**Case Study 1: Who survives the Titanic?**

Mohammad Ismail Ashiq Aslam

**CMU505; Machine Learning**

Table of Contents

[Introduction 3](#_Toc166218188)

[Background 3](#_Toc166218189)

[The stages in finding a solution are: 4](#_Toc166218190)

[State the Problem 4](#_Toc166218191)

[Problem Breakdown 4](#_Toc166218192)

[Problem Solution 4](#_Toc166218193)

[Aim 4](#_Toc166218194)

[Set up the System 4](#_Toc166218195)

[Installing Software 4](#_Toc166218196)

[Importing Libraries: 5](#_Toc166218197)

[Import the Dataset(s) 8](#_Toc166218198)

[Gender\_Syubmission.csv 8](#_Toc166218199)

[test.csv 9](#_Toc166218200)

[train.csv 9](#_Toc166218201)

[Gender\_Syubmission\_copy.csv 10](#_Toc166218202)

[Test\_copy.csv 11](#_Toc166218203)

[Train\_copy.csv 11](#_Toc166218204)

[Perform Data Exploratory Analysis 12](#_Toc166218205)

[Initial examination of the data 12](#_Toc166218206)

[Identify missing data items 16](#_Toc166218207)

[Perform Data Exploratory Analysis 17](#_Toc166218208)

[Data Preporocessing 19](#_Toc166218209)

[Identify Features to Drop from Dataset(s) 19](#_Toc166218210)

[Resolve missing data items 21](#_Toc166218211)

[Convert Feature(s) 22](#_Toc166218212)

[Create Categories 23](#_Toc166218213)

[Create New Feature(s)] 24](#_Toc166218214)

[Build Machine Learning Model(s) 26](#_Toc166218215)

[Create Training Data 26](#_Toc166218216)

[Create Test Data 27](#_Toc166218217)

[Create Machine Learning Model(s) 28](#_Toc166218218)

[Stochastic Gradient Descent (SGD) 28](#_Toc166218219)

[Random Forest 29](#_Toc166218220)

[Logistic Regression 30](#_Toc166218221)

[K Nearest Neighbor (KNN) 31](#_Toc166218222)

[Naïve Bayes 32](#_Toc166218223)

[Perception 33](#_Toc166218224)

[Linear Support Vector Machine 34](#_Toc166218225)

[Decision Tree 35](#_Toc166218226)

[Decide which is the Best Model 36](#_Toc166218227)

[Calculate Best Model 36](#_Toc166218228)

[K-Fold Cross Validation 37](#_Toc166218229)

[Feature Importance 38](#_Toc166218230)

[Revise Dataset by dropping features that do not play a significant role 39](#_Toc166218231)

[Training the Best Model Again 40](#_Toc166218232)

[Hyperparameter Tuning 41](#_Toc166218233)

[Perform Hyperparameter Tuning 41](#_Toc166218234)

[Test New Parameters 42](#_Toc166218235)

[Evaluation 43](#_Toc166218236)

[Generate Confusion Matrix 43](#_Toc166218237)

[Calculate Precision and Recall 44](#_Toc166218238)

[Calculate F-Score 45](#_Toc166218239)

[Generate Precision Recall Curve 46](#_Toc166218240)

[Generate ROC AUC Curve 47](#_Toc166218241)

[Conclusion 48](#_Toc166218242)

# Introduction

In this case study, the challenge, is to predict whether a passenger on the titanic would have been survived or not.

# Background

The RMS Titanic was a British passenger liner that sank in the North Atlantic Ocean in the early morning hours of 15 April 1912, after it collided with an iceberg during its maiden voyage from Southampton to New York City. There were an estimated 2,224 passengers and crew aboard the ship, and more than 1,500 died, making it one of the deadliest commercial peacetime maritime disasters in modern history. The RMS Titanic was the largest ship afloat at the time it entered service and was the second of three Olympic-class ocean liners operated by the White Star Line. The Titanic was built by the Harland and Wolff shipyard in Belfast. Thomas Andrews, her architect, died in the disaster.

A model of a ship

Description automatically generated

# The stages in finding a solution are:

## State the Problem

### Problem Breakdown

The task we're addressing involves determining if a passenger on the Titanic would have survived the catastrophe that happened in 1912.

### Problem Solution

In order effectively calculate the probability of survival, machine learning techniques must be used to assess a variety of factors, such as passenger statistics, seat, and location.

Our goal is to develop predictive algorithms that can calculate the odds that individuals would survive by looking through and modifying previous records from the passenger registers of the Titanic.

### Aim

The aim of this research is to comprehend the factors that influence survival rates and create models that could be helpful in similar circumstances.

## Set up the System

### Installing Software

Downloading Spyder IDE:

We installed the Spyder IDE, a Python IDE for development, a very powerful IDE which is perfect for working on Machine Learning projects as it offers an extensive collection of tools for developing code, testing, and research.

Downloading Anaconda:

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedSpyder leverages Anaconda's package management and deployment features, which streamline the process of creating a scientific Python environment. Since Anaconda includes a wide range of scientific libraries and tools with Spyder, it's a very good option for machine learning, data analysis, and scientific research.

### Importing Libraries:

#### Import Data Processing Libraries

A screenshot of a computer

Description automatically generated

In this Case Study we will be using the panda library from the Data Processing libraries, and we will use the following datasets: gender\_submission.csv, test.csv, train.csv.

#### Import Linear Algebra libraries

A black rectangular object with a white stripe

Description automatically generated

From the Linear Algebra libraries, we will be using numpy which is a mathematical library which will allow us to use functions, operations and statistics. To shorten the name of the library we will use np instead, as we can see numpy is being imported as np.

#### Import Data Visualisation Libraries

A screenshot of a computer

Description automatically generated

We begin setting our environment for analysis and visualisation.

The first step is to import the libraries needed for processing and visualising data.

To generate simple plots, import matplotlib.pyplot; for more intricate statistical visualisations, import seaborn.

Another important tool for data manipulation and analysis that we include in our code is the pandas library, which we shorten to 'pd' for convenience.

To indicate the location of the CSV files containing the Titanic dataset, a variable called directory\_path is created.

We load data from three CSV files—gender\_submission.csv, test.csv, and train.csv into corresponding pandas Data Frames: gender\_submission, test\_data, and train\_data using the pd.read\_csv() function.

#### Import Algorithms

A screenshot of a computer program

Description automatically generated

The code above imports different machine learning algorithms from Scikit-learn, a popular Python predictive modelling toolkit. These algorithms are essential for evaluating past data and forecasting outcomes, such the chance that people on the Titanic will survive.

