UNIVERSITY OF INSUBRIA

Department of Theoretical and Applied Science

Master in Computer Science

Intelligent System Course Final Project

Machine Learning techniques Report

Regression and Classification on various datasets

Advisor:
Prof. Sandro Morasca

Authors:
Marco Bogani
Omar Tornaghi

Genuary 2022

Summary

The purpose of this work is to build regression and classification models by using different machine learning techniques. The project repository is freely available on GitHub

Contents

List of Figures

Chapter 1

Introduction

This report shows a regression and a classification problem solved by different machine learning techniques. For each model shown in this document, results are calculated and compared in order to make a ranking of the best machine learning models for the selected problem. A fixed pipeline has been followed in order to cover all the principal steps in a data analysis problem. Generally, a preliminary analysis was made on the data-set in order to find possible correlations, outliers and missing values. Then, machine learning models were trained on a subset of the original data-set. Finally, performance metrics were obtained on the trained models used on a test data-set and a final ranking was made in order to provide a ranking for each model.

Chapter 2

Regression

Regression is a technique used in machine learning for finding correlations between dependent and independent variables. Therefore, regression algorithms help predict continuous variables such as prices, trends and patterns.

The data-set being used for the regression analysis can be download from Kaggle. It contains various information about used cars. The goal of the analysis is to predict the selling price as accurately as possible. Therefore, several regression techniques will be presented and a ranking based on different metrics such as RMSE, RAE and R SQUARED will be made.

2.1 Preliminary analysis

The data-set contains the following features:

1. Name: The model of the vehicle
2. Year: The year of the vehicle
3. Selling price: The selling price of the vehicle
4. Km driven: The number of Km driven by the vehicle
5. Fuel: The type of fuel the vehicle uses
6. Seller type: The type of seller

7. Transmission: The type of transmission

8. Owner: The type of owner

9. Mileage: The mileage of the car

10. Engine: The engine capacity of the vehicle

11. Max power: The max power of the Engine

12. Seats: The number of seats

The dataset has 8128 records. The following bar charts were obtained.

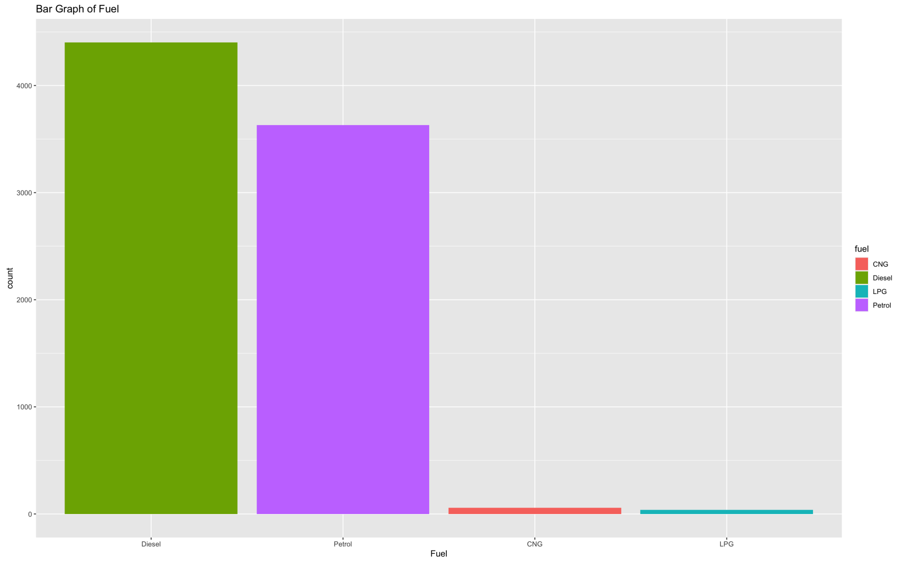

Figure 2.1: Bar chart of fuel feature

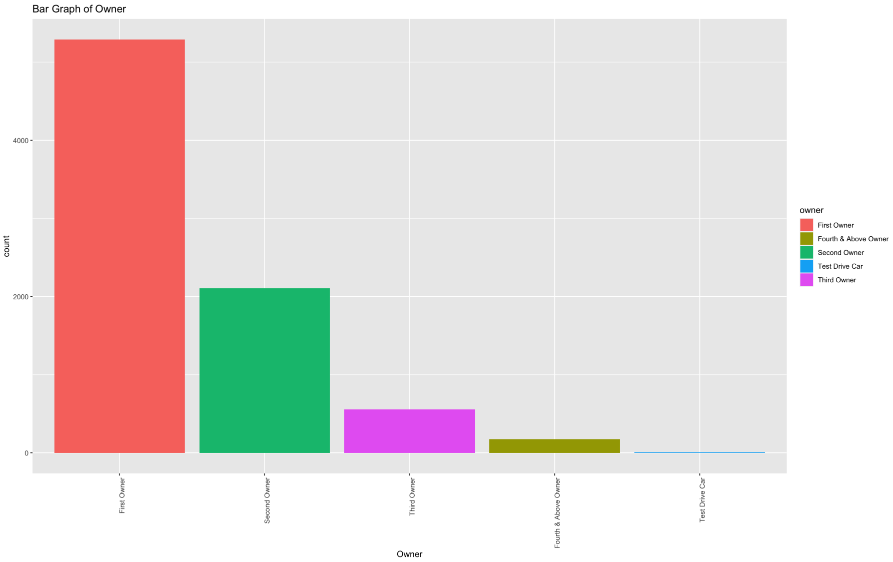

Figure 2.2: Bar chart of owner feature

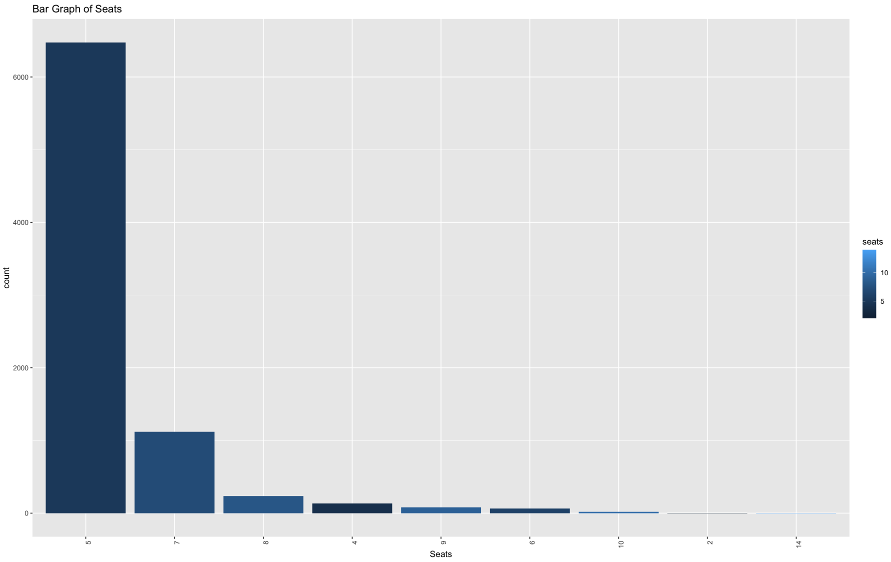

Figure 2.3: Bar chart of seats feature

2.2 Data cleaning

Firstly, the car models were derived by splitting the name field and getting the first word. Therefore, 32 unique names were obtained. Secondly, a unique number was assigned to each car model.

For other features like fuel,owner or selling price, a manual substitution was carried out. The substitution was made by providing an integer number for each string.

For the following columns: mileage,engine and max power, the units of measurement were removed and the numerical values were obtained.

By the preliminary analysis it was discovered that some columns contained missing values. Therefore a median value was provided to replace them.

At the end of the data-cleaning procedure, as the image below shows, no missing values were found.

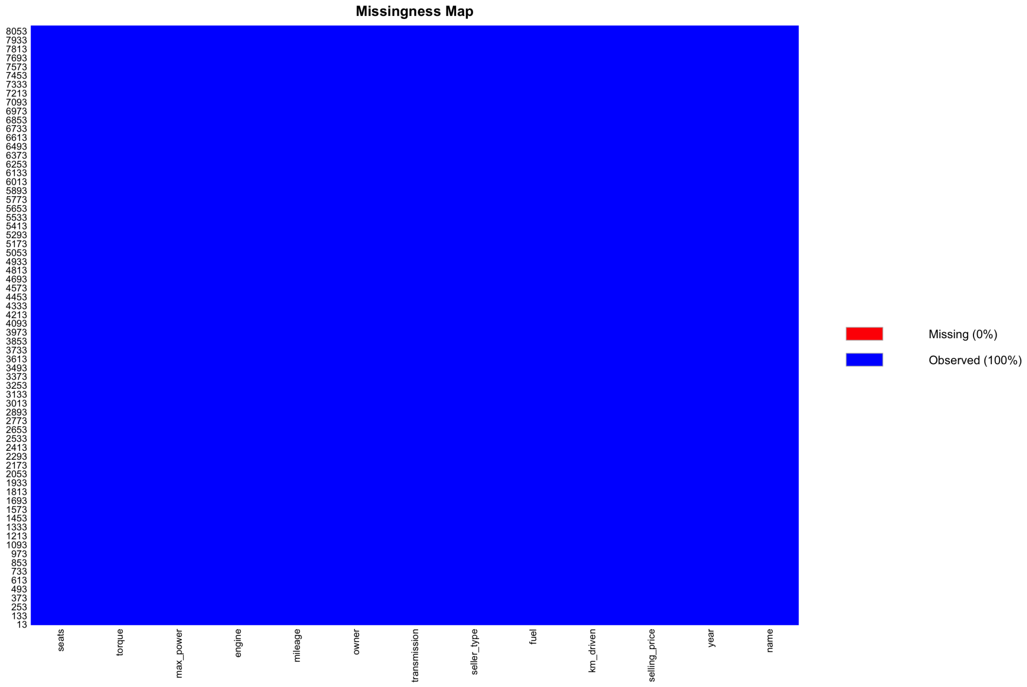

Figure 2.4: Miss map of the car data-set

2.3 Data-set analysis

Firstly, a box plot was made to see if there were outliers. As the image below shows, the column km driven presented a great quantity of them. Therefore a outliers removal procedure was carried out.

