Keenan Flynn: kpfxn8@umsystem.edu

ICP 2

**Problem Statement:** Write a spark program to group the words in a given text file based on their starting letters. Use the Text File provided with ICP (icp.txt)

YOUTUBE: <https://youtu.be/tdRdiGFoRFA>

Box and whisker chart

Description automatically generated

ICP Requirements:

1) Spark Integration with Colab (or IDE that you are using)

2) Creating a well commented Spark program and outputting the correct results and writing it to output file.

3) Code quality, wikiReport quality, video explanation

In this ICP, we learned how to set up Spark in an online environment (CoLab). We learned how Spark can do aggregating and manipulation of data fields within the Spark DataFrames.

There were several challenges in this project. The biggest one was getting the data into the DataFrame. Initially, we tried to directly read the input file using Sparks read() method. This resulted in a single cell with the entire string. We tried to split this cell but then received a list of all of the words inside 1 cell. Rather than continue to struggle with this part of the program, we decided to use the standard Python methods of reading a file. This proved much simpler and allowed us to quickly get the data in the format that we wanted.

Description of the Program

The in-class demonstration helped to create a Spark environment for CoLab. Several dependencies and libraries were set up for Spark. In addition to the libraries that we imported in the example, this ICP also makes use of the following libraries: HiveContext, functions, re. These libraries help with the aggregating the words into buckets and manipulating strings.

Text

Description automatically generated

As shown above, the input file was read in using standard python method. Using re and list methods and comprehension, the paragraph was split into a list of words, with all punctuation and spaces removed. The first letter of each word was capitalized for uniformity. With this list, we can easily create a Spark DataFrame as shown below.

Text

Description automatically generated

This gives us the following output. By creating a new column First\_Letter using the .substr() method, we can group the words together.

Timeline

Description automatically generated with low confidence

Before we get to our desired output, we want to clean and explore the data. This was done by removing any numbers from the DataFrame (ie 2020, 2019, etc). We can also compress our DataFrame by getting only the distinct values. Originally we had 251 rows, but after cleaning we have 151 values.

Text

Description automatically generated with medium confidence

Finally, we can group by First\_Letter and use .collect\_set() to show all words that start with a particular letter.

Text

Description automatically generated