Jason Ballantyne
13432788

Text Analytics COMP47600
Practical Number 5

23/10/2021

Q1:

The category of things I have chosen is “Games”. The 5 instances of this category are "Shooter", "RPG", "Action", "Real-time strategy" & "Sandbox”. Please see the list of feature words below for each instance.

```
ShooterGame = "Fast", "Intense", "Reaction", "Movement", "Tactical"
RPG = "Slow", "Challenge", "Story", "Combat", "Exploration"
Action = "Fast", "Puzzle", "Story", "Combat", "Survival"
Real_timeStrategy = "Arcade", "Story", "Tactical", "Combat", "Turn-based", "Planning"
Sandbox = "Slow", "Relaxed", "Exploration", "Creative", "Puzzle"
```

a) Jaccard Distance measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1. The formula for the Jaccard Distance can be seen below:

$$d_\mu(A,B) = 1 - J_\mu(A,B) = \frac{\mu(A \triangle B)}{\mu(A \cup B)}.$$

A pairwise comparison between each of the 5 instances to calculate the Jaccard distance. A matrix with the distance values can be found below.

Jaccard Distance					
	Shooter	**RPG**	**Action**	**Real Time Strategy**	**Sandbox**
Shooter	0	1	0.89	0.9	1
RPG	1	0	0.75	0.78	0.75
Action	0.89	0.75	0	0.78	0.89
Real Time Strategy	0.9	0.78	0.78	0	1
Sandbox	1	0.75	0.89	1	0

A score of 1.0 illustrates that two sets are completely dissimilar. Based on the above matrix, we can see a score of 1.0 for Shooter and RPG set, the Shooter and Sandbox set and the Strategy and Sandbox set.
On the other hand, the most similar sets from the above matrix are the RPG and Action and the RPG and Sandbox set, both with a Jaccard distance score of 0.75.

b) The Triangle Inequality says that in any triangle, the sum of any two sides must be greater than the third side. Any side of a triangle must be shorter than the other two sides added together. If it is longer, the other two sides won't meet. This is the basis of the Triangle Inequality theorem.
We will now test that the property of the triangle inequality holds for the Jaccard-Distance measure for the values I have found from part (a). Please see the results below:

```python
def TriangleInequality(a, b, c):
    if (a + b <= c) or (a + c <= b) or (b + c <= a):
        return False
    else:
        return True
```

Testing the Triangle Inequality on our Jaccard Distance values

```
TriangleInequality(ShooterRPG,ShooterAction,ShooterStrategy)
```
True

```
TriangleInequality(ShooterSandbox,RPGAction,RPGStrategy)
```
True

```
TriangleInequality(RPGSandbox,ActionStrategy,ActionSandbox)
```
True

```
TriangleInequality(ShooterRPG,StrategySandbox,ActionSandbox)
```
True

```
TriangleInequality(RPGSandbox,ShooterSandbox,ActionSandbox)
```
True

```
TriangleInequality(ShooterSandbox, ActionSandbox, RPGStrategy)
```
True

As we can see from the above output, our results for the Jaccard Distance do satisfy the triangle inequality. The Jaccard distance Distance is known to fulfil all properties of a metric, most notably, the triangle inequality.

c) The Dice Coefficient is a statistical tool which measures the similarity between two sets of data. It is known by several other names such as Sørensen–Dice index or Sørensen index. The formula can be seen below:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

I have implemented the Dice Coefficient and the results can be seen below:

Dice Coefficient					
	Shooter	**RPG**	**Action**	**Real Time Strategy**	**Sandbox**
Shooter	1	0.24	0.22	0.33	0.29
RPG	0.24	1	0.52	0.45	0.5
Action	0.22	0.52	1	0.5	0.35
Real Time Strategy	0.33	0.45	0.5	1	0.24
Sandbox	0.29	0.5	0.35	0.24	1

If we compare the results for Jaccard Distance and Dice Coefficient, there are several similarities and differences which we will discuss below.

Similarities:

The similarities are that Dice Coefficient is also used to compare similarity of two sets and also uses binary-feature similarity.

Differences:

Dice Coefficient does not satisfy the triangle inequality. Due to this, it is not considered a proper metric.