A wide range of machine learning techniques are included by the imported classifiers, which include Gaussian Naive Bayes, Decision Tree, Random Forest, Gradient Boosting, K-Neighbors, Logistic Regression, Support Vector Classifier (SVC), and Multi-layer Perceptron (MLP). All of the classifiers are suitable for different parts of the dataset since they have different benefits and follow different assumptions.

## Import the Dataset(s)

### Gender\_Syubmission.csv

A screenshot of a computer

Description automatically generated

Here we are importing the dataset gender submission. We can see three columns one, representing the index, another containing the primary key, PassingerID. And the last column shows who survived, 1 indicates they survived and 0 indicates they did not survive. We will use this data to see what factors influenced the survival of the passengers.

### test.csv

A screenshot of a computer

Description automatically generated

Here we are importing the dataset test. This dataset is essential to the analysis of the case study. Data including passenger class, name, sex, age, number of parents/children, siblings, and spouses travelling with the ticket, fare, cabin number, and port of departure are included in the dataset.

### train.csv

A screenshot of a computer

Description automatically generated

Here we are importing the dataset train. We can see that women and children had a higher chance of surviving by comparing the Survived column with the Sex and Age columns.

### Gender\_Syubmission\_copy.csv

A screenshot of a computer

Description automatically generated

The gender\_submission\_copy file can be effectively utilized to assess the accuracy of models developed using the train.csv and test.csv datasets.

The gender\_submission\_copy data frame makes it possible to assess the model's prediction accuracy in relation to a fundamental criteria in a simple and clear manner. By using this strategy, the study not only evaluates the performance of the trained model but also highlights the benefits of adding advanced features and approaches.

### Test\_copy.csv

A screenshot of a computer

Description automatically generated

When using the final predictive model to predict survival outcomes, it is necessary to utilise a test\_data\_copy in order to preserve the original test.csv dataset.

We can do preprocessing and feature modification to the training data by making a duplicate of the test dataset. To prevent differences that might distort the model's performance, adjustments such as scaling numerical features, encoding categorical variables, or managing missing values must be implemented uniformly across training and testing datasets.

### Train\_copy.csv

A screenshot of a computer

Description automatically generated

Creating a copy of the training data, such as train\_data\_copy, is crucial to protect the integrity of the original dataset, especially to predict survival outcomes. This maintains the flexibility to experiment and refine the analysis without altering the original data.

## Perform Data Exploratory Analysis

### Initial examination of the data

#### Examine initial rows

A screenshot of a computer

Description automatically generated

This code check the first 5 rows of each dataset and prints it with all the columns and a brief message to identify the data frame.

#### Initial Examination of Gender\_Submission.csv

|  |  |  |
| --- | --- | --- |
| index | PassengerId | Survived |
| 0 | 892 | 0 |
| 1 | 893 | 1 |
| 2 | 894 | 0 |
| 3 | 895 | 0 |
| 4 | 896 | 1 |

Each passenger's ID is shown in column "PassengerID," and the column "Survived," which indicates whether a passenger survived (1) or did not survive (0). This data makes it easier to determine which passengers made it out alive and which did not.

#### Initial Examination of Test.csv

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| index | PassengerId | Pclass | Cabin | Embarked |
| 0 | 892 | 3 | NaN | Q |
| 1 | 893 | 3 | NaN | S |
| 2 | 894 | 2 | NaN | Q |
| 3 | 895 | 3 | NaN | S |
| 4 | 896 | 3 | NaN | S |

The test dataset has many fields: "embarked," which displays the port where passengers boarded; "passengerid," which identifies each person; "Pclass," which indicates the ticket class; and "cabin number," which specifies each passenger's compartment.

#### Initial Examination of Train.csv

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Passenger ID | Survived | Pclass | Fare | Cabin | Embarked |
| 1 | 0 | 3 | 7.2500 | NaN | S |
| 2 | 1 | 1 | 71.2833 | C85 | C |
| 3 | 1 | 3 | 7.9250 | NaN | S |
| 4 | 1 | 1 | 53.1000 | C123 | S |
| 5 | 0 | 3 | 8.0500 | NaN | S |

There are several columns in the train dataset, including "passenger id," "survived," "Pclass," "fare," "cabin," and "embarked." Details on the first passengers, such as their ticket class, fare, cabin, and embarkation port, are included in the dataset. Missing data is shown in the "Cabin" column with NaN (Not a Number) values. In order to analyse the factors that affected the survival rates during the Titanic accident, this dataset is crucial. Understanding how elements like boarding places and economic standing may have affected the chance of survival may be gained by examining elements like ticket class and cost. This dataset closely relates to the case study's goals by providing the foundation for prediction algorithms that calculate the likelihood that passengers aboard the Titanic survived.

#### Examine data types

A screenshot of a computer

Description automatically generated

This code prints the Data types for each data frame column and row. This will help to understand the different data types there are and divide the data accordingly if necessary.

#### Gender Submission data type

|  |  |
| --- | --- |
| Column | Data Type |
| PassengerId | int64 |
| Survived | int64 |

The Gender Submission Dataset is broken down into two columns, "PassengerId" and "Survived," both of which are integer types ("int64"). The passenger's unique identification number and expected survival rate are shown in these columns, accordingly. In the event that the "Survived" column displays as a "object" data type, it suggests that survival results may be encoded as strings rather than numbers, requiring conversion in order to enable precise predictive analysis. This knowledge is essential to getting the dataset ready for accurate survival prediction.

#### Test dataset data type

|  |  |
| --- | --- |
| Column | Data Type |
| PassengerId | int64 |
| Pclass | int64 |
| Name | object |
| Sex | object |
| Age | float64 |
| SibSp | int64 |
| Parch | int64 |
| Ticket | object |
| Fare | float64 |
| Cabin | object |
| Embarked | object |

The table lists the Test Dataset's column names and associated data types; each row describes a distinct passenger attribute. While columns like "Name," "Sex," "Ticket," "Cabin," and "Embarked" are classified as qualitative (object), columns like "PassengerId," "Pclass," "Age," "SibSp," "Parch," and "Fare" are quantitative. This dataset is essential for evaluating how well prediction models built with the Train Dataset perform.

#### Train dataset data type

|  |  |
| --- | --- |
| Column | Data Type |
| PassengerId | int64 |
| Survived | int64 |
| Pclass | int64 |
| Name | object |
| Sex | object |
| Age | float64 |
| SibSp | int64 |
| Parch | int64 |
| Ticket | object |
| Fare | float64 |
| Cabin | object |
| Embarked | object |

The table has two columns: "Data Type," which shows what kind of data each column includes, and "Column," which gives the names of characteristics in the Train Dataset that reflect different aspects of the Titanic passengers. "Name," "Sex," "Ticket," "Cabin," and "Embarked" are classified as category or text data (object), whereas "PassengerId," "Survived," "Pclass," "SibSp," and "Parch" are numerical data (integers). In order to help analysts document the types of data that are there and set the stage for additional research to determine what factors affected survival on the Titanic, this data types table is an essential first step in the data exploration process.

