K Means Assessment

201374125

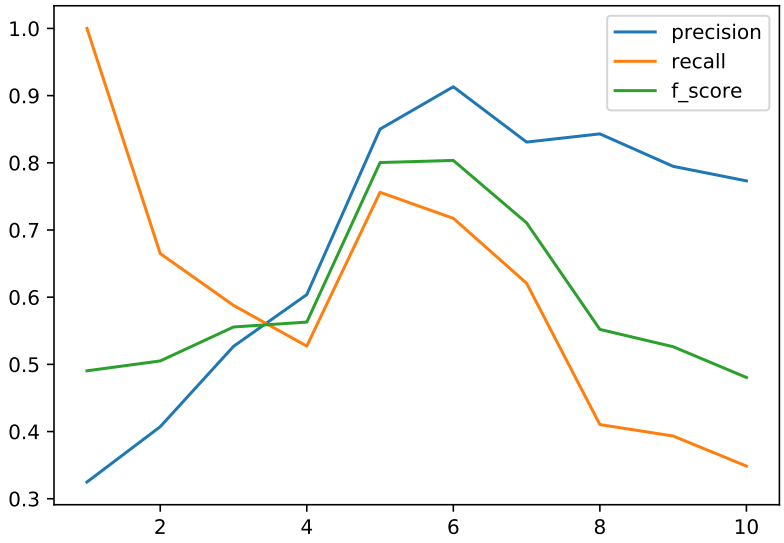
Task 1

See `kmeans.py`.

Task 2

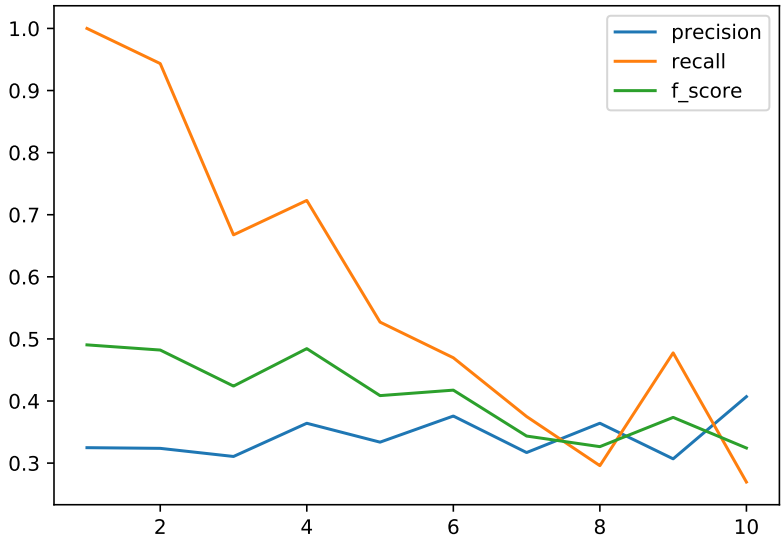
```
import kmeans
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib

matplotlib.rcParams['font.sans-serif'] = ['XCharter', 'sans-serif']



cats = ['animals', 'countries', 'fruits', 'veggies']
df = kmeans.read_data(cats)
```

```
score = {}
for k in range(1, 11):
    km = kmeans.K_Means(data=df, k=k)
    word_clusters = km.fit()
    score[k] = {'precision': km.precision,
                'recall': km.recall,
                'f_score': km.f_score}
score = pd.DataFrame(score).T
score.plot()
```



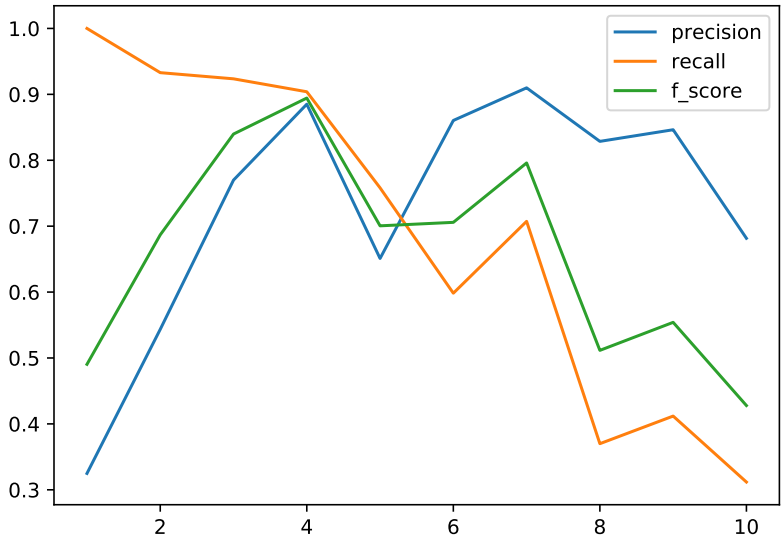
Task 3

```
score = {}
for k in range(1, 11):
    km = kmeans.K_Means(data=df, k=k, normalised=True)
    word_clusters = km.fit()
    score[k] = {'precision': km.precision,
                'recall': km.recall,
                'f_score': km.f_score}
score = pd.DataFrame(score).T
score.plot()
```



