- NAME: SATYAM PUNDIR
- EMAIL: satyam.pundir2021@vitstudent.ac.in
- Branch: IT
- Campus: VIT Vellore

Import the Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

warnings.simplefilter(action='ignore', category=FutureWarning)
```

Import the Dataset

```
df = pd.read_csv("Titanic-Dataset.csv")
df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803

```
df.set_index('PassengerId', inplace=True)
```

```
df.shape
```

```
(891, 11)
```

```
df.describe()
```

	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 891 entries, 1 to 891
```

```
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  891 non-null    int64  
 1   Pclass    891 non-null    int64  
 2   Name      891 non-null    object 
 3   Sex       891 non-null    object 
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64  
 6   Parch     891 non-null    int64  
 7   Ticket    891 non-null    object 
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object 
 10  Embarked  889 non-null    object 
dtypes: float64(2), int64(4), object(5)
memory usage: 83.5+ KB
```

Handling Null Value

```
df.isnull().any()
```

```
Survived    False
Pclass      False
Name        False
Sex         False
Age          True
SibSp       False
Parch       False
Ticket      False
Fare        False
Cabin        True
Embarked     True
dtype: bool
```

```
df.isnull().sum()
```

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

Drop the Column Cabin as it has so many NULL values and it is of no use.

```
df.drop('Cabin', axis=1,inplace=True)
```

Replace the Null value of Ages with the mean and Replace the Null value of Embarked with the Mode

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

```
177
```

```
df[df['Age'].isnull()]
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
PassengerId										
6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	Q
18	1	2	Williams, Mr. Charles Eugene	male	NaN	0	0	244373	13.0000	S
20	1	3	Masselmani, Mrs. Fatima	female	NaN	0	0	2649	7.2250	C
27	0	3	Emir, Mr. Farred Chehab	male	NaN	0	0	2631	7.2250	C
29	1	3	O'Dwyer, Miss. Ellen "Nellie"	female	NaN	0	0	330959	7.8792	Q

```
df['Age'].fillna(df['Age'].mean(),inplace=True)
```

864	0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN	8	2	CA. 2343	69.5500	S

```
df[df['Embarked'].isnull()]
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
PassengerId										
62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	NaN
830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	NaN

To handle the Embarked, we will label encode it.

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
df['Embarked'] = le.fit_transform(df['Embarked'])
```

```
df['Embarked']
```

```
PassengerId
1      2
2      0
3      2
4      2
5      2
      ..
887    2
888    2
889    2
890    0
891    1
Name: Embarked, Length: 891, dtype: int32
```

S (southampton) --> 2

```
Q (QueensTown)  --> 0
C (CherBourg)   --> 1
```

```
df['Embarked'].fillna(df['Embarked'].mode(),inplace=True)
```

```
df.isnull().sum()
```

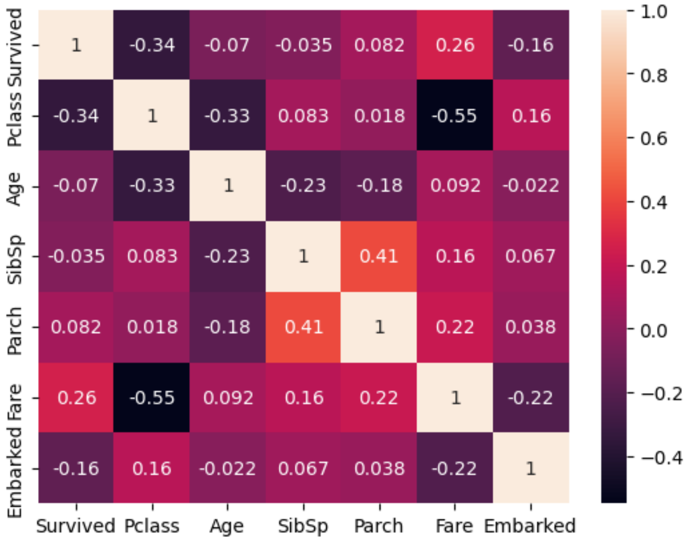
```
Survived    0
Pclass      0
Name        0
Sex         0
Age         0
SibSp       0
Parch       0
Ticket      0
Fare        0
Embarked    0
dtype: int64
```

We Succefully handled all the Null Values.

