titanic_survival_exploration

August 7, 2018

0.1 Introduction and Foundations

0.2 Project: Titanic Survival Exploration

In 1912, the ship RMS Titanic struck an iceberg on its maiden voyage and sank, resulting in the deaths of most of its passengers and crew. In this project, we will explore a subset of the RMS Titanic passenger manifest to determine which features best predict whether someone survived or did not survive.

In [1]:
```
# Import libraries necessary for this project
import numpy as np
import pandas as pd
from IPython.display import display # Allows the use of display() for DataFrames

# Import supplementary visualizations code visuals.py
import visuals as vs

# Pretty display for notebooks
%matplotlib inline

# Load the dataset
in_file = 'titanic_data.csv'
full_data = pd.read_csv(in_file)

# Print the first few entries of the RMS Titanic data
display(full_data.head())
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
```

```
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                            Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

From a sample of the RMS Titanic data, we can see the various features present for each passenger on the ship: - **Survived**: Outcome of survival (0 = No; 1 = Yes) - **Pclass**: Socio-economic class (1 = Upper class; 2 = Middle class; 3 = Lower class) - **Name**: Name of passenger - **Sex**: Sex of the passenger - **Age**: Age of the passenger (Some entries contain `NaN`) - **SibSp**: Number of siblings and spouses of the passenger aboard - **Parch**: Number of parents and children of the passenger aboard - **Ticket**: Ticket number of the passenger - **Fare**: Fare paid by the passenger - **Cabin** Cabin number of the passenger (Some entries contain `NaN`) - **Embarked**: Port of embarkation of the passenger (C = Cherbourg; Q = Queenstown; S = Southampton)

Since we're interested in the outcome of survival for each passenger or crew member, we can remove the **Survived** feature from this dataset and store it as its own separate variable `outcomes`. We will use these outcomes as our prediction targets.
Run the code cell below to remove **Survived** as a feature of the dataset and store it in `outcomes`.

In [2]:
```
# Store the 'Survived' feature in a new variable and remove it from the dataset
outcomes = full_data['Survived']
data = full_data.drop('Survived', axis = 1)

# Show the new dataset with 'Survived' removed
display(data.head())
```

```
   PassengerId  Pclass                                                Name  \
0            1       3                             Braund, Mr. Owen Harris
1            2       1   Cumings, Mrs. John Bradley (Florence Briggs Th...
2            3       3                              Heikkinen, Miss. Laina
3            4       1       Futrelle, Mrs. Jacques Heath (Lily May Peel)
4            5       3                            Allen, Mr. William Henry

      Sex   Age  SibSp  Parch            Ticket     Fare Cabin Embarked
0    male  22.0      1      0         A/5 21171   7.2500   NaN        S
1  female  38.0      1      0          PC 17599  71.2833   C85        C
2  female  26.0      0      0  STON/O2. 3101282   7.9250   NaN        S
3  female  35.0      1      0            113803  53.1000  C123        S
4    male  35.0      0      0            373450   8.0500   NaN        S
```

The very same sample of the RMS Titanic data now shows the **Survived** feature removed from the DataFrame. Note that `data` (the passenger data) and `outcomes` (the outcomes of survival) are now *paired*. That means for any passenger `data.loc[i]`, they have the survival outcome `outcomes[i]`.

To measure the performance of our predictions, we need a metric to score our predictions against the true outcomes of survival. Since we are interested in how *accurate* our predictions are, we will calculate the proportion of passengers where our prediction of their survival is correct.

```
In [4]: def accuracy_score(truth, pred):
            """ Returns accuracy score for input truth and predictions. """

            # Ensure that the number of predictions matches number of outcomes
            if len(truth) == len(pred):

                # Calculate and return the accuracy as a percent
                return "Predictions have an accuracy of {:.2f}%.".format((truth == pred).mean()*

            else:
                return "Number of predictions does not match number of outcomes!"

        # Test the 'accuracy_score' function
        predictions = pd.Series(np.ones(5, dtype = int))
        print(accuracy_score(outcomes[:5], predictions))
```

```
Predictions have an accuracy of 60.00%.
```

1 Making Predictions

If we were asked to make a prediction about any passenger aboard the RMS Titanic whom we knew nothing about, then the best prediction we could make would be that they did not survive. This is because we can assume that a majority of the passengers (more than 50%) did not survive the ship sinking.
The `predictions_0` function below will always predict that a passenger did not survive.

