APM120, Homework #7
Applied Linear Algebra and Big Data
Last updated: Saturday 4th March, 2023, 10:06
Assigned March 21, due March 28, 1:00 pm, via gradescope, **in pdf $\leq$ 20Mb, $\leq$ 30 pages.**
Unsupervised learning: cluster analysis, part I

Show all steps in all calculations explicitly. Attach code used, well documented, and relevant plots and Matlab/python output, attaching code and figures *immediately following* the relevant question solution. A code printout is not a substitute for complete solutions, your solution should stand alone without the Matlab/python code or output. See needed python preliminaries at end of this HW[1]. It is fine to use Matlab/python unless a hand calculation (using only a simple calculator) is required *in orange.* For all questions, make sure you show all steps as if you are doing the problem by hand, and do not use library functions unless explicitly allowed in the question. Make sure you can do all calculations using no more than a hand-held calculator. See the end-note for guidelines and examples of hand-calculations.[2]

1. **Distance metrics:** Let `s1='takeasadsong; s2='makeitbetter;`
`x=[6,-4,-2]; y=[53,-42,-13];` calculate the following distances,
   (a) Hand-calculate the Edit and Jaccard distances between s_1 and s_2; Hamming distance between the first n elements of the two strings, where n is the length of the shorter one.
   (b) Hand-calculate the Cosine, Euclidean (L_2), Manhattan (L_1) and L_∞ distances between $\mathbf{x}$ and $\mathbf{y}$.
   (c) How would you define a circle in the L_n norm? Sketch a circle of radius 2 in the L_1 norm and in L_∞.

2. **Curse of dimensionality: (A)** Numerically examine the distribution of distances of random data as a function of dimension, for the L_3 norm, you may want to start from `curse_of_dimensionality.m/py` on the course web page. Compare to the L_2 case and Discuss your results. **(B)** Explain why one expects random pairs of vectors in the unit cube (that is, with components in the range of $(-1, 1)$) at high dimensions to be rarely close to one another, in the sense of their L_1 distances. **(C) ***Optional extra credit:** can one also examine the distribution of *angles* in a vector space defined with an L_p norm? Explain. (Google. . .)

3. **Hierarchical clustering: (A)** Perform a step-by-step hierarchical clustering of the following set of two-dimensional data points, merging all the way to a single cluster. Show all steps and calculations explicitly. Merge clusters such that the resulting cluster has the *smallest increase in cluster variance* (Ward method, see course notes). Perform a hand-calculation of the first two mergings. Plot the dendrogram plot by hand. What do the vertical distances represent in the dendrogram plot? Discuss the best choice for the number of clusters.

```
X=[ 5,  4.5,  6,  9,  9.5,  4,  2,  2.5,  5;
    9.5,  5,  9,  6,  4.5,  4,  5,  3,  8];
```

Hint: It is helpful to first plot the points using:

```
% Matlab
figure(1);clf; scatter(X(1,:),X(2,:));
box on; hold on;xlabel('x');ylabel('y');
for ii=1:length(X(1,:));
text(X(1,ii)+0.07,X(2,ii),num2str(ii)
  ,'FontSize',20);end
```

```
# python:
plt.figure(1); plt.clf()
plt.scatter(X[0,:],X[1,:])
plt.xlabel('x');plt.ylabel('y')
for ii in range(0,np.size(X[0,:])):
   plt.text(X[0,ii]+0.07,X[1,ii],str(ii))
```

and use the plot to decide what are the likely clusters to merge at each point. You only need to calculate the merging criterion for the most likely cluster merges. Circle each clustering as in Figure 6.2 in the course notes. **(B)** Use Matlab/python to perform the same Hierarchical clustering using library routines, changing the linkage command below to correspond to the above-specified merging strategy. Start from the following, explain what each Code line does.

```
% Matlab:
Y=pdist(X');
distances=squareform(Y),
Z=linkage(Y,'centroid');
dendrogram(Z);
k=3;
idx=cluster(Z,'maxclust',k)'
```

```
# python:
from scipy.cluster.hierarchy import dendrogram,linkage
from scipy.cluster.hierarchy import fcluster
from scipy.spatial.distance import pdist,squareform
Y=pdist(X.T,'euclid'); distances=squareform(Y)
Z=linkage(X.T,method='centroid'); dendrogram(Z)
k=3; idx=fcluster(Z,k,'maxclust')
```

(C) Present and explain the output variables `distances`, `Z` and `idx`, and the dendogram plot. Compare the results of this clustering to your own analysis.

4. **K-means algorithms:** **(A)** Find the clusteroids for the above data for the k-means algorithm for $k = 2, 3, 4$, by choosing the initial clusteroid point to be the fourth data point, and then choosing the following $k - 1$ initial points to maximize the smallest distance to previous initial clusteroids. **(B)** Cluster the above data using a hand-calculation using k-means for $k = 3$. **(C)** Perform k-means clustering for $k = 2, 3, 4, 5$ using Matlab/python,

```
% Matlab:
k=3;
opts = statset('Display','final');
[idx,C] = kmeans(X',k, ...
  'Distance','sqeuclidean' ...
  ,'Replicates',5,'Options',opts);
disp("idx=");disp(idx)
```

```
# python:
from sklearn.cluster import KMeans;
k=3;
kmeans=KMeans(n_clusters=k,n_init=5 \
   ,copy_x=True).fit(X.T);
idx=kmeans.labels_; print("idx="); print(idx);
```

(D) Find the cluster diameter for each cluster and for each of the k values, choose the appropriate value of k based on the *cluster radius*, and explain. **(E)** Compare your k-means results for $k = 3$ to those of the Hierarchical clustering method applied above to the same data set and briefly discuss your conclusions from this comparison.

* [What's the point of *****optional extra credit** challenge problems: apart from the fun of doing them, they may bring the total score of this HW assignment up to 110%, making up for problems you may have missed in this or other HW assignments…]

Python preliminaries & notes

1 python commands within the HW assume you have first used the followings: `import numpy as np; from numpy import linalg; import scipy as scipy; from scipy import linalg; import matplotlib.pyplot as plt; import matplotlib;`
Input a matrix A, column vector **b** and row vector **c** into python in the form
`A=np.array([[a11,a12,a13],[a21,a22,a23],[a31,a32,a33]]; b=np.array([[b1],[b2],[b3]]); c=np.array([[c1,c2,c3]]);` or convert Matlab arrays given in HW directly to python arrays using, e.g., `A=np.array(np.matrixlib.defmatrix.matrix('[1 2 3; 4 5 6]'));`

2 **Hand calculations**, in which you are asked to use only a simple hand-held calculator, are required only when we want to make sure you understand exactly what each step of an algorithm is doing. These also prepare you for the quizzes that involve similar hand calculations. **How much work to show?** Just don't use scratch paper, show all steps that you actually use for the hand calculations, but no more, carrying out calculations to **three significant digits**. Trust the graders to be reasonable, they were students in the course last year. **Examples: (i)** If you are asked, *not* in orange, to calculate the LU decomposition of a matrix, you may use Matlab/python as in `A(2,:)=A(2,:)-A(1,:)*(A(2,1)/A(1,1))` etc, but you may not use a library routine as in `[L,U,P]=lu(A)` except for checking your results. **(ii)** If required to calculate by hand the element $C_{2,3}$ of a matrix product $\mathsf{C} = \mathsf{AB}$, you need to explicitly multiply using a hand calculator the second row of A with the third column of B. You may not use Matlab/python to calculate `C=A*B` and then take the needed element from that product, nor to calculate `C23=A(2,:)*B(:,3)`.