Data mining assignment 2: Data Clustering (k-means and k-medians algorithms)

Saeth Wannasuphoprasit, Student ID: 201585689

Question 1 and Question 2:

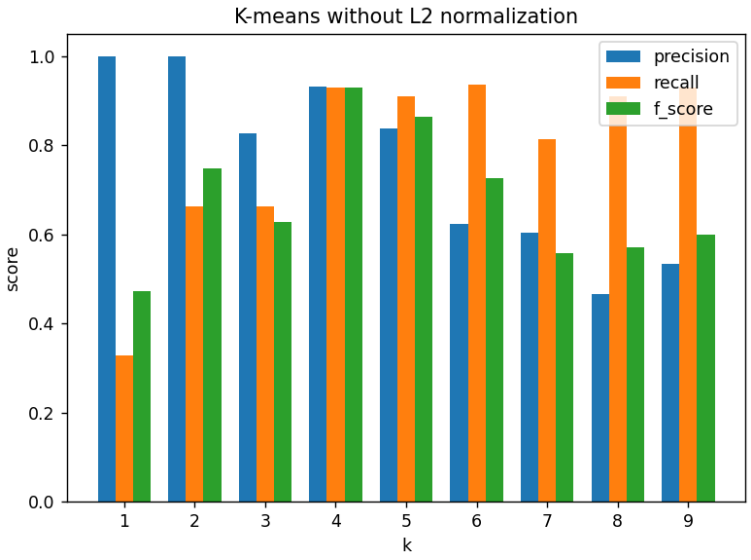
k-means clustering and k-medians clustering algorithms were implemented in 'score function' in the given python file. All the processes are the same except the calculating distances (k-means used euclidean distance, and k-medians used L1 distance), and the optimizing step (k-means used mean method and k-medians used median method to define new centroids). The two different processes are shown below.

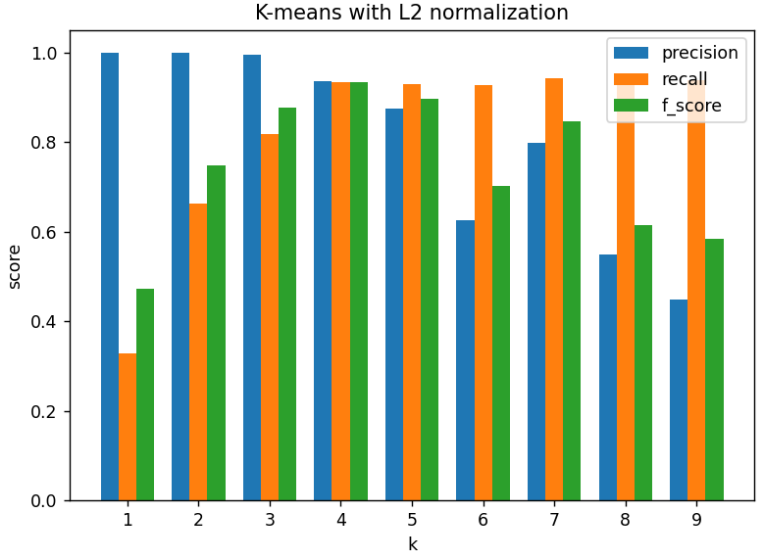
```python
# calculate distances
if method == 'mean':  # k-means method
    # euclidean distance
    distances[:, i] = np.linalg.norm(X - c, axis=1)
else:  # k-medians method
    # l1 distance
    distances[:, i] = np.linalg.norm(X - c, ord=1, axis=1)
```

```python
# optimize step
for c in range(k):
    if method == 'mean':  # k-means method
        # update centroids using mean
        centroids[c] = np.mean(X[classes == c], 0)
    else:  # k-medians method
        # update centroids using median
        centroids[c] = np.median(X[classes == c], 0)
```

Question 3, Question 4, Question 5, Question 6

precision, recall, and F-score for each k cluster were calculated (k-means without L2 normalization, k-means with L2 normalization, k-medians without L2 normalization, and k-medians with L2 normalization). The results, as well as the graphs, are shown below.





k-means without L2 normalization

```
precisions for k = 1 to 9:
k = 1/ precision: 1.0
k = 2/ precision: 1.0
k = 3/ precision: 0.825707067829126
k = 4/ precision: 0.9304017715912729
k = 5/ precision: 0.8374733734050407
k = 6/ precision: 0.6238944747809587
k = 7/ precision: 0.6038180123568639
k = 8/ precision: 0.46575600797242395
k = 9/ precision: 0.5333375252085601
```

```
recalls for k = 1 to 9:
k = 1/ recall: 0.32871344537029357
k = 2/ recall: 0.6623558453999443
k = 3/ recall: 0.6623558453999443
k = 4/ recall: 0.9295314926723446
k = 5/ recall: 0.9089284643520998
k = 6/ recall: 0.9348006206595242
k = 7/ recall: 0.8139797859739916
k = 8/ recall: 0.9094831253722476
k = 9/ recall: 0.9296583264777781
```

```
f-scores for k = 1 to 9:
k = 1/ f-score: 0.4723145424389042
k = 2/ f-score: 0.7468488584696521
k = 3/ f-score: 0.6267572291554255
k = 4/ f-score: 0.9292925719257741
k = 5/ f-score: 0.8626064267401339
k = 6/ f-score: 0.7254882922968137
k = 7/ f-score: 0.5568335567003825
k = 8/ f-score: 0.5708519363431072
k = 9/ f-score: 0.5997137353386113
```

```
average precision:  0.7578209147938052
average recall:  0.7866452168531297
average f-score:  0.6767452388232006
```

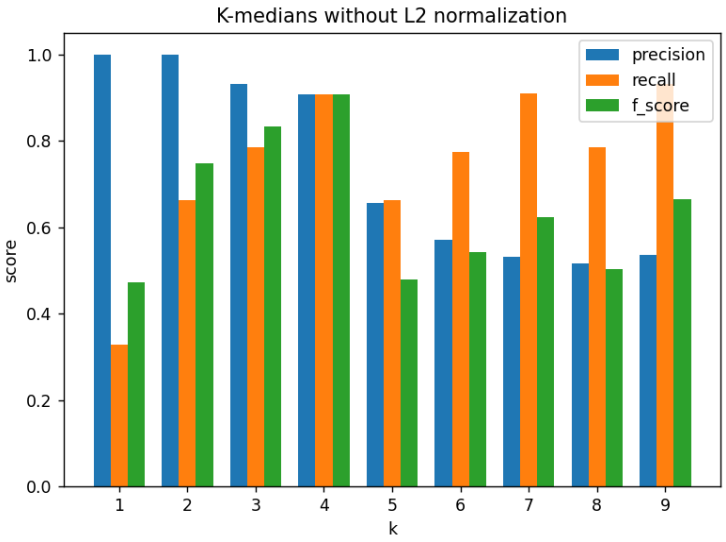
k-means with L2 normalization

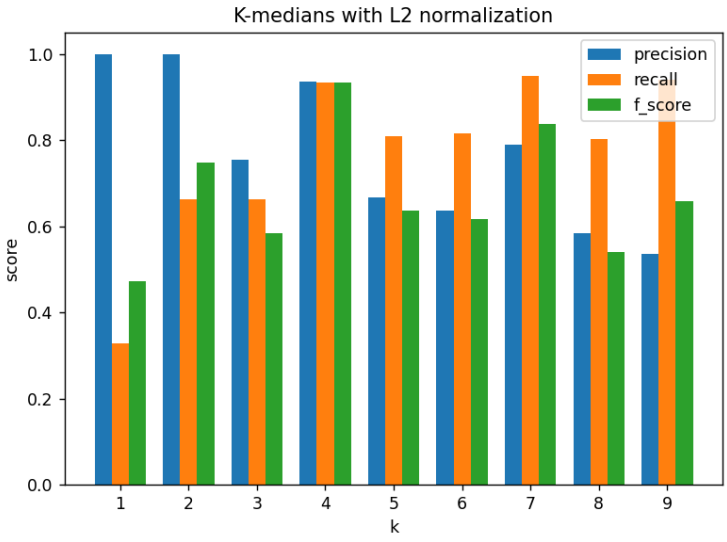
```
precisions for k = 1 to 9:
k = 1/ precision: 1.0
k = 2/ precision: 1.0
k = 3/ precision: 0.9939892439101543
k = 4/ precision: 0.9347252979015129
k = 5/ precision: 0.8735041653485219
k = 6/ precision: 0.6252515943862939
k = 7/ precision: 0.797954233892229
k = 8/ precision: 0.5487341519279678
k = 9/ precision: 0.44905348112313537
```

```
recalls for k = 1 to 9:
k = 1/ recall: 0.32871344537029357
k = 2/ recall: 0.6623558453999443
k = 3/ recall: 0.8181760352060001
k = 4/ recall: 0.9340775365651844
k = 5/ recall: 0.9288449494646652
k = 6/ recall: 0.9275037215116595
k = 7/ recall: 0.9429757497719848
k = 8/ recall: 0.9387740256778508
k = 9/ recall: 0.9382807679282807
```

```
f-scores for k = 1 to 9:
k = 1/ f-score: 0.4723145424389042
k = 2/ f-score: 0.7468488584696521
k = 3/ f-score: 0.8775968384431969
k = 4/ f-score: 0.9339363088464152
k = 5/ f-score: 0.8958643579657513
k = 6/ f-score: 0.700841629746088
k = 7/ f-score: 0.8458423182756605
k = 8/ f-score: 0.6143670391398631
k = 9/ f-score: 0.583517825671222
```

```
average precision:  0.8025791298322016
average recall:  0.8244113418773181
average f-score:  0.7412366354440837
```





k-medians without L2 normalization

```
precisions for k = 1 to 9:
k = 1/ precision: 1.0
k = 2/ precision: 1.0
k = 3/ precision: 0.9322409568479162
k = 4/ precision: 0.9079490398968291
k = 5/ precision: 0.6568465439626204
k = 6/ precision: 0.5699882692991632
k = 7/ precision: 0.5317229982649577
k = 8/ precision: 0.5157395090135126
k = 9/ precision: 0.5367067711229601
```

```
recalls for k = 1 to 9:
k = 1/ recall: 0.32871344537029357
k = 2/ recall: 0.6623558453999443
k = 3/ recall: 0.784618869634163
k = 4/ recall: 0.9074332984667065
k = 5/ recall: 0.6623558453999443
k = 6/ recall: 0.7729485711137088
k = 7/ recall: 0.9084375620338594
k = 8/ recall: 0.7852515985543487
k = 9/ recall: 0.9310666621144655
```

k-medians with L2 normalization

```
precisions for k = 1 to 9:
k = 1/ precision: 1.0
k = 2/ precision: 1.0
k = 3/ precision: 0.7545919045719605
k = 4/ precision: 0.9347252979015129
k = 5/ precision: 0.6677827534463429
k = 6/ precision: 0.6356677493494393
k = 7/ precision: 0.7888938885734055
k = 8/ precision: 0.5828394845827404
k = 9/ precision: 0.5347796350841602
```

```
recalls for k = 1 to 9:
k = 1/ recall: 0.32871344537029357
k = 2/ recall: 0.6623558453999443
k = 3/ recall: 0.6623558453999443
k = 4/ recall: 0.9340775365651844
k = 5/ recall: 0.8096898340127135
k = 6/ recall: 0.8161432105274583
k = 7/ recall: 0.9483401838860899
k = 8/ recall: 0.8026023865203575
k = 9/ recall: 0.940695786139213
```

k-medians without L2 normalization

```
f-scores for k = 1 to 9:
k = 1/ f-score: 0.4723145424389042
k = 2/ f-score: 0.7468488584696521
k = 3/ f-score: 0.8332921774912532
k = 4/ f-score: 0.9068949937167777
k = 5/ f-score: 0.47839667743224246
k = 6/ f-score: 0.5431898395265948
k = 7/ f-score: 0.6241044550162188
k = 8/ f-score: 0.5029941898063909
k = 9/ f-score: 0.6654932315186218
average precision:  0.739021565378662
average recall:  0.7492424108986039
average f-score:  0.6415032183796284
```

k-medians with L2 normalization

```
f-scores for k = 1 to 9:
k = 1/ f-score: 0.4723145424389042
k = 2/ f-score: 0.7468488584696521
k = 3/ f-score: 0.5831866365495567
k = 4/ f-score: 0.9339363088464152
k = 5/ f-score: 0.6355667979994266
k = 6/ f-score: 0.6175180303586136
k = 7/ f-score: 0.8363022796464057
k = 8/ f-score: 0.5406299337608135
k = 9/ f-score: 0.6591752440374267
average precision:  0.7665867459455068
average recall:  0.7672193415356887
average f-score:  0.6694976257896905
```

Question 7: Comparing the different clusterings you obtained in (3)-(6), discuss in which setting you obtained best clustering for this dataset.

To begin with, the overall B-cubed score of k-means was higher than k-medians. For example, the average f-score was around 0.7 and 0.65 for k-means and k-medians respectively. Regarding the trend of the B-cubed score, all scores (precision, recall, and f-score) of each graph fluctuated along with the k values with the peak of those scores around k = 4. However, by applying L2 normalization, the overall precision, recall, and f-score were improved. For example, when L2 normalization was implemented, the f-score of k-means and k-medians increased from 0.68 to 0.74 and 0.64 to 0.67 respectively. Finally, according to the results, the best models were k-means with L2 normalization at k = 4 and k-medians with L2 normalization at k = 4 because all precision, recall, and f-score of them were equally around 0.93 which was the highest value compared to other configurations.