```
In [7]: def predictions_0(data):
            """ Model with no features. Always predicts a passenger did not survive. """

            predictions = []
            for _, passenger in data.iterrows():

                # Predict the survival of 'passenger'
                predictions.append(0)

            # Return our predictions
            return pd.Series(predictions)

        # Make the predictions
        predictions = predictions_0(data)
```

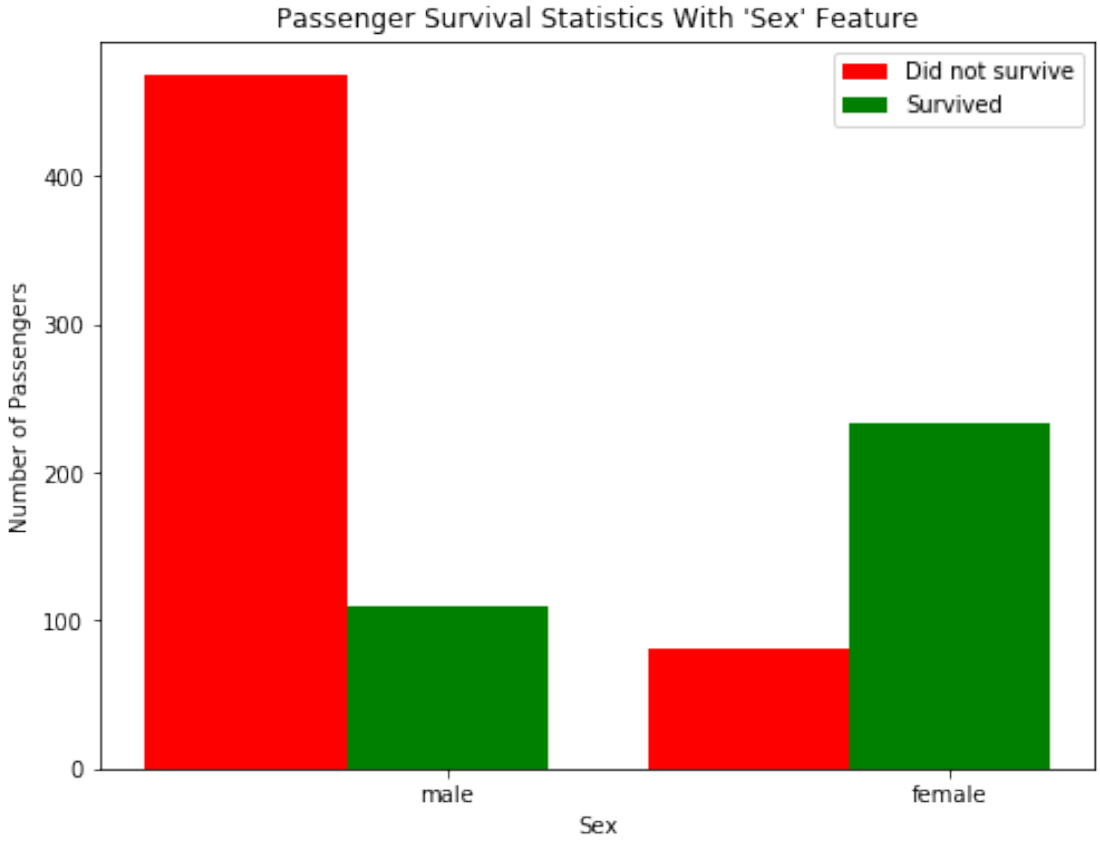
```
In [8]: print(accuracy_score(outcomes, predictions))
```

```
Predictions have an accuracy of 61.62%.
```

Answer: *61.62%*

Let's take a look at whether the feature **Sex** has any indication of survival rates among passengers using the `survival_stats` function. This function is defined in the `visuals.py` Python script included with this project. The first two parameters passed to the function are the RMS Titanic data and passenger survival outcomes, respectively. The third parameter indicates which feature we want to plot survival statistics across.
Run the code cell below to plot the survival outcomes of passengers based on their sex.

```
In [9]: vs.survival_stats(data, outcomes, 'Sex')
```



Examining the survival statistics, a large majority of males did not survive the ship sinking. However, a majority of females *did* survive the ship sinking. Let's build on our previous prediction: If a passenger was female, then we will predict that they survived. Otherwise, we will predict the passenger did not survive.

