Worksheet 05

Name:
UID:

Topics

- Cost Functions
- Kmeans

Cost Function

Solving Data Science problems often starts by defining a metric with which to evaluate solutions were you able to find some. This metric is called a cost function. Data Science then backtracks and tries to find a process / algorithm to find solutions that can optimize for that cost function.

For example suppose you are asked to cluster three points A, B, C into two non-empty clusters. If someone gave you the solution `{A, B}, {C}`, how would you evaluate that this is a good solution?

Notice that because the clusters need to be non-empty and all points must be assigned to a cluster, it must be that two of the three points will be together in one cluster and the third will be alone in the other cluster.

In the above solution, if A and B are closer than A and C, and B and C, then this is a good solution. The smaller the distance between the two points in the same cluster (here A and B), the better the solution. So we can define our cost function to be that distance (between A and B here)!

The algorithm / process would involve clustering together the two closest points and put the third in its own cluster. This process optimizes for that cost function because no other pair of points could have a lower distance (although it could equal it).

K means

a) (1-dimensional clustering) Walk through Lloyd's algorithm step by step on the following dataset:

`[0, .5, 1.5, 2, 6, 6.5, 7]` (note: each of these are 1-dimensional data points)

Given the initial centroids:

`[0, 2]`

Find clusters: [[0, 0.5], [1.5, 2, 6, 6.5, 7]]
Recalculate centroids: [0.25, 4.6]
Find clusters: [[0, 0.5, 1.5, 2], [6, 6.5, 7]]
Recalculate centroids: [1, 6.5]
Find clusters: [[0, 0.5, 1.5, 2], [6, 6.5, 7]] --> convergence

Final clusters: [[0, 0.5, 1.5, 2], [6, 6.5, 7]]
Final centroids: [1.5, 6.5]

b) Describe in plain english what the cost function for k means is.

It is a way to evaluate and compare solutions, in the hope that we can find some algorithm that finds solutions that make the cost small.

c) For the same number of clusters K, why could there be very different solutions to the K means algorithm on a given dataset?

The solution depends on the initial centroids. Since the initial centroids can vary, the algorithm can converge on different solutions.

d) Does Lloyd's Algorithm always converge? Why / why not?

Yes. It always converges because there is a finite number of clusters, and a finite number of datasets.

e) Follow along in class the implementation of Kmeans

```python
import numpy as np
from PIL import Image as im
import matplotlib.pyplot as plt
import sklearn.datasets as datasets

centers = [[0, 0], [2, 2], [-3, 2], [2, -4]]
X, _ = datasets.make_blobs(n_samples=300, centers=centers,
cluster_std=1, random_state=0)

class KMeans():

    def __init__(self, data, k):
        self.data = data
        self.k = k
        self.assignment = [-1 for _ in range(len(data))]
        self.snaps = []

    def snap(self, centers):
        TEMPFILE = "temp.png"

        fig, ax = plt.subplots()
        ax.scatter(X[:, 0], X[:, 1], c=self.assignment)
        ax.scatter(centers[:,0], centers[:, 1], c='r')
        fig.savefig(TEMPFILE)
        plt.close()
        self.snaps.append(im.fromarray(np.asarray(im.open(TEMPFILE))))


    def lloyds(self):
        ...
        return
```

```python
kmeans = KMeans(X, 6)
kmeans.lloyds()
images = kmeans.snaps

images[0].save(
    'kmeans.gif',
    optimize=False,
    save_all=True,
    append_images=images[1:],
    loop=0,
    duration=500
)
```