**Titanic Survival Prediction Report**

**The Challenge**

The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered “unsinkable” RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren’t enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

In this challenge, we must build a predictive model that answers the question: “what sorts of people were more likely to survive?” using passenger data (ie name, age, gender, socio-economic class, etc).

**Problem statement:**

To improve passenger safety and experience in today's competitive environment, the travel and hospitality sector looks for creative solutions. Operating luxury cruise ships, our customer "SafeVoyage Cruises," is interested in using data science to forecast passenger survivability on their journeys. In an emergency, the goal is to maximize resource allocation while adding an extra layer of protection.

**Use Case:**

In order to predict the probability that a passenger would survive a maritime event, SafeVoyage Cruises uses the Titanic Survival Prediction model as an essential tool. The cruise ship may execute focused safety procedures and raise passenger safety levels by using the algorithm to forecast which passengers are most likely to survive by examining past data and passenger characteristics.

**Value Proposition:**

**Improved Safety Measures:** Value Proposition SafeVoyage Cruises will be able to anticipate emergencies and identify guests who could be more susceptible in advance thanks to the predictive technology. By doing this, the crew might perhaps avoid losses by giving aid to those who are more vulnerable priority.

**Resource Optimization:** There are limited resources, including lifeboats and emergency workers, in the case of an emergency evacuation. The cruise liner can optimize resource deployment and ensure a more efficient and effective reaction by precisely forecasting survival odds.

**Customer Loyalty and Trust**: SafeVoyage Cruises can promote the use of cutting-edge safety measures to demonstrate their dedication to the safety of their passengers. Thus, SafeVoyage becomes the preferred option for safety-conscious travelers on their maritime travels, thereby increasing consumer confidence and loyalty.

**Value to the Business:**

Being in the lead when it comes to passenger safety initiatives gives SafeVoyage Cruises a competitive edge. The business establishes itself as a pioneer in the sector by offering guests not only opulent experiences but also the greatest levels of safety through the use of the Titanic Survival Prediction methodology. This endeavour is consistent with the company's basic principles of quality, novelty, and client contentment.

**Data Overview:**

This competition has 2 files, the train and test csv files. I used the training data to build models and submitted my predictions on test data in the Kaggle website.

Dataset has total of **12** columns with names { **PassengerId, Survived, Pclass, Name, Sex, Age , SibSp, Parch, Ticket, Fare, Cabin, Embarked**}. Of which **Name, Sex, Ticket, Cabin and Embarked** are categorical.

This dataset has a total of **891** rows with missing values in some columns.

Here is the overview of data along with the descriptive stats of the numerical data columns:

A screenshot of a computer

Description automatically generated A screenshot of a computer screen

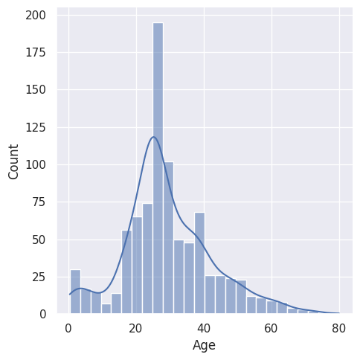
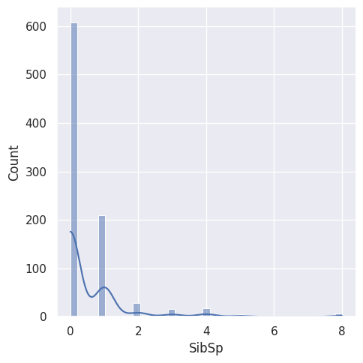
Description automatically generated

**Data Preprocessing:**

1. **EDA:**

After looking at the descriptive stats and missing values in the data. I tried observing the distributions and interactions between the variables in the data:

A graph with a line graph

Description automatically generated 

Looks like the distribution is right skewed for Parch and SibSp and slightly right skewed for Age distribution.

A graph of a number of people

Description automatically generated with medium confidence A graph of a survival count

Description automatically generated A diagram of a distribution of age based on survival

Description automatically generated

Majority of the survival percentage is from Pclass 3. Also, looks like survival percentage is high for male.

A close-up of numbers

Description automatically generated A screenshot of a graph

Description automatically generated

Looking at the Exbarked Survival percentage, port C has a greater survival percentage.

The fare is greater for Pclass 1 when compared to other classes.

A table with numbers and letters

Description automatically generated

On observing the correlation matrix, (Age, Pclass) and (Fare, Pclass) have a significant correlation so I tried including an interaction term during feature engineering.

1. **Data Cleaning/ Imputation:**

I filled the missing values in the embarked column with the mode and missing values in age column with mean grouped by the Pclass column.

I dropped the Passenger, Name, Ticket columns due to interpretation value.

I also dropped the Cabin, due to a lot of missing values and many unique values.

1. **Dummies:**

I used panda’s library’s inbuild getdummies function to generate dummies for my categorical data after data cleaning which includes the Pclass, Sex, Embarked columns. Also, the key thing here is that this does not generate dummy for the first value in a column.

1. **Standardization:**

I used the StandardScaler from the sklearn library to standardize the numerical data inorder to compare oranges with oranges and apples with oranges, leaving aside the dummies so that the categorical data do not lose its categorical significance.

1. **Feature Engineering:**

For the first analysis I tried including an additional feature which captures the interaction between Pclass and Fare

**Pclass\_fare\_interaction = Pclass \* Fare**

For later part of analysis to improve my model performance I introduced few more features:

**FamilySize = SibSp + Parch (This is to see the family size of individual )**

**IsAlsone = 1 if familysize = 0, else 0 (This to see if a member is alone or not)**

**Model Selection:**

**Cross Validation:** I did not split the dataset to evaluate my models because the data is small, instead I used cross validation with a fold of 5 and accuracy as the metric for all the models trained and evaluated the performance of the models based on the mean cv\_score.

**Note:** For Kaggle we cannot evaluate the performance of test data until we submit it on Kaggle, so this contains the evaluation using the cv\_score

**Models Used:**

**PartA:** I tried using some classifiers for initial data without any feature engineering.

1. Logistic Regression
2. Tuned the Logistic Regression using backward feature selection.
3. Random Forest Classifier (Hyperparameter tuning using GridSearch)
4. Gradient Boosting Classifier (With n\_estimators = 1000 and learning rate = 0.0099)
5. Linear Discriminant Analysis as a Classifier
6. XGBoosting ((With n\_estimators = 800 and learning rate = 0.01))
7. Stacking Classifier (Combination of logistic, random forest and gradient boosting)

**Results:**

A table with numbers and text

Description automatically generated

**PartB :** I tried using the following classifiers for the data after feature engineering and also I used MinMaxScaler for standardization here:

1. I used an iterative approach to evaluate the mean cv\_scores first on Logistic, XGBClassifier, Random Forest Classifier and Gradient Boosting Classifier
2. Stacked classifier using the best two models from the iterative approach.

**Results:**

A list of tasks with text

Description automatically generated with medium confidence

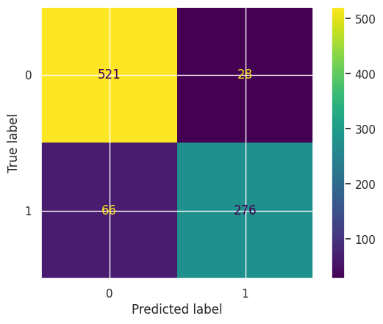
**Conclusion:**

Based on the mean cv scores after running all the models, all are close, so I submitted the XG Boosting and Gradient Boosting from Part B after predicting the values on the test dataset given.

I achieved a score of 79% which achieved a rank of 1919 out of 17k participants.

So, the best predictions I achieved was using the Gradient Boosting on the modified dataset using additional features and below is its performance on training data.

A graph showing a positive rate

Description automatically generated with medium confidence

**Code Link:**

https://colab.research.google.com/drive/1fQppzfDVusgSJVUJVsxXTtM6ozgfeRYC?usp=sharing