Python Similarity Metrics

Veeral Patel

Regis University

April 14th, 2021

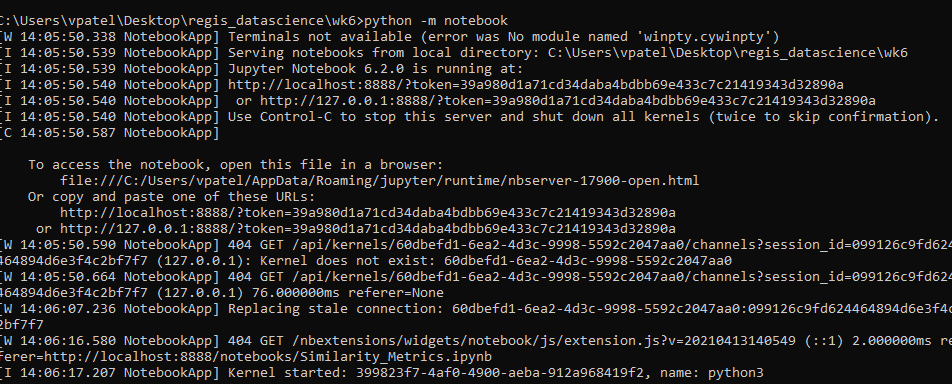
1. **Introduction**

For this project, to have a python notebook installed start installing Spark. In order to install Spark on a windows PC, we had to have installed Oracle Virtual Box, Ubuntu 18.04 and the necessary modules for spark in the virtual environment. For the python portion of the homework, we were given code and we were to identify what is going on, the results, the significance, and make sure that we are able to run the code in the notebook. The code contained Euclidian distance, cosine similarity, Pearson correlation, on a simple movie recommender system.

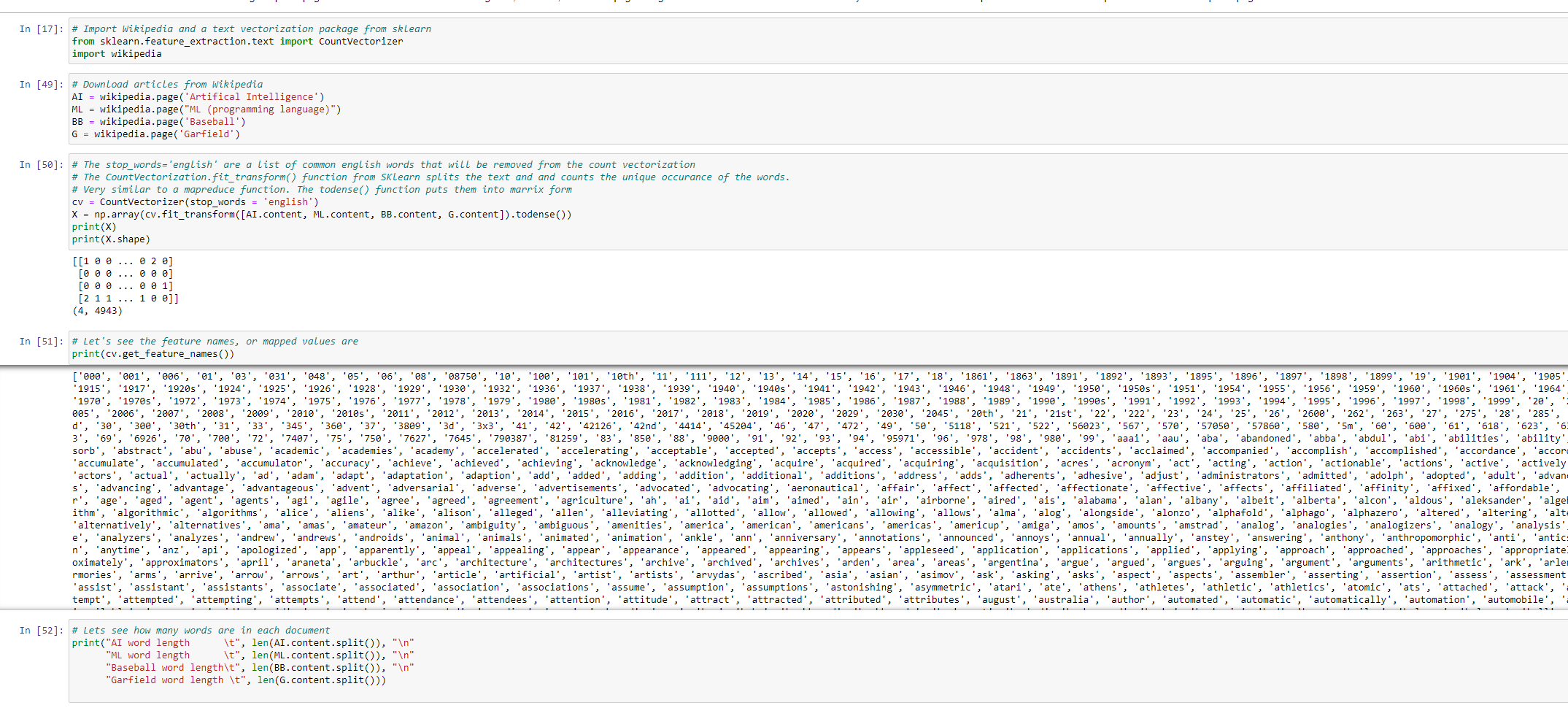
1. **Methods/Code/Screenshots**

I used a windows computer and I had to install spark. I used the PDF that was given to the class as well as some tutorials I found on YouTube. I had difficulty with pip and pyspark but was able to figure it out of a while. Also for the python portion of the homework, I had to install the Wikipedia module to get the code to run. In the Euclidian and Cosine example, I could not download the Wikipedia.page(‘Machine Learning’). Therefore, I changed that Wikipedia page to “ML (programming language).”

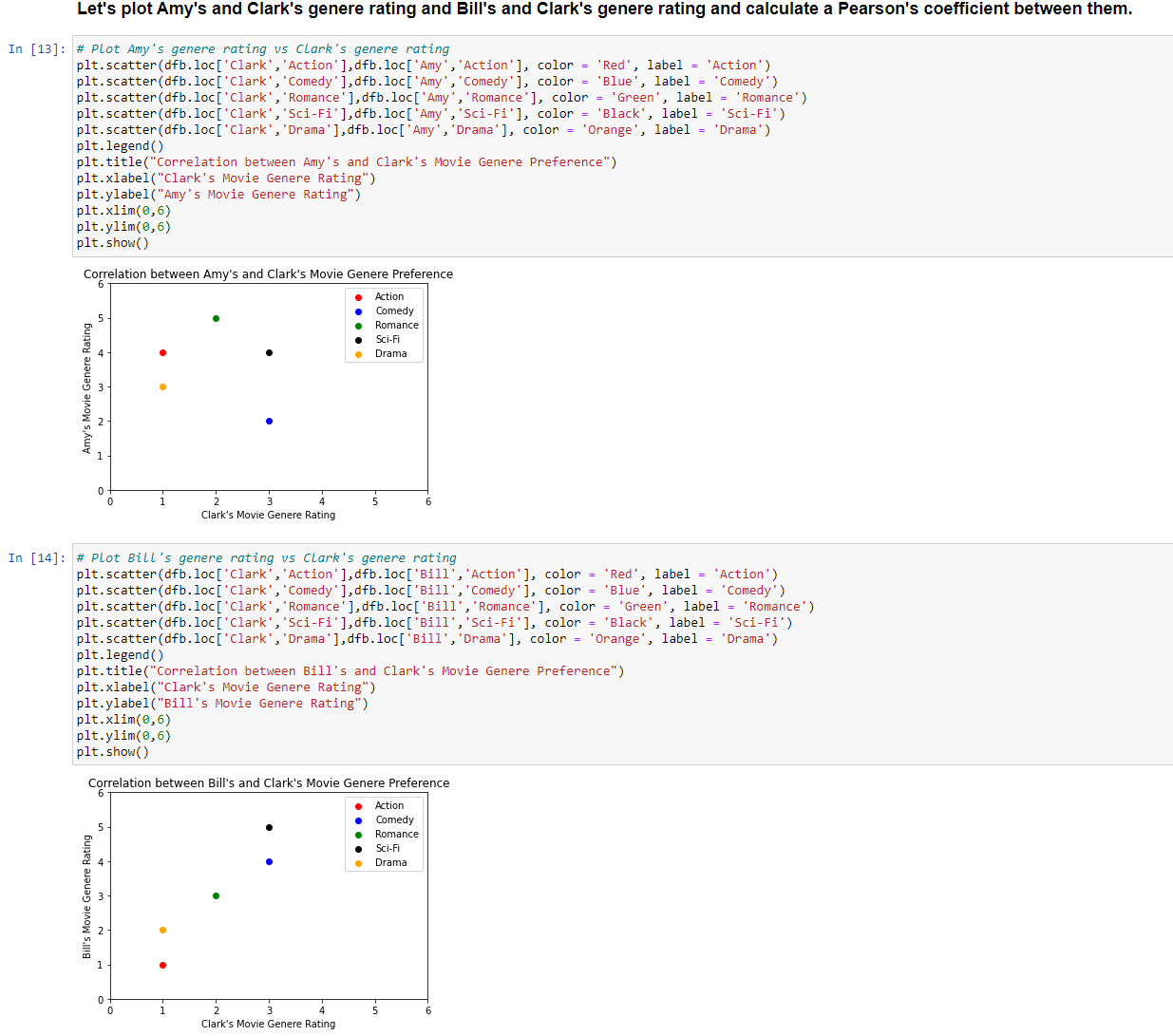
Examples of Python Notebook Running:

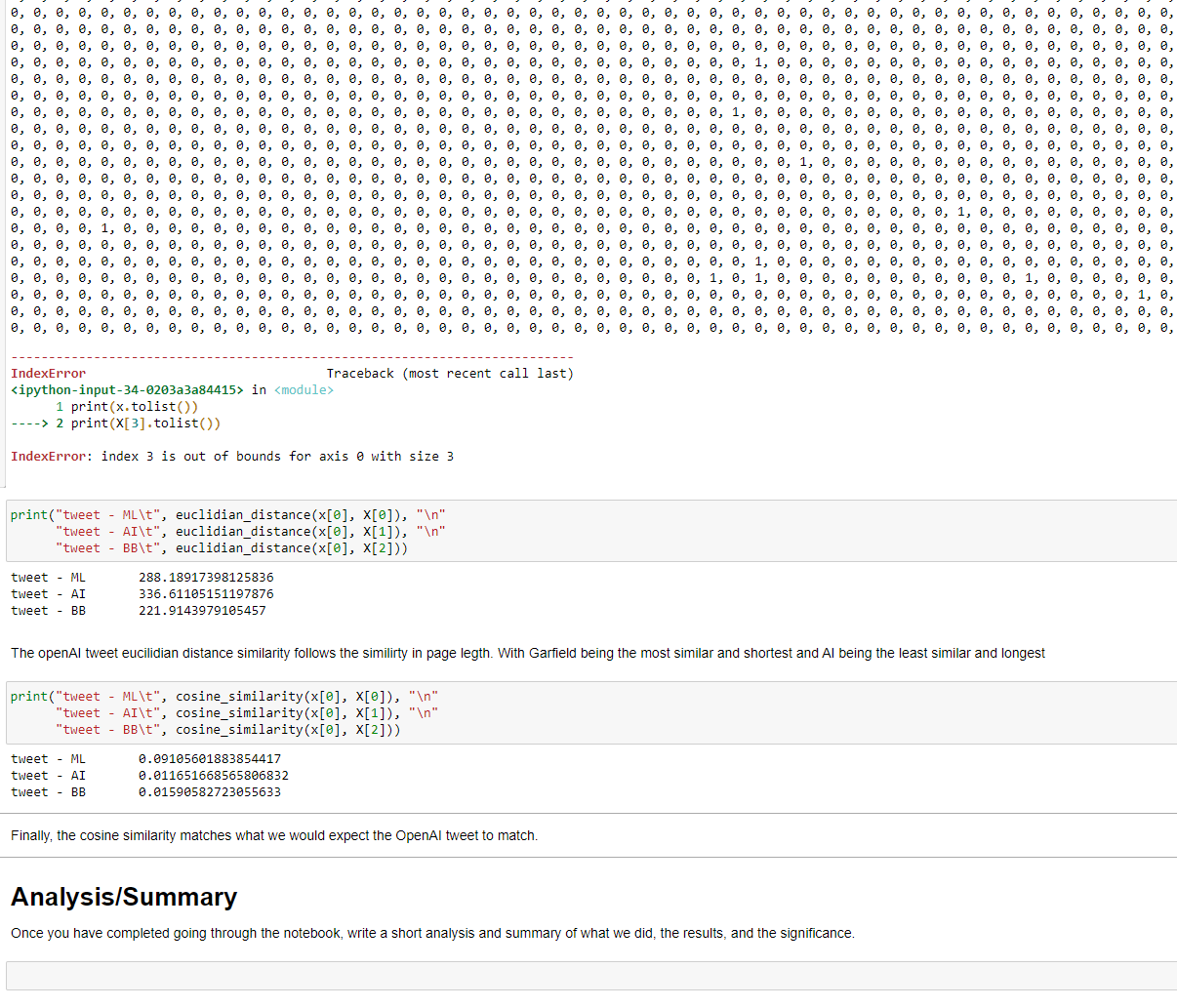


Had to change code from ‘Machine Learning’ to ‘ML (programming language)’:



Scatter Plot from Amy and Clark’s movie genre rating:



1. **Results/Analysis**

1. **Analysis of Results**

In this lab, we did the following step by step:

1. Created an array with movie ratings as well as two columns, with an index. Then we had to convert that array to a dataframe. We use a dataframe in python because it is much easier to work with because it plays nicely with other libraries such as sklearn and matplotlib.
2. Then we use matplotlib to plot the data of action and comedy movies in regard to the movie ratings by Amy, Bill and Clark.
3. Then we write a function to calculate the Euclidean distance. The Euclidean distance is often used to find the closest path.
4. Then we calculate the cosine similarity. The cosine similarity is useful when trying to find the similarity between points in a space. The higher the cosine similarity, the closer the two points usually are. Here we see that Bill and Clark are closer than Amy and Clark.
5. Then we add more dimensions to the matrix by adding genres: Romance, Sci-Fi, and Drama. Additionally, we add more reviews as well.
6. Again, we plot the data and this time we calculate the Pearson Coefficient. The Pearson Coefficient tells us the correlation or association between two continuous variables. Pearson’s coefficient agrees that Clark's genre preference is more similar to Bill's than Amy's and therefore we would use Bill's movie ratings to help inform and recommend movies to Clark.
7. Then we import Wikipedia to look at Euclidian distance and cosine differences in text analysis. We take 3 Wikipedia pages: Artificial Intelligence, Baseball, and Garfield.
8. First, we take all that data, and remove all the common words such as: as, and, for, I, etc. Then we transform that data to an array from a vector.
9. Then, but taking the Euclidian distance from AI to Baseball, then from AI to Garfield. By taking the Euclidean distance, we can see that the distance between AI and ML is less than the distance between AI and Garfield.
10. Taking the Cosine Similarity, we see that the cosine similarity is higher between AI and ML than with ML and G.
11. Then, we take a tweet and use it see the magnitude affect between Euclidian distance and cosine similarity. We see that the OpenAI tweet has the most in common to Machine learning
12. We can also see, through cosine similarity, that the tweet and the Wikipedia ML page has the most similarity since it has the highest cosine similarity number.
13. **Conclusion**

I thought that this was a really good introduction to cosine similarity and Euclidean distance. I think that by using real data that is from the Wikipedia page it made it a lot easier. Note, I could not get this to work: ML = wikipedia.page("Machine Learning"), therefore I changed it to: ML = wikipedia.page("ML (programming language)") for the entire dataset.

I was able to really use it to learn about how Euclidean distance works and how cosine similarity works. I found the example interesting when by just using Euclidean distance we got the information that the Garfield page was more similar to the AI page than the ML page was. This showed that only one similarity test may not be enough to get a real assessment. Here, we were then able to run the cosine similarity test in order to get the actual result. I think that this is something that certainly has applications in many fields. There are many times when you might want to see what is similar to each other. I feel that there is an easy way to do this with the python tools that were used in this lab.

1. **Reference(s):**

Gower, J. C. (1985). Properties of Euclidean and non-Euclidean distance matrices. *Linear Algebra and its Applications*, *67*, 81-97.

Verma, N., Mahajan, D., Sellamanickam, S., & Nair, V. (2012, June). Learning hierarchical similarity metrics. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 2280-2287). IEEE.