Technological Institute of the Philippines	Quezon City - Computer Engineering
Course Code:	CPE 019
Code Title:	Emerging Technologies in CpE 2
Summer Semester	AY 2024-2024

Hands-on Activity 4.1: Advanced Data Analytics and Machine Learning

Name	Ramos, William Laurence M.
Section	CPE32S1
Date Performed:	06/20/24
Date Submitted:	06/20/24
Instructor:	Engr. Roman Richard

PART 1: Do the following objectives:

Part 1: Import the Libraries and Data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

```
titanicFrame = pd.read_csv('/content/titanic_test.csv')
titanicFrame = pd.read_csv('/content/titanic_train.csv')
titanicall = pd.read_csv('/content/titanic_all.csv', delimiter=';')
```

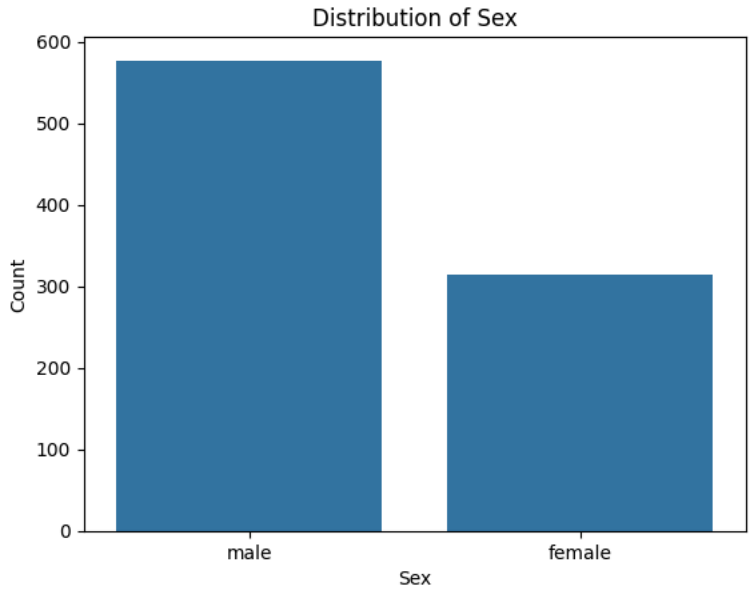
```
print (titanicFrame)
print (titanicFrame2)
```

```
4              5         0       3
..           ...       ...     ...
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         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
..                                                ...     ...   ...    ...
886                            Montvila, Rev. Juozas    male  27.0      0
887                     Graham, Miss. Margaret Edith  female  19.0      0
888         Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                            Behr, Mr. Karl Howell    male  26.0      0
890                              Dooley, Mr. Patrick    male  32.0      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
..     ...               ...      ...   ...      ...
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
     PassengerId  Pclass                                 Name  Sex  \
0            892       3                    Kelly, Mr. James    0
```

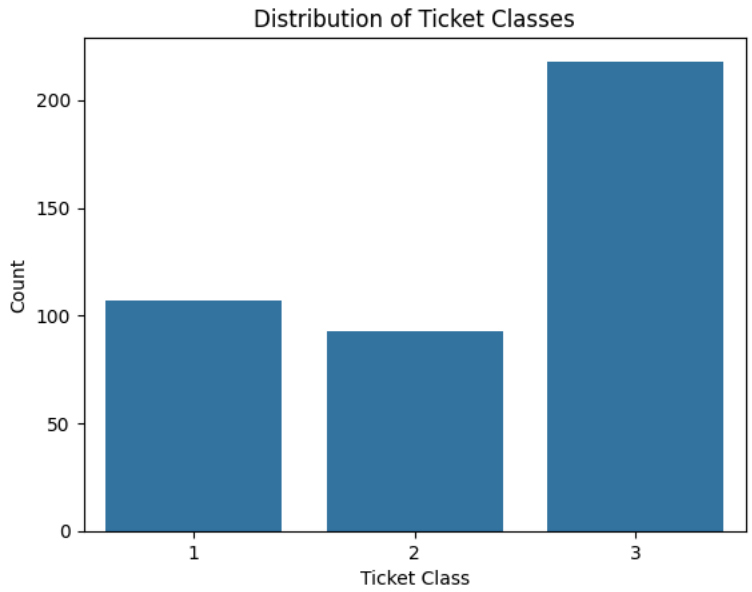
```
414         1306       1                Oliva y Ocana, Dona. Fermina    1
415         1307       3                Saether, Mr. Simon Sivertsen    0
416         1308       3                        Ware, Mr. Frederick    0
417         1309       3                    Peter, Master. Michael J    0

      Age  SibSp  Parch              Ticket      Fare Cabin Embarked
0    34.5      0      0              330911    7.8292   NaN        Q
1    47.0      1      0              363272    7.0000   NaN        S
2    62.0      0      0              240276    9.6875   NaN        Q
3    27.0      0      0              315154    8.6625   NaN        S
4    22.0      1      1             3101298   12.2875   NaN        S
..    ...    ...    ...                 ...       ...   ...      ...
413   NaN      0      0          A.5. 3236    8.0500   NaN        S
414  39.0      0      0           PC 17758  108.9000  C105        C
415  38.5      0      0  SOTON/O.Q. 3101262    7.2500   NaN        S
```