### Identify missing data items

A screenshot of a computer

Description automatically generated

This code check the data frames train data, test data, gender submission. It uses the library panda to check for missing values in the data frame and shows how many values are missing in each column, it also fetches for unidentified data and identifies it.

#### Missing values in gender\_submission\_data:

|  |  |
| --- | --- |
| Passenger ID | 0 |
| Survived | 0 |
| dtype | int64 |

Every passenger has a unique ID and a recorded survival status, as the table verifies that the dataset's "Passenger ID" and "Survived" columns have no missing records.

#### Missing values in test\_data:

|  |  |
| --- | --- |
| Passenger ID | 0 |
| Pclass | 0 |
| Name | 0 |
| Sex | 0 |
| Age | 86 |
| SibSp | 0 |
| Parch | 0 |
| Ticket | 0 |
| Fare | 1 |
| Cabin | 327 |
| Embarked | 0 |
| dtype | int64 |

The "Age" column has 86 entries missing, the "Fare" column has 1 entry missing, and the "Cabin" column has 327 items missing. The table illustrates missing values across various columns in the dataset.

#### Missing values in train\_data:

|  |  |
| --- | --- |
| Passenger ID | 0 |
| Survived | 0 |
| Pclass | 0 |
| Name | 0 |
| Sex | 0 |
| Age | 177 |
| SibSp | 0 |
| Parch | 0 |
| Ticket | 0 |
| Fare | 0 |
| Cabin | 687 |
| Embarked | 2 |
| dtype | Int64 |

The table shows how many missing values are in each column of the train\_data DataFrame. There are large gaps in the columns "Age" (which has 177 missing values), "Cabin" (687), and "Embarked" (2). These missing entries represent gaps in the training dataset, which must be filled to create predictive models that are reliable. During data preprocessing, techniques like imputation—which fills in missing values—and the elimination of rows with incomplete data are used.

### Perform Data Exploratory Analysis

A screenshot of a computer

Description automatically generated

This code uses summary statistics to perform exploratory data analysis on the Titanic dataset. The pandas library is imported, file paths are specified, and train.csv, test.csv, and gender\_submission.csv are loaded into the appropriate DataFrames. It creates and presents summary statistics for every DataFrame using the describe() method, providing information about data variability and distribution that is necessary for additional data preprocessing and survival modelling.

#### Summary statistics of gender\_submission\_copy:

|  |  |  |
| --- | --- | --- |
|  | Passenger ID | Survived |
| Count | 418.000000 | 418.000000 |
| Mean | 1100.500000 | 0.363636 |
| std | 120.810458 | 0.481622 |
| Min | 892.000000 | 0.000000 |
| 25% | 996.250000 | 0.000000 |
| 50% | 1100.500000 | 1.000000 |
| 75% | 1204.750000 | 1.000000 |
| Max | 1309.000000 | 1.000000 |

Summary statistics are shown in the table for the gender\_submission dataset's "Passenger ID" and "Survived" columns. The terms count, mean, and standard deviation are used to represent the total number of non-null entries, the average of values, and the variability around the mean, respectively. Along with percentiles(25,50, and 75%), which indicate the values below which particular percentages of the data fall, the table also includes the minimum and maximum values. The median is represented by 50%.

#### Summary statistics of test\_data:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Passenger ID | Pclass | Age | SibSp | Parch | Fare |
| Count | 418.000000 | 418.000000 | 332.000000 | 418.000000 | 418.000000 | 417.000000 |
| Mean | 1100.500000 | 2.265550 | 30.272590 | 0.447368 | 0.392344 | 35.627188 |
| std | 120.810458 | 0.841838 | 14.181209 | 0.896760 | 0.981429 | 55.907576 |
| min | 892.000000 | 1.000000 | 0.170000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 996.250000 | 1.000000 | 21.000000 | 0.000000 | 0.000000 | 7.895800 |
| 50% | 1100.500000 | 3.000000 | 27.000000 | 0.000000 | 0.000000 | 14.454200 |
| 75% | 1204.750000 | 3.000000 | 39.000000 | 1.000000 | 0.000000 | 31.500000 |
| Max | 1309.000000 | 3.000000 | 76.000000 | 8.000000 | 9.000000 | 512.329200 |

The table shows the summary statistics for a number of the test dataset's numerical columns, including Passenger ID, Pclass, Age, SibSp, Parch, and Fare. The following details are provided for each statistic: The count value, displays the quantity of non-null values; The average value within each column is indicated by the term "mean." The dispersion around the mean is measured by standard deviation (Std); The observed range is delineated by the values of the minimum (Min) and maximum (Max); the data is partitioned into quartiles by the percentiles (25%, 50%, and 75%), with 50% denoting the median.

#### Summary statistics of train\_data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Passenger ID | Survived | Pclass | SibSp | Parch |
| Count | 891.000000 | 891.000000 | 891.000000 | 891.000000 | 891.000000 |
| Mean | 446.000000 | 0.383838 | 2.308642 | 0.52308642 | 0.381594 |
| min | 257.353842 | 0.486592 | 0.836071 | 1.102743 | 0.806057 |
| std | 1.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 |
| 25% | 223.000000 | 0.000000 | 2.000000 | 0.000000 | 0.000000 |
| 50% | 446.000000 | 0.000000 | 3.000000 | 0.000000 | 0.000000 |
| 75% | 668.000000 | 1.000000 | 3.000000 | 1.000000 | 0.000000 |
| Max | 891.000000 | 1.000000 | 3.000000 | 8.000000 | 6.000000 |

For each of the dataset's primary numerical columns—Passenger ID, Survived, Pclass, SibSp, and Parch—summary statistics are included in the table. The features include: Count, which displays entries that are not null; Mean, which shows average values; Min and Max, which denote extremes; and Std, which shows variability. Data is then divided into quartiles by percentiles (25%, 50%, and 75%). Understanding the distribution and variability of characteristics like as passenger IDs, survival status, and familial ties is crucial for developing accurate predicting models of Titanic passenger survival.

## Data Preporocessing

### Identify Features to Drop from Dataset(s)

A screenshot of a computer

Description automatically generated

This code drops features from the gender submission, train data, and test data.

#### Summary Statistics of gender\_submission\_copy:

|  |  |  |
| --- | --- | --- |
|  | Passenger ID | Survived |
| Count | 418.000000 | 418.000000 |
| Mean | 1100.500000 | 0.363636 |
| std | 120.810458 | 0.481622 |
| Min | 892.000000 | 0.000000 |
| 25% | 996.250000 | 0.000000 |
| 50% | 1100.500000 | 0.000000 |
| 75% | 1204.750000 | 1.000000 |
| max | 1309.000000 | 1.000000 |

For the "gender\_submission\_copy" dataset, which includes passenger IDs and their survival status (0 for not surviving, 1 for survived), the code provides summary statistics. These numbers allowed us to see how passengers' survival results are distributed.

