In [1]:
```python
## import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```python
#import train dataset
train=pd.read_csv('train_1.csv')
train.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]:
```python
train.shape
```

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

In [4]:
```python
train.describe()
```

Out[4]:

	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 [5]:
```python
train.isnull().sum()
```

Out[5]:
```
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 [6]:
```python
### import test dataset

test=pd.read_csv('test_1.csv')
```

In [7]:
```python
test.head()
```

Out[7]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [8]:
```python
test.shape
```

Out[8]:
```
(418, 11)
```

In [9]:
```python
test.describe()
```

Out[9]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	3.000000	76.000000	8.000000	9.000000	512.329200

In [10]:
```python
test.isnull().sum()
```

Out[10]:
```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [11]:
```python
male=train.loc[train.Sex=='male']['Survived']
males=sum(male)/len(male)
print("%of mens are survived",males)
```

```
%of mens are survived 0.18890814558058924
```

In [12]:
```python
female=train.loc[train.Sex=='female']['Survived']
female=sum(female)/len(female)
female
```

Out[12]:
```
0.7420382165605095
```

In [13]:
```python
train.Sex.value_counts()
```

Out[13]:
```
male      577
female    314
Name: Sex, dtype: int64
```

In [14]:
```python
train['Embarked'].value_counts()
```

Out[14]:
```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

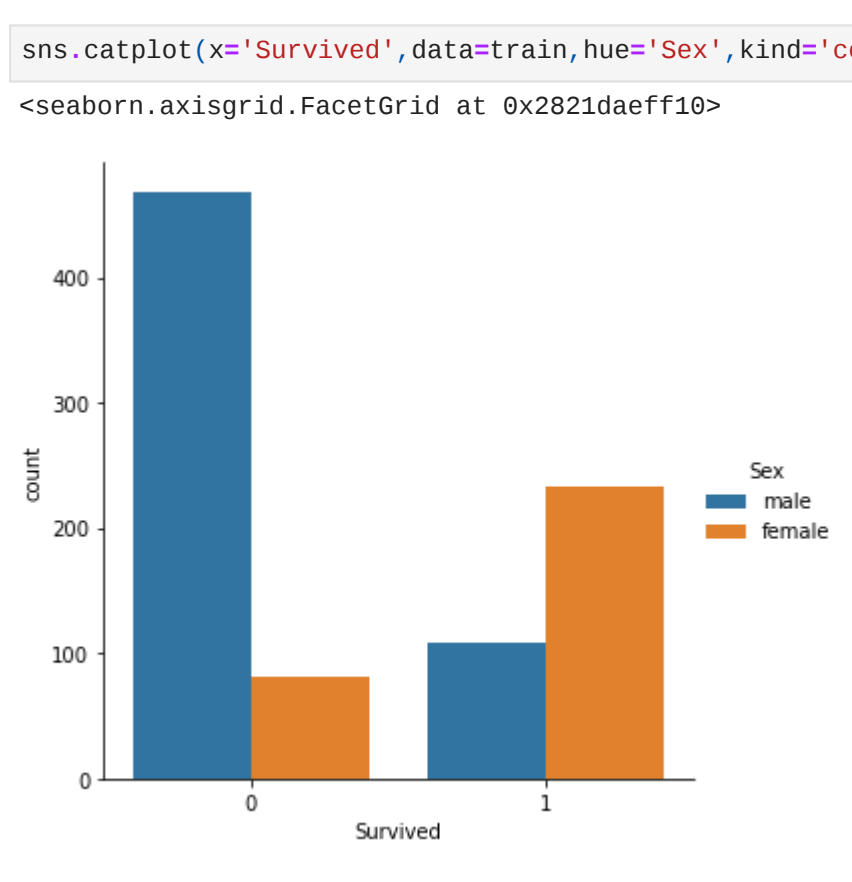
In [15]:
```python
train[['Embarked','Survived']].groupby(['Embarked']).mean().sort_values(by='Survived').reset_index()
```

Out[15]:

	Embarked	Survived
0	S	0.336957
1	Q	0.389610
2	C	0.553571

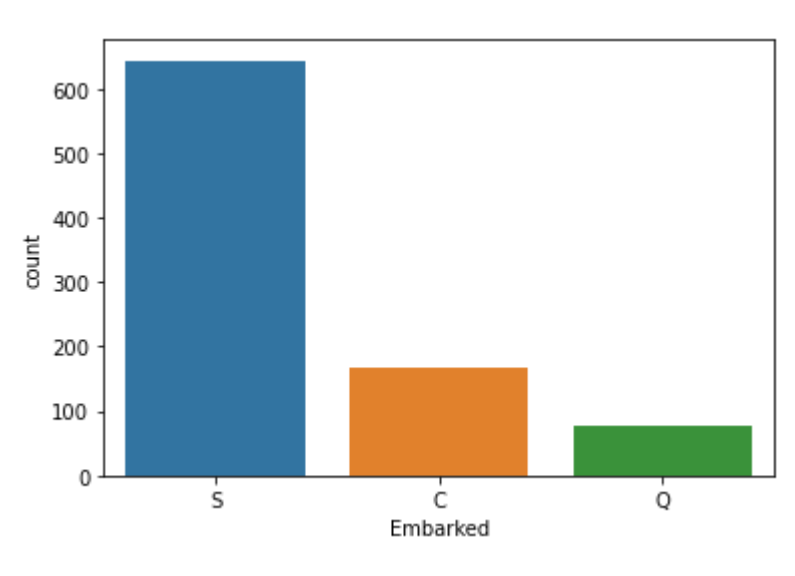
In [16]:
```python
sns.catplot(x='Survived',data=train,hue='Sex',kind='count')
```

Out[16]:
```
<seaborn.axisgrid.FacetGrid at 0x2821daeff10>
```



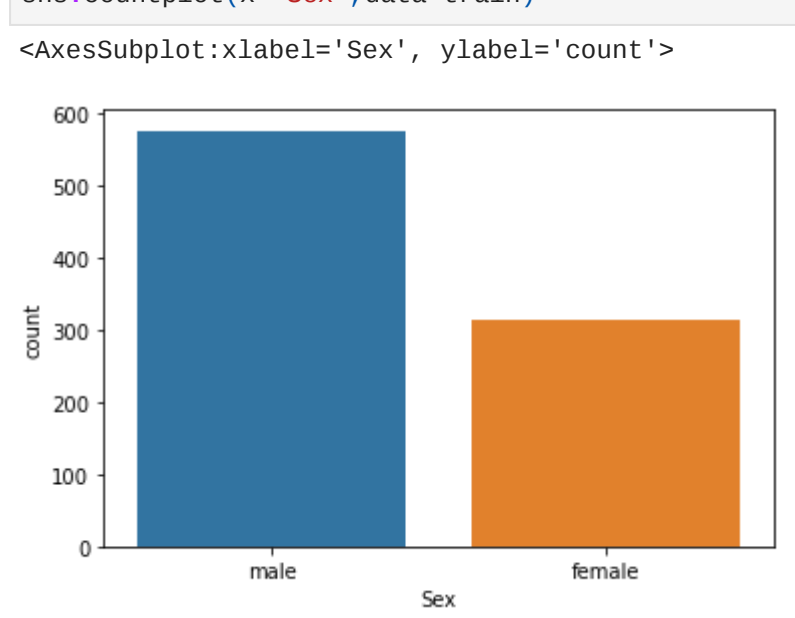
In [17]:
```python
sns.countplot(x='Embarked',data=train)
```

