**COMP 6721**

**40092843** greeshmasunil10@gmail.com

Appendix

[1 Introduction 1](#_Toc25599998)

[2 Technical Details 2](#_Toc25599999)

[3 Significance 2](#_Toc25600000)

[4 Description of Architecture 3](#_Toc25600001)

[4.1 Input Data Set 3](#_Toc25600002)

[4.1.1 Training Model 3](#_Toc25600003)

[4.1.2 Testing Model 4](#_Toc25600004)

[4.2 Classification algorithm: Naïve Bayes Classification 4](#_Toc25600005)

[4.3 Analysis of Results of Experiments on the Model 4](#_Toc25600006)

[4.4 Experimenting with model parameters 5](#_Toc25600007)

[5 Description of any difficulties 5](#_Toc25600008)

1. Introduction

Machine learning has now become the answer to many practical real life problems. It has an extremely broad range of applications in such as spam/email filtering, decision making, categorizing data, sentiment analysis, medical diagnosis, image and pattern recognition and so on. Using this, we can create excellent algorithms by us of supervised/unsupervised learning techniques. This means that code need not be explicitly written. Text mining is an import application of this. There are a huge number of raw data stored in text format which can converted to useful information. It comes under Natural Language Processing (NLP). The challenge in this to create an optimal algorithm and to decide how to utilize this data to produce good results. In this era, Internet has been the major medium of communication and public content. Because of this, it has become a major source for applying machine learning applications and therefore it enables us to have a large amount of data which can be used for training and testing the machine. Using machine learning and NLP, it is now feasible to predict the category of posts once training the model. This is very important in marketing strategies and other business applications.

Hacker News dataset which is taken form Kaggle. It is a technical website which features a huge amount of data in the form of stories posted by authors. These are comment and liked upon by thousands of visitors. In this project, we use this dataset to predict the category of post of each story. The Hacker news data set of 2018 is used for training our machine learning model and the 2019 dataset is used as the testing data. The purpose of this project is to understand practical application on machine learning and NLP by using classification algorithms and to experiment with the algorithms to obtain an optimal solution for this problem.

1. Technical Details

This project is coded in python 3. Python is one of the most popular languages used for Artificial Intelligence and Machine learning applications. This is due to its simplicity and vast libraries which are very helpful for the AI algorithm implementation and ultimately makes the tasks easier. The libraries required for this project are Numpy, Pandas, Nltk, Matplotlib and math.

1. Significance

The scope of this project is import for various real world applications which are useful to evaluate the large amount of data stored in the internet. Classification of these unlabeled data would help easier evaluation and researches on the dataset.

1. Description of Architecture
   1. Input Data Set

All the input files required for this project are stored in the Resources folder, The input data set is taken from Hacker News using Kaggle. The main data used for the project is ‘hn2018\_2019.csv’. The format of the model is as follows,

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | Object ID | Title | Post Type |  | Author | Created At | URL | Points | Number of Comments |

Of this, we only consider the ‘Title’, ‘Post Type’ and ‘Created At’. The project uses pandas to read from this csv file and separates the dataset based on ‘Created At’ into training and testing data. All the data in 2018 is used for training and 2019 data is used for testing. Since the training model is very large, it will take a good amount of time to complete the run. Due to this, processing this data requires code optimization techniques and memory and an organized plan of execution. The Post Type is of four types, ‘story’, ‘ask\_hn’, ‘show\_hn’ and ‘poll’. It can be found from the dataset that most of the posts are of the type ‘story’, followed by ‘ask-hn’ and ‘show-hn’. ‘poll’ category is the least observed Post Type. The project also requires a list of stop words which are used for eliminating certain words which are not useful for classification. This is store in stopwprds.txt

* + 1. Training Model
       1. Pre-processing

The 2018 data in the file ‘hn2018\_2019.csv’ is used as the training model for this project. This data is enormous and therefore requires lot of time for processing and creating the training model. For purposes of tests and demonstration, I have extracted chunks of data of different sizes and stored in various csv files. Before training the classification model, the data needs to be preprocessed. That is, we need to make changes to the ‘title’ values by removing unwanted symbols and words. Secondly, the title need to be lemmatized for better mapping of words of the same meaning not the words in the testing data. In this project, I have used WordNetLemmatiser form nltk.stem. the preprocessing of the titles are done in the function def **preprocess\_model**(df): , where df is the data frame extracted from the input file for the year 2018. Once lemmatization is done, the titles are ready for tokenization.

* + - 1. Tokenization and frequency counter

For tokenizing the preprocessed titles, I have used nltk.word\_tokenize from nltk. The counter() converts this into set of unique words and it’s frequencies. The information required for each word is stored in ‘word’class. This stores the frequency count of the word in each post type, total count and conditional probabilities of the word in each class. The conditional probability is calculated using smoothed conditional probability and is given by p(wi/ci) = (frequency of wi in ci + smoothing value)/(total no.of words in ci + vocabulary size \* smooth value). Since the amount of words are very large, this probability is very small. Therefore, we take log10 for the model. The output of the training model is stored in baseline-result.txt

* + 1. Testing Model

The testing model is extracted from the main dataset using the year as 2019. The testing model titles are also preprocessed in the same way as the training model before evaluation. The words in the testing model are matched with the training model to obtain the smoothed conditional values of the word for each class. A score value is calculated for each post type using the formula score of ci = prior probability of the class ci + sum smoothed conditional probabilities of all the words in ci for that title. The log of this value is taken to prevent numerical underflow and for better comparison. The predicted value generated is the value with the highest score value. This is called Niave bayes classification model.

* 1. Classification algorithm: Naïve Bayes Classification
  2. Analysis of Results of Experiments on the Model
     1. Base-line
        1. Analysis
        2. Result
     2. Stop word filtering
        1. Analysis
        2. Result
     3. Word Length filtering
        1. Analysis
        2. Results
     4. Infrequent word filtering
        1. Analysis
        2. Results
     5. Smoothing
        1. Analysis
        2. Results
  3. Experimenting with model parameters

1. Description of any difficulties

The amount of data in the shapefile was large and therefore it took a good amount of time for the calculation of the crime rates and plotting the graph. This was inconvenient since it was taking more time for bug fixing and testing. Also, there was a learning curve for python libraries and the graphical visualization of data.