Q2:

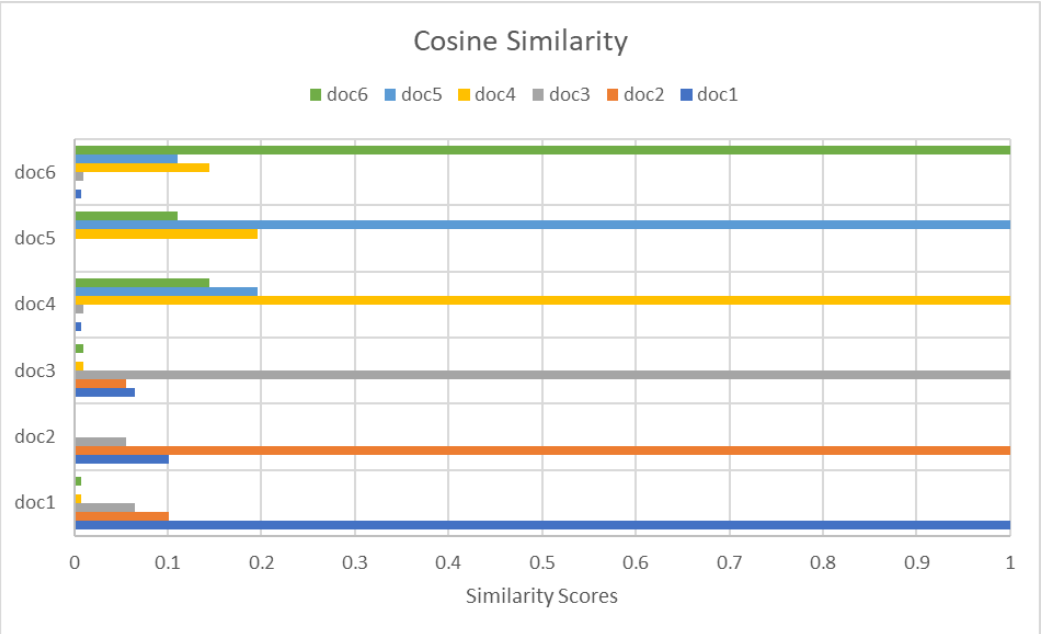
a) The first three online articles I have chosen are related to Prince Harry and Meghan Markle's wedding. The next three online articles I have chosen are related to Kim Kardashian and Kanye West's wedding. Both these events are related under the category of "wedding". Please see the full set of docs below:

```
doc1=('d1', "Here's What Prince Harry and Meghan Markle's Wedding Day Was Like")
doc2=('d2', "Prince Harry and Meghan had just one official ceremony says Justin Welby")
doc3=('d3', "Meghan and Harry finally admit there was no secret backyard wedding")
doc4=('d4', "Kim Kardashian and Kanye West Reenact Their Wedding Vows in Balenciaga Haute Couture")
doc5=('d5', "Kim Kardashian and Kanye West marry in Florence")
doc6=('d6', "Kim Kardashian and Kanye West: A Look Back at Their 2014 Wedding")
```

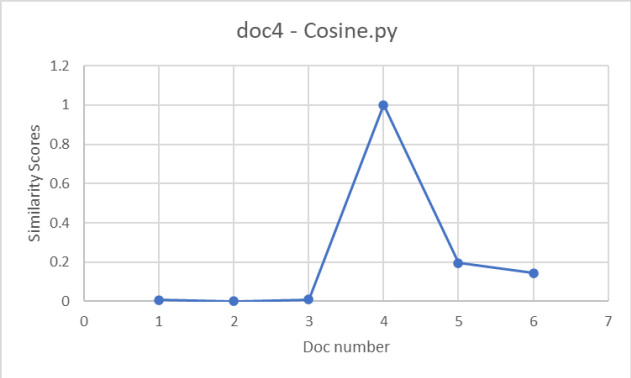
The pairwise cosine-similarity results for each document can be seen below. The output has been stored in a matrix similar to the previous results to see the pairwise for every combination between both the Prince Harry and Meghan Markle wedding and the Kim Kardashian and Kanye West wedding.

Cosine Similarity						
	doc1	**doc2**	**doc3**	**doc4**	**doc5**	**doc6**
doc1	1	0.101	0.065	0.008	0	0.008
doc2	0.101	1	0.056	0	0	0
doc3	0.065	0.056	1	0.01	0	0.01
doc4	0.008	0	0.01	1	0.196	0.145
doc5	0	0	0	0.196	1	0.11
doc6	0.008	0	0.01	0.145	0.11	1

b) The Cosine Similarity scores for every doc have been plotted in the bar chart below:



I created another plot to further illustrate the clustering that is occurring. This time, I produced a scatter plot which shows the similarity between "doc4" and every other "doc". Please see scatter plot below:



From the graphs, it is clear that the similarity scores are highest within their respective groups. By this I mean, that clusters appear for each doc within group 1 (doc1, doc2, doc3). Similarly, clusters appear within group2's respective group (doc4, doc5, doc6). As soon as scores are tested against docs outside of their group, there is a lack of clustering. An example of this is evident in the above graph where doc4 compared with doc2 gives a Cosine Similarity score of 0.

We will now trace where the sources of these feature words exist. For group 1, we can see that the clustering is caused by the following feature words: "harry", "meghan", "prince" & "wedding".

For group 2, we can see that the clustering is caused by the following feature words: "west", "kanye", "kim", "kardashian" & "wedding".

c) I have implemented sklearn's Cosine Similarity method, please see the results below using the same matrix we have seen for previous results.

Cosine_Similarity from sklearn						
	doc1	**doc2**	**doc3**	**doc4**	**doc5**	**doc6**
doc1	1	0.21	0.3	0.09	0.06	0.1
doc2	0.21	1	0.15	0.04	0.05	0.05
doc3	0.3	0.15	1	0.09	0.06	0.1
doc4	0.09	0.04	0.09	1	0.43	0.42
doc5	0.06	0.05	0.06	0.43	1	0.37
doc6	0.1	0.05	0.1	0.42	0.37	1

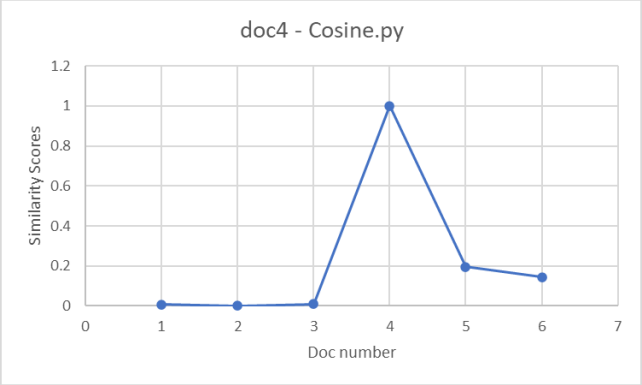
The Manhattan Distance is the distance between two points measured along axes at right angles. It is calculated by the formula below.

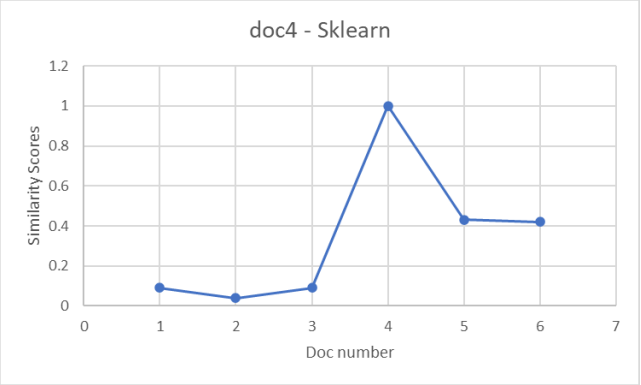
$$d(x,y) = \sum |x_i - y_i|$$

I have implemented sklearn's Manhattan Distance method also, please see the results below:

Manhattan_distances from sklearn						
	doc1	**doc2**	**doc3**	**doc4**	**doc5**	**doc6**
doc1	0	4.91	4.11	6.1	5.68	5.69
doc2	4.91	0	5.4	6.64	5.85	6.34
doc3	4.11	5.4	0	6.06	5.68	5.69
doc4	6.1	6.64	6.06	0	3.66	3.6
doc5	5.68	5.85	5.68	3.66	0	3.62
doc6	5.69	6.34	5.69	3.6	3.62	0

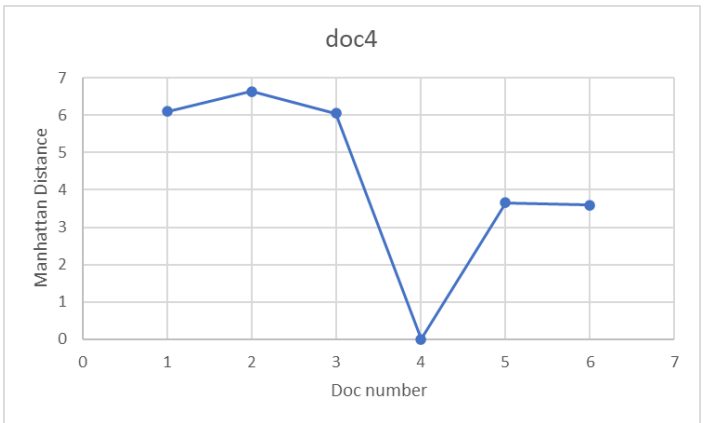
Based on the results for the Cosine Similarity and Manhattan Distance from sklearn above, we can see that the Cosine Similarity scores are very similar between our Cosine.py program and our sklearn method. To better illustrate this, we will recreate our earlier "doc4" plot using sklearn's scores.





As you can see, both plots follow very similar trends, further highlighting the our related scores using both Cosine.py and sklearn method.

If we look at the Manhattan distance results, it appears to be high for docs that are very dissimilar and low for documents that are very similar. This is the stark opposite of the previous results we have seen for the Similarity Scores. Once again, to better understand this, we will use a "doc4" plot for the Manhattan Distance for comparison. Please see plot below:



The Manhattan distance plot is inverted to that of the Cosine similarity plots. This is due to the Cosine Similarity represents the cosine of the angle between two vectors and determines if two vectors point to almost similar directions whereas the Manhattan distance is calculated as the sum of the absolute differences between the two vectors.