Titanic Revisited: Predicting Survival with Machine Learning
Elective project | Artificial Intelligent Systems

David Blazheski, E-mail: db1120@student.uni-lj.si

Faculty of Electrical Engineering

Abstract

This elective project in the course Artificial Intelligent Systems explores the use of machine learning models to predict passenger survival on the Titanic, using a dataset containing attributes such as age, gender, fare, and ticket class. Three models were implemented and compared: Logistic Regression, Random Forest, and K-Nearest Neighbors (KNN). The methodology included data preprocessing, feature scaling, and model evaluation using metrics such as accuracy, precision, and recall. The results showed that Random Forest achieved the best balance between precision and recall, while KNN excelled in overall accuracy when using scaled features.

Contents

1 Introduction

The sinking of the Titanic is one of the most wellknown shipwrecks in history. Many factors, such as a passenger's age, gender, ticket class, and fare played a role in determining whether they survived. This project uses machine learning to predict which passengers were likely to survive based on the dataset. The main goal was to test and compare different machine learning algorithms to find out which one works best for making accurate predictions in this scenario.

2 Theoretical background

2.1 Machine learning

Machine learning focuses on creating algorithms and models that allow computers to perform tasks by learning patterns from data, rather than working by explicit programming. The process begins with collecting and providing data, the more data available, the better the model's performance. Once the data is prepared, a suitable machine learning model is selected and trained to make predictions. During this process, developers tune the model and its parameters with specific metrics and goals in mind.

Among the many algorithms available, three were selected for this project: logistic regression, random forest, and K-Nearest Neighbors (KNN). Each algorithm has unique characteristics that make it suited for specific types of problems.

2.2 Logistic regression

Logistic Regression is an algorithm used for binary classification tasks, and it predicts probabilities that an instance belongs to a particular class. The logistic function maps the input values to a range between 0 and 1. The model calculates the log-odds of the dependent variable as a linear combination of the independent variables and their respective coefficients. These coefficients are learned during the training process using optimization techniques such as Gradient Descent [3].

2.3 Random forest

Random Forest is a powerful machine learning algorithm that ensembles decision trees. It is used for both classification and regression tasks. The primary idea behind Random Forest is to create multiple decision trees during training and predict their predictions to produce more accurate and robust results. In Random Forest, each tree is built using a random subset of the training data and a random selection of features.When making predictions, the algorithm combines the output of all individual trees, typically using a majority vote for classification tasks or an average for regression tasks [2].

2.4 KNN

K-Nearest Neighbors (KNN) is a simple, non-parametric machine learning algorithm used for both classification and regression tasks. In classification, KNN assigns a class to a data point based on the majority class of its k-nearest neighbors in the feature space. The core idea of KNN is to find the closest data points to the one being classified. The proximity is measured using distance metrics such as Euclidean distance, Manhattan distance, or Minkowski distance, with Euclidean distance being the most common. The parameter k, which represents the number of neighbors to consider, is crucial in determining the algorithm's performance. A smaller k may result in a model that is sensitive to noise, while a larger k can lead to smoother decision boundaries but may blur class distinctions [1].

2.5 Aim of the project

In this project, we used these algorithms to predict Titanic survival. The comparison focuses on their ability to handle the given dataset, which includes features like age, gender, ticket class, and embarkation point. By analyzing their performance, we aim to understand the strengths and weaknesses of each method in the context of binary classification problems. This understanding can provide valuable insights into selecting appropriate algorithms for similar predictive tasks in the future.

3 Methodology of the project

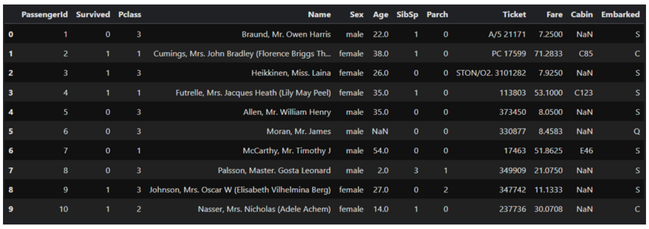
3.1 General

The course AIS provided foundational knowledge of machine learning terminology and techniques. Practical examples were sourced from online tutorials and publicly available notebooks, which supported the implementation process.
The project notebook was designed with a interactive structure, allowing continuous testing. Each block of code was independently tested and verified before moving on to the implementation of the next ML model.
For data exploration, visualizations such correlation matrix were calculated to identify relationships and patterns between features. During the analysis phase, confusion matrices, classification reports, and recall scores were primarily used to evaluate and compare the performance of the models.

