Midterm Exam: Natural Language Processing

Quin'darius Lyles-Woods

October 5, 2021

1 Regular Expression: $[^{\wedge}A-Z|abc]+$

- NLP is an interseting topics
- Regular expressions is easy.
- i like nlp
- Negation operation is fun

$^{\wedge}A-Z$ Does not match any upper case letters.

| or operator.

abc with the negation characters to not match.

+ allows match with the last character to up to infinity occurrences.

The only sentence that matches fully is

i like nlp

All the others ones come up short but they do have matching characters.

2 Regular Expression: $ksu.*edu$

- KSU is a great college
- Ksu is an Edu
- Ksu&Edu
- ksu@edu

ksu matches any of these list of characters.

. can be any character.

* can be any number of the last character before it.

The only sentence that matches fully is:

ksu@edu

3 Classifiers

3.1 Naïve Bayes

The Naïve Bayes classifier is a very fast classifier and doesn't require a lot of storage requirements and is one of the simplest classifiers.

3.2 Logistic Regression

Logistic Regression is another classifier but of the discriminative class. It works by learning the classes by using the features given in the training set and then it assigns a weight to those features. While it is not as simple as the Naïve Bayes it makes up for it in use cases, it has seen use in important applications in medicine and finance.

3.3 Softmax Classifier

The softmax classifier is similar to logistic regression to allows the generalization to multiple dimensions or classes. And the softmax function gives us a normalized probability giving it a more intuitive understanding by squashing the output values of the vector between zero and one.

4 Activation Functions

4.1 σ

This activation function is S shaped and is defined by the function $y = \frac{1}{1+e^{-z}}$. It is non-linear, hence the S shape and as the output from zero to 1, it is differentiable allowing us to compute derivities upon it. This functions' learning rate is slow probably because of the saturation of values near the 1 creating and 0 creating the vanishing gradient problem.

4.2 tanh

Compared to the sigmoid function this function has the range of -1 to 1 instead of 0 to 1. Just like the sigmoid function this function produces an differentiable S curve about the function. Of course it is non-linear given the S shape.

4.3 *Relu*

The Relu or the Reticular Activation Function is a linear function where both the function and the derivative are both monotonic, meaning, varying in such a way the derivatives of the function never increases or decreases.

5 Tagging and Entity Recognition

5.1 Sequence Labeling

Sequence Labeling is assigning labels to all the inputs in the 'sequence' or more appropriate, the text corpus. The Following sections are simply subsets of sequence labeling (POS, Name Entity Recognition). Algorithms that use this are HHM, CRF, and RNN's.

5.2 Parts of Speech Tagging

Parts of speech tagging is where you take all the words and assign them parts of speech in your languages grammar construct. Examples of this are:

- Noun
- Verb
- Conjunction
- Participle
- etc...

5.3 Parts of Speech Baseline Model

If I were to build my parts of speech baseline model I would have all the basic grammatically structures. When I run into conflicts within this baseline I will choose the part of speech that is most frequent within my training set. Then use that as a baseline compare classifier algorithms against that baseline.

- Noun
- Verb
- Pronoun
- Adverb
- Conjunction
- Participle
- Article
- Conjunction
- Interjection

Also I would like to have a place for plural nouns and the different types of verbs to allow my language models know what word is important and what is not. I am aware that most Parts of Speech Tagging can get to around 1000 different data points and I think that is too large because after a certain point we can hard code all the values and we must allow algorithmic decision making take over and allow use to get the many multiple of edge cases that we wouldn't be able to get ourselves.

5.4 POS Tagging vs. Name Entity Recognition

Parts of Speech Tagging is where you try to tag all the parts of speech that you have noted in your Parts of Speech Model, the Name Entity Recognition is where you try to recognize names so it doesn't confuse your algorithm and groups them all under one umbrella. So the Name Entity Recognition is very narrowly focus and the Parts of Speech Tagging is the superset of the two.

6 Metrics

Class A
- 1 True Positive
- 1 False Positive
- 1 False Negative

Class B
- 10 True Positive
- 90 False Positive
- 90 False Negative

Class C
- 1 True Positive
- 1 False Positive
- 1 False Negative

Class D
- 1 True Positive
- 1 False Positive
- 1 False Negative

$$Precision_A = Precision_C = Precision_D = 0.5 \tag{1}$$

$$Precision_B = 0.1 \tag{2}$$

6.1 Micro Average

With the Micro Averaging approach you will want to compute the performance of each of the classes then take the average for all of them.

$$Micro\ Average = \frac{1 + 10 + 1 + 1}{2 + 100 + 2 + 2} = 0.123 \tag{3}$$

6.2 Macro Average

With Macro Averaging you do an additional step, you collect all the decisions in the class then equate the contingency table and get the precision after that with the appropriate formula.

$$Macro\ Average = \frac{0.5 + 0.1 + 0.5 + 0.5}{4} = 0.4 \tag{4}$$

6.3 Recall

The metric is the fraction of relevant marks that were retrieved. It counts the number of positive predictions made out of the positive examples. This is a performance metric.

$$Recall = \frac{1 + 10 + 1 + 1}{1 + 10 + 1 + 1 + 1 + 90 + 1 + 1} = 0.140 \tag{5}$$

6.4 F Measure

Gives you one score that takes into account the precision and the recall metric.

$$F\ Measure = \frac{2 * 0.4 * 0.140}{0.4 + 0.140} = 0.62 \tag{6}$$

7 Word Embeddings

7.1 What is it?

Word Embedding is a way of converting words in vector representations.

7.2 What is word2vec?

The way word2vec works is by iterating over every single word of the whole given body of text and using the vector representations to predict the next words.

7.3 Neural Language Model and Training Word Embeddings

A language model predicts upcoming words from the context of the previous word and this context is giving as a vector. So the language model can train the word embedding for the words given. Working together to build a more powerful prediction engine.

8 Neural Network:

$$a = 2x - y$$

$$b = az$$

$$L = a + 2b$$

8.1 Computational Graphs

8.1.1 $a = 2x - y$

8.1.2 $b = az$

8.1.3 $L = a + 2b$

8.2 Forward Pass Values

8.2.1 $a = 2x - y$

8.2.2 $b = az$

8.2.3 $L = a + 2b$

8.3 Back Propagation Circuit Diagram with Gradient Values

8.3.1 $a = 2x - y$

8.3.2 $b = az$

8.3.3 $L = a + 2b$

9 Language Models

Training Set

b a b a d a f f
a c h a d f f a h
f b a a h c f h d d f
a b f f c c d f h
h h a a c a c d d d

Test Set

h d c d f
d b b c c a

9.1 Unigram Language Model

a	b	c	d	f	h
$\frac{12}{52}$	$\frac{4}{52}$	$\frac{6}{52}$	$\frac{8}{52}$	$\frac{10}{52}$	$\frac{7}{52}$

9.2 Bigram Language Model

	a	b	c	d	f	h
a	$\frac{1}{12}$	$\frac{1}{12}$	$\frac{2}{12}$	$\frac{1}{12}$	$\frac{1}{12}$	$\frac{1}{12}$
b	$\frac{1}{4}$	$\frac{2}{4}$	$\frac{2}{4}$	$\frac{2}{4}$	$\frac{1}{4}$	$\frac{1}{4}$
c	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{3}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
d	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{3}{8}$	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{2}{8}$
f	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{2}{10}$	$\frac{1}{10}$	$\frac{1}{10}$
h	$\frac{1}{7}$	$\frac{1}{7}$	$\frac{1}{7}$	$\frac{2}{7}$	$\frac{1}{7}$	$\frac{1}{7}$

9.3 Joint Probability: Char Unigram

Test Sentence One:

$$Joint\ Probability = \frac{12}{52} * \frac{4}{52} * \frac{6}{52} * \frac{8}{52} * \frac{7}{52} = \frac{16,128}{380,204,032} \tag{7}$$

Test Sentence Two:

$$Joint\ Probability = \frac{8}{52} * \frac{4}{52} * \frac{4}{52} * \frac{6}{52} * \frac{6}{52} * \frac{12}{52} = \frac{55,296}{19,770,609,664} \tag{8}$$

9.4 Perplexity

9.4.1 Unigram

Test Sentence One:

$$Perplexity = (\frac{12}{52} * \frac{4}{52} * \frac{6}{52} * \frac{8}{52} * \frac{7}{52})^{-\frac{1}{5}} \tag{9}$$

Test Sentence Two:

$$Perplexity = (\frac{8}{52} * \frac{4}{52} * \frac{4}{52} * \frac{6}{52} * \frac{6}{52} * \frac{12}{52})^{-\frac{1}{6}} \tag{10}$$

9.4.2 Bigram

Test Sentence One:

$$Perplexity = (\frac{2}{8} * \frac{3}{8} * \frac{3}{8} * \frac{2}{10})^{-\frac{1}{5}} \tag{11}$$

Test Sentence Two:

$$Perplexity = (\frac{2}{8} * \frac{2}{4} * \frac{2}{6} * \frac{2}{6} * \frac{2}{12})^{-\frac{1}{6}} \tag{12}$$

10 Matrices

Training Sentences

$$\underbrace{delta}_{A}\ \underbrace{gamma}_{B}\ \underbrace{sigma}_{C}\ \underbrace{summantion}_{A} \tag{13}$$

$$\underbrace{alpha}_{A}\ \underbrace{sigma}_{C}\ \underbrace{beta}_{D}\ \underbrace{derivative}_{A} \tag{14}$$

$$\underbrace{derivitive}_{A}\ \underbrace{gamma}_{B}\ \underbrace{delta}_{B}\ \underbrace{beta}_{D} \tag{15}$$

$$\underbrace{sigma}_{C}\ \underbrace{summation}_{B}\ \underbrace{beta}_{C}\ \underbrace{alpha}_{D} \tag{16}$$

$$\underbrace{alpha}_{A}\ \underbrace{beta}_{B}\ \underbrace{sigma}_{C}\ \underbrace{derivative}_{A} \tag{17}$$

10.1 Transition Probability Matrix

10.2 Emission Probability Matrix