Out[17]:
```
<AxesSubplot:xlabel='Embarked', ylabel='count'>
```



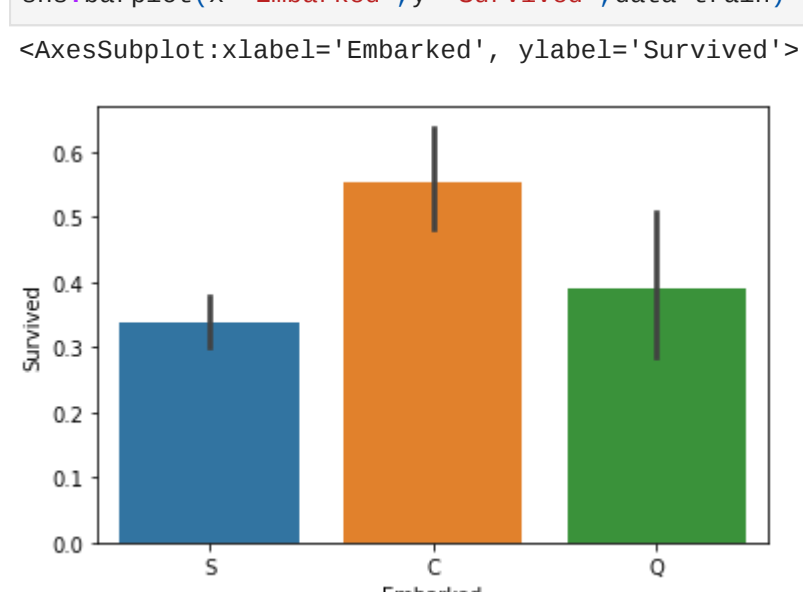
In [18]:
```python
sns.countplot(x='Sex',data=train)
```

Out[18]:
```
<AxesSubplot:xlabel='Sex', ylabel='count'>
```



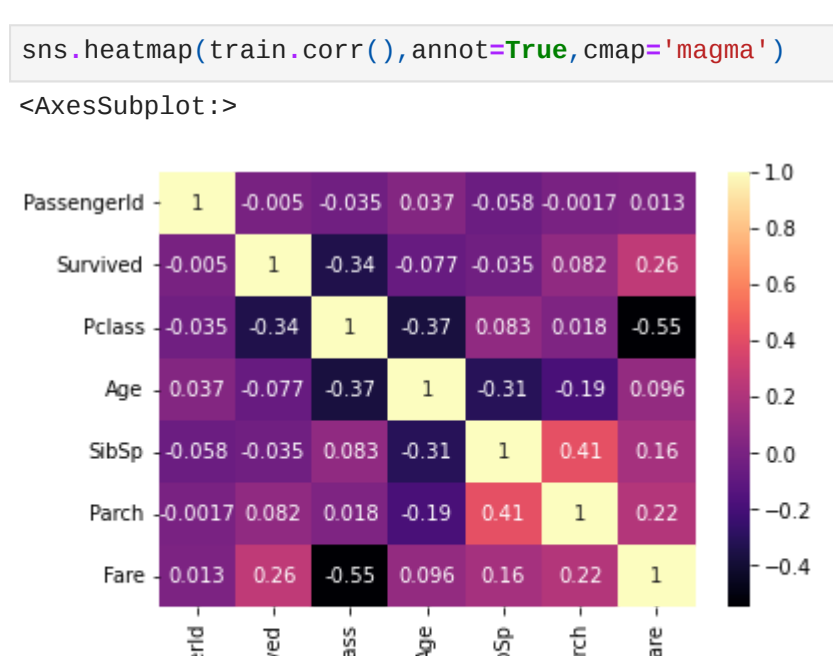
In [19]:
```python
sns.barplot(x='Embarked',y='Survived',data=train)
```

