Bharatiya Vidya Bhavan's

Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India

(Autonomous College Affiliated to University of Mumbai)

BE-ETRX B | **Sub- AIML Lab**

Name: Shubham Sawant | **UID :2019110050**

Name of the Experiment: **Implement the Naïve-Bayes classifier**

Objective: Implement the naïve Bayesian Classifier model to classify a set of documents that you have assumed. Calculate the accuracy, precision, and recall for your data set.

Outcomes:

1. Find the conditional probabilities of attributes of the train data using Bayes theorem and follow the steps of the algorithm.
2. Apply the Naïve-Bayes algorithm to classify the given documents.
3. Apply Parameter smoothing for non-occurring values of attributes while calculation.
4. Find accuracy, precision, recall of the model for test data set.

System Requirements: Windows with MATLAB

Theory:

Naive Bayes algorithm is a classification technique 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:

Likelihood   Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

Posterior Probability   Predictor Prior Probability

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

- P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor.

Naive Bayes algorithm:

Step 1: Convert the data set into a frequency table

Step 2: Create a likelihood table by finding the probabilities

Step 3: Calculate the posterior probability of each feature with respect to the class.

Step 4: If for a certain feature the probability evaluates to zero use feature smoothening for correction.

$$\hat{\theta}_i = \frac{x_i + \alpha}{N + \alpha d} \qquad (i = 1, \dots, d),$$

Step 5: Classify the example into the class for which the probability is highest.

Performance parameters of the model :

Accuracy: It is the ratio of number of correct predictions to the total number of input samples.

$$\boldsymbol{Accuracy} = \frac{No.of\ correct\ prediction}{No.of\ total\ predictions\ made}$$

Precision: Precision is defined as the fraction of the examples which are actually positive among all the examples which we predicted positive.

$$\boldsymbol{Precision} = \frac{No.of\ correct\ prediction}{No.of\ total\ returned\ predictions}$$

Recall: We define recall as, among all the examples that actually positive, what fraction did we detect as positive?

$$\boldsymbol{Recall} = \frac{No.of\ correct\ prediction}{No.of\ actual\ correct\ values}$$

F1-score: F1 Score is the Harmonic Mean between precision and recall.

$$\boldsymbol{Precision} = \frac{2\ x\ Precision\ x\ Recall}{Precision\ +\ Recall}$$

BE-ETRX B **Sub- AIML Lab**
Name: Shubham Sawant **UID :2019110050**

Confusion Matrix: Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.

There are 4 important terms :

- True Positives: The cases in which we predicted YES and the actual output was also YES.
- True Negatives: The cases in which we predicted NO and the actual output was NO.
- False Positives: The cases in which we predicted YES and the actual output was NO.
- False Negatives: The cases in which we predicted NO and the actual output was YES.

Data Set Link: https://archive.ics.uci.edu/ml/machine-learning-databases/00193/CTG.xls

Dataset Description:

Number of Instances: 2126

Number of Attributes: 38

Attribute Information:

FileName	of CTG examination
Date	of the examination
b	start instant
e	end instant
LBE	baseline value (medical expert)
LB	baseline value (SisPorto)
AC	accelerations (SisPorto)
FM	foetal movement (SisPorto)
UC	uterine contractions (SisPorto)
ASTV	percentage of time with abnormal short term variability (SisPorto)
mSTV	mean value of short term variability (SisPorto)
ALTV	percentage of time with abnormal long term variability (SisPorto)
mLTV	mean value of long term variability (SisPorto)
DL	light decelerations
DS	severe decelerations

BE-ETRX B
Name: Shubham Sawant
Sub- AIML Lab
UID :2019110050

DP	prolongued decelerations
DR	repetitive decelerations
Width	histogram width
Min	low freq. of the histogram
Max	high freq. of the histogram
Nmax	number of histogram peaks
Nzeros	number of histogram zeros
Mode	histogram mode
Mean	histogram mean
Median	histogram median
Variance	histogram variance
Tendency	histogram tendency: -1=left assymetric; 0=symmetric; 1=right assymetric
A	calm sleep
B	REM sleep
C	calm vigilance
D	active vigilance
SH	shift pattern (A or Susp with shifts)
AD	accelerative/decelerative pattern (stress situation)
DE	decelerative pattern (vagal stimulation)
LD	largely decelerative pattern
FS	flat-sinusoidal pattern (pathological state)
SUSP	suspect pattern
CLASS	Class code (1 to 10) for classes A to SUSP
NSP	Normal=1; Suspect=2; Pathologic=3

BE-ETRX B **Sub- AIML Lab**
Name: Shubham Sawant **UID :2019110050**

Code:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
from sklearn.metrics import confusion_matrix
data = pd.read_csv(r"C:\Users\Dell\Desktop\Shubham\SEM7\AIML\EXP6\fetal_health.csv")
df = data[data.fetal_health != 2]
df.head()
```
[22] ✓ 0.2s

```
features = df.iloc[:,:-1]
y = df['fetal_health']
x = features.to_numpy()
y = y.to_numpy()
```
✓ 0.1s

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)
```
✓ 0.1s

✓ ...

