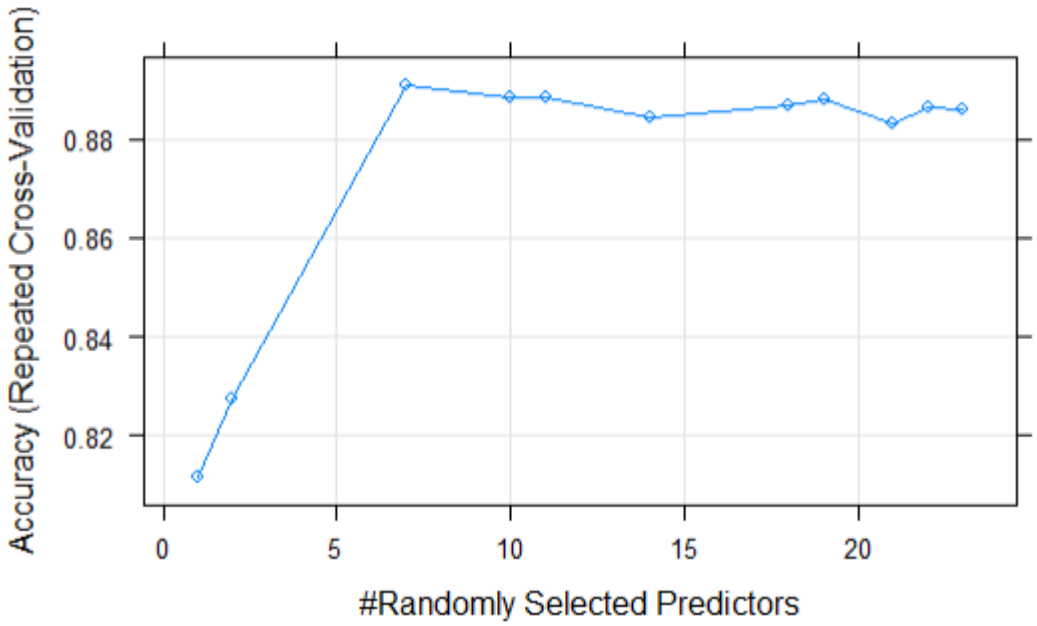


Figure 6.162 The relation between accuracy and mtry values

Figure 6.162 shows the relation between accuracy and mtry values.

```
Confusion Matrix and Statistics

          Reference
Prediction  Bad Good Medium
    Bad      24    0      0
    Good      0 1584     14
    Medium    0    0    330

Overall Statistics

               Accuracy : 0.9928
                 95% CI : (0.988, 0.9961)
    No Information Rate : 0.8115
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9765

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity              1.0000      1.0000        0.9593
Specificity              1.0000      0.9620        1.0000
Pos Pred Value           1.0000      0.9912        1.0000
Neg Pred Value           1.0000      1.0000        0.9914
Prevalence               0.0123      0.8115        0.1762
Detection Rate           0.0123      0.8115        0.1691
Detection Prevalence     0.0123      0.8186        0.1691
Balanced Accuracy        1.0000      0.9810        0.9797
```

Figure 6.163 Confusion matrix and statistics of RF model for the train set after model tuning

When the mtry value is equal to 7, confusion matrix and statistics for the train set were given in Figure 6.163. Accuracy is 0.9928, and p-value is less than 0.05. The kappa coefficient is 0.9765. Sensitivity and precision are 1 for bad cases. So, F1 score is equal to 1.

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Good Medium
    Bad      5    0      0
    Good     1  379     37
    Medium   0   16     48

Overall Statistics

               Accuracy : 0.8889
                 95% CI : (0.8575, 0.9154)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 3.363e-06

                  Kappa : 0.6024

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9595       0.56471
Specificity             1.00000      0.5824       0.96010
Pos Pred Value          1.00000      0.9089       0.75000
Neg Pred Value          0.99792      0.7681       0.91232
Prevalence              0.01235      0.8128       0.17490
Detection Rate          0.01029      0.7798       0.09877
Detection Prevalence    0.01029      0.8580       0.13169
Balanced Accuracy       0.91667      0.7710       0.76240
```

Figure 6.164 Confusion matrix and statistics of RF model for the test set after model tuning

Confusion matrix and statistics for the test set were given in Figure 6.164. Accuracy is 0.8889, and p-value is less than 0.05. The kappa coefficient is 0.6024. For bad cases, sensitivity is 0.8333 and precision is 1, so F1 score is 0.9091.

6.5.4.4 Model Tuning with Grid Search

In the grid search, the grid of mtry values expanded from 1 to 10. For all these mtry values, accuracy rates are calculated. In Figure 6.165, the change of accuracy according to mtry values can be seen. The mtry value that maximizes accuracy is 7 for grid search too, like random search result. Thus, statistics are the same as the result of random search.

```
Random Forest

1952 samples
  14 predictor
   3 classes: 'Bad', 'Good', 'Medium'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 1952, 1952, 1952, 1952, 1952, 1952, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   1    0.8139004  0.0005172937
   2    0.8298891  0.1480750216
   3    0.8516926  0.3480920877
   4    0.8677077  0.4678623411
   5    0.8782127  0.5340397275
   6    0.8826364  0.5639610819
   7    0.8838543  0.5763018483
   8    0.8827683  0.5755002893
   9    0.8831575  0.5791957297
  10    0.8827731  0.5800099426

Accuracy was used to select the optimal model using the
 largest value.
The final value used for the model was mtry = 7.
```

Figure 6.165 The prediction performances of the models tuned with grid search by different mtry values

Figure 6.166 shows the relation of accuracy and mtry values.

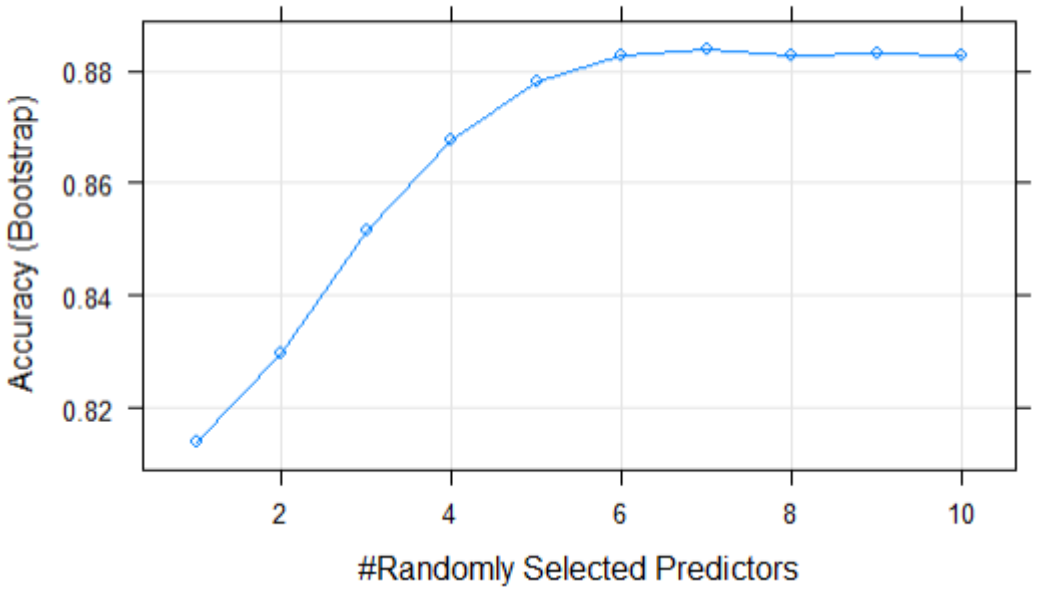


Figure 6.166 The change of accuracy by mtry values

6.5.5 Artificial Neural Network

6.5.5.1 Modeling

The distribution of the classes is as in Figure 6.167 below. It appears to be an unbalanced problem. Here 0 means Bad, 1 means Medium, and 2 means Good.

```
   0    1    2
  30  429 1979

          0          1          2
0.01230517 0.17596390 0.81173093
```

Figure 6.167 Classes of ANN Classification

Since the Artificial Neural Network model will be established, the scaling process was carried out first. Then, the data was split into test and train data with the dependent variable Case_Status. The make.names() function is applied to avoid errors in the Model Tune step.

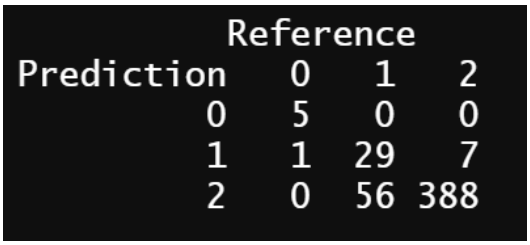
Model outputs are as in Figure 6.168.

