**Client-Service-Rating: Applications of Machine Learning in Document Classification**

Surname: White

Name: Darren

Student Number: 201726807

Module Name: Advanced Machine Learning

1. **Abstract**

Naïve Bayes classifier is a simple classifier which classifies based on probabilities of events. In this study a probabilities approach to feature selection for multi-class text categorization using Naïve Bayes text classifier is proposed to classify data scraped from Coursera course reviews, this classifier is based on the Bayes’s Theorem which helps compute the conditional probabilities of occurrence of two events based on the probability of occurrence of each individual event. Using this theorem, the probability of occurrence of each feature is calculated on the total probability and utilize the values as a ranking criterion.

**Keywords**: Text classification, Naïve Bayes, Probability, Bayes Theorem.

1. **Introduction**

With the current pandemic that we are facing in the world, all the different sectors of industry have been affected and must adjust somehow. All the universities in the country had to transition into online learning since in-class learning is not possible. How effective is online learning in comparison to in-class learning? Will students and universities which are implementing this form of learning for the first time survive? These are the possible questions which were crossing the minds of student as well as university stuff. Conducting reviews of every online lecture is a good way to measure the performance of online learning. In this study I build a probabilistic text classification model for data that was scraped from Coursera course reviews which has about 107018 features.

I will be classifying the reviews given into five different classes (i.e. 1, 2, 3, 4, 5), with this classification I would be able to see how online learning is viewed. The aim of the study is to classify the reviews regarding online learning, whether the reviews are good, bad or neutral. Text classification (or categorization) is the process of classifying text into a specific class or group. Since the text needs to be classified into five different classes a multi-class text classification is followed using Naïve Bayes text classifier. In a study by Borgelt and Gedhardt they point out that Naïve Bayes classifiers make use of a probabilistic approach to assign a class to a case or an object and can be seen as a special type of probabilistic network.

1. **Problem statement**

In recent times we have seen many institutions of high education transition into online learn due to the pandemic taking over the world. Many people are wondering whether students and institutions will survive this online learning which is new for many students and institutions. The only way to measure the success rate of online learning is to get reviews on online lectures. In this study I aim to conduct a probabilistic approach to text classification, where I will be classifying reviews from Coursera reviews data into five different classes (i.e. 1, 2, 3, 4, 5). I will be using a probabilistic approach to multinomial Naïve Bayes to classify the reviews in this approach no python libraries will be used to do the classification.

The objective of this study is to determine how students review online lectures. This information will be used to better online lectures if the reviews are bad and used to improve if the reviews are good. The questions which need to be answered are: Does Naïve Bayes classify the reviews correctly or is there a better classifier to us? How do the students review and rate online classes? It is important to know the reviews of online learning to see if there are areas which needs improvement, this would also help educator and lecturers implement ways to help students adapt to online learning.

1. **Literature Review**

Ioan Pop (2006) proposed a Naïve Bayesian algorithm to rank document or web mining tasks, the study points out that Naïve Bayes algorithm improves the tasks of the web mining by the accuracy documents classification. Naïve Bayes are important in areas such as e-mail spamming, machine learning for Semantic Web, document ranking by text classification, and hierarchical text categorization as pointed out by Ioan Pop (2006). Arar, et al. (2017) pointed out that Naïve Bayes (NB) is one of the most widely used algorithms in classification problems because of its simplicity, effectiveness, and robustness. And Mao, et al. (2020) in their study pointed out that Bayesian Classification is a naturally probabilistic method which performs classification tasks based on the class membership probabilities, i.e. the probability that a given sample belongs to each class.

In Bayesian classification it is important to establish a probability distribution model such as a Gaussian distribution for each class for probability estimation.