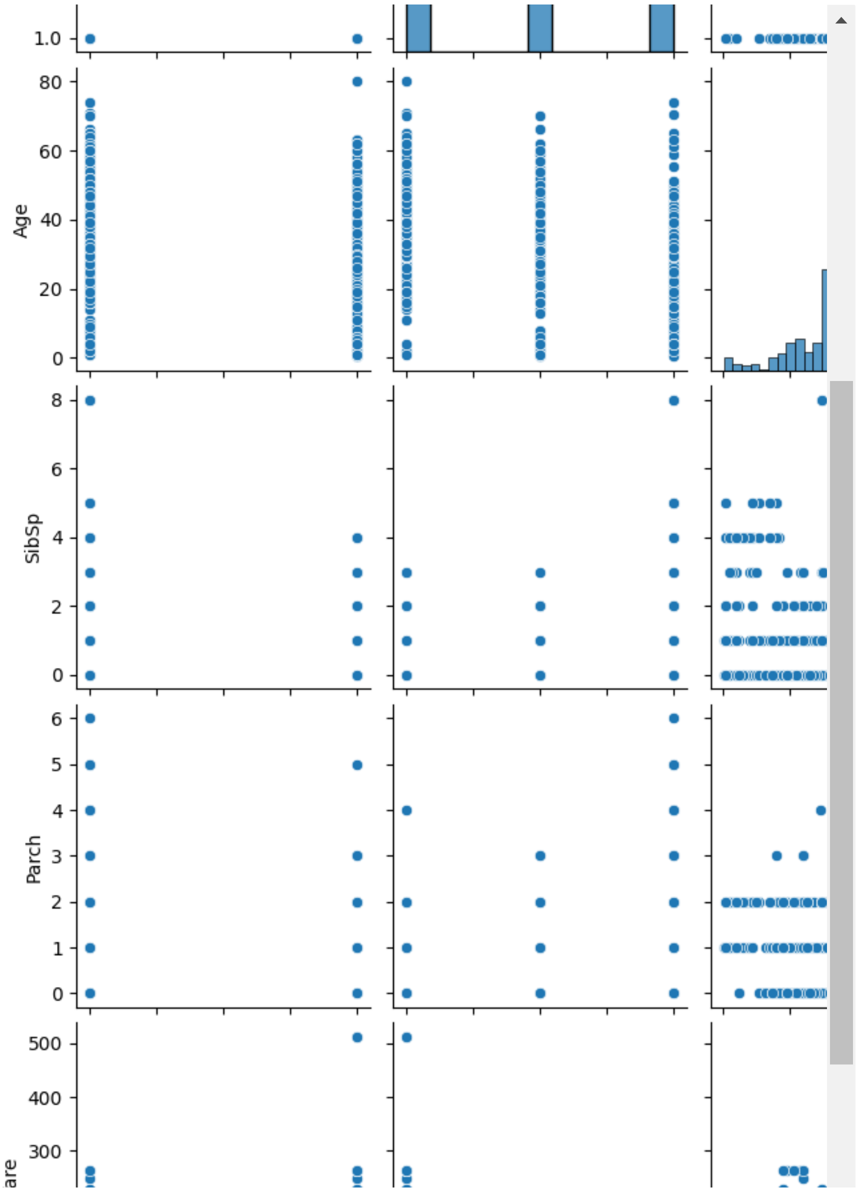
Data Visualisation

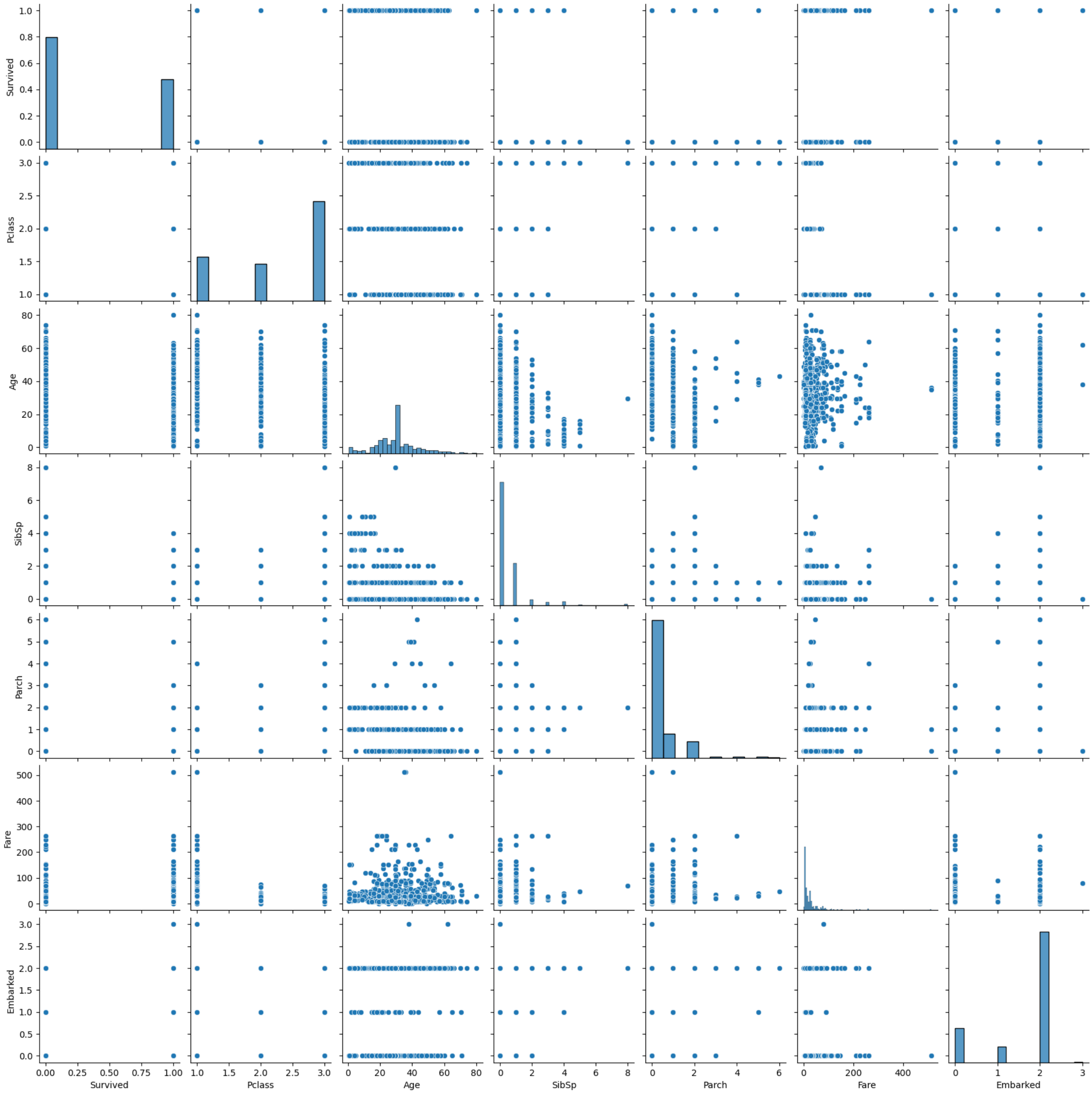
```
sns.heatmap(df.corr(numeric_only=True),annot=True)
```

```
<Axes: >
```



