A Presentation on PCA & SVD

PRESENTED BY

MSc. student, N. I. MD. ASHAFUDDULA

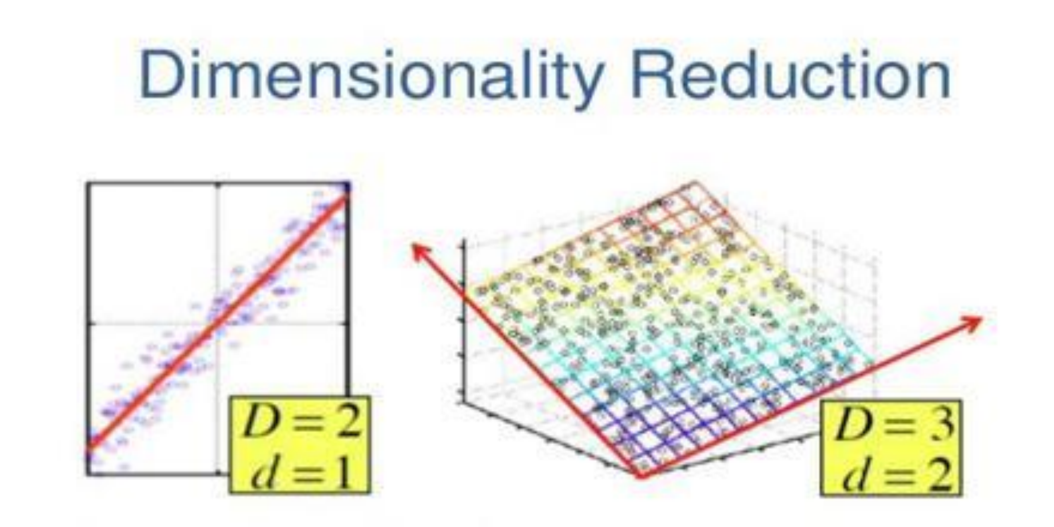
STUDENT ID: 18204016

Outlines

- Feature Reduction
- Why Dimension Reduction?
- Application of SVD & PCA
- Recent Works
- Keywords & Terms
- PCA
- SVD
- Implementation of PCA & SVD
- Result Comparison (SVD/PCA vs Original Data)
- Conclusion

Feature Reduction

- **Dimensionality reduction** refers to techniques for **reducing** the number of input variables in training data.



[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

Feature Reduction

- When dealing with high **dimensional** data, it is often useful to **reduce** the **dimensionality** by projecting the data to a lower **dimensional** subspace which captures the "essence" of the data.

Table: 5D input (Original data)

Record	F1	F2	F3	F4	F5
1	**1**	**1**	**1**	0	0
2	**2**	**2**	**2**	0	0
3	**3**	**3**	**3**	0	0
4	**4**	**4**	**4**	0	0
5	0	2	0	**4**	**4**
6	0	0	0	**5**	**5**
7	0	1	0	**2**	**2**

Dim. Reduction (PCA/SVD) →

Table: 2D input (Reduced Dim)

Record	F1	F2
1	1.72	-0.22
2	5.15	-0.67
3	5.87	-0.89
4	8.58	-1.12
5	1.91	5.62
6	0.90	6.95
7	0.95	2.81

Why Dimension Reduction?

- ❑To compress data
- ❑To remove redundant features
- ❑To use less Computation & Disk space
- ❑To Speed up learning algorithm
- ❑**To Increase system performance**
- ❑**To visualize data**

Application of SVD & PCA

- Fingerprint Recognition
- Recommendation System
- Image Classification
- Computer vision
- Dimension Reduction
- Analysis Input variables

And many more.

Recent Works

[1] A Machine Learning Approach to Detect Self-Care Problems of Children with Physical and Motor Disability. (2018)

[2] Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques. (2020)

[3] Improving detection of Melanoma and Naevus with deep neural networks. (2020)

[4] Dimension reduction of image deep feature using PCA. (2019)

[5] Analysis of Dimensionality Reduction Techniques on Big Data. (2020)

[6] Image Classification base on PCA of Multi-view Deep Representation. (2019)

Recent Works (Result Comparison)

Work	Original Input Features	Reduced Features	Method	Without Feature Reduction	With Feature Reduction
[1]	205	53	PCA, KNN	Acuu: 81.43%	Accu: 84.29%
[2]	15	To Analyze	PCA, RF	-	Accu: 97.4%
[3]	2 Dataset	-	PCA, CNN	-	Accu: 96.8%
[4]	3727	887	PCA	Accu: 88.3%	Accu: 91.3%
[5]	36	26	PCA, RF	Acuu: 98.59%	Acuu: 98.3% (Performane of Other Metrices was better)
[6]	Multiple Image	-	PCA, Proposed	-	Accu: 85.33±0.47

Keywords & Terms

- ❑Variance
- ❑Eigen vector
- ❑Eigen values
- ❑Orthogonal
- ❑Orthonormal
- ❑Co-variance
- ❑Correlation

Keywords & Terms

Variance:

Variance **(σ^2)** in statistics is a measurement of the spread between numbers in a data set. That is, it measures how far each number in the set is from the mean and therefore from every other number in the set.

