
# Machine Learning (Stirling University) – November 2022

**Course Title**:
**Assignment No**:
**Due Date:**
**Submitted To**:

**Submitted By:**
**Student Name**:
**Student Number**:

# 1.0 Introduction

Our company, XYZ Cop, is a data analytics consulting company that helps clients analyze their business needs. Recently, we completed a project for a client, and in this paper, we are going to present the findings and analysis of using machine learning to forecast the performance of stores based on a variety of variables.

This project was requested by the investor of a chain with over 100 stores in the United Kingdom, who is seeking to expand his brand within the country. The goal was to make logistic regression models, decision trees, and neural network models that could accurately predict how well a new store would do based on the owner's descriptive data. The project's data was supplied in the form of a CSV file and included information on 136 shops, such as the town, store ID, staff numbers, floor space, demographic score, and others. To accomplish this objective, we utilized the Cross-Industry Standard Process for data mining framework to guide the data mining process, from issue identification and data analysis to model construction and effectiveness evaluation.

At the beginning of this assignment, we will discuss the data and features needed for analysis. Then, we will present the findings of our study, including the performance of each model, and make suggestions as to which model was best suited for that task. At the end of the study, we will give a summary of our findings and talk about the study's limitations and possible next steps.

## 1.1 CRISP-DM

As one of our business rules, we follow the Cross-Industry Standard Process for Data Mining (CRISP-DM) procedure. CRISP-DM is a popular data mining and machine learning project methodology. It is a structured process that helps practitioners through the various phases of a project, from identifying the business problem and collecting data to developing and evaluating models and delivering the final solution (Wirth, 2000). The technique is intended to be flexible and adaptable to various project types, and it gives a clear framework for organizing and implementing a data mining project. CRISP-DM includes six major phases (Schröer, 2021):

- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation
- Deployment

Each phase provides a series of tasks to be accomplished before proceeding to the next phase. By following CRISP-DM, project managers can be sure that their projects are well-structured and include all important factors.

# 2.0 Business Understanding

Business understanding is the initial step of the CRISP-DM framework (Schröer, 2021), which entails developing a full understanding of the problem at hand, the project's goals, and the environment in which the project is being carried out. During this phase, it is important to clearly explain the problem, list the available information, and describe the project's goals and needs.

## 2.1 Understand the Data

The obtained data for this project was recorded in a CSV file with the name "storedata.csv." This document contained 136 records, each of which represented the store and its associated attributes. The variables for each store included:

- Town: The town in which the store is located.
- Country: The country in which the store is located.
- Store ID: A unique identifier for each store.
- Manager name: The name of the manager of the store.
- Staff numbers: The number of staff members working at the store.
- Floor Space and Window Space: The floor and window space of the store, respectively.
- Car park: Whether or not the store has a parking lot
- Demographic score: A score reflecting the demographic characteristics of the surrounding area.
- Location: The location of the store.
- 40 min, 30 min, 20 min, and 10 min drive time population size: The population size is within 40 minutes, 30 minutes, 20 minutes, and 10 minutes' drive from the store, respectively.
- Store age: The age of the store.
- Clearance space in store: The clearance space available in the store.
- Competition number: The number of competing stores near the store.
- Competition score: The score of the competing stores near the store.
- Performance: Whether or not the store owner considers the store's revenue growth to be "good" or "bad."

The requirements of completed project required the creation of a predictive model capable of reliably predicting the performance of the shop based on its features. The goal

of the model is to aid in the selection of new store locations by estimating the location's performance based on its features.

Since the target variable in this instance is the store's performance, which is either "good" or "bad," and not a continuous number, we have determined that the project is a classification problem. A classification is a form of supervised machine learning in which the objective is to forecast the category label for a provided set of input data (Kotsiantis, 2007). It is used to predict one of several possible outcomes when the target variable is categorical.

## 2.2 Data Mining

We would propose data mining as one of the finest techniques since it provides a scientific method for analyzing large volumes of data and extracting important, meaningful patterns. Data mining techniques are a quick and efficient method for discovering relationships and patterns in massive amounts of data, which would be difficult to examine by hand as the data volume increases.

By utilizing data mining, we can construct a model that can generalize to new data, which is essential for our purpose as the goal is to forecast the performance of unopened stores. Also, data mining gives you the means to test and evaluate the models to make sure they are accurate and reliable, which is important in a business setting where decisions are based on the results.

In conclusion, data mining is a fast and effective way to analyze large amounts of data, find patterns and correlations, and build predictive models that can be used to help companies make decisions.

# 3.0 Data Summary

Data summarization is the crucial phase in the CRISP-DM process since it enables us to comprehend the features of the dataset we are analyzing. By summarizing the variables and their distributions, we can detect any possible flaws or biases in the data and make meaningful selections about which variables would be most beneficial for constructing our prediction model. In this report, we'll give an overview of the variables found in the data file, including whether they should be treated as categorical or numeric variables and whether they're likely to be used in building the model.

## 3.1 Remained Variables

The variables in the dataset have been thoroughly analyzed and appraised for their possible utility in building the predictive model for store performance. Our team has identified the following variables as potentially significant to the modeling process:

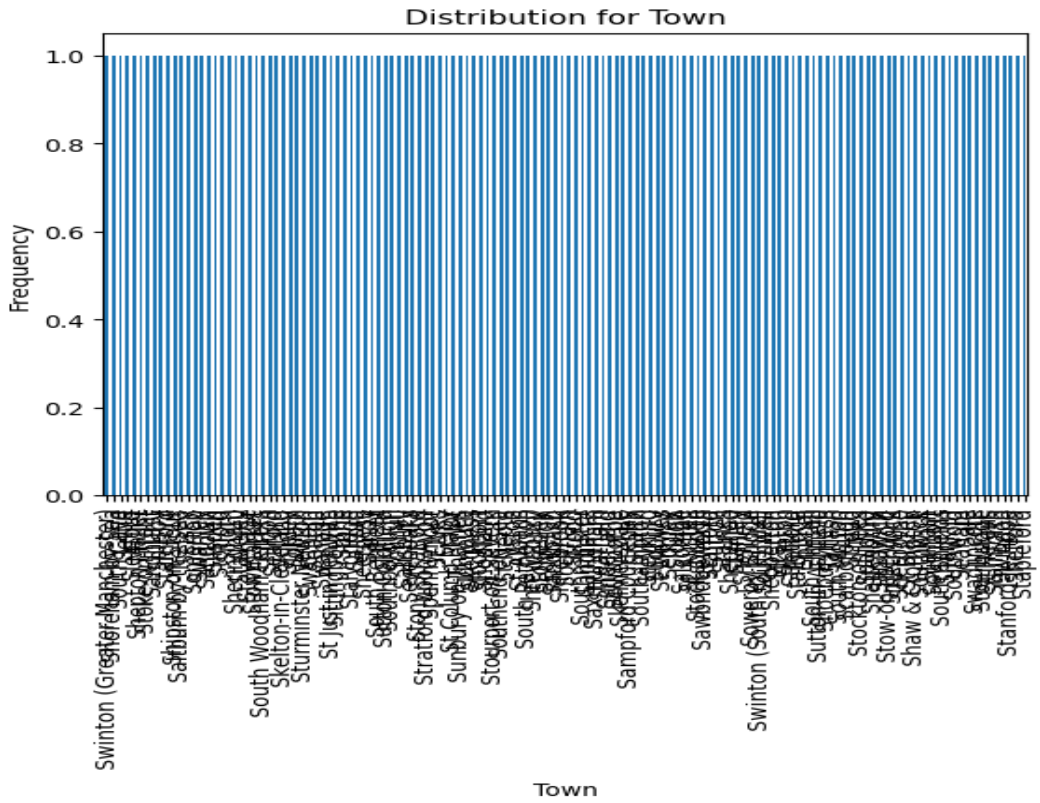
- Staff numbers – numeric, ordinal, and discrete
- Floor Space – numerical, discrete
- Window Space – numerical, discrete, and ordinal
- Car park – categorical, nominal
- Demographic Score – numeric, continuous, and ordinal
- Location – categorical, nominal
- 40 min, 30 min, 20 min, and 10 min drive time population size – numeric, continuous
- Store age – numeric, continuous, and ordinal
- Clearance space in store – numeric, discrete
- Competition number – numeric, continuous
- Competition Score – numeric, continuous
- Performance – nominal, categorical

## 3.2 Excluded Variables

Based on our investigation, our team has concluded that the following variables would be unlikely to contribute significantly to the modeling process and will thus be excluded.

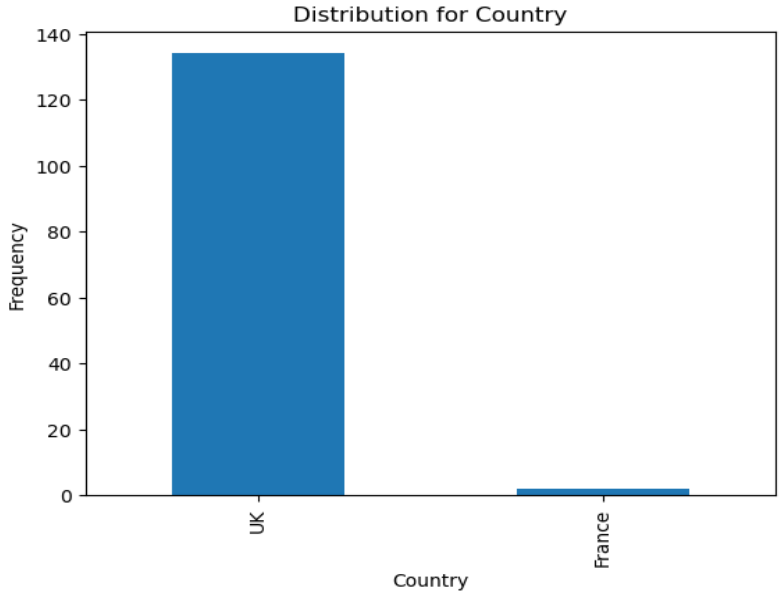
### 3.2.1 Town

This variable should be considered a categorical, nominal variable, and it will likely not be used in model development. Since the Panda’s data description and histogram indicated that there were 136 unique towns and that each town appeared just once, adding "town" as one of our variables would not provide us with any useful information because we cannot aggregate the data.




### 3.2.2 Country

This variable should be handled as a categorical, nominal variable. We will not use this variable but instead attempt to clean the data, as we are only interested in stores within the United Kingdom. By examining the histogram, we can see that the data contains some France-based stores, which will be removed throughout the data cleaning process. We see no reason to keep this data after data cleansing has been done.



### 3.2.3 Store ID and Manager Name

The store ID variable should be regarded as numeric and discrete, and it would be unlikely to be integrated into the model. This is due to the fact that the store ID is a unique identifier for every store and is unlikely to impact its performance.

On the other hand, the variable for the manager’s name should be considered nominal and categorical and is also unlikely to be useful in constructing the model. This is because it is not expected that the manager's name will have a substantial impact on the store's performance.

# 4.0 Data preparation

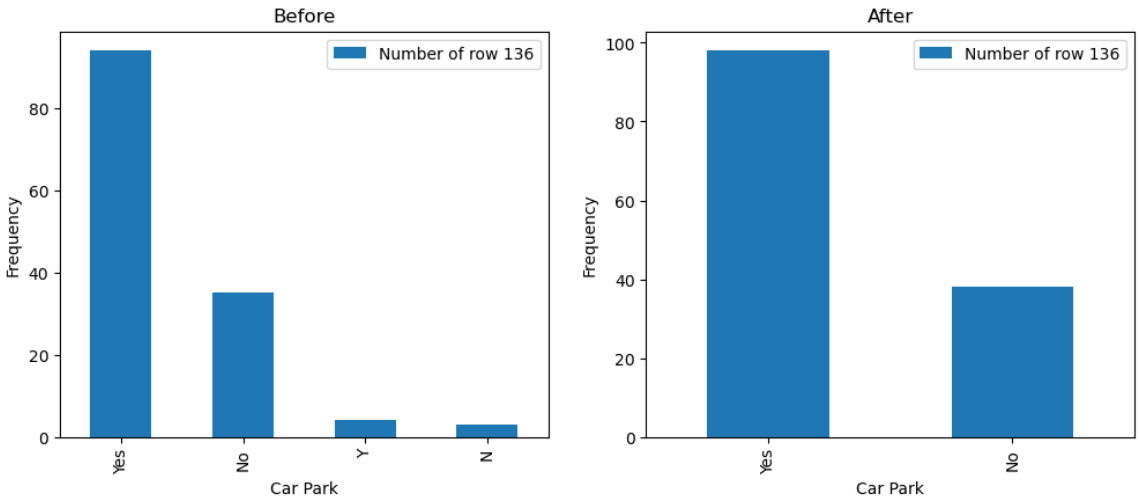
Prior to beginning the modeling process, data preparation is an important and necessary step. This procedure includes a number of processes designed to guarantee that the data are well organized, of high quality, and acceptable for modeling. Failure to appropriately prepare the data may result in incorrect and erroneous modeling results.

## 4.1 Data Cleaning

This step involved the detection and correction of missing or incorrect values. This phase aids in ensuring that the data is consistent and error-free. A histogram was created to examine how the data differed before and after the cleaning process.

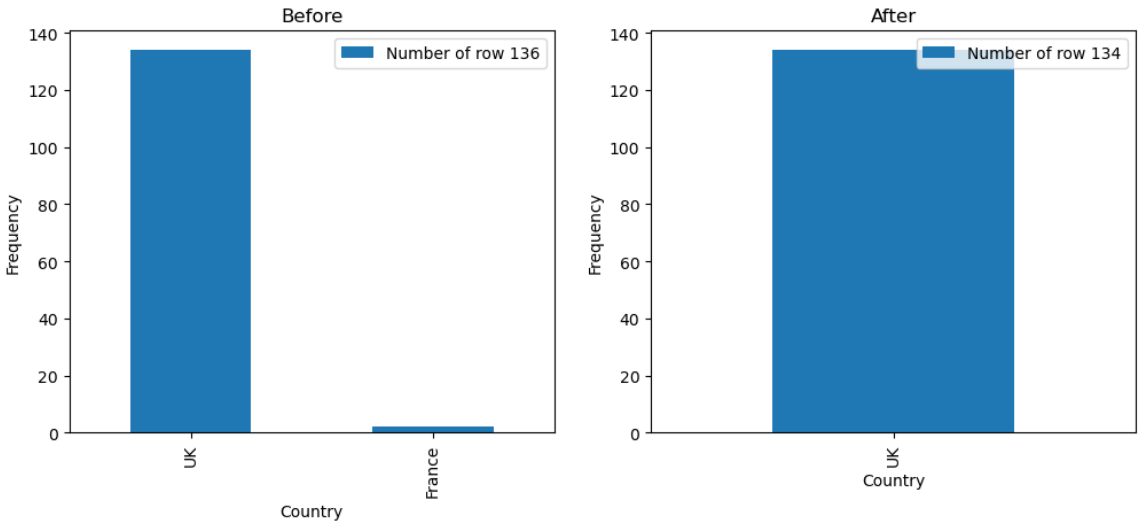
### 4.1.1 Car Park

There were anomalies in the way values were recorded for the “car park” data column, with some values reported as "Yes", "No", "Y", and "N". Our team opted to standardize the values by retaining simply "Yes" and "No".



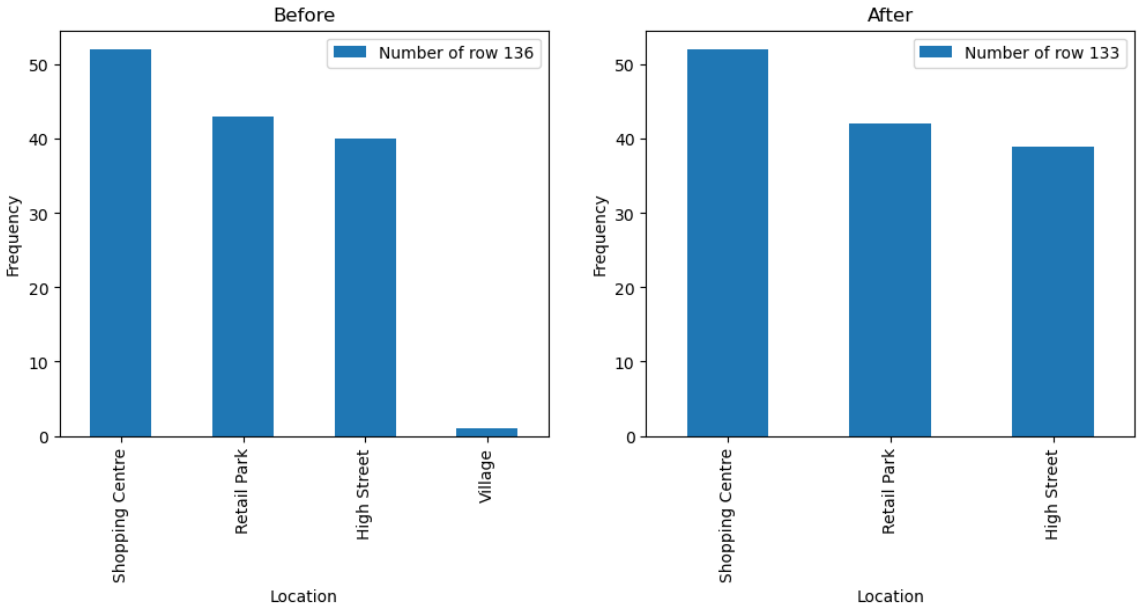

### 4.1.2 Country

Our team worked to clean up the information in the country column, getting rid of French store information that had been added by mistake.



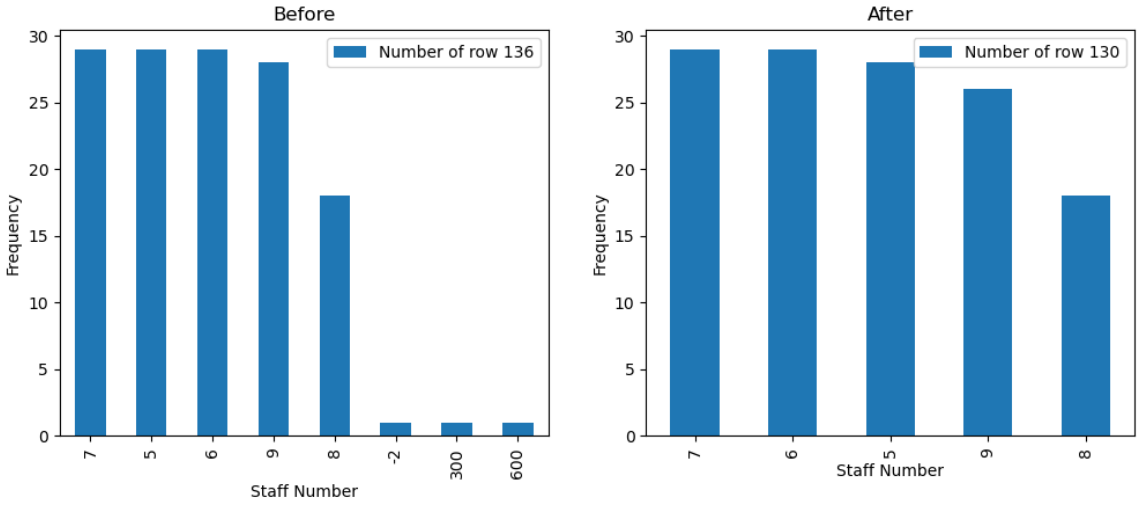

### 4.1.3 Location

After looking at the location data column in the dataset, we decided that shops in rural areas needed to be removed.




### 4.1.4 Staff number

Regarding the staff number column, there were a number of outliers, including -2, 300, and 600. It is impossible to have a negative number of staff, and staff numbers exceeding 100 are likely to be errors or outliers, therefore, we opted to eliminate them during data cleaning.



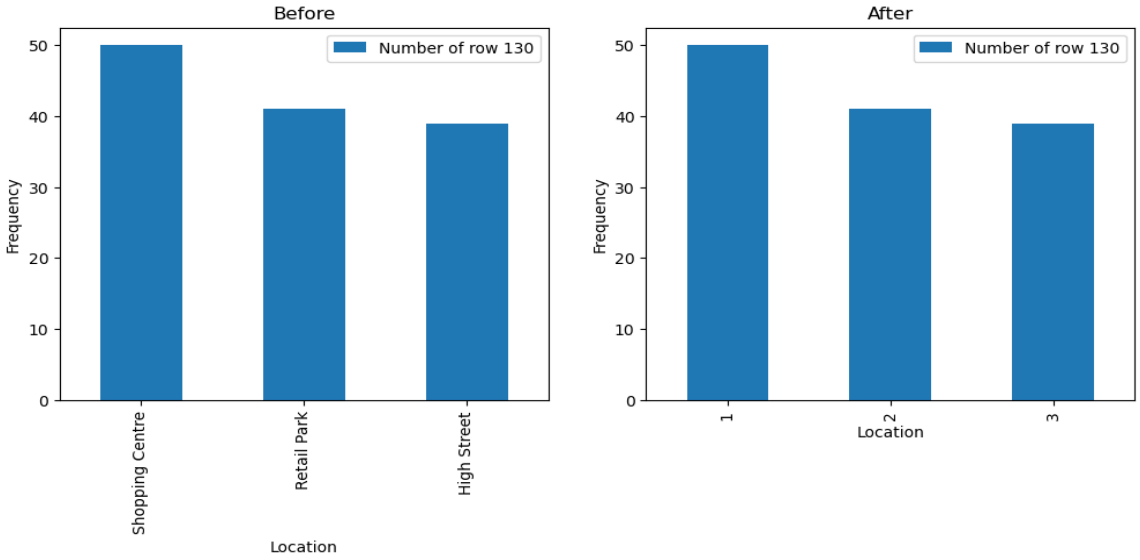

## 4.2 Data transformation

This stage consisted of converting variables from one data type to another to guarantee that all variables were in the proper format for modeling. For instance, transforming categorical variables to numeric variables so they can be utilized in modeling.

### 4.2.1 Location

To make nominal data more relevant, we must conduct a transformation on the location data, which is now divided into three distinct groups. In this step, the categorical data are turned into numbers, which can give the modeling process more information and insights.

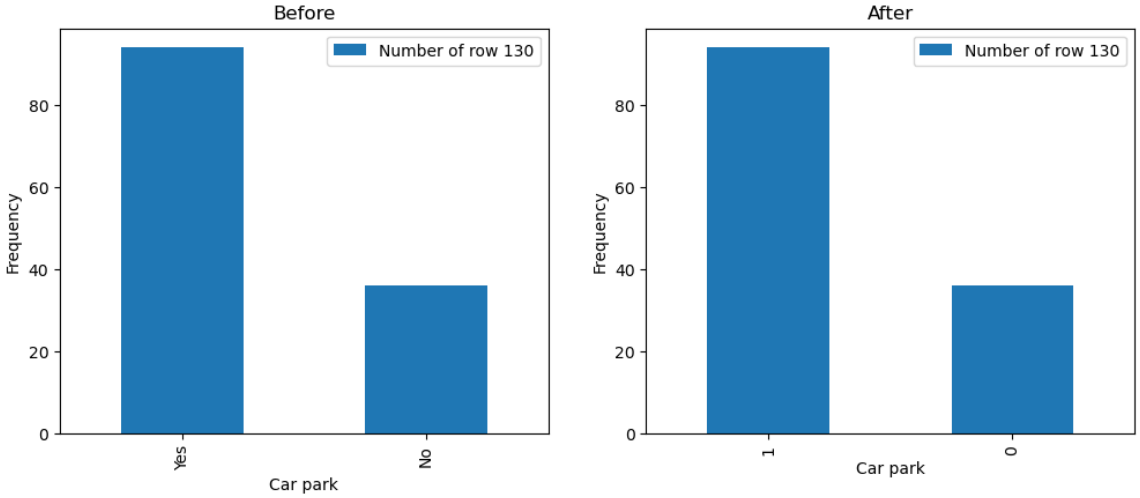
- Shopping Center will be replaced by "1".
- Retail Park will be replaced by “2”.
- High Street will be replaced by “3”.



### 4.2.2 Car Park

The car park data field had to be converted to a numerical representation before it could be utilized effectively in the modeling procedure. This section will involve the following transformations:

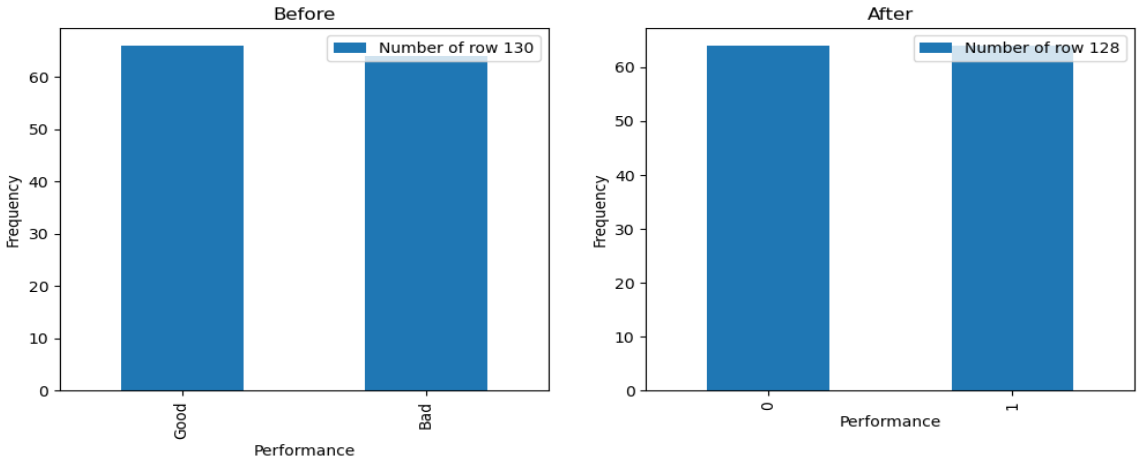
- “Yes” will be replaced by “1”
- “No” will be replaced by “0”




### 4.2.3 Performance

To use the performance data in the data field, which works in a way similar to the car park data field, our team turned the performance data into numbers. In addition, our team did some balancing to ensure that both "yes" and "no" had the same amount of data.

- Good will be replaced by “1”
- Bad will be replaced by “0”
- Rebalance the data



## 4.3 Data Normalization

Scaling is the method of altering the dataset's values so that they range between 0 and 1. The decision to scale or normalize the data depends on the user's machine-learning technique and the types of data. Even though there are many scaling methods, our team chose the Min-Max Scaler to normalize the data, and we explain why we didn't choose one of the other scaling methods.

- Robust Scaler: We did not select the robust scaler because it may not be suitable for data that is not approximately symmetric or has a skewed distribution, as it can lead to large differences between the maximum and minimum values of the scaled data.
- Standard Scaler (Z-Score): After running the T-test on all of the datasets, we found that most of them didn't follow the normal distribution, which is the assumption behind the z-score.
- Max-Abs Scaler: The maximum absolute scaler performs better when the data contains outliers or extreme values that can alter the minimum and maximum values.
- Normalize: The assumption underlying the use of normalize technique is that the data follows the Gaussian distribution (normal distribution). But our tests show that most of the data didn’t fit the normal distribution.
- Log Scaler: The log scaler assumes that the data follows a log-normal distribution. The majority of our data, however, did not fit the log-normal distribution.

### 4.3.1 Min-Max Scaler

Min-Max Scaler is a technique of scaling features in which the values of a given feature are changed to fall in a specific range, often [0, 1]. It is also known as "normalization" or "normalization to a range."

#### 4.3.1.1 Advantages and Disadvantages of Min-Max Scaler

There are several benefits to using the Min-Max scaler. First, it rescales the range of variables to a certain range, which may increase the accuracy and performance of machine learning algorithms like k-nearest neighbors and neural networks (Singh, 2020). Second, it prevents variables with huge magnitudes from dominating the model, which might result in the subpar performance of certain algorithms. In addition, the Min-Max scaler is an easy-to-implement technique that doesn’t require any assumptions about the data distribution. Finally, the approach is resistant to data outliers since extreme values are normalized to the same range as the other data points. In conclusion, the Min-Max scaler is a useful technique for preprocessing data for machine learning, especially when the data has a skewed distribution or when the range of feature values varies substantially (Kang, 2018).

Min-Max scaler may be sensitive to outliers since a single outlier can significantly impact the lowest and highest values of the feature, leading to unexpected scaling outcomes. This approach is unsuitable for normally distributed data because it might falsely accentuate the significance of outliers. It is also not good for data with very different ranges, because it may not be able to normalize the data well in those cases.

#### 4.3.1.2 Equation for Min-Max Scaler

The Min-Max scaler is obtained by subtracting the minimum value of the variable from each value and then dividing by the difference between the variable's maximum and minimum values (Gökhan, 2019). The following equation explains the Min-Max scaler's operation:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Where:

- X is the original variable value
- X' is the scaled variable value
- $X_{min}$ is the minimum value of the variable
- $X_{max}$ is the maximum value of the variable

## 4.4 Data Reduction

This step involved reducing the number of variables to only those that were pertinent to the modeling process. This reduced the amount of overfitting that occurred and enhanced the overall performance of the model. When it comes to simplifying the model, we will use three distinct strategies.

- OLS regression
- Outlier
- Correlation

### 4.4.1 OLS Regression

OLS (Ordinary Least Squares) regression is a common technique for analyzing the correlation between a dependent variable and one or more independent variables. The p-value is a statistical metric used in OLS (Ordinary Least Squares) regression to evaluate the significance of the connection between the dependent variable and each independent variable (Draper, 1998). It is computed on the assumption that there is no substantial correlation between the variables.

