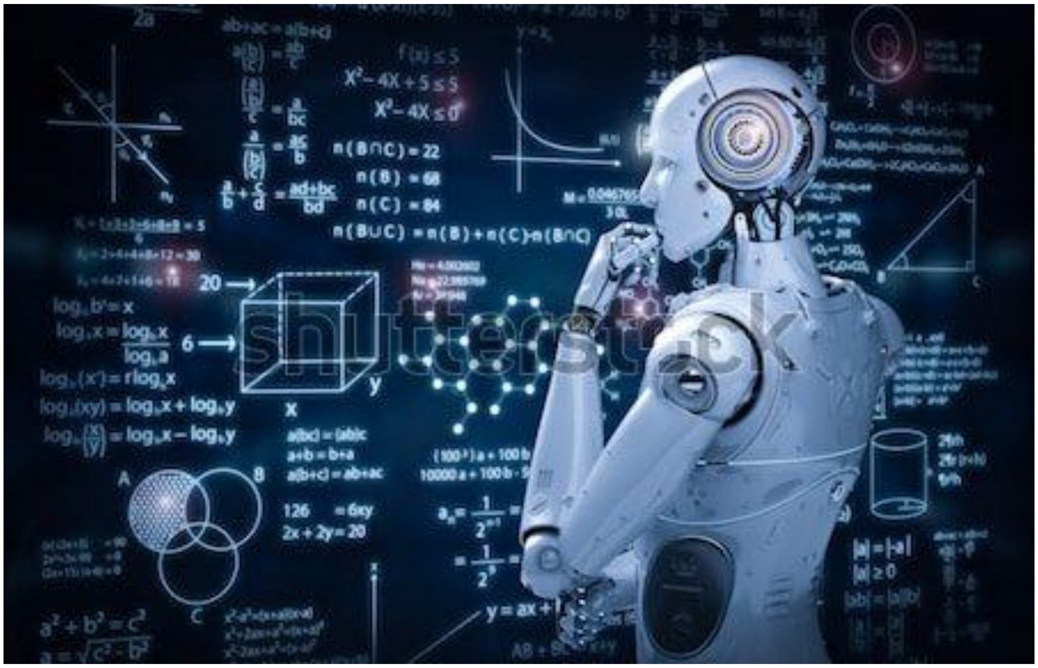
Statistical & Machine Learning 2021-2022

Minh PHAN



Individual Project

Sofie GHYSELS

Table of contents

Introduction

Finding and retaining the best employees is vital for the success of any business. A high employee churn rate is very costly for an organization of any size. If employees often leave the company this not only leads to a decrease in employee morale for the remaining employees, but it also lead to a decrease in productivity because quite simply there are less co-workers to do the job, most importantly a high employee churn rate results in higher recruitment, onboarding and training costs.

The employee churn rate is the percentage of employees leaving the company over some specified period. The turnover rate in the services industry in the Netherlands was 20% in 2021, it was 16% on average in the EU area and 25% in the US. It must be noted that I only look at the voluntary turnover rate.

The dataset provided is an HR dataset that consists of 14 999 observations and 11 variables or columns. The dependent variable is employee churn, or the column 'left' in the dataset. This values in this column are only 0 or 1, 0 for when an employee did not leave and 1 for when an employee left.

There are no missing values in the data. In HR Analytics, employee data is unlikely to feature large ratio of missing values as HR Departments typically have all personal and employment data on-file. However, a common issue in HR analytics is that the type of documentation data is being kept in (often Excel or other specific HR databases) have a massive impact on the accuracy and the ease of access to HR data.

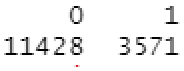
The goals of this project are:

- To identify key factors related to employee churn
- To create models that accurately predict employees who leave

Data exploration

Let’s first do some data exploration to gain some insights into the dataset.

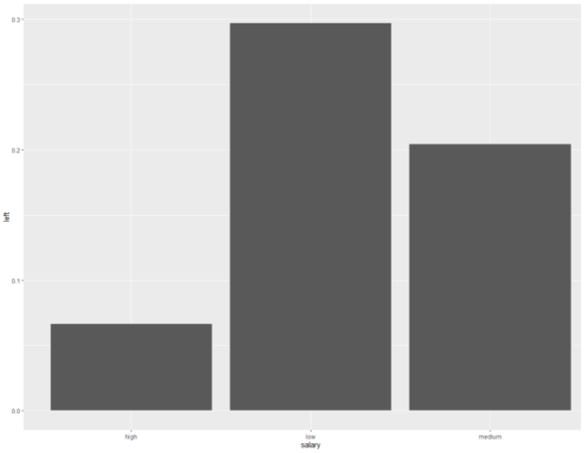
The first step is to calculate the overall employee churn rate. This is put into the following table, which shows that of the 14 999 employees 11 428 did not leave the company and 3 571 did leave the company. This means that the overall employee churn rate is 23.81%



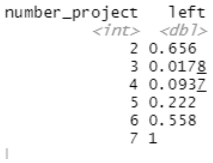
```
    0     1
11428  3571
```

As seen in the graph on the right, some levels of salary have a higher impact on employee churn.

No big surprises here, when the employee has a low salary level, he or she has a higher probability of leaving the company.



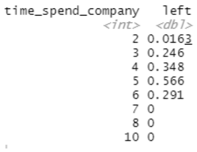
In general, the lower the satisfaction level, the higher the probability is that an employee will leave the company. The result of the code for this is given in R in a tibble, which is a dataframe that sums up all the satisfaction levels and the average of people that left the company.



```
number_project   left
         <int>  <dbl>
             2 0.656
             3 0.0178
             4 0.0937
             5 0.222
             6 0.558
             7 1
```

There is however some fluctuation in the number of projects and the average number of people leaving the company.
The ideal number of projects seems to be 3.
There is a higher chance for employee churn at 2 and 6 and 7 number of projects.

It is interesting that when we look at the amount of time in years an employee spend at the company and the average of leaving, we see that the number goes up until 5 years but then goes down. After 5 years of service at a company people tend to be less likely to leave.



```
time_spend_company   left
             <int>  <dbl>
                 2 0.0163
                 3 0.246
                 4 0.348
                 5 0.566
                 6 0.291
                 7 0
                 8 0
                10 0
```

Information value and splitting the data

Before building our models, it is common to perform variable screening, with for instance the IV function. Information value or IV is a technique to select important variables in a predictive model. It helps ranking variables on the order of their importance. IV is calculated with the following formula:

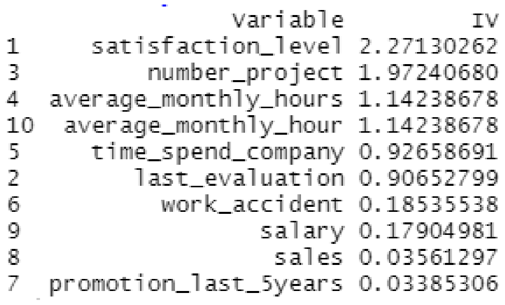
$$IV = \sum (\% \text{ of non-events} - \% \text{ of events}) * WOE$$

The purpose of IV is not the same as multiple-variable selection techniques such as lasso, where the variables that go into the final model are selected. IV is designed to ensure that the approaches deployed during the final modeling phases are set up for success.

The total IV of a variable is the sum of IVs of its categories. In general, if it is low, then it may not be a good explainer for the DV or dependent variable. In general we have the following rating categories:

- < 0.02 = this predictor is not useful for modeling
- 0.01 to 0.02 = this predictor has a weak relationship
- to 0.03 = this predictor has a medium strength relationship
- to 0.05 = the predictor has a strong relationship
- > 0.05 = there is a very high relationship

In the specific case, we can see that there are 2 weak variables: work_accident and salary. There are 2 variables that have a medium relationship: sales and promotion_last_5_years. The other variables have a very high relationship with the DV.



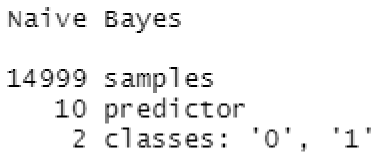
```
                Variable         IV
1     satisfaction_level 2.27130262
3         number_project 1.97240680
4   average_monthly_hours 1.14238678
10   average_monthly_hour 1.14238678
5     time_spend_company 0.92658691
2        last_evaluation 0.90652799
6          work_accident 0.18535538
9                 salary 0.17904981
8                  sales 0.03561297
7  promotion_last_5years 0.03385306
```

Afterwards, I split the dataset into a train (70%) and test (30%) dataset. Both datasets contain 11 columns. The train dataset has 10 499 rows, and the test dataset has 4 500 rows. The train dataset is used to fit the machine learning model for that reason it is the largest corpus whereas the test dataset is used to evaluate the fitted machine learning model.

K-fold Cross-validation

I perform k-fold cross-validation to evaluate a machine learning model. Validation divides the data into folds and ensures that each fold is used as a testing set at some point. This is different in the train and test set splitting because there the accuracy obtained for one test set can be different to the accuracy obtained for a different set. The parameter K refers to the number of different subsets that the given data set is to be split into. For instance, with a 5-fold cross-validation, the data is split into 5 folds. In the first iteration, the first fold is used to test the model and the rest are used to train the model. In the second iteration, 2nd fold is used as the testing set while the rest serve as the training set. This process is repeated until each fold of the 5 folds have been used as the testing set. In sum cross-validation, is used to estimate the skill of a machine learning model on unseen data. It uses a limited sample to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

I used the naïve bayes model in the k-fold cross-validation. Naive Bayes is a probabilistic machine learning algorithm based on the Bayes Theorem, used in a wide variety of classification tasks. Naïve bayes is especially useful for real-time predictions. In this case there are two classes: 0 (employee did not leave) and 1 (employee left).

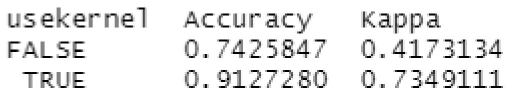


```
Naive Bayes

14999 samples
   10 predictor
    2 classes: '0', '1'
```

The main advantages of k-fold cross-validation is that it is easy to implement, and it generally has a lower bias than other methods.

The usekernel is the parameter and if it is true the model has an accuracy of 0.91, which is very good.



```
usekernel  Accuracy   Kappa
FALSE      0.7425847  0.4173134
 TRUE      0.9127280  0.7349111
```

Applying machine learning algorithms

Multiple logistic regression

Multiple logistic regression is one of the most popular and widely used statistical tests. The variable you want to predict should be binary (it has only 2 possible values so for instance 0 or 1, true or false etc.). The outcome of the multiple logistic regression is dichotomous, meaning it either occurred or it didn't, or in this specific case: the employee either left the company 1 or did not leave the company 0.

It is best to use the multiple logistic regression when the following conditions are met:

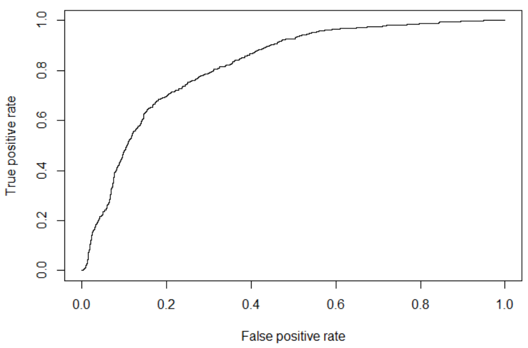
- you want to use one or more variables in a prediction of another
- the variable you want to predict (so the dependent variable) is binary
- the dataset consists of one or more independent variables (which are variables that you are using as a predictor)

The dataset itself must also satisfy the following properties before applying multiple logistic regression:

- linearity: logistic regression assumes that the natural log of the probabilities/odds and the predictor variable is linear
- no outliers: logistic regression is very sensitive to outliers (data points that have unusually large or small values)
- independence: each of your observations should not depend on any of the others
- no multicollinearity: when two or more independent variables are highly correlated the regression analysis could become unstable.

When we run the multiplelog function in R we get coefficients and p-values for each term in the model. So, we fitted the model on the train dataset, created earlier. A p-value less than or equal to 0.05 means that the result is statistically significant, and we can trust that the difference is not due to chance alone. In our case the most significant independent variables are satisfaction_level, last_evaluation, number_project, average_monthly_hours, time_spend_company, work_accident, promotion_last_5years, salarylow and salarymedium.

The final essential task in machine learning is performance measurement. I evaluate the multiplelog model and make predictions based on the test set. To evaluate the model, I use AUC. AUC or area under the curve, represents the degree or measure of separability, it tells how much the model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting 0 classes as 0 and 1 classes as 1. The AUC is 82.21%, which means that the model correctly identifies 82.81 % of the entire employees on the dependent variable. This is also displayed in the plot below, where we can see that the peak is at 82%.



Random forest

Random forest is a supervised and flexible machine learning algorithm, which combines the output of multiple decision trees to reach a single result. It is supervised because it uses labeled datasets to train algorithms to classify or predict outcomes accurately. Therefore, random forest can be used for both classification and regression problems. It randomly samples data, build decision trees on those different samples and builds a decision tree for the samples. This process is repeated until many trees are generated.

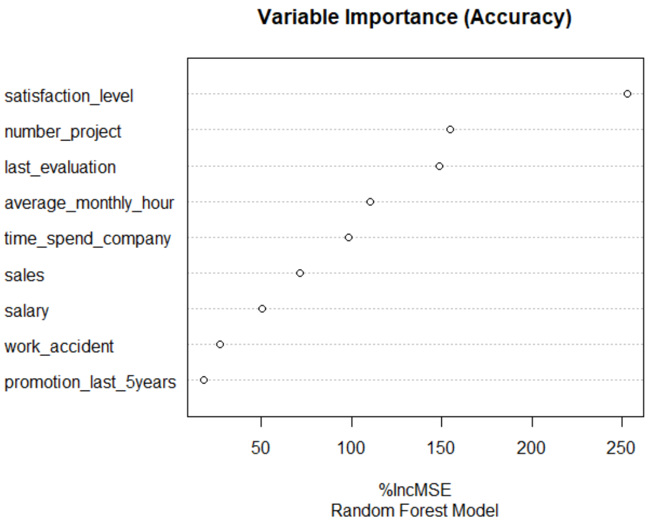
The random forest model uses the bagging or bootstrap aggregation ensemble technique. Ensemble techniques are made up of a set of classifiers, or decision trees, and their predictions are aggregated to identify the most popular result. The first step is when the bagging chooses a random sample from the data set. Each model is generated from the samples provided by the original data with replacement known as row sampling, which is called bootstrap. Each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

The advantages of random forest are:

- It is very flexible and can be used for both classification and regression problems
- It solves the problem of overfitting because the output is based on majority voting
- Because each tree does not consider all the attributes, feature space is reduced, and the model is immune to dimensionality
- There is no need to segregate the data into train and test because there will always be 30% of the data which is not seen by the decision tree made from bootstrap.

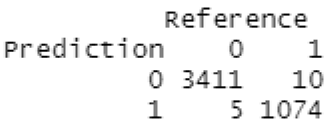
The main disadvantage of the random forest model is that it is a highly complexed model in comparison to decision trees where you can simply follow the path of the tree.

Before applying the code for the random forest on the dataset I select the variables for the model to predict ‘left’ and put them in new variable “res_vars”. The number of trees generated, ntree, is 500 in this example. I then compute the variable importance:



This shows us which independent variables are highly important for predicting the dependent variable, which is left. In this case we can see that satisfaction_level, number_project and last_evaluation are very important variables.

The real test of a model's success is its performance on the test dataset. To have a look at the performance of this random forest model, I compute a confusion matrix. A confusion matrix is a technique for summarizing the performance of a classification algorithm. It gives the number of correct and incorrect predictions broken down by each class.



```
          Reference
Prediction    0    1
         0 3411   10
         1    5 1074
```

In total there are 3411 true negative, which means that the model correctly predicts that 3411 people as not leaving the company. There are 1074 true positive, which means that the model correctly predicts that 1074 are leaving the company. This model does a fairly good job as the error rate is very small. The error rate on class 0 is 0.14% and on class 1 is 0.93%.

Gradient boosting

Boosting is an ensemble technique in which the predictors are not made independently, but sequentially. At each iteration, a new weak, base-learner model is trained with respect to the error of the whole ensemble learnt so far. When combined, these many weak successive trees produce a powerful “committee” that are often hard to beat with other algorithms. The main idea is to leverage the patterns in residuals and strengthen a model with weak predictions and make it better. Once the stage is reached that residuals do not have any pattern that could be modeled, we can stop modeling residuals (otherwise it might lead to overfitting). Algorithmically, we are minimizing our loss function, such that test loss reach its minima.

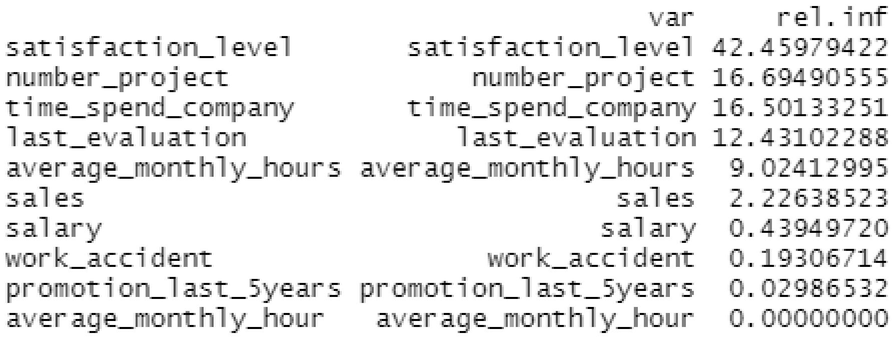
The main advantages of the gradient boosting model are:

- It can deal with large and complex data
- Often produces a very high accuracy score
- No data pre-processing required
- Handles missing data

The main disadvantages are:

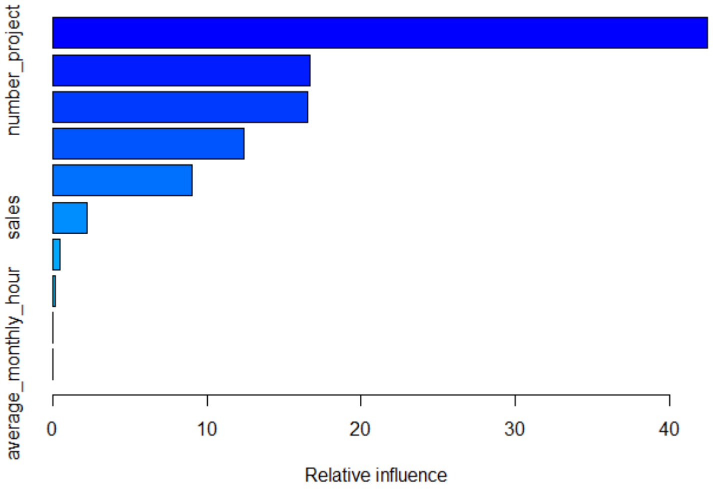
- Possibility of overemphasizing outliers and cause overfitting
- Can be memory exhaustive because it often require many trees

We want to understand which variables have the largest influence on whether an employee will leave or not. The summary method for gbm outputs the following data frame and plot that shows the most influential variables:



```
                                        var     rel.inf
satisfaction_level       satisfaction_level 42.45979422
number_project               number_project 16.69490555
time_spend_company       time_spend_company 16.50133251
last_evaluation             last_evaluation 12.43102288
average_monthly_hours average_monthly_hours  9.02412995
sales                                 sales  2.22638523
salary                               salary  0.43949720
work_accident                 work_accident  0.19306714
promotion_last_5years promotion_last_5years  0.02986532
average_monthly_hour   average_monthly_hour  0.00000000
```

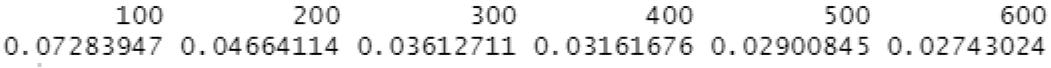
We can see again that satisfaction_level, number_project and time_spend_company are highly influential independent variables.



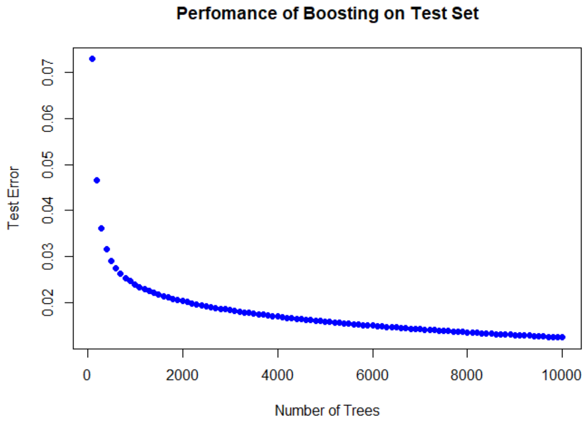
Although the names of all the IV's or independent variables are not displayed, we can also see from the plot that the same IVs are highly influential.

Lastly, we perform a prediction matrix on the train dataset and compute the AUC score. Again, we have a very good AUC of 0.9664676.

We end by calculating the RMSE or root mean square error which allows us to measure how far the predicted values are form the observed values. The smaller the RMSE it indicated the better the model is performing. Both the matrix and the plot shows that the higher the number of trees in the model, the better the model is performing.



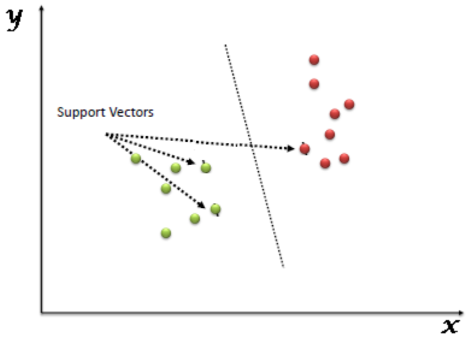
```
       100        200        300        400        500        600 
0.07283947 0.04664114 0.03612711 0.03161676 0.02900845 0.02743024 
```



Support Vector Machine

In essence, support vector machines or SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes. A hyperplane is a line that linearly separates and classifies a set of data. The further the data points lie form the hyperplane, the more confident we are that they have been correctly classified. Therefore, the data points have to be as far away form the hyperplane as possible, while still being on the correct side of it. The support vectors are simply the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-plane/ line).

In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).



Below are the advantages of SVM:

- It is a very accurate and efficient model because it uses a subset of training points
- SVM works very well on smaller datasets

The disadvantages of SVM are:

- SVM is not suited for larger datasets as the trianing time with SVMs can be high
- SVM is not that effective on datasets with overlapping classes

We fit svm on the training dataset and can see how well our model fits the data by usin the the predict() command.

Sources

PHAN, M. (2022). Statistical and Machine Learning Approaches for Marketing. [Course]. Lille: IESEG Management School. MSc in Big Data Analytics.

ANURAG, G. & ABHISHEK, T., 'Human Resources Analytics: Predicting Employee Churn in R', internet, DataCamp, s.a., (https://www.datacamp.com/courses/human-resources-analytics-predicting-employee-churn-in-r).

GIERLAK, A., 'Classification example - HR data', internet, cran.r-project, 08-02-2022, (https://cran.r-project.org/web/packages/rSAFE/vignettes/example_hr.html).

CAUGHLIN, D., 'Chapter 28 Aggregating & Segmenting Employee Survey Data', internet, RforHR, 07-03-2022, (https://rforhr.com/aggregatesegment.html).

VAN VULPEN, E., '7 HR Data Sets for People Analytics', internet, AIHR, s.a., (https://www.aihr.com/blog/hr-data-sets-people-analytics/).

SMITH, G. 'Employee retention: The real cost of losing an employee', internet, Peoplekeep, 17-09-2021, (https://www.peoplekeep.com/blog/employee-retention-the-real-cost-of-losing-an-employee).

SARASWAT, M., 'Practical Guide to Logistic Regression Analysis in R', internet, Hackerearth, s.a., (https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/logistic-regression-analysis-r/tutorial/).

RICHMOND, B., 'Predicting Employee Turnover using R', internet, Medium, 24-10-2018, (https://medium.com/@brian.richmond/predicting-employee-turnover-using-r-dd7cb3136704).

SRUTHI, E., 'Understanding Random Forest', internet, AnalyticsVidhya, 17-06-2021, (https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/#:~:text=Random%20forest%20is%20a%20Supervised,average%20in%20case%20of%20regression.).

ANONYMOUS, 'Random Forest', internet, IBM, 07-12-2020, (https://www.ibm.com/cloud/learn/random-forest).

BROWNLEE, J., 'What is a Confusion Matrix in Machine Learning', internet, Machine Learning Mastery, 18-08-2020, (https://machinelearningmastery.com/confusion-matrix-machine-learning/).

WALIA, A., 'Gradient boosting in R', internet, R bloggers, 24-09-2017, (https://www.r-bloggers.com/2017/08/gradient-boosting-in-r/).

GROVER, P., 'Gradient Boosting from scratch', internet, ML Review, 09-12-2017, (https://blog.mlreview.com/gradient-boosting-from-scratch-1e317ae4587d).

ANONYMOUS, 'Gradient Boosting Machines', internet, UC R, s.a., (http://uc-r.github.io/gbm_regression).

ANONYMOUS, 'Support Vector Machine', internet, UC R, s.a., (http://uc-r.github.io/svm).

SUNIL, 'Understanding Support Vector Machine(SVM) algorithm from examples (along with code)', internet, AnalyticsVidhya, 13-09-2017, (https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/).

ANONYMOUS, 'K-fold Cross Validation in R Programming', internet, Geeks for Geeks, s.a., (https://www.geeksforgeeks.org/k-fold-cross-validation-in-r-programming/).

KRISHNI, 'K-Fold Cross Validation', internet, Medium, 16-12-2018, (https://medium.datadriveninvestor.com/k-fold-cross-validation-6b8518070833).

PRABHAKARAN, S., 'InformationValue', R statistics, s.a., (http://r-statistics.co/Information-Value-With-R.html).