18CS10069_Assignment1_Q10

September 15, 2021

Name: Siba Smarak Panigrahi

Roll No.: 18CS10069

NOTE: The answers to the questions can be found in the text of the this document.

[13]:

```
import pandas as pd
import wikipedia
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans

articles= ['Linear algebra',
'Data Science',
'Artificial intelligence',
'European Central Bank',
'Financial technology',
'International Monetary Fund',
'Basketball',
'Swimming',
'Cricket']

wiki_lst, title = [], []
for article in articles:
    wiki_lst.append(wikipedia.page(article).content)
    title.append(article)
```

[14]:

```
vectorizer = TfidfVectorizer(stop_words={'english'})
X = vectorizer.fit_transform(wiki_lst)
```

[15]:

```
print(f' +++++ Shape of the input data: {X.shape}')
```

```
+++++ Shape of the input data: (9, 8285)
```

0.0.1 Ans 10 (a),(b) with k=4

[43]:

```
# kmeans with k=4
kmeans4 = KMeans(n_clusters=4, random_state=0)
kmeans4 = kmeans4.fit(X)
print(f'+++++ Jclust value: {kmeans4.inertia_/X.shape[0]}')
print(f'+++++ Finding Document Association +++++')
for i, val in enumerate(kmeans4.labels_):
    print(f"\t +++++ Article Name: {articles[i]} {'-' * (30 -␣
 ↪(len(articles[i]))) } Cluster: {kmeans4.labels_[i]}")
```

```
+++++ Jclust value: 0.20494844342027826
+++++ Finding Document Association +++++
         +++++ Article Name: Linear algebra ---------------- Cluster: 2
         +++++ Article Name: Data Science ------------------ Cluster: 1
         +++++ Article Name: Artificial intelligence ------- Cluster: 0
         +++++ Article Name: European Central Bank --------- Cluster: 0
         +++++ Article Name: Financial technology ---------- Cluster: 0
         +++++ Article Name: International Monetary Fund --- Cluster: 0
         +++++ Article Name: Basketball -------------------- Cluster: 0
         +++++ Article Name: Swimming ---------------------- Cluster: 3
         +++++ Article Name: Cricket ----------------------- Cluster: 0
```

0.0.2 Ans 10 (a),(b) with k=8

[44]:

```
# kmeans with k=8
kmeans8 = KMeans(n_clusters=8, random_state=0)
kmeans8 = kmeans8.fit(X)
print(f'+++++ Jclust value: {kmeans8.inertia_/X.shape[0]}')
print(f'+++++ Finding Document Association +++++')
for i, val in enumerate(kmeans8.labels_):
    print(f"\t +++++ Article Name: {articles[i]} {'-' * (30 -␣
 ↪(len(articles[i]))) } Cluster: {kmeans8.labels_[i]}")
```

```
+++++ Jclust value: 0.025551532806781845
+++++ Finding Document Association +++++
         +++++ Article Name: Linear algebra ---------------- Cluster: 2
         +++++ Article Name: Data Science ------------------ Cluster: 1
         +++++ Article Name: Artificial intelligence ------- Cluster: 5
         +++++ Article Name: European Central Bank --------- Cluster: 0
         +++++ Article Name: Financial technology ---------- Cluster: 4
         +++++ Article Name: International Monetary Fund --- Cluster: 7
         +++++ Article Name: Basketball -------------------- Cluster: 6
         +++++ Article Name: Swimming ---------------------- Cluster: 3
         +++++ Article Name: Cricket ----------------------- Cluster: 6
```

0.0.3 Ans 10 (c)

- With the help of only `Jclust` values, from comparison we can say that since it is lower for `k=8`, hence 8 is the optimal number of clusters.