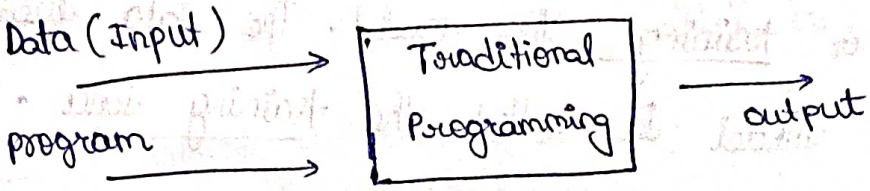
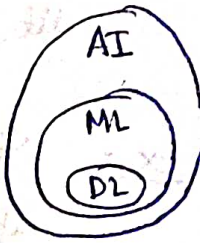
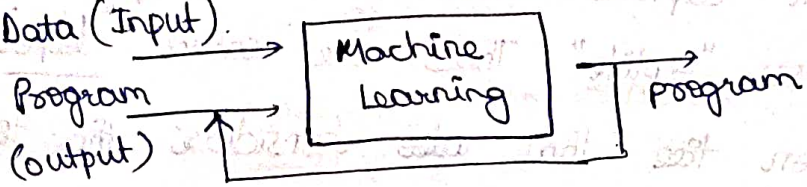
INTRODUCTION TO MACHINE LEARNING

- It is the field of study that gives computers the capability to learn without being explicitly programmed

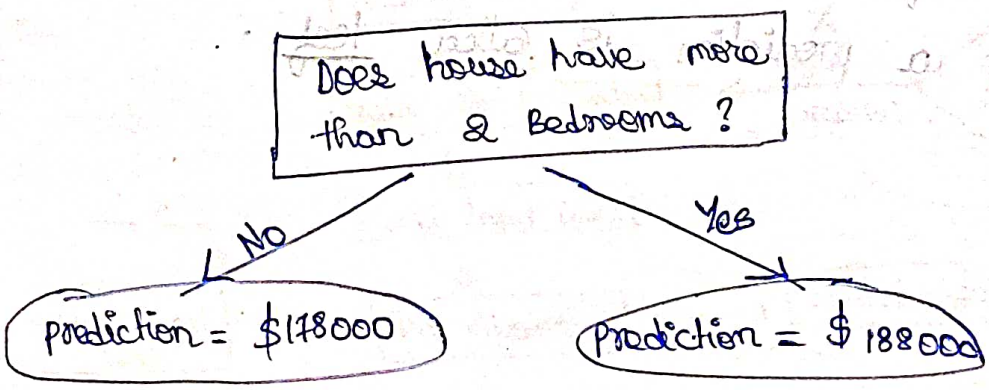
- 



- 

Ex: You ask your cousin how he's predicted real estate values in the past and he says it is just intuition. But more questioning reveals that he's identified price patterns from houses he has seen in the past, and he uses these patterns to make predictions for new houses he is considering.

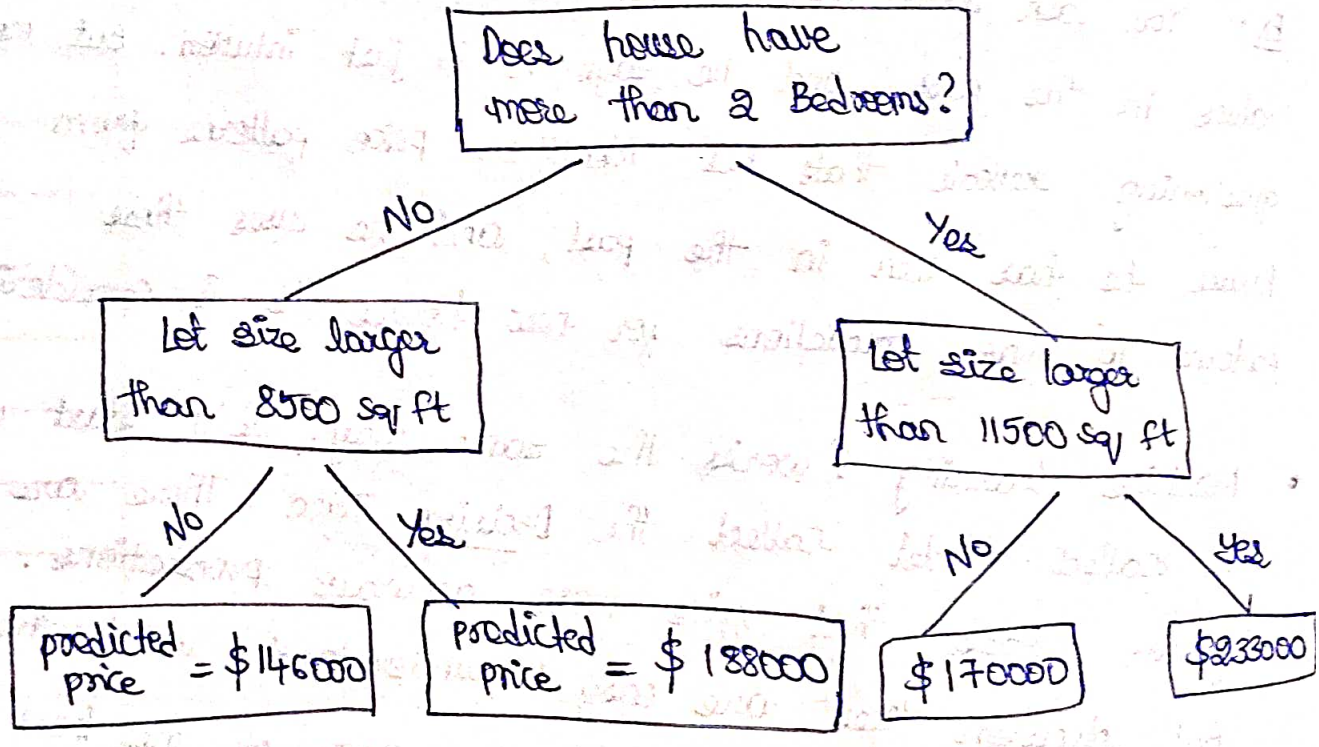
- Machine learning works the same way. We'll start with a called model called the Decision Tree. There are fancier models that give more accurate predictions. But decision trees are easy to understand, and they are the basic building block for some of the best models in data science.



- We use data to decision how to break the houses into two groups, and then again to determine the predicted price in each group.

- This step of capturing patterns from data is called fitting or training the model. The data used to fit the model is called the training data.

Ex: You can capture more factors using a tree that has more "splits". These are called "deeper" trees. A decision tree that also considers the total size of each house's lot might look like this.



- The predicted price for the house is at the bottom of the tree. The point at bottom where we make a prediction is called leaf.

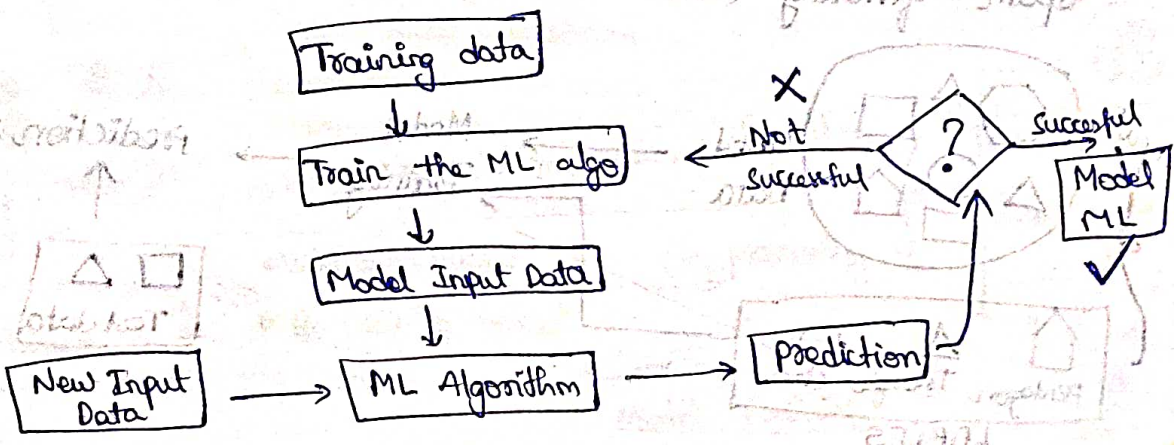
- We use scikit-learn library to create models. It is easily the most popular library for modeling the types of data typically stored in DataFrames.

- The steps to building and using a model are :

1) Define - what type of model will it be?
2) Fit - Capture patterns from provided data. This is heart of modeling.
3) Predict - estimate / guess
4) Evaluate - Determine how accurate the models' predictions are.

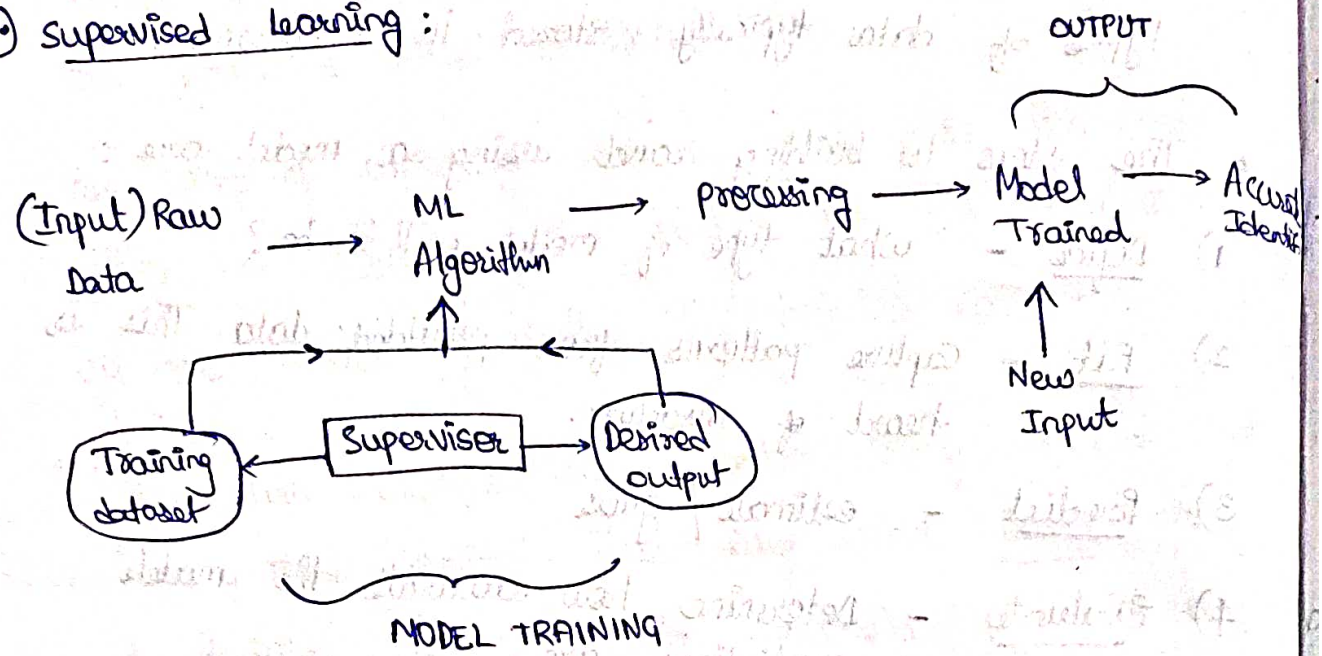
*. INTRODUCTION :

- Machine learning is a subset of Artificial Intelligence.
- It focuses mainly on the designing of systems, thereby allowing them to learn and make predictions based on some experience which is data incase of machines.
- AI - A technique which enables machines to mimic human behaviour.
- ML - subset of AI technique which use statistical methods to enable machines to improve with experience.
- DL - subset of ML which make the computation of multi-layer neural network feasible.

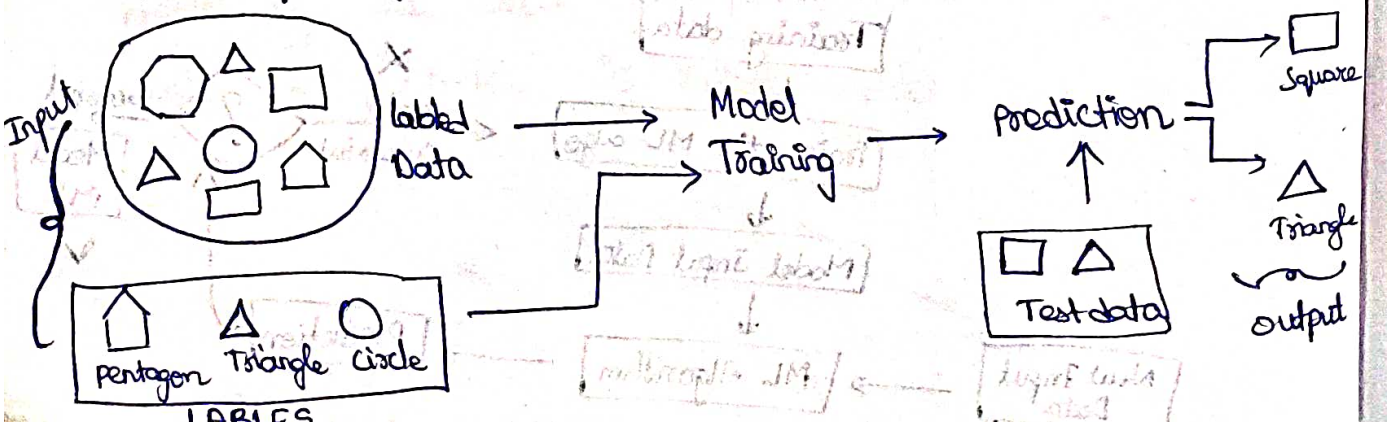


*. Machine Learning Types :

① Supervised Learning :



- It is a type of ML in which machines are trained using well "labelled" training data, and on basis of the data, machines predict the output.
- The "labelled" data means some input data is already tagged with the correct output
- The aim of supervised learning algorithm is to find a mapping function to map the input variable (x) with the output variable (y).
- In real world, Supervised learning can be used for Risk Assessments, Image Classification, Fraud Detection, spam filtering (etc).



- Supervised learning is 2 types:
  1) Regression - relationship between two variables.
  2) Classification - there are two classes (output variable is categorical).

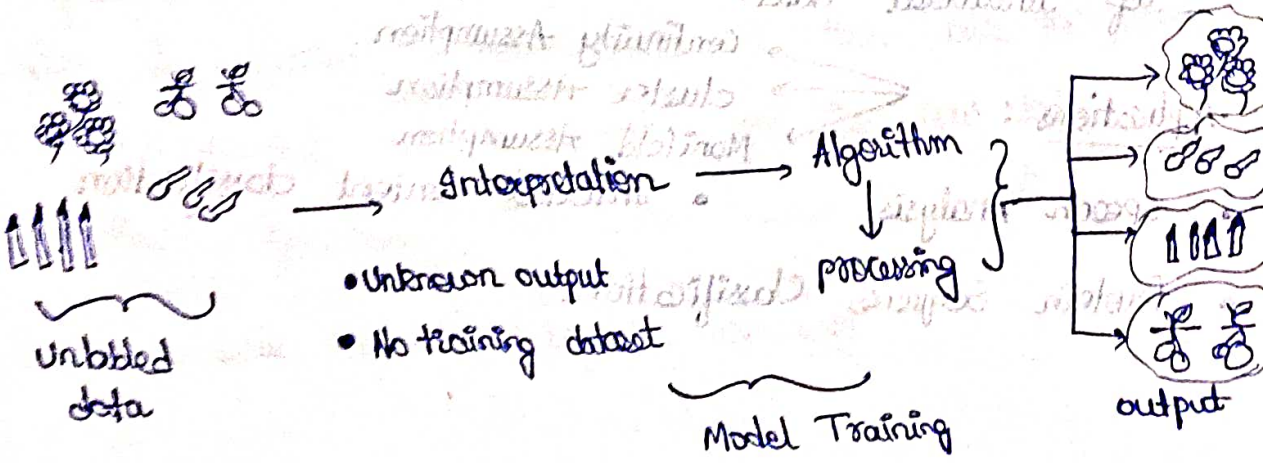
- Advantages:
+ The model can predict the output on basis of prior experience.
+ We have exact idea about classes of objects.

- Disadvantages:
- Not suitable for handling complex tasks.
- Cannot predict correct output if the test data is different from the training dataset.
- Training required lots of computation times.
- We need enough knowledge about classes of object.

② UNSUPERVISED LEARNING:

- It is a type of ML in which models are trained using unlabled datasets and are allowed to act on the data without any supervision.

- The Goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.



- Types of Unsupervised Learning :

① Clustering : similarities & non-similarities are made into different groups.

② Association : determines set of items that occurs together in dataset & finding the relationships between variables.

- Advantages :

+ More complex tasks.
+ Market Basket Analysis.
+ easy to get unlabeled data in comparision to labeled data

- Disadvantages :

- More difficult.
- Result may be less accurate.
- Algorithms don't know exact output in advance.

Note : Semi Supervised Learning :

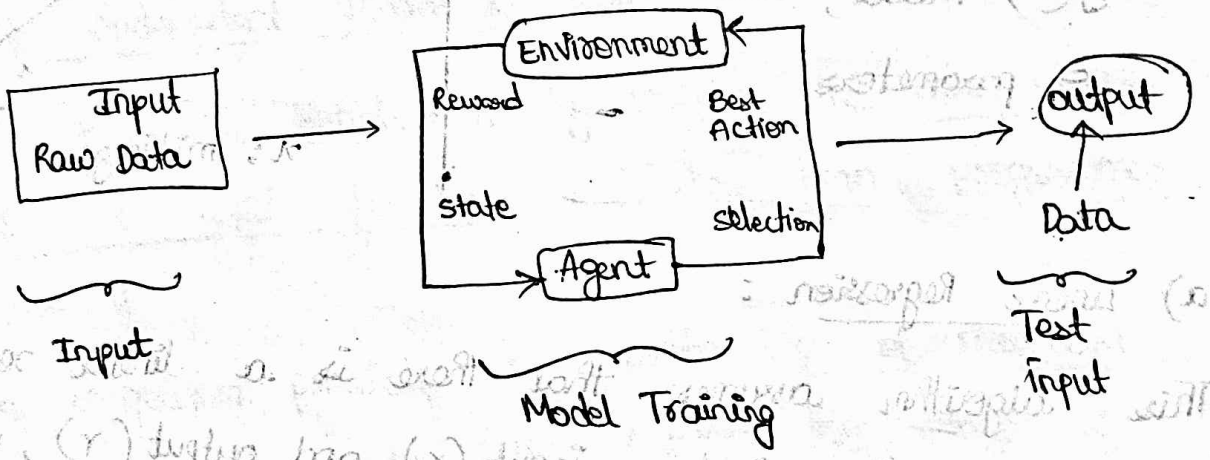
- The algorithm is trained upon a combination of labeled and unlabeled data.
- Typically, this combination will contain a very small amount of labeled data and a very large amount of unlabeled data.

Applications : → Continuity Assumption
→ Cluster Assumption
→ Manifold Assumption

- Speech Analysis
- Internet Content classification
- Protein Sequence Classification

③ Reinforcement Learning :

- It is a feedback based ML in which a machine learns to behave in an environment by performing the actions and seeing the results of action.
- For each good action, the machine gets positive feedback and for each bad action, the machine gets negative feedback
- Here machine learns automatically using feedbacks without any labeled data. It is bound to learn by its experience only.
- Applications : Robotics, Control, Game Playing, Chemistry, Business, Manufacturing, Finance Sector.



*. REGRESSION :

- Supervised learning has been broadly classified into 2 :
  1) Regression 2) Classification
- Regression is a kind of supervised learning that learns from the labeled Datasets and is then able to predict a continuous valued output for the new data given to the algorithm.
- Ex : Linear Regression, Logistic Regression

- Ex: We want to have a system that can predict the price of a used car.
- Inputs are the car attributes. (brand, year, engine capacity, mileage, etc...)
- The output is the price of the car. Such problems where the output is a number are regression problems.

