Assignment 2

November 10, 2021

1 Assignment 2

Author : Tathagato Roy

1.1 Roll Number : 2019111020

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import math
import random
```

1.2 Q1

1.2.1 Subpart A

Single Value Decomposition is more generalizable to Matrices than Eigen Value Decomposition as Single Value Decomposition
can applied for both square and rectangular matrices,whereas Eigen Value Decomposition is only applicable to Square Matrices,
even then it is not applicable to all Square Matrices,just symmetric ones.

1.2.2 Subpart B

We have a matrix M :

$$\begin{bmatrix} 4 & 8 \\ 11 & 7 \\ 14 & -2 \end{bmatrix}$$

We want to compute U, Σ and V such that :

$$M = U\Sigma V^T$$

Where U is 3×3 unitary matrix, Σ is 3×2 rectangular Diagonal Matrix and V is a 2×2 Unitary Matrix

$$C = M^T M = \begin{bmatrix} 333 & 81 \\ 81 & 117 \end{bmatrix}$$

$$det(C - \lambda I) = \begin{bmatrix} 333 - \lambda & 81 \\ 81 & 117 - \lambda \end{bmatrix}$$

$$\implies det(C - \lambda I) = (333 - \lambda)(117 - \lambda) - 81^2$$
$$\implies det(C - \lambda I) = \lambda^2 - 450\lambda + 32400$$
$$\implies det(C - \lambda I) = \lambda^2 - 360\lambda - 90\lambda + 90 * 360$$
$$\implies det(C - \lambda I) = (\lambda - 360)(\lambda - 90)$$

By setting $det(C - \lambda I) = 0$ We get $\lambda = 90, 360$

The Eigen Values of C is therefore $\lambda_1 = 360, \lambda_2 = 90$

The singular Values are given by $\sigma_1 = 6\sqrt{10}$ $\sigma_2 = 3\sqrt{10}$

$$\text{Hence we get } \Sigma = \begin{bmatrix} 6\sqrt{10} & 0 \\ 0 & 3\sqrt{10} \\ 0 & 0 \end{bmatrix}$$

The columns of the Matrix V are the orthonormal eigenvectors of the Matrix $C = M^T M$

The eigenvectors $v_1, v_2$of the Matrix C can be computed by solving the set of Following equations

$$Cv_1 = \sigma_1^2 v_1 \implies (C - \sigma_1^2 I)v_1 = 0$$
$$Cv_2 = \sigma_2^2 v_2 \implies (C - \sigma_2^2 I)v_2 = 0$$

which is equivalent to computing the null space of the matrices $D_i = (C - \sigma_i^2 I)$ for $i = 1, 2$

while this can done by solving the set of linear equations ,

but it is extremely tedious.We can compute the eigen vector more easily by using a numpy function

[2]:
```
M = np.array([[4,8],[11,7],[14,-2]])
print(M)
print(M.shape)
```

```
[[ 4  8]
 [11  7]
 [14 -2]]
(3, 2)
```

[3]:
```
M_T = M.T
print(M_T)
print(M_T.shape)
```

```
[[ 4 11 14]
 [ 8  7 -2]]
(2, 3)
```

[9]:
```
C = np.matmul(M_T,M)
print(C)
print(C.shape)
```

```
[[333  81]
 [ 81 117]]
(2, 2)
```

[13]:
```
w , v = np.linalg.eig(C)
print("EigenValues : ")
print(w)
print("EigenVectors : ")
print(v)
```

```
EigenValues :
[360.  90.]
EigenVectors :
[[ 0.9486833  -0.31622777]
 [ 0.31622777  0.9486833 ]]
```

[21]:
```
e1 = v[:,0]
e2 = v[:,1]
print(e1)
print(e2)
print(M)
s1 = np.sqrt(w[0])
s2 = np.sqrt(w[1])
print(s1)
print(s2)
ans1 = np.matmul(M,e1)/s1
ans2 = np.matmul(M,e2)/s2
```

