data preprocessing

Step-1: importing required libraries

```
In[1]: import numpy as np
       import pandas as pd
       import matplotlib.pyplot as plt
       import seaborn as sns
```

Step-2: importing the dataset

```
In[2]: df=pd.read_csv("Titanic-Dataset.csv")
       df.head()
```

```
Out[2]:   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
```

```
In[3]: df.shape
```

```
Out[3]: (891, 12)
```

```
In[4]: df.info()
```

```
Out[4]: <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
        dtype: float64(2), int64(7), object(3)
        memory usage: 83.7 KB
```

```
In[5]: df.describe()
```

```
Out[5]:       PassengerId    Survived      Pclass         Age       SibSp       Parch        Fare
        count   891.000000  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
        mean    446.000000    0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
        std     257.353842    0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
        min       1.000000    0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
        25%     223.500000    0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
        50%     446.000000    0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
        75%     668.500000    1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
        max     891.000000    1.000000    3.000000   80.000000    8.000000    6.000000  512.329200
```

Step-3: checking and handling null values

```
In[6]: df.isnull().sum()
```

```
Out[6]: 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
```

we can see from above information that columns AGE,CABIN and EMBARKED have null values present in them. we can handle the null values in 2 different ways 1.by deleting them 2.by deleting entire rows/column 3.by replacing with mean or median.

```
In[7]: # we use 3 numerical data to we need mean to replace the null values.
       df["Age"].fillna(df["Age"].mean(),inplace=True)
       df.isnull().sum()
       df["Age"].value_counts().sort()
```

```
Out[7]: 0.420    1
        0.750    1
        0.830    1
        0.833    1
        0.917    1
        ..       ..
        70.000   2
        71.000   2
        74.000   1
        80.000   1
        Name: Age, dtype: int32
```

as there are so many null values in cabin we can delete that column

```
In[8]: df.drop(["Cabin",axis=1,inplace=True)
```

```
In[9]: # we replaced 2 categorical data we can use mode to replace the null values
       df["Embarked"].fillna(df["Embarked"].mode()[0],inplace=True)
       df["Embarked"].value_counts().sort()
```

```
Out[9]: Q    77
        S   644
        C   168
        Name: Embarked, dtype: int64
```

```
In[10]: df1.head(10)
```

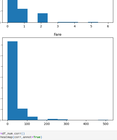
```
Out[10]:   PassengerId  Survived  Pclass                                                 Name     Sex   Age  SibSp  Parch            Ticket     Fare  Cabin  Embarked
         0            1         0       3                             Braund, Mr. Owen Harris    male  22.0      1      0         A/5 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
         5            6         0       3                                    Moran, Mr. James    male  29.7      0      0            330877   8.4583    NaN         Q
         6            7         0       1                             McCarthy, Mr. Timothy J    male  54.0      0      0             17463  51.8625    E46         S
         7            8         0       3                      Palsson, Master. Gosta Leonard    male   2.0      3      1            349909  21.0750    NaN         S
         8            9         1       3   Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)  female  27.0      0      2            347742  11.1333    NaN         S
         9           10         1       2                 Nasser, Mrs. Nicholas (Adele Achem)  female  14.0      1      0            237736  30.0708    NaN         C
```

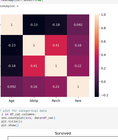
```
In[11]: df1.info()
```

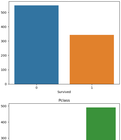
```
Out[11]: <class 'pandas.core.frame.DataFrame'>
         Int64Index: 891 entries, 0 to 890
         Data columns (total 12 columns):
          #   Column       Non-Null Count  Dtype
         ---  ------       --------------  -----
          0   PassengerId  891 non-null    int64
          1   Survived     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
         dtype: float64(2), int64(7), object(3)
         memory usage: 90.5 KB
```

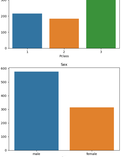
Step-4: data visualization

```
In[12]: df.corr() # df1 "Age", "SibSp", "Parch", "Fare"]] a numerical values
        df1.corr #[["Survived", "Pclass", "Sex", "Ticket", "Cabin", "Embarked"]] # categorical values
        # we can use heatmap to check whether the data is in normal distribution or not
        for i in df.num_columns:
            sns.distplot(df[i])
            plt.show()
            plt.show()
```

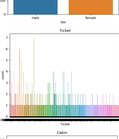




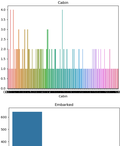




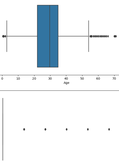

```
In[13]: df1.corr()
        sns.heatmap(df1.corr(),annot=True)
        #boxplot(df1.)
```

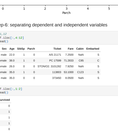


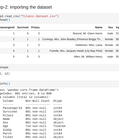

```
In[14]: df1.corr()
        sns.heatmap(df1.corr(),annot=True)
        #boxplot(df1.)
```




```
In[15]: #df1.drop(["Ticket",axis=1)
        #df1.drop(df1.cabin,inplace=True)
        sns.countplot(df1.Sex)
        plt.show()
        plt.show()
```







Step-5: outliers detection

```
In[16]: fig, axes = plt.subplots(2, 3, figsize=(16, 10))
        sns.boxplot(df1[survived], ax=axes[0, 0])
        sns.boxplot(df1[Pclass], ax=axes[0, 1])
        sns.boxplot(df1[Sex], ax=axes[0, 2])
        sns.boxplot(df1[Embarked], ax=axes[1, 0])
        #boxplot(df1.cabin,fig=)
```


Step-6: separating dependent and independent variables

```
In[17]: y=df1.iloc[:,1]
        x=df1.iloc[:,0:12]
        x.head()
```

```
Out[17]:   Sex  Age  SibSp  Parch            Ticket     Fare  Cabin  Embarked
         0    male   22.0      1      0         A/5 21171   7.2500    NaN         S
         1  female   38.0      1      0          PC 17599  71.2833    C85         C
         2  female   26.0      0      0  STON/O2. 3101282   7.9250    NaN         S
         3  female   35.0      1      0            113803  53.1000   C123         S
         4    male   35.0      0      0            373450   8.0500    NaN         S
```

```
In[18]: y.dtype
```

```
Out[18]: dtype('int64')
```

```
In[19]: y.head()
```

```
Out[19]: 0    0
         1    1
         2    1
         3    1
         4    0
         Name: Survived, dtype: int64
```

Step-7: encoding

```
In[20]: from sklearn.preprocessing import LabelEncoder
        le=LabelEncoder()
        x["Sex"]=le.fit_transform(x["Sex"])
        x["Sex"].head()
```

```
Out[20]: 0    1
         1    0
         2    0
         3    0
         4    1
         Name: Sex, dtype: int32
```

```
In[21]: #label encoding for catego data
        df1.Pclass[:] = le.fit_transform(df1.Pclass)
```

```
In[22]: #one hot encoding for catego data
        # df.get_dummies(df[['Sex', 'Ticket', 'Cabin', 'Embarked']])
```

Step-8: splitting into training and testing dataset

```
In[23]: from sklearn.model_selection import train_test_split
        y = df.drop('Survived', axis=1)
        x = df['Survived']
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

```
In[24]: x_train.shape, y_train.shape, x_test.shape, y_test.shape
```

```
Out[24]: ((712, 11), (712, 11), (179,), (179,))
```

Step-9: feature scaling

```
In[25]: from sklearn.preprocessing import StandardScaler
        ss=StandardScaler()
        #df1["Sex"] = ss.fit_transform(df1[["Sex", "Pclass"]])
        #df1.head()
        from sklearn.model_selection import train_test_split
        y = df.drop('Survived', axis=1)
        x = df['Survived']
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

```
In[26]: x_train
```

```
Out[26]:     PassengerId  Pclass                                               Name     Sex   Age  SibSp  Parch            Ticket     Fare  Cabin  Embarked
         331          332       2                                  Partner, Mr. Austen    male  45.0      0      0            28034  10.5000    C124         S
         708          709       1                                  Blackwell, Mr. William    male  42.0000      0      0             36439  26.5500    C77         S
         202          203       3                                       Vendrame, Mr. Luigi    male  31.27500      0      0  STON/O 2. 3101291   6.4958    NaN         S
         783          784       3                                   Johnston, Mr. Hendrik Jaan    male  22.00000      1      1             W./C. 6607  23.4500    NaN         S
         413          414       2                                  Andrews, Miss. Kathleen Mary  female  1.67700      0      0            SC/PARIS  10.5000    NaN         S
         ...          ...     ...                                                         ...     ...    ...    ...    ...                ...     ...    ...       ...
         295          296       1                                   Butt, Major. Archibald Willingham    male  45.0000      0      0            113050  26.5500   B102         S
         278          279       3                                        Carlsson, Mr. Alexander    male  0.00000      0      0            123396   8.0500    NaN         S
         665          666       2                                   Hickman, Mr. Lewis Charles    male  42.0000      2      0             S.O.C. 14879  73.5000    NaN         S
         489          490       3                                   Coutts, Master. Eustace Man    male  1.00000      1      1             C.A. 37671  15.9000    NaN         S
         305          306       1                                   Allison, Master. Hudson Trevor    male  0.92000      1      2             113781  151.5500  C22 C26         S
         ...          ...     ...                                                         ...     ...    ...    ...    ...                ...     ...    ...       ...
         712 rows × 11 columns
```

```
In[27]: y_train
```

```
Out[27]: 331    0
         708    1
         202    0
         783    0
         413    1
         ...    ...
         295    0
         278    0
         665    1
         489    0
         305    0
         Name: Survived, Length: 712, dtype: int64
```