feature engineering

↳ feature transformation.

↳ { match - smooth mathematical transformation }

1). log - transform.
2). Reciprocal
3). power (sq / sqrt)

1). Box - Cox.
2). Yeo - Johnson

↳ { normal distribution }

{ ↳ convert in normal distribution }

function transformer :-

sk - learn

Use ↳ sns.distplot.

function transform
- log
- Reciprocal
- sq / sqrt.
- custom.

Power transform
- Box - cox
- Yeo - Johnson.

Quantile tansform.

pd → pd.skew()
((-, +) ↳ skewed data)

★ Q-Q plot'

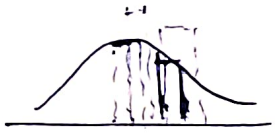
↳ scipy
scipy.stats use.

Normal distribution

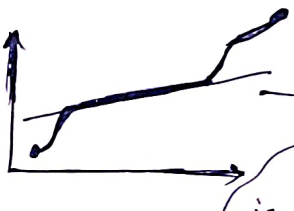
→

'Q-Q' plot

↳ Data peaked in middle.



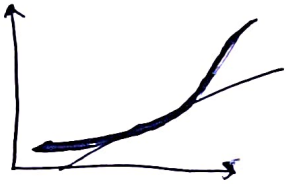
→



As close to this line as much it is normal distribution

↳ skewed - data.

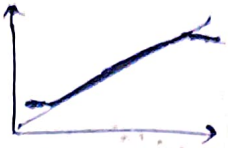
→



1) Thin tails

'distribution'

→



Normal 'Q-Q'

log transform
1) Numerical column
→ each val. → take log.
take any log base

⇒ log transform used on 'Right-skewed' data

⇒ It equivalent in the scale.

Reciprocal	sqrt	Square. (x^2)
⇒ (1/x)	($\sqrt{x}$)	⇒ 'left skewed data'
Image ↔ Text.	Mild skew	Make val non-negative
audio ↔ Text	Vari ∝ Mean.	highlight outliers

Power transformer :-

⇒ Box-Cox transformer :-

⇓

given distrib. $x_i^{(\lambda)} = \begin{cases} \frac{x_i^{(\lambda)} - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln(x_i) & \text{if } \lambda = 0 \end{cases}$

⇓

(convert Normal distribution) {try to calculate 'λ' val.} range [-5 ↔ +5]

only applicable on ($x > 0$) → $-5 < \lambda < 5$

⇒ Yeo-Johnson :-

⇓

(Applicable on every value) $x_i^{(\lambda)} = \begin{cases} [(x_i+1)^{\lambda} - 1]/\lambda, & \lambda \neq 0,\ x_i \geq 0 \\ \ln(x_i)+1, & \lambda = 0,\ x_i \geq 0 \\ -[(-x_i+1)^{2-\lambda} - 1]/(2-\lambda), & \lambda \neq 2,\ x_i < 0 \\ -\ln(-x_i+1), & \lambda = 2,\ x_i < 0 \end{cases}$

we will take each 'λ' then raise it with the power of 'x'.

Binning & Binarization :-

Skewed d-set.

Age
27
35
45
} → category.

Numerical data → categorical data.

Binning (Descritization) | Binarization

Binning :- A process to transform continous variable into discrete variable by creating intervals to span the range values of it.

→ To handle outliers

→ To improve the value spread.

↳ binning apply

Uniform the data-spread.

Age

23, 35, 57, 81 . . . 85

{0-10, 10-20, 20-30 . . .}

Binning

unsupervised | Supervised | Custom binning (ourself created bins) own knowledge

unsupervised:
- equal width (uniform)
- Equal frequency
- K-Mean binning

Supervised:
- Decision tree binning

Uniform binning :-

Select No. of bins - '10' selected self

$$UB = \frac{max - min}{total\ no.\ of\ bins}$$

⇒ {equal interval}

Uniform binning.

Equal frequency :- → select the no. of bin → '%ile' based → {like age - %ile distribution lies.

↳ value - spread even

K-Mean binning → form cluster

Use 'K-Mean'

