Titanic Classifier

Can Wang[1] and Rigpea Wangchuk[1]

[1]Department of Computer Science, CS 365, Professor Babis Tsourakakis, TF. Tianyi Chen

Link to the project repository: GitHub Repository

Abstract

This project addresses the historical Titanic disaster where 1517 lives were lost. The project aims to analyze features such as sex, class, embankment point, and family relationships that had a significant impact on the survival of the passengers. The dataset is analyzed and then reprocessed to handle any missing values, encode categorical variables, and normalize distributions. Then, feature importance is analyzed through chi-square, pi-values, and visual representations. In addition, a heat map is then provided to sum up the correlations between the features. After this, we use several machine learning models to predict the survival of a passenger given their features. This model could serve as a benchmark for further research in predictive analytics within historical contexts.

Keywords: heatmap, logic regression, linear regression, random tree

1. Introduction

The sinking of the RMS Titanic is one of the most infamous maritime disasters in history, occurring on April 15, 1912, during its maiden voyage from Southampton to New York City. Of the approximately 2,224 passengers and crew aboard, more than 1,500 lost their lives in the icy waters of the North Atlantic, a tragedy that has since captivated the public imagination and academic interest alike. This catastrophic event highlights not only human error and technological failures but also the social and economic disparities of the early 20th century that influenced survival outcomes.

In recent years the Titanic data set has provided a strong and simple introduction to baseline Machine Learning models. It offers most of the features attributed to the passenger's trip which gives the ability for aspirants to understand the influences of the features on the survival of a passenger in the Titanic. By doing this it aims to shed light on the complex interplay of factors that determined survival during the event.

The objective of this project is to strengthen the concepts learned in class as well as go out of the box and learn and implement concepts that are being used in the real world. By the end of the analysis, we not only want to gain an understanding of the Titanic tragedy but also gain the ability to apply analytical techniques we come across.

2. Related work

For the Titanic Data set, we approached it with a supervised learning environment. Therefore, we realized the use of a classifier to generate our prediction and get an accuracy score.

This approach was chosen as we were already presented with a data set that we could split into a train and a test set with known target values to make a judgment on the accuracy. In particular, we chose to focus on 4 classifiers which are linear regression, logistic regression, decision tree, and random forest. Out of the 4 we have implemented logistic regression and decision tree from scratch and used libraries in sk-learn for the rest. The reason for implementing from the base was to test our knowledge in the chosen areas and demonstrate some of the learning outcomes of taking CS 365 with Professor Babis. The following sub-section aims to provide a brief description and relevance to the Titanic data set.

2.1 Linear Regression

Linear regression deals with fitting a linear equation to an observed data set. It is usually of the type $Y = aX + b$ where a is the slope of the line and b is the y-intercept. X is the independent variable, the predictor, and Y is the dependent variable or the thing you are trying to predict. Fitting an equation means that linear regression is best used in the case of continuous outcome variables. Thus, it does not very well suit the the binary outcomes of either 'Survived' or 'Not Survived' given a set of features X which we are trying to achieve with the Titanic classifier. However, initially, considerations were given to this method as its relevance to the class before reaching the conclusion of continuous vs binary results (the latter is the aim for TC).

2.2 Logistic Regression

Logistic Regression examples are a basic classifier that aligns with the task assigned of predicting a person dead or alive in the Titanic. There are two parts to logistic regression, maximizing and predicting. Maximizing is handled by taking into account all the features under consideration. Each feature is assigned a weight w_i and the goal is to find the weights that can closely model the behavior of the data. This is done by using the sigmoid function which is talked about more in the methods section of this paper. In short, for each observation you have you generate a value between [0 1] generated by the sigmoid function. Imagine you have a vector that has a sigmoid function for each person. Each sigmoid function has specified feature weights we are trying to find to maximize the true value of that person. The second part is the prediction, we set a threshold for example 0.5, and for any observations in the test set that give a value of < 0.5 when sigmoid is applied with the weights from the train set then we assign binary value 1.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Random Forest

In the context of our study on the Titanic dataset, the Decision Tree model was employed to explore its efficacy in predicting survival outcomes. This approach aligns with traditional uses of Decision Trees in classification tasks, where their ability to model non-linear decision boundaries makes them particularly useful for categorical outcomes.