3.2 Dataset overview

The dataset used for this analysis is sourced from Kaggle and consists of 891 passengers with labeled survival outcomes. This dataset serves as a training set for developing machine learning models to predict the survival of passengers aboard the Titanic. Each row contains information about various features such as age, gender, fare, ticket class, and embarkation point. The target variable is the survival status of the passenger, with values indicating whether the passenger survived or not. The key features in the dataset include:

- **Age**: The age of the passenger (some entries have missing values).
- **Gender**: The gender of the passenger, which is a categorical variable (Male or Female).
- **Fare**: The fare paid by the passenger for the ticket, a continuous numerical feature.
- **Ticket Class**: The class of the ticket purchased (1, 2, or 3), which is a categorical variable.
- **Embarkation Point**: The port where the passenger boarded the Titanic (C = Cherbourg, Q = Queenstown, S = Southampton).



	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	Cumings, 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
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

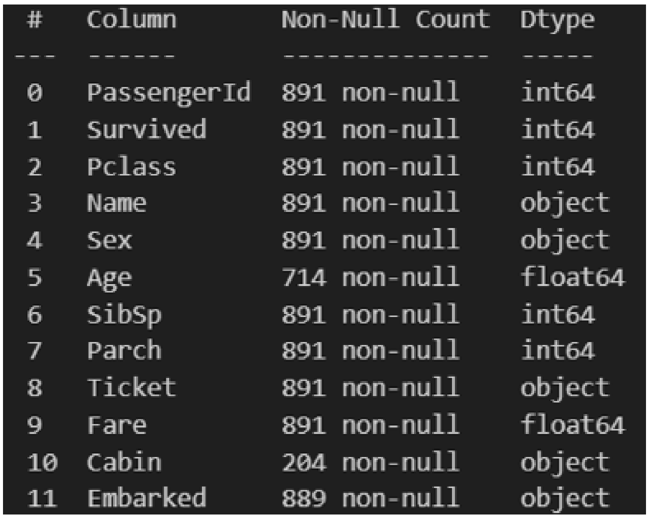
Figure 1: Training dataset.

3.3 Data Preprocessing

Data preprocessing is a crucial step in ensuring that the dataset is clean, consistent, and ready for training. The following steps were taken to preprocess the Titanic dataset:

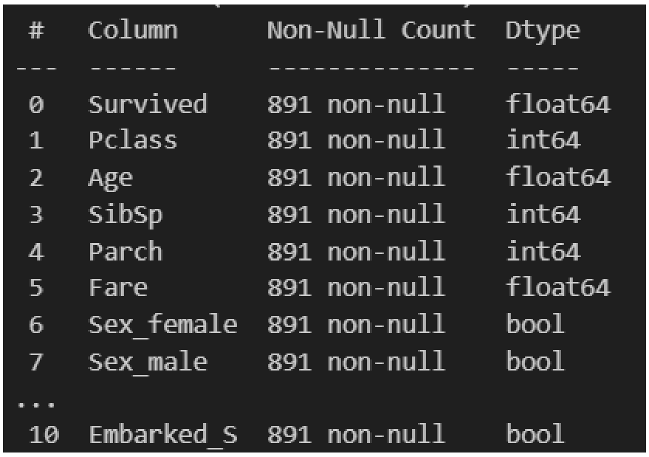
- **Handling Missing Values**: Some features, such as `Age`, contain missing values. These missing values were imputed using the median age of the respective class.
- **Encoding Categorical Variables**: Categorical features like `Gender` and `Embarkation Point` were encoded into dummy variables.. For `Gender`, a binary encoding was used, where 'Male' is represented as 0 and 'Female' as 1. Likewise, for `Embarkation Point`, three dummy variables were created to show the locations.
- **Feature Scaling**: Continuous variables, such as `Fare` and `Age`, were scaled to a standard range to improve model performance

and convergence, which ensures that these features have a mean of 0 and a standard deviation of 1. This scaling helps prevent any feature from dominating the model training process due to its larger range.



```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
```

Figure 2: Dataset before preprocessing.



```
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Survived    891 non-null    float64
 1   Pclass      891 non-null    int64
 2   Age         891 non-null    float64
 3   SibSp       891 non-null    int64
 4   Parch       891 non-null    int64
 5   Fare        891 non-null    float64
 6   Sex_female  891 non-null    bool
 7   Sex_male    891 non-null    bool
...
 10  Embarked_S  891 non-null    bool
```

Figure 3: Dataset after preprocessing.