Suppose 'bins = centroid = 5' → random created

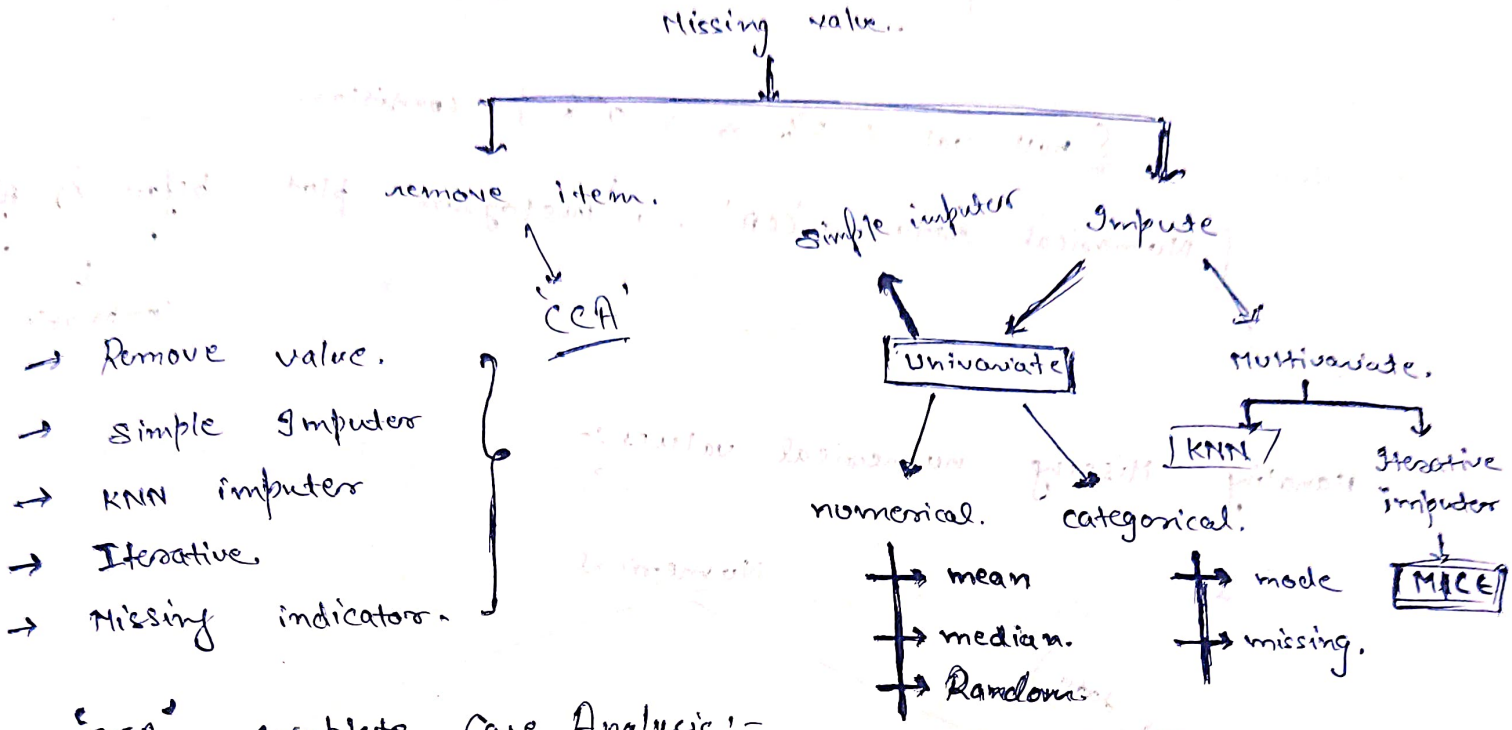
'bisect' at mid-way

OR

{Calc distance of each point from each centroid}

→ 'K-Bin Discritizer'
→ encoding → ordinal / OHE
→ Statergy

Handling missing values :-



- → Remove value.
- → Simple Imputer
- → KNN imputer
- → Iterative.
- → Missing indicator.

'CCA' - Complete Case Analysis :-

⇓

'list-wise deletion' consists in discarding observations if in any of variable are missing.

Advantage | Disadvantage :-

⇒ Easy to impliment as no data manipulation required.

⇒ Preserves variable distribution.

Disadvantage :-

⇒ It can exclude a large fraction of original dataset.

⇒ Excluded observations could be informative for the analysis.

⇒ When using our models in production ; → it will not handle missing values.

[MCAR → 5% < CCA]

CCA condition :-

[coloumn → apply → data missing < 5%]

↓

{ null val < 5% & > 0% } condition.

[Numerical data → 'CCA' → 'histogram' plot before & aft

↓

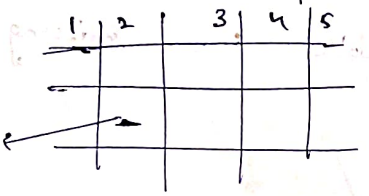
Comparable

Handling Missing numerical values :-

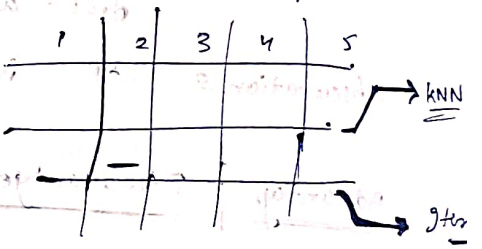
Numerical

apply ⇒ (numerical data)

Univariate imputer



Multivariate imputer



'NA'

['C-2' 'R-2' NA]

50/4

- mean / median,
- Arbitary
- Random
- End of distribution.

Mean / Median :-

Age

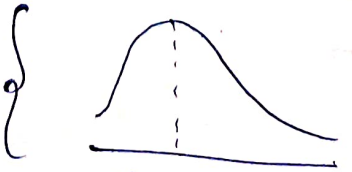
27
35
NA
45

→ (mean/median imputer)

Normal distribution

⇓

mean apply



Skewed data

→ median apply.

Benifits:-

1). Simplicity (server deploy)
2). '5 %' 'less reliable'

When to use

1). less data missing < 5%
2). Missing completly at random.
↳ variance shrink's

Disadvantage:-

1). Distribution shape change.
2). Outlier's formed.
3). Correlation | Covariance changes
↳ mean other colourm se result vary.

Arbitrary Value Imputation :-

↳ Apply on Categorical → 'NA' → 'Missing' ⇒ fill val.

{ Benifit :- easy to apply }

⇒ PDf graph distribute
⇒ Covariance also changes
⇒ Variance

} Disadvantages

Numerical data
↳ "Arbitrary No. use"
(Used when Data isn't missing at Random)

End of distribution Imputer :-

{ Arbitary val → Missing replace }

PDf chayes
Variance vasies.

(IQR - inter quantile range)

Normally distributed
↳ (Mean + 3σ)
or
(Mean - 3σ)
↑ missingness.
→ {Benifit easy}.

skewed data → (IQR proximity)
↓
{ Q1 - 1.5 IQR
Q1 + 1.5 IQR }

Handling Missing Categorical data:-

⇓

replace value by 'mode'

condition:-

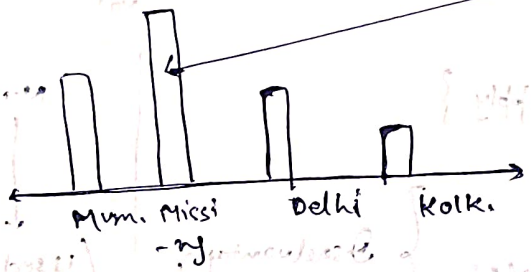
It chaye distribution of data.

completly at random

mode > more time in data

Missing category Imputation:-

- Missing val. > 10% or more.
  replace with mode (x)
- create a new category Name → Missing.



Random Imputation :-

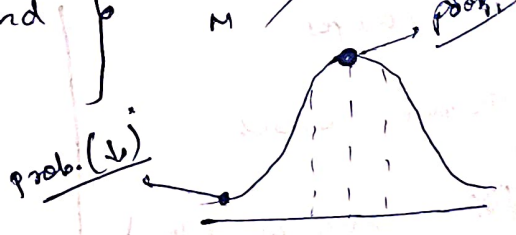
Age
2
9
8
NA
-
-
-

Random No. → data 'numbers' value 'put'

Not much chayes in distribution and variance

sex
M
f
M
-
F
M

→ filled



Memory heavy for deployment

server → column ko deft. there.

Missing Indicator :-

Age	Fare	'Age - NA' → new coloumn
27	29	F → for all present value.
41	32	F
NA	31	T → missing val.
52	41	

Automatically select value for imputation :-

↓

'grid search CV'

↳ 'grid search cross validation'

↳ 'give best parameter'

↳ automates the search with reliable evaluation.

Step :-

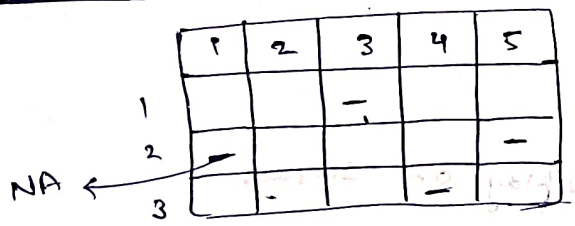
⇒ Define model

⇒ Define a parameter

⇒ GridSearchCV tries All combination

⇒ Cross - val.

KNN Imputer :-



→ KNN Imputer

Iterative Imputer) (Multivariate case)

((-) = NA)

Calculate the distance

to 'find r_2 c_4' → calculate distance from each row

↓

'least value' → 'feature 1' val. 'isse replace'

KNN

↳ 'k = no. of neighbour'

↳ find the value (take val. from neighbour

calculate mean)

general distance ↳ $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$

This case val. is missing more then

nan - euclidan formula

$dist(x,y) = \sqrt{(weight) \times (distance\ from\ pr.\ coordinates)^2}$

↳ Calculate = 33

Apply ↳

SN	F-1	F-2	F-3	F-4
1	33	—	67	21
2	—	45	68	12
3	23	51	71	18

$= \sqrt{weight \times (51-45)^2+(71-68)^2+(18-12)^2}$

↕ (3/3)

weight = total no. of co-ordinates / total no. of pr. coordinates

Advantages/Disadvantages :-

1). More accurate

2). More no. of calculation.

3). Production → train set deploy on server.

Multivariate Imputation by chained equation :-

(MICE)

- MCAR
- MAR → best for it
- MNAR

Adv | Disadv :-

Adv ↓ Accurate

Disadv ↳ Slow

- server
- training
- memory

It-0

R&D	Admin	M.Spend
8	15	30
—	5	20
15	10	41
12	5	26
2	15	—

→ NAN replace by mean of each column.

step - II move left to right in column

replace missing value with NAN. only first one

It-1

R&D	Admin	M.Spend
8	15	30
NAN	5	20
15	MEAN	41
12		
2	15	MEAN

predict

→ training data (input)

R&D	Admin	M.Spend
8	15	30
NAN ⟷	5	20
15	mean	41
12	15	mean

→ 'prediction'

⇒ Apply linear regression / ML model to train & predict.

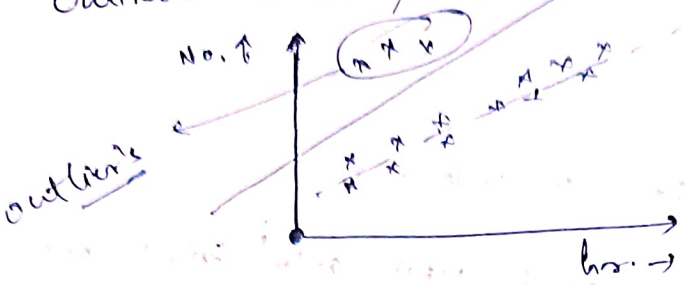
likelihood same for again repeat each step.....

Then :- ((Iteration - 1) - (Iteration - 0))

loop till the difference ≈ 0

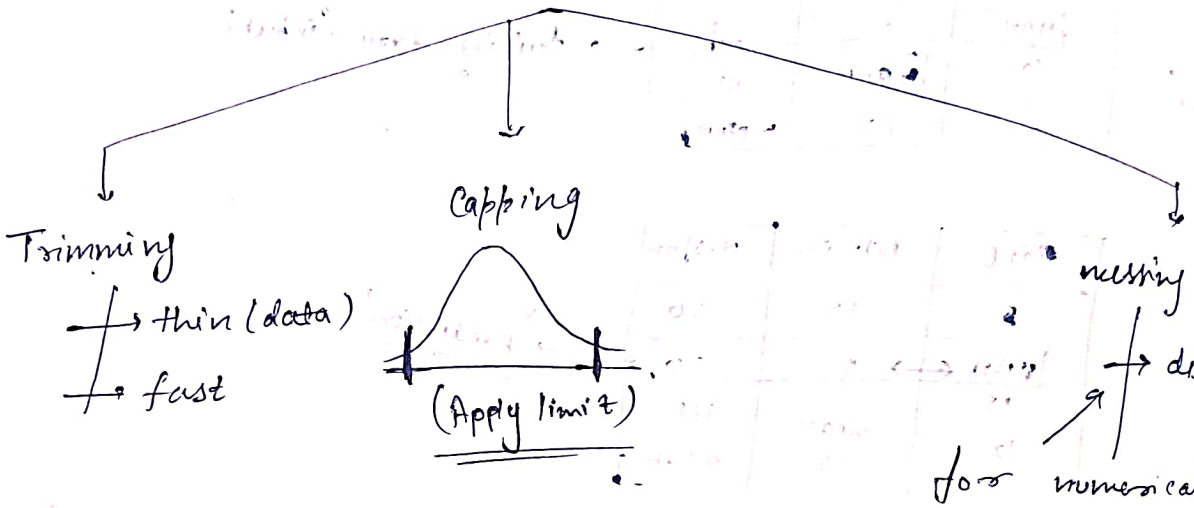
Outliers Detection :-

Outliers :- A datapoint out of bound & a different behaviour



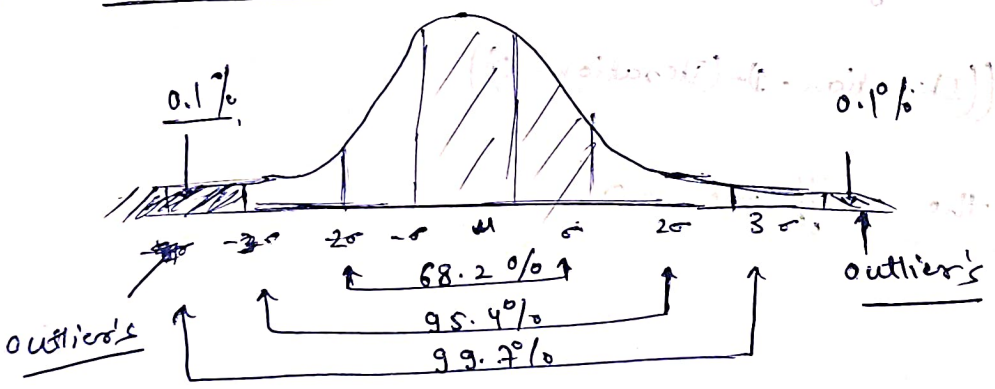
linear algorithm
AdaBoost
logistic regression } affected by outliers
all

Outliers treatment



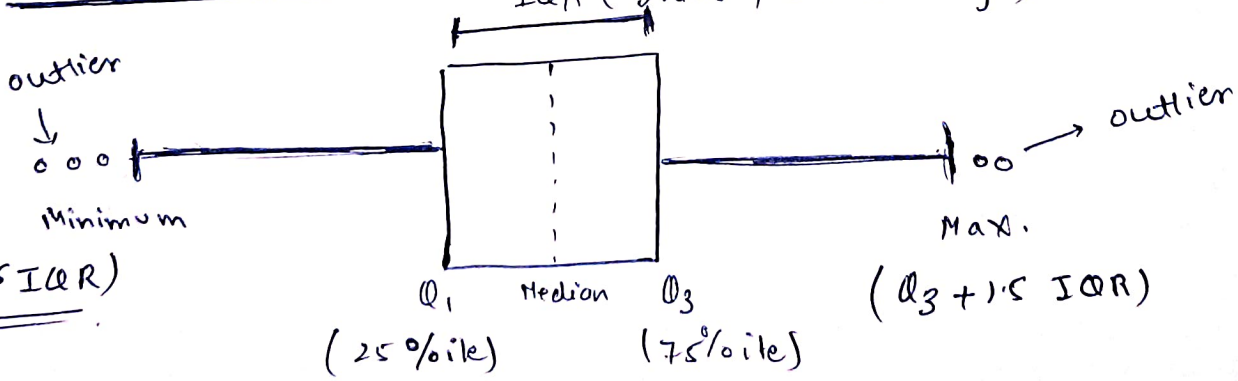
How to detect Outliers :-

1). **Normal distribution :-**

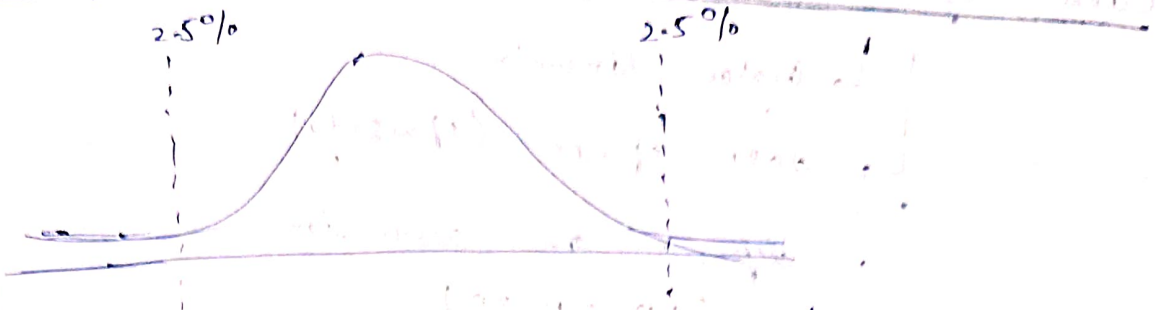


$[(\mu + 3\sigma) > \quad (\mu - 3\sigma) <]$ outlier

2). **Skewed Distribution :-**



3). % ile - based approach :-



we select a particular range & apart from it everything is a outlier.

Technique for outlier Detection and Removal.

- Z-score treatment
- IQR based filtering
- Percentile
- Winsorization
  - 1/sile method - capping

Outlier's removal using Z-score :-

$$Z\text{-Score} = \left(x_i' = \frac{x_i - \mu}{\sigma} \right)$$

consider range [-3 3].

np.where
↳ condition (T/F)

outlier treatment.

- [Trimming] → remove row
- [Capping] → apply cap to a particular range.

Outlier distribution using IQR :-

$$\begin{bmatrix} \text{Max} & Q_3 + 1.5\ IQR \\ \text{Min} & Q_1 - 1.5\ IQR \end{bmatrix}$$ (IQR proximity rule)

⇒ Curse of dimentionality :-

- higher dimension
- data space (sparsity).
- depend on dimension (1D, 2D, 3D)
  - ↑ dimension (↑), finding particular point (difficult).

Dimentionality reduction → soln.

⇒ Performance decreses (↓)

⇒ higher computation.

Dimentionality reduction.

- Feature Selection
  - forward selection
  - Backward substitution
- Feature extraction.
  1) PCA
  2) LDA (Linear discriminate analysis)
  3) TSNE

⇒ Principle Component Analysis (PCA) :-

- Uns upervised ML problem.
- only input no output.
- old technique.

feature extraction technique to reduce curse of dimentionality

Benifit of PCA :- [even reduce '10-D' in '3-D']

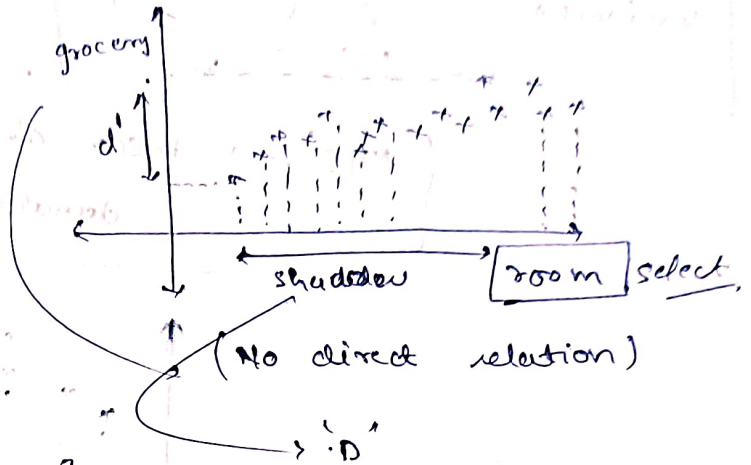
1) visualization.
2) faster

geometric intution :-

No. of rooms	No. of grocery shops	price (L)
3	2	60
4	0	130
5	6	170
2	10	90

↑ (remove)

(PCA)
↳ (feature selection)
to predict to the output.



{ D = shadow
d' = shadow of Y-axis }

[d > d']

changes in 'd-set' :-

No. of room	No. of washroom	Price

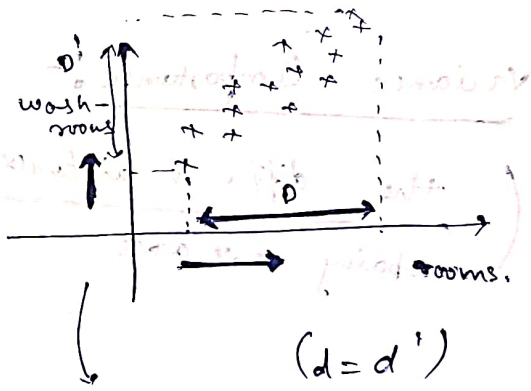
input col. complete change.

'combine & get the size'

Size	Price

['2D' ↓ '1-D']

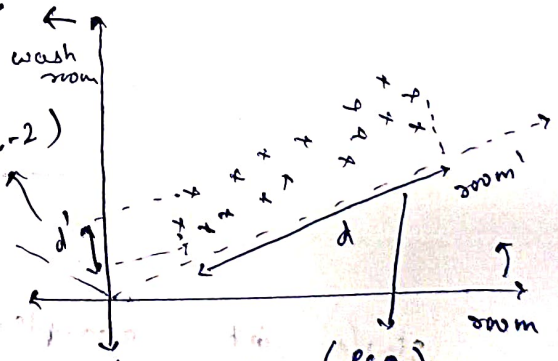
'PCA'



(d = d')

(feature extraction works here)

(d > d') ⟶ washroom



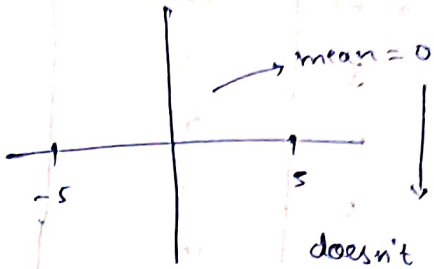
No. of Principle component ≤ 'n'

(total no. of orignal feature in data.)

⇒ **Variance :-**

it describe's the spread of data...

$$\left[\text{variance} = \sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n} \right]$$



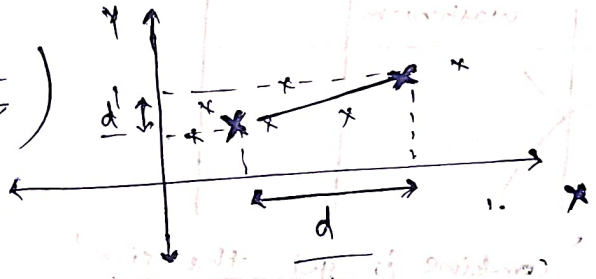
doesn't classify diff. datasets

{ 'MAD' → mean absolute deviation }

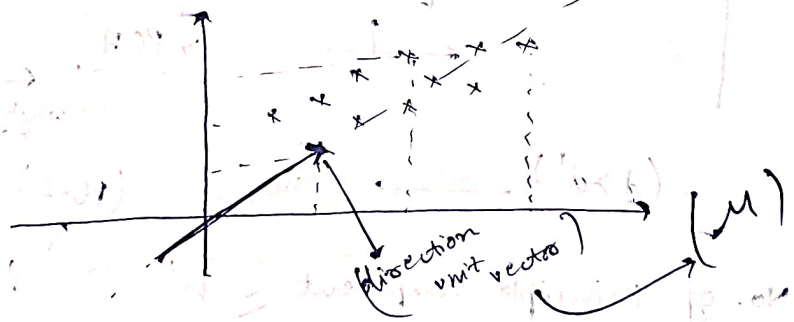
(variance calculate)

Variance Importance :-

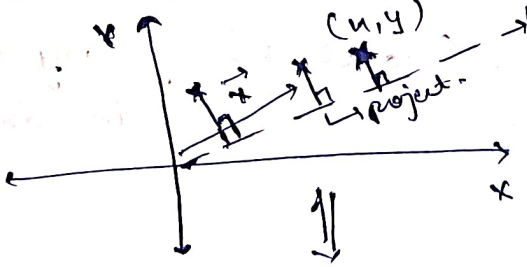
(the diff. in distance containg x,y axis)



⇒ **problem formulation :-**



single-point example.



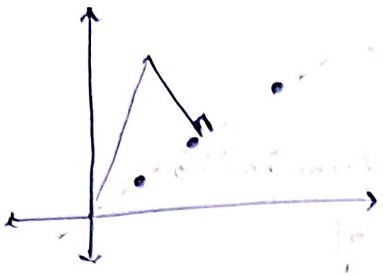
$$\left(\text{project}_{a,b} = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}|} \right)$$

$$\left[\text{projection} = \vec{u} \cdot \vec{x} \quad , |\vec{u}| = 1 \right]$$

$= u^T u$

$$\begin{bmatrix} x_1 & y_1 \end{bmatrix} \begin{bmatrix} u_2 \\ y_2 \end{bmatrix}$$

$(= u_1 u_2 + y_1 y_2 = \text{scalar})$

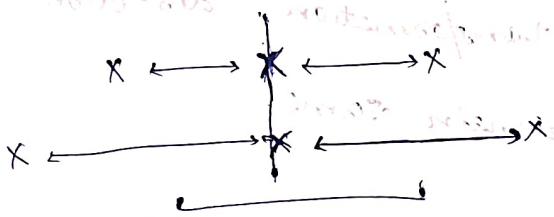


variance allocate $= 1/n \sum_{i=1}^{n} (u_i - \bar{u})^2$

$$\left[\sum_{i=1}^{n} \frac{(u^T u_i - v^T \bar{u})^2}{n} \right] = (\text{on} - \vec{u}) \text{ variance.}$$

↳ 'optimization'

⇒ **Co-variance & Covariance Matrix :-**

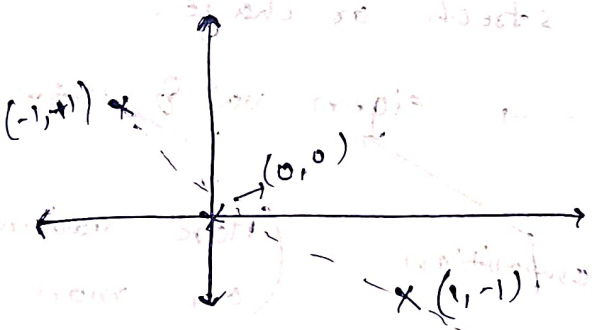


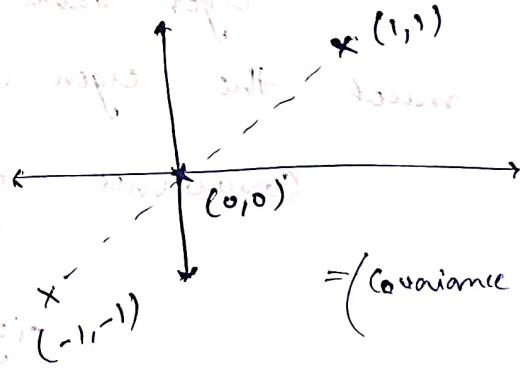
mean problem.

Variance problem

⇓

['diff. on x, y axis']





$=$ (Covariance $= 2/3$)

(Co-variance → relation b/n x, y axis)

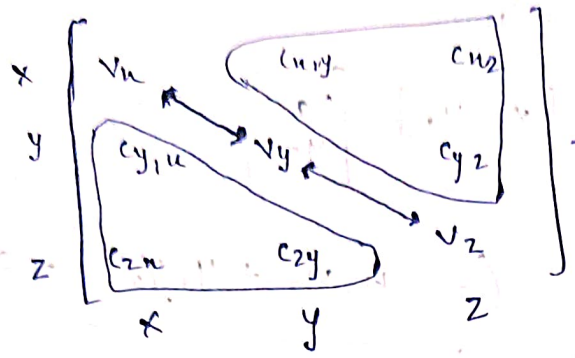
Covariance $= (-2/3)$

(diff co-relation).

$$[2 \times 2] = \begin{matrix} x_1 \\ x_2 \end{matrix} \begin{bmatrix} \text{cov}(u_1, u_1) & \text{cov}(u_2, u_1) \\ \text{cov}(u_1, u_2) & \text{cov}(u_2, u_2) \end{bmatrix}$$ ↳ [-1 1] range.

$\text{cov}(a,b) = \text{cov}(b,a)$

$$= \begin{matrix} u_1 \\ u_2 \end{matrix} \begin{bmatrix} \text{var}(u_1) & \text{cov}(u_2, u_1) \\ \text{cov}(u_1, u_2) & \text{var}(u_2) \end{bmatrix}$$ (= square matrix)



→ cov. Matrix

eigen decomposition of covariance matrix.

linear Transformation, Eigen Vector and Eigen Value :-

Matrix are linear transformation (geogebra.org)

Eigenvector :- special vector after transformation direction don't change span remain same.

Eigen value :- the value which usually desire (how much the eigen vector stretch or change.

covariance matrix → eigen val & vector.

↓ eigen decomposition

(More variance on more strength

- Step-by-step soln :-

1). Mean centre

2). find co-variance matrix.

3). find the eigen val. & vector for covariance matrix. → convert in 2D, 1D

3D → 3 vector

PC1 (1), PC2 (2), PC3 (3)

How to transform point:-

$f_1 \mid f_2 \mid f_3 \mid$ target

↳ (1000, 4)

PCA

↳ (1,3)

↳ $v \rightarrow v^T = (3,1)$

$U^T \cdot v$

$(1000, 3) \cdot (3, 1)$

$= (1000, 1)$

Shape (1000, 3)

Machine Learning Algorithms:-

linear regression :-

- easy to understand.
- foundation of every ml algo.

supervised type

linear regression

- Simple Linear Regression.
  - 1 target var.
  - 1 data var. (input)

$y = mx + c$ → sort of Linear data.

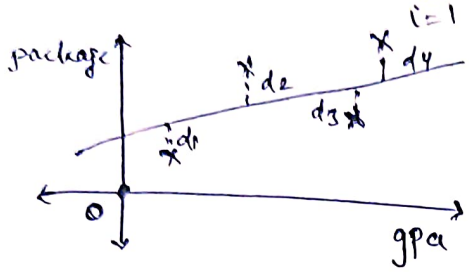
- Multiple Linear regression
  - More than 1 input var.
- Polynomial regression
  - poly. equation [$ax^2 + bx + c$].

(m, c)

- closed form solution
  - direct method.
    - OLS
- Non-closed form.
  - gradient descent.

$B = \bar{y} - m\bar{x}$

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



$\epsilon = d_1 + d_2 + d_3 \cdots d_n$

$\left[\epsilon = d_1^2 + d_2^2 + d_3^2 \cdots d_n^2 = \text{'+ve'}\right]$

$(|d_1| + |d_2| \cdots |d_n|)$ (X)
not differentiable

$\left[\epsilon = \sum_{i=1}^{n} d_i^2\right]$

cost fnct

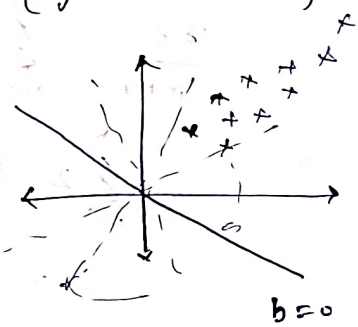
$d_i = (y_i - \hat{y}_i)$

$\left[\epsilon = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2\right]$

$\epsilon_{(m,b)} = \sum_{i=1}^{n} (y_i - mx_i - b)^2$

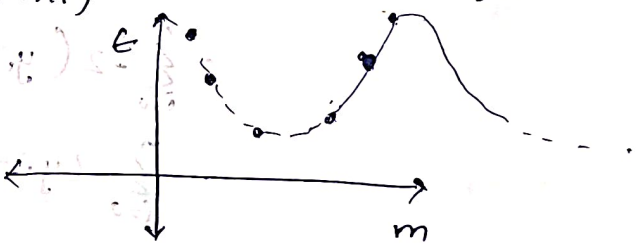
$\{y = f(x)$ $\epsilon(m,b)\}$

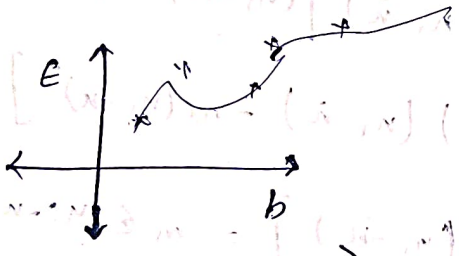
$\left[\epsilon(m,b) = \sum_{i=1}^{n} (y_i - mx_i - b)^2\right]_{\text{minimum}}$



$b = 0, \quad \epsilon(m) = \sum_{i=1}^{n} (y_i - mx_i)^2$

$\epsilon(b) = \sum_{i=1}^{n} (y_i - x_i - b)^2$





local maxima & minima concept :-

$\left[\frac{\partial E}{\partial m} = 0, \quad \frac{\partial \epsilon}{\partial b} = 0\right] \Rightarrow (m,b)$

$$\frac{\partial \epsilon}{\partial b} = \frac{\partial}{\partial b} \sum_{i=1}^{n} (y_i - mx_i - b)^2 = 0$$

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

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

$$\sum_{i=0}^{n} (y_i - mx_i - b) = 0.$$

$$= \sum_{i=0}^{n} y_i - \sum_{i=0}^{n} mx_i - \sum_{i=0}^{n} b = 0$$

divide by n.