#### Summary Statistics of test\_data:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Passenger ID | Pclass | Age | SibSp | Parch | Fare |
| Count | 418.000000 | 418.000000 | 332.000000 | 418.000000 | 418.000000 | 417.000000 |
| Mean | 1100.500000 | 2.265550 | 30.272590 | 0.447368 | 0.392344 | 35.627188 |
| std | 120.810458 | 0.841838 | 14.181209 | 0.896760 | 0.981429 | 55.907576 |
| Min | 892.000000 | 1.000000 | 0.170000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 996.250000 | 1.000000 | 21.000000 | 0.000000 | 0.000000 | 7.895800 |
| 50% | 1100.500000 | 3.000000 | 27.000000 | 0.000000 | 0.000000 | 14.454200 |
| 75% | 1204.750000 | 3.000000 | 39.000000 | 1.000000 | 0.000000 | 31.500000 |
| max | 1309.000000 | 3.000000 | 76.000000 | 8.000000 | 9.000000 | 512.329200 |

Summary statistics for passenger ID, class, age, number of parents/children, siblings/spouses, and fare are all calculated by the code for the RMS Titanic. The demographic details and other factors affecting the chances of survival are better understood thanks to this investigation.

#### Summary Statistics of train\_data:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Passenger ID | Survived | Pclass | Age | SibSp | Parch | Fare |
| Count | 891.000000 | 891.000000 | 891.000000 | 714.00000 | 891.000000 | 891.000000 | 891.000000 |
| Mean | 446.000000 | 0.383838 | 2.308642 | 29.699118 | 0.523008 | 0.381594 | 32.204208 |
| std | 257.353842 | 0.486592 | 0.836071 | 14.526497 | 1.102743 | 0.806057 | 49.693429 |
| Min | 1.000000 | 0.000000 | 1.000000 | 0.420000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 223.500000 | 0.000000 | 2.000000 | 20.125000 | 0.000000 | 0.000000 | 7.910400 |
| 50% | 446.000000 | 0.000000 | 3.000000 | 28.000000 | 0.000000 | 0.000000 | 14.454200 |
| 75% | 668.500000 | 1.000000 | 3.000000 | 38.000000 | 1.000000 | 0.000000 | 31.000000 |
| max | 891.000000 | 1.000000 | 3.000000 | 80.000000 | 8.000000 | 6.000000 | 512.329200 |

Essential passenger information such as passenger ID, survival status, class, age, number of parents/children, siblings, and spouses on board, as well as fare, is calculated by the code.

### Resolve missing data items

A screenshot of a computer program

Description automatically generated

This code shows the missing values in the test\_data and train\_data.

#### Missing values in test\_data:

|  |  |
| --- | --- |
| Passenger ID | 0 |
| Pclass | 0 |
| Name | 0 |
| Sex | 0 |
| Age | 0 |
| SibSp | 0 |
| Parch | 0 |
| Ticket | 0 |
| Fare | 0 |
| Cabin | 327 |
| Embarked | 0 |
| dtype | int64 |

The table displays a method very similar to that of the train dataset, describes the resolved missing values in the test\_data DataFrame. With the exception of "Cabin," all columns currently contain zero missing values. While the missing values in the "Cabin" column were not handled in this preprocessing stage and are still unresolved, the missing entries in the "Age" column were filled using the median value.

#### Missing values in train\_data:

|  |  |
| --- | --- |
| Passenger ID | 0 |
| Survived | 0 |
| Pclass | 0 |
| Name | 0 |
| Sex | 0 |
| Age | 0 |
| SibSp | 0 |
| Parch | 0 |
| Ticket | 0 |
| Fare | 0 |
| Cabin | 687 |
| Embarked | 0 |
| dtype | int64 |

All columns now have zero missing values, with the exception of "Cabin," according to the table that updates the train\_data data frame's missing value status. In the "Embarked" column, the most commonly occurring value was used to fill in missing entries, while the median value was used to fill in those in the "Age" column. Here, 687 missing values are retained in the "Cabin" column because it is not processed during this preprocessing phase.

### Convert Feature(s)

A screenshot of a computer program

Description automatically generated

This code processed all the data so it can be used for calculations. It also encodes categorical feature, fills missing values and scales numerical features,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Passenger | Scaled Age | Scaled Fare | Pclass\_2 | Pclass\_3 | Female | Embarked\_Q | Embarked\_S |
| 1 | -0.58162831 | -0.50329106 | 0 | 0 | 1 | 0 | 1 |
| 2 | 0.65865194 | 0.73474365 | 0 | 0 | 0 | 0 | 0 |
| 3 | -0.27155825 | -0.49024046 | 0 | 0 | 1 | 0 | 1 |

In the table, each row corresponds to a single passenger, and the columns show different parameters such as ticket class, gender, port, along with scaled and imputed values for age and fare. These  processes are essential for creating predictive models that look for the probability of survival.

### Create Categories

A screenshot of a computer

Description automatically generated

This code processes the dataset, groups all passengers with same characteristics.

#### Train.csv Dataset:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Passenger ID | Pclass | Age\_Category | Fare\_Category | Family\_Size\_Category | Title |
| 892 | 3 | Adult | Low | Solo | Mr |
| 893 | 3 | Adult | Low | Small | Mrs |
| 894 | 2 | Senior | Medium | Solo | Mr |
| 895 | 3 | Adult | Medium | Solo | Mr |
| 896 | 3 | Adult | Medium | Small | Mrs |

The table shows processed information of the train dataset, grouping passengers according to age, fare, size of family, and title.

#### Test.csv Dataset:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Passenger ID | Pclass | Age\_Category | Fare\_Category | Family\_Size\_Category | Title |
| 892 | 3 | Adult | Low | Solo | Mr |
| 893 | 3 | Adult | Low | Small | Mrs |
| 894 | 2 | Senior | Medium | Solo | Mr |
| 895 | 3 | Adult | Medium | Solo | Mr |
| 896 | 3 | Adult | Medium | Small | Mrs |

The table displays pre-processed information from the test dataset, arranging travellers according to age, fare, size of family, and title.

### Create New Feature(s)]

A screenshot of a computer program

Description automatically generated

This code creates two new features, Family\_Members and Family\_Survival. This processes whether a passenger with family on board had more chances to survive or not.

