CIND719-DK0T
Assignment 3:

Ramello Peralta 500519802

1. **Load the data file as a Spark DataFrame**

```
>>> df_yelp = spark.read.format('csv').option('delimiter','\t').option('header',
'false').schema('sentence string, sentiment int').load('/home/cu/assignment3/yel
p_labelled.txt')
>>> df_yelp.show(5)
+--------------------+---------+
|            sentence|sentiment|
+--------------------+---------+
|Wow... Loved this...|        1|
|  Crust is not good.|        0|
|Not tasty and the...|        0|
|Stopped by during...|        1|
|The selection on ...|        1|
+--------------------+---------+
only showing top 5 rows

>>>
```

```
>>> type(df_yelp)
<class 'pyspark.sql.dataframe.DataFrame'>
```

2. **Tokenize the reviews text into words. (3 pts)**

```
>>> from pyspark.ml.feature import Tokenizer
```

```
>>> tokenizer = Tokenizer(inputCol='sentence', outputCol='words')
>>> tokenizer.transform(df_yelp).show(5)
+--------------------+---------+--------------------+
|            sentence|sentiment|               words|
+--------------------+---------+--------------------+
|Wow... Loved this...|        1|[wow..., loved, t...|
|  Crust is not good.|        0|[crust, is, not, ...|
|Not tasty and the...|        0|[not, tasty, and,...|
|Stopped by during...|        1|[stopped, by, dur...|
|The selection on ...|        1|[the, selection, ...|
+--------------------+---------+--------------------+
only showing top 5 rows

>>>
```

3. **Transform the Reviews text data into numeric features using the HashingTF class. Report how many features are created. (3 pts)**

```
>>> tokenizer = Tokenizer(inputCol='sentence', outputCol='words')
>>> df_yelp_tokenized = tokenizer.transform(df_yelp)
>>> from pyspark.ml.feature import HashingTF
>>> hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(), outputCol='features'
)
>>> df_yelp_htf = hashingTF.transform(dy_yelp_tokenized)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'dy_yelp_tokenized' is not defined
>>> df_yelp_htf = hashingTF.transform(df_yelp_tokenized)
>>> df_yelp_htf.show(5)
+--------------------+---------+--------------------+--------------------+
|            sentence|sentiment|               words|            features|
+--------------------+---------+--------------------+--------------------+
|Wow... Loved this...|        1|[wow..., loved, t...|(262144,[108541,1...|
|  Crust is not good.|        0|[crust, is, not, ...|(262144,[49815,10...|
|Not tasty and the...|        0|[not, tasty, and,...|(262144,[95889,97...|
|Stopped by during...|        1|[stopped, by, dur...|(262144,[9056,531...|
|The selection on ...|        1|[the, selection, ...|(262144,[15370,67...|
+--------------------+---------+--------------------+--------------------+
only showing top 5 rows

>>>
```

- Two features are created from the original dataframe, words and features.

```
>>> df_yelp_htf.show(1, False)
+------------------------+---------+-----------------------------+---------------
----------------------------------------+
|sentence                |sentiment|words                        |features
                                        |
+------------------------+---------+-----------------------------+---------------
----------------------------------------+
|Wow... Loved this place.|1        |[wow..., loved, this, place.]|(262144,[108541
,177414,216221,239331],[1.0,1.0,1.0,1.0])|
+------------------------+---------+-----------------------------+---------------
----------------------------------------+
only showing top 1 row

>>>
```

- HashingTF creates a hash code for each unique word in ‘sentence’. The corresponding frequencies in the following list.

4. **Split the data into train (70%) and test (30%). For reproducibility, fix the seed as 11. Train a logistic regression model to classify the reviews in positive or negative category. (3 pts)**

```
>>> train, test = df_yelp_htf.randomSplit(weights = [0.70, 0.30], seed = 11)
```

```
>>> from pyspark.ml.classification import LogisticRegression
```

```
>>> lrmodel = LogisticRegression(labelCol='sentiment').fit(train)
21/04/15 22:44:09 WARN BLAS: Failed to load implementation from: com.github.fommi
l.netlib.NativeSystemBLAS
21/04/15 22:44:09 WARN BLAS: Failed to load implementation from: com.github.fommi
l.netlib.NativeRefBLAS
21/04/15 22:44:09 WARN BlockManager: Asked to remove block broadcast_22_piece0, w
hich does not exist
>>>
```

```
>>> train_lr = lrmodel.transform(train)
>>> train_lr.show(5)
```

```
>>> train_lr.show(5, True)
+--------------------+---------+--------------------+--------------------+--------------------+--------------------+----------+
|            sentence|sentiment|               words|            features|       rawPrediction|         probability|prediction|
+--------------------+---------+--------------------+--------------------+--------------------+--------------------+----------+
|!....THE OWNERS R...|        0|[!....the, owners...|(262144,[27576,28...|[20.4898108565260...|[0.99999999873704...|       0.0|
|(It wasn't busy e...|        0|[(it, wasn't, bus...|(262144,[329,1217...|[21.9639182673008...|[0.99999999971080...|       0.0|
|(The bathroom is ...|        1|[(the, bathroom, ...|(262144,[41660,54...|[-20.388620930863...|[1.39744407126579...|       1.0|
|* Both the Hot & ...|        1|[*, both, the, ho...|(262144,[14686,47...|[-20.647963715165...|[1.07820985119999...|       1.0|
|- Really, really ...|        1|[-, really,, real...|(262144,[32705,38...|[-19.021887418026...|[5.48149795708028...|       1.0|
+--------------------+---------+--------------------+--------------------+--------------------+--------------------+----------+
only showing top 5 rows

>>>
```

5. **Apply test data to the trained LogisticRegression model trained in previous step. Compare the predicted values with the actual labels in terms of areaUnderROC using BinaryClassificationEvaluator class (3 pts)**

```
>>> preds = lrmodel.transform(test)
>>> preds.show(5)
+--------------------+---------+--------------------+--------------------+--------------------+--------------------+----------+
|            sentence|sentiment|               words|            features|       rawPrediction|         probability|prediction|
+--------------------+---------+--------------------+--------------------+--------------------+--------------------+----------+
|- They never brou...|        0|[-, they, never, ...|(262144,[38640,41...|[7.37977168829892...|[0.99937664553552...|       0.0|
|- the food is ric...|        1|[-, the, food, is...|(262144,[38640,43...|[-5.1685819500312...|[0.00566041287252...|       1.0|
|5 stars for the b...|        1|[5, stars, for, t...|(262144,[40082,94...|[-6.6737260158566...|[0.00126208659549...|       1.0|
|A FLY was in my a...|        0|[a, fly, was, in,...|(262144,[13925,39...|[-2.2137121049870...|[0.09852587775737...|       1.0|
|A couple of month...|        1|[a, couple, of, m...|(262144,[1546,100...|[-7.0219793610612...|[8.91262980956986...|       1.0|
+--------------------+---------+--------------------+--------------------+--------------------+--------------------+----------+
only showing top 5 rows

>>>
```

```
>>> from pyspark.ml.evaluation import BinaryClassificationEvaluator
```

```
>>> evaluator = BinaryClassificationEvaluator(labelCol='sentiment', metricName='a
reaUnderROC')
>>> evaluator.evaluate(preds)
0.8375822368421045
>>>
```

End of Assignment 3.