Gradient Descent

Gradient Descent lies as an integral part of solving regression problems. It is used in our implementation of logistic regression to efficiently find the optimal parameter values (weights) that minimize the loss function, typically the log-likelihood. We first get the derivative of the loss function in terms of theta. Update the parameter theta in each iteration with theta = theta - alpha('L(theta)). This update reduces prediction error by moving towards the steepest descent direction.

3. Method

In this section, we detail the comprehensive methodology employed to predict survival outcomes for passengers of the Titanic dataset, utilizing a combination of data preprocessing, feature selection, and machine learning algorithms.

3.1 Heatmap and feature selection

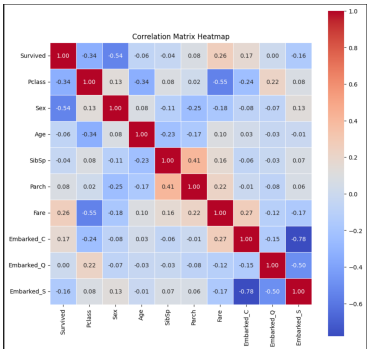

Figure 1: Heatmap of Titanic dataset.

To effectively select the most relevant features for our model, we have implemented a method to visualize the relationships between different variables within the Titanic dataset. This involves generating a heatmap, which is a powerful tool for identifying and displaying correlations across multiple variables.

The heatmap provides a color-coded representation of the correlation coefficients between all pairs of features. This visual aid is particularly valuable for pinpointing strong associations that can inform feature selection for predictive modeling. Our primary focus is on examining how each variable correlates with the 'Survived' outcome. Understanding these relationships is crucial as it influences the accuracy and performance of our predictive model.

Our analysis reveals significant correlations that align well with historical accounts of the Titanic tragedy. Notably, the 'Sex' and 'Pclass' (passenger class) variables show strong links with survival rates. These findings are consistent with documented reports that women and children, as well as passengers in higher classes, were given priority during lifeboat loading. By using the heatmap, we can substantiate these relationships with quantitative data, enhancing the historical accuracy and relevance of our model's feature selection process.

This methodological approach not only supports our feature selection but also ensures that our model is built on statistically significant and historically grounded variables, thereby improving its predictive reliability.

3.2 Preprocessing

We examine the data and find out some issues that could be improved. So we decided to do some preprocessing of the data.Here are some of the method we apply to enhance our overall accuracy.

Handling Missing Values: Missing data presents significant challenges in predictive modeling. We addressed this by imputing missing values in the 'Age' feature using the median age of the dataset, which provides a robust central tendency measure that is less sensitive to outliers than the mean. Additionally, we filled missing 'Embarked' values with 'S', the most frequent embarkation point, thereby minimizing its potential impact on model accuracy. Converting Categorical Variables to Numerical: Many machine learning algorithms require numerical input. Consequently, we transformed the 'Sex' feature from categorical ('male', 'female') into a binary numerical format (0, 1).

Experimentation with the 'Embarked' Feature: The 'Embarked' feature, indicating the passenger's port of embarkation, was initially converted into dummy variables ('Embarked_C', 'Embarked_Q', 'Embarked_S') to maintain the categorical nature without imposing ordinality. To explore the impact of different encoding strategies on our model's performance, we also experimented with directly mapping these categories to numerical values (C: 0, Q: 1, S: 2). This alternative method was assessed to determine whether a simpler numerical representation would yield accuracy improvements.

