UTRECHT UNIVERSITY

INFO-MCM

Assignment 2

Submitted To:
Dr. R.W.F. Nouwen
Linguistics

Submitted By :
Adrian Briceno A. 6189865
Peter Blomsma 5909384

1 Introduction

In this document we report our results from the lab assignment for Bayesian Modeling [1]. The code that we made for this assignment can be obtained at [2].

2 Question 1: dependency between words

Taking the probabilities assigned to the individual words in a text, such that the probabilities are independent from one another does not hold in reality for various reasons including:

1. Syntax: the probability that a word occurs is partly defined by the grammatical structure. E.g. the movie reviews in this assignment should follow a good grammatical structure in order to be understandable. We can expect that in the text in a sentence there is a relationship between the previous worlds and the next word. For example in this sentence: "We are going to ", we normally expect that the probability of the word "*Utrecht*" was a greater probability than the word "*ant*", mainly because it doesn't makes sense in the sentence, so there is a dependency, between each probability.
2. Semantics: words can have different meanings depending on the semantical context. For example: "The class is terrible" is very negative, but "The class is terribly nice" is very positive. The sentiment of terrible thus dependent on the semantical context.

3 Question 2: Classifier based on Stopwords

The answer for this question is divided in three parts. First the training phase is described, second the testing of the model and finally the results.

3.1 Training

The Naive Bayes sentiment classifier is created with the following equation:

$$Prob_C = freq_C / tfreq \tag{1}$$

$$classifier(\langle w_1, \ldots, w_k \rangle) = argmax_{c \in C}[\prod_{1 \leq i \leq k} P(w_i|c)]$$

Figure 1: Formula used for the Naive Bayes sentiment classifier.

Being $Prob_C$ the probability of the word classified as C. Where C is either positive or negative. $freq_C$ is the frequency of the word classified as C and $tfreq$ the total frequency of that word. We created a function named $trainModel$ that applies this formula on the training data and adds the results a new column named prob to the training data. The data to train the Naive Bayes sentiment classifier is provided to us by the assignment [1]

3.2 Testing

In order to test the classifier, we followed the suggestions of the assignment [1] and executed the following steps:

1. The test text is tokenized.
2. The test text is intersected with the training text.
3. The result is intersected again with the words in the stopwords.txt file.

The above three steps are programmed in the function named *preprocess*2.

After the test text is preprocessed by the function *preprocess*2, only the important words remain with the probability of those words of being positive or negative sentiment. The $classifier$ function is used to calculate the probabilities of the test text belonging to either the positive or the negative sentiment class. The calculation is done by the product of the probabilities of all words for each class. The class with the highest probability is returned. The specific formula used for this function is depicted in figure 1. A for loop is used to classify every text in the data set.

3.3 Results and evaluation

In order to evaluate the results of the testing phase, we created the function *statistics*. This function evaluates the performance of the classifier based on the results of the testing phase. This function calculates the number of true positives for class c, the number of false positives for class c, the number of false negatives for class c and the number of true negatives for class c. And with this numbers calculates the accuracy, precision, recall and F1 and prints them. The results are stated in table 1.

Statistics	Accuracy(%)	Precision(%)	Recall(%)	F1(%)
For classification 0	51.5	63.6	7	6.3
For classification 1	51.5	50.7	96	33.21

Table 1: Evaluation of the Naive Bayes sentiment classifier for question 2

4 Question 3: Classifier based on Adjectives

The approach for question 3 is the same as question 2. The difference is that the function *preprocess*3 is used for testing instead of *preprocess*2. *preprocess*2 intersects the intersection of the text with the training data and the adjectives.txt. We can see the evaluation of our classifier in table 2.

Statistics	Accuracy(%)	Precision(%)	Recall(%)	F1(%)
For classification 0	67	79.3	46	29.1
For classification 1	67	61.9	88	36.6

Table 2: Evaluation of the Naive Bayes sentiment classifier only taking intersection with adjectives for question 3

5 Question 4: Classifier based on Sentiment Words

We use the same approach as question 2, but instead of *preprocess*2 we used *preprocess*4 for the test text. This function intersects the intersection of the text with the training data and the sententwords.txt. The performance measurements are evaluation for this classifier are in table 3.

Statistics	Accuracy	Precision	Recall	F1
For classification 0	77.5	85.7	66	37.2
For classification 1	77.5	72.3	89	39.9

Table 3: Evaluation of the Naive Bayes sentiment classifier only taking intersection with sentiment words for question 4

Now that we have the results for the three different intersections, we can see how well it compares our classifier with the intersection of the sentiment word list. We can see the accuracy of question 4 is more than 20% compared to question 2 and more than 10% compared to question 3. For the classification of the both sentiment we can see that the Precision, recall and F1 improves more than 5%.

6 Question 5: Naive Bayes classifier Improvement

The Naive Bayes classifier can be improved in multiple ways:

1. Instead of looking at independent words, the classifier could look at word pairs. Looking at word pairs would result in a more context aware classifier, that could be sensitive to the differences between 'terribly nice' and 'terribly bad'.
2. The classifier of question 5 is not sensitive for negations. Therefore the classifier does not see a difference between 'not bad' and 'bad'. This could be improved by making the classifier negation aware.
3. The classifier could be optimized by using better preprocessing:
    - From the results of question 2 can be derived that stop words are not a reliable indicator for sentiment in a text.
    - From the results of question 3 can be derived that adjectives are good indicators of sentiments.
    - From the results of question 4 can be derived that the sentiment words are a powerful way to derive the sentiments.

For this report, we improved the Naive Bayes classifier by implementing option 2 and 3. For option 2, all words that are preceded by "not" are deleted including "not" itself. For option 3, (i) all stop words are deleted from the text, (ii) only adjective words and sentiment words available in the training model are used.

Statistics	Accuracy(%)	Precision(%)	Recall(%)	F1(%)
For classification 0	78.0	86.8	66.0	37.5
For classification 1	78.0	72.6	90.0	40.2

Table 4: Question 5: Performance measures for the improved Naive Bayes sentiment classifier

The improved classifier is tested as in the previous questions. The results are slightly better (see table 4), as all performance measurements have a slightly higher score than the measurements of the previous questions.

7 Question 6: Application on other Domains

The sentiment classifier as trained in question5 can be applied in different contexts, but the performance cannot be guaranteed. The following factors will influence the performance:

1. Language: The sentiment classifier should be trained on the same language as the language of the corpus it will be used on.
2. Domain specific: The classifier will have the best performance if the training and test sets are from the same domain. For example, if the model is trained on movie reviews and tested on vacuum-cleaner reviews it could have a lower performance than in the case it is tested on movie reviews. The reason is that each domain has specific words related to both sentiment and the domain.
3. Domain specific: The classifier will have the best performance if the training and test sets are from the same domain. For example, if the model is trained on movie reviews and tested on vacuum-cleaner review it supposedly has a lower performance than in the case it is tested on movie reviews. The reason is that each domain has specific words related to both sentiment and the domain.
4. Authors: This factor is related to the domain. The level of language-acquisition of the writer of the text influences the performance. If writers of the trainingset have a different writing-level (and make for example more grammar and spelling errors) than writers of the test set, the performance of the classifier will be negatively influenced.
5. Amount of data: In general it stands that the more texts are used to train the model, the better the performance of the classifier. The more texts a classifier is trained upon, the higher the chance is that classifier knows sentiment words.

In order to test how the classifier from question 5 performs the classification of text from a different source and a slightly different domain. The classifier is tested on 10 positive and 10 negative reviews from Amazon for the product: Harry Potter and the Sorcerer's Stone. This is a DVD movie. The data is available here [2]

Statistics	Accuracy(%)	Precision(%)	Recall(%)	F1(%)
For classification 0	80.0	100.0	60.0	41.7
For classification 1	80.0	71.4	100.0	41.7

Table 5: Question 6: Performance measures for the improved Naive Bayes sentiment classifier on the amazon test set.

The performance measurements are stated in table 5. The measurements at least as good as the results of question 5, except the precision for the positive sentiment.

8 Conclusion

Overall we see that our naive classifier got good results even when we choose to use the probabilities as independent of each other. We can see that the classification of negative or positive sentiment, doesn't depend on the stop words or the adjectives, that is why we got a better evaluation in question 4. Question 4 took the hand-collected list of English words into consideration that are known to have relatively strong (positive or negative) sentiment. In our improved classifier, we can see that (i) we have a small but significant improvement in the performance measurements based on the test set of the assignment and (ii) we have a great evaluation in our Harry Potter text reviews. A graph with all an overview of all the performance measurements is plotted in figure 2. The classifier can be improved by further taking our suggestions into account of question 5 and 6. To do this, more training data is needed and as well more time to develop.

References

[1] All code that is used for this assignment is available on github. `http://ricknouwen.org/baylab17.html`. Accessed: 2018-01-05.

[2] All code that is used for this assignment is available on github. `https://github.com/pblomsma/infomcm/tree/master/Assignment_2`. Accessed: 2018-01-05.

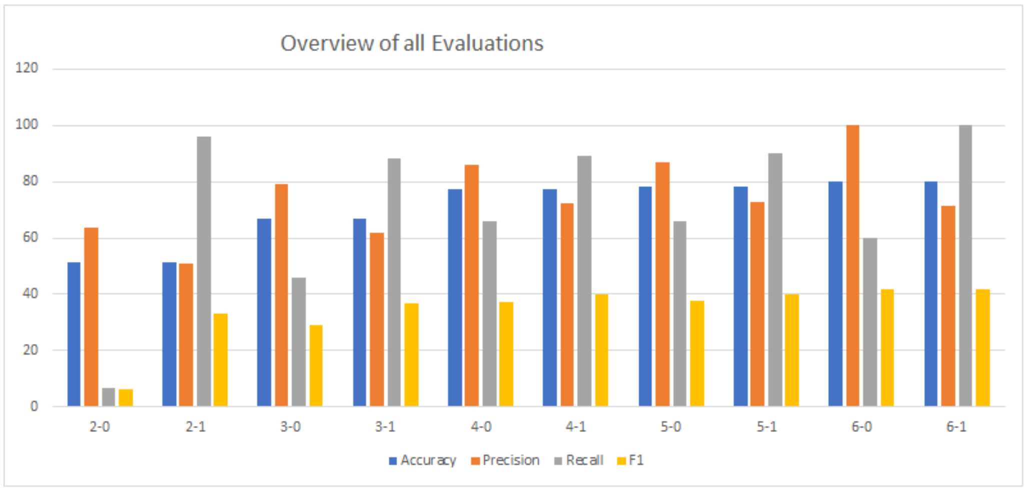

Figure 2: Overview of all performance measurements for the different classifiers.