Yuki Huang & Tommy Liu
May 17th 2024

STA-395 Intro To Machine Learning Final Report:

Predicting Successful Passenger Transportation in a Spaceship Titanic

https://github.com/TML17/Spaceship_Titanic

Abstract

This project addresses the challenge of predicting successful passenger transportation to another dimension in a spaceship Titanic by applying various machine learning techniques. Our approach involved data preprocessing, hyperparameter tuning, model training, and evaluation to identify the most effective model. This study reveals that deep learning models, particularly fully convolutional neural networks (FCNNs) demonstrated the best performance in classifying transported passengers. These findings underscore the potential of advanced machine learning techniques in addressing complex problems.

Keywords: Spaceship Titanic, Machine Learning, Model Selection, Comparative Analysis, Classification Accuracy,

1. Introduction

In a scenario where a spaceship Titanic collided with a spacetime anomaly hidden within a dust cloud, only half of the passengers were transported to another dimension successfully. The challenge of predicting successful passenger transportation is crucial for rescuing lost passengers. This project aims to provide a comprehensive analysis of machine learning models to accurately identify transported passengers. We will implement data cleaning, and preprocessing techniques to prepare the data for modeling, investigate the influence of various features on the prediction outcomes, and explore a variety of machine learning algorithms, including decision trees, random forests, support vector machines, K-nearest neighbors, XGBoost, logistic regression, and neural networks to determine the most effective approach for this unique challenge. By analyzing the model performances based on the accuracy of the predictions, this structured approach will enable us to address the problem with scientific rigor and precision.

2. Methods

2.1 Data Source & Variables

The dataset used in this study to distinguish whether the passenger is transported successfully is from a Kaggle competition called "Spaceship Titanic." While this competition provides three types of datasets — train, test, and submit — in CSV format, we are only using the train.csv dataset.

The dataset contains various features about passengers, including the unique identifier for each passenger (PassengerId), the planet the passenger is from (HomePlanet), whether the passenger elected to be put into cryosleep during the voyage (CryoSleep), cabin number where passenger

stays (Cabin), the age of the passenger (age), the destination of the passenger (Destination), whether the passenger has VIP status (VIP), the amount each passenger spent on different amenities(RoomService, FoodCourt, ShoppingMall, Spa, VRDeck), name of the passenger (Name). And most importantly, the target is whether passengers were transported successfully (Transported).

2.3 Data Preprocessing

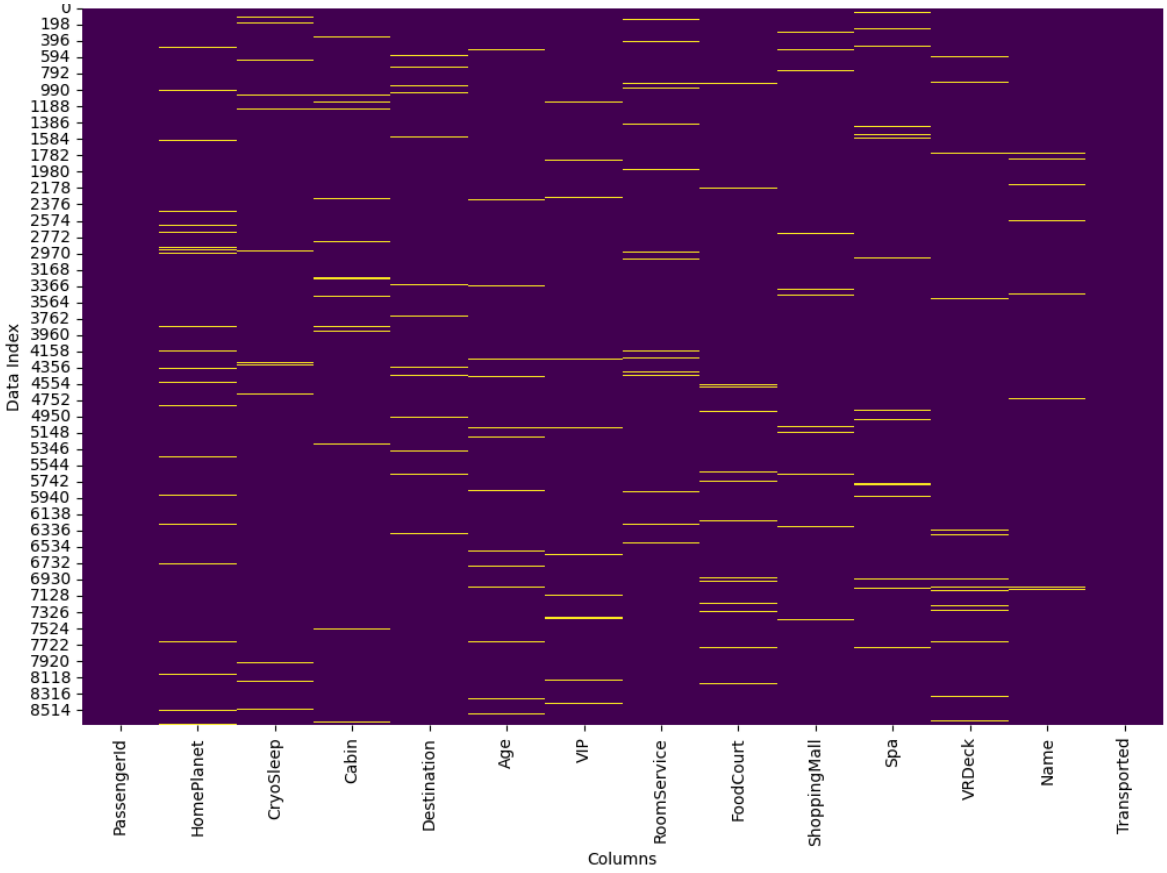

Figure 1. Heatmap of missing values in the Dataset

- Missing values & Imputation

The data contains a lot of missing data and they are randomly distributed across the data(see Figure 1). Thus, if we attempt to delete any rows with missing values, we shall lose a large

number of observations, and the approach of imputation is more desirable for us. In the process of imputation, we handle numerical and categorical variables differently. For numerical variables, we calculate the mean of them and use them as the missing value. For categorical variables, we apply both mode and KNN imputation to our data and will use both of them for evaluation.

- Outliers Handling

To normalize the data and address the issue of potential outliers in the numerical features, a clipping method was applied to the dataset. This involved setting a threshold at the 99th percentile for numerical features. Values above this threshold were changed to threshold values while other data values remained the same. This method helps to reduce the impact of extreme values without removing the data points, preserving the overall structure of the data set.

2.4 Models

The models we explored in this project are listed below.

- K-nearest Neighbors

KNN (K-nearest neighbors) predict the outcome based on grouping individual points to the majority class of its nearest neighbors in the feature space, using distance metrics such as Euclidean distance or Manhattan distance.

- Decision Tree

A decision tree is trained recursively by splitting rules based on features and creating a tree-like model to make classification decisions.

- Random Forest

Random forest is an ensemble method that builds multiple decision trees and aggregates the outputs of each decision tree to make predictions.

- Support Vector Machines

SVM finds the optimal hyperplane that best classifies the data points into classes based on the maximum margin.

- XGBoost

XGBoost is an optimized gradient-boosting implementation that builds an ensemble of decision trees sequentially. Each new tree attempts to correct the errors of the previous trees.

- Logistic Regression

Logistic regression is a method for predicting binary outcomes. It models the probability of the outcome by using a logistic function, which maps linear combinations of all the features to return a probability value.

- Coluntional Neural Networks

CNNs consist of multiple layers, including convolutional layers that apply filters to input data to detect features, pooling layers that downsample the data to reduce dimensionality, and fully connected layers that make the final classification decisions.

- FCNNs

FCNNs consist of multiple layers of neurons where each neuron in one layer is connected to every neuron in the subsequent layer. These networks are used for various classification and regression tasks. FCNNs learn to map input features to output predictions by adjusting the weights of the connections through backpropagation during training.

2.5 Hyperparameter Tuning

In order to find the optimal configuration for a model that minimizes the model's error on the training dataset and accurately predicts successful passenger transportation, hyperparameter tuning is employed in this project. Techniques such as grid search and applying different learning rates are used for selecting the best hyperparameter combinations. After this step, the model is trained by feeding the entire training dataset into the algorithm, which iteratively adjusts the parameters to maximize the accuracy of the model result.

2.6 Performance Evaluation

The accuracy matrix is an essential tool in evaluating the performance of a training model designed to classify transported passengers among all other model evaluation methods. This metric quantifies the proportion of total correct predictions, thereby providing a straightforward assessment of the model's performance in classifying passengers accurately. It is particularly relevant in this scenario where the two classes whether passengers are transported or not are balanced. The accuracy matrix is calculated by

$$Accuracy = \frac{True\ Positive + True\ Negative}{The\ number\ of\ total\ predictions}.$$

The following provides an example of hyperparameter tuning and performance evaluation we used in this project. We applied similar steps to all the models we discussed above.