Part 2: Plot the Data

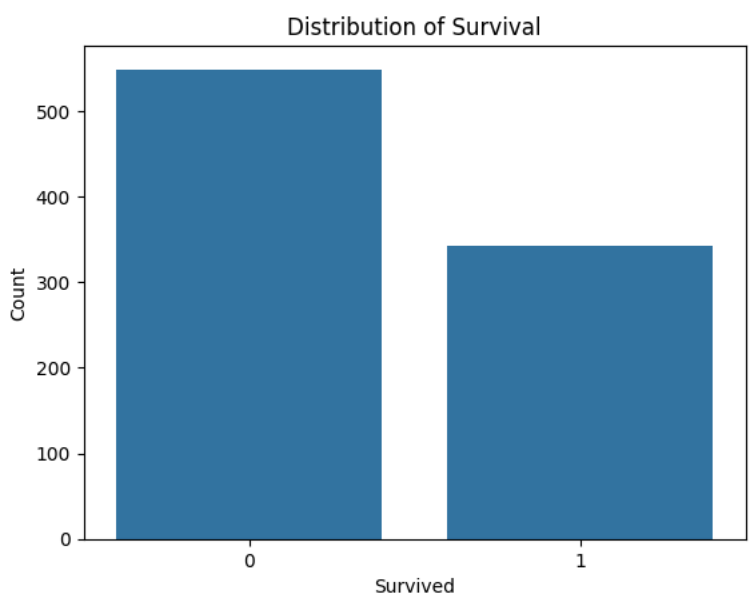
```
sns.countplot(data=titanicFrame, x='Sex')
plt.xlabel('Sex')
plt.ylabel('Count')
plt.title('Distribution of Sex')
plt.show()
```



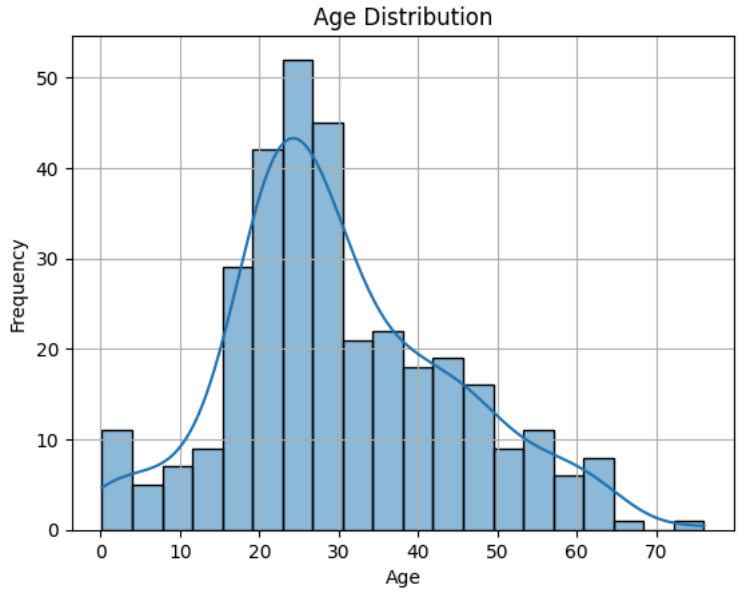

```
sns.countplot(data=titanicFrame2, x='Pclass')
plt.xlabel('Ticket Class')
plt.ylabel('Count')
plt.title('Distribution of Ticket Classes')
plt.show()
```




```
sns.countplot(data=titanicFrame, x='Survived')
plt.xlabel('Survived')
plt.ylabel('Count')
plt.title('Distribution of Survival')
plt.show()
```




```
sns.histplot(titanicFrame2['Age'], bins=20, kde=True)
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Age Distribution')
plt.grid(True)
plt.show()
```