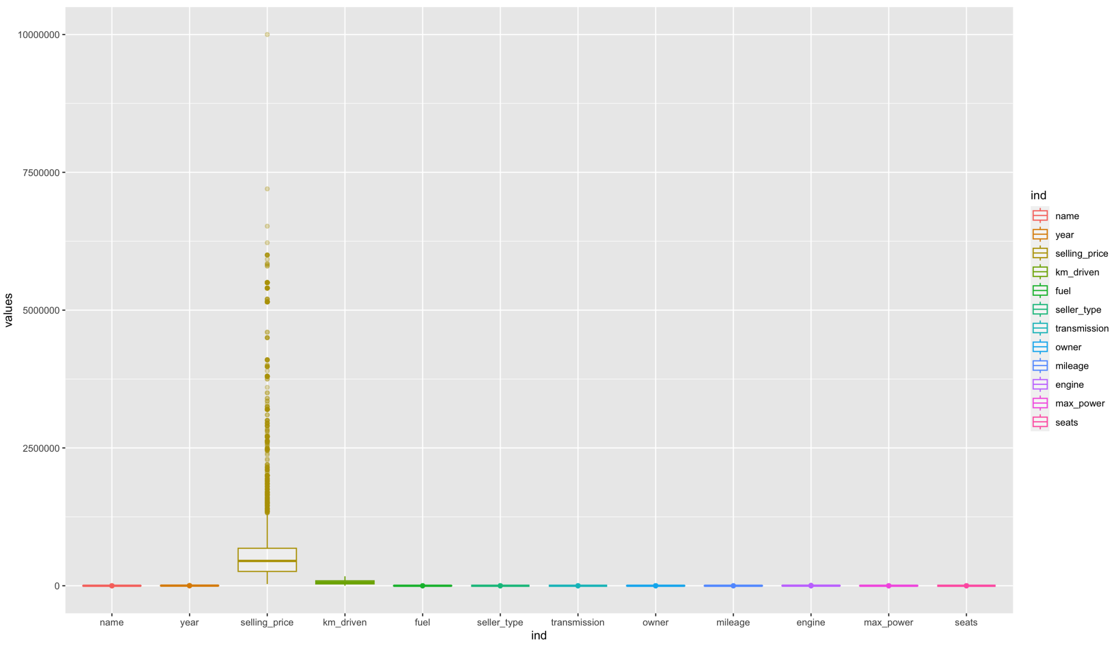

Figure 2.5: Box plot before outliers removal

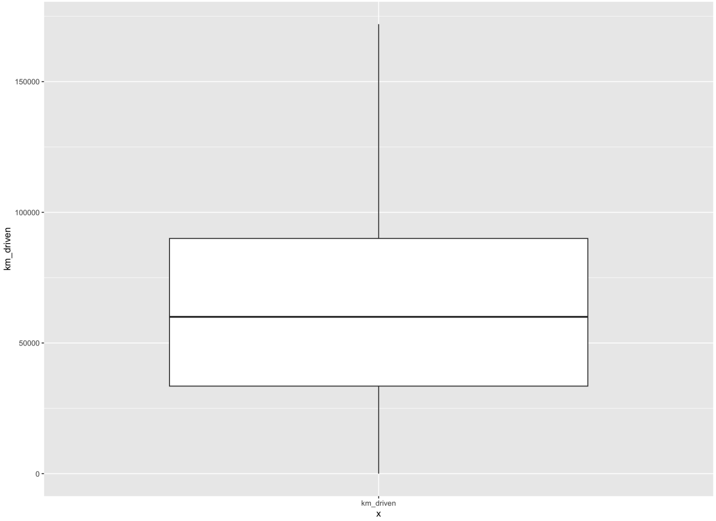

Figure 2.6: Box plot of km driven after outliers removal

Secondly, the distribution analysis of km driven and selling price was made by using two histograms.

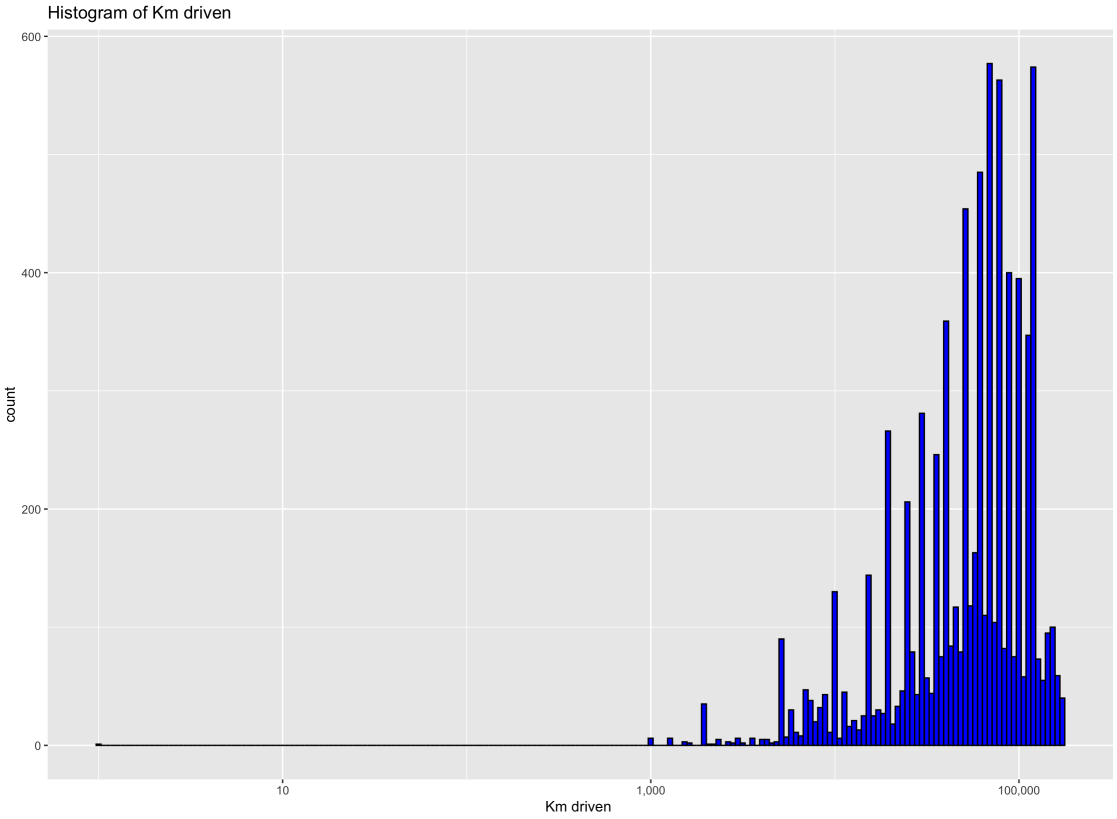

Figure 2.7: Histogram of km driven

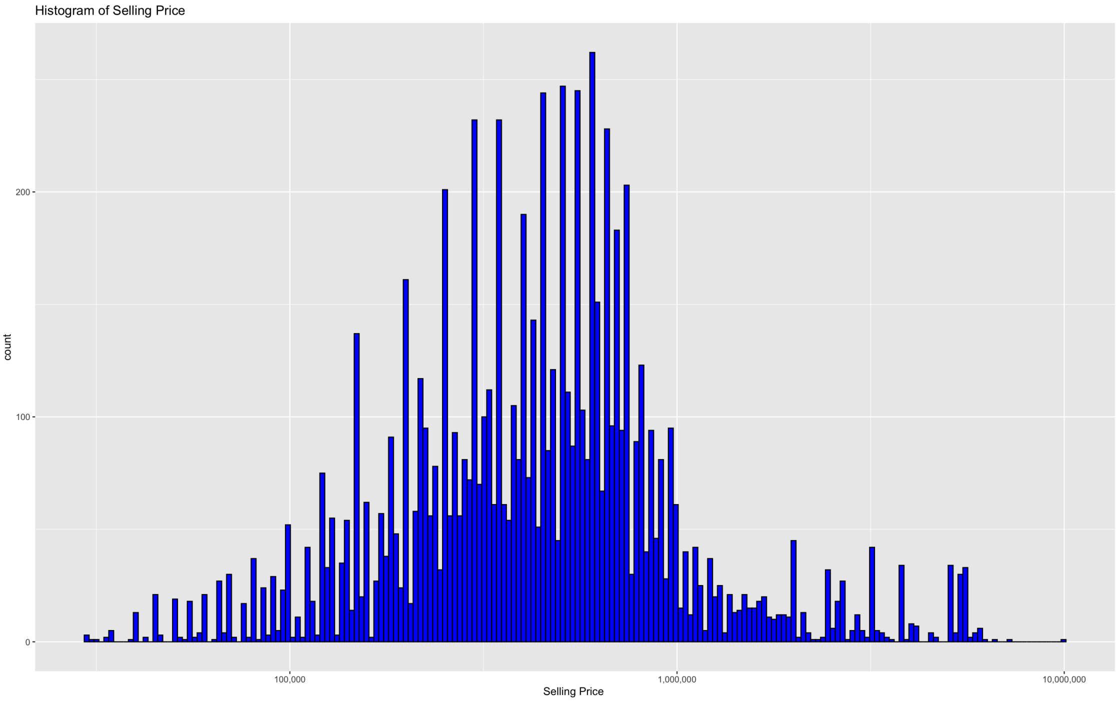

Figure 2.8: Histogram of selling price

The correlation between the independent variables was analyzed using a correlation matrix. For example it was found, as expected, that engine and max power have a positive correlation while e.g. year and owner a negative correlation.