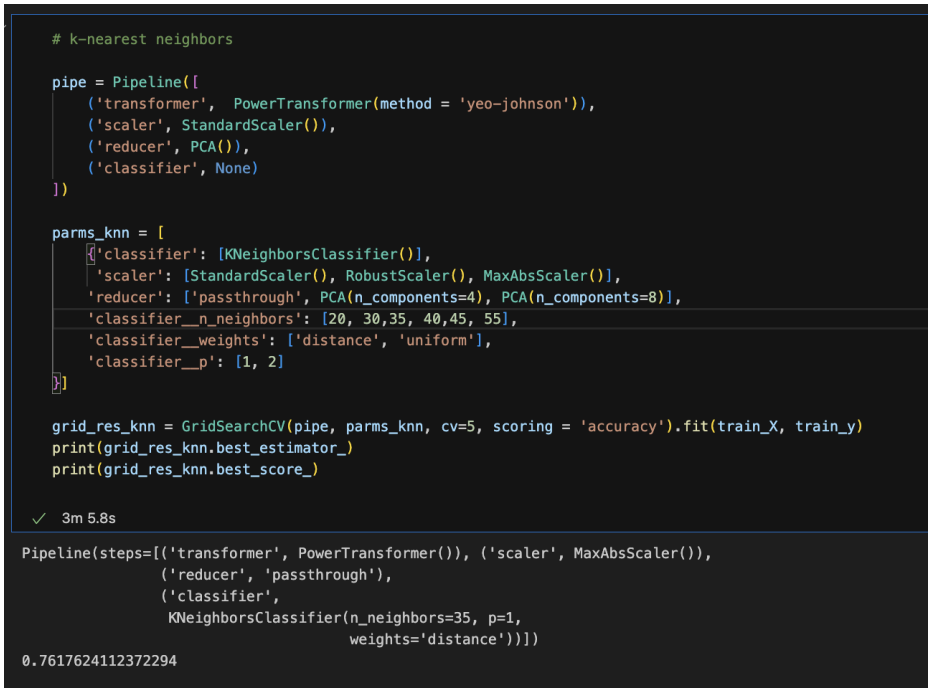

Figure 2. Hyperparameter tuning and model training for KNNs

To evaluate our trained model, we then apply it to a testing dataset, which generates predictions based on the model's learned parameters. These results are subsequently submitted to a Kaggle competition, allowing us to validate the testing accuracy of our model.

3. Results

3.1 Mode Result Comparison

Table 1 shows the comparative analysis of various machine learning models in terms of training and testing accuracy. The Support Vector Machines (SVM) model exhibits a training accuracy of 77.764% and a testing accuracy of 78.372%, indicating a robust generalization from training to validation data. The Convolutional Neural Networks (CNN) model, with a training accuracy of 77.764% and a testing accuracy of 78.25%, also performs well, closely followed by the Fully Convolutional Neural Networks (FCNNs), which achieves the highest testing accuracy of

78.89% with a similar training accuracy. For both deep learning architectures, simpler models like K-nearest Neighbors, Decision Trees, and Logistic Regression show lower testing accuracies, suggesting that more complex models might be better for generalizing an unseen dataset.

Model	Training Accuracy	Testing accuracy
K-nearest Neighbors	76.176%	76.502%
Decision Trees	77.764%	77.32%
Random Forest	76.441%	76.782%
XGBoost	76.187%	76.338%
Support Vector Machines	77.764%	78.372%
Logistic Regression	75.014%	75.66%
Convolutional Neural Network	78.44%	78.25%
Fully Convolutional Neural Network	78. 56%	78.89%

Table 1. Training and testing accuracy of all the models

4. Discussion

4.1 Feature Importance and Data Preprocessing

The preprocessing steps, including handling missing values and outliers, played a crucial role in the model's performance. The imputation of missing values ensured that the data remained robust and complete, while outlier handling helped in normalizing the data distribution. These steps are essential for enhancing model accuracy and ensuring reliable predictions.

The various features, such as passenger age, cabin number, and VIP status, likely contributed differently to the prediction outcomes. Future work could involve a more detailed feature importance analysis to understand which attributes most significantly impact transportation prediction.

4.2 Hyperparameter Tuning Validation

Hyperparameter tuning was a critical step in optimizing model performance. Techniques like grid search helped in identifying the best configurations for each model. The results underscore the importance of fine-tuning hyperparameters to achieve optimal accuracy and prevent overfitting or underfitting. In order to achieve this, we graph out both training and validation accuracy with respect to training epochs for our deep learning models. From the graphs below, a training epoch around 50 would result in a better performance on the validation dataset.

