**CLASSIFICATION USING NAIVE BAYES**

NORTHEASTERN UNIVERSITY - SILICON VALLEY

SUBMITTED TO: SUBMITTED BY:

YIQING WANG ANUBHAV RASTOGI

(PROFESSOR) (STUDENT)

**INTRODUCTION**

With the passage of time, there has been an immense growth in the field of Artificial Intelligence. Artificial Intelligence is nothing but a field of computer science involving Machine Learning wherein machines learn to respond depending upon different circumstances and situations. Various machine learning algorithms have made it possible for the machines to respond, behave and work like humans. The importance of Machine learning has increased with an increase in the amount of Big Data. The increase in the data volumes have not only highlighted the significance but has led to an increase in the usage and application of various machine learning algorithms. Whether its finance or transportation, health care or retail, machine learning is growing rapidly as it has not only resulted in the growth or efficient working in an organization but has also provided an edge over its competitors.

The present assignment is also based on one of the important machine learning algorithms i.e. Naive Bayes. It is a collection of machine learning algorithms which can be understood with the help of Bayes Theorem. The Bayes Theorem defines the conditional probability of an event and is stated by the following formula:



The above formula defines the conditional probability of an event A when B is true. Here, P(A) reflects the probability of event A and P(B) reflects the probability of event B. P(B|A) is the conditional probability of event B when A is true.

Naive Bayes assumes that the presence of each feature in the class is independent and unrelated to the presence of others in the class. It basically classifies each feature to the class independent of other features of the class. The Naive Bayes model is very simple and easy to build. It is quick in making predictions and can be trained easily. Because of these advantages, it is widely used in Recommendation Systems and Real-Time Predictions. It is also used in spam filtering, sentiment analysis, and text classification too. Despite several advantages, this model has one major disadvantage which is that it assumes the feature to be independent even if they are not. Because of this reason, it is called naive as the features may not always be independent of each other, they may be dependent in reality.

The present assignment contains two parts: Part A and Part B. Both the parts involve Naive Bayes Classification performed on two different datasets. Part A involves dataset relating to emails wherein Naive Bayes is used for spam identification. Part B involves the dataset relating to different wine types. In Python, a Naive Bayes model can be built in three different forms:

* Gaussian: This model relies on the assumption that the features follow a normal distribution and is used in classification.
* Multinomial: This model is used mainly for discrete counts
* Bernoulli: This is used in case of binary feature vectors i.e. zeros and ones.

**PART A**

In Part A of the assignment, Naive Bayes has been performed through Python in Jupyter Notebook. In the performance of an algorithm, there are generally five steps involved i.e. Collecting the data, exploring and preparing the data, training the model, evaluating, and improving the model performance. So here also these steps have been performed. Firstly, the necessary package is imported, and the data file is extracted. After extracting, the head of the dataset is printed to have a look at it. In the present dataset, we are trying to predict the type of emails, so while loading features and target values, we are putting mail type as variable y and the other column of text as variable X. We have also printed variables X and y so as to be sure that the dataset is correctly loaded, and the target value is also set correctly.

Now, the next step is to transform and fit the model which is done by importing **CountVectorizer()** and **TfidfTransformer().** After this, the dataset is split into Training Dataset and Test Dataset. Here, the dataset is split into training and test in the ratio of 4:1. Now, the dataset is fitted into a Multinomial model as it is used for text classification problem. In order to perform spam email identification, the multinomial model is appropriate. The final step involves prediction which is done through **confusion\_matrix** and **classification\_report**. Also, the model accuracy has been printed to check how good the model is.

It can be seen that the accuracy of the model is 96.14%

|  |  |  |
| --- | --- | --- |
|  | Ham | Spam |
| Precision | 0.96 | 1.00 |
| Recall | 1.00 | 0.71 |
| F1-score | 0.98 | 0.83 |
| Support | 966 | 149 |

Precision reflects instances which are relevant to the query and is high in both cases. Recall reflects the successfully retrieved instances and in the present case, it is maximum in the case of Ham. F1 score is the harmonic mean of precision and recall. Support represents the number of true response samples present in the respective class. It can be seen from the confusion matrix that the model fits well on the current data set.

|  |  |  |
| --- | --- | --- |
|  | Predicted Ham | Predicted Spam |
| Actual Ham | 966 | 0 |
| Actual Spam | 43 | 106 |

The above table implies that all Ham mails are predicted accurately. It furthers tells us that out of 149 Spam emails, 106 are predicted accurately while the remaining are predicted as Ham.

