Naive Bayes Q1 (Salary Data)

Prepare a classification model using Naive Bayes for salary data

Data Description:

age -- age of a person
workclass -- A work class is a grouping of work
education -- Education of an individuals
maritalstatus -- Marital status of an individulas
occupation -- occupation of an individuals
relationship --
race -- Race of an Individual
sex -- Gender of an Individual
capitalgain -- profit received from the sale of an investment
capitalloss -- A decrease in the value of a capital asset
hoursperweek -- number of hours work per week
native -- Native of an individual
Salary -- salary of an individual

1. Import Libs

In [1]:

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')

from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from mlxtend.plotting import plot_decision_regions
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix
from matplotlib.colors import ListedColormap
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
```

2. Import Data

In [2]:

```
test_data = pd.read_csv('SalaryData_Test.csv')

train_data = pd.read_csv('SalaryData_Train.csv')
train_data
```

Out[2]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race
0	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White
1	50	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White
2	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White
3	53	Private	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black
4	28	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black
...	...	...	...	...	...	...	...	...
30156	27	Private	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White
30157	40	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White
30158	58	Private	HS-grad	9	Widowed	Adm-clerical	Unmarried	White
30159	22	Private	HS-grad	9	Never-married	Adm-clerical	Own-child	White
30160	52	Self-emp-inc	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White

30161 rows × 14 columns

In [3]:

```
test_data
```

Out[3]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race
0	25	Private	11th	7	Never-married	Machine-op-inspct	Own-child	Black
1	38	Private	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White
2	28	Local-gov	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White
3	44	Private	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black
4	34	Private	10th	6	Never-married	Other-service	Not-in-family	White
...	...	...	...	...	...	...	...	...
15055	33	Private	Bachelors	13	Never-married	Prof-specialty	Own-child	White
15056	39	Private	Bachelors	13	Divorced	Prof-specialty	Not-in-family	White
15057	38	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White
15058	44	Private	Bachelors	13	Divorced	Adm-clerical	Own-child	Asian-Pac-Islander
15059	35	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White

15060 rows × 14 columns

3. Data Preprocessing

In [4]:

```
df_temp = test_data.append(train_data)
df_temp
```

Out[4]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race
0	25	Private	11th	7	Never-married	Machine-op-inspct	Own-child	Black
1	38	Private	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White
2	28	Local-gov	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White
3	44	Private	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black
4	34	Private	10th	6	Never-married	Other-service	Not-in-family	White
...	...	...	...	...	...	...	...	...
30156	27	Private	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White
30157	40	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White
30158	58	Private	HS-grad	9	Widowed	Adm-clerical	Unmarried	White
30159	22	Private	HS-grad	9	Never-married	Adm-clerical	Own-child	White
30160	52	Self-emp-inc	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White

45221 rows × 14 columns

In [5]:

```
train = train_data.copy()
test = test_data.copy()
test.head()
```

Out[5]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex
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 [6]:

```
train.head()
```

Out[6]:

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

In [7]:

```
str_c = ["workclass","education","maritalstatus","occupation","relationship","race","sex","
str_c
```

Out[7]:

```
['workclass',
 'education',
 'maritalstatus',
 'occupation',
 'relationship',
 'race',
 'sex',
 'native']
```

In [8]:

```
for i in str_c:
    train[i]= LabelEncoder().fit_transform(train[i])
    test[i]=LabelEncoder().fit_transform(test[i])

mapping = {' >50K': 1, ' <=50K': 2}
train = train.replace({'Salary': mapping})
test = test.replace({'Salary': mapping})
```

In [9]:

```
test.head()
```

Out[9]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex
0	25	2	1	7	4	6	3	2	1
1	38	2	11	9	2	4	0	4	1
2	28	1	7	12	2	10	0	4	1
3	44	2	15	10	2	6	0	2	1
4	34	2	0	6	4	7	1	4	1

In [10]:

```
train.head()
```

Out[10]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex
0	39	5	9	13	4	0	1	4	1
1	50	4	9	13	2	3	0	4	1
2	38	2	11	9	0	5	1	4	1
3	53	2	1	7	2	5	0	2	1
4	28	2	9	13	2	9	5	2	0

In [11]:

```
df = train.append(test)
```

In [12]:

```
df1 = df.copy()
df1
```

Out[12]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	s
0	39	5	9	13	4	0	1	4	
1	50	4	9	13	2	3	0	4	
2	38	2	11	9	0	5	1	4	
3	53	2	1	7	2	5	0	2	
4	28	2	9	13	2	9	5	2	
...	...	...	...	...	...	...	...	...	
15055	33	2	9	13	4	9	3	4	
15056	39	2	9	13	0	9	1	4	
15057	38	2	9	13	2	9	0	4	
15058	44	2	9	13	0	0	3	1	
15059	35	3	9	13	2	3	0	4	

45221 rows × 14 columns

In [13]:

```
df1.describe()
```

Out[13]:

	age	workclass	education	educationno	maritalstatus	occupation	r
count	45221.000000	45221.000000	45221.000000	45221.000000	45221.000000	45221.000000	45
mean	38.548086	2.204507	10.313217	10.118463	2.585148	5.969572	
std	13.217981	0.958132	3.816992	2.552909	1.500460	4.026444	
min	17.000000	0.000000	0.000000	1.000000	0.000000	0.000000	
25%	28.000000	2.000000	9.000000	9.000000	2.000000	2.000000	
50%	37.000000	2.000000	11.000000	10.000000	2.000000	6.000000	
75%	47.000000	2.000000	12.000000	13.000000	4.000000	9.000000	
max	90.000000	6.000000	15.000000	16.000000	6.000000	13.000000	

In [14]:

```
df1.isna().sum()
```

Out[14]:

```
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
```

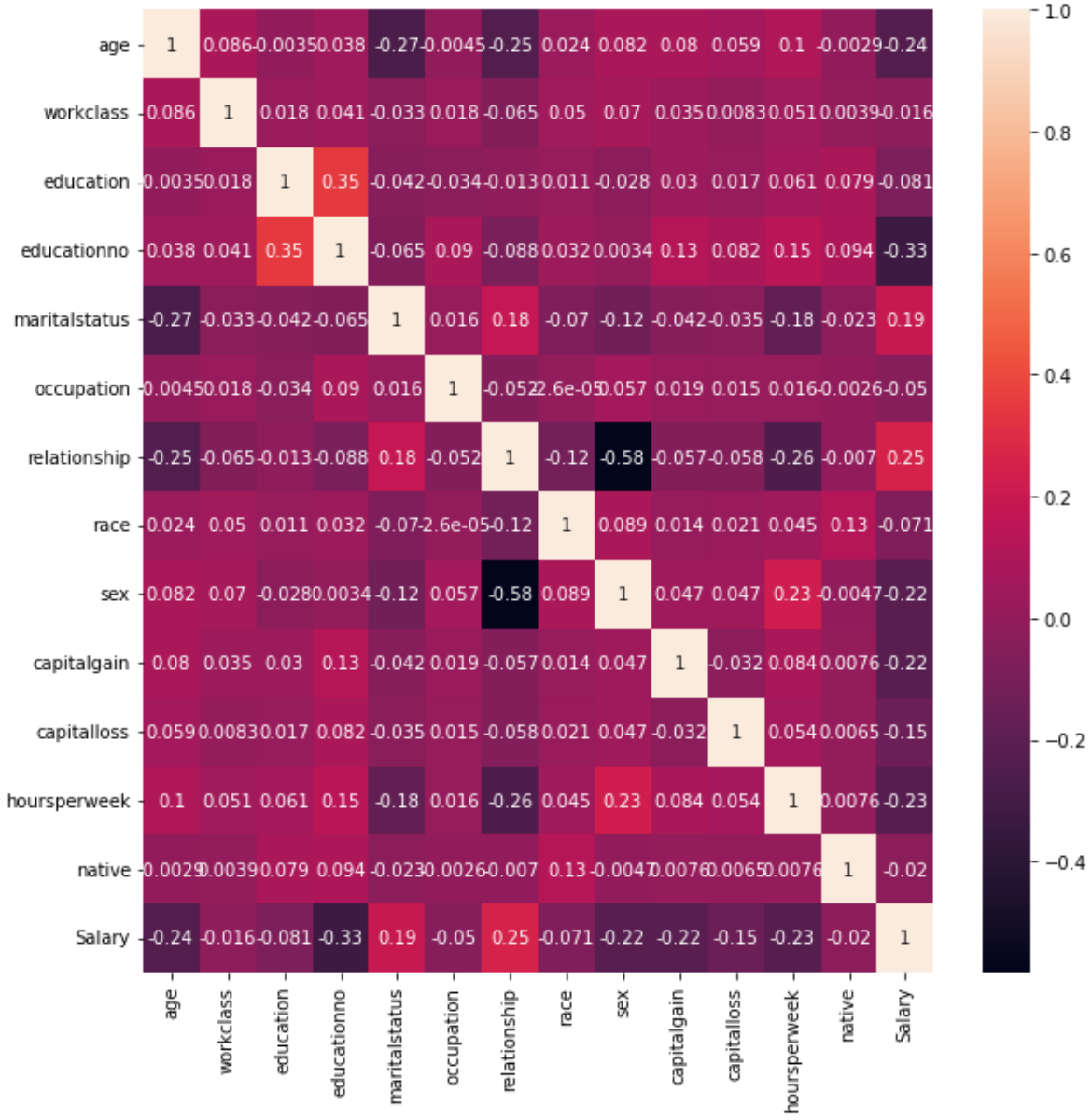
Finding Correlation

In [15]:

```
corr = df1.corr()
```

In [16]:

```
plt.figure(figsize=(10,10))
sns.heatmap(corr,annot=True)
plt.show()
```



In [17]:

```
df1.index.is_unique
```

Out[17]:

```
False
```

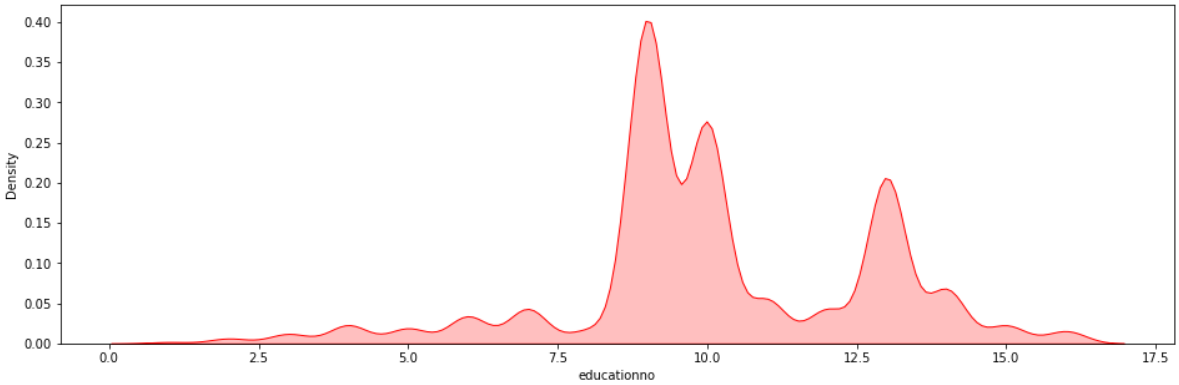
In [18]:

```
df1=df1.loc[~df1.index.duplicated(), :]
```

In [19]:

```
plt.figure(figsize=(16,5))
print("Skew: {}".format(df1['educationno'].skew()))
print("Kurtosis: {}".format(df1['educationno'].kurtosis()))
sns.kdeplot(df1['educationno'],shade=True,color='r')
plt.show()
```

```
Skew: -0.305378355820322
Kurtosis: 0.643604835875955
```



The Data is negatively skewed and has Low Kurtosis value

Most of people have eduction Number of years of education 8 - 11

4. Naive Bayes

In [22]:

```
x_train = train.iloc[:,0:13]
y_train = train.iloc[:,13]
x_test = test.iloc[:,0:13]
y_test = test.iloc[:,13]
```

GaussianNB

In [23]:

```
cls_fr_gnb = GaussianNB().fit(x_train, y_train)
```

In [24]:

```
y_pred_gnb = cls_fr_gnb.predict(x_test)
```

In [25]:

```
confusion_matrix(y_test, y_pred_gnb)
```

Out[25]:

```
array([[ 1209,  2491],
       [  601, 10759]], dtype=int64)
```

In [26]:

```
pd.crosstab(y_test.values.flatten(),cls_fr_gnb)
```

Out[26]:

col_0	GaussianNB()
row_0	
1	3700
2	11360

In [27]:

```
print ("Accuracy",np.mean(y_pred_gnb==y_test.values.flatten()))
```

```
Accuracy 0.7946879150066402
```

MultinomialNB

In [28]:

```
cls_fr_mnb = MultinomialNB().fit(x_train, y_train)
```

In [29]:

```
y_pred_mnb = cls_fr_mnb.predict(x_test)
```

In [30]:

```
confusion_matrix(y_test, y_pred_mnb)
```

Out[30]:

```
array([[  780,  2920],
       [  469, 10891]], dtype=int64)
```

In [31]:

```
pd.crosstab(y_test.values.flatten(),cls_fr_mnb)
```

Out[31]:

col_0	MultinomialNB()
row_0	
1	3700
2	11360

In [32]:

```
print ("Accuracy",np.mean(y_pred_mnb==y_test.values.flatten()))
```

```
Accuracy 0.7749667994687915
```

GaussianNB Model has a better Accuracy than MultinomialNB

END

In [ ]:

```

```