5. Write a program to perform the following tasks a. Determine the outliers in each non-categorical column of Titanic Data and remove them.

```
# importing all the necessary libraries
import pandas as pd
import numpy as np
#we need to read the data
data = pd.read_csv("/content/drive/MyDrive/AI Tools Lab/train.csv")
#print top 5 rows
print(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
```

```
# function to calculate the lower and upperbound
def detect_outliers(data,threshold):
 mean = np.mean(data)
 std =np.std(data)
 lb = max(mean - (threshold * std),min(data))
 ub = min(mean + (threshold * std),max(data))
 return lb,ub
df = data.copy()
lb,ub = detect_outliers(data["Fare"],4)
# removing the rows which are greater than upperbound
df.drop(df[df.Fare > ub].index, inplace=True)
# removing the rows which are less than lowerbound
df.drop(df[df.Fare < lb ].index, inplace=True)


print("lb: ",lb,"ub: ",ub)
df.sort_values(by="Fare",ascending=True,inplace=True)
```

```
lb:  0.0 ub:  230.86634574767106
```

```
print(df)
```

```
     PassengerId  Survived  Pclass  \
271          272         1       3
597          598         0       3
302          303         0       3
633          634         0       1
277          278         0       2
..           ...       ...     ...
527          528         0       1
716          717         1       1
380          381         1       1
557          558         0       1
700          701         1       1

                                             Name     Sex   Age  SibSp  \
271                      Tornquist, Mr. William Henry    male  25.0      0
597                               Johnson, Mr. Alfred    male  49.0      0
302                    Johnson, Mr. William Cahoone Jr    male  19.0      0
633                      Parr, Mr. William Henry Marsh    male   NaN      0
277                        Parkes, Mr. Francis "Frank"    male   NaN      0
..                                                 ...     ...   ...    ...
527                              Farthing, Mr. John    male   NaN      0
716                      Endres, Miss. Caroline Louise  female  38.0      0
380                             Bidois, Miss. Rosalie  female  42.0      0
557                               Robbins, Mr. Victor    male   NaN      0
700  Astor, Mrs. John Jacob (Madeleine Talmadge Force)  female  18.0      1

     Parch    Ticket      Fare    Cabin Embarked
271      0      LINE    0.0000      NaN        S
597      0      LINE    0.0000      NaN        S
302      0      LINE    0.0000      NaN        S
```

```
633      0    112052    0.0000      NaN        S
277      0    239853    0.0000      NaN        S
..     ...       ...       ...      ...      ...
527      0  PC 17483  221.7792      C95        S
716      0  PC 17757  227.5250      C45        C
380      0  PC 17757  227.5250      NaN        C
557      0  PC 17757  227.5250      NaN        C
700      0  PC 17757  227.5250  C62 C64        C

[880 rows x 12 columns]
```

```
lb,ub = detect_outliers(data["Age"],5)
# removing the rows which are greater than upperbound
df.drop(df[df.Age > ub].index, inplace=True)
# removing the rows which are less than lowerbound
df.drop(df[df.Age < lb].index, inplace=True)
```

```
print("lb: ",lb,"ub: ",ub)
df.sort_values(by="Age",ascending=False,inplace=True)
```

```
lb:  0.42 ub:  80.0
```

```
print(df)
```

```
     PassengerId  Survived  Pclass  \
630          631         1       1
851          852         0       3
96            97         0       1
493          494         0       1
116          117         0       3
..           ...       ...     ...
306          307         1       1
334          335         1       1
31            32         1       1
527          528         0       1
557          558         0       1

                                                Name     Sex   Age  SibSp  \
630               Barkworth, Mr. Algernon Henry Wilson    male  80.0      0
851                             Svensson, Mr. Johan    male  74.0      0
96                          Goldschmidt, Mr. George B    male  71.0      0
493                           Artagaveytia, Mr. Ramon    male  71.0      0
116                              Connors, Mr. Patrick    male  70.5      0
..                                                ...     ...   ...    ...
306                            Fleming, Miss. Margaret  female   NaN      0
334  Frauenthal, Mrs. Henry William (Clara Heinshei...  female   NaN      1
31      Spencer, Mrs. William Augustus (Marie Eugenie)  female   NaN      1
527                               Farthing, Mr. John    male   NaN      0
557                             Robbins, Mr. Victor    male   NaN      0

     Parch    Ticket      Fare Cabin Embarked
630      0     27042   30.0000   A23        S
851      0    347060    7.7750   NaN        S
96       0  PC 17754   34.6542    A5        C
493      0  PC 17609   49.5042   NaN        C
116      0    370369    7.7500   NaN        Q
..     ...       ...       ...   ...      ...
306      0     17421  110.8833   NaN        C
334      0  PC 17611  133.6500   NaN        S
31       0  PC 17569  146.5208   B78        C
527      0  PC 17483  221.7792   C95        S
557      0  PC 17757  227.5250   NaN        C

[880 rows x 12 columns]
```

b. Determine missing values in each column of Titanic data. If missing values account for 30% of data, then remove the column.

```
#printing the missing value percentage for every column
df.isnull().mean() * 100
```

```
PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            20.113636
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.000000
Cabin          77.954545
Embarked        0.227273
dtype: float64
```

```
# get all the column names in our dataset
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```
# As we can see cabin column has more than 30% of missing values, so we have to drop that column
df.drop(['Cabin'],inplace=True,axis=1)
```

```
# after removing the column cabin, printing the columns again. If you observe there is no Cabin in the output
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Embarked'],
      dtype='object')
```

c. If missing values are less than 30% of entire data then create a new data frame i. Missing values in numeric columns are filled with the mean of the corresponding column.

```
#printing the percentage of missing values in Age before handling
df['Age'].isnull().mean() * 100
```

```
20.113636363636363
```

```
# Filling the missing values with the mean of respective column
df['Age']=df['Age'].fillna(df['Age'].mean())
```

```
#printing the percentage of missing values in Age after handling
df['Age'].isnull().mean() * 100
```

```
0.0
```

ii. Missing values in categorical columns are filled with the most frequently occurring value.

```
#printing the percentage of missing values in Embarked before handling
df['Embarked'].isnull().mean() * 100
```

```
0.22727272727272727
```

```
# filling with filled with the most frequently occurring value.
df["Embarked"].fillna(df['Embarked'].mode()[0],inplace=True)
```

```
#printing the percentage of missing values in Embarked after handling
df['Embarked'].isnull().mean() * 100
```

```
0.0
```

```
df.to_csv("/content/drive/MyDrive/AI Tools Lab/nonnull_titanic.csv",index=False)
```

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