```
# weights:  51
initial  value 3095.304435
iter  10 value 1088.075493
iter  20 value 962.312457
iter  30 value 899.836887
iter  40 value 863.587211
iter  50 value 852.410674
iter  60 value 846.103268
iter  70 value 843.896403
iter  80 value 841.889154
iter  90 value 841.628406
iter 100 value 841.591267
final  value 841.591267
stopped after 100 iterations
```

Figure 6.168 Model Outputs

6.5.5.2 Prediciton

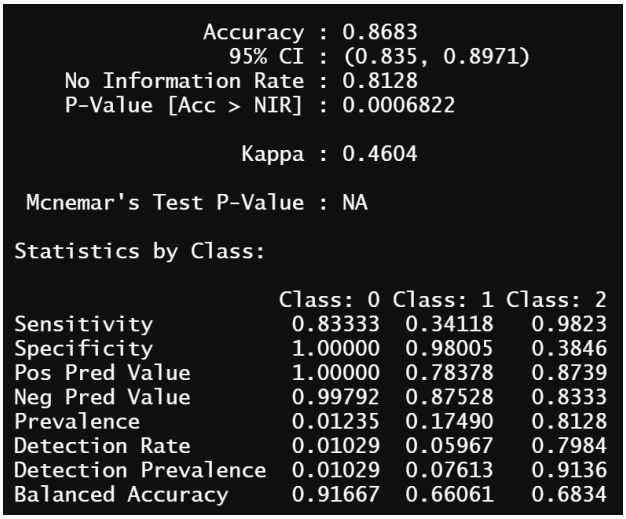
The estimation is as in the Confusion Matrix Figure 6.169. It is seen that 94 observations were estimated incorrectly.



	Reference		
Prediction	0	1	2
0	5	0	0
1	1	29	7
2	0	56	388

Figure 6.169 Confusion Matrix of ANN Classification

Overall statistics are as in Figure 6.170 below.



```
               Accuracy : 0.8683
                 95% CI : (0.835, 0.8971)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.0006822

                  Kappa : 0.4604

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity           0.83333  0.34118   0.9823
Specificity           1.00000  0.98005   0.3846
Pos Pred Value        1.00000  0.78378   0.8739
Neg Pred Value        0.99792  0.87528   0.8333
Prevalence            0.01235  0.17490   0.8128
Detection Rate        0.01029  0.05967   0.7984
Detection Prevalence  0.01029  0.07613   0.9136
Balanced Accuracy     0.91667  0.66061   0.6834
```

Figure 6.170 Overall Statistics of ANN Classification

6.5.5.3 Model Tuning

The graph of the change in the error according to the different layer numbers is as in Figure 6.171.

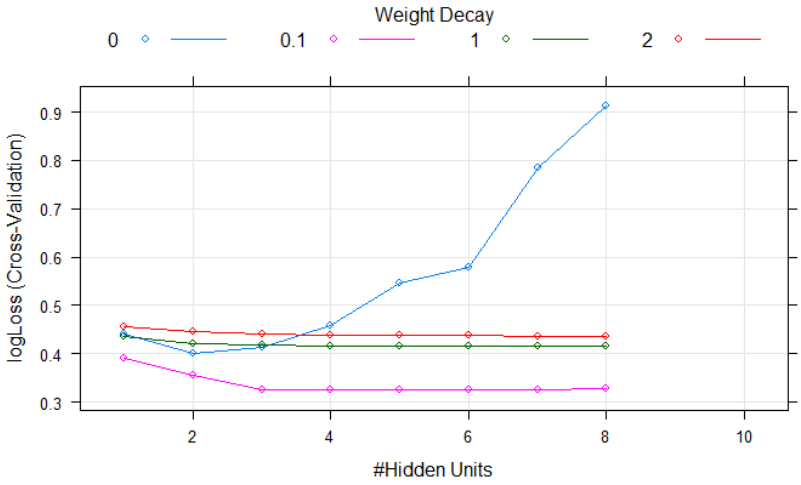


Figure 6.171 Model Tuning Graph of ANN Classification

The best tune parameters extracted from this graph can be accessed from the Figure 6.172 below.

	size <int>	decay <dbl>
14	4	0.1

Figure 6.172 Best Tuning Parameters of ANN Classification

The test error of the tuned model was calculated and the process was completed. Confusion Matrix and overall statistics can be found in the Figure 6.173 below.

```
          Reference
Prediction   0   1   2
         0   4   0   0
         1   2  30   6
         2   0  55 389

Overall Statistics

               Accuracy : 0.8704
                 95% CI : (0.8372, 0.8989)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.000435

                  Kappa : 0.4681

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity           0.66667  0.35294   0.9848
Specificity           1.00000  0.98005   0.3956
Pos Pred Value        1.00000  0.78947   0.8761
Neg Pred Value        0.99585  0.87723   0.8571
Prevalence            0.01235  0.17490   0.8128
Detection Rate        0.00823  0.06173   0.8004
Detection Prevalence  0.00823  0.07819   0.9136
Balanced Accuracy     0.83333  0.66650   0.6902
```

Figure 6.173 Confusion Matrix and Overall Stats of Tuned ANN Classification Model

It is seen that Accuracy and Kappa values improved after Model Tuning.

6.5.6 Support Vector Machines

6.5.6.1 Modeling

The SVM application was started by separating the train and test data. Received as type='C-classification' when building the model. When kernel='linear' in Figure 6.174 and kernel='radial' in Figure 6.175, the outputs and support vectors are seen.

```
Call:
svm(formula = Case_Status ~ ., data = train, type = "C-classification",
    kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  686
```

Figure 6.174 Linear SVM Model

```
Call:
svm(formula = Case_Status ~ ., data = train, type = "C-classification",
    kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  706
```

Figure 6.175 Radial SVM Model

6.5.6.2 Prediction

After the estimation, confusion matrices in Figure 6.176 and Figure 6.177 were formed for the linear and radial method.

	y_pred1		
test_y	Bad	Good	Medium
Bad	5	1	0
Good	0	393	2
Medium	0	74	11

Figure 6.176 Linear Confusion Matrix

	y_pred2		
test_y	Bad	Good	Medium
Bad	5	1	0
Good	0	392	3
Medium	0	64	21

Figure 6.177 Radial Confusion Matris

6.5.6.3 Visualization of Confusion Matrices and Error

Confusion matrices can be seen visually in Figure 6.178 and Figure 6.179.

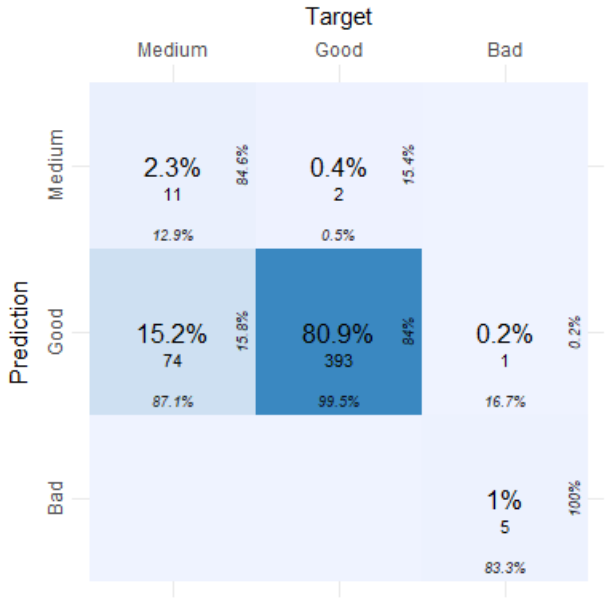


Figure 6.178 Colored Linear Confusion Matrix

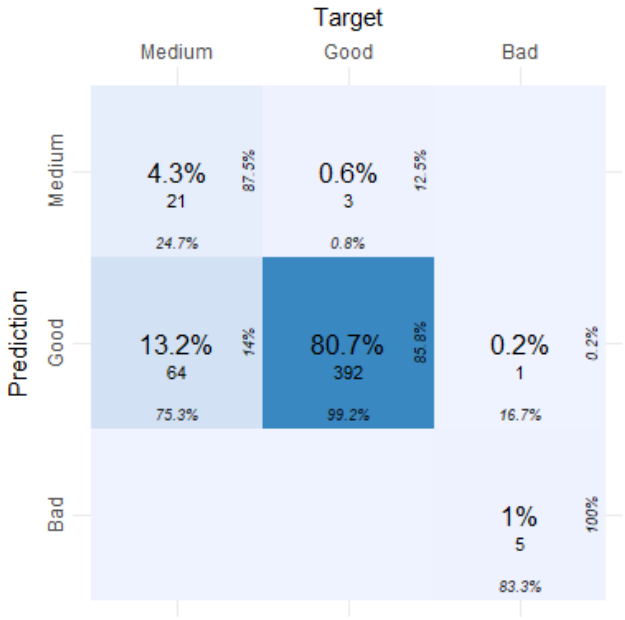


Figure 6.179 Colored Radial Confusion Matrix

Overall statistics and test errors for both methods can be seen in Figure 6.180 and Figure 6.181. According to the results, it can be said that the linear and radial methods are quite close.