With data preprocessing we went from figure 2 to figure 3.
By addressing missing data, encoding categorical variables, and scaling continuous features, the dataset is now in an optimal state for training.

3.4 Data distribution

For better understanding of data, we used correlation matrix, which helps in representing the relationship between two features. It gives the measure of the strength of similarity between two variables. The value can be between -1 to +1. 1 means that they are highly correlated, and while the one increases, the other does too. 0 means no correlation. Representation of correlation coefficients is presented via heatmap.

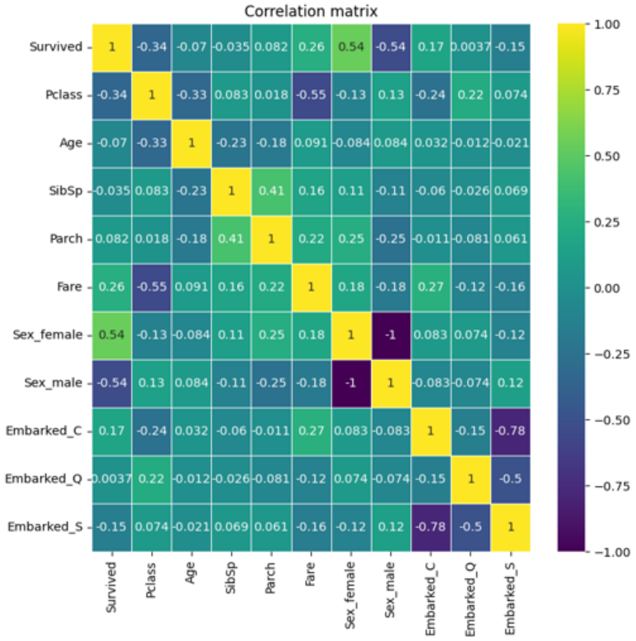


Figure 4: Heatmap correlation.

From Figure 4, we can observe the correlation coefficients of various features with the target variable. The most correlated features are as follows:

- **Sex (female)**: 0.54, indicating that being female had priority while saving
- **Sex (male)**: -0.54, suggesting that males were not likely to survive.
- **Fare**: 0.26, showing that passengers who paid higher fares were more likely to survive.
- **Pclass**: -0.34, indicating that passengers in

lower ticket classes (e.g., third class) were less likely to survive.

- **Embarked (C)**: 0.17, suggesting that passengers who embarked at Cherbourg had a slightly higher likelihood of survival.
- **Embarked (S)**: 0.15, indicating a weaker positive association with survival for passengers embarking at Southampton.

These correlations provide valuable insights into the relationships between features and survival outcomes. The most correlated features are illustrated in the following figures:

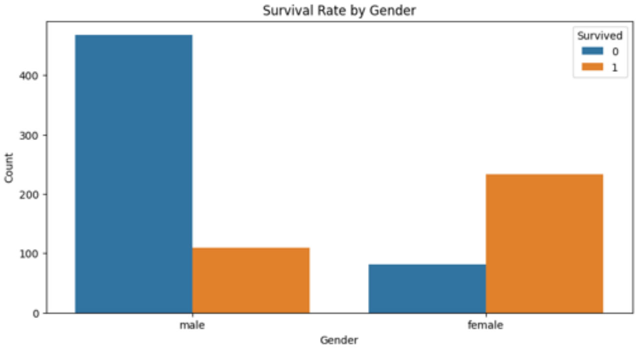


Figure 5: Survival rate by gender.

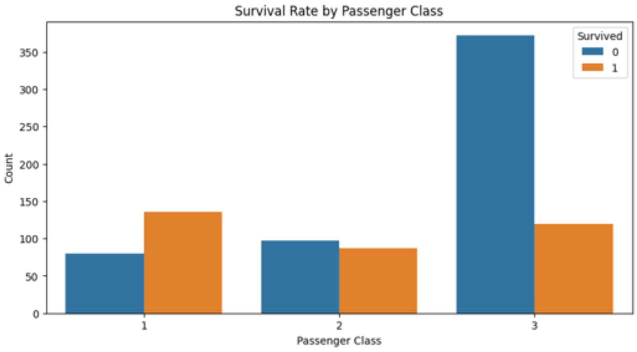


Figure 6: Survival rate by Passenger class.

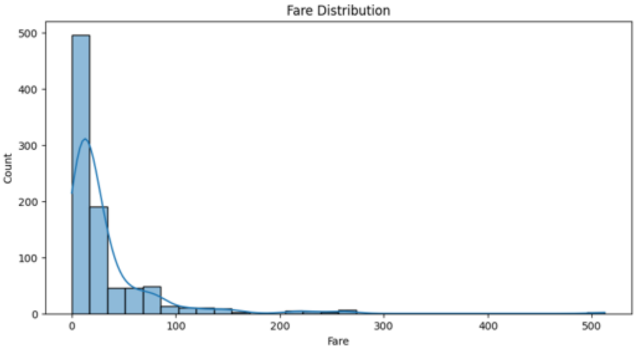


Figure 7: Fare distribution.

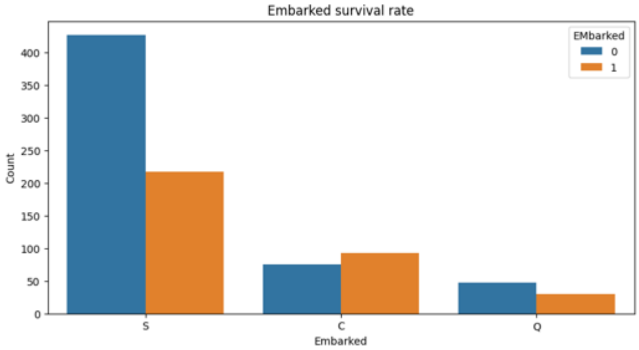


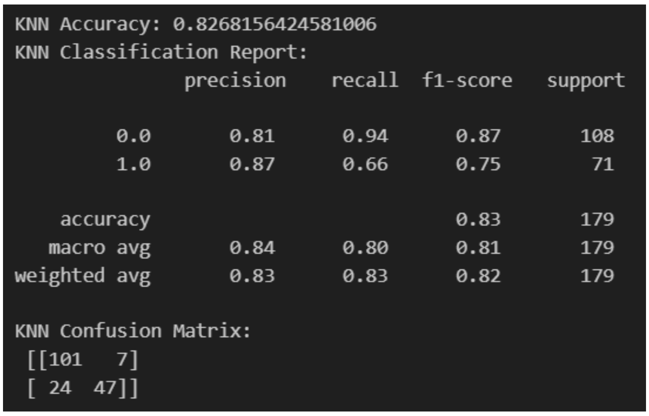
Figure 8: Embarked survival rate.

4 Results

4.1 Results of scaled features models

The performance of the three machine learning models while using scaled features are:

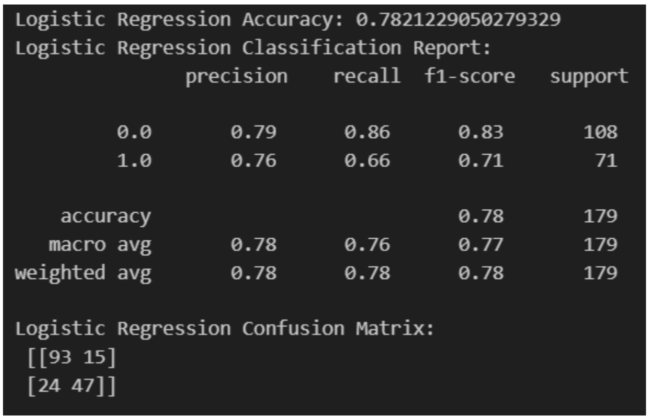
1. **K-Nearest Neighbors (KNN)** achieved the highest accuracy of 82.68%, with strong performance in predicting the majority class (class 0), evident from a recall of 94%. However, its performance in predicting the minority class (class 1) is lower, with a recall of 66%, indicating some misclassifications.



```
KNN Accuracy: 0.8268156424581006
KNN Classification Report:
              precision    recall  f1-score   support

         0.0       0.81      0.94      0.87       108
         1.0       0.87      0.66      0.75        71

    accuracy                           0.83       179
   macro avg       0.84      0.80      0.81       179
weighted avg       0.83      0.83      0.82       179

KNN Confusion Matrix:
 [[101   7]
 [ 24  47]]
```

Figure 9: KNN results.

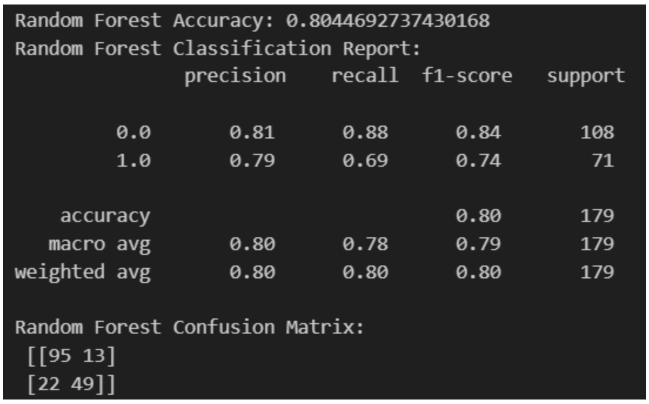
2. **Logistic Regression** scored an accuracy of 78.21%. It has balanced precision and recall values for both classes but slightly underperformed in comparison to KNN and Random Forest. Its recall for class 1 (66%) matches that of KNN, but its recall for class 0 is slightly lower.



```
Logistic Regression Accuracy: 0.7821229050279329
Logistic Regression Classification Report:
              precision    recall  f1-score   support

         0.0       0.79      0.86      0.83       108
         1.0       0.76      0.66      0.71        71

    accuracy                           0.78       179
   macro avg       0.78      0.76      0.77       179
weighted avg       0.78      0.78      0.78       179

Logistic Regression Confusion Matrix:
 [[93 15]
 [24 47]]
```

Figure 10: Logistic regression results.

3. **Random Forest** achieved an accuracy of 80.45%, providing a good trade-off between precision and recall across both classes. It performed slightly better than Logistic Regression but not as well as KNN in overall accuracy and recall for class 0.



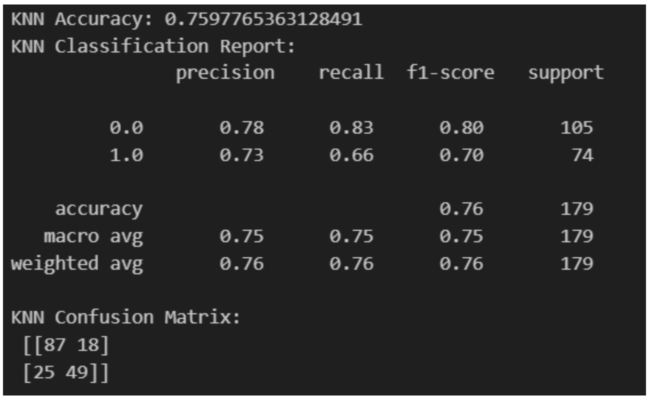
```
Random Forest Accuracy: 0.8044692737430168
Random Forest Classification Report:
              precision    recall  f1-score   support

         0.0       0.81      0.88      0.84       108
         1.0       0.79      0.69      0.74        71

    accuracy                           0.80       179
   macro avg       0.80      0.78      0.79       179
weighted avg       0.80      0.80      0.80       179

Random Forest Confusion Matrix:
 [[95 13]
 [22 49]]
```

Figure 11: Random forest classifier results.

4.2 Results with chosen features

The performance of the three models while using chosen features: Pclass, Fare, Sexfemale, Sexmale, EmbarkedS, EmbarkedC, EmbarkedQ are:

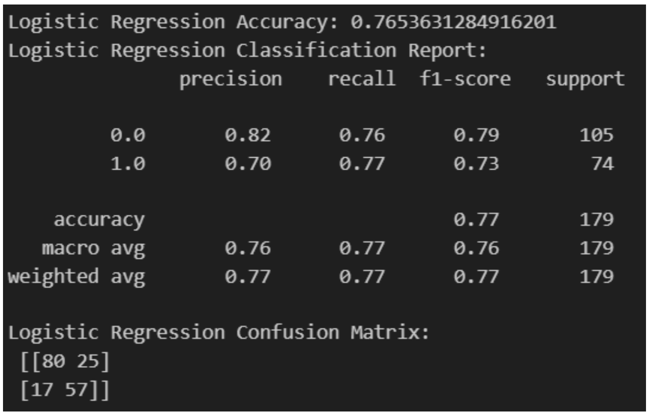
1. **K-Nearest Neighbors (KNN)** achieved the lowest accuracy of 75.97%, with good performance in predicting the majority class (class 0), evident from a recall of 83%. However, its performance in predicting the minority class (class 1) is lower, with a recall of 66%, indicating some misclassifications.



```
KNN Accuracy: 0.7597765363128491
KNN Classification Report:
              precision    recall  f1-score   support

         0.0       0.78      0.83      0.80       105
         1.0       0.73      0.66      0.70        74

    accuracy                           0.76       179
   macro avg       0.75      0.75      0.75       179
weighted avg       0.76      0.76      0.76       179

KNN Confusion Matrix:
 [[87 18]
 [25 49]]
```

Figure 12: KNN results.

