List of all libraries used throughout:

pandas, numpy, matplotlib.pyplot, seaborn, warnings (to ignore warnings), statsmodels.api, linear_model from sklearn, mean_squared_error from sklearn.metrics, cross_validate from sklearn.model_selection, StandardScaler from sklearn.preprocessing, RandomForestClassifier from sklearn.ensemble, confusion_matrix, accuracy_score, plot_confusion_matrix and classification_report from sklearn.metrics, GridSearchCV from sklearn.model_selection. I imported them at the beginning in my notebook.

APPROACH:

I imported the titanic datasets (train.csv and test.csv) from the Kaggle website using *pandas.*

Exploratory Data Analysis

The training dataset has 891 rows and 12 columns while the test data has 418 rows and 11 columns. There are 7 numerical features and 5 categorical features in the training dataset while the test data has 6 numerical features and 5 categorical features since the test dataset has no Survived column which is the target variable. I checked for null values and the table below shows the columns that have null values in both datasets:

	Age	Cabin	Embarked	Fare
Training Data	177	687	2	0
Test Data	86	327	0	1

From the data, 549 passengers did not survive while 342 of the passengers survive. I did some exploratory data analysis by checking the survival probability by gender and plotted a var graph to show the results. The results showed that females, have a higher survival probability of 0.727468 which implies a 7 out of 10 survival probability while for male, the survival probability is 0.190985 which is just below 2 out of 10. This means that a female passenger is a lot more likely to survive than a male passenger. I also checked survival probabilities by Pclass (passenger class), Parch (number of parents and children).

Data Manipulation

1. I dropped some columns which I deemed not to be very useful in the prediction such as PassengerId, Name, Cabin and Ticket. I did this for both the training and test datasets.
2. I filled the missing values in the Age column by the mean of the column for both datasets
3. I created age categories as shown below and created a new column for it called Age_group:

Less than 13	**13-18**	**19-25**	**26-59**	**60 and above**
Children	Adolescence	Youth	Adult	Seniors

4. I filled the empty rows in the Embarked variable of the training data with the mode of the column i.e., the letter that appears the most which was S.
5. For the 1 empty row in the Fare column for the test data, I got the row and saw that the passenger was in Pclass 3. So, I filled that value with the mean of the fares from all the passengers in Pclass 3 which was 12.45967788.

6. After this, all the empty rows in the dataset were filled.

Converting Categorical Variable to Numerical

1. Since all the variables have equal importance, I used one hot encoding with *get_dummies()* instead of label encoding to convert the categorical variables to numerical in the Sex column
2. For the Embarked and Age_group columns, I used *get_dummies()* to create new columns, same as the number of unique values in the column with 1s where there is an occurrence and 0s where there is not.

I did the correlation heatmap to see how each feature correlates with the target variable (Survived). From analyzing the correlation values, I dropped the Age, Embarked_Q and Age_group_Adult columns. I separated the predictors from the target variables and used standard scaler to scale the data to a reasonable range to reduce the potential for bias because the Fare column is significantly higher than the other features.

Modeling and Prediction

I made use of *RandomForestClassifier* for modeling since it is a classification task and plotted the confusion matrix and printed the classification report as well. I then used *crossvalidate* to get the precision, recall, F1 and accuracy scores.

I used a list of trees to get the optimal number of trees with *n_estimators* and plotted the graph of MSE against the number of trees to identify the best performing number of trees which was 120. I made use of GridSearch to get the best performing hyperparameters. The modeling took a long time because GridSearch had to do 24000 fits by doing a combination of all the hyperparameters and got the following:

- n_estimators (number of trees in the forest) = 20,
- max_depth (the maximum depth of the tree) = 12,
- *criterion* (measures the quality of a split) = "entropy",
- *max_leaf_nodes* (number of leaf nodes) = 32,
- *bootstrap* (if bootstrap samples should be used to build the tree) = False and
- *ccp_alpha* (the complexity parameter for pruning the tree) = 0.004.

I used the best hyperparameters from the GridSearch and used them to build the Random Forest model again and used cross validate to get the accuracy score which was 0.82493.

I imported the test data again to get the PassengerId to be put in the submission file and then used my new model to predict on the test data and saved the prediction into a csv file along with the PassengerId. I submitted on Kaggle and got a score of 0.77990.