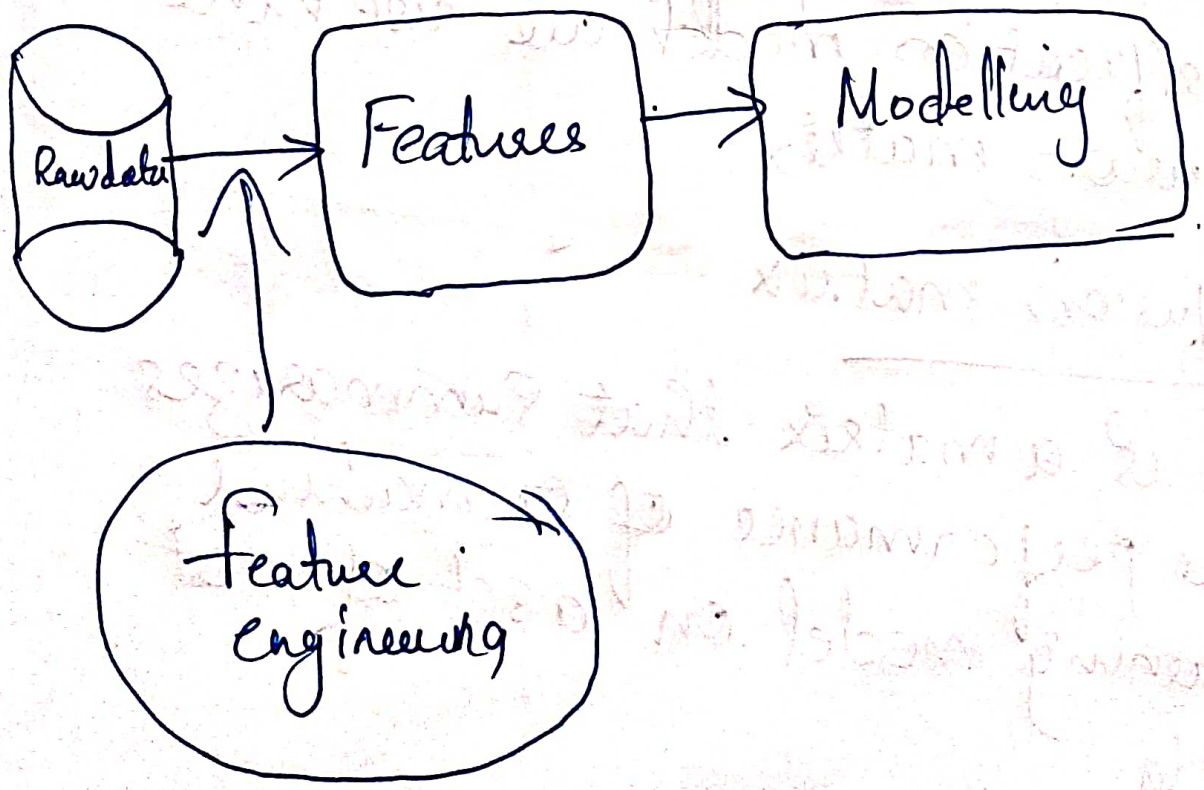
Module-1.

→ Feature engineering is the process of transforming raw data into features that are suitable for machine learning models.

⇒ In other words, it is the process of selecting, extracting and transforming the most relevant features from the available data to build more accurate and efficient m/c learning models.

→ Feature engineering is the process to create feature / extract the feature from existing features by domain knowledge to increase the performance of machine learning model.

→ In data science, the performance of model is depending on data preprocessing and data handling. Suppose if we build a model without handling data, we got an accuracy of around 70%. By applying the Feature engineering on the same model, there is a chance to increase the performance from 70% to more. Simply, by using Feature engineering we improve the performance of the model.



Evaluating Machine learning algms & models

→ Evaluating the statistical or machine learning model build is essential to check its quality
→ Evaluation metrics are used to measure the performance of the model.
⇒ Different types of evaluation metrics are used based on the problem we are solving.

Evaluation metrics for classification

To measure the performance of the classification model we use the Confusion matrix

Confusion matrix

It is a matrix that summarizes the performance of a machine learning model on a set of test data.

problem definition - Confusion Matrix

a) Consider the confusion matrix given below for a binary classifier predicting the presence of a disease.

- The classifier made a total of 150 predictions. Out of those 150 cases, the classifier predicted "yes" 100 times, and "no" 50 times.
- In reality 100 patients in the sample have the disease, and 50 patients do not.

	predicted No	predicted yes
Actual No	TN = 45	FP = 5
Actual Yes	FN = 5	TP = 95

- Accuracy :- Overall, how often is the classifier correct,

Accuracy :- $$\frac{TN+TP}{TN+FP+FN+TP}$$

$$= \frac{45+95}{150} = 93.33\%$$

Misclassification rate

:→ Overall, how often the classification was done wrong.

Misclassification Rate $$= \frac{FN+FP}{TN+FP+FN+TP}$$

$$= \frac{5+5}{150} = 6.67\%$$

True positive Rate no of yes exs classified as yes

When its actually yes, how often does it predict yes?

→ It is also known as "Sensitivity" or "Recall".

True positive rate = $\frac{TP}{\text{Actual yes}}$

$= \frac{95}{100} = 95\%$

False positive Rate :-

When its actually no, how often does it predict yes?

False positive rate = $\frac{FP}{\text{Actual No}}$

$= \frac{5}{50} = 10\%$

* True Negative Rate :- When it's actually no, how often does it predict no?

→ Also known as "Specificity"

ie; True Negative rate = $\frac{TN}{\text{Actual No}}$

$= \frac{45}{50} = 90\%$

* precision :- When it predicts yes, how often is it correct?

$\text{precision} = \frac{TP}{\text{Predicted yes}} = \frac{95}{100} = 95\%$

* prevalence :- How often does the yes condition actually occur in our sample?

$\text{prevalence} = \frac{\text{Actual yes}}{\text{Total}}$

$= \frac{100}{150} = 66.67\%$

Q) For the given confusion Matrix.

True class	predicted class		
	1	2	3
1	8	2	0
2	1	9	0
3	1	2	7

First we need to convert the multi-class matrix into binary matrix.

- Let the True positive (TP) = no. of positive examples correctly predicted.
- False negative (FN) = no. of positive examples wrongly predicted as negative.
- False positive (FP) = no. of negative examples wrongly predicted as positive
- True negative (TN) = no. of negative examples correctly predicted.