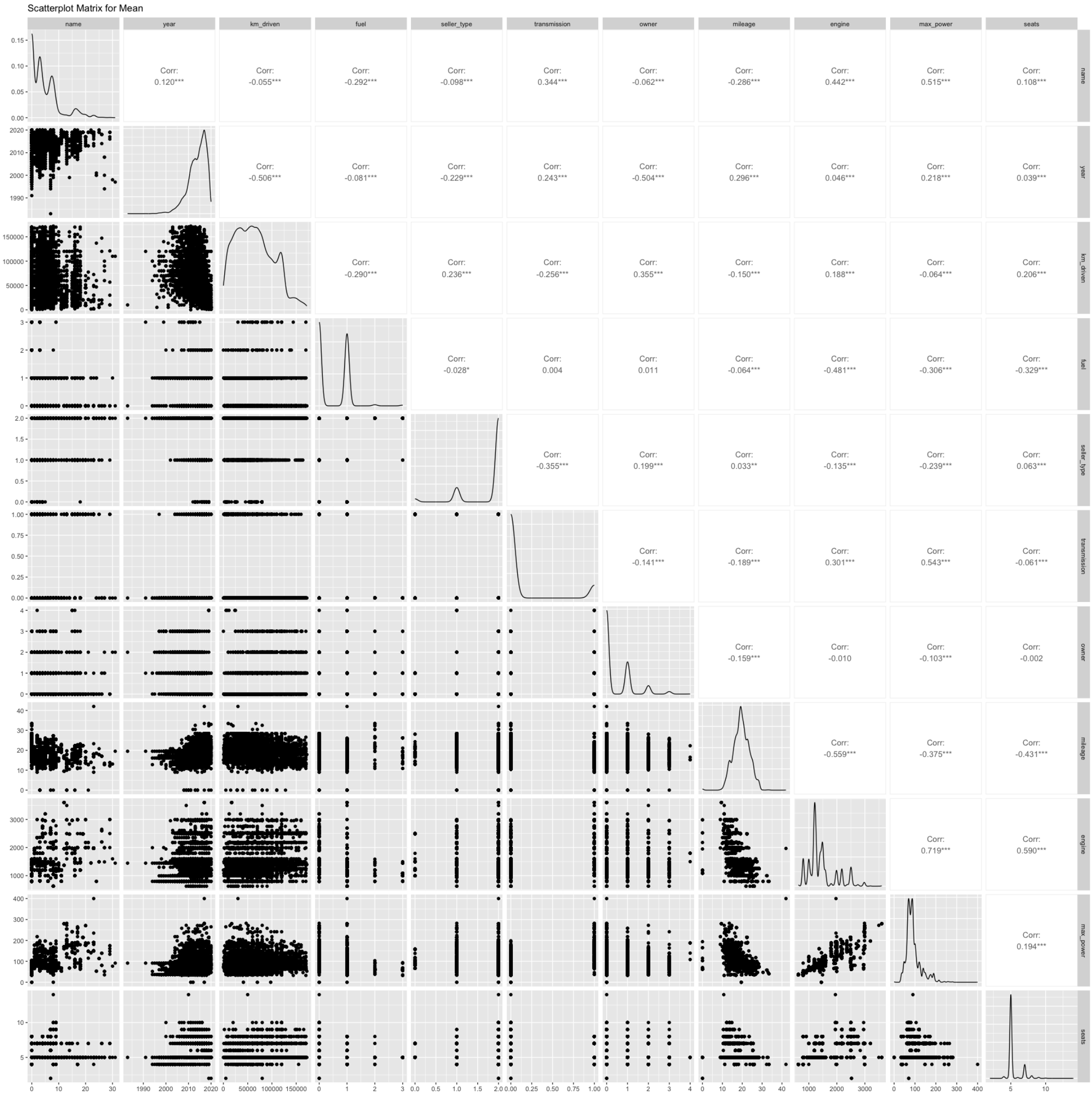

Figure 2.9: Correlations between variables

Finally, the dataset was divided into two parts: Train and Test containing 70% and 30% of the data respectively.

2.4 OLS

Ordinary Least Squares(OLS) linear regression was the first technique being used for predicting the selling price of the vehicles. The analysis was carried out in several step: first of all a feature selection phase was made in order to maximize the model accuracy. Secondly a linear model was trained using the Train data-set. Finally predictions were made using the trained model and RMSE,RAE,R SQUARED were extrapolated in order to quantify the performance of the model.

2.4.1 Feature selection

The feature selection was made by computing the p-value between the independent variables and the dependent variable(selling price). The following table shows the results:

Feature	p-value
Name	$< 2e-16$
Year	$< 2e-16$
Km driven	$< 2e-16$
Fuel	0.9166
Seller type	$< 2e-16$
Transmission	$< 2e-16$
Owner	0.1832
Mileage	$< 2e-16$
Engine	0.0469
Max power	$< 2e-16$
Seats	0.3987

Table 2.1: Variable p values

Therefore, the following feature list contains the selected features: name,year,km driven,seller type,mileage,transmission,max power.

2.4.2 Training

The linear model was trained on the Train data-set(5513 records). The resulted adjusted R squared is 0.7 which is, generally, a good indicator of the model.

By plotting the difference between the residuals and the fitted values, the following chart is obtained:

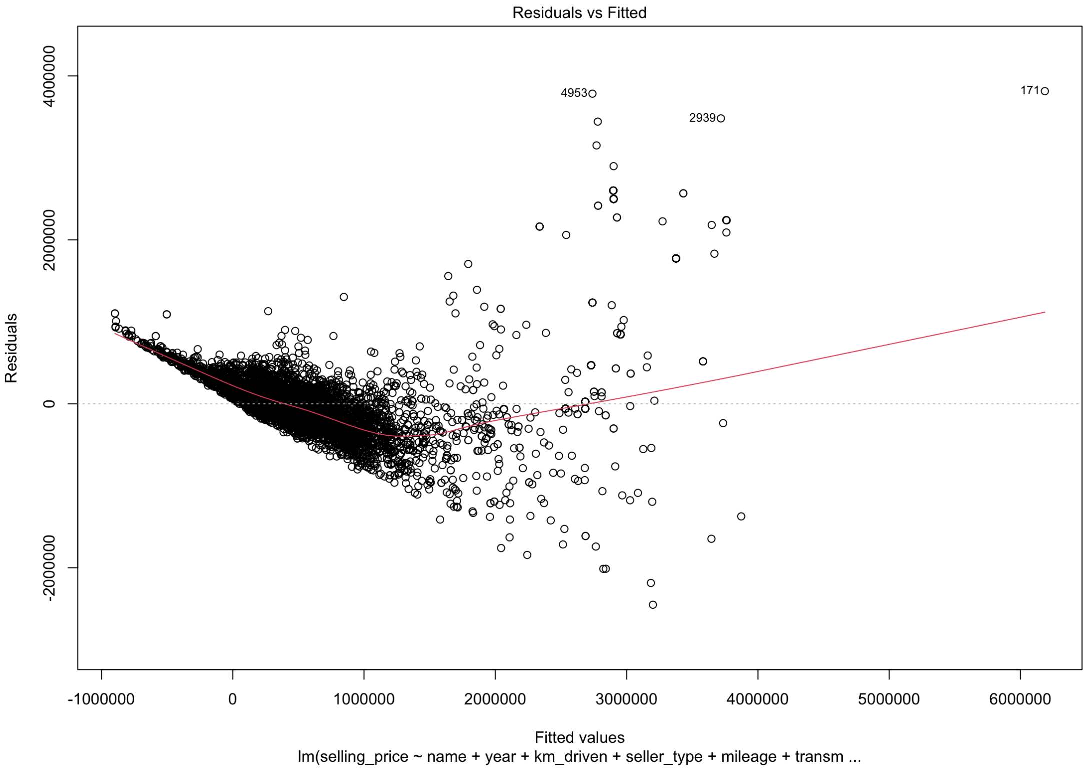

Figure 2.10: Residuals vs fitted values

2.4.3 Performance Metrics

During the test phase, the trained model was used to predict values using the Test data-set(2360 records). Therefore the following metrics were derived:

- Trained R-SQUARED: 0.7

- RMSE: 436881
- RAE: 0.60

The following scatter-plot reflects the actual selling price versus the predicted selling price using the trained linear model on the Test dataset.

