Evaluation scheme :

EST – 35

MST – 35

1 Theory quiz – 10 (3 weeks after MST)

1 Lab quiz – 5 (centralized quiz)

Project – 15 (multiple evaluations of the project)

MST syllabus – (Till random forest)

Lec - 4 Data preprocessing - 1

- Object (point, case, same, entity)

↓

attribute (characteristic of object)

↓

types

nominal | binary | ordinal | interval | ratio

binary → symm. (both are equally imp.) | asymm.

ordinal → in meaningful order

interval → measured on scale of equal sized units, values have order, no true zero pt.

ratio → inherent zero point

Interval scale	ratio scale
- no zero pt.	zero pt.
- AM	HM | GM
- ratio can't be calculated	-
- measures size & magnitude as multiple factors	~~as one~~ in terms of one another

- Discrete and continuous attributes
  - integer values
  - floating pt.
- Measuring central tendency: mean, median, mode

Mean $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\mu = \frac{\sum x}{N}$, $\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$

$$\text{Median} = L_1 + \left(\frac{n/2 - (\sum \text{freq})_l}{(\text{freq})_{\text{median}}} \right) \text{width}$$

Mode = most frequent value in data

-

symm-

+ve skewed

-ve skewed

- Quartiles: Q_1 (25 percentile), Q_3 (75 percentile)

  IQR: $Q_3 - Q_1$

Five no. summary: min, Q_1, median, Q_3, max

Boxplot: ends are quartile, mark median, add whiskers, plot outliers

outliers: higher or lower than $1.5 \times IQR$

- Standard deviation = $\sqrt{\text{variance}}$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2 = \frac{1}{n-1}\left[\sum_{i=0}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2\right]$$

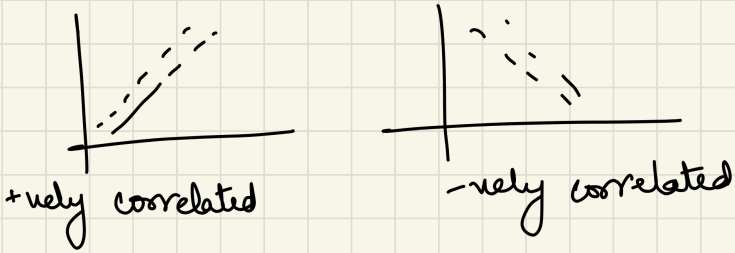
$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{n}(x_i-\mu)^2 = \frac{1}{N}\sum_{i=1}^{n} x_i^2 - \mu^2$$

- lower whisker = $Q_1 - 1.5 * IQR$

  upper whisker = $Q_3 + 1.5 * IQR$

- Histogram vs Bar graph

-



+vely correlated -vely correlated

- Types of data : Record data, data Matrix

  Document data : each doc becomes term vector

- Similarity, dissimilarity, proximity

Attribute type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & x = y \\ 1 & x \neq y \end{cases}$	$s = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$
ordinal	$d = \frac{\lvert x - y \rvert}{n-1}$	$s = 1 - d$
interval or ratio	$d = \lvert x - y \rvert$	$s = -d$, $s = \frac{1}{1+d}$, $s = e^{-d}$

$$s = 1 - \frac{d - \min\_d}{\max\_d - \min\_d}$$

- Euclidean distance = $d(x, y)$

$$d(x,y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

- Minkowski distance

$$d(x,y) = \left(\sum_{k=1}^{n} \lvert (x_k - y_k) \rvert^r \right)^{1/r}$$

'r' is a parameter

$r = 1$, (Hamming distance, Manhattan, city block, L1 norm, taxicab)

$r = 2$, euclidean distance

$r \to \infty$, supremum (Lmax norm, L∞ norm)

- Malanobis distance

$$d(x,y) = \left((x-y)^T \Sigma^{-1} (x-y)\right)^{-0.5}$$

Σ is covariance of matrix.

← for Euclidean distance

- $d(x,y) = d(y,x)$ —— symmetric
- $d(x,z) \leq d(x,y) + d(y,z)$ —— Dar ineq.
- $d(x,y) = 0$ —— if & only if $x=y$

distance that satisfies these properties is a metric.

- similarity matching coefficient count both presence and absence equally and is normally used for symmetric binary attributes.

$$SMC = \frac{f_{00} + f_{11}}{f_{01} + f_{10} + f_{11} + f_{00}}$$

- jaccard coefficient counts only presences and is used for assymmetric binary attributes.

$$JC = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

- **Cosine similarity:**

$$\cos(x,y) = \frac{x \cdot y}{\|x\| \cdot \|y\|}, \quad \|x\| \text{ is length of vector.}$$

$$\|x\| = \sqrt{\sum_{k=1}^{n} x_k^2} = \sqrt{x \cdot x}$$

- **Correlation:**

$$\text{corr}(x,y) = \frac{\text{covariance}(x,y)}{\text{sd}(x) * \text{sd}(y)} = \frac{S_{xy}}{S_x \cdot S_y}$$

$$S_{xy} = \frac{1}{n-1} \sum_{k=1}^{n} (x_k - \bar{x})(y_k - \bar{y})$$

$$S_x = \sqrt{\frac{1}{n-1} \sum_{k=1}^{n} (x_k - \bar{x})^2}$$

$$S_y = \sqrt{\frac{1}{n-1} \sum_{k=1}^{n} (y_k - y)^2}$$

$$\text{corr}(x,y) = \frac{\sum (x_k - \bar{x})(y_k - \bar{y})}{\sqrt{\sum (x_k - \bar{x})^2 \sum (y_k - \bar{y})^2}}$$

- correlation is always b/w -1 to 1.

(Drawbacks: corr might come out to be 0 for non linear data)

Lec 5 : Data preprocessing II

- Data integration : merging data from multiple sources into coherent data stores

problem — schema integration (metadata analysis), feature matching, redundant features (correlation analysis), detection & resolution of data values

- To deal with redundant features, correlation analysis is done.

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

- Data transformation techniques
  - Normalization / scaling
  - Aggregation
  - Generalization
  - Feature construction

- Normalization :

① Min-max :

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

$\frac{47-2}{90-2} = \frac{45}{88} = 512$

② Mean normalisation :

$$x' = \frac{x - \bar{x}}{x_{max} - x_{min}}$$

$\bar{x} = 32.4$

$\frac{2-32.4}{90-2} = \frac{-30.4}{88}$

$= -0.345$

③ Z score : $x' = \frac{x - \bar{x}}{\sigma}$

$$\sigma = \sqrt{\frac{(2-32.4)^2 + (47-32.4)^2 + (90-32.4)^2 + (18-32.4)^2 + (5-32.4)^2}{5}}$$

$= 34.05$

$\frac{2 - 32.4}{34.05} = -0.894$

④ decimal scaling : $x' = \frac{x}{10^j}$

j is max^m count of digits in max^m & min^m value of feature vector corresponding to x.

Here, $j = 2$ $\max(2, 1)$

⑤ Log transformer

$x' = \log(x)$

log reduces impact of too-low & too-high values.

It is used to convert skewed distribution to less skewed distribution.

⑥ Max Abs Scaler (scales data b/w [-1, 1])

$x' = \frac{x}{\max(|x|)}$

$\max(|x|) = 2000$

⑦ Interquartile / Robust Normalization

$x' = \frac{x - Q_1}{Q_3 - Q_1}$

$Q_1 = 5$, $Q_3 = 47$

for finding quartile,

$\{2, 5, 18, 47, 90\}$

↓

median = Q_2

Q_1 = median of $\{2, 5, 18\}$

Q_3 = " " $\{47, 90\}$

- Aggregation: aggregate data to put data in better perspectives.

  Reduces memory consumption
  provides more stable view point than individual data points

- Generalization: generalization from low-level to higher order concepts

- Feature construction: set (input features) → new of powerful features

  When do we need feature construction:

  i) for numeric value inputs — Label Encoding, One hot encoding, dummy encoding

  ii) from numeric to categoric — Rank according to numerical continuous values, Gain Ratio, Gini Index

  iii) text-specific feature construction — BOW, Tf-idf, word embedding

- Label Encoding:

Bridge-Type	Bridge Type
A	0
B	1
C	2

might consider hierarchy

- One hot Encoding:

  ↳ uses N features to represent a feature

Bridge Type	A	B	C
A	1	0	0
B	0	1	0
C	0	0	1

- Dummy Encoding: It uses N-1 features to represent a feature.

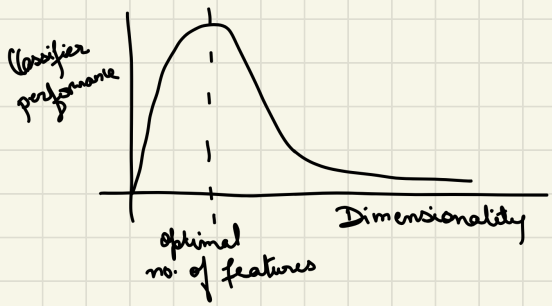
- TF-IDF :

$$\text{Term frequency (TF)} = \frac{\text{No. of occ. of words in sentence}}{\text{No. of words in sentence}}$$

$$\text{Inverse Doc. freq. (IDF)} = \log\left(\frac{\text{No. of sentences}}{\text{No of sentences containing words}}\right)$$

Lecture 6 (Data preprocessing III) :

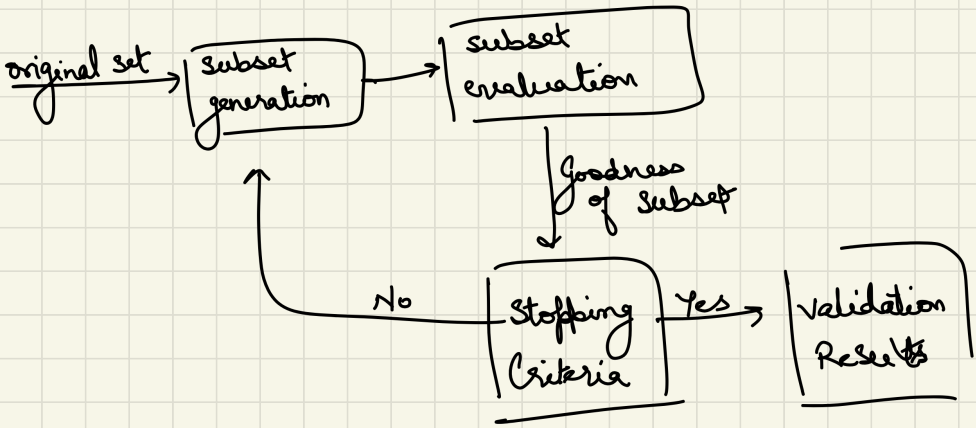
- Dimensionality reduction = no. of input features is called dimensionality



