COMPSCI-589: Homework 2

Will Lillis

March 23, 2023

Question 1

Note: As outlined in this campuswire post, I'll make a quick note here to note how I handled "cases where the test documents contain words that are not in the vocabulary." In such cases, I had the model simply ignore the word entirely.

Standard Implementation:

i)

Evaluating on 20% of the positive and negative test sets yielded an accuracy of

$$\frac{TP + TN}{n} = 0.5642$$

ii)

Evaluating on 20% of the positive and negative test sets yielded a precision of

$$\frac{TP}{TP + FP} = 0.5630$$

iii)

Evaluating on 20% of the positive and negative test sets yielded a recall of

$$\frac{TP}{TP + FN} = 0.5552$$

iv)

The confusion matrix for the evaluation of this implementation is as follows:

	+	−
+	1367	1095
−	1061	1424

Logarithmic Implementation:

i)

Evaluating on 20% of the positive and negative test sets yielded an accuracy of

$$\frac{TP + TN}{n} = 0.5775$$

ii)

Evaluating on 20% of the positive and negative test sets yielded a precision of

$$\frac{TP}{TP + FP} = 0.5772$$

iii)

Evaluating on 20% of the positive and negative test sets yielded a recall of

$$\frac{TP}{TP + FN} = 0.5646$$

iv)

The confusion matrix for the evaluation of this implementation is as follows:

	+	−
+	1390	1072
−	1018	1467

Discussion)

Reviewing the above results, it would appear that classifying instances by computing log-probabilities, instead of simple probabilities does (positively) affect the model's performance. In the results from above, all metrics of interest (accuracy, precision, and recall) were better with the log probability model as compared to the simple probability model. While the performance gains were somewhat small in this experiment, the margin widens greatly in other evaluations completed later on for the assignment (particularly as the training set grew in size).

Assuming that this transformation does have an impact on performance, referring to the above results it would appear that this trick affects the precision metric most strongly, with an improvement of 0.0142 (as compared to accuracy's improvement of 0.0133 and recall's improvement of 0.0094). It seems reasonable that this is the case, as the logarithm "trick" aids in classifying by preventing probabilities from being "smashed" down to 0 due to the finite precision of floating point numbers. As this effect is mitigated (via the use of log probabilities), a larger portion of positive classifications will be correct, as they will more often be the result of an actual computed (log) probability, rather than a flip of a coin due to the aforementioned numeric computational barriers.

Question 2

First using a value of $\alpha = 1$ yielded an accuracy of 0.6817, precision of 0.6943, recall of 0.6437, and the following confusion matrix:

	+	−
+	1622	898
−	714	1830

Next, after evaluating the model for values $\alpha = 0.0001, 0.001, \ldots, 1000$ we have the following accuracy results:

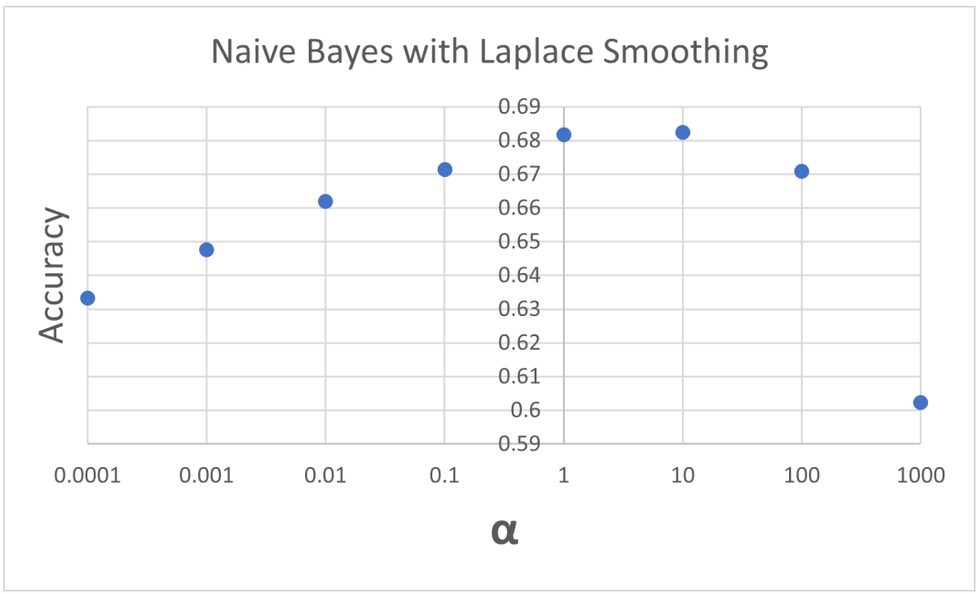

Fig. 1: A plot of the accuracy of my implementation of the Naive Bayes with Laplace Smoothing for various values of α.

Discussion)

As mentioned in the assignment description, Laplace smoothing "works" intuitively by having the algorithm pretend that we have seen that word α additional times in the training set. Thus, the concave downwards shape of the graph seems to fit relatively well. At the low end of the graph (when α is too low), accuracy suffers because the algorithm is pretending to have seen too few instances of the word in the training set to have any measurable impact on the classification. That is, the change introduced by the smoothing is insignificant compared to the other terms in the algorithm. Similarly, on the high end of the graph, (when α is too high), accuracy suffers because the algorithm's pretend sightings (applied uniformly to all words presented from the test set) are now overwhelming the impact of the actual data set. Thus, patterns that the algorithm could have "picked out" from the training data are being "averaged out" by the overly large α value. Accuracy sees the best improvement in the middle "Goldilocks" region, in which α is neither too large or too small.

Question 3

For this question, I utilized Laplace Smoothing with a value of $\alpha = 10$, as this provided the best performance in the previous question's testing.

i)

Evaluating on 100% of the positive and negative test sets yielded an accuracy of

$$\frac{TP + TN}{n} = 0.8426$$

ii)

Evaluating on 100% of the positive and negative test sets yielded a precision of

$$\frac{TP}{TP + FP} = 0.8762$$

iii)

Evaluating on 100% of the positive and negative test sets yielded a recall of

$$\frac{TP}{TP + FN} = 0.7978$$

iv)

The confusion matrix for the evaluation of this implementation is as follows:

	+	−
+	9973	2527
−	1409	11091

Question 4

For this question, I utilized Laplace Smoothing with a value of $\alpha = 10$, as this provided the best performance in Question 2's testing.

i)

Evaluating on 100% of the positive and negative test sets yielded an accuracy of

$$\frac{TP + TN}{n} = 0.8372$$

ii)

Evaluating on 100% of the positive and negative test sets yielded a precision of

$$\frac{TP}{TP + FP} = 0.8718$$

iii)

Evaluating on 100% of the positive and negative test set yielded a recall of

$$\frac{TP}{TP + FN} = 0.7907$$

iv)

The confusion matrix for the evaluation of this implementation is as follows:

	+	−
+	9884	2616
−	1454	11046

Discussion)

Moving from Question 3's to Question 4's respective confusion matrices, the following changes were observed:

	+	−
+	−0.8924%	−3.522%
−	−3.194%	+0.4057%

Looking at the percent changes displayed above, it would appear that the positive class was affected more by changing the size of the training set. The total (absolute value) percent change for instances of the positive class (true positives and false negatives) was 4.414%, while for instances of the negative class (false positives and true negatives) it was 3.600%.

Question 5

Under the assumption that people are utilizing these ratings in order to decide which movie to watch, it makes sense that we would want to avoid dissuading people from watching potentially good movies. In order to accomplish this, we would need to avoid presenting a negative review as a positive review for a given movie. That is, we would like to minimize the number of false negatives our model makes. As the precision of the model is computed via

$$\frac{TP}{TP + FP},$$

minimizing the number of false positives (FP) corresponds to maximizing the precision of the model. Therefore, I would argue that it is more important to have high precision.

Question 6

Training the model with an unbalanced data set (10% of available positive training instances, 50% of available negative training instances) and a value of $\alpha = 10$ yielded an accuracy of 0.5002, precision of 0.8571, recall of 0.00048, and the following confusion matrix:

	+	−
+	6	12494
−	1	12499

Discussion)

This model's performance is drastically different as compared to that of the model displayed in Question 4. The model almost never classified any instance as a member of the positive class, and this is reflected in the confusion matrix's comparably low values for true and false positives. Intutively this seems reasonable, as the training dataset was heavily unbalanced, containing many instances of the negative class and comparatively few of the positive class. As a result of this imbalance, the algorithm tended to classify nearly all instances as a member of the negative class. Again, this is reflected in the the low values for true and false positives. Furthermore, as nearly all classifications were negative while the underlying test dataset was still split evenly between the two classes, it also makes intuitive sense that the remaining classifications were "split" evenly between true and false negatives.

Collaboration:

The assignment description asks us to explicitly list all students with whom you discussed this assignment.

- I spoke with Pranav Shekar regarding how to implement Laplace Smoothing.