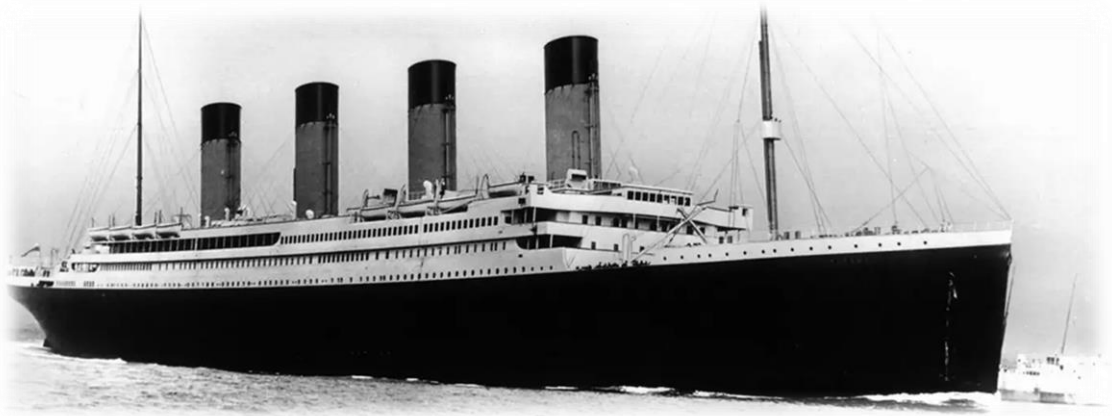


# DASC591 - Non-Thesis Master Term Project

Data Science - Koç University - 2022 Fall

## Titanic Dataset



**Ersin Sönmez**

ersinsonmez21@ku.edu.tr - (ID: 0079025)

### Abstract

In order to realize some of the data science applications on the selected project topic (Titanic Dataset), examinations were made. First of all, what the data looks like and what it consists of is explained and the details of the elements that make up the data are explained. Preparations were made with pre-processing on the visualized data and the influencing features were highlighted. The processed model, which was prepared for the fit and predict processes, was analyzed with various classifiers, and the results were compared within itself. The data of survivors were tried to be estimated without being included in the assessment. When models are run with existing data, prediction and actual data converge. It is ensured that people who live or die are predicted over data. This Project aims to gain practical experience over the "Titanic" dataset, which is frequently used in data science. Within the scope of the course, it is an end-to-end application in which we can see all of the various analysis applications taught during the master's degree. The script and data on which the study was carried out will be attached to this report, and each title that composes the content of this document can be viewed under "Table of Contents" - quoted on the next page.

## Table of Contents

Abstract.....	1
1) Introduction .....	2
2) Related Work .....	2
3) Approach.....	3
Step 1: Understand the Problem .....	3
Step 2: Import Libraries and Datasets.....	3
Step 3: Perform Data Visualization .....	4
Step 4: Drop the Target Column Before Splitting .....	5
Step 5: Splitting, Perform Pre-Processing .....	5
Step 6: Train Logistic Regression Classifier Model.....	5
Step 7: Assess Trained Model Performance for Logistic Regression .....	7
Step 8: Train Naive Bayes Classifier Model .....	7
Step 9: Assess Trained Model Performance for Naive Bayes .....	7
Step 10: Train Decision Tree Classifier Model .....	7
Step 11: Assess Trained Model Performance for Decision Tree .....	7
Step 12: Hyperparameter Tuning for Decision Tree ( <i>GridSearchCV</i> ).....	7
Step 13: Train Decision Tree Classifier Model with Optimized Hyperparameter .....	7
Step 14: Optimized Trained Model Performance for Decision Tree.....	7
Step 15: Comparison of Classifier Models and Results.....	7
4) Implementation Details .....	9
5) Evaluation Details, Results and Analysis/Discussion.....	9
6) Conclusion.....	9
7) References .....	9

## 1) Introduction

The sinking of the "Titanic" is one of the most tragic tragedies in history. The tragedy took place on April 15th, 1912. The Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers. The numbers of survivors were low due to lack of lifeboats for all passengers. Some passengers were more likely to survive than others, such as women, children, and upper-class. This project analyses what sorts of people were likely to survive this tragedy. To achieve this aim, comparisons will be made between the classifier models.