Task 4

```
score = {}
for k in range(1, 11):
    km = kmeans.K_Means(data=df, k=k, distance='manhattan_distance')
    word_clusters = km.fit()
    score[k] = {'precision': km.precision,
                'recall': km.recall,
                'f_score': km.f_score}
score = pd.DataFrame(score).T
score.plot()
```

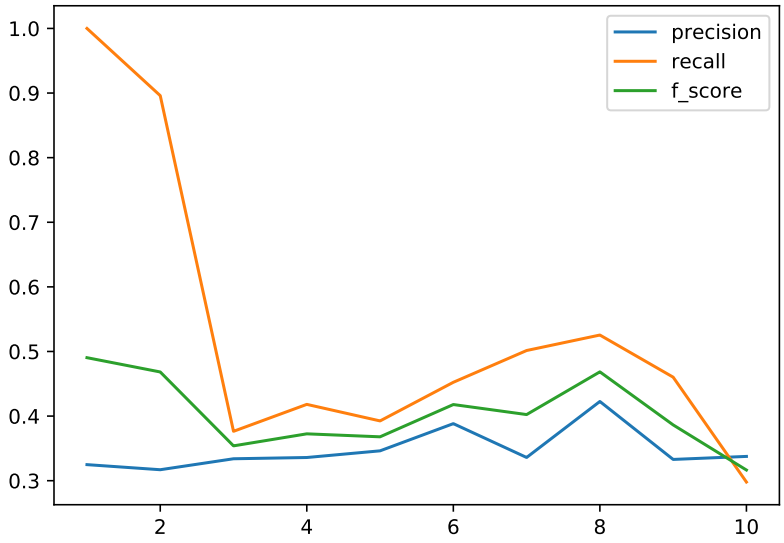


Task 5

```
score = {}
for k in range(1, 11):
    km = kmeans.K_Means(data=df,
                  k=k,
                  distance='manhattan_distance',
                  normalised=True)
    word_clusters = km.fit()
    score[k] = {'precision': km.precision,
                'recall': km.recall,
                'f_score': km.f_score}
```

```
//usr/lib/python3.8/site-packages/numpy/core/fromnumeric.py:3334: RuntimeWarning: Mean of empty slice.
  return _methods._mean(a, axis=axis, dtype=dtype,
//usr/lib/python3.8/site-packages/numpy/core/_methods.py:153: RuntimeWarning: invalid value encountered in
  ret = um.true_divide(
```

```
score = pd.DataFrame(score).T
score.plot()
```

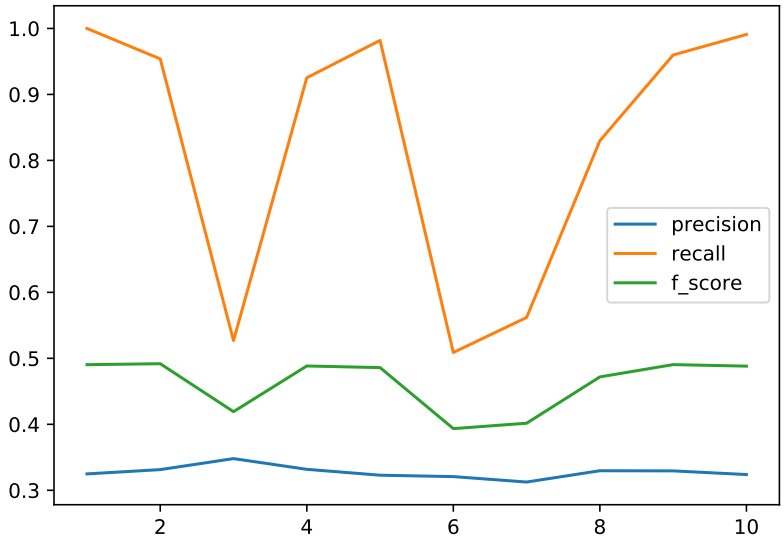


Task 6

```
score = {}
for k in range(1, 11):
    km = kmeans.K_Means(data=df,
                  k=k,
                  distance='cosine_distance',
                  normalised=True)
    word_clusters = km.fit()
    score[k] = {'precision': km.precision,
                'recall': km.recall,
                'f_score': km.f_score}
```

```
//usr/lib/python3.8/site-packages/numpy/core/fromnumeric.py:3334: RuntimeWarning: Mean of empty slice.
  return _methods._mean(a, axis=axis, dtype=dtype,
//usr/lib/python3.8/site-packages/numpy/core/_methods.py:153: RuntimeWarning: invalid value encountered i
  ret = um.true_divide(
```

```
score = pd.DataFrame(score).T
score.plot()
```



Task 7

The best result is given by the highest F Score, this appears to be the unnormalised Manhattan distance with 4 clusters from task 4. However, it should be noted that without a set seed, these results vary wildly.