```
In [10]: def predictions_1(data):
             """ Model with one feature:
                     - Predict a passenger survived if they are female. """

             predictions = []
             for _, passenger in data.iterrows():

                 # Remove the 'pass' statement below
                 # and write your prediction conditions here
                 if(passenger['Sex'] == 'female'):
                     predictions.append(1)
                 else:
                     predictions.append(0)

             # Return our predictions
             return pd.Series(predictions)

         # Make the predictions
         predictions = predictions_1(data)
```

- How accurate would a prediction be that all female passengers survived and the remaining passengers did not survive?

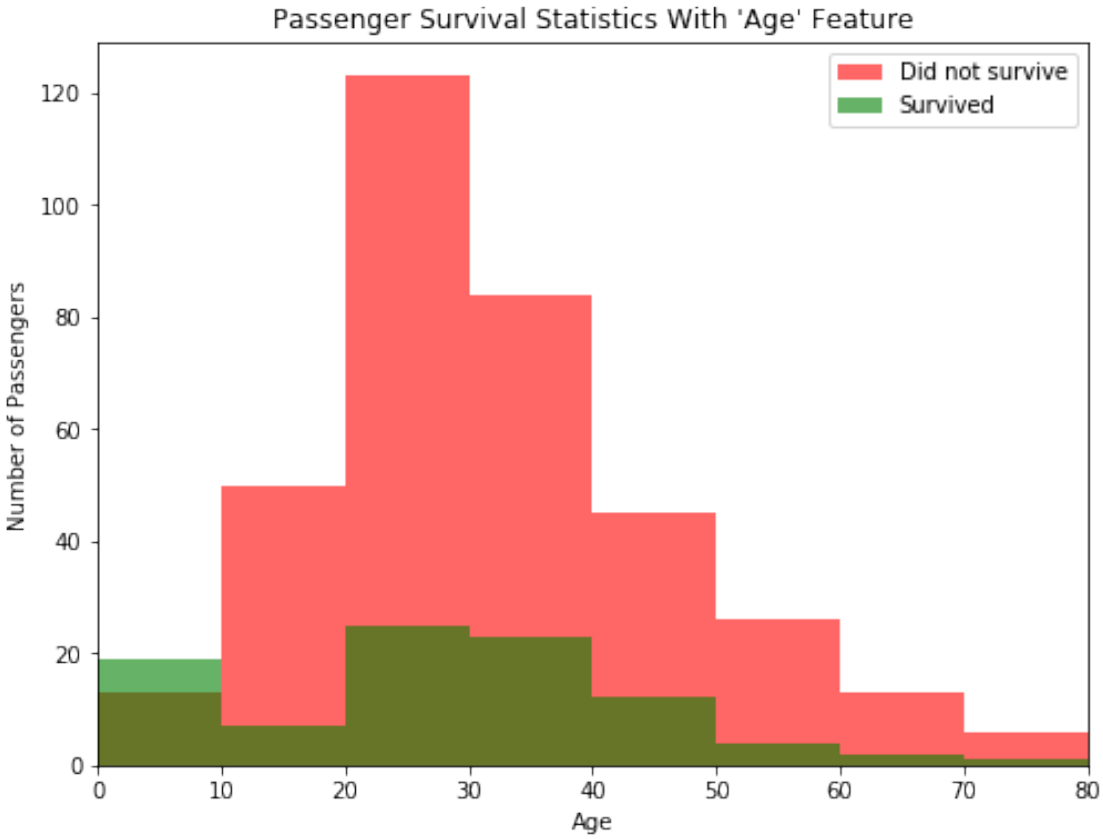
```
In [11]: print(accuracy_score(outcomes, predictions))
```

```
Predictions have an accuracy of 78.68%.
```

Answer: *78.68%*

Using just the **Sex** feature for each passenger, we are able to increase the accuracy of our predictions by a significant margin. Now, let's consider using an additional feature to see if we can further improve our predictions. For example, consider all of the male passengers aboard the RMS Titanic: Can we find a subset of those passengers that had a higher rate of survival? Let's start by looking at the **Age** of each male, by again using the `survival_stats` function. This time, we'll use a fourth parameter to filter out the data so that only passengers with the **Sex** 'male' will be included.
Run the code cell below to plot the survival outcomes of male passengers based on their age.