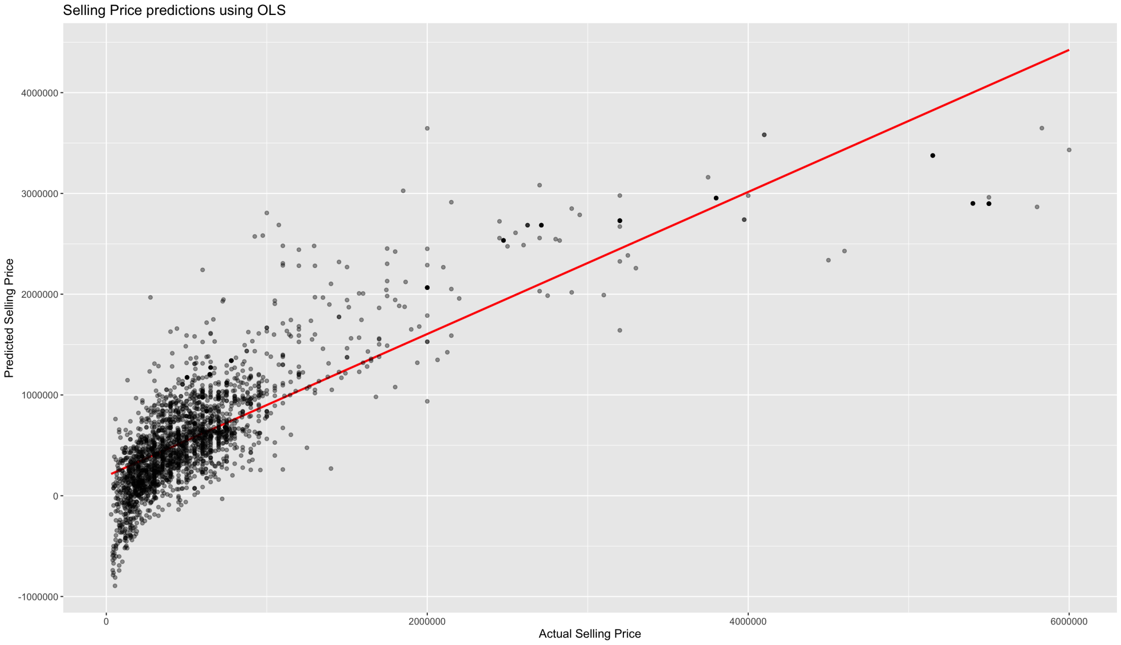

Figure 2.11: Linear model: Actual selling prices vs Predicted selling prices

2.5 Random forest

A similar approach was followed using a random forest model.

2.5.1 Feature selection

Firstly, the following visualization of the importance of the variables was generated.

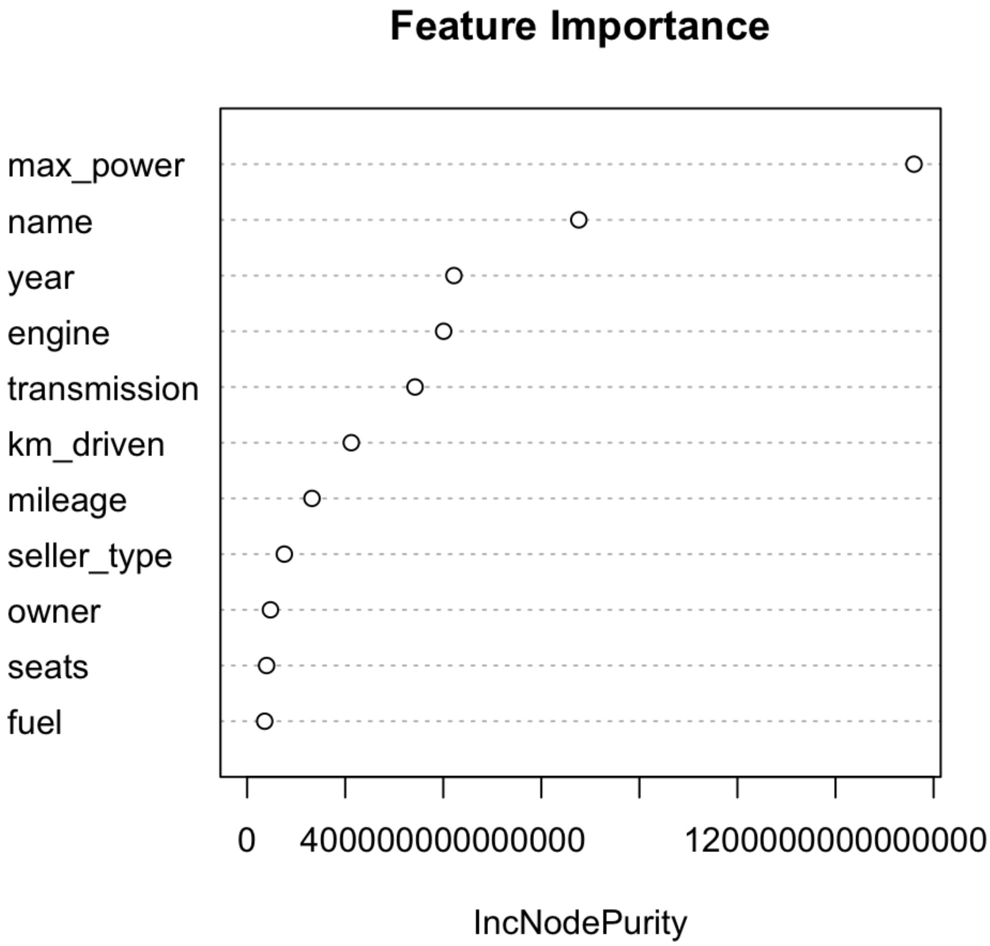

Figure 2.12: Random forest: Variable importance

As a results, all features were selected. Secondly, some measurements of the error rate based on the number of tree were provided. As expected, the error rate decreases as the number of trees increases.

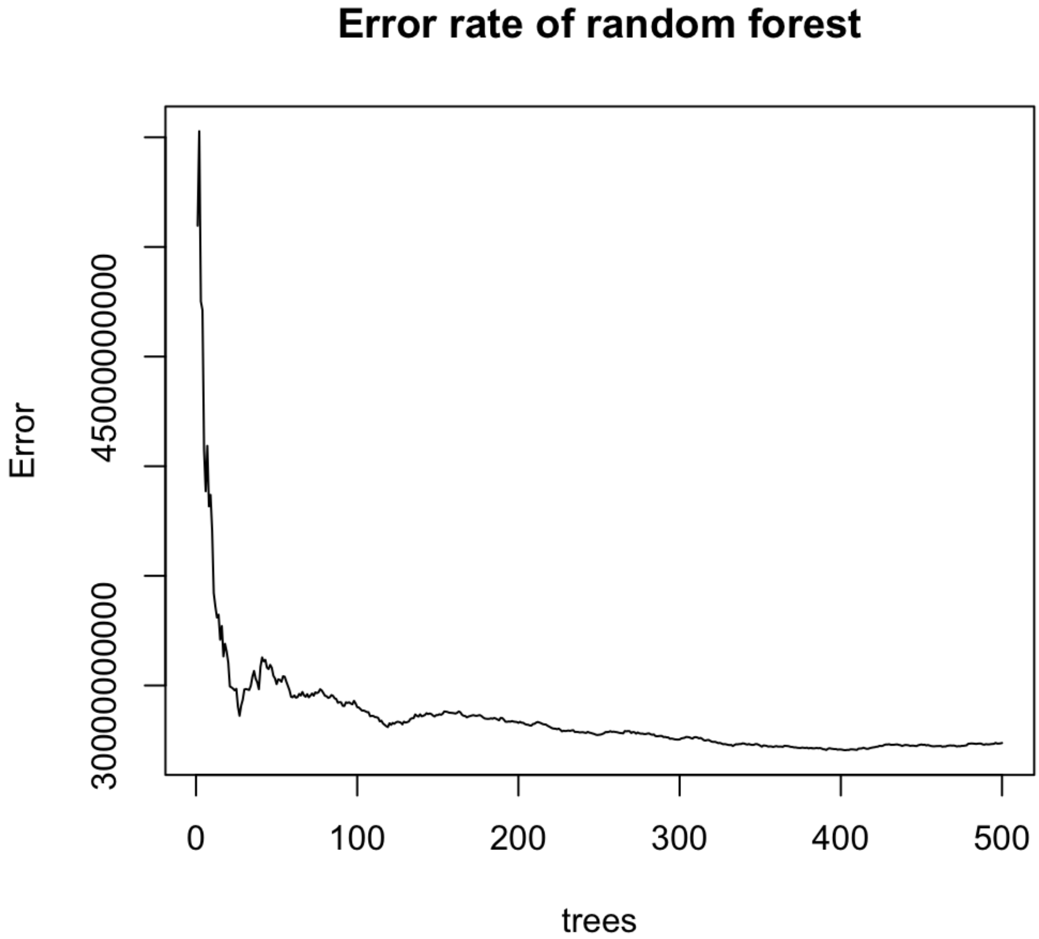

Figure 2.13: Random forest: Error rate by changing the number of trees

2.5.2 Performance Metrics

The obtained metrics were:

- RMSE: 116776
- RAE: 0.16

As the following scatter-plot confirms, this model predicts the selling price very well.