True class	predicted class	
	1-yes	2&3-No
1-yes	TP=8	FN=2
2&3-No	FP-2	TN-18

True class	predicted class	
	2-yes	1&3-No
2-yes	TP-9	FN-1
1&3-No	FP-4	TN-16

True class	Predicted class	
	3-yes	1&2-No
3-yes	TP=7	FN-3
1&2-No	FP=0	TN=20

$$\text{Accuracy (class-1)} = \frac{TP + TN}{TP + FN + FP + TN}$$

$$\text{Accuracy (class-1)} = \frac{8+18}{8+2+2+18} = \frac{26}{30} = 0.87$$

$$\text{Accuracy (class-2)} = \frac{TP + TN}{TP + FN + FP + TN}$$

$$\text{Accuracy (class-2)} = \frac{9+16}{9+1+4+16} = \frac{25}{30} = 0.83$$

$$\text{Accuracy (class-3)} = \frac{TP + TN}{TP + FN + FP + TN}$$

$$= \frac{7+20}{7+3+0+20} = \frac{27}{30} = 0.9$$

- $\text{precision (class-1)} = \frac{TP}{TP + FP} = \frac{8}{8+2} = \frac{8}{10} = 0.8$

~~Precision (class)~~

- $\text{Recall (class-1)} = \frac{TP}{TP + FN} = \frac{8}{8+2} = \frac{8}{10} = 0.8$

- F-1 score (class-1) $= \frac{2 * \text{precision} * \text{Recall}}{\text{precision} + \text{Recall}}$

$= \frac{2 \times 0.8 \times 0.8}{0.8 + 0.8}$

$= \frac{1.28}{1.6} = 0.8$

- precision (class-2) $= \frac{TP}{TP+FP} = \frac{9}{9+4} = \frac{9}{13} = 0.69$

- Recall (class-2) $= \frac{TP}{TP+FN} = \frac{9}{9+1} = \frac{9}{10} = 0.9$

- F-1 score (class-2) $= \frac{2 * \text{precision} * \text{Recall}}{\text{precision} + \text{Recall}}$

$= \frac{2 * 0.69 * 0.9}{0.69 + 0.9}$

$= \frac{1.242}{1.59} = 0.78$

- Precision (class-3) $= \frac{TP}{TP+FP} = \frac{7}{7+0} = \frac{7}{7} = 1.0$

- Recall (class-3) $= \frac{TP}{TP+FN} = \frac{7}{7+3} = \frac{7}{10} = 0.7$

- F-1 score (class-3) $= \frac{2 * \text{precision} * \text{Recall}}{\text{precision} + \text{Recall}}$

F-1 score (class-3) $= \frac{2 * 1.0 * 0.7}{1.0 + 0.7} = \frac{1.242}{1.59}$

$= 0.82$

Weighted F1-score $= \sum_{i=1}^{N} w_i * F_1 \text{score}_i$

Weighted F1 $= \frac{10 \times 0.8 + 10 \times 0.78 + 10 \times 0.82}{30}$

$= 0.8$

Macro F1 score $= \frac{F1\text{-score}(C1) + C2 + C3}{3}$

$= \frac{0.8 + 0.78 + 0.82}{3} = 0.8$

Overfitting

In ML, there is a term called train data and test data which machine learning model will learn from train data & try to predict the test data based on its learning. Overfitting is a concept in m/c learning which states a common problem that occurs when a model learns the train data too well including the noisy data, resulting in poor generalization performance on test data.

Feature Engineering processes

Feature engineering in ML contains mainly four processes

- Feature Creation
- Transformations
- Feature extraction
- Feature selection

1. Feature creation

→ It is finding the most useful variables to be used in a predictive model.

→ The new features are created by mixing existing features using addition, subtraction and these new features have great flexibility

2. Transformations

The transformation step of feature engineering involves adjusting the predictor variable to improve the accuracy and performance of the model. For ex; it ensures that the model is flexible to take input of the variety of data; it ensures that all the variables are on the same scale, making the model easier to understand.

3. Feature extraction

→ The main aim of this step is to reduce the volume of data so that it can be easily used and managed for data modelling

→ Feature extraction methods include cluster analysis, text analytics, edge detection algms

4. Feature selection

It is very important to identify and select the most appropriate features from the data and remove the irrelevant or less important features, which is done with the help of feature selection in m/c learning

"Feature selection is a way of selecting the subset of the most relevant features from the original features set by removing the redundant, irrelevant, or noisy features."

Problems.

Q) Consider the following training set Table for predicting the sales of the items.

x_i Items	y_i (sales)
I_1	80
I_2	90
I_3	100
I_4	110
I_5	120

→ Consider 2 fresh items I_6 and I_7, whose actual values are 80 and 75, respectively.

Test items	Actual value	predicted value
I_6	80	75
I_7	75	85

Find MAE, MSE, RMSE, Rel MSE and CV?

Mean Absolute Error (MAE)

$$MAE = \frac{1}{n} \sum_{i=0}^{n-1} |y_i - y_i^1|$$

y_i^0 = Actual value
y_i^1 = predicted value

$$MAE = \frac{1}{2} \times \left[|80-75| + |75-85| \right] = \frac{15}{2} = 7.5$$

Mean Squared Error (MSE)

$$MSE = \frac{1}{n} \sum_{i=0}^{n-1} (y_i^0 - y_i^1)^2$$

$$MSE = \frac{1}{2} \times \left[|80-75|^2 + |75-85|^2 \right] = \frac{125}{2} = 62.5$$

Root Mean Square Error (RMSE)

$$RMSE = \sqrt{MSE} = \sqrt{62.5} = 7.91$$

Relative MSE

$$\text{Rel MSE} = \frac{\sum_{i=0}^{n-1} (y_i^0 - y_i^1)^2}{\sum_{i=0}^{n-1} (y_i^0 - \bar{y}_i^0)^2}$$

For finding Rel mse and cv, the training table should be used to find the average of y. $\bar{y}_0$

The average of y is $= \frac{80+90+100+110+120}{5}$

$= \frac{500}{5} = 100$

$$\text{Rel MSE} = \frac{(80-75)^2+(75-85)^2}{(80-100)^2+(75-100)^2}$$

$$= \frac{125}{1025} = 0.1219$$

CV = Coefficient of variation.

$$CV = \frac{RMSE}{\bar{y}}$$

$$CV = \frac{\sqrt{62.5}}{100} = 0.08$$

Cross validation

→ Cross validation is a technique in ML where we divide our data into smaller parts to train and test our model multiple times.

→ It helps us understand how well our model will perform on unseen data and avoid overfitting.

→ By using cross validation, we can get a more accurate estimate of our models' performance & make better decisions about its effectiveness

4 Types of Cross validation techniques

(i) Hold out method

This method is the simplest cross-validation technique among all, In this method Divide the dataset into 2parts:- The training set & the test set. Usually take 80% as training set and 20% as test set. It's a simple and quick way to evaluate a model. The major drawback of this method is that it may possible the remaining 20% of the data contains some important information which we are leaving while training our model. ie; higher bias.