3.2.1 Exploring Encoding Methods for 'Embarked'

After converting the 'Embarked' feature into both dummy variables and direct numerical mappings, we closely examined the impact of these encoding strategies on the performance of our logistic regression model. Interestingly, both methods resulted in a similar accuracy of 0.84. This observation is somewhat unexpected as dummy variables are generally presumed to provide a more nuanced representation by avoiding arbitrary numerical relationships between categories. However, in this instance, the logistic regression model did not show a significant difference in performance between the two encoding approaches.

This could suggest that the 'Embarked' feature, regardless of the encoding method, may not have a strong differential impact on the survival prediction in the context of other more dominant features like 'Sex' or 'Pclass'. It is also possible that the inherent similarities between the ports with respect to passenger survival outcomes are such that they do not distinctly affect the model's ability to predict based on port of embarkation alone. Further analysis could involve examining interaction effects between 'Embarked' and other features to better understand this phenomenon.

3.3 Machine Learning Models

To deepen our understanding of the model training process and the underlying mathematical principles, we employed four distinct machine learning models: Linear Regression, Logistic Regression, Decision Tree, and Random Forest. Our approach was twofold. First, we implemented Logistic Regression and Decision Tree models from scratch. This exercise allowed us to gain a hands-on experience with the algorithms' intricacies and to appreciate the computational challenges involved in their practical applications.

Second, we compared our implementations with those provided by the widely-used scikit-learn library. This comparison was crucial for assessing the accuracy and efficiency of our models against established, optimized versions. It also provided valuable insights into potential discrepancies in performance, which could stem from differences in implementation details such as parameter settings, optimization techniques, or data handling

practices.

Through this comprehensive analysis, not only did we enhance our technical expertise, but we also gained practical perspectives on the trade-offs and decision-making processes involved in deploying machine learning algorithms. We are going to emphasize and provide details on the model we implement below.

3.3.1 Logistic Regression

Logistic Regression is used to predict a passenger's survival. We try to assign weight values that reflect the proportion of influence they have on the target. We have two main mathematical function that help in achieving the results: Sigmoid Function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where z is the linear combination of the features X weighted by the coefficients θ, expressed as $z = X\theta$. Cost Function:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log(\sigma(x^{(i)}\theta)) + (1 - y^{(i)}) \log(1 - \sigma(x^{(i)}\theta)) \right]$$

where $y^{(i)}$ is the actual label of the i-th sample, and $\sigma(x^{(i)}\theta)$ is the predicted probability that the outcome is 1 given the features $x^{(i)}$. Gradient of Cost Function (Important):

$$\nabla_\theta J(\theta) = \frac{1}{m} X^T (\sigma(X\theta) - y)$$

where X^T is the transpose of the feature matrix, and $\sigma(X\theta) - y$ represents the prediction errors.

We first get the features we are handling and pre-process them to be ready to deal with the function handling. For every observation (person) we get their sigmoid function with feature weights attached. Our goal is to get the value of these weights using gradient descent. For every iteration we compute the gradient and with a step sized defined and update our weights. By the end of our implementation, the updated weights will represent the proportion of influence over survival. We take the weights and test them on our test data set against a specified threshold, eg. 0.5.

3.3.2 Decision Tree

We implemented the Decision Tree algorithm, which recursively splits the training data into subsets based on an attribute that maximizes information gain, continuing until the maximum depth is reached or all data at a node belong to the same class.

Mathematical Background: The core concepts in our implementation are *entropy* and *information gain*. Entropy, which measures dataset impurity, is calculated as:

$$H(S) = -\sum_{i=1}^{c} p_i \log_2(p_i)$$

where p_i is the proportion of class i instances in dataset S. Information Gain is defined as the decrease in entropy after a dataset is split on an attribute:

$$IG(D, a) = H(D) - \left(\frac{|D_{left}|}{|D|} H(D_{left}) + \frac{|D_{right}|}{|D|} H(D_{right}) \right)$$

Implementation Details: Our implementation involves several key functions and classes:

- The `Node` class represents each tree node, capable of being a decision node or a leaf node.
- The `calculate_entropy` function computes the entropy at a node.
- The `best_split` function finds the split maximizing information gain.
- The `build_tree` function constructs the tree recursively.
- The `predict` function traverses the tree to return predictions based on the reached leaf node.
- The `decision_tree_predictions` function applies `predict` to generate predictions from validation data.

