COMP4901K and MATH4824B Midterm Exam

October 23, 2018
In Class: 4:30pm-5:50pm
Instructor: Yangqiu Song

Name: _____ **Student ID:** _____

Question	**Score**	**Question**	**Score**
1	/ 10	5	/ 15
2	/ 5	6	/ 10
3	/ 10	7	/ 25
4	/ 5	8	/ 20
Total:		**/ 100**	

Q1. Yes/No Questions

Indicate whether each statement is true (✓) or false (×).

1. A word more frequently appearing in a document implies that it is more informative for document classification.
2. In practice, we use dense vectors to store document VSM representations.
3. WordNet is a knowledge base about synonyms of words. You can retrieve information such as hypernyms and hyponyms of nouns.
4. Word similarities can be evaluated based on both knowledge bases and corpora.
5. Clustering requires labeled data. We know how many classes there are and we have labeled examples for each class.
6. High recall of a class means that an algorithm returned most of the relevant data about that class.
7. The evaluation of a classifier on dev set can be used to determine the model's hyper-parameters.
8. Naive Bayes' assumption is that features are dependent to each other.
9. In Naive Bayes classifier without smoothing, the more features you add to your classifier, the easier your classifier gets overfitting.
10. Laplace smoothing can be used to avoid overfitting.

FFTTF TTFTT (10 points)

Q2. Pre-processing: Read Code and Report the Result

```
stop_words = ["a", "and", "has", "in", "of", "the"] + list(string.punctuation)
def tokenize(text):
    tokens = []
    for word in nltk.word_tokenize(text):
        word = word.lower()
        if word not in stop_words and not word.isnumeric():
            tokens.append(word)
    return tokens
```

Input:
text = "Eighteen Olympic and World Championship gold medals and 21 world records later, Lewis has become the richest man in the history of track and field."

Output:
tokens = ["eighteen","olympic","world","campionship","gold","medals","world","recoreds","later",

"lewis","become","richest","man","history","track","field"] (2 points for every error)

Q3. TF-IDF: Read Code and Report the Result

Here's a program turning Bag-of-words into TF-IDF vectors, please read the code and report the result.

```
from collections import defaultdict

def tfidf(corpus, vocab):
    def termfreq(matrix, doc, term):
        try:
            return matrix[doc][term] / float(sum(matrix[doc].values()))
        except ZeroDivisionError:
            return 0

    def inversedocfreq(matrix, term):
        try:
            return float(len(matrix)) / sum(
                [1 for i,_ in enumerate(matrix) if matrix[i][term] > 0])
        except ZeroDivisionError:
            return 0

    matrix = [{k:v for k,v in zip(vocab, i[1])} for i in corpus]
    tfidf = defaultdict(dict)
    for doc,_ in enumerate(matrix):
        for term in matrix[doc]:
            tf = termfreq(matrix,doc,term)
            idf = inversedocfreq(matrix, term)
            tfidf[doc][term] = tf*idf

    return [[tfidf[doc][term] for term in vocab] for doc,_ in enumerate(tfidf)]


if __name__ == "__main__":
    corpus = [('this is a foo bar', [1, 1, 0, 1, 1, 0, 0, 1]),
              ('foo bar bar black sheep', [0, 2, 1, 1, 0, 0, 1, 0]),
              ('this is a sentence', [1, 0, 0, 0, 1, 1, 0, 1])]
    vocab = ['a', 'bar', 'black', 'foo', 'is', 'sentence', 'sheep', 'this']
    print(tfidf(corpus, vocab))
```

Output:

[[0.3, 0.3, 0.0, 0.3, 0.3, 0.0, 0.0, 0.3],
  [0.0, 0.6, 0.6, 0.3, 0.0, 0.0, 0.6, 0.0],
  [0.375, 0.0, 0.0, 0.0, 0.375, 0.75, 0.0, 0.375]]

Each non-zero element worth 1 point. Greater than or equal to 10 incorrect values gets no point.

Q4. WordNet

A subgraph of WordNet is shown in Figure 1.

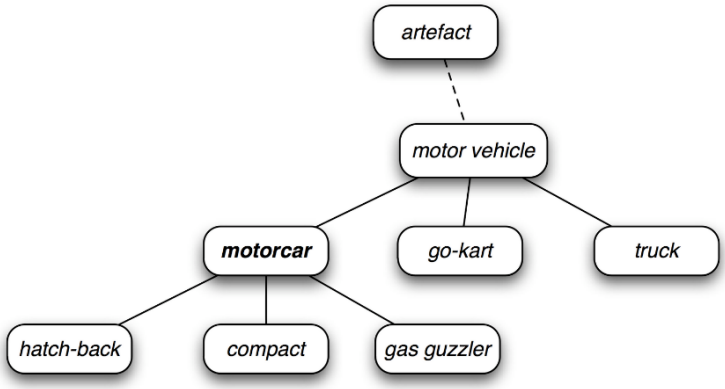

Figure 1: Each rectangle denotes a synset.

Please compute following two similarities between two synsets: `compact` and `truck`.

1. Shortest path based similarity: $1/\text{Len}(w_1, w_2)$, where $\text{Len}(w_1, w_2)$ returns the length of the shortest path between w_1 and w_2. For example, $\text{Len}(\texttt{motorcar}, \texttt{compact}) = 1$. (5 points)

2. Least common subsumer based similarity: $2*\text{Depth}\big(\text{LCS}(w_1, w_2)\big)/\big(\text{Depth}(w_1)+\text{Depth}(w_2)\big)$, where $\text{LCS}(w_1, w_2)$ returns the least common subsumer of w_1 and w_2, and $\text{Depth}(w)$ returns the path length from the root node to the synset w. Suppose the depth of synset `motor vehicle` is 10. (5 points)

Shortest path based similarity:
$1/\text{Len}(\texttt{compact}, \texttt{truck})$
$= 1/3$

Least common subsumer based similarity:
$2 * \text{Depth}\big(\text{LCS}(\texttt{compact}, \texttt{truck})\big)/\big(\text{Depth}(\texttt{compact}) + \text{Depth}(\texttt{truck})\big)$
$2 * \text{Depth}(\texttt{motor vehicle})/\big(\text{Depth}(\texttt{compact}) + \text{Depth}(\texttt{truck})\big)$
$= 2 * 10/(11 + 12)$
$= 20/23$

Q5. Document Similarity

Consider the following document collection D = D1, D2, D3 (given as one document per line):

```
D1: clear sky blue sky
D2: the blue car
D3: sky is nice
```

Assume that the stopword list contains words "the" and "is", and words are not stemmed. Please answer following questions relevant to *Document Vector Space Model*.
a). Write down the vector representations of D1, D2, and D3 using term frequency $f_{t,d}$ as term weighting (the dimensions should be listed alphabetically and stopwords should be taken into consideration). (9 points)
b). Given a new document D4:

```
D4: the sky is blue
```

compute the cosine similarity based on TF weights between D4 and D1, D4 and D2, D4 and D3, respectively. (6 points)

a). After removing stops words, documents are as follows:

D1: clear sky blue sky
D2: blue car
D3: sky nice

The vocabulary is ["blue", "car", "clear", "nice", "sky"]
So the BOW representation of each document is as follows:

D1: [1, 0, 1, 0, 2]
D2: [1, 1, 0, 0, 0]
D3: [0, 0, 0, 1, 1]

So the answer is [1, 0, 1, 0, 2]

(If the vocabulary is not in alphabetical order, then there are 3 points discounted (1 for each vector).)

b). The document 4's vector is [1, 0, 0, 0, 1]

The cosine similarities are as follows:
between D1 and D4: 0.8660,
between D2 and D4: 0.5
between D3 and D4: 0.5

Q6. Classification

Given the training data shown as Figure 2, answer the following questions.