(ii) Leave-p-out cross validation

In this approach the p datasets are left out of the training data. It means, if there are total n datapoints in the orginal input dataset.

Then n-p datapoints will be used as the training dataset and the p data points as the validation set. This complete process is repeated for all samples, and the average error is calculated to know the effectiveness of the model. But it can be computationally difficult for large p.

(iii) Leave one-out cross validation

This method is similar to the leave-p-out cross validation, but instead of p, we need to take 1 dataset out of training. It means in this approach, for each learning set, only one datapoint is reserved and the remaining dataset is used to train the model. This process repeats for each datapoint. Hence for n samples, we get n different training set and n test set.

- Bias is minimum
- Execution time is high

(IN) K-fold cross-validation

Steps for K-fold cross validation

- Split the i/p dataset into k groups of equal sizes and known as folds
- For each group.
  - Take one group as the reserve or test dataset
  - Use remaining group (K-1 fold) as the training dataset
  - Fit the model on the training set and evaluate the performance of the model using the test set

The o/p is less biased than other method.

test K-Fold Cross validation.

1 block | training set

3-fold cross

$K=3$

① | 1 2 3 | 4 5 6 | 7 8 9 | → E_1.

② Training | Testset | Training

| 1 2 3 | 4 5 6 | 7 8 9 | → E_2

Error estimate

Dataset ↓ 9 exs.

③ Testset

| 1 2 3 | 4 5 6 | 7 8 9 | → E_3

based on training example constructed classif.

9 tuples 9 entries

Specifically, using a set which cannot be repeated.

$$E = \frac{1}{K} \sum_{i=1}^{K} E_i$$

~~Training etc~~

Dimensionality Reduction

→ It is a technique used to reduce the number of features in a dataset while retaining as much of the important information as possible.

→ It is a process of transforming high-dimensional data into lower-dimensional space that still preserves the essence of the original data.

→ High dimensional data refers to data with a large number of features or variables.

→

Curse of dimensionality

It is a common problem in m/c learning, where the performance of the model deteriorates as the no. of features increases. This is beoz the complexity of a model rises with the number of features, and it becomes more difficult to find a good solution.

→ The high-dimensional data can also lead to overfitting., where the model fits the the training data too closely and does not generalize well to new data.

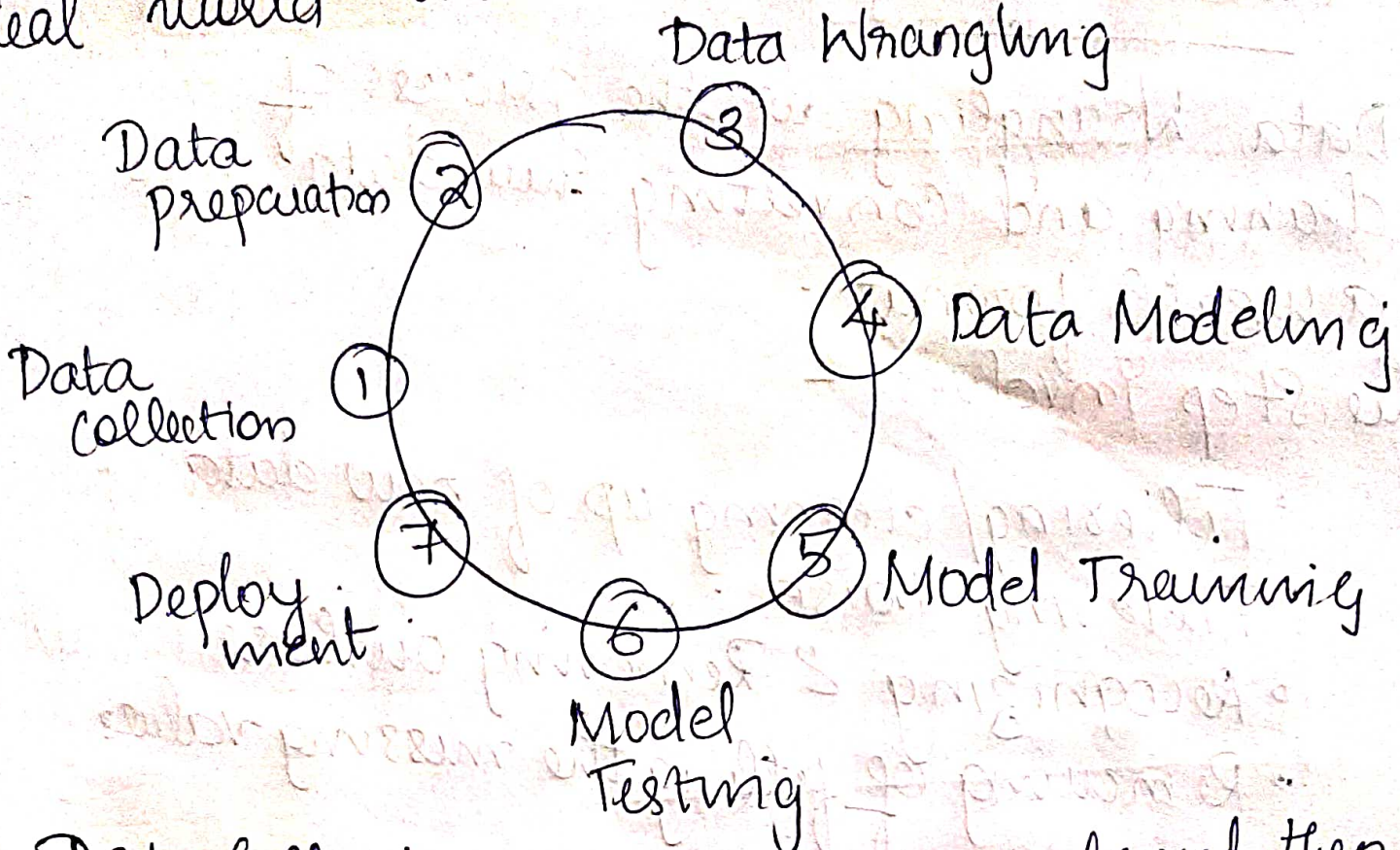
→ Dimensionality reduction can help to mitigate these problems by reducing the complexity of the model and improving generalization performance.

→ There are 2 main approaches to dimensionality reduction:
Feature selection & extraction.

Machine Learning

→ It is the science of getting computers to learn and act like humans do, and improve their learning over time in autonomous fashion, by feeding them data and information in the form of observations and

Real world interactions



Data collection

~~one problem statement cleared then the next step is to collect raw data~~

→ Identify the various source of inform
→ Gather data
→ Combine data acquired from various data source

Data preparation

→ Study your data, understand the data quality of data, features of data, beco2 better understanding of data ~~result~~ leading to generation of best results
→ Data preparation deals with exploration of your data to generate for better results

③ Data Wrangling

Data Wrangling is the process of cleaning and converting raw data into a useable format.

