# Informative Versus Uninformative Variables

In our last session, I showed how it is possible to identify informative variables, if they exist, by computing the cross correlations between variables over a set of 10 data points. These 10 data points are organized into two clusters, one with 5 data points, and another with 5 data points.

In this session, let us examine this process more carefully, by using 20 data points for one cluster, and 30 data points for the other cluster. Also, we want 10 informative variables, and 40 uninformative ones.

In the Jupyter QtConsole, we first use

import numpy as np

import matplotlib.pyplot as plt

to import numpy and matplotlib.

Then, we use

X = np.zeros((50,50))

to initiate our array of data points. Rows in X refer to the data points, while columns in X refer to the variables. The first 20 rows of X are in one cluster, while the last 30 rows of X are in the other cluster. The first 10 variables are informative, while the next 40 variables are not.

To generate data points for the first 10 informative variables, we need the centres of the two clusters in these 10 variables.

X1 = np.random.randint(2,5,10)

X1

array([2, 2, 3, 2, 2, 3, 3, 3, 4, 2])

X2 = np.random.randint(-6,-1,10)

X2

array([-4, -2, -3, -6, -4, -5, -3, -2, -6, -6])

Then for each X1, we generate 20 normally-distributed data points, and for each X2, we generate 30 normally-distributed data points.

for k in range(10):

for n in range(20):

X[n,k] = X1[k] + 0.5\*np.random.randn()

for n in range(20,50):

X[n,k] = X2[k] + 0.4\*np.random.randn()

We then fill out the remaining 40 uninformative variables by:

for k in range(10,50):

for n in range(50):

X[n,k] = 5\*np.random.randn()

We check that we have done this correctly, by plotting a color map of X:

plt.matshow(X)

plt.show()

to get the following figure:



If we compute the pairwise distances between the first 20 data points, using only the first 10 variables, we find that the distances are small, and less than 3.

D = np.zeros((20,20))

for m in range(20):

for n in range(20):

D[m,n] = np.linalg.norm(X[m,:10] - X[n,:10])

Similarly, if we compute the pairwise distances between the next 30 data points, using the first 10 variables, we find that the distances are small, and less than 3.

D = np.zeros((30,30))

for m in range(30):

for n in range(30):

D[m,n] = np.linalg.norm(X[20+m,:10] - X[20+n,:10])

If we compute the pairwise distances of all 50 points, using only the first 10 variables,

D = np.zeros((50,50))

for m in range(50):

for n in range(50):

D[m,n] = np.linalg.norm(X[m,:10] - X[n,:10])

plt.matshow(D)

plt.colorbar()

plt.show()

to get



where we can clearly distinguish between the two clusters.

However, if we use all 50 variables,

D = np.zeros((50,50))

for m in range(50):

for n in range(50):

D[m,n] = np.linalg.norm(X[m,:10] - X[n,:10])

plt.matshow(D)

plt.colorbar()

plt.show()

to get



Here, based on pairwise distances alone, we cannot tell that there are two clusters of data points.

To identify the two clusters of data points, we first have to identify the informative variables. This can be done by computing the cross correlation between the variables. We can do so using:

C = np.corrcoef(np.transpose(X))

plt.matshow(C)

plt.show()

to obtain the color map shown below:

## Regression vs Correlation

Correlation does not imply causal relations.

We represent causal relations through regressions.

Suppose X[m, k] = coordinate of the mth data point for the kth variable, and X[n, k] = coordinate of the nth data point for the kth variable.

If |X[m, k] – X[n, k]| is small, then m and n are in the same cluster, what then can I predict?

I will predict that |X[m, l] – X[n, l]| is also small, if l is a informative variable.

Two way street: if |X[m, k] – X[n, k]| is small predicts that |X[m, l] – X[n, l]| is also small, then |X[m, k] – X[p, k]| can also be small, if data point p is in the same cluster as m.

Fix the variable k, but vary the data points m, n, p, q, …

We say that variable k is predictive (or informative), if I know that |X[m, k] – X[n, k]| is small, and |X[m, k] – X[p, k]| is also small, then |X[n, k] – X[p, k]| is small.

If plot a graph of height vs age. Have this for many people. Then if I know your age now, and your height now, I might be able to predict your height 5 years later.

y = mx + c

Variable 8 has prediction power, but variable 28 does not.



Here, we can see