The p-value is used to figure out if an independent variable in the model is statistically important or not.

- The correlation between the dependent and independent variables is statistically significant if the p-value is less than the commonly used significance threshold of 0.05 (Draper, 1998).
- When the p-value is higher than the significance threshold (0.05), the correlation is not statistically significant, and the independent variable can be taken out of the model.

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Performance | R-squared (uncentered): | 0.763 |
| Model: | OLS | Adj. R-squared (uncentered): | 0.734 |
| Method: | Least Squares | F-statistic: | 26.26 |
| Date: | Tue, 14 Feb 2023 | Prob (F-statistic): | 2.94e-29 |
| Time: | 20:11:26 | Log-Likelihood: | -45.031 |
| No. Observations: | 128 | AIC: | 118.1 |
| Df Residuals: | 114 | BIC: | 158.0 |
| Df Model: | 14 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| Staff | 0.1540 | 0.023 | 6.561 | 0.000 | 0.107 | 0.200 |
| Floor Space | 0.0001 | 2.78e-05 | 3.781 | 0.000 | 5e-05 | 0.000 |
| Window | -0.0254 | 0.008 | -3.247 | 0.002 | -0.041 | -0.010 |
| Car park | 0.1606 | 0.076 | 2.115 | 0.037 | 0.010 | 0.311 |
| Demographic score | -0.0100 | 0.011 | -0.874 | 0.384 | -0.033 | 0.013 |
| Location | -0.1803 | 0.040 | -4.456 | 0.000 | -0.261 | -0.100 |
| 40min population | -3.118e-07 | 1.51e-07 | -2.061 | 0.042 | -6.12e-07 | -1.21e-08 |
| 30 min population | -1.257e-07 | 2.7e-07 | -0.466 | 0.642 | -6.6e-07 | 4.09e-07 |
| 20 min population | 5.952e-07 | 4.36e-07 | 1.365 | 0.175 | -2.69e-07 | 1.46e-06 |
| 10 min population | -2.331e-07 | 6.4e-07 | -0.364 | 0.716 | -1.5e-06 | 1.03e-06 |
| Store age | -0.0071 | 0.012 | -0.575 | 0.566 | -0.031 | 0.017 |
| Clearance space | 9.796e-05 | 0.001 | 0.117 | 0.907 | -0.002 | 0.002 |
| Competition number | 0.0304 | 0.011 | 2.667 | 0.009 | 0.008 | 0.053 |
| Competition score | 0.0584 | 0.011 | 5.154 | 0.000 | 0.036 | 0.081 |

| | | | |
|---|---|---|---|
| Omnibus: | 1.953 | Durbin-Watson: | 1.237 |
| Prob(Omnibus): | 0.377 | Jarque-Bera (JB): | 1.541 |
| Skew: | 0.084 | Prob(JB): | 0.463 |
| Kurtosis: | 2.489 | Cond. No. | 5.91e+06 |

Notes:
[1] $R^2$ is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 5.91e+06. This might indicate that there are strong multicollinearity or other numerical problems.

The results of the OLS regression indicated that the variables "Demographic score," "30 min population," "20 min population," "10 min population," “store age” and "clearing space" all had p-values greater than 0.05, indicating that these variables are not statistically significant and may be of less importance to the model.

### 4.4.2 Outlier

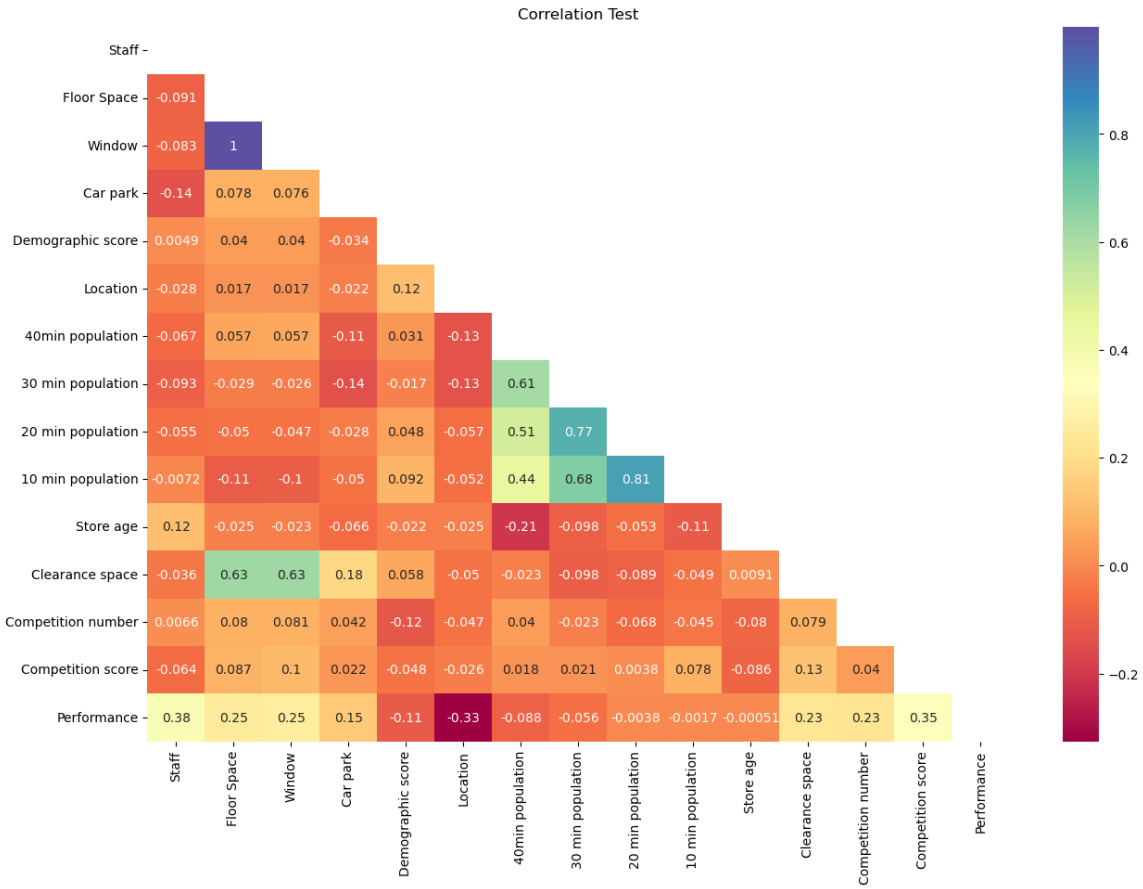
Our team did a thorough analysis and found that some of the variables had one or more outliers. The outliers can have a substantial effect on the outcomes of our study and modeling. Outliers can distort the data and produce erroneous patterns or relationships. Prior to continuing with our investigation, it is necessary to identify and address these outliers. We also included an algorithm to find the outliers and a table indicating the number of outliers for each variable.

- Determine the first and third quartiles of the variable.
- Using the formula: IQR = Q3 - Q1, compute the interquartile range (IQR) (Devore, 2008).
- The lower and upper bounds can be calculated using the following formulas: lower bound = Q1 - 1.5 IQR, and upper bound = Q3 + 1.5 IQR (Devore, 2008).
- Any values that fall outside of the lower and upper bounds would be classified as outliers.

| | 30 min population | 20 min population | 10 min population | Clearance Space |
|---|---|---|---|---|
| Number of Outliers | 1 | 2 | 10 | 1 |

### 4.4.3 Correlation

In addition, the correlation test was run to establish the relationship between the variables in our dataset. Through the use of a heatmap, the correlation test gives a visual depiction of the correlation between each pair of variables. The heatmap is generated by visualizing the correlation coefficients between each pair of variables in the form of a matrix, where the color of each cell denotes the degree of correlation between two variables. Negative correlations are represented by hues of red, whereas positive correlations are represented by shades of blue. The heatmap made it easy to find things that were strongly connected, which gave us important data-driven insights.



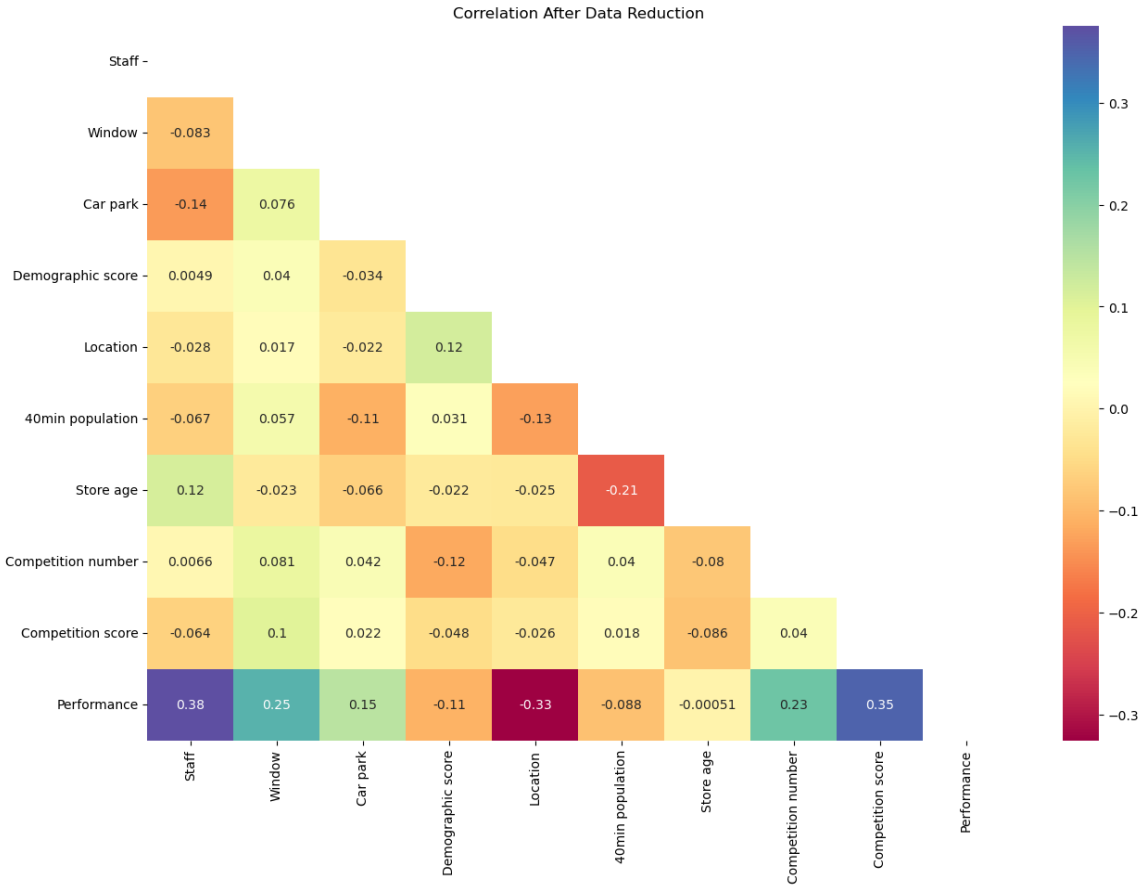

After doing a thorough correlation analysis, it was found that some of the variables in the dataset were strongly correlated to other variables.

- The correlation between floor space and windows is perfectly correlated, with a coefficient of 1.
- There is a strong positive correlation of 0.77 between the population within a 20-minute radius and the population within a 30-minute radius.
- There is a strong positive correlation of 0.81 between the population within a 20-minute radius and the population within a 10-minute radius.
- There is a moderate positive correlation of 0.61 between the population within a 30-minute radius and the population within a 40-minute radius.
- This is a moderately positive correlation of 0.63 between the clearance space and windows.
- This is a moderately positive correlation of 0.63 between the clearance space and floor space.

### 4.4.4 Analysis of the Result

Based on the information provided, our team has determined to exclude the following variables from our model selection: "30 minute population," "20 minute population," "10 minute population," and "clearance space." These variables displayed a moderate-to-

perfect correlation with other variables and contained multiple outliers. However, despite having high p-values, variables such as "demographic score" and "store age" may still be useful in the model, and we have decided to keep them because removing them would oversimplify the model.




## 4.5 Data Splitting

To evaluate the performance of the models, training, validation, and test datasets were created. The training dataset was utilized to construct the models, the validation dataset was utilized to evaluate the performance of a machine learning model during the training process, and the test dataset was utilized to evaluate the performance of the models after they had been trained.

In our model, we chose the 50:25:25 split method to split the data, i.e., 50% of the data will be utilized for training, 25% of the data will be utilized for validation, and 25% of the data will be utilized for testing. In addition, we applied the leave-one-out cross-validation technique for hyperparameter tuning.

# 5.0 Pre-Modeling

Before processing into modeling, our team will discuss some of the steps we will follow during the modeling process.

## 5.1 Modeling Step

In this project, we used the scikit-learn Python library to process the three different machine-learning models. For each of the models, our team will follow the following steps:

1. The training procedure will begin with the initialization of the model, which will have no predefined parameters.

2. Carry out the cross-validation method and check to determine how well the model will perform with the given training data.

3. After fitting the model to the training data and making predictions using the validation dataset, we will make performance measures like the accuracy score, AUC score, and confusion matrix to determine how well the model works.

4. To improve the performance of the model, we will first make a list of hyperparameters, then use the leave-one-out split method for grid search to explore the range of hyperparameters, and then choose the best parameter based on the grid search results.

5. Repeat steps one through three, but this time we will use the hyperparameter that was found in step 4 instead of the model, which has no predefined parameters.

6. Compare and contrast the two models that were produced by steps 1 and 5. Perform an analysis of the model.

7. We choose the best model out of three models that have had their hyperparameters adjusted, then carry out the final test with the test dataset and analyses the results.

## 5.2 Grid Search vs. Random Search

Grid search and random search are two common approaches for hyperparameter tuning, the process of determining the optimal set of hyperparameters for a machine learning model. Grid search is a deterministic technique in which the algorithm tries every conceivable combination of the hyperparameters supplied in a preset grid. In other words, grid search thoroughly explores all conceivable hyperparameter combinations. Random search, on the other hand, is a probabilistic strategy in which the algorithm randomly picks hyperparameters from a given distribution (Zabinsky, 2009). Grid search is exhaustive and deterministic, while random search is less exhaustive and more probabilistic. The decision between grid search and random search is determined by the size of the hyperparameter space and the available processing resources. If the hyperparameter search space is small and the model is computationally inexpensive, grid

search may be a viable option. If the hyperparameter space is huge and the model is computationally costly, a random search may be a more effective method for finding the best hyperparameters.

## 5.3 Leave One Out Cross-validation

Leave-One-Out Cross-validation (LOOCV) is a type of cross-validation process used in machine learning to evaluate a model's performance. LOOCV separates its data into a training set and a test set. The training set is used to fit the model, while the test set is used to evaluate the model's performance (Beyeler, 2017). In LOOCV, the size of the test set is equivalent to a single sample, but in other cross-validation techniques, the test set is often significantly bigger (Wong, 2015). This implies that, in LOOCV, the model is fitted and evaluated n times, where n is the number of data samples, and each time a new sample serves as the test set. LOOCV offers a comprehensive evaluation of the model since each data sample is utilized precisely once as the test set. The model must be fitted and assessed n times, which is computationally costly.

# 6.0 Model - Logistic Regression

Logistic regression is a statistical and supervised machine learning approach for assessing a dataset whose outcome is dependent on one or more independent variables. It is used to simulate the binary outcome, where the result may be "yes" or "no" (expressed by 1 or 0), or, in specific circumstances, the probability between 0 and 1 (Nick, 2007). The purpose of logistic regression is to determine the best relationship between independent variables and the dependent variable (Draper, 1998).

**Keyword:**

Logistic Regression, Binary Classification, Statistical Modeling, Predictive Modeling, Machine Learning, Data Analytics, Supervised Learning, Logit Model, Maximum Likelihood Estimation

## 6.1 Features

Logistic regression is the statistical method for assessing a dataset whose outcome is dependent on one or more independent variables. The most important features of logistic regression include:

- Logistic regression is a statistical model that is used to figure out how likely it is that something will happen or not (Nick, 2007).
- Logistic regression presupposes a linear relationship between independent variables and the dependent binary result.
- Regularization of logistic regression helps prevent overfitting and improve model stability (Komarek, 2004).
- Logistic regression can handle multicollinearity, which happens when independent variables are highly correlated.
- Logistic regression offers coefficients that may be used to demonstrate the impact of each independent variable on the result.

## 6.2 Equation

Logical regression can be shown with an equation, which can be written as follows (Hosmer, 2013):

$$P(y = 1|x) = \frac{1}{\left(1 + e^{-(B_0 + B_1x_1 + B_2x_2 + .... + B_nx_n)}\right)}$$

Where:

- p(y=1|x) is the probability of y being 1 given the independent variables x

- $B_0$ is the intercept
- $B_1 \ldots B_n$ are the coefficients for the independent variables $x_1$ to $x_n$
- e is the natural logarithmic base

Using the above equation, a probability score between 0 and 1 may be calculated. This score is then compared to a threshold number in order to generate binary predictions.

## 6.3 Advantages and Disadvantages

In the field of statistical modeling, logistic regression is especially useful for situations requiring binary classification. It is simple to understand and interpret since it implies a linear relationship between the independent variables and the result. Also, logistic regression models the likelihood that an event will happen to give a clearer picture of the expected result (Peng, 2002).

Regularization may be used in logistic regression to reduce overfitting and enhance model stability, making it a reliable predictive tool. It can also deal with multicollinearity, which is when different variables are strongly correlated to each other. This makes it a flexible tool for analyzing large datasets.

Furthermore, logistic regression produces simple-to-interpret coefficients that reveal the relative importance of each independent variable to the outcome. This simplifies the comprehension and dissemination of the model's predictions.(Hosmer, 2013).

Logistic regression is a useful tool for solving classification problems, but before you use it, you should think about a number of problems and important issues. One of the biggest problems is that it depends on a number of assumptions, such as linearity, independence of errors, homoscedasticity, and the lack of multicollinearity (Komarek, 2004). Second, logistic regression is also affected by outliers and extreme results, so it is very important to find and get rid of outliers before making the model. (Andersen, 2010).

Not only that, Logistic regression assumes a linear relationship between the independent variables and the outcome (Hosmer, 2013); if this relationship is not linear, the model may not reliably predict the outcome. Also, logistic regression can't fully explain complex relationships between variables, so it's important to look closely at the independent variables in the model. Finally, logistic regression is sensitive to overfitting, which happens when the model fits the training data too precisely, resulting in poor performance on new data. Regularization might help stop overfitting, but this problem still needs to be kept in mind when building the logistic regression model.

# 6.4 Input and Output

There are two primary inputs in logistic regression: independent variables and binary dependent variables. They are used to forecast the result of the binary dependent variable, also known as the response variable.

The logistic regression model predicts an outcome based on the likelihood. The outputs of logistic regression include regression coefficients representing the expected effect of each independent variable on the outcome. The model intercept and the event occurrence probability are also outputs for a given set of independent variable values. In addition, model statistics such as R-squared, AIC, and BIC are supplied in the output to provide information on the model's fit and performance.

## 6.5 Hyperparameters

Among the hyperparameters that must be adjusted in logistic regression are:

- Regularization strength
- Solver
- Penalty
- C (inverse of regularization strength)
- Tolerance
- The maximum number of iterations

Regularization strength controls the amount of regularization done to the model and helps avoid overfitting. Newton-Raphson, BFGS, and L-BFGS are common solvers used in logistic regression. The selection of the penalty may affect the model's performance and interpretability. The parameter C reflects the inverse of the regularization strength, with lower values producing stronger regularization and a simpler model and bigger values producing weaker regularization and a more complicated model. The tolerance establishes the stopping criteria for the solver and the required degree of precision in the model's coefficients. The maximum number of iterations specifies the maximum number of iterations permitted for the solver to converge to a solution, with larger values perhaps yielding more accurate models but increasing computing time. It is essential to remember that the particular hyperparameters and their ideal values will vary based on the situation and data at hand.

## 6.6 Process of the Model

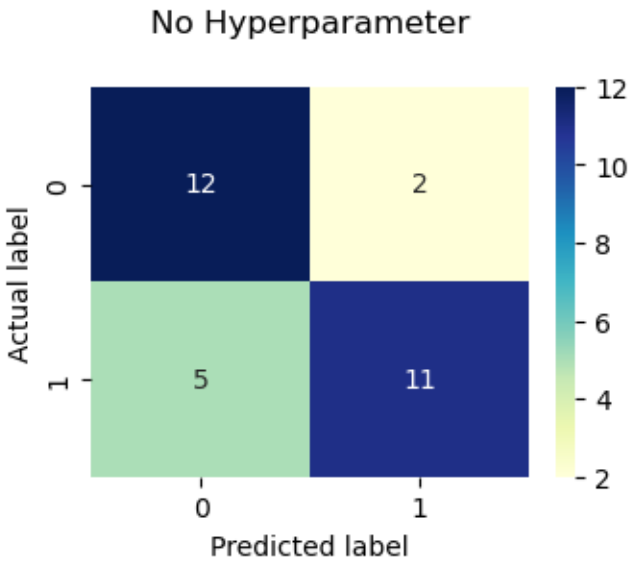
To begin with, our team developed the logistic regression model using its default parameters. We then sought to evaluate the model's performance by implementing a cross-validation technique using the "cross_val_score" function available in the Scikit-Learn library. As a result of this process, we obtained a list of five data points, which provide insights into the accuracy of the model across different training and test sets. To acquire a better understanding of the model's overall accuracy, we took the mean and standard deviation of the list using the NumPy library. This allowed us to estimate the model's expected accuracy and assess its reliability. By examining these statistics, we can get some idea of how the training data will fit the model.

```
Cross Validation accuracy scores: [0.71428571 0.92857143 1.
0.61538462 0.84615385]
```

```
Cross Validation accuracy: 0.821 +/- 0.140
```

After cross-validation, we fitted a model to the data. This consists of finding a mathematical function that properly represents the connection between the variables in the data using statistical techniques. Once the model has been fitted, it must be validated using a distinct collection of data known as the validation data. This enables us to see whether the model can properly anticipate the result of new data. To evaluate the performance of the fitted model, we produced a series of predictions from the validation data using the predict function. The predicted values are then compared to the actual values in the dependent (y-validation) dataset. Various evaluation metrics supplied by the scikit-learn package were used to determine the accuracy of the model. The accuracy score represents the percentage of accurate predictions, the ROC-AUC score evaluates the quality of the model's binary classification predictions, and the confusion matrix offers a breakdown of the model's predictions by category.

```
Validation Accuracy: 0.767
Validation AUC: 0.772
```




In addition to fitting the model to the training data and evaluating its performance using various metrics, we extract essential information about the model using methods supplied by the scikit-learn package. One such function is "coef_", which enables us to determine the logistic regression model's coefficients. These coefficients describe the relative significance of each model component and may be used to comprehend the connections between independent and dependent variables. The "intercept_" is an essential function that supplies the intercept of the logistic regression model. This intercept value is a constant term added to the product of the feature values and their respective coefficients. It may be used to calculate the likelihood of the dependent variable in the absence of any characteristics.

By applying these functions to extract crucial model information, we may acquire a better understanding of the model's underlying structure and the variables influencing its predictions. This information may assist us in refining and enhancing the model, as well as in making better-educated judgments based on its output.

```
Coef:
array([[ 1.61027882,  1.26200736,  0.68673313, -0.57495418, -1.65466
975,-0.35608953,  0.13328038,  0.27889744,  1.85094309]])

Intercept:
array([-1.84501958])

Feature name:
array(['Staff', 'Window', 'Car park', 'Demographic score', 'Location
','40min population', 'Store age', 'Competition number','Competition
 score'], dtype=object)


logistic regression equation:
Performance = 1.61027882 * Staff + 1.26200736 * Window + 0.68673313
* 'Car Park' - 0.57495418 * 'Demographic score' - 1.65466975 * Locat
ion - 0.35608953 * '40min population' + 0.13328038 * 'Store age' +
0.27889744 * 'competition number' + 1.85094309 * 'competition score'
 - 1.84501958
```

Following the original logistic regression model, we will re-initialize a new model to enhance its performance further. So, we set up a set of parameters that we think will help make the model work best.

To identify the ideal set of parameters, we used the grid search algorithm. This entails searching extensively through a specific range of parameter values to determine the combination that produces the greatest performance score. By methodically altering the parameters and analyzing the resultant model performance, we may successfully fine-tune and increase the model's accuracy.

As part of the grid search procedure, we must choose an appropriate cross-validation splitting technique for our training dataset. Due to the relatively modest size of the training dataset, we used the LeaveOneOut method in this instance. This entails training using all but one data point and validating with the remaining data point. We can obtain a robust assessment of the model's performance and prevent overfitting to the training data by repeating this method over all the data points.

```
parameter = [
    {'penalty' : ['l1', 'l2', 'elasticnet'],
    'C' : np.logspace(-4, 4, 20),
    'solver' : ['lbfgs','newton-cg','liblinear','sag','saga'],
     'multi_class': ['auto', 'ovr', 'multinomial'],
    'max_iter' : [100, 1000,2500, 5000]
```

```
    }
]
```

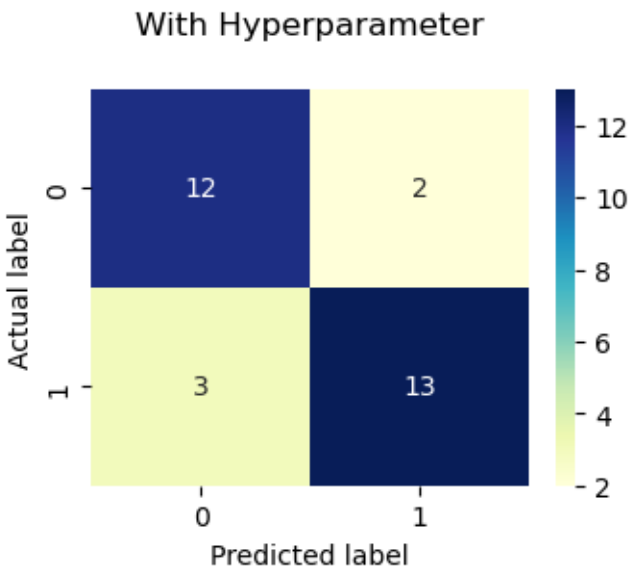
Upon completion of the "grid searches" function, we may extract the optimal feature model parameters. This was a crucial stage in the process of enhancing the performance of the model, as it helped us find the particular combination of parameters that produced the best accuracy score.  Once the optimal parameter values have been determined, we may utilize them to modify and enhance the model, thereby boosting its predictive accuracy.

```
Best Hyperparameters found: {'C': 1.623776739188721, 'max_iter': 10
0, 'multi_class': 'multinomial', 'penalty': 'l2', 'solver': 'lbfgs'}
```

Once the parameters have been adjusted, the model's performance may be further refined by repeating the cross-validation and data-validation steps using the optimized parameters. This enables us to collect numerous performance indicators, including cross-validation scores, validation accuracy scores, confusion matrices, classification reports, and an AUC curve.

```
Cross Validation accuracy scores: [0.71428571 0.78571429 1.
0.61538462 0.84615385]
Cross Validation accuracy: 0.792 +/- 0.129

Validation Accuracy: 0.833
Validation AUC: 0.835
```




```
Coef:
array([[ 1.36927977,  1.06348897,  0.48153722, -0.50816729, -1.20919
648,-0.32769053,  0.08488207,  0.20164999,  1.53138428]])

Intercept:
```

```
array([-1.48384138])

Feature name:
array(['Staff', 'Window', 'Car park', 'Demographic score', 'Location
','40min population', 'Store age', 'Competition number','Competition
 score'], dtype=object)

logistic regression equation:
Performance = 1.36927977 * Staff + 1.06348897 * Window + 0.48153722
* 'Car Park' - 0.50816729 * 'Demographic score' - 1.20919648 * Locat
ion - 0.32769053 * '40min population' + 0.08488207 * 'Store age' +
0.20164999 * 'competition number' + 1.53138428 * 'competition score'
 - 1.48384138
```

## 6.7 Analysis of the Result

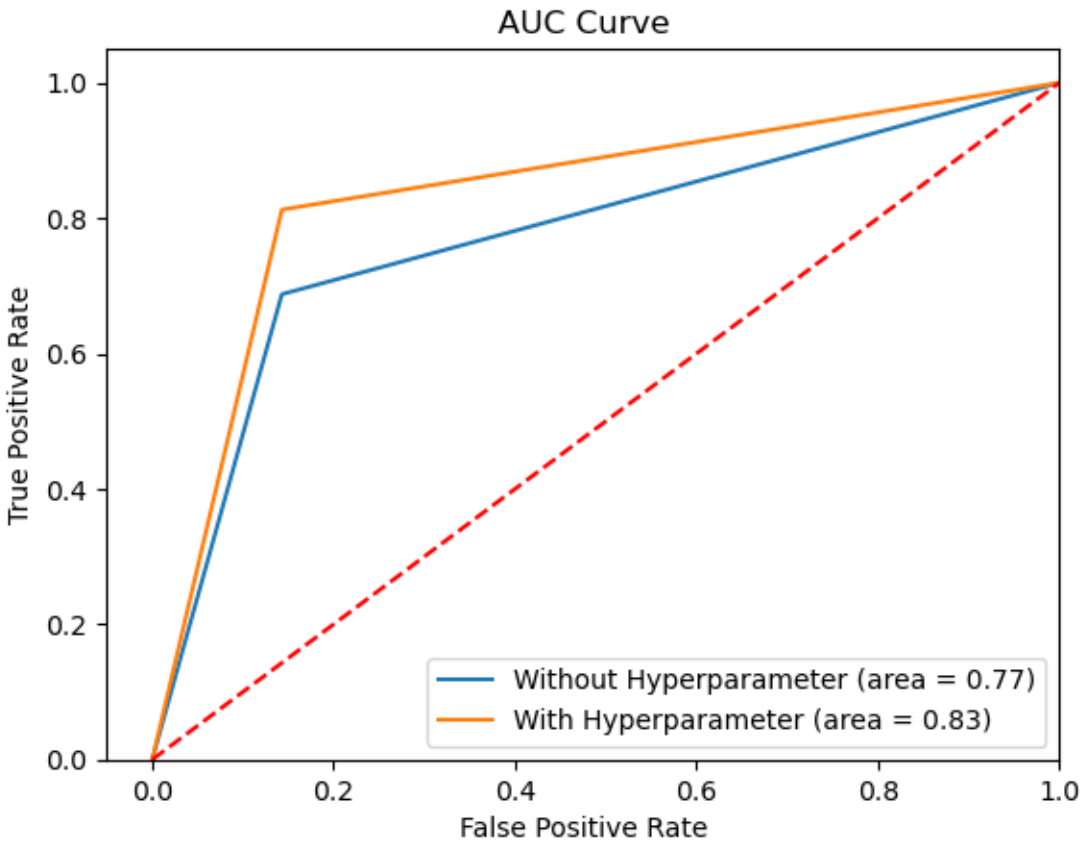
Based on the evaluation metrics and classification report, it looks like the validation dataset with adjusted hyperparameters does better than the one without. The model with optimized parameters exhibited a better accuracy score (0.83 vs. 0.77) and greater precision, recall, and F1 score values for both classes than the model without hyperparameter adjustment (0 and 1). The improved model has higher weighted average and macro average values for accuracy, recall, and F1-score. In addition, a comparison of the AUC curves reveals that those with hyperparameter adjustment cover a greater area than those without. Consequently, the validation dataset with hyperparameters adjusted would be a superior model.

Validatiaon dataset without hyperparameter tuning:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.71 | 0.86 | 0.77 | 14 |
| 1.0 | 0.85 | 0.69 | 0.76 | 16 |
| accuracy | | | 0.77 | 30 |
| macro avg | 0.78 | 0.77 | 0.77 | 30 |
| weighted avg | 0.78 | 0.77 | 0.77 | 30 |

Validatiaon dataset with hyperparameter tuning:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.80 | 0.86 | 0.83 | 14 |
| 1.0 | 0.87 | 0.81 | 0.84 | 16 |
| accuracy | | | 0.83 | 30 |
| macro avg | 0.83 | 0.83 | 0.83 | 30 |
| weighted avg | 0.84 | 0.83 | 0.83 | 30 |


AUC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Without Hyperparameter (area = 0.77)
With Hyperparameter (area = 0.83)

# 7.0 Model – Random Forest

The random forest classifier is an approach for machine learning used to categorize problems. It is a technique for ensemble learning that combines the predictions of numerous decision trees to provide a final prediction (Rokach, 2016).

The technique operates by generating numerous decision trees using various samples of the training data and averaging the outcomes of each tree. The final prediction of the Random Forest Classifier is decided by taking the average of all the forecasts from the trees or by picking the class with the highest number of votes. The method is renowned for its robustness, flexibility, and precision, as well as its ability to handle missing data, a high number of features, and outliers (Parmar, 2019).

**Keyword:**

Random Forest Classifier, Ensemble Learning, Decision Trees, Classification, Machine Learning, Data Mining, Prediction, Missing Values, Outlier Resistance, Binary Classification, Multi Class Classification

## 7.1 Feature

Tree-based models are a type of machine learning model that generates predictions using tree structures. These models are commonly used in both regression and classification problems, and their distinctive characteristics make them a popular option for many data science projects:

1. The Random Forest Classifier is an ensemble method that generates a final prediction by combining the results of multiple decision trees (Rokach, 2016).
2. The method is resistant to missing data and a high number of features, and it is resistant to outliers. The method is known for being accurate, especially when compared to single decision tree classifiers.
3. The Random Forest Classifier may be used to establish the significance of each feature in the prediction. And it applies to both binary and multiclass classification tasks.
4. The Random Forest Classifier can effectively deal with unbalanced data sets, making it ideal for several real-world applications. The method is capable of dealing with nonlinear relationships between features and the target variable (Zhang, 2012).

## 7.2 Equation

As opposed to other machine learning models, such as linear regression and logistic regression, the Random Forest approach lacks a single classification equation. Instead, it constructs a set of decision trees, each of which makes a prediction depending on the

input attributes. The final forecast is then derived by combining the predictions of all the trees. This can be done by using the majority vote or the arithmetic mean.

When applying random forests to classification data, it is essential to remember that the Gini index formula controls how nodes are organized in a decision tree branch. This formula estimates the Gini of each branch on a node based on the class and probability, which helps forecast which branch is most likely to occur (Breiman, 2001).

Gini Index equation:

$$Gini = 1 - \sum_{t=1}^{T} (p_i)^2$$

The following equation can be used to determine how nodes in a decision tree branch using entropy:

$$Entropy = \sum_{t=1}^{T} -p_i log(p_i)$$

Where:

- $p_i$ represents the class's relative frequency in the data set
- T is the number of classes

## 7.3 Advantages and Disadvantages

Due to its remarkable accuracy, robustness, and adaptability, the random forest classifier is widely acknowledged as a strong machine-learning technique. In terms of accuracy, this method generally outperforms single-decision tree classifiers by combining the predictions of multiple decision trees to produce a final prediction. It is also capable of handling missing data and is resistant to outliers. This makes it a good choice for a wide range of practical uses.

In addition, this method provides a method for estimating the significance of each variable in the prediction, which is useful for variable selection and dimensionality reduction. Lastly, the random forest classifier can deal with relationships between variables and the target variable that are not linear. And it is also easy to parallelize, which makes it perfect for large data sets and applications that need to work quickly.

Like the vast majority of machine learning algorithms, the random forest classifier has a number of challenges and limitations that must be carefully considered. For instance, overfitting might be problematic when there are too many trees in a forest. In addition, the computational cost of random forest classifiers can be high, particularly when dealing with huge datasets and a large number of trees. As highlighted by Hatwell, the algorithm

tends to favor categorical variables with high cardinality, meaning that it may assign more weight to variables having a large number of categories (Hatwell, 2020).

Furthermore, the algorithm may have a preference for balanced data, resulting in a subpar performance with imbalanced datasets. Lastly, the success of the random forest classifier depends a lot on how the hyperparameters are chosen, and it can be hard to find the right ones.

## 7.4 Input and Output

The Random Forest Classifier algorithm needs three inputs for proper operation. The first input is "Features", which is a collection of independent variables that characterize the features of each dataset sample. The second input is "Labels", which is a collection of variables that specify the target classes or labels for each sample in the dataset. The third input is "Hyperparameters", which is a collection of parameters that shape the behavior of the algorithm.

The Random Forest Classifier algorithm's outputs include a trained model, predictions, feature significance, confidence scores, and decision trees. The trained model is a representation of the learning from the training data, and it may be used to make predictions on new, unseen data. The algorithm has assigned class labels to a given collection of variables, which are the predictions. The relevance of a variable indicates the relative contribution of each variable to the prediction. The confidence ratings measure the amount of certainty in each prediction and are derived from the forecasts of each individual decision tree in the forest. The decision trees in the forest are also important because they help understand how the model works and how to understand the predictions.

## 7.5 Hyperparameters

To get the best performance using the random forest classifier, a number of hyperparameters must be fine-tuned. Tuning considerations include the number of trees in the forest, the maximum depth of the trees, the minimum number of samples required to split a node, the minimum number of samples required to be at a leaf node, the maximum number of features to consider when splitting a node, the criterion for splitting nodes, such as Gini impurity, information gain, and the bootstrapping method for building trees.

## 7.6 Process of the Model

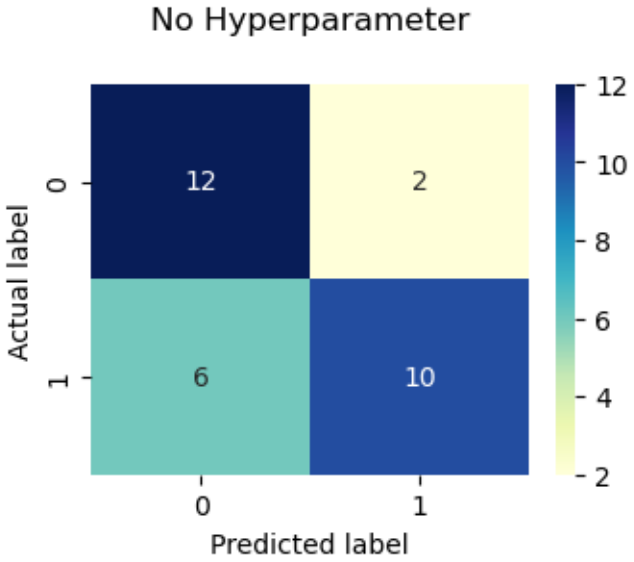
Similar to our method for logistic regression, we started by constructing a random forest decision tree model with no parameters specified. Next, we used the cross-validation method to estimate the model's accuracy. Our dataset was divided into many folds for cross-validation, each time using a new fold for testing and the remaining folds for training. This procedure was done fivefold, and a list of cross-validation scores was

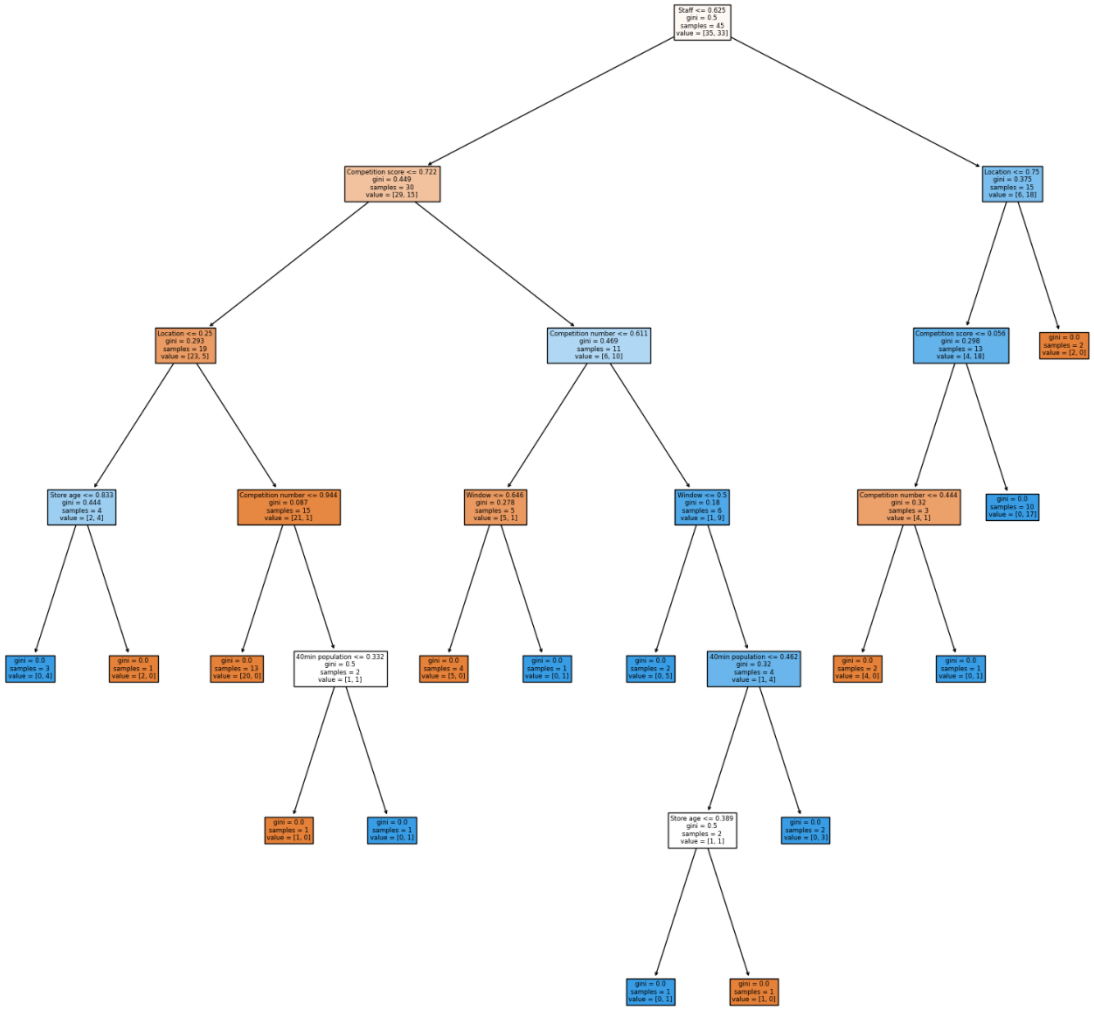
generated.  To summarize the cross-validation results, we estimated the list's mean and standard deviation. The mean value offered an estimate of the model's accuracy, but the standard deviation supplied a measure of the variation in those estimations. By evaluating both of these numbers, we may have a deeper understanding of the model's overall performance.

```
Cross Validation accuracy scores: [0.78571429 0.57142857 0.64285714
0.53846154 0.84615385]
Cross Validation accuracy: 0.821 +/- 0.120
```

The RandomForestClassifier model was fitted with no parameters after receiving the cross-validation scores. We next made predictions on the validation dataset using the fitted model and obtained the validation accuracy, the AUC score, and a confusion matrix. Since a random forest comprises several decision trees, we only construct one decision tree for illustrative reasons.

```
Validation Accuracy: 0.733
Validation AUC: 0.741
```





Even though the hyperparameters for the random forest model and those for the logistic regression model are different, the process of employing grid search to optimize those hyperparameters is the same in both models. Due to the limited size of the training dataset, we continue to use the "LeaveOneOut" technique when it comes to the cross-validation splitting process.

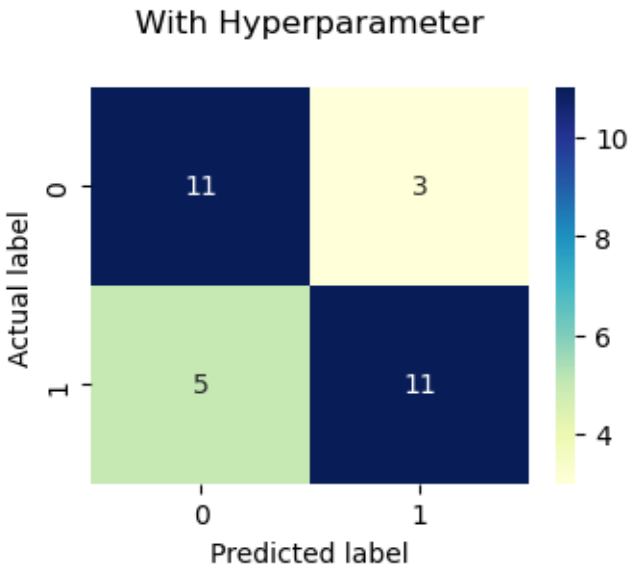
```
parameter = [{
      'criterion': ['gini', 'entropy', 'log_loss'],
      'max_features': ['sqrt', 'log2', None],
      'class_weight': ['balanced', 'balanced_subsample'],
            }
]
```

After completing the grid search to improve the random forest classifier's hyperparameters, we repeated the procedure of fitting the model with the optimal parameters and predicting the validation dataset. This enables us to generate a more precise estimate of how well the model will perform with new data. The validation accuracy was subsequently computed, revealing the percentage of cases in the validation dataset that were properly classified. Additionally, we computed the AUC score, which offers an overall evaluation of the model's ability to distinguish between positive and negative classifications. We also provided the confusion matrix, which offers a breakdown of the number of true positives, false positives, true negatives, and false negatives. Finally, we provided a classification report that details the accuracy, recall, and F1 score.

```
Cross Validation accuracy scores: [0.78571429 0.57142857 0.57142857
0.61538462 0.76923077]
Cross Validation accuracy: 0.663 +/- 0.095

Best Hyperparameters found: {'class_weight': 'balanced_subsample', '
criterion': 'gini', 'max_features': None}

Validation Accuracy: 0.733
Validation AUC: 0.737
```




## 7.7 Analysis of the Result

Based on our findings, the first model without a hyperparameter adjustment appears to be superior. It has greater accuracy for class 1 (0.83) and recall for class 0 (0.86), suggesting that it is more accurate at accurately detecting positive instances and avoiding false negatives. In addition, the first model has a greater cross-validation accuracy (0.821 +/- 0.120) than the second model (0.663 +/- 0.095).

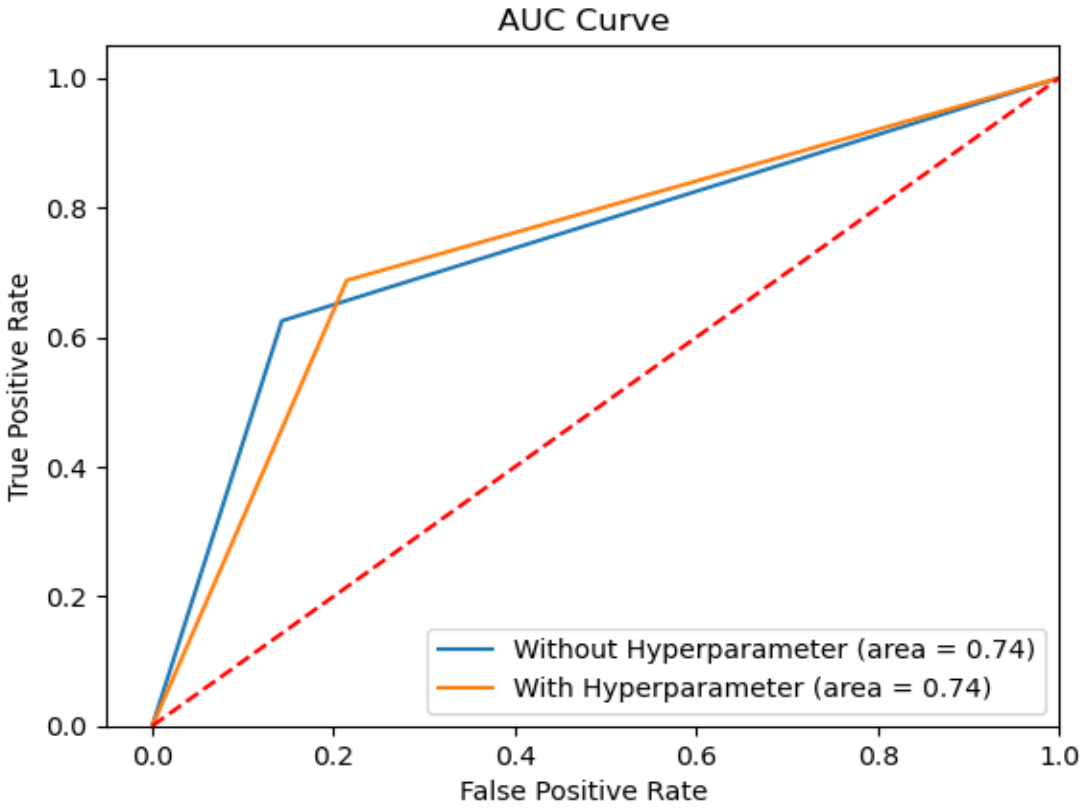
According to the results of the validation dataset, it seems that tuning the model's hyperparameters did not greatly improve its performance. The model with hyperparameter adjustment has a worse cross-validation accuracy than the model without adjustment. This indicates that the model's default parameters may be well-suited to the data and that efforts to improve the parameters through grid search may have resulted in overfitting or other problems. It may be worthwhile to examine additional techniques for enhancing the performance of the model, such as feature engineering or investigating other model topologies.

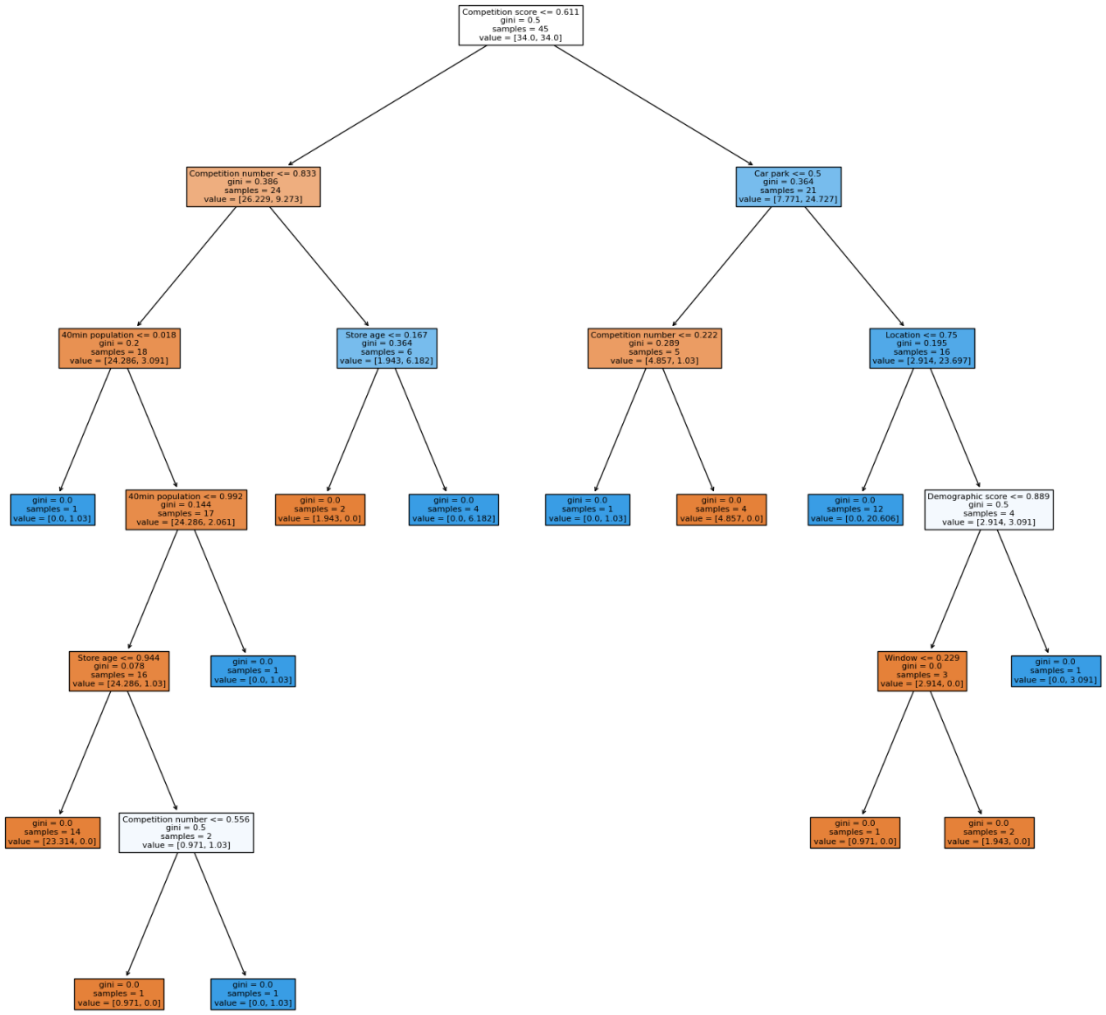
Validatiaon dataset without hyperparameter tuning:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.67 | 0.86 | 0.75 | 14 |
| 1.0 | 0.83 | 0.62 | 0.71 | 16 |
| accuracy | | | 0.73 | 30 |
| macro avg | 0.75 | 0.74 | 0.73 | 30 |
| weighted avg | 0.76 | 0.73 | 0.73 | 30 |

Validatiaon dataset with hyperparameter tuning:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.69 | 0.79 | 0.73 | 14 |
| 1.0 | 0.79 | 0.69 | 0.73 | 16 |
| accuracy | | | 0.73 | 30 |
| macro avg | 0.74 | 0.74 | 0.73 | 30 |
| weighted avg | 0.74 | 0.73 | 0.73 | 30 |




Competition score <= 0.611
gini = 0.5
samples = 45
value = [34.0, 34.0]
Competition number <= 0.833
gini = 0.386
samples = 24
value = [26.229, 9.273]
Car park <= 0.5
gini = 0.364
samples = 21
value = [7.771, 24.727]
40min population <= 0.018
gini = 0.2
samples = 18
value = [24.286, 3.091]
Store age <= 0.167
gini = 0.364
samples = 6
value = [1.943, 6.182]
Competition number <= 0.222
gini = 0.289
samples = 5
value = [4.857, 1.03]
Location <= 0.75
gini = 0.195
samples = 16
value = [2.914, 23.697]
gini = 0.0
samples = 1
value = [0.0, 1.03]
40min population <= 0.992
gini = 0.144
samples = 17
value = [24.286, 2.061]
gini = 0.0
samples = 2
value = [1.943, 0.0]
gini = 0.0
samples = 4
value = [0.0, 6.182]
gini = 0.0
samples = 1
value = [0.0, 1.03]
gini = 0.0
samples = 4
value = [4.857, 0.0]
gini = 0.0
samples = 12
value = [0.0, 20.606]
Demographic score <= 0.889
gini = 0.5
samples = 4
value = [2.914, 3.091]
Store age <= 0.944
gini = 0.078
samples = 16
value = [24.286, 1.03]
gini = 0.0
samples = 1
value = [0.0, 1.03]
Window <= 0.229
gini = 0.0
samples = 3
value = [2.914, 0.0]
gini = 0.0
samples = 1
value = [0.0, 3.091]
gini = 0.0
samples = 14
value = [23.314, 0.0]
Competition number <= 0.556
gini = 0.5
samples = 2
value = [0.971, 1.03]
gini = 0.0
samples = 1
value = [0.971, 0.0]
gini = 0.0
samples = 2
value = [1.943, 0.0]
gini = 0.0
samples = 1
value = [0.971, 0.0]
gini = 0.0
samples = 1
value = [0.0, 1.03]

# 8.0 Model - Multi-layer Perceptron Classifier

The multi-layer perceptron (MLP) classifier is an artificial neural network used for supervised learning, particularly for classification problems. The input layer receives the input data, the hidden layers do calculations to extract features, and the output layer generates the classification result (Shepherd, 2012). MLP classifiers can be used for both binary and multi-class classification problems. They are often used in a wide range of applications because they can be trained and tuned on large datasets to achieve high accuracy.

**Keyword:**

Artificial Neural Network, Supervised Learning, Classification, Multi-layer Perceptron, Neural Network, Machine Learning, Deep Learning, Pattern Recognition, Predictive Modeling, Binary Classification, Multi-class Classification

## 8.1 Feature

The Multi-layer Perceptron (MLP) classifier is held in high esteem because of its distinctive characteristics, which may be categorized as follows:

- MLP classifiers are made up of many layers of connected nodes, which lets them find and learn high-level characteristics from the data they receive.
- Each node in the MLP classifier uses a non-linear activation function to introduce non-linearity into the model, such as the sigmoid or rectified linear unit (ReLU) function (Sharma, 2017). This permits the MLP classifier to simulate complicated input-output interactions.
- The MLP classifier uses an optimization algorithm, like gradient descent, to change the weights of the connections between the nodes. This is done so that the loss function is minimized.
- MLP classifiers can be applied to binary classification problems, in which the output can only take on two values (such as positive or negative), and multi-class classification problems, in which the output can take on more than two values (e.g., red, green, or blue).
- MLP classifiers may be used for a wide range of problems and can be tailored by modifying the number of layers, the number of nodes in each layer, and the activation functions employed. MLP classifiers can be trained on huge datasets and are readily parallelizable, making them appropriate for use with big data.

## 8.2 Equation

The Multi-layer Perceptron (MLP) classifier may be summed up by the equation that is shown below, which reflects the computation that is carried out at a single node in the network:

$$y = f(Wx + b)$$

**where:**

- y represents the output of the node
- f is the activation function
- x represents the input to the node
- b represents the bias term for the node
- W represents the weight matrix for the connections between the input and node.

## 8.3 Advantages and Disadvantages

Due to its numerous advantages, the multi-layer perceptron (MLP) classifier is a widely preferred machine-learning technique for classification problems. Using non-linear activation functions, the MLP classifier is able to model complex non-linear interactions between inputs and outputs, achieving amazing accuracy on big datasets. In addition, the classifier is scalable, flexible, and capable of generalizing to unknown data, making it a significant tool in a variety of real-world situations.

Importantly, MLP classifiers can perform both binary and multi-class classification tasks and give insight into the relationships between inputs and outputs. This feature facilitates a greater comprehension of the logic underlying a given categorization. All of these benefits render the MLP classifier a versatile and extensively utilized machine learning approach for a wide variety of classification applications.

The multi-layer perceptron (MLP) classifier is a popular and potent machine learning approach; however, it is not devoid of recognized problems and limitations. These problems include the possibility of overfitting, which can lead to poor performance on unknown data, and a substantial computational cost, which can be both time and resource intensive. In addition, non-convex optimization might result in suboptimal solutions due to the difficulty of locating the global minimum. To get the best results when employing the MLP classifier in real-world applications, it is essential to be aware of these possible hurdles and devise techniques for overcoming them.

## 8.4 Input and Output

The inputs of a multi-layer perceptron (MLP) classifiers are the features or variables that characterize each instance in the training dataset. They could be continuous, categorical, or binary. The output of the MLP classifier is a prediction for the class or label associated with each occurrence in the dataset.

The outcome of a binary classification is either positive or negative. Multiclass classification produces one of several class labels. Based on the patterns contained in the training dataset, the MLP classifier is trained to map inputs to the appropriate outputs.

## 8.5 Hyperparameters

In order to get optimum performance using a Multi-layer Perceptron (MLP) classifier, it is necessary to tweak a number of hyperparameters. These hyperparameters include:

- Number of hidden layers
- Number of nodes in each hidden layer
- The activation function used in each node
- Learning rate
- Momentum
- Regularization intensity
- Batch size

The number of hidden layers influences the model's complexity and, if not properly regulated, may lead to overfitting. The number of nodes in each hidden layer may have an effect on the model's performance, and the activation function employed in each node defines the model's non-linearity. The learning rate defines the weight update step size during training, while the momentum hyperparameter determines the speed and direction of weight updates. The regularization strength is a way to stop overfitting, and the batch size is the number of cases that are processed before the MLP classifier's weights are updated.

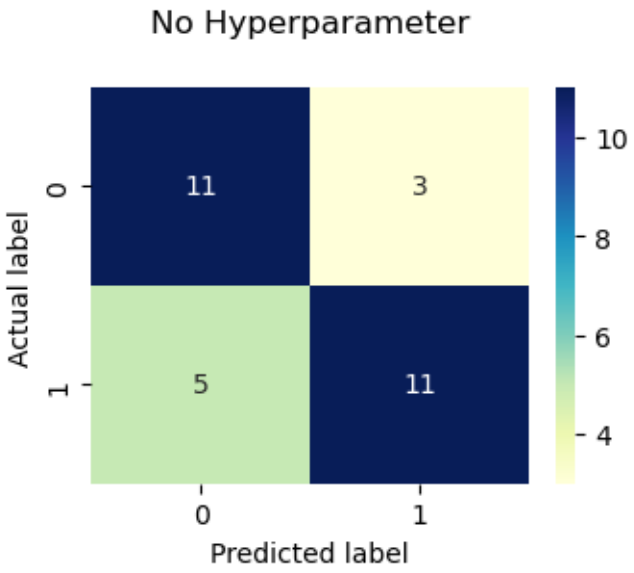
## 8.6 Process of the Model

We followed a similar procedure to train the multi-layer perceptron (MLP) classifier as we did with the previous models. First, we started the MLP model with no parameters specified. Then, we performed a cross-validation procedure in order to produce a list of cross-validation scores. Next, we determined the average cross-validation score by calculating the mean and standard deviation of the list of cross-validation scores.

```
Cross Validation accuracy scores: [0.71428571 0.71428571 1.
0.53846154 0.84615385]
Cross Validation accuracy: 0.763 +/- 0.154
```

After generating the average cross-validation score for the Multi-layer Perceptron classifier, we proceed to fit the model without specifying any parameters. The fitted model is then used to predict the validation dataset, and its performance may be evaluated by calculating validation accuracy, the AUC score, and the confusion matrix.

```
Validation Accuracy: 0.733
Validation AUC: 0.737
```




After figuring out which hyperparameters needed to be adjusted, we created a list of all the possible parameter values and then did a grid search to see which combination of parameters would give the best results.

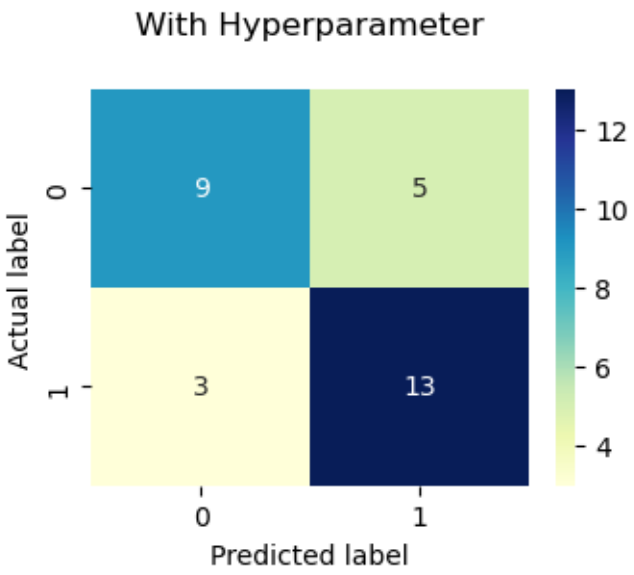
```
parameters = {
    "hidden_layer_sizes": [(50,50,50), (50,100,50), (100,)],
    "activation": ['identity', 'logistic', 'tanh', 'relu'],
    "solver": ['lbfgs', 'sgd', 'adam'],
    'learning_rate': ['constant', 'invscaling', 'adaptive'],
    "alpha": [0.0001, 0.05]
}
```

```
Best Hyperparameters found: {'activation': 'tanh', 'alpha': 0.05, 'h
idden_layer_sizes': (50, 50, 50), 'learning_rate': 'constant', 'solv
er': 'lbfgs'}
```

After receiving the optimal hyperparameters using the grid search, the training and validation procedures were repeated with the new parameters. We used the model fitted with the optimal hyperparameters to forecast the validation dataset. To assess the performance of the model with the new hyperparameters, we calculated the validation accuracy, AUC score, and confusion matrix.

```
Cross Validation accuracy scores: [0.71428571 0.71428571 1.
0.92307692 0.92307692]
Cross Validation accuracy: 0.855 +/- 0.118

Validation Accuracy: 0.733
Validation AUC: 0.728
```




## 8.7 Analysis of the Result

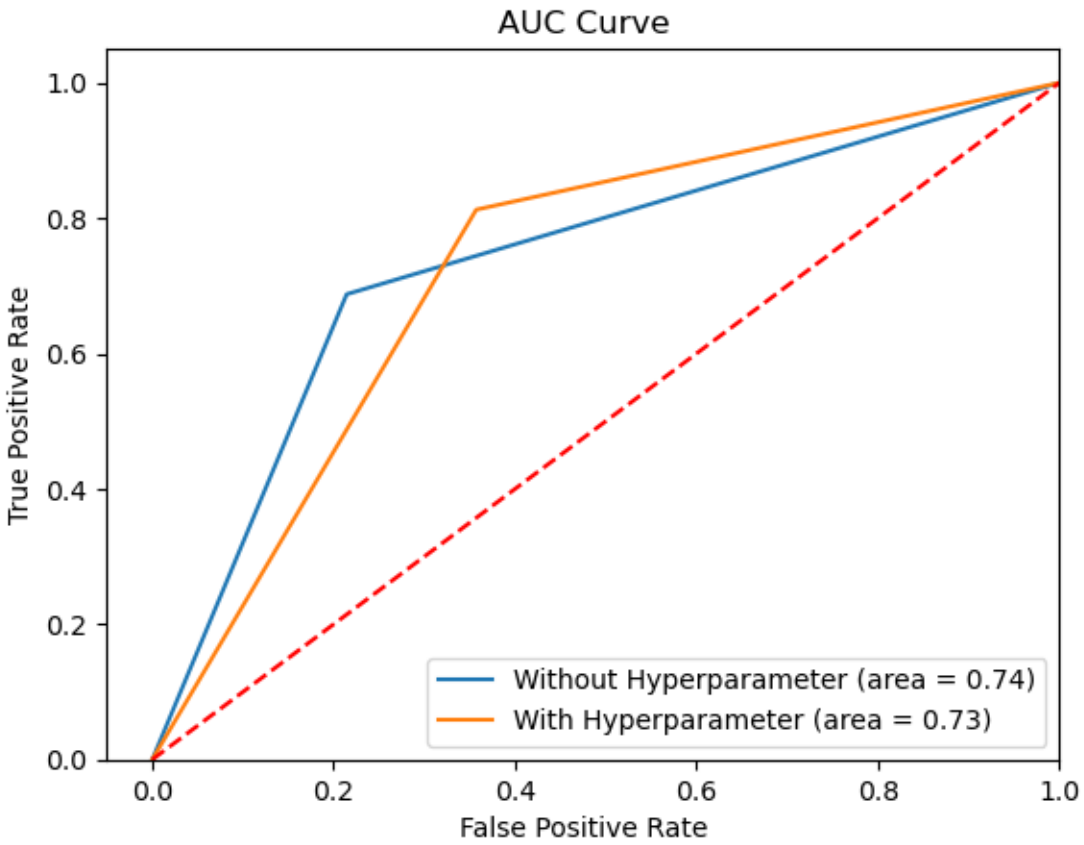
Compared to the one with hyperparameter adjustment, the model without hyperparameter adjustment has a poorer cross-validation accuracy (0.763) and worse precision, recall, and F1-score for both classes. The model with hyperparameter adjustment shows better cross-validation accuracy (0.855) as well as greater precision, recall, and F1 score for both classes. This indicates that the hyperparameter adjustment has helped increase the model's performance.

Validatiaon dataset without hyperparameter tuning:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.69 | 0.79 | 0.73 | 14 |
| 1.0 | 0.79 | 0.69 | 0.73 | 16 |
| accuracy | | | 0.73 | 30 |
| macro avg | 0.74 | 0.74 | 0.73 | 30 |
| weighted avg | 0.74 | 0.73 | 0.73 | 30 |

Validatiaon dataset with hyperparameter tuning:

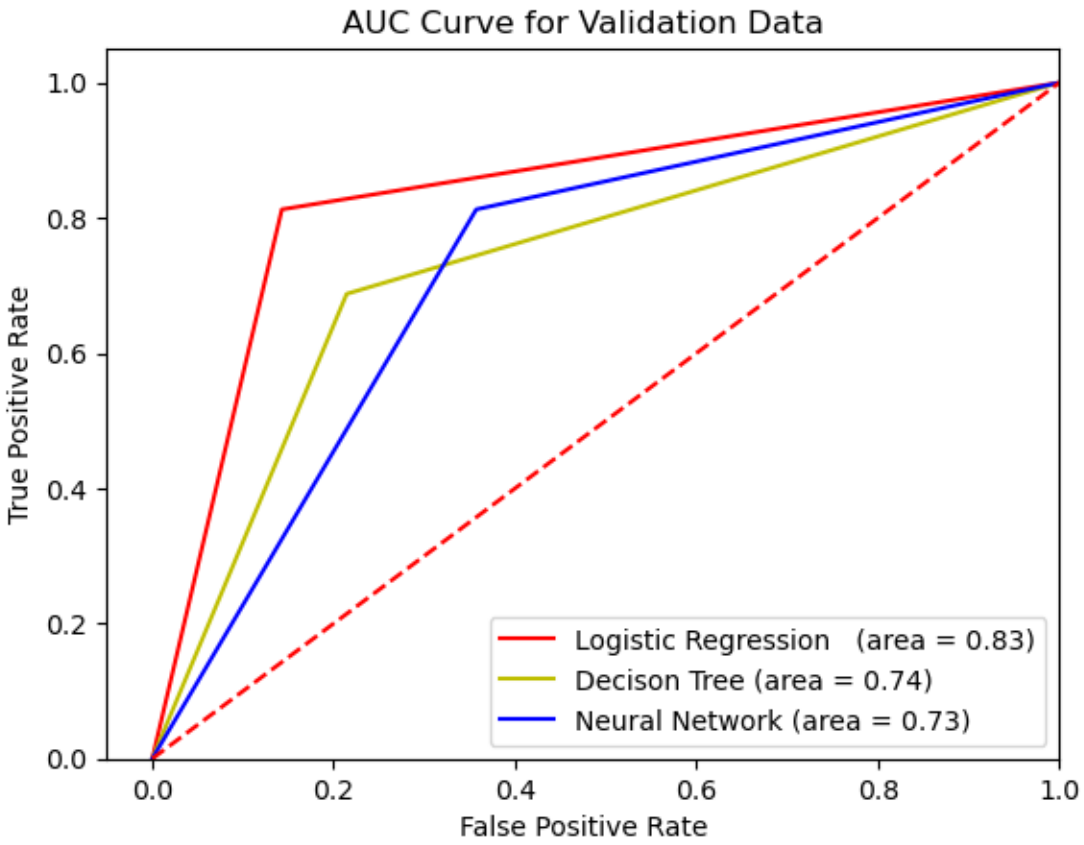
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.75 | 0.64 | 0.69 | 14 |
| 1.0 | 0.72 | 0.81 | 0.76 | 16 |
| accuracy | | | 0.73 | 30 |
| macro avg | 0.74 | 0.73 | 0.73 | 30 |
| weighted avg | 0.74 | 0.73 | 0.73 | 30 |


AUC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Without Hyperparameter (area = 0.74)
With Hyperparameter (area = 0.73)

# 9.0 Result and Errors

We investigated the receiver operating characteristic (ROC) curve for each model and discovered that the logistic regression model had a greater area under the curve (AUC) than the other models. This shows that the logistic regression model is better at telling the difference between positive and negative classes.

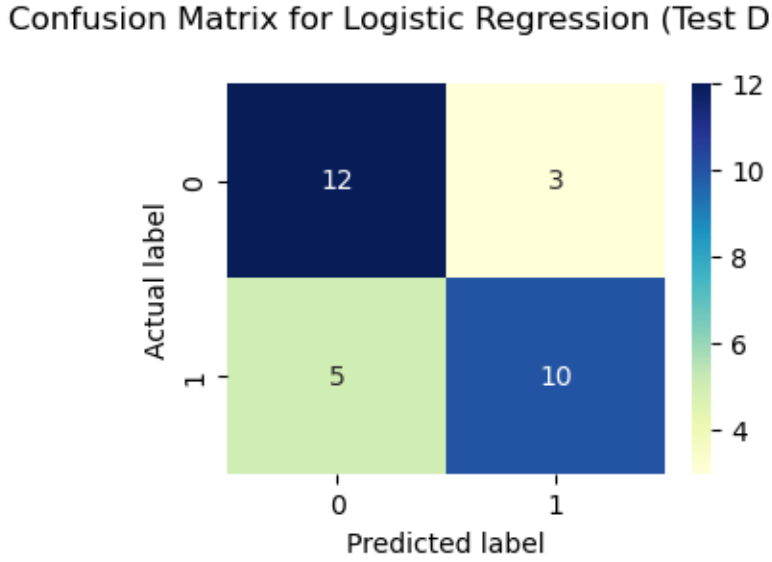
In addition, after a comprehensive examination of the classification report, we eventually decided to use logistic regression for our model. This decision was based on a number of important indicators, including AUC, cross-validation accuracy, precision, recall, and the F1 score. These performance indicators indicate that the logistic regression model can reliably predict the target variable and beat the other models we investigated.

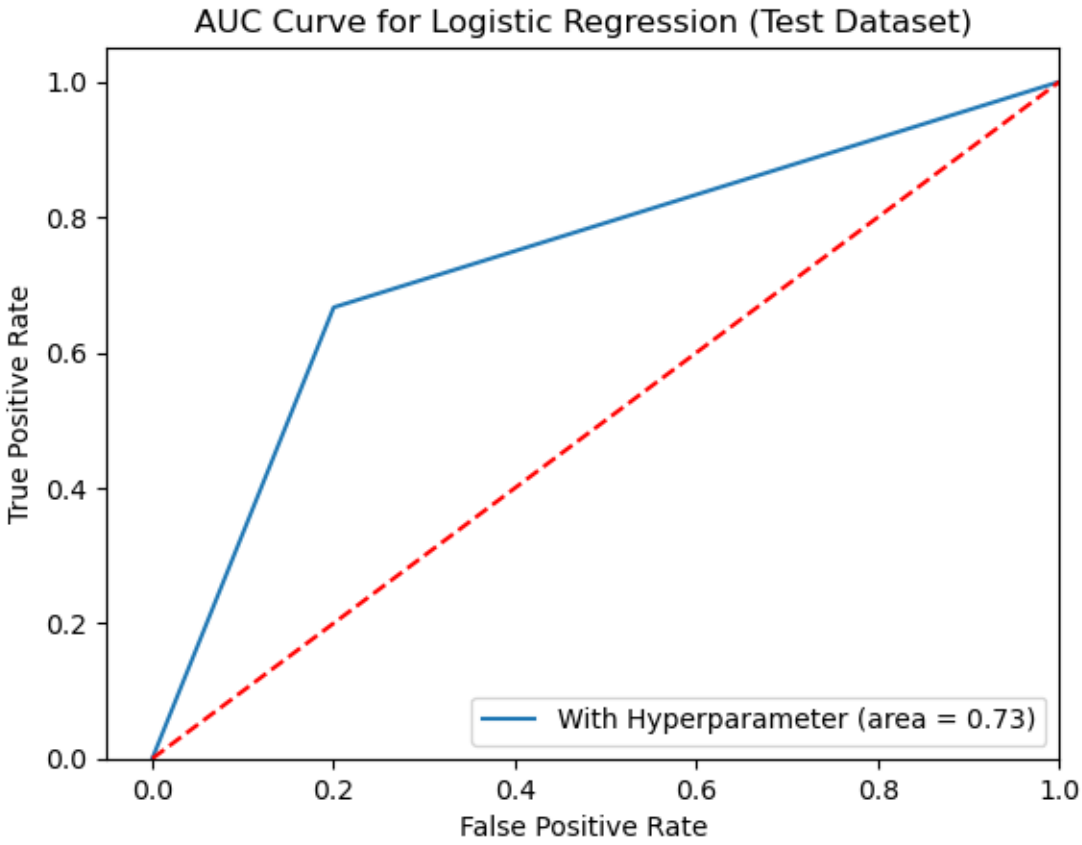



## 9.1 The Test Dataset's Outcome

Using the optimal hyperparameters gained throughout the tuning procedure, we finally assessed the logistic regression model on the test dataset. This evaluation led to the creation of a classification report and a confusion matrix, which show how well the model works on data that was not known before.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0.0 | 0.71 | 0.80 | 0.75 | 15 |
| 1.0 | 0.77 | 0.67 | 0.71 | 15 |
| accuracy | | | 0.73 | 30 |
| macro avg | 0.74 | 0.73 | 0.73 | 30 |
| weighted avg | 0.74 | 0.73 | 0.73 | 30 |



## 9.2 Analysis of the Result

On the validation dataset, the logistic regression model achieved an overall accuracy of 0.83, whereas the accuracy on the test dataset was 0.73. According to the confusion matrix, the model made five errors with the test dataset. In particular, it categorized three instances of class 1 as class 0 and five instances of class 0 as class 1. These errors can be interpreted as false positives and false negatives, respectively. Also, the model appears to be less accurate at predicting instances of class 1 than class 0, as demonstrated by class 1's lower recall score.

The “World of Bargains” retailer may wish to avoid false positives more than other types of error (i.e., when the model predicts that a store will perform well but it really performs poorly). This is due to the possibility that the client would experience financial losses, preventing them from generating revenue. On the other side, false negatives (i.e., when the model predicts that a store is not likely to perform well, but they actually do) may not be as harmful as false positives since they are less likely to result in a decrease in revenue.

Overall, the logistic regression model shows promising results with reasonable accuracy and good precision and recall scores. However, the model's performance can be further improved by considering other classification algorithms, feature engineering, or data preprocessing techniques.

# 10.0 Conclusion

This project was intended to provide logistic regression, decision trees, and neural network models to predict the performance of stores in a UK retail chain with over 100 locations. We utilized a dataset comprising information on 136 stores, such as staff number, location, and other variables. After examining the data and comparing the performances of the three models, we determined that logistic regression performed the best for this particular project.

The logistic regression model exhibited the greatest cross-validation accuracy, precision, recall, F1 score, and area under the AUC curve. In addition, it was possible to identify the variables that had the greatest impact on store performance, such as staff number, location, and competition score. Our investigation also showed several areas for enhancement, such as the need to normalize numerical data to a standard range in order to improve the accuracy and efficiency of the machine-learning model. We also recognized the possibility of enhancing the performance of our models via hyperparameter tuning.

Overall, our models provide useful insights into the aspects that drive store performance and can assist the owner of the retail chain in making educated decisions on business expansion. By utilizing data-driven tactics such as these, we can assist businesses in making smarter, more informed decisions and achieving greater success.

# 11.0 Appendix - Terminology

AIC: The Akaike Information Criterion is a statistical metric used to assess alternative models and decide which one best fits a given dataset based on the trade-off between model accuracy and complexity.

Artificial Neural Network: It uses layers of linked nodes, or artificial neurons, to learn from and forecast complicated data sets to simulate the human brain.

BIC: The "Bayesian Information Criterion," like the AIC, is a statistical metric used to evaluate models to find the best fit for a dataset based on model accuracy and complexity.

Binary classification: In machine learning, binary classification predicts one of two classes, typically 0 or 1.

Categorical variable:  A variable that represents a group or category of data.

Classification: Classification uses labeled training data to train a model to assign input data to one of many predefined classes or categories.

Coefficients: Coefficients are numerical values allocated to variables in a mathematical calculation or model.

Confusion matrix: This is a table used to evaluate the performance of a model for machine learning.

Continuous variable: Those that can take on any value within a specified range, such as height, weight, or temperature

Cross-Validation techniques: A technique that prevents overfitting and evaluates machine learning models.

Data analysis: Data analysis involves analyzing, cleansing, manipulating, and modeling data to find insights, make conclusions, and help with decision-making.

Data mining: Data mining uses statistical and computational tools to find patterns in large datasets.

Decision trees: A tree-like structure type of machine learning that divides data based on their properties or features over and over again.

Deep learning: A subtype of machine learning, that employs neural networks with several layers to extract high-level characteristics from complicated data like pictures, audio, and text.

Discrete variable: Those that can only take on specific values, such as the number of children in a family or the number of items in a basket.

Distributions: A distribution is the pattern of data points or values over a particular range. Measures like the mean, standard deviation, and skewness can reveal the data's central tendency, variability, and shape.

Ensemble learning: Ensemble learning takes the predictions of many models and combines them to improve performance. This often improves accuracy, robustness, and generalization.

Gaussian distribution: Commonly known as the "normal distribution," is a continuous probability distribution with a symmetrical, bell-shaped mean and standard deviation.

Grid Search: Grid search is a machine learning strategy for hyperparameter tuning that searches a specific parameter space for the optimum hyperparameter combination for a validation set, commonly using a predetermined performance metric like accuracy or the F1 score.

Heatmap: A heatmap is a graphical representation of data in which each cell represents a value or observation and the color indicates its magnitude or density.

Hyperparameter: The hyperparameter is a parameter that is specified before the training process and controls the algorithm's behavior, such as the learning rate, the number of hidden layers, or the regularization strength.

Intercept: An intercept is a point where a regression line or curve crosses the y-axis.

K-nearest neighbors: K-NN is a basic machine learning technique used for classification and regression that finds the K nearest data points to a new query point based on a distance measure and assigns the label or value of the majority of those nearest neighbors to the query point.

Leave One Out: LOO cross-validation uses each data point as the validation set and the rest as the training set. This makes n different training sets, where n is the number of data points.

Log-normal distribution: It is a continuous probability distribution with a mean and standard deviation of the logarithm of the interest variable.

Log scale: Log scaling calculates the logarithm of your values to compress a broad range into a narrow range.

Logistic Regression: A machine learning approach for binary classification tasks fits a logistic function to model the likelihood of the positive class given one or more input variables to predict a binary result.

Machine Learning: It is an area of artificial intelligence that uses statistical and computational methods to enable computer systems to automatically learn from data and improve their performance on a given job over time without being explicitly programmed.

Maximal absolute Scaler: Scale each characteristic by its absolute maximum value.

Maximum Likelihood Estimation: MLE is a statistical approach used to estimate the parameters of a probability distribution by finding the values that maximize the likelihood of observing the data given the distribution, provided the data are independent and identically distributed.

Min-Max Scale: It is a way to prepare data that rescales all features to the range [0, 1] by getting rid of the lowest value and dividing by the difference between the highest and lowest values.

Multicollinearity: It happens when two or more predictor variables are strongly connected, making it hard to figure out how each predictor variable is related to the response variable.

Multi-class Classification: It uses one or more input characteristics to predict a category target variable that has more than two classes.

Multi-layer Perceptron: MLP is an artificial neural network with many layers of linked nodes.

Neural networks: It's a type of machine learning model inspired by the brain that uses layers of linked nodes to learn complex patterns and correlations by performing mathematical operations on incoming data.

Normalize: Each row of the input data is transformed to have a Euclidean length of 1 as a result of the row-wise normalization.

Nominal variable: A type of categorical variable that represents a characteristic that can be divided into categories, but the categories do not have any inherent order or ranking.

Numeric variable: A type of variable that stands for a number, like a count or a ratio.

NumPy: A Python library for scientific computing and data analysis provides a powerful array of data structures for handling large and multi-dimensional datasets and a wide range of mathematical functions for numerical operation.

Ordinal variable: Those that have an order or ranking, but do not have an inherent meaning to the difference between the values. An example of an ordinal numeric variable in the provided data set is "Demographic Score".

Outliers: Data points that deviate from the remainder of a dataset owing to measurement or recording mistakes, infrequent events, or extreme values.

Outlier Resistance: A statistical or machine-learning method can give consistent and reliable results even when there are outlier values.

Overfitting: when a model learns both the underlying patterns and the noise and quirks of the training data, resulting in high variance and low bias.

Pattern Recognition: It is a branch of machine learning and artificial intelligence that uses patterns to recognize and classify things, events, and phenomena.

Prediction: It is using data to estimate or forecast an unknown value or result.

Predictive model: Predictive models use past and present data to forecast future results.

Random Forest Classifier: It is a supervised machine learning method that uses subsamples of a dataset to fit several decision tree classifiers.

Robust scaler: Utilize statistics that are robust against outliers for scale features.

ROC-AUC score: The ROC-AUC score shows how well a binary classifier can tell the difference between classes that are positive and classes that are negative.

R-squared: It indicates the proportion of the variance in the dependent variable that can be attributed to the independent variable.

Scikit-Learn: A Python machine learning framework that offers fast methods for classification, regression, clustering, and dimensionality reduction.

Skewed distribution: It has more data points, or weight, on one side of the mean.

Standard scaler (z-score): To standardize the features, remove the mean and scale to unit variance.

Statistical Modeling: It generates sample data and real-world predictions by applying mathematical models and assumptions to observed data and similar data from a larger population.

Supervised Learning: It utilizes labeled datasets to train algorithms to classify data or predict outcomes.

Variables: A variable is something that can fluctuate, especially in an unpredictable manner.

# Reference:


Andersen, Charlotte Møller, and Rasmus Bro. "Variable selection in regression—a tutorial." Journal of chemometrics 24.11-12 (2010): 728-737.

Beyeler, Michael. *Machine Learning for OpenCV*. Packt Publishing Ltd, 2017.

Breiman, Leo. "Random forests." *Machine learning* 45 (2001): 5-32.

Devore, Jay L. "Probability and Statistics for Engineering and the Sciences." (2008).

Draper, Norman R., and Harry Smith. *Applied regression analysis*. Vol. 326. John Wiley & Sons, 1998.

Gökhan, A. K. S. U., Cem Oktay Güzeller, and Mehmet Taha Eser. "The effect of the normalization method used in different sample sizes on the success of artificial neural network model." *International Journal of Assessment Tools in Education* 6.2 (2019): 170-192.

Hatwell, Julian, Mohamed Medhat Gaber, and R. Muhammad Atif Azad. "CHIRPS: Explaining random forest classification." Artificial Intelligence Review 53 (2020): 5747-5788.

Hosmer Jr, David W., Stanley Lemeshow, and Rodney X. Sturdivant. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.

Kang, Myeongsu, and Jing Tian. "Machine Learning: Data Pre-processing." Prognostics and Health Management of Electronics: Fundamentals, Machine Learning, and the Internet of Things (2018): 111-130.

Komarek, Paul. Logistic regression for data mining and high-dimensional classification. Carnegie Mellon University, 2004.

Kotsiantis, Sotiris B., Ioannis Zaharakis, and P. Pintelas. "Supervised machine learning: A review of classification techniques." Emerging artificial intelligence applications in computer engineering 160.1 (2007): 3-24.

Kotsiantis, Sotiris B. "Decision trees: a recent overview." Artificial Intelligence Review 39 (2013): 261-283.

Parmar, Aakash, Rakesh Katariya, and Vatsal Patel. "A review on random forest: An ensemble classifier." International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018. Springer International Publishing, 2019.

Peng, Chao-Ying Joanne, Kuk Lida Lee, and Gary M. Ingersoll. "An introduction to logistic regression analysis and reporting." The journal of educational research 96.1 (2002): 3-14.

Nick, Todd G., and Kathleen M. Campbell. "Logistic regression." Topics in biostatistics (2007): 273-301.

Rokach, Lior. "Decision forest: Twenty years of research." Information Fusion 27 (2016): 111-125.

Schröer, Christoph, Felix Kruse, and Jorge Marx Gómez. "A systematic literature review on applying CRISP-DM process model." *Procedia Computer Science* 181 (2021): 526-534.

Singh, Dalwinder, and Birmohan Singh. "Investigating the impact of data normalization on classification performance." Applied Soft Computing 97 (2020): 105524.

Sharma, Sagar, Simone Sharma, and Anidhya Athaiya. "Activation functions in neural networks." Towards Data Sci 6.12 (2017): 310-316.

Shepherd, Adrian J. Second-order methods for neural networks: Fast and reliable training methods for multi-layer perceptrons. Springer Science & Business Media, 2012.

Wong, Tzu-Tsung. "Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation." Pattern Recognition 48.9 (2015): 2839-2846.

Wirth, Rüdiger, and Jochen Hipp. "CRISP-DM: Towards a standard process model for data mining." Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining. Vol. 1. 2000.

Zabinsky, Zelda B. "Random search algorithms." Department of Industrial and Systems Engineering, University of Washington, USA (2009).

Zhang, Cha, and Yunqian Ma, eds. Ensemble machine learning: methods and applications. Springer Science & Business Media, 2012.