Name: Satyam Mishra

Reg : 21BCE8247

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

dataset=pd.read_csv("dataset\Titanic-Dataset.csv")

dataset.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
```

```
dataset.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
```

```
dataset.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
```

Checking for Null Values

```
print(dataset.isnull().any())
```

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age             True
SibSp          False
Parch          False
Ticket         False
```

```
Fare            False
Cabin            True
Embarked         True
dtype: bool
```

```
print(dataset.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
```

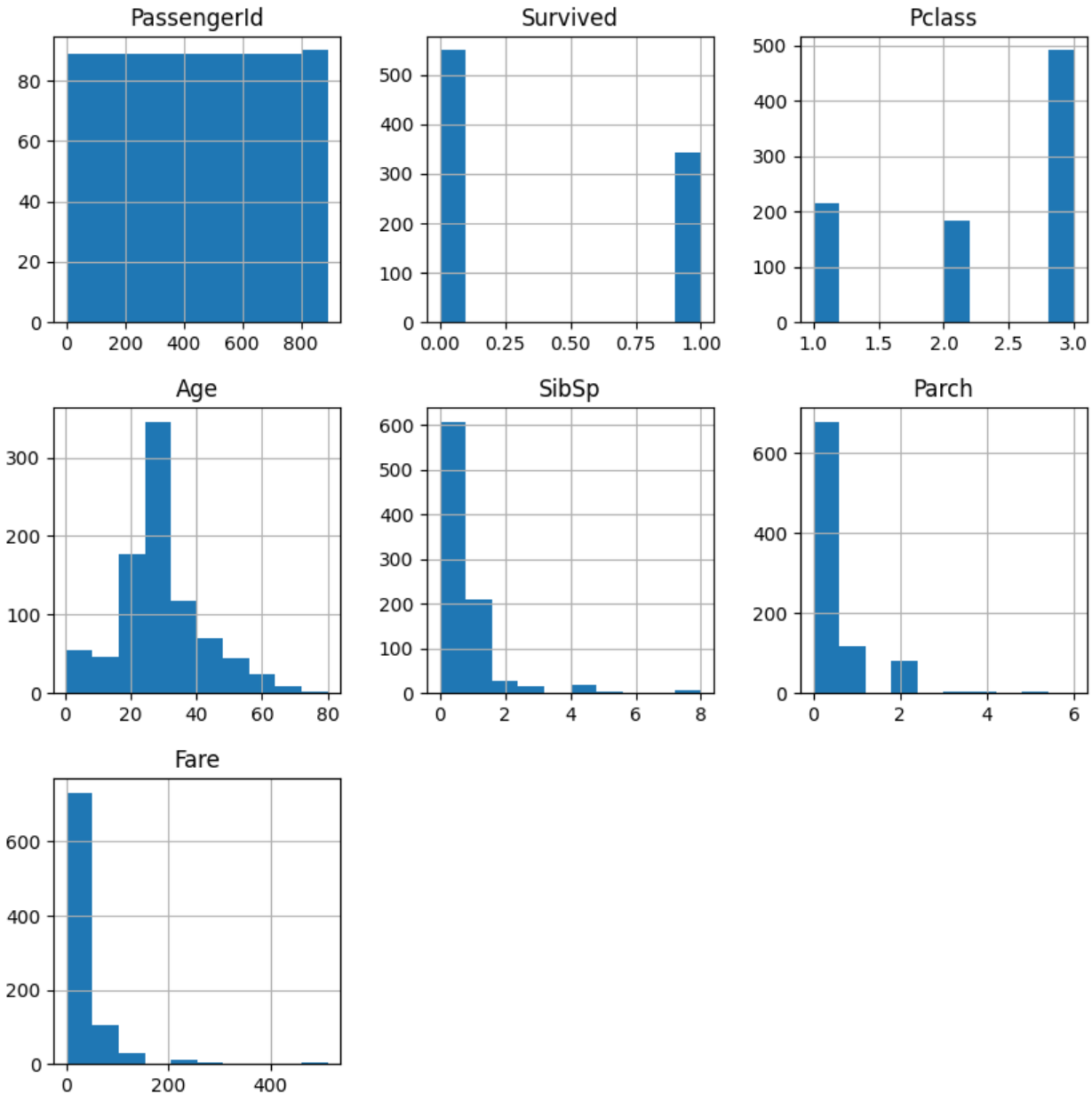
```
# Age, Cabin, Embarked columns has null value, so we will remove it.
# Age is numerical value so we will asssign mean of dataset to it

dataset["Age"].fillna(dataset["Age"].mean(),inplace=True)

# For Cabin and Embarked which are Categorical value we use mode

dataset["Embarked"].fillna(dataset["Embarked"].mode()[0],inplace=True)
dataset["Cabin"].fillna(dataset["Cabin"].mode()[0],inplace=True)
```

```
print(dataset.isnull().any())
```

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

Data Visualization (Histogram for each feature)

```
dataset.hist(figsize=(10,10))
plt.show()
```



Splitting Dependent and Independent variables

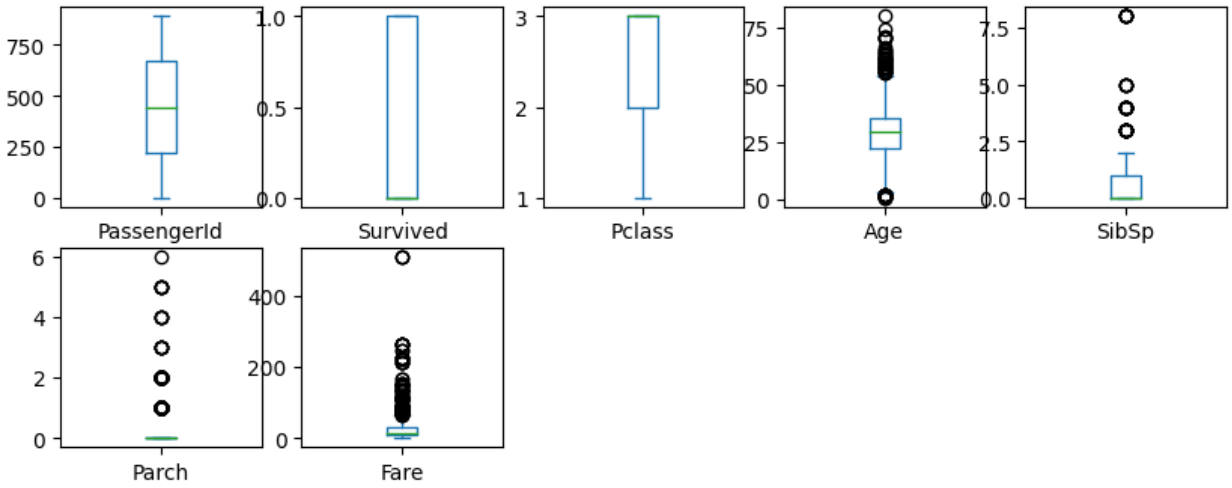
```
dataset.head()
```

```
   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  B96 B98        S
1      0          PC 17599  71.2833      C85        C
2      0  STON/O2. 3101282   7.9250  B96 B98        S
3      0            113803  53.1000     C123        S
4      0            373450   8.0500  B96 B98        S

x = dataset.drop(['Survived','PassengerId','Name','Ticket','Cabin'],
axis=1)  # Independent variables

y = dataset['Survived']  # Dependent variable

print(x.head())
print(y.head())

   Pclass     Sex   Age  SibSp  Parch     Fare Embarked
0       3    male  22.0      1      0   7.2500        S
1       1  female  38.0      1      0  71.2833        C
2       3  female  26.0      0      0   7.9250        S
3       1  female  35.0      1      0  53.1000        S
4       3    male  35.0      0      0   8.0500        S
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

Outlier Detection (Boxplot for each feature)

```
dataset.plot(kind='box', subplots=True, layout=(5,5), sharex=False,
sharey=False, figsize=(10,10))
plt.show()
```



Perform Encoding

```
x.head()
```

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

```
x["Sex"].value_counts()
```

```
Sex
male      577
female    314
Name: count, dtype: int64
```

```
x["Sex"].nunique()
```

```
2
```

```
from sklearn.preprocessing import LabelEncoder

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

```
x["Sex"].head()
```

```
0    1
1    0
2    0
3    0
4    1
Name: Sex, dtype: int32
```

```
Embarked=pd.get_dummies(x["Embarked"])
Embarked.head()
```

```
       C      Q      S
0  False  False   True
1   True  False  False
2  False  False   True
3  False  False   True
4  False  False   True
```

```
# adding Embarked DataFrame in x
x=pd.concat([x,Embarked],axis=1)

# Drop Embarked
x.drop(["Embarked"],axis=1,inplace=True)

x.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	C	Q	S
0	3	1	22.0	1	0	7.2500	False	False	True
1	1	0	38.0	1	0	71.2833	True	False	False
2	3	0	26.0	0	0	7.9250	False	False	True
3	1	0	35.0	1	0	53.1000	False	False	True
4	3	1	35.0	0	0	8.0500	False	False	True

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
# Feature Scaling
scaler = StandardScaler()
x = scaler.fit_transform(x)

x
```

```
array([[ 8.27377244e-01,  7.37695132e-01, -5.92480600e-01, ...,
        -4.82042680e-01, -3.07562343e-01,  6.15838425e-01],
       [-1.56610693e+00, -1.35557354e+00,  6.38789012e-01, ...,
         2.07450510e+00, -3.07562343e-01, -1.62380254e+00],
       [ 8.27377244e-01, -1.35557354e+00, -2.84663197e-01, ...,
        -4.82042680e-01, -3.07562343e-01,  6.15838425e-01],
       ...,
       [ 8.27377244e-01, -1.35557354e+00, -2.23290646e-16, ...,
        -4.82042680e-01, -3.07562343e-01,  6.15838425e-01],
       [-1.56610693e+00,  7.37695132e-01, -2.84663197e-01, ...,
         2.07450510e+00, -3.07562343e-01, -1.62380254e+00],
       [ 8.27377244e-01,  7.37695132e-01,  1.77062908e-01, ...,
        -4.82042680e-01,  3.25137334e+00, -1.62380254e+00]])
```

Splitting into training and testing set

```
from sklearn.model_selection import train_test_split


x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,rando
m_state=0)

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

```
((712, 9), (179, 9), (712,), (179,))
```