In [2]:

```
#!pip install pyforest ----inbuilt pandas,matplotlib and seaborndata

import pyforest
```

In [5]:

```
data=pd.read_csv('Titanic-Train-Data.csv')
data
```

<IPython.core.display.Javascript object>

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	
...	...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	

891 rows × 12 columns

In [6]:
```
data.shape #check the how many rows and colums
```

Out[6]:
```
(891, 12)
```

In [7]:
```
data.isna().sum()  #check the how many null values present data set(isna()---isnu
```

Out[7]:
```
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
```

In [8]:
```
data.describe()
```

Out[8]:

	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

In [9]:
```
data.info() #data types findout
```

```
<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
```

In [10]:
```
data.dtypes  #separate datatype findout
```

Out[10]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [11]:
```
#Import label encoder
from sklearn import preprocessing #sklearnmodule

#label_encoder object knows how to understand word labels.
label_encoder=preprocessing.LabelEncoder()

#encode labels in column 'Gender'
data['Sex']=label_encoder.fit_transform(data['Sex'])

data['Sex'].value_counts() #converting male as 1 and female as 2
```

Out[11]:
```
1    577
0    314
Name: Sex, dtype: int64
```

In [12]:
```
data
```

Out[12]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	1	0	3	Braund, Mr. Owen Harris	1	22.0	1	0	A/5 21171	7.2500	Na
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	38.0	1	0	PC 17599	71.2833	C8
2	3	1	3	Heikkinen, Miss. Laina	0	26.0	0	0	STON/O2. 3101282	7.9250	Na
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	35.0	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	1	35.0	0	0	373450	8.0500	Na
...	...	...	...	...	...	...	...	...	...	...	.
886	887	0	2	Montvila, Rev. Juozas	1	27.0	0	0	211536	13.0000	Na
887	888	1	1	Graham, Miss. Margaret Edith	0	19.0	0	0	112053	30.0000	B4
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	0	NaN	1	2	W./C. 6607	23.4500	Na
889	890	1	1	Behr, Mr. Karl Howell	1	26.0	0	0	111369	30.0000	C14
890	891	0	3	Dooley, Mr. Patrick	1	32.0	0	0	370376	7.7500	Na

891 rows × 12 columns

In [27]:
```
data
```

Out[27]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Cabin	Embarke
0	1	0	3	Braund, Mr. Owen Harris	1	22.0	1	0	7.2500	NaN	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	38.0	1	0	71.2833	C85	
2	3	1	3	Heikkinen, Miss. Laina	0	26.0	0	0	7.9250	NaN	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	35.0	1	0	53.1000	C123	
4	5	0	3	Allen, Mr. William Henry	1	35.0	0	0	8.0500	NaN	
...	...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	1	27.0	0	0	13.0000	NaN	
887	888	1	1	Graham, Miss. Margaret Edith	0	19.0	0	0	30.0000	B42	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	0	NaN	1	2	23.4500	NaN	
889	890	1	1	Behr, Mr. Karl Howell	1	26.0	0	0	30.0000	C148	
890	891	0	3	Dooley, Mr. Patrick	1	32.0	0	0	7.7500	NaN	

891 rows × 11 columns

In [28]:
```
data=data.drop(['Cabin','Name'],axis=1)
data
```

Out[28]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	1	22.0	1	0	7.2500	S
1	2	1	1	0	38.0	1	0	71.2833	C
2	3	1	3	0	26.0	0	0	7.9250	S
3	4	1	1	0	35.0	1	0	53.1000	S
4	5	0	3	1	35.0	0	0	8.0500	S
...	...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	0	0	13.0000	S
887	888	1	1	0	19.0	0	0	30.0000	S
888	889	0	3	0	NaN	1	2	23.4500	S
889	890	1	1	1	26.0	0	0	30.0000	C
890	891	0	3	1	32.0	0	0	7.7500	Q

891 rows × 9 columns

In [29]:
```
data['Age'].median()
```

Out[29]: 28.0

In [30]:
```
data['Age']=data['Age'].fillna(value=28) #filling null values
data
```

Out[30]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	1	22.0	1	0	7.2500	S
1	2	1	1	0	38.0	1	0	71.2833	C
2	3	1	3	0	26.0	0	0	7.9250	S
3	4	1	1	0	35.0	1	0	53.1000	S
4	5	0	3	1	35.0	0	0	8.0500	S
...	...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	0	0	13.0000	S
887	888	1	1	0	19.0	0	0	30.0000	S
888	889	0	3	0	28.0	1	2	23.4500	S
889	890	1	1	1	26.0	0	0	30.0000	C
890	891	0	3	1	32.0	0	0	7.7500	Q