```
          Reference
Prediction Bad Good Medium
    Bad      5    0      0
    Good     1  393     74
    Medium   0    2     11

Overall Statistics

               Accuracy : 0.8416
                 95% CI : (0.806, 0.8729)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.05601

                  Kappa : 0.2546

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9949       0.12941
Specificity             1.00000      0.1758       0.99501
Pos Pred Value          1.00000      0.8397       0.84615
Neg Pred Value          0.99792      0.8889       0.84355
Prevalence              0.01235      0.8128       0.17490
Detection Rate          0.01029      0.8086       0.02263
Detection Prevalence    0.01029      0.9630       0.02675
Balanced Accuracy       0.91667      0.5854       0.56221
```

Figure 6.180 Overall Stats of Linear SVM

For radial, sensitivity is 0.833 and recall is 1 for bad class. So, F1 Score is 0.9091 for bad class. The kappa coefficient is 0.3836, and the accuracy 0.8601.

```
          Reference
Prediction Bad Good Medium
    Bad      5    0      0
    Good     1  392     64
    Medium   0    3     21

Overall Statistics

               Accuracy : 0.8601
                 95% CI : (0.826, 0.8897)
    No Information Rate : 0.8128
    P-Value [Acc > NIR] : 0.003495

                  Kappa : 0.3836

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Bad Class: Good Class: Medium
Sensitivity             0.83333      0.9924       0.24706
Specificity             1.00000      0.2857       0.99252
Pos Pred Value          1.00000      0.8578       0.87500
Neg Pred Value          0.99792      0.8966       0.86147
Prevalence              0.01235      0.8128       0.17490
Detection Rate          0.01029      0.8066       0.04321
Detection Prevalence    0.01029      0.9403       0.04938
Balanced Accuracy       0.91667      0.6391       0.61979
```

Figure 6.181 Overall Stats of Radial SVM

6.5.7 Comparison the Prediction Performances of the Methods

Table 6.5 Comparison of the classification methods

Classification	Test Set		
	F1 Score	Accuracy	Kappa
Multinomial Logistic Regression	0,9091	0,8601	0,4531
K-Nearest Neighbor Algorithm	NaN	0,8049	0,0618
Classification and Regression Trees	0,9091	0,8704	0,4745
Random Forest	0,9091	0,9012	0,6292
Artificial Neural Network	0,8031	0,8704	0,4681
SVM	0.9091	0.8601	0.3836

Table 6.5 gives the prediction performances of classification methods. The aim of classification is prediction of the bad cases. So, F1 score is given for bad class in Table 6.5. F1 score is equal for every model except k-nearest neighbor (KNN) and support vector machine methods (SVM). In KNN, it could not be calculated because precision could not calculated and recall was 0. Also, accuracy and the kappa coefficient of KNN are the lowests according to applied methods. So, it can be said that KNN is the worse method than others for the data set.

The highest accuracy and the kappa coefficient belong to random forest method. Thus, the best method is random forest for our data set.

6.6 Clustering

6.6.1 K-Means

6.6.1.1 Modeling

The data that is ready to be used is first scaled with the *scale* function like Figure 6.182. Because the standard deviations/variances in the initial state of the data are very different from each other. Clustering is done according to the distance of the observations from each other.

Sex <dbl>	Patient_Followup_Status <dbl>	Case_Status <dbl>	Contact <dbl>	Pregnancy <dbl>
-1.1718963	-0.7013147	0.4662873	0.7084847	0.08377954
0.8529678	-0.7013147	0.4662873	0.7084847	0.08377954
-1.1718963	-0.7013147	0.4662873	-1.4108842	0.08377954
0.8529678	1.4253086	-1.8584742	-1.4108842	0.08377954
-1.1718963	-0.7013147	0.4662873	-1.4108842	0.08377954
0.8529678	-0.7013147	0.4662873	0.7084847	0.08377954
-1.1718963	-0.7013147	0.4662873	0.7084847	0.08377954
-1.1718963	-0.7013147	0.4662873	0.7084847	0.08377954
-1.1718963	-0.7013147	0.4662873	-1.4108842	0.08377954
0.8529678	-0.7013147	0.4662873	0.7084847	0.08377954

Figure 6.182 Scaled Data

The distance matrix is calculated. The kmeans function is run by assigning multiple starting points. Output is like Figure 6.183.

```
List of 9
 $ cluster     : int [1:2438] 1 1 1 1 1 1 1 1 1 1 ...
 $ centers     : num [1:2, 1:13] -0.00254 0.16025 -0.02168 1.36934 0.05461 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:13] "Sex" "Patient_Followup_Status" "Case_Status" "Contact" ...
 $ totss       : num 31681
 $ withinss    : num [1:2] 23000 2403
 $ tot.withinss: num 25403
 $ betweenss   : num 6278
 $ size        : int [1:2] 2400 38
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

Figure 6.183 First K-Means Outputs

The descriptions of the values in the output are as follows:

• cluster: vector expressing clusters from 1 to k

• centers: matrix expressing the center of clusters

• totts: the sum of the square sums

• withinss: intra-set sum of squares

• top.withinss: the sum of squares for the whole set

• betweens: the sum of squares between clusters

• size: number of observations for each cluster

When k2 is run, the output in Figure 6.184 is obtained. As can be seen, the difference between the clusters is large, but considering that the cluster with few observations represents a bad result, COVID cases can be accepted for this study.

```
K-means clustering with 2 clusters of sizes 2400, 38

Cluster means:
          Sex Patient_Followup_Status Case_Status     Contact    Pregnancy
1 -0.002537309             -0.02168129  0.05461076 -0.00238697 -0.001326509
2  0.160251109              1.36934486 -3.44910050  0.15075602  0.083779542
  Intensive_care Intubation         Age           CT    Pneumonia     Ferritin
1      0.1108741  0.1153023 -0.02326142  0.005995478  0.001714222  0.004961827
2     -7.0025730 -7.2822526  1.46914205 -0.378661786 -0.108266651 -0.313378562
         CRP      Oxygen
1 -0.07790278  0.07104504
2  4.92017569 -4.48705513
```

Figure 6.184 K-Means Clustering with Two Clusters

6.6.1.2 Visualization of Clusters

Clusters obtained using the *fviz_cluster* function are visualized as in Figure 6.185.

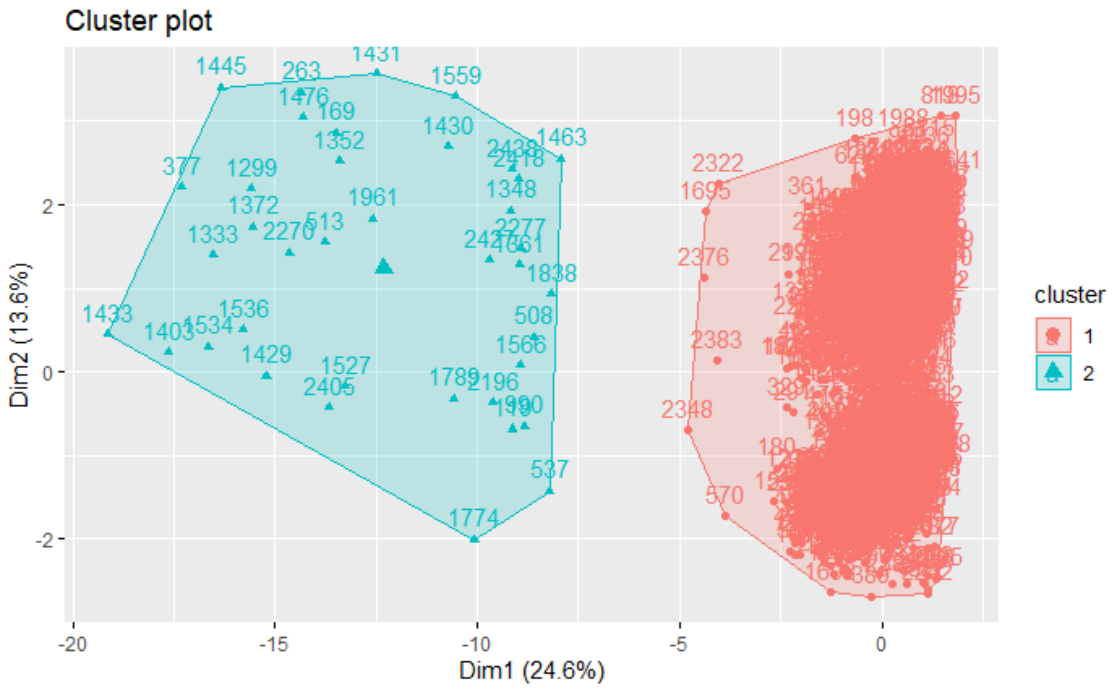


Figure 6.185 Visualization of Two Clusters

It appears that the values at the boundary can be clustered differently using another distance metric. Also, the function we use reduces n variables to two, which explains the

vast majority of their variances. The "dimension reduction" method is used on the reverse side.

Figure 6.186 is formed as a result of visualizing the clusters with Scatter Plot based on two variables (Oxygen and Age).

Figure 6.186 Scatter Plot Clustering with Age and Oxygen Variables

6.6.1.3 Clusters by Different K Values

One of the disadvantages of K-Means is that the number of clusters is given manually. Different clusters are shown in Figure 6.187. This situation reduces reliability.

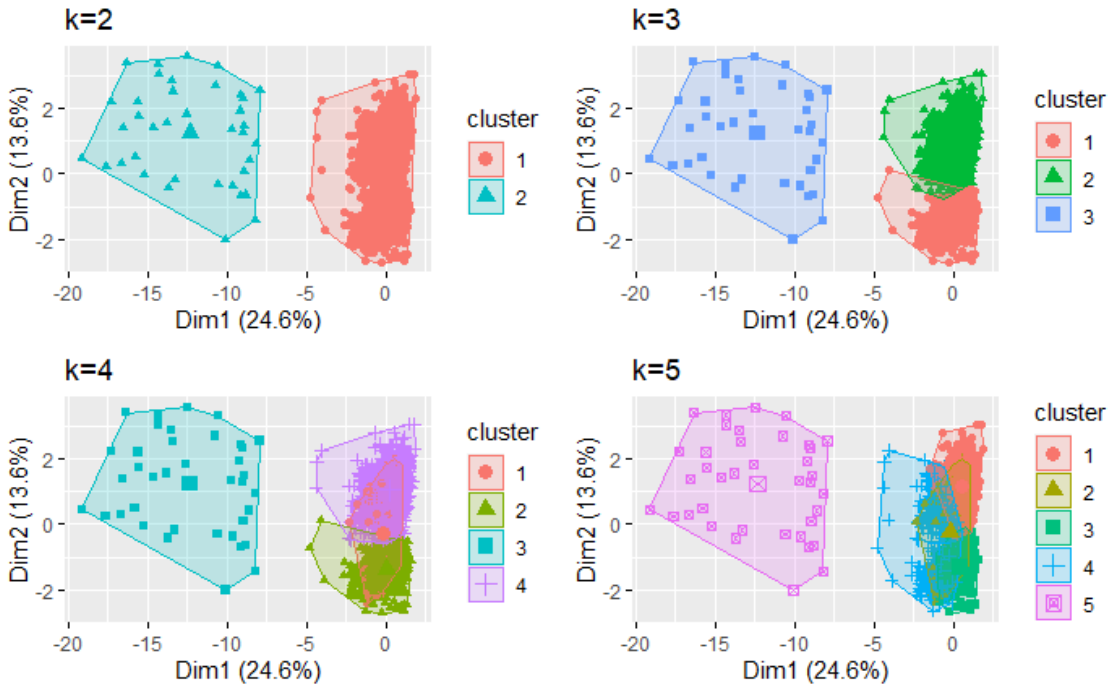


Figure 6.187 Clusters with Different k values

6.6.1.4 Determining the Optimum Number of Clusters

The algorithms used while finding the optimum number of clusters try to keep the within-cluster error minimum and the inter-cluster error maximum. The optimum number of clusters can be decided by using more than one algorithm and considering the domain information. The sum of the in-cluster errors is used.

For the Elbow method, firstly, the central k value was determined. Errors are shown on the graph and 3, the point where the slope changes net, was determined as the optimum number of clusters, as seen in Figure 6.188.

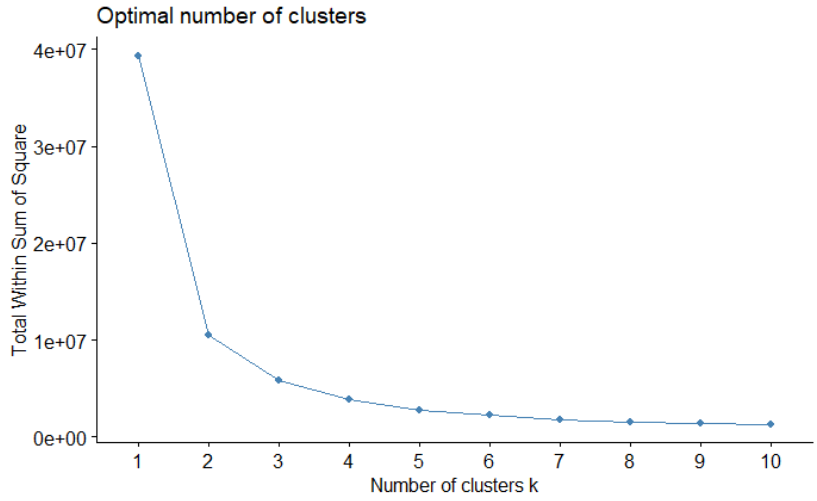


Figure 6.188 Elbow Method

In the Average Silhouette method, it is measured how similar each observation is to the cluster it is in when compared to other clusters. As a result of the method, the optimum number of clusters was determined as 2, as can be seen in Figure 6.189.

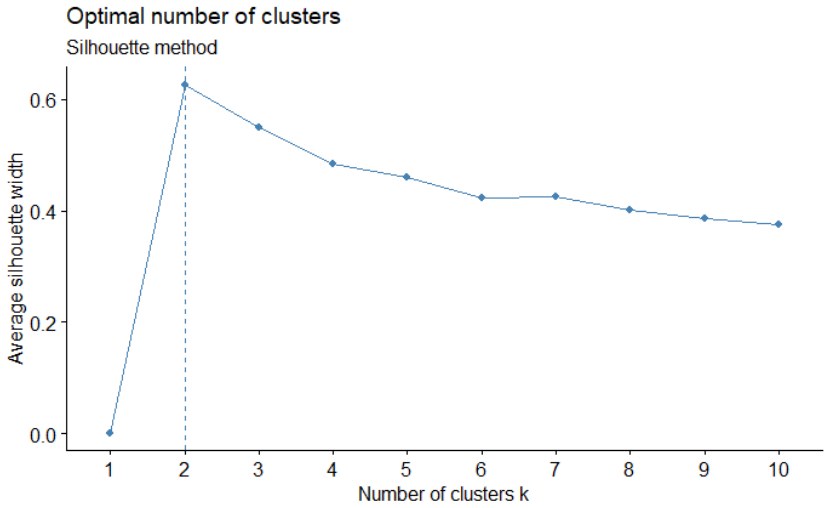


Figure 6.189 Average Silhouette Method

In the GAP method, a sample data set is first created with Monte-Carlo simulation. The errors of the actual clustered values and the expected values were compared. As seen in Figure 6.190, 2 clusters are proposed.

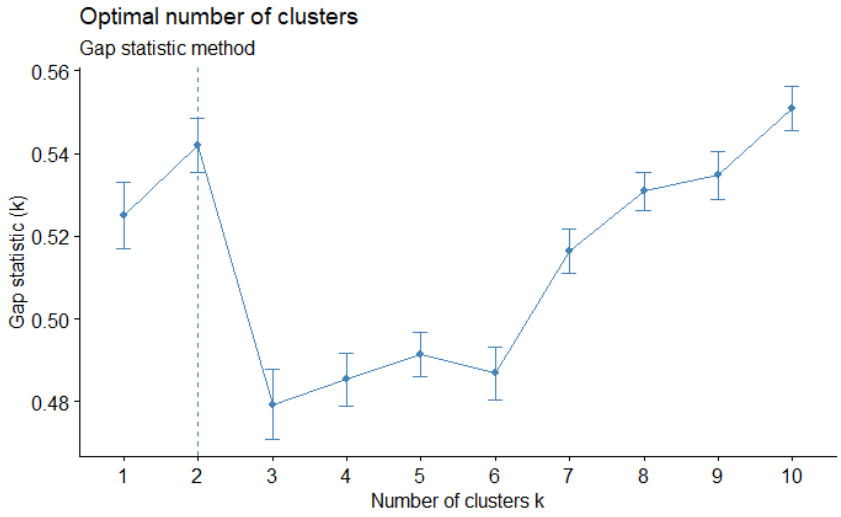


Figure 6.190 Gap Method

The outputs in Figure 6.191 were obtained by running the last K-Means with 2 clusters. Number of 2 is selected because 2 out of 3 method suggested that. The values in the centers of the clusters are also seen.

```
K-means clustering with 2 clusters of sizes 1427, 1011

Cluster means:
        Sex Patient_Followup_Status Case_Status   Contact Pregnancy Intensive_care
1 0.2838122               0.3412754    1.781359 0.6426069 0.9880869      0.9824807
2 0.9950544               0.3135509    1.824926 0.6983185 1.0000000      0.9861523
  Intubation      Age        CT Pneumonia Ferritin       CRP   Oxygen
1  0.9859846 41.89278 0.4660126 0.9656622 118.3708 1.0773110 97.71689
2  0.9881306 36.84768 0.4965381 0.9683482 338.8195 0.9614162 97.87043

Clustering vector:
   [1] 1 2 1 1 1 2 1 1 1 2 2 1 1 1 2 2 2 1 2 2 1 1 1 2 1 2 1 1 1 1 1 1 2 1 2 2 2 1
  [39] 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 2 2 2 1 1 2 1 2 1 1 1 2 2 2 1 2 2 2 1 2 2 2
  [77] 2 2 2 2 1 2 2 2 1 2 1 1 2 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 1 2 1 2 2
 [115] 1 2 2 2 1 1 1 2 1 1 1 1 2 2 2 1 1 1 2 2 2 2 2 1 1 1 1 2 1 1 2 2 1 1 1 1 2 1
 [153] 1 2 1 2 1 2 1 1 2 1 1 2 2 1 1 1 2 2 2 2 2 2 2 1 1 2 1 2 1 2 2 2 2 1 1 1 1 1
 [191] 1 1 1 1 1 1 2 2 1 2 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 1
```

Figure 6.191 K-Means with 2 Clusters (Outputs)

6.6.2 Hierarchical Clustering

6.6.2.1 Agglomerative Hierarchical Clustering

A scaling was made due to the variances of the variables. The distance matrix was created using the Euclidean method. Distance metrics were calculated to compare each observation with other observations on the basis of variables. After clustering with the hculst function, the dendogram seen in Figure 6.192 was formed.

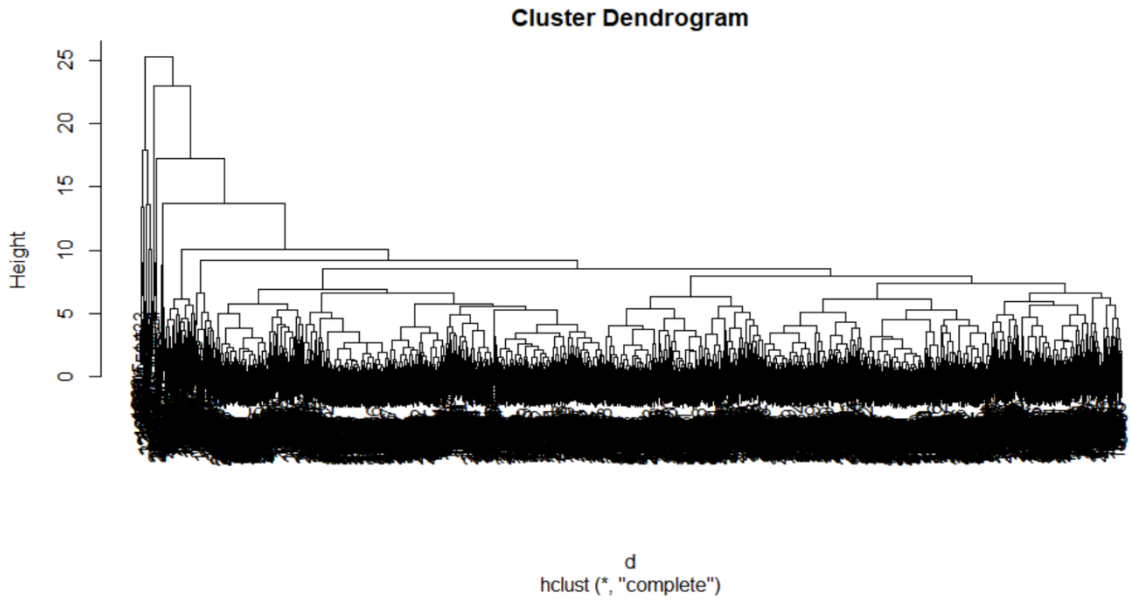


Figure 6.192 Cluster Dendrogram using complete

The left axis represents the distances. The legs touch the observations at the lowest level. In the upper levels, it shows clusters. It can be divided into 3 clusters with the leg cutting method.

6.6.2.2 Comparison of Combining Methods

Which method to use was decided by agnes. Outputs are as in Figure 6.193. The best method is the ward method.

```
  average    single  complete      ward
0.9625790 0.9544687 0.9751414 0.9943852
```

Figure 6.193 Comparison of Combining Methods

The dendrogram drawn with the ward, which was decided to be the best method, is as in Figure 6.194. Three clusters can be formed from this dendrogram.

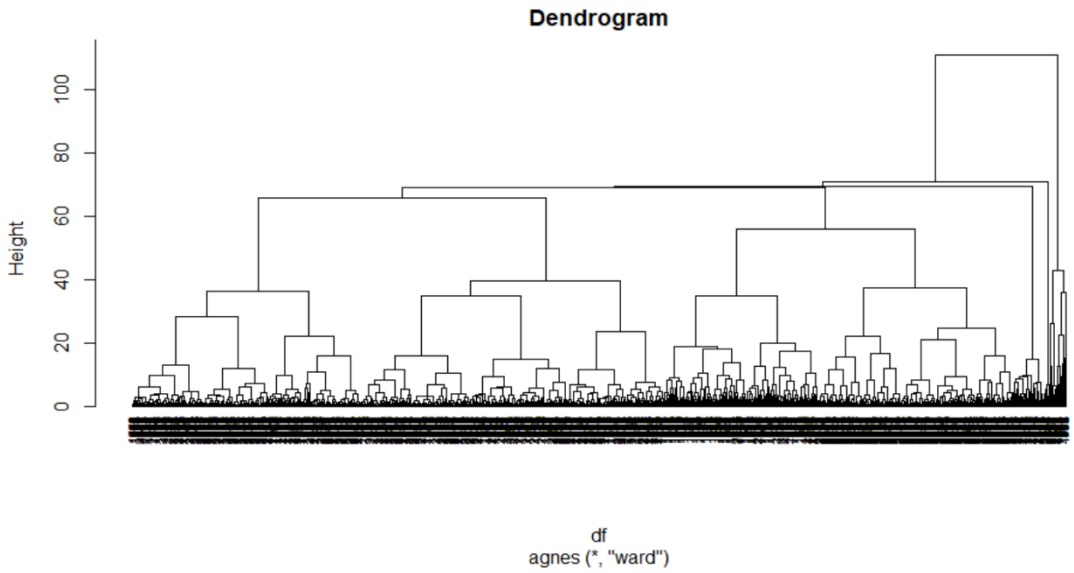


Figure 6.194 Cluster Dendrogram using ward

6.6.2.3 Divisive Hierarchical Clustering

Clustering was carried out with the diana approach. The partitioning coefficient was found to be 0.9723871 by drawing the dc component. The dendogram in Figure 6.195 was obtained.

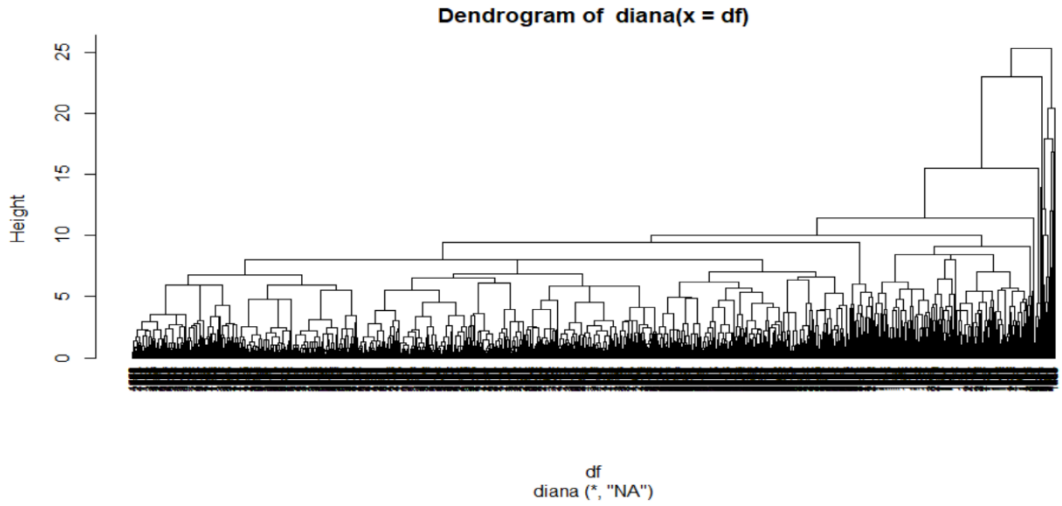


Figure 6.195 diana Dendrogram

6.6.2.4 Editing Dendrograms and Accessing Components

Clustering was performed with ward.D2. While dividing with cutree, 2 clusters were selected in Figure 6.196 and 3 clusters in Figure 6.197.