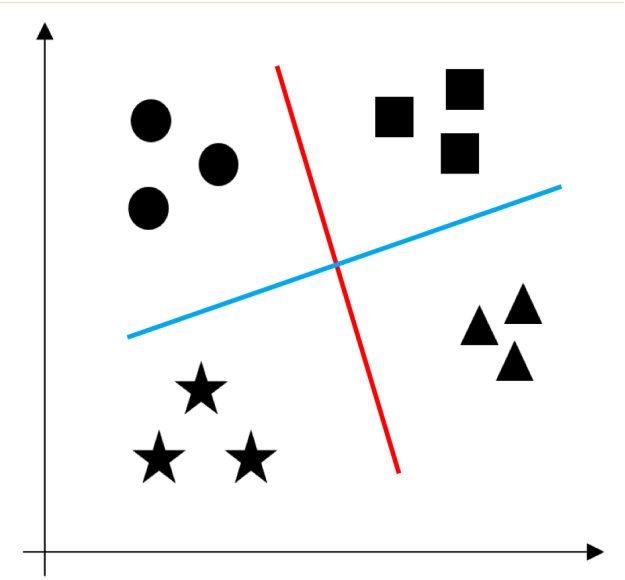

Figure 2: Each point with either symbol of circle, square, star, or triangle represents a sample from one of four classes. The points with the same symbol are from the same class.

(i) Are the data shown in Figure 2 linearly separable? (4 points)

Yes.

(ii) If we use binary classifiers to classify the data points, how many classifiers at least do we need to use? Draw the classification boundaries of corresponding feasible classifiers. (6 points)

At least two classifier. (3 points)
One of possible ways to classify the data is shown in the Figure 2. (3 points)

Q7. Naive Bayes

Note: To make this question easier, you don't need to handle stop words. But you still need to apply the Laplace smoothing (add-one smoothing) to probabilities of each class and each word in order to avoid zero-probability.

We work on a sentiment analysis problem with two classes positive (+) and negative (−), and take the following miniature training and test documents.

Table 1: training and test documents

	Class	Documents
Training	−	just plain boring
	−	entirely predictable and lacks energy
	−	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

Please compute:

(i) the probability for each class, $P(y = +)$ and $P(y = -)$

$$P(y = +) = \frac{N_+ + 1}{N_{doc} + 2} = \frac{3}{7}$$
$$P(y = -) = \frac{N_- + 1}{N_{doc} + 2} = \frac{4}{7}$$

(5 points. 2.5 points for computation and the Laplace smoothing)

(ii) the probability for every word in the test document, $P(w_i|y = +)$ and $P(w_i|y = -)$

$$P(predictable|y = +) = \frac{0+1}{9+20} = \frac{1}{29}$$
$$P(predictable|y = -) = \frac{1+1}{14+20} = \frac{2}{34}$$
$$P(no|y = +) = \frac{0+1}{9+20} = \frac{1}{29}$$
$$P(no|y = -) = \frac{1+1}{14+20} = \frac{2}{34}$$
$$P(fun|y = +) = \frac{1+1}{9+20} = \frac{2}{29}$$
$$P(fun|y = -) = \frac{0+1}{14+20} = \frac{1}{34}$$

(8 points. 4 points for computation and the Laplace smoothing)

(iii) the **necessary** steps, and your final decision of the test document, positive or negative.

$$P(d|y = +) = \frac{2}{29^3}$$
$$P(d|y = -) = \frac{4}{34^3}$$
$$P(d|y = +)P(y = +) = \frac{6}{7 \times 29^3}$$
$$P(d|y = -)P(y = -) = \frac{16}{7 \times 34^3}$$

Because $P(d|y = -)P(y = -) > P(d|y = +)P(y = +)$, this test document is negative.
(12 points. 6 points for calculation formulas and intermediate results, 0 points for final decision)

Q8. Evaluation of Classification Results

Calculate precision, recall, and F1-score of the following model's predictions and document labels. Note: please compute the precision, recall, and F1-score of each class and take the average of each to report the result.

Document	Prediction	Label
0	Research	Sports
1	Research	Sports
2	Research	Sports
3	Research	Research
4	Research	Government
5	Sports	Sports
6	Sports	Sports
7	Sports	Research
8	Government	Research
9	Government	Government
10	Government	Sports
11	Research	Sports
12	Research	Sports
13	Research	Research
14	Research	Government
15	Sports	Research
16	Sports	Sports
17	Government	Research
18	Government	Government
19	Government	Government

2 points for each	Research	Sports	Government	Precision	F1
Research	2	5	2	2/9	4/15=0.267
Sports	2	3	0	3/5	3/7 = 0.429
Government	2	1	3	1/2	6/11=0.545
Recall	1/3	1/3	3/5		

$$\text{average precision} = 119/270 = 0.441$$
$$\text{average recall} = 19/45 = 0.422$$
$$\text{average f1} = 1433/3465 = 0.414$$

2 points for averages.