891 rows × 9 columns

In [31]:
```
data['Age'].isna().sum()
```

Out[31]:
```
0
```

In [32]:
```
data.isna().sum()
```

Out[32]:
```
PassengerId    0
Survived       0
Pclass         0
Sex            0
Age            0
SibSp          0
Parch          0
Fare           0
Embarked       2
dtype: int64
```

In [33]:
```
data['Embarked'].value_counts()
```

Out[33]:
```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

In [34]:
```
g=data.groupby('Survived')
g['Embarked'].value_counts() #group by multiple colums
```

Out[34]:
```
Survived  Embarked
0         S           427
          C            75
          Q            47
1         S           217
          C            93
          Q            30
Name: Embarked, dtype: int64
```

In [35]:
```
data['Embarked']=data['Embarked'].fillna(value='S')
data
```

Out[35]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	1	22.0	1	0	7.2500	S
1	2	1	1	0	38.0	1	0	71.2833	C
2	3	1	3	0	26.0	0	0	7.9250	S
3	4	1	1	0	35.0	1	0	53.1000	S
4	5	0	3	1	35.0	0	0	8.0500	S
...	...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	0	0	13.0000	S
887	888	1	1	0	19.0	0	0	30.0000	S
888	889	0	3	0	28.0	1	2	23.4500	S
889	890	1	1	1	26.0	0	0	30.0000	C
890	891	0	3	1	32.0	0	0	7.7500	Q

891 rows × 9 columns

In [36]:
```
#Import label encoder
from sklearn import preprocessing #sklearnmodule

#label_encoder object knows how to understand word labels.
label_encoder=preprocessing.LabelEncoder()

#encode labels in column 'Gender'
data['Embarked']=label_encoder.fit_transform(data['Embarked'])

data['Embarked'].value_counts()
```

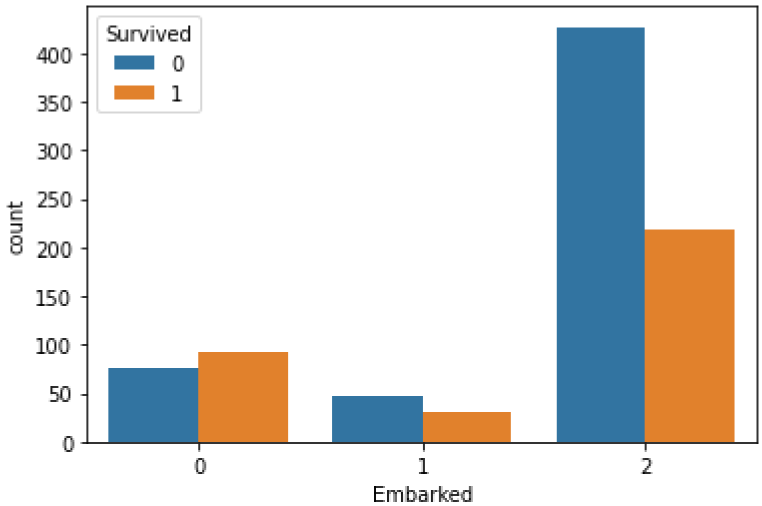
Out[36]:
```
2    646
0    168
1     77
Name: Embarked, dtype: int64
```

In [37]:
```
sns.countplot(data['Embarked'],hue=data['Survived'])
plt.show()
```

```
<IPython.core.display.Javascript object>

G:\Users\HP\Downloads\lib\site-packages\seaborn\_decorators.py:36: FutureWarnin
g: Pass the following variable as a keyword arg: x. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without a
n explicit keyword will result in an error or misinterpretation.
  warnings.warn(