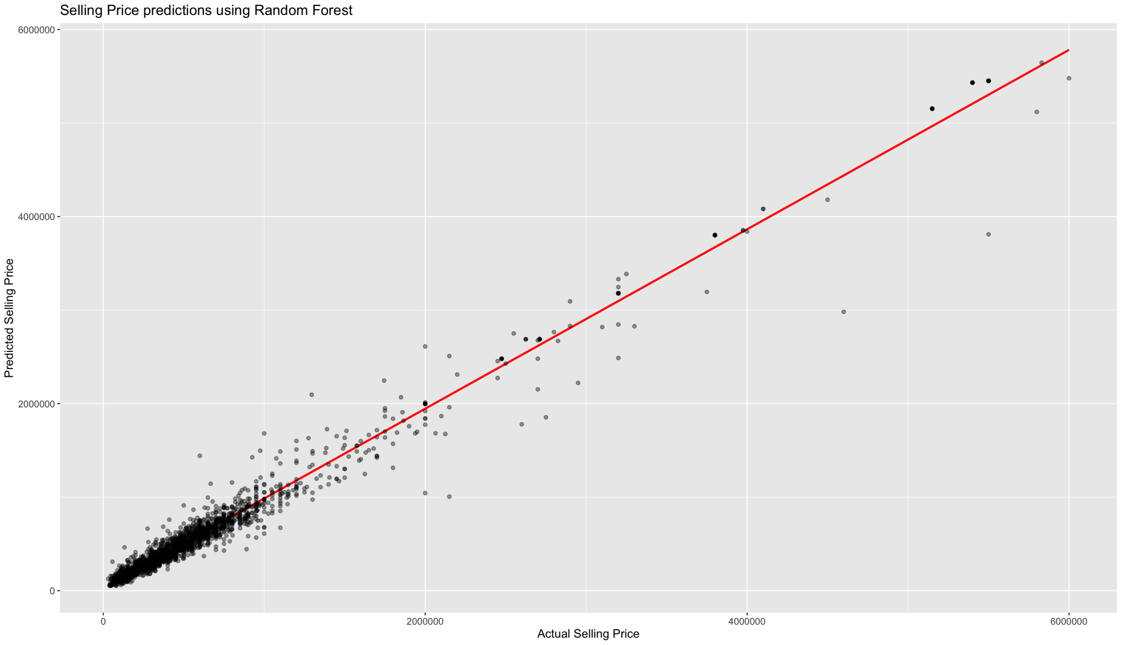

Figure 2.14: Random forest: Actual selling prices vs Predicted selling prices

2.6 Ridge regression

A Ridge regression model was evaluated using the same process as above.

2.6.1 Training

The model was trained using a k-fold cross-validation to find the optimal lambda value based on the Mean Squared Error(MSE).

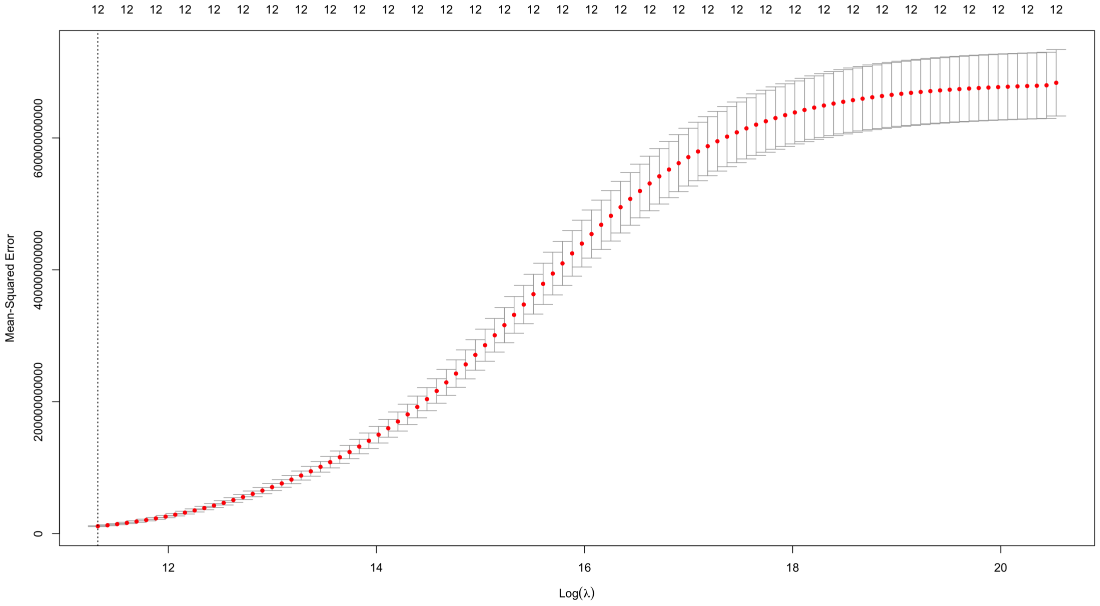

Figure 2.15: Ridge regression: Mean Square Error for Lambda values

2.6.2 Performance Metrics

As a results of the selling price predictions, the following metrics were obtained:

- Trained R-SQUARED: 0.98
- RMSE: 99785
- RAE: 0.15

As the following scatter-plot confirms, also this model predicts the selling price very well.

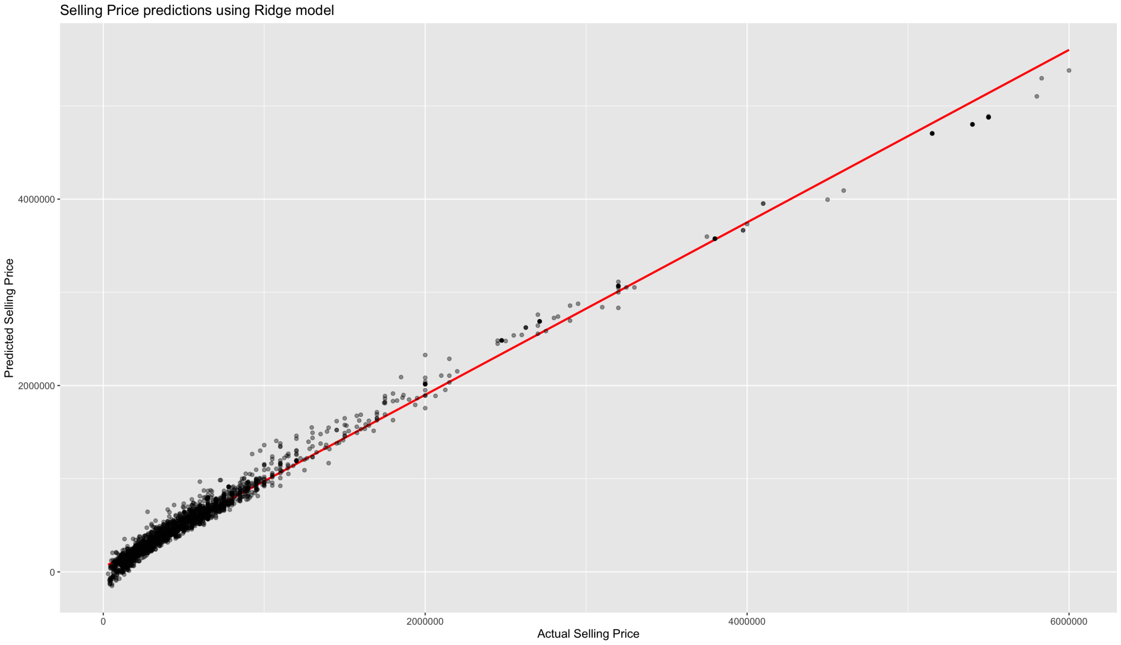

Figure 2.16: Ridge regression: Actual selling prices vs Predicted selling prices

2.7 Results

The following table is a summary of the performance metrics being calculated from all the models being used. As the RMSE and RAE metrics show, Ridge is the best model to predict the selling price of a vehicle based on the car data-set, followed by Random Forest and OLS.

Model	RMSE	RAE
Ridge	99785	0.15
Random Forest	116776	0.16
OLS	436881	0.60

Table 2.2: Model results

Chapter 3

Classification

Classification is a machine-learning technique used to identify the category of new observations based on training data.

This study aims to train different models with various classification algorithms to classify whether a breast tumour is benign or malignant. The data set used in this analysis is the Breast Cancer Wisconsin (Diagnostic) Data Set (available on Kaggle).

3.1 Preliminary analysis

The data set consists of thirty-three columns where the first is the observation id, the second is the diagnosis of tumour (M = malignant; B = benign) and the others are features grouped by *mean*, *se* and *worst*. Each of these groups contains ten features:

1. radius (mean of distances from center to points on the perimeter)
2. texture (standard deviation of gray-scale values)
3. perimeter
4. area
5. smoothness (local variation in radius lengths)
6. compactness ($perimeter^2/area - 1.0$)

7. concavity (severity of concave portions of the contour)

8. concave points (number of concave portions of the contour)

9. symmetry

10. fractal dimension ("coastline approximation" - 1)

Since column ID was useless and the last column contains only NaN, these were removed.

The total number of observations in this data set is 569, where the 63% indicates the absence of cancer cells and the 37% the presence of cancerous cells.

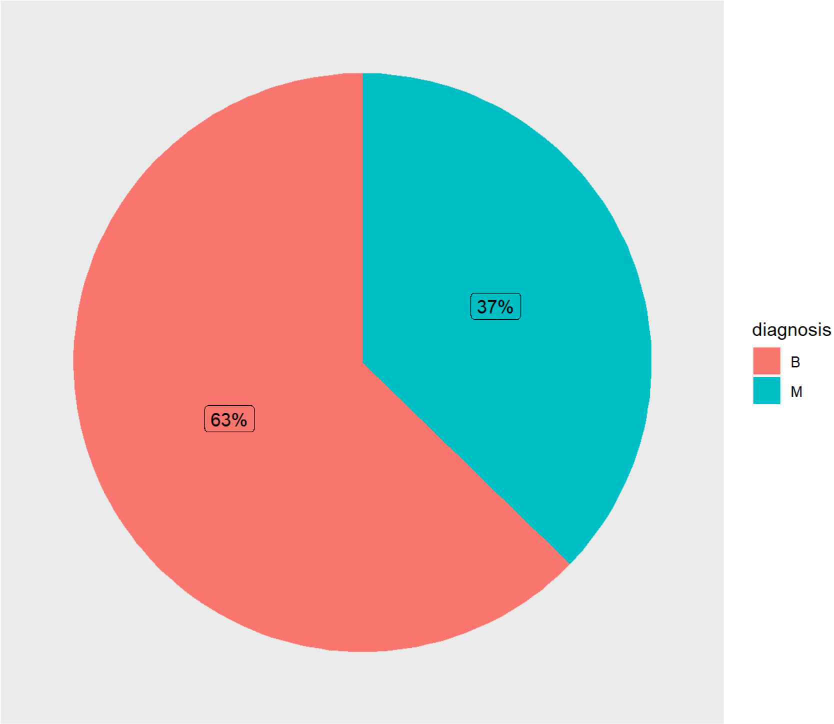

Figure 3.1: Frequency of cancer diagnosis.

The following matrix of scatterplot represents the correlation between features of the same group. On the diagonal of the matrix is represented the feature distribution.

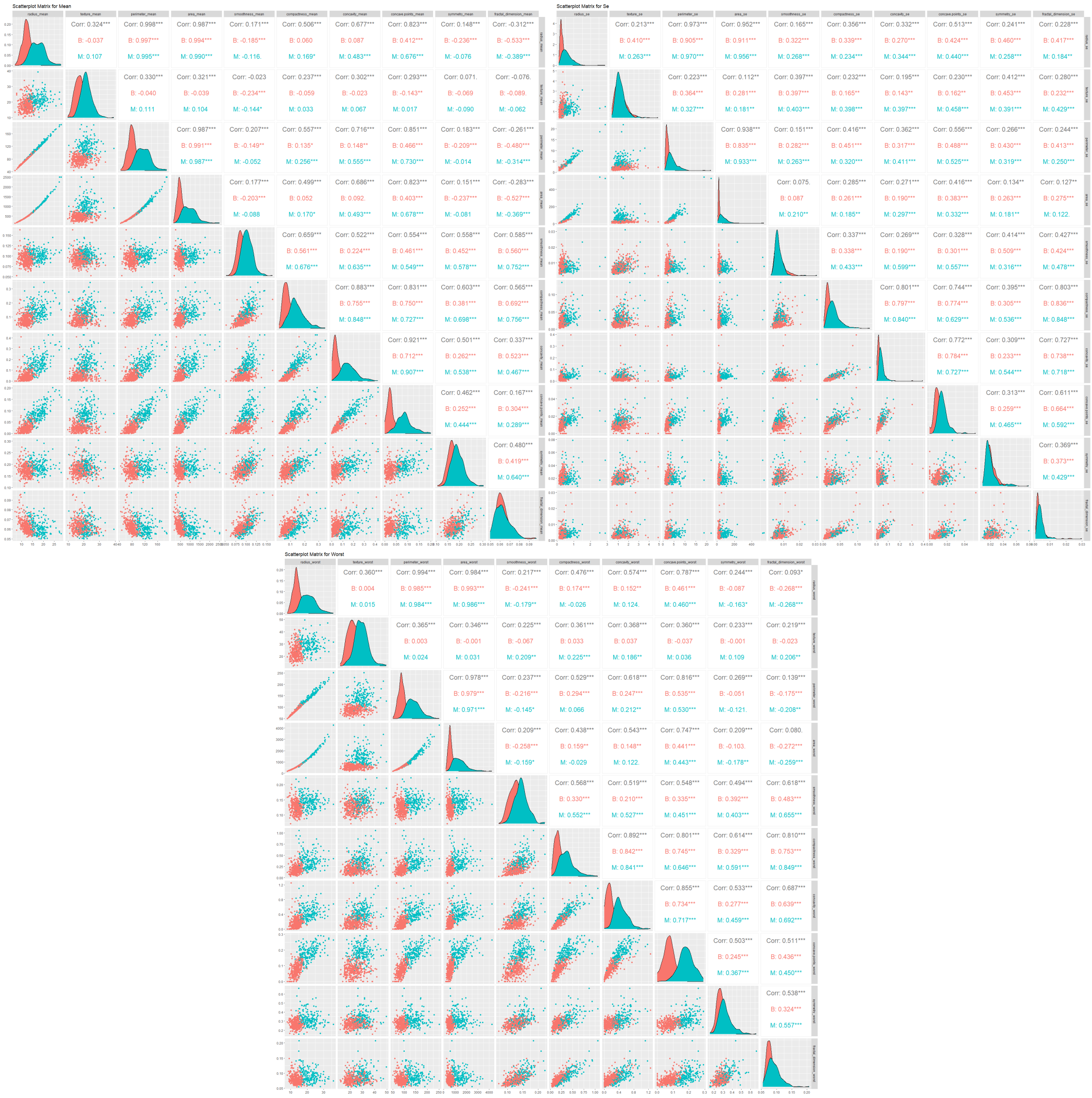

Figure 3.2: Features matrix of scatterplot.

3.2 Data-set analysis

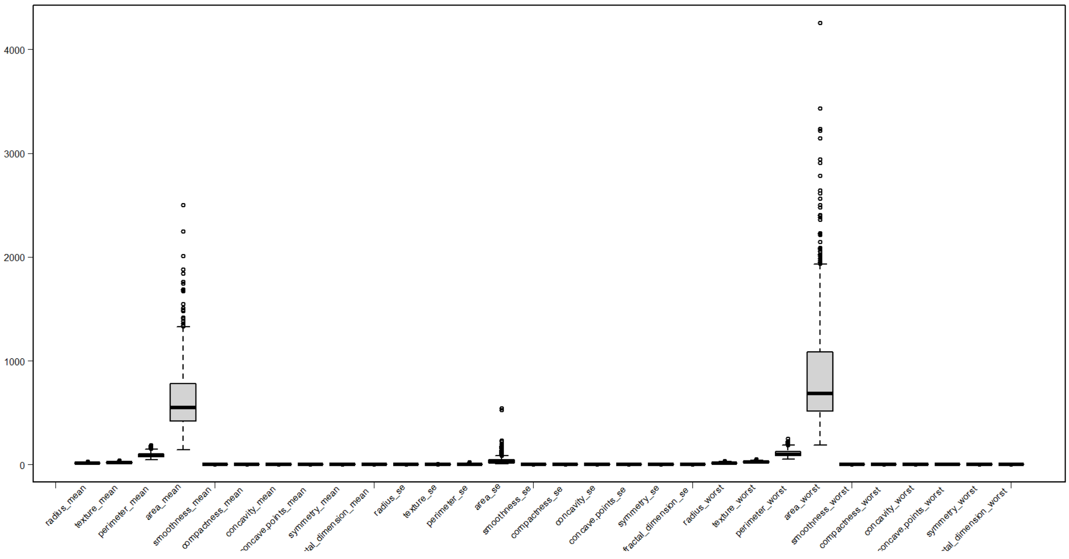

Figure 3.3: Features boxplot.

The first analysis aims to identify potential outliers. As it is possible to notice in figure 3.3, in the following features possible outliers are presents: *area_worst*, *area_mean*, *area_se*, *perimeter_worst* and *perimeter_mean*. However, by removing them, the models were less accurate and therefore no cleaning was carried out.

As shown in figure 3.2, it is possible to notice that the features are normally distributed and only some of them are quite separated (B = red; M = blue).

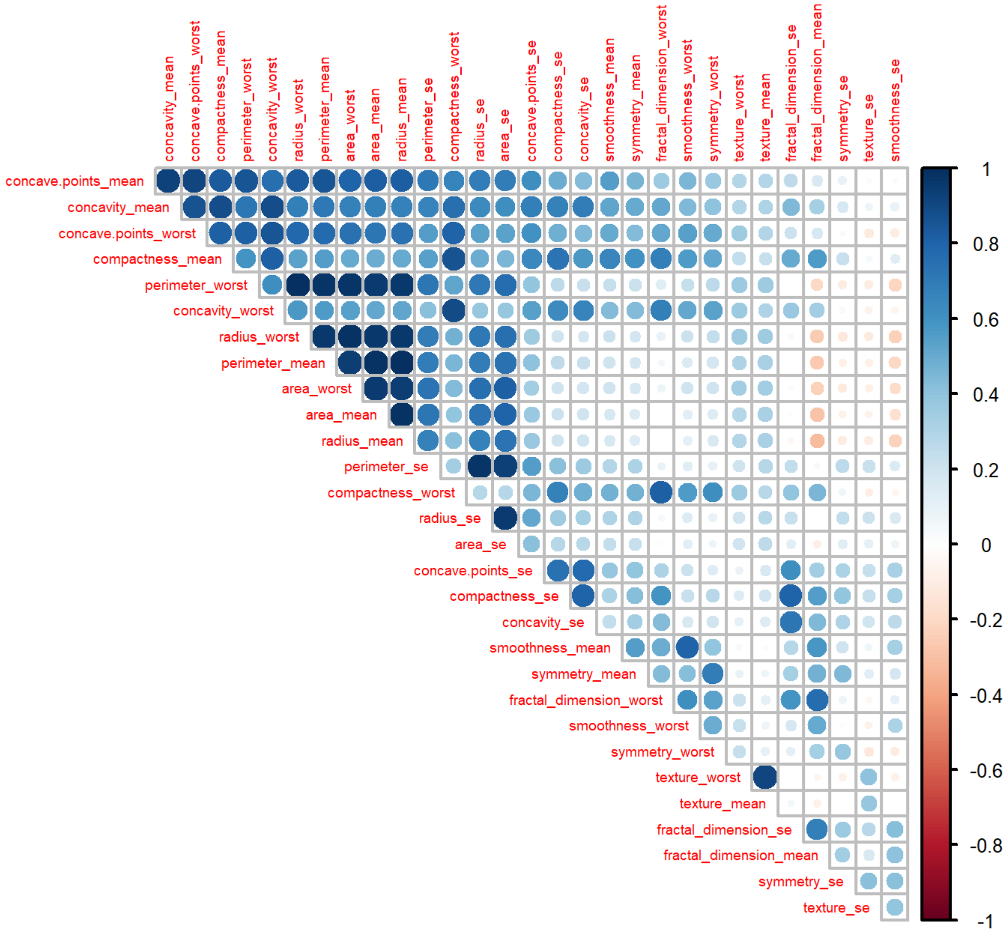

Figure 3.4: Features corrplot.

Then, the correlation between the features was calculated and the twelve most correlated features were removed to avoid bias in model training (figure 3.4).

3.3 Data preparation

Firstly, the entire data set was randomly divided to prepare data for train and testing: 70% of data are contained in *train_set* and the other 30% are contained in *test_set*. Secondly, to reduce the number of features a Primary Component Analysis (PCA) was made. In the function *prcomp()* was specified to scale and center data for better performance. Thirdly, the variance of each Primary Component (PC) was calculated and then a graph with the cumulative proportion of variance explained was produced (figure 3.5). Finally, eight of the eighteen PC were selected. These explain about 96% of the total variance.

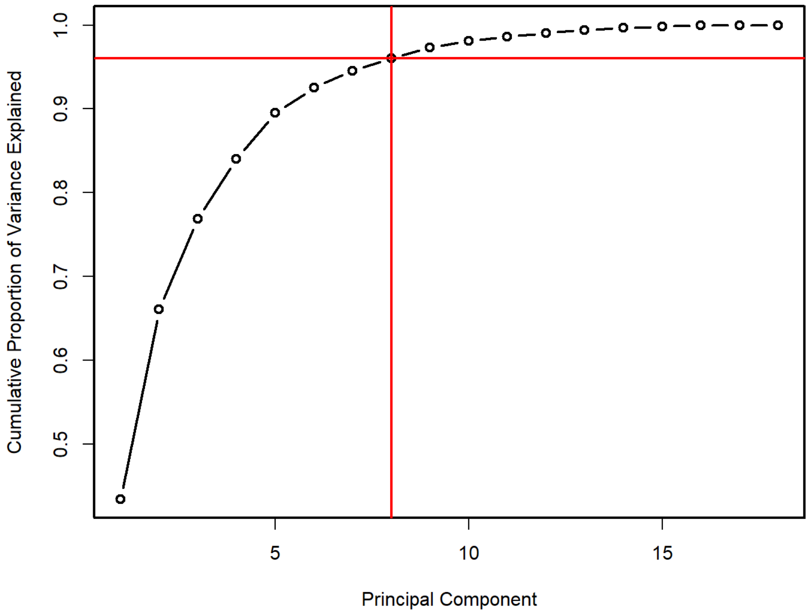

Figure 3.5: Cumulative Proportion of Variance Explained.

3.4 k-NN

3.4.1 Train model

The k-Nearest Neighborhood is a supervised learning classifier which uses proximity to make classifications. Therefore, the purpose is to find the optimal k value to maximize the accuracy. The model was trained several times varying the k value from 1 to 30. Taking into account the accuracy, the optimal k is $k = 14$ with $accuracy = 93.52\%$.

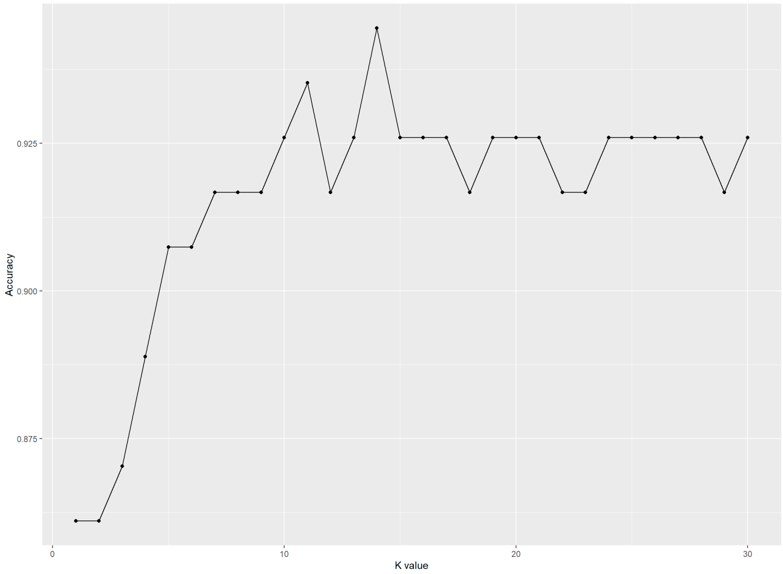

Figure 3.6: Optimal k-value.

After that, k-fold cross-validation was introduced. The number of folds is 5 and after an evaluation the new optimal k for k-NN is $k = 16$ with $accuracy = 94.58\%$.

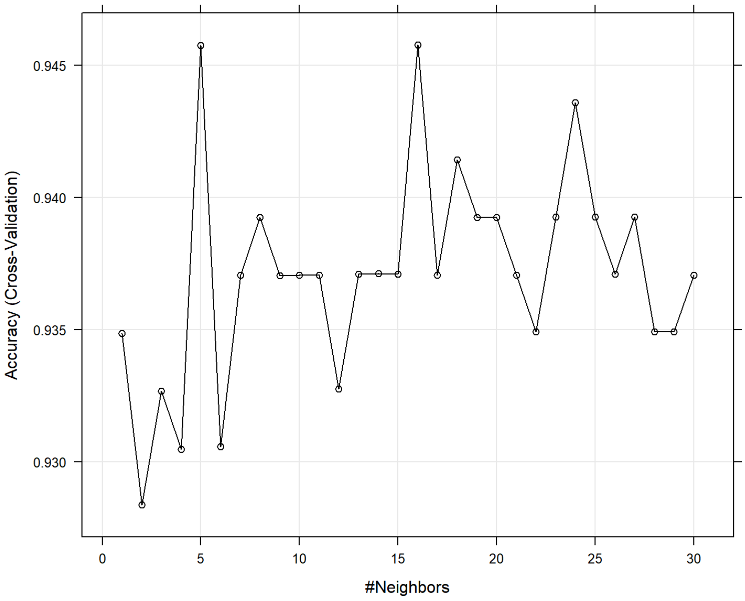

Figure 3.7: Optimal k-value (cross-validation).

3.4.2 Testing and evaluation

Both k-NN with and without cross-valitation were tested, and the following evaluations and confusion matrix are obtained (positive class = M):

No Cross-Validation

		Actual values		
		Positive	Negative	Total
Predicted values	Positive	35	5	40
	Negative	2	66	68
	Total	37	71	108

Cross-Validation

		Actual values		
		Positive	Negative	Total
Predicted values	Positive	33	4	37
	Negative	4	67	71
	Total	37	71	108

Metrics	No CV	CV
Accuracy	0.9352	0.9259
Prevalence	0.3426	0.3426
F-measure	0.9091	0.8919
ϕ	0.8604	0.8356
Precision	0.8750	0.8919
Recall	0.9459	0.8919
Jac	0.8333	0.8049
J	0.8755	0.8356
Mk	0.8456	0.8356

Table 3.1: Performance metrics k-NN

3.5 Classification tree

3.5.1 Train model

Classification tree learning is a supervised learning approach used as a predictive model to draw conclusions about a set of observations. The goal is to find the optimal size of the tree and by varying this hyperparameter it results that the optimal depth is 5 (figure 3.8).

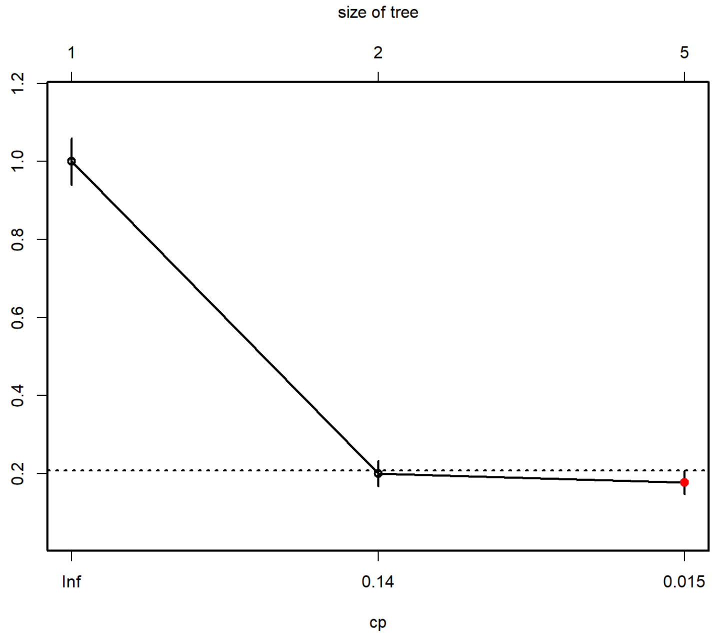

Figure 3.8: Optimal tree size.