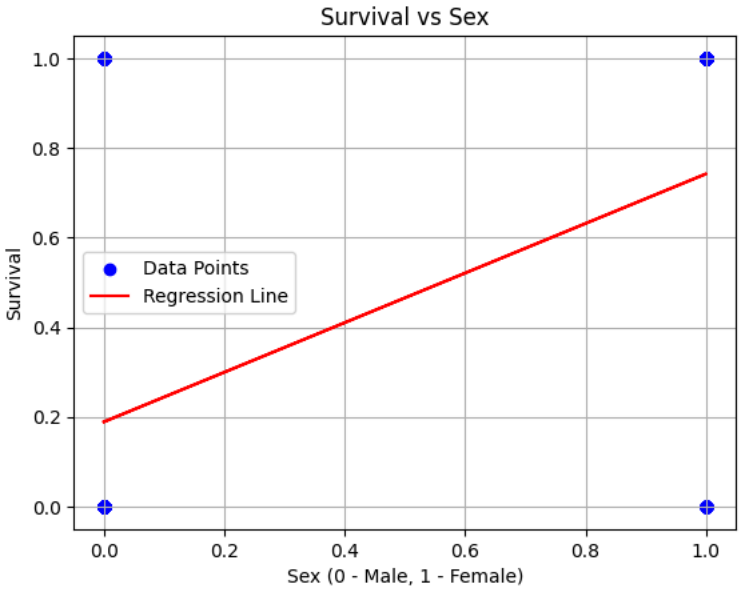


Part 3: Perform Simple Linear Regression on the SURVIVAL feature column (you can check the internet on how you can perform simple linear regression)

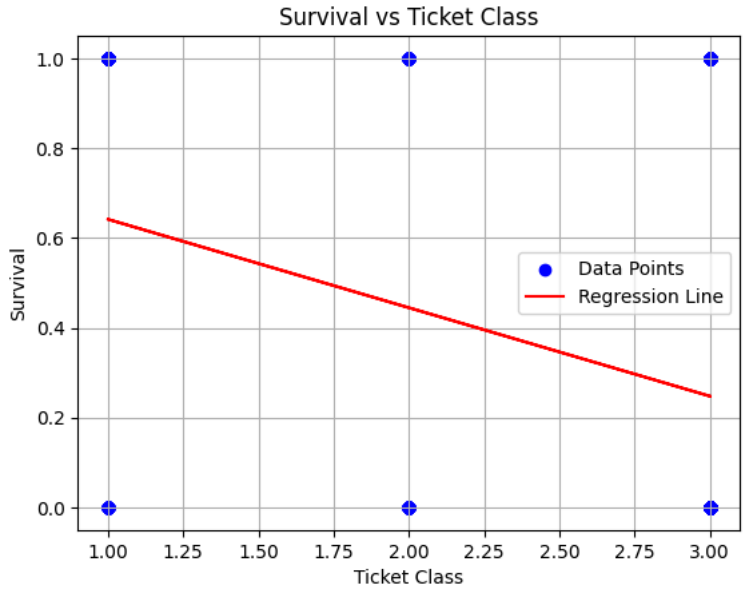
```
print (titanicFrame2)
```

```
     PassengerId  Pclass                                          Name  Sex  \
0            892       3                              Kelly, Mr. James  NaN
1            893       3              Wilkes, Mrs. James (Ellen Needs)  NaN
2            894       2                     Myles, Mr. Thomas Francis  NaN
3            895       3                              Wirz, Mr. Albert  NaN
4            896       3  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  NaN
..           ...     ...                                           ...  ...
413         1305       3                            Spector, Mr. Woolf  NaN
414         1306       1                  Oliva y Ocana, Dona. Fermina  NaN
415         1307       3                  Saether, Mr. Simon Sivertsen  NaN
416         1308       3                           Ware, Mr. Frederick  NaN
417         1309       3                      Peter, Master. Michael J  NaN

      Age  SibSp  Parch              Ticket      Fare Cabin Embarked
0    34.5      0      0              330911    7.8292   NaN        Q
1    47.0      1      0              363272    7.0000   NaN        S
2    62.0      0      0              240276    9.6875   NaN        Q
3    27.0      0      0              315154    8.6625   NaN        S
4    22.0      1      1             3101298   12.2875   NaN        S
..    ...    ...    ...                 ...       ...   ...      ...
413   NaN      0      0           A.5. 3236    8.0500   NaN        S
414  39.0      0      0            PC 17758  108.9000  C105        C
415  38.5      0      0  SOTON/O.Q. 3101262    7.2500   NaN        S
416   NaN      0      0              359309    8.0500   NaN        S
417   NaN      1      1                2668   22.3583   NaN        C

[418 rows x 11 columns]
```

