Course: DataScience

Project Report

Graded Assessment Project 2

Prepared by: Georges Assaf

Date: Dec 2024

# Code Update

After reviewing Assessment Project 2, I identified several issues and applied fixes in the notebook code. These fixes may be specific to my environment due to the versions of Python libraries in use. **Three of the fixes are code-related, while the other three address issues in the analysis**.

## Code Fix 1

In the first section, i faced an issue and fixed it by adding the encoding='utf-8'

A screenshot of a computer

Description automatically generated

## Code Fix 2

In the print\_cluster\_sample() function, replace=False is used to ensure no duplicates, resulting in 5 unique bullets being returned. However, when printing cluster 6, duplicates were observed, and the function was unable to return 5 distinct bullets. I modified the code to ensure it returns a number of bullets ranging from 5 to the total number of unique bullets in a specific cluster. Below is the updated code:

A computer screen shot of a computer code

Description automatically generated

## Code Fix 3

The original function for computing cluster groups used total\_bullets. When attempting to use total\_bullet\_700, it raised an error indicating that the objects do not have the required sample length.

The updated version of the function is to add a new variable for total\_bullet implemented as below.

A screenshot of a computer program

Description automatically generated

# Loading Data

In this project, we loaded all HTML files available in the "job\_posting" folder into a list named html\_content. A total of 1,458 HTML files were successfully loaded.

# Parsing HTML Files

This code parses the loaded HTML files into BeautifulSoup objects called soup\_objects, ensuring each file has a title and body. The necessary import statement is:

A screenshot of a computer program

Description automatically generated

# Checking Title and Body Texts for Duplicates

We processed the soup\_objects to extract the <title> and <body> content from each HTML file, storing them in a dictionary called html\_dict. The body text is retrieved using soup.find('body') the function .get\_text(strip=True) can be used to extract and cleans the text inside the <body> tag by removing extra whitespace. The data is then converted into a pandas DataFrame (df\_jobs), where a summary of the extracted content is generated using df\_jobs.describe() and printed for analysis.

A screenshot of a computer code

Description automatically generated

# Display HTML ContentA screenshot of a computer Description automatically generated

# Examining the Top-Ranked Words in the HTML Bullet

A screenshot of a computer

Description automatically generated

# Vectorizing our Skills and The Job-Posting Data

This code combines the Body column of the df\_jobs DataFrame with an existing\_skills list to create a combined text\_list. It then uses TfidfVectorizer to transform the text data into a TF-IDF matrix, excluding English stop words, and converts it into an array for further analysis.

A computer code with text

Description automatically generated with medium confidence

# Clustering the job Skills into 15 Clusters

I used KMeans as clustering algorithm to group data from the shrunk\_norm\_matrix into 15 clusters. It fits the model and predicts the cluster assignments for each data point, associating each with a specific cluster. A DataFrame is created to store the index, cluster labels, and corresponding bullet, providing a structured overview of the clustering results.

A screenshot of a computer program

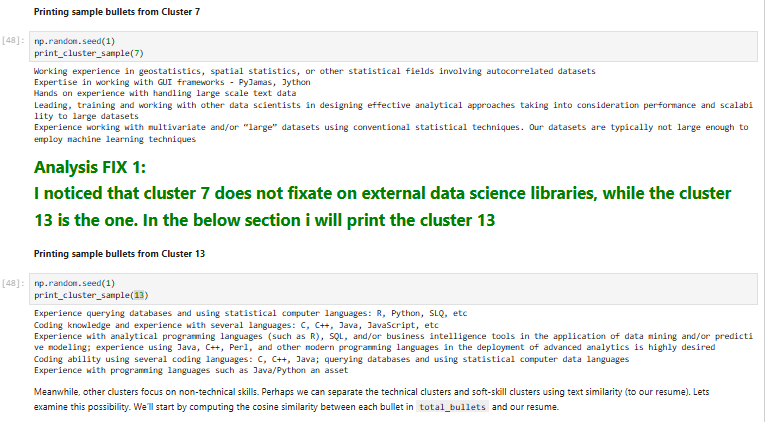
Description automatically generated

# Printing Cluster that Fixates on external data Science Libraries(Scikit-Learn,Pandas)

In this section as per the notebook its mentioned that cluster 7 fixates on external data science libraries such as Scikit-Learn, Pandas, Numpy, etc. After printing the cluster 7, I noticed that this cluster content does not match what is written however, the cluster 13 is the one that focuses on the data science libraries.

A collage of words

Description automatically generated



# Printing Sample Bullets from Clusters that Contains Minimal Count of Years

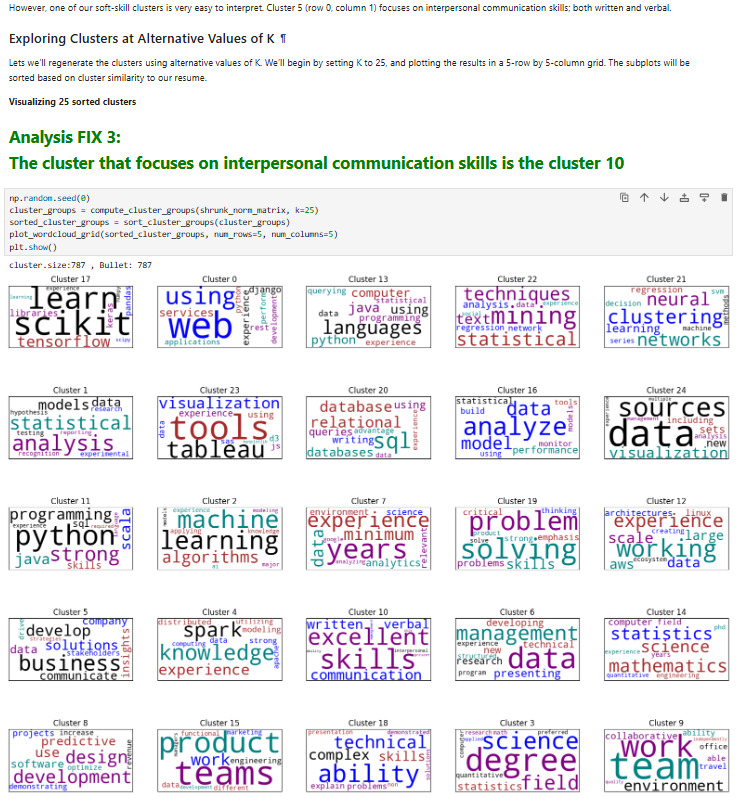
After fixing the function print\_cluster\_sample(), its now returning the unique element. In the following, we can see that cluster 6 is now returning only 1 element and not 5.

In addition the cluster 4 is the cluster that contains the minimum number of years.

A screenshot of a computer

Description automatically generated

# Printing Cluster that Focuses on Interpersonal Communication Skills



# Visualizing 20 Sorted Cluster

A screenshot of a computer

Description automatically generated

# Analyzing the 700 Most-Relevant Postings

The below code takes the first 700 job descriptions from the sorted\_df\_jobs DataFrame and gathers all the unique bullet points, removing extra spaces. It then converts these bullet points into numerical values using TF-IDF vectorization, which helps capture the importance of each word while ignoring common words. After that, the matrix is shrunk to reduce its size using the shrink\_matrix function. Finally, the code prints how many bullet points have been processed and shrunk.

A screenshot of a computer program

Description automatically generated

In the function compute\_cluster\_groups, I added a variable var\_totalbullet which contains the total bullets

A screenshot of a computer

Description automatically generated