$$= \sum_{i=0}^{n} \frac{y_i}{n} \left[- \sum_{i=0}^{n} \frac{mx_i}{n} \right] \sum_{i=0}^{n} \frac{b}{n} = \frac{0}{n}$$

$$= \bar{y} - m\bar{x} - \frac{nb}{n} = 0$$

$$= \bar{y} - m\bar{x} - b = 0$$

$$b = \bar{y} - m\bar{x}$$

$$\varepsilon = \sum (y_i - mx_i - \bar{y} + m\bar{x})^2$$

$$\frac{\partial \epsilon}{\partial m} = \sum_{i=0}^{n} \frac{\partial}{\partial m} (y_i - mx_i - \bar{y} + m\bar{x})^2 = 0$$

$$= \sum_{i=0}^{n} 2 (y_i - mx_i - \bar{y} + m\bar{x})$$

$$= \sum_{i=0}^{n} 2 (y_i - mx_i - \bar{y} + m\bar{x}) (-x_i + \bar{x}) = 0$$

$$= \sum_{i=0}^{n} -2 (y_i - mx_i - \bar{y} + m\bar{x}) (x_i - \bar{x}) = 0$$

$$= \sum_{i=0}^{n} (y_i - mx_i - \bar{y} + m\bar{x}) (x_i - \bar{x}) = 0$$

$$= \sum_{i=0}^{n} [(y_i - \bar{y}) - m(x_i - \bar{x})] (x_i - \bar{x}) = 0$$

$$= \sum_{i=0}^{n} [(y_i - \bar{y})(x_i - \bar{x}) - m(x_i - \bar{x})^2] = 0$$

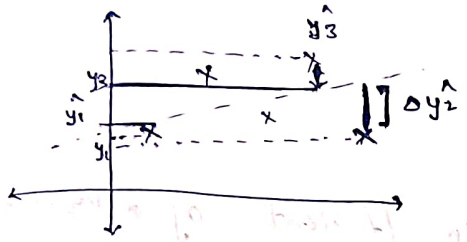
$$= \sum_{i=0}^{n} [(y_i - \bar{y})(x_i - \bar{x})] = m \sum (x_i - \bar{x})^2$$

$$\left[m = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \right]$$

Regression Metrics :-

1) MAE
2) MSE
3) RMSE
4) R2 score
5) Adjusted R-2 score.

⇒ MAE :- Mean absolute Error :-



Absolute error $= |y_1 - \hat{y_1}| + |y_2 - \hat{y_2}| + |y_3 - \hat{y_3}| \ldots$

$=$ = total ↗

mean absolute error $= \frac{|y_1 - \hat{y_1}| + |y_2 - \hat{y_2}| + |y_3 - \hat{y_3}| \ldots}{n}$

$$\left[mae = \sum_{i=1}^{n} \frac{|y_i - \hat{y_i}|}{n} \right] \Rightarrow$$

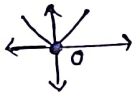
Advantages ⇓

⇒ Same unit (LPA vs LPA) ↑ direct o/p.

⇒ robust to error (detect & less diff. —).

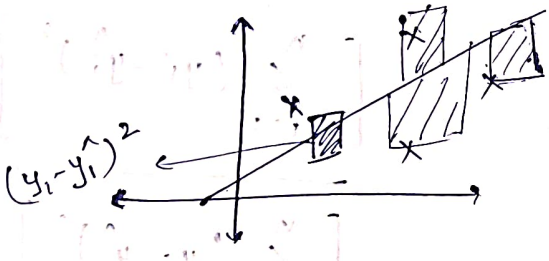
⇓

Disadvantage :-

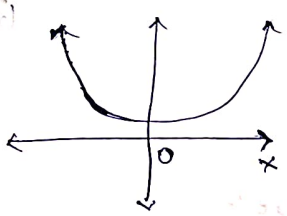
$|z| =$ 

⇒ Mod. fun^n is not diffntiable at '0'.

⇒ MSE : Mean Squared Error :-



→ differentiable at every point.



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

$\{ Y = LPA \quad mse = (LPA)^2 \}$

Advantage :-

use as loss fun^n.

Disadvantage :- Not robust to outliers

RMSE :- Root mean square error.

$= \sqrt{mse}$

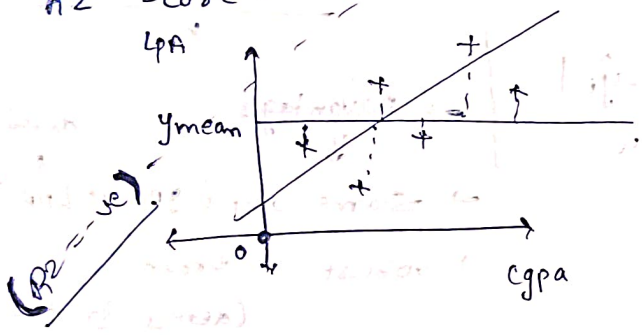
$$\left[= \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{n}} \right]$$

rmse = Lpa ↳ benefit

y = Lpa

disadvantage :- Robust - collision.

R2 score :-



R-2 = co-efficient of determination or goodness of fit.

$$\left[R_2 = 1 - \frac{SS_R}{SS_M} \right]$$

SS_R → sum of squared error in regression line.

SS_M → sum of sq. error in mean line.

$$R_2 = 1 - \frac{\left[\sum_{i=1}^{n} (y_i - \hat{y_i})^2 \right]_{reg.}}{\left[\sum_{i=1}^{n} (y_i - \hat{y_i})^2 \right]_{mean}}$$

Adjusted R-2 score :-

(R-2 score
↳ R-2 adjusted)

$$R^2_{adj.} = 1 - \left[\frac{(1-R_2)(n-1)}{(n-1-k)} \right]$$

R_2 =

n = no. of rows

k = independent

k = 1, 2, 3 ---