This step involves:-

- Filtering/ cleaning up of raw data
- Filtering Noise
- Recognizing & Removing outliers
- Removing or filling the missing values.

④ Data Modeling

Data Modeling is the step in which we take the data and select a m/c learning algm to built a model.

This step involves:-

→ Selecting m/c learning algm
→ Building the models
→ Validating the results.

Model Training

- You are going to teach skill to your model, with the help of data & algm.
- An ML training model is a process in which a ML algm is fed with sufficient training data to learn from.

Model verification (Test a Model)

Whether the model is suitable for intended appln or not.

→ This stage of ML lifecycle involves checking for the accuracy of the model by providing with the i/ps that are unseen.

Model Deployment

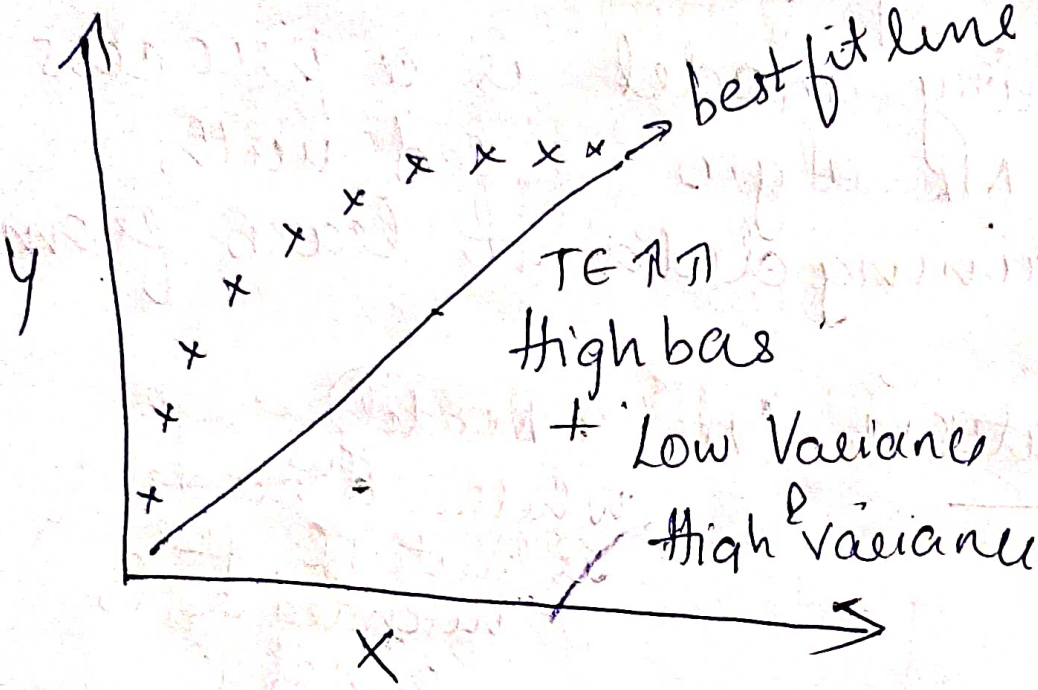
This is the final step in the machine learning life cycle where we have a brilliant model ready to go to production.

Machine learning

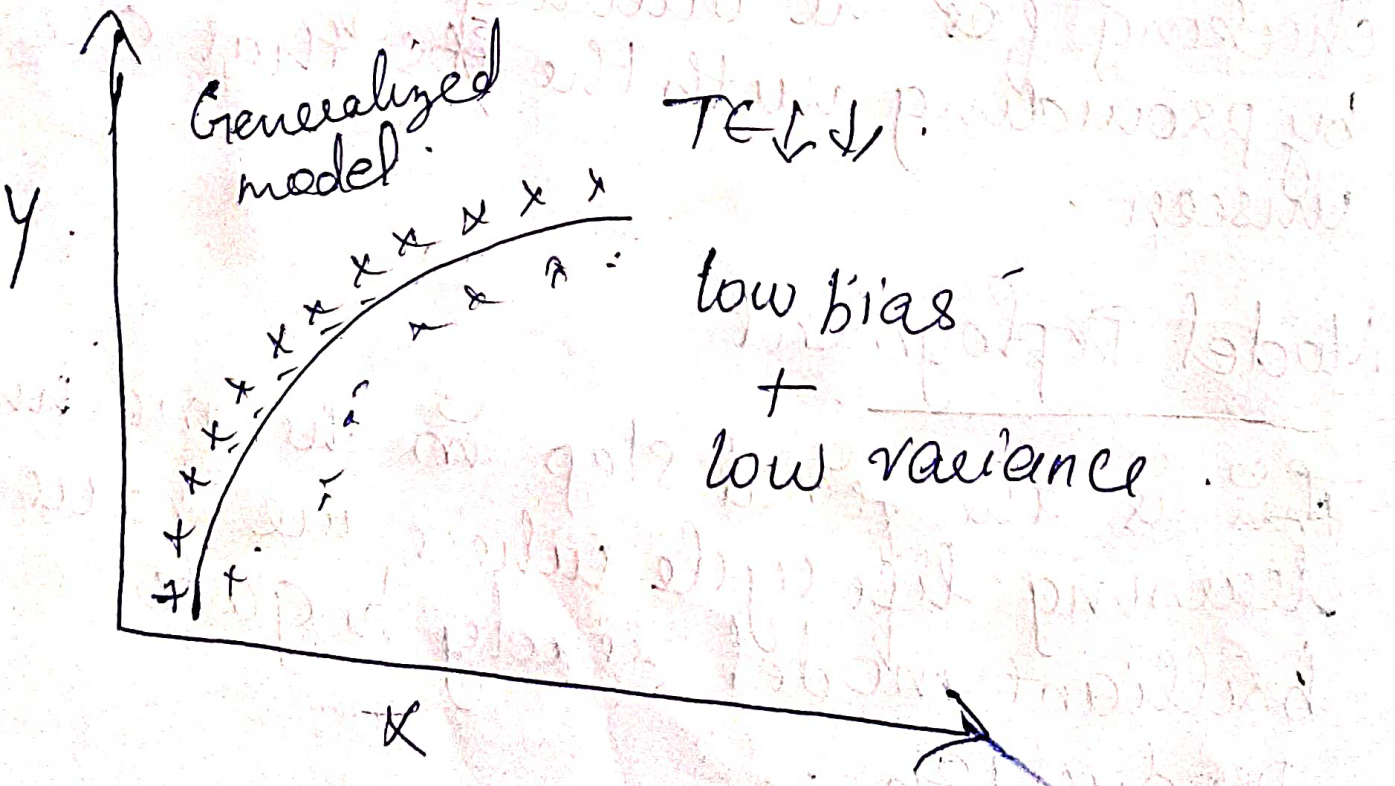
Regression problem

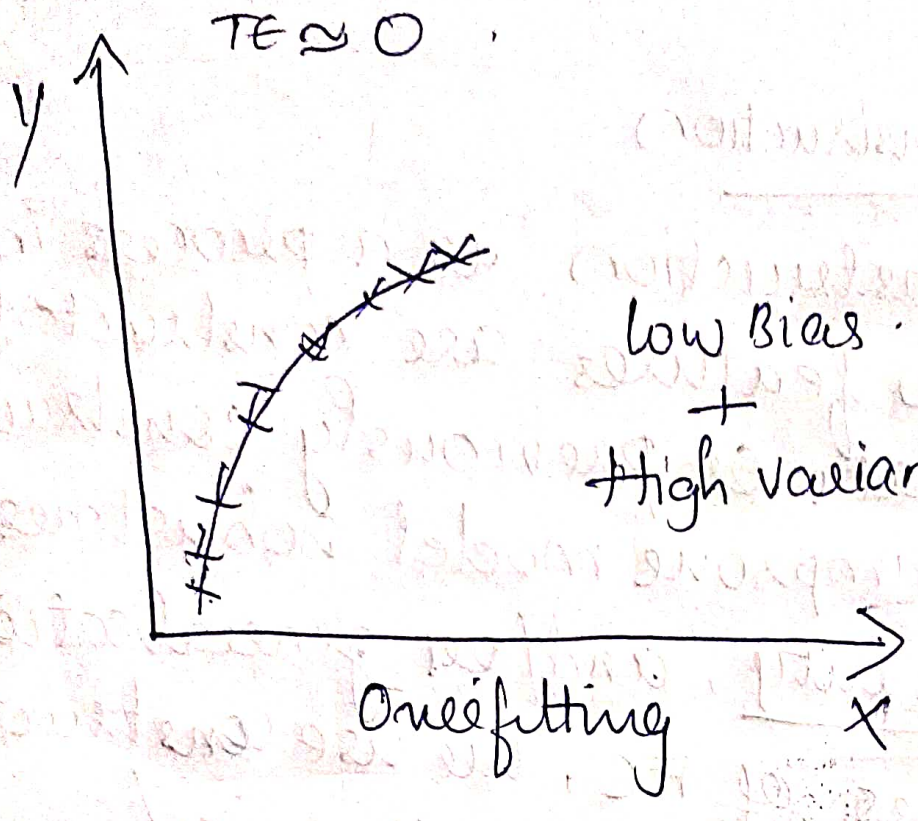
Bias → Training data

Variance → Test data



Underfitting line





Classification problem

Model 1

Train Error = 2%

Test Error = 25%

⇓

overfitting

low Bias

+

high variance

Model 2

Train error = 25%

Test error = 26%

underfitting

High Bias

+

high variance

Model 3

Train error < 10%

Test error < 10%

Generalized model

low Bias

+

low variance

Feature construction

Feature construction is a process in which new features are constructed from raw data or previously constructed features to improve model robustness, interpretability, and/or generalisation.

In the process of FC, we use constructive operators and existing attributes/features. As a result, we get features that may better describe the target concept.

Dimensionality Reduction

Common techniques of Dimensionality Reduction

1) principal component analysis

It is an unsupervised learning algm that is used for the dimensionality reduction in machine learning. It is a statical process that converts the observations of correlated features into a set of linearly uncorrelated ~~techniques~~ features with the help of orthogonal transformation.

Q) Given the data in Table, reduce the dimension from 2 to 1 using the principle component analysis algorithm (PCA).

step1 Dataset

Feature	ex1	ex2	ex3	ex4
x	4	8	13	7
y	11	4	5	14

No. of features $n = 2$

No. of samples $N = 4$

step2 :- Computation of mean of variables

$$\bar{x} = \frac{4 + 8 + 13 + 7}{4} = 8$$

$$\bar{y} = \frac{11 + 4 + 5 + 14}{4} = 8.5$$

Step3

Computation of covariance matrix.

for this first we have to write ordered pairs

Ordered pairs are :-

$(x,x), (x,y) (y,x) (y,y)$

1) Covariance of all ordered pairs

$$Cov(x,x) = \frac{1}{N-1} \sum_{K=1}^{N} (x_{ik} - \bar{x_i})(x_{jk} - \bar{x_j})$$

$$= \frac{1}{4-1} \times \left[(4-8)^2 + (8-8)^2 + (13-8)^2 + (7-8)^2 \right]$$

$$= 14$$

$$Cov(x,x) = \frac{1}{N-1} \sum_{K=1}^{N} (x_i - \bar{x})^2$$

$$\text{Cov}(x,y) = \frac{1}{4-1}\Big[(4-8)(11-8.5) + (8-8)(4-8.5) + (13-8)(5-8.5) + (7-8)(14-8.5)\Big]$$

(mean of x = 8, mean of y = 8.5)

$$= -11$$

$$\text{Cov}(y,x) = -11 = \text{Cov}(x,y)$$

$$\text{Cov}(y,y) = \frac{1}{4-1}\Big[(11-8.5)^2 + (4-8.5)^2 + (5-8.5)^2 + (14-8.5)^2\Big]$$

$$= 23$$

Covariance matrix $n \times n$, 2×2

$$S = \begin{bmatrix} \text{Cov}(x,x) & \text{Cov}(x,y) \\ \text{Cov}(y,x) & \text{Cov}(y,y) \end{bmatrix} = \begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix}$$

Step4 Eigen value, Eigen vectors, Normalized eigen vector

(i) Eigenvalue

$$\det(S - \lambda I) = 0$$

λ = eigen value

I = Identity matrix (2×2)

S = covariance matrix

$$\lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \Rightarrow \lambda$$

$$\det\left(\begin{bmatrix} 14-\lambda & -11 \\ -11 & 23-\lambda \end{bmatrix}\right) = 0$$

$$(14-\lambda)(23-\lambda) - (-11 \times -11) = 0$$

$a = 1$
$b = -37$
$c = 201$

$$\lambda^2 - 37\lambda + 201 = 0 \Rightarrow \frac{1}{2a}\sqrt{b^2 - 4ac}$$

find the roots

$\lambda_1 = 30.3849$, 6.6151

$\lambda_1 = 30.3849$ ⟹ largest eigen value consider for the principle component.

$\lambda_2 = 6.6151$

(ii) eigen vector of λ_1. is 2×1 mat

$$U_1 = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$(S - \lambda_1 I) U_1 = 0$$

$$\begin{bmatrix} 14-\lambda_1 & -11 \\ -11 & 23-\lambda_1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} (14-\lambda_1)u_1 - 11u_2 \\ -11u_1 + (23-\lambda_1)u_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Now we get two linear equations.

$$(14-\lambda_1)u_1 - 11u_2 = 0$$

$$-11u_1 + (23-\lambda_1)u_2 = 0$$

$$14-\lambda_1 u_1 = 11u_2$$

$$\frac{u_2}{14-\lambda}$$

No we we want to find the value of u_1 & u_2 from any of these linear equ

$\frac{u_1}{11} = \frac{u_2}{14-\lambda_1} = t$ we are assigning with a t variable.

when $t=1$

$u_1 = 11$

$u_2 = 14 - \lambda_1$

So Eigen vector v_1 of $\lambda_1 = \begin{bmatrix} u_1 \\ 11 \\ u_2 \\ 14-\lambda_1 \end{bmatrix}$

α eigen

$= \begin{bmatrix} 11 \\ \hline 14 - 30.3841 \end{bmatrix}$

$= \begin{bmatrix} 11 \\ \hline -16.3849 \end{bmatrix}$

(iii) Normalize the eigen vector v_1.

To Normalize the eigen vector just divide the eigen vector with its length.

$$e_1 = \begin{bmatrix} \dfrac{11}{\sqrt{11^2 + (-16.38)^2}} \\ \dfrac{-16.3849}{\sqrt{11^2 + (-16.38)^2}} \end{bmatrix}$$

$\frac{\text{of eigen vector}}{\text{length}}$

$$= \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}$$

This is unit eigen vector corresponds to higher λ value (eigen value)

λ_2

$$e_2 = \begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$$

$\therefore$

Step 5:- Derive new dataset

	Ex_1	Ex_2	Ex_3	Ex_4
First principle component PC1	P_{11}	P_{12}	P_{13}	P_{14}

$$P_{11} = e_1^T \begin{bmatrix} x_1 - \bar{x} \\ 4 - 8 \\ y_1 - \bar{y} \\ 11 - 8.5 \end{bmatrix} \qquad F = \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \\ \vdots \\ e_p^T \end{bmatrix}$$

$$= \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} -4 \\ 2.5 \end{bmatrix}$$

1x1

$$= \begin{bmatrix} -4.3052 \end{bmatrix}$$

$$P_{12} = \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} 8 - 8 \\ 4 - 8.5 \end{bmatrix}$$

$$= 3.7361$$

$P_{13} = 5.6928$

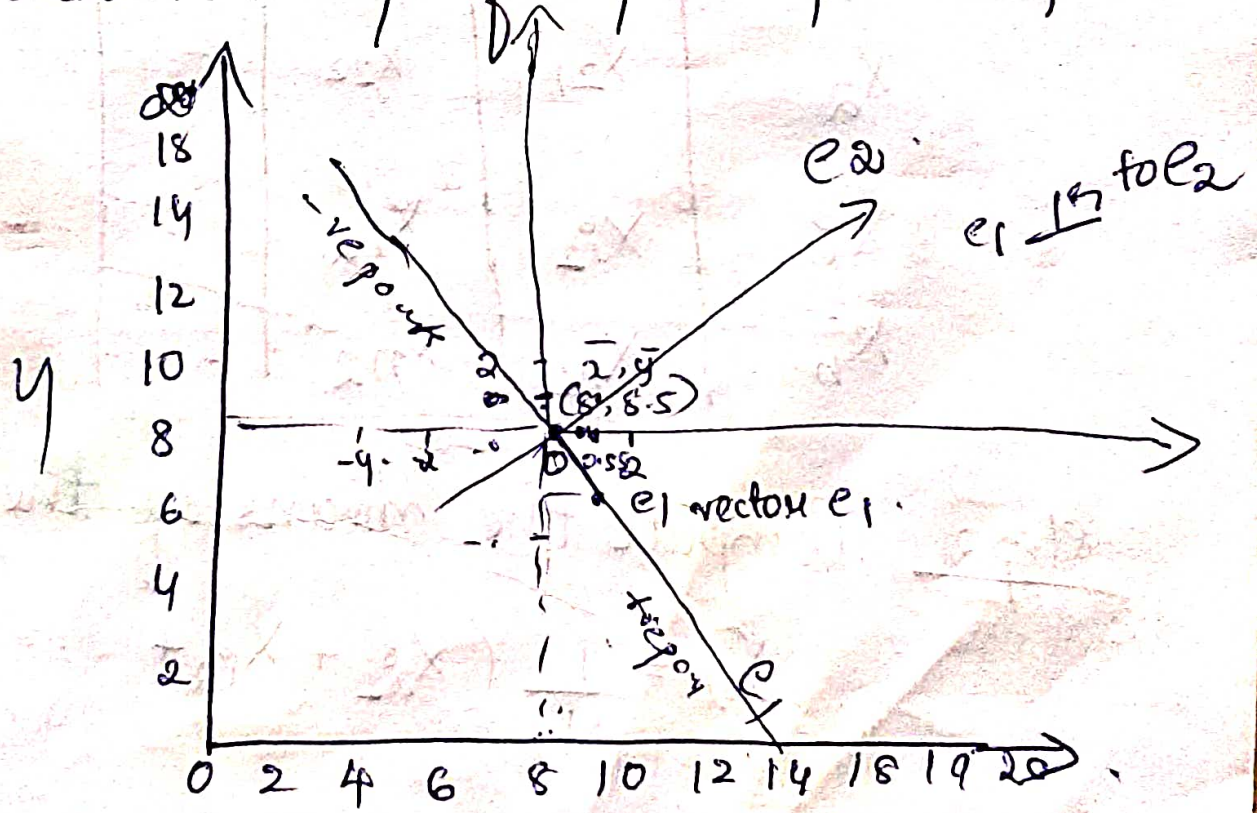
$P_{14} = -5.1238$

New dataset

	ex1	ex2	ex3	ex4
PC1	-4.3052	3.7361	5.6928	-5.1238

step

Co-ordinate s/m for principal component

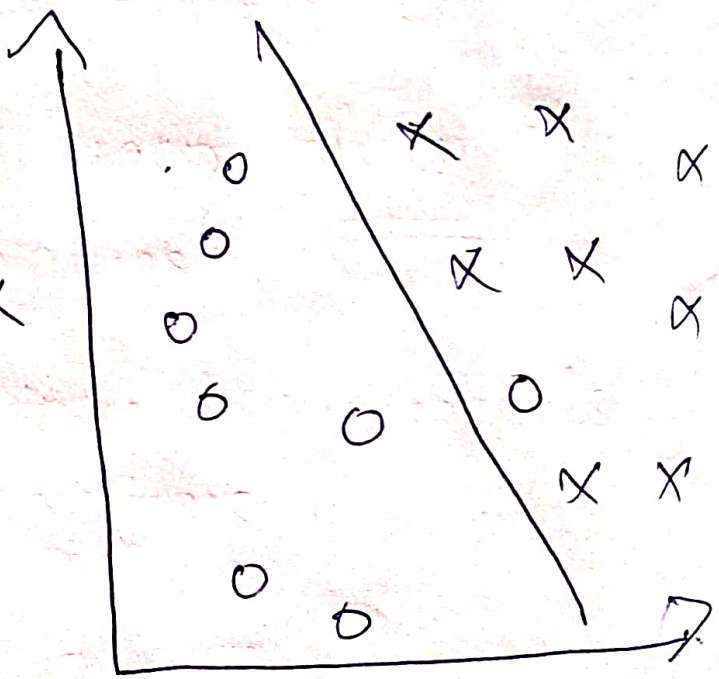


Linear Discriminant Analysis

→ Linear discriminant Analysis or Normal discriminant Analysis or Discriminant function analysis is a dimensionality reduction technique that is commonly used for supervised classification problems.

→ It is used to project the features in higher dimension space into a lower dimension space.

→ Suppose we have two sets of data points belongs to two different classes that we want to classify.



→ When the data points are plotted on 2D plane, there's no straight line that can seperate the two classes of the data points completely

then, LDA uses both the axes (x and y) to create a new axis and projects data onto a new axis in a way to maximize the seperation of the two categories and hence, reducing the 2D graph into a 1D graph

→ 2 criteria are used by LDA to create a new axis

- Maximize the distance b/w means of the two classes
- Minimize the variation within each class

Steps

1. Compute the class means of dependent variable $\mu_1 = \frac{1}{N_1} \sum_{x \in \omega_i} x$

2. Derive the covariance matrix of the class variable $S_1 = \sum_{x \in \omega_i} (x - \mu_1)(x - \mu_1)^T$

3. Compute the within class-scatter matrix $(S_1 + S_2)$

$$S_W = S_1 + S_2$$

4. Compute the between class scatter matrix

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

5. Compute the eigen values and eigen vectors from the within class and b/w class scatter matrix.

$$S_W^{-1} S_B \, w = \lambda w \Rightarrow |S_W^{-1} S_B - \lambda I| = 0$$

6. Sort the values of eigen values and select the top k values.

7. Find the eigen vectors corresponds to the top k eigen vectors.

$$\left[S_W^{-1} S_B - \lambda I \right] \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0$$

eigen vector

or directly

$$w = S_W^{-1}(\mu_1 - \mu_2)$$

8. Obtain the LDA by taking the dot product of eigen vectors and original data

LDA Solved Example

Q) Compute the ~~linear~~ Discriminant projection for the following two dimensional dataset.

Samples for class ω_1

$X_1 = (x_1, x_2) = \{(4,2), (2,4), (2,3), (3,6), (4,4)\}$.

Samples for class ω_2:

$X_2 = (x_1, x_2) = \{(9,10), (6,8), (9,5), (8,7), (10,8)\}$.

Our task is to find a new axis, to project the data values

Step 1

The classes mean are:-

$$\mu_1 = \frac{1}{N_1} \sum_{x \in \omega_1} x = \frac{1}{5}\left[\begin{bmatrix}4\\2\end{bmatrix} + \begin{bmatrix}2\\4\end{bmatrix} + \begin{bmatrix}2\\3\end{bmatrix} + \begin{bmatrix}3\\6\end{bmatrix} + \begin{bmatrix}4\\4\end{bmatrix}\right]$$

$$= \begin{bmatrix}3\\3.8\end{bmatrix}$$

$$\mu_2 = \frac{1}{N_2} \sum_{x \in \omega_2} x = \frac{1}{5}\left[\begin{bmatrix}9\\10\end{bmatrix} + \begin{bmatrix}6\\8\end{bmatrix} + \begin{bmatrix}9\\5\end{bmatrix} + \begin{bmatrix}8\\7\end{bmatrix} + \begin{bmatrix}10\\8\end{bmatrix}\right] = \begin{bmatrix}8.4\\7.6\end{bmatrix}$$

Step 2

Covariance matrix of the first class:-

$$S_1 = \sum_{x \in \omega_1} (x-\mu_1)(x-\mu_1)^T$$

$$= \left[\begin{bmatrix}4\\2\end{bmatrix} - \begin{bmatrix}3\\3.8\end{bmatrix}\right]\left[\begin{bmatrix}4\\2\end{bmatrix} - \begin{bmatrix}3\\3.8\end{bmatrix}\right]^T + \left[\begin{bmatrix}2\\4\end{bmatrix} - \begin{bmatrix}3\\3.8\end{bmatrix}\right]\left[\begin{bmatrix}2\\4\end{bmatrix} - \begin{bmatrix}3\\3.8\end{bmatrix}\right]^T$$

Note: 1st term $\begin{bmatrix}1\\-1.8\end{bmatrix}\begin{bmatrix}1 & -1.8\end{bmatrix} \Rightarrow \begin{bmatrix}1 & -1.8\\-1.8 & 3.24\end{bmatrix}$

$$+\left[\begin{bmatrix}2\\3\end{bmatrix}-\begin{bmatrix}3\\3.8\end{bmatrix}\right]\left[\begin{bmatrix}2\\3\end{bmatrix}-\begin{bmatrix}3\\3.8\end{bmatrix}\right]^T+\left[\begin{bmatrix}3\\6\end{bmatrix}-\begin{bmatrix}3\\3.8\end{bmatrix}\right]\left[\begin{bmatrix}3\\6\end{bmatrix}-\begin{bmatrix}3\\3.8\end{bmatrix}\right]^T$$

$$+\left[\begin{bmatrix}4\\4\end{bmatrix}-\begin{bmatrix}3\\3.8\end{bmatrix}\right]\left[\begin{bmatrix}4\\4\end{bmatrix}-\begin{bmatrix}3\\3.8\end{bmatrix}\right]^T$$

$N=1$

$$=\begin{bmatrix}1 & -0.25\\ -0.25 & 0.2\end{bmatrix}$$

$$S_2 = \frac{\sum_{x \in w_2} (x-\mu_2)(x-\mu_2)^T}{N-1}$$

$$= \left[\begin{pmatrix}9\\10\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]\left[\begin{pmatrix}9\\10\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^T + \left[\begin{pmatrix}6\\8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]$$

$$\left[\begin{pmatrix}6\\8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^T + \left[\begin{pmatrix}9\\5\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]\left[\begin{pmatrix}9\\5\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^T +$$

$$\left[\begin{pmatrix}8\\7\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]\left[\begin{pmatrix}8\\7\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^T +$$

$$\left[\begin{pmatrix}10\\8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]\left[\begin{pmatrix}10\\8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^T \Big/ \frac{5-1}{N-1}$$

$$= \begin{pmatrix}2.3 & -0.05\\ -0.05 & 3.3\end{pmatrix}$$

(answer)

Step 3

Within-class scatter matrix

$$S_w = S_1 + S_2 = \begin{pmatrix} 1 & -0.25 \\ -0.25 & 2.2 \end{pmatrix} + \begin{pmatrix} 2.3 & -0.05 \\ -0.05 & 3.3 \end{pmatrix}$$

$$= \begin{pmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{pmatrix}$$

Step 4

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

$$= \left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]\left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]^T$$

$$= \begin{pmatrix} -5.4 \\ -3.8 \end{pmatrix}\begin{pmatrix} -5.4 & -3.8 \end{pmatrix} = \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix}$$

Steps

Find eigen values

$$\Rightarrow |S_W^{-1} S_B - \lambda I| = 0$$

$$\Rightarrow \left| \begin{pmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{pmatrix}^{-1} \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = 0$$

$$\Rightarrow \left| \begin{pmatrix} 0.3045 & 0.0166 \\ 0.0166 & 0.1827 \end{pmatrix} \begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = 0$$

$$\Rightarrow \left| \begin{pmatrix} 9.2213 - \lambda & 6.489 \\ 4.2339 & 2.9794 - \lambda \end{pmatrix} \right|$$

$$\Rightarrow (9.2213 - \lambda)(2.9794 - \lambda) - 6.489 \times 4.2339 = 0$$

$$\Rightarrow \lambda^2 - 12.2007\lambda = 0 \Rightarrow \lambda(\lambda - 12.2007) = 0$$

$$\Rightarrow \lambda_1 = 0, \lambda_2 = 12.2007$$

- Find eigen vector

$$\left(S_W^{-1} S_B - \lambda I\right)\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 0$$

$$w_1 = \begin{pmatrix} -0.5755 \\ 0.8178 \end{pmatrix} \qquad w_2 = \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix} = w$$

or directly

$$w^* = S_W^{-1}(\mu_1 - \mu_2) = \begin{pmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{pmatrix}^{-1}\left[\begin{pmatrix} 3 \\ 3.8 \end{pmatrix} - \begin{pmatrix} 8.4 \\ 7.6 \end{pmatrix}\right]$$

$$= \begin{pmatrix} 0.3045 & 0.0166 \\ 0.0166 & 0.1827 \end{pmatrix}\begin{pmatrix} -5.4 \\ -3.8 \end{pmatrix}$$

$$= \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix}$$

* Obtain the LDA by taking the dot product of eigen vectors and original data

$$w_2 = \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix} = w \cdot \underset{1\times 2}{[4\ 2]} \cdot \begin{bmatrix} 0.9 \\ 0.4 \end{bmatrix}$$

x_1	4	2	2	3	4	9	6	9	8	10
x_2	2	4	3	6	4	10	8	5	7	8
1st LD	4.46	3.48	3.06	5.2	5.3	12.35	8.8	10.2	10.19	12.4

GDA

→ Generalized discriminant analysis (GDA) is a commonly used method for dimensionality reduction. Also known as Kernel discriminant Analysis (KDA).

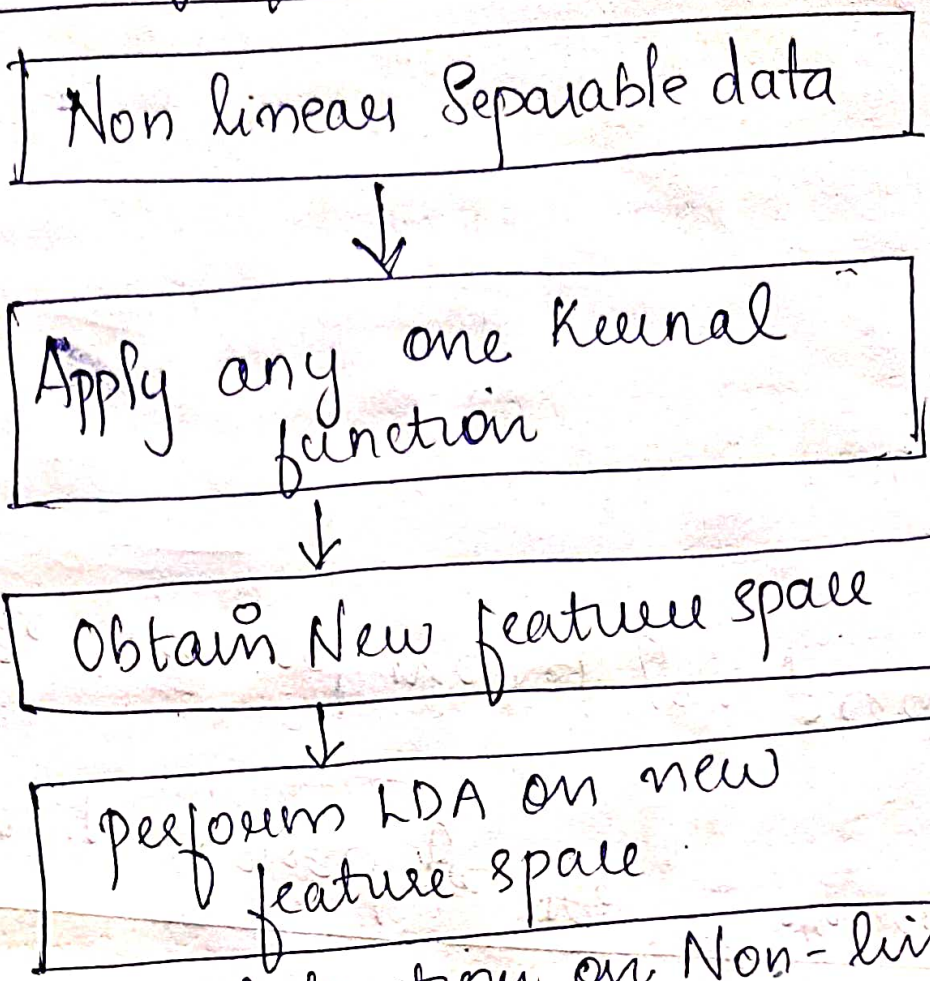
→ GDA is an extension of LDA to non-linear distribution, since conventional LDA is not suitable for non linear data sets.

→ GDA deals with non linear discriminant analysis using kernel function operator.

→ The objective of GDA is to find non linear projection that simultaneously maximizes the between-class dissimilarity and minimizes the within-class dissimilarity to increase class separability.

→ The main idea is to map the input space into a convenient feature space in which variables are non linearly related to the input space.

Working of GDA



Step1:- Apply Kernel function on Non-linear seperable data to map to input space into high dimensional feature space x.

Step2:- Perform LDA on new feature space to extract most significant discriminant features