***Nikhil Lahoti***
***012448514***

# *Assignment 4*

**Answer 3)** Suppose that we want to learn about a topic and we have a book for that topic. The best way to get as much information about the topic would be to start reading the longest chapter from the book because the most important topic would require the most explanation and would cover majority of the book. We then start reading the chapter which is second largest and connected to this chapter. This is identical the idea of taking the longest road in a town and then the longest road connected to this road and so on.

If we only had to consider two most significant dimensions, then we would read the two longest chapters as they contain most of the explanation about the topic.

**Answer 4)** Please refer to the code inside the code folder for all 3 parts: ***File Name: question4.py***

a) Covariance Matrix =
[[2.5 2.5 4. ]
[2.5 5. 4.5]
[4. 4.5 6.5]]

b) The Eigenvalues are 1.5 and 12.5
Sample output:
Eigen Vectors --->
[-0.424 -0.566 -0.707]
[ 0.408 -0.816 0.408]

Eigen Values --->
12.499999999999984
1.5000000000000004

c) The verification part. The dot product of C * eigenvector = lambda * eigenvector
Eigenvector 1 results:
[-5.303 -7.071 -8.839]
[-5.303 -7.071 -8.839]

Eigenvector 2 results:
[ 0.612 -1.225 0.612]
[ 0.612 -1.225 0.612]

As dot product of c * eigenvector = eigenvalue * eigenvector. Hence verified.

**Answer 9)**

a) The total variance in the input space is 6.0625. Please refer to the code in the code folder. ***Filename: question9.py***

b) The total variance in the projection space if given by the summation of the eigen values.
Lambda1 = 4.0833 Lambda2 = 1.2364 Lambda3 = 0.7428
Summation = 4.0833 + 1.2364 + 0.7428 = 6.0625.
It is same as the total variance in the part a.

The reason is, that the eigen vectors are the basis vectors in the projection space and are unit vectors and orthogonal to each other. This eigen vectors are analogues to finding new x, y and z axis in the projection space and are orthogonal to each other.

When we perform a linear transformation, which is the covariance matrix, on the eigen vectors, the span of the eigen vectors remain the same. But, the dimensions of the eigen vectors changes which is given by the eigen values. The eigen values gives us the effect that this transformation has on the eigen vectors i.e. does it stretches or squishes the eigen vector. This is nothing but the variance in the projected space. So, if the eigen value is large, the vector's length is large which is nothing but the spread of the vector and thus the variance is large.

c) When we consider only the most dominant eigenvector, we only take into account the first eigen value as the variance is determined by the corresponding eigen value.
Thus, the variance taken into consideration
= most dominant eigen value / Total variance of all the features
= (4.0833 / 6.065) * 100 = 67.35 % (approx.)
Similarly, when two are considered
= (two most dominant eigen value) / Total variance of all the features)
= (4.0833 + 1.2364) / 6.0625 = 87.74 % (approx.)

Similarly, for three, all the eigen values are considered, so its 100%.

In general, for l non-trivial out of k non-trivial

= ((sum of first l most dominant eigen values) / (sum of k dominant eigen values)) * 100

**Answer 10)**

Please refer to the code inside the code folder. ***File Name: question10.py*** for the calculations. Here is the output of the program --->

[ 2.507575 -0.88785 ]

Minimum distance Y1:  5.5901699437463756e-05

[-6.083925  0.31135 ]

Minimum distance Y2:  3.7278210026937995

[2.229075 0.34155 ]

Minimum distance Y3:  0.9719411135068835

[ 2.043875 -0.75055 ]

Minimum distance Y4:  0.48363818927479263

**Answer 11)**

Please refer the code inside the code folder: **File Name: question11.py.** It contains the code for both the parts a and b.

**a) The Delta matrix:**

[[-4.7125498   4.3842183   4.0654171  -3.7370856 ]

 [ 1.15784724 -3.26562104  3.54380389 -1.43603009]

 [ 1.52520956  0.69953493 -0.53743633 -1.68730815]]

**b) Sample Output:**

For Y1 -->

[ 1.50839507 -2.97964816 -1.02716098]

Minimum distance Y1:  3.36654399883552

---

For Y2 -->

[ 4.3842183  -3.26562104  0.69953493]

Minimum distance Y2:  4.965068306494546e-16

---

For Y3 -->

[-2.36126822 -1.71775642 -1.97592161]

Minimum distance Y3:  1.433715777908635

---

For Y4 -->

[-0.48801761 -0.18140578 -0.46863535]

Minimum distance Y4:  3.689944218960036

---

**Answer 13)**

a) Please refer to the code inside the code folder. ***File Name: question13a.py.***
**Output --->**
[-1.558825 -1.14825 ]
Minimum distance Y1:  1.1227926981972232

[-0.816625 -2.51495 ]
Minimum distance Y2:  2.275736802252185

[1.989775 0.38765 ]
Minimum distance Y3:  0.7283359136586636

[-0.500225  0.80715 ]
Minimum distance Y4:  0.9575558877292754

b) Please refer to the code inside the code folder. ***File Name: question13b.py***

**Output --->**

```
[[-1.41155725 -1.78517483  2.05192301  1.14480906]
 [-1.22204258  1.53740337  1.52194619 -1.83730697]
 [ 1.46257242 -1.03660105  0.77349322 -1.19946459]]
```

c) Please refer to the code inside the code folder. ***File Name: question13c.py***
**Output --->**

**For Y1 -->**

**Y1 is Benign!**

**----**

**For Y2 -->**

**Y2 is Benign!**

**----**

**For Y3 -->**

**Y3 is Malware**

**----**

**For Y4 -->**

**Y4 is Malware**

**----**

**Answer 17)**

a) Feature 2 has the greatest positive impact on the projection space determined by u1
Feature 4 has the greatest negative impact on the projection space determined by u2

Feature 2 has the greatest overall impact on the projection space determined by u1

b) The code is in the file ***question17.py*** file
Here is the output:

***Component Loading Vector -->***
***CLV1: [ 0.65904462 2.56349574 -1.06329132 -2.20049037 1.89342621 0.30706416]***
***CLV2: [ 0.30205252 0.1322948 -0.99122188 0.52880828 -0.16975772 -0.35905056]***
***CLV3: [-0.0527388 0.21793752 0.29355456 0.25544892 0.27067632 -0.52612524]***

c) We sum the two CLV to get the relative importance. Here is the relative importance of each feature:

***The relative importance of each features ->***
***[ 0.96109714 2.69579054 -2.0545132 -1.67168209 1.72366849 -0.0519864 ]***

As we can see, the feature 2 has the greatest impact on the feature space. It stretches the vector by 2.6957.

The features with most to least importance are as follows:
***Feature 2***
***Feature 3***
***Feature 5***
***Feature 4***
***Feature 1***
***Feature 6***

***The negative sign just shows the direction of the vector.***
Please refer to the code in the code folder. ***File Name: question17.py***