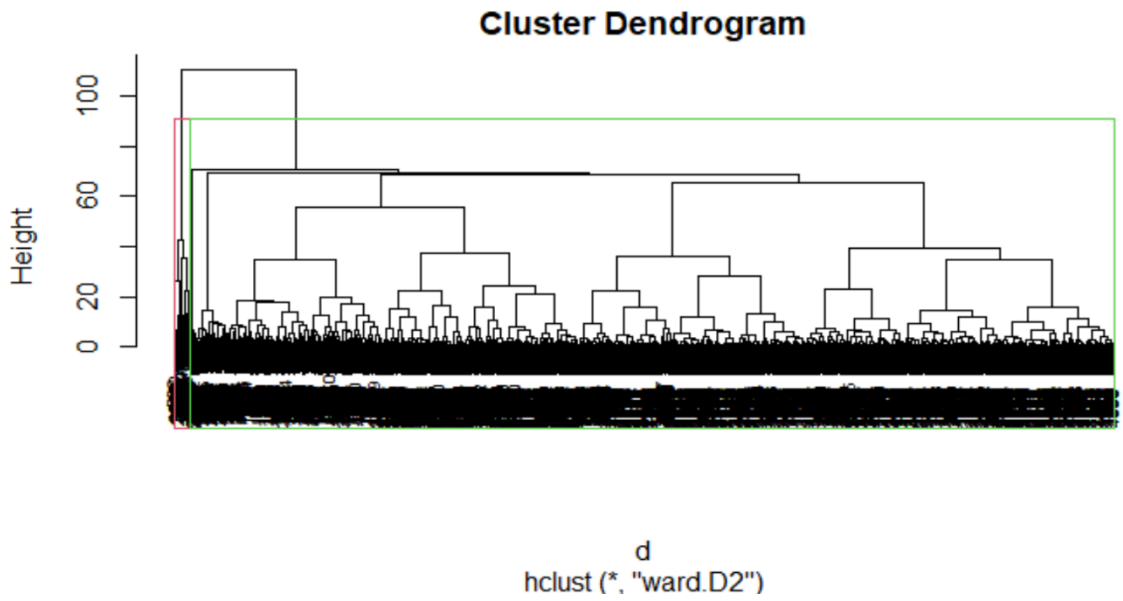


Figure 6.196 Cluster Dendrogram using ward.D2 with Two Clusters

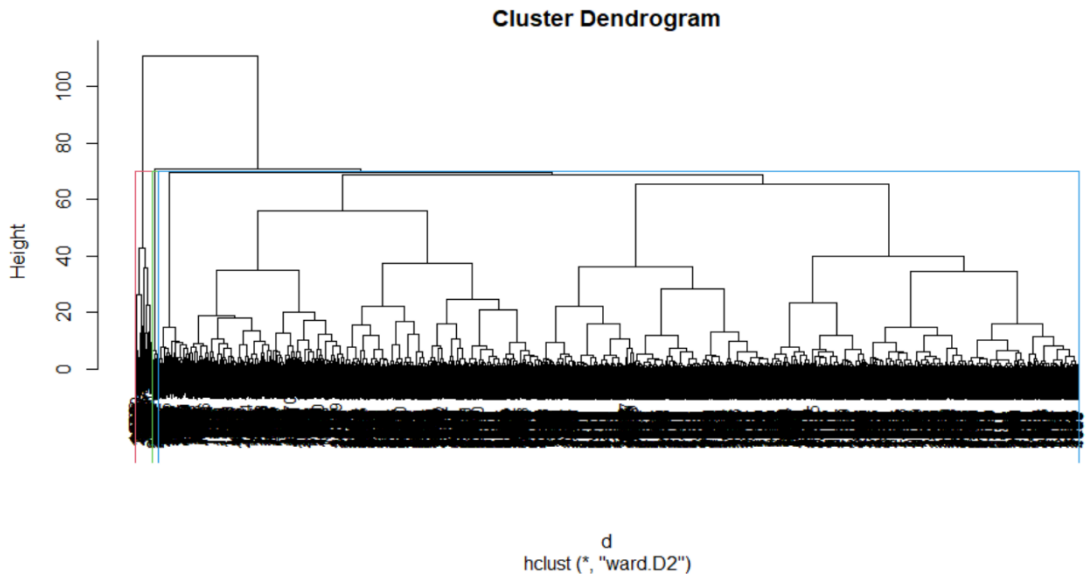
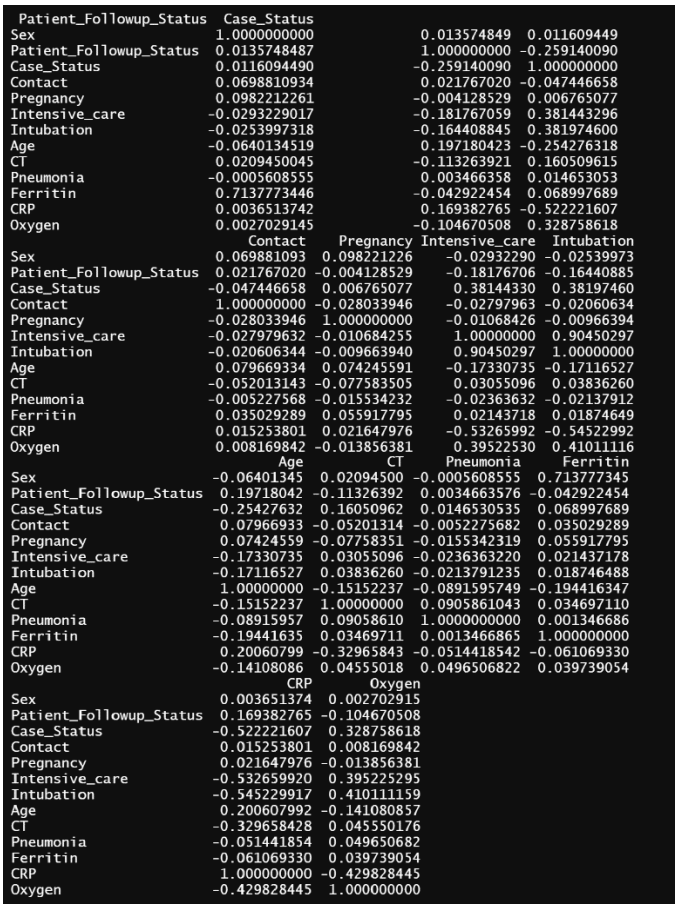


Figure 6.197 Cluster Dendrogram using ward.D2 with Three Clusters

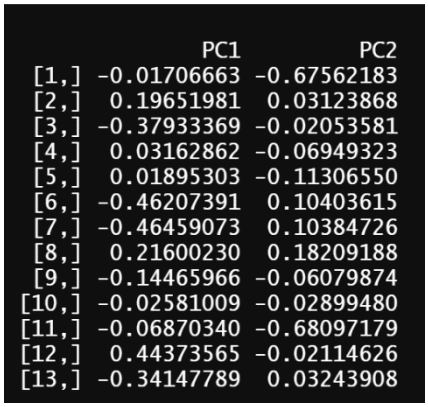
6.6.3 Principal Component Analysis : PCA

After the data is standardized, the covariance matrix in Figure 6.198 is created. Then, eigenvalues and eigenvalue vectors are created over this covariance matrix. These are as in Figure 6.199. Since eigenvalue vectors are automatically created in negative direction in R, the conversion is done.



	Patient_Followup_Status	Case_Status		
Sex	1.0000000000	0.013574849	0.011609449	
Patient_Followup_Status	0.0135748487	1.000000000	-0.259140090	
Case_Status	0.0116094490	-0.259140090	1.000000000	
Contact	0.0698810934	0.021767020	-0.047446658	
Pregnancy	0.0982212261	-0.004128529	0.006765077	
Intensive_care	-0.0293229017	-0.181767059	0.381443296	
Intubation	-0.0253997318	-0.164408845	0.381974600	
Age	-0.0640134519	0.197180423	-0.254276318	
CT	0.0209450045	-0.113263921	0.160509615	
Pneumonia	-0.0005608555	0.003466358	0.014653053	
Ferritin	0.7137773446	-0.042922454	0.068997689	
CRP	0.0036513742	0.169382765	-0.522221607	
Oxygen	0.0027029145	-0.104670508	0.328758618	
	Contact	Pregnancy	Intensive_care	Intubation
Sex	0.069881093	0.098221226	-0.02932290	-0.02539973
Patient_Followup_Status	0.021767020	-0.004128529	-0.18176706	-0.16440885
Case_Status	-0.047446658	0.006765077	0.38144330	0.38197460
Contact	1.000000000	-0.028033946	-0.02797963	-0.02060634
Pregnancy	-0.028033946	1.000000000	-0.01068426	-0.00966394
Intensive_care	-0.027979632	-0.010684255	1.00000000	0.90450297
Intubation	-0.020606344	-0.009663940	0.90450297	1.00000000
Age	0.079669334	0.074245591	-0.17330735	-0.17116527
CT	-0.052013143	-0.077583505	0.03055096	0.03836260
Pneumonia	-0.005227568	-0.015534232	-0.02363632	-0.02137912
Ferritin	0.035029289	0.055917795	0.02143718	0.01874649
CRP	0.015253801	0.021647976	-0.53265992	-0.54522992
Oxygen	0.008169842	-0.013856381	0.39522530	0.41011116
	Age	CT	Pneumonia	Ferritin
Sex	-0.06401345	0.02094500	-0.0005608555	0.713777345
Patient_Followup_Status	0.19718042	-0.11326392	0.0034663576	-0.042922454
Case_Status	-0.25427632	0.16050962	0.0146530535	0.068997689
Contact	0.07966933	-0.05201314	-0.0052275682	0.035029289
Pregnancy	0.07424559	-0.07758351	-0.0155342319	0.055917795
Intensive_care	-0.17330735	0.03055096	-0.0236363220	0.021437178
Intubation	-0.17116527	0.03836260	-0.0213791235	0.018746488
Age	1.00000000	-0.15152237	-0.0891595749	-0.194416347
CT	-0.15152237	1.00000000	0.0905861043	0.034697110
Pneumonia	-0.08915957	0.09058610	1.0000000000	0.001346686
Ferritin	-0.19441635	0.03469711	0.0013466865	1.000000000
CRP	0.20060799	-0.32965843	-0.0514418542	-0.061069330
Oxygen	-0.14108086	0.04555018	0.0496506822	0.039739054
	CRP	Oxygen		
Sex	0.003651374	0.002702915		
Patient_Followup_Status	0.169382765	-0.104670508		
Case_Status	-0.522221607	0.328758618		
Contact	0.015253801	0.008169842		
Pregnancy	0.021647976	-0.013856381		
Intensive_care	-0.532659920	0.395225295		
Intubation	-0.545229917	0.410111159		
Age	0.200607992	-0.141080857		
CT	-0.329658428	0.045550176		
Pneumonia	-0.051441854	0.049650682		
Ferritin	-0.061069330	0.039739054		
CRP	1.000000000	-0.429828445		
Oxygen	-0.429828445	1.000000000		