#### Train Dataset:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Passenger ID | Survived | Pclass | Name | Sex | Age | SibSp | Parch | Ticket | Fare | Cabin | Embarked | Family\_Members | Family\_Survival |
| 1 | 0 | 3 | Braund, Mr. Owen Harris | Male | 22 | 1 | 0 | A/5 21171 | 7.25 | NaN | S | 2 | 0 |
| 2 | 1 | 1 | Cumings, Mrs. John Bradley (Florence Briggs Thayer) | Female | 38 | 1 | 0 | PC 17599 | 71.28 | C85 | C | 2 | 0 |
| 3 | 1 | 3 | Heikkinen, Miss. Laina | Female | 26 | 0 | 0 | STON/O2. 3101282 | 7.92 | NaN | S | 1 | 0 |
| 4 | 1 | 1 | Futrelle, Mrs. Jacques Heath (Lily May Peel) | Female | 35 | 1 | 0 | 113803 | 53.1 | C123 | S | 2 | 1 |
| 5 | 0 | 3 | Allen, Mr. William Henry | Male | 35 | 0 | 0 | 373450 | 8.05 | NaN | S | 2 | 0 |

The attached table adds "Family\_Members" and "Family\_Survival," two more features to the dataset. By adding one to include the passenger and adding the sum of the "SibSp" (siblings/spouses) and "Parch" (parents/children) columns, the "Family\_Members" feature counts the number of family members that each passenger has on board. If someone has the same ticket number and surname as you, they are considered family members, and the "Family\_Survival" function lets you know if any of them made it through the calamity. If at least one member of the family survived, it is set to 1, otherwise it is set to 0.

These characteristics allow for a detailed knowledge of the dynamics of families on board the Titanic. They are also necessary for predictive modelling and further research to determine the factors influencing survivors' odds of surviving the disaster.

#### Test Dataset:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Passenger ID | Pclass | Name | Sex | Age | SibSp | Parch | Fare | Cabin | Embarked | Family\_Size | Family\_Survial |
| 892 | 3 | Kelly, Mr. James | Male | 34.5 | 0 | 0 | 7.83 | NaN | Q | 1 | 0 |
| 893 | 3 | Wilkes, Mrs. James (Ellen Needs) | Female | 47 | 1 | 0 | 7 | NaN | S | 2 | 0 |
| 894 | 2 | Myles, Mr. Thomas Francis | Male | 62 | 0 | 0 | 9.69 | NaN | Q | 1 | 0 |
| 895 | 3 | Wirz, Mr. Albert | Male | 27 | 0 | 0 | 8.69 | NaN | S | 1 | 0 |
| 896 | 3 | Hirvonen, Mrs. Alexander (Helga E Lindqvist) | Female | 22 | 1 | 1 | 12.29 | NaN | S | 3 | 0 |

There are two new columns in the modified dataset: "Family\_Members" and "Family\_Survival." In order to highlight family size, "Family\_Members" computes each passenger's total onboard family count, taking into account spouses, parents, siblings, kids, and oneself. "Family\_Survival" denotes the presence of any surviving family members; 1 indicates a survival and 0 indicates non-survival. These contributions show the passenger families dynamics and how those interactions have affected their odds of surviving.

## Build Machine Learning Model(s)

### Create Training Data

A screenshot of a computer

Description automatically generated

This code trains the dataset by analysing, encoding variables, handling missing data and creating new features.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Passenger | ID | Pclass | Sex | Age | Parch | Fare | Embarked | Family Size |
| 331 | 332 | 1 | 1 | 45.5 | 0 | 28.5000 | 2 | 1 |
| 733 | 734 | 2 | 1 | 23.0 | 0 | 13.0000 | 2 | 1 |
| 382 | 383 | 3 | 1 | 32.0 | 0 | 7.9250 | 2 | 1 |
| 704 | 705 | 3 | 1 | 26.0 | 0 | 7.8542 | 2 | 2 |
| 813 | 814 | 3 | 0 | 6.0 | 2 | 31.2750 | 2 | 7 |

The table provides information such as Passenger ID, Passenger Class, Sex, Age, Parch (number of parents/children on board), Fare, Embarked port, and Family Size. These components shed light on the several variables that might have affected the tragedy's survival rates.

|  |  |  |
| --- | --- | --- |
| Name | Survived | dtype |
| 331 | 0 | int64 |
| 733 | 0 | int64 |
| 382 | 0 | int64 |
| 704 | 0 | int64 |
| 813 | 0 | int64 |

This table shows the passenger ID and the "Survived" column, which indicates whether the passenger survived (shown by a 1) or did not survive (shown by a 0). The "Survived" column contains only int64 type data.

### Create Test Data

A screenshot of a computer

Description automatically generated

This code handles missing data such as age, embarked columns data, by filling them with the median. Using LabelEncoder, the procedure encodes categorical variables like "Sex" and "Embarked". In addition, titles are taken from the "Name" column, and the number of siblings/spouses and parents/children aboard is determined by adding the values from the "SibSp" and "Parch" columns. Following that, columns such as "Name," "Ticket," "Cabin," and "Title" are eliminated so that only the features that are necessary for testing the machine learning model remain.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Passenger | ID | Pclass | Sex | Age | SibSp | Parch | Fare | Embarked | Family Size |
| 0 | 892 | 3 | 1 | 34.5 | 0 | 0 | 7.8292 | 1 | 1 |
| 1 | 893 | 3 | 0 | 47.0 | 1 | 0 | 7.0000 | 2 | 2 |
| 2 | 894 | 2 | 1 | 62.0 | 0 | 0 | 9.6875 | 1 | 1 |
| 3 | 895 | 3 | 1 | 27.0 | 0 | 0 | 8.6625 | 2 | 1 |
| 4 | 896 | 3 | 0 | 22.0 | 1 | 1 | 12.2875 | 2 | 3 |

With regard to passenger information, the table shows a sample of the test dataset that includes logistical information like class, fare, and embarkation port in addition to demographic information like age, gender, and family size. The integration of these data sets offers a thorough understanding of the passenger's situation, which facilitates the examination of variables that could impact the records.

## Create Machine Learning Model(s)

### Stochastic Gradient Descent (SGD)

A screenshot of a computer

Description automatically generated

This code handles multiple preprocessing tasks, including title extraction to generate a new family members feature, encoding categorical variables, and filling in missing data. Additionally, it divides the dataset into testing and training sets. The Stochastic Gradient Descent (SGD) classifier is then initialised, trained on training data, and its accuracy is assessed. Finally, predictions are made on the test set.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.5921787709497207 |

The accuracy metric value, which is around 59.2% in the table, shows how well the trained model performed.

