Import all the libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import accuracy_score
import seaborn as sns
```

Load the data

In [2]:
```
data = pd.read_csv("/home/tamanna/Downloads/Titanic.csv")
```

In [3]:
```
data
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298
...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668

418 rows × 12 columns

data preprocessing

In [4]:
```
data.head()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.82
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.00
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.68
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.66
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.28

In [5]:
```
data.shape          # the number of rows and columns present in the data
```

Out[5]:
```
(418, 12)
```

In [6]:
```
data.columns        # Name of the columns present in the data
```

Out[6]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [7]:
```
data.info()         # information about columns : datatype and non-null count
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  418 non-null    int64  
 1   Survived     418 non-null    int64  
 2   Pclass       418 non-null    int64  
 3   Name         418 non-null    object 
 4   Sex          418 non-null    object 
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64  
 7   Parch        418 non-null    int64  
 8   Ticket       418 non-null    object 
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object 
 11  Embarked     418 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [8]:
```
data.describe()           # descriptive statistics of data
```

Out[8]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	F
count	418.000000	418.000000	418.000000	332.000000	418.000000	418.000000	417.000
mean	1100.500000	0.363636	2.265550	30.272590	0.447368	0.392344	35.627
std	120.810458	0.481622	0.841838	14.181209	0.896760	0.981429	55.907
min	892.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.000
25%	996.250000	0.000000	1.000000	21.000000	0.000000	0.000000	7.895
50%	1100.500000	0.000000	3.000000	27.000000	0.000000	0.000000	14.454
75%	1204.750000	1.000000	3.000000	39.000000	1.000000	0.000000	31.500
max	1309.000000	1.000000	3.000000	76.000000	8.000000	9.000000	512.329

dropping the unnecessary columns

In [9]:
```
new_data = data.drop(["PassengerId", "Name", "Ticket","Cabin"], axis = 1)
```

In [10]:
```
new_data
```

Out[10]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	34.5	0	0	7.8292	Q
1	1	3	female	47.0	1	0	7.0000	S
2	0	2	male	62.0	0	0	9.6875	Q
3	0	3	male	27.0	0	0	8.6625	S
4	1	3	female	22.0	1	1	12.2875	S
...	...	...	...	...	...	...	...	...
413	0	3	male	NaN	0	0	8.0500	S
414	1	1	female	39.0	0	0	108.9000	C
415	0	3	male	38.5	0	0	7.2500	S
416	0	3	male	NaN	0	0	8.0500	S
417	0	3	male	NaN	1	1	22.3583	C

418 rows × 8 columns

Checking for null values

In [11]:
```
new_data.isnull().sum()
```

Out[11]:
```
Survived     0
Pclass       0
Sex          0
Age         86
SibSp        0
Parch        0
Fare         1
Embarked     0
dtype: int64
```

filling the mean value in place of null values

In [12]:
```
new_data['Age'].fillna(new_data['Age'].mean(), inplace=True)
```

In [13]:
```
new_data['Fare'].fillna(new_data['Fare'].mean(), inplace=True)
```

In [14]:
```
new_data
```

Out[14]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	34.50000	0	0	7.8292	Q
1	1	3	female	47.00000	1	0	7.0000	S
2	0	2	male	62.00000	0	0	9.6875	Q
3	0	3	male	27.00000	0	0	8.6625	S
4	1	3	female	22.00000	1	1	12.2875	S
...	...	...	...	...	...	...	...	...
413	0	3	male	30.27259	0	0	8.0500	S
414	1	1	female	39.00000	0	0	108.9000	C
415	0	3	male	38.50000	0	0	7.2500	S
416	0	3	male	30.27259	0	0	8.0500	S
417	0	3	male	30.27259	1	1	22.3583	C

418 rows × 8 columns

In [15]:
```
new_data.isnull().sum()
```

Out[15]:
```
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    0
dtype: int64
```

creating the dummy variables

In [16]:
```
new_data.replace({"Sex" : {"male" : 0, "female" : 1}}, inplace = True)
```

In [17]:
```
new_data
```

Out[17]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	0	34.50000	0	0	7.8292	Q
1	1	3	1	47.00000	1	0	7.0000	S
2	0	2	0	62.00000	0	0	9.6875	Q
3	0	3	0	27.00000	0	0	8.6625	S
4	1	3	1	22.00000	1	1	12.2875	S
...	...	...	...	...	...	...	...	...
413	0	3	0	30.27259	0	0	8.0500	S
414	1	1	1	39.00000	0	0	108.9000	C
415	0	3	0	38.50000	0	0	7.2500	S
416	0	3	0	30.27259	0	0	8.0500	S
417	0	3	0	30.27259	1	1	22.3583	C

418 rows × 8 columns

In [18]:
```
new_data.replace({"Embarked" : {"Q" : 0, "S" : 1, "C" : 2}}, inplace = True)
```

In [19]:
```
new_data
```

Out[19]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	0	34.50000	0	0	7.8292	0
1	1	3	1	47.00000	1	0	7.0000	1
2	0	2	0	62.00000	0	0	9.6875	0
3	0	3	0	27.00000	0	0	8.6625	1
4	1	3	1	22.00000	1	1	12.2875	1
...	...	...	...	...	...	...	...	...
413	0	3	0	30.27259	0	0	8.0500	1
414	1	1	1	39.00000	0	0	108.9000	2
415	0	3	0	38.50000	0	0	7.2500	1
416	0	3	0	30.27259	0	0	8.0500	1
417	0	3	0	30.27259	1	1	22.3583	2

418 rows × 8 columns

In [20]:
```
x = new_data.drop(["Survived"], axis = 1)
```

In [21]:
```
print(x)
```

```
     Pclass  Sex       Age  SibSp  Parch      Fare  Embarked
0         3    0  34.50000      0      0    7.8292         0
1         3    1  47.00000      1      0    7.0000         1
2         2    0  62.00000      0      0    9.6875         0
3         3    0  27.00000      0      0    8.6625         1
4         3    1  22.00000      1      1   12.2875         1
..      ...  ...       ...    ...    ...       ...       ...
413       3    0  30.27259      0      0    8.0500         1
414       1    1  39.00000      0      0  108.9000         2
415       3    0  38.50000      0      0    7.2500         1
416       3    0  30.27259      0      0    8.0500         1
417       3    0  30.27259      1      1   22.3583         2

[418 rows x 7 columns]
```

In [22]:
```
y = new_data["Survived"]
```

In [23]:
```
print(y)
```

```
0      0
1      1
2      0
3      0
4      1
      ..
413    0
414    1
415    0
416    0
417    0
Name: Survived, Length: 418, dtype: int64
```

splitting the data into training and testing datasets

In [24]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y,random_state = 2, te
```

In [25]:
```
print(x.shape, x_train.shape, x_test.shape)
```

```
(418, 7) (376, 7) (42, 7)
```

In [26]:
```
model = LogisticRegression()
```

In [27]:
```
model.fit(x_train, y_train)
```

```
/home/tamanna/.local/lib/python3.11/site-packages/sklearn/linear_model/_logis
tic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion
  n_iter_i = _check_optimize_result(
```

Out[27]:
```
▾ LogisticRegression

LogisticRegression()
```

In [28]:
```
# accuracy on training data
x_train_prediction = model.predict(x_train)
training_data_accuracy = accuracy_score(x_train_prediction, y_train)
```

In [29]:
```
training_data_accuracy
```

Out[29]:
```
1.0
```

In [30]:
```
# accuracy on testing data
x_test_prediction = model.predict(x_test)
testing_data_accuracy = accuracy_score(x_test_prediction, y_test)
```

In [31]:
```
testing_data_accuracy
```

Out[31]: 1.0

checking the accuracy on given data

In [37]:
```
input_data = ( 2, 0, 62.00000, 0, 0, 9.6875, 0)
input_data
```

Out[37]: (2, 0, 62.0, 0, 0, 9.6875, 0)

In [38]:
```
data_as_array = np.asarray(input_data)
data_as_array
```

Out[38]: array([2. , 0. , 62. , 0. , 0. , 9.6875, 0.])

In [39]:
```
input_data_reshaped = data_as_array.reshape(1,-1)
input_data_reshaped
```

Out[39]: array([[2. , 0. , 62. , 0. , 0. , 9.6875, 0.]])

In [40]:
```
prediction = model.predict(input_data_reshaped)
print(prediction)

if (prediction[0] == 0):
    print('person has not survived')
else:
    print('person has survived')
```

```
[0]
person has not survived
/home/tamanna/.local/lib/python3.11/site-packages/sklearn/base.py:439: UserWa
rning: X does not have valid feature names, but LogisticRegression was fitted
with feature names
  warnings.warn(
```