Spaceship Titanic

Wassim Mecheri

November 2024

https://github.com/wmecheri/machine_learning/tree/main/project/Wassim_Spaceship_Titanic

Abstract

This project focuses on the "Spaceship Titanic" competition from Kaggle, where the task is to predict if passengers on the Spaceship Titanic were transported to an alternate dimension following a collision with a spacetime anomaly. The most challenging part of this project was the data preparation step, as almost all features contained missing values that required careful imputation and feature engineering to extract useful data from some columns.

This report discusses the implementation of four machine learning algorithms: Logistic Regression, MLP Classifier, Gradient Boosting Classifier, and Random Forest Classifier. The goal was to compare these models, each representing a different kind of machine learning approach: Logistic Regression as a simple linear model, MLP Classifier as a neural network, and Gradient Boosting and Random Forest as tree-based ensemble models.

Hyperparameter tuning experiments improved results for most models, with the exception of Random Forest. The models were evaluated using accuracy, F1 score, recall, precision, and a confusion matrix to measure their effectiveness.

Results showed that Random Forest outperformed other models, slightly surpassing Gradient Boosting, which itself was better than the MLP Classifier. Logistic Regression performed the worst of the four models. This outcome highlighted the effectiveness of tree-based models, including Random Forest and Gradient Boosting, in binary classification tasks. The neural network model (MLP) did not perform as well as expected, possibly due to the nature of the dataset and the need for more sophisticated feature engineering.

In short, the tree-based models, Random Forest and Gradient Boosting, showed the highest levels of effectiveness for this classification task. Future work could focus on improving feature engineering, exploring more imputation techniques, and fine-tuning hyperparameters to further enhance model performance. Additionally, exploring other model types could potentially produce even better performance. Another potential solution would be to collect more data to enhance the training process.

1 Introduction

This project is centered around the "Spaceship Titanic" dataset from a Kaggle competition, which presents a binary classification problem: predicting if passengers on the Spaceship Titanic were transported to an alternate dimension after a collision with a spacetime anomaly. Inspired by the widely-used "Titanic" dataset, the "Spaceship Titanic" challenge offers a similar structure but with new features that reflect the complexity of interstellar travel and a unique anomaly, making it a fresh take on the classic problem and an ideal educational project.

The provided train and test datasets include approximately 13,000 passenger records. For this project, only the training data is utilized, as it is the only dataset containing the target feature, which is necessary for calculating metrics. The analysis ultimately focuses on around 8,700 entries. The data includes different attributes, such as the passenger's home planet, age, VIP and cryosleep status, cabin details, and their use of various spaceship amenities. The target variable, "Transported," indicates whether a passenger was transported to another dimension. Detailed descriptions of the dataset features are available in the Data section of the competition. Despite the dataset's richness, almost all the features have missing values that must be imputed carefully. Additionally, some features required feature engineering to extract valuable insights, while others were considered irrelevant and removed.

The primary goal of this project is to predict which passengers were transported to an alternate dimension by applying various machine learning algorithms. The aim is to compare the performance of four different models: Logistic Regression, MLP Classifier, Gradient Boosting Classifier, and Random Forest Classifier, each representing a distinct approach to solving classification problems. This task also includes addressing missing data, performing feature engineering, and evaluating model performance using accuracy, F1 score, recall, precision, and confusion matrix metrics.

The research questions guiding this analysis include:

- How can the dataset be preprocessed and feature-engineered to improve model performance?
- Which machine learning model (Logistic Regression, MLP Classifier, Gradient Boosting, or Random Forest) will be most effective in predicting passenger transport, and why?
- What insights can be gained from comparing tree-based models to neural networks in a classification task with missing data?

This project seeks to demonstrate the application of machine learning techniques, model evaluation, and comparison to address a "real-world" context. By the end of the analysis, the objective is to identify the most effective model and provide recommendations for improving prediction results.

2 Methodology and Preprocessing

2.1 Feature Engineering

The first step in this project was feature engineering, focusing on transforming and extracting relevant information from the existing dataset:

- **PassengerId**: The "PassengerId" feature was considered irrelevant, as it contained no direct predictive value. However, it was useful for extracting family or group-related information. PassengerId was split into two new features: "PassengerGroup" (representing the group the passenger is traveling with) and "PassengerNumber" (indicating the individual's number within the group). After extraction, the original "PassengerId" feature was removed.
- **Cabin**: The "Cabin" feature was split into "Deck", "Num", and "Side" based on its format (deck/num/side). While "Num" was irrelevant, both "Deck" and "Side" were retained, as they might have an influence on whether a passenger was transported, given that the anomaly might behave differently depending on their location on the ship.

After completing feature extraction, handling missing data is the next critical step.

2.2 Missing Data Imputation

Missing Data Imputation is done using several strategies to fill missing values, relying on both calculations and the relationships between features:

- **VIP**: By examining the expenses for services (RoomService, FoodCourt, ShoppingMall, Spa, VRDeck) between VIP and non-VIP passengers, VIP passengers had higher service expenses. Based on this, missing values in the "VIP" feature were imputed using the mean VIP value. Additionally, passengers with service values higher than the mean VIP expenses were classified as VIP, while those with lower values were classified as non-VIP.

VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck
False	217.22	426.34	173.88	301.71	282.72
True	473.62	1811.39	247.73	760.71	1234.86

Table 1: Average spending in various services by VIP status

- **CryoSleep**: Since passengers in cryosleep are confined to their cabins and do not use onboard services, any missing or zero values in the "CryoSleep" feature were imputed as "True" (indicating the passenger was in cryosleep). If any service-related expenses were recorded, the "CryoSleep" status was marked as "False".

CryoSleep	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck
False	350.15	713.00	270.59	486.09	475.72
True	0.00	0.00	0.00	0.00	0.00

Table 2: Average spending in various services by CryoSleep status

- **Service Features**: Once the "VIP" and "CryoSleep" features were filled, missing values in the service-related columns (RoomService, FoodCourt, ShoppingMall, Spa, VRDeck) were imputed. For VIP passengers, missing values were filled using the mean service expenses of other VIP passengers. Non-VIP passengers were filled with the mean values for non-VIP passengers. Afterward, cryosleeping passengers had all service expenses set to zero.
- **HomePlanet**: Non-VIP passengers were predominantly from Earth, and VIP passengers mostly came from Europa. Therefore, missing "HomePlanet" values were imputed based on these observations, filling missing values with "Earth" for non-VIP passengers and "Europa" for VIP passengers.
- **Deck**: Using a graph showing the relationship between "Deck" and "HomePlanet", there is a strong correlation between Earth-origin passengers and Deck G, and Mars-origin passengers and Deck F. Missing values in the "Deck" feature were imputed based on these relationships. For Europa-origin passengers, who did not have a clear deck association, a probable deck based on the available data was chosen.

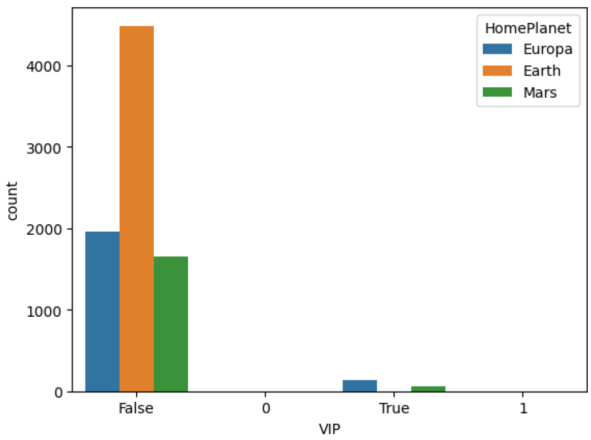


Figure 1: Relation VIP HomePlanet

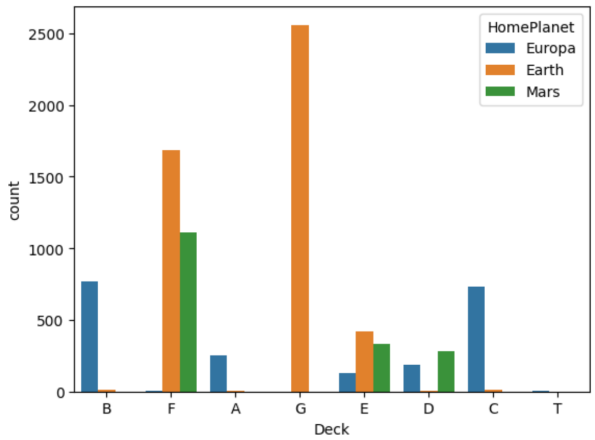


Figure 2: Relation Deck HomePlanet

- **Side and Destination**: The "Side" feature was symmetrical and showed no significant trends, missing values were filled with the mode of the column. For "Destination," there were no meaningful relationships with other features, so missing values were also filled with the mode.
- **Age**: "Age" was imputed using the mean value after confirming that the distribution of ages was approximately normal, making the mean a reasonable choice for imputation.

2.3 Data Types and Categorical Features

After handling missing values, binary categorical features (e.g., "Side") were converted into boolean types and categorical features like "HomePlanet," "Destination," and "Deck" were encoded using one-hot encoding.

2.4 Feature Correlations

Checking for highly correlated features is important because they can create redundancy in the model. Removing one of the correlated features helps simplify the model, reduce overfitting, and improve its interpretability.

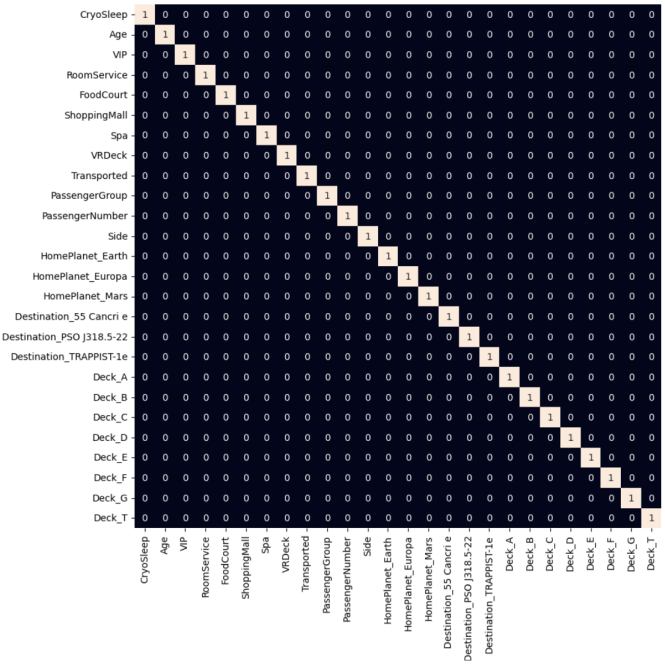


Figure 3: Feature Correlations

2.5 Normalization

Since many machine learning algorithms, especially the MLP classifier, are sensitive to the scale of input features, standardization was applied to normalize the dataset. This step ensures that features are on the same scale, preventing features with larger numerical values from dominating the learning process. Two scaling methods were applied to standardize the dataset:

- **StandardScaler**, which scales the data to have a mean of 0 and a standard deviation of 1, ensuring that the model can learn efficiently without bias toward certain features.
- **MinMaxScaler**, which scales the data to a specified range, typically [0, 1], by transforming features based on their minimum and maximum values, ensuring that all features are on the same scale.

The scalers were applied to the training data (excluding the target variable, "Transported") and then the fitted scalers were used to transform the test data.

2.6 Hyperparameter Tuning

To improve model performance, hyperparameter tuning for the models was attempted, experimenting with different values for key hyperparameters. Grid search was used for key hyperparameters, searching for the best combination:

- **Logistic Regression**: No hyperparameter tuning was done for this model.
- **MLP Classifier**: "hidden_layer_sizes" and "learning_rate_init" parameters, testing various combinations of layer sizes ((10, 10), (50, 50), (100,)) and learning rates.
- **Gradient Boosting**: the number of estimators ("n_estimators") and learning rate, testing different combinations to find the optimal balance between bias and variance.
- **Random Forest**: the number of estimators ("n_estimators"), the maximum tree depth ("max_depth"), and the minimum samples required for a split ("min_samples_split"), trying different values to improve generalization.

In the end, the hyperparameter tuning improved the results for almost all models but did not lead to better performance than using the default values for the Random Forest model. However, this process highlights how different hyperparameters affect model performance.

3 Algorithm Comparison

In this project, four distinct models were evaluated: Logistic Regression, MLP Classifier, Gradient Boosting Classifier, and Random Forest Classifier. Each model type provides unique benefits and limitations, and their comparative analysis gives insight into which approaches are better suited for this dataset's complexity.

3.1 Logistic Regression

Logistic Regression was selected as a baseline model due to its simplicity and interpretability. It is a linear model, making it suitable as a starting point for comparison with more complex algorithms. Logistic Regression is computationally efficient, interpretable, and performs well when the relationship between features and the target variable is approximately linear. Its assumption of linear relationships limits its ability to capture more complex, non-linear dependencies, which likely exist in this dataset due to the interactions between extracted features (e.g., CryoSleep, spending habits, family groups).

3.2 MLP Classifier

The MLP Classifier represents a neural network approach, selected to test the model's ability to capture non-linear relationships in the data. Neural networks can model complex, non-linear dependencies, which can be beneficial in a dataset with intricate feature interactions. They can theoretically outperform simpler models by learning complex patterns across multiple layers. MLP Classifiers require extensive tuning and sufficient data to avoid overfitting. In practice, they may underperform on structured/tabular data without advanced configuration or larger data volumes. This model's reliance on parameter tuning and sensitivity to data size may have limited its potential here.

3.3 Gradient Boosting Classifier

Gradient Boosting was chosen as a powerful tree-based ensemble method, particularly suited for capturing non-linear relationships and feature interactions. Gradient Boosting's iterative, boosting process allows it to learn from errors sequentially, improving its sensitivity and adaptability to complex patterns. This quality makes it highly effective for structured/tabular data. This method is sensitive to hyperparameters, and its iterative learning can increase training time, especially as the dataset grows. However, the method can yield strong results when configured with suitable parameter tuning.

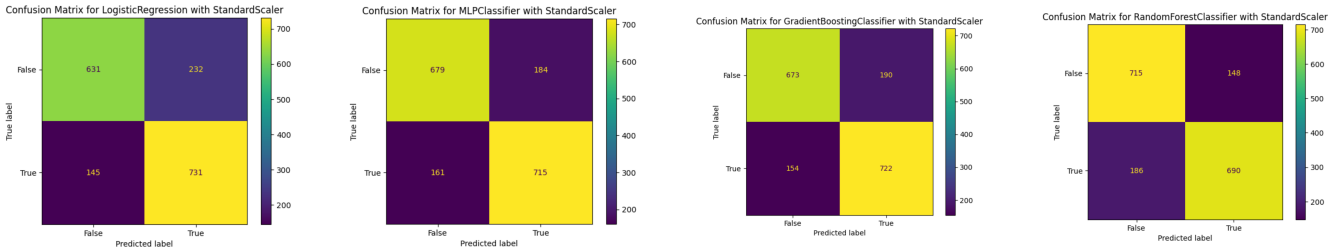
3.4 Random Forest Classifier

Random Forest, another tree-based ensemble method, is well-known for its robustness, accuracy, and ability to handle imbalanced data. It differs from Gradient Boosting in that it builds trees independently and averages their results, balancing bias and variance. Random Forest is robust, less prone to overfitting, and performs consistently well on structured data. Its feature importance capabilities also make it more interpretable. While it generally requires less tuning than Gradient Boosting, it can still be computationally demanding if the number of trees or depth increases significantly. However, its ensemble structure often leads to high generalization capability.

The results of this analysis shows the unique strengths of tree-based ensemble methods, specifically Random Forest and Gradient Boosting, for structured data with complex feature interactions. Random Forest performed better due to its averaging approach and balanced handling of variance, while Gradient Boosting benefited from its iterative refinement, which improved its recall. MLP captured some non-linear relationships but was limited by the need for extensive tuning, while Logistic Regression provided a helpful linear baseline, illustrating the added benefit of non-linear models for this dataset.

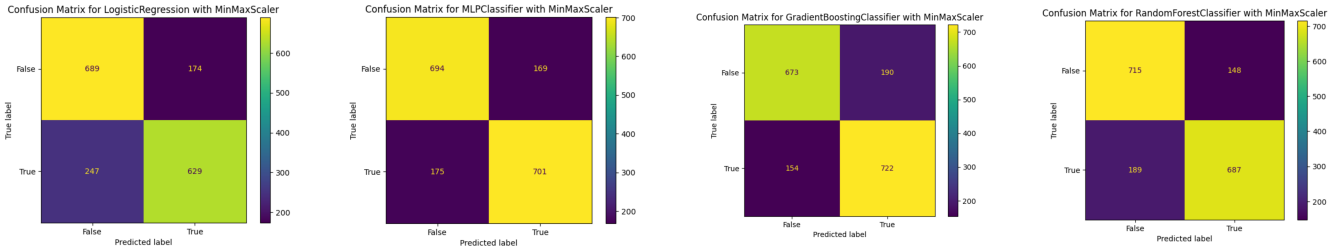
4 Results and Discussion

The table below summarizes the performance of each model using both StandardScaler and MinMaxScaler, highlighting their Accuracy, F1 Score, Recall, Precision metrics and Confusion matrices:



Model	Accuracy	F1 Score	Recall	Precision
Logistic Regression	0.7832	0.7950	0.8345	0.7591
MLP Classifier	0.8016	0.8056	0.8162	0.7953
Gradient Boosting	0.8022	0.8076	0.8242	0.7917
Random Forest	0.8079	0.8051	0.7877	0.8234

Table 3: Model performance using StandardScaler



Model	Accuracy	F1 Score	Recall	Precision
Logistic Regression	0.7579	0.7493	0.7180	0.7833
MLP Classifier	0.8022	0.8030	0.8002	0.8057
Gradient Boosting	0.8022	0.8076	0.8242	0.7917
Random Forest	0.8062	0.8030	0.7842	0.8228

Table 4: Model performance using MinMaxScaler

4.1 Random Forest's Superiority

With an accuracy of 80.79% and the highest precision of 82.34%, Random Forest performed best overall using StandardScaler. Using MinMaxScaler, its performance slightly decreased to 80.62% accuracy and 82.28% precision. The model's strong precision indicates a low false-positive rate, meaning it effectively minimizes cases where non-transported passengers are incorrectly classified as transported. Random Forest's ensemble nature allows it to capture complex interactions without overfitting, contributing to its high accuracy and precision. The slight decline in performance with MinMaxScaler suggests that scaling features within a fixed range may not have been as beneficial for this model, as it may have introduced a different dynamic in the feature contributions that reduced the model's ability to handle some variations in the data. Random Forest could potentially benefit from further hyperparameter tuning, specifically exploring a greater number of trees or adjusting depth constraints to increase accuracy and F1 score.

4.2 Gradient Boosting and Recall

Gradient Boosting achieved a recall of 82.42%, the highest among all models using StandardScaler, making it particularly effective at identifying transported passengers. Its accuracy (80.22%) and F1 score (80.76%) were also strong, slightly trailing behind Random Forest. With MinMaxScaler, Gradient Boosting's performance remained stable, with the same recall of 82.42% and an accuracy of 80.22%. Gradient Boosting's iterative error-correction process likely contributed to its high recall, progressively improving on prior mistakes. This high recall suggests that it is particularly sensitive to positive cases, making it suitable for scenarios where identifying true positives is critical. The consistency of its performance with both scalers indicates that Gradient Boosting is relatively less sensitive to the type of feature scaling compared to other models. Fine-tuning the learning rate and the number of estimators may allow Gradient Boosting to enhance its precision, striking a better balance between recall and precision.

4.3 Comparing MLP and Tree-Based Models

The MLP Classifier performed similarly with both scalers, achieving an accuracy of 80.16% and a recall of 81.62% with StandardScaler. When using MinMaxScaler, its accuracy and precision increased slightly to 80.22% and 80.57%, respectively. However, the F1 score decreased to 80.30%, and recall diminished to 80.02%. The MLP's performance reflects the potential of neural networks to capture non-linear relationships, though it was constrained by a limited dataset and minimal tuning. The fact that MLP performs similarly with both scalers suggests that the model is not highly sensitive to the choice of scaling method in this case, though further tuning and more data could allow it to outperform tree-based models. To further optimize MLP's performance, increasing the complexity of the neural network (e.g., adding layers or neurons) and utilizing more data would likely allow it to better capture patterns and perform closer to tree-based models.

4.4 Logistic Regression's Limitations

Logistic Regression, with an accuracy of 78.32%, recall of 83.45%, and precision of 75.91%, demonstrated decent performance given its linear nature. However, its performance lagged behind the other models due to its limited capacity to capture non-linear relationships. Using MinMaxScaler, Logistic Regression's performance dropped significantly, with an accuracy of 75.79% and a recall of 71.80%. Interpretation: Logistic Regression's performance indicates that while some linear relationships exist in the data, they do not fully explain the complex interactions required for optimal classification. The high recall but lower precision suggests that it identifies transported passengers well but at the cost of more false positives. The drop in performance with MinMaxScaler suggests that the model might not benefit from feature scaling that confines the features to a fixed range, which may reduce its ability to capture the underlying linear relationships. Suggestions for Improvement: While Logistic Regression is primarily a baseline model here, using polynomial features to introduce non-linearity could improve its fit to the data.

4.5 Feature Importance Insights

Both Gradient Boosting and Random Forest provided insights into feature importance:

- Both models identified service-related features (RoomService, FoodCourt, ShoppingMall, Spa, VRDeck) as highly influential in predicting transported status, suggesting that spending behavior might have a strong association with the likelihood of being transported.

- Both models ranked CryoSleep and PassengerGroup as significant features, highlighting the importance of sleep status and family grouping.
- Random Forest ranked Age as a more influential feature, while Gradient Boosting emphasized Deck G. This distinction reflects each model's unique approach to interpreting feature interactions—Random Forest's ensemble averaging versus Gradient Boosting's sequential refinement.

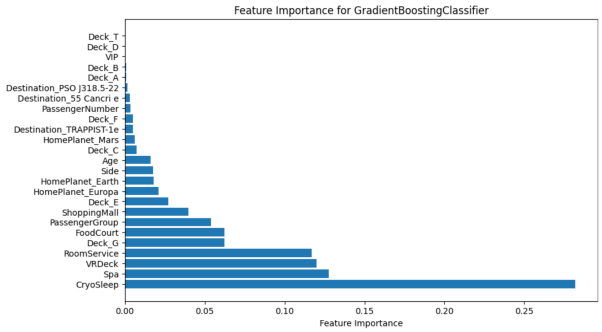


Figure 4: Feature importance Gradient Boosting

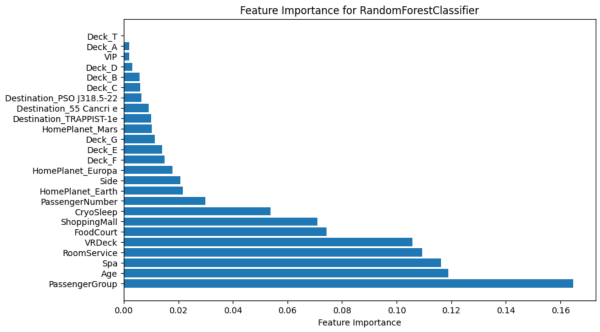


Figure 5: Feature importance Random Forest

4.6 StandardScaler over MinMaxScaler

Overall, StandardScaler generally led to better performance compared to MinMaxScaler across the models, particularly for Random Forest and Logistic Regression. MinMaxScaler did improve the performance of models like MLP, but it seemed to decrease the performance of Logistic Regression and slightly impacted Random Forest.

4.7 Future improvements could include

More detailed hyperparameter tuning, particularly for tree-based models, to optimize precision and accuracy. Increased dataset size or network complexity for MLP to capture underlying patterns more effectively. Additional feature engineering to further uncover potential patterns and improve the models' predictive power. These recommendations aim to refine model performance by enhancing the recall-precision balance and improving interpretability, making the models more robust and applicable in future iterations.

5 Conclusion

This project aimed to predict whether passengers aboard the "Spaceship Titanic" were transported to an alternate dimension following a spacetime anomaly. After conducting extensive data preprocessing and feature engineering, four machine learning models were evaluated: Logistic Regression, MLP Classifier, Gradient Boosting, and Random Forest. Through rigorous comparison, key insights were drawn regarding the strengths and weaknesses of each model.

5.1 Key Findings

Random Forest emerged as the top performer, with the highest accuracy (80.79%) and precision (82.34%) when using StandardScaler. This model's ensemble nature allowed it to capture complex interactions between features, maintaining robustness and generalizability. Despite a slight drop in performance with MinMaxScaler, it consistently outperformed other models. Gradient Boosting demonstrated strong recall (82.42%) and was particularly effective at identifying transported passengers, though its precision lagged behind Random Forest. The model's iterative learning process made it highly sensitive to errors, enhancing its ability to detect true positives. MLP Classifier showed moderate performance with both scalers, with a slight improvement in F1 score and precision when using MinMaxScaler. However, it did not surpass the tree-based models, possibly due to limited data and minimal tuning. Logistic Regression, while computationally efficient, performed the worst among all models, struggling to capture the non-linear relationships inherent in the dataset. It exhibited good recall but lower precision and suffered a performance drop with MinMaxScaler.

5.2 Contributions

This project provided a comprehensive exploration of machine learning models applied to a real-world classification task, demonstrating the importance of data preprocessing, feature engineering, and model evaluation. It showcased the value of tree-based ensemble models, especially Random Forest, for complex, structured datasets. Additionally, it offered insights into the effectiveness of various scalers on model performance, highlighting that the choice of scaling method can significantly influence outcomes, especially for models like Logistic Regression and MLP.

5.3 Limitations

While Random Forest and Gradient Boosting performed well, the project did not achieve significant performance improvements through hyperparameter tuning. The MLP Classifier's potential was limited by the available data and the lack of advanced tuning. Moreover, the relatively small dataset size may have restricted the models' capacity to fully capture complex patterns, especially for neural network models.

5.4 Future Improvements

Future work could involve further hyperparameter tuning for tree-based models to optimize precision and recall, as well as exploring more advanced feature engineering techniques. Increasing the dataset size or using more sophisticated architectures for the MLP Classifier may also improve its performance. Additionally, experimenting with more diverse imputation techniques and feature scaling methods could yield further improvements, providing a more robust solution for the prediction task. Expanding the dataset and applying more complex neural networks may enhance performance, particularly for non-linear models like MLP.

In summary, while Random Forest proved to be the most effective model for this task, further exploration of model tuning and feature engineering could lead to even better results in future iterations of this project.

References

[1] Kaggle, *Spaceship Titanic Competition*.
https://www.kaggle.com/competitions/spaceship-titanic/overview.

[2] GeeksforGeeks, *Spaceship Titanic Guide*.
https://www.geeksforgeeks.org/spaceship-titanic-project-using-machine-learning-python/.

[3] Kaggle, *Titanic - Machine Learning from Disaster*.
https://www.kaggle.com/c/titanic/overview.

[4] Kaggle, *First Kaggle Attempt - Feedback Welcome!*.
https://www.kaggle.com/code/liamstuart3141/first-kaggle-attempt-feedback-welcome.

[5] Kaggle, *Spaceship Titanic: Logistic Regression Model Train*.
https://www.kaggle.com/code/aslemimolu/spaceship-titanic-logistic-regression-model-train.

[6] Kaggle, *Spaceship Titanic Using Random Forest and Xgboost*.
https://www.kaggle.com/code/hardikgarg03/spaceship-titanic-using-random-forest-and-xgboost.