<IPython.core.display.Javascript object>
```



In [39]:
```
data['Embarked'].value_counts()
```

Out[39]:
```
2    646
0    168
1     77
Name: Embarked, dtype: int64
```

In [40]:
```
data
```

Out[40]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	1	22.0	1	0	7.2500	2
1	2	1	1	0	38.0	1	0	71.2833	0
2	3	1	3	0	26.0	0	0	7.9250	2
3	4	1	1	0	35.0	1	0	53.1000	2
4	5	0	3	1	35.0	0	0	8.0500	2
...	...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	0	0	13.0000	2
887	888	1	1	0	19.0	0	0	30.0000	2
888	889	0	3	0	28.0	1	2	23.4500	2
889	890	1	1	1	26.0	0	0	30.0000	0
890	891	0	3	1	32.0	0	0	7.7500	1

891 rows × 9 columns

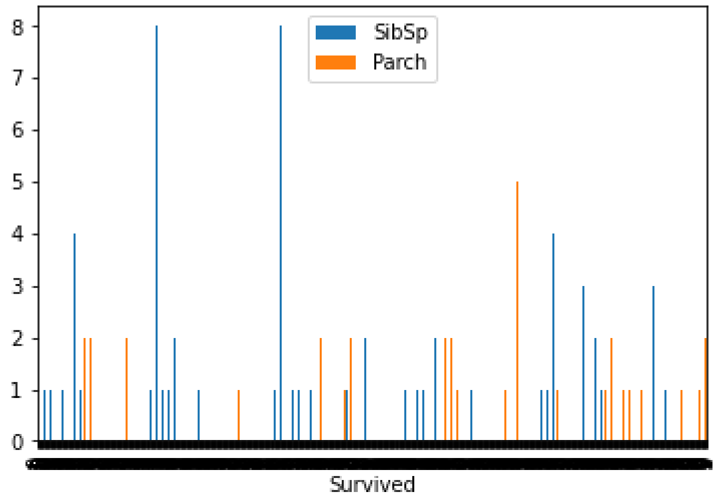
In [41]:
```
data.corr() #corealation---corr
```

Out[41]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fa
PassengerId	1.000000	-0.005007	-0.035144	0.042939	0.034212	-0.057527	-0.001652	0.0126
Survived	-0.005007	1.000000	-0.338481	-0.543351	-0.064910	-0.035322	0.081629	0.2573
Pclass	-0.035144	-0.338481	1.000000	0.131900	-0.339898	0.083081	0.018443	-0.5495
Sex	0.042939	-0.543351	0.131900	1.000000	0.081163	-0.114631	-0.245489	-0.1823
Age	0.034212	-0.064910	-0.339898	0.081163	1.000000	-0.233296	-0.172482	0.0966
SibSp	-0.057527	-0.035322	0.083081	-0.114631	-0.233296	1.000000	0.414838	0.1596
Parch	-0.001652	0.081629	0.018443	-0.245489	-0.172482	0.414838	1.000000	0.2162
Fare	0.012658	0.257307	-0.549500	-0.182333	0.096688	0.159651	0.216225	1.0000
Embarked	0.013128	-0.167675	0.162098	0.108262	-0.018754	0.068230	0.039798	-0.2247

In [44]:
```
data.plot(x="Survived",y=['SibSp','Parch'],kind='bar')
plt.show()
```

