COMP 550 A1

Yue Violet Guo

(260606976)

1 Ambiguity

- Every student took a course

The subject object agreement is not clear. It could mean each student took their own chosen courses, or everyone took that exact same one course. This is semantics ambiguity. It is not clear how "student" and "course" are quantified. A human or machine needs to know whether students ⟵ course(s) is a many to one or many to many relation.

- John was upset at Kevin but he didn't care.
  - The ambiguity is pronoun reference. Who is "he" referring to?
  - The "he" has two possible cases. It could be Kevin doesn't care, and John is upset at the act of not caring. Or it could be John is upset at Kevin, but Kevin doesn't care about John's feelings.
  - Pragmatics. Under each different context, the meaning of "he" changes. After conducting some research, this particular pragmatic ambiguity of pronouns or adjective of pronouns is also called anaphoric ambiguity. "He" refers to a male. It is possible to use "he" to refer to all possible male subjects in the text.

need the linguistic adj, lexical? phonological?

  - We can either specify what he didn't care about, such as "... but he didn't care about Kevin". This would clarify that he is not Kevin. Or we can specify who "he" refers to, "... but Kevin didn't care."

- Sara owns the newspaper
  - The ambiguity is "newspaper"
  - We can interpret Sara owns a copy of newspaper issued on a certain day, or Sara owns the company that produces a particular newspaper.
  - Lexical. The word for a publisher and a printed copy of an issue of a newspaper has the same form.
  - The machine needs to know more modifiers or nouns, whether the newspaper is the publisher that produces this particular newspaper, or an issue of a newspaper, such as "newspaper publisher" versus "this pile of newspaper." A human may infer the status of Sara from more descriptions, and distinguish the kind of newspaper she owns.
  - (TODO)

- He is my ex-father-in-law-to-be.
  - The order to parse the phrase, and which modifier in the phrase "ex-father-in-law-to-be" goes first is unclear.
  - The difference interpretations are "(ex)-(father)-(in-law)-(to-be)", a previous partner's father's future in law; or "ex-(father-in-law)-to-be", the subject's potential father-in-law from a previous relationship.
  - Syntactical. The order of adjectival modifier is undefined in the phrase.

- A machine needs to know the dependency structure of the phrase "ex-father-in-law-to-be." It needs to learn how to propoerly group between lexicons the hyphens, and which one modifies which one(s). A human may be able to infer the syntactical hierarchy from more descriptions of the subject "he", or the "ex" in the phrase. add some more

- ttyl ;)
  - This spelling is not defined in a proper English dictionary. And this punctutaion ';)' is not a defined combination in standard English.
  - We can interpret ttyl as a new word, or 'T.T.Y.L.' that represents the acronym of 4 words. The ';)' represents an emoji, or new hybrid punctuation.
  - Orthographical. This spelling is not defined in a proper English dictionary.
  - A human or machine needs proper spelling to disambiguate. Use "T.T.Y.L." to clearly indicate that it is an acronym. For ";)", a human may need the instruction to tilt our screen to read this hybrid punctuation as an emoji. A machine can also learn emoji representations by having access to training corpus with labels of emojis.

2 Naive Bayes and Logistic Regression

We look at a specific case of naive bayes with bernoulli distribution. As explained in lecture, a case where we have more than 2 categories can also be proven.

Let $P(y=1)=\mu$, $P(x_i|y=1)=\theta_{+i}^{x_i}(1-\theta_{+i})^{1-x_i}$, $P(x_i|y=0)=\theta_{-i}^{x_i}(1-\theta_{-i})^{1-x_i}$

$$
\begin{aligned}
P(y=1|x) &= \frac{P(y=1)\prod_i P(x_i|y=1)}{P(y=1)\prod_i P(x_i|y=1) + P(y=0)\prod_{i\in x} P(x_i|y=0)} \\
&= \frac{1}{1+\frac{P(y=0)\prod_i P(x_i|y=0)}{P(y=1)\prod_i P(x_i|y=1)}} \\
&= \frac{1}{1+exp(log(\frac{P(y=0)\prod_i P(x_i|y=0)}{P(y=1)\prod_i P(x_i|y=1)}))} \\
&= \frac{1}{1+exp(log(\frac{\mu}{1-\mu}) + log(\frac{\prod_i (\theta_{-i}^{x_i}(1-\theta_{-i})^{1-x_i}}{\prod_i \theta_{+i}^{x_i}(1-\theta_{+i})^{1-x_i}}))} \\
&= \frac{1}{1+exp(log\frac{\mu}{1-\mu} + \sum_i x_i(log\frac{\theta_{-i}}{\theta_{+i}} - log\frac{1-\theta_{-i}}{1-\theta_{+i}}) + \sum_i log\frac{1-\theta_{-i}}{1-\theta_{+i}})}
\end{aligned}
\tag{1}
$$

We can rewrite the formula for Naive bayes into the same form as logistic regression, $P(y = 1|x) = \frac{1}{1+exp(\sum_i w_i x_i + c)}$.

