

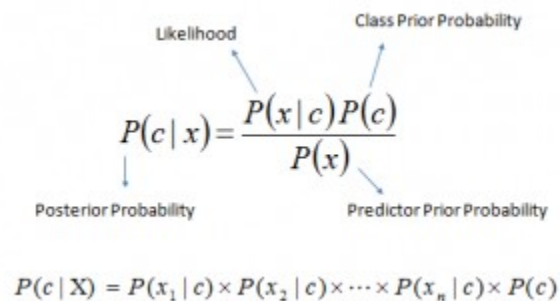
**No. of Experiment:** 03

**Name of the Experiment:** Classification using Naïve Bayes Algorithm

**Theory:**

Naïve Bayes Algorithm based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Bayes theorem provides a way of calculating posterior probability  $P(c|x)$  from  $P(c)$ ,  $P(x)$  and  $P(x|c)$ . Look at the equation below:



The diagram shows the equation  $P(c|x) = \frac{P(x|c)P(c)}{P(x)}$  with arrows pointing from labels to the terms in the equation. 'Likelihood' points to  $P(x|c)$ , 'Class Prior Probability' points to  $P(c)$ , 'Posterior Probability' points to  $P(c|x)$ , and 'Predictor Prior Probability' points to  $P(x)$ . Below the equation is the expanded formula:  $P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$ .

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Posterior Probability

Likelihood

Class Prior Probability

Predictor Prior Probability

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

**Advantages and Disadvantages:**

**Advantage:**

1. This algorithm works quickly and can save a lot of time.
2. Naive Bayes is suitable for solving multi-class prediction problems.
3. If its assumption of the independence of features holds true, it can perform better than other models and requires much less training data.
4. Naive Bayes is better suited for categorical input variables than numerical variables.

**Disadvantage:**

1. Naive Bayes assumes that all predictors (or features) are independent, rarely happening in real life. This limits the applicability of this algorithm in real-world use cases.
2. This algorithm faces the 'zero-frequency problem' where it assigns zero probability to a categorical variable whose category in the test data set wasn't available in the training dataset. It would be best if you used a smoothing technique to overcome this issue.
3. Its estimations can be wrong in some cases, so you shouldn't take its probability outputs very seriously.

**Algorithm:**

- 1 START
- 2 Import GaussianNB, confusion\_matrix
- 3 Initialize GaussianNB ( )
- 4 FIT GaussianNB ( ) Classifier
- 5 Calculate 'y\_pred' and 'y\_pred\_train' and their confusion matrix and compare them
- 6 END

**Psuodocode:**

```
IMPORT GaussianNB
IMPORT confusion_matrix
CALL GaussianNB( ) classifier
FIT Classifier ( X_train, y_train )
CALCULATE y_pred
CALCULATE cm_test
CALCULATE y_pred_train
CALCULATE cm_train
```

**Dataset:** Used a dataset that was based on Car Sells Report .

**Screenshot of the task:**

```
import pandas as pd
```

```
dataset = pd.read_csv('./car.csv')
```

```
print(dataset.head(3))
print(dataset.shape)
```

	Car No.	Maker	Type	Color	Sell
0	1	TATA	SPORTS	RED	YES
1	2	FORD	SPORTS	BLACK	YES
2	3	TATA	SUV	RED	NO

(10, 5)

```
X = pd.DataFrame(dataset.drop(['Sell'], axis=1))
```

```
y = pd.DataFrame(dataset['Sell'])
```

```
columns = dataset.select_dtypes(include=['object']).columns.to_list()
from sklearn.preprocessing import StandardScaler, LabelEncoder
label = LabelEncoder()
```

```
def encode_labels(df, labels_to_encode):
    for column in labels_to_encode:
        df[column] = label.fit_transform(df[column])
    return df
```

```
df_labelled = encode_labels(dataset, columns)
```

```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
```

```
X = pd.DataFrame(
    sc_X.fit_transform(df_labelled.drop(['Sell'], axis = 1))
)
y = df_labelled.Sell
```

```
from sklearn.model_selection import train_test_split
# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=9)
```

```
from sklearn.naive_bayes import GaussianNB
nv = GaussianNB() # create a classifier
nv.fit(X_train, y_train) # fitting the data
```

```
GaussianNB()
```

```
from sklearn.metrics import accuracy_score
y_pred = nv.predict(X_test) # store the prediction data
accuracy_score(y_test, y_pred) # calculate the accuracy
```

```
0.5
```

#### Result:

Accuracy = 0.5

#### Conclusion:

Larger dataset can improve the result.

#### Contribution by Members:

Abdullah Al Hasib (1105081)

Md Abdullah Al Mamun mozumder (1105054)