2021 6th International Conference for Convergence in Technology (I2CT)
Pune, India. Apr 02-04, 2021

A Model Based Linear Algebraic Approach for Machine Learning

Samrudhi Kaware
Assistant Professor
G H Raisoni College of Engineering and Management
Wagholi, Pune, India
samrudhikaware17@gmail.com

***Abstract*—Linear algebra is sub field of mathematics and contains matrix's, Operations on dataset, Vectors. Linear algebra is core base for purely statistics and mathematics person to achieve goals with basic logics of linear algebra using machine learning. As a machine learning aspirant, the aspirant should be best at linear algebra to work on creating a new module or bundle. In other areas, such as statistics, engineering, research, linear algebra plays an important role and has strongest impact Although solving any problem mathematically with linear algebra is crucial for experienced engineers, linear algebra is crucial in machine learning, and in the case of any practical implementation of linear algebra, it is quite probable that the technical person may not have mathematical knowledge. Easier implementation is provided by linear algebra.**

Coronavirus disease (COVID-19) has raised urgent questions regarding minimizing and improving effective analytical strategies for analyzing information obtained from standardized protocols. In order to understand what quality of care interventions are the most successful and to test them as quickly as possible, it becomes necessary to evaluate available data immediately. It is necessary to integrate and clean up data from large multi-center hospitals and needs sophisticated data processing. To collect and store their data in a standardized manner, case report forms (CRF) for patients with suspected or confirmed COVID-19 are required. The goal of this study is to determine if it is possible to avoid or postpone local outbreaks of COVID-19 by travel restrictions from abroad. The problem statement is that we are unable to gather clear suspicious data from big data of covid-19 patients, and during the corona outbreak it becomes difficult to distinguish symptotic and asymptotic patients. The number of beds required or ventilators needed for COVID-19 cases at this point is difficult to estimate.

Keywords—Dimension Reduction, Singular Value Decomposition, Singular Vectors & Singular Values, Eigen Value and Eigen Vector.

I. INTRODUCTION

Linear Equation is an equation which may be represented as follows

$$a_1x_1 + \cdots + a_nx_n + b = 0,$$

Where $x_1, \ldots, x_n$ are variables, $b, a_1, \ldots, a_n$ are the coefficients which are often real numbers

Linear algebra is a part of mathematics with respect to linear equations

$$a_1x_1 + \cdots + a_nx_n = b,$$

Linear mapping as,

$$(x_1, \ldots, x_n) \mapsto a_1x_1 + \ldots + a_nx_n,$$

Consists of vector spaces and through matrices

The reason to use Linear algebra, it is sub field of mathematics and contains matrix's, Operations on dataset, Vectors. Linear algebra is core base for purely statistics and mathematics person to achieve goals with basic logics of linear algebra using machine learning.

As a Machine learning aspirant to work on developing new module or package the aspirant should be best at linear algebra. Linear algebra plays an important role and have best impact on other fields such as Statistics, Engineering, Science.

The reason why linear algebra essential in machine learning, while solving any problem mathematically with linear algebra is crucial for technical person and in case of some practical implementation of linear algebra it is quite possible the technical person won't have mathematical knowledge person is good at his programming skills. Linear algebra supports easier implementation.

Linear Algebra's Field to be studied

5 fields of linear algebra every Machine Learning aspirant must learn are

- Linear Algebra Notation
- Linear Algebra Arithmetic
- Linear Algebra for Statistics
- Factorization of matrix

Linear Algebra Notation: Learn to write and read matrix and vector. Notation allows specific operators to accurately diagnose operations on data.

Linear Algebra Arithmetic: Arithmetic operations are performed by pairing with linear algebra notation. Arithmetic operations consist of scalars, vectors and matrices being added, subtracted, and multiplied.

There are several difficulties facing freshers in the field of linear algebra for performing operations such as matrix multiplication, tensor multiplication that are not implemented as direct element multiplication.

Linear Algebra for statistics: Describing, understanding, sampling data is concerned to statistics. Chunks of data is analyzed and then data cleaning process takes place in machine learning.

2021 6th International Conference for Convergence in Technology (I2CT) | 978-1-7281-8876-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/I2CT51068.2021.9418083

Factorization of Matrix: Constructing on arithmetic and notation is the core concept of matric factorization also known as matrix decomposition.

In linear algebra, factoring matrix is an enhance the integrity. Elements are commonly used in many more complex operations in both linear algebra (inverse matrix) and machine learning (Principal Component Analysis, Singular Value Decomposition, least square)

II. MACHINE LEARNING TECHNIQUES WITH LINEAR ALGEBRA

Linear algebra is a short notation for describing some compact operation.

Some common machine learning technique via linear algebra are describe below

1. Principal Component Analysis (PCA)
2. Linear Least Square for linear Regression
3. Singular Value Decomposition (SVD)

A. Principal Component Analysis (PCA)

Data Compression: Data compression is a method of reducing the number of bits needed for either data storage or transmission.

Data compression is categorized into methods that are either lossy or lossless.

In lossy technique, as name suggests there is loss of information or data. Where there is no loss, as in a lossless method. There are various forms of algorithm for coding, such as Huffman coding, coding for run length.

Lossy Compression technique

Lossy Compression technique at the cost of data quality one can achieve higher compression ratio.

There are two methods of lossy data compression:

- Wavelet transforms
- Principal Component Analysis PCA

Principal Component Analysis (PCA): A representation of data with orthogonal fundamental vectors is provided by Principal Component Analysis (PCA). Eigen values of data matrix covariance. This original projection dataset is reduced with little loss of knowledge, which is derived using the singular value decomposition method.

PCA is interpreted by the covariance matrix's own value/eigen vector approach.

Before moving forward let's understand core concepts like Dimensionality, correlation, orthogonal, eigen vectors, covariance matrix.

Dimensionality: The number of random variables in a dataset or either the number of features or, more specifically, the number of columns in the dataset.

Correlation: It shows how strongly two variables are correlated to each other. The same range value for +1 to-1. Positive indicates that as one variable increases, the other also increases. Whereas negative suggests the other decrease of the increase of the former.

Orthogonal: irrelevant to each other, ie. Correlation between any variable pair ie 0.

Eigen vectors: Eigen values and Eigen vectors are a major domain of themselves. Let's limit ourselves to the awareness of the same thing that we would need here.

Consider non zero vector V. It is an eigen vector of square matrix A, if AV is scalar multiple of V.

$$AV = \lambda . V$$

V is eigen vector
λ is eigen value.

Covariance matrix: Layer is calculated consisting of covariances between pairs of variables. The (i, j) component is the covariance between the i^{th} and the j^{th} variable.

III. LINEAR LEAST SQUARE FOR LINEAR REGRESSION

It is also known as linear model; Linear regression is a predictive algorithm which mostly provides a linear relationship between both the [calls Y] prediction and the [call is X] data.

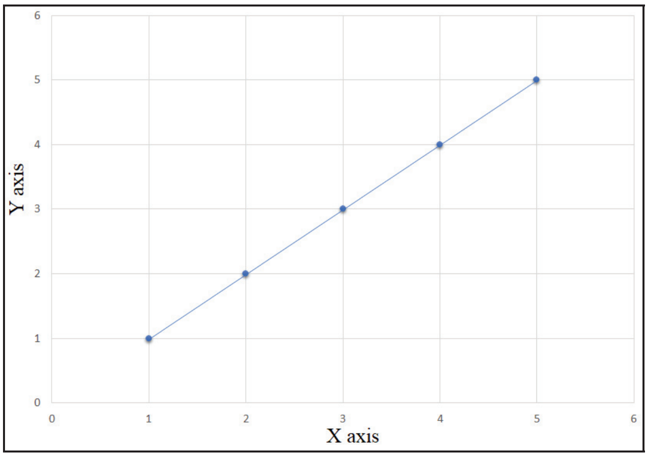
If we plot an X, the Y linear relationship will always come up with a straight line.