Formula:

$$\sigma^2 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n}$$

Where: x^i = the i^{th} data point

$\bar{x}$ = the mean of all data points

n = the number of data points

Keywords & Terms

❑ Eigenvector & Values:

✓ **Eigenvectors and values** exist in pairs: every Eigenvector has a corresponding Eigenvalue.

✓ An **Eigenvector is a direction**, in the example the **Eigenvector is the direction of the line** (vertical, horizontal, 45 degrees etc.) . An **Eigenvalue is a number, telling us how much Variance there is in the data in that direction.**

✓ In the example the **Eigenvalue** is a number telling us how spread out the data is on the line.

✓ **The Eigenvector with the highest Eigenvalue is therefore the Principal Component.**

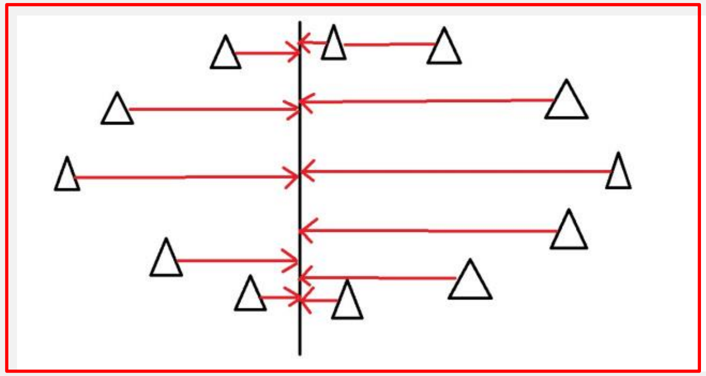


Fig. (a)

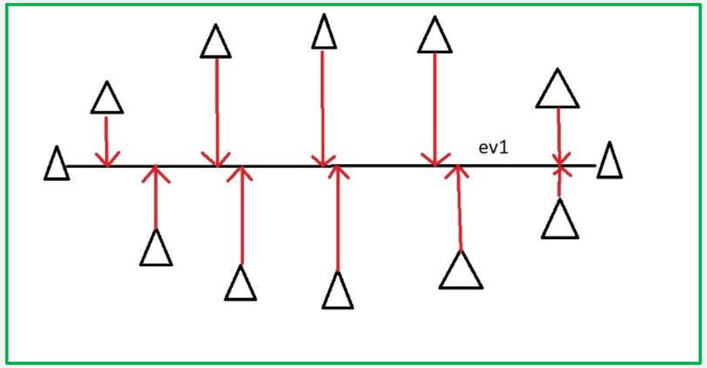


Fig. (b)

[1] https://georgemdallas.wordpress.com/2013/10/30/principal-component-analysis-4-dummies-eigenvectors-eigenvalues-and-dimension-reduction/

Keywords & Terms

Eigenvector & Values:

✓ If we take **Age and Height** of students, there are 2 variables, it's a 2-D data set, therefore there are 2 eigenvectors/values.

✓ Similarly If take **Age, height, Class** of students there are 3 variables, 3-D data set, so 3 eigenvectors/values.

✓ The Eigenvectors have to be able to span the whole x-y area, in order to do this (most effectively), the two directions need to be orthogonal (i.e. 90 degrees) to one another.

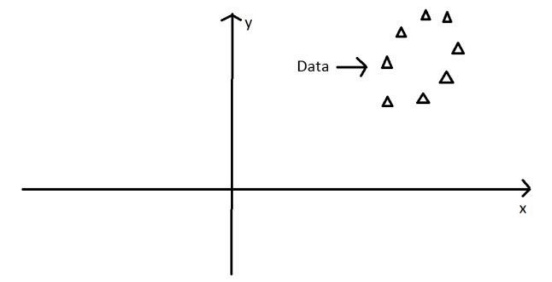
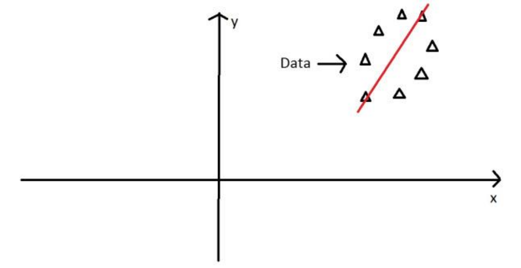


Fig. (c)

Fig. (d)



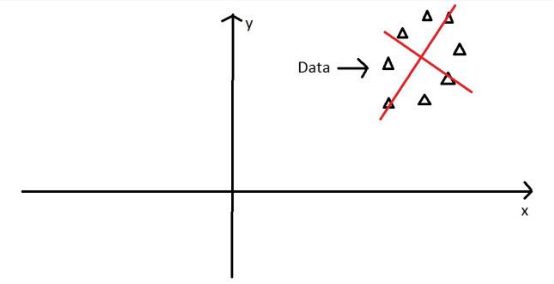


Fig. (e)

[1] https://georgemdallas.wordpress.com/2013/10/30/principal-component-analysis-4-dummies-eigenvectors-eigenvalues-and-dimension-reduction/

Keywords & Terms

❑ **Eigenvector & Values:**

✓ The **Eigenvectors** gives us much more useful axis to frame the data in. So, We can now re-frame the data in these new dimensions.