Out[19]:
```
<AxesSubplot:xlabel='Embarked', ylabel='Survived'>
```



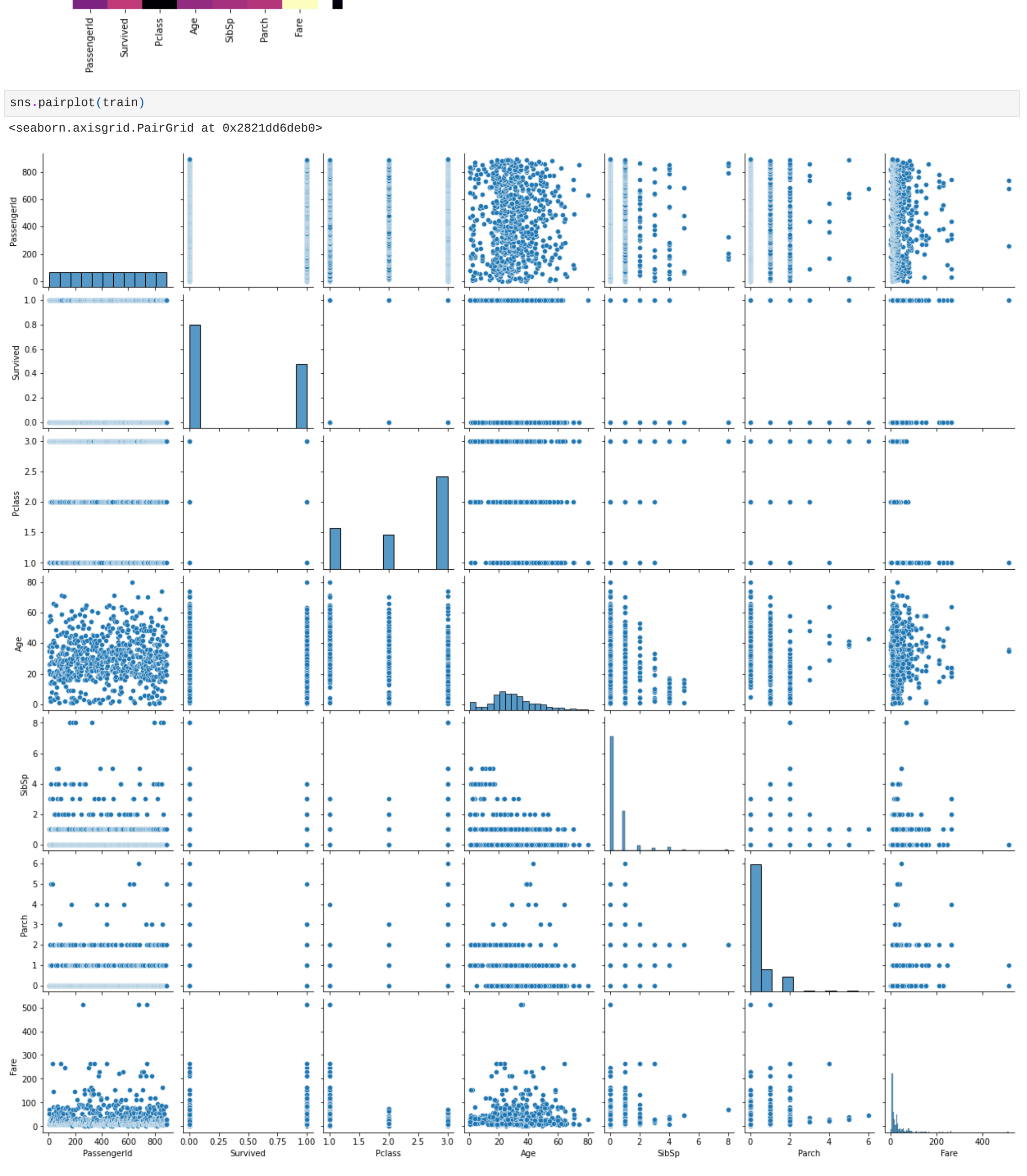
In [20]:
```python
sns.heatmap(train.corr(),annot=True,cmap='magma')
```

Out[20]:
```
<AxesSubplot:>
```



In [21]:
```python
sns.pairplot(train)
```

Out[21]:
```
<seaborn.axisgrid.PairGrid at 0x2821dd6deb0>
```



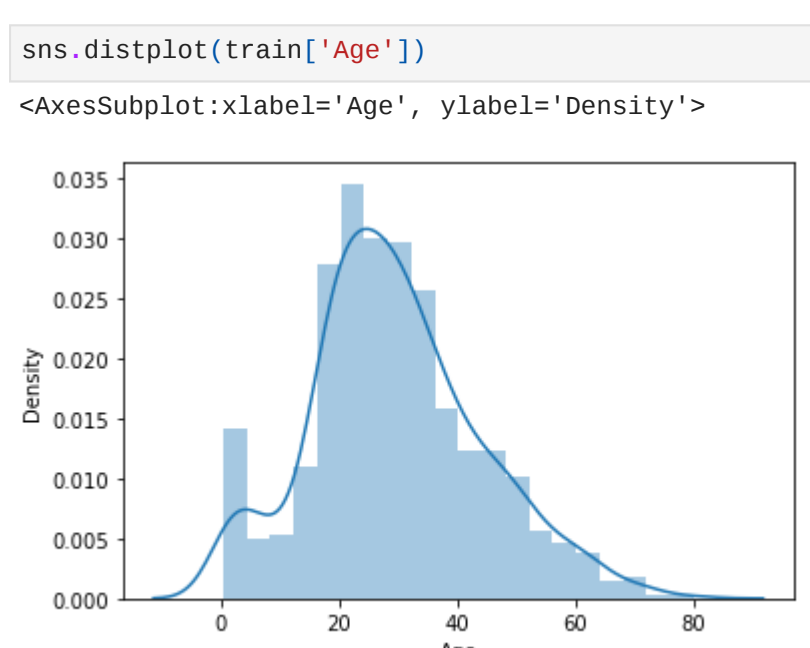
In [22]:
```python
test.corr()
```

Out[22]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.026751	-0.034102	0.003818	0.043080	0.008211
Pclass	-0.026751	1.000000	-0.492143	0.001087	0.018721	-0.577147
Age	-0.034102	-0.492143	1.000000	-0.091587	-0.061249	0.337932
SibSp	0.003818	0.001087	-0.091587	1.000000	0.306895	0.171539
Parch	0.043080	0.018721	-0.061249	0.306895	1.000000	0.230046
Fare	0.008211	-0.577147	0.337932	0.171539	0.230046	1.000000

In [23]:
```python
sns.distplot(train['Age'])
```

Out[23]:
```
<AxesSubplot:xlabel='Age', ylabel='Density'>
```



In [24]:
```python
### treat the missing values
train.drop('Cabin',axis=1,inplace=True)
```

In [25]:
```python
train.isnull().sum()
```

Out[25]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Embarked         2
dtype: int64
```

In [26]:
```python
train['Age'].fillna(train['Age'].median(),inplace=True)
train.Embarked.fillna(method='ffill',inplace=True)
```

In [27]:
```python
test.drop('Cabin',axis=1,inplace=True)
test['Age'].fillna(test['Age'].median(),inplace=True)
test.Embarked.fillna(method='bfill',inplace=True)
test.Fare.fillna(method='ffill',inplace=True)
```

In [28]:
```python
test.isnull().sum()
```

Out[28]:
```
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [29]:
```python
train_df=pd.get_dummies(train,columns=["Pclass","Embarked","Sex"],drop_first=True)
```

In [30]:
```python
train_df.head()
```

Out[30]:

	PassengerId	Survived	Name	Age	SibSp	Parch	Ticket	Fare	Pclass_2	Pclass_3	Embarked_Q	Embarked_S	Sex_male
0	1	0	Braund, Mr. Owen Harris	22.0	1	0	A/5 21171	7.2500	0	1	0	1	1
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	38.0	1	0	PC 17599	71.2833	0	0	0	0	0
2	3	1	Heikkinen, Miss. Laina	26.0	0	0	STON/O2. 3101282	7.9250	0	1	0	1	0
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	35.0	1	0	113803	53.1000	0	0	0	1	0
4	5	0	Allen, Mr. William Henry	35.0	0	0	373450	8.0500	0	1	0	1	1

