In [104]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.preprocessing import LabelEncoder,StandardScaler
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report

import warnings
warnings.filterwarnings('ignore')
```

In [105]:
```
data = pd.read_csv('SalaryData_Train.csv')
data.head()
```

Out[105]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex
0	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male
1	50	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male
2	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male
3	53	Private	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male
4	28	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female

In [106]:
```
data.shape
```

Out[106]:
```
(30161, 14)
```

In [107]:
```
data.isna().sum()
```

Out[107]:
```
age             0
workclass       0
education       0
educationno     0
maritalstatus   0
occupation      0
relationship    0
race            0
sex             0
capitalgain     0
capitalloss     0
hoursperweek    0
native          0
Salary          0
dtype: int64
```

In [108]:
```
data.dtypes
```

Out[108]:
```
age              int64
workclass       object
education       object
educationno      int64
maritalstatus   object
occupation      object
relationship    object
race            object
sex             object
capitalgain      int64
capitalloss      int64
hoursperweek     int64
native          object
Salary          object
dtype: object
```

In [109]:
```
data.describe()
```

Out[109]:

	age	educationno	capitalgain	capitalloss	hoursperweek
count	30161.000000	30161.000000	30161.000000	30161.000000	30161.000000
mean	38.438115	10.121316	1092.044064	88.302311	40.931269
std	13.134830	2.550037	7406.466611	404.121321	11.980182
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	47.000000	13.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	4356.000000	99.000000

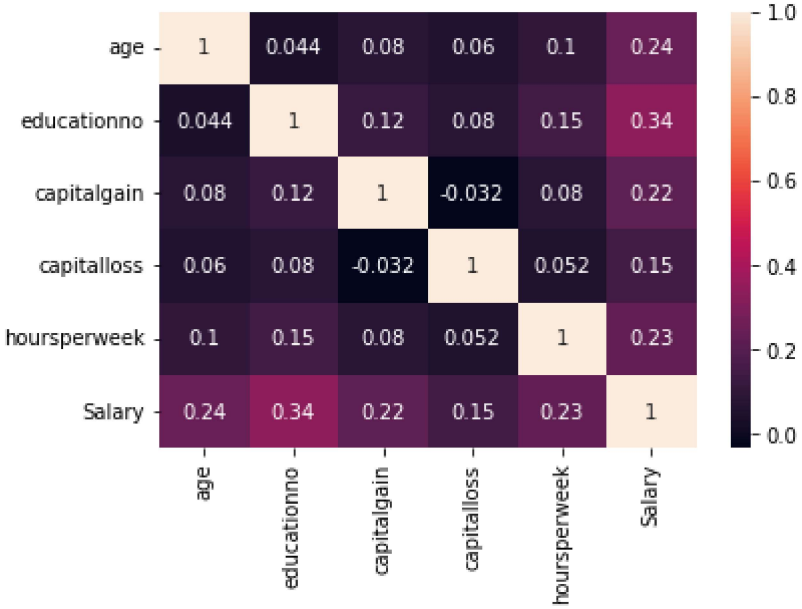
In [110]:
```
data.describe(include='object')
```

Out[110]:

	workclass	education	maritalstatus	occupation	relationship	race	sex	native	Salary
count	30161	30161	30161	30161	30161	30161	30161	30161	30161
unique	7	16	7	14	6	5	2	40	2
top	Private	HS-grad	Married-civ-spouse	Prof-specialty	Husband	White	Male	United-States	<=50K
freq	22285	9840	14065	4038	12463	25932	20380	27504	22653

In [111]:
```
le = LabelEncoder()
data['Salary'] = le.fit_transform(data['Salary'])
```

In [112]:
```
sns.heatmap(data.corr(),annot = True)
plt.show()
```



In [113]:
```
data.drop('education',axis=1,inplace=True)
data.drop('race',axis=1,inplace=True)
data.drop('relationship',axis=1,inplace=True)
```

In [114]:
```
data['workclass'] = le.fit_transform(data['workclass'])
data['maritalstatus'] = le.fit_transform(data['maritalstatus'])
data['occupation'] = le.fit_transform(data['occupation'])
data['native'] = le.fit_transform(data['native'])
data['sex'] = le.fit_transform(data['sex'])
```

In [115]:
```
X = data.iloc[:,:-1]
y = data.iloc[:,-1]
```

In [116]:
```
std = StandardScaler()
std = std.fit_transform(X)
x_scaled = pd.DataFrame(std,columns = X.columns)
```

In [117]:
```
t,y_train,y_test = train_test_split(x_scaled,y,test_size=0.25,random_state=12,str
```

In [118]:
```
x_train.shape,y_train.shape
```

Out[118]:
```
((22620, 10), (22620,))
```

In [119]:
```
x_test.shape,y_test.shape
```

Out[119]:
```
((7541, 10), (7541,))
```

In [120]:
```
gnb = GaussianNB()
gnb.fit(x_train,y_train)
```

Out[120]:
```
GaussianNB()
```

In [121]:
```
y_pred = gnb.predict(x_test)
```

In [122]:
```
accuracy_score(y_test,y_pred)
```

Out[122]:
```
0.7947221853865535
```

In [123]:
```
confusion_matrix(y_test,y_pred)
```

Out[123]:
```
array([[5406,  258],
       [1290,  587]], dtype=int64)
```

In [124]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.81      0.95      0.87      5664
           1       0.69      0.31      0.43      1877

    accuracy                           0.79      7541
   macro avg       0.75      0.63      0.65      7541
weighted avg       0.78      0.79      0.76      7541
```

In [125]:
```
test_data = pd.read_csv('SalaryData_Test.csv')
test_data.head()
```

Out[125]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	ca
0	25	Private	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	
1	38	Private	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	
2	28	Local-gov	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	
3	44	Private	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	
4	34	Private	10th	6	Never-married	Other-service	Not-in-family	White	Male	

In [126]:
```
test_data.drop('education',axis=1,inplace=True)
test_data.drop('race',axis=1,inplace=True)
test_data.drop('relationship',axis=1,inplace=True)
```

In [127]:
```
test_data['workclass'] = le.fit_transform(test_data['workclass'])
test_data['maritalstatus'] = le.fit_transform(test_data['maritalstatus'])
test_data['occupation'] = le.fit_transform(test_data['occupation'])
test_data['native'] = le.fit_transform(test_data['native'])
test_data['sex'] = le.fit_transform(test_data['sex'])
test_data['Salary'] = le.fit_transform(test_data['Salary'])
```

In [128]:
```
X = test_data.iloc[:,:-1]
y = test_data.iloc[:,-1]
```

In [129]:
```
std = StandardScaler()
std = std.fit_transform(X)
X_scale = pd.DataFrame(std,columns=X.columns)
```

In [131]:
```
y_pred = gnb.predict(X)
```

In [132]:
```
print('accuracy',accuracy_score(y,y_pred))
print('\nconfusion_matrix\n',confusion_matrix(y,y_pred))
print('\nclassification report','\n',classification_report(y,y_pred))
```

```
accuracy 0.7753652058432935

confusion_matrix
 [[10536   824]
 [ 2559  1141]]

classification report
               precision    recall  f1-score   support

           0       0.80      0.93      0.86     11360
           1       0.58      0.31      0.40      3700

    accuracy                           0.78     15060
   macro avg       0.69      0.62      0.63     15060
weighted avg       0.75      0.78      0.75     15060
```