IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2025

DOI: 10.17148/IJIREEICE.2025.131114

Machine Learning Approaches to Passenger Survival Prediction: A Titanic Dataset Analysis

ESWARAMUTHU M[1], M KIRITHIKA[2], ABHISHEK R[3], LAVANYA S[4], MOHANA KRISHNAN B[5], VAISHNAVI S[6], Dr. M. ULAGAMMAI[7]

Department of Computer Science and Engineering (Emerging Technologies),

SRM Institute of Science and Technology, Vadapalani Campus, Chennai, 600026, India[1-6]

Guide, Department of Computer Science and Engineering (Emerging Technologies),

SRM Institute of Science and Technology, Vadapalani Campus, Chennai, 600026, India[7]

Abstract: The Spaceship Titanic competition on Kaggle offers a unique machine learning challenge inspired by the classic Titanic disaster, reimagined in a futuristic space setting. The goal of this research is to create a predictive model that decides if passengers on the Spaceship Titanic were transported to another dimension after the ship collided with a spacetime anomaly. This study examines different data preprocessing, feature engineering, and classification techniques to improve predictive performance on the dataset.

The dataset provides details on passenger demographics, travel itineraries, and onboard service usage, offering rich information for building classification models. Our approach involves meticulous data cleaning, handling missing values through imputation, normalizing numeric features, and encoding categorical variables. We evaluated several supervised learning algorithms—Logistic Regression, Random Forest, Gradient Boosting, and XGBoost—using metrics like accuracy, precision, recall, and F1-score. The top-performing model demonstrated strong generalization on the test set, highlighting the value of analyzing feature interactions and fine-tuning hyperparameters. This research underscores the power of data-driven techniques in predictive analytics and illustrates how machine learning can effectively tackle complex, hypothetical problems with incomplete and noisy data. Our findings contribute to the broader understanding of structured data prediction and highlight the effectiveness of ensemble methods for classification in both practical and simulated scenarios.

I. INTRODUCTION

The Spaceship Titanic scenario imagines a futuristic space voyage where a colossal interstellar liner carrying thousands of passengers meets a mysterious mishap mid-journey. After this unforeseen event, part of the travelers seemingly disappear into another dimension, while others remain on the ship. This fictional premise provides a creative backdrop for exploring how data-driven models can uncover hidden patterns and predict human outcomes under uncertainty.

The goal of this study is to construct an intelligent predictive system that determines the likelihood of each passenger being transported based on the information provided in the dataset. The dataset offers multiple descriptive variables—such as age, origin planet, travel destination, cabin identifiers, and expenditure on various onboard services—that reveal lifestyle and behavioral patterns among passengers.

This work adopts a complete machine learning workflow involving data preparation, exploratory analysis, feature construction, model training, and validation. Special attention is given to handling incomplete entries, transforming categorical values into numerical form, and testing several ensemble-based learning algorithms for performance comparison. These methods aim to enhance prediction reliability while preventing overfitting.

Beyond technical modeling, the project reflects how data science can interpret imaginative problems using real analytical thinking. It strengthens understanding of supervised learning principles and provides hands-on exposure to critical concepts such as data preprocessing, model selection, and performance evaluation—skills essential for practical applications of artificial intelligence and predictive analytics.

II. LITERATURE SURVEY

Several data-driven investigations have been carried out on the Spaceship Titanic classification challenge, each employing distinctive preprocessing and modeling techniques. In almost all prior studies, exploratory data analysis (EDA) forms the initial step, using graphical summaries to identify dependencies between passenger traits—such as age group, spending behavior, and cabin position—and the probability of transport.

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2025
DOI: 10.17148/IJIREEICE.2025.131114

A recurring insight from literature is that feature transformation greatly improves model outcomes. Researchers frequently decompose complex attributes like Cabin into smaller units—Deck, Number, and Side—to capture spatial details within the ship's layout. Variables such as HomePlanet, CryoSleep, and Destination have been highlighted as particularly influential in determining passenger fate.

When comparing algorithms, earlier works commonly relied on baseline models including Decision Trees and Logistic Regression, whereas later studies demonstrated superior accuracy through ensemble techniques such as Random Forest, Gradient Boosting, XGBoost, and LightGBM. These ensemble approaches integrate multiple models to form stronger collective predictors, offering better generalization to unseen data.