```
In [12]: vs.survival_stats(data, outcomes, 'Age', ["Sex == 'male'"])
```



Examining the survival statistics, the majority of males younger than 10 survived the ship sinking, whereas most males age 10 or older *did not survive* the ship sinking. Let's continue to build on our previous prediction: If a passenger was female, then we will predict they survive. If a passenger was male and younger than 10, then we will also predict they survive. Otherwise, we will predict they do not survive.

```
In [13]: def predictions_2(data):
             """ Model with two features:
                     - Predict a passenger survived if they are female.
                     - Predict a passenger survived if they are male and younger than 10. """

             predictions = []
             for _, passenger in data.iterrows():

                 if(passenger['Sex'] == 'female'):
                     predictions.append(1)
                 elif((passenger['Sex'] == 'male') and (passenger['Age'] < 10)):
                     predictions.append(1)
                 else:
                     predictions.append(0)
```

```
        # Return our predictions
        return pd.Series(predictions)

    # Make the predictions
    predictions = predictions_2(data)
```

- How accurate would a prediction be that all female passengers and all male passengers younger than 10 survived?

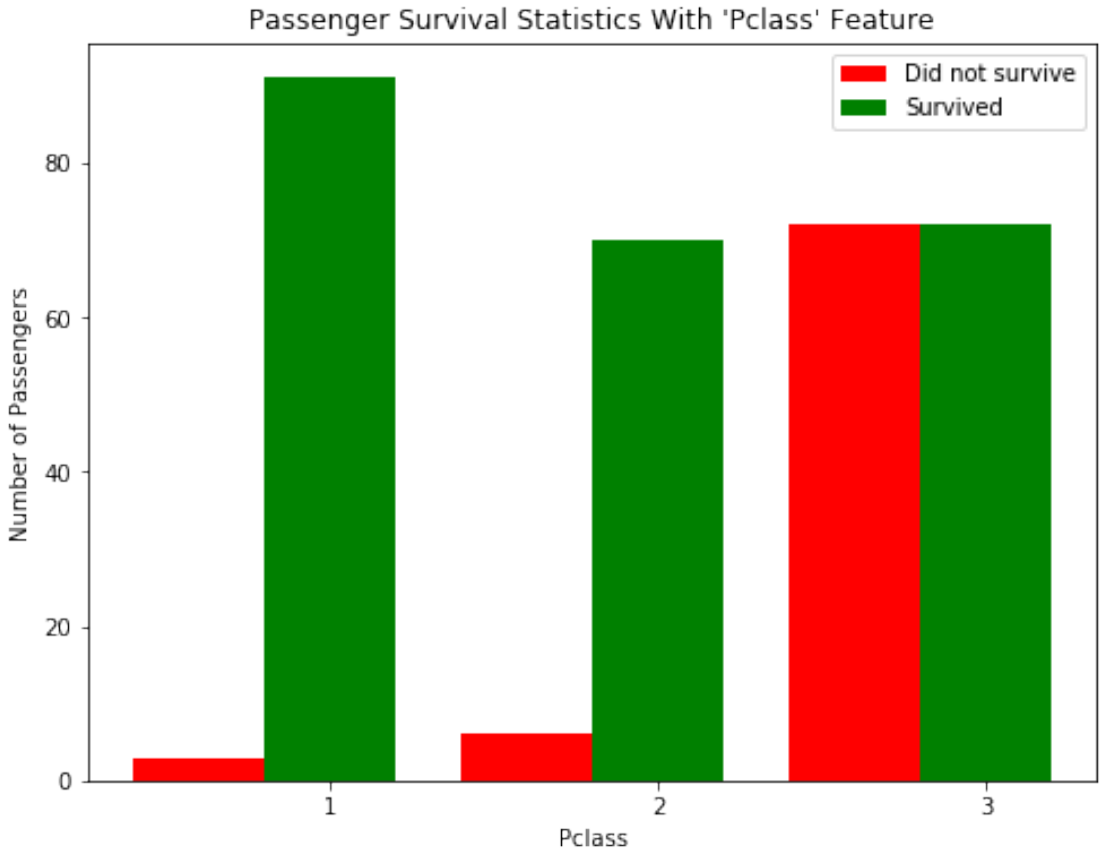
```
In [14]: print(accuracy_score(outcomes, predictions))
```

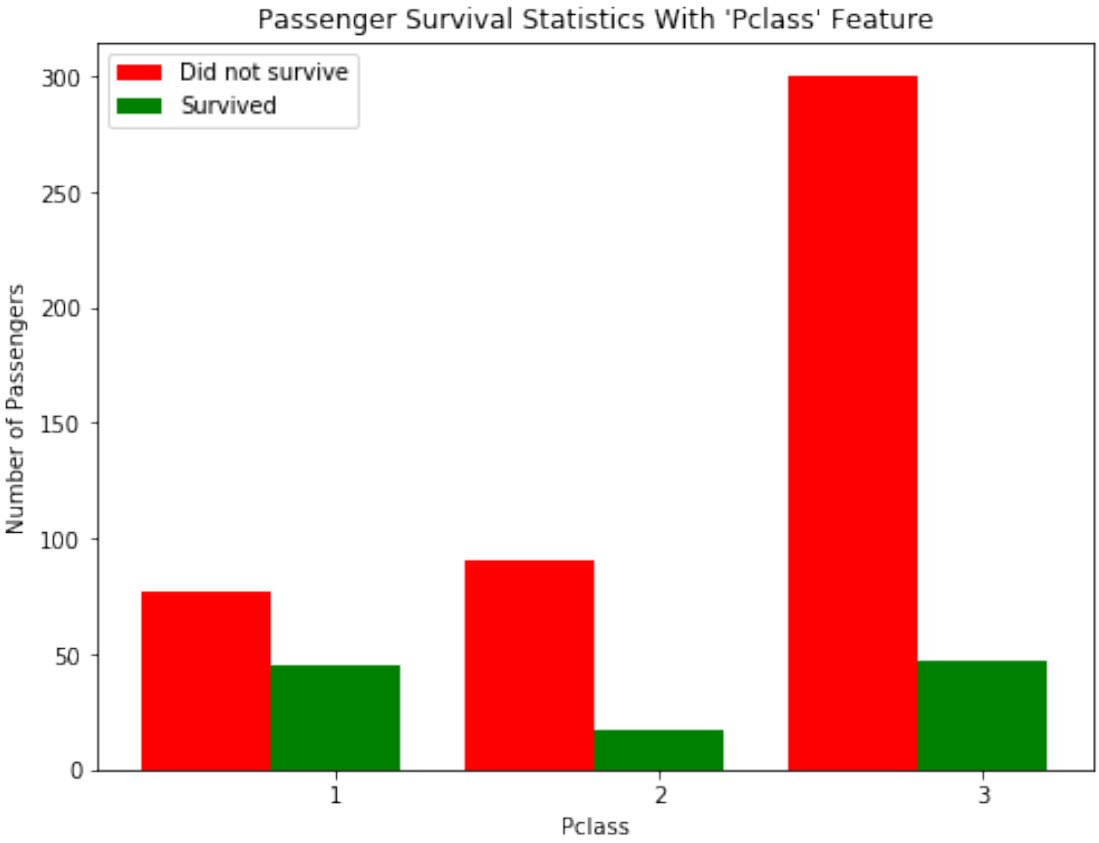
```
Predictions have an accuracy of 79.35%.
```

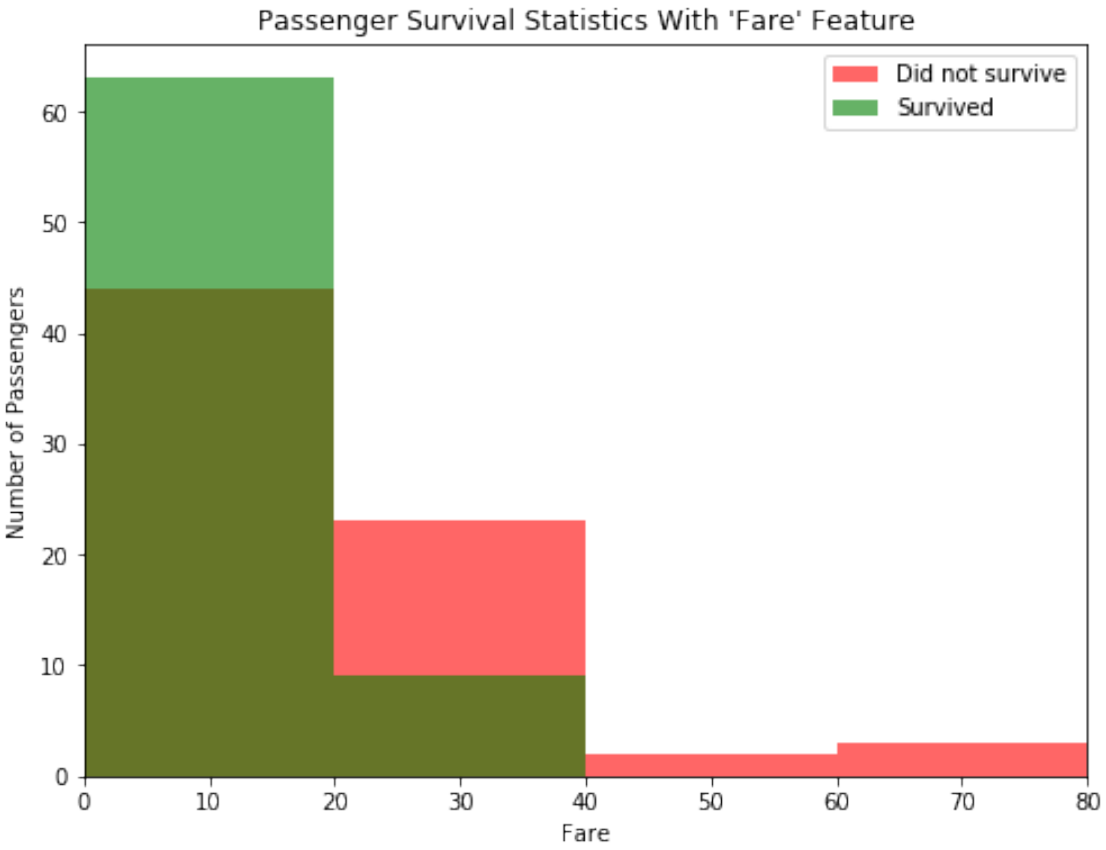
Answer: *79.35%*

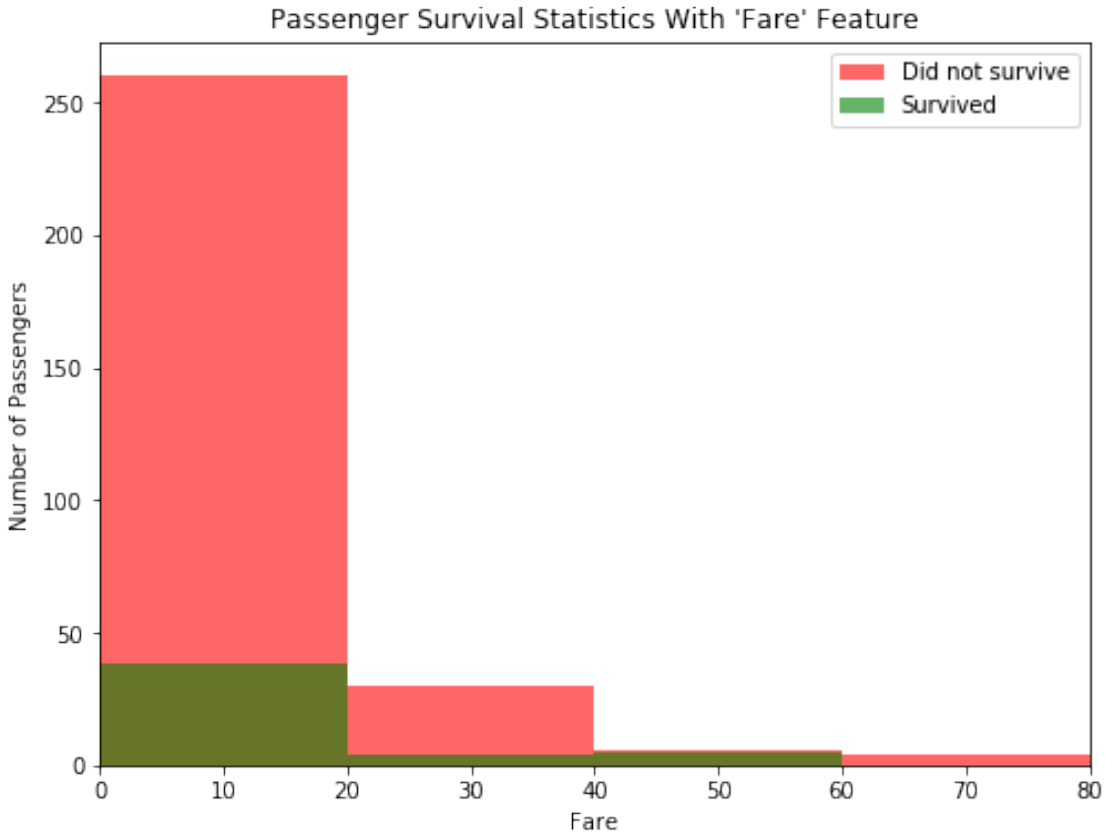
Adding the feature **Age** as a condition in conjunction with **Sex** improves the accuracy by a small margin more than with simply using the feature **Sex** alone. We want to find a series of features and conditions to split the data on to obtain an outcome prediction accuracy of at least 80%. This may require multiple features and multiple levels of conditional statements to succeed. **Pclass**, **Sex**, **Age**, **SibSp**, and **Parch** are some suggested features to try.

