CSE 498 Text Mining

Fall 2016

Project 1

Sachin Joshi

Task 2

To accomplish this task I have used Java programming language. Initially the reviews are tagged using the Stanford POS Tagger. However, I have used only the first 100 reviews to show the functionality of this task. The first 100 reviews are tagged using the Stanford POS Tagger and stored in the file **Tagged_100_Reviews.txt**. However, I have tagged all the reviews using the Stanford POS Tagger and stored in the file **Tagged_Reviews.txt**.

The Java file **AttributesExtract.java** contains the code for extracting the attributes from the text file **Tagged_100_Reviews.txt**. This Java program reads the text file **Tagged_100_Reviews.txt**, extracts the attributes, prints the extracted attribute and stores the extracted attributes in a new text file named **Attributes_Extract.txt**. We must create a new text file and rename it as **Attributes_Extract.txt** before running our program to store the extracted attributes.

Note: All the above mentioned files, i.e., **AttributesExtract.java (Java Program), Tagged_100_Reviews.txt (Input Text File)** and **Attributes_Extract.txt (Output Text File)** must be stored in the same folder.

To compile the above Java file, we use the following syntax in the command prompt:

javac AttributesExtract.java

Once the file is successfully compiled, a .class file named **AttributesExtract.class** is created and we can run the program using the following syntax in the command prompt:

java AttributesExtract

The next step is to count the frequency of each extracted attribute. To accomplish this task, I have designed a simple Java program to count the frequency of each extracted attribute in an input file. The associated Java file is **CountAttributesFrequency.java** and the input file is **Attributes_Extract.txt**.

In this program, there are two classes-**Counter** and **CountAttributesFrequency** class. In the Counter class, two data structures are implemented- **LinkedList** and **TreeMap**. The Counter class has four methods. The first method, **readWords()** is called to read the contents from the input file and split the content into words. Before adding these words to the LinkedList, the regular express is used to filter words by removing all symbol characters from the words. The **Pattern** class is used to define the pattern to be matched. The string pattern "\\W+" matches any

symbol except underscore in a word. The **Matcher** class is able to remove the symbols from the words that match the string pattern. The second method is called **countWords()**. This method uses two loops to process all words and count the word frequency. The **TreeMap** is used to store the unique words and their frequencies. The words are stored automatically in **TreeMap**. The **addToMap** method is called by the **countWords()** method to add words and frequencies to the TreeMap. The **showResult** method is invoked after the words and frequencies are added to the TreeMap to show the table of the words, frequencies, and the percentages and further to write the results to two different text files namely **Attributes.txt** and **Frequency.txt**. **Attributes.txt** file contains the unique attributes without any repetition and the **Frequency.txt** file contains the frequency of the corresponding attribute.

The final method, **processCounting** combines the methods above in a single code block. This method will be invoked from the CounterAttributesFreqency class to start analyzing the content of the input file and displaying the word frequency table. Finally the attributes and the frequency of each attribute is stored in separate text files namely **Attributes.txt** and **Frequency.txt respectively**.

The program can be compiled using the following syntax in the command prompt:

javac CountAttributesFrequency.java

Once the above Java file is compiled, we can run the program using the following syntax in the command prompt. We need to provide the input text file **Attributes_Extract.txt** as the command line argument while running the program as described below:

java CountAttributesFrequency Attributes_Extract.txt

Finally, the desired results of this task, i.e., the top 50 attributes with highest frequency are stored in the file named **Project_1_Task_2.csv**.