After that, k-fold cross-validation was introduced. The number of folds was 5 and once the model has been trained what emerged is that both models have the same size and the same structure (figure 3.9).

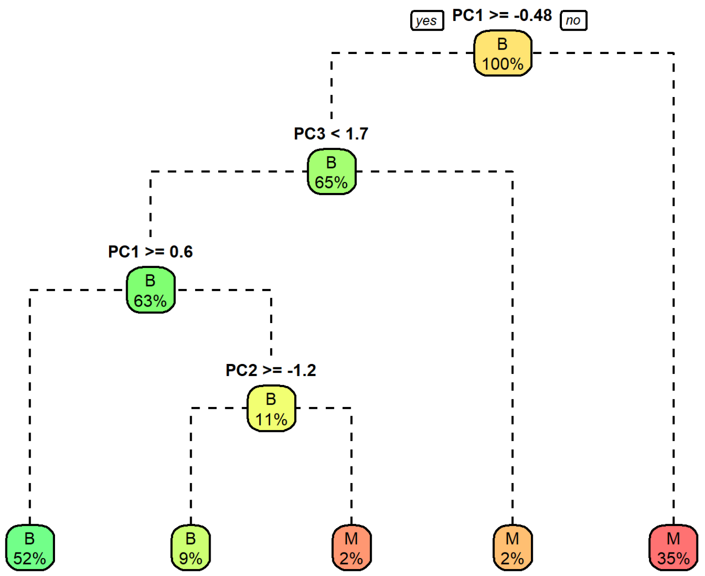

Figure 3.9: Classification tree.

3.5.2 Testing and evaluation

Both classification trees with and without cross-valitation were tested, and the following evaluations and confusion matrix were obtained (positive class = M):

No Cross-Validation

Actual values

		Positive	Negative	Total
Predicted values	Positive	35	4	39
	Negative	2	67	69
	Total	37	71	108

Cross-Validation

Actual values

		Positive	Negative	Total
Predicted values	Positive	35	4	39
	Negative	2	67	69
	Total	37	71	108

Metrics	No CV	CV
Accuracy	0.9444	0.9444
Prevalence	0.3426	0.3426
F-measure	0.9211	0.9211
ϕ	0.8790	0.8790
Precision	0.8974	0.8974
Recall	0.9459	0.9459
Jac	0.9024	0.9024
J	0.8896	0.8896
Mk	0.8685	0.8685

Table 3.2: Performance metrics classification tree

3.6 Random forest

3.6.1 Train model

Since in this context is better to not have false negatives, a random forest was implemented trying to maximize the recall. For the training was used k-fold cross-validation with 5 folds. Trying to maximize the recall, various model were trained varying the number of trees. The optimal model count 600 trees.

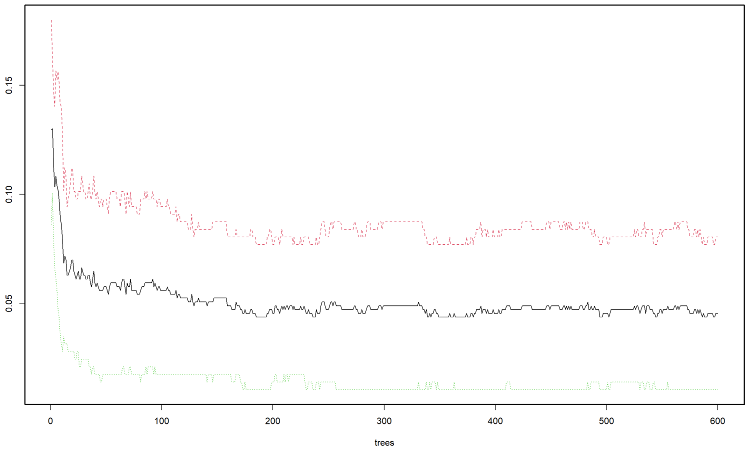

Figure 3.10: Error rate of random forest.

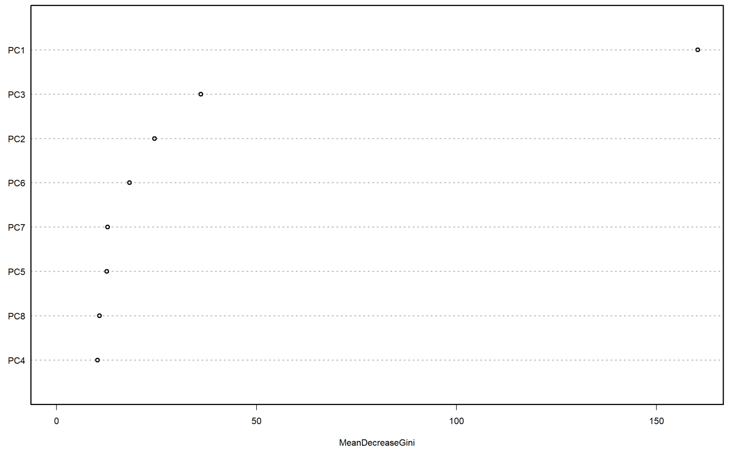

Figure 3.11: Variable importance.

3.6.2 Testing and evaluation

The random forest model were tested, and the following evaluations and confusion matrix are obtained (positive class = M):

No Cross-Validation

		Actual values		
		Positive	Negative	Total
Predicted values	Positive	37	7	44
	Negative	0	64	64
	Total	37	71	108

Metrics	RF
Accuracy	0.9352
Prevalence	0.3426
F-measure	0.9136
ϕ	0.8706
Precision	0.8409
Recall	1.0000
Jac	0.8409
J	0.9014
Mk	0.8409

Table 3.3: Performance metrics random forest.

It is important to highlight that there are zero false negatives but the number of false positives is greater compared to other models. However, in this context is better to be sure that who have cancer will be correctly classified. Therefore, it is possible to save many more lives.

3.7 Results

For comparison, only models trained with cross-validation are taken into account. The model that performs better is the classification tree, meanwhile, the worst is the k-NN model. However, as mentioned in chapter 3.6.2, it would be better to use the model trained with the random forest, as it has zero false negatives ($recall = 1.0$). It could save many more lives than other trained models.

Metrics	k-NN	Classification tree	Random Forest
Accuracy	0.9259	0.9444	0.9352
Prevalence	0.3426	0.3426	0.3426
F-measure	0.8919	0.9211	0.9136
ϕ	0.8356	0.8790	0.8706
Precision	0.8919	0.8974	0.8409
Recall	0.8919	0.9459	1.0000
Jac	0.8049	0.9024	0.8409
J	0.8356	0.8896	0.9014
Mk	0.8356	0.8685	0.8409

Table 3.4: Performance metrics comparison.

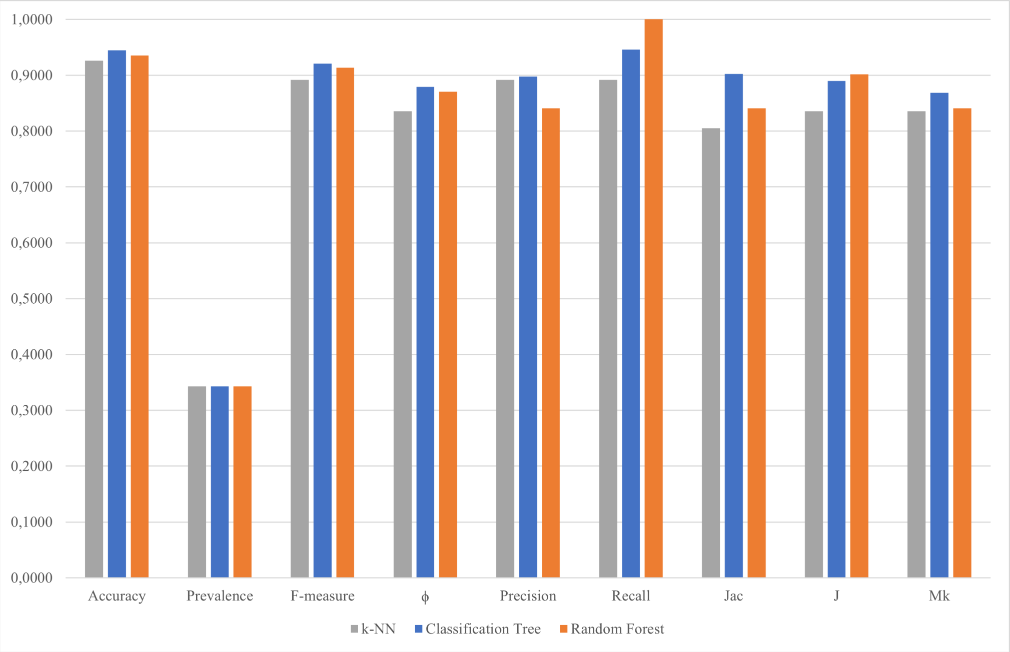

Figure 3.12: Comparison of models.

Bibliography

[1] *Project Repository*. URL: https://github.com/marcobogani/IntelligentSystemProject.