USCMarshall
School of Business

DSO 560 – Text Analytics & Natural Language Processing
Instructor: Yu Chen Midterm Exam

Due Tuesday, April 20th, 8:10pm PST (90 minutes)

Instructions:

- **WRITE ALL ANSWERS ON SEPARATE SHEETS OF PAPER**
- **SCAN EACH PAGE (AS A PDF OR IMAGE) AND SEND TO ME AND THE TA VIA SLACK.**

- **DO ALL SECTIONS.**

SHOW ALL WORK TO RECEIVE CREDIT

Short Answer (5 pts, recommended 20 minutes)

Pick 5 of the 7 to answer. Provide at most two sentences explaining your answer.

1. What are **two reasons** why you may decide to use a word2vec embedding model instead of count vectorization to convert your text documents into numeric vectors?
2. Which of the following code snippets is better suited to handle Big Data-sized text datasets? Explain why.

Option A	**Option B**	**Option C**

import pandas as pd
 	
df = pd.read_csv(“datasource”)
	with open(“datasource”) as file:
	    lines =file.readlines()
		with open(“datasource”) as file:
		  while True:
		    line = file.readline()
		    if not line:
		      break

3. You are working to build an NLP model. Your manager reviews your code and asks you to **reduce the n-gram size** in your model. What is one reason why she may ask you to reduce the n value of your n-grams?
4. You notice a junior data scientist’s regex pattern for finding all references to **male children** is **re.findall(r’son’).** Identify one issue w/ this regex pattern and propose an improved version that would reduce the number of false positives/negatives.
5. According to Zipf’s Law (assume an alpha α = 1), the top 5 words in your corpus would comprise what percentage of your total word count?
6. If you want to make sure your text search query model, when it is provided an input “baby care” returns all relevant products baby care products, would you optimize for recall or precision?
7. Would a bag of words model work better for representing as documents, **tweets** or **New York Times articles**? Why?

Naïve Bayes (3 pts, recommended 15 minutes)

You are a data scientist at Amazon implementing a **Naïve Bayes NLP model on product tags to classify Above Benchmark and Below Benchmark products by conversion rate.** This model will be used to quickly alert customer service to follow up with customers.

You've collected a small dataset below:

Above Benchmark Product:

1. Bargain, Artisan, Furniture, Wood, Green
2. Discount, Toys, Homemade, Green
3. Personalized, Wool, Clothing, Premium

Below Benchmark Product:

1. On-Sale, Furniture
2. Plastic, Recycleable, On-Sale
3. Discount, Bargain, Clothing
4. Wool, Blanket, Old, Premium, Clothing
5. Wooden, Chair, Green
6. Old, On-Sale, Toys, Blue
7. Personalized Clothing

Note – feel free to group together any terms you think are identical in semantic meaning.

A. **Perform any preprocessing and grouping you deem appropriate, and calculate the following probabilities, assuming a Naïve Bayes classifier (2 pts):**

   i. The **prior** for a below benchmark product **(0.5pts)**

   ii. The **likelihood** for the product tags "**Green, Wool, Clothing**" given you know it is an Above Benchmark product **(0.5pts)**

   iii. The **evidence in Bayes Rule for "Green, Wool, Clothing" (1pt)**

   iv. The **posterior** for an Above Benchmark product with the tags **"Green, Wool, Clothing" (1pts)**

Vectorization and Similarity (3 pts, recommended 15 minutes)

You work as a data analyst for Netflix. Your team is tasked with identifying customer segments This is what 3 customers wrote:

User A: comedy G-rated Michael B. Jordan Sandra Bullock NYC

User B: G-rating animation adventure action LA

User C. romantic comedies Sandra Bullock New York City animated

User D. Michael Douglas horror adventure thriller action Los Angeles

These are your definitions for **Term Frequency** and **Inverse Document Frequency**:

TF = n(t,d)

IDF = 1 / df(t)

n(t,d) is the number of times term *t* appears in document *d*

df(t) is the document frequency of term *t*

Please group any tokens you feel should be collocated together.

a. Generate **TF-IDF document vectors** (you may write them as a matrix or table). Calculate IDF for each of the words, then term frequency (TF) for each of document – word combinations (**2pt**)

b. A new show on Netflix called *Toothy Tim's Tank Engine* is set to debut, with the search tags: **G-rated action comedy**. Assuming **TF-IDF vectorized** documents and **Cosine Similarity**, should we recommend this show for **User A** or **User B**? **(1pt)**

N-Gram Language Models (3 pts, recommended 15 minutes)

Given the following documents:

1. I bought a banana cake to sell.
2. I eat cake.
3. He loves to eat the bananas.
4. I love her.

A. Perform lemmatization and stopword removal. For this exercise, consider the words **“the”** and “a”, as stopwords. Then construct the transition matrix for this corpus. (**2 pts**)

B. Using a **bigram language model (the only language model for transitions we have studied so far),** what is the perplexity score for the sentence ***I love to eat cake*** (**1pt**)

True/False (3 pts, recommended 15 minutes)

Pick 3 of the statements below, indicate if it is true or false. **If it is false, explain why it is false and provide an example. You may provide an explanation if it is true in case you are wrong and would like to receive partial credit.**

A. Using count vectorization to convert words to numeric vectors, the word **dog** is as similar to the word **Serbia** as the word **puppy**.

B. Assuming that both documents are in the same corpus of 10,000 documents, using TF-IDF vectorization, both the document **Happy fun time** and **fun happy time** would have the same vectorization.

C. Euclidean distance is better suited to compare two documents of different lengths.

D. UTF8 characters can be anywhere between 1 and 4 bytes in size. The English alphabet and common characters will usually take 1 byte to represent but characters from East Asian languages will often take 3-4 bytes to represent.

E. In terms of number of bits required, compared to **UTF8, Latin-1** would more efficiently represent the character "💖"