Download the titanic dataset from https://www.kaggle.com/competitions/titanic/data

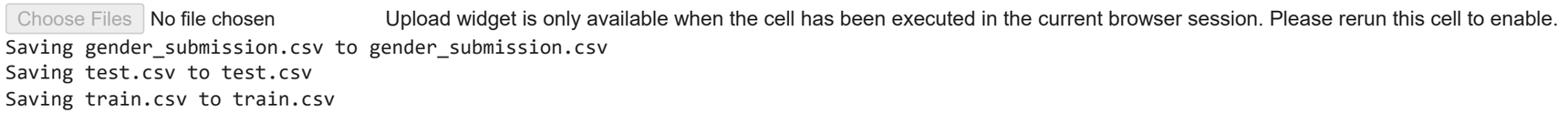
Mridul Goyal

2020BTechCSE051

Section - A

Importing Data:

```
from google.colab import files
uploaded = files.upload()
```




```
Saving gender_submission.csv to gender_submission.csv
Saving test.csv to test.csv
Saving train.csv to train.csv
```

Survival - Survival (0 = No; 1 = Yes)

class - Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)

name - Name

sex - Sex

age - Age

sibsp - Number of Siblings/Spouses Aboard

parch - Number of Parents/Children Aboard

ticket - Ticket Number

fare - Passenger Fare

cabin - Cabin

embarked - Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

boat - Lifeboat (if survived)

body - Body number (if did not survive and body was recovered)

Question 1: -

Create training and testing dataframes from the downloaded csv files. Find

A) number of rows in training and test sets

B) display the structure of the dataset along with the datatypes of the fields

```
import pandas as pd
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
train = pd.read_csv('train.csv')
train
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C

```
test = pd.read_csv('test.csv')
test
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q

```
gender = pd.read_csv('gender_submission.csv')
gender
```

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1
...	...	...
413	1305	0
414	1306	1
415	1307	0
416	1308	0
417	1309	0

418 rows × 2 columns

```
#a
print("Rows in training: ", train.shape[0])
print("Rows in test: ", test.shape[0])
print("Rows in gender submission", gender.shape[0])
```

```
Rows in training:  891
Rows in test:  418
Rows in gender submission 418
```

```
#b
print(train.dtypes)
print(test.dtypes)
print(gender.dtypes)
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
```

```
Embarked        object
dtype: object
PassengerId      int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
PassengerId    int64
Survived       int64
dtype: object
```

```
print(train.describe())
print(test.describe())
print(gender.describe())
```

```
       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000
mean    446.000000    0.383838    2.308642   29.699118    0.523008
std     257.353842    0.486592    0.836071   14.526497    1.102743
min       1.000000    0.000000    1.000000    0.420000    0.000000
25%     223.500000    0.000000    2.000000   20.125000    0.000000
50%     446.000000    0.000000    3.000000   28.000000    0.000000
75%     668.500000    1.000000    3.000000   38.000000    1.000000
max     891.000000    1.000000    3.000000   80.000000    8.000000

            Parch        Fare
count  891.000000  891.000000
mean     0.381594   32.204208
std      0.806057   49.693429
min      0.000000    0.000000
25%      0.000000    7.910400
50%      0.000000   14.454200
75%      0.000000   31.000000
max      6.000000  512.329200
       PassengerId      Pclass         Age       SibSp       Parch        Fare
count   418.000000  418.000000  332.000000  418.000000  418.000000  417.000000
mean   1100.500000    2.265550   30.272590    0.447368    0.392344   35.627188
std     120.810458    0.841838   14.181209    0.896760    0.981429   55.907576
min     892.000000    1.000000    0.170000    0.000000    0.000000    0.000000
25%     996.250000    1.000000   21.000000    0.000000    0.000000    7.895800
50%    1100.500000    3.000000   27.000000    0.000000    0.000000   14.454200
75%    1204.750000    3.000000   39.000000    1.000000    0.000000   31.500000
max    1309.000000    3.000000   76.000000    8.000000    9.000000  512.329200
       PassengerId    Survived
count   418.000000  418.000000
mean   1100.500000    0.363636
std     120.810458    0.481622
min     892.000000    0.000000
25%     996.250000    0.000000
50%    1100.500000    0.000000
```

```
75%    1204.750000    1.000000
max    1309.000000    1.000000
```

Data Cleaning

Question 1 :-

Analyse the data and identify which columns are not relevant for survivor prediction task. Drop those columns from the dataframes.

```
print(train.columns.values)
```

```
['PassengerId' 'Survived' 'Pclass' 'Name' 'Sex' 'Age' 'SibSp' 'Parch'
 'Ticket' 'Fare' 'Cabin' 'Embarked']
```

```
#counting number of null values in each columns of train, test and gender
print(train.isnull().sum())
print(test.isnull().sum())
print(gender.isnull().sum())
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
PassengerId    0
Survived       0
dtype: int64
```

```
#finding categotical data
print("Names of coloumns which have categorical data in data-1-Train - >")
print(train.select_dtypes(include=['object']).columns.tolist())
```

```
print("Names of coloumns which have categorical data in data-2-Test ->")
print(test.select_dtypes(include=['object']).columns.tolist())


print("Names of coloumns which have categorical data in data-3-Gender_submission ->")
print(gender.select_dtypes(include=['object']).columns.tolist())
```

```
Names of coloumns which have categorical data in data-1-Train - >
['Name', 'Sex', 'Ticket', 'Cabin', 'Embarked']
Names of coloumns which have categorical data in data-2-Test ->
['Name', 'Sex', 'Ticket', 'Cabin', 'Embarked']
Names of coloumns which have categorical data in data-3-Gender_submission ->
[]
```

Question 2: -

Check how many columns have missing values in them (NA) and how many have NaN values. Logically impute the dataset.

```
train.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

```
train['Age'] = train['Age'].fillna((train['Age'].median()))
```

```
train['Age'].isnull().sum()
```

```
0
```

Question 3: -

Identify any categorical valued columns (non-numeric) and convert them to numeric.

```
cleanup = {"Sex": {"female": 1, "male": 2}}
```

```
train.replace(cleanup).dtypes
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex              int64
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

Exploratory Analysis (On training set):

Question 1 :-

Show how many passengers were male and female and plot using matplotlib. On the same plot depict the people who survived and who died.
Make accurate axis and legend. Save the plot in a png file.

```
train['Survived'].value_counts()
```

```
0    549
1    342
Name: Survived, dtype: int64
```

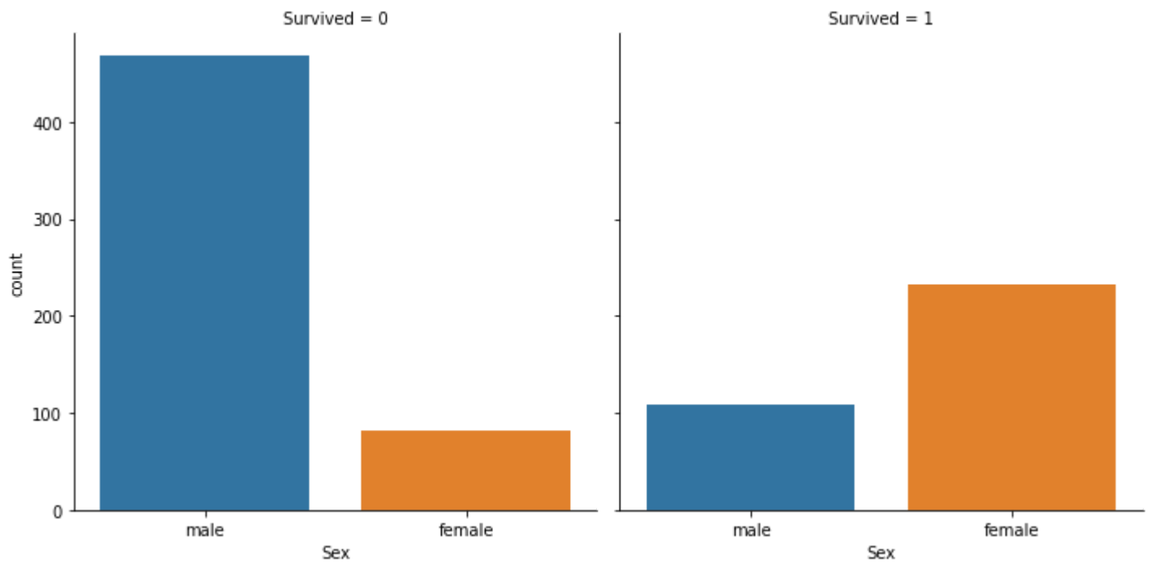
```
train.groupby(['Survived','Sex'])['Survived'].count()
```

```
Survived  Sex
0         female     81
          male      468
1         female    233
          male      109
Name: Survived, dtype: int64
```

```
train['Survived']==0
```

```
0       True
1      False
2      False
3      False
4       True
       ...
886     True
887    False
888     True
889    False
890     True
Name: Survived, Length: 891, dtype: bool
```

```
sns.catplot(x='Sex', col='Survived', kind='count', data=train);
plt.savefig('plot.png', dpi=300, bbox_inches='tight')
```



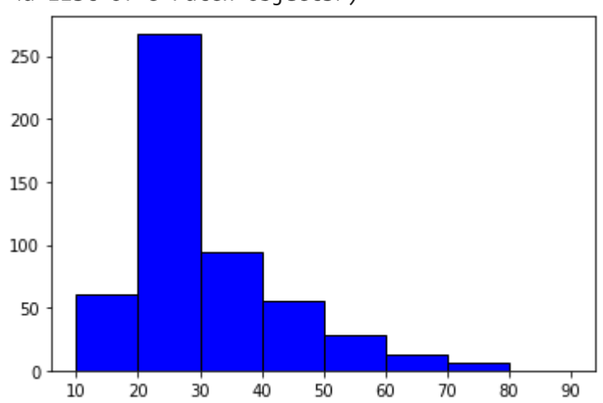
Question 2 :-

Show the histogram of the count of passengers who died (according to their age). Age ranges should be <10, 10 to <20, 20 to <30 and so on.

How many minor children died and how many of them survived (<16 years). Create a separate plot for the passengers who survived.

```
b=train.query('Survived==0')
a=b['Age']
plt.hist(a,bins=[10,20,30,40,50,60,70,80,90],edgecolor='black',color='blue')
```

```
(array([ 61., 268.,  94.,  55.,  28.,  13.,   6.,   0.]),
 array([10, 20, 30, 40, 50, 60, 70, 80, 90]),
 <a list of 8 Patch objects>)
```

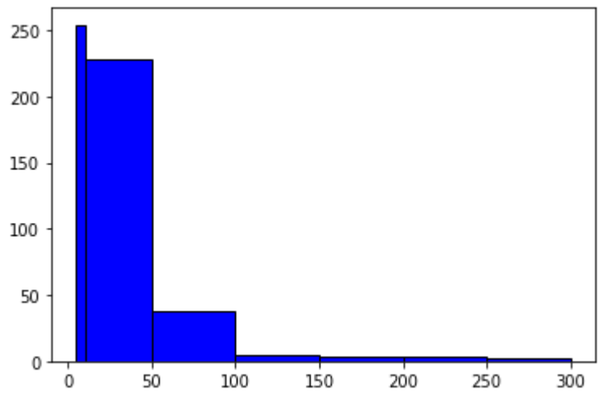


Question 3: -

Show the distribution on the count of passengers who died (according to the fare they paid). Choose fare ranges such that the mean lies in the middle range.