Figure 6.198 The Covariance Matrix



	PC1	PC2
[1,]	-0.01706663	-0.67562183
[2,]	0.19651981	0.03123868
[3,]	-0.37933369	-0.02053581
[4,]	0.03162862	-0.06949323
[5,]	0.01895303	-0.11306550
[6,]	-0.46207391	0.10403615
[7,]	-0.46459073	0.10384726
[8,]	0.21600230	0.18209188
[9,]	-0.14465966	-0.06079874
[10,]	-0.02581009	-0.02899480
[11,]	-0.06870340	-0.68097179
[12,]	0.44373565	-0.02114626
[13,]	-0.34147789	0.03243908

Figure 6.199 Eigenvalues and Eigenvalue Vectors

Figure 6.200 shows the effects of the components on the variables.

Cases <chr>	PC_ONE <dbl>	PC_TWO <dbl>
1	0.139815208	1.536350283
2	-0.892196363	-0.980941851
3	0.006719065	1.858420419
4	0.735485077	-0.327362195
5	-0.452724359	1.582604804
6	-0.586899059	-1.040692518
7	-0.517551686	1.288907861
8	-0.155278051	0.851038275
9	-0.056204384	1.206199090
10	-0.504729984	-1.238544766

1-10 of 2,438 rows Previous 1 2 3 4 5 6

Figure 6.200 Effects of the Components on the Variables

Currently, the observation units are reduced to 2 variables and the score is calculated for each observation.

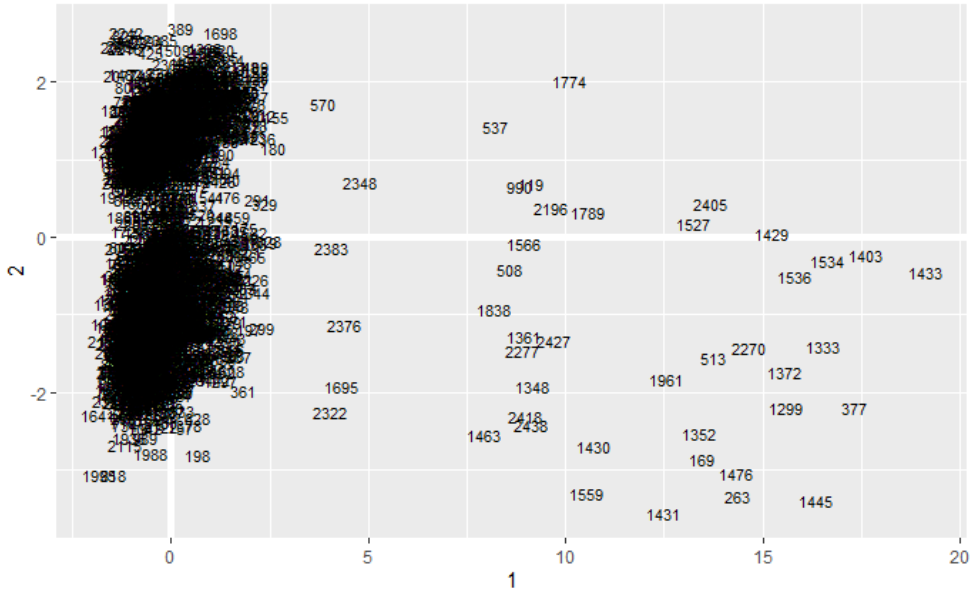


Figure 6.201 Locations of the Cases

The locations of the cases are seen in Figure 6.201, which is composed of principal component scores.

We are in the step of deciding on the number of components. The graphs in Figure 6.202 were used to decide how many components to choose. As can be seen in the cumulative scree plot, the explainability of the model is close to 1 with 9 components.

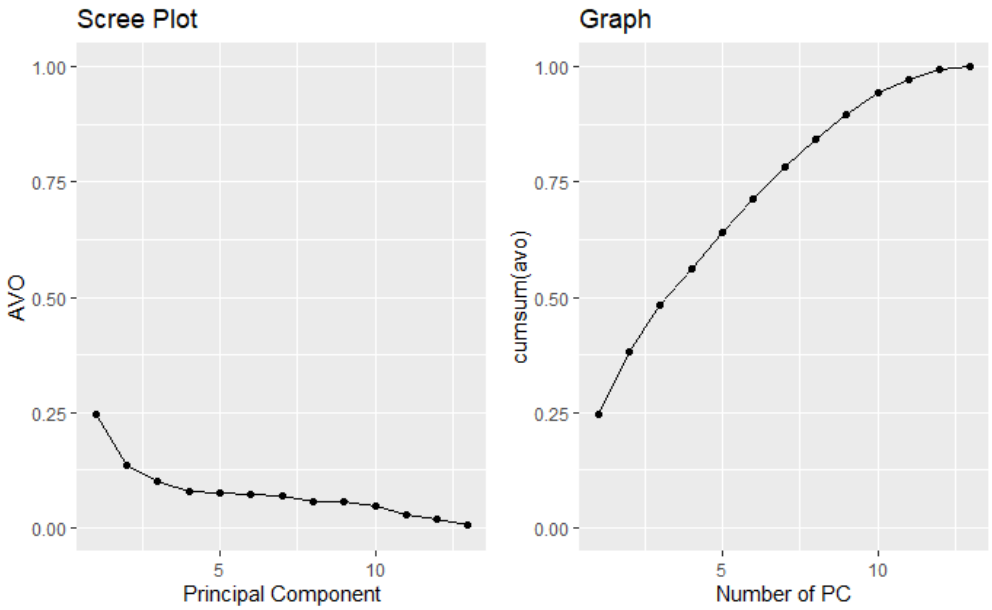


Figure 6.202 Explainability of the Model

As a result, in Figure 6.203, the variables are reduced to 2 components so that the information they carry is maximum. The red expressions express the effect of the variables on the components.

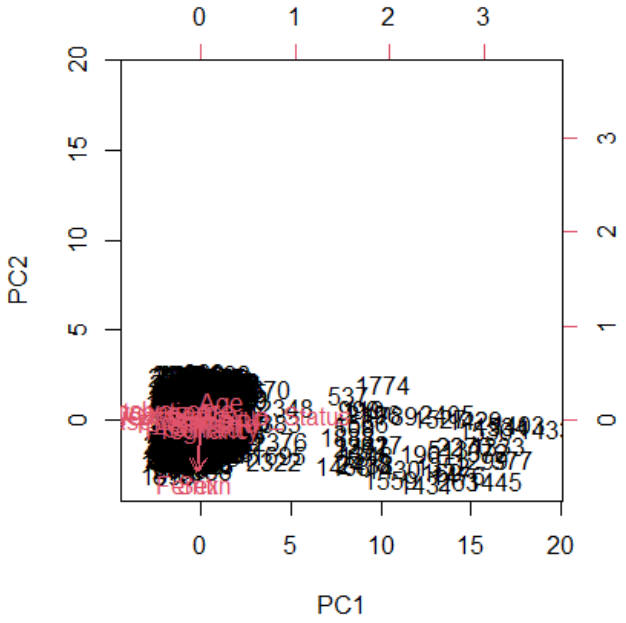


Figure 6.203 Final PCA

7. SECTION

CONCLUSION AND RECOMMENDATIONS

The study is completed after 16 different machine learning methods are applied for linear regression, non-linear regression, classification and clustering. The study aims to determine which methods are more suitable for the data set consists of COVID-19 cases.

Despite a lot of methods are applied, not all of them gave successful results. Some algorithms have higher prediction performance. For linear regression, multiple linear regression has the highest R-squared value, which is 0.69. Support vector regression gives the best result among non-linear applications. The R-squared of the model is 0.4926, and RMSE is 0.8787. In classification, random forest is more successful than other methods. The accuracy of the model is 0.9012, and the F1 Score of the model is 0.9091. It can be said that classification is a more appropriate method among machine learning methods. In the three optimum cluster number determination methods used in the K-Means method, the optimum cluster number was determined as 2(2 out of 3 methods). In the hierarchical clustering method, the highest success rate was obtained in the ward method.