## 2) Related Work

A competition was held on the Kaggle platform on behalf of Titanic data, and many codes and learning models were shared in this context. The models seem to be very similar to each other, but the features, classifiers and hyperparameters of the developed models vary. In this project, "Survived" feature will be taken as the target function and classifications will be made accordingly.

(Competition Reference: <https://www.kaggle.com/c/titanic>)

### 3) Approach

Approaches carried out in parallel with the “Jupyter Notebook” script will be summarized in 15 steps in the following stages.

(Reference: DASC591\_NonThesisMasterTermProject\_ErsinSONMEZ\_Notebook.ipynb)

#### Step 1: Understand the Problem

In addition to the information given in the "Introduction" section, the data includes the following features, the target class is the "Survived" feature. The details are as follows:

- **Pclass** : Ticket class (1 = 1st Class, 2 = 2nd Class, 3 = 3rd Class)
- **Sex** : Sex (Male, Female)
- **Age** : Age in years
- **SibSp** : Number of siblings / spouses aboard the Titanic
- **Parch** : Number of parents / children aboard the Titanic
- **Ticket** : Ticket number
- **Fare** : Passenger fare
- **Cabin** : Cabin number
- **Embarked** : Port of Embarkation (C = Cherbourg, Q = Queenstown, S = Southampton)
- **Target Class** : Survived (0 = No, 1 = Yes)

#### Step 2: Import Libraries and Datasets

The basic libraries "Pandas, Numpy, Matplotlib and Seaborn" were used for data manipulations and visualizations, and the "scikit-learn" library and sub-models were used for artificial learning and classification. (It may be necessary to install with "pip install...") The dataset is as follows:

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cummings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows x 12 columns

(With "Pandas Profiling", more detailed data analysis can be accessed, it has been commented in the code, details will not be entered in order not to overshadow the actual work. In the following sections, data will be continued to be analyzed and edited with coding Pandas functions.)

### Step 3: Perform Data Visualization

The basic statistics of passengers and the meanings extracted from the data are as follows:

Total = 891

---

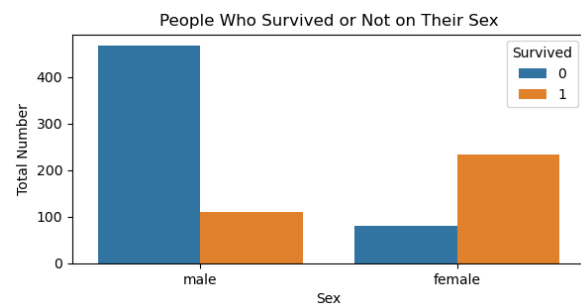
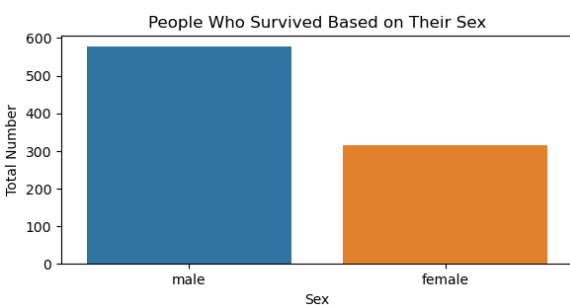
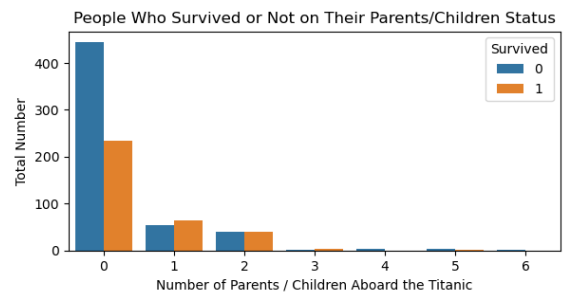
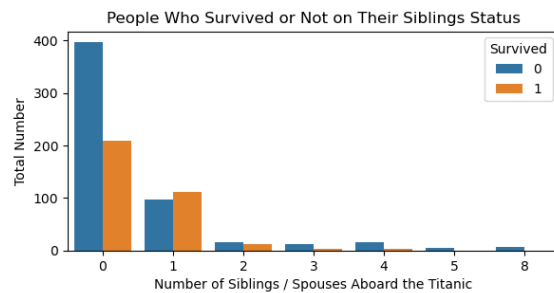
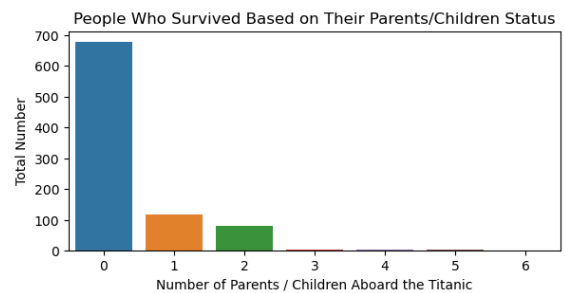
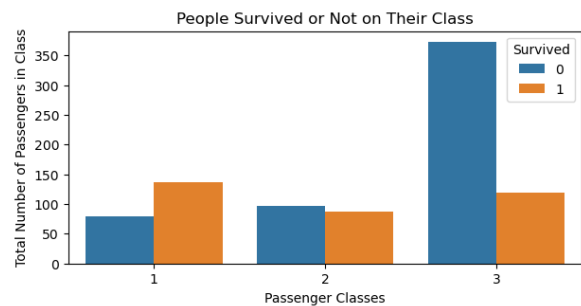
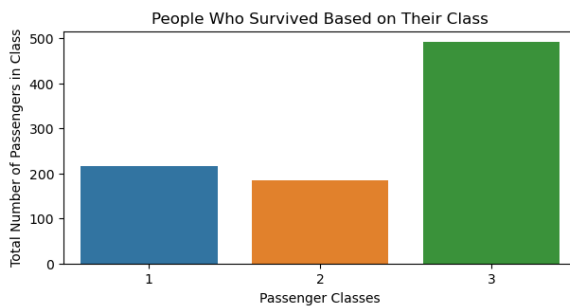
Number of passengers who survived = 342  
Percentage Survived = % 38.38

---

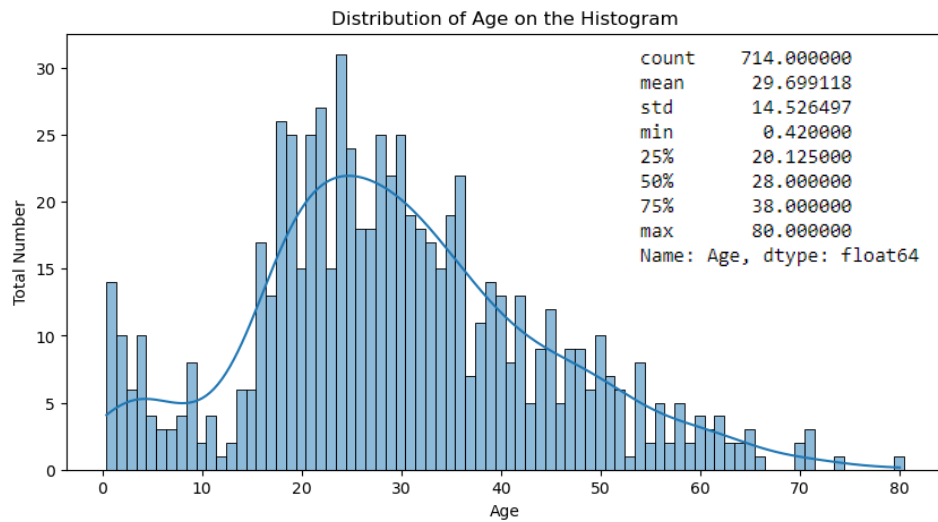
Number of passengers who did not survive = 549  
Percentage who did not survive = % 61.62

Some inferences supported by visualizations:

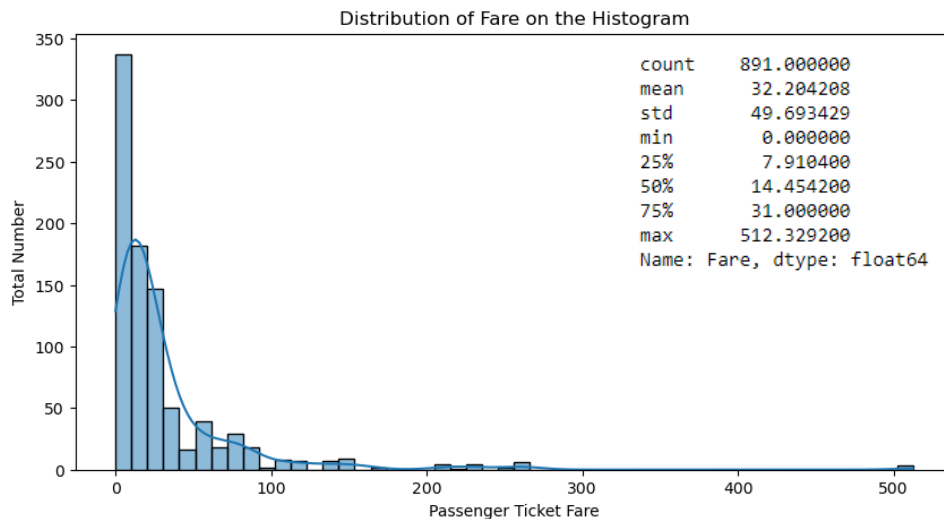
- If you are a first class, you have a higher chance of survival!
- Those who have one sibling seem to have a higher chance of survival than those without a sibling!
- If you have 1, 2, or 3 family members, you have a higher chance of survival compared to being alone!
- If you are a female, you have a higher chance of survival compared to other people!



- The average age is around 30 (approx. uniformly), the majority of the passengers on board are young.



- Many passengers showed interest in affordable tickets, and very few (>500 \$) bought expensive tickets.  
(probably 1st class passengers)

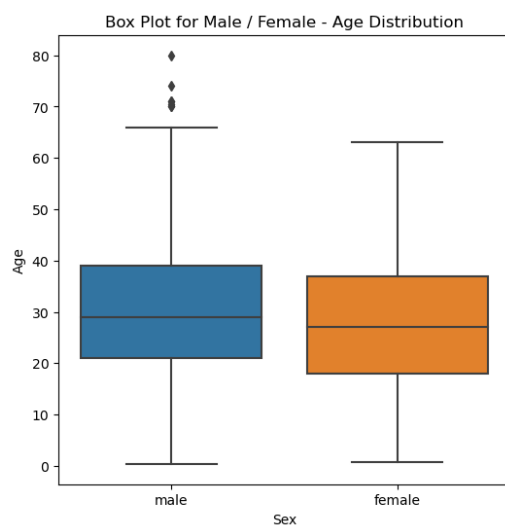
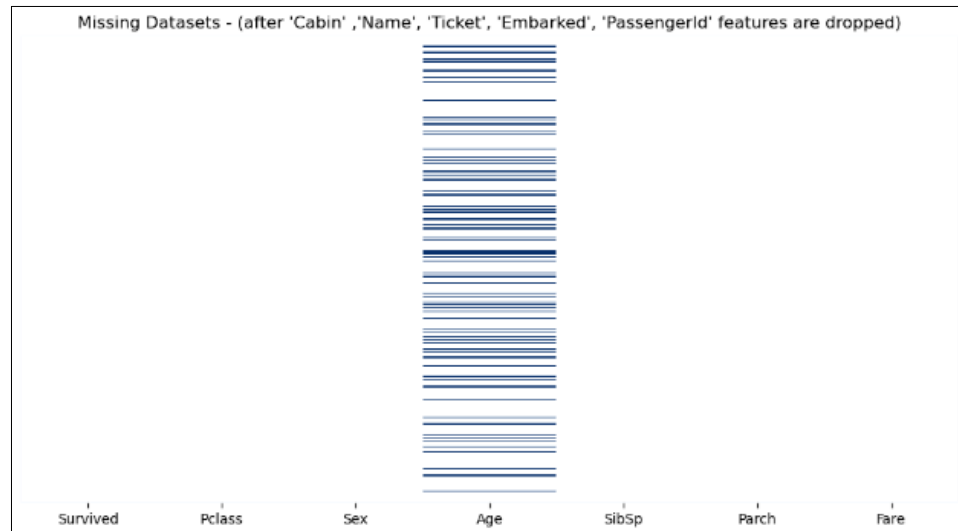
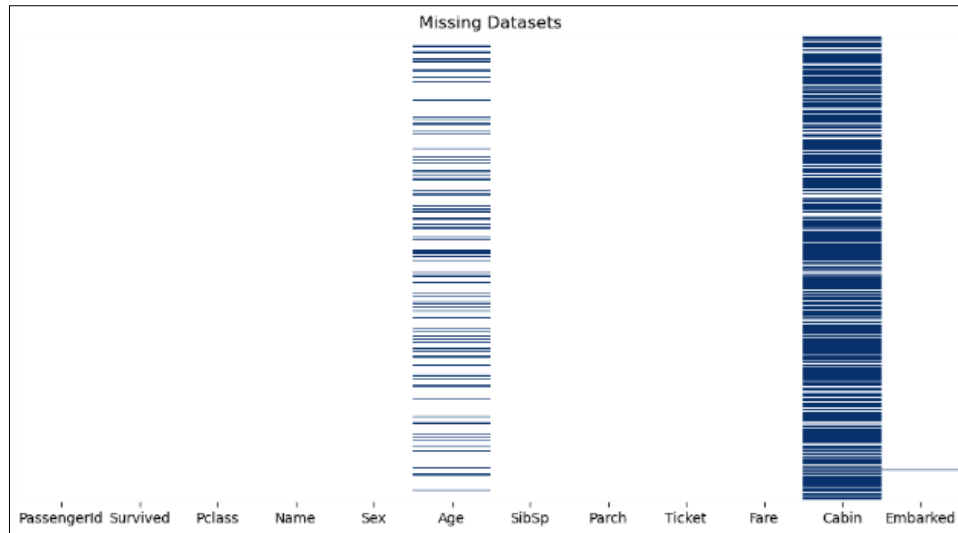


#### Step 4: Drop the Target Column Before Splitting

Before the train and test splitting, the target column in the data is dropped. It will be tried to be predicted with the models to be used. (X, y)

#### Step 5: Splitting, Perform Pre-Processing

First of all, it is necessary to split the data and then apply pre-processing to the train data. (It is a common mistake to apply the same pre-processing to the entire dataset, I had this mistake at first, too.) When the missing data were examined, a situation like the one below was encountered. The columns "Cabin, Name, Ticket, Bindi, PassangerId" have been removed from the dataframe. (It is assumed that the effects of these data on survival are low) Value assignments were made in the missing data of the column related to "Age". While performing this value assignment, manipulations were performed on the train data set. Instead of missing data with the written function, the mean age was accepted as 28 for women and 31 for men. (The mean age values for different split ratios converge to similar numbers.) The train data are plotted on the "Box Chart", the mean values are calculated and the numerical value is rounded up. After the operations, the data set is as follows:



	Pclass	Age	SibSp	Parch	Fare	male
7	3	2.0	3	1	21.0750	1
765	1	51.0	1	0	77.9583	0
339	1	45.0	0	0	35.5000	1
374	3	3.0	3	1	21.0750	0
183	2	1.0	2	1	39.0000	1
...	...	...	...	...	...	...
369	1	24.0	0	0	69.3000	0
320	3	22.0	0	0	7.2500	1
527	1	31.0	0	0	221.7792	1
125	3	12.0	1	0	11.2417	1
265	2	36.0	0	0	10.5000	1

623 rows × 6 columns

The same features were dropped on the test dataset and the dataframe was dimensionally matched, similar operations were repeated in the script.

### Step 6: Train Logistic Regression Classifier Model

The data has been split into test and train so that 30 percent will be divided into tests. In the first case, it will work with the "Logistic Regression" classifier. (X\_train, X\_test, y\_train, y\_test)

### Step 7: Assess Trained Model Performance for Logistic Regression

This is the part where the model is predicted, the results are transferred onto a confusion matrix. The main outputs are classification report and accuracy. Even if the classifier changes in the following steps, the process will remain the same.

### Step 8: Train Naive Bayes Classifier Model

The process is the same as in "Step 6", except that the data is trained according to Naive Bayes.

### Step 9: Assess Trained Model Performance for Naive Bayes

The process is the same as in "Step 7", the outputs will be compared in the following sections.

### Step 10: Train Decision Tree Classifier Model

It is trained according to "Decision Trees" without changing the basic parameters.

### Step 11: Assess Trained Model Performance for Decision Tree

Predictions were created without changing the basic parameters, the outputs will be compared.

### Step 12: Hyperparameter Tuning for Decision Tree (*GridSearchCV*)

Since decision trees have many hyperparameters, optimizing them will increase the accuracy of the model and handle overfitting/underfitting problems. For this, the functions of the "GridSearchCV" module are used. It tries to find the optimum of the selected parameters within itself. 5-fold-cross validation is given as the input for the optimization. The best parameters obtained were used as the input of the next classifier. The main purpose is to tune hyperparameters for best results.

### Step 13: Train Decision Tree Classifier Model with Optimized Hyperparameter

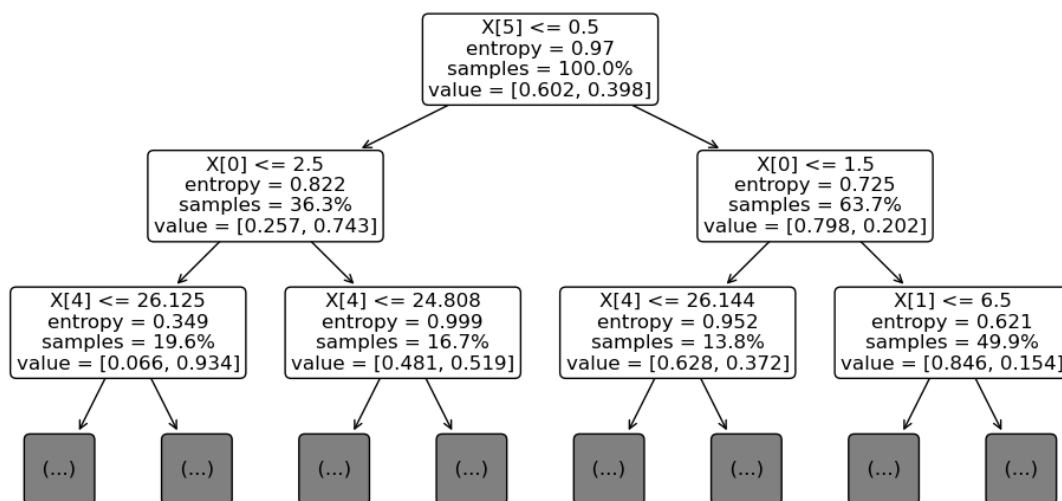
The model was fitted according to the best parameters obtained in the previous step. (Optimized)

### Step 14: Optimized Trained Model Performance for Decision Tree

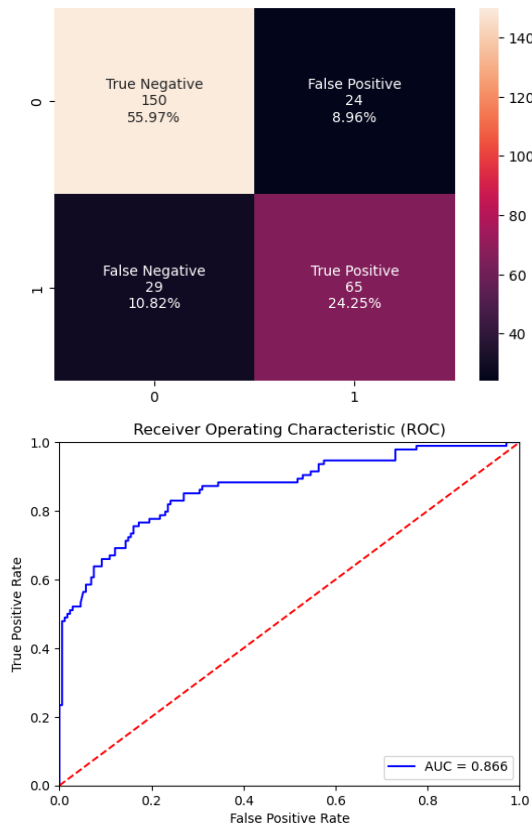
"Optimized Decision Tree" is saved to evaluate predictions and metrics, outputs will be compared.