The earliest attempts to classify textual data by machine instead of by hand were rule-based classifiers for which knowledge engineers and domain expects created if-then rules for sorting text based on their content, Caitlin Cassidy (2020). Those systems classified data which their we designed for well, however their rules were expensive to create and did not work well on new datasets, this brought about machine learning as a preferable approach to text classification. Text classification is a process of automatically assigning an unknown textual feature to its appropriate one or more classes, Feng, et al. (2018). The most popular approach to text classification is to use machine learning techniques that inductively build a classification model of pre-defined classes from a training set of labelled text data. The machine learning methods for text classification include Naïve Bayes, k-nearest neighbors, decision trees, support vector machine (SVM), and deep learning methods just to mention a few. Kim, et al. (2018) stated that “the Naïve Bayes learning algorithm has a number of superior advantages compared to the other learning algorithms in constructing the operational text classification system even though it is old.

Arar, et al. (2017) proposed the Feature Dependent Naïve Bayes (FDNB) method, in this method the assumption is that the features having continuous (ordinal) values are converted to categorical (nominal) values. Discretization pre-processing is done in order to convert the values. FDNB consists of two processing standard NB and discretization of data. NB is a classification method which makes use of Bayes theorem, an instance’s posterior distribution is proportional to prior distribution and likelihood, it can be expressed using the following formula:

Where, refers to a certain class;

And represents metric values of a feature in the data.

Arar, et al. (2017) in their study process point out that when features are discretized with an entropy-based method, NB performance increases significantly. A discretization method that was used is based on the minimal entropy heuristic, at the end of the discretization process the numeric feature values are converted to , . Uysal, et al. (2012) in their study proposed a novel filter based probabilistic feature selection method, namely distinguishing feature selector (DFS) for text classification. The best filter based feature selection method should assign high scores to distinctive features while assigning lower scores to irrelevant ones. In text classification each distinct tern corresponds to a feature.

1. **Methodology**

In the methodology I followed a series of sequential steps throughout the process. I did not do any data collection during my methodology since a dataset from Coursera containing reviews and labels was provided to us. This study is a continuation of the research done by other researchers; this study has adopted methods from Uysal, et al. (2012), Arar, et al. (2017), Kim, et al. (2018) and Cassidy (2020). I chose to adopt their method of approaching text classification since in their studies they use Naïve Bayes to classify text into multiple classes. The figure below depicts the steps which was followed during the study process.

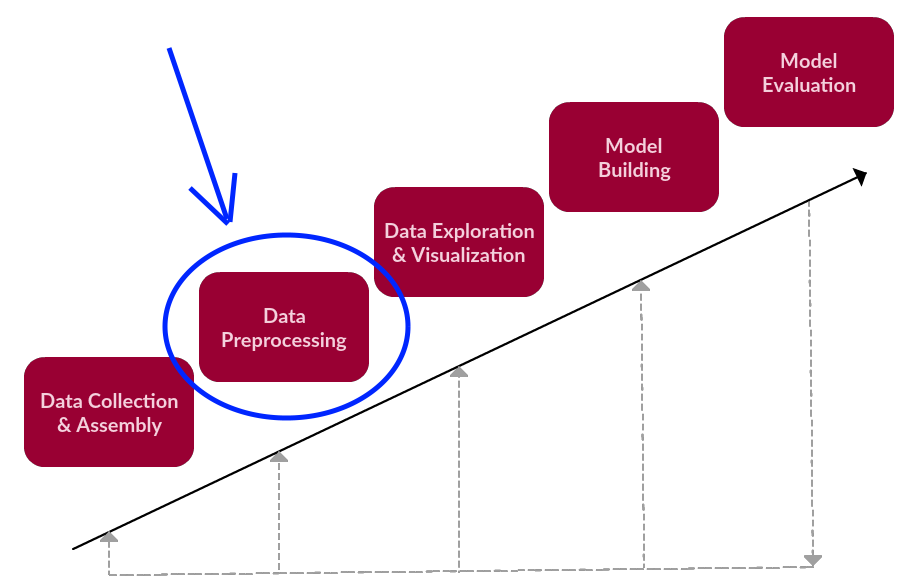


Figure 1: Text Classification steps, figure adopted from KDnuggets

As depicted in Figure 1, the first step which was data collection & Assembly was skipped since that was provided to us. The first step as depicted in Figure 1 is data pre-processing, in machine learning process data pre-processing is a process by which the data gets transformed or encoded to bring it to such a state that the machine can easily parse it, the features of the data can now be easily interpreted by the algorithm. I used these python libraries which are needed for pre-processing which are:

* Pandas – a library needed in order to read in the dataset into the Jupyter Notebook
* NLTK- the natural language toolkit is the most-used natural language processing (NLP) libraries in python, it is useful for tasks such as tokenization and part of speech tagging to mention a few. NLTK is used to remove the noise in the dataset which include common words known as stopwords, transforming all the text to lowercase, and removing part of speech as well as white spaces.

The cleaned data was then split into single features where each word is a feature, all the cleaned data was stored using a new name. The next step is data exploration & visualization, I explored and visualized the data that was cleaned from the previous step. The dataset two columns of importance namely Review and Label, the Review column is all the textual reviews collected from Coursera and the Label column is the ratings which was giving ranging from 1 to 5, 1 being very and 3 being fair and 5 being excellent. The following python libraries where used in this step:

* Matplotlib – is a visualization library I used to visualize the count of the Labels; how much reviews fall under each label.
* Numpy - is a mathematical library I used to the value of each label in the dataset, the distribution of the reviews in the dataset.
* WordCloud – used to visually represent the reviews, visualized the most frequent words in the dataset.
* MultiLabelBinarizer – I used this library to convert the cleaned data into vector of binaries, 1 for the word being present in the label/class and 0 for not. The textual data needs to be converted since text cannot be processed in the algorithms/model.
* Sklearn – This library was used to split the binary vector into training and test sets.

Model building is the step that follows, in this study I build a multinomial Naïve Bayes classifier to classify the reviews into the 5 classes/labels given. I followed a probabilistic approach which involves not using python libraries to do the classification. NB is a classification method which makes use of Bayes theorem, an instance’s posterior distribution is proportional to prior distribution and likelihood, it can be expressed using the following formula:

The prior which is the probability of , it is the total number of features divided by the total number of observations. I created a function groups the reviews into their respective classes. The likelihood ) was calculated which is the probability of a feature given that it is in a class divided by the total number of observations. And the posterior was calculated which is the probability that feature is in class X. is the probability of a class, this is dropped from the Bayesian theorem formula since the classes do not change. I implemented the Laplace smoothing technique to solve the problem of zero probability by applying this method the prior and likelihood probability can be written as:

Image for post

After all the probabilities were calculated the model was created by multiplying the priors with the likelihood.

A class was created with all the functions needed such as the prior, likelihood, Laplace smoothing and prediction. The training set was used to train the model. And the test was used to evaluate the model using the prediction function, this was done in the last step which is model evaluation. A Sklearn specifically the accuracy\_score python library was used to get the accuracy of the model. Although this was a probabilistic model some python were used doing the pre-processing stage, data exploration & visualization stage and the model evaluation stage.

1. **Analysis**

In the analysis section I will be follow more or less the steps depicted in Figure 1 to analyse and evaluate the results from the model. During the pre-processing step which was explained in the methodology. Figure 2 below depicts a sample of the data:

A screenshot of a computer

Description automatically generated

Figure 2: Sample of the data

The WordCloud in Figure 3 depicts the common words which are found in the dataset. As seen in the figure most of the words in the WordCloud are positive words.

A screenshot of a social media post

Description automatically generated

Figure 3: WordCloud

Figure 2 and Figure 3 don not fully depict the reviews, I printed out two random sentences as well as the label in which they in from the dataset to see how it looks like, Figure 4 below shows the two sentences.