Results: Evaluating our Decision Tree on the validation dataset yielded a validation accuracy of:

Validation Accuracy: 0.84

4. Experimental results

Dataset Configuration and Rationale for Training-to-Test Ratio:

In configuring the dataset for our analysis, we opted for a training-to-test ratio of 90% to 10%. This decision was primarily influenced by the relatively small size of the Titanic dataset. Allocating a larger proportion to the training set maximizes the information available for the model to learn, which is especially crucial in smaller datasets where each data point is valuable. This approach aims to enhance the model's ability to generalize effectively in real-world scenarios, an essential factor for the successful deployment of machine learning models.

Feature Selection:

For feature selection, we chose `Pclass`, `Sex`, `Age`, `SibSp`, `Parch`, `Fare`, and `Embarked` based on their significant correlations observed in the previously generated heatmap. This decision ensures that our model leverages variables most relevant to predicting survival outcomes on the Titanic.

4.1 Logistic Regression

Results:

Our implementation resulted in an accuracy score of 83.3% with the sk-learn giving a score of 84.4%. We were off by 1%. The model considers an expanded feature using one-hot encoding casting the Embarked and Age Object types to boolean types so that we can process it using np arrays with no bias given towards order.

Current Status:

At this stage in our implementation, we were only able to guess the correct values for the learning rate and the iteration size. This could be further improved using the L1 and L2 regularization techniques which we intend to use in the future. Furthermore, more attention needs to be given to our pick of the features and the categorization of the Age into more relevant sized buckets.

4.2 Decision Tree

Results:

Decision Tree Performance: Our custom Decision Tree model achieved an accuracy of 0.822, which was notably higher than the 0.767 accuracy obtained with the scikit-learn Decision Tree model. This outcome prompts an intriguing consideration of the differences between our implementation and the conventional approach provided by scikit-learn.

Analysis of Performance Differences: Several factors may contribute to the superior performance of our custom model compared to the scikit-learn implementation:

- **Model Complexity:** Scikit-learn's Decision Tree models often incorporate more complex mechanisms for handling overfitting and outliers, such as pruning strategies and various parameters that can be adjusted for optimization. While these features are generally beneficial, they might lead to a model that is 'over-tuned' for a small dataset, potentially obscuring essential patterns in the data with overfitting controls.

- **Focus on Core Features:** Our simpler model may be better suited for small datasets like the Titanic because it inherently focuses on the most influential features without being distracted by noise or less relevant data points. This direct approach could help in capturing clearer signals from the core attributes that are most predictive of the outcome.

- **Custom Tailoring:** By designing and implementing our Decision Tree, we were able to tailor the model specifically to the characteristics of the Titanic dataset, possibly achieving a better fit than the more generic model provided by scikit-learn. This customization includes specific decisions on how to handle missing data, feature encoding, and splitting criteria, which are optimized based on our in-depth understanding of the dataset's nuances.

These factors suggest that in scenarios involving smaller or specific datasets, a customized approach can sometimes offer substantial advantages over a generalized tool, particularly in terms of model simplicity and focus.

4.3 Random Forest

As part of an experimental approach to assess different modeling techniques on the Titanic dataset, we utilized the `RandomForestClassifier` from scikit-learn. The Random Forest is an ensemble learning method that constructs a multitude of decision trees at training time and outputs the class that is the mode of the classes of the individual trees.

Model Training:

The Random Forest model was trained using the same features as the logistic regression model: `Pclass`, `Sex`, `Age`, `SibSp`, `Parch`, `Fare`, and `Embarked`. This approach enables direct comparison between the models.

Model Validation:

The Random Forest achieved a validation accuracy of 0.82, demonstrating its robust predictive capability, likely enhanced by the ensemble approach. This setup reduces the risk of overfitting and provides a more generalized solution.

This study's implementation of the Random Forest was experimental, aimed at assessing its potential effectiveness in comparison to simpler models.

4.4 Linear Regression

In our ongoing exploration of different predictive models for the Titanic dataset, we also applied Linear Regression, utilizing scikit-learn's `LinearRegression` class. This model, typically used for predicting continuous outcomes, was adapted to our classification task by applying a threshold to convert continuous outputs into binary predictions.

Model Training:
The Linear Regression model utilized the same features as the other models: `Pclass`, `Sex`, `Age`, `SibSp`, `Parch`, `Fare`, and `Embarked`, ensuring a fair comparison across models.

Model Validation:
After training, continuous outputs were dichotomized using a 0.5 threshold; values above were deemed '1' (survived), and below '0' (did not survive). This method resulted in a validation accuracy of 0.82, demonstrating the model's effective adaptation for binary classification.

Analysis of Performance:
The achieved accuracy of 0.82, comparable to more complex models like Random Forest and Decision Trees, highlights the efficacy of Linear Regression for classification tasks. This suggests that simpler models can rival more intricate ones under certain conditions, especially when linear relationships are prevalent.

This evaluation confirms Linear Regression's adaptability and potential as a binary classification tool in datasets with complex interdependencies, such as the Titanic data.

5. Conclusion

Throughout our analysis, we have employed several machine learning models to predict survival outcomes on the Titanic dataset, each revealing unique insights and demonstrating distinct capabilities. Our approach encompassed the implementation and validation of Logistic Regression, Decision Trees, Random Forest, and Linear Regression models.

Our experimental implementations of Decision Trees and custom Logistic Regression provided us with deep insights into the fundamental algorithms. Our Decision Tree model when compared to the scikit learn pacakge, constructed from scratch, achieved an unexpected accuracy, suggesting that simpler, less optimized models can sometimes outperform more complex algorithms on small datasets like the Titanic.

As expected, Logistic Regression derived the best results out of all the classifications. This was expected as we are trying to achieve binary results, either dead or alive, and logistic regression as explained previously handles learning from the the data to achieve the targets. Associating weights to the features accordingly.

In contrast, the use of Random Forest and Linear Regression, both implemented via scikit-learn, allowed us to explore the effectiveness of ensemble techniques and the adaptability of regression models to classification tasks. Remarkably, all models tested achieved a similar accuracy level around 0.82, reinforcing the concept that no single model universally outperforms others across all scenarios. Instead, the choice of model should be driven by the specific characteristics of the dataset, the complexity of the relationships within the data, and the computational resources available.

This project not only enhanced our understanding of various predictive modeling techniques but also underscored the importance of matching the model to the task at hand. As we move forward, these insights will guide our further explorations into machine learning applications, continuously refining our approach to data science challenges.

6. Resources

References

[1] "Decision Tree." *GeeksforGeeks*, GeeksforGeeks, https://www.geeksforgeeks.org/decision-tree/. Accessed 28 Apr. 2024.

[2] "ML | Logistic Regression Using Python." *GeeksforGeeks*, GeeksforGeeks, https://www.geeksforgeeks.org/ml-logistic-regression-using-python/. Accessed 28 Apr. 2024.

[3] "Logistic Regression." *Wikipedia*, Wikimedia Foundation, 23 Apr. 2024, https://en.wikipedia.org/wiki/Logistic_regression. Accessed 28 Apr. 2024.

[4] "ChatGPT 4.0." *Conversational AI provided by OpenAI*, 28 Apr. 2024, interaction via conversational interface.

[5] "Titanic: Machine Learning from Disaster." *Kaggle*, Kaggle Inc., https://www.kaggle.com/c/titanic. Accessed 28 Apr. 2024.

Machine Specs: CPU: AMB 7950 X3d 16 cores, GPU: RTX4090, RAM: 64 GB DDR5, PSU: 1200 Watt

Data Set: Kaggle Titanic Data set as formally introduced in the introduction and abstract.