Istanbul Technical University
Mobile Communications
Machine Learning Homework
Nov, 22, 2018

Raw Data:

The raw data (https://www.dropbox.com/s/k55c7wlylwgth32/MobileCommData.7z?dl=0) includes sensor measurements as a function of time with 1s frequency.

Overall columns include 401 brands and 5 additional information on customers. The structure of the data is summarized in Table 1.

Table 1: The transaction numbers and the demographic information on customers

C0	C1	C2	C3	C4	C5	C6	C7	...	C401
t=1									
t=2									
…									
t=N									

The columns with indices `262, 81, 82, 80, 84` are outputs to be predicted by machine learning.
Objective:

1) Use unsupervised learning algorithms to determine:
- The number of different processing regimes
- The number of clusters for columnsç
- The hierarchical clusters.

Demonstrate the elements in the clusters (i.e. using histograms) and explain the main characteristics of the elements within the cluster.

Show the impact of:
- The number of clusters
- The algorithm of the cluster

Use codes from:
https://www.mathworks.com/help/stats/kmeans.html
https://www.mathworks.com/help/stats/dendrogram.html

Kmeans is a function which incorporates several clustering methods. A typical usage is given below:

```
[idx,C] = kmeans(X,NumCluster,'Distance','DistFunc');
```

where X is the data matrix with N customers and M brands (NxM); NumCluster is the number of clusters; DistFunc is the distance function which affect the clustering; idx is the index vector of dimension N with the cluster index of sample data.

The following sample code plots the 2-dimensional X data and colors with the corresponding cluster index. Plotting X is not possible when the dimension is high.

```
figure;
plot(X(idx==1,1),X(idx==1,2),'r.','MarkerSize',12)
```

```
hold on
plot(X(idx==2,1),X(idx==2,2),'b.','MarkerSize',12)
plot(C(:,1),C(:,2),'kx',...
        'MarkerSize',15,'LineWidth',3)
legend('Cluster 1','Cluster 2','Centroids',...
           'Location','NW')
title 'Cluster Assignments and Centroids'
hold off
```

The following code plots the histogram to show the number of elements within a particular cluster:

```
hist(idx,1:NumCluster)
```

Later on you should calculate the characteristics of clusters (i.e high transaction customers) and evaluate why they end up in the same cluster. You should also do this with different distance functions and determine the optimal algorithm for clustering at different NumClusters.

Once you transpose the data matrix, X, the clustering focuses on the brands. Do the same and show which brands are in the same cluster.

Determine the impact of NumCluster on the computation speed.

In order to obtain a dendogram plot use the following:

```
tree = linkage(X,'average');
dendrogram(tree)
```

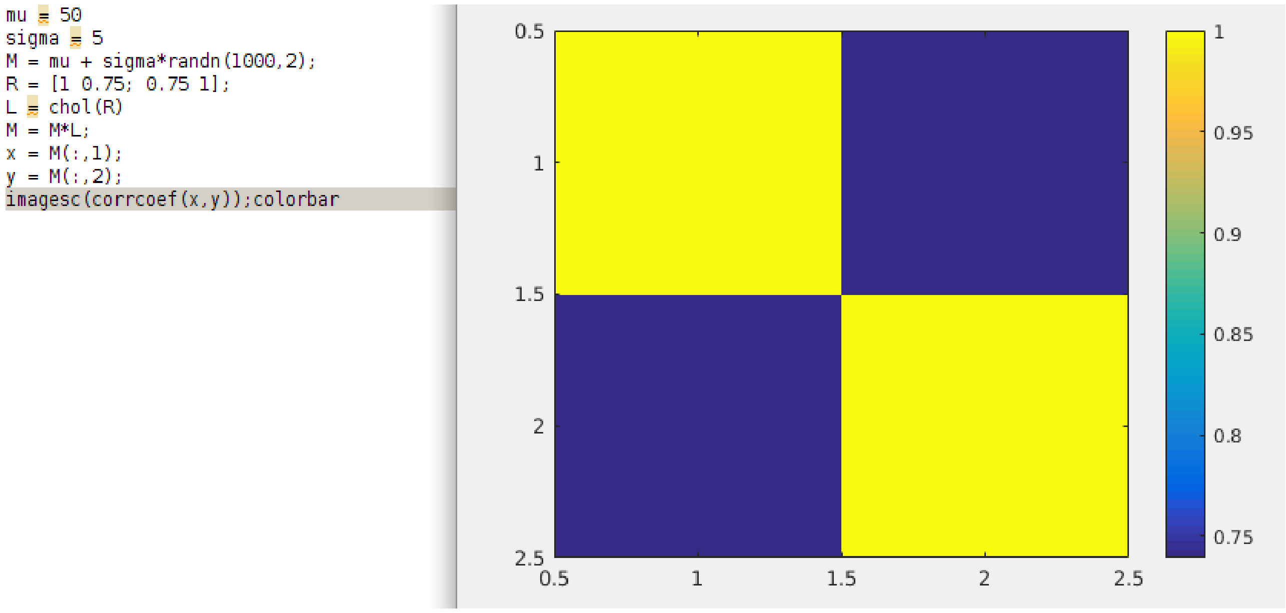
Where X is the data matrix. Note that you will have to take the transpose of the original data matrix to avoid a memory error. The dendrogram can only be calculated for brands since the computation is challenging due to its nature.

On the dendrogram, demonstrate which brands are similar. Focus on the vertical line distances. Show the impact of different algorithms on the dendrogram. Try to determine the optimal algorithm.

2) Calculate the correlations between columns of the row data (i.e. determine if there is a correlation between variables). Determine which columns are important and needs to be used in the machine learning. The codes you will need is here:

https://www.mathworks.com/help/matlab/ref/corrcoef.html
https://www.mathworks.com/help/matlab/ref/cov.html
https://www.mathworks.com/help/matlab/ref/imagesc.html

The following code results in the following figure:



```
mu = 50
sigma = 5
M = mu + sigma*randn(1000,2);
R = [1 0.75; 0.75 1];
L = chol(R)
M = M*L;
x = M(:,1);
y = M(:,2);
imagesc(corrcoef(x,y));colorbar
```

Thus, it show the correlation between x and y on an image. Try to eliminate high correlation resulting variables for the machine learning purposes. How many and which brand should be satisfactory for the machine learning?

3)Use supervised learning algorithms to construct a neural network (NN) architecture to exploit the interactions between inputs and outputs.

1) Divide the data into training and test samples. How do you determine the number of samples?
2) Fit NN to the data and show the prediction accuracy.
3) Show the impact of neuron size in the prediction accuracy?
4) Determine the maximum number of neurons which improve the prediction performance. Why any more increase does not bring additional accuracy?
5) Change the training algorithm and show the impact on accuracy and computation time.
6) Determine which inputs and outputs have no impact on the overall data. Do fitting and observe the outputs after data elimination.

Hint:

1) Go through examples:
https://www.mathworks.com/help/nnet/examples/wine-classification.html
https://www.mathworks.com/examples/neural-network/mw/nnet-ex97900244-cancer-detection

2) Here is a code to train a neural network from the editor:

```
net = patternnet(NumNeurons);
[net,tr] = train(net,NN_inputs',NN_outputs');
```

Where NumNeurons is the neuron number in the NN. NN_inputs and NN_outputs are the input and output matrices, respectively.

Once the training is accomplished, the prediction for a particular input data can be obtained from:

```
prediction = net(inputs);
```

HINT:

An example for the implementation of classification problems in python is given below:

```
from sklearn.neural_network import MLPClassifier
from numpy import *
```

```
num_data=1000

circle_rad=0.6

inputs=random.rand(num_data,2)*2-1 # generate data between -1 and 1
outputs=zeros((num_data,1)) # initialize output vector

for i in range(0,len(inputs)):
    if (inputs[i,0]**2+inputs[i,1]**2<circle_rad**2): # check if the number is inside circle
        outputs[i]=1

X = inputs
y = outputs

clf = MLPClassifier(solver='lbfgs', alpha=1e-5,hidden_layer_sizes=(5, 2), random_state=1)

clf.fit(X, y)

import matplotlib.pyplot as plt
plt.figure(figsize=(10,10))
plt.subplot(1, 1, 1)
for i in range(0,len(inputs)):
    if (inputs[i,0]**2+inputs[i,1]**2<circle_rad**2):
        plt.plot(inputs[i,0],inputs[i,1],'ro')
    else:
        plt.plot(inputs[i,0],inputs[i,1],'bo')

plt.show()
```