Example if we plot the graph of these values:

X= 1,2,3,4,5 …… (input)

Y= 1,2,3,4,5 …… (output)

It will be perfect straight line.



Least square regression

The least square regression is a statistical technique that minimizes the error in such a way that the amount of all square error is minimized.

Steps to estimate the minimum square regression.

Formula for the estimation of m=slope

$$m = \frac{\text{Sum of all } (X - \text{xmean})*(Y - \text{ymean})}{\text{Sum of } (X - \text{xmean})**2}$$

**2 syntax in python

Mean value of all 'x'= (1+2+3+4+5)/5 = 3

Calculating X – xmean for all X value

At X = 1: (X - xmean) = 1 - 3 = -2

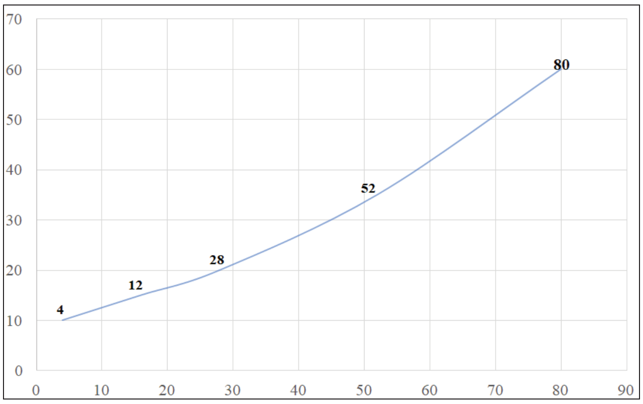
At X = 2: (X - xmean) = 2 - 3 = -1

At X = 3: (X - xmean) = 3 - 3 = 0

At X = 4: (X - xmean) = 4 - 3 = 1

At X = 5: (X - xmean) = 5 - 3 = 2

Calculating all Y – ymean values for all Y



Y – ymean is: (4+1+28+52+80)/5 = 35.2

At Y= 4: (Y - Ymean) = 4- 35.2= -31.2

At Y= 12: (Y - Ymean) = 12- 35.2= -23.2

At Y= 28: (Y - Ymean) = 28 - 35.2= -7.2

At Y= 52: (Y - Ymean) = 52 - 35.2= 16.8

At Y= 80: (Y - Ymean) = 80 - 35.2= 44.8

The availability of these values enables us to determine the sum of all values

(x,y) = (X-xmean)*(Y-ymean)

(1,4) = (X-xmean) *(Y-ymean) = (-2*-31.2) =62.4

(2,12) = (X-xmean) *(Y-ymean) = (-1*-23.2) =23.2

(3,28) = (X-xmean) *(Y-ymean) = (0*-7.2) =0

(4,52) = (X-xmean) *(Y-ymean) = (1*16.8) =16.8

(5,80) = (X-xmean) *(Y-ymean) = (2*44.8) =89.6

Sum of all = (62.4+23.2+0+16.8+89.6) =192

Now denominator part,

Sum (X-xmean) **2

= sum (-2**2, -**2,0**2,1**2,2**2) =sum (4,1,0,1,4) =10

$$m = \frac{\text{Sum of all (X – xmean)*(Y – ymean)}}{\text{Sum of (X - xmean)**2}}$$

=192/10=19.2

Calculate of Y intercept, formula

b= ymean – m * xmean

=35.2-19.2*3

=35.2-57.6

= -22.4

Overall formula can now be written inform of

Y= mx+b

Y= mx+b

= 19.2x+(-22.4)