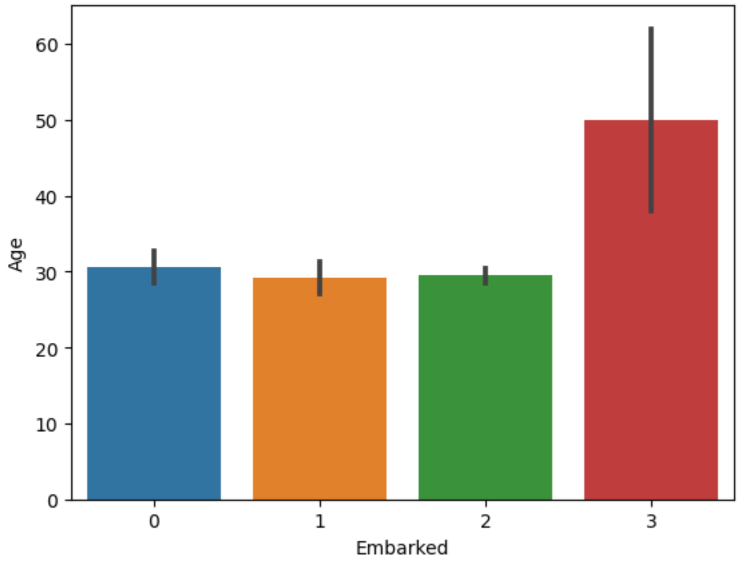
```
sns.pairplot(df)
```


1.0
80
60
Age
40
20
0
8
6
SibSp
4
2
0
6
5
4
Parch
3
2
1
0
500
400
300
are



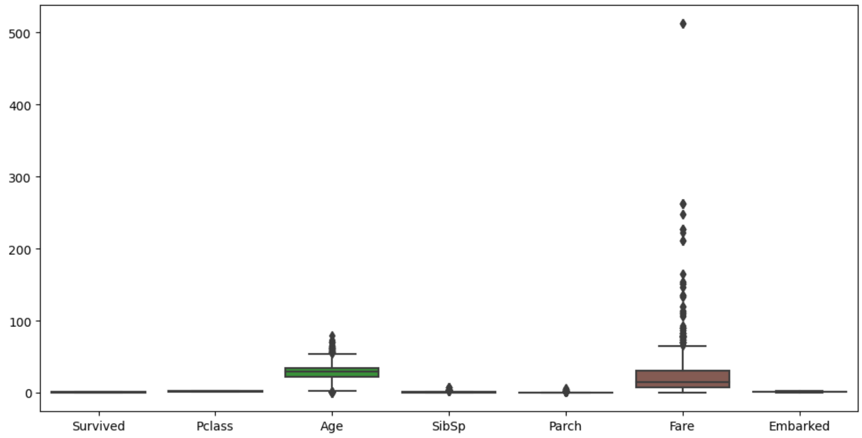
```
sns.barplot(x=df["Embarked"],y=df["Age"]) #printing barplot between embarked and age
```

```
<Axes: xlabel='Embarked', ylabel='Age'>
```

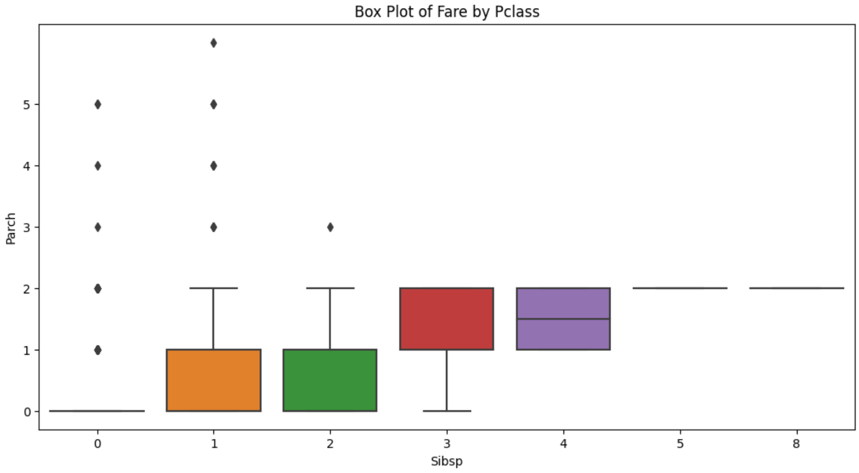


Outlier Detection

```
plt.figure(figsize=(12,6))
sns.boxplot(df)
plt.show()
```

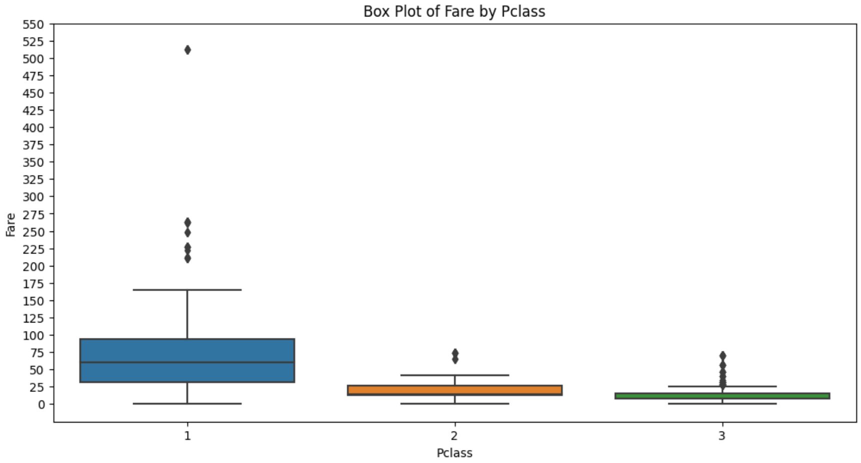


```
plt.figure(figsize=(12,6))
sns.boxplot(data=df,y='Parch',x='SibSp')

plt.yticks(np.arange(0, df['Parch'].max(), 1))
plt.xlabel("Sibsp")
plt.ylabel("Parch")
plt.title("Box Plot of Fare by Pclass")
plt.show()
```



We have Outliers in Age, Sibsp, Parch, Fare

```
plt.figure(figsize=(12,6))
sns.boxplot(data=df,y='Fare',x='Pclass')

plt.yticks(np.arange(0, df['Fare'].max() + 50, 25))
plt.xlabel("Pclass")
plt.ylabel("Fare")
plt.title("Box Plot of Fare by Pclass")
plt.show()
```



We can see that the Fare Pricess depends on the Class so we cant Fully remove it. We need to Deal it with respect to the Pclass.

```
outlier_id_1 = df[(df['Pclass']==1) & (df['Fare'] > 180)].index.to_numpy()
```

```
outlier_id_1
```

```
    array([ 28,  89, 119, 259, 300, 312, 342, 378, 381, 439, 528, 558, 680,
           690, 701, 717, 731, 738, 743, 780], dtype=int64)
```

```
outlier_id_2 = df[(df['Pclass']==2) & (df['Fare'] > 50)].index.to_numpy()
outlier_id_2
```

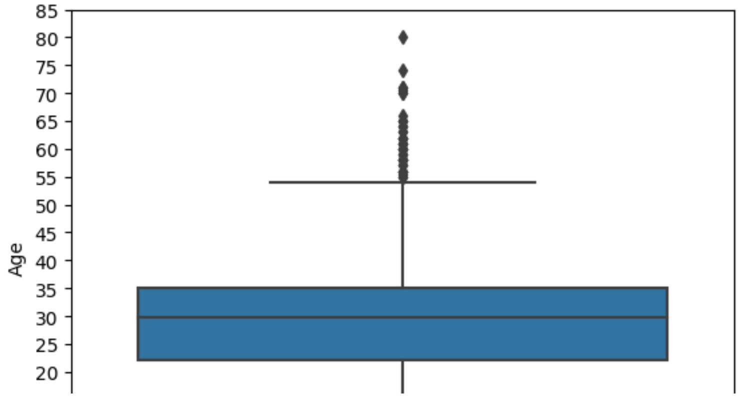
```
    array([ 73, 121, 386, 616, 656, 666, 755], dtype=int64)
```

```
outlier_id_3 = df[(df['Pclass']==3) & (df['Fare'] > 25)].index.to_numpy()
outlier_id_3
```

```
    array([ 14,  17,  26,  51,  60,  64,  72,  75,  87, 120, 148, 160, 165,
           168, 170, 172, 177, 181, 183, 202, 230, 234, 262, 267, 279, 325,
           361, 387, 410, 437, 481, 486, 510, 542, 543, 611, 635, 639, 643,
           644, 679, 684, 687, 693, 737, 788, 793, 814, 820, 825, 827, 839,
           847, 851, 864, 886], dtype=int64)
```

For Age

```
sns.boxplot(data=df, y=df['Age'])
plt.yticks(np.arange(0, df['Age'].max() + 10, 5))
plt.show()
```



```
def outliers (df, ft):
    q1 = df[ft].quantile(0.25)
    q3 = df[ft].quantile(0.75)
    iqr = q3-q1
    lower_bound = q1- 1.5*iqr
    upper_bound = q3 + 1.5*iqr
    ls = df[(df[ft]<lower_bound) | (df[ft] > upper_bound)].index.to_numpy()
    print(ls.shape)
    return ls
```

```
outlier_id_4 = outliers(df,'Age')
```

```
(66,)
```

```
outlier_id_5 = outliers(df,'SibSp')
```

```
(46,)
```

```
outlier_id_6 = outliers(df,'Parch')
```

```
(213,)
```

```
type(outlier_id_6)
```

```
numpy.ndarray
```

```
outlier_final = np.concatenate((outlier_id_1 ,outlier_id_2 , outlier_id_3 , outlier_id_4 , outlier_id_5,outlier_id_6))
```

```
len(outlier_final)
```

```
408
```

```
outlier_final = np.unique(outlier_final)
```

```
outlier_final
```

```
array([  8,   9,  11,  12,  14,  16,  17,  25,  26,  28,  34,  44,  51,
        55,  59,  60,  64,  66,  69,  72,  73,  75,  79,  86,  87,  89,
        94,  95,  97,  98,  99, 103, 117, 119, 120, 121, 125, 129, 137,
       141, 146, 148, 149, 153, 154, 156, 160, 161, 165, 166, 167, 168,
       170, 171, 172, 173, 175, 176, 177, 181, 183, 184, 185, 189, 194,
       196, 198, 202, 206, 230, 233, 234, 238, 248, 249, 252, 253, 255,
       256, 259, 260, 262, 263, 267, 269, 273, 274, 276, 279, 280, 281,
       298, 300, 306, 312, 313, 315, 319, 320, 324, 325, 327, 329, 330,
       333, 341, 342, 349, 353, 357, 361, 363, 367, 375, 378, 381, 382,
       386, 387, 391, 395, 408, 410, 417, 418, 420, 424, 425, 436, 437,
       438, 439, 441, 446, 447, 449, 451, 457, 468, 470, 473, 480, 481,
       484, 486, 488, 490, 493, 494, 499, 507, 510, 524, 528, 530, 531,
       533, 534, 536, 540, 541, 542, 543, 546, 549, 550, 551, 556, 558,
       559, 568, 571, 581, 582, 586, 588, 594, 596, 601, 609, 611, 616,
       617, 619, 623, 626, 627, 631, 635, 638, 639, 643, 644, 645, 648,
       652, 656, 658, 660, 666, 671, 673, 679, 680, 684, 685, 686, 687,
       690, 692, 693, 695, 699, 701, 703, 710, 717, 721, 727, 731, 737,
       738, 743, 746, 747, 751, 752, 755, 756, 764, 773, 775, 780, 784,
       788, 789, 793, 800, 802, 803, 804, 814, 818, 820, 821, 824, 825,
       827, 828, 830, 832, 836, 839, 847, 849, 851, 852, 853, 854, 856,
       857, 859, 864, 870, 872, 880, 881, 886, 889], dtype=int64)
```

```
len(outlier_final)
```

```
269
```

Total Outlier Spotted are 269, we can remove them also since we have the value

Splitting Dependent and Independent variables

```
Dependent column is Survived, and Independent is remaing others
```

```
df.head()
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Far
PassengerId									
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.250
2	1	1	Cumings, Mrs. John Bradley (Florence	female	38.0	1	0	PC 17599	71.283

```
x=df.drop(columns=["Name","Ticket","Survived"],axis=1)
x.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
PassengerId							
1	3	male	22.0	1	0	7.2500	2
2	1	female	38.0	1	0	71.2833	0
3	3	female	26.0	0	0	7.9250	2
4	1	female	35.0	1	0	53.1000	2
5	3	male	35.0	0	0	8.0500	2

```
x.shape
```

```
(891, 8)
```

```
y=df["Survived"]
y.head()
```

```
PassengerId
1    0
2    1
3    1
4    1
5    0
Name: Survived, dtype: int64
```

Encoding

```
x["Sex"]=le.fit_transform(x["Sex"])
```

```
x
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
PassengerId							
1	3	1	22.000000	1	0	7.2500	2
2	1	0	38.000000	1	0	71.2833	0
3	3	0	26.000000	0	0	7.9250	2
4	1	0	35.000000	1	0	53.1000	2
5	3	1	35.000000	0	0	8.0500	2

We already encoded the Embarked while dealing with the NULL values

Splitting into

890 1 1 26.000000 0 0 30.0000 0

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
((623, 7), (268, 7), (623,), (268,))
```

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

```
x_data_train=sc.fit_transform(x_train)
x_data_test=sc.fit_transform(x_test)
```

```
x_data_train
```

```
array([[-1.5325562 ,  0.72592065,  1.62393675, ..., -0.47299765,
        -0.12253019,  0.56011053],
       [-1.5325562 , -1.37756104,  1.47020331, ..., -0.47299765,
         0.91812372, -2.02469583],
       [ 0.84844757,  0.72592065, -2.21939923, ...,  1.93253327,
         0.29950338,  0.56011053],
       ...,
       [ 0.84844757,  0.72592065, -0.0133922 , ..., -0.47299765,
        -0.51276504, -0.73229265],
       [ 0.84844757, -1.37756104,  0.47093596, ..., -0.47299765,
        -0.31228976,  0.56011053],
       [-0.34205431,  0.72592065,  2.31573723, ...,  0.72976781,
         0.13566725,  0.56011053]])
```