✓ These directions are where there is most variation found, and that is where there is more information.

✓ Another way we can say, **No variation of data means No information**. In this case the **Eigenvalue for that dimension would equal Zero**, because there is no variations.

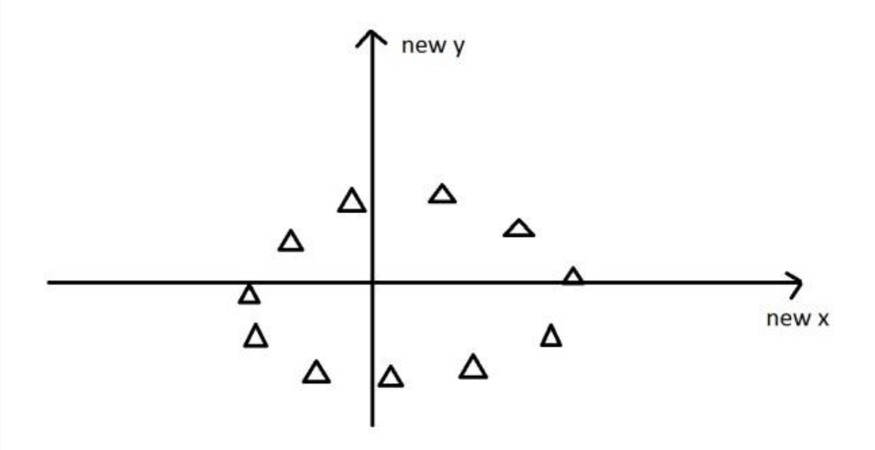


Fig. (f)

[1] https://georgemdallas.wordpress.com/2013/10/30/principal-component-analysis-4-dummies-eigenvectors-eigenvalues-and-dimension-reduction/

Keywords & Terms

- **Orthogonal:**

✓Two vectors are Orthogonal if they are perpendicular to each other. (i.e. the **dot product of the two vectors is 0**)

✓In Matrix, A **square matrix with real numbers** or elements is said to be an orthogonal matrix, **if its transpose is equal to its inverse matrix** or
when the **product** of **a square matrix** and its **transpose** gives an **identity matrix**, then the square matrix is known as an orthogonal matrix.

if, $\mathbf{A^T = A^{-1}}$ is satisfied, then, $\mathbf{A\,A^T = I}$

- **Orthonormal:**

➢A **set** of vectors is **orthonormal** if it is an orthogonal **set** having the property that every vector is a unit vector (a vector of magnitude 1).

[2] https://byjus.com/maths/orthogonal-matrix/#:~:text=If%20m%3Dn%2C%20which%20means,3%20rows%20and%203%20columns.

Keywords & Terms

❑ **<u>Co-Variance & Correlation:</u>**

✓ Both the terms measure the relationship and the dependency **between** two variables, suppose (x,y).

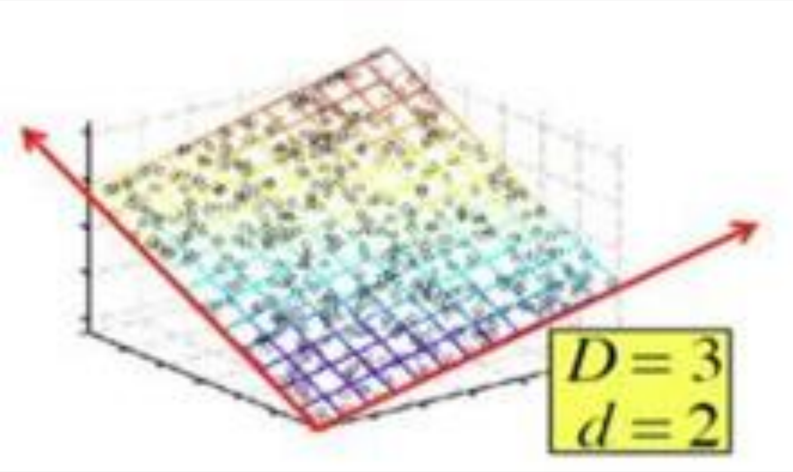
✓ "**Covariance**" indicates **the direction** of the linear relationship **between** variables.

✓ "**Correlation**" on the other hand measures **both the strength and direction** of the linear relationship **between** two variables.

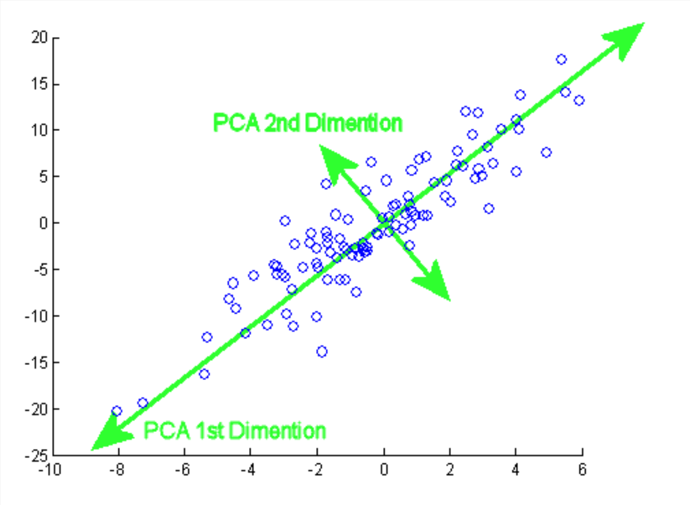
[3] https://towardsdatascience.com/let-us-understand-the-correlation-matrix-and-covariance-matrix-d42e6b643c22
[4] https://www.youtube.com/watch?v=xZ_z8KWkhXE

PCA

- PCA stands for Principal Component Analysis.
- PCA finds the principal components of data.
- PCA **finds the directions** where there is the most variance, the directions where the data is most spread out.



Fig(g). PCA-1

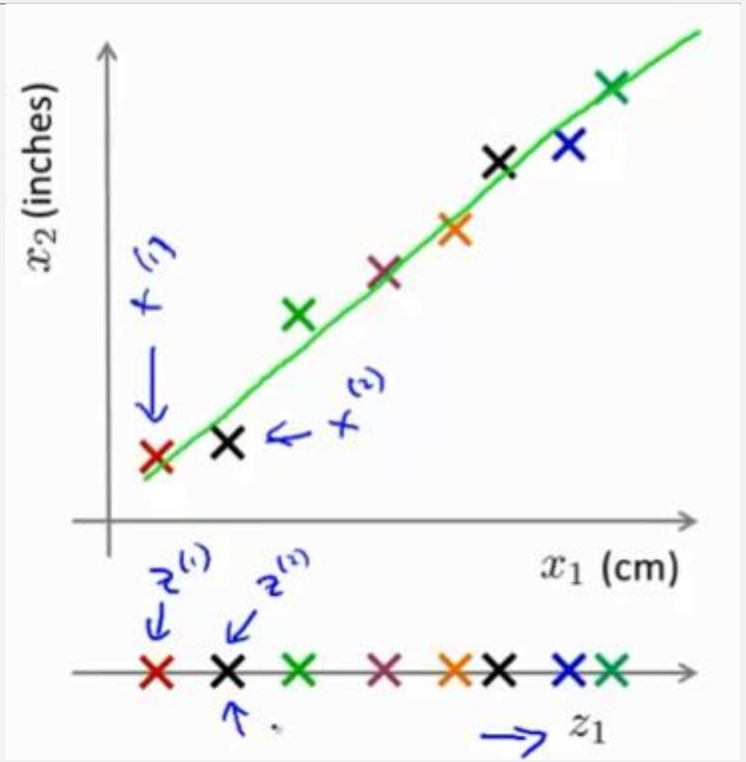


Fig(h). PCA-2

[5] https://dataconomy.com/2016/01/understanding-dimensionality-reduction/

PCA (Cont'd)

- Dimension Reduction tries to find a **Line or Plane** in which most of the data lies and project onto that line So that Every projected data to the line is smallest.



Fig(i). PCA (2D to 1D)

<u>2D to 1D</u>

$X^1 \in R^2 \;\text{–>}\; Z^1 \in R^1$

$X^1 \in R^2 \;\text{–>}\; Z^1 \in R^1$

………………..

$X^m \in R^2 \;\text{–>}\; Z^m \in R^1$

$$Z^i = \begin{bmatrix} Z_1^i \end{bmatrix} = \begin{bmatrix} X^i \end{bmatrix}$$

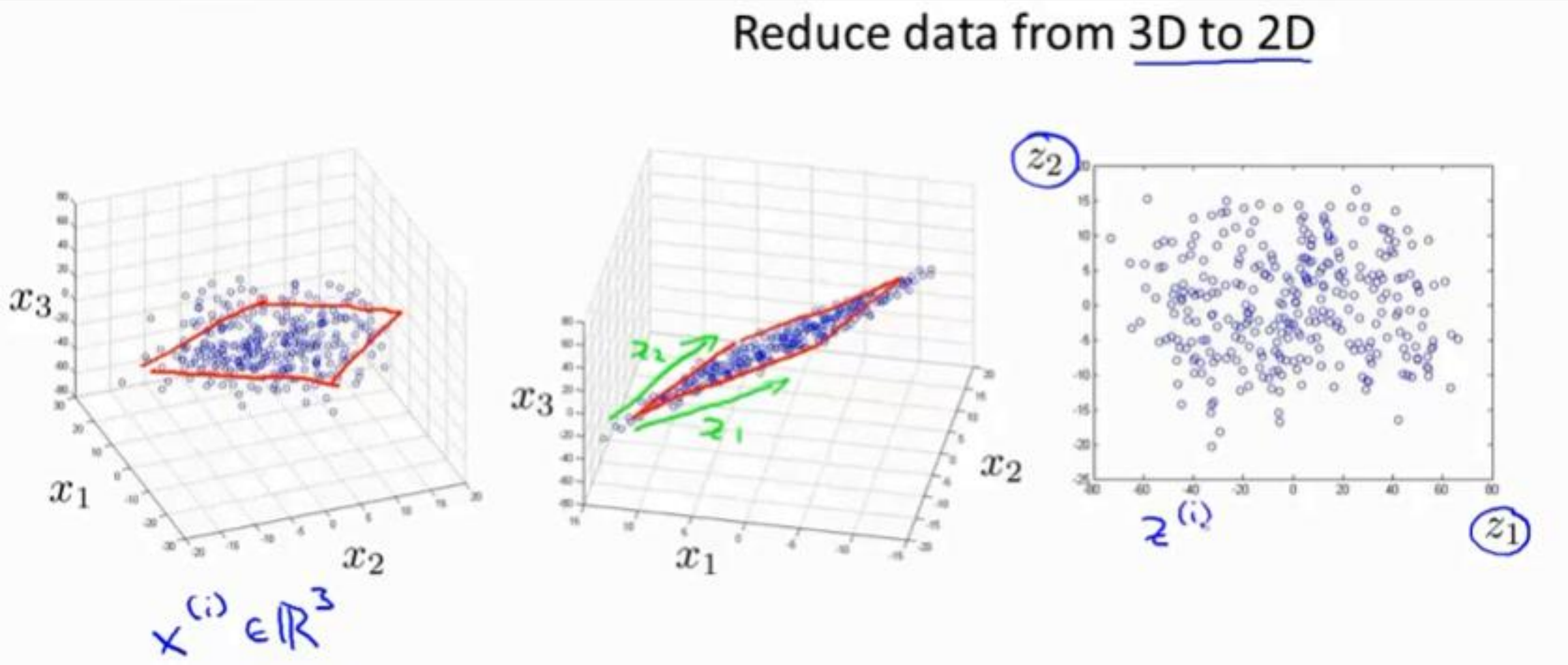
[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

PCA (Cont'd)

3D to 2D

$X^1 \in R^3 \ \to \ Z^1 \in R^2$
$X^1 \in R^3 \ \to \ Z^1 \in R^2$
…………………
$X^m \in R^3 \ \to \ Z^m \in R^2$

$$Z^i = \begin{bmatrix} Z_1^{\,i} \\ Z^i_{\,2} \end{bmatrix} = \begin{bmatrix} X^i \\ Y^i \end{bmatrix}$$



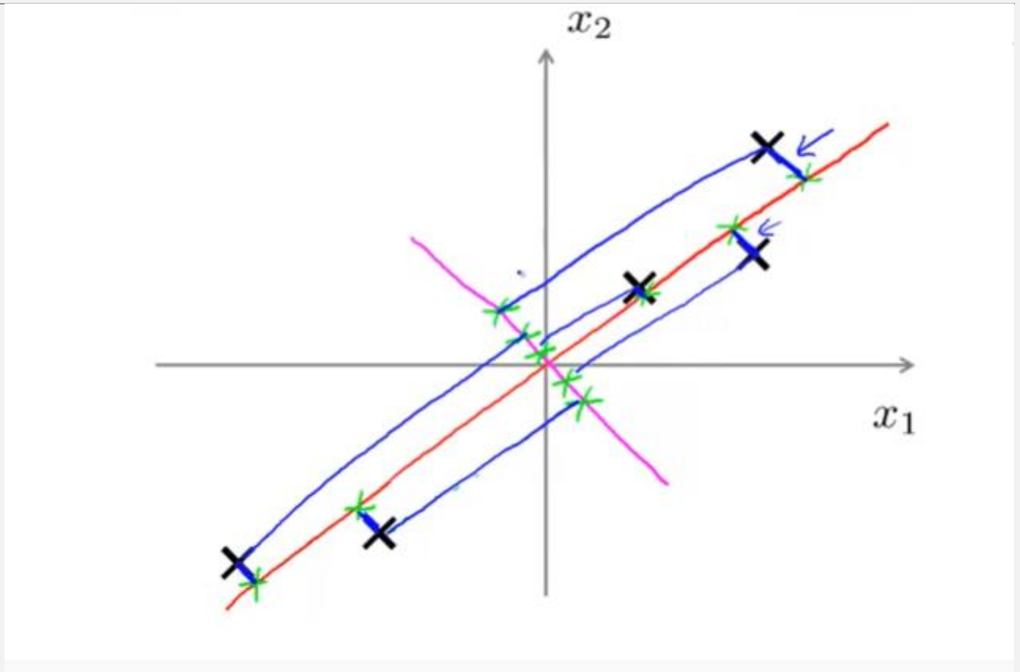
Fig(j). PCA (3D to 2D)

[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

PCA (Cont'd)

❑ Wrong-Right projection line

✓ The projection of every point on the Magenta line (Fig) is huge and also greater than the Red line.
So this is wrong line.



Fig(L). PCA (Wrong-Right projection line)

[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

Implementation of PCA

Algorithm:

1. Collect dataset

2. Preprocessing data (Feature Scaling/ Mean Normalization)

$\mu_j = (1/m) * \sum_{i=1}^{m} (x^i{}_j)$

Where, j = Dimension no. [1,2,3,…]
m = total no. of point on that dimension

Replace each x^i_j with $(x_j - \mu_j)$.

We scale data to have comparable range values. E.g if we have x_1 and x_2 features of data where,

x_1 = Size of house

x_2 = no. of bedroom

[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

Implementation of PCA (Cont'd)

- **Algorithm** (Reduced N Dim to K Dim):

3. Compute "Co-variance matrix"

Sigma, $\Sigma = (1/m) * \sum_{i=1}^{n} x^i \, (x^i)^T$ [Here, x^i (n*1) matrix]

In code, Sigma, $\Sigma = (1/m) * X^T * X$

4. Computer "Eigen Vectors" of matrix Sigma.

    [U,S,V] = svd(Sigma) or U = eig(Sigma)

5. U is a (n*n) matrix. Where to reduce in K dim we need to choose K column from U matrix that is (n*k) matrix.

U_{reduce} = [:, 1:k]

6. $Z_{new\_space} = U^T_{reduce} * X_{input\_set}$

[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

Implementation of PCA (Cont'd)

Algorithm (Choose 'K' the no. of Principle Component):

- PCA minimizes average projection Error.

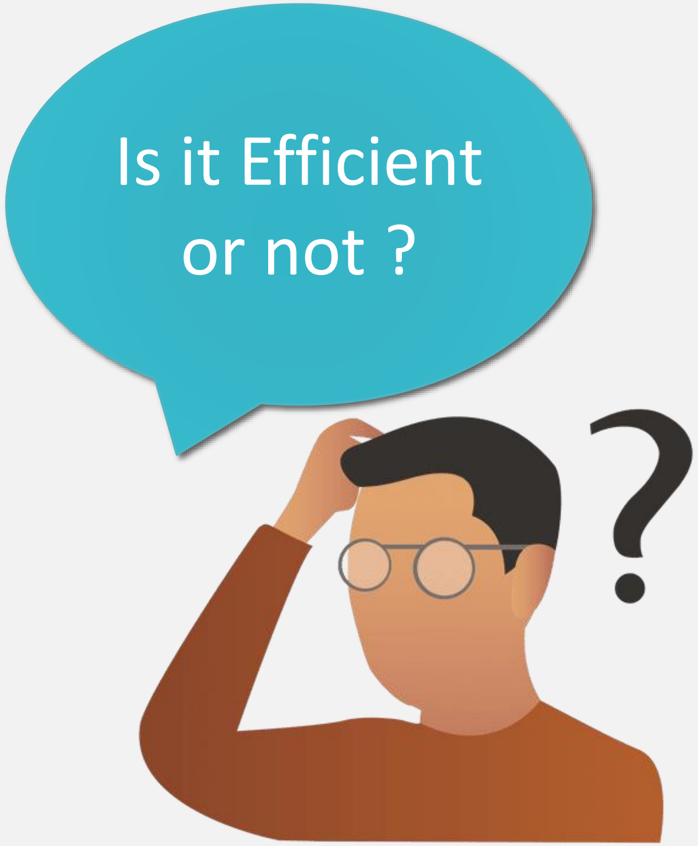
Eq 1. Avg. projection Error = (1/m) * $\sum_{i=1}^{m}(x^{i} - x^{i}{}_{approx})^2$

Eq 2. Total variation in the data = (1/m) * $\sum_{i=1}^{m}(x^{i})^2$

Now, we choose K to be smallest value so that,

$$\frac{Eq.1}{Eq.2} \leq 0.01\ (1\%)$$

So, that we can maintain 99% variance or our desired variance.



[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

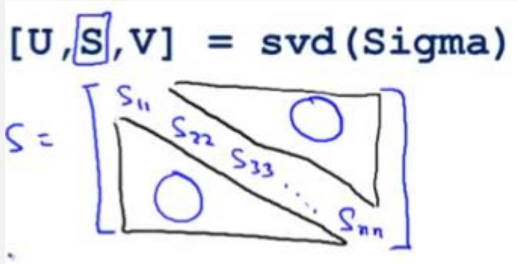
Implementation of PCA (Cont’d)

<u>Algorithm</u> <u>(Choose ‘K’ the no. of Principle Component):</u>

- Previous method was not Efficient to **choose ‘K’**

Compute, [U,S,V] = svd(sigma)

Where, S is a (r*r) diagonal matrix.



Now, for given ‘K’ we need to check if, $\frac{\sum_{i=1}^{k} s^{ii}}{\sum_{i=1}^{n} s^{i}} \geq 0.99\ (99\%\ variance\ retained)$

[6] Coursera Machine Learning (PCA & Feature| Dimension Reduction)

Implementation of PCA (code)

Input:

Dataset = np.array(
[[1, 1, 1, 0, 0],
[3, 3, 3, 0, 0],
[4, 4, 4, 0, 0],
[5, 5, 5, 0, 0],
[0, 2, 0, 4, 4],
[0, 0, 0, 5, 5],
[0, 1, 0, 2, 2]])

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler() #Standardize features by removing the mean and scaling to unit variance
scaler.fit(Dataset)
scaled_data=scaler.transform(Dataset) #making input arrays -> Tranpose

from sklearn.decomposition import PCA
pca = PCA(n_components = 2) #Here code suggests 25 is good #SCADI-paper took 53 features
pca.fit(scaled_data)
x_pca=pca.transform(scaled_data) #making input arrays -> reverse Tranpose = original

print('PCA on Dataset(7*5):\n\n',x_pca)
_plot_data(x_pca)
```

Output: (Reduced Dim.)

[[0.06153582 1.4876476] [-1.41476273 0.32780438] [-2.152912 -0.25211723] [-2.89106128 -0.83203884] [2.01452609 -0.69985245] [2.9755685 -0.71530184] [1.40710559 0.68385838]]

Shape: (7, 2)

SVD

- SVD stands for Singular Value Decomposition
- SVD is specific way to reduce features
- SVD is nothing more than decomposing vectors onto orthogonal axes

Implementation of SVD

❑ **Singular Value Decomposition:**

$A_{m\times n} = U_{m\times m}\ \Sigma_{m\times n}\ V^T_{n\times n}$

Where, $A_{m\times n}$ = input Matrix

$U_{m\times m}$ = Orthogonal Matrix

$\Sigma_{m\times n}$ = Diagonal Matrix

$V^T_{n\times n}$ = Orthogonal Matrix

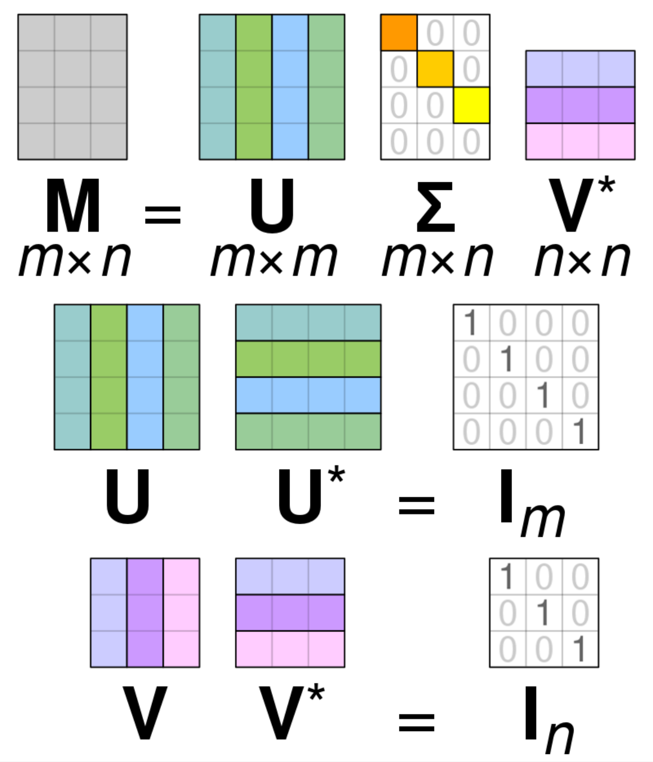


Fig (M). SVD computation

[7] https://en.wikipedia.org/wiki/Singular_value_decomposition#/media/File:Singular_value_decomposition_visualisation.svg

Implementation of SVD (Cont'd)

- **Singular Value Decomposition Dimension reduced to 'k' Dimension:**

$A_{m\times n} = U_{m\times m} \sum_{m\times n} V^{T}_{n\times n}$

To find reduced dim of A matrix 'n' dimension to 'k' dimension,

$\mathbf{Z_{reduced} = U[:,1:k] * \sum [1:k,1:k]}$

Implementation of SVD -1

Input:

Dataset = np.array(
[[1, 1, 1, 0, 0],
[3, 3, 3, 0, 0],
[4, 4, 4, 0, 0],
[5, 5, 5, 0, 0],
[0, 2, 0, 4, 4],
[0, 0, 0, 5, 5],
[0, 1, 0, 2, 2]])

```
from scipy.linalg import svd

n_elements = 2
U, S, V = svd(Dataset)
U = U[:,0:n_elements]
S = np.diag(S)
S = S[0:n_elements,0:n_elements]
rd = U.dot(S)
print('Original Data(7*5):\n\n',Dataset)
print('\nShort Hand SVD(7*2):\n\n',rd)
```

Output: (Reduced Dim.)

Short Hand SVD **(7*2)**:

[[-1.71737671 -0.22451218]
[-5.15213013 -0.67353654]
[-6.86950685 -0.89804872]
[-8.58688356 -1.12256089]
[-1.9067881 5.62055093] [-
0.90133537 6.9537622] [-
0.95339405 2.81027546]]

Implementation of SVD-2

Input:
Dataset = np.array(
[[1, 1, 1, 0, 0],
[3, 3, 3, 0, 0],
[4, 4, 4, 0, 0],
[5, 5, 5, 0, 0],
[0, 2, 0, 4, 4],
[0, 0, 0, 5, 5],
[0, 1, 0, 2, 2]])

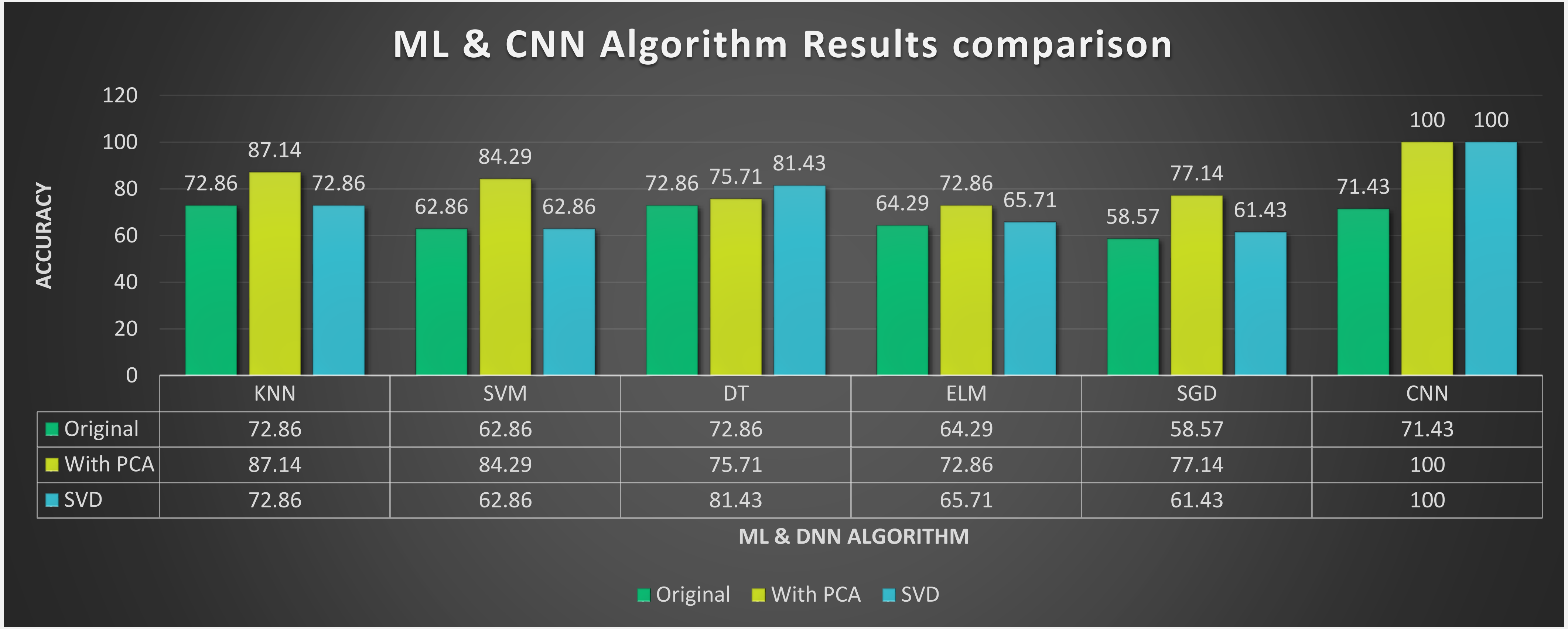
```python
from sklearn.decomposition import TruncatedSVD

#Dataset = A
# svd
svd = TruncatedSVD(n_components=2)
svd.fit(Dataset)
dataset_svd = svd.transform(Dataset)
print(dataset_svd.shape)
print(dataset_svd)

_plot_data(dataset_svd)
```

Output: (Reduced Dim.)

[[1.71737671 -0.22451218] [5.15213013 -0.67353654] [6.86950685 -0.89804872] [8.58688356 -1.12256089] [1.9067881 5.62055093] [0.90133537 6.9537622] [0.95339405 2.81027546]]

Shape: (7, 2)

Result Comparison (SVD/PCA vs Original Data)



	KNN	SVM	DT	ELM	SGD	CNN
Original	72.86	62.86	72.86	64.29	58.57	71.43
With PCA	87.14	84.29	75.71	72.86	77.14	100
SVD	72.86	62.86	81.43	65.71	61.43	100

Disadvantages of Dimensionality Reduction

- It may lead to some amount of **data loss**.
- PCA tends to find linear correlations between variables, which is sometimes undesirable.
- PCA **fails in cases where mean and covariance are not enough** to define datasets.
- We **may not know how many principal components to keep-** in practice, some thumb rules are applied.

THE END