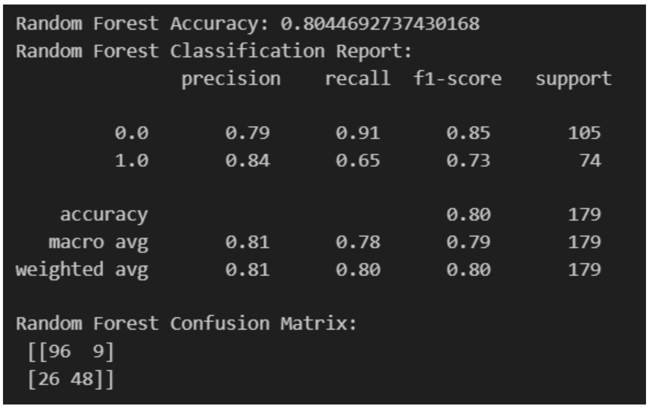
2. **Logistic Regression** got an accuracy of 76.53%. It has almost balanced precision and recall values for both classes . Its recall for class 1 is slightly better with score of 77%, but its recall for class 0 is slightly lower.



```
Logistic Regression Accuracy: 0.7653631284916201
Logistic Regression Classification Report:
              precision    recall  f1-score   support

         0.0       0.82      0.76      0.79       105
         1.0       0.70      0.77      0.73        74

    accuracy                           0.77       179
   macro avg       0.76      0.77      0.76       179
weighted avg       0.77      0.77      0.77       179

Logistic Regression Confusion Matrix:
 [[80 25]
 [17 57]]
```

Figure 13: Logistic regression results.

3. **Random Forest** achieved the best accuracy of 80.44%, providing a excellent relations between precision and recall across both classes. It got the best recall for predicting the majority class(class 0) of 91%, but stil 65% for predicting the minority class(class 1).



```
Random Forest Accuracy: 0.8044692737430168
Random Forest Classification Report:
              precision    recall  f1-score   support

         0.0       0.79      0.91      0.85       105
         1.0       0.84      0.65      0.73        74

    accuracy                           0.80       179
   macro avg       0.81      0.78      0.79       179
weighted avg       0.81      0.80      0.80       179

Random Forest Confusion Matrix:
 [[96  9]
 [26 48]]
```

Figure 14: Random forest classifier results.

The evaluation of the machine learning models reveals that feature scaling and feature selection significantly impact their performance. K-Nearest Neighbors (KNN) achieved the highest accuracy when scaled features were used but struggled with predicting the minority class (class 1) in both scenarios. Logistic Regression demonstrated consistent and balanced performance across classes, making it a reliable but slightly less accurate option compared to Random Forest and KNN. Random Forest consistently provided a good trade-off between precision and recall, achieving the best performance with the chosen features, particularly excelling in predicting the majority class (class 0). Overall, Random Forest is the most suitable model for applications requiring balanced performance, while KNN is preferable for tasks prioritizing overall accuracy.

5 Conclusions

The project demonstrated that machine learning algorithms can effectively predict survival outcomes in a binary classification problem like the Titanic dataset. Random Forest emerged as the most robust model, balancing precision and recall effectively, while KNN performed well in overall accuracy with scaled features. Logistic Regression, although slightly less accurate, provided consistent and interpretable results. Feature scaling and selection significantly impacted model performance, reaffirming their importance in the preprocessing part.

To enhance the model further, additional features such as family size and cabin location could be explored to improve predictions. Furthermore, testing more advanced models, such as Gradient Boosting or Neural Networks, may yield better accuracy and generalization.

References

1. GeeksforGeeks, *K-Nearest Neighbours (KNN)*, available at: https://www.geeksforgeeks.org/k-nearest-neighbours/

2. GeeksforGeeks, *Random Forest Algorithm in Machine Learning*, available at: https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/

3. GeeksforGeeks, *Understanding Logistic Regression*, available at: https://www.geeksforgeeks.org/understanding-logistic-regression/

4. Will Cukierski, *Titanic - Machine Learning from Disaster*, available at: https://www.kaggle.com/competitions/titanic, 2012, Kaggle.