```python
In [1]: import pandas as pd
        import numpy as np
```

```python
In [2]: df = pd.read_csv("D:\Summer Training Video\ML\Social_Network_Ads.csv", usecols = ['Age','EstimatedSalary','Purchased']
```

```python
In [3]: df.head()
```

Out[3]:

	Age	EstimatedSalary	Purchased
0	19	19000	0
1	35	20000	0
2	26	43000	0
3	27	57000	0
4	19	76000	0

```python
In [4]: x = df.drop(columns = ['Purchased'])      # Independent column
        y = df['Purchased']     # Dependent
```

```python
In [5]: from sklearn.model_selection import train_test_split
```

```python
In [6]: x_train , x_test , y_train , y_test = train_test_split(x,y,test_size = 0.2 , random_state = 23)
```

```python
In [7]: from sklearn.preprocessing import StandardScaler
```

```python
In [8]: sc = StandardScaler()
```

```python
In [9]: x_train_new = sc.fit_transform(x_train)
```

```python
In [10]: x_test_new = sc.transform(x_test)
```

from sklearn.naive_bayes import BurnolliNB

```python
In [11]: from sklearn.naive_bayes import GaussianNB , MultinomialNB , BernoulliNB

         classifier = GaussianNB()
```

```python
In [12]: classifier.fit(x_train_new , y_train)
```

Out[12]:
```
▾ GaussianNB
GaussianNB()
```

```python
In [13]: y_pred = classifier.predict(x_test_new)
```

```python
In [14]: y_pred
```

Out[14]: array([0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
               1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0,
               1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0], dtype=int64)

```python
In [15]: from sklearn.metrics import confusion_matrix
```

```python
In [17]: cn = confusion_matrix(y_test , y_pred)
```

```python
In [18]: cn
```

Out[18]: array([[48, 2],
               [5, 25]], dtype=int64)

```
In [ ]:    [[TP , FN ]
            [FP , TN ]]
```

```
In [20]:   # accuracy_score(y_test , y_pred)

           from sklearn.metrics import accuracy_score
```

```
In [22]:   ac = accuracy_score(y_test , y_pred)
```

```
In [23]:   ac
```

Out[23]:   0.9125

```
In [24]:   # isme random forest classifier lagana h
```

```
In [ ]:
```