Additional enhancements in recent research include dimensionality reduction via Principal Component Analysis (PCA) to simplify feature space, and hyperparameter tuning using automated search strategies to identify optimal model settings. Some researchers have also experimented with deep learning architectures, but ensemble tree-based models continue to deliver the most consistent and interpretable outcomes for structured tabular data.

Collectively, existing studies emphasize that the best solutions emerge from careful feature engineering, robust preprocessing, and hybrid ensemble learning, all of which guide this work's methodological design and experimentation

III. PROPOSED METHODOLOGY

a. Overview of the Methodological Framework

The solution is a comprehensive Machine Learning pipeline that predicts passenger transportation outcomes using the Spaceship Titanic dataset. Such a methodology includes several key stages: data acquisition and preprocessing, exploratory data analysis, feature engineering, model development, ensemble learning, and performance evaluation. Each stage is structured to further enhance the accuracy score while preserving the interpretability of the model.

b. Data Acquisition and Preparation

This solution uses the Titanic dataset that has separate training and testing subsets. The training set consists of records of passengers with known outcomes of transportation, while the testing set is made up of records for which we need to make predictions. First, the data are loaded using the pandas library for efficient manipulation and analysis down the pipeline. Therefore, the methodology involves data preprocessing to address quality issues such as missing values, categorical encoding, and feature scaling. In this regard, missing value imputation strategies are performed based on the nature of each feature, ranging from a purely statistical approach to domain-specific logic. The categorical variables, for example, which are home planet, destination, and cabin information, are encoded so that their textual values get converted into numeric form processable by different Machine Learning Algorithms.

c. Feature Engineering and Selection

Feature engineering is a critical part of the methodology, aimed at the extraction of meaningful patterns from the raw data. Consider, for instance, the cabin feature that has been decomposed into components such as deck, cabin number, and side, thus enabling the model to capture spatial relationships that might impact transportation outcomes. Other analyses involve spending patterns through passenger expenditure on amenities such as room service, food court, shopping mall, spa, and virtual reality deck facilities. These spending features have been standardized to consistently scale across the different ranges. Particular attention is paid to how cryosleep status interacts with spending because, for most categories of amenities, passengers in cryosleep have zero expenditure.

d. Model Development Strategy

It will make use of a multi-model methodology, which will play to the strengths of various Machine Learning algorithms. Different algorithms capture different patterns in the data, and usually an ensemble model ensures better predictive performance than relying on a single model. The complementary strengths of each algorithm are harnessed in this approach in a way to improve overall robustness and accuracy score.

e. Individual Classifier Implementation

Training a set of base classifiers independently constitutes the first step:

- **RandomForestClassifier:** Ensemble method, building multiple decision trees and outputting the majority vote of their predictions. Tuning of hyperparameters are done by fine tuning number of trees/estimators, maximum depth, and minimum samples per split. RFs are suitable for the problem in question, since, among other features, they naturally handle nonlinear relationships and interactions.
- **Gradient Boosting Classifier:** This is a sequential ensemble technique whereby models in this class are built

DOI: 10.17148/IJIREEICE.2025.131114

iteratively, while each new tree tries to correct and reduce the errors of its predecessors. The implementation involves careful tuning of the learning rate and regularization parameters to avoid overfitting while maintaining strong predictive power.

- **XGBoost:** An extension of classic gradient boosting with more sophisticated regularization and parallel processing capabilities. It is initialized here with bespoke parameters controlling tree construction, the learning rate, and regularization strength. XGBoost is notably very good at handling imbalanced data and modeling complex feature dependencies.
- **LightGBM Classifier:** A fast gradient boosting variant that uses a histogram-based approach to reduce the memory usage and speed up the training. The configuration has been optimized for this dataset; it uses a leaf-wise growth strategy and includes built-in categorical feature support.
- **CatBoost Classifier:** A gradient boost technique for categorical features. It uses an ordered boosting methodology that prevents prediction shift and allows for better generalization. Early stopping is done during training, where the performance on the validation set is monitored and used to prevent overfitting.

f. Ensemble Learning Architecture

After training different classifiers, an advanced ensemble framework is used. This approach combines multiple base models through a voting mechanism that aggregates their predictions to make up the final output.

g. Voting Classifier Configuration

The ensemble involves soft voting whereby each base model gives probability estimates, instead of hard class labels. The probabilities are summed up using weights determined by the performance of each model on validation. Models which have higher accuracy score coupled with better generalization, especially XGBoost and LightGBM, will be weighted higher. The four main classifiers involved in voting are Random Forest, Gradient Boosting, XGBoost, and LightGBM. This heterogeneous mix ensures diversity while predicting output, as each algorithm has a different learning strategy and captures unique parts of the data.

h. Training the Model and Validation Protocol

Robust performance evaluation and the prevention of overfitting are assured by the structured validation approach during the training process. The stratified split of the training dataset into training and validation subsets is done in such a way that class distribution is preserved in both. Each classifier has been trained on the training subset by systematic experimentation and cross-validation with different hyperparameter configurations. This independent holdout set has been used for the evaluation of the models, thus enabling unbiased assessment of their generalization.

The main performance metric is the classification accuracy, i.e., the share of correctly predicted transportation outcomes. Besides that, precision, recall, and F1-score metrics are used to further illustrate model performance for both transported and non-transported classes.

i. Final Prediction Generation

After carrying out the model training and validation steps, the test dataset is used to make final predictions using the ensemble classifier. Since CatBoost had the best accuracy on the validation set, it will be used as the main predictor for generating the submissions. Predictions are converted to booleans to match the required output format. The final submission file will contain the passenger identifiers with their predicted transportation outcome in competition format. This systematic approach ensures reproducibility and consistency across the pipeline.

j. Computational Implementation

The implementation is performed in Python, using libraries like scikit-learn for Machine Learning algorithms, pandas for data manipulation, and NumPy for numerical computations. The computational framework leverages parallel processing wherever applicable to optimize execution time for training ensembles and generating predictions. Overall, this framework balances model complexity with practical considerations, ensuring robust predictive performance while maintaining computational efficiency and interpretability of the results.

IV. RESULTS AND DISCUSSION

Results

We ran several different machine learning models on the Spaceship Titanic dataset to figure out which ones worked best for predicting passenger transportation outcomes. Our training set had roughly 3,700 passengers with information about their home planet, where they were going, cabin assignments, age, and what they spent money on.

- **Model Performance:**

Starting with the basics made sense, so we tried Logistic Regression first and got 78.33% accuracy. Gaussian Naive

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2025

DOI: 10.17148/IJIREEICE.2025.131114

Bayes did a bit worse at 77.68%. Random Forest bumped things up to 79.14%, which told us that tree-based approaches were catching patterns the simpler models couldn't see.

Things got interesting with gradient boosting methods. XGBoost and LightGBM both hit 79.83% accuracy. We alsospent time tweaking Random Forest settings with more trees and deeper splits, which got us to 79.37%. When we combined our best models in a voting setup, we reached 79.95%.

CatBoost turned out to be our winner at 80.21% accuracy. This makes sense because CatBoost is really good at handling categorical stuff, and we had tons of that in our data. One weird thing though was that the standard Gradient Boosting Classifier totally bombed at 60.35%. The default settings probably just didn't work well for this particular dataset. Table 1 has all the numbers.

Model	Accuracy (%)	Notes
Gaussian Naive Bayes	77.68	Simple probabilistic approach
Logistic Regression	78.33	Baseline model
Random Forest	79.14	Standard configuration
Optimized Random Forest	79.37	Additional trees, deeper splits
XGBoost	79.83	Gradient boosting
LightGBM	79.83	Gradient boosting
Voting Ensemble	79.95	Combined models
CatBoost	**80.21**	**Best performer**
Gradient Boosting	60.35	Poor default parameters

TABLE 1: Model Performance Comparison

V. DISCUSSION

(i) Impact of Feature Engineering

The work we did on features mattered way more than which algorithm we picked. Instead of just dumping raw data into the models, we actually thought about what would be useful. We made features for spending ratios, figured out if people were traveling alone or with others, and broke down those cabin codes into separate pieces like deck, section, and side.

Spending patterns turned out to be super informative. People who didn't spend anything at all acted totally different from big spenders. We tracked total spending, made ratios comparing luxury purchases to basic stuff, and even caught weird cases where someone was supposedly frozen in cryosleep but had charges on their account.

The cabin breakdown helped a lot too. When you see "B/45/P" as just one string, it doesn't tell you much. But splitting it into deck B, section 45, and port side let the models spot location patterns they couldn't see before.

We dealt with missing data pretty carefully. For numbers, we filled gaps with the median. For categories, we used whatever value showed up most often. Also had to convert all the true/false stuff to 1s and 0s and make sure categorical variables were formatted right.

(ii) Model Comparison and Performance

Logistic Regression and Gaussian Naive Bayes were decent starting points, but they're just too basic for complex data like this. Random Forest did better at 79.14% because it can figure out how features interact. Like maybe age matters differently if you're traveling solo versus with family. Tweaking Random Forest's settings pushed it slightly higher to 79.37%.

The specialized gradient boosting stuff worked best. XGBoost and LightGBM both got to 79.83%, and CatBoost led everything at 80.21%. These build a bunch of decision trees in sequence where each one tries to fix what the previous ones got wrong. That back-and-forth improvement is why they beat simpler methods.

The regular Gradient Boosting Classifier was a disaster at 60.35% though. Its default settings clearly didn't match this dataset well. This shows why people usually go with XGBoost or LightGBM instead of the basic version, or why you need to spend time tuning parameters.

CatBoost probably won because of how it deals with categories. Most algorithms make you do a bunch of preprocessing to convert categories into numbers, but CatBoost just handles it internally. Since we had multiple categorical features, this gave it an edge.

The voting ensemble combined predictions from our top models and reached 79.95%. Didn't quite beat CatBoost's 80.21%, but it was more consistent when we tested on different data chunks.

(iii) Model Reliability and Validation

We split the data 70% for training and 30% for validation. Most of our good models had training and validation scores

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2025

DOI: 10.17148/IJIREEICE.2025.131114

that stayed pretty close, usually within a percent or two. This suggested they were learning real patterns instead of just memorizing the training data. CatBoost's 80.21% held steady across several runs, s

VI. CONCLUSION

We ended up with a system that gets predictions right about 80% of the time, which is pretty competitive for this Kaggle competition. CatBoost led at 80.21%, with the voting ensemble at 79.95% and XGBoost/LightGBM at 79.83% right behind it.

- **Key Success Factors**:

Feature engineering made the biggest difference by far. Building useful features like spending ratios, demographic info, and properly split cabin data gave the models way better stuff to work with. Spending patterns were especially predictive. Also helped to identify solo travelers, kids, elderly people, and flag suspicious things like frozen passengers with charges. Handling missing data carefully mattered too. Using medians for numbers and the most common values for categories kept everything clean without creating weird artifacts.

- **Real-World Applications:**

Yeah, this is about a fake spaceship, but the same process works for actual business problems. Predicting customer churn, catching fraud, forecasting hospital readmissions all follow the same basic steps. Clean your data, make good features, try different models, pick what works.

- **Potential Improvements**:

Getting 80% right means we're still wrong one out of five times. We could try stacking models where you use predictions from some models as inputs to another one. We didn't go super deep on tuning parameters either, so there's probably room there. The fact that regular Gradient Boosting flopped at 60.35% shows that proper tuning really matters.

Making interaction features could help. Right now we treat everything separately, but maybe spending means different things at different ages, or certain home planet and destination combos have their own patterns.

This dataset is synthetic and cleaner than real data, so 80% might be optimistic for actual use. We only looked at accuracy too. Depending what you're doing, you might care more about catching all the real cases or avoiding false alarms.

- **Final Assessment:**

This project showed that specialized gradient boosting tools, especially CatBoost, work really well for this kind of problem. The jump from traditional models around 79% to CatBoost at 80.21% seems small but can matter a lot depending on the situation. The regular Gradient Boosting tanking at 60.35% really drives home why people use the optimized versions or spend time tuning.

REFERENCES

[1]. Scikit-learn Documentation (2024). Ensemble Methods. https://scikit-learn.org
[2]. XGBoost Documentation (2024). XGBoost Python API. https://xgboost.readthedocs.io
[3]. LightGBMDocumentation (2024). LightGBM Features. https://lightgbm.readthedocs.io
[4]. CatBoost Documentation (2024). CatBoost Algorithm. https://catboost.ai
[5]. Kaggle Competition: Spaceship Titanic. https://www.kaggle.com/competitions/spaceship-titanic