**MACHINE LEARNING - PROJECT 2**

**Naïve Bayes Classification**

**INTRODUCTION:**

Naïve Bayes is a classification algorithm for binary and multiclass classification problems. Naive Bayes is a simple supervised machine learning algorithm that uses the Bayes' theorem with strong independent assumptions between the features to produce results. The algorithm just assumes that each input variable is independent. It really is a naive assumption to make about real-world data. Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

**BAYES THEOREM:**

# Text Description automatically generated with medium confidence

Bayes' Theorem discovers the likelihood of an occasion happening given the likelihood of another occasion that has effectively happened. Bayes' hypothesis is expressed numerically as the accompanying condition:

where A and B are events,

* **P(A|B)** is **posterior probability** – Probability of hypothesis A on the observed event B.
* **P(B|A)** is the **likelihood probability** – Probability of hypothesis before observing the evidence.
* **P(A)** is **prior probability** – Probability of hypothesis before observing the evidence.
* **P(B)** is the **marginal probability** – Probability of evidence.

Now, with regards to our dataset, we can apply Bayes’ theorem in the following way:

Graphical user interface, text

Description automatically generated with medium confidence

Naïve Bayes works also on text categorization. NB bears the cost of quick model structure and scoring and can be utilized for both paired and multi-class arrangement issues.

The guileless Bayes classifier is extremely valuable in high-dimensional issues in light of the fact that multivariate strategies like QDA and even LDA will separate.

**MODEL BUILDING:**

**Importing required libraries**

**A picture containing chart

Description automatically generated**

**Stop Words & Vocabulary:**

For building the model, we have created few stop words referring certain documents. The words which are used generally are taken as the stop words here and implemented to build the model. We split the documents in data folders separately. We also have a similar function for basic vocabulary in which the vocabulary is sorted and also a feature list is provided for the key to access the vocabulary.

**Implementation:**

Naive Bayes classifiers are a collection of classification algorithms based on Bayes’ Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. The given dataset has been split into 50% for the testing and training.

• **Train Dataset**: Used to fit the machine learning model.

• **Test Dataset**: Used to evaluate the fit machine learning model. With this given information about the model, we will be splitting the data into train and test data. We can apply the model to the training data, and predict labels for the test data.

The model which we have used to build it will be counting total data points, total data and total count as a whole. This is the manner by which we hope to utilize the model practically speaking. In particular, to fit it on accessible information with known data sources and yields, at that point make forecasts on new models later on where we don't have the normal yield or target esteems.

The train-test system is fitting when there is an adequately huge dataset accessible. Grouping is a prescient demonstrating issue that includes relegating a name to guaranteed input information test.

The issue of grouping prescient demonstrating can be outlined as ascertaining the restrictive likelihood of a class mark given an information gives a principled method to figuring this contingent likelihood, albeit by and by requires a tremendous number of tests (enormously-measured dataset) and is computationally costly.

All things considered, the count of Bayes Theorem can be improved by making a few suspicions, for example, each info variable is autonomous of any remaining information factors. Albeit an emotional and ridiculous presumption, this makes the outlook of the restrictive likelihood manageable and brings about a viable order model alluded to as Naive Bayes.

Therefore, the time taken for the code to execute will take minimum of 30-40 minutes and then will be able to produce the accuracy results. The below set of code is which I have used for the prediction function of Y\_test.

Text

Description automatically generated

The below image represents the confusion matrix between the true and predicted labels for the test data:

Chart

Description automatically generated

The model which is built predicts a F-score of approximately 84.2%. Also, I tried using sk-learn and got the same accuracy. Below is the image for the model’s classification accuracy:

Chart, bar chart

Description automatically generated

**Conclusion:**

In conclusion we built a model, added stop words and vocabulary for the given dataset. We also split the dataset into 80% train and 20% test data to predict the accuracy of the dataset for naïve bayes classification and got an accuracy rate of 84.375% in inbuilt model and 84.6 in model which has been created

Graphical user interface

Description automatically generated with medium confidence

When the data is split in 50% training and 50% testing the Accuracy was

Graphical user interface

Description automatically generated

When the data is split in 75% training and 25% testing the Accuracy was

Graphical user interface

Description automatically generated

From all these it is evident that the model which we built is having more or less the same accuracy of the inbuilt model hence the model can be used for applications

**References:**

<https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c>

<https://towardsdatascience.com/multinomial-naive-bayes-classifier-for-text-analysis-python-8dd6825ece67>

<https://www.javatpoint.com/machine-learning-naive-bayes-classifier>

<http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html>