A screenshot of a computer

Description automatically generated

Figure 4: Sampled sentences

As seen in Figure 5, the ratio of reviews is not equally distributed across the five classes, which makes me believe that the data is bias towards Label “5” which might affect the model. The chances of the model misclassifying Labels 1, 2 and 3 is very high since their values are fewer than that of Labels 4 and 5.

A screenshot of a computer

Description automatically generated

Figure 5: Distribution of Labels

The clean data from the pre-processing stage was passed through the MultiLabelBinarizer() in order to convert the text into a vector of binary values, Figure 6 below depicts the vector. As seen in the figure the vector contains 1’s and 0’s.

A screenshot of a computer

Description automatically generated

Figure 6: Vector of text

The overall accuracy and performance of the model was 74.2%, which is not bad nor is it good. It is not bad because it performed well given the data was bias on Label 5 and Label 4. But it also did not perform good since that bias might have affected the model and its accuracy. To show the bias I created a heatmap to depict the actual sentiments against the predicted sentiments.

A screenshot of a social media post

Description automatically generated

Figure 7: Heatmap of Model

1. **Conclusion**

From the given dataset apart from the bias most of the reviews were classified into Label 5 and Label 4 respectively. This means that the reviews and sentiments about the online classes was very good. A probabilistic approach to Naïve Bayes classification was a good classifier seeing that is works with both continuous and discrete data, it is highly scalable, and it is simple but not so easy to implement. The only problem I has is the fact that it took a long time to train the model. As pointed out in many literature naïve bayes performs better in text classification than other classification algorithms.

1. **References**
2. 2020. [online] Available at: <https://www.researchgate.net/publication/251255921\_An\_approach\_of\_the\_Naive\_Bayes\_classifier\_for\_the\_document\_classification\_1> [Accessed 27 July 2020].
3. Arar, Ö. and Ayan, K., 2020. A Feature Dependent Naive Bayes Approach And Its Application To The Software Defect Prediction Problem.
4. Borgelt.net. 2020. [online] Available at: <https://www.borgelt.net/papers/eufit\_99.pdf> [Accessed 27 July 2020].
5. Cassidy, C., 2020. Parameter tuning Naïve Bayes for automatic patent classification. World Patent Information, 61, p.101968.
6. Feng, G., Li, S., Sun, T. and Zhang, B., 2018. A probabilistic model derived term weighting scheme for text classification. Pattern Recognition Letters, 110, pp.23-29.
7. Fragos, K., Belsis, P. and Skourlas, C., 2014. Combining Probabilistic Classifiers for Text Classification. Procedia - Social and Behavioral Sciences, 147, pp.307-312.
8. Kim, H., Kim, J., Kim, J. and Lim, P., 2018. Towards perfect text classification with Wikipedia-based semantic Naïve Bayes learning. Neurocomputing, 315, pp.128-134.
9. Mao, C., Lu, L. and Hu, B., 2020. Local probabilistic model for Bayesian classification: A generalized local classification model. Applied Soft Computing, 93, p.106379.
10. Mayo, M., 2020. Text Data Preprocessing: A Walkthrough In Python - Kdnuggets. [online] KDnuggets. Available at: <https://www.kdnuggets.com/2018/03/text-data-preprocessing-walkthrough-python.html> [Accessed 27 July 2020].
11. Medium. 2020. Multinomial Naive Bayes Classifier For Text Analysis (Python). [online] Available at: <https://towardsdatascience.com/multinomial-naive-bayes-classifier-for-text-analysis-python-8dd6825ece67> [Accessed 27 July 2020].
12. Silva-Palacios, D., Ferri, C. and Ramírez-Quintana, M., 2018. Probabilistic class hierarchies for multiclass classification. Journal of Computational Science, 26, pp.254-263.
13. Uysal, A. and Gunal, S., 2012. A novel probabilistic feature selection method for text classification. Knowledge-Based Systems, 36, pp.226-235.