Kuzey Cimen - kc18182 – 1803189

1: Your proposed text classification method(you can use relevant NLTK text classification functions or other text classification functions)

In this assignment I have used Naïve Bayes Classifier using
“from nltk.classify import NaiveBayesClassifier“.

Movie_reviews data set is imported using “from nltk.corpus import movie_reviews”.

Document list is created by using the movie_reviews categories and fileids inside a nested for loop. The categories for the movies and the fileids of the categories are used in generating a list of tuples that contain the word from the fileid and its category (list(movie_reviews.words(fileid)), category).

The docment is then shuffle so the training sets and the test sets are unbiased.

2: Pre-processing methods such as removing stop words, punctuations.

Convert all words to lowercase, I have used the built-in lower method to a string and looping through the words list in the movie_reviews.

To remove the stop words, I have used “from nltk.corpus import stopwords” and check if the current word in the movie corpus that is checked is not a stop word. Then lowercase it and append it to a list.

To remove the punctuations, I have used “from nltk.tokenize import RegexpTokenizer”. Tokenizing using a regex that checks if it is a word.

To further improve the accuracy, I have decided to lemmatize each word in the list after removing the punctuations by using the lemmatize function from “WordNetLemmatizer()” which is imported “from nltk.stem.wordnet import WordNetLemmatizer”. Lemmatized words are put in a list. This will be the final step in pre-processing.

3: Feature selection methods such as selection the 3000 most important words.

There is a feature finding function called “find_features()” which gets the document variable as a parameter. A dictionary is created (key: word, value: true/false) and loops through the most important 3000 words. Then, checks if the word is in the document parameter that was given and sets it as a value in the created dictionary with the key being the word that is being checked. After that, returns the dictionary.

Frequency Distribution is created using the last list in the pre-processing where all the needed pre-processing is done.

Then, top 3000 most important words are selected.

A list is created by looping through the document list and find_features() function is called

4: You should evaluate your model on 10 percent of movie review model, please report the precision, recall, accuracy, and F-score of each class.

There is an evaluation function called “evaluate()” which creates two dictionary sets (key: pos/neg, value: set of current loop index), one being for the training set “temp_train” and the other one being

for the testing set “temp_test”. The loop inside the function loops through the main testing set (not the one that is declared in the function). Inside the loop, “temp_train” and “temp_test” dictionaries are getting filled with the key being label for “temp_train” and classified feast for “temp_test”. The values are I (index of the loop). After that, a tuple is returned which includes the “temp_train[type]” and “temp_test[type]” with type being the input parameter (pos / neg). This will be used in getting the precision, recall and f_measure for pos and neg.

Training set is 90% and testing set is 10%. This means I do featuresets[:1800] to get 90% and featuresets[1800:] for the 10%.

First model is trained by using “classifier = NaiveBayesClassifier.train(training_set)”

For the Accuracy, I have used “from nltk.classify import accuracy“ called the accuracy function with the paramenets “classifier” and “test_set”, multiplied by 100 to get the percentage.

To get the Precision, Recall and F-score I have used
“from nltk.metrics import precision, recall, f_measure“

For each class (pos / neg) have their own precision, recall, accuracy, and F-score. So I have printed them seperately first pos precision, recall, accuracy, and F-score then neg precision, recall, accuracy, and F-score.