Naive Bayes Algorithm

This algorithm works on Bayes Theorem i.e Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Posterior=(Likelihood+Prior)/Evidence

Assumptions: All variable are independent of each other

P(c|X)=P(x1|c)*P(x2|c)*.....*P(xn|c)*P(c)

1. Reading the Data

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

In [2]:
```
df=pd.read_csv("loans.csv")
df.head()
```

Out[2]:

	accepted	creditscore	amount	age	marital	health_ins	creditgrade
0	0	387	42580093	87	0	0	A
1	0	400	80816186	49	0	1	A
2	0	360	37264552	46	1	0	A
3	0	378	7209235	38	1	0	A
4	0	387	71637479	51	1	0	A

In [3]:
```
df.shape
```

Out[3]:
```
(2500, 7)
```

In [4]:
```
df.describe()
```

Out[4]:

	accepted	creditscore	amount	age	marital	health_ins
count	2500.000000	2500.000000	2.500000e+03	2500.000000	2500.000000	2500.000000
mean	0.518800	286.824400	3.364925e+07	58.156400	0.494800	0.440000
std	0.499746	74.884278	2.897636e+07	21.082678	0.500073	0.496486
min	0.000000	100.000000	5.113590e+05	18.000000	0.000000	0.000000
25%	0.000000	246.000000	8.954224e+06	40.000000	0.000000	0.000000
50%	1.000000	298.000000	2.360884e+07	61.000000	0.000000	0.000000
75%	1.000000	343.250000	5.592593e+07	76.000000	1.000000	1.000000
max	1.000000	400.000000	9.995992e+07	90.000000	1.000000	1.000000

2. Loading the library

In [5]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import accuracy_score,roc_auc_score,confusion_matrix

import matplotlib.pyplot as plt
```

In [6]:
```
x=df[["creditscore","amount","age","marital","health_ins"]]
y=df["accepted"]
```

In [7]:
```
print(y.value_counts())
```

```
1    1297
0    1203
Name: accepted, dtype: int64
```

3. Model Building

In [8]:
```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.7,random_state=0)
xtrain=pd.DataFrame(xtrain)
xtest=pd.DataFrame(xtest)
ytrain=pd.DataFrame(ytrain)
ytest=pd.DataFrame(ytest)
```

In [9]:
```
model = GaussianNB()
model.fit(xtrain,ytrain)
```

```
C:\Users\Gaurav\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: Da
taConversionWarning: A column-vector y was passed when a 1d array was expected.
Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[9]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

4. Predicting the Value

In [10]:
```
pred=model.predict(xtest)
```

In [11]:
```
print("       Accuracy Score")
print("---------------------------------")
accuracy_score(ytest,pred)
```

```
       Accuracy Score
---------------------------------
```

Out[11]:
```
0.7297142857142858
```

This shows that our model calculate the correct result for 73% of times.