### Random Forest

A screenshot of a computer

Description automatically generated

The initial steps of this code include encoding categorical variables, handling missing values, and loading data from a CSV file. In order to develop new features, it additionally extracts helpful data. The data is then divided into testing and training sets. Next, using the training set of data, a Random Forest classifier is initialised and trained. On the test set, predictions are made, and the accuracy of the model is assessed. Lastly, the model's accuracy score on the test set is shown.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.8212290502793296 |

With an accuracy of 82.12% on the dataset, the Random Forest classifier proved to be highly accurate in predicting the survival outcomes of passengers.

### Logistic Regression

A screenshot of a computer

Description automatically generated

After loading the dataset and addressing any missing values, this code encodes categorical variables and extracts more data, including titles from names. New features like family size are also produced by it. The data is first prepared, then divided into training and testing sets. A logistic regression classifier is then initialised, and the model is trained using the training set. Next, predictions are created using the testing data, and the correctness of the model is assessed. The accuracy score, which provides a gauge of the model's performance in predicting survival outcomes, is printed last.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.8044692737430168 |

With an accuracy of roughly 80.4 %, the logistic regression model was able to accurately predict the survival status of roughly 80% of the passengers in the testing dataset. This performance measure demonstrates how well the logistic regression algorithm predicts survival outcomes based on characteristics such as family size, age, sex, and class. The model's accuracy in estimating passenger survivability on the Titanic by using these characteristics is demonstrated by this.

### K Nearest Neighbor (KNN)

A screenshot of a computer

Description automatically generated

This code predicts survival outcomes in the dataset using the K Nearest Neighbours (KNN) algorithm. First, categorical variables are encoded and missing values are addressed. To improve the model's input data, new features are generated and additional information is retrieved from existing features. Next, the dataset is divided into testing and training sets. The KNN classifier is initialised, and then the model is trained using the training set. After that, it forecasts the testing set and assesses the accuracy of the model, giving a thorough indication of how well the KNN algorithm predicts survival based on the processed data.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.659217877094972 |

The accuracy statistic, which measures how well the K Nearest Neighbours (KNN) model can predict passenger survival when applied to the Titanic dataset, is displayed in the table. The KNN model accurately predicted the survival status for roughly 65.92% of the passengers in the test dataset, with an accuracy value of about 65.92%. Based on the properties in the dataset, this metric proves how well the KNN algorithm classified passengers into those who survived and those who did not.

### Naïve Bayes

A screenshot of a computer

Description automatically generated

This code processes the dataset the fills missing data, encodes categorical variables, and extracts featuers to enhance the dataset. Then the data is split into testing and training sets. Next, a Gaussian Naïve Bayes classifier is initialised, trains the data, and is used to predict test data by the code. The accuracy of the model's performance is measured and printed to the console to give a clear indication of how well the classifier predicts survival outcomes.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.7821229050279329 |

The Naïve Bayes classifier's accuracy value of roughly 78.21% indicates how well it predicts the survival outcomes of Titanic passengers. This shows that the model used important characteristics like age, gender, and ticket class to predict whether or not people would survive in roughly 78.21% of the events in the test dataset.

### Perception

A screenshot of a computer

Description automatically generated

This code predicts survival outcomes by training a classification model on passenger data using the Perception algorithm. Features including age, gender, embarkation port, and family size are used. The accuracy of the model is used to quantify its efficacy, indicating how well it can determine whether passengers survived the accident or not.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.6201117318435754 |

The classification model's accuracy measure, which employs the Perceptron method on passenger data from the Titanic. About 62% of the passengers in the test dataset had their survival outcomes correctly predicted by the model, which had an accuracy of 0.6201.

### Linear Support Vector Machine

A screenshot of a computer

Description automatically generated

In order to handle missing values in the dataset, this code replaces the 'Age' column with median values and the 'Embarked' column with mode values. It extracts titles from the 'Name' column, translates categorical variables like 'Sex' and 'Embarked' into numerical formats, and adds a new feature called 'FamilySize' by adding the 'SibSp' and 'Parch' columns. The data is split into training and testing sets following preprocessing. After that, a Support Vector Machine (SVM) classifier with a linear kernel is initialised by the code, trained on training data, and used to predict test data. By comparing the expected results with the actual target values, it calculates the accuracy score of the model and prints it.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.77653612849165 |

The accuracy score for a Linear Support Vector Machine (SVM) model that predicts survival outcomes on the Titanic dataset is shown in the table. With an accuracy score of roughly 77.65%, the model was able to accurately predict the survival status of roughly 77.65% of the test dataset's passengers. This measure shows how well the SVM model performs when it comes to correctly categorising passengers using the given features.

### Decision Tree

A screenshot of a computer

Description automatically generated

This code processes and analyses the dataset in multiple steps. It loads the data first, preprocessing it by creating new features, encoding category variables into numerical ones, and filling in missing values. The data is split into training and testing sets following preprocessing. Next, using the training data, a Decision Tree classifier is initialised and trained. The classifier is used to provide predictions on test data after it has been trained.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.7541899441340782 |

The Decision Tree classifier that was used on the dataset yielded an accuracy value of 0.7541, as shown in the table. This performance indicator demonstrates how accurately the classifier can predict survival outcomes. The classifier efficiently divides the feature space into discrete areas for prediction by utilising information such as gender, age, embarkation point, and family size. By using this technique, the Decision Tree is able to recognise and take advantage of survival trends among travellers, illustrating how useful it is for comprehending and evaluating the intricate dynamics of the dataset.

## Decide which is the Best Model

### Calculate Best Model

A screenshot of a computer

Description automatically generated

This code calculates the average accuracy and top performer by analysing performance characteristics of different machine learning models. First, a dictionary called accuracy\_scores is defined, with each model name serving as a key and the value being the accuracy score that corresponds to it. By taking the mean of the dictionary values, the algorithm determines the average accuracy across all models. In order to identify the best-performing model, it looks up the dictionary's greatest accuracy score. The code then provides a clear summary of the models' efficacy based on the given data by printing the average accuracy, the name of the top-performing model, and its individual accuracy score.

|  |  |
| --- | --- |
| Metric | Value |
| Average Accuracy | 0.7264688690873939 |
| Best Model | Random Forest |
| Accuracy of Best Model | 0.82122905027922965 |

The table presents the average accuracy of multiple machine learning models and identifies Random Forest as the top-performing model, with an accuracy of 0.8212 and the average of all models of 0.7265. Based on the accuracy measure supplied, the Random Forest model is the most effective model listed in the table, demonstrating that it outperforms other models in predicting outcomes.

