Data Mining

Homework 3

Due: 3/12/2023, 23:59

Instructions

You must hand in the homework electronically and before the due date and time.

This homework has to be done by each **person individually**.

Handing in: You must hand in the homework by the due date and time by an email to Gianluca (`decarlo@diag.uniroma1.it`) that will contain as attachment **(not links to some file-uploading server!)** a .zip file with your answers. The filename of the attachment should be `DM_Homework_3__StudentID_StudentName_StudentLastname.zip`;
for example:
`DM_Homework_3__1235711_Robert_Anthony_De_Niro.zip`.
The email subject should be
`[Data Mining] Homework_3StudentID StudentName StudentLastname`;
For example:
`[Data Mining] Homework_31235711 Robert Anthony De Niro`.
After you submit, you will receive an acknowledgement email that your project has been received and at what date and time. If you have not received an acknowledgement email within 2 days after you submit, then contact Gianluca.

The solutions for the theoretical exercises must contain your answers either typed up or hand written clearly and scanned.

For information about collaboration, and about being late check the web page.

Problem 1. We will now study some questions of k-means on 1 dimension.

1. Recall that in the k-means problem we want to minimize the total squared ℓ_2 distance between each point and the center to which it is assigned to:

$$\sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2,$$

where C_i is the set of points that belong to the ith cluster, $\boldsymbol{\mu}_i$ the mean of the points in the ith cluster, and

$$\|\mathbf{x}\|^2 = \sum_{j=1}^{d} x_j^2,$$

if $\mathbf{x} = (x_1, x_2, \ldots, x_d)$.

In class, we said that in general the k-means problem is NP-hard. However, for $d = 1$ the problem is polynomial. Design an algorithm that solves the k-means problem in time polynomial in the number of points n and the number of clusters k, for $d = 1$.

(**Hint:** Can you solve the problem for k clusters if you assume that you can solve it for fewer than k clusters?)

2. We are given a set P of n points in $\mathbb{R}$. For simplicity, assume that $\mu(P) = 0$, that is, $\sum_{x \in P} x = 0$. Let $\|P\|^2 = \sum_{x \in P} x^2$ be the optimal 1-means cost. Show that by adding carefully $O(1/\epsilon)$ centers, we can make the k-means cost at most $\epsilon \cdot \sum_{x \in P} x^2$.

Hint: First show that by adding 2 centers at locations $-\ell$ and ℓ, for an appropriate value of ℓ, the cost decreases by a factor of 3/4.

Problem 2.

Benchmark Data Set Consider an instance generated as follows. Let I_k be the $k \times k$ identity matrix. Let $\mathcal{N}^{k,d}(0,\sigma^2)$ be the $k \times d$ matrix in which every entry is a Gaussian random variable with mean 0 and variance σ^2. We now consider the instance A obtained by stacking n copies of the matrix I_k $\mathcal{N}(0,\sigma^2)^{k,d}$, that is,

$$A = \begin{bmatrix} I_k & \mathcal{N}^{k,d}(0,\sigma^2) \\ I_k & \mathcal{N}^{k,d}(0,\sigma^2) \\ \vdots & \\ I_k & \mathcal{N}^{k,d}(0,\sigma^2) \end{bmatrix}$$

A has $k \cdot n$ points and $k + d$ dimensions. The "correct" clustering of the rows of A is to simply put A_i into cluster $i \bmod k$, that is, the matrix I_k is also an indicator matrix for cluster membership and the correct clustering matrix satisfies

$$X_{i,j} = \begin{cases} \frac{1}{\sqrt{n}} & \text{if } i \bmod k = j + 1 \\ 0 & \text{otherwise} \end{cases}.$$

To set n, k, d, and σ^2, we suggest the following values (experiment with them).

- $k = 50, 100, 200$
- $n = 1.000, 10.000, 100.000$ (go to a million if your computer is too good)
- $d = k, 100k, 100k^2$
- $\sigma^2 = 1/k, 1/\sqrt{k}, 0.5$

To generate this data set, we recommend the loosely python inspired pseudocode (do **not** copy/paste, check for errors):

```
import numpy as np
import math
from random import gauss
set n,k,d and σ as necessary
data = np.ndarray(shape=(k*n, d+k), dtype=float, order='F')
for i in range(k*n): do
  for j in range(d): do
    data[i][k+j] = gauss(0,σ)
  end for
  for j in range(k): do
    if i%k == l: then
      data[i][j] = 1
    else
      data[i][j] = 0
    end if
  end for
end for
```

Algorithms We have seen the following two algorithms in class:

PCA: project the data onto the first m principal components. Recall that $m \geq k$. If we set m close to k we remove more noise. If $m \approx k/\varepsilon$, we remove less noise but have more accuracy wrt the k-means objective.

```
import numpy as np
set m as necessary
u,s,vh = np.linalg.svd(data, full_matrices=True)
smat = np.zeros((k*n, d+k), dtype=float)
smat[:s.size, :s.size] = np.diag(s)
for  i in range(0,s.size): do
  if i > m-1: then
    s[i] = 0
  end if
end for
smat[:d+k, :d+k] = np.diag(s)
projected = np.dot(u, np.dot(smat, vh))
```

k-means++:

```
C={random input point}
for i=1 to k-1 do
  pick point p proportionate to min_{c∈C} ||p − c||^2
  add p to C
end for
```

Tasks Experiment with a data analysis chain. In particular, try the following two options:

1. Run k-means++ on the data set.
2. Apply PCA and then apply k-means++.

For all variants, consider whether the algorithm was able to recover the ground clustering. Try to explain why certain combinations were more successful than others.

Finally, consider the running time of the various steps. Which parts were the most expensive parts? Did this behavior change, depending on the order in which the algorithms were executed? Remember, you can log the running time using:

```
from datetime import datetime
tStart = datetime.now()
run Algorithm
tEnd = datetime.now()
```

Problem 3. For this exercise you will perform some clustering and use some feature engineering techniques. The lecture by Pablo Duboue[1] gives us a lot of interesting insights on the topic of

[1] http://duboue.net/

feature engineering. It is a procedure that is sometimes underestimated but that, as you saw, can be really game changing in machine learning applications; the correct way to preprocess and model the features can increase the accuracy of many ML models. For this exercise we will use the dataset collected in the previous homework from the amazon.it website. It presents different kinds of features, a fact that strongly suggests us to use some engineering techniques to better work with such data. What you have to do is the following:

- Choose one clustering algorithm (k-means++, DBSCAN, etc.) to cluster the samples in the dataset without any feature engineering technique applied, just on raw data. You can use library implementations. Use the elbow method to find the most suitable number of clusters. You are also free to use a different method to find the optimal number of clusters, if you want (like Silhouette, etc).

- Inspect and understand your data and choose the best feature engineering technique (or techniques) that you would apply. Do whatever you think may be necessary (normalization, scaling, one-hot encoding and more). After that, perform the clustering again using the same clustering method used with the raw data.

Do you see any differences in the clusters obtained? What about running times? Make a plot of the clusters in both cases and comment what you see. Did your feature engineering help in creating more distinct clusters? Did it help with the shape of the elbow curve (or with the different method chosen)? Write a very short report (max 3 pages) in which you describe the clustering algorithm employed and the feature engineering techniques used (and why you think they were the proper techniques to use) along with the plots, comments and any observation on the results obtained that you think to be important. Hand in the report along with the code and instructions to run it.