```python
print(ans1)
print(ans2)
```

```
[0.9486833  0.31622777]
[-0.31622777  0.9486833 ]
[[ 4  8]
 [11  7]
 [14 -2]]
18.973665961010276
9.486832980505138
[0.33333333 0.66666667 0.66666667]
[ 0.66666667  0.33333333 -0.66666667]
```

The normalized Eigenvectors are :

$$v_1 = \begin{bmatrix} 0.948 \\ 0.316 \end{bmatrix}$$

$$v_2 = \begin{bmatrix} -0.316 \\ 0.948 \end{bmatrix}$$

Note that this vectors are not unique,given there are 3 other choice of combination(by selectively multiplying v_1, v_2 with - 1. Also note that $v_1, v_2$are ordered according to decreasing order of the modulus their respective Eigenvalue.

$$V = \begin{bmatrix} v_1 & v_2 \end{bmatrix}$$
$$\implies V = \begin{bmatrix} 0.948 & -0.316 \\ 0.316 & 0.948 \end{bmatrix}$$

Now we want to compute U which a 3X3 Matrix. We know columns of U forms of Orthonormal Basis of the Space Ax.

The singular Values of A is $\sqrt{90}$ and $\sqrt{360}$

hence the rank of A is 2 as there are two singular Values.

Therefore the rank of Ax is also 2.The first two columns of U is therefore given by :

$$u_1 = \frac{Av_1}{\sigma_1}$$
$$u_2 = \frac{Av_2}{\sigma_2}$$

$$u_1 = \frac{1}{\sigma_1}\begin{bmatrix} 4 & 8 \\ 11 & 7 \\ 14 & -2 \end{bmatrix}\begin{bmatrix} 0.948 \\ 0.316 \end{bmatrix}$$

$$\implies u_1 = \begin{bmatrix} 1/3 \\ 2/3 \\ 2/3 \end{bmatrix}$$

$$u_2 = \frac{1}{\sigma_1}\begin{bmatrix} 4 & 8 \\ 11 & 7 \\ 14 & -2 \end{bmatrix}\begin{bmatrix} 0.316 \\ 0.948 \end{bmatrix}$$

$$\implies u_2 = \begin{bmatrix} 2/3 \\ 1/3 \\ -2/3 \end{bmatrix}$$

We need to compute u_3 such that $u_1^T u_3 = 0$ and $u_2^T u_3 = 0$ that is u_3 is normal to space spanned by u_1 and u_2

$$\text{One such example of } u_3 \text{ is } \begin{bmatrix} 2/3 \\ -2/3 \\ 1/3 \end{bmatrix}$$

$$U = \begin{bmatrix} 1/3 & 2/3 & 2/3 \\ 2/3 & 1/3 & -2/3 \\ 2/3 & -2/3 & 1/3 \end{bmatrix}$$

$$\text{Hence } M = \begin{bmatrix} 1/3 & 2/3 & 2/3 \\ 2/3 & 1/3 & -2/3 \\ 2/3 & -2/3 & 1/3 \end{bmatrix}\begin{bmatrix} 3\sqrt{10} & 0 \\ 0 & 6\sqrt{10} \\ 0 & 0 \end{bmatrix}\begin{bmatrix} 0.948 & -0.316 \\ 0.316 & 0.948 \end{bmatrix}^T$$

Sanity Check :

[25]:
```
U = np.array([[1/3,2/3,2/3],[2/3,1/3,-2/3],[2/3,-2/3,1/3]])
S = np.array([[np.sqrt(360),0],[0,np.sqrt(90)],[0,0]])
V = np.array([[0.948,-0.316],[0.316,0.948]])
M_1 = np.matmul(U,np.matmul(S,V.T))
print(M_1)
print("Difference")
print(M - M_1)
```

```
[[ 3.99711896  7.99423792]
 [10.99207715  6.99495818]
 [13.98991637 -1.99855948]]
Difference
[[ 0.00288104  0.00576208]
 [ 0.00792285  0.00504182]
 [ 0.01008363 -0.00144052]]
```

1.3 Q2

1.3.1 Subpart A

Assumption : The Decomposition is a SVD Decomposition and hence D is a diagonal Matrix

Option B is correct as one Single Value being larger than the other suggest one is the principle component

If both the diagonal values are equal,then there is no way to find the principle component.

Option C is also Correct.As all the points in X lie in a straight line, therefore the columns of X

which is NX2 Matrix are not linearly Independent and therefore its rank is 1,and as such the number of

nonzero singular value in D is also 1.Hence D is not full rank

1.3.2 Subpart B

It is False.It is not neccessary PCA preserves Useful Information for MultiClass Classification,

only just that it maximizes the variance

1.3.3 Q3

1.3.4 Subpart A

In Bayesian Statistic The Prior Probability indicates the probability of Random Event / Uncertain Quantity before any data is collected/sampled. It is a expression of our belief about this Random Event / Uncertain Quantity before any evidence is taken into account.

The Posterior Probability is the conditional probability of a Random Event / Uncertain Quantity given we have now seen the Evidence/Data X.

Bayes Theorem :

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

Where :

$P(\theta|X)$ is the posterior Probability

$P(X|\theta)$is the Likelyhood of seeing X given the parameter θ

$P(\theta)$ is the prior probability of θ or our belief of quantity θ

$P(X)$ is the probability of seeing the evidence/Data X

1.3.5 Subpart B

Let E be the event when someone have Sore throat and Headache

Let F be the event when someone have the flu

$P(E)$is the probability of someone having the symptoms Sore Throat and Headache

$P(F)$ is the probability of someone having the flu

$P(E|F)$ is the probability someone having the headache and sore throat given they have the flu

$P(F|E)$ is the probability someone having the flu given they have a headache and sore throat

Given in the problem :

$$P(F) = 0.05$$

$$P(E) = 0.2$$

$$P(E|F) = 0.9$$

We want to now compute P(F|E) ,i.e the probability of having the Flu given someone already has headache and

Using Bayes Theorem :

$$P(F|E) = \frac{P(E|F) * P(F)}{P(E}$$

$$\implies P(F|E) = \frac{0.9 * 0.05}{0.2}$$

$$\implies P(F|E) = \frac{9 * 5 * 10}{2 * 100 * 10}$$

$$\implies P(F|E) = \frac{9}{40}$$

$$\implies P(F|E) = 0.225$$

Hence there is a 0.225 probability of someone having the Flu given they have headache and sore throat

[ ]: