3_creating_features_quiz

November 24, 2019

1 Creating Features Quiz

Use this Jupyter notebook to find the answers to the quiz in the previous section. There is an answer key in the next part of the lesson.

```
In [22]: from pyspark.sql import SparkSession
         from pyspark.ml.feature import RegexTokenizer, CountVectorizer, IDF, StringIndexer, Vec
         from pyspark.sql.functions import udf
         from pyspark.sql.types import IntegerType, StringType

         import re

         # TODOS:
         # 1) import any other libraries you might need
         # 2) run the cells below to read dataset and build body length feature
         # 3) write code to answer the quiz questions
```

```
In [8]: spark = SparkSession.builder \
            .master("local") \
            .appName("Creating Features") \
            .getOrCreate()
```

1.0.1 Read Dataset

```
In [9]: stack_overflow_data = 'Train_onetag_small.json'
```

```
In [10]: df = spark.read.json(stack_overflow_data)
         df.persist()
```

```
Out[10]: DataFrame[Body: string, Id: bigint, Tags: string, Title: string, oneTag: string]
```

1.0.2 Build Body Length Feature

```
In [11]: regexTokenizer = RegexTokenizer(inputCol="Body", outputCol="words", pattern="\\W")
         df = regexTokenizer.transform(df)
```

```
In [12]: body_length = udf(lambda x: len(x), IntegerType())
         df = df.withColumn("BodyLength", body_length(df.words))
```

```
In [13]: df.head()
```

```
Out[13]: Row(Body="<p>I'd like to check if an uploaded file is an image file (e.g png, jpg, jpeg
```

2 Question 1

Select the question with Id = 1112. How many words does its body contain (check the BodyLength column)?

```
In [14]: df.select(["id", "BodyLength"]).where(df.Id == "1112").collect()
```

```
Out[14]: [Row(id=1112, BodyLength=63)]
```

3 Question 2

Create a new column that concatenates the question title and body. Apply the same functions we used before to compute the number of words in this combined column. What's the value in this new column for Id = 5123?

```
In [20]: appendTitleAndBody = udf(lambda x,y: x+y, StringType())
         df = df.withColumn("TitleAndBody", appendTitleAndBody(df.Title, df.Body))

         df = df.withColumn("TitleAndBodyLength", body_length(df.TitleAndBody))
         df.select(["id", "TitleAndBodyLength"]).where(df.Id == "5123").collect()
```

```
Out[20]: Row(Body="<p>I'd like to check if an uploaded file is an image file (e.g png, jpg, jpeg
```

4 Create a Vector

Create a vector from the combined Title + Body length column. In the next few questions, you'll try different normalizer/scaler methods on this new column.

```
In [23]: assembler = VectorAssembler(inputCols=["TitleAndBodyLength"], outputCol="TitleAndBodyLe
         df = assembler.transform(df)
```

5 Question 3

Using the Normalizer method what's the normalized value for question Id = 512?

```
In [26]: #scaler = Normalizer(inputCol="TitleAndBodyLengthVector", outputCol="ScaledNumFeatures'
         #df = scaler.transform(df)


         df.select(["id", "ScaledNumFeatures"]).where(df.Id == "512").collect()
```

```
Out[26]: [Row(id=512, ScaledNumFeatures=DenseVector([1.0]))]
```

6 Question 4

Using the StandardScaler method (scaling both the mean and the standard deviation) what's the normalized value for question Id = 512?

```
In [27]: scaler2 = StandardScaler(inputCol="TitleAndBodyLengthVector", outputCol="ScaledNumFeatu
         scalerModel = scaler2.fit(df)
         df = scalerModel.transform(df)
         df.select(["id", "ScaledNumFeatures2"]).where(df.Id == "512").collect()
```

```
Out[27]: [Row(id=512, ScaledNumFeatures2=DenseVector([0.2003]))]
```

7 Question 5

Using the MinMAxScaler method what's the normalized value for question Id = 512?

```
In [29]: from pyspark.ml.feature import MinMaxScaler
         scaler3 = MinMaxScaler(inputCol="TitleAndBodyLengthVector", outputCol="ScaledNumFeature
         scalerModel3 = scaler3.fit(df)
         df = scalerModel3.transform(df)

         df.select(["id", "ScaledNumFeatures3"]).where(df.Id == "512").collect()
```

```
Out[29]: [Row(id=512, ScaledNumFeatures3=DenseVector([0.0071]))]
```

```
In [ ]:
```