CS 6350

ASSIGNMENT \_\_\_\_\_3\_\_\_\_\_\_\_

Names of students in your group:

Aditya Kulkarni (axk230069@utdallas.edu)

Number of free late days used: \_\_\_\_\_\_\_\_0\_\_\_\_\_\_\_\_\_\_\_   
Note: You are allowed a **total** of 4 free late days for the **entire semester**. You can use at most 2 for each assignment. After that, there will be a penalty of 10% for each late day.

Please list clearly all the sources/references that you have used in this assignment.

Solution:

Part 1: Report is from page 2.

Solution Link: <https://github.com/adityavkulkarni/6350_assignment3/tree/master/Part1>

Part 2: Report is from page 8.

Solution Link:

Google Colab link:

<https://colab.research.google.com/drive/1yP-7jeaYKonvGEgznyWzRck4rDjr39nS?usp=sharing>

Github link:

<https://github.com/adityavkulkarni/6350_assignment3/tree/master/Part2>

Dataset used:

<https://snap.stanford.edu/data/wiki-Vote.html>

Dataset is present in the GitHub repository: <https://github.com/adityavkulkarni/6350_assignment3/tree/master/Part2/input>

Attached in Submission:

1. *CS6350\_Assignment3 – combined report*
2. *ReadMe files for part 1 and part 2*
3. *Zip file of repository:* [*https://github.com/adityavkulkarni/6350\_assignment3*](https://github.com/adityavkulkarni/6350_assignment3)
4. *Output for Part 2 can be found in:* [*https://github.com/adityavkulkarni/6350\_assignment3/tree/master/Part2/output*](https://github.com/adityavkulkarni/6350_assignment3/tree/master/Part2/output)

Assignment 3 Part 1

Spark Streaming with Real Time Data and Kafka

**Problem Statement**

**In this part, you will create a Spark Streaming application that will continuously read text data from a real time source, analyze the text for named entities, and send their counts to Apache Kafka. A pipeline using Elasticsearch and Kibana will read the data from Kafka and analyze it visually.**

**Execution steps**

**Detailed steps are present in** [README.md](https://github.com/adityavkulkarni/6350_assignment3/blob/master/Part1/README.md)

**A screenshot of a computer program

Description automatically generated**

**Kafka, Spark and ELK need to be downloaded and setup for running the above steps. Libraries required to run are mentioned in:** [requirements.txt](https://github.com/adityavkulkarni/6350_assignment3/blob/master/Part1/requirements.txt)

**Data and Dashboard Description:**

**We have fetched top 1000 posts and then streamed 29,832 posts (submissions) from “r/all” which is a less filtered feed of the most popular posts on Reddit.**

**These posts are then published in Kafka topic “reddit”. The published posts are read in Pyspark streaming. After cleaning text and tokenizing it, we have identified the named entities using NLTK – “*pos\_tag*” and “*ne\_chunk*”. The named entities and their count are published to the topic “*ner*”. Logstash is configured to read from Kafka topic “*ner*” and push the data to “*nerkibana*” index in Kibana.**

**The dashboard has the following visualizations:**

1. **Bar graph showing 10 most frequent named entities**
2. **Pie chart showing frequency distribution**
3. **Word cloud of frequent named entities**
4. **Time steps showing the count of records analyzed per 5 minutes**
5. **Semi-circle meter plot for unique word count vs total word count**

**Output Analysis:**

**A total of 7.8 million unique named entities were identified and total named entity count is 24 million.**

**According to bar graph, the most mentioned word on “r/all” from 13:30 to 14:45 on 21st July 21, 2024, is “*Biden*” with about 358,000 mentions, followed by “*Joe*”, “*news*” and “*Trump*” with about 150,000 counts. This is due to the news about Joe Biden dropping out of presidential race which led to widespread news and follow up discussions across Reddit. All the top 10 frequent named entities are related to politics except for “*Engine”* which has about 68,000 mentions.**

**According to timestep graph, which shows number of records(named entities) pushed to Kibana in 5 minutes interval, we observed that: at *13:30* only 5,344 named entities were processed. But at *13:55*, the count increased to 14,882 which was when the news was out about the presidential candidates changing. And the count increased steadily to 24,252 at *14:30*.**

**Output:**

**Output of reddit\_scrapper.py:**

A black screen with blue text

Description automatically generated

**Dashboard:A screenshot of a graph

Description automatically generated**

**Bar Plots:**

***15 minutes:*A graph with green squares

Description automatically generated**

***A graph with green squares

Description automatically generated30 minutes:***

***45 minutes:***A graph with green squares

Description automatically generated

*A graph with green squares

Description automatically generated60 minutes:*

**Unique words vs Total word count:**

**A grey and black circular object

Description automatically generated with medium confidence**

**A graph of a bar chart

Description automatically generated with medium confidenceTimestep plot:**

Assignment 3 Part 2

Analyzing Social Networks using GraphX/GraphFrame

**Problem Statement**

**In this part, you will use Spark GraphX/GraphFrame to analyze social network data. You are free to choose any one of the social network datasets available from the SNAP repository.**

**You will use this dataset to construct a GraphX/GraphFrame graph and run some queries and algo- rithms on the graph.**

**Output of Queries**

1. Find the top 5 nodes with the highest outdegree and find the count of the number of outgoing edges in each.

A black background with white text

Description automatically generated

1. Find the top 5 nodes with the highest indegree and find the count of the number of incoming edges in each.

A black and white image of numbers

Description automatically generated

1. Calculate PageRank for each of the nodes and output the top 5 nodes with the highest PageRank values. You are free to define any suitable parameters.

A black and white screen with numbers and letters

Description automatically generated

1. Run the connected components algorithm on it and find the top 5 components with the largest number of nodes.

*Connected Components: Strongly Connected Components:*

A black and white screen with numbers and a white text

Description automatically generated with medium confidence A black and white screen with white text

Description automatically generated

1. 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.

A black background with white text

Description automatically generated

**Summary:**