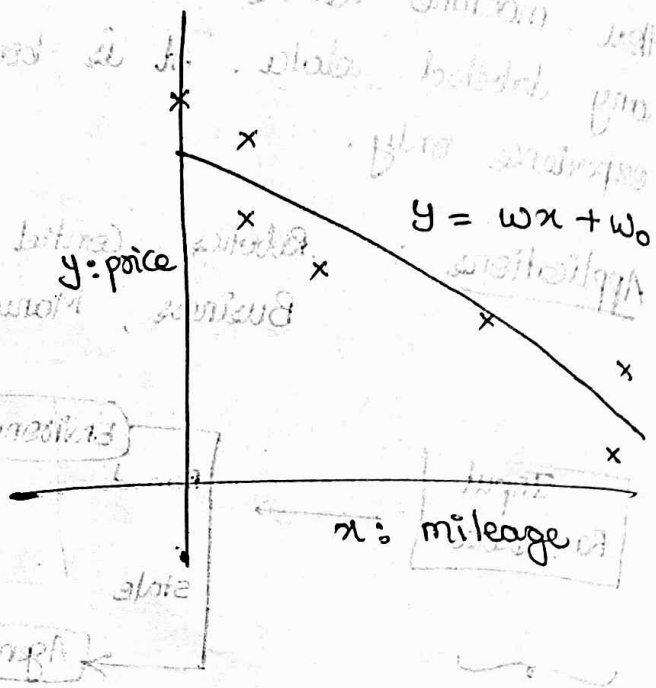
x: car attributes

y: price of the car

$y = g(x|\theta)$

$g(\cdot)$ model,

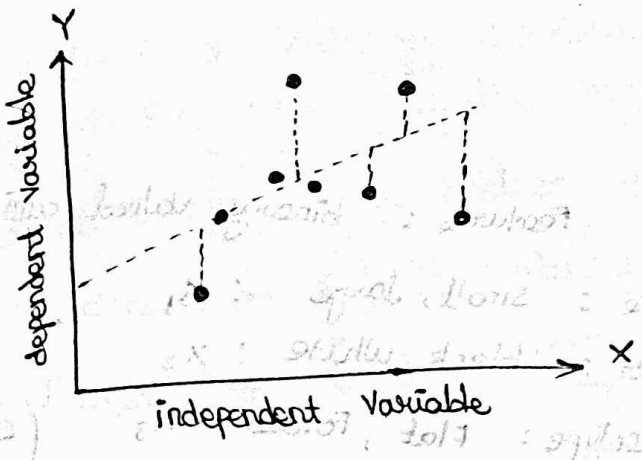
θ parameters



a) Linear Regression:

- This algorithm assumes that there is a linear relationship between the 2 variables, input (x) and output (Y), of the data it has learnt from.
- The Input variable is called the "Independent variable" and the output variable is called Output/Dependent variable"

When unseen data is passed to the algorithm, it uses the function, calculates and maps the input to a continuous value for the output.

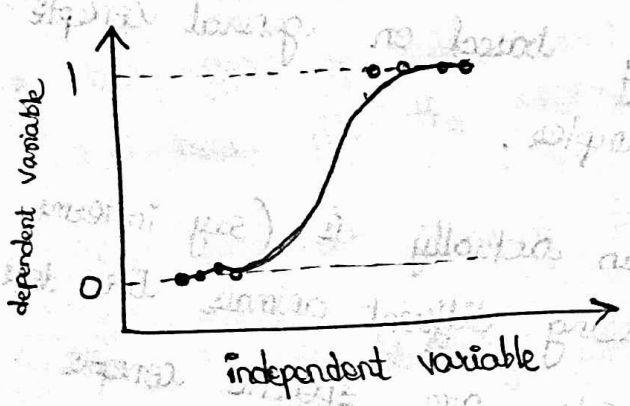


$$h_\theta(x) = \theta_0 + \theta_1 \cdot x$$

Idea : choose θ_0, θ_1 such that $h_\theta(x)$ is close to y for our training (x, y) examples.

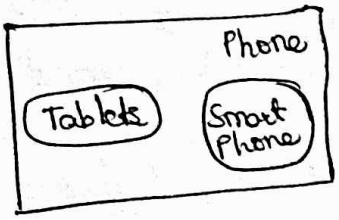
b) Logistic Regression :

- This algorithm predicts discrete values for the set of independent variables that have been passed to it.
- It does the prediction by mapping the unseen data to the logit function that has been programmed into it.
- The algorithm predicts the probability of new data and so its' output lies between the range of 0 and 1.



* CONCEPT LEARNING :

Ex:



Universe

Features : binary valued attributes

Size : small, large : x_1

color : black, white : x_2

screenType : Flat, Folded : x_3

shape : square, rectangle : x_4

} d

Concept, $C = \langle x_1, x_2, x_3, x_4 \rangle$

Tablet = < large, black, Flat, Square >

SmartPhones = < small, grey, Folded, rect >

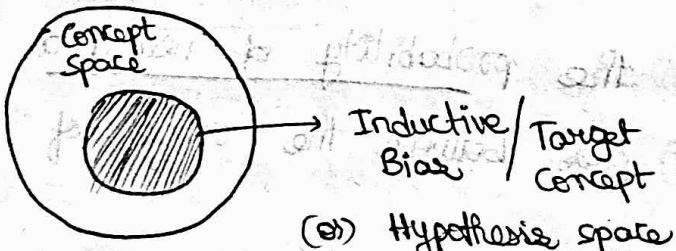
2^d = possible instances

2^{2^d} = possible concepts

Here, $d = 4$

possible instances = $2^d = 16$

possible concepts = 2^{16}



- It is a type of learning based on general concepts from specific training examples.
- Ex: People learn what an actually is (say in terms of its features) by seeing different animals like dogs, cats, etc.. Here dogs, cats are specific concepts whereas Animal is general concept.

$C(x) = x$ is bird If x is cat $\rightarrow C(x) = 0$

If x is parrot $\rightarrow C(x) = 1$

- Learning a category discription (concept) from a set of positive and negative training examples.
- Target Function : a boolean function $c : X \rightarrow \{0, 1\}$.
- Experience : a set of training instances $D : \{\langle x, c(x) \rangle\}$
- Instances of $c(x) = 1 \longrightarrow$ positive examples
  $c(x) = 0 \rightarrow$ negative examples

IMP
- The most general hypothesis — "accept all" is represented by $\langle ?, ?, ?, ? \rangle$

IMP
- The most specific possible hypothesis — "reject all" is represented by $\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$

$\langle x_i, \theta_i \rangle$; $\theta_i = \langle +, - \rangle$ so, $\langle x_i, c(x_i) \rangle$ is the training sample with θ_i as attributes. Like example $\langle$ Redmi, 1 $\rangle$, $\langle$ Apple, 0 $\rangle$.

- Inductive learning Hypothesis : An hypothesis that approximates well over a sufficiently large set of training examples will also approximate the target function (hypothesis) over other unobserved examples.

*. CONCEPT LEARNING AS SEARCH :

- Main goal of this search is to find the hypothesis that best fits the training example.
- Ex : Enjoysport learning Task (6 attributes)

$\langle$ Sky, Air Temp, Humidity, wind, water, Forecast $\rangle$

Sky has values (3) : Rainy, cloudy, sunny;
remaing has only 2 values.

∴ Different instances possible = $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$.

⇒ Syntactically distinct possible hypothesis (Additionall ? & ∅ are added)

So, $= (3+2) \times (2+2) \times (2+2) \times (2+2) \times (2+2) \times (2+2)$

$= 5 \times 4 \times 4 \times 4 \times 4 \times 4 = 5120$

⇒ Semantically distinct possible hypothesis (Null is taken common)

$= 1 + ((5-1) \times (4-1) \times (4-1) \times (4-1) \times (4-1) \times (4-1))$

$= 1 + (4 \times 3 \times 3 \times 3 \times 3 \times 3) = 973$

After finding all the syntactically & semantically distinct hypothesis, we search the best match from all those (i.e. much closer to our learning process).

*. FIND-S ALGORITHM : (finding a maximally specific hypothesis)

- S means most specific, this algorithm considers only positive examples.
- Algorithm :

Step 1 : Initialize with most specific hypothesis (∅)

$h_0 = \langle \phi, \phi, \phi, \phi, \phi \rangle \Rightarrow$ 5 attributes

Step 2 : For each +ve sample,

for each attribute

if (value = hypothesis value) ⇒ Ignore

else Replace with most general hypothesis (?)

Example :

origin	Manufacturer	color	Year	Type	class
Japan	Honda	Blue	1980	economy	+ (Yes)
Japan	Toyota	Green	1970	Sport	− (No)
Japan	Toyota	Blue	1990	economy	+ (Yes)
USA	Audi	Red	1980	economy	− (No)
Japan	Honda	White	1980	economy	+ (Yes)
Japan	Toyota	Green	1980	economy	+ (Yes)
Japan	Honda	Red	1980	economy	− (No)

solution :

$h_0 = \langle \phi, \phi, \phi, \phi, \phi \rangle$

$h_1 = \langle$ Japan, Honda, Blue, 1980, economy $\rangle$

$h_2 = h_1$ (since −ve class)

$h_3 = \langle$ Japan, ?, Blue, ?, economy $\rangle$

$h_4 = h_3$ (since −ve class)

$h_5 = \langle$ Japan, ?, ?, ?, economy $\rangle$

$h_6 = \langle$ Japan, ?, ?, ?, economy $\rangle$

$h_7 = \langle$ Japan, ?, ?, ?, economy $\rangle$ → This is most specific hypothesis

- Disadvantages :

→ Considers only +ve values / ignores −ve values.

→ Here, h_7 may not be sole hypothesis that fits the data.

→ There can be more than one "most specific hypothesis"

→ If training data is inconsistent ⇒ algorithm will mislead.

*. VERSION SPACES :

- subset of hypothesis H consistent with the training examples (d)

$$VS_{H,d} = \{ h \in H \mid \text{consistent}(h, d) \}$$

(h: given hypothesis, H: entire hypothesis)

H → hypothesis (entire) d = training example

If $h(x) = c(x)$ ⇒ the hypothesis is consistent with the training example

- Algorithm : (List then Eliminate)

1. Version Space ← list containing every hypothesis in H
2. From this step, we keep on removing inconsistent hypothesis from version space

   For each training example, $\langle x, c(x) \rangle$ remove any hypothesis that is $h(x) \neq c(x)$

3. output the list of hypotheses into version space after checking for all training examples.

- Advantages :

+ It guarantees to output all hypotheses consistent with training data

- Disadvantages :

- The hypotheses space must be finite.
- Enumeration of all the hypotheses, rather inefficient.

*. CANDIDATE ELIMINATION ALGORITHM :

- It uses the concept of version space.
- Considers both +ve and -ve values (Yes, No). (+ve, -ve)
- Both specific and general hypotheses.
- For positive samples, move from specific to general.
- For negative samples, move from general to specific.

$$S = \{ \phi \ \phi \ \phi \ \phi \ \phi \} \quad + \downarrow$$

$$G = \{ ? \ ? \ ? \ ? \ ? \} \quad - \uparrow$$

- Algorithm :

1. Initialize both general and specific hypothesis (S, G)

$$S = \{ \phi, \phi, \phi, \ldots, \phi \} \quad \& \quad G = \{ ?, ?, ? \ldots, ? \}$$

2. For each example,
   if example is positive → make specific to general
   else example is negative → make general to specific

Example : EnjoySport Learning Task.

S.No	Sky	Temperature	Humidity	Wind	Water	Forecast	Enjoy
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

No. of Attributes = 6

$\Rightarrow S_0 = \{\phi, \phi, \phi, \phi, \phi, \phi\}$, $G_0 = \{?, ?, ?, ?, ?, ?\}$

(1) +ve

$S_1 = \{Sunny, Warm, Normal, Strong, Warm, Same\}$

$G_1 = \{?, ?, ?, ?, ?, ?\}$

(2) +ve, (specific to general)

$S_2 = \{Sunny, Warm, ?, Strong, Warm, Same\}$

$G_2 = \{?, ?, ?, ?, ?, ?\}$

(3) -ve, (general to specific)

$S_3 = \{Sunny, Warm, ?, Strong, Warm, Same\}$

$G_3 = \{\langle Sunny, ?, ?, ?, ?, ?\rangle \langle ?, Warm, ?, ?, ?, ?\rangle$

$\langle ?, ?, ?, ?, ?, Same\rangle\}$

(4) +ve (specific to general)

$S_4 = \{Sunny, Warm, ?, Strong, ?, ?\}$ — Same is removed because it is changed

$G_4 = \{\langle Sunny, ?, ?, ?, ?, ?\rangle \langle ?, Warm, ?, ?, ?, ?\rangle\}$

Finally, S_4 & G_4 are Final Hypothesis

* LEARNED VERSION SPACE :

- The learned Version space correctly describes the target concept, provided :
  1. There are no errors in the training examples.
  2. There is some hypothesis that correctly describes the target concept.
- If S and G converge to a single hypothesis, the concept is exactly learned
- In case of errors in the training, useful hypothesis are discarded, no recovery possible.
- An empty version space means no hypothesis in H is consistent with training examples.
- The learned version space doesnot change with different orderings of training examples.
- Optimal strategy (if you are allowed to choose) :
  - Generate instances that satisfy half the hypotheses in the current version space.
  - Ex: < sunny, Warm, Normal, Light, Warm, Same >
    satisfies 3/6 hyp.
  - Ideally the VS can be reduced by half at each experiment.
  - Correct target found in $\lceil \log_2 |VS| \rceil$ experiments.

*. INDUCTIVE BIAS :

- Remarks on Candidate Elimination & Version Space Algorithm

1) Will the Candidate Elimination algorithm give us correct hypothesis ?

2) What training example should the learner request next ?

- Inductive Learning - From examples we derive rules.
- Deductive Learning - Already existing rules are applied to our examples.

- Biased hypothesis space : Doesnot consider all types of training examples. (Solution → include all hypothesis)

ex : Sunny ∧ warm ∧ normal ∧ strong ∧ cool ∧ change = Yes

- Unbiased hypothesis space : Provided a hypothesis capable of representing set of all examples.

Possible Instances = $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$

Target concepts = 2^{96} (huge) [Practically not possible]

- Idea of Inductive Bias :

Learner generalizes beyond the observed training examples to infer new examples.

'>' → inductively infered from

$x > y$ → y is inductively inferred from x.

Example:

Suppose, learning algorithm = L

training data $D_c = \{x, c(x)\}$

New instance = x_i

Represented as $L(x_i, d_c)$

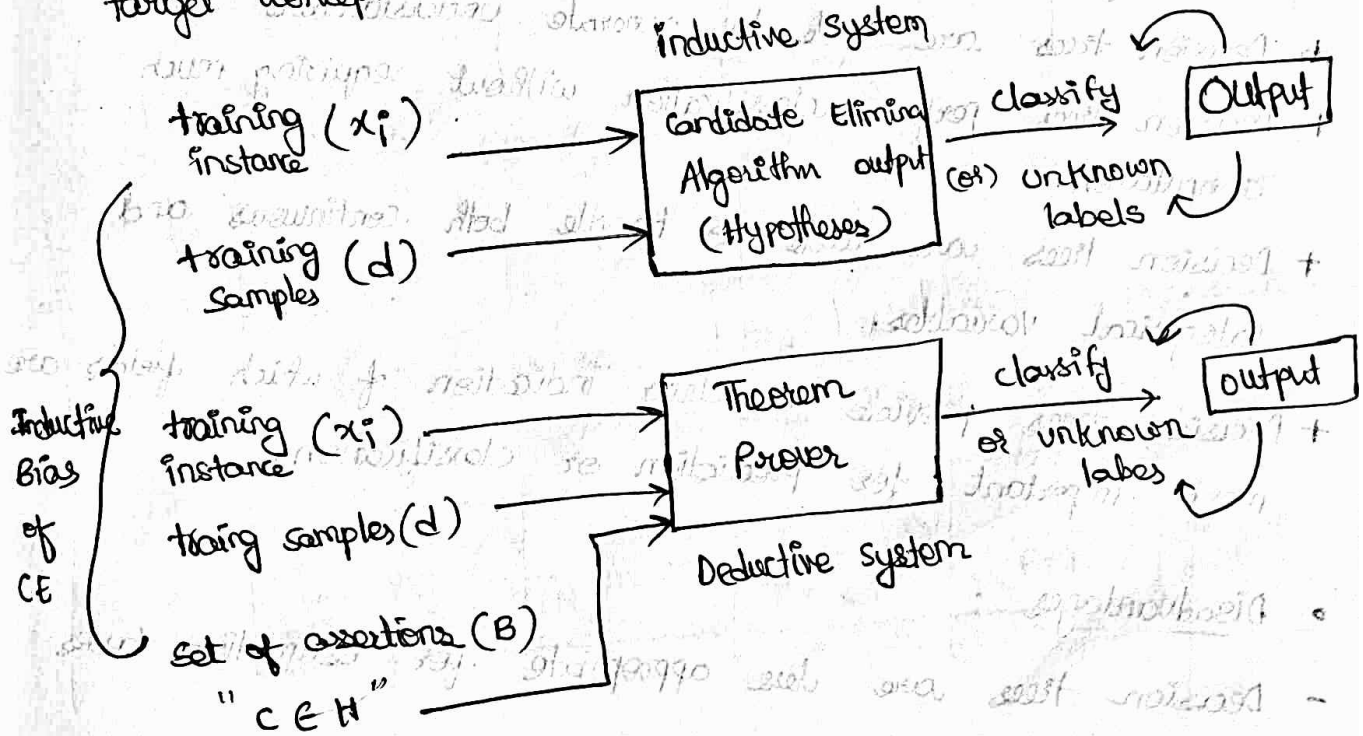
→ $(D_c \wedge x_i) \succ L(x_i, d_c)$: is inductively inferred.

definition: For all instances, which belongs to X, it will hold the assertions (B) & has training instances (d) with target concept (c) for all values of x it implies, Generalizing (L) that has values of x & training instance with target concept.

$$\forall x_i \in X \left[(B \wedge D_c \wedge x_i) \vdash L(x_i, d_c) \right]$$ * IMP

* Case of Candidate Elimination Algorithm:

- Inductive Bias of Candidate elimination says that your "target concept is consistent with the hypothesis."



① ② ③

- Benefit: ① It essentially follows non-procedural way of classifying instances.
  ② → It will give a comparitive strength of different classification algorithms.

- Rote Learner : No Bias
- Find-S Algo : most specific hypotheses (+ve examples)
- Candidate Elimination : $VS_{H,d}$ = classification

Rank these algorithms based on Inductive Bias : Rote learner < Candidate Elimination < Find-S

Produces higher inductive Bias

*. DECISION TREE LEARNING :

- It is a flow chart like tree structure, where each internal node denotes a test on an attribute, each branch represents an output / outcome of the test, & each leaf node (terminal node) holds a class label.

- Advantages :

+ Decision trees are able to generate understandable rules.
+ Decision trees perform classification without requiring much computation.
+ Decision trees are able to handle both continuous and categorical variables.
+ Decision trees provide a clear indication of which fields are most important for prediction or classification.

- Disadvantages :

- Decision trees are less appropriate for estimation tasks

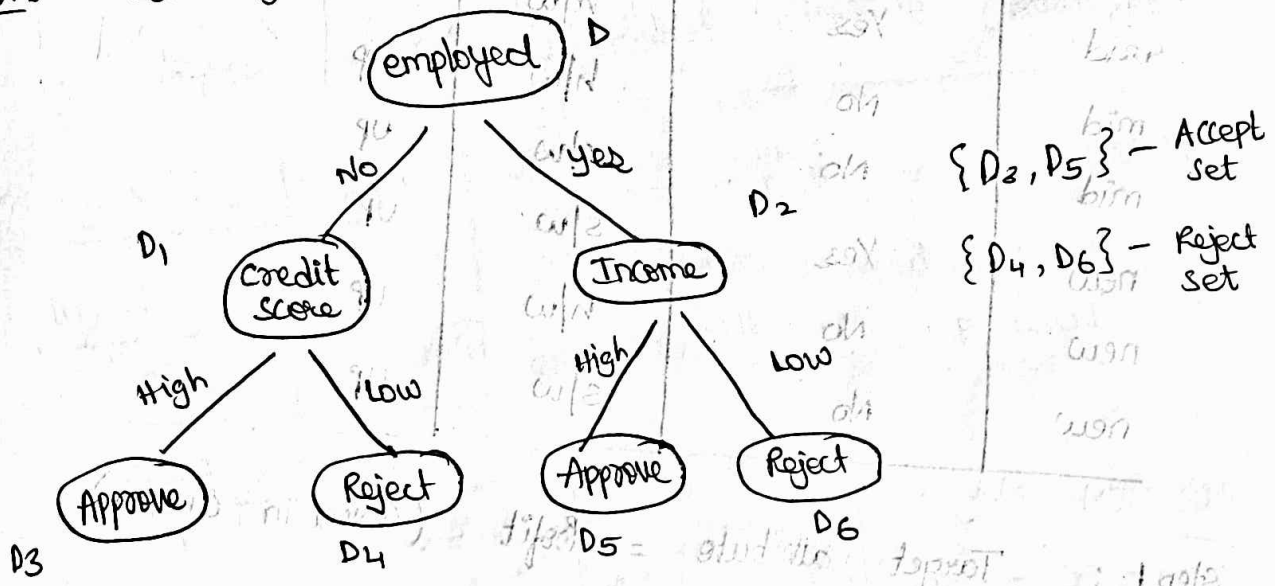
where the goal is to predict the value of a continuous attribute.

- Decision Trees are prone to errors in classification problems with many class and relatively small no. of training examples.
- Decision Tree can be computationally expensive to train.
- The process of growing a decision tree is expensive.
- Used in: tree structed classification & Regression

Raw Data → Decision Tree Algorithm → Classifies the data → yes class / NO class / ... / positive class

- 2 Nodes → Decision Node – root
  → Leaf Node – last

Ex: Loan System (decides if the loan should be approved or rejected



$\{D_3, D_5\}$ – Accept set

$\{D_4, D_6\}$ – Reject set

- Algorithm (ID3):

1. In the given dataset, choose a target attribute.
2. Calculate Information gain of target attribute.

$$IG = \frac{-P}{P+N} \log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N} \log_2\left(\frac{N}{P+N}\right) \quad *$$

3. For remaining attributes, find entropy

Entropy = IG × probability

$$E(A) = \sum \frac{P_i + N_i}{P+N} I(P_i \cdot N_i)$$

4. Calculate **Gain = IG − E(A)**

Example:

Age	Competetion	Type	Profit
old	Yes	S/W	Down
old	No	S/W	Down
old	No	H/W	Down
mid	Yes	S/W	Down
mid	Yes	H/W	Down
mid	No	H/W	up
mid	No	S/W	up
new	Yes	S/W	up
new	No	H/W	up
new	No	S/W	up

Step 1 : Target attribute = Profit (Given in Question)

Step 2 : Information Gain (IG)

$$IG = \frac{-P}{P+N} \log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N} \log_2\left(\frac{N}{P+N}\right)$$

here P - Count of down class = 5 → how many times down occured

N - Count of up class = 5 → how many times up occured

$$= -\frac{5}{10}\log_2\left(\frac{5}{10}\right) - \frac{5}{10}\log_2\left(\frac{5}{10}\right) = 1$$

So, Information Gain, IG = 1

step3 : Calculate entropy for remaining attributes

[Entropy = IG × Probability] → only for non-target attributes

attributes → { age, Competition, Type, Profit }

Target attributes → { Profit } (step 1)

Non-Target attributes → { age, Competition, Type }

For Age ; finding Entropy :

(1) Prepare a table for each attribute

rows - values of undertaken attribute (old, mid, new)

Columns - values of target attribute (down, up)

Age values	down	up
old	3	0
mid	2	2
new	0	3

Entropy = IG × probability

P = down count

N = up count

$$IG(old) = -\left(\frac{3}{3}\log\left(\frac{3}{3}\right) + \frac{0}{3}\log\left(\frac{0}{3}\right)\right) = 0$$

Probability = 3/10 $\left[\frac{\text{No. of times old}}{\text{Total values}}\right]$

Entropy (old) = $0 \times \frac{3}{10} = 0$

$$IG(mid) = -\left(\frac{2}{4}\log\left(\frac{2}{4}\right) + \frac{2}{4}\log\left(\frac{2}{4}\right)\right) = 1$$

Probability = 4/10

Entropy (mid) = $1 \times \frac{4}{10} = 0.4$

$$IG(new) = -\left[\left(\frac{0}{3}\log\left(\frac{0}{3}\right) + \frac{3}{3}\log\left(\frac{3}{3}\right)\right]\right. = 0$$

Probability = 3/10

$$\text{Entrophy (new)} = 0 \times \frac{3}{10} = 0$$

Entrophy (age) = Entrophy (old) + Entrophy (mid) + Entrophy (

$$= 0 + 0.4 + 0 = 0.4$$

Step4 : Gain = IG − E(A)

$$= 1 - 0.4 = 0.6$$

Repeat the same calculation of Gain of Competition, Type

So, we obtain:

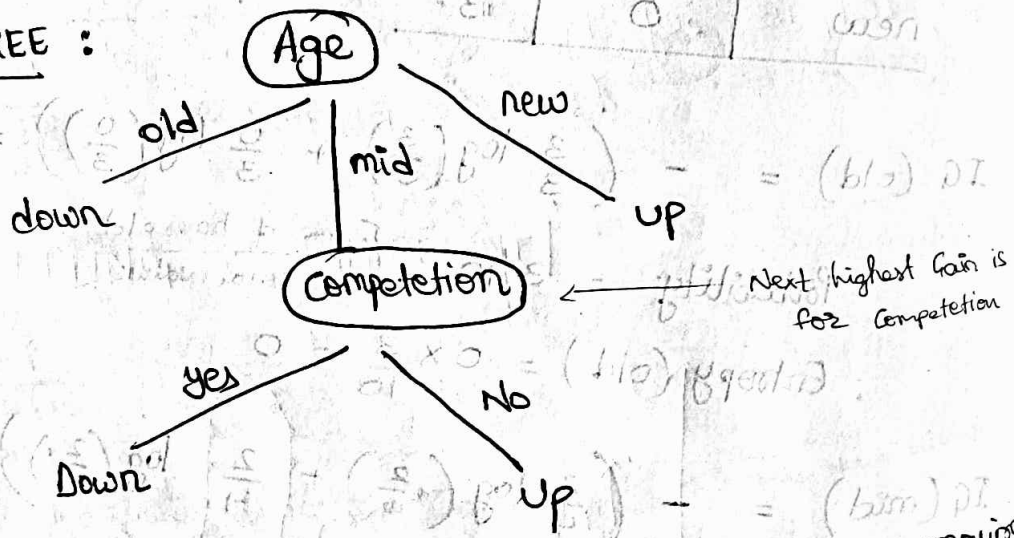
Gain (competition) = 0.124

Gain (Age) = 0.6

Gain (Type) = 0

Highest Gain → decision node (root) → Age.

DECISION TREE :



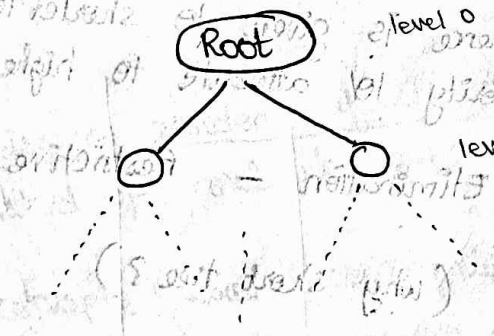
Type → gain = 0 & even in Decision Tree, no requirement

- DT Algorithm is best suited to problems with the following characteristics :

1. Instances are represented by attribute value pairs.
2. The target function has discrete o/p values.
3. Training data can have errors.
4. May contain missing attribute values also.

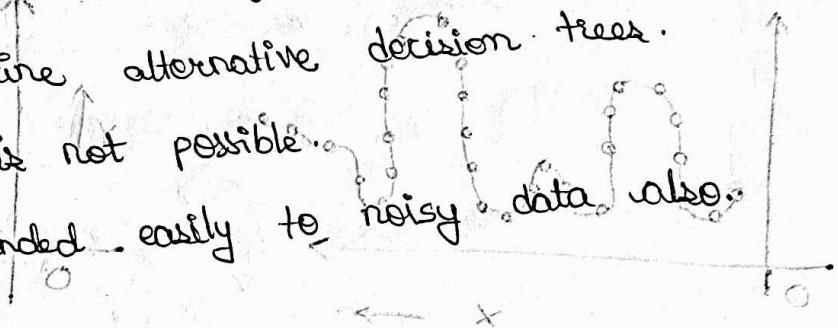
*. HYPOTHESIS SPACE SEARCH IN DTL : (ID3 is DTL algo)

- ID3 can be characterized as searching a space of hypothesis for one that fits the training examples.
- ID3 - will search set of possible decision trees from available hypothesis.
- ID3 performs simple to complex searching.



*. First start with empty tree & keep on adding nodes.

- Every discrete valued (finite) function can be described by some decision tree
- Avoids major risk of [searching] incomplete hypothesis
- Has only single current hypothesis.
- Cannot determine alternative decision trees.
- Backtracking is not possible.
- Can be extended easily to noisy data also.



*. INDUCTIVE BIAS IN ID3 | DTL : {Preference Bias}

→ set of assumptions

- Inductive Bias of ID3 consists of describing the basis by which ID3 chooses one consistent decision tree over all the possible ~~ID3~~ DT's.

- ID3 Search strategy :

1) Select in favour of shorter trees over longer ones
2) Selects element with highest IG as root attribute over lowest IG ones.

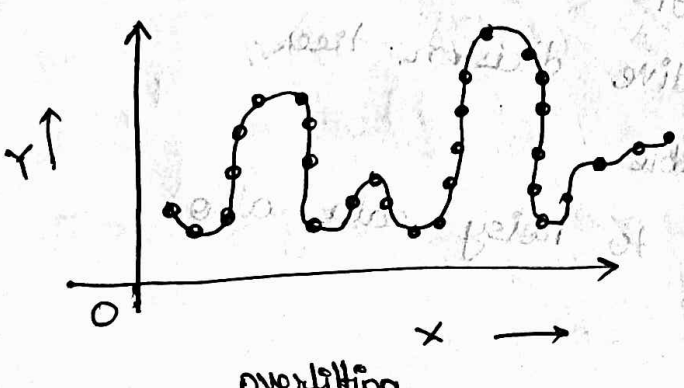
- Types of Inductive Bias :

i. Restrictive bias - based on conditions.
ii. Preference bias - based on priorities.

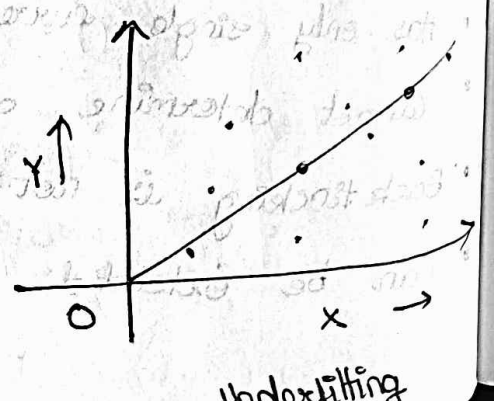
- ID3 - Preference (pefrence is given to shorter tree & priority to attribute to highest IG)
- Version Space & Candidate Elimination - Restrictive Bias
- Why short hypothesis ? (why short tree ?)

→ According to "occums Razor" - Prefer simplest hypothesis that fits the data (into table).

*. OVERFITTING : [IMPORTANT]



overfitting



Underfitting

- Given a hypothesis space H, a hypothesis $h \in H$ is said to OVERFIT the training data if there exists some alternative hypothesis $h' \in H$, such that h has smaller error than h' over the training examples but h' has smaller error than h over the entire distribution of instances (including the instances beyond the training set)

- Reasons for overfitting:
  - Errors and noise in training examples.
  - Small no. of training samples to produce a representative sample

- overfitting happens when a model learns the detail & noise in the training data to the extent that it negatively impacts the performance of the model on new data.

- The noise or random fluctuations in the training data is picked up and learned as concepts by the model.

- These concepts are not applicable to new data and negatively impact the models ability to generalize.

- Avoid overfitting?
  - stop growing when data split not statistically significant
  - Grow full tree, then post-prune.
  - use separate set of examples, distinct from training set.
  - Reduced error Pruning.

(*) Fails to classify test data (new data)

*. REDUCED ERROR PRUNING (REP):

- The process of adjusting Decision Tree to minimize the misclassification error is Pruning.
- Aim is to minimize the overfitting of the model.
- Pre pruning | Early stopping Rule:
  - Halt subtree construction @ some node after checking some measures
  - Stops the tree growth pre maturely.
  - Avoids generating overlay complex subtrees which overfits the training data.
- Post Pruning:
  - Grows DT entirely
  - Trims the nodes of DT in bottom-up fashion
  - Replace nodes by leaf.
  - If error increases after trimming replace sub-tree by a leaf node.
- REP hold out some available instances called Pruning set
- Pruning set provides a estimate of error rate on future instances than training data.

```
if (error (parent) < error(child))
{
    prune the tree & replace node with leaf
}
else dont prune, repeat the process.
```

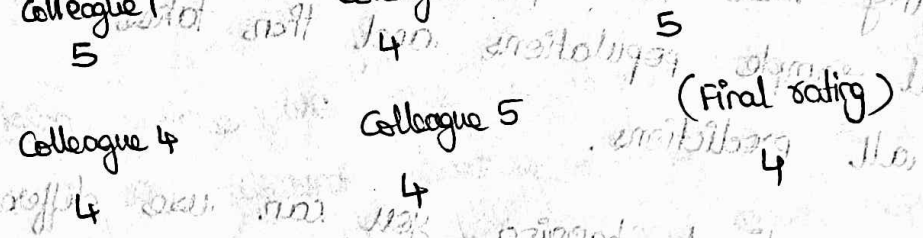
*. ENSEMBLE LEARNING :

- It is a process by which multiple models, such as classifiers or experts, are strategically generated & combined to solve a problem.
- Ensemble learning is primarily used to improve the (classification, prediction, function approximation) performance of a model.

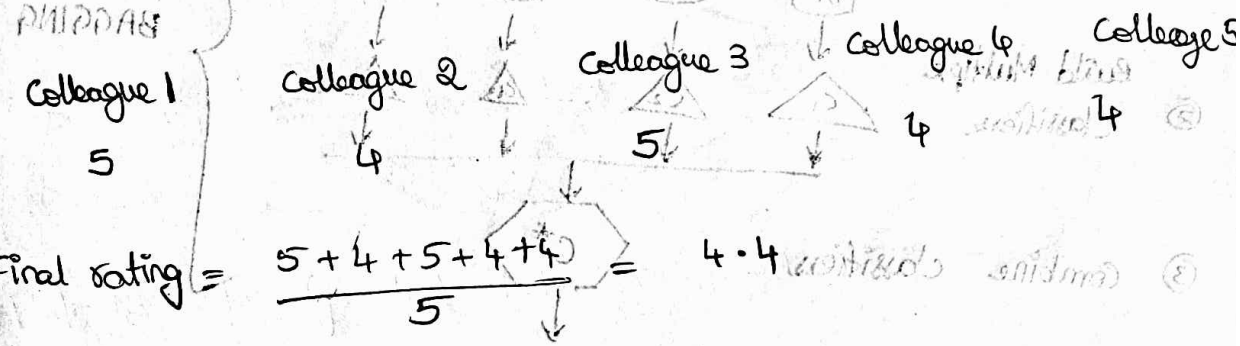
*. Basic Ensemble Techniques :

① Max Voting : The max voting method is generally used for classification problems. In this technique, multiple models are used to make predictions for each data point.

Ex:



② Averaging : Multiple predictions are made for each data point in averaging. In this method, "we take an average of predictions from all the models and use it to make predictions (final)."



③ Weighted Average : "All models are assigned different weights defining the importance of each model for prediction."

Ex:

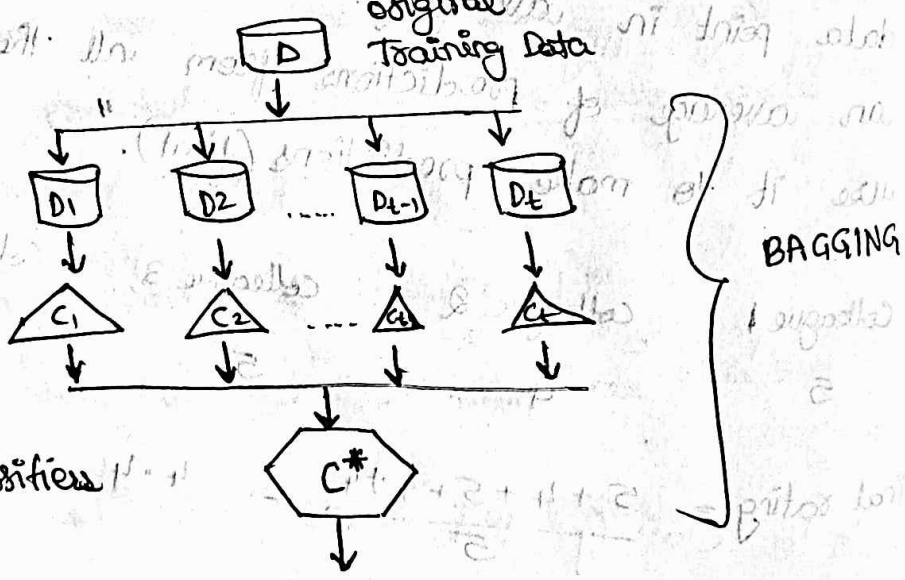
	Colleague 1	Colleague 2	Colleague 3	Colleague 4
Weight	0.23	0.23	0.18	0.18
rating	5	4	5	4

Final rating = $(5 \times 0.23) + (4 \times 0.23) + (5 \times 0.18) + (4 \times 0.18) + (4 \times 0.18) = 4.41$

*. ADVANCED ENSEMBLE LEARNING :

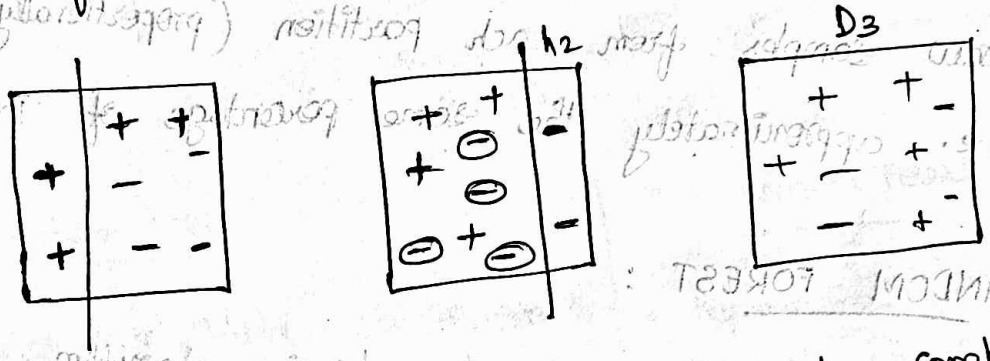
① Bagging :

- Bagging tries to implement similar learners on small sample populations and then takes a mean of all predictions.
- In generalized bagging, you can use different learners on different population. This helps to reduce variance er

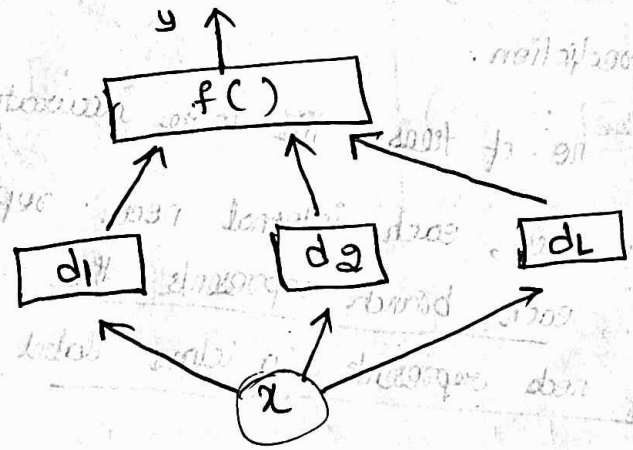


② Boosting :

- Boosting is an iterative technique which adjust the weight of an observation based on the last classification.
- If an observation was classified incorrectly, it tries to increase the weight of this observation and vice-versa.
- In general, it decreases the bias error and builds strong predictive models. However, they may sometimes over fit on the training data.



③ Stacking: Here we use a learner to combine output from different learners. This can lead to decrease in either bias or variance error depending on the combining learner we use.



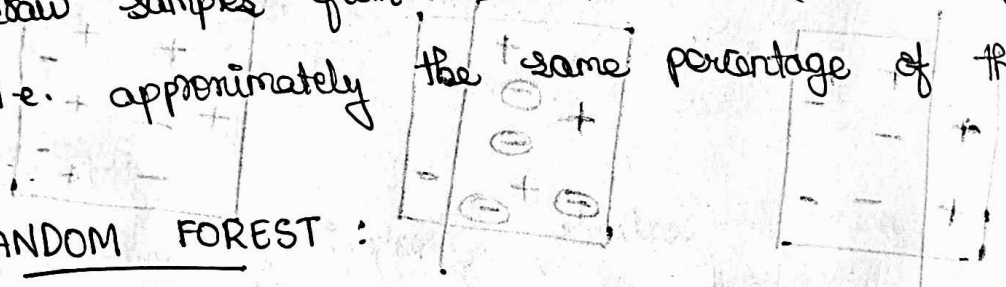
*. TYPES OF SAMPLING :

① Simple random sampling : There is an equal probability of selecting any particular item.

② Sampling without replacement : once an object is selected, it is removed from the population.

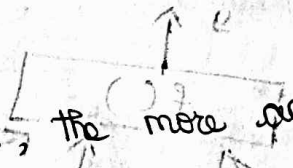
③ Sampling with replacement : A selected object is not removed from population.

④ Stratified sampling : Partition the data set, and draw samples from each partition (proportionally i.e. approximately the same percentage of the data)



*. RANDOM FOREST :

- Random Forest is a supervised learning algorithm.
- The forest it builds, is an ensemble of decision trees, usually trained with the bagging method.
- Working Rule: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

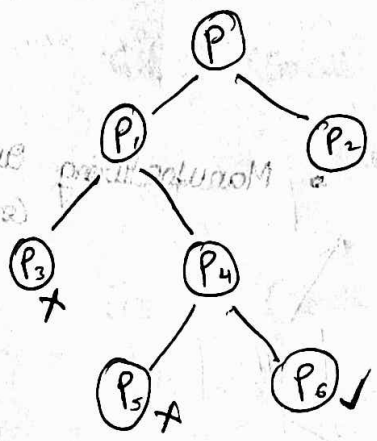


- The larger the no. of trees, the more accurate the result.
- In a decision tree, each internal node represents a "test" on a attribute, each branch represents the outcome of test, and each leaf node represents a class label

- The output is,
  - the mode of the predicted classes (in case of classification)
  - the mean of predicted value (in case of regression) provided by individual trees.

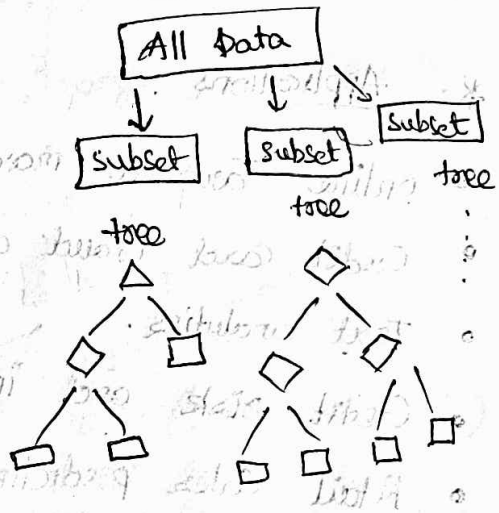
Decision Tree

- Decision tree builds classification models in the form of tree.
- It breaks down a dataset into smaller & smaller subsets



Random Forest

- Random Forest is an ensemble classifier made of many decision tree
- ensemble models combine the results from different models.



*. Advantages of Random Forests :

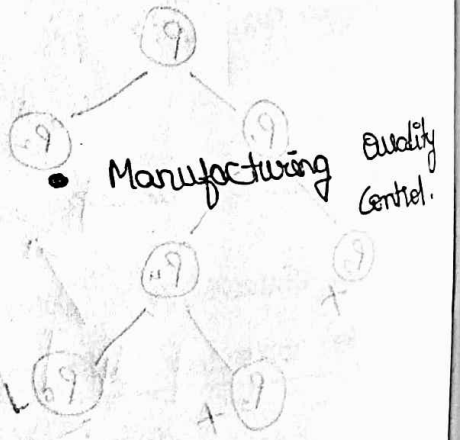
- Automatic identification of important predictors.
- each decision tree is independent. Therefore trees can be grown on different cores or different computers, allowing for quicker analysis.
- It can be used for both classification and regression
- In Random Forest algorithm, if there are enough trees in the forest, the classifier won't overfit the model.
- The Random forest can handle missing values, ~~inconsistent data~~, new data, etc...
- It is modelled for categorical values.

*. Disadvantages :

- Suited for wide datasets with only a moderate no. of rows. Breiman recommends the use of other tools for larger datasets.
- Large memory needed to store built models.
- overfitting might be used/seen with noisy data.

*. Applications :

- Online Targeted marketing.
- Credit card Fraud detection.
- Text analytics.
- Credit risk and insurance risk.
- Retail Sales prediction.
- Biological & Medical Research
- Manufacturing Quality Control.



*. Candidate Elimination Algorithm Example :

Given Dataset,

Size	Color	Shape	class / Label
Big	Red	Circle	No
Small	Red	Triangle	No
Small	Red	Circle	Yes
Big	Blue	Circle	No
Small	Blue	Circle	Yes

S1:

S0 : (∅, ∅, ∅) } Initialization

G0 : (?, ?, ?)

S1 : (∅, ∅, ∅)

G1 : (Small, ?, ?) (?, Blue, ?) (?, ?, Triangle)

S2 : (∅, ∅, ∅)

G2 : (Small, Blue, ?) (Small, ?, Circle)

(?, Blue, ?) (Big, ?, Triangle) (?, Blue, Triangle)

S3 : (Small, Red, Circle)

G3 : (Small, ?, Circle)

S4 : (Small, Red, Circle)

G4 : (Small, ?, Circle)

S5 : (Small, ?, Circle)

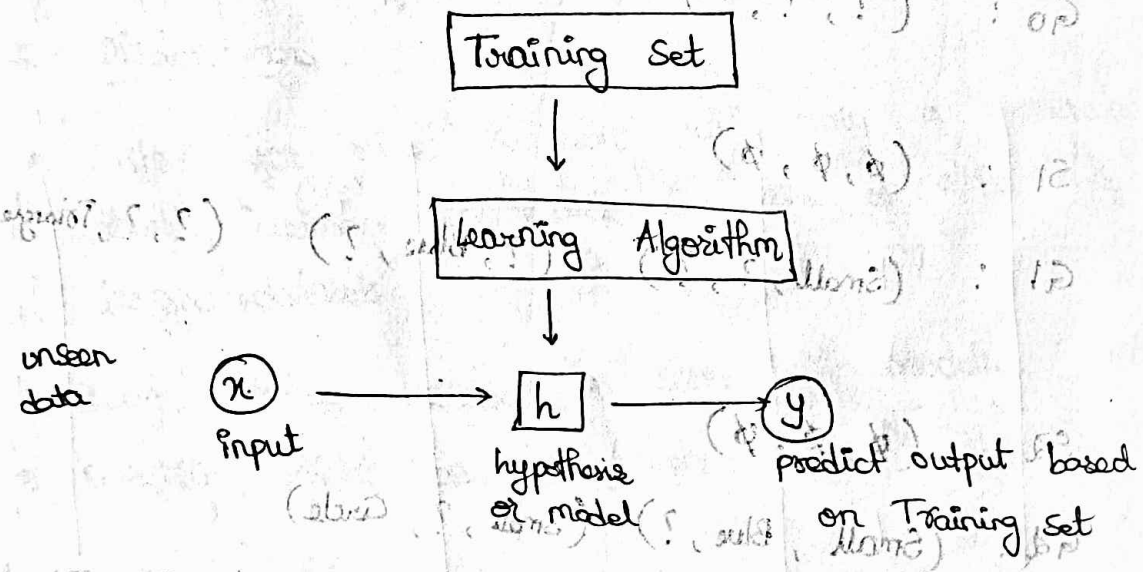
G5 : (Small, ?, Circle)

*. MODULE-2

*. LINEAR REGRESSION :

Labelled Data	Supervised Learning	Unsupervised learning
Discrete	Classification	clustering
Continuous	Regression	Dimensionality Reduction

- In a simple way, regression is estimating unknown values from set of known values.



Training Set notation :

m = Number of training examples.

x = Input variable / features.

y = Output variable / target variable.

(x, y) = One training example

$(x^{(i)}, y^{(i)})$ = i^{th} training example.

$$y = h_\theta(x) = \theta_0 + \theta_1 x$$

θ_0, θ_1 are parameters / weights that control hypothesis

$h_\theta(x)$ - predicted output from Training set

y - actual output

- error = $|y - h_\theta(x)|$ → this error should be very low

Minimizing error is our goal & that is the best fit lin

*. Cost Function :

Idea : choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training example (x, y)

minimize θ_0, θ_1 $\quad \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 = J(\theta_0, \theta_1)$

Simplified version : (1 parameter)

1. Hypothesis : $h_\theta(x) = \theta_1 x$, $\theta_0 = 0$
2. Parameters : θ_1
3. Cost Function : $J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$
4. Goal : minimize $J(\theta_1)$ θ_0, θ_1 → we use method called Gradient Descent

*. Gradient Descent :

Repeat until convergence {

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(for } j = 0 \text{ ; } j = 1\text{)}$$

α : Learning rate (step size)

- Gradient descent can converge to a local minimum, even with the learning rate α fixed
- As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.
- Linear regression model:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$$

- "Batch": Each step of gradient descent uses all the training examples

*. MULTIPLE LINEAR REGRESSION :

Ex:

$x^{(i)}$	Size in ft^2 (x_1)	No. of Bedrooms (x_2)	No. of Floors (x_3)	Age of home (x_4)	Price (y)
x^1	2104	5	1	45	460
x^2	1416	3	2	40	232
x^3	1534	3	2	30	315
x^4	852	2	1	36	178
⋮	⋮	⋮	⋮	⋮	⋮

Notation:

n = No. of features

$x^{(i)}$ = input features of i^{th} training example

$x_j^{(i)}$ = value of feature j in i^{th} training example

$x_3^{(2)} = 2$

$x_3^{(4)} = 1$

$n = 4$

$x_4^{(3)} = 30$

Now, Hypothesis $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots \theta_n x_n$

$$\Rightarrow \quad x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \in R^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in R^{n+1}$$

- $h_\theta(x) = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n = \theta^T \cdot x$

$$h_\theta(x) = \underset{1\times n}{\begin{bmatrix} \theta_0 & \theta_1 & \theta_2 & \dots & \theta_n \end{bmatrix}} \underset{n\times 1}{\begin{bmatrix} x_0 \\ \vdots \\ x_n \end{bmatrix}} = [H]_{1\times 1}$$

Now, Gradient Descent will be:

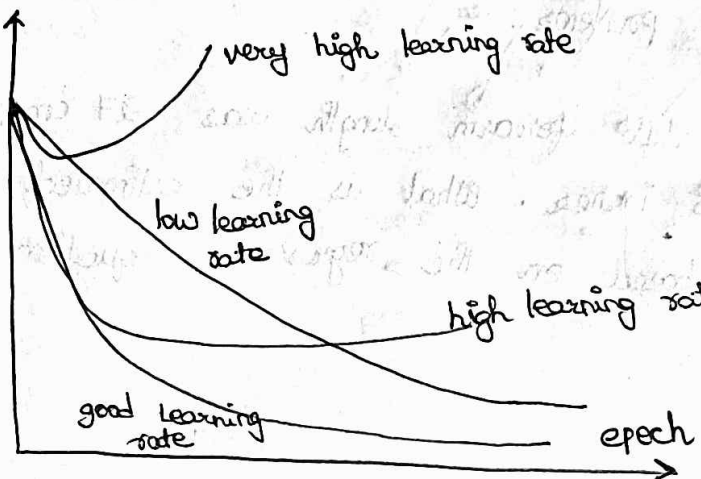
Repeat until convergence $\{$

$$\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)\, x_j^i$$

$\}$ $\quad n \geq 1$

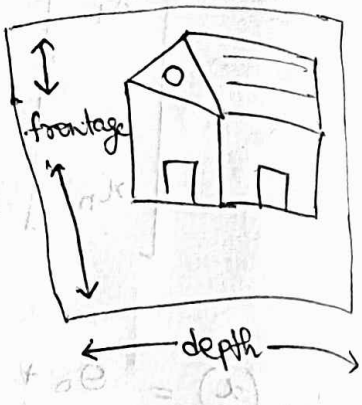
*. Gradient Descent : Learning rate

- Automatic convergence test
- α too small : slow convergence
- α too large : may not converge



*. POLYNOMIAL REGRESSION :

→ House price prediction :



- Features : frontage, depth.
- $h_\theta(x) = \theta_0 + \theta_1 \times \text{frontage} + \theta_2 \times \text{depth}$
- From given features we can generate new feature called Area.
- Area, x = frontage × depth
- $h_\theta(x) = \theta_0 + \theta_1 x$
- $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \ldots$

  $= \theta_0 + \theta_1 (\text{size}) + \theta_2 (\text{size})^2 + \theta_3 (\text{size})^3 \ldots$

  (or)

  $= \theta_0 + \theta_1 (\text{size}) + \theta_2 (\sqrt{\text{size}}) + \theta_3 (\sqrt[3]{\text{size}}) \ldots$

*. Example Problems : (LINEAR REGRESSION)

The question is based on a regression equation for 55 college students with X = left forearm length (cm) & y = height. The forearm lengths ranged from 22 cm to 31 cm. The regression equation is $\hat{y} = 30.3 + 1.49x$.

Solve the following problems :

① One student's left forearm length was 27 cm, and his height was 75 inches. What is the estimated height for this student, based on the regression equation ?

② One students' left forearm length was 22 cm, and her height was 63 inches. What is the estimated height for this student, based on the regression equation?

③ What is the residual (Error) in problem 1, problem 2?

Δy = residual error = |given − estimated|

① $\hat{y} = 30.3 + 1.49\,x$

Given, forearm length = 27 cm

height of student = 75 inch = 190.5 cm

height = 30.3 + 1.49 (forearm length)

height = 30.3 + 1.49 (27) = 68.1 cm

$\hat{y} - y = \Delta y$

$\Delta y = 190.5 - 68.1 = 122.4$ cm (residual error)

② $x = 22$ cm

$y = 63$ inch = 160.02 cm

According to our given regression line,

$\hat{y} = 30.3 + 1.4 \times 22$

$= 61.1$ cm

$\Delta y = 160.02 - 61.1 = 98.92$ (residual error)

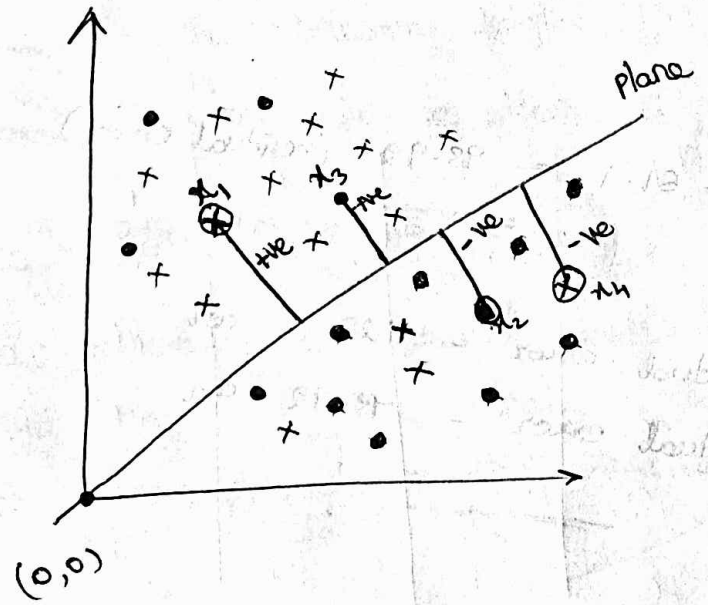
③ Problem 1 residual error = 122.4 cm

Problem 2 residual error = 98.92 cm

*. LOGISTIC REGRESSION :

- Regression used to fit a curve to data in which the dependent variable is binary, or dichotomous.
- Ex: Coronary Heart Disease (CD) and Age : In this study sampled individuals were examined for signs of CD (present = 1 / absent = 0) and the potential relationship between this outcome & their ages (yrs.) was considered.
- Logistic Regression is basically a Binary Classification problem statement.
- MP Sometimes, a (Linear Regression) straight line or best fit line can sometimes classify the data (yes/No). But if there is any outlier point that completely changes the best fit line, then the "Linear Regression" concept fails to solve Binary classification problem. (high error rate)

  Hence "LOGISTIC REGRESSION" comes into picture



$x \longrightarrow y_i$: +ve = 1

$\bullet \longrightarrow y_i$: -ve = -1

above the plane > 0

below the plane < 0

case ①: $y_1 = +1$ (because it is ×)

$w^T x_1 > 0$ (above the plane)

$\boxed{(y_1 \cdot w^T x_1) > 0}$ ⟹ This is easily classified.

case ②:

$y_2 = -1$ (because it is •)

$w^T x_2 < 0$ (below the plane)

$\boxed{(y_2 \cdot w^T x_2) > 0}$ ⟹ This is getting classified properly

case ③:

$y_3 = -1$ (because it is •)

$w^T x_3 > 0$ (above the plane)

$\boxed{(y_3 \cdot w^T x_3) < 0}$ ⟹ This (x_3, y_3) is incorrectly classified.

[OR] $y_4 = +1$ (because it is ×).

$w^T \cdot x_4 < 0$ (below the plane)

$\boxed{(y_4 \cdot w^T x_4) < 0}$ ⟹ This (x_4, y_4) point is not properly classified.

The best fit line that linearly seperates (×, •) points make sure that the sum of all the points along with the distance should be maximum (cost function must be maximum

$\left\{ = \sum_{i=1}^{n} y_i \, w_i^T \, x_i \right\}$ → this should be maximum.

Sigmoid Function : $F(z) = \frac{1}{1+e^{-z}}$

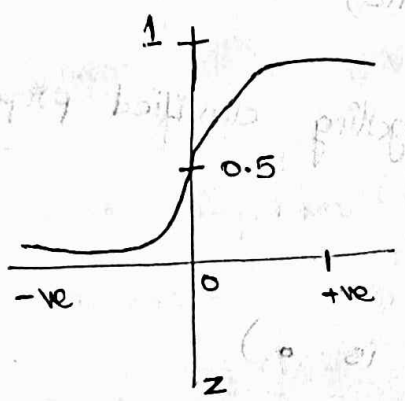
where $\boxed{z = y_i \cdot \omega^T x_i}$

Here, to prevent chance of best fit line, (outliers)

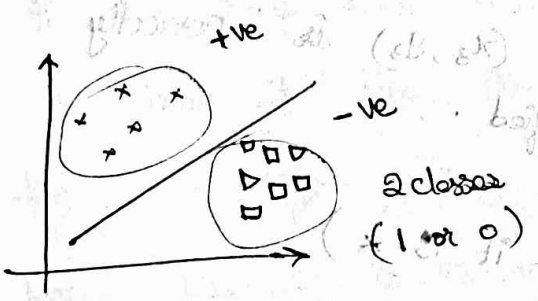
$\left\{ = \max \sum_{i=1}^{n} f\left(y_i \cdot \omega^T x_i\right) \right\}$ between 0 to 1

any no. of outliers that may interrupt the possibility of best fit line

$\Rightarrow$ The value comes out to be between 0 & 1.

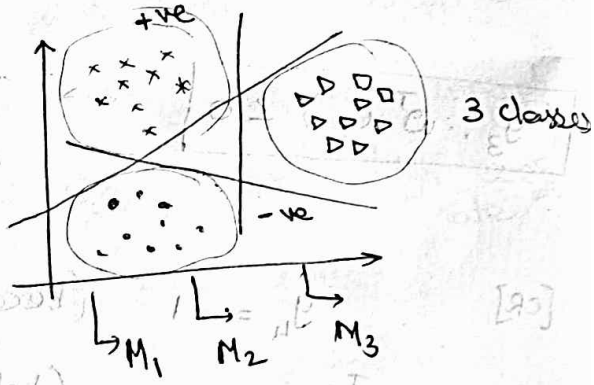


*. **Multiclass classification (one vs Rest) :**



2 classes (1 or 0)

Binary class classification



3 classes

Multiclass classification

Suppose, Example :

f_1	f_2	f_3	output
I_1	I_2	I_3	O_1
I_4	I_5	I_6	O_2
I_7	I_8	I_9	O_3
I_{10}	I_{11}	I_{12}	O_1
I_{13}	I_{14}	I_{15}	O_2

O_1	O_2	O_3
+1	-1	-1
-1	+1	-1
-1	-1	+1
+1	-1	-1
-1	+1	-1

$h_\theta(x)$ = estimated probability that $y = 1$ on input x

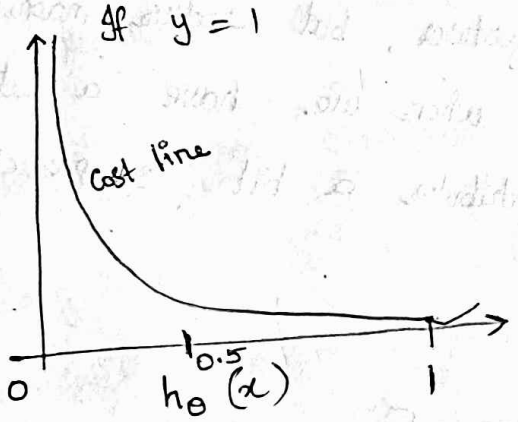
Ex: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix} \Rightarrow h_\theta(x) = 0.7$

Tell the patient that 70% chance of tumor being malignant

Probability that $y = 1$, given

*. Logistic Regression Cost Function :

$$\text{Cost}\left(h_\theta(x), y\right) = \begin{cases} -\log\left(h_\theta(x)\right) & \text{if } y = 1 \\ -\log\left(1 - h_\theta(x)\right) & \text{if } y = 0 \end{cases}$$



$h_\theta(x)$ = predicted value

y = Actual value

*. REGULARIZATION :

• Overfitting : If we have too many features, the learned hypothesis may fit the training set very well. but fail to generalize to new examples (predict prices on new examples).

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 \approx 0$$

underfitting — high bias, low variance.

overfitting — high variance, low bias.

- **Bias - Variance Trade off :**

1) **Bias** - difference between what you expect to learn and truth. It decreases with more complex model.

2) **Variance** - difference between what you expect to learn and what you learn from a particular dataset. It measures how sensitive learner is to specific dataset. Increases with more complex model.

- Reduce overfitting → 1. Reduce no. of features (not so easy)
  → 2. Regularization. (better sometimes)

- **REGULARIZATION :** Keep all features, but reduce magnitude of parameter θ. Works well when we have a lot of features, each of which contributes a bit to predicting y.

- **Regularized Linear regression :**

$$\theta_j = \theta_j \left(1 - \alpha \frac{\eta}{m}\right) - \alpha \frac{1}{m} \left[\sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) x_j^{(i)} \right]$$

- **Unregularized Linear regression :**

$$\theta_j = \theta_j - \alpha \frac{1}{m} \left[\sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) x_j^{(i)} \right]$$

Here $1 - \alpha \frac{\eta}{m} < 1$: Weight decay (or) $\alpha\left(\frac{\eta}{m}\right) > 0$

*. INSTANCE BASED LEARNING :

- Instance based learning is often termed "lazy learning", as there is typically no "transformation" of training instances into more general "statements".
- Instead, the presented training data is simply stored and, when a new query instance is encountered, a set of similar, related instances is retrieved from memory and used to classify new query instance.
- Hence, instance-based learners never form an explicit general hypothesis regarding the target function. They simply compute the classification of each new query instance as needed.

*. Nearest neighbor classifier:

- Training Data : $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}) \ldots (x^{(N)}, y^{(N)})$
- Learning : Do nothing
- Testing : $h(x) = y^{(k)}$, where $k = \underbrace{\text{argmin}_i \; D(x, x^{(i)})}_{\text{take Min distance}}$.

- K-NN Approach : (K must be odd)

- The simplest, most used instance-based learning algorithm is the K-NN algorithm.
- K-NN assumes that all instances are points in some n-dimensional space and defines neighbors in terms of distance (usually Euclidean in R-space).
- K is the no. of neighbors considered.
- Voronoi cell : Graphic depiction of K-nearest neighboring points. works when K = 1

Algorithm :

1. Load Data.
2. Initialize K. (Rule of thumb)
3. For each sample in the training data,
   - 3.1 Calculate distance between query point & current point.
   - 3.2 Add the distance & index of example to an ordered collection.
4. Sort → ordered collection of distances & indexes from small to large.
5. Pick first K entries from sorted collection.
6. Get the labels of selected K entries.
7. If regression → return mean of K labels.
   If classification → return mode of K labels.

How to choose K ?

- Rule of thumb, $K = \sqrt{n}$, n is no. of examples in data
- In practice, K = 1 is often used for efficiency, but can be sensitive to "noise"

CAT

Ex: We have data from the questionnaires survey and objective testing with two attributes (acid durability and strength) to classify whether a special paper tissue is good or not. Consider below 4 training samples.

x_1 = Acid Durability	x_2 = Strength	Y = classification
7	7	BAD
7	4	BAD
3	4	GOOD
1	4	GOOD

Now the factory produces a new paper tissue that passes the laboratory test with $x1 = 3$, $x2 = 7$. Guess the classification of new tissue.

Sol: Step 1 : Initialize & define k (assume k=3)

Step 2 : Compute distance between input sample and training sample

- Coordinate of the input sample is (3, 7).
- Instead of calculating the Euclidean distance, we calculate the squared Euclidean distance.

X1	X2	Squared Euclidean distance	Rank	?
7	7	$(7-3)^2 + (7-7)^2 = 16$	3	Yes
7	4	$(7-3)^2 + (4-7)^2 = 25$	4	No
3	4	$(3-3)^2 + (4-7)^2 = 09$	1	Yes
1	4	$(1-3)^2 + (4-7)^2 = 13$	2	Yes

7 7 – Bad
7 4 – X
3 4 – Good
1 4 – Good

Majority = Good $\Rightarrow$ (3, 7) $\Rightarrow$ GOOD

∴ The classification of new tissue = GOOD (3,7)

Conclusions on KNN :

- KNN works well on many practical problems and is fairly robust (depending on value of k)

- K-NN is subject to the curse of dimensionality (i.e., presence of many irrelevant attributes).
- K-NN needs adequate distance measure.
- K-NN relies on efficient indexing.

Applications of KNN :

- Used in classification.
- Used to get Missing values.
- Used in pattern recognition.
- Used in gene expression.
- Used to get 3D structure.
- Used to measure document similarity.

*. DISTANCE - WEIGHTED KNN :

- Weights = F (distance) = $\frac{1}{\text{distance}}$

Give weights based on distance & Predict class labels based on the weighted - sum not on majority vote.

Ex :

Neighbors	Time label	Distance	weight	Sum of Weights
x_1	Positive	0.1	10	10 + 3.3 = 13.3
x_2	Positive	0.3	3.3	
x_3	Negative	1	1	1 + 0.5 + 0.3 = 1.8
x_4	Negative	2	0.5	
x_5	Negative	3	0.3	

- KNN is a lazy algorithm because it doesnot learn a discriminative function from the training data but memorizes the training dataset instead. There is no training time in K-NN.

*. LOCALLY - WEIGHTED REGRESSION :

- To overcome the problem of non-linearly seperable data, linear locally weighted Regression assigns weights to data to overcome the ~~probability~~ problem.
- This is computationally more expensive.
- Finding / Assigning weights → Kernal Smoothing $\left[\text{weight} \propto \frac{1}{\text{Dist}}\right]$

$$D = a\, e^{-\frac{\|x - x_0\|}{2c^2}}$$

x – each training input.

x_0 – value we are predicting.

We construct a weight matrix (w)
- for each training input (x) and for the value we are trying to predict (x_0)

weight matrix – diagonal matrix

$$\boxed{\beta = (x^T w x)^{-1} \cdot x^{-1} \cdot w \cdot y}$$

β = Model Parameter

then prediction can be defined as,

* $\boxed{y = \beta x_0}$; y → prediction

Drawbacks :

1. Need to evaluate whole dataset everytime.
2. Computation cost is more.
3. Memory requirement is more.

*. RADIAL BASIS FUNCTIONS (Module-5 with Neural Networks)

*. CASE BASED LEARNING :

- All instance Based learners have 3 properties.
  1) They are lazy learners.
  2) classification is different from each instance.
  3) Instances are represented with n dimensional euclidean space.
- In case based Reasoning, everything is considered as case and based on previous cases - we propose a solution.
- Instances are represented as symbols (not values).

CBR has 3 Components :

1. Similarity functions or distance measures.
2. Approximation / Adjustment of instances.
3. Symbolic representation of instances.

- For modelling Case based Reasoning, we use case based design tool (CADET) system. that has 75 predefined libraries.

*. EXAMPLES : (KNN)

The table shows lists of dataset that was used to create a nearest neighbour model that predicts whether it will be a good day to go surfing.

ID	Wave Size	Wave Period	Wind Speed	Good Surf
1	6	15	5	yes
2	1	6	9	no
3	7	10	4	yes
4	7	12	3	yes
5	2	2	10	no
6	10	2	20	no

Assuming that the model uses Euclidean distance to find the nearest neighbour, what prediction will the model return for each of the following query instances.

ID	Wave Size	Wave Period	Wind Speed	Good Surf
Q1	8	15	2	?
Q2	3	2	18	?
Q3	6	11	4	?

Sol: Given query instance Q1 has

Wave Size = 8 , Wave Period = 15

Wind Speed = 2 ,

Finding the Euclidean distance between Q1 & the given lists in dataset for ID1, ID2, ID3, ... ID6.

ID1 :

Wave size ⇒ $(6-8)^2$

Wave Period ⇒ $(15-15)^2$

Wind Speed ⇒ $(5-2)^2$

$d = \sqrt{(-2)^2 + (0)^2 + (3)^2}$

$d = \sqrt{4+0+9} = \sqrt{13}$

$d = 3.605$

ID2 :

Wave size ⇒ $(1-8)^2$

Wave Period ⇒ $(6-15)^2$

Wind Speed ⇒ $(9-2)^2$

$d = \sqrt{(-7)^2 + (-9)^2 + (7)^2}$

$d = \sqrt{49+81+49} = \sqrt{179}$

$d = 13.379$

ID3 :

Wave size ⇒ $(7-8)^2$

Wave Period ⇒ $(10-15)^2$

Wind Speed ⇒ $(4-2)^2$

$d = \sqrt{(-1)^2 + (-5)^2 + (2)^2}$

$d = \sqrt{1+25+4} = \sqrt{30}$

$d = 5.477$

ID4 :

Wave size ⇒ $(7-8)^2$

Wave Period ⇒ $(12-15)^2$

Wind Speed → $(3-2)^2$

$d = \sqrt{(-1)^2 + (-3)^2 + (1)^2}$

$d = \sqrt{1+9+1} = \sqrt{11}$

$d = 3.316$

ID5 :

Wave size ⇒ $(2-8)^2$

Wave Period ⇒ $(2-15)^2$

Wind Speed ⇒ $(10-2)^2$

$d = \sqrt{(-6)^2 + (-13)^2 + (8)^2}$

$d = \sqrt{36+169+64} = 16.401$

ID6 :

Wave size ⇒ $(10-8)^2$

Wave Period ⇒ $(2-15)^2$

Wind Speed ⇒ $(20-2)^2$

$d = 22.293$

ID	Euclidean distance	Rank	output
1	3.605	2	yes
2	13.379	4	no
3	5.477	3	yes
4	3.316	1	yes
5	16.401	5	no
6	22.293	6	no

Assuming $K = 3$, considering 3 nearest neighbors

ID = 4 → yes
ID = 1 → yes
ID = 3 → yes
} majority = YES

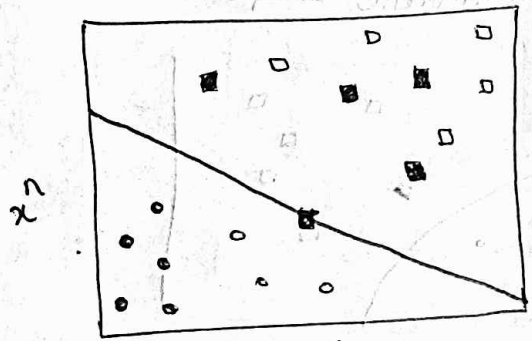
∴ New Query Instance a1 with Wave Size = 8 FT
Wave Period = 15 sec and Wind Speed = 2 MPH
It will be a GOOD SURF (YES).

---

① $\min \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i$ s.t. $\forall i = 1, \ldots n$ $C = 0.1$

$\xi_i \geq 0$ & $(w \cdot x_i + b) y_i - (1 - \xi_i) \geq 0$

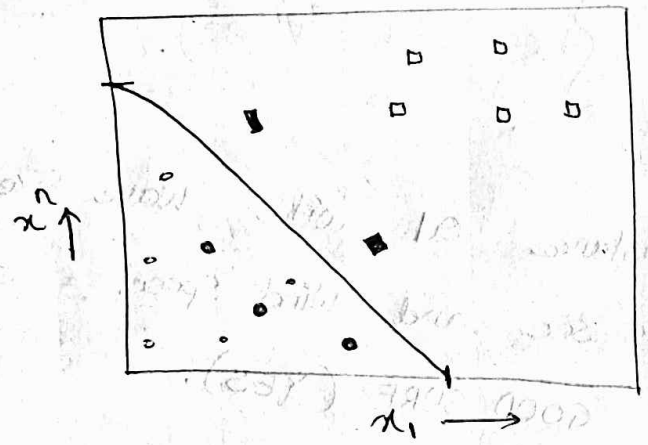
- The smaller error penalty C means that the decision boundary will have a larger margin



(2) $\min \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i$ s.t. $\forall i = 1, \ldots n$

$\xi_i \geq 0$, $(w \cdot x_i + b) y_i - (1 - \xi_i) \geq 0$ $C = 1$

- From the distance of the support vectors to the decision boundary, we can infer that margin here is small because the primal optimization reduces the width of the margin as the penalty of margin as the penalty increase, it is Linear kernel with the larger error penalty C.

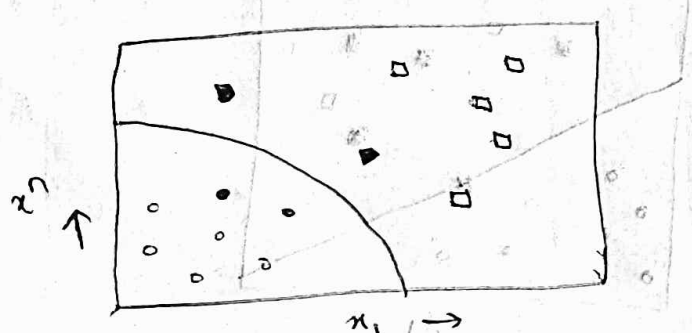


(3) $\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i, \alpha_j y_i y_j k(x_i, x_j)$

such that $\sum_{i=1}^{n} \alpha_i y_i = 0$; $\alpha_i \geq 0 \quad \forall i = 1, \ldots n$

where $K(u, v) = u \cdot v + (u \cdot v)^2$

- It is sum of linear kernel & Quadratic kernel so, it will be in Quadratic shape

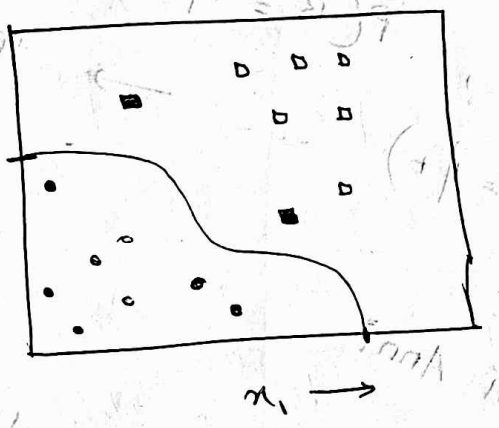


④ $$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

such that $\sum_{i=1}^{n} \alpha_i y_i = 0$; $\alpha_i \geq 0$, $\forall i = 1, \ldots n$

where $K(u,v) = \exp\left(-\frac{\|u-v\|^2}{2}\right)$

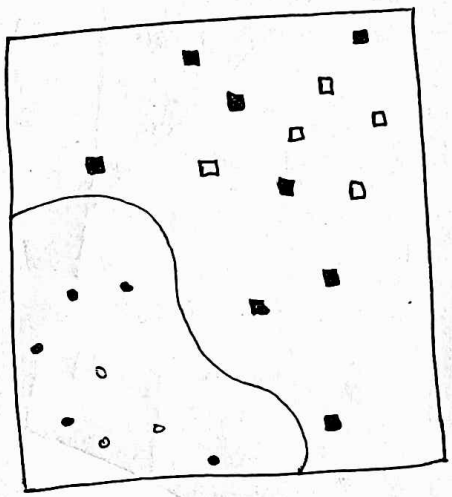
- Gaussian RBF kernel, $K(u,v) = \exp(-\gamma \|u-v\|^2)$, a smaller γ results a decision boundary with smoother curvature.



⑤ $$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

such that, $\sum_{i=1}^{n} \alpha_i y_i = 0$; $\alpha_i \geq 0 \quad \forall i = 1, \ldots n$

where $K(u,v) = \exp\left(-\|u-v\|^2\right)$



- Larger γ will result in gaussian RBF kernel to have more distigushed curvature.

- A larger γ means Gaussian distribution has smaller variance & more distinguished curve.

- What is Bagging ? (decrease variance)

- An ensemble learning method that is commonly used to reduce variance within a noisy dataset. A random sample of data in a training set is selected with replacement.

- What is Boosting ? (decrease bias)

- Boosting creates a collection of predictors. It refers to a family of algorithms which converts weak learner to strong learner. It is an ensemble method for improving model predictions of given learning algorithm.

- What is pruning ?

- A data compression technique in ML that reduces the size of decision tree by removing sections of trees which are not required to classify instances

- What is pre-pruning ?

- Early stopping the growth of tree before it classify completely the training set.

- What is post-pruning ?

- Remove the sections of tree after it has completly grown.

- What is version space ?

- Intermediate space between general hypothesis space & specific hypothesis space.
  It is also hierarchial representation of knowledge.

- Disadvantages of Candidate Elimination Algo ?

- Fails for noisy / inconsistent data
- Uses partially learning, sometimes fails to predict right hypothesis for new training sample.

- Why we only consider particular rules for decision trees?

- A small change in data can cause a large change in the structure of decision tree causing unstability.

For Decision Trees, sometimes calculations can go for more complex compared to other algorithms.

Because of certain rules, decision trees requires less effort for data preparation during pre-processing.

- Applications of Machine Learning?

- Fraud detection
- Price prediction / Stock Markets Trading
- Products Recommendation
- Speech Recognition
- Self driving cars, Medical diagenosis

- What is Regression?

- A statistical method to predict continuous outcome based on one or more predictor variable values.

- What is Linear Regression?

- A supervised ML model in which it finds the best fit linear line between the independent and dependent variable.

- What is cost function?

- Cost function (J) of Linear Regression is the Root Mean Squared error between predicted y value & true y value.

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

- What is hypothesis of Linear Regression?

θ_1 - intercept

$h_\theta(x) = y = \theta_1 + \theta_2 \cdot x$ (θ_1, θ_2 are parameters that control the hypothesis y)

- What is gradient descent?

- To update θ_1 & θ_2 values in order to minimize cost function (i.e. minimizing RMSE value) and achieving best fit line the model uses Gradient Descent.

Here, the idea is to start with random θ_1, θ_2 values and then iteratively update them, reaching minimum cost.

- Why Logistic Regression needed?

- When our data has outliers, the best fit line predicted using linear regression may deviate and incorrectly predicts the output values.

- The output in linear regression is sometimes >1 or <0.

- What is sigmoid function?

- $F(z) = \frac{1}{1+e^{-z}}$; to prevent the output of linear regression (>1 or <0). This sigmoid function is an activation function that has range $(0, 1)$. Since we have to predict probability (lies between $(0, 1)$) this sigmoid function is right choice.

- What is Bias?

- The difference between what you expect to learn & truth.

- What is variance?

- The difference between what you expect to learn & what you learnt.

Small variance, high bias — underfitting (increase features)

High variance, small bias — overfitting (decision trees)

↓ Regularization

- What is Regularization ?

- Keep all features, but reduce parameter θ magnitude. It discourages learning more complex or flexible model, to prevent overfitting.

- Define Logistic Regression ?

- Regression used to fit a curve of data in which the dependent variable is Binary value or dichotomous

- What is Confusion Matrix ?

- A summarized table of no. of correct & incorrect predictions obtained from by a classifier or classification model. It is a performance measurement for ML algorithm.

Predicted values \ Actual values	1	0
1	TP	FP
0	FN	TN

TP - True Positive
FP - False Positive
FN - False Negative
TN - True Negative

TP - You predicted positive & it is True.

True Negative - You predicted negative & it is True.

False Negative - You predicted negative & it is False. (Type II error)

False Positive - You predicted positive & it is False. (Type I error)

$$\text{Precision} = \frac{TP}{TP + FP} \quad \text{(as high as possible)} \qquad \text{Accuracy} = \frac{TP + TN}{\text{Total}} \quad \text{(high as possible)}$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{(as high as possible)}$$

• What is K-NN ? (An instance based Learner)

- A supervised ML algorithm that is used to solve both classification & regression problems.

It works by finding the distances between a query instance & all the data (training examples) nearest to query & then takes K-nearest examples & votes for most frequent output.

• Why is KNN called lazy Algorithm ?

- Because it doesnot learn a discriminate function from training data rather memorizes the training data & stores it. to classify new training example.

• What is difference between KNN & weighted-KNN ?

- Weighted KNN is modified version of KNN. While taking majority vote, the nearest neighbors vary widely in their distance & ~~closely~~ closest neighbors more reliably indicate the class of object

• What is Support Vector Machine ?

- The objective of SVM algorithm is to find a hyperplane in an N-dimensional space (N - the no. of features) that distinctly classifies the data points.

• What is hyperplane ?

- In an n-dimensional has a flat, n-1 dimensional subset of that space that divides the space into two disconnected parts. called hyperplane

• What is margin ?

- The distance between the hyperplanes drawn to differentiate the different classes.

- What is dimension of Hyperplane?

– Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes.

The dimension of hyperplane depends upon the no. of features. If the no. of input features is 2, then hyperplane is just a line. If the no. of input features is 3, then the hyperplane becomes 2-D plane.

- What are support vectors?

– The data points that are closer to the hyperplane and influence the position & orientation of hyperplane.

Using these, we can maximize the margin of the classifier.

- Explain Large Margin Intuition?

– We take the output of the linear function and if that output is greater than 1, identified is in other class. The reinforcement range [-1, 1] acts as margin (also called threshold values)

- What is Hinge Loss?

– The Loss function that helps maximize the margin is hinge loss.

- What is Kernal-trick?

– A method of using linear classifier to solve a non-linear problem by mapping non-linear data into higher dimensional space. ~~without~~

- Why do we need Kernel trick?

- If the data is not linearly separable in 2-dimensional space, to build a linear classifier, we have to transform our data into 3D space while dealing with 2-D data.

- Types of Kernels?

1) Linear Kernel -
2) Gaussian Kernel -
3) Polynomial Kernel -
4) String Kernel -
5) Chi-square Kernel -

- What is Linear Kernel?

- Used when data is linearly separable, i.e. it can be separated using single line.

- What is Gaussian Kernel?

- Gaussian is one such Kernel that gives good linear seperation in higher dimension for many non linear problems.

- Logistic Regression (vs) SVM?

- n = no. of features, m = no. of training examples

1. n is large than m : Use logistic regression or SVM without linear kernel

2. n is small, m is intermediate : Use SVM with Gaussian kernel

3. n is small, m is large : Create/add more features, then use logistic regression or linear SVM.

- SVM multiclass classification ?

- Use one (vs) all method. Train K SVMs, one to distinguish $y = i$ from the rest, get $\theta^{(1)}, \theta^{(2)} \ldots \theta^{(k)}$

Then pick class i with the largest $\left[\theta^{(i)T} \cdot x\right]$

- SVM Regression ?

- Support Vector Regression is supervised learning algorithm that is used to predict discrete values. The basic idea behind SVR is to find best fit line.

In SVR, the best fit line is the hyperplane that has the maximum no. of points. Here, SVR tries to fit the best line within the distance between hyperplane & boundary line.

- Disadvantages of SVM ?

- 1) Not suitable for large data.
2) Doesnot fit for data with noise (target class are overlapping).
3) no. of features >>> no. of training samples, SVM fails
4) Long training time on datasets.
5) choosing a "good" Kernel is not easy.

- What is Radial Basis ?

$$K(x_1, x_2) = e^{\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right)}$$

{ Find a non linear classifier or regression line.

σ - variance of hyperparameter.

$\|x_1 - x_2\|$ - Euclidean distance between x_1, x_2 points.

MODULE - 4 :

*. NAIVE BAYES CLASSIFIER :

EX : Given Data below. Classify this New Instance (Color = Green, Legs = 2, Height = Tall, Smelly = No) using Naive Bayes classifier.

No	Color	Legs	Height	Smelly	Species
1	White	3	short	Yes	M
2	Green	2	Tall	No	M
3	Green	3	short	Yes	M
4	White	3	Short	Yes	M
5	Green	2	short	No	H
6	White	2	Tall	No	H
7	White	2	Tall	No	H
8	White	2	short	Yes	H

Sol :

$$P(M) = \frac{4}{8} = 0.5$$

$$P(H) = \frac{4}{8} = 0.5$$

Color	M	H
White	$\frac{2}{4}$	$\frac{3}{4}$
Green	$\frac{2}{4}$	$\frac{1}{4}$

Legs	M	H
2	$\frac{1}{4}$	$\frac{4}{4}$
3	$\frac{3}{4}$	$\frac{0}{4}$

Height	M	H
Tall	$\frac{3}{4}$	$\frac{2}{4}$
Short	$\frac{1}{4}$	$\frac{2}{4}$

Smelly	M	H
Yes	$\frac{3}{4}$	$\frac{1}{4}$
No	$\frac{1}{4}$	$\frac{3}{4}$

$$\vartheta'_{NB} = \underset{\vartheta_j \in \{yes, No\}}{\operatorname{argmax}} P(\vartheta_j) \prod_i P(a_i | \vartheta_j)$$

$$\vartheta'_{NB} = \underset{\vartheta_j \in \{yes, No\}}{\operatorname{argmax}} P(\vartheta_j)$$

$$\vartheta_{NB}(yes) = \frac{\vartheta'_{NB}(yes)}{\vartheta'_{NB}(yes) + \vartheta'_{NB}(No)}$$

$$\vartheta_{NB}(No) = \frac{\vartheta'_{NB}(No)}{\vartheta'_{NB}(yes) + \vartheta'_{NB}(No)}$$

$$P(M | \text{New Instance}) = P(M) * P(\text{Color} = \text{Green} | M) * P(\text{legs} = 2 | M) * P(\text{height} = \text{tall} | M) * P(\text{Smelly} = No | M)$$

$$P(M | \text{New Instance}) = 0.5 \times \frac{2}{4} \times \frac{1}{4} \times \frac{3}{4} \times \frac{1}{4} = 0.0117$$

$$P(H | \text{New Instance}) = P(H) * P(\text{Color} = \text{Green} | H) * P(\text{legs} = 2 | H) * P(\text{height} = \text{tall} | H) * P(\text{Smelly} = No | H)$$

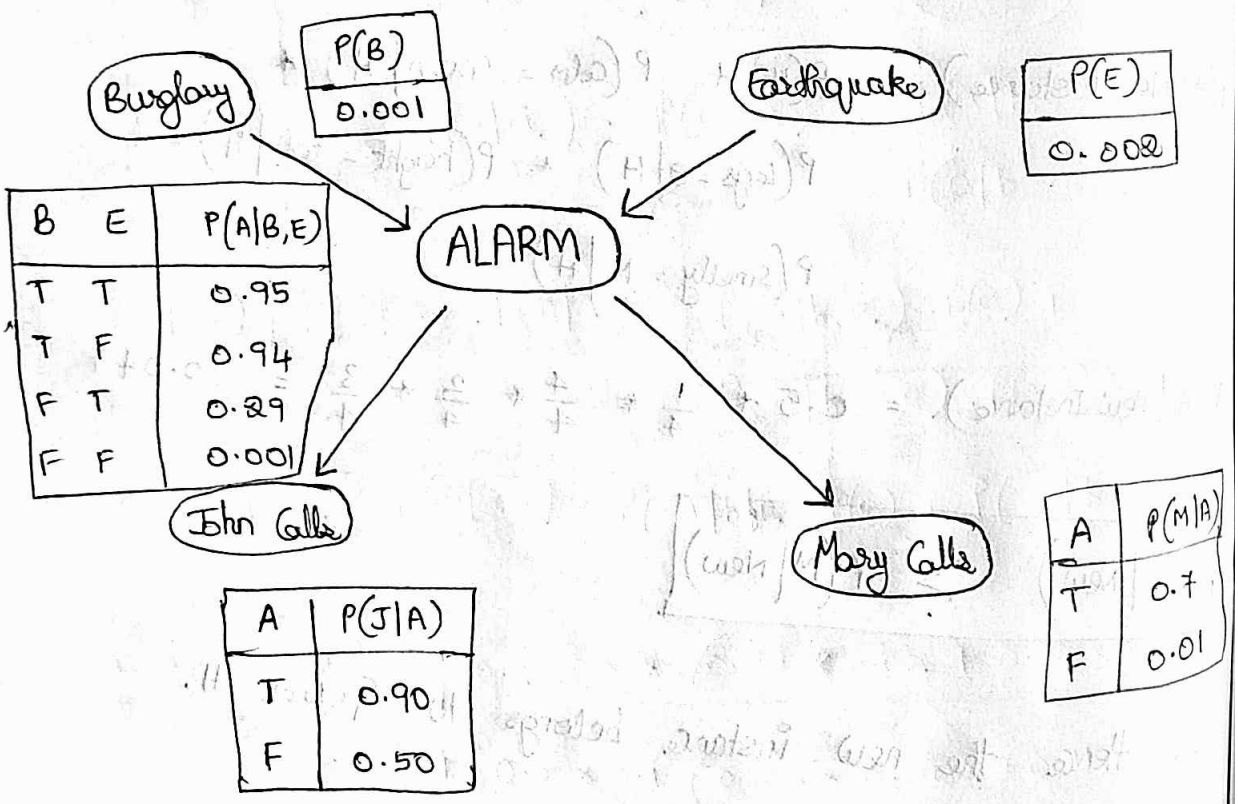
$$P(H | \text{New Instance}) = 0.5 * \frac{1}{4} * \frac{4}{4} * \frac{2}{4} * \frac{3}{4} = 0.047$$

$$\boxed{P(H | \text{New}) > P(M | \text{New})}$$

$\therefore$ Hence the new instance belongs to Species H.

*. BAYESIAN BELIEF NETWORKS :

- You have a new burglar alarm installed at home.
- It is fairly reliable at detecting burglary, but also sometimes responds to minor earthquakes.
- You have two neighbors, John & Mary, who promised to call you at work when they hear the alarm.
- John always calls when he hears the alarm, but sometimes confuses telephone ringing with the alarm and calls too.
- Mary likes loud music and sometimes misses the alarm. Given the evidence of who has or has not called, we would like to estimate the probability of a burglary.



① What is the probability that the alarm has sounded that but neither a burglary nor an earthquake has occured, and both John & Mary call?

Sol:

$$P(j \wedge m \wedge a \wedge \neg b \wedge \neg e) = P(j|a) \cdot P(m|a) \cdot P(a|\neg b, \neg e) \cdot P(\neg b) \cdot P(\neg e)$$

$$= 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998 = 0.00062$$

② What is the probability that John Call?

Sol:

$$P(j) = P(j|a) \cdot P(a) + P(j|\neg a) \cdot P(\neg a)$$

$$= P(j|a) \cdot \{P(a|b,e) * P(b,e) + P(a|\neg b, e) * P(\neg b, e) + P(a|b, \neg e) * P(b, \neg e) + P(a|\neg b, \neg e) * P(\neg b, \neg e)\}$$

$$+ P(j|\neg a) \cdot \{P(\neg a|b,e) * P(b,e) + P(\neg a|\neg b, e) * P(\neg b, e) + P(\neg a|b, \neg e) * P(b, \neg e) + P(\neg a|\neg b, \neg e) * P(\neg b, \neg e)\}$$

$$= 0.90 * 0.00252 + 0.05 * 0.9974 = 0.0521$$

③ What is the probability that there is a burglary given that John and Mary calls?

Sol:

Suppose we have given for the evidence variables $E_1, E_2, \ldots E_m$ their values $e_1, \ldots e_m$, and we want

to predict whether the query variable X has the value x or not.

For this, we compute and compare the following:

$$P(x \mid e_1, \ldots e_m) = \frac{P(x, e_1 \ldots e_m)}{P(e_1, \ldots e_m)} = \alpha \cdot P(x, e_1 \ldots$$

$$P(\neg x \mid e_1 \ldots e_m) = \frac{P(\neg x, e_1 \ldots e_m)}{P(e_1, \ldots e_m)} = \alpha \cdot P(\neg x, e_1 \ldots$$

$$\Rightarrow \boxed{\alpha = \frac{1}{P(x, e_1 \ldots e_m) + P(\neg x, e_1 \ldots e_m)}} \quad *$$

$$P(b \mid j, m) = \alpha \cdot P(b) \sum_a P(j \mid a) \cdot P(m \mid a) \sum_e P(a \mid b, e) \cdot P(e)$$

$$= \alpha \cdot P(b) \sum_a P(j \mid a) \cdot P(m \mid a) \cdot \{ P(a \mid b, e) P(e) + P(a \mid b, \neg e) \cdot P(\neg e) \}$$

$$= \alpha \cdot P(b) \left[P(j \mid a) \cdot P(m \mid a) \{ P(a \mid b, e) \cdot P(e) + P(a \mid b, \neg e) \cdot P(\neg e) \} \right.$$

$$\left. + P(j \mid \neg a) \cdot P(m \mid \neg a) \{ P(\neg a \mid b, e) P(e) + P(\neg a \mid b, e) P(\neg e) \} \right]$$

$$= \alpha * (0.001) * [0.9 \times 0.7 \times (0.95 * 0.02 + 0.94 \times 0.998) + 0.5 \times 0.1 (0.05 \times 0.002 + 0.71 \times 0.998)]$$

$$= \alpha * 0.00059$$

Similarly,

$$P(\sim b|j,m) = \alpha \cdot P(\sim b) \sum_a P(j|a) \cdot P(m|a) \sum_e P(a|\sim b,e) P(e)$$

$$= \alpha \cdot P(\sim b) \sum_a P(j|a) \cdot P(m|a) \{ P(a|\sim b, e) \cdot P(e) + P(a|\sim b, \sim e) \cdot P(\sim e) \}$$

$$= \alpha \cdot P(\sim b) [P(j|a) P(m|a) \{ P(a|\sim b, e) \cdot P(e) + P(a|\sim b) \cdot P(\sim e) \} + P(j|\sim a) \cdot P(m|\sim a) \{ P(\sim a|\sim b, e) \cdot P(e) + P(\sim a|\sim b, \sim e) \cdot P(\sim e) \}]$$

$$= \alpha * 0.999 * \{ (0.9 \times 0.7 (0.29 \times 0.02 + 0.01 \times 0.998) + 0.05 * 0.01 (0.71 \times 0.02 + 0.999 \times 0.998) \}$$

$$= \alpha * 0.0015$$

We got, $P(b|j,m) = \alpha \cdot P(b,j,m)$

$P(\sim b|j,m) = \alpha \cdot P(\sim b, j, m)$

And, $P(b,j,m) = 0.00059$

$P(\sim b, j, m) = 0.0015$

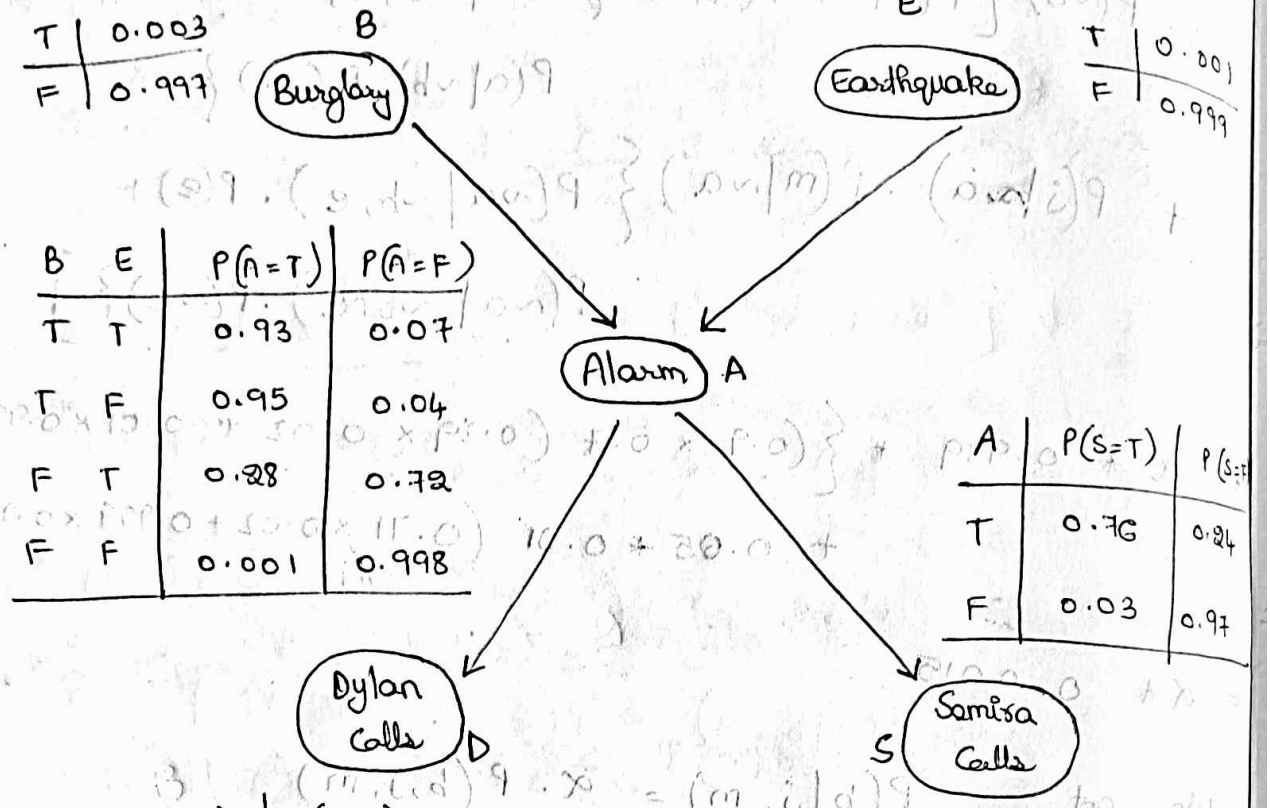
We know that, $\alpha = \frac{1}{P(b,j,m) + P(\sim b, j, m)} = \frac{1}{(0.00059 + 0.0015)}$

$$\alpha = \frac{1}{0.00209} = 478.4688 \approx 478.5$$

$\rightarrow P(b|j,m) = 478.5 \times 0.00059 = 0.28$

$P(\neg b|j,m) = 478.5 \times 0.0015 = 0.72$

Example 2 : (Consider same Burglar Alarm case before)



T	0.003
F	0.997

T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.93	0.07
T	F	0.95	0.04
F	T	0.28	0.72
F	F	0.001	0.998

A	P(S=T)	P(S=F)
T	0.76	0.24
F	0.03	0.97

A	P(D=T)	P(D=F)
T	0.92	0.08
F	0.04	0.96

Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occured, and Dylan & Samira both called Munish.

Sol : Given Probabilities are :

$P(B = True) = 0.003$ $\quad P(E = True) = 0.001$

$P(B = False) = 0.997$ $\quad P(E = False) = 0.999$

- Probability of Alarm : (A)

Burglary (B)	Earthquake (E)	P(A=T)	P(A=F)
T	T	0.93	0.07
T	F	0.95	0.04
F	T	0.28	0.72
F	F	0.001	0.998

- Probability of Dylan Calls : (D)

Alarm (A)	P(D = True)	~D (D = False)
T	0.92	0.08
F	0.04	0.96

- Probability of Samira Calls : (S)

Alarm (A)	P(S = True)	~S (S = False)
T	0.76	0.24
F	0.03	0.97

Now,

Alarm sounded	⇒	A = True ⇒ P(A = F)
No Burglary	⇒	B = False ⇒ P(B = F)
No Earthquake	⇒	E = False ⇒ P(E = F)
Dylan called	⇒	D = True ⇒ P(D = T)
Samira called	⇒	S = True ⇒ P(S = T)

Now $P(A, \sim B, \sim E, D, S) = ?$

$\Rightarrow = P(D|A) \cdot P(S|A) \cdot P(A|\sim B, \sim E) \cdot P(\sim B) \cdot P(\quad$

$= 0.92 \times 0.76 \times 0.001 \times 0.997 \times 0.999$

$= 0.0006964 \approx \underline{\underline{0.0007}}$.

*. In an oximeter manufacturing factory, Wing A, B, C produce 25%, 36% and 42% oximeters respectively. Out of total 5.2%, 3.5%, 2.5% are defective ones. An oximeter is drawn at random from product. If oximeter drawn is found to be defective. What is the probability it is manufactured by wing B machine.

Sol: Let A_1 be event that oximeter manufactured by Wing A. & B_2 for Wing B, C_3 be for Wing C

$P(B|E) = ?$

E : Defective oximeter

$B|E$: Defective oximeter given that is manufactured by Wing B.

Given,

$P(A) = 25\% = 0.25$

$P(B) = 36\% = 0.36$

$P(C) = 42\% = 0.42$

$P(E|A_1) = 0.052$

$P(E|B) = 0.035$

$P(E|C) = 0.025$

$$P(B|E) = \frac{P(B) \times P(E|B)}{P(A) \times P(E|A) + P(B) \times P(E|B) + P(C) \times P(E|C)}$$

$$= \frac{0.36 \times 0.035}{(0.25 \times 0.052) + (0.36 \times 0.035) + (0.42 \times 0.025)} = \frac{126}{361} = \underline{\underline{0.349}}$$

(*) Rishav has joined a new company and his age is 48 years and he is getting a salary amount of Rs. 142 000/-. In that company, the salary amount is labeled as High (H) or Low (L) based on the employee's age as shown below.

S.No. (ID)	Employee Name	Age	Salary	Salary Type (High/Low)
1	Amit	33	150000	H
2	Bikash	48	220000	H
3	Seema	60	100000	H
4	Pratik	40	62000	L
5	Arnab	23	95000	H
6	Ridhi	52	18000	L
7	Biju	35	120000	H
8	Rakesh	20	20000	L
9	Kiran	45	80000	H
10	Jhuma	35	60000	L
11	Piyus	25	40000	L

Use K-NN algorithm and find the Rishav's Salary label for K=5 & K=7 ?

New Instance ⇒ (12, Rishav, 48, 142000, ?)

For ID1 :

Age $\Rightarrow (33-48)^2$

Salary $\Rightarrow (150000-142000)^2$

$\Rightarrow d = \sqrt{225+64000000}$

$d = 8000.014069$

For ID2 :

Age $\Rightarrow (48-48)^2$

Salary $\Rightarrow (220000-142000)^2$

$\Rightarrow d = \sqrt{0+(78000)^2}$

$d = 78000$

For ID3 :

Age $\Rightarrow (60-48)^2$

Salary $\Rightarrow (100000-142000)^2$

$\Rightarrow d = \sqrt{144+1764000000}$

$d = 42000.0017$

For ID4 :

Age $\Rightarrow (40-48)^2$

Salary $\Rightarrow (62000-142000)^2$

$\Rightarrow d = \sqrt{64+6400000000}$

$d = 80000.0004$

For ID5 :

Age $\Rightarrow (23-48)^2$

Salary $\Rightarrow (95000-142000)^2$

$\Rightarrow d = \sqrt{625+2209000000}$

$d = 47000.00665$

For ID6 :

Age $\Rightarrow (58-48)^2$

Salary $\Rightarrow (18000-142000)^2$

$\Rightarrow d = \sqrt{16+15376000000}$

$d = 124000.0001$

For ID 7:

Age ⇒ $(35-48)^2$

Salary ⇒ $(120000-142000)^2$

$\Rightarrow d = \sqrt{169 + 169000000}$

$d = 13000.0065$

For ID 8:

Age ⇒ $(20-48)^2$

Salary ⇒ $(80000-142000)^2$

$\Rightarrow d = \sqrt{784 + 14884000000}$

$d = 122000.0032$

For ID 9:

Age ⇒ $(45-48)^2$

Salary ⇒ $(80000-142000)^2$

$\Rightarrow d = \sqrt{9 + 3844000000}$

$d = 62000.00007$

For ID 10:

Age ⇒ $(35-48)^2$

Salary ⇒ $(60000-142000)^2$

$\Rightarrow d = \sqrt{169 + 6724000000}$

$d = 82000.00103$

For ID 11:

Age ⇒ $(25-48)^2$

Salary ⇒ $(40000-142000)^2$

$d = \sqrt{529 + 10404000000}$

$d = 102000.0026$

ID	Euclidean distance	Rank	output
1	8000.014062	1	H
2	78000	6	H
3	42000.00171	3	H
4	80000.0004	7	L
5	47000.00665	4	H
6	124000.0001	11	L
7	13000.0065	2	H
8	122000.0032	10	L
9	62000.00007	5	H
10	82000.00103	8	L
11	102000.0026	9	L

i. For $K=5 \Rightarrow \{ID = 1, 7, 3, 5, 9\}$

Corresponding outputs are H, H, H, H, H

Majority vote = H for $K=5$

∴ New Query Instance : (12, Rishav, 48, 142000, H)

So, Rishav gets High Salary.

ii. For $K=7 \rightarrow \{ID = 1, 7, 3, 5, 9, 2, 4\}$

Corresponding outputs are H, H, H, H, H, H, L

Majority vote = H for $K=7$

∴ New Query Instance : (12, Rishav, 48, 142000, H)

So, Rishav gets High Salary.

* EM Algorithm : (Expectation - Maximization)

- used to find latent variable (not directly derivable)
- basic for many unsupervised clustering algorithm.

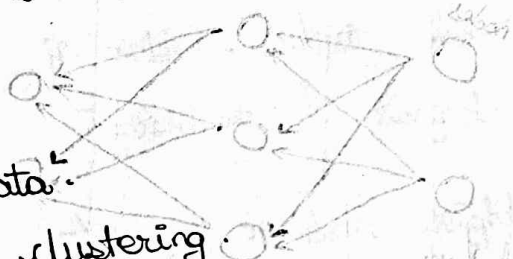
⊙ steps involved in EM-algorithm:

① Initially, a set of initial values are considered. A set of incomplete data is given to system.

② Next step - expectation step(E-step). Here we use observed data to estimate or guess the values of missing / incomplete data.

③ Maximisation step or M-step. Here we use the complete data generated in preceeding e-step to update the values.

④ We check if values are converging or not.
If converging - stop
Else, Repeat step 2 & step 3 till the convergence occurs.



⊙ USAGE:

1) used to fill missing data.
2) used for unsupervised clustering.
3) used to discover values of latent variables.

⊙ Advantages:

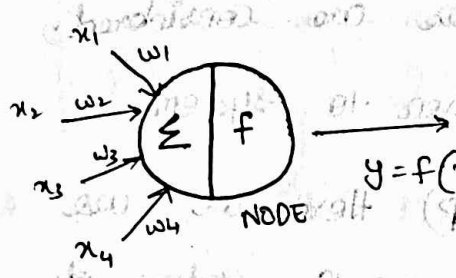
+ With each iteration, likelihood increases.
+ E-step & M-step are easy to implement.

⊙ Disadvantages:

- slow convergence.
- makes convergence to local optimal only.

MODULE - 5

- Human Brain - millions of neurons.
- Biological - Neurons, Formal term → Nodes.
- Natural functions are done artificially.



x_i : Input signal

w_i : weight of corresponding input signal

Σ : Summation function

f : Activation function

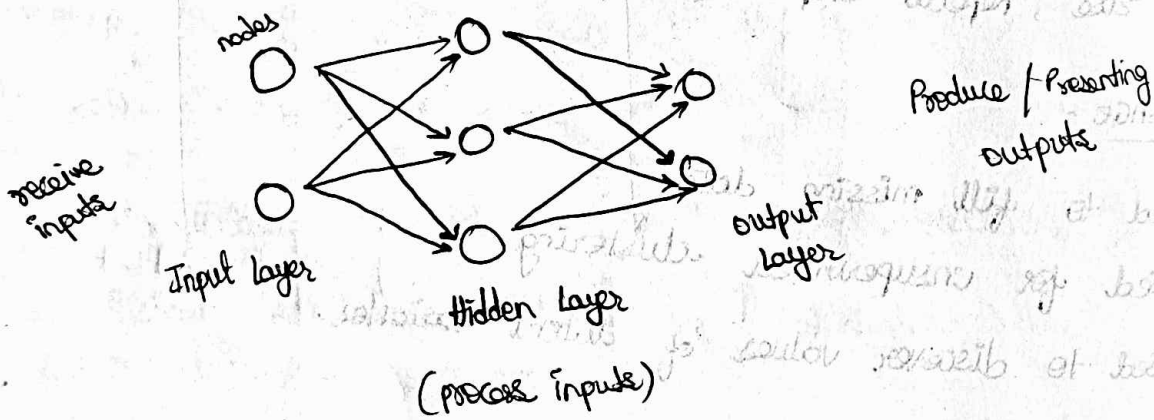
y : output function

- A node has 2 parts :

1. Summation : $x_1 w_1 + x_2 w_2 + x_3 w_3$
2. Activation : Generate output based on input. $\Sigma x_i w_i$

- Representation of Artificial Neural Networks : (3 parts)

1. Input layer
2. Hidden layer
3. output layer



- Appropriate problems for Neural networks Learning :

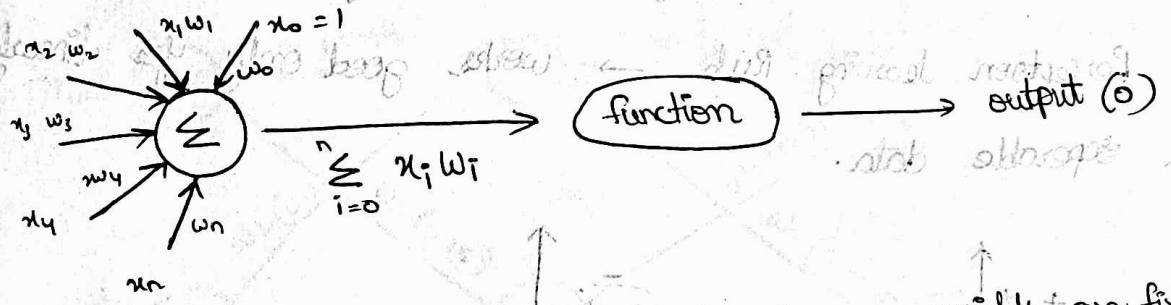
1) Instances have many attribute value pairs.
2) Target function has discrete values, continuous values or combination of both.

3. Training examples with errors or missing values.
4. Long training times are acceptable.
5. Fast evaluation of the target function learnt.
6. Ability for humans to understand the target function learnt by machine is not important.

*. PERCEPTION :

- basic unit used to build ANN.
- takes real valued input, calculates linear combination of these inputs and generates output.

output = 1 if the result $\geq$ threshold
= -1 otherwise result $\leq$ threshold



o : actual output
t : target output

if actual = target → weights are fixed
otherwise → changed.

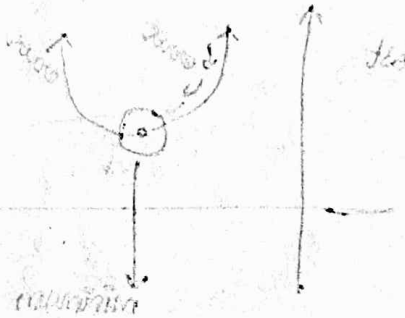
How to change weights ? : Initially, random weights later, keep on applying iterations and check if (o = t).

Formula :

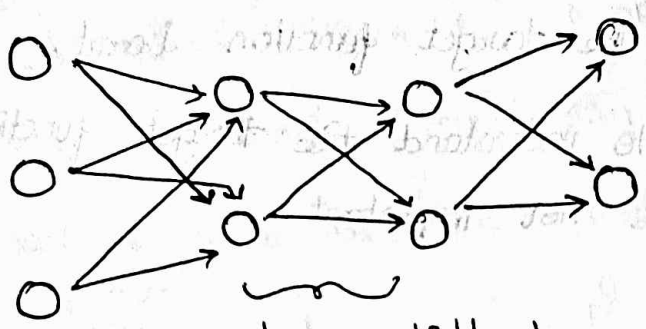
$$* \begin{bmatrix} w_i \leftarrow w_i + \Delta w_i \\ \Delta w_i = \eta (t - o) x_i \end{bmatrix} *$$

η : learning rate
t : target output
o : actual output

x_i : input associated with w_i



* . MULTI LAYER NEURAL NETWORKS :



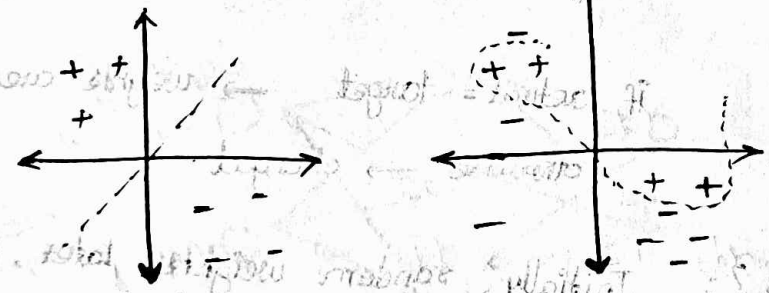
has 2 hidden layers

- Linearly separable data is for single layer Neural Network.
- Multilayer NN works for all type of data.

- output is not matching → go back & modify weights.
- Back Propogation is possible.

* . GRADIENT DESCENT AND DELTA RULE :

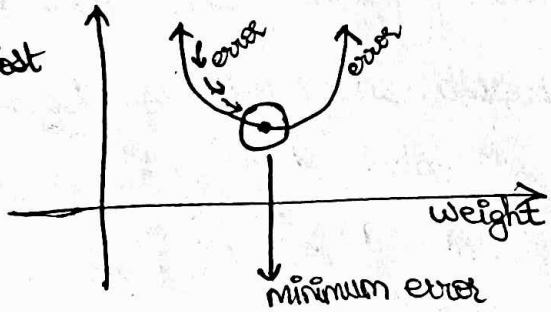
- Delta Rule → used for non-linearly separable data
- Perceptron learning Rule → works good only for linearly separable data.



Linearly Separable

Non-linearly Seperable

- Main idea behind delta rule is, it uses gradient descent rule and finds out the best weight.



Modifing weights :

$$w_i = w_i + \Delta w_i$$

$$\Delta w_i = -\eta \, \Delta E(w)$$

Here, $\Delta E(w)$ is derivative of error w.r.t weights. Also called as gradient.

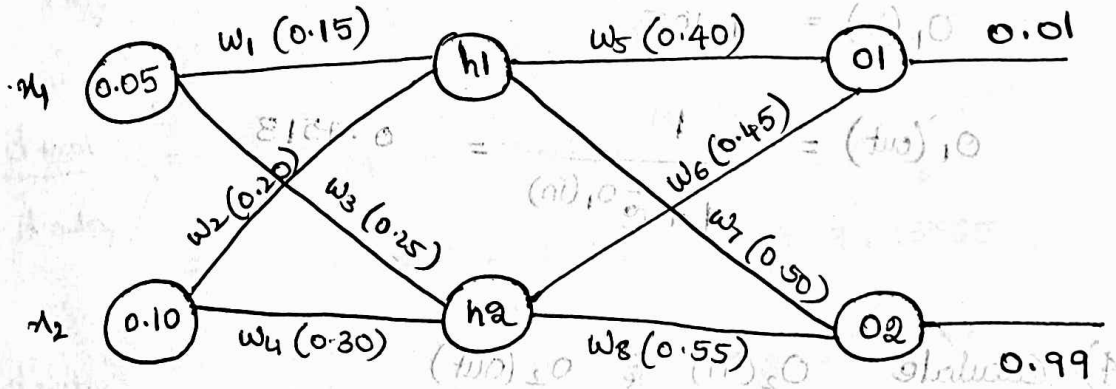
$$\Delta w_i = +\eta \sum (t_d - o_d) x_d$$

*. BACK PROPAGATION ALGORITHM :

- When error occurs, we go in backward direction & fix the error. output → hidden → input layer

Part 1: Calculate forward Propagation (error)

Ex:



$b_1 = 0.35$, $b_2 = 0.60$ (Bias Factors)

1) Calculate h1 (in & out)

$$h_1(in) = w_1 x_1 + w_2 x_2 + b_1$$

$$= (0.15 \times 0.005) + (0.20 \times 0.10) + 0.35$$

$$= 0.377$$

$$h_1(out) = \frac{1}{1 + e^{-h_1(in)}} = \frac{1}{1 + e^{-0.377}} = 0.5932$$

2) Calculate $h_2(in)$ & $h_2(out)$

$$h_2(in) = x_1 w_3 + x_2 w_4 + b_1$$

$$= (0.05 \times 0.25) + (0.1 \times 0.3) + 0.35$$

$$h_2(in) = 0.3925$$

$$h_2(out) = \frac{1}{1+e^{-h_2(in)}} = \frac{1}{1+e^{-0.3925}} = 0.5968$$

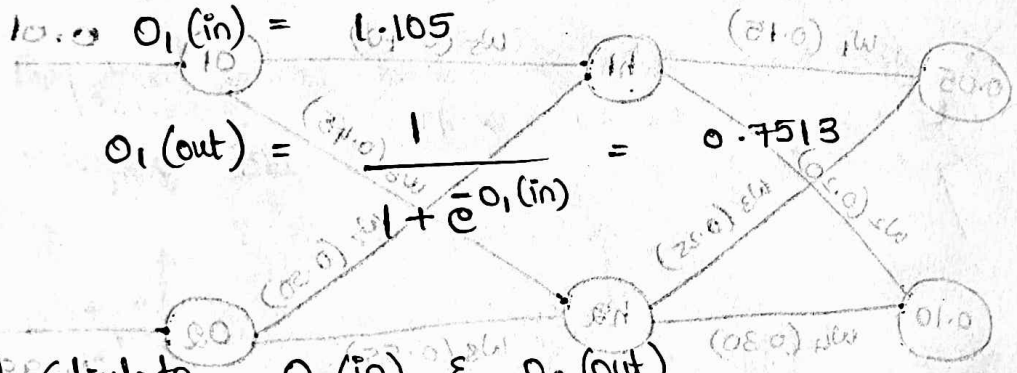
3) Calculate $O_1(in)$ & $O_1(out)$

$$O_1(in) = h_1(out) * w_5 + h_2(out)\, w_6 + b_2$$

$$= (0.593 \times 0.4) + (0.596 \times 0.45) * 0.6$$

$$O_1(in) = 1.105$$

$$O_1(out) = \frac{1}{1+e^{-O_1(in)}} = 0.7513$$



4) Calculate $O_2(in)$ & $O_2(out)$

$$O_2(in) = h_1(out) \times w_7 + h_2(out) \times w_8 + b_2$$

$$= (0.593 \times 0.5) + (0.5968 \times 0.45) + 0.6$$

$$O_2(in) = 1.22484$$

$$O_2(out) = \frac{1}{1+e^{-O_2(in)}} = \frac{1}{1+e^{-1.2248}} = 0.7729$$

5) Calculate E_{Total}

$$E_{Total} = \sum \frac{1}{2} (target - output)^2$$

$$E_{total} = E_{o1} + E_{o2}$$

$$= \frac{1}{2}(0.01 - 0.7513)^2 + \frac{1}{2}(0.99 - 0.7729)^2$$

$$= 0.274 + 0.0235 = 0.29837$$

Part 2 : Calculating Backward Propagation Error
(Output Layer → Hidden Layer)

weights are w_5, w_6, w_7, w_8

$$w_5^* = w_5 - \eta \frac{dE_{total}}{dw_5}$$

$$\frac{dE_{total}}{dw_5} = \frac{dE_{total}}{dout_{o1}} \times \frac{dout_{o1}}{dnet_{o1}} \times \frac{dnet_{o1}}{dw_5}$$

$$\frac{d_{total}}{d\,out_{o1}} = Out_{o1} - target_{o1}$$

$$= 0.751365 - 0.01 = 0.7413565$$

$$\frac{dout_{o1}}{dnet_{o1}} = out_{o1}(1 - out_{o1})$$

$$= 0.751365(1 - 0.751365) = 0.186815602$$

$$\frac{dnet_{o1}}{dw_5} = out_{h1} = 0.59326992 = h_1(out)$$

$$\frac{dE_{total}}{dw_5} = 0.7413565 \times 0.186815602 \times 0.59326992$$

$$= 0.08216704$$

$$w_5^* = w_5 - \eta \frac{dE_{total}}{dw_5} = 0.4 - 0.6 \times 0.08216704$$

$$= 0.35069776$$

$$w_8^* = w_8 - \eta \frac{d E_{total}}{d w_8}$$

$$\frac{d E_{total}}{d w_8} = \frac{d E_{total}}{d out_{o_2}} \times \frac{d out_{o_2}}{d net_{o_2}} \times \frac{d net_{o_2}}{d w_8}$$

$$\frac{d E_{total}}{d out_{o_2}} = out_{o_2} - target_{o_2}$$

$$= 0.7729 - 0.99 = -0.2171$$

$$\frac{d out_{o_2}}{d net_{o_8}} = out_{o_2}(1 - out_{o_2}) = 0.7729(1 - 0.7729)$$

$$= 0.17552559$$

$$\frac{d net_{o_2}}{d w_8} = out_{h_2} = 0.5968$$

$$\frac{d E_{total}}{d w_8} = -0.2171 \times 0.17552559 \times 0.5968$$

$$= -0.022742$$

$$w_8^* = 0.55 - 0.6 \times (-0.022742) = 0.563645$$

Similarly Calculating :

$w_6^* =$ $\qquad$ $w_7^* =$

Part 3 : Calculating Backward Propogation error

$\{w_1\ w_2\ w_3\ w_4\}$ (hidden layer ⟶ input layer)

$$w_1^* = w_1 - \eta \frac{\partial E_{total}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h_1}} \times \frac{\partial out_{h_1}}{\partial net_{h_1}} \times \frac{\partial net_{h_1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial out_{h_1}} = \frac{\partial E_{o_1}}{\partial out_{h_1}} + \frac{\partial E_{o_2}}{\partial out_{h_1}}$$

$$\frac{dE_{o_1}}{dnet_{o_1}} \times \frac{dnet_{o_1}}{dout_{h_1}} \qquad \frac{dE_{o_2}}{dnet_{o_2}} \times \frac{dnet_{o_2}}{dout_{h_1}}$$

$$\frac{dE_{o_1}}{dout_{o_1}} \times \frac{dout_{o_1}}{dnet_{o_1}} \qquad \frac{dE_{o_2}}{dout_{o_2}} \times \frac{dout_{o_2}}{dnet_{o_2}}$$

$$\frac{\partial E_{o_2}}{\partial out_{o_2}} = out_{o_2} - target_{o_2}$$

$$= 0.7729284165 - 0.99$$

$$= -0.2170715835$$

$$\frac{\partial out_{o_2}}{\partial net_{o_2}} = out_{o_2}(1 - out_{o_2})$$

$$= 0.7729(1 - 0.7729) = 0.175510052$$

$$\frac{\partial net_{o_1}}{\partial out_{h_1}} = \text{on } o_1 \text{ from } h_1 = w_5 = 0.4$$

$$\frac{dE_{o_2}}{\partial net_{o_2}} = -0.21707 \times 0.1755 = -0.038$$

$$\frac{\partial net_{o_2}}{\partial out_{h_1}} = \text{on } o_2 \text{ from } h_1 = w_7 = 0.5$$

$$\frac{\partial E_{o_2}}{\partial out_{h_1}} = -0.038 * 0.5 = -0.01904$$

$$\frac{\partial E_{o_1}}{\partial net_{o_1}} = 0.1384\,9856, \qquad \frac{\partial E_{o_1}}{\partial out_{h_1}} = 0.1384 \times 0.4$$

$$= 0.0554$$

$$\frac{\partial E_{total}}{\partial out_{h_1}} = 0.05539 + (-0.01904)$$

$$= 0.036350306$$

$$\frac{\partial out_{h_1}}{\partial net_{h_1}} = out_{h_1}(1 - out_{h_1})$$

$$= 0.241300709$$

$$\frac{\partial net_{h_1}}{\partial w_1} = \frac{\partial}{\partial w_1}(w_1 x_1 + w_2 x_2 + b_1)$$

$$= x_1 = 0.05$$

$$\frac{\partial E_{total}}{\partial w_1} = 0.036350306 \times 0.241300709 \times 0.05$$

$$= 0.00438568$$

$$w_1^* = w_1 - \eta \frac{\partial E_{total}}{\partial w_1} = 0.15 - (0.6) \times 0.00438568$$

$$= 0.149736859 2$$

Similarly,

$w_2^* =$ $w_4^* =$

$w_3^* =$

Repeat this all 3 parts until you get minimum error.