### Step 15: Comparison of Classifier Models and Results

A summary visualization of the "Optimized Decision Tree" is as follows, because it branches too much, it's down to the 2nd maximum depth. The decision tree presents the idea of the classifier.

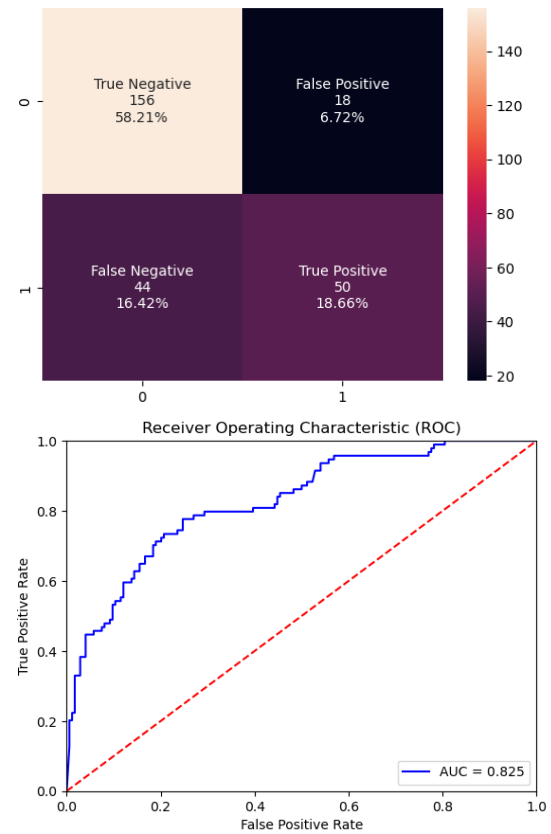


**[Logistic Regression Classifier Model]**

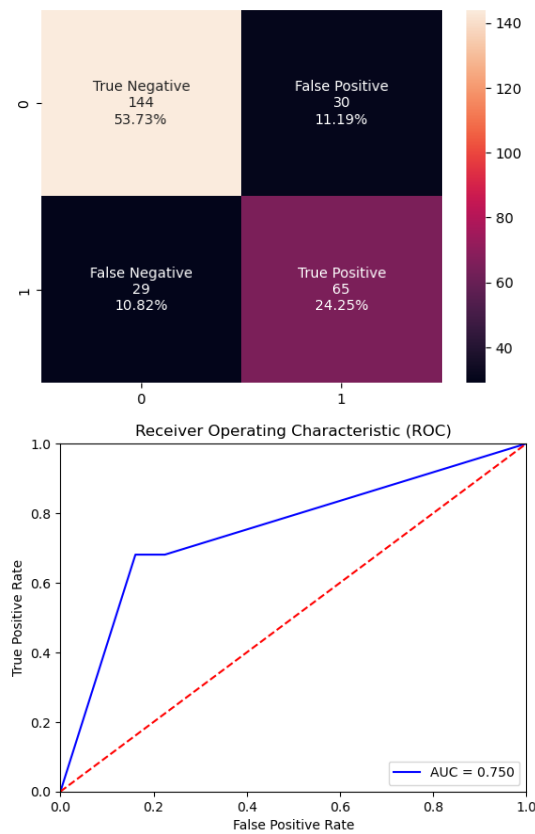


vs.

