cs-119 Quiz7 q4 (LDA)

The first 8 cells of this notebook install PySpark. The quiz questions follow.

```
!apt-get  install  openjdk-8-jdk-headless  -qq  >  /dev/null
```

```
!wget  -qN  https://archive.apache.org/dist/spark/spark-3.2.1/spark-3.2.1-bin-hadoop3.2.tgz
!tar  xf  spark-3.2.1-bin-hadoop3.2.tgz
```

```
!pip  install  -q  findspark
```

```
import  os
os.environ["JAVA_HOME"]  =  "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"]  =  "spark-3.2.1-bin-hadoop3.2"
```

```
!pip  install  pyspark
```

```
Collecting pyspark
  Downloading pyspark-3.2.1.tar.gz (281.4 MB)
     |████████████████████████████████| 281.4 MB 25 kB/s
Collecting py4j==0.10.9.3
  Downloading py4j-0.10.9.3-py2.py3-none-any.whl (198 kB)
     |████████████████████████████████| 198 kB 51.8 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.2.1-py2.py3-none-any.whl size=281853642 sha256=c
  Stored in directory: /root/.cache/pip/wheels/9f/f5/07/7cd8017084dce4e93e84e92efd1e1d5334db05f2
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.3 pyspark-3.2.1
```

```
import  pyspark
from  pyspark.sql  import  SparkSession
spark  =  SparkSession.builder.master("local[1]")  \
                              .appName('Reviews  LDA')  \
                              .getOrCreate()
```

```
sc  =  spark.sparkContext
```

```
from  pyspark.sql.types  import  *
```

```
import  nltk
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
import pandas as pd
import pyspark
from pyspark.sql import SQLContext
```

Q1: Read the reviews data from the source and scrub it

The data source is here.

Download it to a Pandas DataFrameClean the text in the Title and Review Text columns according to these rules:

1. Remove all punctuations,
2. Turn all words into lowercase,
3. Reject all 3-character or smaller words.

```
df = pd.read_csv('reviews.csv')
df.dropna(inplace=True)
for (columnName, columnData) in df.iteritems():
    if pd.api.types.is_string_dtype(df[columnName].dtype):
        df[columnName] = df[columnName].str.lower()
        df[columnName]=df[columnName].str.findall('\w{3,}').str.join(' ')
        df[columnName] = df[columnName].str.replace(r'[^\w\s]+', '')
df
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: FutureWarning: The defa
  import sys
```

	recNo	Clothing ID	Age	Title	Review Text	Rating	Recommended IND	Positive Feedback Count	Divisio Nam
2	2	1077	60	some major design flaws	had such high hopes for this dress and really ...	3	0	0	gener
3	3	1049	50	favorite buy	love love love this jumpsuit fun flirty and fa...	5	1	0	gener petit
					this shirt very				

Each review is organized as a row in the PySpark DataFrame, and the objective is to do the same processing on each review in parallel!

Q2: Convert the Pandas DataFrame into a PySpark DataFrame

not for reese

```
sparkDF = spark.createDataFrame(df)
sparkDF.printSchema()
sparkDF.show()
```

```
root
 |-- recNo: long (nullable = true)
 |-- Clothing ID: long (nullable = true)
 |-- Age: long (nullable = true)
 |-- Title: string (nullable = true)
 |-- Review Text: string (nullable = true)
 |-- Rating: long (nullable = true)
 |-- Recommended IND: long (nullable = true)
 |-- Positive Feedback Count: long (nullable = true)
 |-- Division Name: string (nullable = true)
 |-- Department Name: string (nullable = true)
 |-- Class Name: string (nullable = true)

+-----+-----------+---+--------------------+--------------------+------+---------------+--------
|recNo|Clothing ID|Age|               Title|         Review Text|Rating|Recommended IND|Positive
+-----+-----------+---+--------------------+--------------------+------+---------------+--------
|    2|       1077| 60|some major design...|had such high hop...|     3|              0|
|    3|       1049| 50|        favorite buy|love love love th...|     5|              1|
|    4|        847| 47|    flattering shirt|this shirt very f...|     5|              1|
|    5|       1080| 49|not for the very ...|love tracy reese ...|     2|              0|
|    6|        858| 39|cagrcoal shimmer fun|aded this basket ...|     5|              1|
|    7|        858| 39|shimmer surprisin...|ordered this carb...|     4|              1|
|    8|       1077| 24|          flattering|love this dress u...|     5|              1|
```

```
|    9|       1077| 34|      such fun dress|and 125 lbs order...|     5|             1|
|   10|       1077| 53|dress looks like ...|dress runs small ...|     3|             0|
|   12|       1095| 53|             perfect|more and more fin...|     5|             1|
|   13|        767| 44|           runs big|bought the black ...|     5|             1|
|   14|       1077| 50|pretty party dres...|this nice choice ...|     3|             1|
|   15|       1065| 47|nice but not for ...|took these out th...|     4|             1|
|   16|       1065| 34|you need least av...|material and colo...|     3|             1|
|   17|        853| 41|looks great with ...|took chance this ...|     5|             1|
|   18|       1120| 32| super cute and cozy|flattering super ...|     5|             1|
|   19|       1077| 47|stylish and comfo...|love the look and...|     5|             1|
|   20|        847| 33|    cute crisp shirt|this product was ...|     4|             1|
|   21|       1080| 55|               torn|upset because for...|     4|             1|
|   22|       1077| 31| not what looks like|first all this no...|     2|             0|
+-----+-----------+---+--------------------+--------------------+------+--------------+--------
only showing top 20 rows
```

Q3: Tokenize Review Text

NLTK provides its own stop words. Using these has the advantage that we can use NLTK-provided stop words for a variety of supported languages.

This is an excellent place to further process the text. A tokenizer for the Review Text is provided for you here, you may use it as-is or modify it as you see fit.

```
def tokenize(pyspark_DataFrame):
    import re
    from nltk.corpus import stopwords
    reviews = pyspark_DataFrame.rdd.map(lambda x : x['Review Text'])     \
        .filter(lambda x: x is not None)
    StopWords = stopwords.words("english")
    tokens = reviews
        .map( lambda doc: doc.strip().lower())
        .map( lambda doc: re.split(" ", doc))
        .map( lambda word: [x for x in word if x.isalpha()])                    \
        .map( lambda word: [x for x in word if x not in StopWords])       \
        .zipWithIndex()
    return tokens
```

```
def tokenize(pyspark_DataFrame):
    import re
    from nltk.corpus import stopwords
    reviews = pyspark_DataFrame.rdd.map(lambda x : x['Review Text'])     \
        .filter(lambda x: x is not None)
    StopWords = stopwords.words("english")
```

```
    tokens = reviews
           .map( lambda doc: doc.strip().lower())
           .map( lambda doc: re.split(" ", doc))
           .map( lambda word: [x for x in word if x.isalpha()])                    \
           .map( lambda word: [x for x in word if x not in StopWords])        \
           .zipWithIndex()
    return tokens

token = tokenize(sparkDF)
sparkDF.show()
```

```
+-----+-----------+---+--------------------+--------------------+------+--------------+--------
|recNo|Clothing ID|Age|               Title|         Review Text|Rating|Recommended IND|Positive
+-----+-----------+---+--------------------+--------------------+------+--------------+--------
|    2|       1077| 60|some major design...|had such high hop...|     3|             0|
|    3|       1049| 50|        favorite buy|love love love th...|     5|             1|
|    4|        847| 47|    flattering shirt|this shirt very f...|     5|             1|
|    5|       1080| 49|not for the very ...|love tracy reese ...|     2|             0|
|    6|        858| 39|cagrcoal shimmer fun|aded this basket ...|     5|             1|
|    7|        858| 39|shimmer surprisin...|ordered this carb...|     4|             1|
|    8|       1077| 24|          flattering|love this dress u...|     5|             1|
|    9|       1077| 34|      such fun dress|and 125 lbs order...|     5|             1|
|   10|       1077| 53|dress looks like ...|dress runs small ...|     3|             0|
|   12|       1095| 53|             perfect|more and more fin...|     5|             1|
|   13|        767| 44|            runs big|bought the black ...|     5|             1|
|   14|       1077| 50|pretty party dres...|this nice choice ...|     3|             1|
|   15|       1065| 47|nice but not for ...|took these out th...|     4|             1|
|   16|       1065| 34|you need least av...|material and colo...|     3|             1|
|   17|        853| 41|looks great with ...|took chance this ...|     5|             1|
|   18|       1120| 32| super cute and cozy|flattering super ...|     5|             1|
|   19|       1077| 47|stylish and comfo...|love the look and...|     5|             1|
|   20|        847| 33|    cute crisp shirt|this product was ...|     4|             1|
|   21|       1080| 55|                torn|upset because for...|     4|             1|
|   22|       1077| 31| not what looks like|first all this no...|     2|             0|
+-----+-----------+---+--------------------+--------------------+------+--------------+--------
only showing top 20 rows
```

Q4: TF.IDF Calculation

Feed the resulting tokens into a TF.IDF calculation. TF calculation is provided by `CountVectorizer`, and IDF calculation by `IDF`, both are available in the `pyspark.ml.feature` library.

```
def tfidf(sc, tokens):
    from pyspark.sql import SQLContext
    from pyspark.ml.feature import CountVectorizer , IDF
    sqlContext = SQLContext(sc)
    df_txts = sqlContext.createDataFrame(tokens, ["list_of_words",'index'])
    #
```

```
        #  TF
        #
        cv  =  CountVectorizer(inputCol="list_of_words",  outputCol="raw_features",  \
               vocabSize=5000,  minDF=10.0)
        cvmodel  =  cv.fit(df_txts)
        result_cv  =  cvmodel.transform(df_txts)
        #
        #  IDF
        #
        idf  =  IDF(inputCol="raw_features",  outputCol="features")
        idfModel  =  idf.fit(result_cv)
        tfidf_result  =  idfModel.transform(result_cv)
        #
        return  tfidf_result



def  tfidf(sc,  tokens):
        from  pyspark.sql  import  SQLContext
        from  pyspark.ml.feature  import  CountVectorizer  ,  IDF
        sqlContext  =  SQLContext(sc)
        df_txts  =  sqlContext.createDataFrame(tokens,  ["list_of_words",'index'])
        #
        #  TF
        #
        cv  =  CountVectorizer(inputCol="list_of_words",  outputCol="raw_features",  \
               vocabSize=5000,  minDF=10.0)
        cvmodel  =  cv.fit(df_txts)
        result_cv  =  cvmodel.transform(df_txts)
        #
        #  IDF
        #
        idf  =  IDF(inputCol="raw_features",  outputCol="features")
        idfModel  =  idf.fit(result_cv)
        tfidf_result  =  idfModel.transform(result_cv)
        #
        return  tfidf_result,  cvmodel

tf_res,  cvmodel  =  tfidf(sc,  tokenize(sparkDF))
```

```
/usr/local/lib/python3.7/dist-packages/pyspark/sql/context.py:79: FutureWarning: Deprecated in 3
  FutureWarning
```

Q5: LDA Training

The TF.IDF calculations form the input into LDA. An `lda_train()` function (shown below) takes columns `index` and `features` from the `tfidf` DataFrame and calculates the LDA Model. This calculation is referred to as *Training* the model.

```
def lda_train(result_tfidf):
        from pyspark.ml.linalg import Vectors, SparseVector
        from pyspark.ml.clustering import LDA
        #
        lda = LDA(k=10, seed=1, optimizer="em")
        lda.setMaxIter(100)
        #
        model = lda.fit(result_tfidf[['index', 'features']])
        return model
```

With the reviews dataset, LDA training takes about 15-20 minutes in a Colab environment.

```
def lda_train(result_tfidf):
        from pyspark.ml.linalg import Vectors, SparseVector
        from pyspark.ml.clustering import LDA
        #
        lda = LDA(k=10, seed=1, optimizer="em")
        lda.setMaxIter(100)
        #
        model = lda.fit(result_tfidf[['index', 'features']])
        return model
```

▾ This will take about 15 minutes

```
model = lda_train(tf_res)
```

```
newDF = model.describeTopics(10)
newDF.show()
```

```
/usr/local/lib/python3.7/dist-packages/pyspark/sql/context.py:127: FutureWarning: Deprecated in
  FutureWarning
+-----+--------------------+--------------------+
|topic|         termIndices|         termWeights|
+-----+--------------------+--------------------+
|    0|[10, 1, 29, 14, 4...|[0.03909962665316...|
|    1|[7, 64, 16, 76, 4...|[0.02875142105420...|
|    2|[33, 5, 71, 31, 2...|[0.02314743749983...|
|    3|[96, 17, 107, 57,...|[0.01768791402246...|
```

```
|    4|[77, 131, 145, 92...|[0.01589315688429...|
|    5|[38, 59, 50, 51, ...|[0.02935966152497...|
|    6|[53, 102, 34, 140...|[0.02674571223454...|
|    7|[0, 95, 41, 98, 1...|[0.05758129915368...|
|    8|[19, 32, 69, 79, ...|[0.02366700487689...|
|    9|[36, 11, 70, 97, ...|[0.03011856656754...|
+-----+--------------------+--------------------+
```

Q6: Reporting on the LDA Model

Examine the model generated during training. It will show the 10 topics it generated.

What is a topic? Just a list of words that "hang together." Does the collection of words describe a topic to you? What might it be?

```
def  getDescrib(input,  voca):
    res  =  []
    for  i  in  input:
            res.append(voca[i])
    return  res

voca  =  cvmodel.vocabulary

pandaDF  =  newDF.toPandas()
descrip  =  []
for  i  in  range(pandaDF.shape[0]):
    input  =  pandaDF.at[i,  'termIndices']
    temp  =  getDescrib(input,  voca)
    descrip.append(temp)

pandaDF['description']  =  descrip
sparkDF1  =  spark.createDataFrame(pandaDF)
sparkDF1.printSchema()
sparkDF1.show()
res  =  sparkDF1.toPandas()
res
```

```
root
 |-- topic: long (nullable = true)
 |-- termIndices: array (nullable = true)
 |    |-- element: long (containsNull = true)
 |-- termWeights: array (nullable = true)
 |    |-- element: double (containsNull = true)
 |-- description: array (nullable = true)
 |    |-- element: string (containsNull = true)

+-----+--------------------+--------------------+--------------------+
|topic|         termIndices|         termWeights|         description|
+-----+--------------------+--------------------+--------------------+
|    0|[10, 1, 29, 14, 4...|[0.03909962665316...|[small, size, lar...|
|    1|[7, 64, 16, 76, 4...|[0.02875142105420...|[great, black, pe...|
|    2|[33, 5, 71, 31, 2...|[0.02314743749983...|[much, like, onli...|
|    3|[96, 17, 107, 57,...|[0.01768791402246...|[first, one, time...|
|    4|[77, 131, 145, 92...|[0.01589315688429...|[design, blouse, ...|
|    5|[38, 59, 50, 51, ...|[0.02935966152497...|[petite, pants, t...|
|    6|[53, 102, 34, 140...|[0.02674571223454...|[skirt, high, len...|
|    7|[0, 95, 41, 98, 1...|[0.05758129915368...|[dress, body, wai...|
|    8|[19, 32, 69, 79, ...|[0.02366700487689...|[well, material, ...|
|    9|[36, 11, 70, 97, ...|[0.03011856656754...|[sweater, color, ...|
+-----+--------------------+--------------------+--------------------+
```

1 to 10 of 10 entries Filter

termWeights
32853,0.026314045794474043,0.026154858932446977,0.02240158084410348,0.019837806102351044,0.01941
085606,0.020509776893423835,0.020479219965671175,0.019449530911569892,0.017983882680044725,0.016
4322,0.018605007314040297,0.018502901997674955,0.018425677218458326,0.018172557591483868,0.01763

I think most of them tells me something(I have to change the df to pandas so that I can see full content of description:

1. feels like someone its complaining that the small size thing runs loose to large size
2. someone is happy with the black jeans he/she bought
3. someone bought a shirt that looks like the picture online, positive feedback
4. someone like cloth he bought and it is his first time
5. can't tell if this is a positive or nagtive review but it is about a blouse
6. looks someone bought a small size pants
7. someone is happy with the skir it bought
8. can hardly tell what it says too many words
9. someone really like the thing he bought maybe about the style and matrerial
10. someone bought a sweater and jacket with light green color

✓ 0 秒 完成时间：21:50