In [31]:
```python
test_df=pd.get_dummies(test,columns=["Pclass","Embarked","Sex"],drop_first=True)
test_df
```

Out[31]:

	PassengerId	Name	Age	SibSp	Parch	Ticket	Fare	Pclass_2	Pclass_3	Embarked_Q	Embarked_S	Sex_male
0	892	Kelly, Mr. James	34.5	0	0	330911	7.8292	0	1	1	0	1
1	893	Wilkes, Mrs. James (Ellen Needs)	47.0	1	0	363272	7.0000	0	1	0	1	0
2	894	Myles, Mr. Thomas Francis	62.0	0	0	240276	9.6875	1	0	1	0	1
3	895	Wirz, Mr. Albert	27.0	0	0	315154	8.6625	0	1	0	1	1
4	896	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	22.0	1	1	3101298	12.2875	0	1	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...
413	1305	Spector, Mr. Woolf	27.0	0	0	A.5. 3236	8.0500	0	1	0	1	1
414	1306	Oliva y Ocana, Dona. Fermina	39.0	0	0	PC 17758	108.9000	0	0	0	0	0
415	1307	Saether, Mr. Simon Sivertsen	38.5	0	0	SOTON/O.Q. 3101262	7.2500	0	1	0	1	1
416	1308	Ware, Mr. Frederick	27.0	0	0	359309	8.0500	0	1	0	1	1
417	1309	Peter, Master. Michael J	27.0	1	1	2668	22.3583	0	1	0	0	1

418 rows × 12 columns

Train and Test Split

In [32]:
```python
x_train=train_df.drop(['PassengerId','Survived','Name','Ticket'],axis=1)
y_train=train_df['Survived']
```

In [33]:
```python
x_test=test_df.drop(['PassengerId','Name','Ticket'],axis=1)
```

In [34]:
```python
seed=10
num_trees=30
### import models
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
kfold = model_selection.KFold(n_splits=10, random_state=seed, shuffle=True)
```

LogisticRegression

In [35]:
```python
logreg=LogisticRegression()
logreg.fit(x_train,y_train)
y_pre=logreg.score(x_train,y_train)
y_pre
```

Out[35]:
```
0.8080808080808081
```

In [36]:
```python
pre=logreg.predict(x_test)
pre
```

Out[36]:
```
array([0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
       0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0,
       1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0],
      dtype=int64)
```

In [37]:
```python
 model_selection.cross_val_score(logreg, x_train,y_train, cv=kfold).mean()
```

Out[37]:
```
0.8024469413233458
```

Random Forest Classifier

In [38]:
```python
random_forest=RandomForestClassifier(n_estimators=num_trees,max_depth=5,random_state=0)
random_forest.fit(x_train,y_train)
y_pred=random_forest.score(x_train,y_train)
y_pred
```

Out[38]:
```
0.8507295173961841
```

In [39]:
```python
predict=random_forest.predict(x_test)
predict
```

Out[39]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1,
       1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0,
       1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
       0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
       0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
       1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0],
      dtype=int64)
```

In [40]:
```python
results = model_selection.cross_val_score(random_forest, x_train,y_train, cv=kfold)
```

In [41]:
```python
results.mean()
```

Out[41]:
```
0.8147565543071161
```

AdaBoostClassifier

In [42]:
```python
from sklearn.ensemble import AdaBoostClassifier
ada_boost = AdaBoostClassifier(n_estimators=num_trees, random_state=0)
```

In [43]:
```python
ada_boost.fit(x_train,y_train)
y_predict=ada_boost.score(x_train,y_train)
y_predict
```

Out[43]:
```
0.8226711560044894
```

In [44]:
```python
pred=ada_boost.predict(x_test)
pred
```

Out[44]:
```
array([0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1,
       1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
       0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0,
       1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0],
      dtype=int64)
```

In [45]:
```python
results = model_selection.cross_val_score(ada_boost, x_train,y_train, cv=kfold)
results.mean()
```

Out[45]:
```
0.8035580524344569
```

In [ ]: