Homework 3: Building BabyClassifier - Naive Bayes for Sentiment Analysis

Due: October 11, 2024 (2:00pm)

1 Recap: Naive Bayes Classifiers

A Naive Bayes classifier is a probabilistic machine learning model used for classification tasks. The classifier is based on applying Bayes' theorem with strong (naive) independence assumptions between the features. For example, consider a review $R = ['I', 'love', 'this', 'restaurant']$. Your Naive Bayes model will calculate the probability of each class (positive or negative sentiment) given this sequence of words, assuming each word is independent of the others.

More formally, we can write this as $p(C|w_1, w_2, ..., w_n)$, where C is the class (sentiment) and w_i are the words in the review. In this homework, you will first need to create a Naive Bayes model and then use it to classify a set of reviews.

2 Creating a Naive Bayes Model

Given the input CSV file 'training_set.csv', you should use Python dictionaries to calculate and store $p(w_i|C)$ and $p(C)$. In practice, this involves keeping a count of word occurrences for each class, and then using that to calculate the probabilities of each word given the class, as well as the prior probabilities of each class.

HINT: Use nested dictionaries to keep track of counts and probabilities.

3 Classifying the Sequence

Now that you have a Naive Bayes Model, you can start classifying reviews. Your classifier should be able to handle reviews of any length. You will use a maximum-likelihood estimate (MLE) to select the most probable class for each review. This simply means that given any review, you will pick the class (sentiment) with the highest probability.

4 Evaluation

Calculate the following metrics for your Naive Bayes classifier:

- Accuracy
- Precision
- Recall
- F1 Score

Compare these metrics for:

- The Naive Bayes model without any text preprocessing
- The Naive Bayes model with lemmatization
- The Naive Bayes model with both lemmatization and stopword removal

How do these different preprocessing steps affect the performance of your classifier?

5 Submission

For your final submission, you should submit a Colab notebook (or a .py file) with your work. Additionally, submit a Word document with your observations, including:

1. A brief explanation of your implementation choices
2. The performance metrics for each version of your model
3. An analysis of how lemmatization and stopword removal affected your model's performance
4. Any challenges you faced and how you overcame them

Your submission should also include the 'test_set_group_X.csv' file with your best model's predictions, where X is your group number.