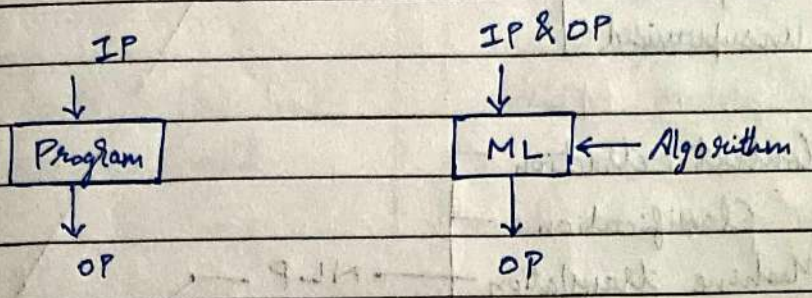
5th August, 2024

# Machine Learning

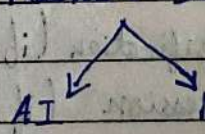


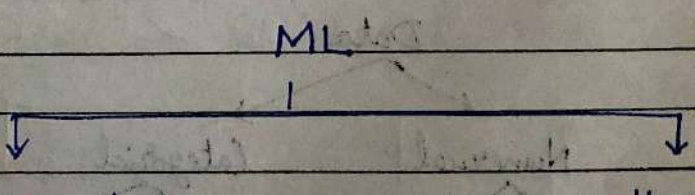

* ML is a subset of AI that focuses on the development of algos & statistical models that enables computers to make predictions based on data.

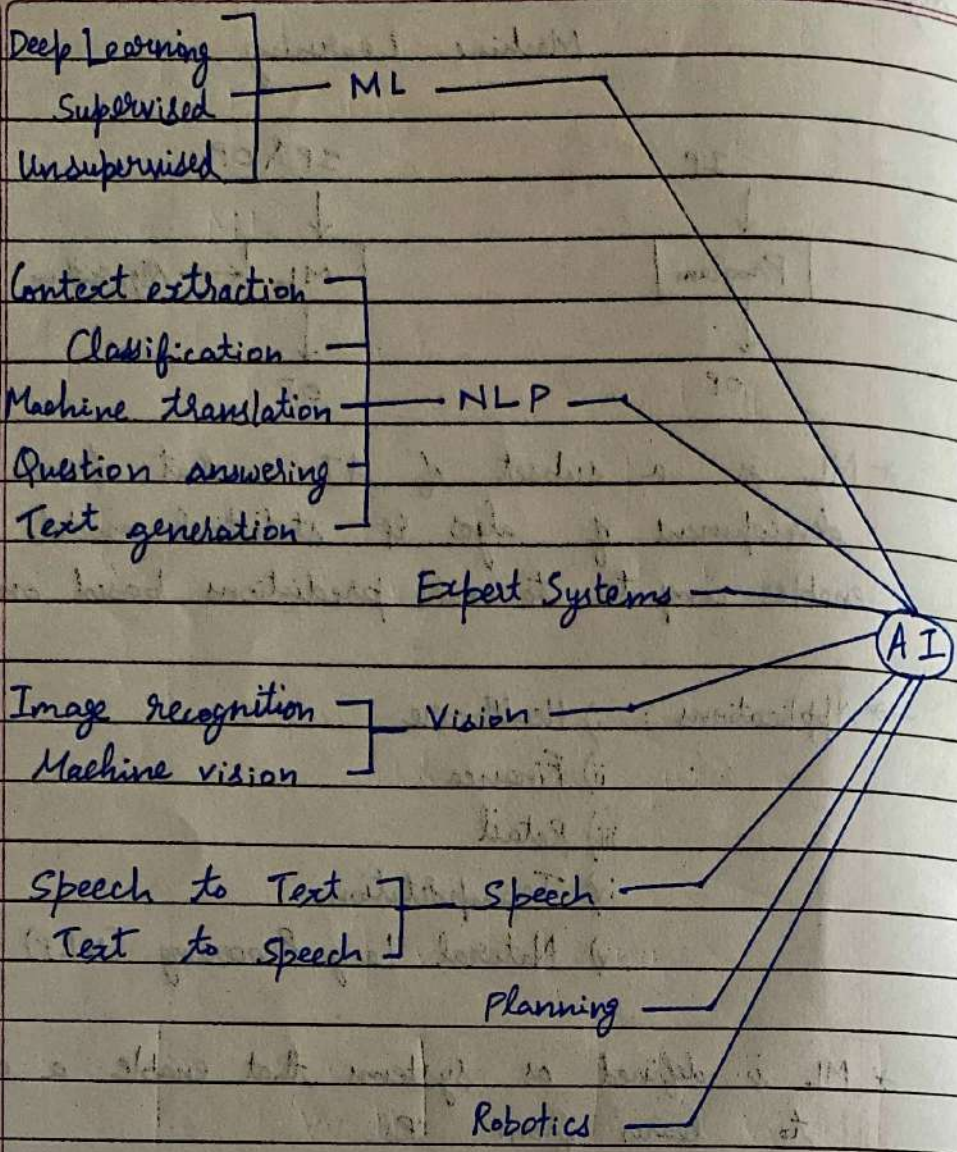
→ Applications : i) Healthcare
ii) Finance
iii) Retail
iv) Transportation
v) Natural Lang. Processing (NLP)

* ML is defined as systems that enable a computer to learn from IPs.

* AI is composed of systems that allow computers to imitate human cognitive processes.

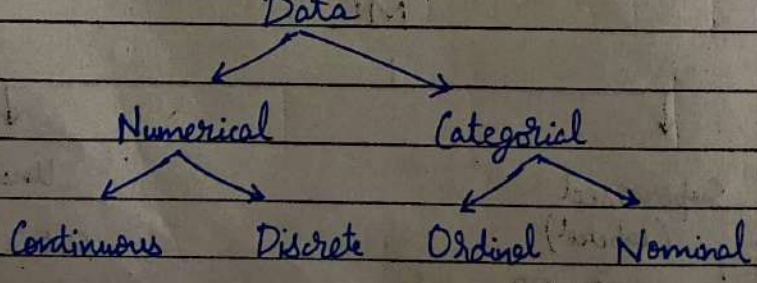





Unsupervised : Clustering
Associativity

Supervised : Classification (if OP is class wise / category)
Regression (if OP data is continuous)



| Training | | Prediction |
|---|---|---|
| * using data to create & fine tune a predictive model | * | answer questions predictive model is used to serve up predictions based on unseen data |

→ 7 steps to ML

i) Gathering data
ii) Data preparation
iii) Choosing a model
iv) Training
v) Evaluation
vi) Parameter tuning
vii) Prediction

→ PTE ML Model Definition

ML is the study of algos that:
- improve their PERFORMANCE (P)
- at some TASK (T)
- with EXPERIENCE (E)

6th August, 2024

* if we have labeled data along with labels
=> Supervised learning

small amount: labeled data
large amount: unlabeled data

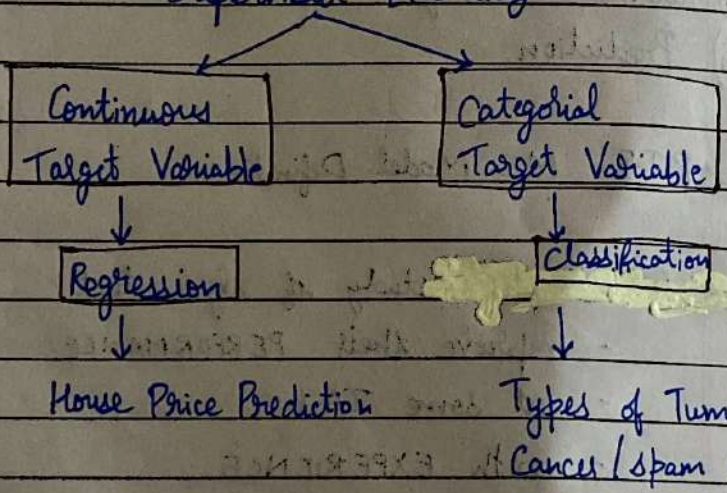
Semi supervised - only a portion of data is labeled
Reinforcement learning

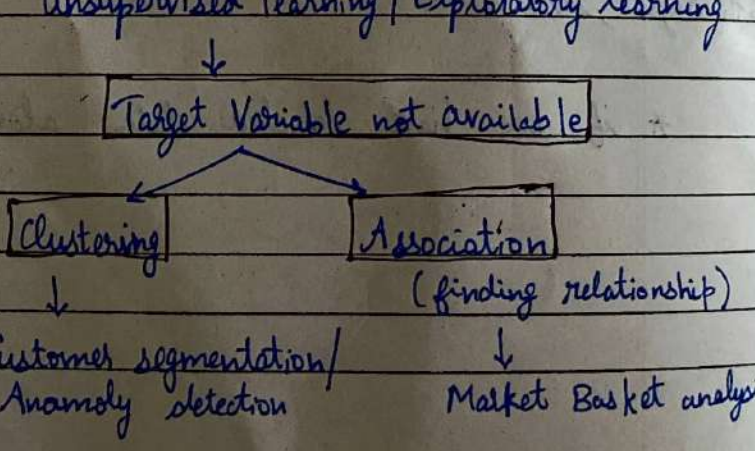
→ Types of ML

i) Supervised (inductive) learning -
training data + desired outputs (labels)

ii) unsupervised learning -
training data (without desired OP)

iii) Reinforced learning -
rewards from sequence of actions



* initially to choose classification/categorical we check OP label

→ Supervised Learning

$$Y = f(X)$$

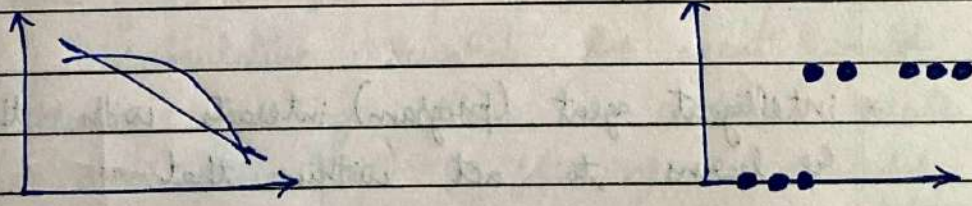
OP variable, IP variable

* algorithm to learn the mapping func. from IP to OP.

* func. approximation is also called hypothesis func. (an "/claim)

* goal: to approximate the mapping func. so well that when you have new IP data (X) you can predict the OP var. (Y) for that data

— Regression — Categorical - Classification



→ Unsupervised Learning

given - training data (w/o desired OP)

* unlabeled IP data is fed to ML model to train it.

* Interpret raw data → hidden patterns → clustering

Association:

An association rule is an unsupervised learning method which is used for finding relationship b/w variables in a large data set.

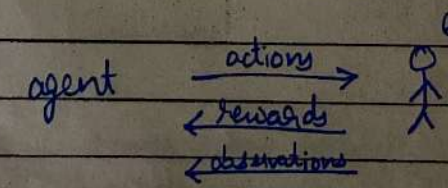
→ Reinforced Learning

* RL involves lot of computing
* No training data set is there for training any model.

RL: what to do and how to map situations to actions.

* intelligent agent (program) interacts with the env & learns to act within that.

* learner is not told which action to take but instead it must discover which action yields max reward



Basic Terms

1. Agent : Learner / decision maker that interacts with the env.

2. Environment : External system with which the agent interacts.

3. State : A representation of current situation of the env.

4. Action : The choices available to the agent at each state.

5. Policy : A strategy / rule that the agent follows to select actions.

6. Reward : A scalar feedback signal from env indicating how good / bad the action taken by user was.

7. Value func : It predicts the expected cumulative reward the agent would receive starting from a certain state (or state-action pair) if following a particular policy.

* The goal of the agent is to learn the optimal policy that maximizes cumulative rewards.

- Q-table in RL

* look up table - max expected future rewards for an action
  - tells us which action is best at each state

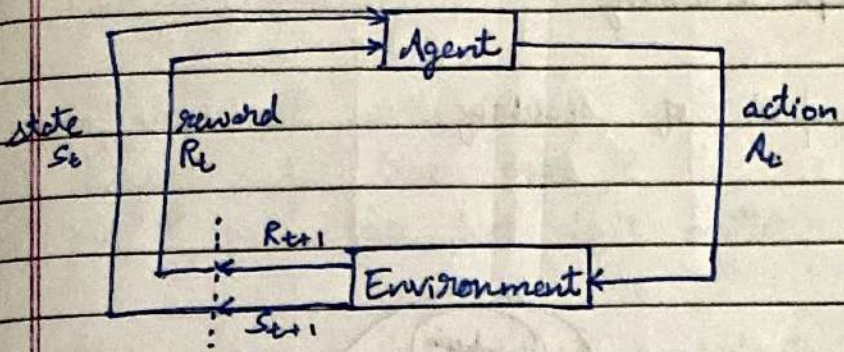
* Q func : takes 2 ips - state & action
  & uses Bellman's eqn:

$$Q'(s_t, a_t) = E[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | (s_t, a_t)]$$

→ Q learning algo

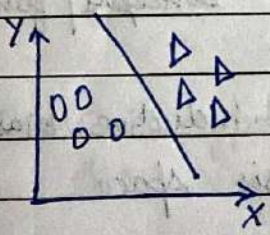
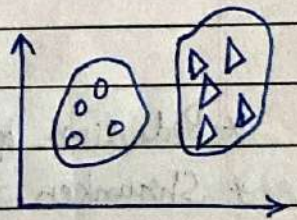
initialize the Q-table
↓
choose an action ←
↓
Perform an action
↓
compute the reward
↓
update the reward

many iterations to get good Q-table



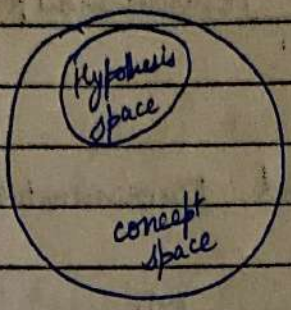

# Generative and Discriminative models

ML

| Discriminative | Generative |
|---|---|
| L makes prediction based on conditional probability and is either used for classification / regression | L revolves around the distribution of a dataset to return a probability for given eg |
| * Logistic regression = binary classification | |
|   |  |
| * draws boundaries in data space | * models how data is placed throughout the space |
| * simple data more data | * complex data less data |
| Types : 1. Logical Regression 2. Support Vector Machine 3. Decision Tree 4. Random Forest | Types: 1. Bayesian Network 2. Hidden Markov Model 3. Autoregressive Model 4. Generative adversarial NW |

# Concept Learning

The prob. of searching




* Domain specific
* concept: category / class (boolean value func.)
* concept can be multi class
* The concept / func. to be learned = target concept

eg: for 3 attribute obj

$2^{2^d}$

for binary feature concept space has $2^{2^d}$ no. of concepts / functions

* Reduction of concept space : inductive bias
* Shrunken space : hypothesis space

→ Inductive Bias / Learning Bias

* set of assumptions that learner uses to predict op

* concepts are of a particular kind called conjuctive concepts because of which the hypothesis space is restricted. Hence also called as Restriction Bias

# Find S algorithm

i) initialize all with $\phi$
ii) choose only true ones
iii) put ? with ones that don't matter

* Version Space : Set of all consistent hypothesis

8th August, 2024

| | (A,B) | (M,N,O) | (C,D) | (0,1) |
|---|---|---|---|---|
| eg: | X | Y | Z | O/P |
| | A | M | C | 0 |
| | B | N | D | 1 |
| | A | O | C | 0 |
| | (2) | (3) | (2) | |

CS = 2 X 3 X 2 = 12 $\Rightarrow 2^{12}$ $(2^{2^d})$

HS = (3 X 4 X 3) + 1 = 37

A
B
A
? → either or
$\phi$ → none of them

* consistent hypothesis is set of assumption that classifies the concept space appropriate-ly

* Accuracy of a model

⇓

# CONFUSION METRICS

| Model Prediction class | Actual class: Disease (10) | Actual class: No Disease (20) | |
|---|---|---|---|
| + | 8 (TP) | 4 (FP) | Type I error |
| − | 2 (FN) | 16 (TN) | |
| | Type II error | | |

$$\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN} = \frac{24}{30}$$
(correct Predicts / Total)

Precision

| | |
|---|---|
| 8 | 4 |
| 2 | 16 |

Recall Sensitivity

$$\text{Recall (Total +ve classes)} = \frac{TP}{TP + FN}$$ sensitivity

(True Positive Rate) $= \frac{TP}{\text{Total Positive}}$

True by actual

$$\text{Precision (model predicted +ve)} = \frac{TP}{TP + FP}$$

True by model

Balanced data – Equally +ve & -ve in data OP

↓

Accuracy ✓

Imbalanced data → Recall, Precession ✓

$$\text{specificity (about negative class)} = \frac{TN}{FP + TN}$$

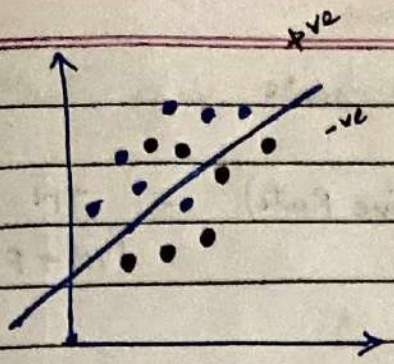
$$\text{False Negative Rate (FNR)} = \frac{FN}{TP + FN}$$

↓

Type-1 error

↓ reduce this

slides q:




blue → +ve
black → -ve

| prediction \ actual | + | − |
|---|---|---|
| + | 6 TP | 2 FP |
| − | 1 FN | 5 TN |

Prediction

| actual data | + | − |
|---|---|---|
| + | 6 TP | 1 FN. |
| − | 2 FP | 5 TN |

→ Precision Recall trade off

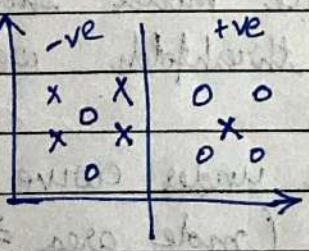
↓

scenario when Recall ↓ Precision ↑

↓↓

use Precision Recall Tradeoff, F1 score

eg:



→ Specificity (Fallout) & F1 score

Specificity (True Negative Rate) $= \frac{TN}{(TN + FP)} = 1 - FPR$

$$FPR = \frac{FP}{TN + FP}$$

F1 score (HM) $= \frac{2 * (recall * precesion)}{(recall + precesion)}$

Harmonic mean

$$= \frac{2TP}{(2TP + FP + FN)}$$

* Higher TNR & a lower FPR is desired since we want to correctly classify -ve class.

→ ROC - Received Operating Characteristics

X → FPR (specificity)
Y → TPR (sensitivity)

plotted with multiple TPR, FPR values

used when threshold comes into picture & set by user manually, since model should work correctly for multiple thresholds we need ROC graph.

* Maximum area under curve (AUC)
(more area ⇒ better model)

eg: converting probabilities to Yes/No
set threshold

False Positive ⇒ Actual : Actually X
Model : But model ✓

False Negative ⇒ Actual : Real ✓
Model : AI X

# Multi-class confusion matrix

9.

| | | Actual | | | |
|---|---|---|---|---|---|
| | | A | B | C | D |
| Predicted | A | 100 | 0 | 0 | 0 |
| | B | 80 | 9 | 1 | 1 |
| | C | 10 | 0 | 8 | 0 |
| | D | 10 | 1 | 1 | 9 |

Class A : TP = 100
TN = 9+1+1, +0+8+0+1+1+9
FP = 0+0+0
FN = 80+10+10

class B : TP = 9
TN = 100+10+8+10+1+1+9
FP = 80+1+1
FN = 0+0+1

class C : TP = 98
TN = 100+0+0+80+9+1+10+10+1+1+9
FP = 10+8
FN = 0+1+1

class D : TP = 9
TN = 100+80+9+1+10+8
FP = 10+1+1
FN = 0+1+0

11th August, 2024

# Decision Tree - ID3 algorithm

Graphical representation of all possible solns. to a decision

eg:

| IP | | OP |
|---|---|---|
| Gender | Age | |
| F | old | snap |
| M | middle | insta |
| F | young | PUBG |
| M | young | PUBG |
| F | young | PUBG |
| F | middle | snap |

* Classification model ⇒ Decision Tree (works best)

Supervised learning

* Decision Tree creates a tree structure with decision node as root node,

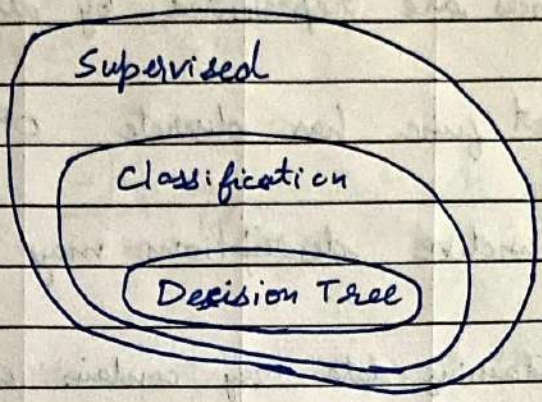
* pruning : makes decision tree less complex

→ Entropy : tells impurity factor (checks purity of node) alone is not enough

→ Information Gain :

* with numbers DT take more time, CART algo.
(regression)

Gini impurity
Gini index

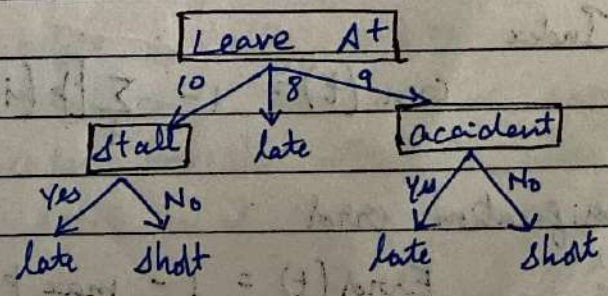



Base Classifiers
(1 model)

Ensemble classifier
(more than 1 model)
- Random Forest
- Boosting
- Bagging

* DT works nicely with text data (NLP)
(decision)

root, intermediate : IP
Leaf nodes : OP
categories of IP / classes / attributes : edges / branches



→ CART : Classification And Regression Tree

→ Appropriate Problem for DT Learning

i) instances are represented by attribute-value pairs

ii) target func has discrete OP values

iii) disjunctive descriptions may be required

iv) the training data may contain errors

v) the training data may contain missing attribute values

* DT : Disjunction of conjunctions (Inductive Bias)
* tends to look for short fat trees
* makes use of heuristics for split

→ Building a DT

Two step — Construction
└ Pruning

→ Impurity Measures

1. Entropy

$$Entropy(t) = -\sum_i p(i|t) \log_2 p(i|t)$$

2. Gini Index

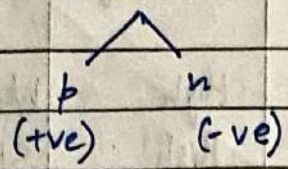
$$Gini(t) = 1 - \sum_i [p(i|t)]^2$$

3. Misclassification error

$$Error(t) = 1 - \max p(i|t)$$

# Entropy

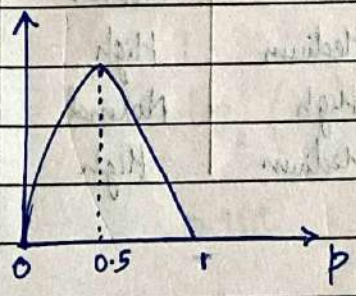
given collection S




$$\text{Entropy}(S) = \frac{-p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

All -ve classes / All +ve classes $\Rightarrow$ Entropy = 0

equal +ve, -ve $\Rightarrow$ Entropy = maximum (0.5)




# Information Gain

↓

used to test goodness of a split

$$\Delta = I(\text{parent node}) - \sum_{i=1}^{k} \frac{N(v_i)}{N} I(v_i)$$

* compare the degree of impurity of parent node before & after splitting

N: total no. of instances at parent node

k: total possible attribute values

$N(v_i)$: total no. of instances associated with child node $v_i$

Slides

ID3 algo problem

| Outlook | Temp | Humidity | Windy | Play Tennis |
|---|---|---|---|---|
| Sunny | High | High | Weak | No |
| Sunny | High | High | Strong | No |
| Overcast | High | High | Weak | Yes |
| Rainy | Medium | High | Weak | Yes |
| Rainy | Cool | Normal | Weak | Yes |
| Rainy | Cool | Normal | Strong | No |
| Overcast | Cool | Normal | Strong | Yes |
| Sunny | Medium | High | Weak | No |
| Sunny | Cool | Normal | Weak | Yes |
| Rainy | Medium | Normal | Weak | Yes |
| Sunny | Medium | Normal | Strong | Yes |
| Overcast | Medium | High | Strong | Yes |
| Overcast | High | Normal | Weak | Yes |
| Rainy | Medium | High | Strong | No |

Step 1. $$\sum(S) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right)$$

$$= 0.94$$

Entropy of whole data set

Step 2. $$\sum_{\text{outlook (sunny)}} = -\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right)$$

5

$$= 0.971$$

$$\sum_{\text{outlook (overcast)}} = -\frac{4}{4}\log_2\left(\frac{4}{4}\right) - \frac{0}{4}\log_2\left(\frac{0}{4}\right)$$

4

$$= 0$$

$$\sum_{\text{outlook (Rainy)}_5} = -\frac{3}{5}\log_2\left(\frac{3}{5}\right) - \frac{2}{5}\log_2\left(\frac{2}{5}\right)$$

$$= 0.971$$

$$\text{avg Entropy} = \frac{5}{14}(0.971) + \frac{4}{14}(0) + \frac{5}{14}(0.971)$$

$$= 0.693$$

Information Gain = Parent E - Avg

$$= 0.94 - 0.693$$

$$= 0.24$$

$$\sum_{\text{temp (High)}_4} = -\frac{2}{4}\log_2\left(\frac{2}{4}\right) - \frac{2}{4}\log_2\left(\frac{2}{4}\right)$$

$$= 1$$

$$\sum_{\text{temp (Medium)}_6} = -\frac{2}{6}\log_2\left(\frac{2}{6}\right) - \frac{4}{6}\log_2\left(\frac{4}{6}\right)$$

$$= 0.918$$

$$\sum_{\text{temp (cool)}_4} = -\frac{1}{4}\log_2\left(\frac{1}{4}\right) - \frac{3}{4}\log_2\left(\frac{3}{4}\right)$$

$$= 0.811$$

$$\text{avg Entropy} = \frac{4}{14}(1) + \frac{6}{14}(0.918) + \frac{4}{14}(0.811)$$

$$= 0.910$$

Information Gain = $0.94 - 0.91 = 0.03$

$$\sum_{\text{Hum}}(H) = -\frac{4}{7}\log_2\left(\frac{4}{7}\right) - \frac{3}{7}\log_2\left(\frac{3}{7}\right)$$

$$= 0.985$$

$$\sum_{\text{Hum}}(N) = -\frac{1}{7}\log_2\left(\frac{1}{7}\right) - \frac{6}{7}\log_2\left(\frac{6}{7}\right) = 0.591$$

$$\text{avg Entropy} = \frac{7}{14}(0.985) + \frac{7}{14}(0.591)$$

$$= 0.788$$

$$\text{Information Gain} = 0.94 - 0.788 = 0.152$$

$$\sum_{\text{windy}}(W) = -\frac{2}{8}\log_2\left(\frac{2}{8}\right) - \frac{6}{8}\log_2\left(\frac{6}{8}\right) = 0.811$$

$$\sum_{\text{windy}}(S) = -\frac{3}{6}\log_2\left(\frac{3}{6}\right) - \frac{3}{6}\log_2\left(\frac{3}{6}\right) = 1$$

$$\text{avg Entropy} = \frac{8}{14}(0.811) + \frac{6}{14}(1) = 0.892$$

$$\text{Info Gain} = 0.94 - 0.892 = 0.048$$

12th August, 2024

q.

| Salary | Location | Job acceptance |
|---|---|---|
| T1 | MUM | Yes |
| T2 | BLR | Yes |
| T1 | BLR | No |
| T1 | HYD | No |
| T2 | MUM | Yes |
| T1 | HYD | No |
| T1 | HYD | No |

① $E(S) = -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right)$

$= 0.985$

Salary: ② $\sum_{\substack{\text{salary} \\ 5}}(T1) = -\frac{1}{5}\log_2\left(\frac{1}{5}\right) - \frac{4}{5}\log_2\left(\frac{4}{5}\right)$

$= 0.722$

$\sum_{\substack{\text{salary} \\ 2}}(T2) = 0$

avg entropy $= \frac{5}{7}(0.722) + \frac{2}{7}(0) = 0.515$

$IG_{\text{salary}} = 0.985 - 0.515 = 0.47$

Location: $\sum_{\substack{\text{location} \\ 2}}(MUM) = 0$

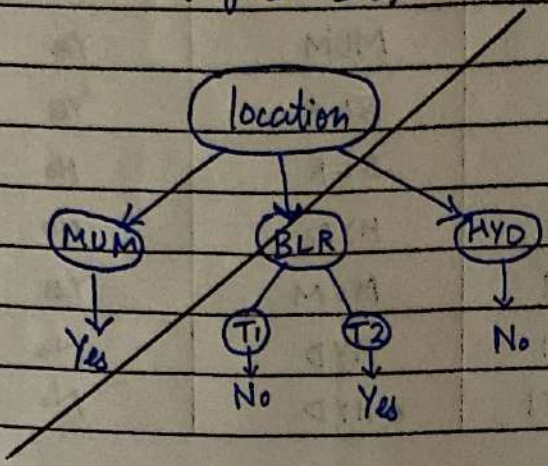
$\sum_{\substack{\text{location} \\ 2}}(BLR) = 1.0$

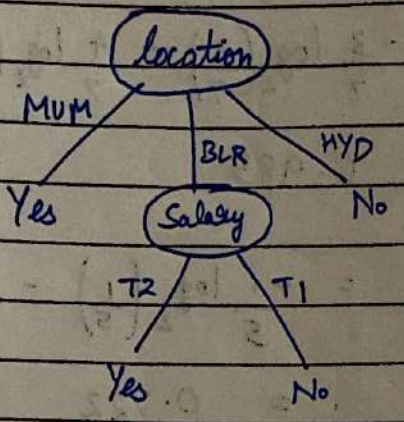
$\sum_{\substack{\text{location} \\ 3}}(HYD) = 0$

avg entropy $= 0 + \frac{2}{7}(1) + 0 = 0.286$

$IG_{\text{location}} = 0.985 - 0.286 = 0.699$

③ Consider location as root (highest IG)




q: numerical IP attribute

| Temp | Playing |
|---|---|
| 40 | No |
| 48 | No |
| 60 | Yes |
| 72 | Yes |
| 80 | Yes |
| 90 | No |

option 1: take decision for all values 40, 48...

Step 1. Sort the numerical Attribute

Step 2. Whereever there is a change in OP consider threshold

eg:

| | |
|---|---|
| 22 | No |
| 23 | Yes |
| 26 | No |
| 28 | No |
| 29 | Yes |
| 30 | Yes |
| 32 | No |

| Temp | Playing |
|---|---|
| 40 | No |
| 48 | No |
| 60 | Yes |
| 72 | Yes |
| 80 | Yes |
| 90 | No |

54

85

$E(S) = 1$

Step 3. $\leq 54$ , $>54$ $\quad\vdots\quad$ $\leq 85$ $\quad >85$

to find which is a better split find IG

$E(\leq 54) = 0$

$E_4(>54) = -\frac{1}{4}\log_2\left(\frac{1}{4}\right) - \frac{3}{4}\log_2\left(\frac{3}{4}\right) = 0.811$

$\text{avg } E = \frac{2}{6}(0) + \frac{4}{6}(0.811) = 0.54$

$IG = 1 - 0.54 = 0.46$

wrt 54

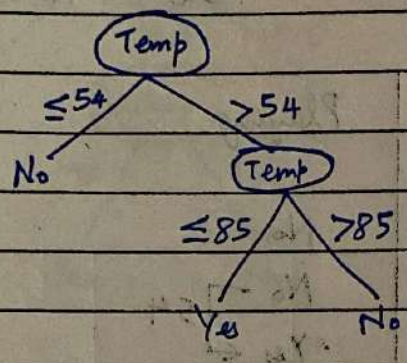
$$E\underset{5}{(\leq 85)} = \frac{-2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right) = 0.971$$

$$E\underset{1}{(>85)} = 0$$

$$\text{avg } E = \frac{5}{6}(0.971) + 0 = 0.809$$

$$IG_{\text{wrt 85}} = 1 - 0.809 = 0.19$$

Take 54 as root node




* overfitting
* underfitting

# CART algo.

uses Gini Index instead of Entropy

$$\text{Gini Impurity} = 1 - \sum_i [p(i|t)]^2$$

Step 1 G.I(s) Gini Impurity
Step 2 Attribute G.I
Step 3 Find Gini Index
Step 4 Gain = Overall G.I - Gini Index

| q. | Income | Student | Credit | Buys |
|---|---|---|---|---|
| 1 | High | No | Fair | No |
| 2 | High | No | Excellent | No |
| 3 | High | Yes | Fair | Yes |
| 4 | High | Yes | Excellent | Yes |
| 5 | Medium | Yes | Fair | Yes |
| 6 | Low | Yes | Fair | Yes |
| 7 | Low | No | Excellent | No |
| 8 | Low | Yes | Excellent | Yes |
| 9 | Medium | No | Fair | No |
| 10 | Low | No | Fair | No |
| 11 | Medium | Yes | Excellent | Yes |
| 12 | Medium | No | Excellent | Yes |

① $GI(S) = 1 - \left[\left(\frac{7}{12}\right)^2 + \left(\frac{5}{12}\right)^2\right] = 0.486$

income ② $GI_{\text{income}}(\text{High})\,4 = 1 - \left[\left(\frac{2}{4}\right)^2 + \left(\frac{2}{4}\right)^2\right] = 0.5$

$GI_{\text{income}}(\text{Medium})\,4 = 1 - \left[\left(\frac{3}{4}\right)^2 + \left(\frac{1}{4}\right)^2\right] = 0.375$

$GI_{\text{income}}(\text{Low})\,4 = 1 - \left[\left(\frac{2}{4}\right)^2 + \left(\frac{2}{4}\right)^2\right] = 0.5$

$\text{avg } GI = \frac{4}{12}(0.5) + \frac{4}{12}(0.375) + \frac{4}{12}(0.5) = 0.485$

$\text{Gain}_{\text{income}} = 0.486 - 0.485 = 0.001$

student $GI_{\text{student}}(\text{No})\,6 = 1 - \left[\left(\frac{5}{6}\right)^2 + \left(\frac{1}{6}\right)^2\right] = 0.277$

$GI_{\text{student}}(\text{Yes})\,6 = 1 - \left[\left(\frac{6}{6}\right)^2 + \left(\frac{0}{6}\right)^2\right] = 0$

$\text{avg } GI = \frac{6}{12}(0.277) = 0.1385$

$\text{Gain}_{\text{student}} = 0.486 - 0.1385 = 0.3475$

credit

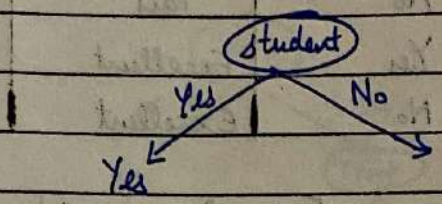
$$GI_{credit\ (Fail)6} = 1 - \left[\left(\frac{3}{6}\right)^2 + \left(\frac{3}{6}\right)^2\right] = 0.5$$

$$GI_{credit\ (excellent)6} = 1 - \left[\left(\frac{2}{6}\right)^2 + \left(\frac{4}{6}\right)^2\right] = 0.444$$

$$\text{avg } GI = \frac{6}{12}(0.5) + \frac{6}{12}(0.444) = 0.472$$

$$\text{Gain}_{credit} = 0.486 - 0.472 = 0.014$$

Take student as root




| income | student | credit | buys |
|---|---|---|---|
| High | No | Fair | No |
| High | No | Exc | No |
| Low | No | Exc | No |
| Medium | No | Fair | No |
| Low | No | Fair | No |
| Medium | No | Exc | Yes |

$$GI_{6}(S) = 1 - \left[\left(\frac{1}{6}\right)^2 + \left(\frac{5}{6}\right)^2\right] = 0.277$$

income

$GI_{income\ (H)2} = 0$

$GI_{income\ (M)} = 1/2$

$GI_{income\ (L)} = 0$

avg GI = $\frac{2}{6}\left(\frac{1}{2}\right)$ = 0.166

Gain = 0.277 − 0.166 = 0.111 (take)

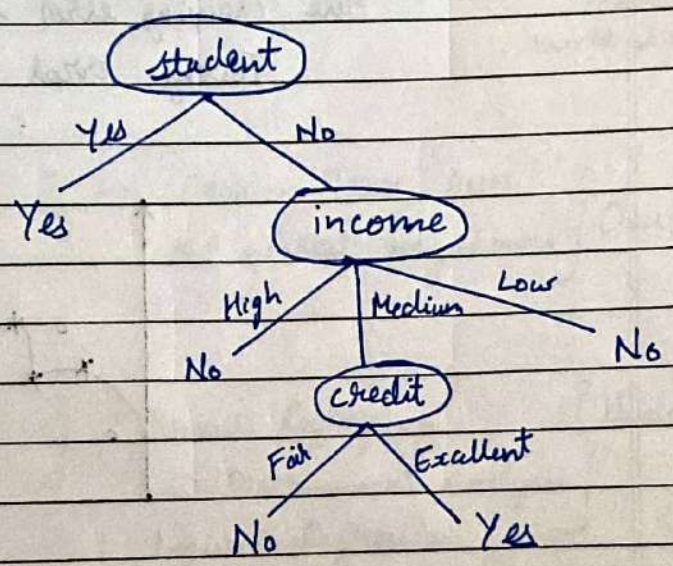
credit

GI credit = 0
(F) 3

GI credit = $1 - \left[\left(\frac{1}{3}\right)^2 + \left(\frac{2}{3}\right)^2\right]$ = 0.444
(E) 3

avg = $\frac{3}{6}$ (0.444) = 0.222

Gain = 0.05




* Hypothesis which understands data completely
- Consistent hyp

* Hypothesis which understands some/most data
- Satisfied hyp

~~Symatic~~

Syntactically diff Hyp = no along with ? ϕ

3 × 2 × 2
? ?
ϕ ϕ

⇒ 5 × 4 × .....

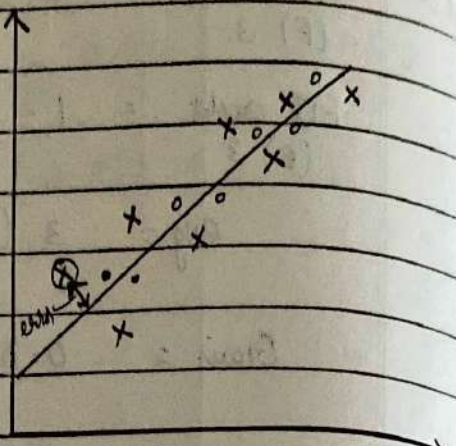
Semantically diff = ϕ ϕ ϕ ⇒ 1 ⇒ (4 × 3 × 3) + 1

22nd August, 2024

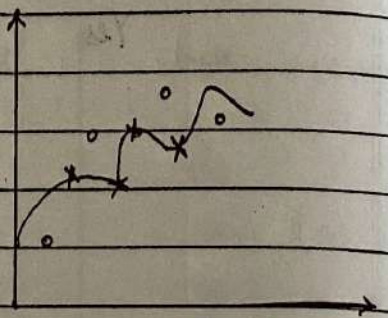
* Decision Tree creates problem of over fitting which can be overcome by pruning.

*: Training data ik
o: Testing data
/: model

Training Error
⇓
Bias




Here Training error is more ⇒ Bias ↑
Testing error is less ⇓
Underfitting



Bias ↓

More Testing error ⇒ Variance ↑
⇓
Overfitting ⇒ model highly complex

* Model we prefer : Low Bias (less training error)
Low Variance (less testing error)

Err (s) = Bias$^2$ + Variance + Irreducible error
noise

Err (s) =

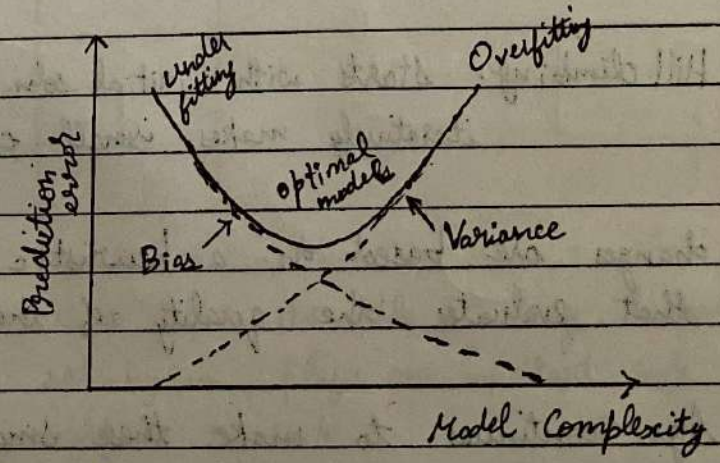
* Bias = diff b/w avg prediction & correct values

* Variance = how are prediction made on same observation different from each other

* Decision Tree → complex → High Variance → Overfitting
it tries to understand training data more & more
↓ make model simple

Low Bias & High Variance → Decision Trees, ANN, k-nearest neighbours, SVM → Overfitting

High Bias & Low Variance → Linear Regression, Linear Discriminant Analysis, Logistic Regression → Underfitting




model ⇒ function

# Hypothesis Search and Inductive bias - ID3

* ID3 can be characterized as searching a hypotheses space for one that fits the training examples.

* The hypothesis space (searched by ID3) is the set of all possible DTs.

* ID3 performs a simple-to-complex, hill climbing search.

↓

way of identifying func.
searching or trying to reach last leaf node
height of tree ↑ complexity ↑

* Hill climbing → simple optimization algo to find best possible solution.

* It belongs to local search algorithm

Goal: Find best soln. from set of all possible soln.

* Hill climbing: starts with initial soln. and then iteratively makes small changes

* changes are based on a heuristic func. that evaluates the quality of the soln.

* Algo continues to make these small changes until it reaches a local maximum.

* The evaluation func. that guides hill climbing search is information gain measure

→ Advantages of Favoring Shorter Trees

i) Simplicity and Interpretability
ii) Generalization
iii) Reduced Overfitting

→ Limitations

i) Underfitting - Did not understand training data
ii) Loss of Information
iii) Bias against complex pattern

26th August, 2024

# KNN : K Nearest Neighbour

| Height | Weight | Target |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

* Training time it sits idle - lazy learner
* k-value set by us, plays an important role. how many neighbours to check

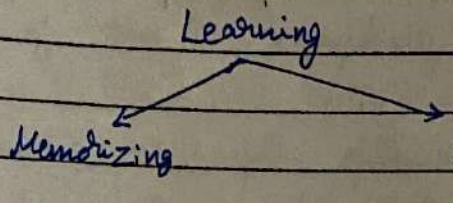
→ Instance based learning

* It is an ML technique in which the model learns the training examples by heart and then generalizes to new instances based on some similarity measure

* number as IP attribute

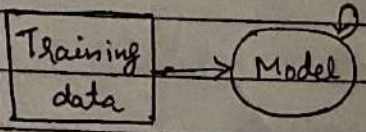
* Memory-based learning / lazy-learning

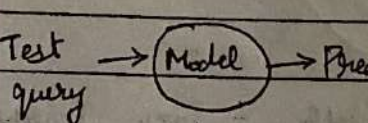
* Worst case O(n) → size of training data



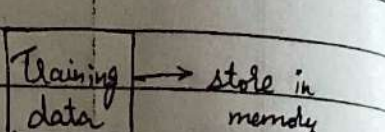

- instance based learning

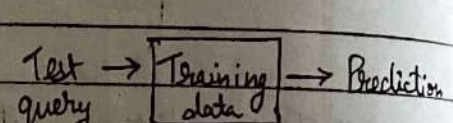
- Eager Learning




- Lazy Learning



→ KNN

* instance based, lazy learner ML algo, supervised learning

KNN Inductive Bias

requires 3 things : i) set of stored records
ii) distance metric : compute dist b/w records
iii) value of k : no. of nearest neighbours to retrieve

To classify an unknown record :

i) compute distance to other training records
ii) identify k nearest neighbours
iii) use class labels of nearest neighbours to determine class of unknown record

* for odd classes take even k-value
for even classes take odd k-value

- K-NN working algo

Step 1: Select k of the neighbours

Step 2: Calculate Euclidean dist. of k no. of neighbours

Step 3: Take k nearest neighbours as per calculated euc dist.

Step 4: Among k neighbours, count no. of data points in each category

Step 5: Assign the new data point to that category for which no. of neighbour is maximum

Step 6: Model is ready.

* All instances correspond to points in n-D space

* nearest neighbour : dist measure - dist $(x_1, x_2)$

* Target func : discrete or real valued

* for discrete (classes) : kNN returns most common value

* for real valued (numbers) : kNN returns mean values of k nearest neighbours

- Choosing the value of k

* if k too small → overfitting

* if k too large → underfitting

- Consider learning discrete-valued target functions of the form $f: R^n \to V$ where V is finite set

* Training Algo : For each training example $(x, f(x))$ add the eg. to the list of training egs

* Classification Algo : Given a query instance $x_q$ to be classified

1. Let $x_1 \ldots x_k$ denote k instances from training egs. that are nearest to $x_q$

2. Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a,b) = 1$ if $a = b$ eq

$\delta(a,b) = 0$ otherwise

* Argmax - operation that finds the arg that gives the max from a target func.

↓

used to find class with largest predicted probability.

→ KNN Applications

* In KNN, every data instance is assumed to be a point in d-dimensional space ($R^d$) where d is the no. of features.

* KNN can be used for both classification & regression

– KNN classification

* target func. is discrete valued (m classes)
* each of the data instances may belong to one of the m classes
* We define set V as set of all classes
* target func $f: R^d \rightarrow V$
* any data instance $x_i$ will have a label $f(x_i)$ (one of the class from set V)

– KNN regression

* target func. is real valued (continuous value)
* target func $f: R^d \rightarrow R$

* query instance is represented as $x_q$
* we are interested in estimating the $f(x_q)$ value, denoted by $\hat{f}(x_q)$

→ Minkowski Distance

$$= \left( \sum_{i=1}^{k} \left( |x_i - y_i| \right)^q \right)^{1/q}$$

$q = 1 \Rightarrow$ Manhattan
$q = 2 \Rightarrow$ Euclidean

k : no. of features/attributes
$x_i$ $y_i$ : $i^{th}$ attributes

→ How to choose no. of neighbours

* k : odd if classes - even
  k : even if classes - odd

BEST PRACTICE :

* start with k = no. of classes + 1
* in case of tie decrease k by 1
* $k = \sqrt{n}$ perform well too

Elbow method : used to determine best k value

- Calc. error rate for diff k-value
- choose optimal k-value as elbow value of curve
- retrain with new k value
- retrain model with best k value

→ KNN algo for real valued target

* store each training example $\langle x, f(x) \rangle$
* find k nearest neighbour $x_1 \cdots x_k$ of q

$$f(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

mean value of k nearest neighbours is returned as appropriate value of $x_q$.

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

→ Weighted KNN

* More closer ⇒ more importance ⇒ more weight

* $w_i = \frac{1}{d(x_q, x_i)^2}$

Distance wtd. Voting

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} w_i \, \delta(v, f(x_i))$$

$$w_i = \frac{1}{d(x_q, x_i)^2}$$

→ Issues with KNN

i) * KNN slow algo : as data set ↑ KNN struggles

ii) * Curse of dimensionality : no. of IP features ↑ KNN struggles

→ Overcoming curse of dimensionality

* Assigning weights

* Leave-one-out

→ Issues with KNN

iii) Optimal no. of neighbours
iv) Imbalanced data causes problems
v) Outliers sensitivity
vi) Missing value treatment
vii) K-NN needs homogeneous features

→ KNN computational complexity

* Basic KNN algo stores all examples
* n examples of dim d

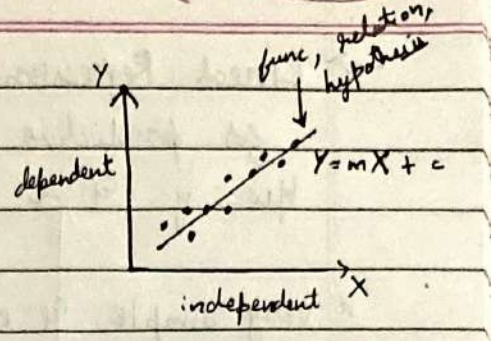
compute dist to one eg: $O(d)$
find 1 nearest neighbour: $O(nd)$
find k closest egs: $O(knd)$

↓

prohibitly expensive for large no. of samples

27th August, 2024

# Linear Regression




Simple Linear Regression

⇓

1 IP , 1 OP

it assumes linear relation

Optimization - improving the model → decreasing error

↳ Gradient Descent algo

Regression : method / statistical measure for understanding relationship b/w independent variable (X) & dependent variable (Y)
(features) (outcome)

Types of Regression
- Linear
- Polynomial
- Support vector
- Decision Tree
- Random Forest
- Ridge
- Lasso
- Logistic

→ Scatter Plot

↓

* Visualization tool to choose 'type of regression'
* shows individual points as dots in 2D plane

→ Linear Regression : statistical regression method for predictive analysis, shows linear relationship b/w y & x (one/more)

* very simple & easy
* used when continuous numeric data

Types of Linear Regression : i) Simple 1 IP 1 OP
ii) Multiple many IP 1 OP

→ Simple Linear Regression
only 1 IP variable

$$y = mx + b$$

y : dependent, m : slope, x : independent, b : y-intercept

→ Best Fitted Line

* Straight line that minimizes the vertical distance b/w the data points & the regression line

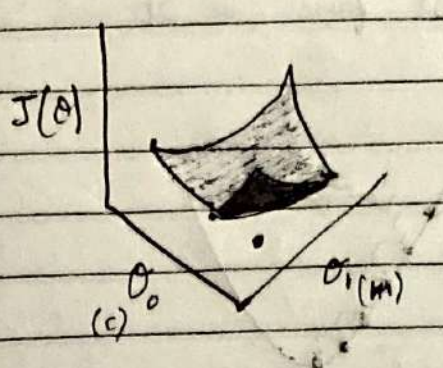
Error = Actual value − Predicted value

$$= y_i - (mx_i + b)$$

Squared Error $= (y_i - (mx_i + b))^2$

Sum of Squared Error (SSE) $= \sum_{i=1}^{n} (y_i - (mx_i + b))^2$

Mean Squared Error (MSE) $= \frac{1}{n} \sum_{i=1}^{n} (y_i - (mx_i + b))^2$



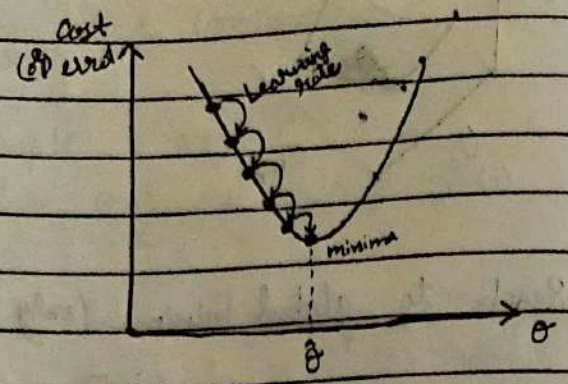

Note: Reach to global minima (only 1 global minima)

rate of improving → learning rate

→ Gradient Descendent

i) initialize
ii) derivative of loss func.
iii) find min of loss func ⇒ best-fit value of func.

→ Visualization of Cost func: in 2D




Learning rate alpha : +ve constant determines how much bigger the steps are gonna be

Note: If $\alpha$ is very large then it is gonna overshoot the min val of cost func (J) & might diverge.
If $\alpha$ is smaller value then smaller steps are taken to reach global min.

→ Steps to find optimum values of m & b

1. Initialize m & b with random values

2. Define the loss func :

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

n: no. of data points
$y_i$: actual y value

3. Calculate the gradient : partial derivatives of loss func with respect to m & b

$$\frac{\partial MSE}{\partial m} = \frac{-2}{n} \sum_{i=1}^{n} x_i (y_i - (mx_i + b))$$

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

4. Update parameters

$$m \leftarrow m - \alpha \frac{\partial MSE}{\partial m}$$

$$b \leftarrow b - \alpha \frac{\partial MSE}{\partial b}$$

5. Iterate : Repeat ③, ④ until parameters converge

q. $y = mx + b$, $m = 10$, $b = 300$, $LR(\alpha) = 0.0001$

| $Y = 10x + 300$ | Age (x) | Salary $y_i$ |
|---|---|---|
| 600 | 30 | 800 |
| 670 | 37 | 950 |
| 550 | 25 | 600 |
| 730 | 43 | 1050 |
| 800 | 50 | 1200 |
| 590 | 29 | 740 |
| 760 | 46 | 1100 |

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

$$= 74485.71$$

$$GD \rightarrow \frac{\partial MSE}{\partial M} = \frac{-2}{n} \sum_{i=1}^{n} x_i (y_i - (mx_i + b))$$

$$= -20388.57$$

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

$$= -497.14$$

$$m_1 = m - \alpha\left(\frac{\partial MSE}{\partial m}\right)$$

$$= 12.038$$

$$b_1 = b - \alpha\left(\frac{\partial MSE}{\partial b}\right)$$

$$= 300.0497$$

| $x$ | $mx + b$ | $m_1x + b_1$ |
|---|---|---|
| 30 | 600 | 661.189 |
| 37 | 670 | 745.455 |
| 25 | 550 | 600.999 |
| 43 | 730 | 817.6837 |
| 50 | 800 | 901.949 |
| 29 | 590 | 649.152 |
| 46 | 760 | 853.798 |

$$MSE = \frac{1}{7}\sum_{i=1}^{7}(y_i - \hat{y})^2$$

$$= 38957.2$$

$$\frac{\partial MSE}{\partial m} = -\frac{2}{n}\sum x_i(y_i - (mx_i + b)) = -1443.2$$

$$\frac{\partial MSE}{\partial b} = -\frac{2}{n}\sum(y_i - (mx_i + b)) = -345.59$$

$$m_2 = m_1 - \alpha\left(\frac{\partial MSE}{\partial m_1}\right) = 13.4831$$

$$b_2 = b_1 - \alpha\left(\frac{\partial MSE}{\partial b_1}\right) = 300.084$$

→ Measuring model Performance

* The goodness of fit determines how the line of regression fits the set of observations. Process of finding best model out of various models: optimization

R-squared method

└ determines goodness of fit

* lies bw 0 and 1
* R-squared = 1 model perfectly fits
* R-squared = 0 model does not predict any variability
* measures strength of relationship b/w dependent & independent variable. (0-100%)

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

$$R^2 = 1 - \frac{\text{sum squared regression (SSR)}}{\text{total sum of squares (SST)}}$$

- R squared method

1. Calculate mean of the target/dependent variable y & denote it with $\bar{y}$

2. $$SS_{Tot} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

3. $$R^2 = 1 - \frac{SSR}{SST}$$

$$= 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y}_i)^2}$$

* High $R^2$ ⇒ less diff b/w predicted & actual values
⇓
good model

* for multiple regression

$$R^2 = \frac{\text{Explained variation}}{\text{Total variation}}$$

↓
coefficient of determination or multiple determination

Linear Regression

| Advantages | Disadvantages |
|---|---|
| * Simple to implement & easier to interpret OP efficiencies | Outliers can have a huge effect on the regression |
| * We know the relationship b/w ind & dep variable (linear) algo is the best to use because of its less complexity | LR assumes linear relation ⇓ it assumes straight line relationship, assumes independence b/w attributes |
| * LR is susceptible to over fitting but it can be avoided using some dimensionality reduction tech, regularization | It also looks at the relationship b/w the mean of dep & ind vars, mean is not a complete desc of relationship b/w vars |

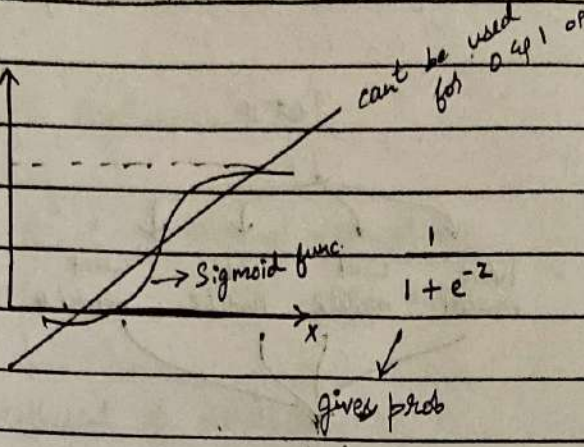
KNN → classification

2nd September, 2024

# Logistic Regression - Not a regression model but a classification model

↓

discrete OP (0, 1)




$$\text{odd} = \frac{\text{Prob. of event occuring}}{\text{Prob. of event not occuring}} = \frac{p}{1-p}$$

z → linear eqn: for 1 parameter $z = mx + c$

for multiple: $z = \beta_0 + \beta_1 x_1 + \beta_2 x_1 + \beta_3 x_3$

Q: Predict likelihood of admission

| Patient ID | Age | BMI | BP | Glucose level | days in hospital | no. of prev admission |
|---|---|---|---|---|---|---|
| 6 | 45 | 27 | 138 | 102 | 4 | 1 |
| 7 | 62 | 25 | 132 | 106 | 5 | 2 |

Q:

| Restaurant | location | food quality | High Rating |
|---|---|---|---|
| 1 | 3 | 2 | 0 |
| 2 | 4 | 3 | 1 |
| 3 | 2 | 1 | 0 |
| 4 | 5 | 2 | 1 |
| 5 | 4 | 2 | 1 |

$w_0 = -3, \quad w_1 = 0.4, \quad w_2 = 2$

predict whether restaurant with location = 2 & food quality = 2 will receive high rating or not

$$z = -3 + (0.4)(2) + 2(2) = -1.8$$

$$y = \frac{1}{1 + e^{-1.8}} = 0.858$$

if threshold = 0.5 $\Rightarrow$ $y$ is high rating

Logistic Regression: supervised ML algo mainly used for binary classification tasks where goal is to predict the prob that an instance belongs to one of the 2 given classes

* ~~Of~~ : independent vars can be categorical / numerical
* Logistic Regression : * Binary classification problems
  * handling imbalanced data
  * capturing non linear relationships
* Linear Regression : * continuous numeric data
  * not well suited for classification tasks

* Logistic Regression more robust to outliers
  └ due to sigmoid func.

## 3 types of Logistic Regression

i) Binomial : 2 possible types of dependent variable
ii) Multinomial : 3 or more possible unordered types of dependent variable
iii) ordinal : 3 or more possible ordered types of dependent variable

$p$ = outcome of interest / all possible outcomes

$$\text{Odds} = \frac{p \text{ of event occurring}}{p \text{ of event not occurring}} = \frac{p}{1-p}$$

odds ratio : ratio of 2 odds

* dependent variable in logistic regression follows Bernoulli distribution

Logit func : maps the linear combination of IP variables ↔ OP variable

natural log of odds

goal of logistic regression : estimate $p$

$$\text{Logit}(p) = \ln(\text{odds}) = \ln\left(\frac{p}{1-p}\right) = \ln(p) - \ln(1-p)$$

/ pred value b/w 0 & 1

- Relationship bw Logit & Sigmoid func.

$\text{Logit}^{-1}(z) = \frac{1}{1+e^{-z}}$, $z$: linear combination of IP variables

$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_n x_n$

$(x_1, x_2 \cdots x_n$ : IP features)

$\sigma(z) = \frac{1}{1+e^{-z}}$

sigmoid func

$\sigma(z)$ tends to 1 as $z \to \infty$ ; $P(y=1) = \sigma(z)$

$\sigma(z)$ tends to 0 as $z \to -\infty$ ; $P(y=0) = 1-\sigma(z)$

- Loss Function in Logistic Regression

Binary cross-entropy loss in logistic regression:

$$L(y, \hat{y}) = -[y \log(\hat{y}) + (1-y) \log(1-\hat{y})]$$

$$\frac{\partial L}{\partial m} = -x(y-\hat{y}) \qquad \frac{\partial L}{\partial b} = -(y-\hat{y})$$

- Finding coefficients using Gradient Descent

1. Initialize coefficients
2. Calculate Predicted Probabilities

$$P(Y=1|X) = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$$

3. Calculate log-likelihood

$$L(y, \hat{y}) = -[y \log(\hat{y}) + (1-y) \log(1-\hat{y})]$$

4. Calculate Gradient $\frac{\partial L}{\partial m} = -x(y - \hat{y})$

$$\frac{\partial L}{\partial b} = -(y - \hat{y})$$

5. Update coefficients

$$m = m - \alpha\left(\frac{\partial L}{\partial m}\right)$$

$$b = b - \alpha\left(\frac{\partial L}{\partial b}\right)$$

Repeat ② to ⑤ until coefficients converge

6. Convergence & Termination
7. Final coefficient values

- Assumption of Logistic Regression
  - i) Independent observations
  - ii) Binary dependent variables
  - iii) Linearity b/w ind vars & log odds
  - iv) No outliers
  - v) Large sample size

- Advantages : i) performs better when data is linearly seperable
  ii) doesn't req. too many computational resources
  iii) doesn't req. tuning
  iv) easy to implement & train
  v) gives measure of how relevant a predictor is

- Disadvantages : i) fails to predict continuous outcome
  ii) assumes linearity b/w dep & ind vars
  iii) may not be accurate if sample size is too small.