Future activities will aim to apply more methods. If more methods are applied, more results are achieved. Some of them can have higher prediction performance than the methods used in the thesis.

REFERENCES

[1] World Health Organization, [Çevrimiçi]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019. [Erişildi: 13 June 2021].

[2] H. B. Syeda, M. Syed, K. W. Sexton, S. Syed, S. Begum, F. Syed, F. Prio ve F. Y. J, *Role of Machine Learning Techniques to Tackle the COVID-19 Crisis: Systematic Review,* 2021.

[3] F. Tezza, G. Lorenzoni, D. Azzolina, S. Barbar, L. A. C. Leone ve D. Gregori, «Predicting in-Hospital Mortality of Patients with COVID-19 Using Machine Learning Techniques,» *Journal of Personalized Medicine,* 2021.

[4] P. Melin, J. C. Monica, D. Sanchez ve O. Castillo, *Analysis of Spatial Spread Relationships of Coronavirus (COVID-19) Pandemic in the World using Self Organizing Maps,* 2020.

[5] N. S. Punn, S. K. Sonbhadra ve S. Agarwal, *COVID-19 Epidemic Analysis using MachineLearning and Deep Learning Algorithms,* 2020.

[6] «LAB TESTS ONLINE,» [Çevrimiçi]. Available: https://labtestsonline.org/tests/c-reactive-protein-crp.

[7] Y. E. Bulut and Datajarlabs, "Veri Bilimi Nedir ve Nasıl Öğrenilebilir?," 29 Aug 2018. [Online]. Available: https://medium.com/datajarlabs/veri-bilimi-nedir-ve-nas%C4%B1l-%C3%B6%C4%9Frenilebilir-b5ff8c581bbc.

[8] N. Thakur, «The differences between Data Science, Artificial Intelligence, Machine Learning, and Deep Learning,» 17 April 2020. [Çevrimiçi]. Available: https://medium.com/ai-in-plain-english/data-science-vs-artificial-intelligence-vs-machine-learning-vs-deep-learning-50d3718d51e5.

[9] A. S. Dayanıklı, "Aykırı Değer (Outlier) Analizi Nedir? Uç Değerler Nasıl Tespit Edilir?," 11 February 2021. [Online]. Available: https://ravenfo.com/2021/02/11/aykiri-deger-analizi/.

[10] https://www.techrepublic.com/, "https://www.techrepublic.com/article/3-best-practices-for-exploratory-data-visualizations/," 2021. [Online].

[11] «R ile Veri Bilimi ve Machine Learning,» [Çevrimiçi]. Available: https://www.udemy.com/course/veri-bilimi-ve-makine-ogrenmesi-egitimi/learn/lecture/13074674#overview.

[12] https://kevserkose.wordpress.com/, 2021. [Online].

[13] https://www.mathsisfun.com/, 2021. [Online].

[14] http://www.matematik.us/, 2021. [Online].

[15] https://datavizcatalogue.com/, 2021. [Online].

[16] https://www.r-graph-gallery.com/, 2021. [Online].

[17] https://towardsdatascience.com/, 2021. [Online].

[18] https://chartio.com/, 2021. [Online].

[19] N. Sharma, R. Sharma and N. Jindal, "Machine Learning and Deep Learning Applications-A Vision," in *Global Transitions Proceedings*, 2021.

[20] M. Antunes, "Knowledge extraction from semi-structured sources - Scientific Figure on ResearchGate," [Online]. Available: https://www.researchgate.net/figure/Machine-learning-taxonomy-Machine-learning-methods-organized-based-on-the-training_fig2_331114192. [Accessed 15 Jun 2021].

[21] J. Brownlee, «A Gentle Introduction to the Bootstrap Method,» 25 May 2018. [Çevrimiçi]. Available: https://machinelearningmastery.com/a-gentle-introduction-to-the-bootstrap-method/.

[22] J. Brownlee, "Supervised and Unsupervised Machine Learning Algorithms," 16 March 2016. [Online].

[23] M. V. Keskin, "Veri Bilimi Okulu," 2021. [Online].

[24] «Underfitting and Overfitting in Machine Learning,» [Çevrimiçi]. Available: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/.

[25] J. Frost, "How To Interpret R-squared in Regression Analysis," [Online].

[26] «ROC curve analysis,» [Çevrimiçi]. Available: https://www.medcalc.org/manual/roc-curves.php.

[27] D. S. (. Kampakis, "Performance Measures: Cohen’s Kappa statistic," [Online].

[28] J. B. PhD, "Probabilistic Model Selection with AIC, BIC, and MDL," Machine Learning Mastery, [Online]. Available: https://machinelearningmastery.com/probabilistic-model-selection-measures/.

[29] «Introduction to linear regression,» [Çevrimiçi]. Available: https://rpubs.com/DevanshuMehrotra/548560.

[30] M. Scholz, «Nonlinear PCA,» [Çevrimiçi]. Available: http://www.nlpca.org/pca_principal_component_analysis.html.

[31] «Principal Component Regression (PCR),» [Çevrimiçi]. Available: https://learnche.org/pid/latent-variable-modelling/principal-components-regression.

[32] M. Kuhn ve K. Johnson, Applied Predictive Modeling.

[33] [Çevrimiçi]. Available: https://www.saedsayad.com/support_vector_machine_reg.htm.

[34] K. Vala, «Tree-Based Methods: Regression Trees,» 18 March 2019. [Çevrimiçi]. Available: https://towardsdatascience.com/tree-based-methods-regression-trees-4ee5d8db9fe9.

[35] «Regression Trees,» [Çevrimiçi]. Available: https://uc-r.github.io/regression_trees.

[36] «Bootstrap aggregating,» [Çevrimiçi]. Available: https://en.wikipedia.org/wiki/Bootstrap_aggregating.

[37] «Gradient Boosting,» [Çevrimiçi]. Available: https://docs.paperspace.com/machine-learning/wiki/gradient-boosting.

[38] J. K. Jaiswal and R. Das, "Artificial Neural Network Algorithms based Nonlinear Data," *International Journal of Engineering & Technology,* pp. 169-176, 2018.

[39] Ž. Živković ve I. M. a. Đ. Nikolić, «ARTIFICIAL NEURAL NETWORK METHOD APPLIED,» *Serbian Journal of Management,* pp. 143 - 155 , 2009.

[40] E. Hatipoğlu, «Machine Learning — Classification — Logistic Regression,» 12 July 2018. [Çevrimiçi]. Available: https://medium.com/@ekrem.hatipoglu/machine-learning-classification-logistic-regression-part-8-b77d2a61aae1.

[41] «Classification and Regression Trees,» [Çevrimiçi]. Available: https://ldapwiki.com/wiki/Classification%20and%20Regression%20Trees.

[42] V. Tabora, «SVM Classification Algorithms In R,» Medium, 27 September 2019. [Çevrimiçi]. Available: https://medium.com/0xcode/svm-classification-algorithms-in-r-ced0ee73821.

[43] J. Le, "Support Vector Machines in R," datacamp, 22 August 2018. [Online].

[44] «RDocumentation,» [Çevrimiçi]. Available: https://www.rdocumentation.org/packages/MASS/versions/7.3-54/topics/dropterm.

[45] «MathWorks,» [Çevrimiçi]. Available: https://www.mathworks.com/help/stats/cooks-distance.html.

[46] J. Brownlee, «Machine Learning Mastery,» 5 February 2016. [Çevrimiçi]. Available: https://machinelearningmastery.com/tune-machine-learning-algorithms-in-r/.

[47] M. V. Keskin, *R ile Veri Bilimi ve Makine Öğrenmesi,* 2021.

[48] İ. Kocabıyık, 12 Ekim 2015. [Çevrimiçi]. Available: http://devveri.com/veri-madenciligi/r-ile-korelasyon-regresyon-ve-zaman-serisi-analizleri.

RESUME

PERSONAL INFORMATIONS

Name-Surname : Gaye Nur Onur

Date of Birth and Place : 03.03.1999 / Tosya

Foreign Language : English

E-mail : l6117049@std.yildiz.edu.tr

EDUCATIONAL SITUATION

Degree	Study Field	School/University	Graduation Year
Bachelor's	Industrial Engineering	Yıldız Technical University	
High School	Science school	Kastamonu Science High School	2017

RESUME

PERSONAL INFORMATIONS

Name-Surname : Yağmur Ecem Pehlevan

Date of Birth and Place : 27.10.1998 / Bahçelievler

Foreign Language : English

E-mail : l6116047@std.yildiz.edu.tr

EDUCATIONAL SITUATION

Degree	Study Field	School/University	Graduation Year
Bachelor's	Industrial Engineering	Yıldız Technical University	
High School	Anatolian school	Cahit Elginkan Anatolian High School	2016