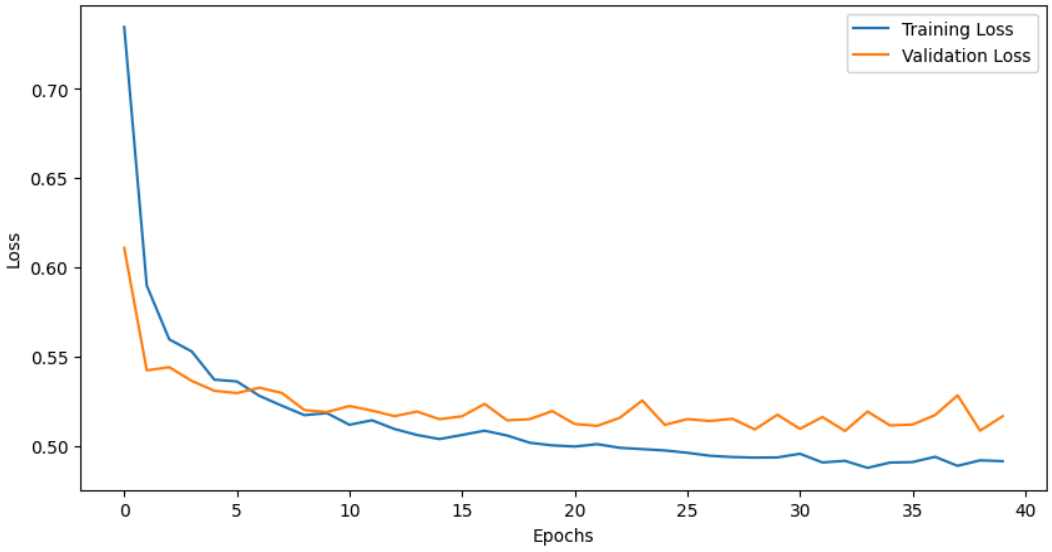

Figure 3. Training and Validation Losses for CNNs

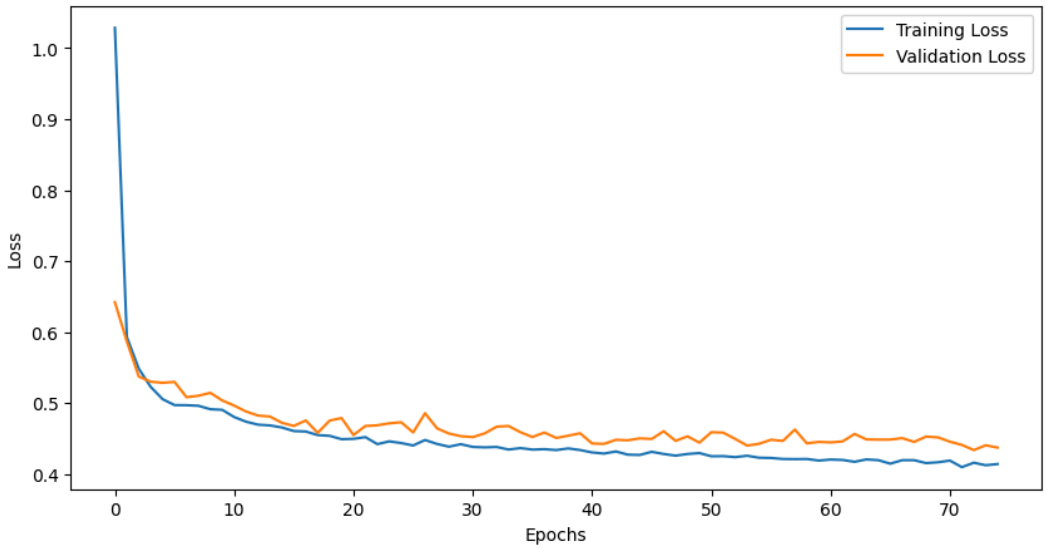

Figure 4. Training and Validation Losses for FCNNs

4.3 Limitations and Future Work

One limitation of this study is the reliance on a single dataset for training and testing. Cross-validation could be employed in future work to ensure the robustness of the model across different data splits. Additionally, exploring other advanced models like ensemble methods combining multiple classifiers or employing more sophisticated neural network architectures could further enhance prediction accuracy. Overall, the structured approach to data preprocessing, model training, hyperparameter tuning, and performance evaluation provided a comprehensive analysis of machine learning models for this unique challenge. The findings suggest that deep learning models, particularly FCNNs, are highly effective for this classification task, while simpler models may require further refinement or feature engineering to improve their performance.

5. Reference

1. **Addison Howard, Ashley Chow, Ryan Holbrook. (2022). Spaceship Titanic. Kaggle. https://kaggle.com/competitions/spaceship-titanic**

2. **Ryan Miller. (2024). Intro to Machine Learning. https://remiller1450.github.io/MLs24.html**