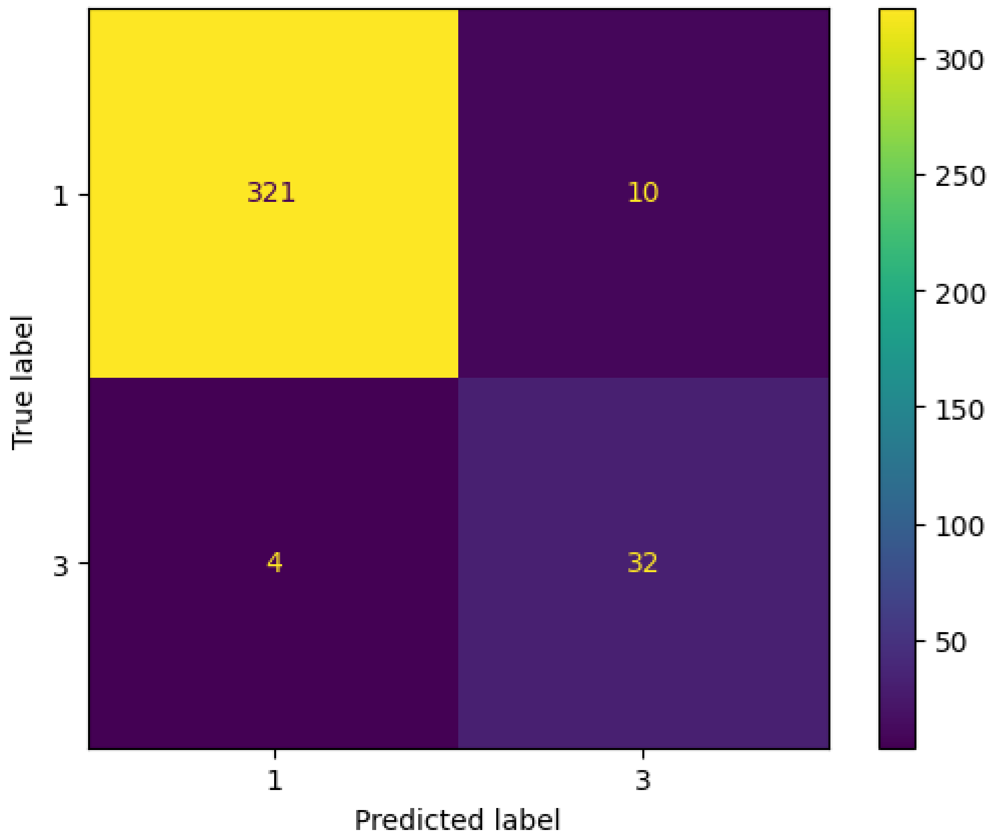
```
Gaussian Naive Bayes model accuracy(in %): 96.18528610354224
```

```
import matplotlib.pyplot as plt
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = con_mat, display_labels = [1,3])
cm_display.plot()
plt.show()
```
✓ 0.6s

```
Accuracy = metrics.accuracy_score(y_test, y_pred)
Precision = metrics.precision_score(y_test, y_pred)
recall = metrics.recall_score(y_test, y_pred)
Specificity = metrics.recall_score(y_test, y_pred, pos_label=3)
F1_score = metrics.f1_score(y_test, y_pred)
```
✓ 0.4s

```
print({"Accuracy":Accuracy,"Precision":Precision,"Recall":recall,"Specificity":Specificity,"F1_score":F1_score})
print('Training set score: {:.4f}'.format(gnb.score(X_train, y_train)))
print('Test set score: {:.4f}'.format(gnb.score(X_test, y_test)))
```
✓ 0.3s

Interpretation of output:




```
{'Accuracy': 0.9618528610354223, 'Precision': 0.9876923076923076, 'Recall': 0.9697885196374623, 'Specificity': 0.8888888888888888, 'F1_score': 0.9786585365853658}
Training set score: 0.9460
Test set score: 0.9619
```

Interpretation:

- There are total 21 input features each with different units having values in different ranges. Hence, we need to standardize and normalize the data for better performance.
- The number of positive labels is more than negative labels. The data has slight imbalance.
- Due to this imbalance, the model might fail to correctly classify negative examples. But accuracy is high and training and test set scores are almost equal. Hence there is no overfitting.
- The class follows a gaussian distribution.

Conclusion:

- Naïve Bayes is fast and easy to implement but its biggest disadvantage is that the requirement of predictors to be independent.
- Naïve Bayes performs very well on high dimensional data like text or documents or even images.
- This is a probability-based classifier which uses Bayes theorem for classification.