Least Square regression

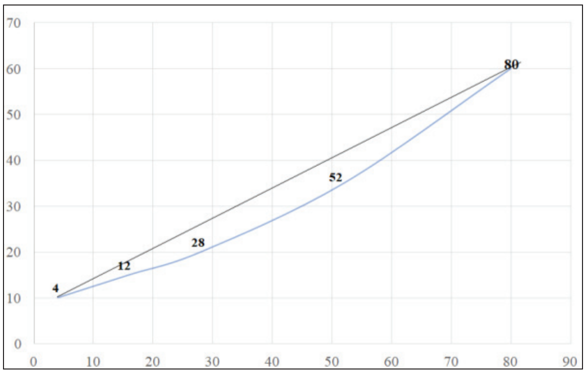
x=1 $\Rightarrow$ y= 19.2 *1 – 22.4= -3.2

x=2 $\Rightarrow$ y=19.2 * 2 – 22.4 = 16

x=3 $\Rightarrow$ y= 19.2 * 3 – 22.4 = 35.2

x=4 $\Rightarrow$ y= 19.2 * 4 – 22.4 = 54.4

x=5 $\Rightarrow$ y= 19.2 * 5 – 22.4 = 73. 6



IV. Singular Value Decomposition

The solution for PCA in terms of the covariance matrix's Eigen vectors. There is, however, another way to obtain a solution based on a singular decomposition value, or SVD. This generally generalizes the notion of Eigen vectors from square matrices to any kind of matrix.

There is a factorization (Singular Value Decomposition = SVD) as follows for $A_{m \times n}$ having a rank r:

$$A = \underbrace{U}_{m \times m}\underbrace{\Sigma}_{m \times n}\underbrace{V^T}_{V \text{ is } n \times n}$$

Columns U are orthogonal eigenvectors of AA^T.

Columns V are orthogonal eigenvectors of A^TA.

Eigenvalues $\lambda_1 \ldots \lambda_r$ of AA^T are the eigenvalues of A^TA.

$\sigma_i = \sqrt{\lambda_i}$ ………….. Singular value

$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$

Singular vectors and singular values

Properties of AA^T and A^TA:

- Symmetrical
- Square
- At least positive semi definite (eigen values are zero or positive)
- Both matrices have same positive eigen values

Both have same rank r as A.

V. PROPOSED METHODS

In the event that COVID-19 has started to spread in India, we have developed a mathematical model that could minimize disease transmission. Reduction of dimension is the solution to the curse of dimensionality. Dimensional reduction models minimize the size of the data by removing the relevant information and discarding the rest of the data as noise.

The most convenient mathematical language to express data models is “Linear Algebra”. The main concept we have used is Singular Value Decomposition (SVD).

The singular value decomposition (SVD) of the matrix is the fundamental method of decomposition of the matrix in linear algebra. It was referred to as the "Fundamental Theorem of Linear Algebra" (Strang, 1993) since it can be extended to all matrices, not just square matrices, and it always remains. Instead of lumping together the size of the population as a whole, we are monitoring it in one amount, regardless of age or stage of growth.

In pandemic situation, rapid and valid information flow and reporting is crucial. Long-lasting reporting guidelines might do more harm than good. Specific reporting guidelines are needed for pandemic settings from standard protocols. To enable analysis of data gathered during COVID-19 pandemic, principles of open science and raw data sharing will be of paramount importance. International norms have been proposed for data sharing during international health emergencies.

VI. POSSIBLE OUTCOMES

It becomes easier to visualize the high dimensional data when reduced to very low dimensions such as 2D or 3D. we can easily extract relevant summaries quickly and most efficiently by reducing space complexity. The new direction revealed will be an affordable way to reduce future pandemic. Then it will look at how countries can reduce, mitigate and adapt to future pandemic and climate risks. It will update reporting guidelines for observational studies. This proposal describes a six-month study, using existing data and studies to analyze through a systems approach. The proposed focus will be global. The project will look at the COVID 19 origin. This is an innovative movement that needs to be studied to see how it can enable risk reduction, mitigation and adaptation to proceed quickly. The best possible outcome we can extract will be to be able to predict the situation and prevent devastating impacts on human life and welfare. It’s possible to take the right actions quickly, then we can save our life and save people’s lives in our community by preventing community spread of the virus.

It can be concluded that about a few per cent of infected patients will need intensive care and that half of those admitted to the intensive care unit will need mechanical ventilation Over time, appropriate numbers for health planning can be generated once the model has also been validated.

Dimension Reduction: Dimension Reduction operates on a data winding function in a bundle (zipping). Compressing huge data set features into a new lower dimensional subspace function without losing the significance of information.

Factorization of Matrix: Matrix Factorization: linear algebra strategies used to minimize dimensionality. Matrix Factorization methodologies can be used to reduce the dataset matrix to its constituent parts.

Examples are Eigen decomposition and singular value decomposition.

Applications of proposed model are:

- Timely identification of Covid-19 infected and healthy patients.
- Prediction of situations through call recordings.
- Image Compression.
- Google Page Ranking.
- Finding Structure in Movie Ratings and Consumers.

VII. IMPLEMENTATION

Prerequisites: Python 3.x, and IDE: Anaconda/ pycharm Core Packages: numpy, scipy, matplotlib

Advance linear algebraic technique to identify COVID 19 patients at earlier stage.

Given Matrix: $A = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix}$

Solution: $AA^T = \begin{pmatrix} 17 & 8 \\ 8 & 17 \end{pmatrix}$, $A^TA = \begin{pmatrix} 13 & 12 & 2 \\ 12 & 13 & -2 \\ 2 & -2 & 8 \end{pmatrix}$

These matrices have at least positive semi definite (all eigenvalues are positive or zero). They share the same positive eigen values

$AA^T = \begin{pmatrix} 17 & 8 \\ 8 & 17 \end{pmatrix}$ $A^TA = \begin{pmatrix} 13 & 12 & 2 \\ 12 & 13 & -2 \\ 2 & -2 & 8 \end{pmatrix}$

eigenvalues: $\lambda_1 = 25, \lambda_2 = 9$ eigenvalues: $\lambda_1 = 25, \lambda_2 = 9, \lambda_3 = 0$

eigenvectors eigenvectors

$u_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$ $u_2 = \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}$ $v_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{pmatrix}$ $v_2 = \begin{pmatrix} 1/\sqrt{18} \\ -1/\sqrt{18} \\ 4/\sqrt{18} \end{pmatrix}$ $v_3 = \begin{pmatrix} 2/3 \\ -2/3 \\ -1/3 \end{pmatrix}$

Implementation of above example:

Step 1: Import packages

```
In [1]: import numpy as np
        import scipy.linalg
        import seaborn as sns
```

Step 2: Declaration of given matrix

```
In [2]: A = np.array([[3,2,2], [2,3,-2]])
```

Step 3: Calculate transpose of initial matrix A

```
In [3]: Atrans= A.transpose()
        print("A Transpose is :")
        print(Atrans)

        A Transpose is :
        [[ 3  2]
         [ 2  3]
         [ 2 -2]]
```

Step 4: Matrix multiplication of initial matrix A and transpose of matrix A

```
In [4]: # matrix multipication for A.At
        AAt= A.dot(Atrans)
        print("Matrix multipication for A.At")
        print(AAt)

        Matrix multipication for A.At
        [[17  8]
         [ 8 17]]
```

Step 5: Matrix multiplication of transpose of matrix A and initial matrix A

```
In [5]: # matrix multipication for At.A
        AtA= Atrans.dot(A)
        print("Matrix multipication for At.A")
        print(AtA)

        Matrix multipication for At.A
        [[13 12  2]
         [12 13 -2]
         [ 2 -2  8]]
```

Step 6: Calculate eigen value and eigen vector of matrix multiplication of initial matrix A and transpose of matrix A

```
In [6]: W, V = np.linalg.eig(AAt)
        print("Eigen Values for A.At")
        print(W)
        print("Eigen Vectors for A.At")
        print(V)

        Eigen Values for A.At
        [25.  9.]
        Eigen Vectors for A.At
        [[ 0.70710678 -0.70710678]
         [ 0.70710678  0.70710678]]
```

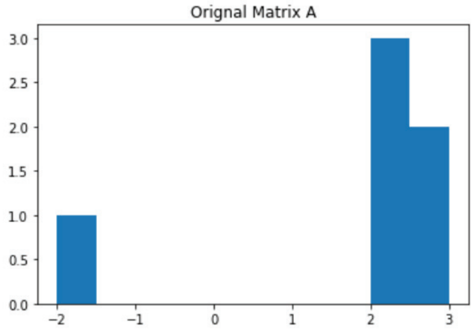
Step 7: Calculate eigen value and eigen vector of matrix multiplication of transpose of matrix A and initial matrix A.

```
In [ ]: WT, VT = np.linalg.eig(AtA)
        #eig() calculates eigen for only square matrix.
        #linalg.eigvals(a):Compute the eigenvalues of any matrix.
        print("Eigen Values for At.A")
        print(WT)
        print("Eigen Vector for At.A")
        print(VT)

        Eigen Values for At.A
        [2.50000000e+01 3.61082692e-15 9.00000000e+00]
        Eigen Vector for At.A
        [[-7.07106781e-01 -6.66666667e-01  2.35702260e-01]
         [-7.07106781e-01  6.66666667e-01 -2.35702260e-01]
         [-1.16614446e-17  3.33333333e-01  9.42809042e-01]]
```

Step 8: Visualization of initial matrix A.

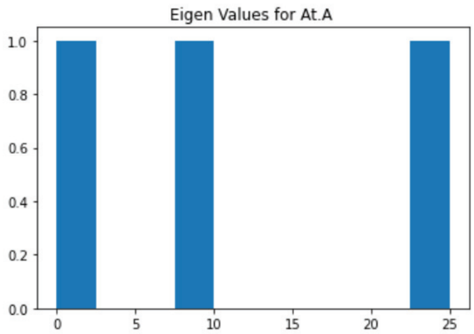
Ravel(): This feature allows you to flatten the arrays. This means that whether you ever have 2D, 3D or n-D arrays, you can use this feature to flatten all of them to a 1-D array.

```
In [ ]: # Import numpy and matplotlib
        import matplotlib.pyplot as plt
        # Construct the histogram
        plt.hist(A.ravel())
        # Add a title to the plot
        plt.title('Orignal Matrix A')
        # Show the plot
        plt.show()
```



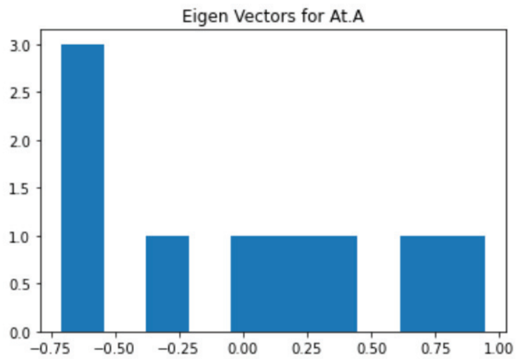
Step 9: Eigen value of matrix multiplication of A transpose and initial matrix A

```
In [ ]: plt.hist(WT.ravel())
        # Add a title to the plot
        plt.title('Eigen Values for At.A ')
        # Show the plot
        plt.show()
```



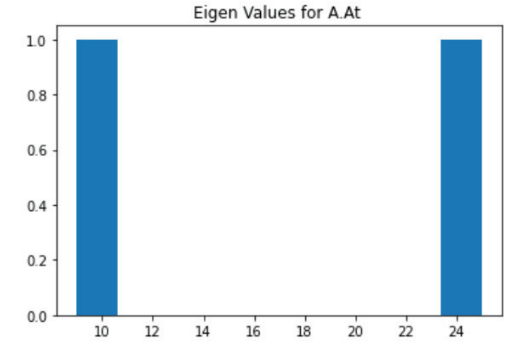
Step 10: Eigen vectors of matrix multiplication of A transpose and initial matrix A

```
In [ ]: plt.hist(VT.ravel())
        # Add a title to the plot
        plt.title('Eigen Vectors for At.A')
        # Show the plot
        plt.show()
```



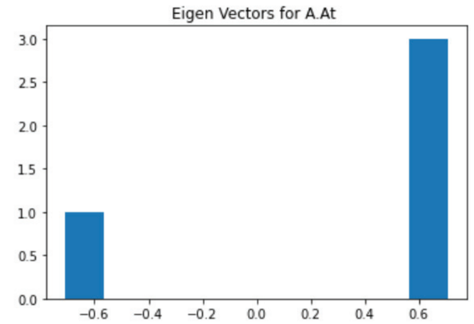
Step 11: Eigen value of matrix multiplication of initial matrix A and A transpose

```
In [ ]: plt.hist(W.ravel())
        # Add a title to the plot
        plt.title('Eigen Values for A.At ')
        # Show the plot
        plt.show()
```



Step 12: Eigen vectors of matrix multiplication of A transpose and initial matrix A

```
In [ ]: plt.hist(V.ravel())
        # Add a title to the plot
        plt.title('Eigen Vectors for A.At ')
        # Show the plot
        plt.show()
```



Step 13: Two ways of computing SVD in python

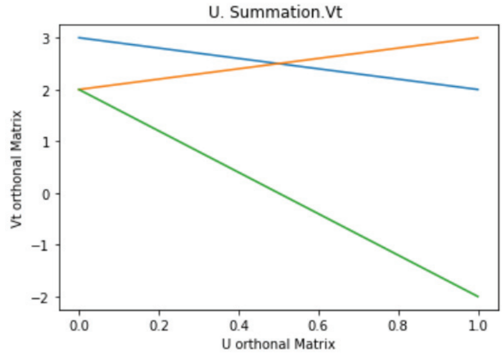
1. By using numpy package: linalg (linear algebra) package
2. By using '@' operator: @ is for matrix multiplication.

```
In [ ]: u, s, vt = np.linalg.svd(A, full_matrices=False)
        print("\nFactoration of matrix with Singular Value Decomposition:")
        print("\nU=", u, "\n\ns=", s, "\n\nVt=",vt)

        xa=u @ np.diag(s) @ vt
        print("\n U. Summation.Vt \n",xa)
```

```
Factoration of matrix with Singular Value Decomposition:

U= [[-0.70710678 -0.70710678]
 [-0.70710678  0.70710678]]

s= [5. 3.]

Vt= [[-7.07106781e-01 -7.07106781e-01 -5.55111512e-17]
 [-2.35702260e-01  2.35702260e-01 -9.42809042e-01]]

 U. Summation.Vt
 [[ 3.  2.  2.]
 [ 2.  3. -2.]]
```

Step 14: Visualization of Singular Value Decomposition as line graph.

```
In [ ]: plt.plot(xa)
        plt.title("U. Summation.Vt ")
        plt.xlabel("U orthonal Matrix")
        plt.ylabel("Vt orthonal Matrix")
        plt.show()
```



VIII. CONCLUSION

Extraction of related summaries easily and effectively. Reduces the complexity of space. High dimensional data visualization. Appropriate numbers for health planning can be created. Infected and stable patients may be identified during the healthy patient during the corona epidemic.

ABBRIVATIONS

PCA- Principal Component Analysis, SVD- Singular Value Decomposition.

REFERENCES

[1] Ethem Alpaydın, "Introduction to Machine Learning Second edition ", The MIT Press Cambridge, Massachusetts London, England.

[2] Kelvin P. Murphy, "Machine Learning: A Probabilistic Perspective", The MIT Press Cambridge, Massachusetts London, England.

[3] Marc Peter Deisenroth, A Aldo Faisal, Cheng soon Ong, "Mathematics for Machine Learning", Cambridge University Press.

[4] Li M. Chen, Zhixun Su, Bo Jiang, "Mathematical problems in data science, Theoretical and Practical Methods", Springer (2015).