In the same feature space, if $w_i = log\frac{\theta_{-i}}{\theta_{+i}} - log\frac{1-\theta_{-i}}{1-\theta_{+i}}$ and $c = log\frac{\mu}{1-\mu} + \sum_i log\frac{1-\theta_{-i}}{1-\theta_{+i}})$, naive bayes has the same form as logistic regression.

3 Sentiment Analysis

3.1 Problem Setup

We are given two groups of movie reviews with label positive or negative, and we would like to predict the sentiment given a review. The problem is a classification problem where we predict whether the review is positive or negative. The data is relatively uniform, with roughly 50% of both classes. We decided to divide the corpus into training set and test set. On the training set, we use cross validation in the sklearn's GridSearch function with default 3 fold split, which leaves one fold as the validation set. We then retrain on the original train set with parameters from cross validation, and results are in Table 2.

Range of parameters There are two kinds of parameters, one is associated with preprocssing and unigrams, the other one is regularization parameters in the model of choice to make an actual prediction. The first is intended to regularize unigram counts. The latter is intended to regularize the classification models. It is possible to take a combination of methods in Table 1 and find the best combination of parameters in unigram. However, due to lack of computation resource, only some combinations are experimented.

3.2 Experiment procedure

The experiment has mainly three steps. First, it preproceses the text, such as applying a stemmer or lemmatizer from NLTK. Second, it applies unigram model from sklearn to the text from preprocessings outlined in Table 1. Third, we run cross validation using sklearn's grid search. The experiment tries different setup outlined in the previous section.

We then compare model performance and choose according to validation accuracy and the corresponding parameters. Then, we retrain on training set with the given hyperparam-

eters, and compare the test accuracies.

3.3 Results

Discussion: Success: We observe that under the same classification model, no preprocessing in unigram gives the best test accuracy. Under the same unigram preprocessing, Naive Bayes model has the best test accuracy of 78.13%. The overall best model is naive bayes with no preprocessing or additional parameters in unigrams. Grid search gives $\alpha = 1.0$ (Table 1) as the best parameter. All methods have test accuracies 13% to 28% above random dummy classifier.

Logstic regression and naive Bayes are very close. It is shown in Question 2 of this assignment that they are both predict a probability. SVM's lower performance may be due to the fact that it is a non-probablistic model. In this problem, it is hard to say that a word/sequence is always positive or negative.

Failure: we can speculate that unigram regularization hurts performance. Both lemmatizer and stemmer failed to reduce overfitting, as we can observe that the difference between train and test accurarcies only drop around 2% for models using the two difference methods. The reason of lemmatizer and stemmer not working well could be that they cannot apply the same set of rules to foreign text, typos, etc in the noisy data.

Overfitting: Despite the high test accuracy, we can see many cases of overfitting in all methods except removing infrequent words and the last two rows in table 2. When we remove infrequent words, the accuracy is lower than the best ones in other unigram methods, but the train and test accuracies are both around $65 \pm 2\%$. The low performance, while still well above the dummy method, is due to the models' limited computational power. This shows that removing infrequent words is effective in preventing overfitting. We can explore

this preprocessing with a more powerful model to achieve better results.

Error analysis/Confusion Matrix (eqn 2): The diagonal represents cases that are correctly classified. Precision is 76.24%, recall is 78%, and accuracy is 78.13%. The classifier model makes a balanced prediction on both positive and negative classes. The model does not predict biased results.

$$\begin{bmatrix} 1050 & 283 \\ 327 & 1006 \end{bmatrix} \tag{2}$$

4 Appendix

Table 1: parameters

Parameter	Range	best param
Lemmatizer	[with lemmatizer, without]	without
stemmer	[with stemmer, without]	without
unigram stopwords	[english, none]	none
unigram frequency	min_freq[0.001, 0.01, 0.1]	0
C in logistic	np.logspace(-4, 4, 20)	0.6158
C in SVM	np.logspace(-4, 4, 20)	545.5
α in Naive bayes	(0.1, 0.5, 1.0, 1.5, 2)	1.0

Table 2: Results

Model	**Unigram**	**Train/Test Accuracy(%)**	$train - test$ **(%)**
Dummy	raw/lemma/stem	50.09/50.41 average	
Logistic	Raw	97.66/77.34%	20
SVM	raw	50.53/48.42	2
naive bayes	**raw**	**93.30/78.13**	
Log	stop words	97.42/75.88	
SVC	stop words	50.42/48.72	
Naive Bayes	stopwords	93.57/76.51	
log	infreq	67.28/65.04	2
SVC	infreq	65.72/63.01	
NB	infreq	65.54/63.35	
log	stem	96.16/76.89	20
SVC	stem	50.16/49.54	
NB	stem	90.74/77.31	
log	lemma	97.26/74.90	23
SVC	lemma	54.62/54.05	0.6
NB	lemma	92.23/77.11	
NB	stop words, stem	90.68/76.33	
log	stem, infreq	68.63/66.91	
NB	stem, infreq	67.10/64.17	