```
<IPython.core.display.Javascript object>
```



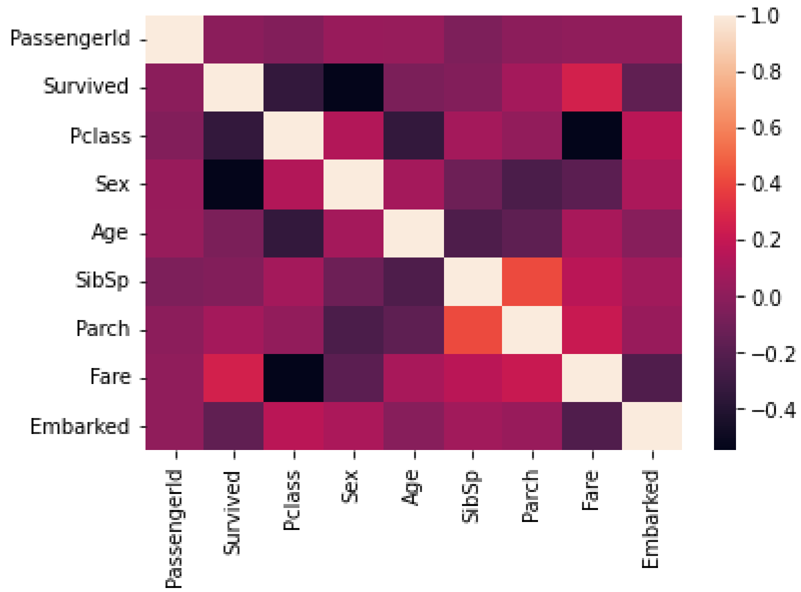
In [45]:
```
correlation=data.corr()
correlation['Survived'].sort_values(ascending=False)
```

Out[45]:
```
Survived       1.000000
Fare           0.257307
Parch          0.081629
PassengerId   -0.005007
SibSp         -0.035322
Age           -0.064910
Embarked      -0.167675
Pclass        -0.338481
Sex           -0.543351
Name: Survived, dtype: float64
```

In [46]:
```
sns.heatmap(data.corr())
```

<IPython.core.display.Javascript object>

Out[46]: <AxesSubplot:>



In [47]:
```
correlation['Fare'].sort_values(ascending=False)
correlation['Fare']
```

Out[47]:
```
PassengerId    0.012658
Survived       0.257307
Pclass        -0.549500
Sex           -0.182333
Age            0.096688
SibSp          0.159651
Parch          0.216225
Fare           1.000000
Embarked      -0.224719
Name: Fare, dtype: float64
```

In [48]:
```
data.head()
```

Out[48]:

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

In [54]:
```
data['family']=data['SibSp']+data['Parch']+1
data
```

Out[54]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	family
0	1	0	3	1	22.0	1	0	7.2500	2	2
1	2	1	1	0	38.0	1	0	71.2833	0	2
2	3	1	3	0	26.0	0	0	7.9250	2	1
3	4	1	1	0	35.0	1	0	53.1000	2	2
4	5	0	3	1	35.0	0	0	8.0500	2	1
...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	0	0	13.0000	2	1
887	888	1	1	0	19.0	0	0	30.0000	2	1
888	889	0	3	0	28.0	1	2	23.4500	2	4
889	890	1	1	1	26.0	0	0	30.0000	0	1
890	891	0	3	1	32.0	0	0	7.7500	1	1

891 rows × 10 columns

In [57]:
```
data=data.drop(['SibSp','Parch'],axis=1)
data
```

Out[57]:

	PassengerId	Survived	Pclass	Sex	Age	Fare	Embarked	family
0	1	0	3	1	22.0	7.2500	2	2
1	2	1	1	0	38.0	71.2833	0	2
2	3	1	3	0	26.0	7.9250	2	1
3	4	1	1	0	35.0	53.1000	2	2
4	5	0	3	1	35.0	8.0500	2	1
...	...	...	...	...	...	...	...	...
886	887	0	2	1	27.0	13.0000	2	1
887	888	1	1	0	19.0	30.0000	2	1
888	889	0	3	0	28.0	23.4500	2	4
889	890	1	1	1	26.0	30.0000	0	1
890	891	0	3	1	32.0	7.7500	1	1

891 rows × 8 columns

In [59]:
```
data=data.drop(['PassengerId'],axis=1)
data
```

Out[59]:

	Survived	Pclass	Sex	Age	Fare	Embarked	family
0	0	3	1	22.0	7.2500	2	2
1	1	1	0	38.0	71.2833	0	2
2	1	3	0	26.0	7.9250	2	1
3	1	1	0	35.0	53.1000	2	2
4	0	3	1	35.0	8.0500	2	1
...	...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	2	1
887	1	1	0	19.0	30.0000	2	1
888	0	3	0	28.0	23.4500	2	4
889	1	1	1	26.0	30.0000	0	1
890	0	3	1	32.0	7.7500	1	1

891 rows × 7 columns

In [60]:
```
data=data.drop(['Embarked'],axis=1)
data
```

Out[60]:

	Survived	Pclass	Sex	Age	Fare	family
0	0	3	1	22.0	7.2500	2
1	1	1	0	38.0	71.2833	2
2	1	3	0	26.0	7.9250	1
3	1	1	0	35.0	53.1000	2
4	0	3	1	35.0	8.0500	1
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	1
887	1	1	0	19.0	30.0000	1
888	0	3	0	28.0	23.4500	4
889	1	1	1	26.0	30.0000	1
890	0	3	1	32.0	7.7500	1

891 rows × 6 columns

In [61]:
```
x=data.drop('Survived',axis=1).values
y=data['Survived'].values
```

In [63]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

In [64]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=10
```

In [65]:
```
from sklearn.metrics import accuracy_score
```

In [66]:
```
lr=LogisticRegression()
lr.fit(x_train,y_train) #sending data to train 70%
lrpred=lr.predict(x_test)
```

In [67]:
```
accuracy_score(y_test,lrpred)
```

Out[67]: 0.7910447761194029

In [71]:
```
from sklearn.model_selection import GridSearchCV

#creating the hyperparameter grid
c_space=np.logspace(-5,815)
param_grid={'C':c_space}

#Instantiating the GridsearchCV object
logreg_cv=GridSearchCV(lr,param_grid,cv=5)

logreg_cv.fit(x_train,y_train)

#print the tuned parameters and score
print("Tuned Logistic Regression Parameters: {}".format(logreg_cv.best_params_))
print("Best score is {}".format(logreg_cv.best_score_))
```

```
<IPython.core.display.Javascript object>

G:\Users\HP\Downloads\lib\site-packages\numpy\core\function_base.py:277: Runtim
eWarning: overflow encountered in power
  return _nx.power(base, y)

Tuned Logistic Regression Parameters: {'C': 542867543932.3859}
Best score is 0.8042193548387097
```

In [ ]: