Date: 10/06/2023

PCA: Used in feature engineering. Dimensionality reduction

1. Data ingestion
2. EDA
3. Preprocessing ⇒ feature engineering
4. Model building
5. Model evalution

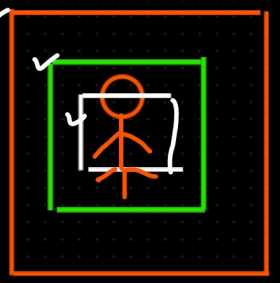
Dimension Reduction → Reducing the dimension → Feature

- Feature Selection
- Feature extraction

Acc to curse of dim. "As the number of features or dimensions grows, the amount of data we need to generalize accurately grows exponentially."

Follow this beautiful article explaining this topic is most beautiful manner with set of examples:

https://medium.com/flutter-community/curse-of-dimensionality-an-intuitive-and-practical-explanation-with-examples-399af3e38e70#:~:text=Example%202%3A,dimension%20they%20can%20move%20in.



Curse of Dimensionality :- Data (Row X column)

Eg: Say, you dropped a coin on a 100-meter line. How do you find it? Simple, just walk on the line and search. But what if it's 100 x 100 sq. m. field? It's already getting tough, trying to search a (roughly) football ground for a single coin. But what if it's 100 x 100 x 100 cu.m space?! You know, football ground now has thirty-story height. Good luck finding a coin there! That, in essence, is "curse of dimensionality". In order to get rid of this using dimensionality reduction approaches like PCA we are basically trying reducing the dimensions in such a way that resultant (vector/set of dimensions) have essence of all the dimensions making it easier for ML algo to comprehend as volume or traversal space decreases.

$f_1 \dots f_{10}$ Y → M_1 ⇒ 75%.
$f_1 \dots f_{100}$ Y → M_2 ⇒ 80%.
$f_1 \dots f_{1000}$ Y → M_3 ⇒ 72%.

less optimal over ⇒ overfitting ⇒ sparsity

Underfitting

① Feature Selection ⇒ Subset of the feature

$f_1 \dots f_{100} \Rightarrow$ Model $\Rightarrow$ 75%

50, 60, 70, 80

$\equiv \{1, 2, 3, 4, 5, 6, 7\}$

Set theory

$\{1, 2, 3\}$

Below are the methods that can be used to perform feature selection:

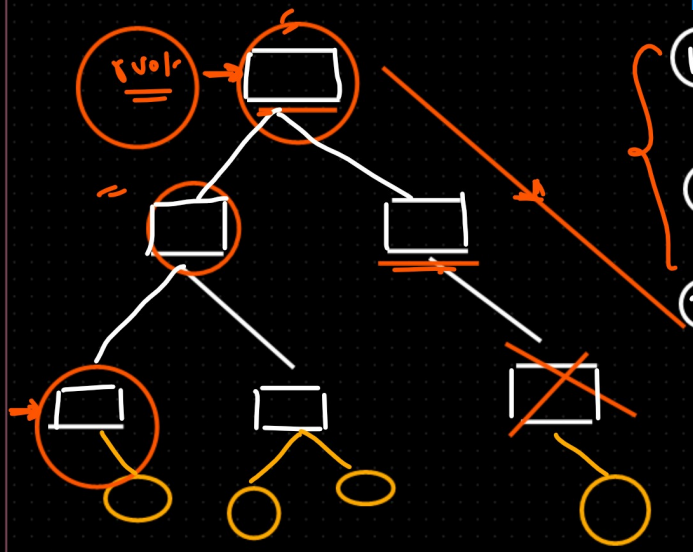
Scikit learn

① Filter method ⇒ Corr (correlation)

Info gain → feature importance

② Wrapper method.

③ Lasso (Regularization based method)



In DT root level has high importance as it has largest info gain

Google feature importance method

= Wafer fault Detection

= root

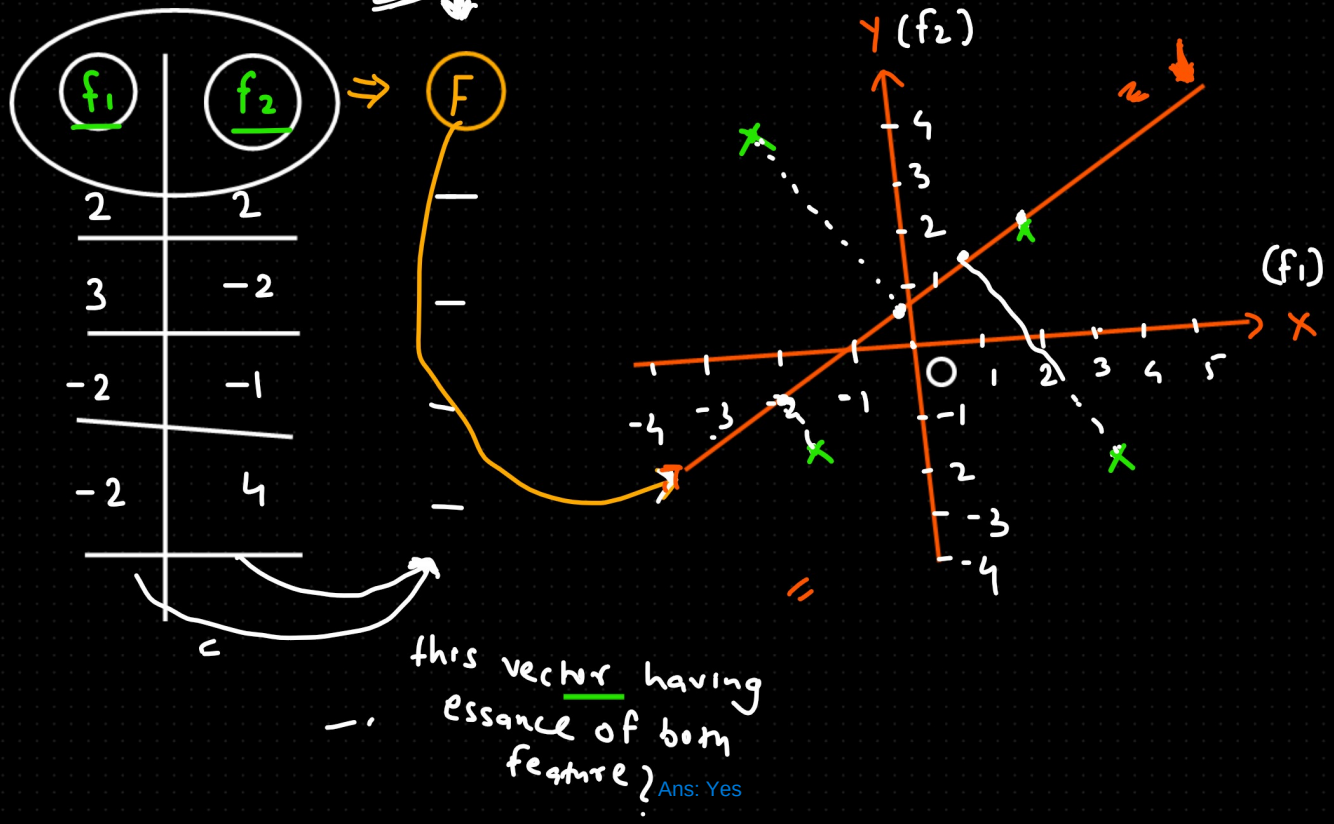
= feature Selection

VV Imp:
Google skikit learn feature selection.
Using this we can implement different feature selection approaches.
https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_selection

Can follow this blog as well for feature for feature selection methods available in sklearn lib:
https://towardsdatascience.com/5-feature-selection-method-from-scikit-learn-you-should-know-ed4d116e4172

Please note that having more and more number of features will definitely help in building a model having high accuracy. But beyond some threshold on increasing number of features further will result into curse of dimensionality which will increase the volume and sparsity. To get rid of this we reduce number of features by taking a subset of features which are important(feature selection) or by reducing the dimension in such a way that the resultant reduced dimensions have essence of whole dataset(feature extraction)

Dimension Reduction | Feature extraction | Feature transformation



In this example through feature extraction we are reducing 2 dimension data(F1 and F2) into 1 dimension by projecting them on vector F such that this vector F have essence of both the features(F1 and F2)

Dimension reduction or feature extraction can be used to:
1. To get rid off the curse of dimensionality problem.
2. To visualize higher dimension dataset. For Eg; For visualizing dataset with 4 features we can first convert into 3 dimension using dimensionality reduction approaches such as PCA followed by visualizing resultant 3D dataset.

Following are some of the latest techniques used for dimensionality reduction or feature extraction:

1. PCA
2. t-SNE
3. LDA

} State of art (State of art means latest)

1. Standardized the Data
2. Covariance matrix
3. Eigen value and ergen vector
4. Principle component

PCA

These are the 4 steps in which PCA is implemented

Step 1:

1 Scaling of the data

Standardization of data can be done using any of the below approaches:

- Min-Max ⇒ [0-1]
- Standard Scalar
- Z-Score

This is the general formula for Min Max scaling where X represents the data point. Also, we may customize this formula according to our need.

$$\text{Min-Max} = \frac{X - \text{Min}}{\text{Max} - \text{Min}}$$

In min-max based approach we suppress the dataset under min and max range such that all the data points after scaling lie in the interval [0,1]

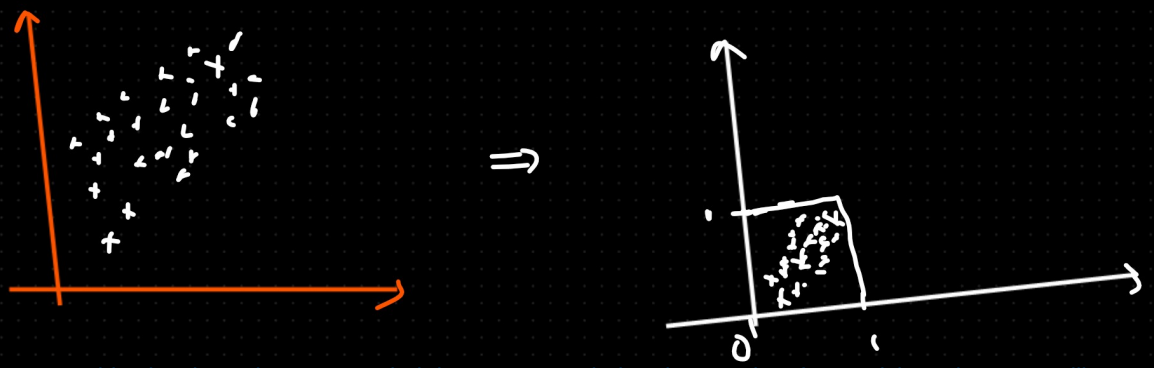
$[a_1, a_2, a_3, a_4, \ldots, a_n]$

- a_1 → Min
- a_n → Max

In this example
a1 == min
an == max

scaling min data point a1 $= \frac{\text{Min} - \text{Min}}{\text{Max} - \text{Min}} = 0$

scaling max datapoint an $= \frac{\text{Max} - \text{Min}}{\text{Max} - \text{Min}} = 1$



This plot shows how a unscaled dataset get scaled under 0 and 1 after applying min max scalling

SND

Standardization => $\frac{X-\mu}{\sigma}$

Z-Score

μ = mean

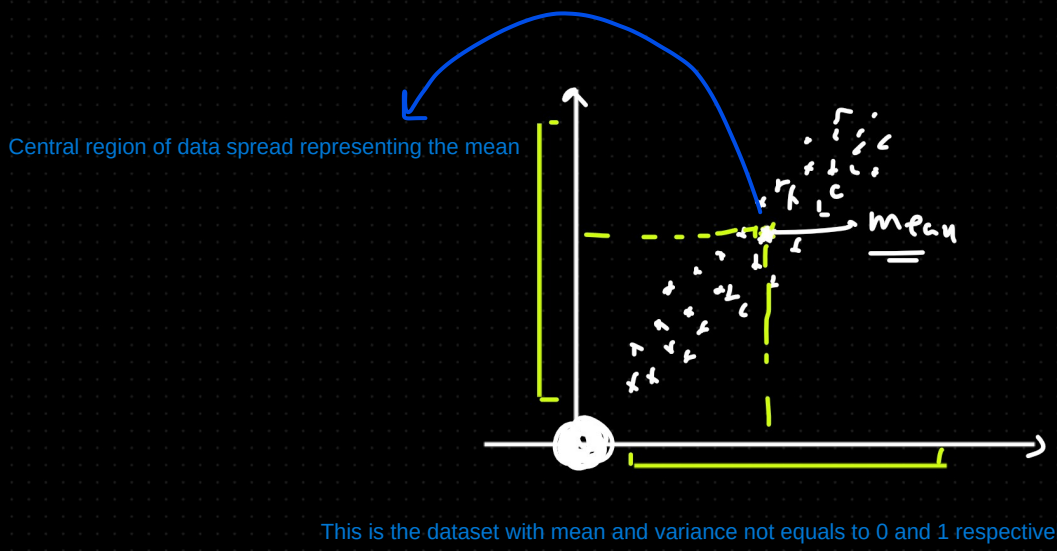
σ = std

⇓

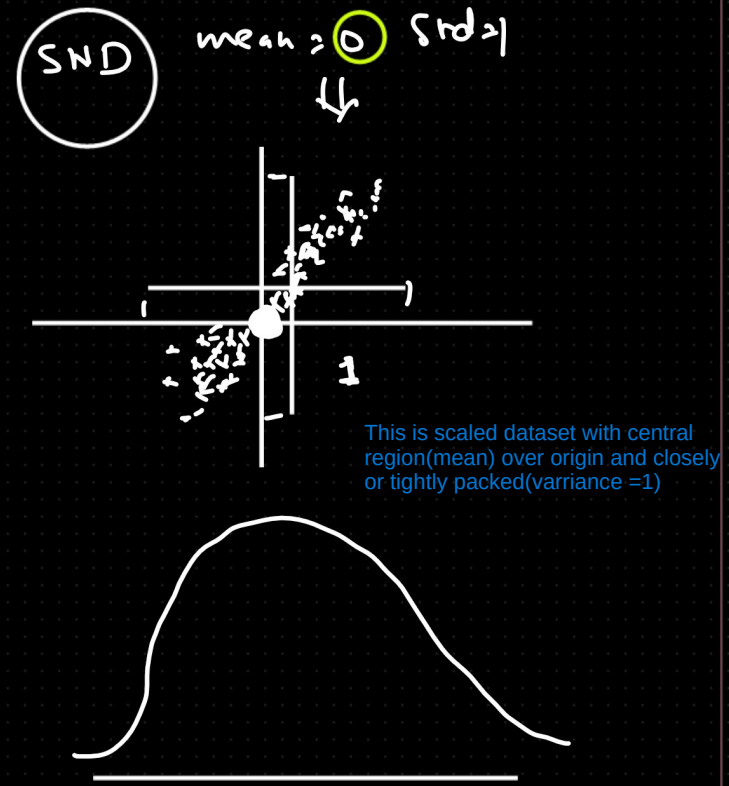
In Standard scaler or Z score based approach we perform standardization on the dataset(with mean and variance not equals to 0 and 1 respectively) in such a way that the resultant standardized dataset have mean equal to zero and standard deviation =1

Data → Standardization => Standard Data

$\mu = 0$ $\sigma = 1$



This is the dataset with mean and variance not equals to 0 and 1 respectively



This is scaled dataset with central region(mean) over origin and closely or tightly packed(varriance =1)

Step 2:

Variance =>

Follow this flow to remember the Variance formula in easy way

2, 8, 10, 12, 6, 5, 13, 14

↓

→ Mean

↓

Deviation => $(\text{Mean} - \text{DP})^2$

↓

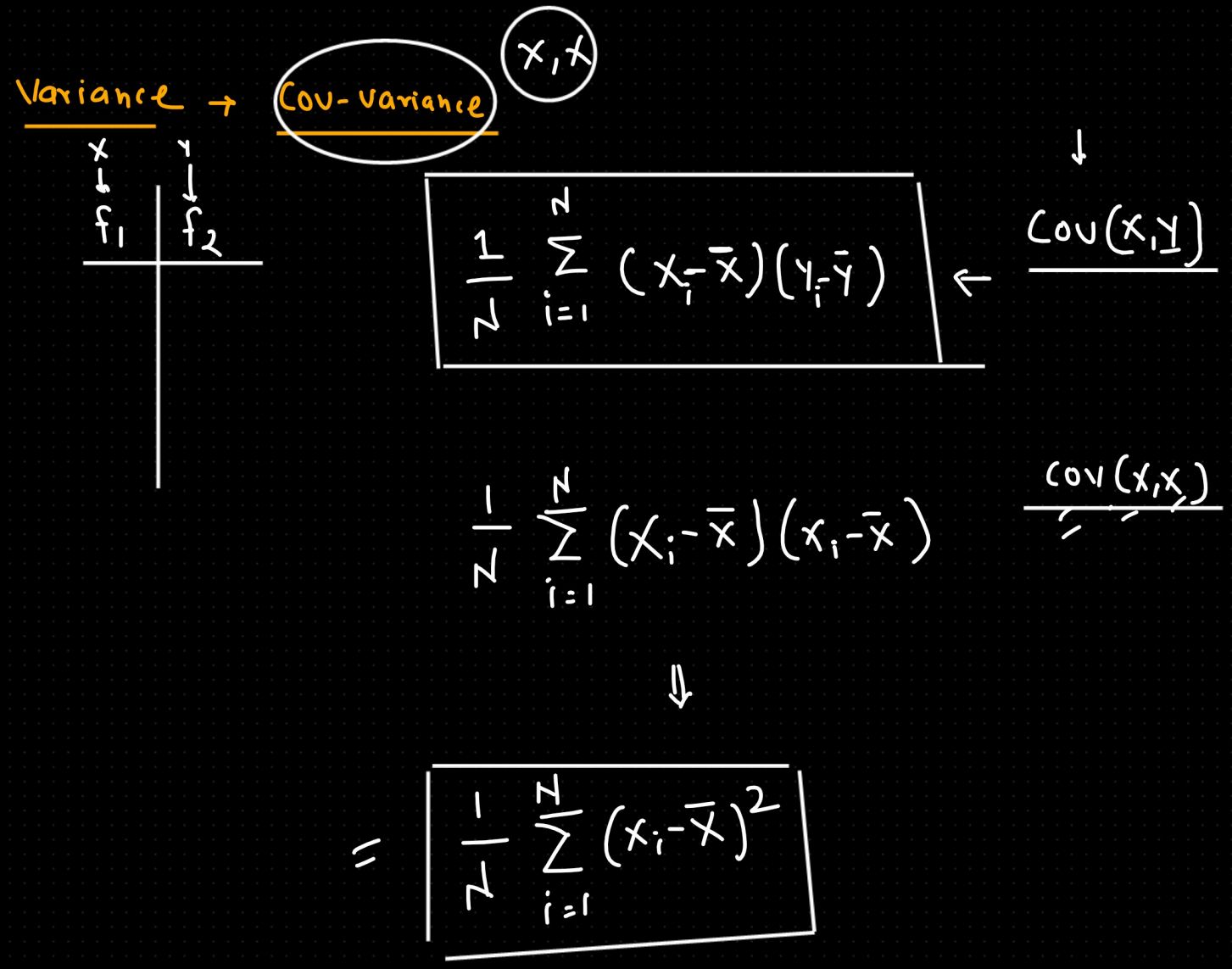
Mean of deviation

Variance is the spread of data around the mean

Spread of the data around the mean

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

Formula of variance



Relationship b/w variance and covariance w.r.t a random variable X
Variance of X == Co variance of(X,X)

Covariance is commutative in nature

$$Cov(x,y) = Cov(y,x)$$

x	y

$$= \begin{bmatrix} Cov(x,x) & Cov(x,y) \\ cov(y,x) & Cov(y,y) \end{bmatrix} \equiv \Rightarrow \text{Covariance matrix of } x,y \text{ feature}$$

1 Std mean : 0

$$\Rightarrow \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})$$

$\bar{x}$: mean, $\bar{y}$: mean

Why we do Standarization?

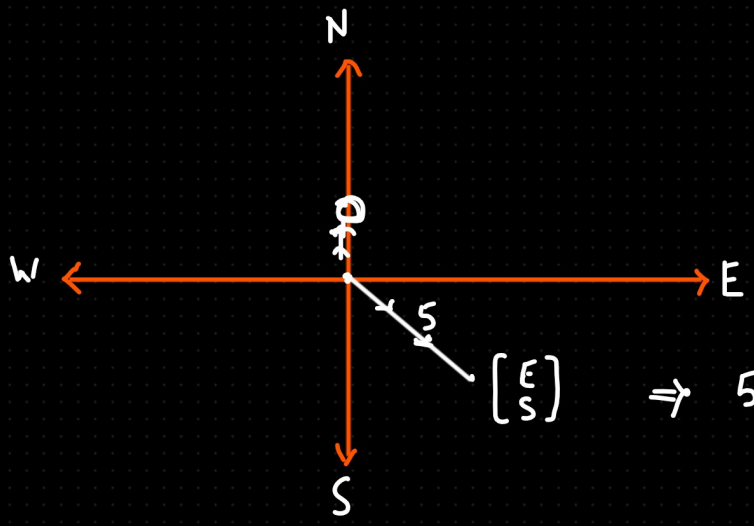
Since in the Standarization step we are making the mean 0. So, our cov / Var formula is also getting updated(since mean =0) that is less calculation. This is why we are doing the Standarization which is kind of optimizing the model learning phase.

$$\Rightarrow \frac{\frac{1}{N} \sum_{i=1}^{N} (x_i)(y_i)}{N}$$

$$\Rightarrow \frac{x_1y_1 + x_2y_2 + x_3y_3 \cdots x_ny_n}{N}$$

Considering X and Y are 2 vectors then the numerator is dot product of these 2 vectors.

$\Rightarrow$ Dot Product of two vector

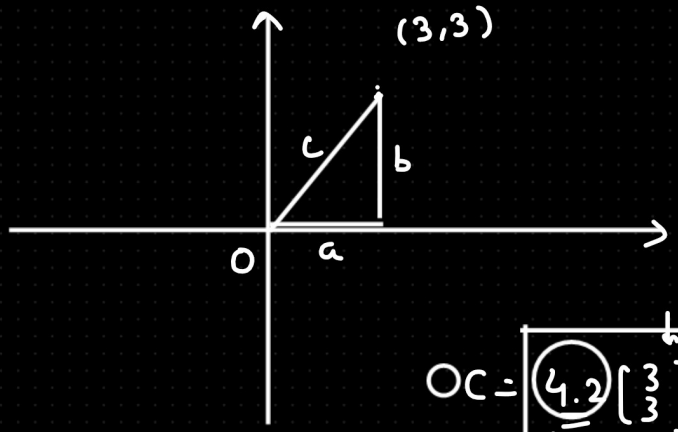


Scalar value (Magnitude)

Vector (Magnitude + Direction)

≡ 5 km / 1 hr

Representation of a person moving 5 Km in East - South direction.
This is vector quantity.



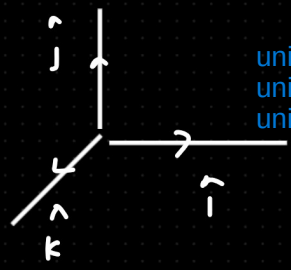
$c^2 = a^2 + b^2$

$c = \sqrt{a^2+b^2} = \sqrt{3^2+3^2}$

$= \sqrt{18} = 4.2$

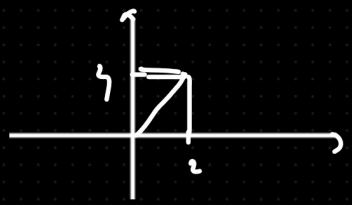
$OC = 4.2 \begin{bmatrix} 3 \\ 3 \end{bmatrix}$

$3\hat{i} + 3\hat{j}$



unit vector along the x axis i^
unit vector along the y axis j^
unit vector along the z axis k^

$V = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$



$\sqrt{2^2+4^2}$

$= \sqrt{4+16} = \sqrt{20}$

Step 3:

Eigen value & eigen vector

We are using cov matrix since it is square matrix that will be used for calculating eigen vector

$$AV = \lambda V$$

A = Square Matrix

V = Vector

λ = Scalar value

Please note matrix can have multiple rows and columns but vector can have multiple rows and only 1 column.

if we multiplying a square Matrix (A) with vector (V) and if we get a new vector (λV) that is λ times of vector V, then vector V is called eigen vector of Matrix A

and lamda is called the eigen value

Also,

= if we multiply square matrix (A) by vector (V) the new vector (λV) does not change the direction. than only it is eigen vector)

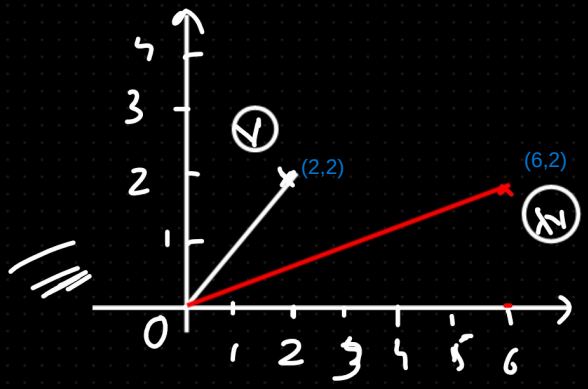
Example 1:

$$A = \begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix}$$

$$V = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$AXV = \lambda XV$$

$$\begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} X \begin{bmatrix} 2 \\ 2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1X2 + 2X2 \\ 1X2 + 2X0 \end{bmatrix} \Rightarrow \begin{bmatrix} 2+4 \\ 2+0 \end{bmatrix} \Rightarrow \begin{bmatrix} 6 \\ 2 \end{bmatrix}$$



From above plot we can say that both the vectors are having different direction since they are not coinciding over each other. If they were coinciding over each other or on the same line then we would have said that both are having the same direction but different magnitude where respectively direction would have represented eigne vector whereas, respectively magnitude would have represented the eigen value.

$$AV = \begin{bmatrix} 6 \\ 2 \end{bmatrix} = \lambda V$$

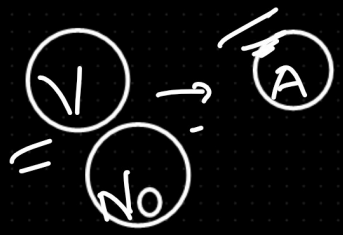
$$\begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 6 \\ 2 \end{bmatrix}$$

unit movement along x axis = i^ = 6

unit movement along y axis = j^ = 2

$$\begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} = 2 \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

$$A\,V = \lambda\,V$$



It is clear that direction has changer after A is multiplied with V. It both were equal then we would have said that same direction. Same can be also visualized through graphical representation

In this example AV=lamda*V but since direction is changing V cannot be considered as Eigne value and Lamda cannot be considered as Eigen value.

Example 2:

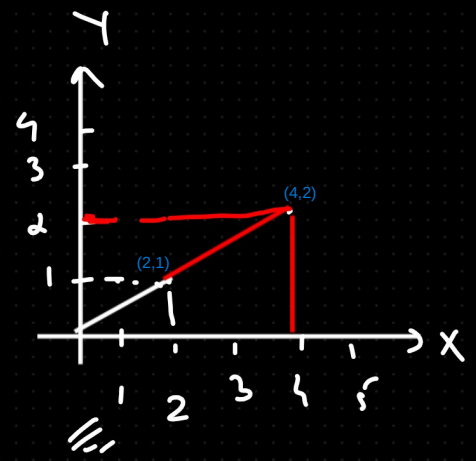
$$\begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1\times2+2\times1 \\ 1\times2+0\times1 \end{bmatrix} = \begin{bmatrix} 2+2 \\ 2+0 \end{bmatrix} \Rightarrow \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

$\lambda 1$

$$A \times V = \lambda \times V$$

$$\begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

λ = Eigen value



$$\Rightarrow \begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = 2 \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Here since both the vectors are coinciding over each other or are on the same line so we will say that both are having the same direction.

Here this vector is eigen vector of matrix A

Principle Component

In this example since AV=lamda*V and direction is same for vector V in the process so V will act as Eigen vector and Lamda will be considered as Eigen value.

Eigen value tells us how much the eigen vector changes in size when we multiply it(Eigen vector) with square matrix

$A \rightarrow v$

λv

$$Av = \lambda v$$

$$Av - \lambda v = 0$$

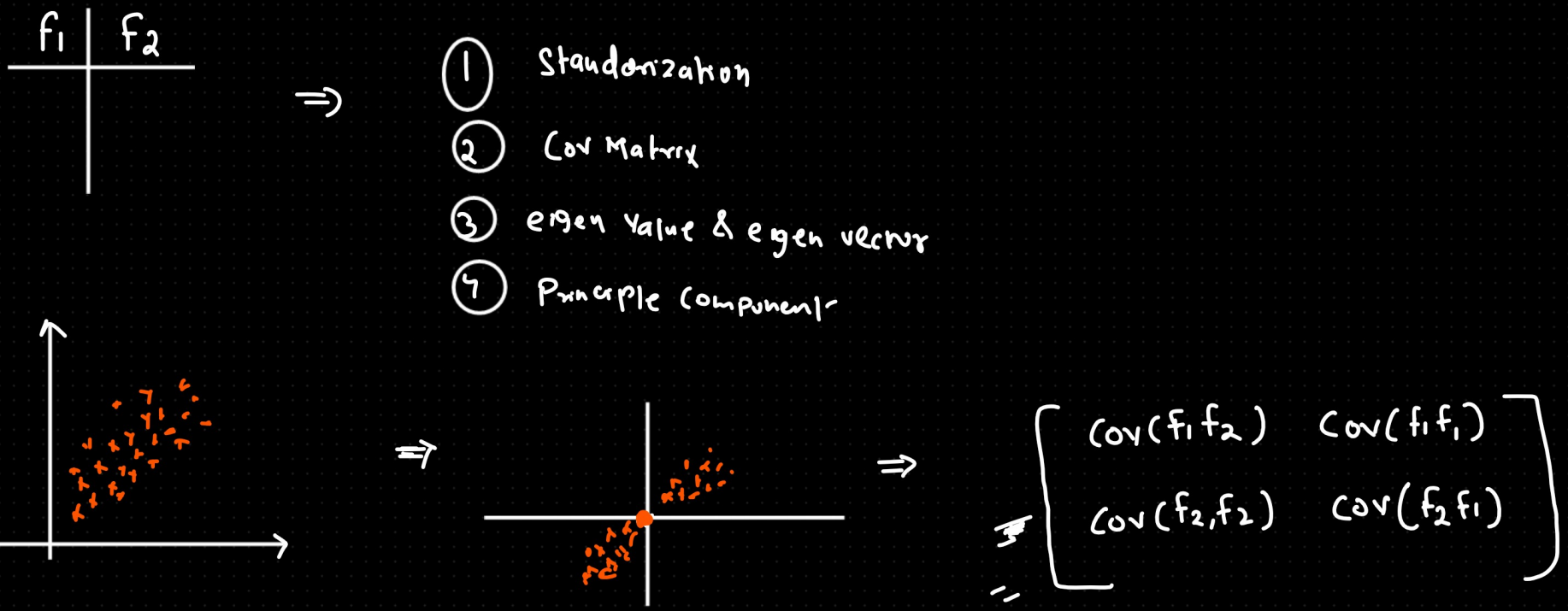
$$(A - \lambda I)v = 0$$

$$\text{Det}\,|A - \lambda I|$$

I = Identity matrix

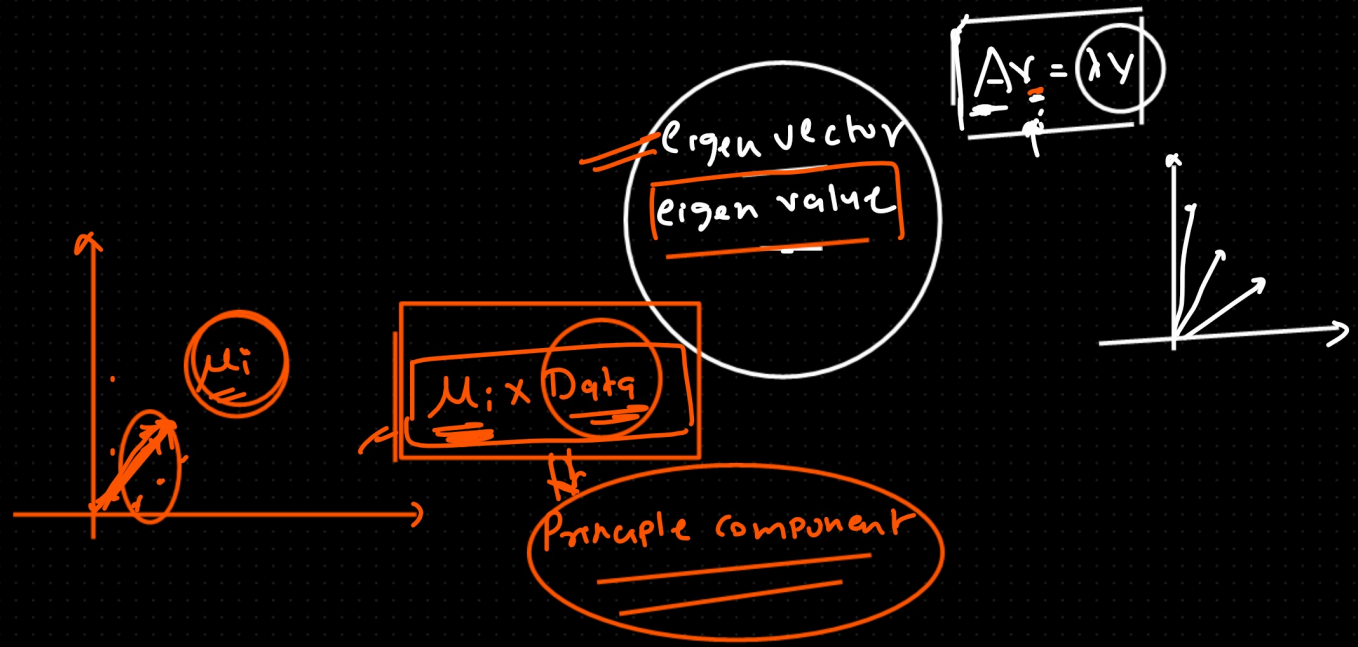
We can say that this is the Eigen equation

Please note that here we have multiplied scaler lambda with Identity matrix so that to convert it into matrix, later which can undergo like term subtraction with square covariance matrix A



Step 4:

ui is the eigen vector which when multiplied with the datapoints(obtained after scaling) gives principal component which will project all the datapoints on the same vector resulting in dimensionality reduction.



Important: Refer to implementation ipynb for better understanding

While training the model we will be providing the PCA transformed dataset and while prediction we will first of all tranform new data on which prediction needs to be made into PCA transformed data and then use the existing learning (trained over PCA transfored train dataset) to carry out the prediction.

Means for any new data set that is not the part of the test split of exisiting dataset then we need to prepare a data pipeline that we convert this new data in PCA transformed data which will be then passed to the model for carrying out the prediction.

Again please note PCA is applicable only for numerical colums and not for categorical columns. Even if we are transforming categorical col into numeric using then it will generate sparse matrix(0s and 1s) will can not be taken forward for PCA

Apply PCA on numerical col and encoding on categorical col and combine to generate the final dataset.

In Feature Selection we are taking the subset of the features whereas, in Feature Extraction we are producing vector having essence of whole dataset with all the features