```
In [15]: vs.survival_stats(data, outcomes, 'Pclass', ["Sex == 'female'"])
         vs.survival_stats(data, outcomes, 'Pclass', ["Sex == 'male'"])
         vs.survival_stats(data, outcomes, 'Fare', ["Sex == 'female'", "Pclass == 3"])
         vs.survival_stats(data, outcomes, 'Fare', ["Sex == 'male'","Pclass == 3"])
```


Passenger Survival Statistics With 'Pclass' Feature
Did not survive
Survived
Number of Passengers
0
20
40
60
80
1
2
3
Pclass


Passenger Survival Statistics With 'Pclass' Feature
Did not survive
Survived
Number of Passengers
0
50
100
150
200
250
300
1
2
3
Pclass


Passenger Survival Statistics With 'Fare' Feature
Did not survive
Survived
Number of Passengers
Fare
0
10
20
30
40
50
60
70
80



After exploring the survival statistics visualization,we will code a function so that it will make your prediction.
We will make sure to keep track of the various features and conditions we tried before arriving at our final prediction model.

```
In [16]: def predictions_3(data):
             """ Model with multiple features. Makes a prediction with an accuracy of at least 8

             predictions = []
             np.random.seed(11)
             for _, passenger in data.iterrows():

                 if(passenger['Sex'] == 'female'):
                     if((passenger['Pclass'] == 1) or (passenger['Pclass'] == 2)):
                         predictions.append(1)
                     else:
                         if((passenger['Fare'] > 0) and (passenger['Fare'] < 20)):
                             predictions.append(1)
                         elif((passenger['Fare'] > 20) and (passenger['Fare'] < 40)):
                             predictions.append(np.random.randint(0,2))
                         else:
```

```
                predictions.append(0)

            elif((passenger['Sex'] == 'male') and (passenger['Age'] < 10)):
                predictions.append(1)
            elif((passenger['Sex'] == 'male') and (passenger['Pclass'] == 1)):
                if(passenger['Fare'] > 80 and passenger['Fare'] < 100):
                    predictions.append(1)
                elif(passenger['Fare'] > 120 and passenger['Fare'] < 200):
                    predictions.append(1)
                elif(passenger['Fare'] > 300):
                    predictions.append(1)
                else:
                    predictions.append(0)
            else:
                predictions.append(0)

        # Return our predictions
        return pd.Series(predictions)

    # Make the predictions
    predictions = predictions_3(data)
```

```
In [17]: print(accuracy_score(outcomes, predictions))
```

```
Predictions have an accuracy of 80.70%.
```

Answer: *80.70%*

2 Conclusion

After several iterations of exploring and conditioning on the data, we have built a useful algorithm for predicting the survival of each passenger aboard the RMS Titanic. The technique applied in this project is a manual implementation of a simple machine learning model, the *decision tree*. A decision tree splits a set of data into smaller and smaller groups (called *nodes*), by one feature at a time. Each time a subset of the data is split, our predictions become more accurate if each of the resulting subgroups are more homogeneous (contain similar labels) than before. The advantage of having a computer do things for us is that it will be more exhaustive and more precise than our manual exploration above. This link provides another introduction into machine learning using a decision tree.

A decision tree is just one of many models that come from *supervised learning*. In supervised learning, we attempt to use features of the data to predict or model things with objective outcome labels. That is to say, each of our data points has a known outcome value, such as a categorical, discrete label like `'Survived'`, or a numerical, continuous value like predicting the price of a house.