### K-Fold Cross Validation

A screenshot of a computer

Description automatically generated

This code uses the preprocessed Titanic dataset to perform k-fold cross-validation using a Random Forest Classifier. The first step in preparing the dataset is to one-hot encode categorical variables and fill in any missing values. After that, to evaluate the model's performance more thoroughly and reduce overfitting to the training set, the algorithm employs k-fold cross-validation. To offer a statistical overview of the model's performance across various data subsets, it computes the mean accuracy and standard deviation from the cross-validation scores. Lastly, the classifier's mean accuracy and standard deviation are presented, providing information on how consistently and dependably its predictions perform.

|  |  |
| --- | --- |
| Metric | Value |
| Mean Accuracy | 0.8092963404682694 |
| Standard Deviation | 0.04883749307924223 |

The result above indicates that the model maintains a mean accuracy of approximately 80.93% with a standard deviation of 0.0488, indicating consistent performance across various subsets of the data. The comparatively low standard deviation highlights the classifier's dependability in this setting by showing little variation in performance across the folds.

### Feature Importance

A screenshot of a computer

Description automatically generated

This code evaluates the significance of several characteristics in predicting survival aboard the Titanic by using a Random Forest classifier. The dataset is first preprocessed, with the mode being used to replace missing values in the "Embarked" column and the median being used to fill in the "Age" column. One-hot encoding converts categorical variables like "Sex" and "Embarked" into numerical representations. This data is preprocessed and then used to train the Random Forest classifier. After determining each feature's significance, the model extracts the data and arranges it into a DataFrame. The most important characteristics in predicting survival outcomes are highlighted in this DataFrame by sorting it in descending order.

|  |  |  |
| --- | --- | --- |
| Value | Feature | Importance |
| 0 | Passenger Id | 0.187196 |
| 5 | Fare | 0.181608 |
| 2 | Age | 0.169467 |
| 7 | Sex\_Male | 0.145658 |
| 6 | Sex\_Female | 0.130505 |
| 1 | Pclass | 0.078234 |
| 3 | Sibsp | 0.041788 |
| 4 | Parch | 0.032521 |
| 10 | Embarked\_S | 0.014151 |
| 8 | Embarked\_C | 0.011290 |
| 9 | Embarked\_Q | 0.007581 |

The feature significance values from a Random Forest classifier, which was used to forecast survival on the RMS Titanic, are displayed in the table. Passenger Id, Fare, Age, and categorical variables such as Sex (Male and Female), Pclass, SibSp, Parch, and Embarked S, Embarked C, and Embarked Q are all assigned to a corresponding row in the database. Higher values indicate a greater influence on determining survival outcomes. The 'Importance' column measures the relative importance of each feature in the prediction process.

## Revise Dataset by dropping features that do not play a significant role

A screenshot of a computer

Description automatically generated

This code uses a Random Forest classifier to process the Titanic dataset and predict survivor outcomes. First, the data is loaded, and any missing values are filled in by adding the median and mode to the "Age" and "Embarked" columns, respectively. Then, using one-hot encoding, it turns categorical variables, such "Sex" and "Embarked," into numerical representations. Following data preparation, the algorithm extracts feature importances and trains a Random Forest classifier before putting the results into a DataFrame for improved visualisation. After the features are ranked in order of significance, a threshold is applied to determine which features are most important. After this threshold is reached, features are kept to produce a refined dataset that is used to train the model.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Passenger | ID | Fare | Age | Sex\_Male | Sex\_Female | Pclass | Survived |
| 0 | 1 | 7.2500 | 22.0 | True | False | 3 | 0 |
| 1 | 2 | 71.2833 | 38.0 | False | True | 1 | 1 |
| 2 | 3 | 7.9250 | 26.0 | False | True | 3 | 1 |
| 3 | 4 | 53.1000 | 35.0 | False | True | 1 | 1 |
| 4 | 5 | 8.0500 | 35.0 | True | False | 3 | 0 |

Part of the dataset is shown in the table, where each row corresponds to a single passenger. Passenger ID, Fare, Age, Sex (Male/Female), Passenger Class (Pclass), and Survival status (1 for survived, 0 for did not survive) are among the features that are shown.

### Training the Best Model Again

A screenshot of a computer

Description automatically generated

This code uses a Random Forest classifier to process the Titanic dataset in order to analyse survival. It starts by loading the dataset and takes care of preprocessing chores like encoding categorical variables and filling in missing values. The data is split into training and testing sets following preprocessing. After that, the code initialises and trains a Random Forest classifier using the training set. It uses the test data to generate predictions after training. The model's accuracy on the test set is then computed and printed, giving an indication of how well the classifier predicts survival outcomes.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.8435754189944135 |

The performance of a Random Forest classifier trained on the Titanic dataset is shown in the table, with an accuracy of roughly 84.36%. This graph shows the proportion of correctly predicted survival outcomes out of all the predictions made for passengers in the test set, demonstrating how well the classifier performed in predicting survival outcomes based on the given attributes.

## Hyperparameter Tuning

### Perform Hyperparameter Tuning

A screenshot of a computer program

Description automatically generated

With the Titanic dataset, this method uses Grid Search Cross Validation to do hyperparameter tuning for a Random Forest classifier. After loading the dataset, preprocessing is done to manage missing values and encode category variables. The data is divided into training and testing sets, and then several combinations of hyperparameters for the Random Forest are outlined in a parameter grid. After initialising the classifier, the optimal set of hyperparameters is found by running Grid Search CV on this parameter grid. The ideal hyperparameters are then shown, along with the accuracy score that goes with them. The tuned model is then put to use by using the best model to generate predictions on the test set. The accuracy of these predictions is then calculated and presented, demonstrating its efficacy.

|  |  |
| --- | --- |
| Hyperparameter | Value |
| max\_depth | None |
| Max\_feature | ‘sqrt’ |
| Min\_sample\_leaf | 2 |
| Min\_sample\_spilt | 10 |
| n\_estimators | 100 |
| Accuracy | 0.8156424581005587 |

The table demonstrates the best hyperparameters for a Random Forest classifier on the Titanic dataset that were found using Grid Search Cross Validation. After adjusting these hyperparameters to optimise the classifier's performance, the test data accuracy score was 0.8156.

### Test New Parameters

A screenshot of a computer

Description automatically generated

This code uses a Random Forest classifier to process the dataset in order to predict survival. The first step involves loading the dataset and performing preprocessing operations, like encoding category variables and filling in missing values. Next, the data is split into testing and training sets. After that, a Random Forest classifier is trained using the training set of data and initialised with particular hyperparameters. The classifier uses the test set to generate predictions after training. The accuracy of the model is calculated to assess its performance, and the results are printed to show how well the additional parameters improved the survival outcome prediction.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.815642458100587 |

This table shows the accuracy of the Random Forest.