- Remarks on Backpropagation Algorithm :

1. Convergence & local minima
2. Representational power of feed-forward networks
3. Hypothesis Space Search and inductive Bias

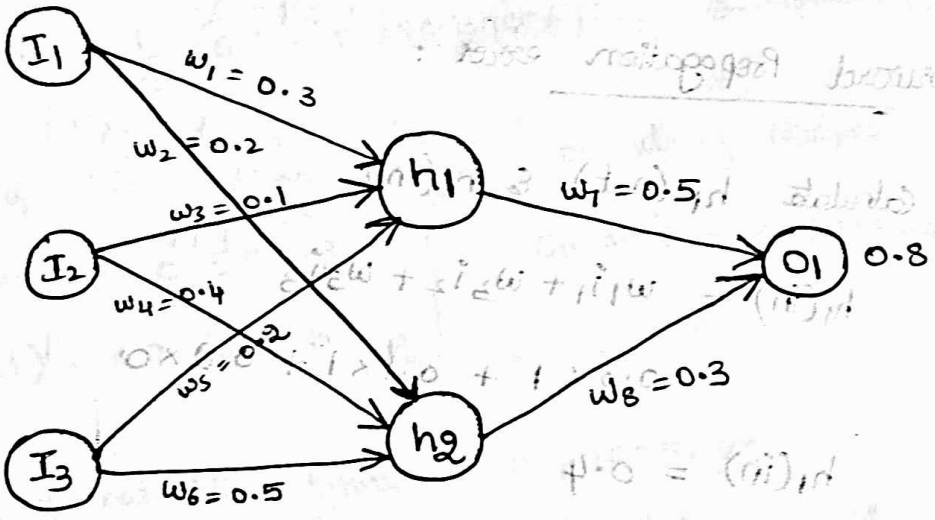
4. Hidden layer Representation

5. Generalization, overfitting & stopping criteria.

functions : Boolean, Continuous, Arbitary

Ex: The following is a network where the output of each neuron in hidden and output layer is computed by applying the hidden layer and output layer neurons for the given :

1. Compute the output of hidden layer and output layer neurons for the given input (1,1,0)
2. What is the error of this network with the given weights.
3. Update the weights of w7, w8 by applying the backpropogation algorithm.



Sol:

- It computes the gradient of the loss function for a single weight by the chain rule.

1. Inputs x, arrive through the preconnected graph-path.
2. Input is modeled using real weights W. The weights

are usually randomly selected

3. Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.

4. Calculate the error in the outputs.

5. Travel back from output layer to the hidden layer to adjust the weights such that error is minimized.

6. Keep repeating this process until the desired output is achieved.

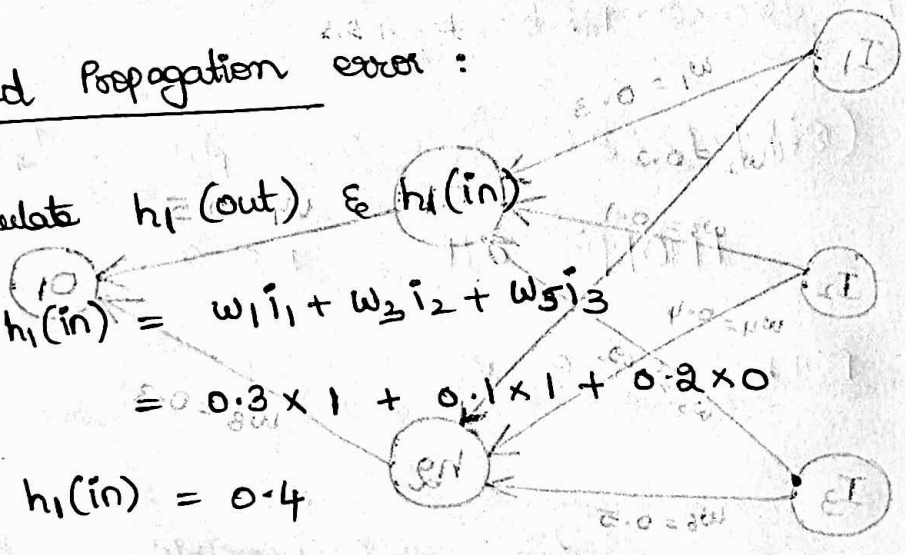
$w_1 = 0.3$ $\quad w_5 = 0.2$ $\quad i_1 = 1$

$w_2 = 0.2$ $\quad w_6 = 0.5$ $\quad i_2 = 1$

$w_3 = 0.1$ $\quad w_7 = 0.5$ $\quad i_3 = 0$

$w_4 = 0.4$ $\quad w_8 = 0.3$ $\quad o_1 = 0.8$

Forward Propagation error :



1) Calculate $h_1(out)$ & $h_1(in)$

$$h_1(in) = w_1 i_1 + w_3 i_2 + w_5 i_3$$

$$= 0.3 \times 1 + 0.1 \times 1 + 0.2 \times 0$$

$$h_1(in) = 0.4$$

$$h_1(out) = \frac{1}{1+e^{-h_1(in)}} = \frac{1}{1+e^{-0.4}} = 0.5986876601$$

2) $h_2(in) = w_2 i_1 + w_4 i_2 + w_6 i_3$

$= 0.2 \times 1 + 0.4 \times 1 + 0.5 \times 0$

$h_2(in) = 0.6$

$h_2(out) = \frac{1}{1+e^{-h_2(in)}} = \frac{1}{1+e^{-0.6}} = 0.6456563062$

3) Calculate $O_1(in)$ & $O_1(out)$

$O_1(in) = h_1(out) * w_7 + h_2(out) * w_8$

$= 0.5986876601 \times 0.5 + 0.6456563062 \times 0.3$

$= 0.2993438301 + 0.1936968919$

$O_1(in) = 0.4930407220$

$O_1(out) = \frac{1}{1+e^{-0.4930407220}} = 0.6208224$

4) Calculate E_{total}

$E_{total} = \sum \frac{1}{2}(target - output)^2 = E_{O1}$

$E_{O1} = \frac{1}{2}(0.8 - 0.6208224)^2 = 0.01605229064$

① output of hidden layer:

$h_1(out) = 0.598688$

$h_2(out) = 0.645656$

output of Neurons:

$O_1(out) = 0.620822$

② Error of network, $E_{total} = 0.01605229$

③ Updating weights w_7, w_8 using Backpropogation:

$w_7^* = w_7 - \eta \frac{dE_{total}}{dw_7}$ η: learning rate $= 0.5$ (assume)

$w_8^* = w_7 - \eta \frac{dE_{total}}{dw_8}$

$$w_7^* = w_7 - 0.5 \frac{dE_{total}}{dw_7}$$

$$\frac{dE_{total}}{dw_7} = \frac{dE_{total}}{dout_{o1}} \times \frac{dout_{o1}}{dnet_{o1}} \times \frac{dnet_{o1}}{dw_7}$$

$$\frac{dE_{total}}{dout_{o1}} = out_{o1} - target_{o1}$$

$$= 0.620822 - 0.8 = -0.179178$$

$$\frac{dout_{o1}}{dnet_{o1}} = out_{o1}(1 - out_{o1})$$

$$= (0.620822)(1 - 0.620822) = 0.235402044316$$

$$\frac{dnet_{o1}}{dw_7} = out_{h1} = 0.598688$$

$$\frac{dE_{total}}{dw_7} = -0.179178 \times 0.235402044316 \times 0.598688$$

$$= -0.025251981822$$

$$w_7^* = w_7 - 0.5(-0.025251981822)$$

$$= 0.5 + 0.012625990911$$

$$w_7^* = 0.512625990911$$

$$w_8^* = w_8 - 0.5 \frac{\partial E_{total}}{\partial w_8}$$

$$\frac{\partial E_{total}}{\partial w_8} = \frac{\partial E_{total}}{\partial out_{o1}} \times \frac{dout_{o1}}{dnet_{o1}} \times \frac{dnet_{o1}}{dw_7}$$

$$\frac{dE_{total}}{dout_{o1}} = -0.179178 \qquad \frac{dout_{o1}}{dnet_{o1}} = 0.2354020443$$

$$\frac{dnet_{o1}}{dw_8} = out_{h_2} = 0.645656$$

$$\frac{dE_{total}}{dw_8} = -0.179178 \times 0.235402044316 \times 0.645656$$

$$= -0.02723305774$$

$$w_8^* = w_8 - (0.5)(-0.02723305774)$$

$$w_8^* = 0.3 + 0.01361652887 = 0.3136165287$$

updated weights are:

$$w_7^* = 0.5126259909 \approx 0.5126$$

$$w_8^* = 0.3136165287 \approx 0.3136$$

*. FACE RECOGNITION

- Face Recognition is a category of biometric software that maps individuals facial features and stores the data as a face print.
- Machine learning and Image processing are backbones of Face Recognition.

*. IMAGE PROCESSING:

- involves the process of Computer vision.
- Image Reading : Reads any image between the range of 0-255.
- For any image, there are 3 colors — RGB.
- OpenCV : Python Library to solve computer vision.

*. Machine Learning:

- Takes dataset as input and learn from that data
  - identifies the pattern (height, width, color, etc---)
  - ML does 3 major functions in Face Recognition.

  1) Deriving the Feature Vector
  2) Matching Algorithms
  3) Face Recognition operations
     - Face detection
     - Face Analysis
     - Image to data conversion.
     - Match Finding

- Face Recognition Softwares : DeepVision AI, SenseTime, Amazon Rekognition, faceFirst, TrueFace, Cognitec....

- **Applications :**

1) Hospital Security
2) Mobile Devices
3) Airline Industry
4) Banking sectors ....

*. **Advancements in Neural Networks :** (3 advanced Topics)

(1) **Alternative error functions :**

- Adding an extra penalty term to $E(w)$ can reduce the problem of overfitting.
- each weight is multiplied with a constant in each iteration. So, we can redefine E as

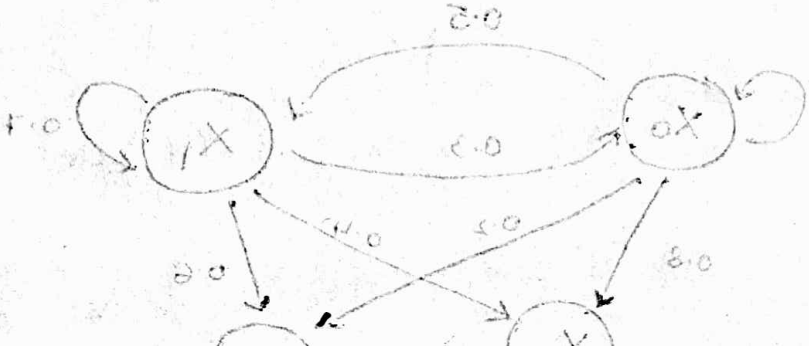
$$E(\bar{w}) = \frac{1}{2} \sum_{d \in D} \sum_{k \in K} (t_{kd} - o_{kd})^2 + \gamma \sum w_{ji}^2$$

(2) **Alternative error Minimization procedures :**

A) Line Search
B) Conjugate Gradient Descent method.

(3) **Recurrent Neural Networks (RNN) :**

- ANN that can be applied to time series data and uses the output of network units at time t, as input to other units at time $t+1$.



MODULE - 6

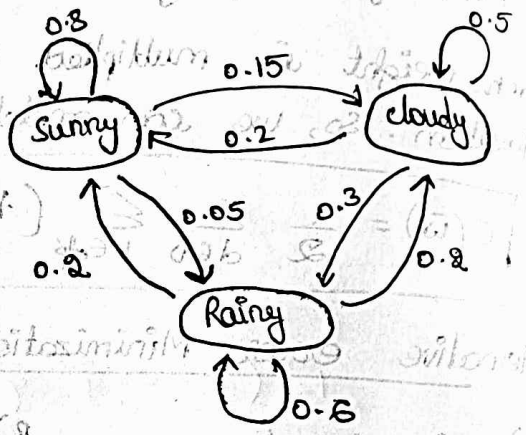
*. MARKOV MODEL :

- $P(q_1, q_2, \ldots q_n) = \prod_{i=1}^{n} P(q_i | q_{i-1})$

Ex: What is the probability of given sequence S, R, R, R, C, C

Initial probabilities given as $\pi = \{0.7, 0.25, 0.05\}$

Sol: $P(S) = 0.7$ (initial)



$P(R|S) = 0.05$

$P(R|R) = 0.6$

$P(C|R) = 0.2$

$P(C|C) = 0.5$

$P(\text{sequence}) = P(S, R, R, R, C, C)$

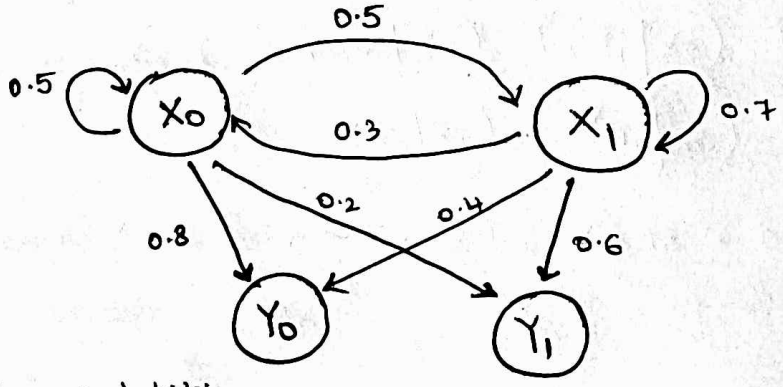
$= P(S) \times P(R|S) \times P(R|R) \times P(R|R) \times P(C|R) \times P(C|C)$

$= 0.7 \times 0.05 \times 0.6 \times 0.6 \times 0.2 \times 0.5$

$P(S, R, R, R, C, C) = 0.00126$

*. FORWARD ALGORITHM :

EX:



What is the probability of $\{Y_0, Y_0, Y_1\}$ sequence given the parameters of our HMM?

$$A = \begin{array}{c} \\ x_0 \\ x_1 \end{array} \begin{array}{c} \begin{array}{cc} x_0 & x_1 \end{array} \\ \begin{bmatrix} 0.5 & 0.5 \\ 0.3 & 0.7 \end{bmatrix} \end{array}, \quad B = \begin{array}{c} \\ x_0 \\ x_1 \end{array} \begin{array}{c} \begin{array}{cc} Y_0 & Y_1 \end{array} \\ \begin{bmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{bmatrix} \end{array}$$

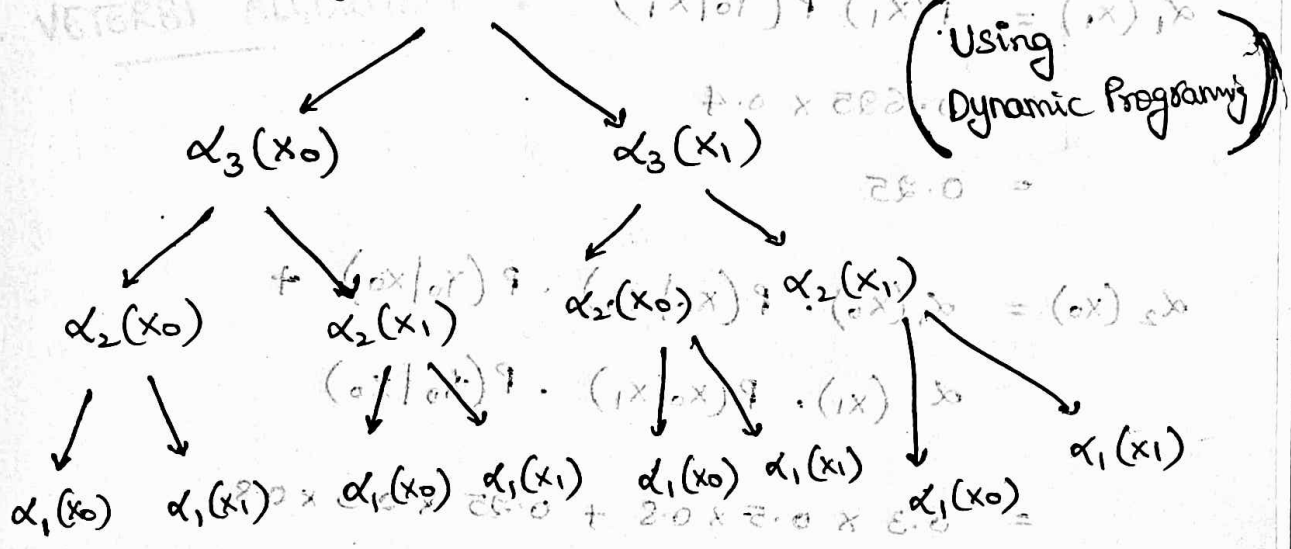
Transition Matrix

Emission Matrix

Stationary distribution for Hidden states : $\pi \cdot A = \pi \Rightarrow \pi = [0.375 \quad 0.625]$

Here no. of Hidden states =

$$P(Y = Y_0, Y_0, Y_1)$$



$$\alpha_1(x_0) = P(x_0) \cdot P(Y_0 | x_0)$$

$$\alpha_2(x_1) = P(x_1) \cdot P(Y_0 | x_1)$$

Formula :

$$\alpha_t(x_i) = \sum_{j=0}^{n-1} \alpha_{t-1}(x_j) \cdot P(x_i | x_j) \cdot P(Y_t | x_i)$$

For $j=0$, $n=1$

$$\alpha_t(x_i) = \alpha_{t-1}(x_0) \cdot P(x_i | x_0) \cdot P(Y^t | x_i)$$

Here, n is no. of Hidden states.

$$\alpha_1(x_i) = \pi[i]\, P(Y^0 \mid x_i)$$

$$P(Y^1 Y^2 \ldots Y^t) = \sum_{i=0}^{n-1} \alpha_{t-1}(x_i)$$

$$\alpha_1(x_0) = P(x_0) \cdot P(Y_0 \mid x_0)$$

$$= 0.375 \times 0.8$$

$$= 0.3$$

$$\alpha_1(x_1) = P(x_1)\, P(Y_0 \mid x_1)$$

$$= 0.625 \times 0.4$$

$$= 0.25$$

$$\alpha_2(x_0) = \alpha_1(x_0) \cdot P(x_0 \mid x_0) \cdot P(Y_0 \mid x_0) +$$
$$\alpha_1(x_1) \cdot P(x_0 \mid x_1) \cdot P(Y_0 \mid x_0)$$

$$= 0.3 \times 0.5 \times 0.8 + 0.25 \times 0.3 \times 0.8$$

$$= 0.18$$

$$\alpha_2(x_1) = \alpha_1(x_0)\, P(x_1 \mid x_0)\, P(Y_0 \mid x_1) +$$
$$\alpha_1(x_1) \cdot P(x_1 \mid x_1)\, P(Y_0 \mid x_1)$$

$$= (0.3 \times 0.5 \times 0.4) + (0.25 \times 0.7 \times 0.4)$$

$$\alpha_3(x_0) = \alpha_2(x_0)\, P(x_0 \mid x_0) \cdot P(Y_1 \mid x_0) +$$
$$\alpha_2(x_1) \cdot P(x_0 \mid x_1) \cdot P(Y_1 \mid x_0)$$

$$= (0.18 \times 0.5 \times 0.2) + (0.13 * 0.3 \times 0.2)$$

$$= 0.0258$$

$$\alpha_3(X_1) = \alpha_2(X_0) \cdot P(X_1|X_0) \cdot P(Y_1|X_1) + \alpha_2(X_1) \cdot P(X_1|X_1) \cdot P(Y_1|X_1)$$

$$= 0.18 \times 0.5 \times 0.6 + 0.13 \times 0.7 \times 0.6$$

$$= 0.1086$$

$$P(Y = Y_0, Y_0, Y_1) = \alpha_3(X_0) + \alpha_3(X_1)$$

$$= 0.0258 + 0.1086 = 0.1344$$

---

* VETERBI ALGORITHM :