Give the percentage of passengers who survived as had paid more than $100. Justify if there was any bias in the rescue operation towards the rich (Yes/No/not enough evidence).

```
fare=train.query('Survived!=1')
fare
#m=fare.groupby(['Fare'])['Survived']
plt.hist(fare['Fare'],bins=[5,10,50,100,150,200,250,300],edgecolor='black', color='blue')
plt.show()
```




```
train=pd.read_csv("train.csv")
face=train.query('Survived==1 and Fare>=100')
#print(df_1)
#m=fare.groupby(['Fare'])['Survived']
plt.hist(fare['Fare'],bins=[5,10,50,100,150,200,250,300],edgecolor='black',color='blue')
plt.show()
# perecenatge of passenger who survived and they paid fare>100
total_sur_pess=train.query('Fare>=100')
n_f=total_sur_pess['PassengerId'].count()
n_s=fare['PassengerId'].count()
print("Number of Passenger who survived and have fare>=100 = ", n_s)
#n_f=fare['PassengerId'].count()
print(" Number of PassengerId who have paid fare>=100 = ",n_f)
per=(n_s/n_f)*100
print("Percentage of passenger survived = ",per,"%")
```

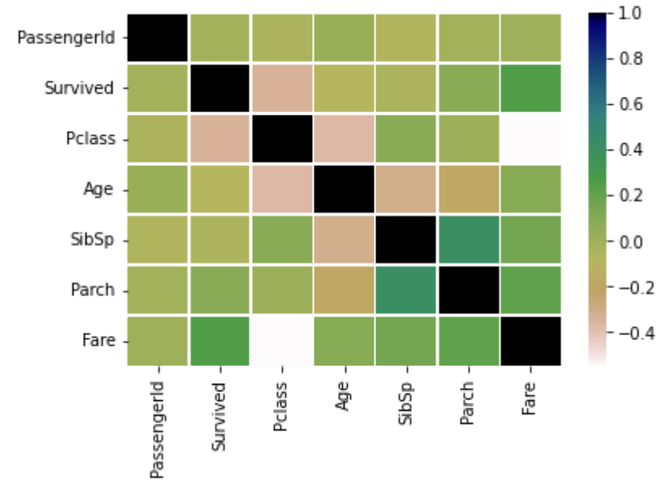
Question 4: -

Plot graphs showing correlation between different pairs of attributes. Infer if there is any significant correlation between survivors and any specific feature.

```
cor=train.corr()
print(cor)
#plotting the corrraltion matrix
sns.heatmap(cor,linewidth = 0.5 , cmap = 'gist_earth_r')
```

```
             PassengerId  Survived    Pclass       Age     SibSp     Parch  \
PassengerId     1.000000 -0.005007 -0.035144  0.036847 -0.057527 -0.001652
Survived       -0.005007  1.000000 -0.338481 -0.077221 -0.035322  0.081629
Pclass         -0.035144 -0.338481  1.000000 -0.369226  0.083081  0.018443
Age             0.036847 -0.077221 -0.369226  1.000000 -0.308247 -0.189119
SibSp          -0.057527 -0.035322  0.083081 -0.308247  1.000000  0.414838
Parch          -0.001652  0.081629  0.018443 -0.189119  0.414838  1.000000
Fare            0.012658  0.257307 -0.549500  0.096067  0.159651  0.216225

                 Fare
PassengerId  0.012658
Survived     0.257307
Pclass      -0.549500
Age          0.096067
SibSp        0.159651
Parch        0.216225
Fare         1.000000
<matplotlib.axes._subplots.AxesSubplot at 0x7f611751ae90>
```



Question 5: -

Find the number of passengers who were married.

```
n=train['Name'].tolist()
c=0
for i in range(len(n)):
    for j in range(i,len(n[i])):
        if('Mrs.' in n[j] ):
            c=c+1
print(c)
```

```
128
```