- Benefits of dimensionality reduction
  - Accuracy improvement
  - Overfitting risk red^{n}
  - Improved data visualization
  - Reduce storage cost
  - ↑ in explainability of model
- Data reduction techniques
  - feature extraction
  - feature selection
- Feature selection : finding best of features
  - Maximizing feature relevance
  - Minimizing " redundancy
    - ↳ features that provide similar info

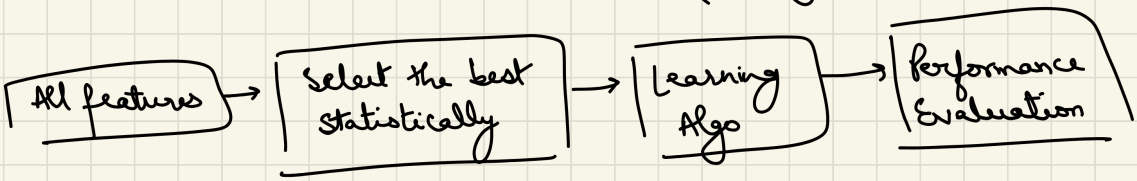
Measuring feature redundancy

- Correlation
- SMC
- JC
- Cosine similarity
- Minkowski distance (euclidian, manhattan)
  - L_1
  - L_2



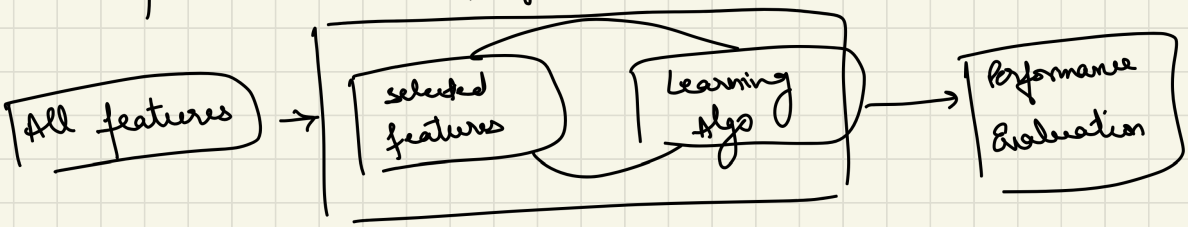
Feature Selection Process

- Filter Approach : feature subset is selected on the basis of statistical measures. (No algo)



- Wrapper Approach :

More expensive but better performance



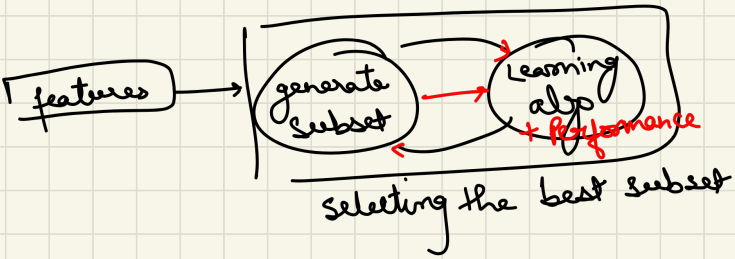
i) ~~forward~~ wrapper selection : we choose the candidate subset & keep selecting the features

ii) Backward wrapper " : we start with whatever features we have & train the model & then see where the model performs the best

ii) Forward wrapper : we see which variables perform the best against the target. Next we select the best feature against the first selected feature.

iii) Exhaustive wrapper : tries all the feature combination and return the best performing model.

- Embedded Approach : takes care of each iteration of model training and carefully extract those features which contribute most to training for a particular iteration



- Feature extraction : creates new features from a combination of original features.

  - PCA (Principal Component Analysis)
  - SVD (single valued decomposition)
  - LDA (Linear Discriminant Analysis)

- PCA is used for dimensionality red^n.

- Steps of PCA :

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

ii) for eigenvalues, $\det(A - \lambda I) = 0$

iii) eigenvector $[A - \lambda I]x = 0$

iv) sort eigenvector in descending order & choose K values.

v) Transform the data along PCA.

eg:

feature						
x	126	128	128	130	130	132
y	78	80	82	82	84	86

Sol^n $\bar{x} = \dfrac{126 + 128 + 128 + 130 + 130 + 132}{6}$, $\bar{y} = 82$

$= 129$

x	$x - \bar{x}$	y	$y - \bar{y}$
126	-3	78	-4
128	-1	80	-2
128	-1	82	0
130	1	82	0
130	1	84	2
132	3	86	4

$$cov(xx) = \frac{1}{n-1}\sum(x-\bar{x})(x-\bar{x})$$

$$cov(xy) = \frac{1}{n-1}\sum(x-\bar{x})(y-\bar{y})$$

$$cov(yx) = \frac{1}{n-1}\sum(y-\bar{y})(x-\bar{x})$$

$$cov(yy) = \frac{1}{n-1}\sum(y-\bar{y})(y-\bar{y})$$

Covariance matrix:

xx	xy
yx	yy

$$cov(xx) = \frac{(-3)^2 + (-1)^2 + (-1)^2 + (1)^2 + (1)^2 + (3)^2}{5} = 4.4$$

$$cov(yy) = \frac{(-4)^2 + (-2)^2 + (0)^2 + (0)^2 + (2)^2 + (4)^2}{5}$$

$$= 8$$

$$\text{cov}(xy) = \text{cov}(yx) = \frac{(-3)(-4) + (-1)(-2) + (-1)(0) + (1)(0) + (1)(2) + (3)(4)}{5}$$

$$= 28/5 = 5.6$$

$$\begin{array}{c|c} 4.4 & 5.6 \\ \hline 5.6 & 8 \end{array} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{array}{c|c} 4.4-\lambda & 5.6 \\ \hline 5.6 & 8-\lambda \end{array}$$

$$|A - \lambda I| = 0$$

$$(4.4-\lambda)(8-\lambda) - (5.6)(5.6) = 0$$

$$\lambda = 0.32, 12.08$$

for eigen vector,

$$\begin{bmatrix} 4.4 & 5.6 \\ 5.6 & 8 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 12.08 \begin{bmatrix} x \\ y \end{bmatrix}$$

here we get,

$$y = 1.37x$$

$$1.37x = y$$

$$v_2 = \begin{bmatrix} 1 \\ 1.37 \end{bmatrix} = \begin{bmatrix} 0.59 \\ 0.81 \end{bmatrix}$$?

$$\begin{bmatrix} 4.4 & 5.6 \\ 5.6 & 8 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0.32 \begin{bmatrix} x \\ y \end{bmatrix}$$

$$4.4x + 5.6y = 0.32x$$
$$4.08x + 5.6y = 0$$
$$x = \frac{-5.6}{4.08}y$$

$$\Rightarrow v_1 = \begin{bmatrix} -1.37 \\ 1 \end{bmatrix} = \begin{bmatrix} -0.81 \\ 0.59 \end{bmatrix}$$

final dataset = feature vector^T * original dataset

$$V = \begin{bmatrix} 0.59 & -0.81 \\ 0.81 & 0.59 \end{bmatrix} \qquad D = \begin{bmatrix} -3 & -4 \\ -1 & -2 \\ -1 & 0 \\ 1 & 0 \\ 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$DV = \begin{bmatrix} PC_1 & PC_2 \\ -5 & 0.1 \\ -2.2 & -0.4 \\ -0.6 & 0.8 \\ 0.6 & -0.8 \\ 2.2 & 0.4 \\ 5 & 0.1 \end{bmatrix}$$

(contains values of PC)

Lec 7 (Linear Regression)

- Regression is mapping f" from input variables to output variable.

$$y = f(x)$$

where y is continous or real valued variable.

- Regression is said to be linear regression if output variable is a linear f" of input variable.
- SLR: output variable is related to a single predictor

$$\hat{y_i} = y_{pred} = \beta_0 + \beta_1 x$$

$$\varepsilon = \text{Error} = y_i - \hat{y_i}$$

$$\text{Total square error} = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (y_i - \hat{y_i})^2$$

(Minimized error)

$$\hat{\beta_1} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

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

$$\hat{\beta_0} = \bar{y} - \hat{\beta_1}\bar{x}$$

y_i	i	x_i	y_i	x_i^2	$y_i x_i$
	1				
	2				
	1				
	.				
	15				
	Σ	24.76	931.17	41.0532	1548.2453

$$\hat{\beta_1} = \frac{15(1548.24) - (24.76)(931.17)}{15(41.053) - (24.76)^2}$$

$$\bar{x} = \frac{24.76}{15}, \quad \bar{y} = \frac{931.17}{15}, \quad \beta_0 = \bar{y} - \hat{\beta_1}\bar{x}$$

- MLR : this model describes how a single response variable Y depends linearly on a number of predictor values

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_K x_K$$

$$\text{Total error} = \sum_{i=1}^{n} \varepsilon_i = \left(\sum_{i=1}^{n} y_i - \beta_0 - \sum_{j=1}^{K} \beta_j x_{ij} \right)$$

$$\varepsilon = y - X\beta$$

$$\varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_K \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1K} \\ 1 & x_{21} & x_{22} & \cdots & x_{2K} \\ \vdots & & & & \\ 1 & & & & \end{bmatrix}$$

- Total square error $= J(\beta) = \sum_{i=1}^{n} \varepsilon_i^2 = \varepsilon^T \varepsilon$

$$= (y - X\beta)^T (y - X\beta)$$

$$= yy^T - 2y^T X\beta + X\beta X^T \beta^T$$

$$\frac{\partial J(\beta)}{\partial \beta} = 0 \Rightarrow \boxed{\hat{\beta} = (X^T X)^{-1} X^T y}$$

$$\frac{\partial^2 J(\beta)}{\partial \beta^2} = 0 \Rightarrow \quad 2XX^T = +ve$$

eg:

x_1	x_2	y
7	560	16.68
3	220	11.50
3	340	12.03

$$A^{-1} = \frac{1}{|A|} adj(A)$$

Soln a) $y = \beta_1 + \beta_2 x_1 + \beta_3 x_2$

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

$$X = \begin{bmatrix} 1 & 7 & 560 \\ 1 & 3 & 220 \\ 1 & 3 & 340 \end{bmatrix} \quad X^T = \begin{bmatrix} 1 & 1 & 1 \\ 7 & 3 & 3 \\ 560 & 220 & 340 \end{bmatrix}$$

$$(X^T X)^{-1} X^T y = \begin{bmatrix} 7.7696 \\ 0.9196 \\ 0.0044 \end{bmatrix} = \hat{\beta}$$

$$y = 7.7696 + 0.9196\beta_1 + 0.0044\beta_2$$

$x_1 = 4, \; x_2 = 80, \; y = $ ——

Lec 7 Linear Regression (Gradient Descent optimization)

$$\hat{y} = \beta_0 + \beta_1 x$$

$$J(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x)^2 \quad \text{—— cost fn}$$

x_i	y_i
1	1
2	2
3	3

S.no	β	
1	1	$\frac{1}{3}(1-1)^2 + (2-2)^2 + (3-3)^2 = 0$
2	0.5	$\frac{1}{3}(1-0.5)^2 + (2-0.5)^2 + (3-0.5)^2 = 1.67$
3	0	4.67
4	1.5	1.67
5	2	4.67

- Gradient Descent Optimization :

gradient descent is an optimizing algorithm to minimize the cost f^n at local minimum

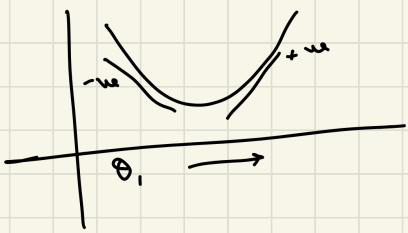
gradient of cost f^n → The higher the gradient, the steeper the slope and the faster the model can learn. If the slope is zero, the model stops learning.

$\frac{\partial J(\beta)}{\partial \beta}$ = gradient of cost f^n, α = step size

$$\beta_j = \beta_j - \alpha \frac{\partial J(\beta)}{\partial \beta}$$

convergence of β_j will happen $\frac{\partial J(\beta)}{\partial \beta} = 0$

- start from any pt. & the f^n will converge at local/global minima.



if $\frac{\partial J(\theta_1)}{\partial \theta_1} < 0$, $\theta_1 = \theta_1 -$ positive quantity

$\frac{\partial J(\theta_1)}{\partial \theta_1} > 0$, $\theta_1 = \theta_1 +$ positive quantity

- Learning rate : Learning rate is fixed.

if learning rate is too small, then convergence will take time.

if " " " high, the gradient descent overshoots min^n.

i) Initialize $\beta_0 = 0$ & $\beta_1 = 0$

ii) update β_0 & β_1 until convergence or for fixed no. of iterations.

$$temp0 := \beta_0 - \frac{\alpha}{n} \times \sum_{i=1}^{n} \beta_0 + \beta_1 x_i - y_i$$

$$temp1 := \beta_1 - \frac{\alpha}{n} \times \sum_{i=1}^{n} (\beta_0 + \beta_1 x_i - y_i) \times x_i$$

$\beta_0 := temp0$

$\beta_1 := temp1$

eg: $\beta_1, \beta_0 = 0$, learning rate = 0.01

Step 1

x_i	y_i
2	15
3	28
5	42
13	64
8	50
16	90

x_i	y_i	$x_i y_i$	x_i^2
11	58		
1	8		
9	54		

Gradient Des. Optimization for MLR :

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_K x_K$$

$$J = \frac{1}{2n} \sum_{i=1}^{n} (y_i - \hat{y})^2 \quad \text{—— cost fn}$$

gradient of cost fn :

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_K x_{iK} - y_i)$$

gradient descent for MLR :

i) Initialize $\beta_0 = \beta_1 = \text{——} = 0$

ii) $$\beta_j = \beta_j - \frac{\alpha}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_K x_{iK} - y_i) \times x_{ij}$$

for $j = 1, 2, \ldots K$

eg. $\beta_0, \beta_1, \beta_2 = 0$, $\alpha = 0.01$

Sol^n

x_{i1}	x_{i2}	y_i	$x_1 x_{i2}$	$x_{i1} y_i$	$x_{i2} y_i$	$(x_{i1})^2$	$(x_{i2})^2$
2.45	5.3	1464					
2.5	5.3	1394					
⋮	⋮	⋮					

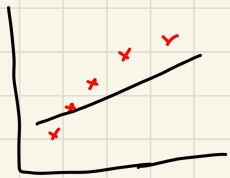
$$\text{temp0} := \beta_0 - \frac{\alpha}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} - y_i)$$

$$\text{temp1} := \beta_1 - \frac{\alpha}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} - y_i) \times x_{i1}$$

$$\text{temp2} := \beta_2 - \frac{\alpha}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} - y_i) \times x_{i2}$$

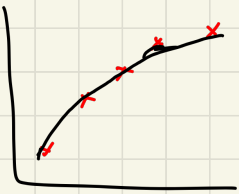
Lec 9 Linear Regression (Problems & Sol"):

-



Underfitting
or
high bias

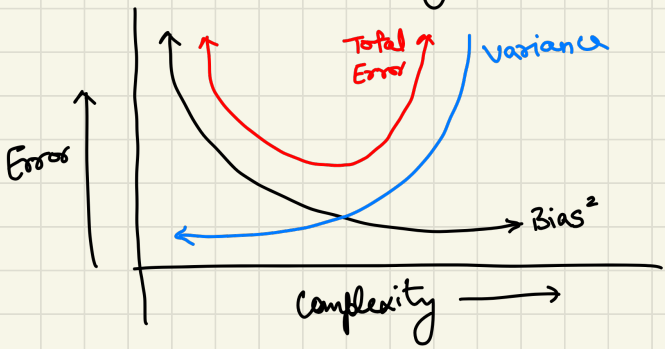
good fit



Overfitting
or
high variance

↓

uses less features, can be fixed by using more features in training data



- variance is an error resulting from fluctuations in the training dataset.
- Bias is an error resulting from faulty assumption in the algorithm.

Sol" to regression problems: i) remove highly correlated predictors (LDA, PCA)
ii) Regularization

Regularization: Keep all the features but reduce magnitude of parameters.

Regularization | Shrinkage

cost f^n = $f(\beta_0, \beta_1, \beta_2, \beta_3, \beta_4) + 5000\beta_3^2 + 5000\beta_4^2$

We add two extra terms at the end to inflate cost of β_3 & β_4. Now in order to get it close to zero, we will reduce values of β_3 & β_4 to near zero.

Regularization

- Ridge Regression (L2 regularization)
- LASSO Regression (L1 regularization) — Least Absolute Selection & Shrinkage operator

Ridge Regression performs L2 regularization. It adds $\Sigma(\text{coeff})^2$.

cost f^n = Mean Square Error + λ (sum of square of coeff.)

$$J = \frac{1}{2n}\sum_{i=1}^{n}\left((y_i - \beta_0 - \beta_1 x_{i1} \cdots - \beta_K x_{iK})^2 + \lambda \sum_{i=0}^{K} \beta_i^2\right)$$

n = no. of training examples λ = regularization parameter

K = no. of features

- Ridge Regression using gradient descent

$$\beta_j = \beta_j - \alpha \frac{\partial(J(\beta))}{\partial \beta} \quad \text{for } i = 1, 2 \ldots K$$

In general, $\frac{\partial J}{\partial \beta_j} = \frac{1}{n}\sum_{i=1}^{n}(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_K x_{iK} - y_i)x_{ij} + \lambda\beta_j$

$\Rightarrow \beta_j = \beta_j\left(1 - \frac{\alpha}{n}\right) - \frac{\alpha}{n}\sum_{i=1}^{n}(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_K x_{iK} - y_i)\times x_{ij}$

since $\boxed{1 - \frac{\alpha}{n} < 1}$, values will shrink.

Ridge Regression using Least square error fit method:

$$\hat{\beta} = (X^T X + \lambda I)^{-1} X^T y$$

It will solve the problem of overfitting and multicollinearity as $|X^T X + \lambda I|$ will not be zero for correlated features.

Lasso Regression (L1 regularization)

It adds $\lambda \sum_{i=0}^{K} |(\text{coeff})|$

Ridge Vs Lasso

Ridge → Keeps all/none of the features
major adv. of ridge regression is coeff. shrinkage and reducing model complexity

Lasso → performs feature selection as well

Ridge prevents overfitting, not very useful in case of a lot of features

Lasso provides sparse soln

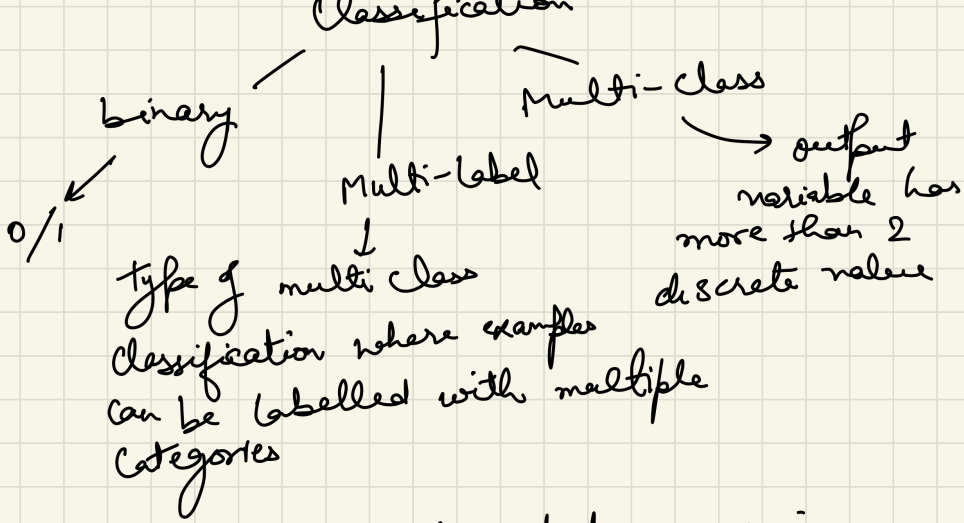
Ridge regression work well in presence of even highly correlated features. Lasso doesn't work that well.

L-10 Classification and Logistic Regression

- Classification uses an algo to find mapping f^m from input variables to output variable.

$$y = f(x)$$

-



- Classification models are not used for regression :

i) Regression model gives continous value of output variable and doesn't give probabilistic values.

ii) Linear regression models are insensitive to imbalance data.

- Logistic Regression :

Logistic regression is similar to linear regression but it is used for classification problems.

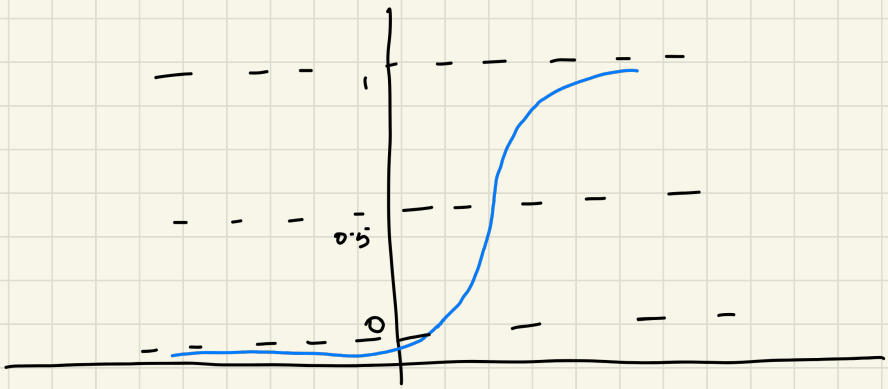
It is used for predicting the categorical dependant variable using a given set of independent variable.

hypothesis f^n : (sigmoid logistic f^n)

$$\hat{y} = f(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x + \dots + \beta_K x_K)}}$$

$x_1, x_2 \dots x_K$ are independent features

hypothesis f^n : $\hat{y} = f(x) = \frac{1}{1 + e^{-z}}$



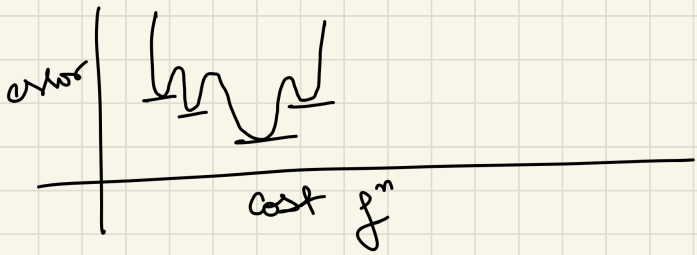
logistic regression must be b/w 0 & 1.

$$\text{Label} = \begin{cases} 1 & \hat{y} \geq 0.5 \quad (z \geq 0) \\ 0 & \hat{y} < 0.5 \quad (z < 0) \end{cases}$$

- The decision boundary (basically eq^n represented by z) is insensitive to balanced or imbalanced data and is characteristic of hypothesis f^n.
- Logistic Regression – cost f^n :

we don't use mean square error method for cost f^n of logistic regression. (as we get multiple minima)

$$MSE = \frac{1}{n} \Sigma \left(y_i - \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_k x_{ik})}} \right)^2$$



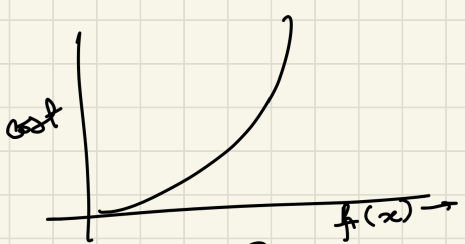
So here,

$$\text{cost fn or error} = \begin{cases} -\log(f(x)) & \text{if } y = 1 \\ -\log(1 - f(x)) & \text{if } y = 0 \end{cases}$$

- The cross entropy fn with the logistic regression fn gives convex curve with one local or global minima.

if $y = 1$

$f(x) = 0$, cost $= \infty$



$y = 0$

$f(x) = 0$, cost $= 0$

- combined eqn:

$$\text{cost} = -y \log(f(x)) - (1-y) \log(1 - f(x))$$

Total error :

$$J = -\frac{1}{n} \sum_{i=1}^{n} \left(y_i \log(f(x_i)) + (1 - y_i) \log(1 - f(x_i)) \right)$$

* gradient descent optimization for logistic regression :

$$\frac{\partial f(x)}{\partial z} = f(x)(1 - f(x)) \frac{\partial x}{\partial z}$$

$$\frac{\partial J}{\partial \beta_j} = \frac{1}{n} \sum_{i=1}^{n} (f(x_i) - y_i) \times x_{ij}$$

(i) initialize $\beta_0, \beta_1 \cdots \beta_K = 0$

(ii) $$\beta_j = \beta_j - \frac{\alpha}{n} \sum_{i=1}^{n} \left(\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_K x_{iK})}} - y_i \right) \times x_{ij}$$

for $j = 0, 1, 2 \ldots K$

Logistic regression for multiclass classification :

Input variables : x_1, x_2

classes : A, B, C

we make beniary classifiers,

$A=1, \quad B,C=0$

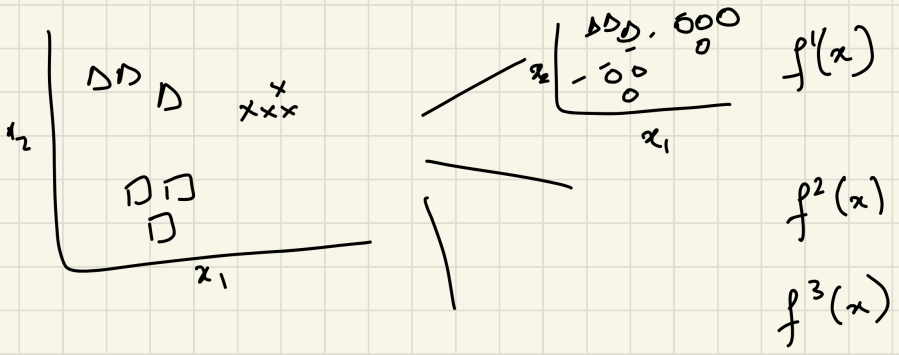
$B=1, \quad A,C=0$

$C=1, \quad A,B=0$

each binary classifier will give probability of i^{th} label for which probability is max^m.

$$f^i(x) = P(y=i|x_1,x_2)$$

$$i = \text{argmax}\, f^i(x)$$



Regularization for Logistic Regression :

- overfitting — decision boundary too good to be true.
  ∴ we shrink the coeff. of input variable.

Ridge regularization :

$$J = \frac{-1}{m}\sum_{i=1}^{m} y_i \log(f(x_i)) + (1-y_i)\log(1-f(x_i)) + \frac{1}{2n}\lambda\sum_{j=0}^{k}\beta_j^2$$

$\frac{\partial J}{\partial \beta_j} = 0$ —— gives a factor of $\left(1-\frac{\alpha\lambda}{n}\right) < 1$

so coeff. will be shrinked

Lec 11 Evaluation measures for Regression

(i) Mean Absolute error : $MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y_i}|$

(ii) Mean Squared error : $MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y_i})^2$

(iii) Root mean Squre error : $RMSE = \sqrt{MSE}$

(iv) R^2 Score : It measures proportion of variation independent variables explained by all the independent variables in the model.

$$R^2 = \frac{\text{Explained variance}}{\text{Total variance}} = \frac{\sum_{i=1}^{n} (\hat{y_i} - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

y_i = actual value, $\hat{y_i}$ = predicted value

$$R^2 = 1 - \frac{\text{Unexplained variance}}{\text{Total variance}} = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

SSE / SST

R^2 lies b/w -1 & 1.

0 % R^2 means model explains none of the variability of data. 100 % R^2 means ____ all ____.

Higher the R^2, the better the model fits the data.

eg: $\hat{y} = 0.143 + 1.229x$

Sol:

x	y	$\hat{y}$	$\frac{\hat{y}-y}{n}$	$\frac{\Sigma\lvert y-\hat{y}\rvert^2}{n}$	$(\hat{y}-\bar{y})^2$	$(y-\bar{y})^2$
2	2					
3	4					
4	6					
6	7					

$$R^2 = \frac{1}{n} \frac{\Sigma(\hat{y}-\bar{y})^2}{\Sigma(y-\bar{y})^2}$$

Adjusted R^2 score: It measures proportion of variation explained by only those independent variable that really affect dependent variables.

- Adjust R^2 score never declines on add^n of an independent variable even if it doesn't affect the dependent variable.

$$\text{Adjusted } R^2 = 1 - \frac{(1-R^2)(N-1)}{(N-K-1)}$$

R^2 = Sample R^2, N = Sample size, K = no. of predictor

Lec 12 sampling methods for supervised learning:

supervised learning occurs when an algorithm learns from example data and associate target responses.

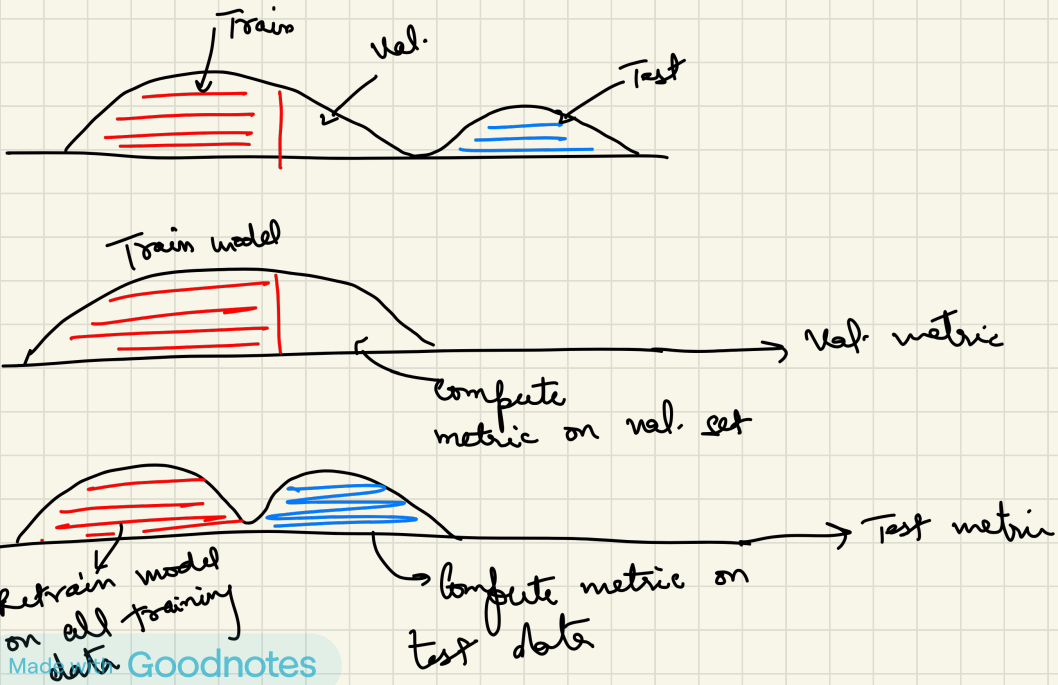
Sampling methods
- Holdout method
- K-fold Cross validation
- Bootstrap Sampling

- Holdout method : Train – Test

overfitting results in good performance with training data but poor performance in test data.

overfitting can be fixed by — K fold method
Holdout val data
feature selection

Holdout of a validation dataset : It is usually for parameter selection & to avoid overfitting



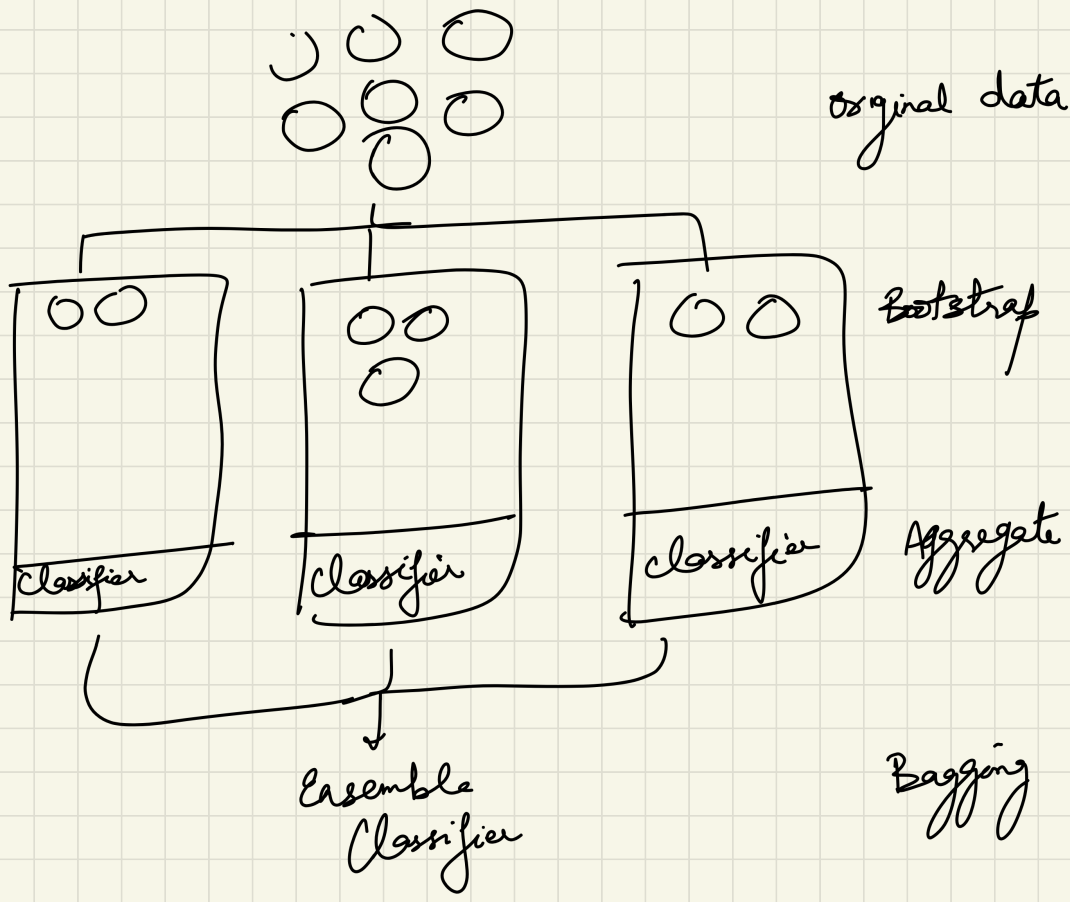
- overfitting can be avoided by :

(i) hold back a validation data set

(ii) using resampling techniques like k-fold cross val.

(iii) remove the data points which have little or no-predictive power

Bootstrap Sampling:



Pros of holdout strategy : Lower computation cost as it runs only once

Cons of holdout strategy : Performance eval. is subject to higher variance given smaller data

K-fold Cross Validation : when dataset is randomly split up in K-groups.

The model is trained on training set & scored on test set & the process is repeated until each unique group is used as test set.

- Leave one cross val. : one set is left for validation & n-1 sets are trained.

Pros of K-fold strategy : less variation as it uses entire data

Cons of K-fold strategy : high computation cost

- Bootstrap Sampling : uses Simple Random Sampling with Replacement (SRSWR)

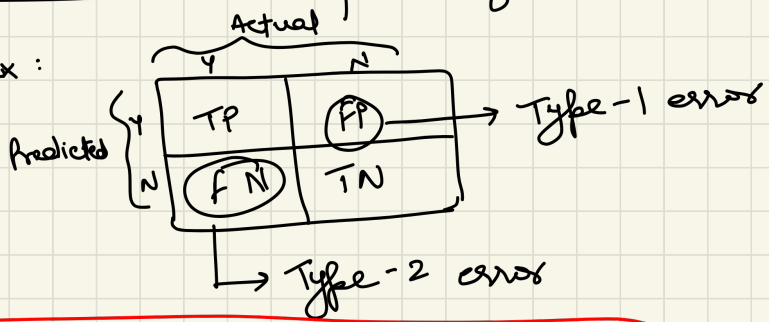
randomly picks input dataset with replacement

Bootstrap sampling used in ensemble algorithm called bootstrap aggregation (bagging).

↳ Avoids overfitting & improves stability of ML algos.

Lec-11 (part-2) Eval. Metric for classification :

- Confusion matrix :



- Accuracy = $\frac{\text{Correct predictions}}{\text{Total predictions}} = \frac{TP + TN}{TP + TN + FP + FN}$

↳ important for balanced datasets

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

- Precision = $\frac{\text{Correct positives}}{\text{Positive predictions}} = \frac{TP}{FP + TP}$

∴ precision must be used for minimizing FP error. (eg. spam mail)

- Recall = sensitivity / hit ratio / true +ve rate

$= \frac{\text{correct positive}}{\text{Total correct +ve predictions}} = \frac{TP}{TP + FN}$

↓ used to minimize FN error. (medical diag)

- Fβ score : weighted HM of precision & recall

$F_\beta \text{ Score} = (1+\beta^2)\frac{\text{Precision} * \text{Recall}}{\beta^2 \text{Precision} + \text{Recall}}$

if $\beta > 1$, we give more weightage to recall.

$\beta = 2$ —— for recall > precision

$\beta = 0.5$ —— " precision > recall

- **F1 score** : equal weightage to precision & recall

$$\text{f1 score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

- **Macro & Weighted precision:**

$$\text{Precision}_{+ve} = \frac{TP}{FP + TP}$$

$$\text{precision}_{-ve} = \frac{TN}{FN + TN}$$

$$\text{Macro} = \frac{\text{precision}_{+ve} + \text{precision}_{-ve}}{2}$$

$$\text{Weighted} = \frac{n_1 \times \text{precision}_{+ve} + n_2 \times \text{precision}_{-ve}}{n_1 + n_2}$$

- **Macro & weighted recall :**

$$\text{Recall}_{+ve} = \frac{TP}{TP + FN} \qquad \text{Recall}_{-ve} = \frac{TN}{TN + FP}$$

$$\text{Macro} = \frac{\text{Recall}_{+ve} + \text{Recall}_{-ve}}{2}$$

$$\text{weighted} = \frac{n_1 \times \text{Recall}_{+ve} + n_2 \times \text{Recall}_{-ve}}{n_1 + n_2}$$

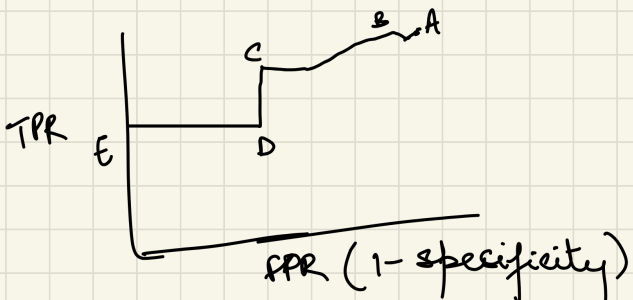
eg. Look at the example !!! (Imp.)

Sensitivity & Specificity:

sensitivity — TPR, recall
specificity — TNR,

In medical diagnosis both are important

ROC: (Receiver operator characteristic)



A has highest sensitivity & lowest specificity

B has same sensitivity as A but higher specificity

Relate specificity with negative class pts & sensitivity with positive class pts

Area under curve is measure of ability of a classifier to distinguish b/w two classes. (AUC)

Lec-13 Data preprocessing IV:

Single Valued decomposition : factorization of a real or complex matrix

$$SVD = U\Sigma V^T$$

eg: Find SVD of A where

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

Sol^n

$$AA^T = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix} = \begin{bmatrix} 17 & 8 \\ 8 & 17 \end{bmatrix}$$

$$|AA^T - \lambda I| = \begin{vmatrix} 17-\lambda & 8 \\ 8 & 17-\lambda \end{vmatrix} = (17-\lambda)^2 - 64 = (\lambda-25)(\lambda-9)$$

$$\lambda = 25, 9$$

singular values $= \sqrt{25}, \sqrt{9} = 5, 3$

$$\Sigma = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix}$$

eg: $A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ -1 & 1 \end{bmatrix}$

Sol^n $A^TA = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$

Step 1 : compute V (eigen values matrix)

Step 2 : compute Σ.

Step 3 : eigen vector for AA^T.

For eigen value & eigen vector,

$\det|A^TA - \lambda I| = 0$, $\begin{pmatrix} 2-\lambda & 0 \\ 0 & 3-\lambda \end{pmatrix} = 0$

$\lambda = 3, 2$

for SVD write λ_i in $\downarrow$ order.

for $\lambda = 3$,

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

$-a + b = 0$

$0a + 3b = 3b$

$\Rightarrow a = 0, b = 1$ $\Rightarrow x_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0/1 \\ 1/1 \end{bmatrix}$

Similarly $x_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

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

(2) $\sigma_1 = \sqrt{\lambda_1} = \sqrt{3}$

$\sigma_2 = \sqrt{\lambda_2} = \sqrt{2}$

No. of non zero eigen value = Rank of matrix (R)

$$\Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix}$$ i.e. Σ_1 is a diagonal matrix whose diagonal entries are σ_1, σ_2.

$\Sigma_1 = R \times R = 2 \times 2$

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

(3) $AA^T = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}$

$\lambda = 3, 2, 0$

find eigen vectors and write U.

SVD : (1) eigen vector for AA^T.
(2) Σ
(2) eigen vector for A^TA.

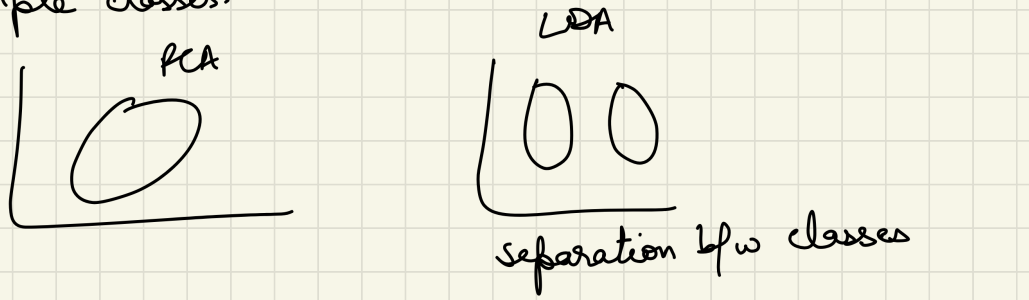
- SVD is used for dimensionality reduction by using compressed SVD.
- SVD is useful for dimensionality red" for a sparse matrix.
  eg: recommender system, TF-IDF, BoW

Relation b/w PCA & SVD ??

LDA (Linear Discriminant Analysis):

PCA & LDA both are linear transformation techniques that are used for dimensionality red".

But PCA is unsupervised algo. that maximizes variance
LDA is supervised and it maximizes the separation b/w multiple classes.



eg: $C_1 \Rightarrow X_1 = (x_1, x_2) = \{(4,1), (2,4), (2,3), (3,6), (4,4)\}$

$C_2 \Rightarrow X_2 = (x_1, x_2) = \{(9,10), (6,8), (9,5), (8,7), (10,8)\}$

Step 1: Compute within class-scatter matrix (S_W):

$$S_W = S_1 + S_2$$

S_1 = cover matrix of C_1

$$S_1 = \sum (x - \mu_1)(x - \mu_1)^T$$

$$\mu_1 = \left\{ \frac{4+2+2+3+4}{5}, \frac{1+4+3+6+4}{5} \right\}$$

$$\mu_1 = [3 \quad 3.6]$$

$$\mu_2 = [8.4 \quad 7.6]$$

$$x_1 - \mu_1 = \begin{bmatrix} 4 & 2 & 2 & 3 & 4 \\ 1 & 4 & 3 & 6 & 4 \end{bmatrix} - \begin{bmatrix} 3 \\ 3.6 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -1 & -1 & 0 & 1 \\ -2.6 & -0.4 & -0.6 & +2.4 & 0.4 \end{bmatrix}$$

$$(x-\mu_1)(x-\mu_1)^T = \begin{bmatrix} 1 \\ -2.6 \end{bmatrix} \begin{bmatrix} 1 & -2.6 \end{bmatrix} = \begin{bmatrix} 1 & -2.6 \\ -2.6 & 6.76 \end{bmatrix}$$

$$\vdots$$

$$\sum (x-\mu_1)(x-\mu_1)^T =$$

Add all the above to get S_1 (& divide each element by n)

$$S_1 = \begin{bmatrix} 0.8 & -0.4 \\ -0.4 & 2.6 \end{bmatrix}$$

$$S_2 = [8.4 \quad 7.6]$$

$$S_W = S_1 + S_2 = \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 5.28 \end{bmatrix}$$

Step 2: Between class scatter matrix (S_B)

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

Step 3: find best LDA projection vector

$S_W^{-1} \cdot S_B = \lambda V$ → projection vector

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

we get $\lambda = 15.65$ (highest λ)

then find the eigen vector.

(or)

$$W = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = S_W^{-1} (\mu_1 - \mu_2)$$

→ projection vector

Step 4: $Y = W^T X$ → input data sample

Lec 2 Types of ML:

- supervised learning: Mapping of x to Y to find $Y = f(x)$.

Classification, Regression (continuous input)

- unsupervised learning discovers patterns from data.

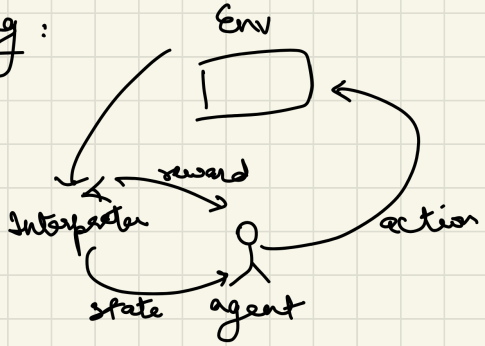
Clustering (segregation), Learning Associations, Dimensionality Red^n

↓ checks for dependency of one data item on another

- Apriori
- FP growth
- ECLat

→ transformation of data from low to high dimensional space while retaining some meaningful properties.

- Reinforcement learning :



- Semi supervised learning : Supervised + unsupervised

- Multi Instance learning : data is arranged in bags & entire bag is labelled
(if one entity fits, entire bag is true & vice-versa)

- Multi Task learning : Word embeddings, Spam filtering
↳ one dataset addresses multiple related problems

- Ensemble learning : Two or more models are fit & their results are combined

- Transfer learning : NLP, Image Classification

- Supervised learning
  - Classification
    - Binary
    - Multilabel
  - Regression

In regression, the input variables are mapped to a continuous fn.

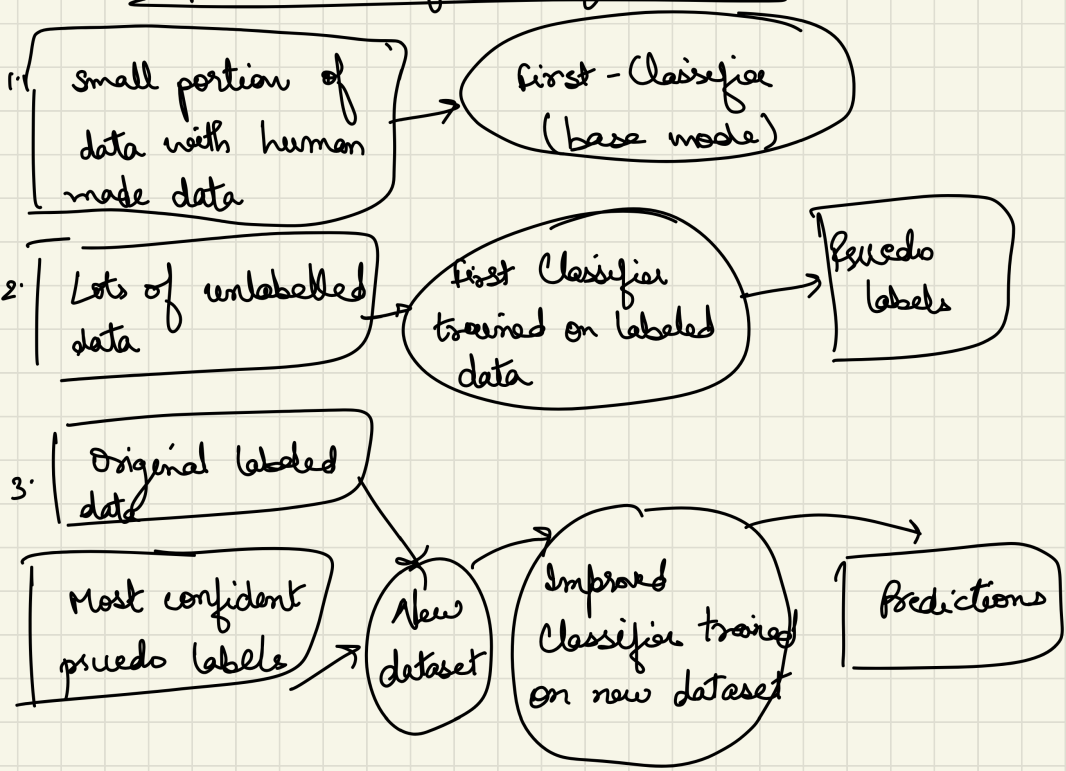
- unsupervised learning : algo. determine data on its own and restructures data into something else such as new features that may represent a class or new series of uncorrelated values.

- clustering : dividing population into no. of groups such that data pts in same group are more similar to other data pts in same group than those in other groups

- Reinforcement learning : Concerned with how intelligent agents ought to take actions in an env. to maximize the notion of cummulative reward

- Self - Supervised learning : It is a more advanced version of unsupervised learning which requires supervisory data along with it.

- Semi-Supervised Self-Training Method :



Lec-1 : Intro to ML

ML : field of study that gives computers the ability to learn without being explicitly programmed.

- T : task T
  E : experience E
  P : performance measure P

computer learns from E of some class T against performance measures P

- playing checkers
  T : playing checkers
  E : having played tens of thousands of games
  P : ability to win against new opponent

- Stock price pred. / House price pred. / Bitcoin price pred.

T : ↑

E : Tens of thousands of prev. year stock prices or housing prices based on factors like (no. of rooms, area, no of floors, crime rate)

P : how close the predicted price is to the actual price using mean error, RMSE

- Task (T): Image Segmentation / Breast Cancer Detection / Cartoonify or Emojify an Image

E: Images with labelling of diff. obj (for img segmentation)
ROI area (for breast cancer detection)
emoji's or cartoons are to diff. (for cartoonify)
moods in image

P: whether or not image is properly segmented or classified measured in terms of no. of metrics such as precision, recall.

COMPLETE THIS!

Lec – Data Preprocessing – 1

– Structured	unstructured data
i) can be displayed in rows	can't be
ii) No. & strings	images, audio, video
iii) requires less storage	requires more storage
iv) Easier to manage and protect with legacy sol"	more difficult to manage and protect with legacy sol"

- Data preprocessing : phase of any ML process, which transforms or encodes the data to bring it to such a state where it can be easily interpreted by the learning algo.

- Need of data preprocessing : Messy, Noisy, inconsistent,

- unstructured data (Text):
  - LowerCase
  - Normalization (remove punctuation)
  - Stopwords
  - Stemming / Lemmatization

unstructured data (Images):
- Read image
- Resize img
- Denoise
- Segmentation
- Morphology (smoothing edges)

- Data preprocessing – Cleaning
  - Integration
  - Transformation
  - Reduction

Data Cleaning – procedure to clean data by filling in missing values, smoothening noisy data, identifying & removing outliers

- Dealing with missing values : remove the entity, imputation, manually filling
  - remove the entity → loss of data
  - manually filling → too difficult
  - imputation → Hot Deck, Cold Deck, Model Based
    - Model Based → Nearest Neighbour, Tree based

- Central tendency imputation : Replacing by mean, not effective for outliers, can be replaced by median in such cases

- Hot Deck Imputation : computes how many number of features have same values in entire training example and choose it for replacement

  Cold Deck " : same as hot deck imputation, the data set selected is diff from the one under consideration.

Nearest Neighbour : rely on distance metrics, evaluate dist. b/w receipient & donors

- Noise : Random variance in a measured variable

  Reason — Malfunctioning of collection instruments
  - Data entry lags
  - Data transmission problems

  Methods → Binning method, Regression, Outlier Analysis

Ex 9, 21, 29, 28, 4, 21, 8, 24, 26

Sol" ① ascending order : 4, 8, 9, 21, 21, 24, 26, 28, 29

② Bin size = $\frac{29-4}{9} = 2.77 \approx 3$

③ Bin size : 4, 8, 9
21, 24, 24
26, 28, 29

④ Data smoothening : 7, 7, 7 (replace by mean)
22, 22, 22
27, 27, 27

- Outlier Analysis :

outlier - object that deviates significantly from rest of objects

- Univariate : extreme value on one variable
- Multivariate : combination of unusual scores ~~of~~ at least two variables

outlier Analysis Range : (Median - IQR*1.5 , Median + IQR* 1.5)

- Extreme Values Analysis : Z-score

$$Z_i = \frac{x_i - \mu}{\sigma}$$

An outlier is then a normalized data pt., which has an absolute value greater than Z_{thr}.

$|Z_i| > Z_{thr}$

- Outlier Analysis :

Linear Model : Projection methods that model the data into lower dimensions using linear correlations

eg. PCA

Proximity based Model : Data instances that are isolated from mass of data as determined by cluster, density or nearest neighbour analysis.

Information Theoretic Model : Outliers are detected as data instances that increase the complexity.

After MST !!

Lec-15 : Random forest

- Decision trees are easy to build, use and interpret but
  i) they work well for data used to create them, but are not flexible when it comes to classifying new examples.
  ii) They are computationally quite expensive.

- Random forest :

Random forest combine simplicity of decision trees with flexibility resulting in vast improvement in accuracy.

Random forest are ensemble learning methods for classification and regression.

Ensemble learning is the process by which multiple models such as classifiers, are strategically generated and combined to solve a particular computational intelligence problem in order to improve the performance of model.

Random forest operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of classes (classification) or mean / avg. prediction (regression) of the individual trees.

*** Random Forest Algo :

Random Forest Algorithm

Consider a training set with N examples and D- dimensions

1. For b=1 to B trees:
   a. Draw a bootstrap sample Z* of size N from the training examples.
   b. Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the D-variables (generally m<<D)
      ii. Pick the best variable/ split variable among the m according to any of the evaluation metric such as Information Gain , Gain Ratio, or Gini Index
      iii. Split the node into its daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$ i.e. outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

This kind of ensemble where the output from various bootstrapped sample is computed by voting for majority or averaging is called **bagging (or bootstrap aggregating).**

Step 1 : Create a bootstrapped dataset

randomly select samples in the original dataset (we are allowed to pick the same sample, more than once)

Step 2 : Create a decision tree, using bootstrapped dataset, but only use a random subset of variables (columns) at each step.

randomly select 2 variables for the root, check better performing variable & put it in the root
(& same for other level nodes)

Make multiple such trees.

Now for a particular sample (~~from test data~~), check the output from all trees generated & see what is the output that occurs max. no. of times.

- bootstrapping the data using the aggregate to make a decision is called "bagging".
- but we missed some of the entries in the bootstrapped dataset, called "Out-of-bag" dataset.
- Ultimately, we can measure how accurate our random forest is by proportion of Out-of-Bag samples that were correctly classified by the Random Forest.

- The proportion of Out-of-Bag samples that were incorrectly classified is the "Out-of-Bag" error.

- Now we choose 1 variable to make a no. of decision trees & in this way we try a bunch of different settings.

- Also we come across two types of missing data,
  ① missing data from columns while building random forest. (fill with initial guess then refine)
  ② missing data from the test sample columns

Tuning parameter for Random forest :

① How to determine optimal no. of features to consider at each split point.
  - Choose optimal no. of features
  - calculate out of bag error
  - Choose optimal value for which error is min^{m}

② How to determine no of trees the algo builds ?
  - more no. of trees, better
  - same method as ①

③ What features to select at each step?
- usually at random
- But we can compute correlation among randomly selected features at each step and reject features that are highly correlated.
- We must also check correlation bw features and output variable as we need features that have at least some predictive power.

- Advantages of Random forest:

① Random forest classifier has improved accuracy over decision tree classifier.

② Random forest are computationally less expensive

③ Random forest can be used to rank the importance of variables in a regression or classification problem in a natural way.

④ can also be used to handle missing values in dataset.

Lec-16: Naive Bayes Classifier:

- It is a probabilistic classifier that uses Bayes theorem & Naive assumption,

Acc to Bayes theorem, $P(A|B) = \frac{P(A) \cdot P(B|A)}{P(B)}$

Acc. to naive assumption, prob. of each feature in the input is conditionally independent of each other.

$$y^* = \arg\max_y P(y_i) \prod_{i=1}^{n} P(x_i | y_i)$$

- eg ① $P(\text{Yes}) = \frac{9}{14}$ $\quad P(\text{No}) = \frac{5}{14}$

② $P(\text{Outlook} = \text{Rainy} \mid \text{Yes}) = \frac{P(\text{Rainy})\,(P(\text{yes}|\text{rainy}))}{P(\text{yes})}$

$$= \frac{\frac{5}{14} \cdot \frac{2}{5}}{\frac{9}{14}} = 2/9$$

$P(\text{Outlook} = \text{rainy} \mid \text{No}) =$

⋮

⑧ Test : < Rainy, Cool, high, True >

$P(\text{Yes} \mid \text{Test}) = ?$

$= P(\text{yes}) \cdot P(\text{Rainy}|\text{yes}) \cdot P(\text{Cool}|\text{yes}) \cdot P(\text{high}|\text{yes}) \cdot P(\text{True}|\text{yes})$

$$= \frac{5}{14} \cdot \frac{3}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} = 0.007936$$

$P(\text{No} \mid \text{Test}) = 0.013714$

$\Rightarrow$ (No)

- Advantages of Naive Bayes Classifier :

① Simple and easy to understand
② No hyper-parameter tuning is req.
③ It is scalable i.e. if a new instance is added it is easy to adjust class prior and likelihood probabilities.
④ It can be used for real time classifications.
⑤ It is suitable for multi-class classification.

- Naive Bayes Classifier : Problems

① Zero frequency problem :
if value is missing, will be labeled as zero & we'll get prob. as zero.

Sol^n → Smoothing technique
we add a parameter α in numerator and $\alpha \times$ no. of features in denominator

$$P(x_j = c | y_i) = \frac{n_{x_j=c, y_i} + \alpha}{(n_{y_i} + \alpha) \times K}$$

where K is no. of features, α is added so that prob. is never 0 and $\alpha \times K$ is added in denominator so that prob is never greater than 1.

$\alpha = 1 \longrightarrow$ Laplace smoothing

$\alpha < 1 \longrightarrow$ Lidstone smoothing

α should not be greater than 1 as it will give higher prob. mass to zero freq. counts.

(2) Independence Assumption :

(Naive assumption)

Sol^n $\longrightarrow$ dimensionality red^n if feature are correlated

$\therefore$ of this feature, naive bayes is best for text classification.

(3) Numerical underflow :

we know likelihood is,

$$P(y_i|x) \propto p(y_i) \prod_{j=1}^{K} p(x_j|y_i)$$

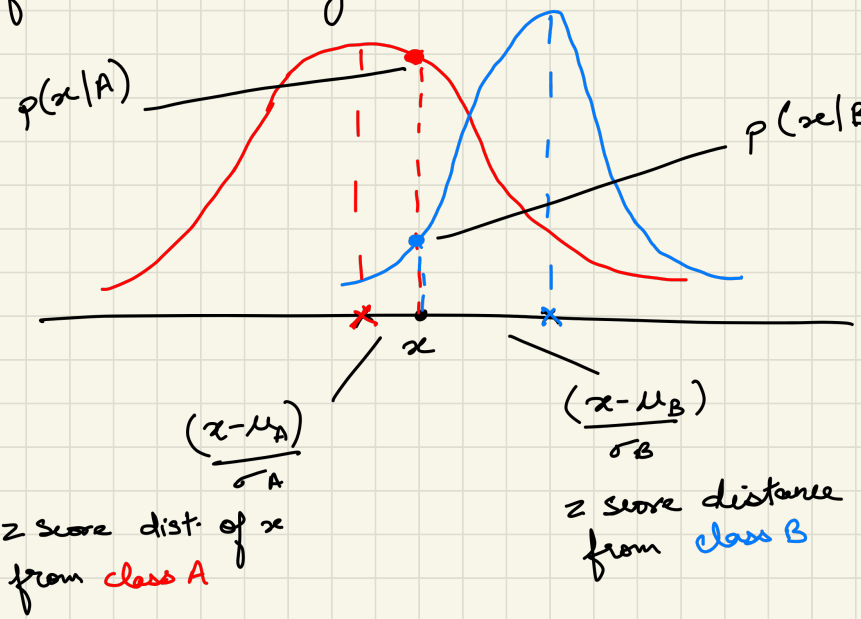
If K (no. of features are large) then product is small and approx to zero. This problem is called numerical underflow

Sol^n $\longrightarrow$ compute log likelihoods

$$\log(P(y_i|x)) \propto \log(p(y_i)) + \sum_{j=1}^{K} \log(P(x_j|y_i))$$

Variant : Gaussian Naive Bayes Classifier :

- The version of Naive Bayes algo that deal with continous feature values is called Gaussian Naive Bayes classifier. cts values are assumed to be distributed according to a Gaussian dist. So we use z-score for observing each feature value given a label.



- Probability of observing any feature value c for feature x_j given a class label y_i is computed as :

$$P(x_j = c \mid y_i) = \frac{1}{\sqrt{2\pi\sigma^2_{x_j,y_i}}} e^{-\frac{1}{2}\left(\frac{c-\mu_{x_j,y_i}}{\sigma_{x_j,y_i}}\right)^2}$$

$$\sigma = \sqrt{\frac{\Sigma(x_i - \mu)^2}{N}}$$

KNN (K-nearest neighbour) :

- KNN is instance based supervised learning as it constructs hypotheses directly from training instance themselves
- based on the fact that similar objects lie in the close proximity to each other
- KNN is used for classification and regression :
  1. In KNN classification, output is class membership with obj being assigned to the class most common among its K nearest neighbours.
  2. In KNN regression, output is a property value for the obj. This value is avg. of values of K nearest neighbours.

KNN Algo:

- K-Nearest Neighbor algorithm works in the following steps:
  1. Determine parameter K = number of nearest neighbors.
  2. Calculate the distance between the query-instance and all the training samples.
  3. Sort the distance and determine nearest neighbors based on the K-th minimum distance.
  4. Gather the category Y of the nearest neighbors.
  5. Use simple majority of the category of nearest neighbors as the label of the query instance in case of classification and average of the values of k nearest neighbors in case of regression.

(For classification, K must be set as even number – so as to avoid tie during simple majority)

① value of K : $\sqrt{n/2}$ or odd no for classification

- small value of K means that noise will have higher influence on the result and a large value makes it computationally expensive.

if K = 1 then algo behaves as overfitting and gives a non smooth decision surface.

As K increases, our decision gets smoother and if we choose K as very large then our algo behaves as underfitting and it gives a smooth decision surface and everything becomes one class which is majority class in our dataset.

Dice coefficient :

$$S_{A,B} = \frac{2 \sum_{j=1}^{n} x_{jA} x_{jB}}{\sum_{j=1}^{n} (x_{jA})^2 + \sum_{j=1}^{n} (x_{jB})^2}$$

eg

x_1	x_2	Y = classification
7	7	B
7	4	B
3	4	G
1	4	G

$x_1 = 3, x_2 = 7$

Sol" ① find K.

Let K = 3

② Calculate distance

x_1	x_2	dist		
7	7	$\sqrt{(7-3)^2+(7-7)^2} = 4$	③	B
~~7~~	~~4~~	~~$\sqrt{(7-3)^2+(7-4)^2} = 5$~~	~~④~~	~~B~~
3	4	$\sqrt{(3-3)^2+(4-7)^2}$	①	G
1	4	$\sqrt{(3-1)^2+(4-7)^2} = \sqrt{13}$	②	G

⇒ (Good)

Advantages of KNN:

(under-partitions)

① Prediction quality

② Short cycle (No training)

(large no. of classes)

③ Multi-class classification

④ Interpretability

Limitations of KNN:

① Costly inference

Reducing Inference Cost:

① Sub sampling

② Dimension reduction — using L2 & cosine distance (not good for sparse data)

③ Avoiding region quickly (clustering)

Association Metric	Formula
1. Minkowski Distance	$d(X_i, X_j) = \left(\sum_{k=1}^{p} \lvert X_{ik} - X_{jk}\rvert^m\right)^{1/m}$
2. Manhattan/ City-Block Distance	$d(X_i, X_j) = \sum_{k=1}^{p} \lvert X_{ik} - X_{jk}\rvert$
3. Euclidean Distance	$d(X_i, X_j) = \sqrt{\sum_{k=1}^{p} \lvert X_{ik} - X_{jk}\rvert^2}$

Association Metric	Formula
4. Canberra Distance	$d(X_i, X_j) = \sum_{k=1}^{p} \frac{\lvert X_{ik} - X_{jk}\rvert}{X_{ik} + X_{jk}}$
5. Czekanowski Coefficient	$d(X_i, X_j) = 1 - \frac{2\sum_{k=1}^{p} min\,(X_{ik}, X_{jk})}{\sum_{k=1}^{p}(X_{ik} + X_{jk})}$

Distance Metrics for Categorical Features

- If x and y are the feature vectors for two categorical variables, than following metrics are used for K-NN Classifier:

Association Metric	Formula
1. Cosine Distance	$d(x, y) = 1 - \frac{x.y}{\lvert x\rvert\lvert y\rvert}$
2. Tanimato Coefficient	$d(x, y) = 1 - \frac{x.y}{\lvert x\rvert^2 + \lvert y\rvert^2 - \lvert x\rvert\lvert y\rvert}$
3. Jaccard Distance	$d(x, y) = 1 - \frac{\lvert x \cap y\rvert}{\lvert x \cup y\rvert}$
4. Sorensen-Dice Coefficient	$d(x, y) = 1 - \frac{2\lvert x \cap y\rvert}{\lvert x\rvert + \lvert y\rvert}$
5. Hamming Distance	$d(x, y) = \sum_{i=1}^{k} \lvert x_i - y_i\rvert$

Lec 18.1 Association Rule Learning :

- Associating one feature with other
- $X \rightarrow Y$ (X: antecedent, Y: consequent)
- Evaluating Association rule strength:

① support
② confidence
③ lift

- **Support** :

$$\text{support}(x) = \frac{\text{no. of times } x \text{ appears in transactions}}{\text{no. of transactions}} = \frac{n(x)}{N}$$

$$\text{support}(x, y) = \frac{n(x \cap y)}{N}$$

- **confidence** : (Chances) $X \rightarrow Y$

$$\text{confidence} = P(Y|X) = \frac{P(X \cap Y)}{P(X)} = \frac{\text{support of } X \text{ and } Y}{\text{support of } X}$$

A high level of confidence shows that rule is often enough to justify a decision based on it.

- **lift** : lift is a metric that considers joint frequency of X and Y, frequency of X and frequency of Y.

$$\text{lift}(X \to Y) = \frac{n(X \cap Y) \times N}{n(X) \times n(Y)} = \frac{P(Y|X)}{P(Y)} = \frac{P(X \cap Y)}{P(X) \cdot P(Y)}$$

- **Association Rule Mining – Naive Algo** :

Approach to find association rules that satisfy the threshold value of support and confidence.

It is naive approach because it works by generating every possible pairs of items in transaction database.

1. find frequency of each item and all possible item pairs.
2. identify item pairs that qualify threshold value of support
3. generate and identify rules that qualify the threshold value of confidence.

— Given d items, there are 2^d possible candidate itemsets.

Limitations of Naive Algo:

for n items in the transaction database, Naive algo needs to find frequency of $2^n - n - 1$ subsets

An improved version of Naive Alg. is to compute only non-zero frequency pairs.

This can be done by generating item pairs for items that appear in the same transaction.

Lec 18.2: Apriori Algo for Association Rule Mining:

- works in 2 phases:

Phase 1: Identification of frequent item sets → support ≥ threshold

Phase 2: Generation of association mining rules

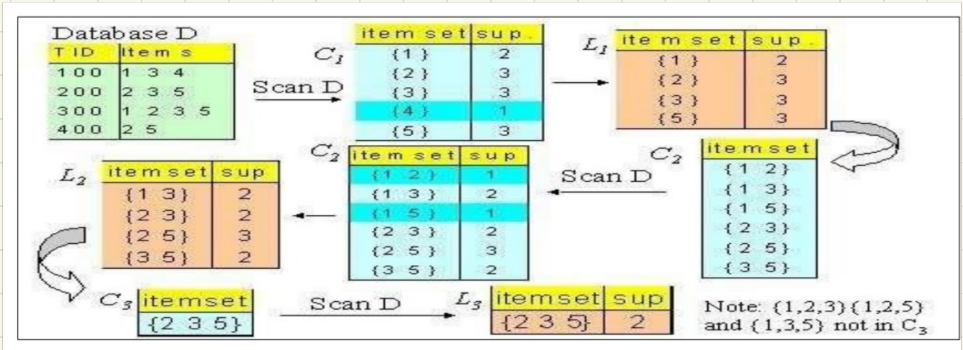
Algo??

- support (min^m) = 50%, confidence (min^m) = 75%

TID	items
100	1 3 4
200	2 3 5
300	1 3 5 2
400	2 5

Sol^n

(1)



confidence $(2 \to 3,5) = 2/3$

$(3,5 \to 2) = 2/2$ ✓

$(3 \to 2,5) = 2/3$

$(2,5 \to 3) = 2/3$

$(5 \to 2,3) = 2/3$

$(2,3 \to 5) = 2/2$ ✓

(2)

$(3,5 \to 2) = 1$ ✓

$(3 \to 2) = 0.67$

$(5 \to 2) = 1$ ✓

$(2,3 \to 5) = 1$ ✓

$(2 \to 5) = 1$ ✓

$(3 \to 5) = 1$ ✓

Improvement of Apriori Algo : Take a look at the example from slides

Lec 19 : Clustering

- unsupervised technique that involves automatically discovering natural grouping in data
  points are divided such that data points in one group are more similar
  segregate groups with similar traits into clusters
- Applications : City Planning, Market Analysis, Social N/w analysis, Organize Computing Clusters
- Eval. metrics : DRAPS No ground truth

① Silhouette Coeff. :

$$S = \frac{b-a}{\max(b,a)}$$

a = mean dist. b/w a sample and all other points the same cluster
b = mean dist. b/w a sample and all other points in the nearest cluster

Score is bounded b/w -1 and 1 for incorrect clustering and +1 for highly dense clustering.

score around zero for overlapping clusters.
score is higher when clusters are dense and well separated.

(2) **Dunn Index** :

$$D = \frac{\text{min separation}}{\text{max diameter}}$$

amongst clusters

→ ek hi cluster main

$$0 < D < \infty$$

more compact clusters, D ↑

(3) **Rand Index** : measure of how similar clustering results or groupings are to ground truth

$0 < RI < 1$, 1 being best

$$RI = \frac{A + B}{{}^{n}C_{2}}$$

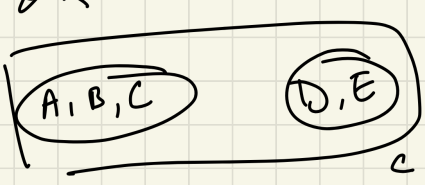
C = ground truth class labelling

K = clustering assignment

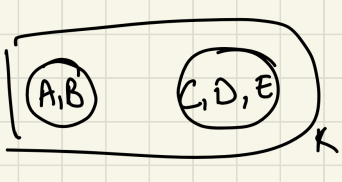
A = no. of element pairs that lie in same set in C & K

B = no. of element pairs that lie in diff sets of both C & K

eg:



but



RI = ?

Sol: 5 items $= {}^5C_2 = \frac{5!}{3!\,2!} = 10$

A = 2 (A,B) (D,E)

B = 4 (A,D) (A,E) (B,D) (B,E)

$$RI = \frac{2+4}{10} = 0.6$$

② **Adjusted Rand Index**: Its drawback is that it yields a high value for pairs of random partitions of a given set.

To fix, we consider grouping by chance.

no. of items C_2

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

a_i = sum of contingency cells in row i

b_j = " " " " column j

n_{ij} = no. of examples common to cluster i & j

⑤ **Purity** : external evaluation criterion of cluster quality

% of total no. of objects that were classified correctly.

$0 < \text{Purity} < 1$ ← best

$$\text{Purity} = \frac{1}{N} \sum_{i=1}^{K} \max_j [c_i \cap t_j]$$

N = no. of objects
K = no. of clusters

c_i = cluster in clustering algo
t_j = cluster in ground truth

$$\text{Purity} = \frac{\text{max contigency in each row}}{\text{Total samples}}$$

	A,B	C,D,E
A,B,C	2	1
D,E	0	2

$\text{Purity} = \frac{2+3}{5} = 0.8$

– Types of clustering Algo :

① Partition → K-means, CLARANS (objective fn)

② Hier → (divisive, agglomerative)

③ Density (threshold val.) → DBSCAN, OPTICS

④ Grid

⑤ Model