**[Naive Bayes Classifier Model]**

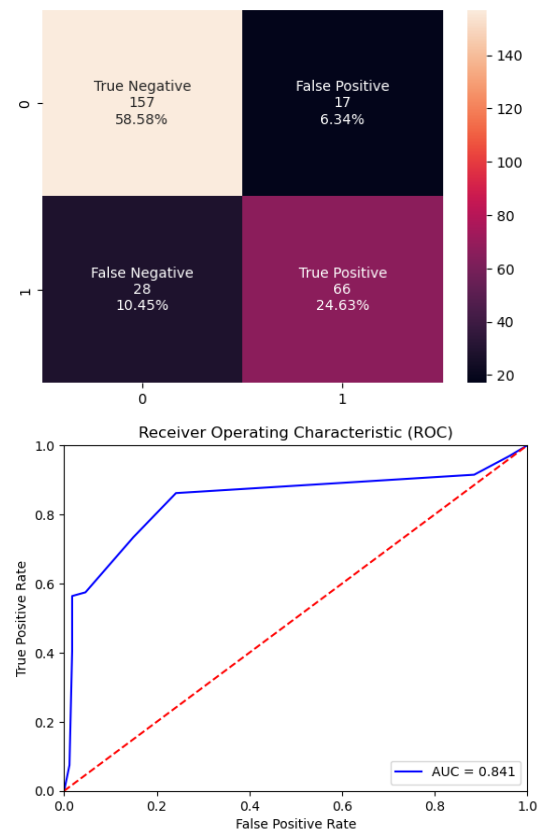


**[Decision Tree Classifier Model]**



vs.

**[Decision Tree Classifier Model (with Optimized)]**





#### 4) Implementation Details

When it comes to data science and learning models, the first thing that comes to mind is fast applicability and access to resources. "Jupyter Notebook" is also a pretty useful IDE here. It has been preferred because it is web-based, can provide interactive outputs and is lightweight. Many libraries come preloaded with "Anaconda" in the environment settings. Most of the online training resources and repositories provide fast implementation for this notebook. For this reason, it has been preferred.

(Note: For the notebook to work properly, the "titanic.csv" dataset must be in the same directory as the script.)

#### 5) Evaluation Details, Results and Analysis/Discussion

While comparing the scores of the models, the outputs of the classifiers were taken into account. In other words, confusion matrices are an evaluation criterion for our project. The Accuracy metric is also a score shared by each, calculated as follows: (True Positive, True Negative, False Positive, False Negative)

When Accuracy is calculated for all models, a summary table is obtained from the script as follows:

$$\text{accuracy (ACC)} \\ \text{ACC} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

```
- Logistic Regression --> Accuracy Score: 0.802
- Naive Bayes --> Accuracy Score: 0.769
- Decision Tree (unoptimized) --> Accuracy Score: 0.784
- Decision Tree (optimized) --> Accuracy Score: 0.81
```

When AUC Score is calculated for all models, a summary table is obtained from the script as follows:

$$\text{TPR} = \text{Sensitivity} = \frac{\text{TruePositives}}{(\text{TruePositives} + \text{FalseNegatives})}$$

and

$$\text{FPR} = 1 - \text{Specificity} = \frac{\text{TrueNegatives}}{(\text{TrueNegatives} + \text{FalsePositives})}$$

```
- Logistic Regression --> AUC Score: 0.866
- Naive Bayes --> AUC Score: 0.825
- Decision Tree (unoptimized) --> AUC Score: 0.75
- Decision Tree (optimized) --> AUC Score: 0.841
```

#### Some important outputs and notes:

- Since the data set is imbalanced, it would not be right to only look at accuracy. In addition to confusion matrix, f-1 score examinations, ROC curve and AUC score examinations are also important, the accuracy of the models should not be evaluated over a single parameter. (I was also mistaken in the beginning, in unbalanced datasets, the "Accuracy Paradox" is common)
- Splitting the Test/Train sets at different rates (30% - 20%) causes minor differences.
- When the data set is considered numerically, it generally showed a uniform distribution. In parallel, the models work linearly in the spaces they are trained in.
- In this project, all of the classifiers examined give fast results, there are no bottlenecks that slow down the model. The entire script can deliver output in seconds, Resource consumption is low.

#### 6) Conclusion

Only the accuracy score should not be considered as a criterion. Pre-processing operations can be differentiated, missing data in the model (except for age) may need to be processed with different methods. Thus, thanks to more features, better models can be developed, categorical data can be added to models, synthetic data can be created and deep learning can be applied, and the number of parameter optimizations can be increased.

#### 7) References

- Kaggle, Coursera, Lecture Notes and Applications, Python Cheat Sheets etc.