Lastly, predictions are done for a single string implying the further evaluation of the model. X is taken as "I am going for shopping tomorrow. Wanna join?", to which the prediction was accurate that it’s a Ham. In the second time, x was equal to "CONGRATS! Your mobile number has been selected in a Lucky draw. Call on 8447621901 to claim your guaranteed reward. Hurry!". This was also predicted accurately as Spam. Since, the accuracy of the model is extremely high i.e. 96.14%, further improvement is not necessarily required.

**PART B**

Part B of the assignment includes the same algorithm i.e. Naive Bayes to be performed on the different dataset. This part of the assignment is also performed through Python in Jupyter notebook only. In this dataset, the prediction is made about the types of wine. The data is in numeric form; therefore, the Gaussian Naive Bayes Model is applied to classify the types of wines. Firstly, the necessary package or library is imported, and then the data file is extracted so as to have a look at the data by printing the data head. Here we are classifying different wine types, so the target values will be the column of Wine\_types. After data collection, the next step involves Preparation of the data; the features and target are loaded by importing **GaussianNB**. Since we are predicting Wine types, therefore variable y contains the column of Wine\_types and variable X contains all other columns. The next step involves splitting the data into training and test sets in the ratio of 4:1. After the split, the model is fit, and prediction is made out. The accuracy of the model turns out to be 1 which implies that the model is fully correct. The Precision and Recall, both are completely 1.00 for all the three wine types reflecting the accuracy.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Type 1 | Type 2 | Type 3 |
| Precision | 1.00 | 1.00 | 1.00 |
| Recall | 1.00 | 1.00 | 1.00 |
| F1-score | 1.00 | 1.00 | 1.00 |
| Support | 12 | 14 | 10 |

The confusion matrix is also completely clean reflecting the predictions about the three wine types to be fully accurate.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Predicted Type 1 | Predicted Type 2 | Predicted Type 3 |
| Actual Type 1 | 1.00 | 1.00 | 1.00 |
| Actual Type 2 | 1.00 | 1.00 | 1.00 |
| Actual Type 3 | 1.00 | 1.00 | 1.00 |

Since the model is fully accurate, there is no need for any improvement.

**CONCLUSION**

This brings an end to the report. On the basis of the analysis, it can be concluded that Naive Bayes presumes the presence of a feature in class is independent of others. In other words, it classifies each feature as independent or unrelated to others. The model is easy to build and works well in multiclass prediction. Also, it performs well in categorical input variables as compared to numerical variables. However, it does suffer from a few limitations such as assumptions of independent variables. In reality, the existence of independent feature may or may not be true. But Naive Bayes always assume it to be true. Because of this assumption, it is called as Naive. Another important demerit of Naive Bayes is zero frequency wherein the model is unable to predict if the category in the categorical variable is not observed in training dataset. Thus, it becomes necessary to use smoothing technique.

In the present assignment, the Naive Bayes is performed through Multinomial and Gaussian model in Part A and Part B respectively. The accuracy in both the parts is extremely high reflecting the variables have a normal distribution. In Part B, the model is completely accurate reflecting that the Wine dataset is a normally distributed and does not require any scaling or smoothing.

Thus, it can be concluded that Naive Bayes is a collection of machine learning algorithm based on Bayes Theorem and conditional probability. Naive Bayes has become easier to perform in Python with the help of Sci-kit learn and does require normalization and scaling to perform well in case of discrete continuous data with zero frequency issue.

**REFERENCES**

* Sunil Ray. (2017, September 11). 6 Easy Steps to Learn Naive Bayes Algorithm (with codes in Python and R). Retrieved from <https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/>
* Prashant Gupta. (2017, November 6). Naive Bayes in Machine Learning. Retrieved from <https://towardsdatascience.com/naive-bayes-in-machine-learning-f49cc8f831b4>