## Evaluation

### Generate Confusion Matrix

A screenshot of a computer

Description automatically generated  
The code loads the dataset first, then preprocesses it by encoding categorical variables and handling missing values. It then separates the data into testing and training sets. After that, a Random Forest classifier with predetermined hyperparameters is built up, the model is trained with training data, and predictions are produced on the test set. The accuracy of the model is then assessed, and the confusion matrix—which includes the numbers of true negatives, false positives, false negatives, and true positives—is computed.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 0.815642458100587 |

The model's performance as trained on the dataset is indicated by this accuracy value. This indicates that 81.56% of the test set's passengers had their survival outcomes predicted by the model with accuracy.

|  |  |  |
| --- | --- | --- |
|  | Predicted Not Survived | Predicted to Survived |
| Actual Not Survived | 92 | 13 |
| Actual Survived | 20 | 54 |

The model's predictions for survival outcomes on the Titanic dataset are compiled in the given matrix. Out of the passengers that did not survive, it shows that 13 were falsely expected to have survived (false positives) and 92 were correctly predicted as such (true negatives). Moreover, of the passengers who did make it out alive, 20 were incorrectly projected to have died (false negatives), whereas 54 were correctly predicted to have lived (true positives).

### Calculate Precision and Recall

A screenshot of a computer

Description automatically generated  
The given code loads the Titanic dataset, preprocesses it by encoding categorical variables and managing missing values, divides it into training and testing sets, trains a Random Forest classifier with specified hyperparameters, assesses its effectiveness using precision and recall metrics, and displays the outcomes.

|  |  |
| --- | --- |
| Metric | Value |
| Precision | 0.8059701492537313 |
| Recall | 0.7297297297297297 |

For the prediction model trained on RMS Titanic disaster data, the precision and recall metrics are shown in the table. The percentage of accurately anticipated survival cases among all predicted survival instances is what precision measures, and it is used to evaluate the accuracy of positive forecasts. Approximately 0.8060 was the precision number, meaning that 80.60% of the predicted survival cases were correct. Recall indicates the percentage of true survival cases that were accurately predicted as such and quantifies the model's capacity to correctly identify all pertinent instances. The model detected almost 73.00% of the real survival cases, according to a recall value of about 0.7300.

### Calculate F-Score

A screenshot of a computer

Description automatically generated

The provided code loads passenger data, handles missing values, encodes categorical variables, divides the data into training and testing sets, trains a Random Forest classifier with given hyperparameters, creates predictions for the test set, and computes the F-score, which is a performance metric for the model.

|  |  |
| --- | --- |
| Metric | Value |
| F-score | 0.7746478873239436 |

The F-score value, which indicates how well the model predicts passenger survival outcomes, is shown in the table. This score provides a fair assessment of the model's performance since it is the harmonic mean of precision and recall. Based on the passenger data from the Titanic, a higher F-score value denotes greater overall performance in properly predicting survival outcomes.

### Generate Precision Recall Curve

A screenshot of a computer

Description automatically generated

The code that is provided loads passenger data from the Titanic and preprocesses it by encoding categorical variables and addressing missing values. The data is then divided into testing and training sets. Using the given hyperparameters, a Random Forest classifier is trained on the training set. For the test set, probabilities of the affirmative class are predicted; a Precision-Recall curve is then produced using these probabilities. This curve provides information about the performance of the classifier by illustrating the trade-off between precision and recall for different categorization levels.

### Generate ROC AUC Curve

A screenshot of a computer

Description automatically generated  
In addition to handling missing values and loading passenger data, the code also one-hot encodes categorical variables such as "Sex" and "Embarked." On the basis of the preprocessed data, a Random Forest classifier is then trained. The test set's survival probabilities are predicted, and the classifier's performance in predicting survival outcomes is assessed by computing the AUC score and ROC curve. Lastly, it displays the trade-off between the true positive rate and false positive rate by plotting the ROC curve and giving us a AUC score of 0.84, meaning the perfomance of the classifier is good.

# Conclusion

Overall, based on different parameters such as age, gender, and embarkation port, the machine learning models developed in this investigation, especially the Random Forest Classifier, showed promising performance in predicting the survival outcomes of Titanic passengers. The models yielded an approximate 81.56% accuracy, an approximate 80.60% precision, an approximate 72.97% recall, and an F-score of 76.60%. Furthermore, an AUC score of 0.84 was obtained from the ROC curve analysis, demonstrating strong discriminatory power.

It's important to recognise that these models may not generalise flawlessly to new, unseen data because they were trained and assessed exclusively on previous data. Furthermore, when using prediction models to analyse past disasters like the Titanic accident, ethical considerations must be properly considered. Although the models provide useful information on survival probability, they are not able to conclusively predict an individual's fate. As a result, judgement calls should be made carefully, and moral issues like honouring the victims' memories should always come first.

In conclusion, even though the models offer insightful information about the variables affecting survival aboard the Titanic, they are unable to conclusively identify those who ought to have died or survived. They are useful instruments for contemplation and analysis of past occurrences, but their use must be conscientious and considerate of the human statements concealed within the data.

Guidance

1. **Tools to use**: You should start with a new Template Case Study Jupyter notebook, to contain your machine learning model and notes as you reflect on what the code has generated at each stage. This notebook should be worked with in Google Colab.
2. **Datasets to use**: The following three CSV files, which can be downloaded from the [Titanic](https://www.kaggle.com/c/titanic) (<https://www.kaggle.com/c/titanic>) page on the Kaggle site, are the three datasets that are required for this case study:
   1. **train.csv**,
   2. **test.csv**,
   3. **gender\_submission.csv.**
3. **Understand the data**: Read the Overview, and make sure you understand the difference between the training and testing data, and also the **pclass**, **sibsp** and **parch** data fields. Also note the **Embarked** column is the Port of Embarkation, and contains a C, Q, or S, depending on whether the passenger joined the ship in Cherbourg, Queenstown, or Southampton.
   1. We recommend viewing the data files with *Excel*. In *Excel*, it will be convenient to select "Filter", from the "Sort and Filter" tab item, and "Freeze Top Row" from the "View" menu.
   2. To ensure that you understand the **sibsp** and **parch** data fields, verify their values for the following people, or otherwise explain any discrepancies. (If you're having trouble squaring these up, remember that the training file is not a complete listing of the passengers. Together, the training and testing files make up the full set.)
      1. The Futrelle family (click the pull-down menu in Column C and select "Contains" and type in "Futrelle")
      2. The Palsson family
      3. The Skoog family.
4. **Construct the Machine Learning Model**: We want to use machine learning to figure out who survived and who did not, based on the other data. Use the template Jupyter notebook to help you construct a machine learning model to predict who survived and who did not.
5. **Reflection**: At each stage of the build of your machine learning model, you might want to reflect on:
   1. problems encountered and how you resolved them
   2. applicability of these tools and techniques to problems you foresee encountering in the future.
   3. benefits of using parallelism to approach larger scale problems.
6. **Submission:** Download and submit your Jupyter notebook via the submission link on CMU505 – Machine Learning Moodle page.