Neelanjan Goswami

Student ID: 400414867

Assignment 1 – Mean, Covariance of Multivariate Data

1) Using the Excel file dataA.xlsx, which contains a 500x3 data matrix (500 data points with 3 attributes), calculate both the mean and the covariance matrix.

```
import numpy as np
import pandas as pd

dataA = pd.read_excel('dataA.xlsx', header = None)
data = pd.DataFrame.to_numpy(dataA)

shape = data.shape

data
```

```
array([[ 1.55689089,  4.01059603,  1.29323385],
       [-1.39048821,  3.24872268, -1.01189758],
       [ 0.92749172,  2.11432065, -2.21430133],
       ...,
       [ 3.09422969,  4.06386355,  1.72786312],
       [-0.78509245, -1.46375948,  2.68672626],
       [ 2.14907545,  2.35843938,  2.11973336]])
```

```
# Mean
data_mean = np.mean(data, axis = 0)
print('The mean of the dataA.xlsx data:\n')
print(data_mean)
```

```
The mean of the dataA.xlsx data:

[0.34750193 1.02563712 0.80122132]
```

```
data_m = data - data_mean
covariance = np.dot(np.transpose(data_m),(data_m))/(shape[0])
print('The covariance of the dataA.xlsx data:\n')
print(covariance)

# Covariance
data_m = data - data_mean
covariance = np.dot(np.transpose(data_m),(data_m))/(shape[0])
print('The covariance of the dataA.xlsx data:\n')
print(covariance)
```

Neelanjan Goswami

400414867

```
The covariance of the dataA.xlsx data:

[[4.06234772 0.1499012  0.26155949]
 [0.1499012  2.55794521 0.01465669]
 [0.26155949 0.01465669 3.176846  ]]
```

2) Using the Excel file dataB.xlsx, which contains a 500x10 data matrix (500 data points with 10 attributes), calculate both the mean and the covariance matrix.

```
import numpy as np
import pandas as pd
dataB = pd.read_excel ('dataB.xlsx', header = None)
data2 = pd.DataFrame.to_numpy(dataB)
```

data2

```
array([[  8.65459847,   4.5620827 ,   1.38692383, ...,   4.97059994,
         -5.21064875,   4.00363613],
       [  8.73913433,   7.14120061,   7.8046532 , ...,  10.05300533,
         -4.23675514,   0.65769703],
       [ 11.83876324,   8.93293461,   8.75195321, ...,   7.62535922,
        -10.91127599,  -0.41110291],
       ...,
       [ 13.89179674,   4.14493518,   7.03398343, ...,   5.50311762,
         -0.63970448,   0.72106266],
       [ 10.84950119,   2.59352958,   3.84345036, ...,   3.09446443,
         -1.83492417,   1.56929856],
       [  8.42435559,   9.80180673,  10.04606526, ...,   6.15884425,
         -1.12177575,   6.53539108]])
```

```
# Mean
data_mean = np.mean(data2, axis = 0)
print('The mean of the dataB.xlsx data:\n')
print(data_mean)
```

```
The mean of the dataB.xlsx data:

[9.57062029 6.15014874 8.08016477 9.55989208 8.8040749  2.19491256
 0.20634971 4.54942571 0.06659806 4.65575632]
```

```
data_m = data2 - data_mean
covariance = np.dot(np.transpose(data_m),(data_m))/(shape[0])
print('The covariance of the dataB.xlsx data:\n')
print(covariance)

#Covariance Matrix
```

Neelanjan Goswami

400414867

```
The covariance of the dataB.xlsx data:

[[ 9.55495678e+00  1.57110665e-01  6.89623982e-01 -4.30708218e-02
  -1.54984822e-01  1.12708273e+00  2.63934697e-02 -4.85572927e-01
   9.54450984e-01  5.32211658e-01]
 [ 1.57110665e-01  9.49737298e+00  4.76615526e-01  4.12508342e-01
   5.56261025e-03  5.30880675e-01  1.10779531e-01  1.69990669e-01
   8.39373125e-01  1.33647214e+00]
 [ 6.89623982e-01  4.76615526e-01  8.64010504e+00 -3.10998204e-01
   1.65235043e-01  1.91868055e-01  1.85483343e-01  4.09352351e-01
   2.28436981e-01 -1.53964737e-01]
 [-4.30708218e-02  4.12508342e-01 -3.10998204e-01  1.02499863e+01
   2.50503194e-01  3.48114355e-01  6.88545857e-01  3.21912898e-01
   7.21089201e-01  1.04788892e+00]
 [-1.54984822e-01  5.56261025e-03  1.65235043e-01  2.50503194e-01
   9.63187327e+00  6.49585344e-01  1.52340162e-01 -1.65722441e-01
   1.35517125e+00 -1.97654085e-01]
 [ 1.12708273e+00  5.30880675e-01  1.91868055e-01  3.48114355e-01
   6.49585344e-01  1.08932147e+01  7.99488174e-01  3.38786259e-01
   9.66947914e-02  1.33740311e+00]
 [ 2.63934697e-02  1.10779531e-01  1.85483343e-01  6.88545857e-01
   1.52340162e-01  7.99488174e-01  9.24639090e+00 -1.98792292e-01
  -2.14388372e-01  8.57907167e-01]
 [-4.85572927e-01  1.69990669e-01  4.09352351e-01  3.21912898e-01
  -1.65722441e-01  3.38786259e-01 -1.98792292e-01  9.14332920e+00
   2.94751875e-01  1.36098542e-01]
 [ 9.54450984e-01  8.39373125e-01  2.28436981e-01  7.21089201e-01
   1.35517125e+00  9.66947914e-02 -2.14388372e-01  2.94751875e-01
   1.01275109e+01  1.95897691e+00]
 [ 5.32211658e-01  1.33647214e+00 -1.53964737e-01  1.04788892e+00
  -1.97654085e-01  1.33740311e+00  8.57907167e-01  1.36098542e-01
   1.95897691e+00  1.05847513e+01]]
```

Q3) **The data generated is random and normally distributed with a mean for dataA, dataB and covariance for dataA and dataB given in meanA.xlsx, meanB.xlsx, covarianceA.xlsx and covarianceB.xlsx respectively. Briefly explain why your answers are different from the parameters used to generate the data.**

Since the parameters used to construct this data (Mean, Covariance) are taken from extremely large data, the mean and covariance of (dataA, dataB) diverge from the real mean and covariance. The other argument would be that the mean covariance does not represent the data's underpinning function. The terms mean and covariance are purely mathematical. The reason for this is that the parameters represent the mean and covariance values for the entire population (i.e., the bigger collection of data). The data that (has been supplied to us and) that we are acting on, on the other hand, is a subset of the complete data. As a result, the mean and covariance values are not the same. The meanA, meanB, CovarianceA, and CovarianceB values differ from the parameters used to produce the data.

Neelanjan Goswami

400414867

Q4)
A) Plot 2D Scatterplot graph for each example.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

covlist=[]
var_one=[]
var_two=[]

for i in range(0,20,2):
  data = pd.read_excel ('multNormal.xlsx', header = None,usecols=[i,i+1],names=['colA', 'colB'])
  data_np = pd.DataFrame.to_numpy(data)
  mean = np.mean(data_np, axis = 0)

  print('Mean: ',mean, '\n')
  q= data_np - mean
  shape = data_np.shape

  cov=np.dot(np.transpose(q),(q))/(shape[0])
  print("Covariance:\n")
  print(cov,"\n")
  print("variance: ",cov[0,0]," ",cov[1,1])

  sns.scatterplot(data = data, x = 'colA', y = 'colB',color='red')
  plt.title('scatter plot %d' %v)
  plt.show()

  np.shape(data)
  mX = np.mean(data, axis=0)
  shape=data.shape
  cov = np.dot(np.transpose(data-mX),(data-mX))/(shape[0]-1)

  covlist.append(cov[0,1])
  var_one.append(cov[0,0])
  var_two.append(cov[1,1])
```

Neelanjan Goswami

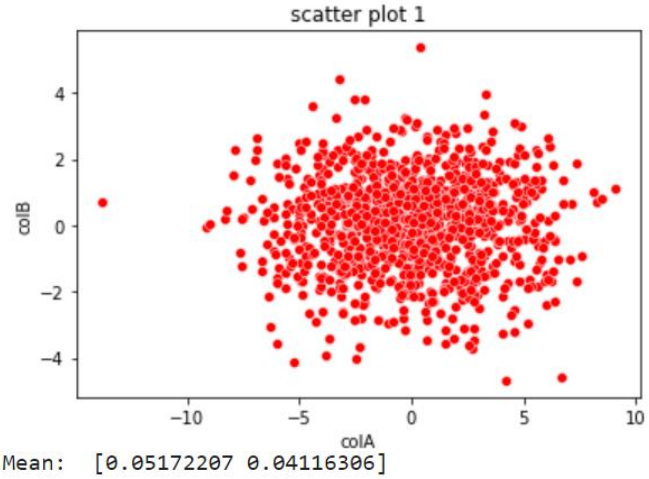
400414867

```
Mean:  [-0.04740766  0.04488932]

Covariance:

[[10.3298651  -0.01202236]
 [-0.01202236  2.05878922]]

variance:  10.329865104160632   2.058789224422458
```

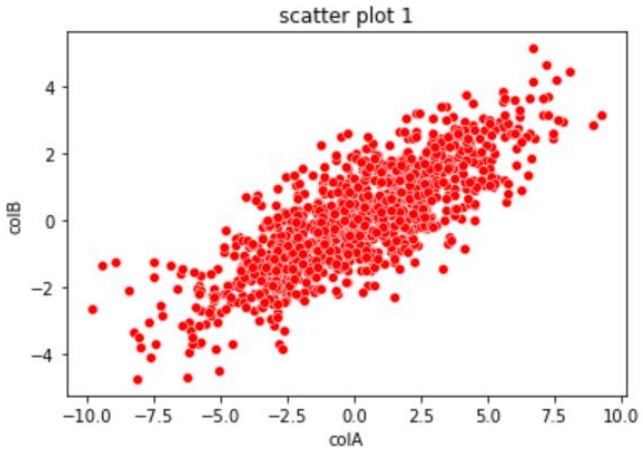



```
Mean:  [0.05172207 0.04116306]

Covariance:

[[9.72412206 3.90489124]
 [3.90489124 2.48007349]]

variance:  9.724122063207144   2.480073489211185
```



Neelanjan Goswami

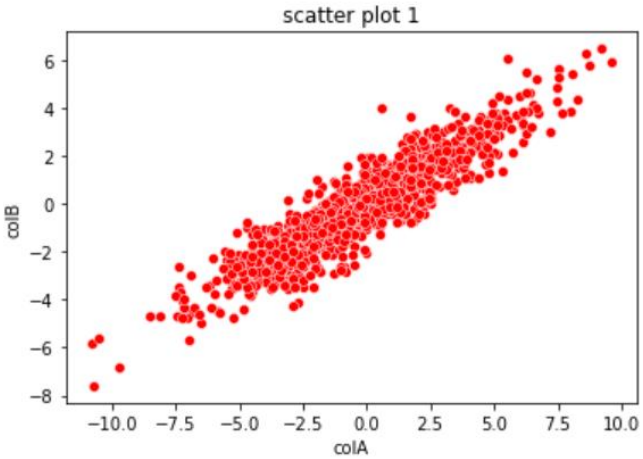
400414867

```
Mean:  [-0.10527492 -0.05752108]

Covariance:

[[9.80349765 5.91605223]
 [5.91605223 4.23722908]]

variance:  9.803497653843339   4.237229080580697
```

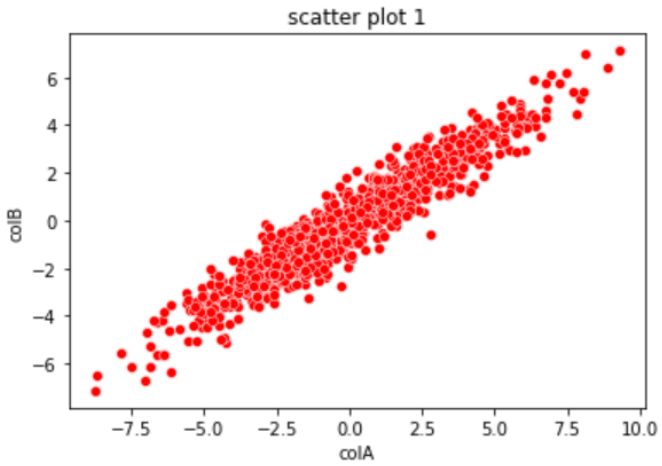



```
Mean:  [-0.03339159 -0.05383925]

Covariance:

[[8.70992967 6.33112001]
 [6.33112001 5.1061709 ]]

variance:  8.70992967084372   5.106170903741053
```



Neelanjan Goswami

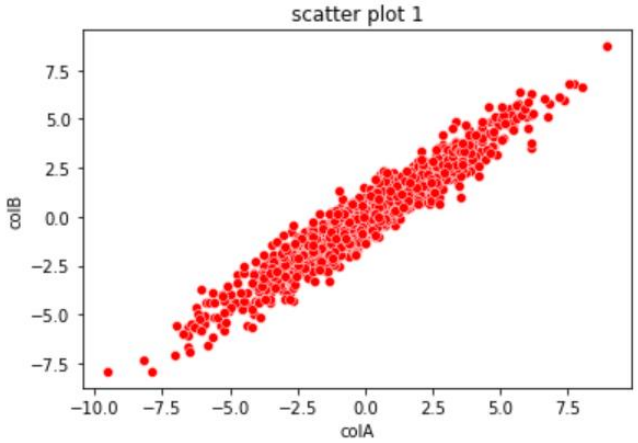
400414867

```
Mean:  [0.090774   0.05885067]

Covariance:

[[8.0793274  7.12489571]
 [7.12489571 6.79829309]]

variance:  8.079327398831369   6.798293090569988
```

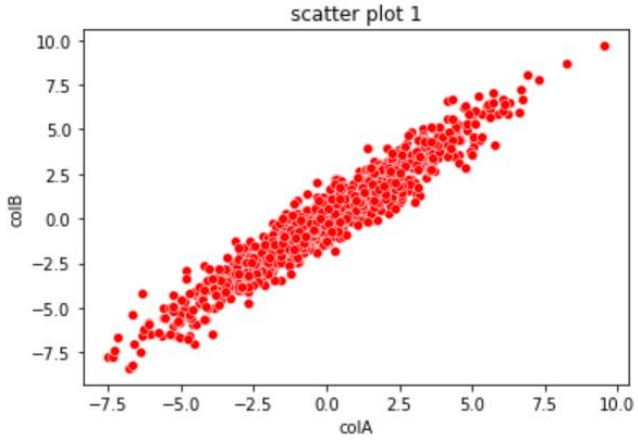



```
Mean:  [0.15811834 0.1380098 ]

Covariance:

[[7.1207944  7.44404877]
 [7.44404877 8.40313235]]

variance:  7.120794402632483   8.403132347687023
```



Neelanjan Goswami

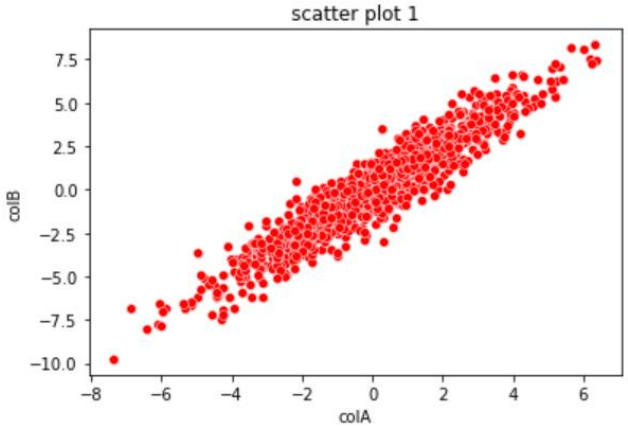
400414867

```
Mean:  [0.03867344 0.04598896]

Covariance:

[[5.1066438  6.25534383]
 [6.25534383 8.59176228]]

variance:  5.106643798466135   8.591762279077884
```

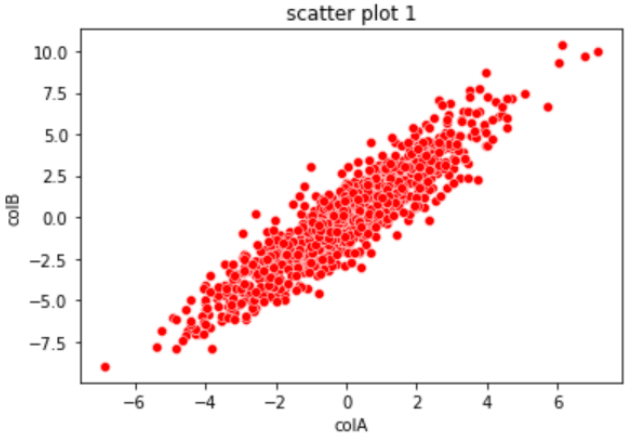



```
Mean:  [-0.01507954 -0.05184973]

Covariance:

[[3.81481296 5.41569181]
 [5.41569181 9.12033767]]

variance:  3.814812960521768   9.120337665533148
```



Neelanjan Goswami

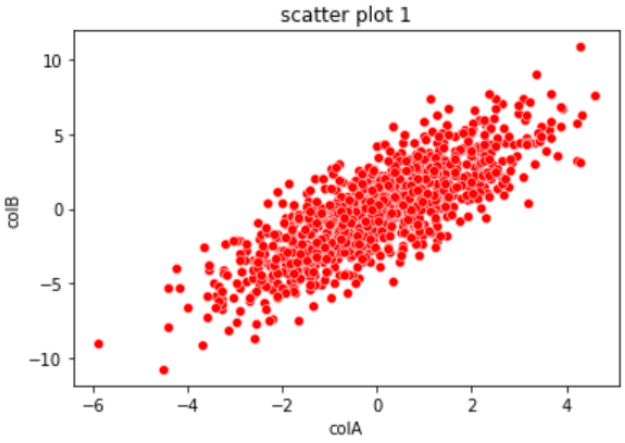
400414867

```
Mean:  [ 0.01622623 -0.05016126]

Covariance:

[[2.62015812 3.99806731]
 [3.99806731 9.33748922]]

variance:  2.620158120691828   9.33748922249571
```

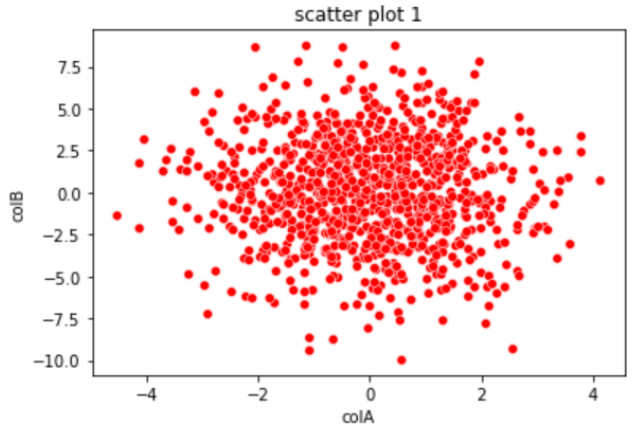



```
Mean:  [-0.02265412  0.08309586]

Covariance:

[[ 1.98447005 -0.09357812]
 [-0.09357812  9.53860132]]

variance:  1.9844700467096144   9.53860132400415
```

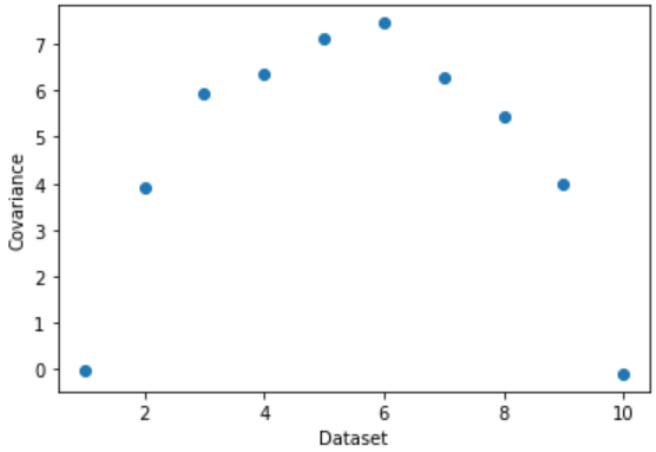


dataset = [1,2,3,4,5,6,7,8,9,10]

covlist

```
[-0.01203439057100108,
 3.908800042506928,
 5.921974201419026,
 6.337457468165683,
 7.132027739623565,
 7.45150027134725,
 6.26160543100501,
 5.421112921806911,
 4.002069380384973,
 -0.09367179090385218]
```

```
plt.scatter(dataset,covlist)
plt.xlabel("Dataset")
plt.ylabel("Covariance")
```



var_one

```
[10.340205309470104,
 9.73385591912627,
 9.813310964808146,
 8.718648319162885,
 8.087414813645013,
 7.12792232495744,
 5.111755554020156,
 3.8186315921138823,
 2.6227809015934214,
 1.9864565032128272]
```

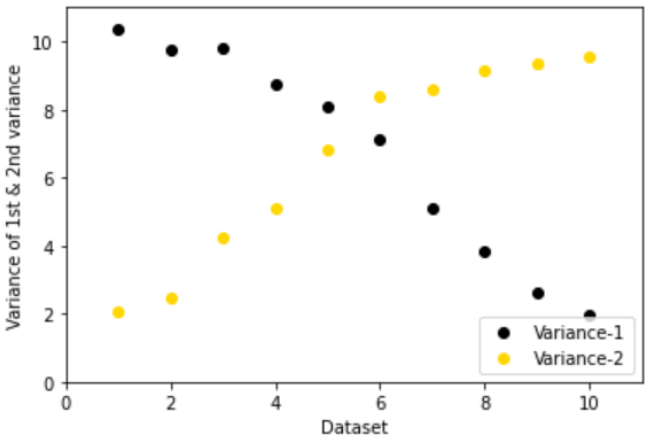
var_two

```
[2.060850074496955,
 2.4825560452564415,
 4.241470551131829,
 5.11128218592698,
 6.8050981887587465,
 8.411543891578603,
 8.600362641719604,
 9.129467132665814,
 9.346836058554265,
 9.548149473477627]
```

```
plt.scatter(dataset,var_one,color='black',label="Variance-1")
plt.scatter(dataset,var_two,color='gold',label="Variance-2")
plt.xlabel("Dataset")
plt.ylabel("Variance of 1st & 2nd variance")
plt.legend(loc='lower right')
plt.xlim(0,11)
plt.ylim(0,11)
```

Neelanjan Goswami

400414867



B) When we look at the graphs geometrically, we can see how the dots start to accumulate towards a specific location as we proceed through the samples. When we look at scatterplot 1, we can see that the points are scattered out, but as we go, the spread narrows and they become centred on a certain location, yet the end plot resembles layout 1. We can also view the data distribution in general.

C)

1. The covariance matrix has undergone rotational and scaling transformations.
2. A. The variance is represented by the diagonals of covariance. We may also say that the variance is determined by the breakdown of eigen values. As a result, we may argue that the eigenvalues reflect the diagonal of the matrix.
   B. A specified covariance matrix must be a semi-decisive matrix that is positive.
3. We must calculate the covariance matrix from the diagonal, which is the variance, and then use it to create fresh data.