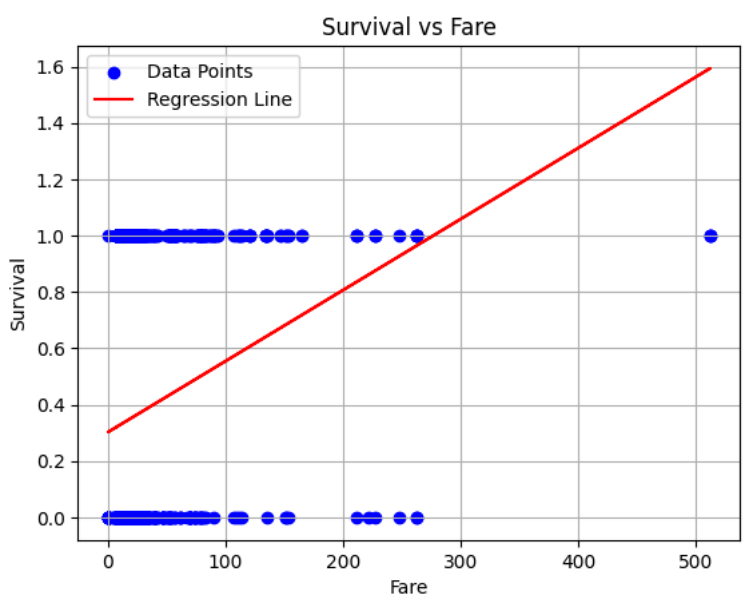
```
titanicFrame['Sex'] = titanicFrame['Sex'].map({'male': 0, 'female': 1})
x = titanicFrame['Sex']
y = titanicFrame['Survived']

# coefficients of the regression line
coefficients = np.polyfit(x, y, 1)
m, b = coefficients
regression_line = m * x + b

# Plotting
plt.scatter(x, y, color='blue', label='Data Points')
plt.plot(x, regression_line, color='red', label='Regression Line')
plt.xlabel("Sex (0 - Male, 1 - Female)")
plt.ylabel("Survival")
plt.title('Survival vs Sex')
plt.legend()
plt.grid(True)
plt.show()
```



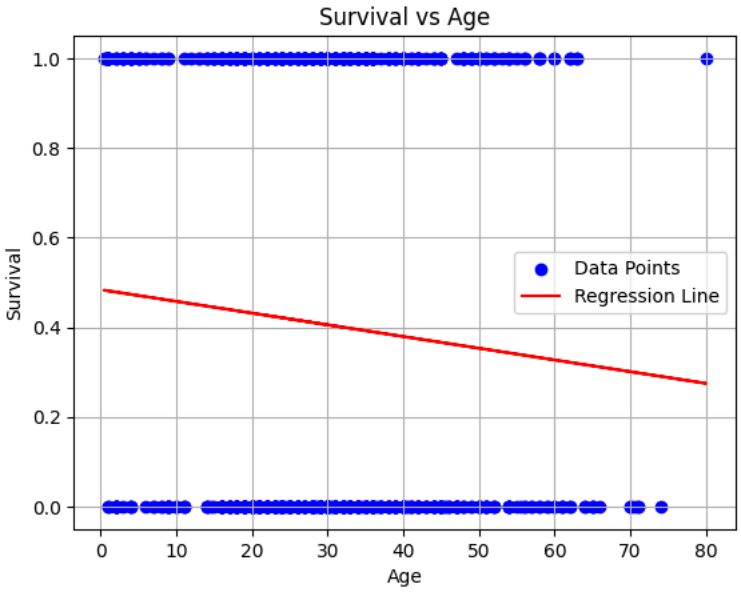

```
x = titanicFrame['Pclass']
y = titanicFrame['Survived']

# coefficients of the regression line
coefficients = np.polyfit(x, y, 1)
m, b = coefficients
regression_line = m * x + b

# Plotting
plt.scatter(x, y, color='blue', label='Data Points')
plt.plot(x, regression_line, color='red', label='Regression Line')
plt.xlabel("Ticket Class")
plt.ylabel("Survival")
plt.title('Survival vs Ticket Class')
plt.legend()
plt.grid(True)
plt.show()
```



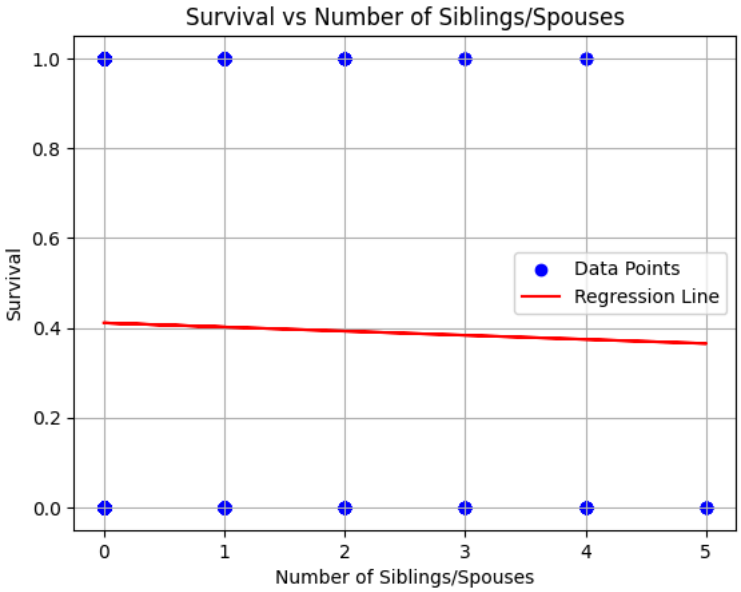

```
x = titanicFrame['Fare']
y = titanicFrame['Survived']

# coefficients of the regression line
coefficients = np.polyfit(x, y, 1)
m, b = coefficients
regression_line = m * x + b

# Plotting
plt.scatter(x, y, color='blue', label='Data Points')
plt.plot(x, regression_line, color='red', label='Regression Line')
plt.xlabel("Fare")
plt.ylabel("Survival")
plt.title('Survival vs Fare')
plt.legend()
plt.grid(True)
plt.show()
```



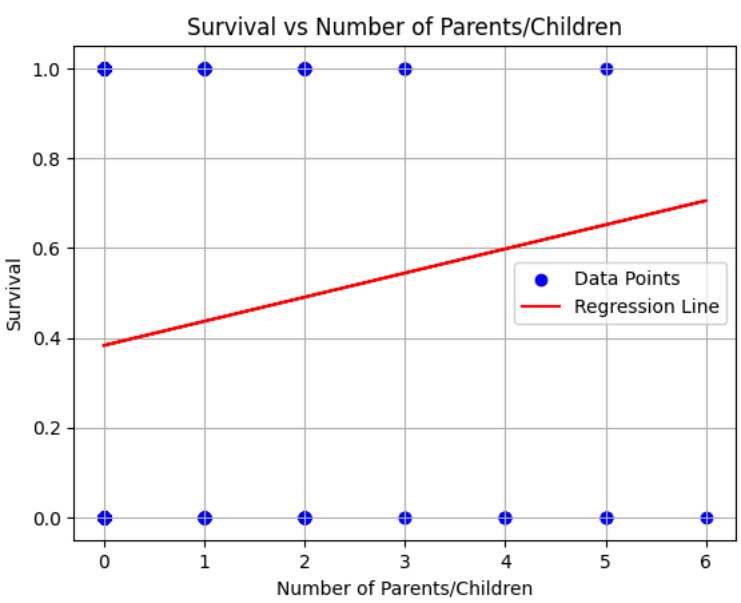
```
titanicFrame = titanicFrame.dropna(subset=['Age', 'Survived'])
x = titanicFrame['Age']
y = titanicFrame['Survived']

# coefficients of the regression line
coefficients = np.polyfit(x, y, 1)
m, b = coefficients
regression_line = m * x + b

# Plotting
plt.scatter(x, y, color='blue', label='Data Points')
plt.plot(x, regression_line, color='red', label='Regression Line')
plt.xlabel("Age")
plt.ylabel("Survival")
plt.title('Survival vs Age')
plt.legend()
plt.grid(True)
plt.show()
```




```
x = titanicFrame['SibSp']
y = titanicFrame['Survived']

# coefficients of the regression line
coefficients = np.polyfit(x, y, 1)
m, b = coefficients
regression_line = m * x + b

# Plotting
plt.scatter(x, y, color='blue', label='Data Points')
plt.plot(x, regression_line, color='red', label='Regression Line')
plt.xlabel("Number of Siblings/Spouses")
plt.ylabel("Survival")
plt.title('Survival vs Number of Siblings/Spouses')
plt.legend()
plt.grid(True)
plt.show()
```




```
x = titanicFrame['Parch']
y = titanicFrame['Survived']

# coefficients of the regression line
coefficients = np.polyfit(x, y, 1)
m, b = coefficients
regression_line = m * x + b

# Plotting
plt.scatter(x, y, color='blue', label='Data Points')
plt.plot(x, regression_line, color='red', label='Regression Line')
plt.xlabel("Number of Parents/Children")
plt.ylabel("Survival")
plt.title('Survival vs Number of Parents/Children')
plt.legend()
plt.grid(True)
plt.show()
```



Part 2: Decision Tree Classification

- Objectives In this lab, you will use a decision tree classifier model to determine who survived the Titanic cruise ship disaster.
- *Part 1: Create a Decision Tree Classifier **
- *Part 2: Apply the Decison Tree Model**

- *Part 3: Evaluate the Decison Tree Model** Scenario / Background In this lab you will create a decision tree classifier that will work with a data set which contains the details about the more than 1300 hundred passengers who were onboard the passenger liner Titanic on its infamous maiden voyage.

Step 1: Create the dataframe

a.) Import pandas and the csv file

```
# Code Cell 1
#create a pandas dataframe called "training" from the titanic-train.csv file
training = pd.read_csv('/content/titanic_train.csv')
```

b.) Verify the import and take a look at the data.

```
#Code Cell 2
training.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Are there Missing Values in the dataset?

- Yes, some of the entries are missing, particularly in the cabin number data, which has only 204 entries.

```
#Code Cell 3
training.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley (Florence	female	38.0	1	0	PC 17599	71.2833

Next steps: Generate code with `training` View recommended plots

```
#Code Cell 4
training["Sex"] = training["Sex"].apply(lambda toLabel: 0 if toLabel ==
'male' else 1)
```

```
#Code Cell 5
training.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	C
0	1	0	3	Braund, Mr. Owen Harris	1	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence	1	38.0	1	0	PC 17599	71.2833	

Next steps: Generate code with `training` View recommended plots

c.) Address Missing Values in the Dataset.

```
#code cell 6
training["Age"].fillna(training["Age"].mean(), inplace=True)
```

d) Verify the values have been replaced.

```
#Code Cell 7
training.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    int64
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(6), object(4)
memory usage: 83.7+ KB
```

What was the valuie used to replace the missing ages?

- The average of all ages is used to fill the missing ages

```
training["Age"].mean()
```

```
29.69911764705882
```

Step 3:Train and Score the Decision Tree Model.

a.) Create an array object with the variable that will be the target for the model.

```
Start coding or generate with AI.
```

```
#code cell 8
#create the array for the target values
y_target = training["Survived"].values
```

b.) Create an array of the values that will be the input for the model.

```
#code cell 9
columns = ["Fare", "Pclass", "Sex", "Age", "SibSp"]
#create the variable to hold the features that the classifier will use
X_input = training[list(columns)].values
```

Double-click (or enter) to edit

c.) Create the learned model

```
#code cell 10
#import the tree module from the sklearn library
from sklearn import tree
#create clf_train as a decision tree classifier object
clf_train = tree.DecisionTreeClassifier(criterion="entropy", max_depth=3)
#train the model using the fit() method of the decision tree object.
#Supply the method with the input variable X_input and the target variable y_target
clf_train = clf_train.fit(X_input, y_target)
```

d.) Evaluate the model

```
#code cell 11
clf_train.score(X_input,y_target)
```

```
0.7216610549943884
```

Step 6: Visualize the Tree

a.) Create the intermediate file output

```
#code cell 12
from six import StringIO
with open("/content/titanic.dot", 'w') as f:
  f = tree.export_graphviz(clf_train, out_file=f, feature_names=columns)
```

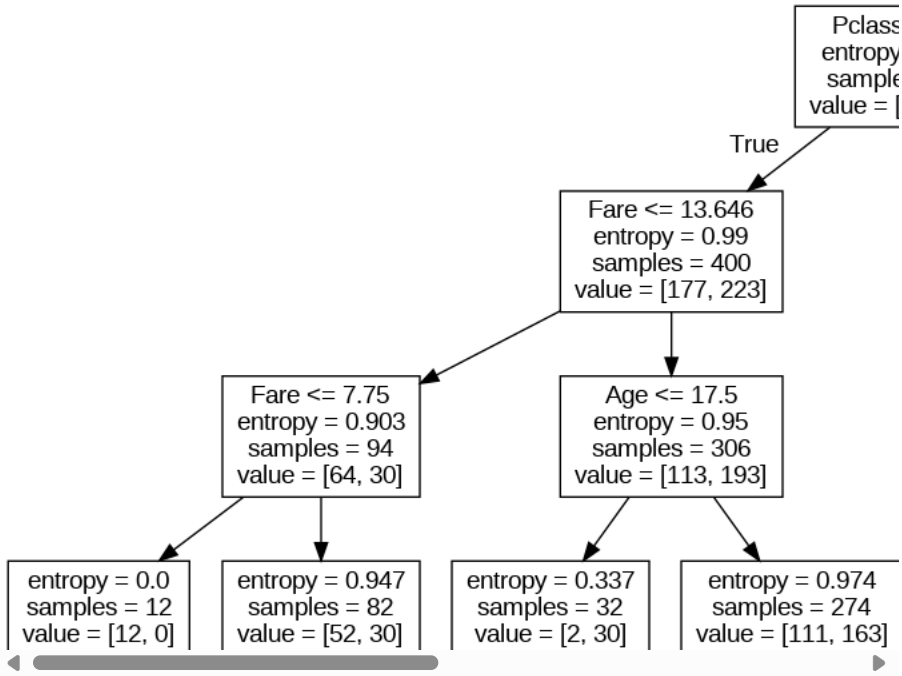
b.) Install Graphviz**

c.) Convert the intermediate file to a graphic

d.) Display the image

```
 #code cell 13
#run the Graphviz dot command to convert the .dot file to .png
!dot -Tpng /content/titanic.dot -o /content/titanic.png
```

```
#code cell 14
#import the Image module from the Ipython.display libary
from IPython.display import Image
#display the decison tree graphic
Image("/content/titanic.png")
```



What describes the group that had the most deaths by number? which group has the msot survivors?

- According to the decision tree, the individuals with the highest survival rates were those in class 2.5 or higher, who paid a fare of at least 13.646, and were younger than 17.5 years old.
- Conversely, the group with the highest mortality rates comprised those in the lower classes, who paid lower fares, and had an average age of 32.5 years.

Apply the Decision Tree Model.

Step 1: Import and Prepare the Data

a.) Import the data

```
#code cell 15
#import the file into the 'testing' dataframe.
testing = pd.read_csv('/content/titanic_test.csv')
testing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  418 non-null    int64  
 1   Pclass       418 non-null    int64  
 2   Name         418 non-null    object 
 3   Sex          418 non-null    object 
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64  
 6   Parch        418 non-null    int64  
 7   Ticket       418 non-null    object 
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object 
 10  Embarked     418 non-null    object 
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

How many records are in the data set?

- 418 records

Which important variables are missing and how many are missing?

- The age variable only has 332 data in it instead of 418 which has 86 missing values.

b) Use a lambda expression to replace the "male" and "female" values with 0 for male and 1 for female.

```
#code cell 16
#replace the Gender labels in the testing dataframe
testing["Sex"] = training["Sex"].apply(lambda toLabel: 0 if toLabel == 'male' else 1)
```

c) Replace the missing age values with the mean of the ages.

```
#code cell 17
#Use the fillna method of the testing dataframe column "Age"
#to replace missing values with the mean of the age values.
testing["Age"].fillna(testing["Age"].mean(), inplace=True)
testing["Fare"].fillna(testing["Fare"].mean(), inplace=True)
testing["Cabin"].fillna("-", inplace=True)
```

d) Verify that the values have been replaced.