CS 6350 - ASSIGNMENT 6

Please read the instructions below before starting the assignment.

- This assignment can be done on Databricks or the UTD cluster. For both cases, you have to provide your code, README file indicating the dependencies of your code, and the output. For Databricks, you also have to submit a public URL of your notebook. Create separate folders for each part.

- You should use a cover sheet, which can be downloaded at: http://www.utdallas.edu/~axn112530/cs6350/CS6350_CoverPage.docx

- You are allowed to work in pairs i.e. a group of two students is allowed. Please write the names of the group members on the cover page. Only one submission per team is required.

- You have a total of 4 free late days for the entire semester. You can use at most 2 days for any one assignment. After that, there will be a penalty of 10% for each late day. The submission for this assignment will be closed 2 days after the due date.

- Please ask all questions on Piazza, and not through email to the instructor or TA.

ASSIGNMENT 6

Part I: GraphX

In this assignment, you will use Spark GraphX to analyze social network data. You are free to choose any one of the ***Social network*** datasets available from the SNAP repository. https://snap.stanford.edu/data/index.html#socnets

You will use this dataset to construct a GraphX graph and run some queries and algorithms on the graph.

Below are the steps that you will perform. You should use Scala or Python under Spark to accomplish all of these tasks. You are free to use either UTD cluster or Databricks.

Step I:

Load the data into a dataframe or RDD using Spark. Define a parser so that you can identify and extract relevant fields.
Note that GraphX edges are directed, so if your dataset has undirected relationships, you might need to convert those into 2 directed relationships. That is, if your dataset contains an undirected friendship relationship between X and Y, then you might need to create 2 edges – one from X to Y and the other from Y to X.

Step II:

Define edge and vertex structure and create property graphs.

Step III:

Run the following queries using the GraphX API:

a. Find the nodes with the highest outdegree and find the count of the number of outgoing edges

b. Find the nodes with the highest indegree and find the count of the number of incoming edges

c. Calculate PageRank for each of the nodes and output the top 5 nodes with the larges PageRank values. You are free to define the threshold parameter.

d. Run the connected components algorithm on it and find the node ids of the connected components.

e. Run the triangle counts algorithm on each of the vertices and output the top 5 vertices with the largest triangle count. In case of ties, you can randomly select the top 5 vertices.

What to submit:

1. Your code or public link to your Databricks notebook
2. A Readme file containing details of which dataset you used and any instructions on how to run your code

Part II: Spark Streaming

For this part, you will use Spark Streaming along with Twitter API to extract continuous tweets, filter them based on certain keywords, and then analyze their sentiment. You can do all this using Databricks community edition. You are free to choose the parameters for window length, slide interval, and total number of times to run. You should collect several hours' worth of tweets for successfully completing this part.

Below are the steps:

Step I:

Sign up for a Twitter account. You might need to setup an Twitter application account also. More details about this can be found in the tutorial below:
http://ampcamp.berkeley.edu/3/exercises/realtime-processing-with-spark-streaming.html

Make sure you copy your consumer key, consumer secret, access token, etc.

Step II:

Create a Spark streaming app using Databricks. An example of how to do this can be viewed here:
https://docs.cloud.databricks.com/docs/latest/databricks_guide/07%20Spark%20Streaming/03%20Twitter%20Hashtag%20Count%20-%20Scala.html

You are free to choose any values for parameters, such as slideInterval and windowLength.

Step III:

Instead of looking for hashtags as shown in the example mentioned in Step II, you will filter the tweets to those that contain the following word: ***iPhone X***. Your search should be case insensitive i.e. it should match ***iphone X, iphoneX,*** etc.

Step IV:
You will evaluate the sentiment of tweets using any one of the following techniques:

1. Count the number of positive and negative words in the tweet. A list of positive and negative words can be downloaded from:
http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar
If the number of positive words is greater than negative words, you would classify the tweet as positive else you would classify it as negative.

2. Using the CoreNLP API, which is available from:
https://stanfordnlp.github.io/CoreNLP/index.html

This API assigns a sentiment score to text ranging from 1 (most negative) to 5 (most positive). You will classify a tweet as positive if its score is greater than or equal to 2.5 else it would be classified as negative.

3. Using Python's NLTK tools:
http://text-processing.com/docs/sentiment.html

Step V:

For each window length, you will output the following:

Number of tweets containing the word iPhone X: _____
Number of positive tweets: _____
Number of negative tweets: _____

What to submit:

1. Your code or public link to your Databricks notebook
2. A Readme file containing details of which sentiment analysis technique you used and instructions on how to run your code.
3. Output of your code for several hours' worth of tweets.