CS 124 / LING 180 From Languages to Information
Dan Jurafsky, Winter 2020
Week 3: Group Exercises on Naive Bayes and Sentiment - Solutions
Jan 21, 2020

Part 1: Group Exercise

We want to build a naive bayes sentiment classifier using add-1 smoothing, as described in the lecture (not binary naive bayes, regular naive bayes). Here is our training corpus:

Training Set:

```
- just plain boring
- entirely predictable and lacks energy
- no surprises and very few laughs
+ very powerful
+ the most fun film of the summer
```

Test Set:

```
predictable with no originality
```

1. Compute the prior for the two classes + and -, and the likelihoods for each word given the class (leave in the form of fractions).

   |V| = 20, n- = 14, n+ = 9
   P(-) = 3/5, P(+) = 2/5
   P(and | -) = (2 + 1) / (14 + 20) = 3/34
   P(any_other_vocab_word_in_-_sentence | -) = (1 + 1) / (14 + 20) = 2/34, e.g. P('plain' | -)
   P(any_vocab_word_not_in_-_sentence | -) = (0 + 1) / (14 + 20) = 1/34, e.g. P('powerful' | -), P('with' | -)
   P(the | +) = (2 + 1) / (9 + 20) = 3/29
   P(any_other_vocab_word_in_+_sentence | +) = (1 + 1) / (9 + 20) = 2/29, e.g. P('powerful' | +)
   P(any_vocab_word_not_in_+_sentence | +) = (0 + 1) / (9 + 20) = 1/29, e.g. P('plain' | +), P('with' | +)

2. Then compute whether the sentence in the test set is of class positive or negative (you may need a computer for this final computation).

   C = {+, -}
   P(c | "predictable with no originality") ∝ P(c) * P ("predictable with no originality" | c) = P(c) * P(predictable | c) * P(with | c) * P(no | c) * P(originality | c) ~= P(c) * P(predictable | c) * P(no | c), 'with' and 'originalty' are unknown
   P(- | "predictable with no originality") = (3/5) * (2/34) * (2/34) = 0.002076
   P(+ | "predictable with no originality") = (2/5) * (1/29)^2 = 0.0004756
   P(- | "predictable with no originality") is greater, so the test set sentence is classified as class negative.

3. Would using binary multinomial Naive Bayes change anything?

   *No, using binary NB would not change anything - under this scheme n+ = 8, P(+ | "predictable with no originality") = (2/5) * (1/28)^2 = 0.0005102, which is still less than P(- | "predictable with no originality").*

4. Why do you add |V| to the denominator of add-1 smoothing, instead of just counting the words in one class?

   In add-1 smoothing we assume we have seen each word once regardless of whether they appear in the original class or not and thus add |V| to the denominator. Note that words that do not appear in the training set are 'unk' and we just pretend they aren't there, and they are not included in the vocab.

5. What would the answer to question 2 be without add-1 smoothing?

   P(c | "predictable with no originality") = 0 for class positive because at least one of the words in the test set does not appear in the positive train examples; P('predictable' | +) = P('no' | +) = 0.

6. Can you think of any other features (or preprocessing) that you could add that might be useful in predicting sentiment? (This will come in handy for the next PA!).

   We leave these for you to consider in PA3

Part 2: Challenge Problems

7. Ethics question: For discussion

8. Go to the Sentiment demo at http://nlp.stanford.edu:8080/sentiment/rntnDemo.html. Come up with 5 sentences that the classifier gets wrong. Can you figure out what is causing the errors?

   One example that the classifier gets wrong: "I don't not like you." The double negation is interpreted incorrectly.

9. It is sometimes the case that more complex features (like trigrams or bigrams) perform better than simple features (like unigrams) on the **training** set, but perform worse than simple features on the **test** set. This is a particular case of the phenomenon called `overfitting' in machine learning. Discuss why this might be. Can you create a tiny training set with 2 3-word documents and a test set with one document for which this overfitting situation holds?

   In overfitting, your model is too complicated for the small amount of data you have, and the model will fit to random patterns in the data. So a complex feature might just occur accidentally in the *training set, but will give it a very high probability. Such a rare `accidental' feature might never occur in the test set, or if it does might simply randomly occur with the other class.*

10. Binary multinomial NB seems to work better on some problems than full count NB, but full count works better on others. For what kinds of problems might binary NB be better, and why? (There is no known right answer to this question, but I'd like you to think about the possibilities.)

    Binary NB works better when word occurrence is more important than word frequency, such as in sentiment classification.