Machine Learning

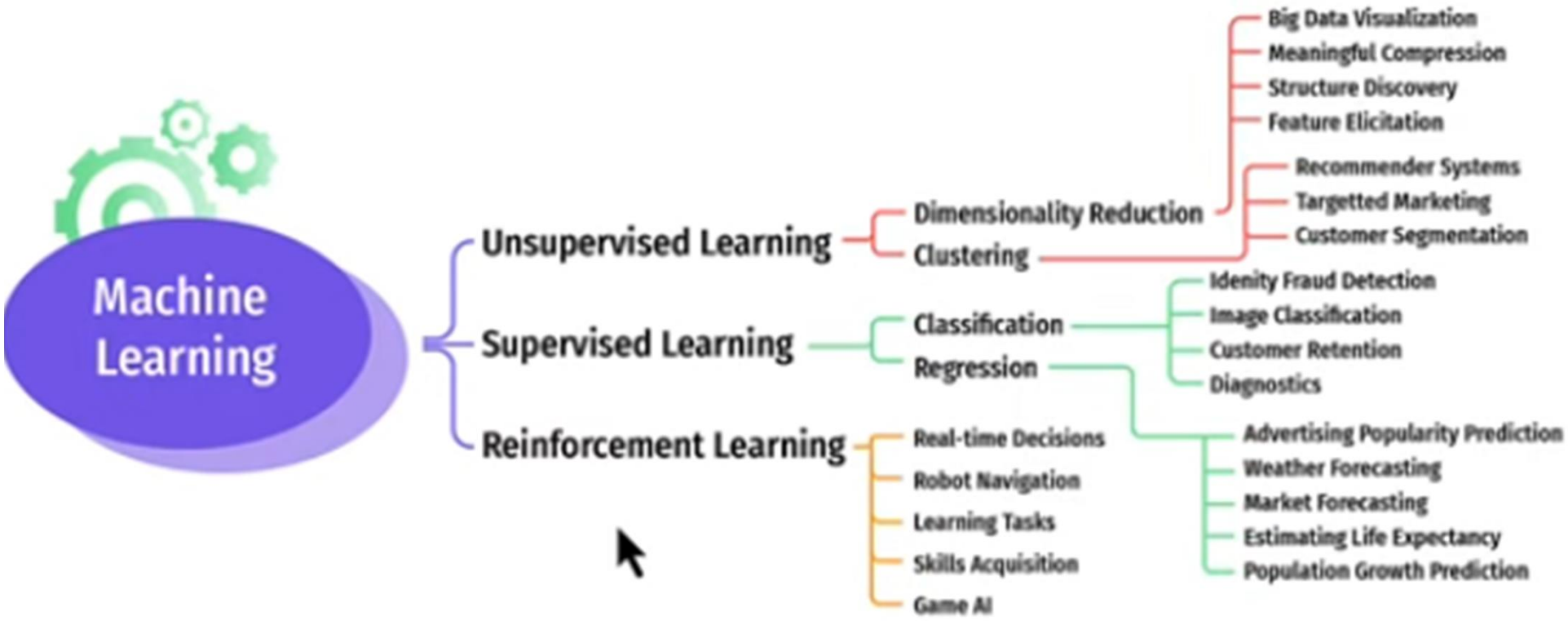
Dr Binu P Chacko

Introduction

- We need an algorithm to solve a problem using computer
- Customer likes to get the right product
- Supermarket wants to predict about the product based on customer demand (no algorithm)
- Customer behavior changes in time any by location
- If we lack in knowledge, make up in data…then learn from it
- A system in a changing environment should have the ability to learn
- Computer should extract the algorithm automatically for such task
- We have to construct a good and useful approximation. This approximation may not explain everything

Cont…

- Identifying the complete process may not be possible, still we can detect certain patterns or regularities…machine learning
- Programming computers to optimize a performance criteria using example data or past experience
- Uses statistical theory to build mathematical models
- Need efficient algorithms to solve optimization problem, and process large amount of data
- Data mining: applications of machine learning methods to large databases. Large volume of data is processed to construct a model
- This model may be predictive to make predictions, or descriptive to gain knowledge from the data, or both
- Pattern recognition: face recognition, speech recognition, DAR


Machine Learning
Unsupervised Learning
Dimensionality Reduction
Big Data Visualization
Meaningful Compression
Structure Discovery
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Classification
Idenity Fraud Detection
Image Classification
Customer Retention
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Market Forecasting
Estimating Life Expectancy
Population Growth Prediction
Reinforcement Learning
Real-time Decisions
Robot Navigation
Learning Tasks
Skills Acquisition
Game AI

Applications

Learning Associations

- Basket analysis: finding association between products brought by the customers
- Association rule: learning a conditional probability P(Y/X), P(Y/X,D)

Regression

- Autonomous car: *inputs* are provided by the sensors. *Output-the* angle by which steering wheel should be turned to advance the vehicle without hitting obstacles and deviating from the route
- Recommendation system: e.g. for movies. Learn relative positions, Learn a ranking function

Cont…

Classification

- E.g. discriminant, a function that separates low-risk(0) and high-risk(1) class customers based on income and savings. P(Y/X)…make predictions P(Y=1/X=x)
- Knowledge extraction: learning a rule from data as above
- NLP: spam filtering, analyze blogs or posts to extract trending topics
- Machine translation
- Biometrics: recognition or authentication of people using their physiological (face, finger print, iris) or behavioral characteristics (signature, voice)

Concept Learning

- Inferring a boolean-valued function from training examples of its input and output
- Task: learn to predict the value of EnjoySport
- Hypothesis: a vector of 6 constraints, specifying the values of attributes
- ? : any value is acceptable, specify a single value, θ: no value is acceptable
- (?, warm, high, ?, ?, ?)
- If some instance x satisfies all constraints of hypothesis h, h(x) = 1

Example	Sky	air	Humidity	Wind	Forecast	water	Enjoy sport
1	Sunny	warm	Normal	Strg	Same	Warm	yes
2	Sunny	warm	high	Strg	Same	Warm	Yes
3	Rainy	Cold	high	Strg	Change	Warm	No
4	Sunny	warm	high	Strg	change	cool	yes

Cont…

- EnjoySport concept learning task requires learning the set of days for which EnjoySport=yes
- Given
  - Instances X: with following attributes
    - Sky (Sunny, Cloudy, Rainy)
    - Air (Warm, Cold)
    - Humidity (Normal, High)
    - Wind (Strong, Weak)
    - Water (Warm, Cool)
    - Forecast (Same, Change)
  - Target concept c: EnjoySport: X -> {0, 1}
  - Training samples D
- Determine
  - Hypothesis h in H such that h(x) = c (x) for all x in X
- Inductive learning hypothesis: any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples

Supervised Learning

https://www.sciencedirect.com/topics/computer-science/machine-learning-approach

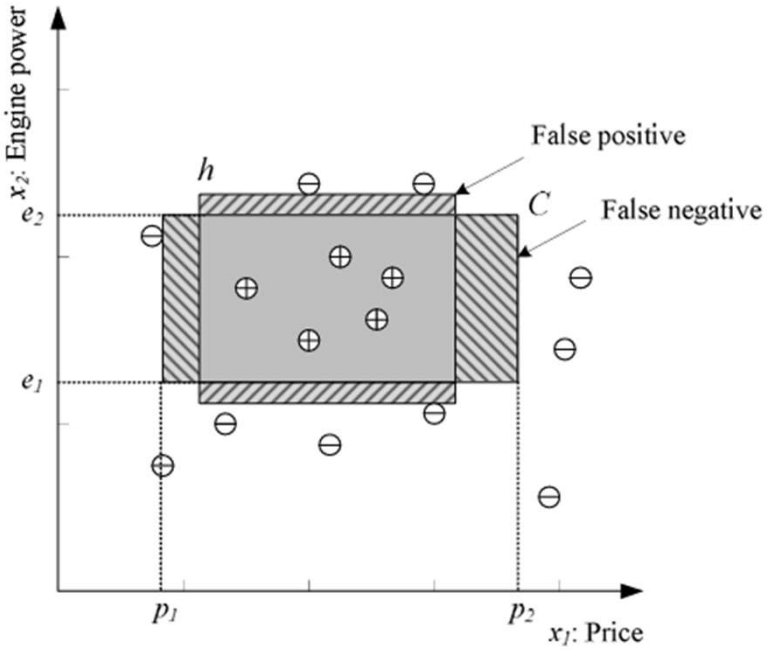
- Family car: predict by learning from positive examples and negative examples (knowledge extraction) – two class
- Features (attributes): price (x_1), engine power (x_2). These are inputs to class recognizer. Ignore other attributes
- Represent each car $\mathbf{x}=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
- Label $r=\begin{cases} 1 & positive\ \ example \\ 0 & negative\ \ example \end{cases}$ $(\mathbf{x}, r)$
- Training set of N examples $\chi = \left\{x^t, r^t\right\}_{t=1}^{N}$
- $(p_1 \le \text{price} \le p_2)$ AND $(e_1 \le \text{engine power} \le e_2)$
- Learning algorithm finds the hypothesis $h \in \mathrm{H}$ specified by the quadruple $\left(p_1^h, p_2^h, e_1^h, e_2^h\right)$ to approximate C
- Hypothesis *h* makes the prediction for an instance **x** such that $h(\mathbf{x}) = \begin{cases} 1 & positive\ \ example \\ 0 & negative\ \ example \end{cases}$

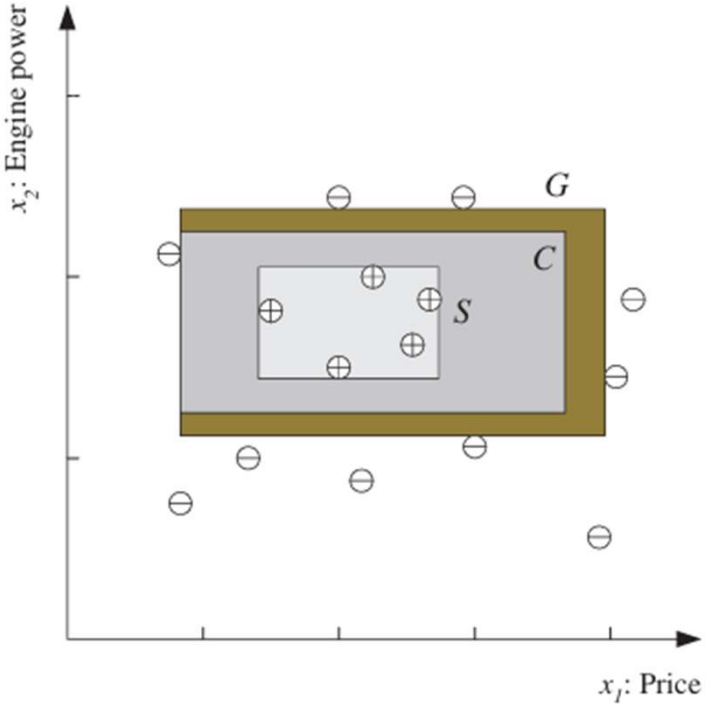
Cont…

- Empirical error: proportion of training instances where prediction of *h* don't match the required values given in χ

$$E(h \mid \chi) = \sum_{t=1}^{N} 1\left(h\left(x^t\right) \neq r^t\right)$$

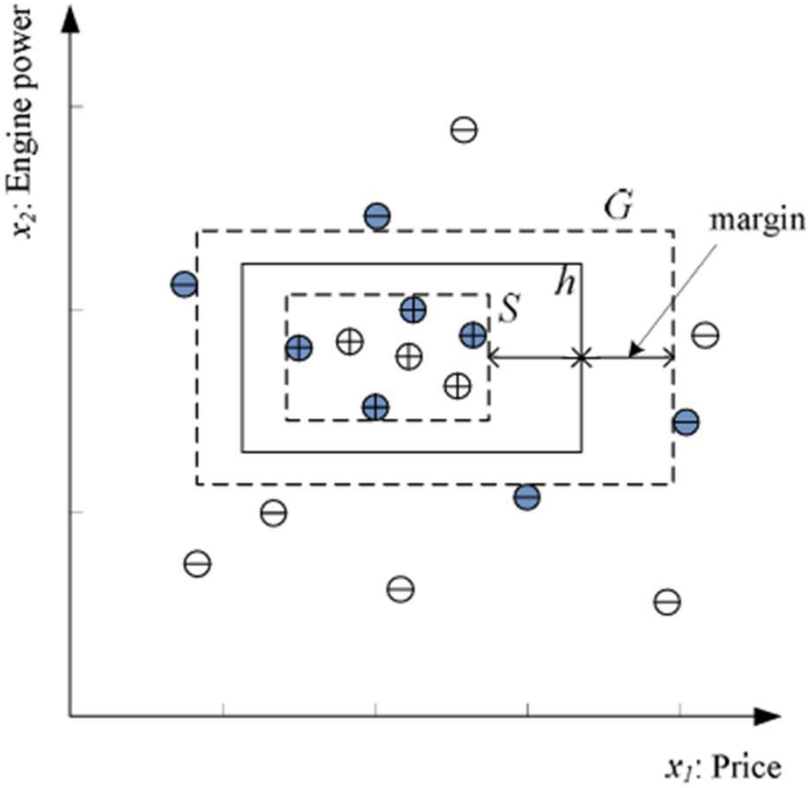
- Generalization: how well the hypothesis will correctly classify future examples that are not part of the training set
- Most specific hypothesis (S): tightest rectangle that includes all the positive examples and none of the negative examples
- Most general hypothesis (G): largest rectangle that includes all the positive examples and none of the negative examples
- Any h ∈H between S and G is a valid hypothesis with no error, said to be *consistent* with the training set, and such *h* make up the *version space*





Cont…

- Margin: distance between boundary and instances closest to it
- Use an error (loss) function to check whether an instance is on the correct side of the boundary and how far away it is



Learning Multiple Classes

- Training set $\chi = \left\{x^t, r^t\right\}_{t=1}^{N}$

$$r_i^t = \begin{cases} 1 & if\ x^t \in C_i \\ 0 & if\ x^t \in C_j, j \neq i \end{cases}$$

where i=1,2,..,K classes

- Hypothesis $h_i\left(x^t\right) = \begin{cases} 1 & if\ x^t \in C_i \\ 0 & if\ x^t \in C_j, j \neq i \end{cases}$

- Empirical error $E\left(\left\{h_i\right\}_{i=1}^{K} \mid \chi\right) = \sum_{t=1}^{N}\sum_{i=1}^{K} 1\left(h_i\left(x^t\right) \neq r_i^t\right)$

Regression

- Learn a numeric function when the output is a numeric value, e.g. time series prediction
- Training set $\chi = \left\{x^t, r^t\right\}_{t=1}^{N}$ $r^t \in \Re$

If there is no noise, task is *interpolation*

- Find the function f(x) such that $r^t = f(x^t)$
- Extrapolation: In polynomial interpolation, given N points, find $(N-1)^{th}$ degree polynomial that can be used to predict the output of any **x**
- In regression, noise(ϵ) added to the output of the unknown function $r^t = f(x^t)+\epsilon$

Noise – extra hidden variables ($\mathbf{z}^t$) $r^t=f^*(\mathbf{x}^t, \mathbf{z}^t)$

- Approximate the output by the model g(**x**)
- Empirical error

$$E(g \mid \chi) = \frac{1}{N}\sum_{t=1}^{N}\left[r^t - g\left(x^t\right)\right]^2$$

Loss function – square of difference, absolute value of the difference

- Find g(.) that minimizes empirical error. If g(**x**) is linear

$g(\mathbf{x})=w_1x_1+...+w_dx_d +w_0 =$ $\sum_{j=1}^{d} w_j x_j + w_0$

w_j are to be learned from data

Model Selection and Generalization

- Remove those hypotheses that aren't consistent with the training data
- Ill-posed problem: data is not sufficient to find a unique solution
- As we see more training examples, we know more about the underlying function
- The set of assumptions we make to have learning possible is called the *inductive bias* of the learning algorithm
- E.g. rectangle shape, rectangle with largest margin (in family car e.g.), linear function (in linear regression)
- Model selection: choosing the right bias between possible hypothesis class
- Generalization: How well a model trained on the training set predicts the right output for new instances
- For best generalization, we should match the complexity of the hypothesis class H with the complexity of the function underlying the data. If H is less complex than the function, we have *underfitting*. In such a case, as we increase the complexity, the training error decreases
- If there is noise, an overcomplex hypothesis may learn not only the underlying function but also the noise in the data and may make a bad fit. This is called *overfitting*. In such a case, having more training data helps

Cont…

Triple trade-off (in learning algorithms)

1. the complexity of the hypothesis we fit to data, namely, the capacity of the hypothesis class,
2. the amount of training data, and
3. the generalization error on new examples

- As the amount of training data increases, the generalization error decreases. As the complexity of the model class H increases, the generalization error decreases first and then starts to increase
- Divide the dataset into two – training set and validation set. Given a set of possible hypothesis classes H_i, for each we fit the best $h_i \in H_i$ on the training set. The hypothesis that is the most accurate on the validation set is the best one (the one that has the best inductive bias)
- Test set: examples that aren't used in training or validation

Bayesian Learning

- Coin tossing: only observable variable (x) is the outcome of the toss. Extra pieces of knowledge that we do not have access to are named the unobservable variables (**z**)
- x = f(**z**) deterministic function that defines the outcome
- X=1, outcome is head; X=0, outcome is tail
- $P(X=1)=p_o$ $P(X=0)=1-P(X=1)=1-p_o$
- Sample $\mathcal{X}$ containing examples, p(x) – probability distribution of observables x^t
- Build an approximator $\hat{p}(x)$

$$\hat{p}(x) = \frac{\#\{tosses \quad with \quad head\}}{\#\{tosses\}} = \frac{\sum_{t=1}^{N} x^t}{N}$$

Classification

- Credibility of the customer, C (high risk=1, low risk=0), on observables, income (X_1), savings (X_2),

$$\begin{cases} C=1 & \text{If } P(C=1|x1,x2)>0.5 \\ C=0 & \text{otherwise} \end{cases} \quad \text{OR} \quad \begin{cases} C=1 & \text{If } P(C=1|x1,x2)>P(C=0|x1,x2) \\ C=0 & \text{otherwise} \end{cases}$$

- $\mathbf{x}=[x_1, x_2]^T$, Using Bayes' rule

$$P(C|x)=\frac{P(C)p(x|C)}{p(x)}$$

Posterior probability=(prior X likelihood)/evidence

- P(C=1) is the prior probability regardless of **x**. It is the knowledge. p(**x**|C) is called class likelihood – conditional probability. p(**x**), evidence
- P(C=0|x)+P(C=1|x)=1
- In digit recognition

$$P(C_i|x)=\frac{p(x|C_i)P(C_i)}{\sum_{k=1}^{K}p(x|C_k)P(C_k)}$$

Bayes' classifier chooses the class with highest posterior probability

Risk and Loss

- Action α_i is the decision to assign the input to class C_i. λ_{ik} is the loss incurred when the input actually belongs to C_k
- Expected risk, $$R(\alpha_i \mid x) = \sum_{k=1}^{K} \lambda_{ik} P(C_k \mid x)$$
- Define K actions α_i,i= 1,...,K, where α_i is the action of assigning **x** to C_i. In the special case of the 0/1 loss where

$$\lambda_{ik} = \begin{cases} 0 & \text{if i=k} \\ 1 & \text{If i}\neq\text{k} \end{cases}$$

all correct decisions have no loss and all errors are equally costly

- In the case of wrong recognition, define an additional action of reject or doubt, α_{k+1}
- A possible loss function is

$$\lambda_{ik} = \begin{cases} 0 & \text{If i=k} \\ \lambda & \text{If i =K+1} \\ 1 & \text{otherwise} \end{cases}$$

where $0 < \lambda < 1$ is the loss incurred for choosing the $(K+1)^{st}$ action of reject. Risk of reject is

$$R(\alpha_{K+1} \mid x) = \sum_{k=1}^{K} \lambda P(C_k \mid x) = \lambda$$

Discriminant Functions

- Classification can also be seen as implementing a set of discriminant functions, $g_i(\mathbf{x})$, i = 1,...,K
- choose C_i if $g_i(\mathbf{x}) = \max_k g_k(x)$
- Represent the Bayes classifier in this way

$g_i(\mathbf{x}) = -R(\alpha_i|\mathbf{x})$, maximum discriminant function corresponds to minimum conditional risk

- When we use the 0/1 loss function, $g_i(\mathbf{x}) = P(C_i|\mathbf{x})$
- Ignoring the common normalizing term, p(x)

$g_i(\mathbf{x}) = p(\mathbf{x}|C_i)P(C_i)$

- This divides the feature space into K decision regions $R_1,...,R_K$, where $R_i = \{\mathbf{x}|g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})\}$. The regions are separated by decision boundaries

Association Rule

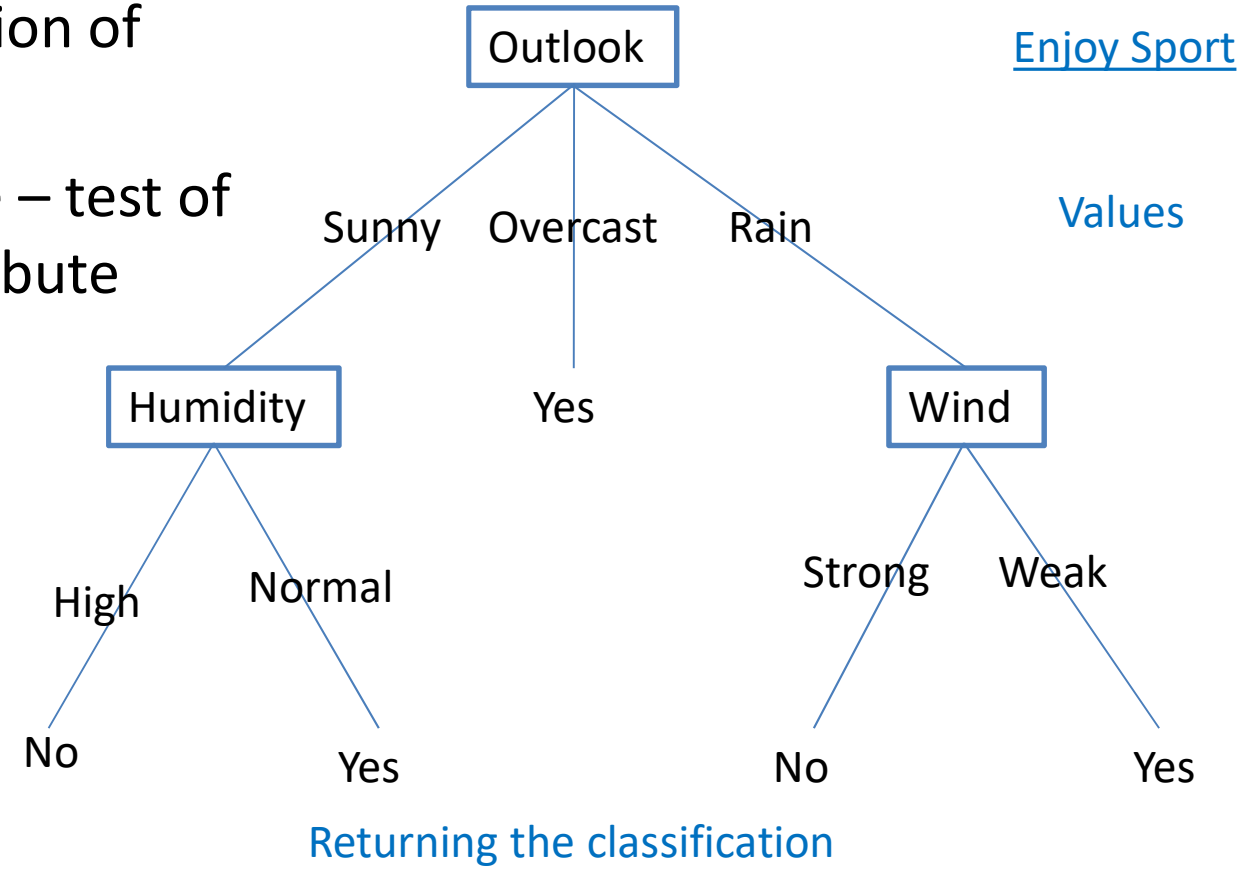
- It is an implication of the form X → Y. e.g. find the dependency between two items X and Y in basket analysis

Three measures

- Support (X,Y) ≡ P(X,Y)= $\frac{\#\{\text{customers bought X and Y}\}}{\#\{\text{customers}\}}$
- Confidence (X → Y)≡P(Y|X) = $\frac{P(X,Y)}{P(X)} = \frac{\#\{\text{customers bought X and Y}\}}{\#\{\text{customers bought X}\}}$
- Lift (X → Y)= $\frac{P(X,Y)}{P(X)P(Y)} = \frac{P(Y \mid X)}{P(Y)}$ (Interest)
- Apriori algorithm: generalization for more than two items
- finding frequent itemsets, that is, those which have enough support
- converting them to rules with enough confidence, by splitting the items into two

Decision Tree Learning

- Classification of instance
- Each node – test of some attribute



Cont…

Best for …

- Instances are represented by attribute-value pairs
- The target function has discrete output values
- Represent disjunctive expressions
- The training data may contain errors or missing attribute values

- Learns decision tree by constructing it top-down
- Greedy search for an acceptable decision tree - no backtracking

ID3 algorithm

ID3(*Examples, Target_attribute, Attributes*)

Examples are the training examples. Target attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given Examples.

- Create a Root node for the tree
- If all *Examples* are positive, Return the single-node tree Root, with label = +
- If all *Examples* are negative, Return the single-node tree Root, with label = -
- If *Attributes* is empty, Return the single-node tree *Root*, with label = most common value of *Target_attribute* in *Examples*
- Otherwise Begin

  A ← the attribute from *Attributes* that best classifies *Examples*

  The decision attribute for Root ← A

  For each possible value, v_i, of A,

  Add a new tree branch below Root, corresponding to the test A = v_i

  Let $Examples_{vi}$, be the subset of *Examples* that have value v_i for A

  if $Examples_{vi}$, is empty

  Then below this new branch add a leaf node with label = most common value of *Target_attribute* in *Examples*

  Else below this new branch add the subtree

  ID3($Examples_{vi}$, *Target attribute, Attributes* – {A}))

End

Return *Root*

Cont…

- Entropy characterizes the (im)purity of an arbitrary collection of examples (S)
- Entropy (S) ≡ $-p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$ Boolean classification

Entropy 0 if all members are in same class; 1 if members are equal in two classes; if unequal, entropy is from 0 to 1

$$Entropy(S) \equiv \sum_{i=1}^{c} - p_i \log_2 p_i$$ c classes

Information gain: how well a given attribute separates the training examples according to their target classification. Expected reduction in entropy caused by partitioning the examples according to the attribute

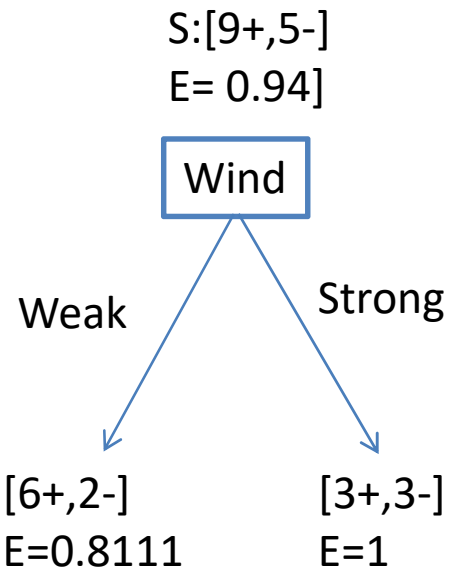
$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

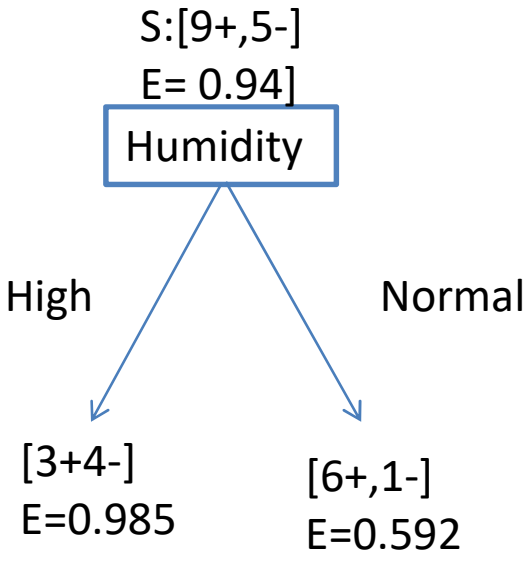
Attribute A
S_v the subset of S having value v of A

information provided about the target function value, given the value of some attribute A

expected value of the entropy after S is partitioned using attribute A.

Cont…





Gain (S, Humidity)

=0.94-(7/14)0.985-(7/14)0.592

=0.151

Gain (S, Wind)

=0.94-(8/14)0.811-(6/14)1

=0.048

Hypothesis Space Search

- ID3 (Iterative Dichotomiser) searches through the space of possible decision trees from simplest to increasingly complex, guided by the information gain heuristic

Capabilities and limitations of ID3

- complete space of finite discrete-valued functions, relative to the available attributes
- maintains only a single current hypothesis
- no backtracking; converge to locally optimal solutions that are not globally optimal
- ID3 can be easily extended to handle noisy training data

Inductive Bias

- Describing the basis by which it chooses one of the consistent hypothesis over the others
- ID3 search strategy selects in favor of shorter trees over longer ones
- selects trees that place the attributes with highest information gain closest to the root
- Preference bias (search bias): preference for certain hypotheses over others (e.g., for shorter hypotheses). Works within a complete hypothesis space
- Restriction bias (language bias): categorical restriction on the set of hypotheses considered. Limits the set of potential hypotheses

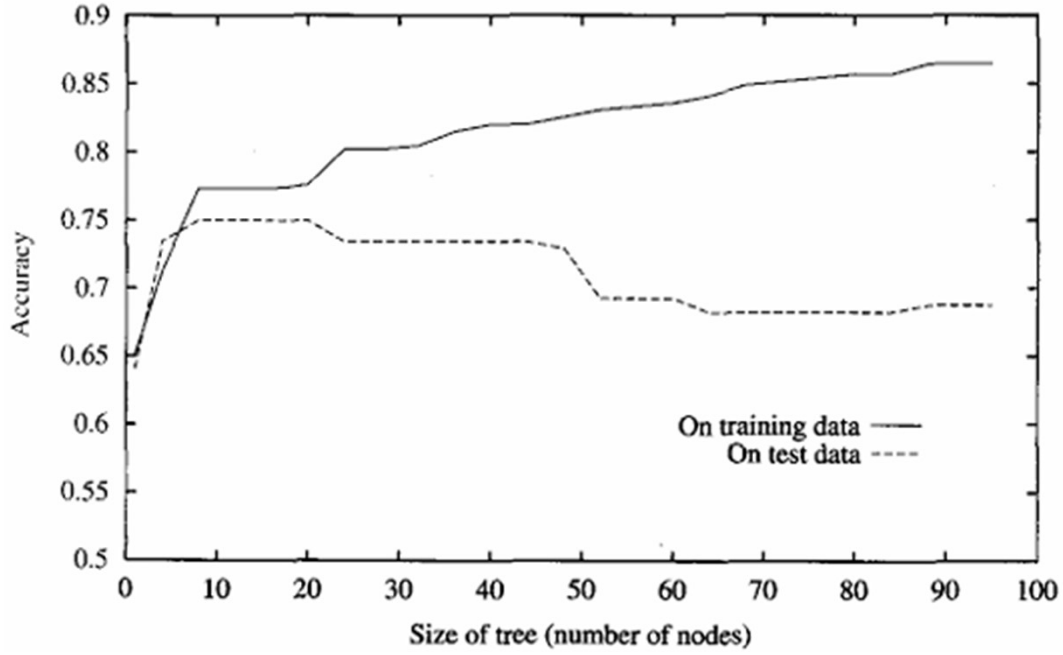
Prefer the simplest hypothesis that fits the data

- ➢ There are relatively few such trees
- ➢ Size of a hypothesis is determined by the particular representation used internally by the learner. Learners would generalize in different ways
- Inductive bias can alter the representation easier than it can alter the learning algorithm

Issues

Avoid overfitting the data

- When there is noise in the data
- Training samples are less
- *Given a hypothesis space H, a hypothesis h Ɛ H is said to overfit the training data if there exists some alternative hypothesis h' Ɛ H, such that h has smaller error than h' over the training examples, but h' has a smaller error than h over the entire distribution of instances*



Cont…

Solution

- Approaches that stop growing the tree earlier
- Approaches to post-prune the tree

To determine the correct tree size

- ❑ Use a separate set of examples - training (to form the learned hypothesis) and validation set (to evaluate the accuracy of this hypothesis) approach
- ❑ Estimate whether further expanding a node is likely to improve performance over the entire instance distribution (chi-square test)
- ❑ Minimum Description Length principle - halting growth of the tree
- Reduced error pruning: Prune a decision node and assigning it the most common classification of the training examples affiliated with that node
- Rule post-pruning: To find high accuracy hypotheses. To estimate rule accuracy, use validation set or training set

1. Grow the tree and allow overfitting
2. Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node
3. Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy
4. Sort the pruned rules by their estimated accuracy

Cont…

Incorporating Continuous-Valued Attributes (in decision nodes)

- For a continuous-valued attribute, create a new boolean attribute A_c, where c is the threshold. E.g. temperature range in EnjoySport

Alternative Measures for Selecting Attributes

- Information gain measure may favors attributes with many values over those with few values (E.g. date)
- But it is not a useful predictor, because it is bound to separate the training examples into very small subsets

Solution

- SplitInformation(S, A)≡ $-\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$
- GainRatio(S,A)≡ $\frac{Gain(S,A)}{SplitInformation(S,A)}$

earlier Gain measure

Cont…

Handling Training Examples with Missing Attribute Values

- Based on other examples
- Assign a probability to each of the possible values of A. This probability is used to compute information Gain. E.g. C4.5

Handling Attributes with Differing Costs

- In some learning tasks the instance attributes may have associated costs. Prefer to use low-cost attributes

$$\frac{Gain^2(S,A)}{Cost(A)}$$

Multivariate Methods

- Emerged in response to increasingly complex datasets
- Allow DA to identify relationships between variables, Reduce dimensionality through PCA, Improve model accuracy

Steps

- Data collection and preprocessing: Data source (survey, experiment, database)
- Data cleaning: addressing missing values, outliers, inconsistencies
- Standardization: essential when variables are measured in different scales
- Min-Max Normalization , StandardScaler, GaussianNormalization $z = \frac{x - \mu}{\sigma}$
- Feature engineering: Creating new features or modifying existing ones may help in capturing the underlying relationships in the data

Cont…

Exploratory data analysis: to identify trends, patterns, anomalies

- Visualization techniques: graphs can reveal distributions and correlations
- Statistical summaries: Mean, median, variance, and correlation matrices provide a numerical snapshot of the data
- Outlier detection: Analyzing the spread and variability in data helps in identifying anomalous points

Selection of relevant variables (through): Correlation analysis (examining the correlation coefficients to identify interrelated variables)

- Variable importance measures: Techniques like backward elimination and forward selection in regression models help in identifying key predictors
- Domain knowledge: Combining statistical techniques with expert insights can improve variable selection

Multivariate Techniques

Dimensionality reduction approaches

- These methods transform the original variables into a new set of variables while retaining most of the original information
- PCA finds a new set of principal components that capture the maximum variance in the dataset, **Z=XW**

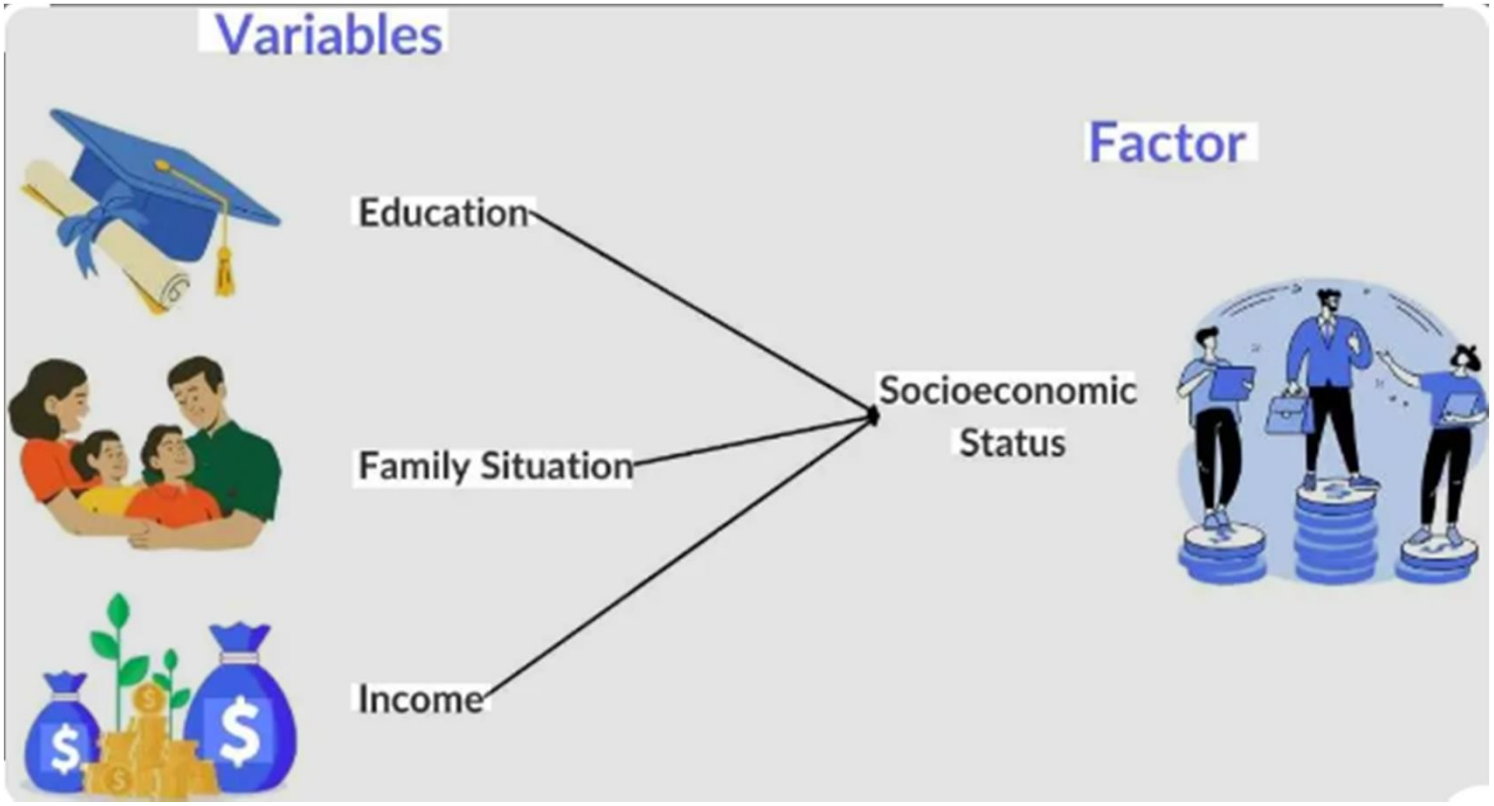
where **X** is the data matrix, **W** is the matrix of eigenvectors, and **Z** is the transformed data matrix

- Steps: calculate covariance matrix to understand how variables interact

Compute eigenvectors and eigenvalues: The eigenvectors determine the direction of the new feature space, while the eigenvalues determine their magnitude (importance)

Project the data: The original data is then projected onto the new axes to attain a reduced dimensional representation

- Factor analysis helps in interpreting how different variables collectively contribute to the outcome. It is especially useful in psychological and sociological research


Variables
Factor
Education
Family Situation
Income
Socioeconomic Status

Cont…

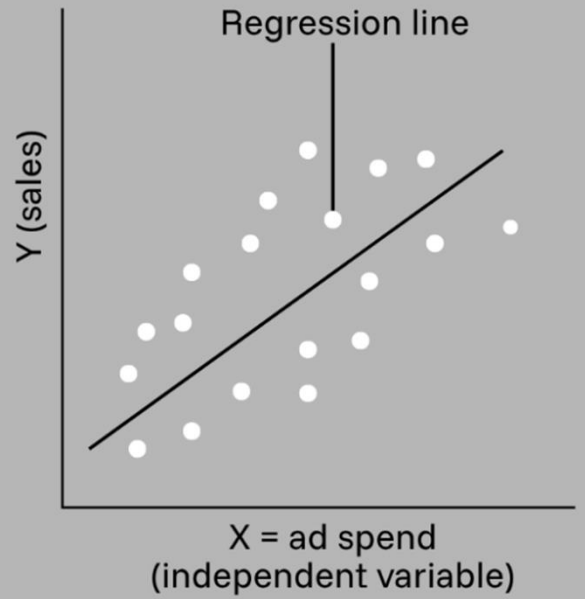
Multivariate Regression Techniques

- These models are used to predict multiple dependent variables simultaneously
- Multiple Regression**:** When the relationship between one dependent variable and several independent variables is of interest, regression models are helpful. E.g. predict the risk of developing disease
- $y=\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_k x_k+\epsilon$

where β_0 is the intercept (a constant that finds the value of y when x_1 .. x_k are 0), β_1 to β_k are the coefficients, and ϵ is the error term

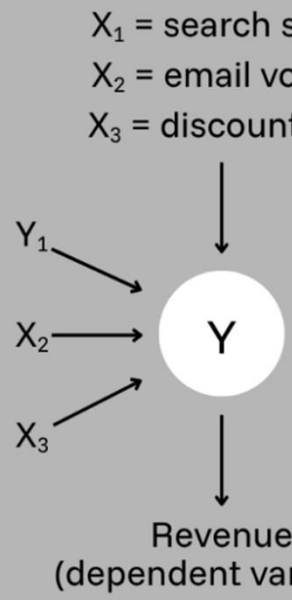
- Multivariate Analysis of Variance (MANOVA): This test extends the univariate ANOVA to multiple dependent variables. To test whether the vectors of means differ significantly across groups

Simple regression



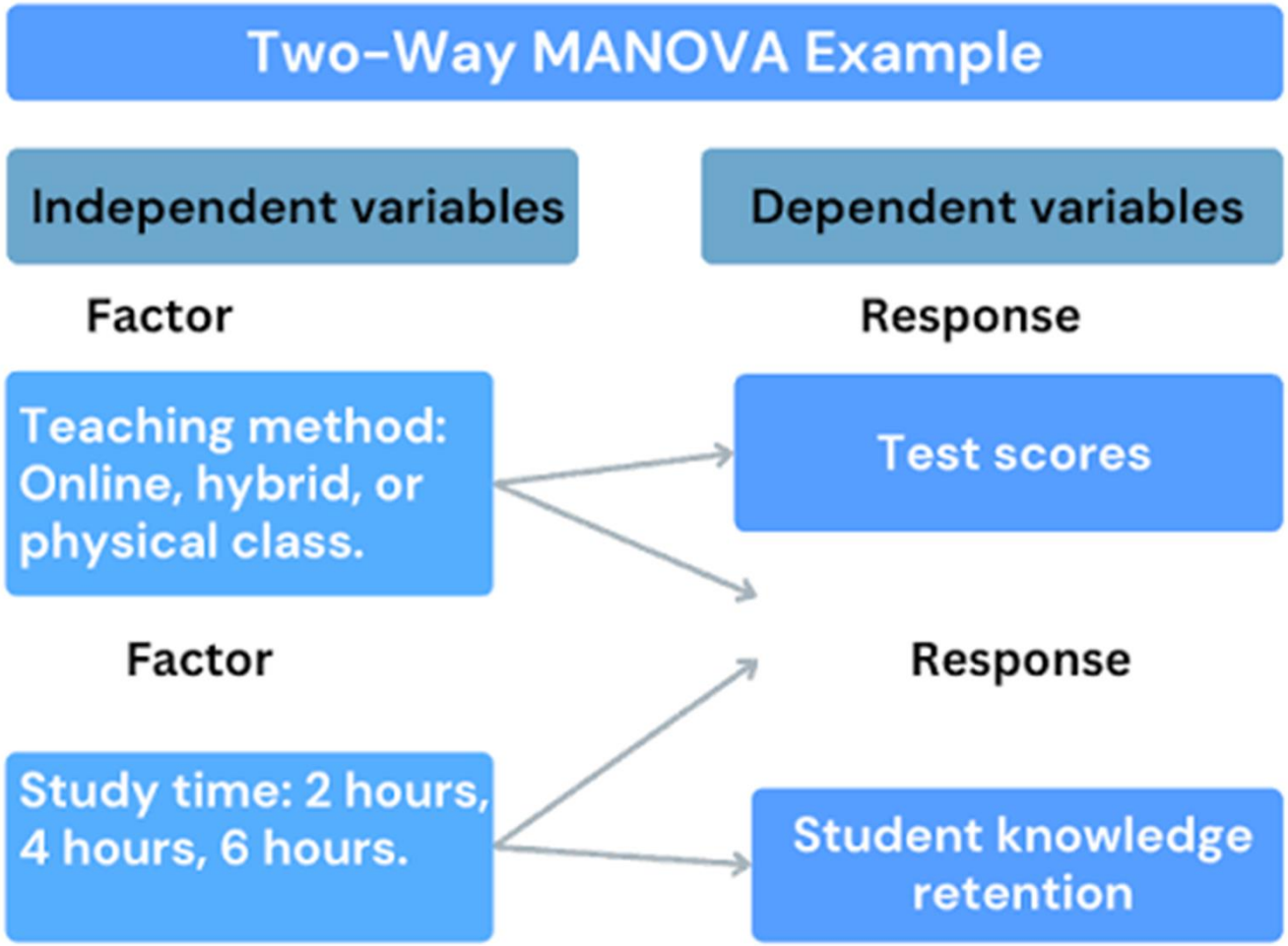
One predictor ⟶ One outcome

Multiple regression



$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$$

Multiple predictors ⟶ One outcome


Two-Way MANOVA Example
Independent variables
Dependent variables
Factor
Response
Teaching method: Online, hybrid, or physical class.
Test scores
Factor
Response
Study time: 2 hours, 4 hours, 6 hours.
Student knowledge retention

Multivariate Classification

- Bivariate: model will operate on two features
- Multivariate: model will operate on more than two features
- Supervised learning is used for solving predictive problems
- In unsupervised learning you want to aggregate each of the locations into one of a specified number of groups or clusters
- Hypothesis or ML model should predict the label based on inputs
- Multivariate statistics are calculated on training samples to establish the relationships within and between the classes
- Building blocks: output function, gradient descent algorithm, loss function, learning rate

Steps

- Create and analyze the input data
- Produce signatures for class and cluster analysis
- Evaluate and edit classes and clusters
- Perform the classification

Multivariate Regression

- Regression predictive modeling is the task of approximating a mapping function (f) from input variables (X) to a continuous output variable (y)
- Shows the linear relationship between more than one predictor or independent variable and more than one output or dependent variable
- It is a supervised machine learning algorithm
- E.g. estimate the price of a house, predict crop yield, predict GDP, predict electricity bill
- Answers to - What are the critical variables that impact output? Which ones should we ignore? How do they interact with each other?
- Cost function, MSE= $\frac{1}{2m}\sum\left(h_\theta(x)^{(i)} - y^i\right)^2$

Actual value Predicted value

Smaller MSE implies better performance. Cost function allows a cost to samples when the model differs from observed data

Cont…

Steps

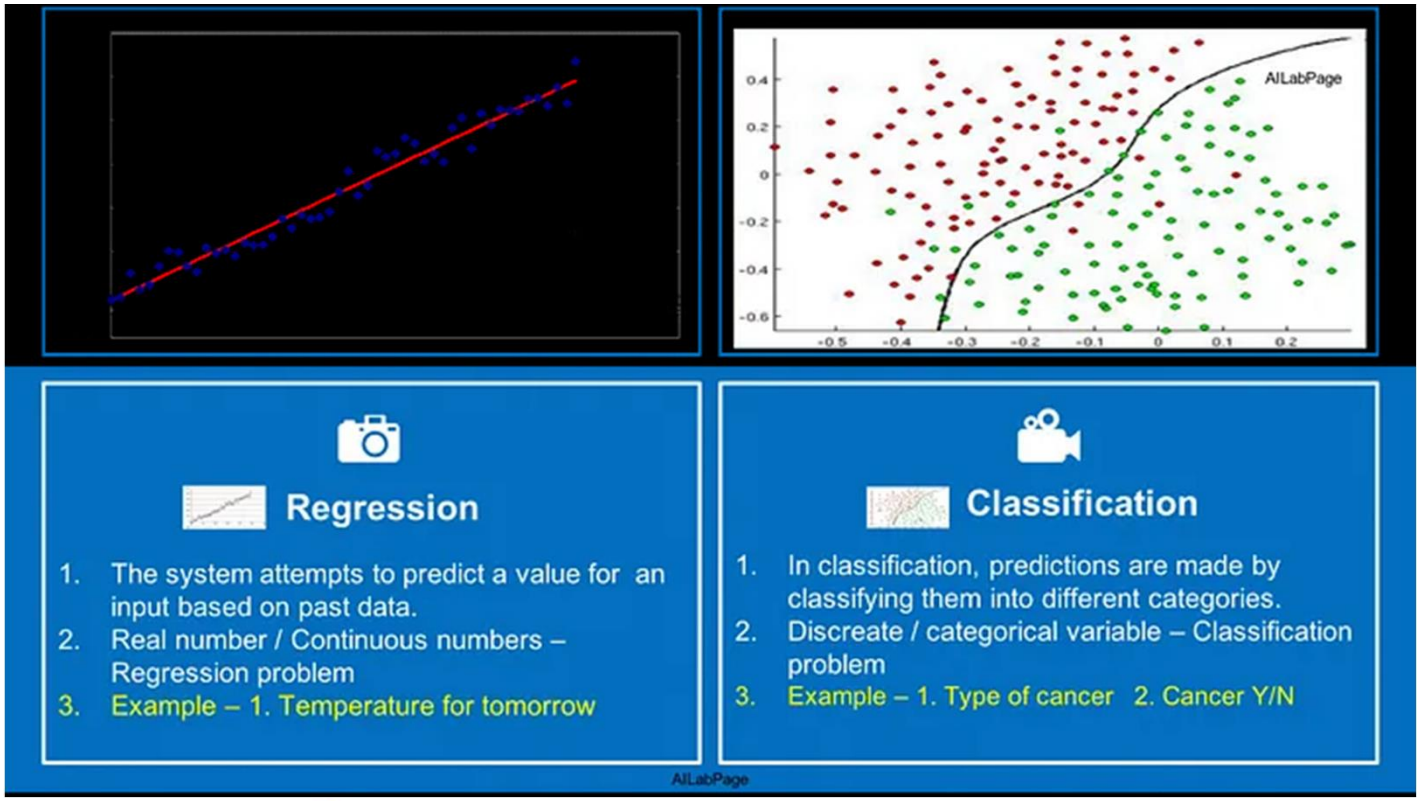
- Feature selection for better model building
- Normalizing (scale) features
- Select loss function and hypothesis: loss function predicts whenever there is an error. i.e., when the hypothesis prediction deviates from actual values
- Set hypothesis parameters in such a way that it reduces the loss function and predicts well
- Minimize the loss function by using a loss minimization algorithm (gradient descent) on the dataset, which will help in adjusting hypothesis parameters
- Test the hypothesis function as it is predicting the values

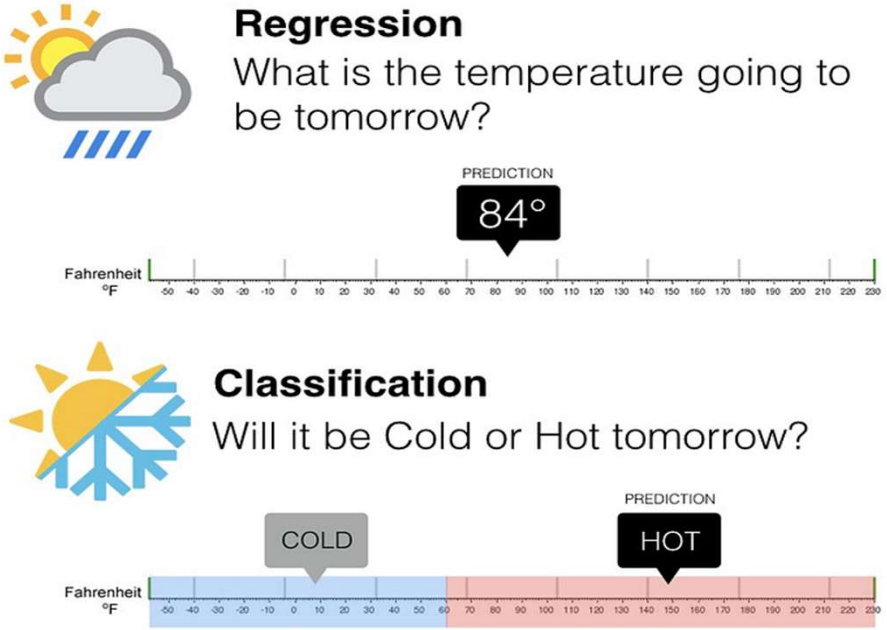
Advantages

- Improved predictive accuracy, Handle complex relationships, Reduces bias, Identifies key predictors, Improves model fit to data

Disadvantages

- It is complex and requires a high level of mathematical calculation
- The output produced by multivariate models is sometimes not accessible to interpret because it has some loss and error outputs that are not identical
- These models are not suitable for smaller datasets

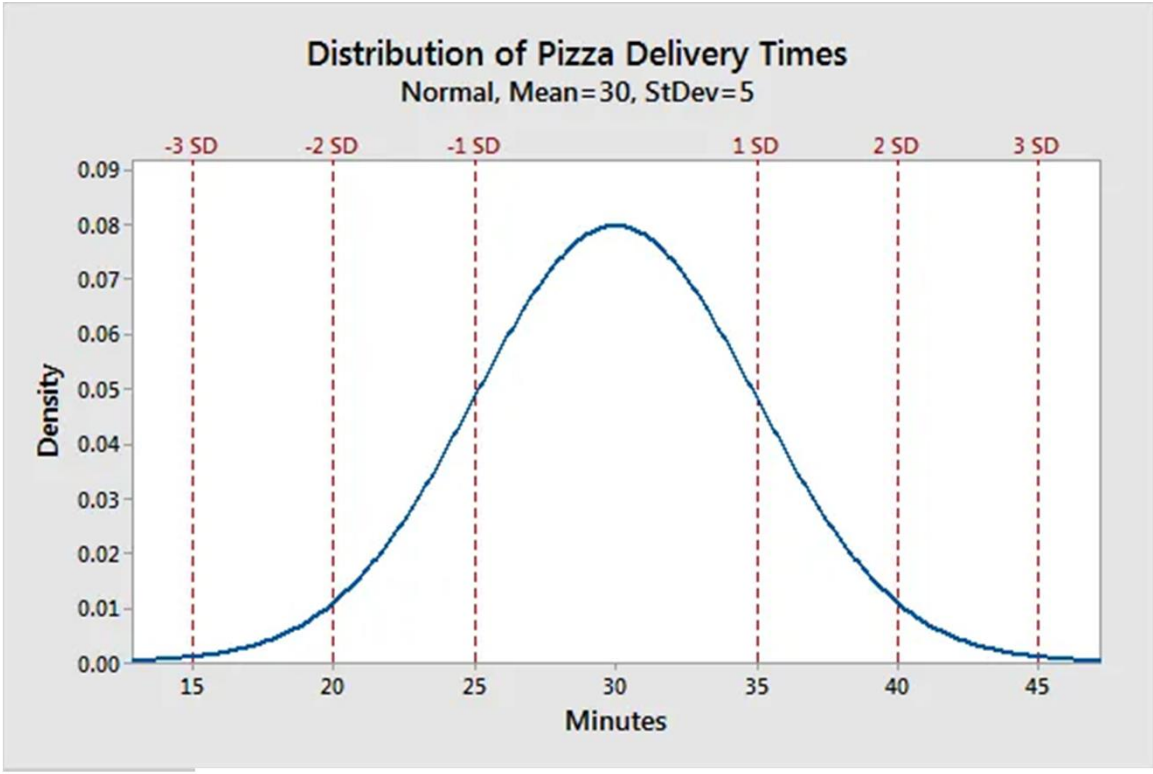

AILabPage
Regression
1. The system attempts to predict a value for an input based on past data.
2. Real number / Continuous numbers – Regression problem
3. Example – 1. Temperature for tomorrow
Classification
1. In classification, predictions are made by classifying them into different categories.
2. Discreate / categorical variable – Classification problem
3. Example – 1. Type of cancer 2. Cancer Y/N
AILabPage


Regression
What is the temperature going to be tomorrow?
PREDICTION
84°
Fahrenheit
°F
Classification
Will it be Cold or Hot tomorrow?
PREDICTION
COLD
HOT
Fahrenheit
°F

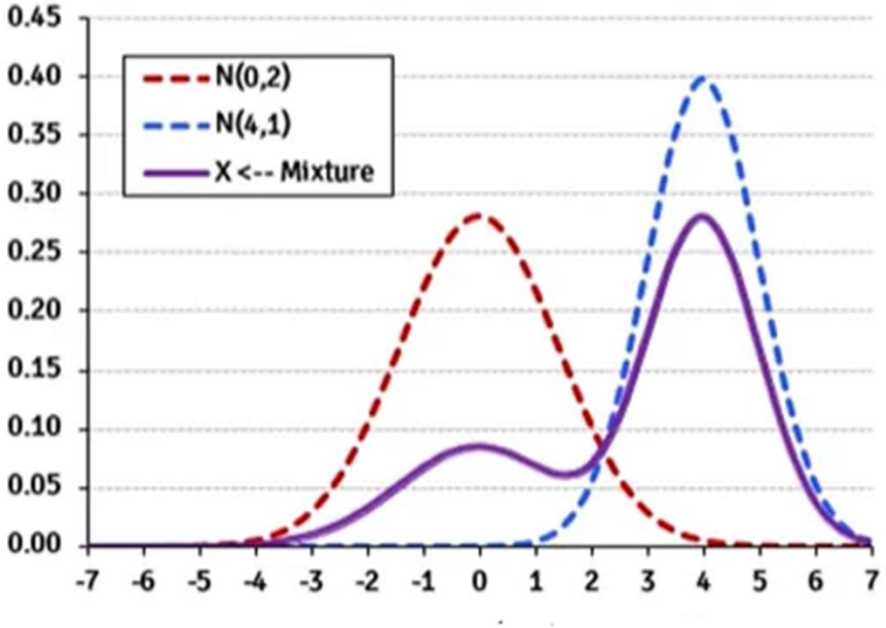
Clustering

- Allows learning the mixture parameters from data
- Mixture densities $p(x) = \sum_{i=1}^{k} p(x|G_i)P(G_i)$
- G_i - mixture components (groups or clusters), $p(x|G_i)$ – component densities, $P(G_i)$ – mixture proportions
- Learning corresponds to estimating the component densities and proportions
- Parametric classification is a bonafide mixture model where groups, G_i, correspond to classes, C_i, component densities $p(x|G_i)$ correspond to class densities $p(x|C_i)$, and $P(G_i)$ correspond to class priors, $P(C_i)$

$$p(x) = \sum_{i=1}^{K} p(x|C_i)P(C_i)$$

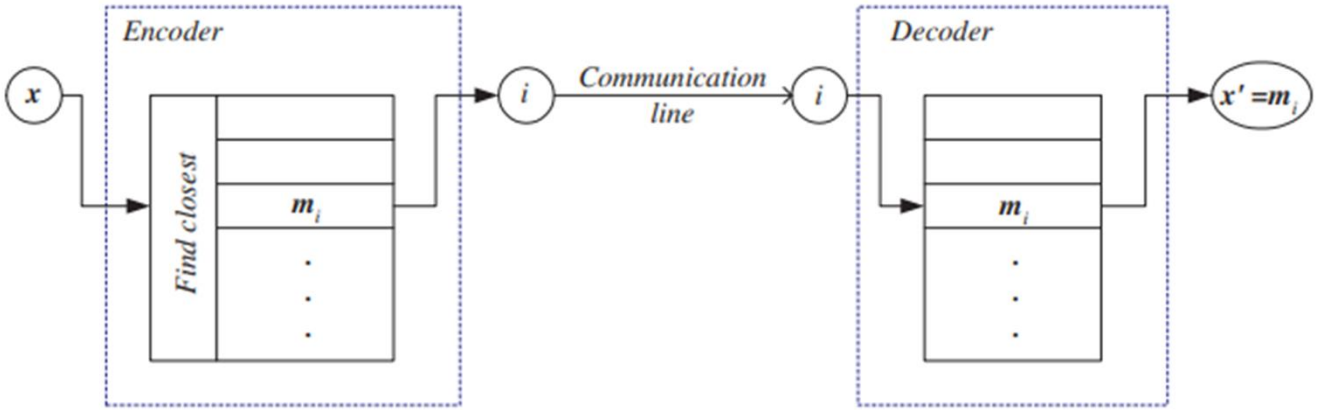


- If you add up 2 normal distributions (red and blue), where each distribution have its own mean and standard deviation, it will gives back the bimodal general distribution (purple)



k-means Clustering

- Colour quantization (compression): map from high to lower resolution. E.g. displaying 24 bit image in 8 bit screen
- Vector quantization: map from a continuous space to a discrete space
- Let sample $\mathcal{X} = \{x^t\}_{t=1}^N$ We have k reference vectors, $\mathbf{m}_j$, j = 1,...,k. Represent the image with most similar entry, $\mathbf{m}_i$ (codebook vector or code words) in the colour map, satisfying



Given $\boldsymbol{x}$, the encoder sends the index of the closest code word and the decoder generates the code word with the received index as $\boldsymbol{x}'$. Error is $\|\boldsymbol{x}' - \boldsymbol{x}\|^2$

Cont…

- Total reconstruction error $E(\{m_i\}_{i=i}^{k} \mid \chi) = \sum_t \sum_i b_i^t \left\| x^t - m_i \right\|^2$

where $b_i^t = \begin{cases} 1 & if \left\| x^t - m_i \right\| = \min_j \left\| x^t - m_j \right\| \\ 0 & otherwise \end{cases}$

- The best reference vectors are those that minimize the total reconstruction error
- An iterative procedure named k-means clustering is used to solve the above optimization problem

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$
Repeat
  For all $\boldsymbol{x}^t \in \mathcal{X}$
    $b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$
  For all $\boldsymbol{m}_i, i = 1, \ldots, k$
    $\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$
Until $\boldsymbol{m}_i$ converge

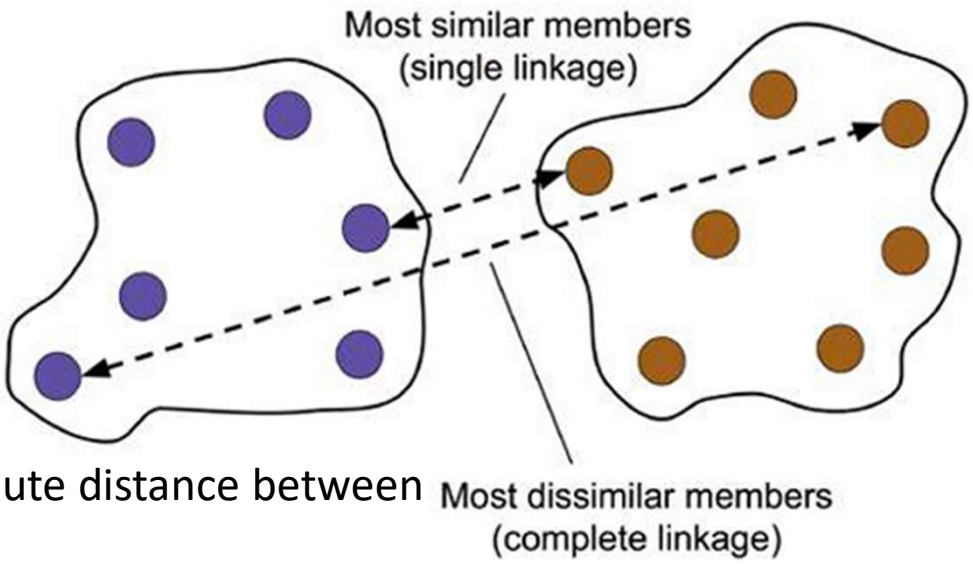
Hierarchical Clustering

- Need not specify the No. of clusters in advance
- Divisive clustering: start with one cluster, and iteratively split the cluster into smaller clusters until each cluster only contains one example

Agglomerative clustering

- start with each example as an individual cluster and merge the closest pairs of clusters until only one cluster remains

Algorithms



Other linkages
Average linkage
Ward's linkage
Centroid linkage

- **Single Linkage**

  $D(c_1,c_2) = \min D(x_1,x_2)$

  Minimum distance or distance between closest elements in clusters

- **Complete Linkage**

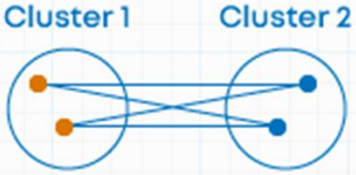
  $D(c_1,c_2) = \max D(x_1,x_2)$

  Maximum distance between elements in clusters

- **Average Linkage**

  $D(c_1,c_2) = \frac{1}{|c_1|}\frac{1}{|c_2|}\Sigma\Sigma D(x_1,x_2)$

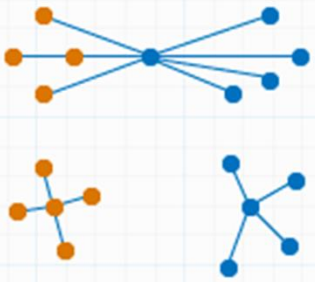
  Average of the distances of all pairs



- **Centroid Method**

  Combining clusters with minimum distance between the centroids of the two clusters

- **Ward's Method**
  - Combining clusters where increase in within cluster variance is to the smallest degree.
  - Objective is to minimize the total within cluster vairance



Complete Linkage

1. Compute the distance matrix of all examples
2. Represent each data point as a singleton cluster
3. Merge the two closest clusters based on the distance between the most dissimilar (distant) members
4. Update the similarity matrix
5. Repeat steps 2-4 until one single cluster remains

Observations

Features	X	Y	Z
ID_0	6.964692	2.861393	2.268515
ID_1	5.513148	7.194690	4.231065
ID_2	9.807642	6.848297	4.809319
ID_3	3.921175	3.431780	7.290497
ID_4	4.385722	0.596779	3.980443

Euclidean distance

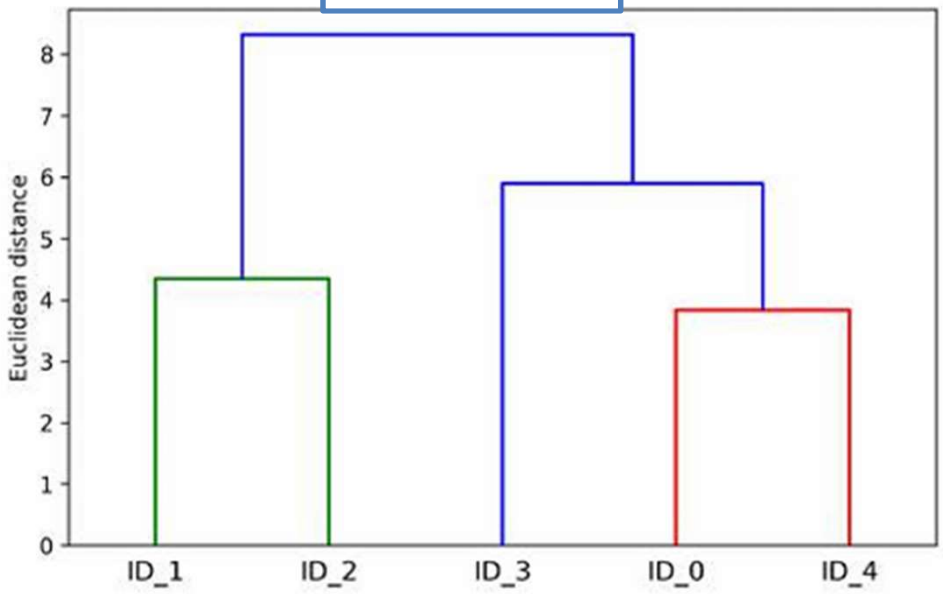
	ID_0	ID_1	ID_2	ID_3	ID_4
ID_0	0.000000	4.973534	5.516653	5.899885	3.835396
ID_1	4.973534	0.000000	4.347073	5.104311	6.698233
ID_2	5.516653	4.347073	0.000000	7.244262	8.316594
ID_3	5.899885	5.104311	7.244262	0.000000	4.382864
ID_4	3.835396	6.698233	8.316594	4.382864	0.000000

Cont...

Linkage Matrix

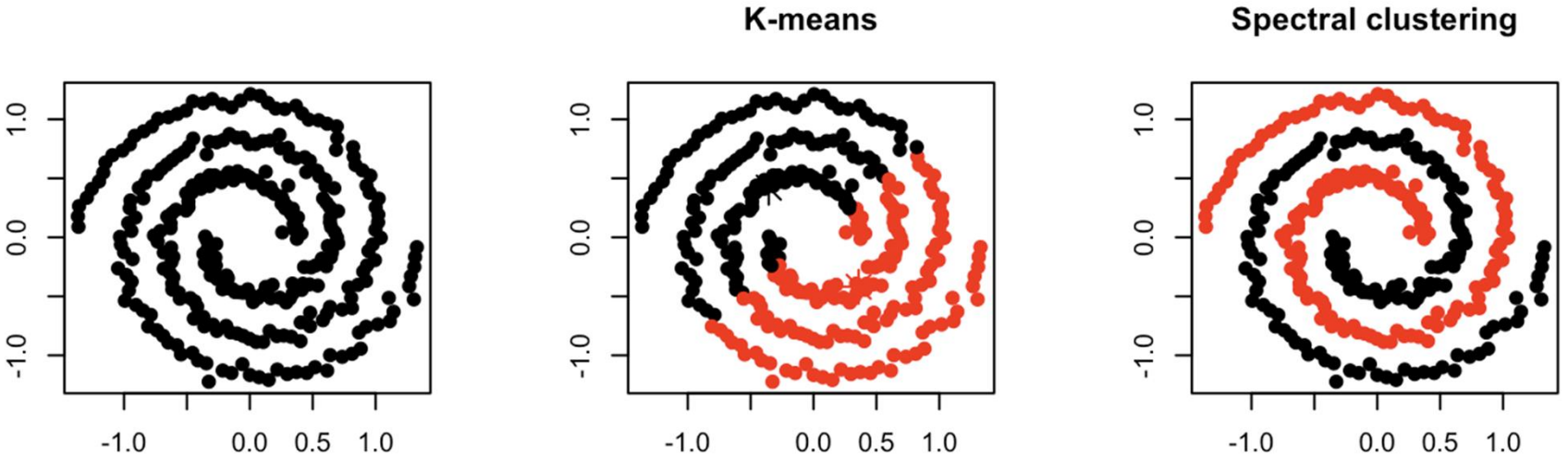
	row label 1	row label 2	distance	no. of items in clust.
cluster 1	0.0	4.0	3.835396	2.0
cluster 2	1.0	2.0	4.347073	2.0
cluster 3	3.0	5.0	5.899885	3.0
cluster 4	6.0	7.0	8.316594	5.0

Dendrogram



Spectral Clustering

- Based on spectral graph theory
- It treats data clustering as a graph partitioning problem
- To uncover hidden structures and patterns within the data
- It explores relationship among data points
- It clusters data points together into categories
- Applications: image segmentation, social network analysis, document clustering
- In connectivity clustering approach, the points in a cluster are either immediately next to each other or connected.
- K-means (compactness clustering): points are closer to each other and are compact towards the cluster center



Procedure

- Start with data points for clustering
- Build a similarity graph: Connect points that are close to each other and assign edge weights based on similarity. Use Euclidean distance, cosine similarity, or mutual information
- Make a graph Laplacian: From the similarity graph, compute a mathematical matrix Laplacian. This captures the structure of the graph

L = D – A, where A is the adjacency matrix (if two nodes are connected, store the weight in the matrix), D is the degree matrix (a diagonal matrix where each diagonal entry is the sum of edge weights connected to that node)

- Calculate a few eigenvectors of the Laplacian. Project the data points onto the Laplacian matrix's leading eigenvectors. This gives a new low-dimensional representation of the data
- Eigenvectors associated with smaller eigenvalues contain valuable insights regarding the underlying structure of clusters

For a square matrix **A**, an eigenvector v satisfies: $Av = \lambda v$, where λ is the eigenvalue. Multiplying v by **A** just stretches or shrinks it, but doesn't change its direction.

- Cluster in new space: Apply k-means clustering on the new representation v

Cont…

Advantages

- Handling non-convex and irregularly shaped clusters
- Hidden structure discovery
- Robustness to noise
- It can be applied to a wide range of domains

Disadvantages

- Sensitivity to hyperparameters such as the number of clusters and the affinity measure
- Computationally intensive
- It may not scale well to extremely large datasets due to the eigenvalue decomposition step

Expectation-Maximization algorithm

- Iterative, unsupervised – to find unknown values (hidden or missing or unobserved latent variable) in statistical models
- It is widely used in clustering like Gaussian Mixture Models

Steps

- initialize parameter values
- Expectation step: Estimates missing or hidden values using current parameters
- Calculate the posterior probability of each latent variable based on the observed data
- Compute the log-likelihood of the observed data using the current parameters
- Maximization step: Updates model parameters to maximize the likelihood based on the estimated values from the E-step
- The processes repeat until the model reaches a stable solution

Key Terms

- Latent variables: These are hidden parts of the data. We try to guess their values using the visible data
- Likelihood: This refers to the probability of seeing the data we have based on certain assumptions or parameters. The EM algorithm tries to find the best parameters that make the data most likely
- Log-likelihood: This is the natural log of the likelihood function. It's used to make calculations easier and measure how well the model fits the data. The EM algorithm tries to maximize the log-likelihood to improve the model fit
- Maximum likelihood estimation: This is a method to find the best values for a model's settings called parameters. It looks for the values that make the data we observed most likely to happen
- Posterior probability: it helps to estimate the "best" parameters when there's uncertainty about the data
- Convergence: if the changes in the model's parameters or the log-likelihood are small enough to stop the process

- Assume that we have two coins, C1 and C2
- Assume the bias of C_1 is θ_1 (i.e., probability of getting heads with C_1)
- Assume the bias of C_2 is θ_2 (i.e., probability of getting heads with C2)
- We want to find θ_1, θ_2 by performing a number of trials (i.e., coin tosses)

First experiment

- We choose 5 times one of the coins.
- We toss the chosen coin 10 times

$$\theta_1 = \frac{number\ of\ heads\ using\ C1}{total\ number\ of\ flips\ using\ C1}$$

$$\theta_2 = \frac{number\ of\ heads\ using\ C2}{total\ number\ of\ flips\ using\ C2}$$

Coin A	Coin B
	5 H, 5 T
9 H, 1 T	
8 H, 2 T	
	4 H, 6 T
7 H, 3 T	
24 H, 6 T	9 H, 11 T

$$\theta_1 = \frac{24}{24+6} = 0.8$$

$$\theta_2 = \frac{9}{9+11} = 0.45$$

Assume a more challenging problem

H T T T H H T H T H

H H H H T H H H H H

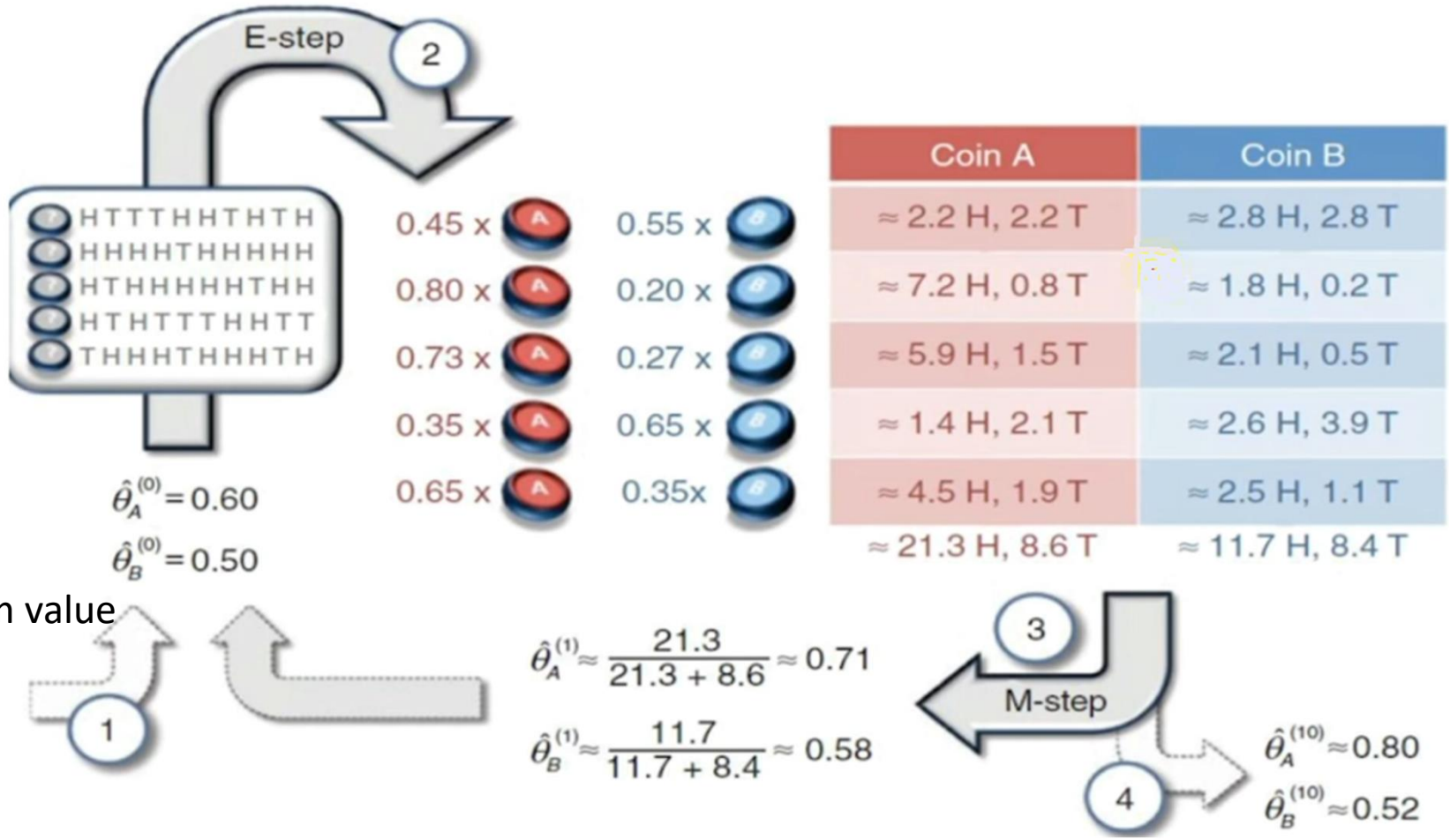
H T H H H H H T H H

H T H T T T H H T T

T H H H T H H H T H

- We do not know the identities of the coins used for each set of tosses (we treat them as hidden variables).

Solution: apply EM algorithm



Convergence
after 10 iterations

Binomial Distribution

Experiment E

Expectation step

$$P(E \mid Z_A) = P(HHHHHHHHHT \mid \text{A chosen}) = \binom{n}{x} \theta_A^x (1-\theta_A)^n$$

$$P(E \mid Z_B) = P(HHHHHHHHHT \mid \text{B chosen}) = \binom{n}{x} \theta_B^x (1-\theta_B)^n$$

$$P(E|Z_A) = \binom{9}{1} * (0.6)^9 * (0.4)^1 = 0.036$$

Likelihood probability

$$P(E|Z_B) = \binom{9}{1} * (0.5)^9 * (0.5)^1 = 0.009$$

$$P(Z_A|E) = \frac{0.036}{0.036 + 0.009} = 0.80$$

Actual probability

$$P(Z_B|E) = \frac{0.009}{0.036 + 0.009} = 0.20$$

Cont…

Advantages

- Improves result: With each step, the algorithm improves the likelihood (chances) of finding a good solution
- Simple to implement, Quick math solution

Disadvantages

- It converges slowly, i.e., it may take many iterations to reach the best solution
- Get stuck in local best: Instead of finding the absolute best solution, it might settle for a *good local* one
- Need extra probability: EM requires both forward and backward probabilities making it slightly more complex

Latent Variable Models

Example: Generating a Face

Observed data (image)

Pose
Head position, angle, viewpoint

Expression
Smile, frown, open mouth, etc.

Identity
Person's unique facial features

The observed face is explained by underlying latent factors: pose, expression, identity, and more.

What Are Latent Variables?

- Latent variables (z) are hidden factors that we cannot observe directly.
- They capture the underlying structure that generates the data.
- By learning a model, we can infer these hidden factors from data.

Examples of Latent Factors

Pose

Expression

Identity

Lighting, age, style, etc.

Generative Modeling with Latent Variables

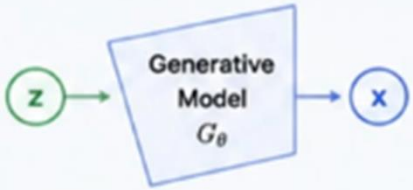
1. Latent Space (hidden)

Sample a latent code

$z \sim p(z)$

2. Generative Process

A generative model $p_\theta(x \mid z)$ maps latent code to data



3. Observed Data (visible)

Generate data sample

$z \sim p_\theta(z)$

Model learns to:

- Infer z from x (inference)
- Generate new x (generation)
- Interpolate in z (smooth transitions)
- Edit specific factors (controlled generation)

1 Compress Data

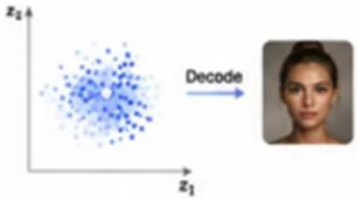
Learn a lower-dimensional representation that captrues the essential structure.

High-dimensional data

Low-dimensional latent space

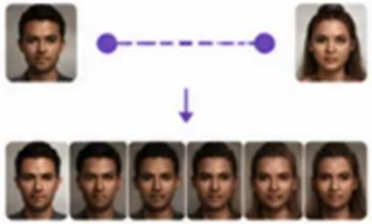
2 Enable Generation

Sample from the latent space and decode to generate new, realistic data.



3 Allow Interpolation

Smoothly morph between data points in latent space.



Interpolation (morphing)

4 Provide Interpretability

Latent factors often correspond to meaningful, interpretable attributes.

Latent Dimension	Semantic Attribute
z_1	Smile
z_2	Pose (yaw)
z_3	Age
z_4	Lighting
...	...

5 Generalize & Disentangle

Disentangled representations improve generalization and controlled manipulation.

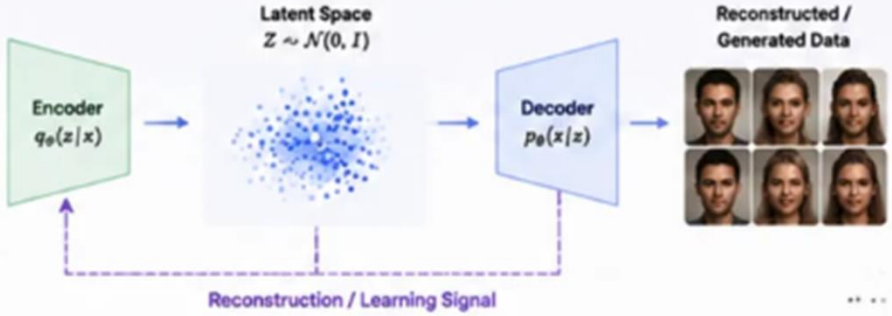
Change pose (z_2)

Change expression (z_1)

Latent Space (Z)

Each point in the latent space corresponds to a (potential) data point.

Latent Variable Model (e.g., VAE)



How Generation Works

1 Sample z from prior $\mathcal{N}(0, I)$

2 Decode z to generate x

3 Repeat to create diverse outputs

The Generative Process

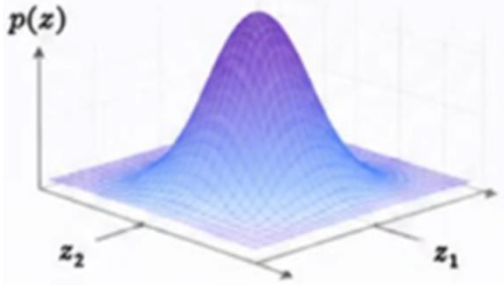
Latent Variable Models

In a latent variable model, data is generated by first sampling a latent variable z, then generating data x conditioned on z.

1 Sample a latent variable

$$z \sim p(z)$$

Sample z from a prior distribution $p(z)$.

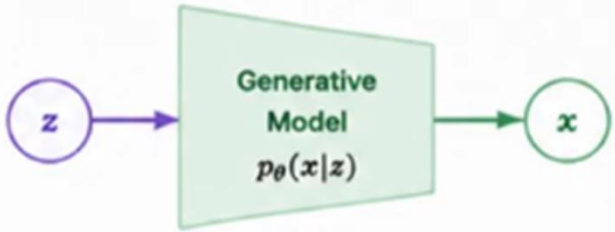


Example: $p(z) = \mathcal{N}(0, I)$

2 Generate data from z

$$x \sim p(x|z)$$

Generate data x from the conditional distribution $p(x|z)$.



Example (images):

...

3 Observed data

$$x$$

The generated data x is a sample from $p(x)$.

...

The marginal distribution over data is:

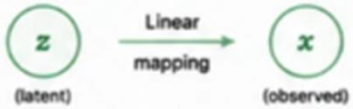
$$p(x) = \int p(x|z)\, p(z)\, dz$$

We integrate (marginalize) over all possible latent variables z to obtain the model distribution $p(x)$.

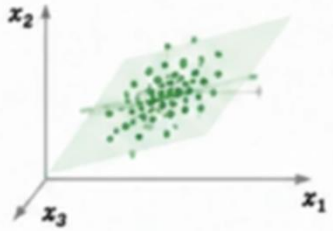
Latent variable models capture hidden structure by explaining observed data through **unobserved (latent) variables.** Different models make different assumptions and offer different strengths.

1 Factor Analysis

Linear latent factors



Assumes data are generated by a linear combination of latent factors plus Gaussian noise.



Example:
Discovering underlying attributes like height, weight, age from measurements.

2 Mixture Models

Clustered latent structure

Assumes data are generated from one of several components (clusters), each with its own distribution.

Example:
Customer segmentation, topic modeling, or clustering in high dimensions.

3 Variational Autoencoders (VAEs)

Deep generative models

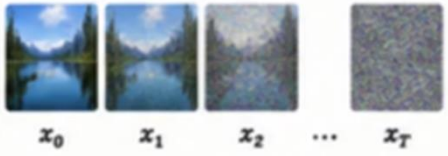
Uses a neural encoder to infer latent variables and a decoder to generate data. Learnable, continuous latent space.

Example:
Generating faces, digits, molecules, text, and more with smooth interpolations in latent space.

4 Diffusion Models

Hierarchical noise process

Data is gradually noised up to pure noise, then learned to be denoised step by step. Can be viewed as a latent variable model with a hierarchical (multi-step) latent variable process.



Example:
High-quality image generation, audio synthesis, video generation, and more.

What Is A Latent Variable Model?

- Latent variable models help researchers understand things that cannot be measured directly.
- They represent relationships between observable variables (measurable) and latent variables (inferred).
- An example of a latent variable is intelligence, which is assessed through test answers.
- The model assumes observable variables are influenced by hidden variables.
- Patterns among observed variables help estimate properties of latent variables.
- Factor analysis reduces many observed variables into fewer underlying factors.
- Item response theory analyzes discrete observed variables, like test scores.

Cont…

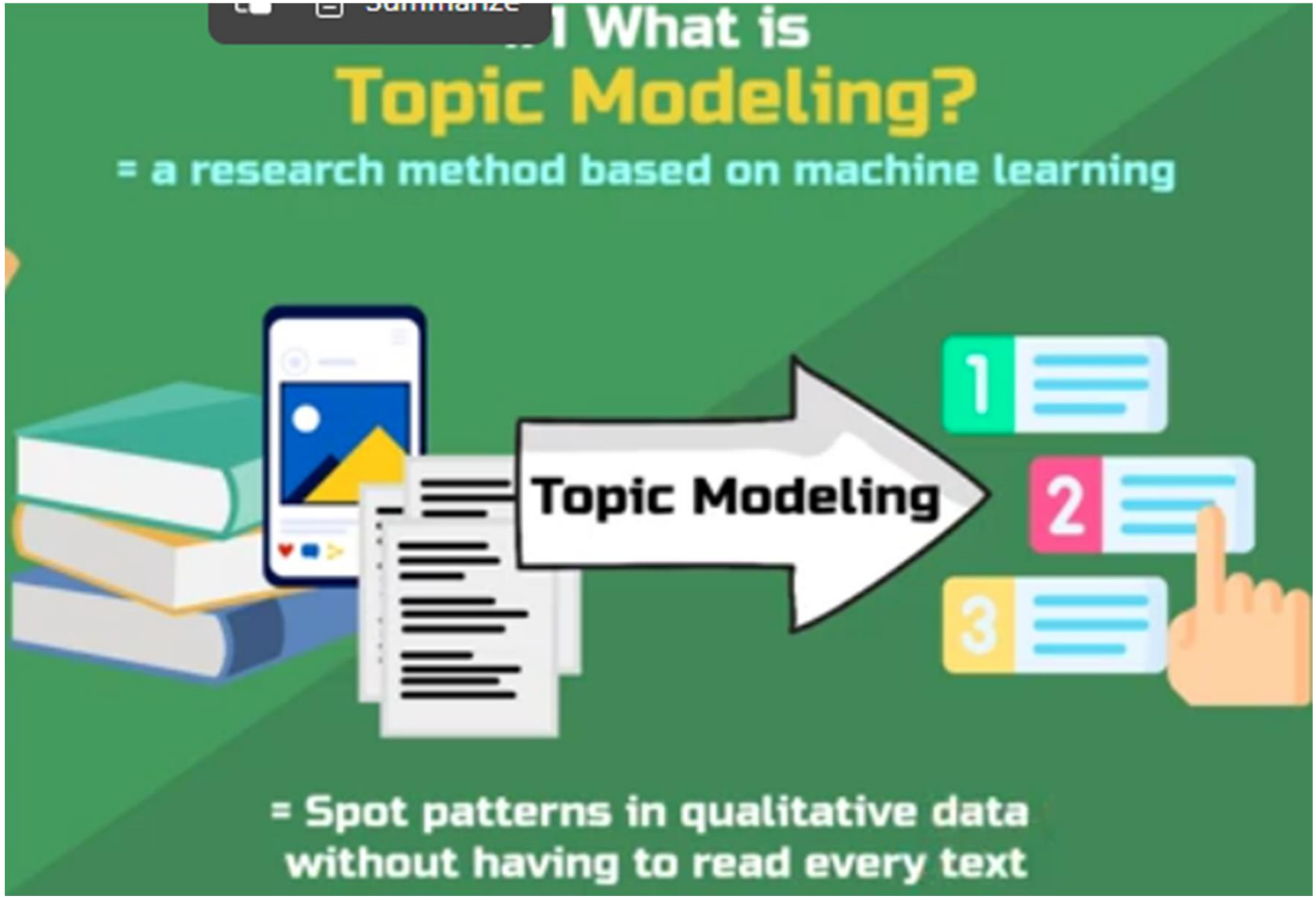
- Latent class analysis groups individuals based on categorical response patterns.
- Latent profile analysis deals with continuous observed variables and categorical latent variables.
- These models are used in psychology, education, marketing, and social sciences.
- In education, they assess a student's ability from test responses.
- In market research, they identify consumer attitudes from survey data.
- Multilevel latent variable models handle nested or grouped data, like students in schools.
- They provide accurate estimates of latent variables across different data levels.

Latent Variable Mixture Model

- A statistical technique for identifying unobserved subpopulations within a larger, heterogeneous population by analyzing observable patterns in data

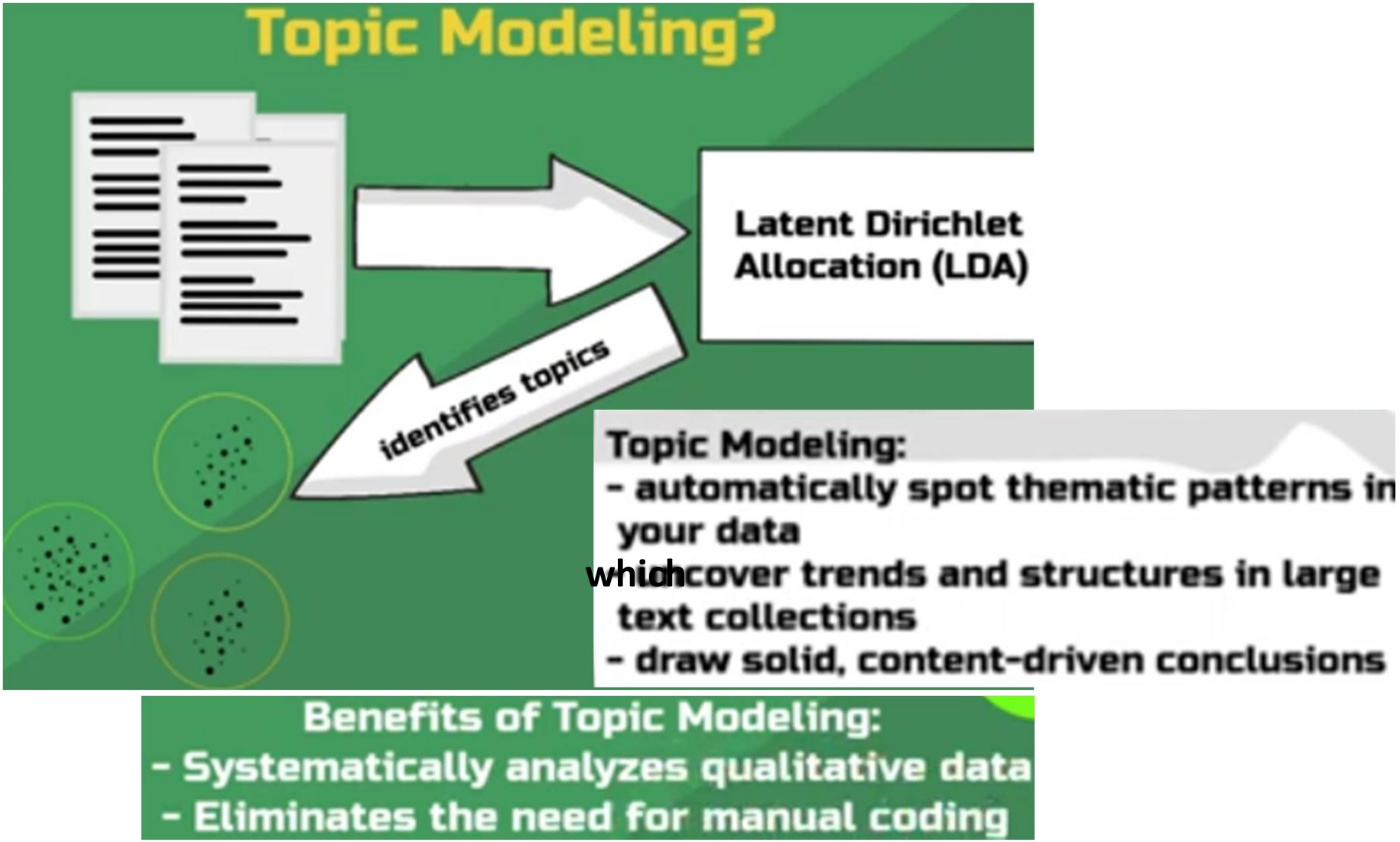
Procedure

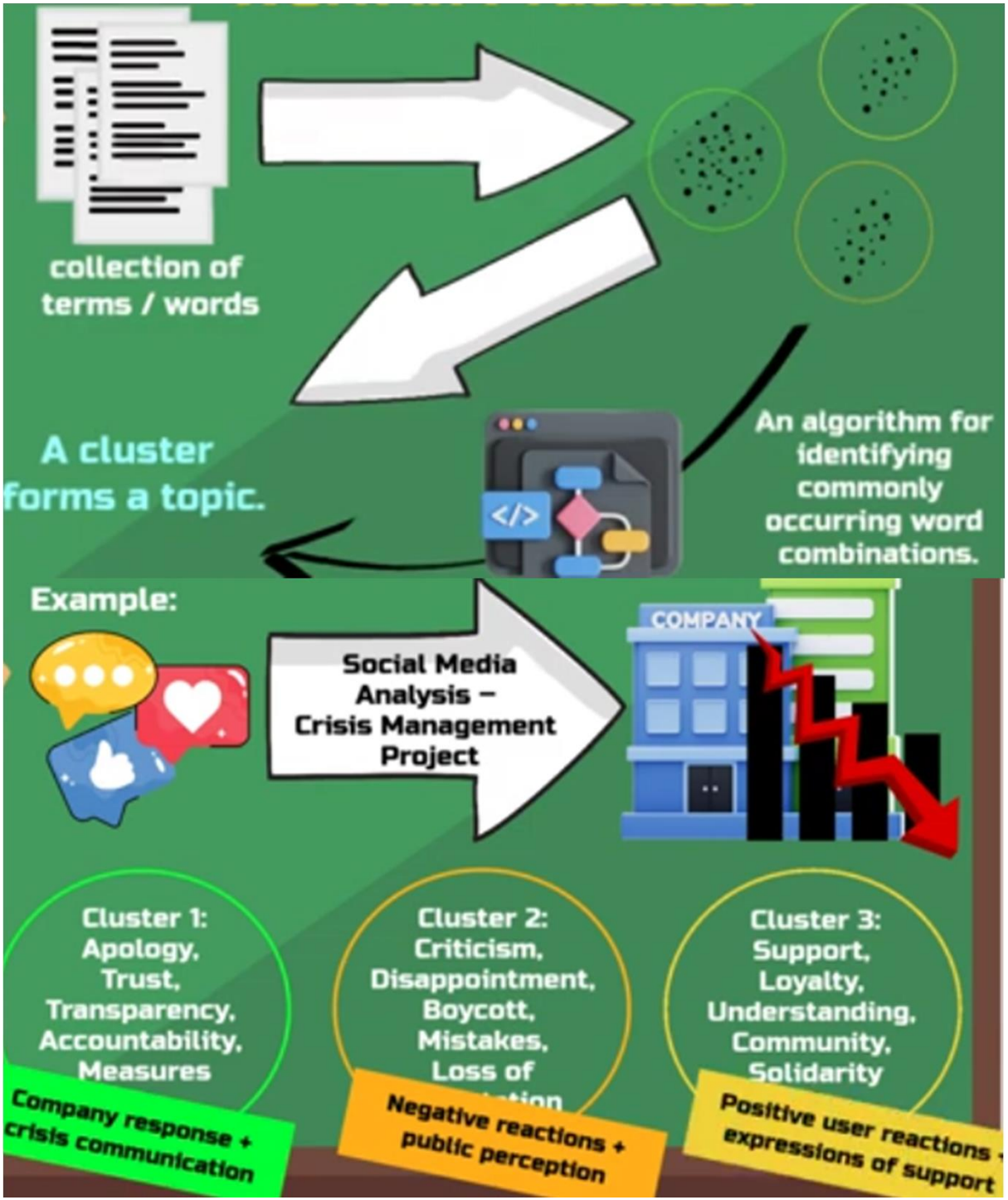
- Identifies heterogeneity: LVMM acknowledges that the population is not uniform but contains different subgroups, each with unique characteristics
- Uses observed data: The model uses observed indicators (e.g., survey responses, behavioral patterns) to infer the existence of unobserved (latent) groups
- Combines models: It merges aspects of latent class analysis (for categorical latent variables) and traditional factor models or structural equation models (for continuous latent variables)
- Classify individuals: The goal is to determine how many latent classes are present and to classify each individual into the most likely class based on their response patterns



Topic = statistical distribution of words that often appear together

You interpret what the clusters actually mean


Topic Modeling?
Latent Dirichlet Allocation (LDA)
identifies topics
Topic Modeling:
- automatically spot thematic patterns in your data
which cover trends and structures in large text collections
- draw solid, content-driven conclusions
Benefits of Topic Modeling:
- Systematically analyzes qualitative data
- Eliminates the need for manual coding


collection of terms / words
An algorithm for identifying commonly occurring word combinations.
A cluster forms a topic.
Example:
Social Media Analysis – Crisis Management Project
COMPANY
Cluster 1: Apology, Trust, Transparency, Accountability, Measures
Company response + crisis communication
Cluster 2: Criticism, Disappointment, Boycott, Mistakes, Loss of
Negative reactions + public perception
Cluster 3: Support, Loyalty, Understanding, Community, Solidarity
Positive user reactions + expressions of support

Step-By-Step Guide

Step 1: Data Collection

Be sure to save them in a readable format

Step 2: Text Preparation (Preprocessing)

- Remove punctuation

- Reduce words to their base form (lemmatization)

- Remove stop words (and, or, ...)

The quality of your results strongly depends on the quality of your preprocessing.

Step 3: Choosing the Right Algorithm

LDA: How many topics do you want to generate?

GUIDELINES

5 to 15 topics, depending on the dataset size and variety

Step 4: Evaluation + Interpretation

Which topics emerge from the word distributions?

Are there overlaps or clear distinctions?

Which topics are more dominant?

Which terms appear in multiple topics?

Step 5: Validating the Results

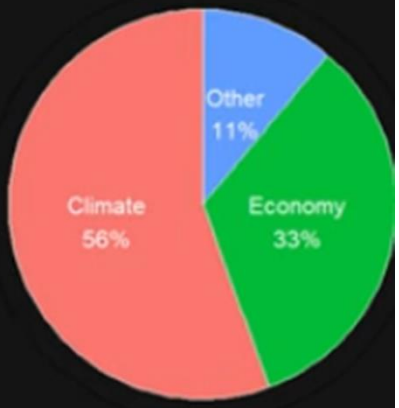
Development of topics over time:

- Break down by publication date
- Analyze how topics shift over the course of a crisis

Latent Dirichlet Allocation

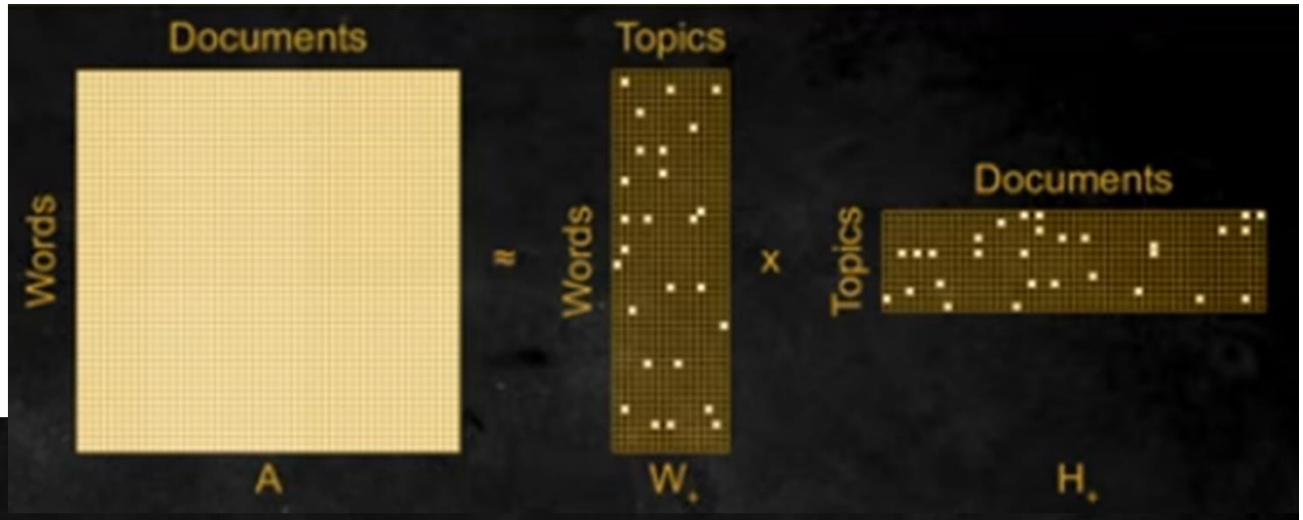
Goal – Topic Modelling

- Tons of documents (corpus)
  - Labeling and filtering
- Cluster words that tend to go together into "topics"
  - A form of dimensionality reduction
    - 20-100k words → 20-100 topics
  - Human / LLM labeling
- Label documents according to topics



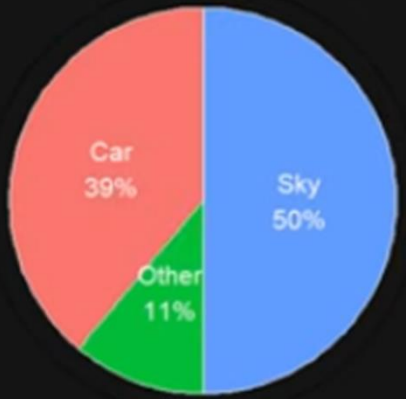
Economics	Climate	Love	…
Bank	Renewable	Heart	…
Interest	Global_Warming	Kiss	…
Inflation	Emissions	Passion	…
GDP	Carbon	Relationship	…
⋮	⋮	⋮	⋱

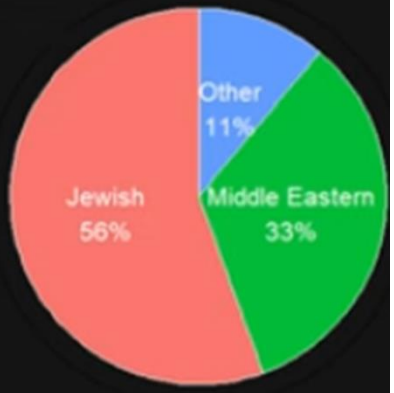
Cont…



Goal – Other uses

- Genetics
  - Assign individual (=document) to populations (=topics)
- Recommendation Systems
  - Finding favorite genres (=topics)
- Computer Vision
  - Assign topics to an image (=document)
- Etc.





LDA Model

- Each document $d = 1 \ldots D$ has a distribution of topics

$$\theta_d = (\theta_{d1}, \ldots, \theta_{dK}), \qquad \sum_k \theta_{dk} = 1, \qquad \theta_{dk} \geq 0 \; \forall k$$

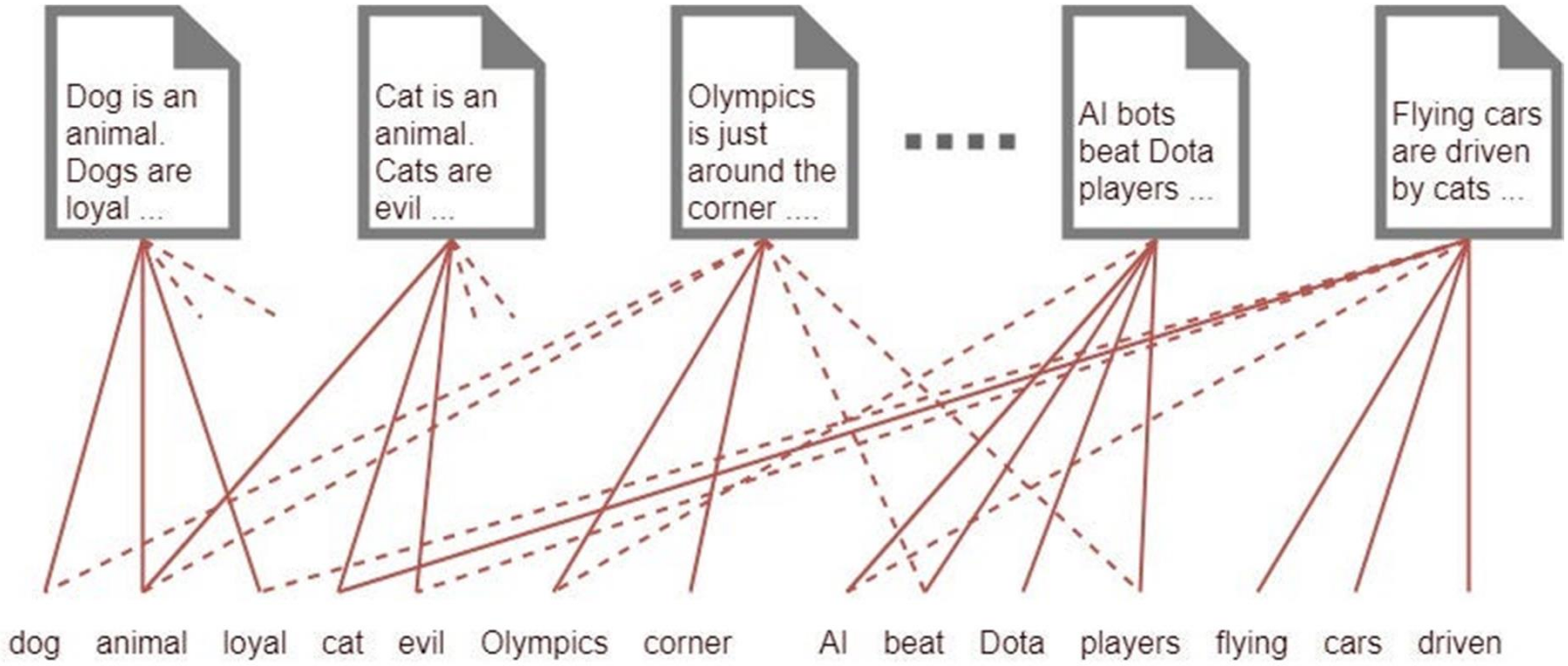
- Each topic $k = 1 \ldots K$ has a distribution of words

$$\beta_k = (\beta_{k1}, \ldots, \beta_{kV}), \qquad \sum_v \beta_{kv} = 1, \qquad \beta_{kv} \geq 0 \; \forall v$$

- $\boldsymbol{\theta} = (D \times K)$
- $\boldsymbol{\beta} = (K \times V)$

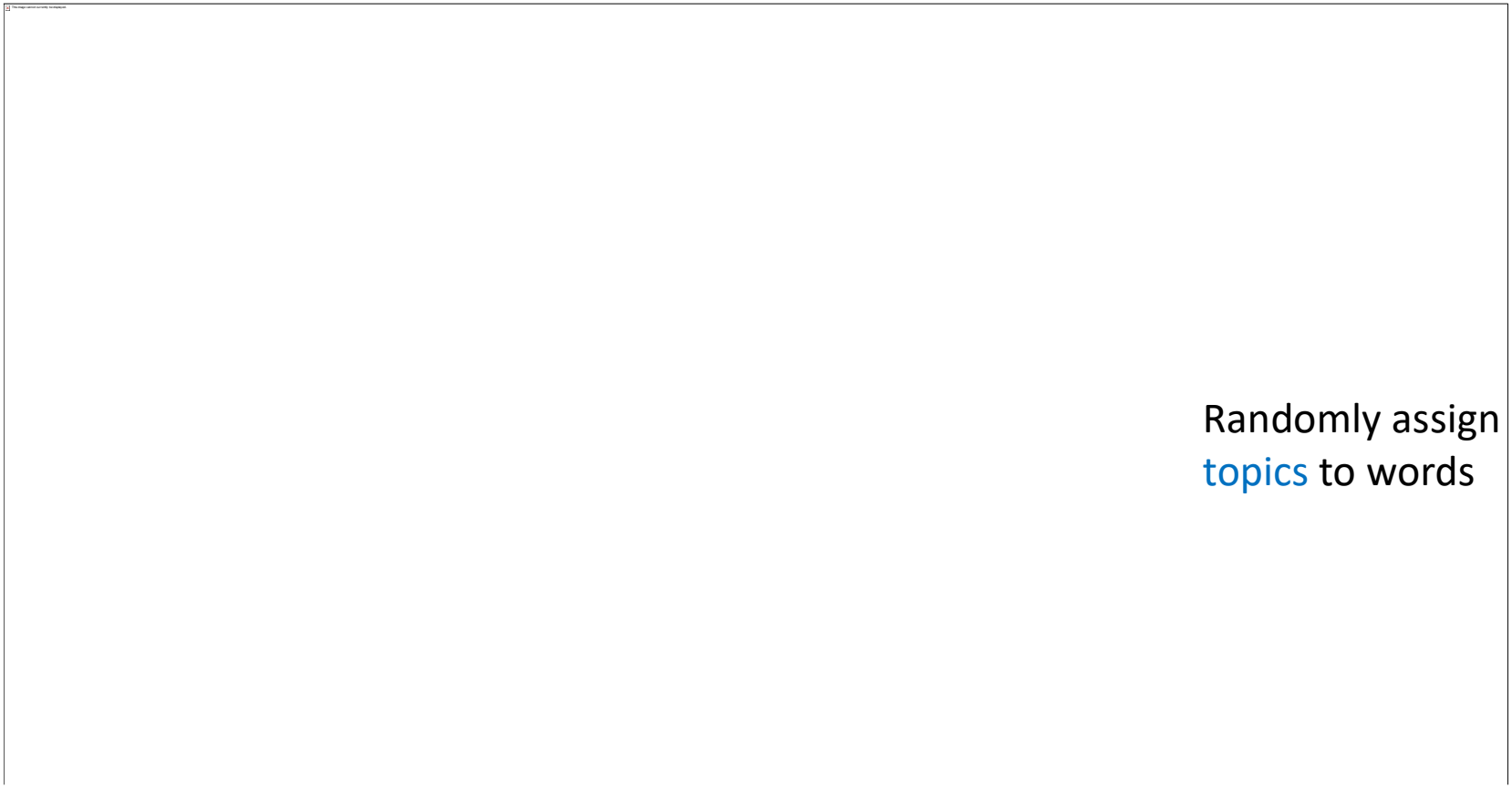
Latent Dirichlet Allocation

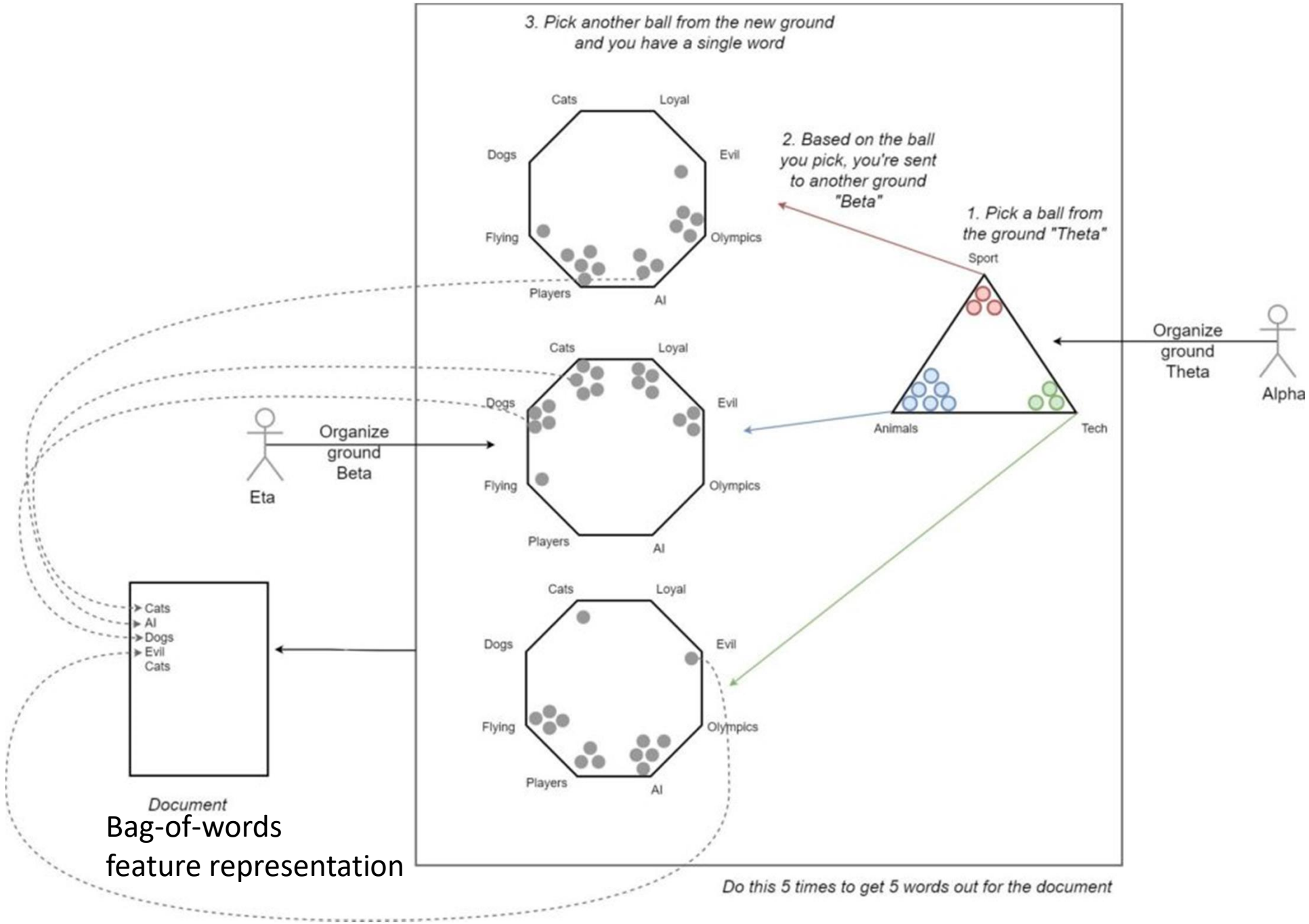
- Topic modeling technique - task of identifying topics that best describes a set of documents
- Goal - map all the *documents* to the *topics* such that the *words* in each document are mostly captured by those hidden topics
- Application – sentiment analysis, content recommendation, document clustering



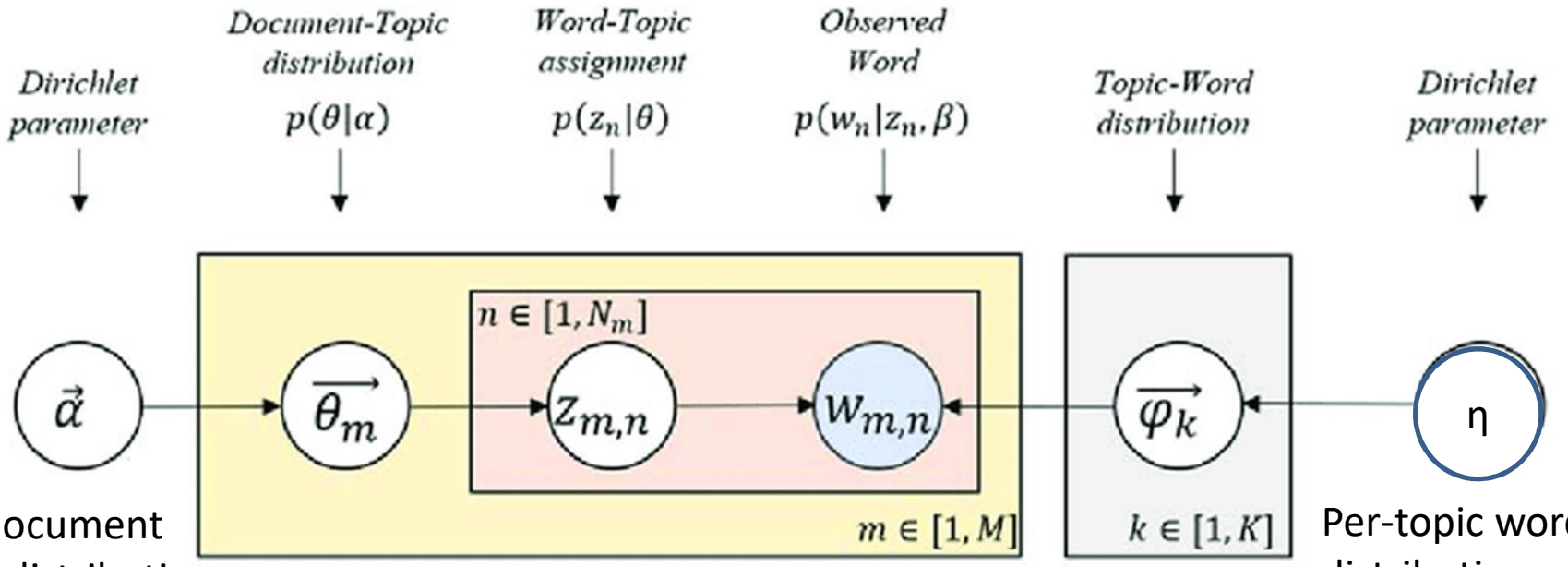
Cont…

- documents are probability distributions over latent topics
- topics are probability distributions over words




3. Pick another ball from the new ground and you have a single word
2. Based on the ball you pick, you're sent to another ground "Beta"
1. Pick a ball from the ground "Theta"
Cats
Loyal
Dogs
Evil
Flying
Olympics
Players
AI
Sport
Animals
Tech
Organize ground Theta
Alpha
Organize ground Beta
Eta
Cats
AI
Dogs
Evil
Cats
Document
Bag-of-words feature representation
Do this 5 times to get 5 words out for the document

Process



Per-document topic distribution

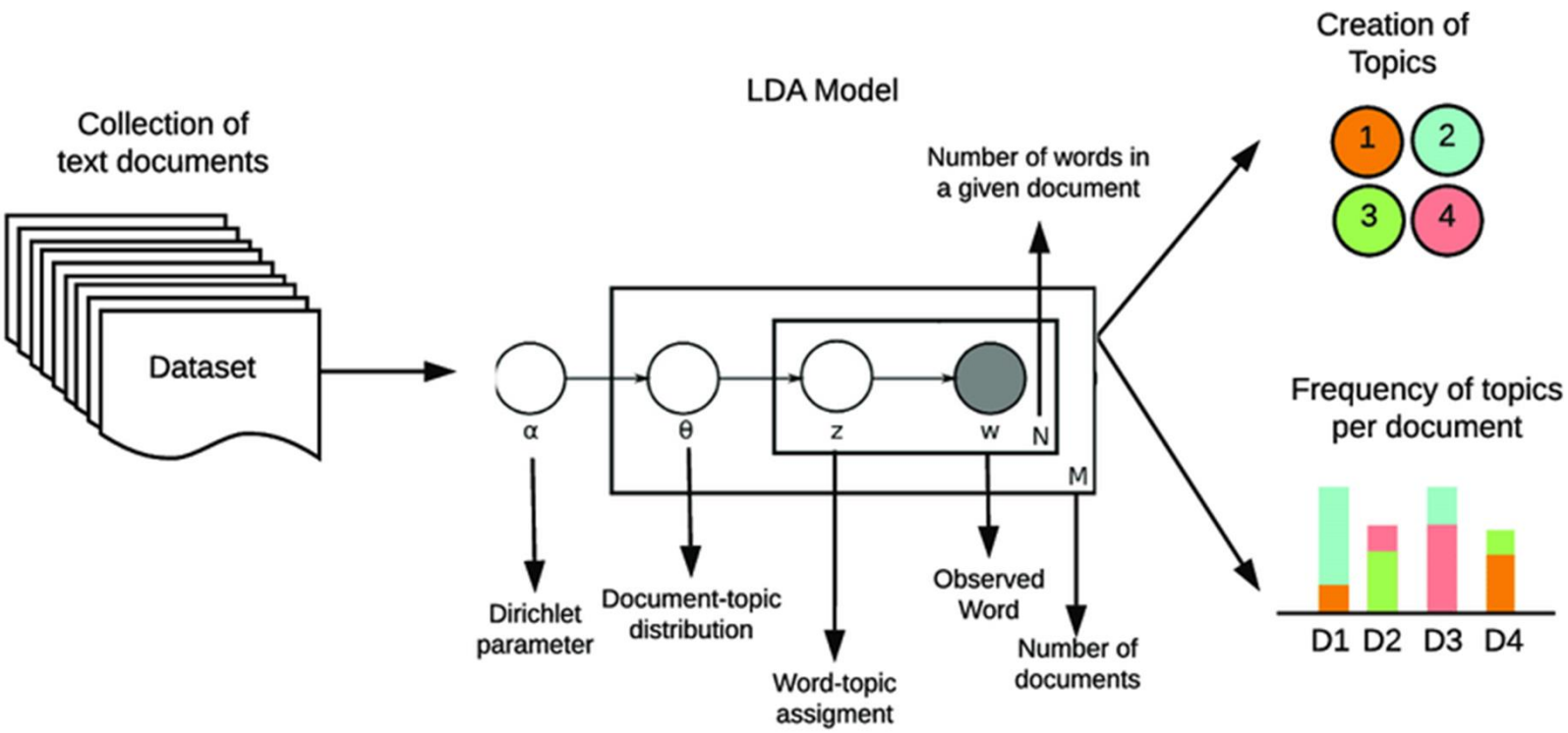
Per-topic word distribution

Iterate through documents, reassigning words to topics based on probabilities

- LDA assumes that all assigned topics are correct except for the current word
- It iterates over each document and word to compute two probabilities, p1 and p2, for each topic (k)
- p1 = p(k | D) represents the proportion of words in the document currently assigned to topic k
- p2 = p(w | k) is the proportion of assignments to topic k across all documents that come from the current word w
- Reassignment of word ‘w’ of the document ‘D’ to a new topic ‘k’ via the product probability of p1 * p2

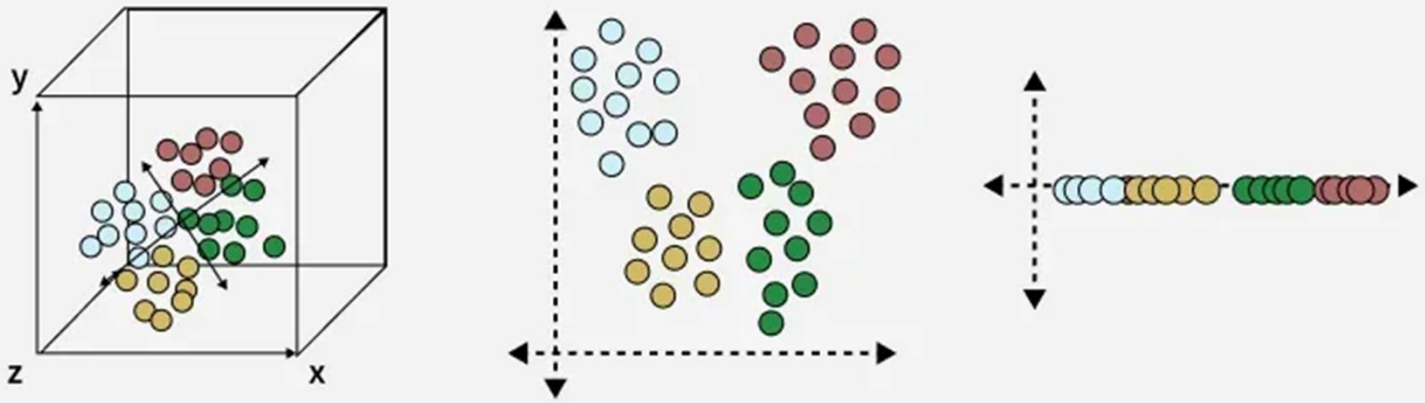
Convergence: Continue iterations until topics stabilize

Cont…


Collection of
text documents
Dataset
LDA Model
Number of words in
a given document
α
θ
z
w
N
M
Dirichlet
parameter
Document-topic
distribution
Word-topic
assigment
Observed
Word
Number of
documents
Creation of
Topics
1
2
3
4
Frequency of topics
per document
D1 D2 D3 D4

What is Dimensionality Reduction

Dimensionality Reduction is the process of reducing the number of input variables (features) in a dataset while preserving as much important information as possible.

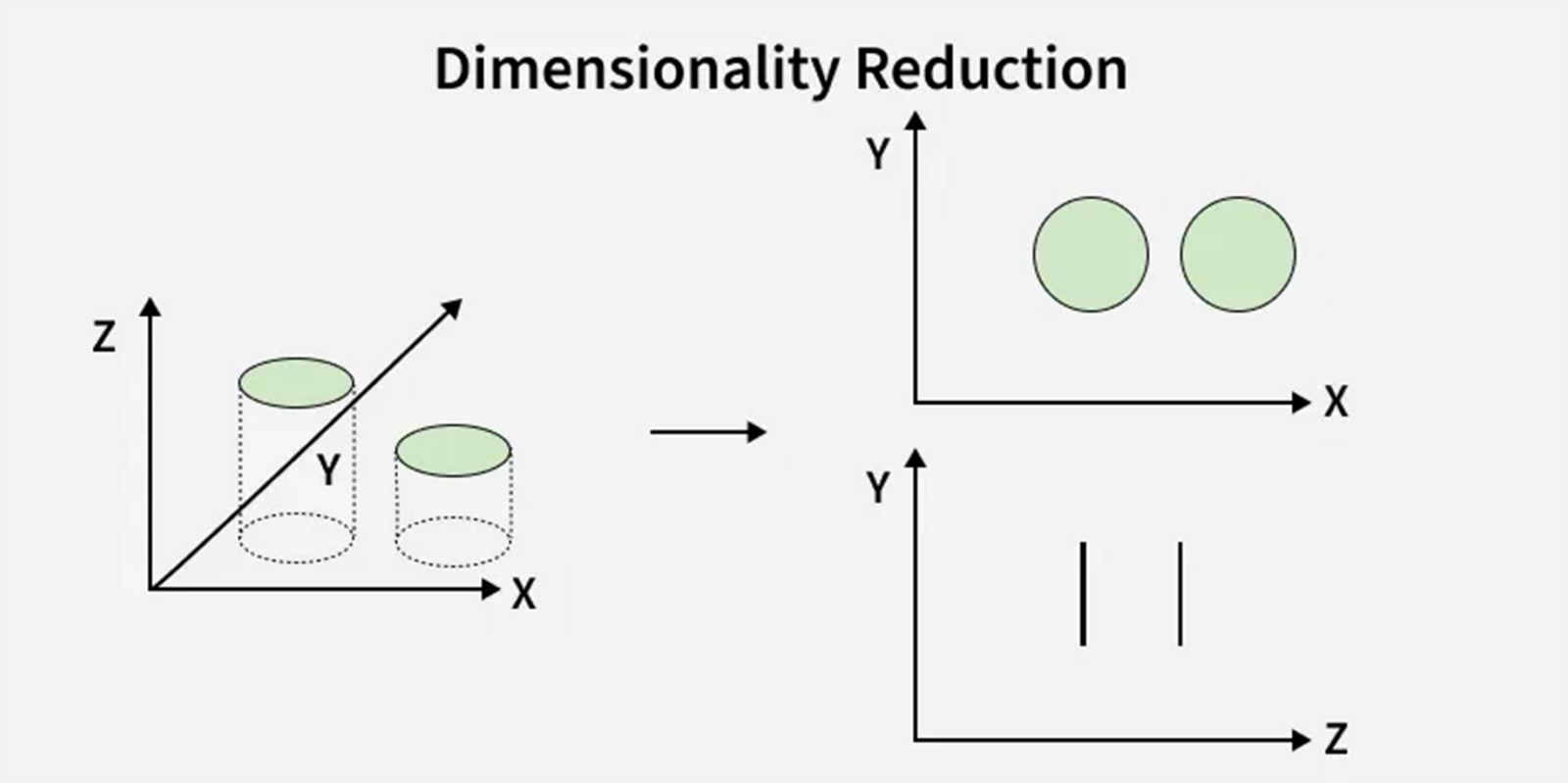


Advantages

- Faster computation
- Prevent overfitting
- Better visualization

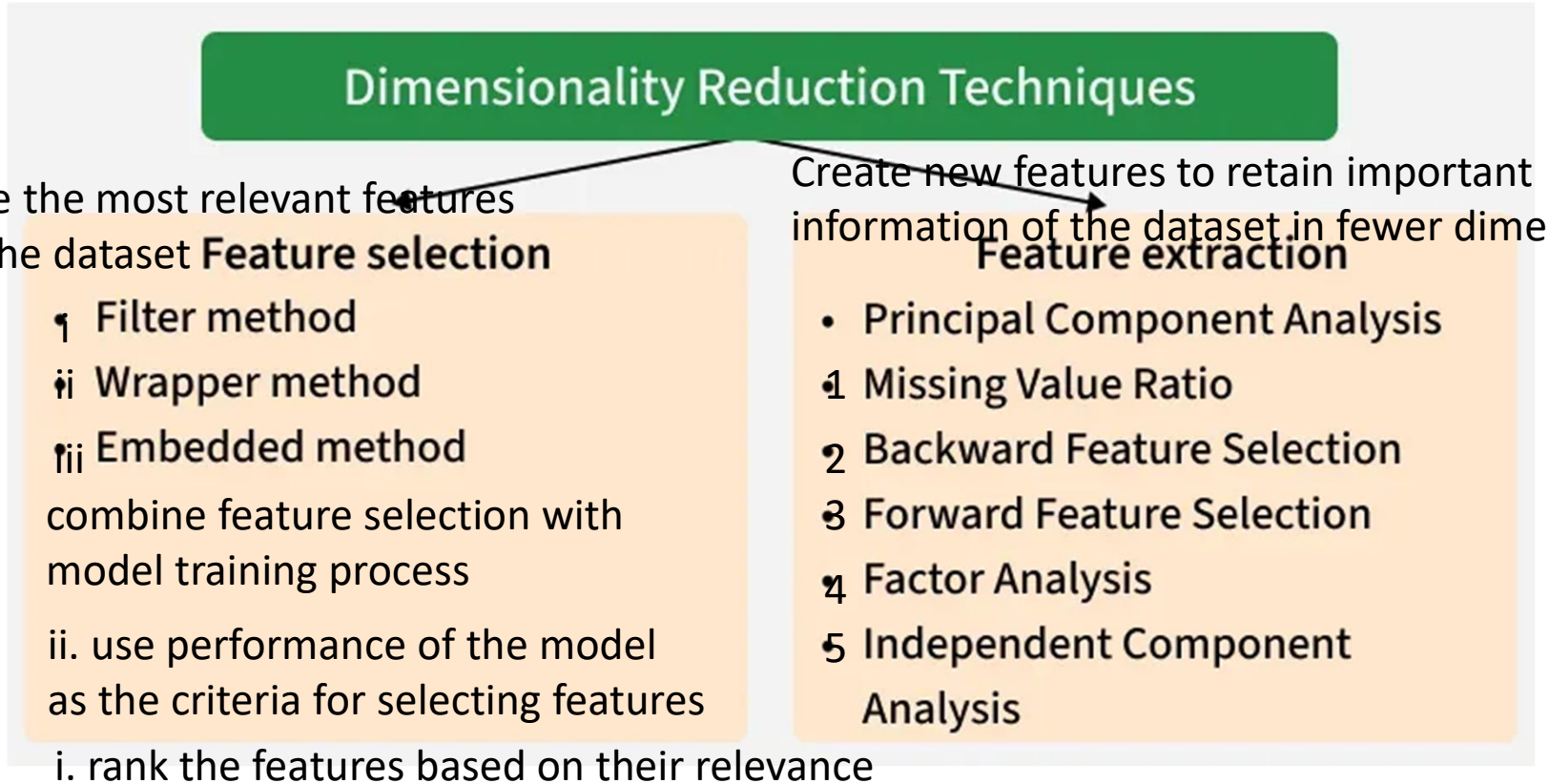
Disadvantages

- Data loss and reduced accuracy
- Choosing the right components is difficult



Applications

- Text categorization
- Image retrieval based on colour, texture, shape rather than just text descriptions
- Gene expression analysis
- Intrusion detection: helps to analyze user activity patterns to detect suspicious behaviors and intrusions



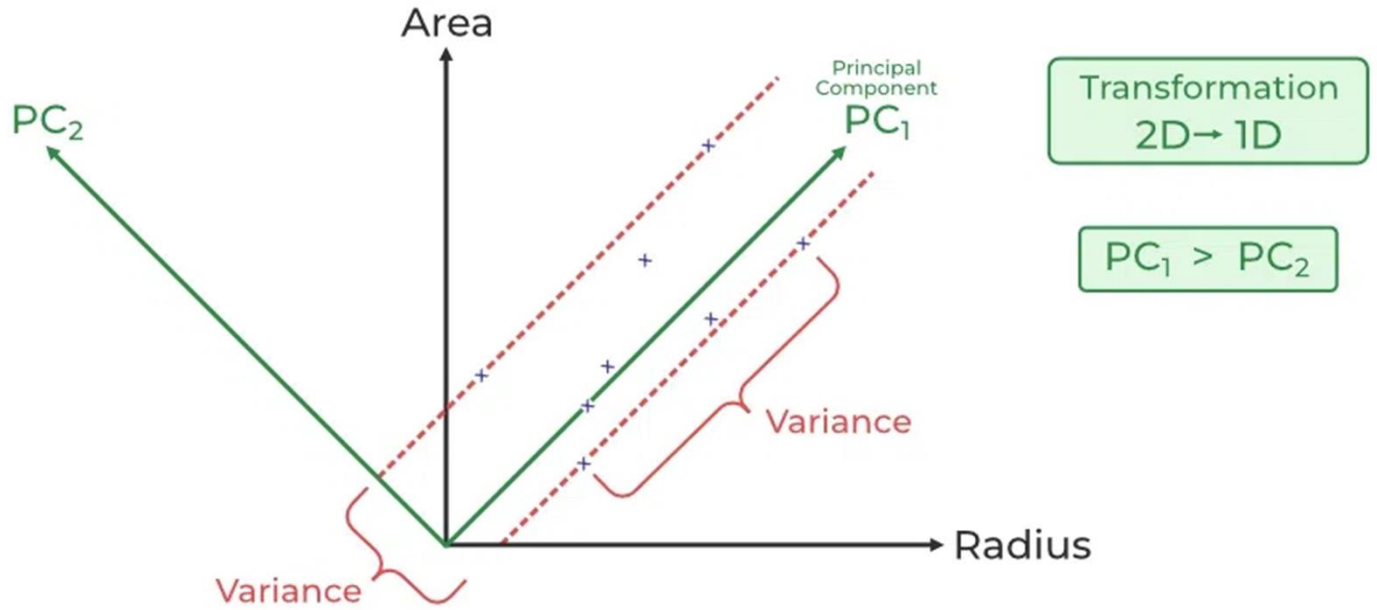
1. Variables with missing data beyond a set threshold are removed
2. Starts with all features and removes the least significant ones in each iteration
3. It begins with one feature, adds others incrementally
4. Groups variables by correlation
5. Identifies statistically independent components

Principal Component Analysis

- Standardize the data, $Z = (X - \mu)/\sigma$
- Calculate covariance matrix to see how feature relate to each other

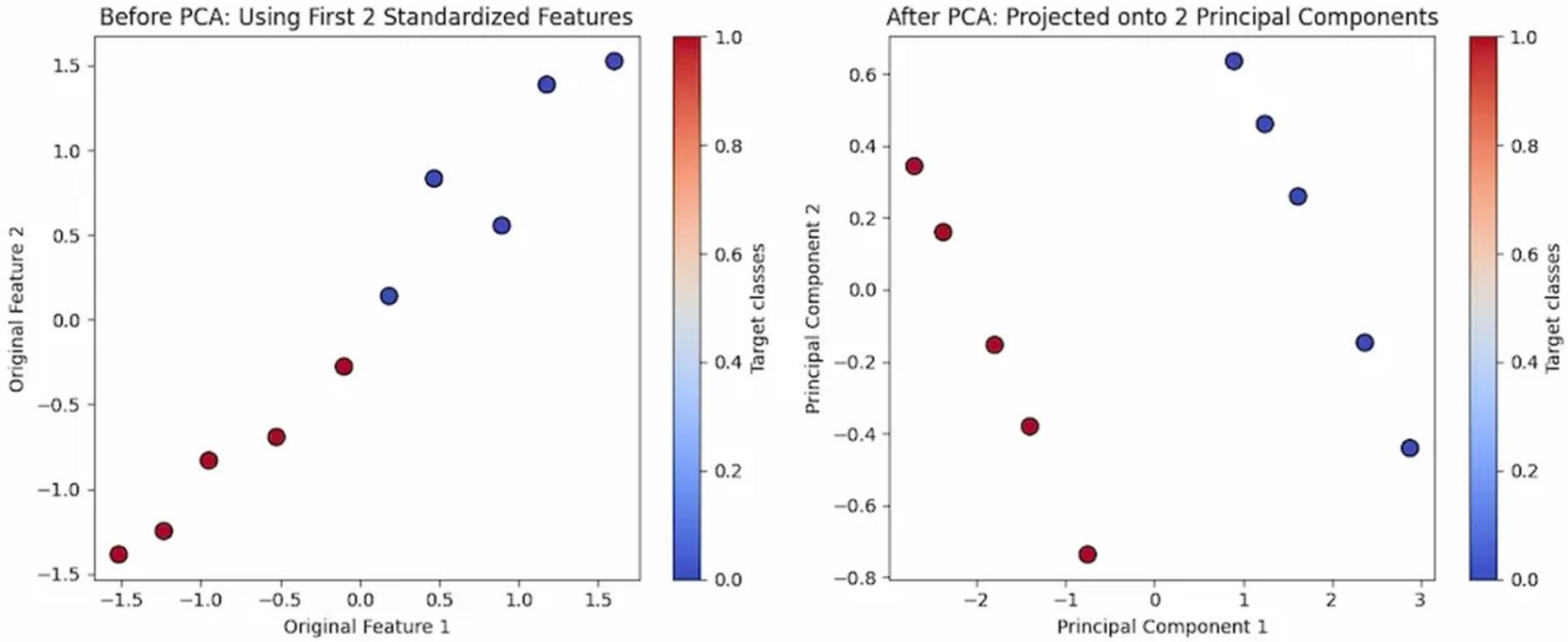
$$cov(x1, x2) = \frac{\sum_{i=1}^{n}(x1_i - \overline{x1})(x2_i - \overline{x2})}{n - 1}$$

- Find principal components
  - PCA identifies **new axes** where the data spreads out the most
  - **1st Principal Component (PC1):** The direction of maximum variance
  - **2nd Principal Component (PC2):** The next best direction, *perpendicular to PC1*
  - These directions come from the **eigenvectors** of the covariance matrix and their importance is measured by **eigenvalues**
  - Rank the directions by importance using eigenvalues
- **Select the top k components** that capture most of the variance
- **Transform the original dataset** by projecting it onto these top components



Transformation
2D→ 1D

$PC_1 > PC_2$



Advantages

- Creates new, uncorrelated variables to address issues when original features are highly correlated
- Eliminates components with low variance enhance data clarity
- Represents data with fewer components reduce storage needs and speeding up processing
- Identifies unusual data points by showing which ones deviate significantly in the reduced space

Disadvantages

- New components are combinations of original variables which can be hard to explain
- Requires proper scaling of data before application, otherwise results may be misleading
- Information loss
- Works best when relationships between variables are linear
- Can be slow and resource-intensive on very large datasets
- Using too many components or working with a small dataset may lead to overfitting of the model

Linear Discriminant Analysis

- Supervised machine learning technique that is used for both classification and dimensionality reduction
- Find directions (linear combination of features) that maximizes the distance between different class means while minimizing the variance within each class
- Separates two or more classes by converting higher-dimensional data space into a lower-dimensional space

Advantages: Simple and computationally efficient

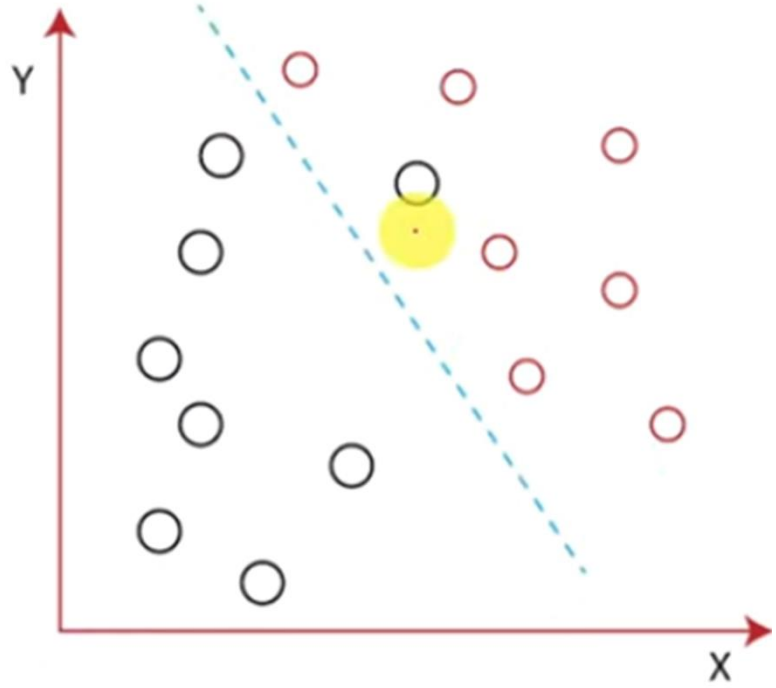
- Works well even when the number of features is much larger
- Can handle multicollinearity

Disadvantages: Assumes Gaussian distribution of data which may not always be the case

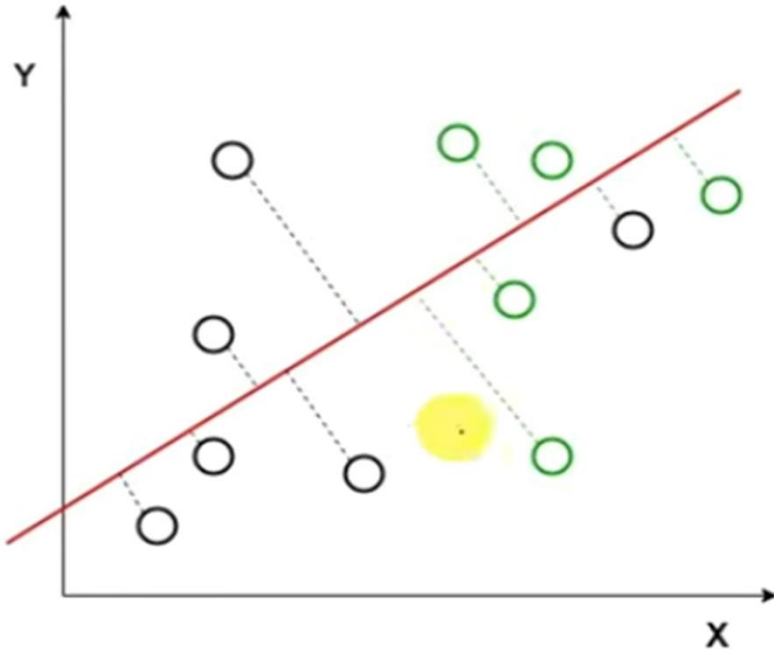
- Assumes equal covariance matrices for different classes which may not hold in all datasets
- Assumes linear separability which is not always true
- May not always perform well in high-dimensional feature spaces

Applications: Face recognition, medical diagnosis, customer identification

- Suppose we have two sets of data points belonging to two different classes that we want to classify.
- When the data points are plotted on the 2D plane, there's no straight line that can separate the two classes of the data points completely.



- Here, Linear Discriminant Analysis uses both the axes (X and Y) to create a new axis and projects data onto a new axis in a way to maximize the separation of the two categories and hence, reducing the 2D graph into a 1D graph.



- Two criteria are used by LDA to create a new axis:
  - Maximize the distance between means of the two classes.
  - Minimize the variation within each class.

Example

- Compute the Linear Discriminant projection for the following two dimensional dataset.

Samples for class ω_1 : $\mathbf{X}_1=(x_1,x_2)=\{(4,2),(2,4),(2,3),(3,6),(4,4)\}$

Sample for class ω_2 : $\mathbf{X}_2=(x_1,x_2)=\{(9,10),(6,8),(9,5),(8,7),(10,8)\}$

The classes mean are :

$$\mu_1 = \frac{1}{N_1}\sum_{x\in\omega_1} x = \frac{1}{5}\left[\begin{pmatrix}4\\2\end{pmatrix}+\begin{pmatrix}2\\4\end{pmatrix}+\begin{pmatrix}2\\3\end{pmatrix}+\begin{pmatrix}3\\6\end{pmatrix}+\begin{pmatrix}4\\4\end{pmatrix}\right] = \begin{pmatrix}3\\3.8\end{pmatrix}$$

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

Covariance matrix of the first class:

$$S_1 = \sum_{x \in \omega_1} \frac{(x-\mu_1)(x-\mu_1)^T}{\text{N-1}} = \left[\begin{pmatrix}4\\2\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]\left[\begin{pmatrix}4\\2\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]^T + \left[\begin{pmatrix}2\\4\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]\left[\begin{pmatrix}2\\4\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]^T$$
$$+\left[\begin{pmatrix}2\\3\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]\left[\begin{pmatrix}2\\3\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]^T + \left[\begin{pmatrix}3\\6\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]\left[\begin{pmatrix}3\\6\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]^T + \left[\begin{pmatrix}4\\4\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]\left[\begin{pmatrix}4\\4\end{pmatrix}-\begin{pmatrix}3\\3.8\end{pmatrix}\right]^T \Big/ \text{N-1}$$
$$= \begin{pmatrix}1 & -0.25\\ -0.25 & 2.2\end{pmatrix}$$

Similarly calculate the covariance matrix of the second class

Within-class scatter matrix:

$$S_w = S_1 + S_2 = \begin{pmatrix}1 & -0.25\\ -0.25 & 2.2\end{pmatrix} + \begin{pmatrix}2.3 & -0.05\\ -0.05 & 3.3\end{pmatrix}$$
$$= \begin{pmatrix}3.3 & -0.3\\ -0.3 & 5.5\end{pmatrix}$$

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

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

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

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

Between-class scatter matrix

- Find Eigen Values

$$S_W^{-1} S_B w = \lambda w$$

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

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

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

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

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

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

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

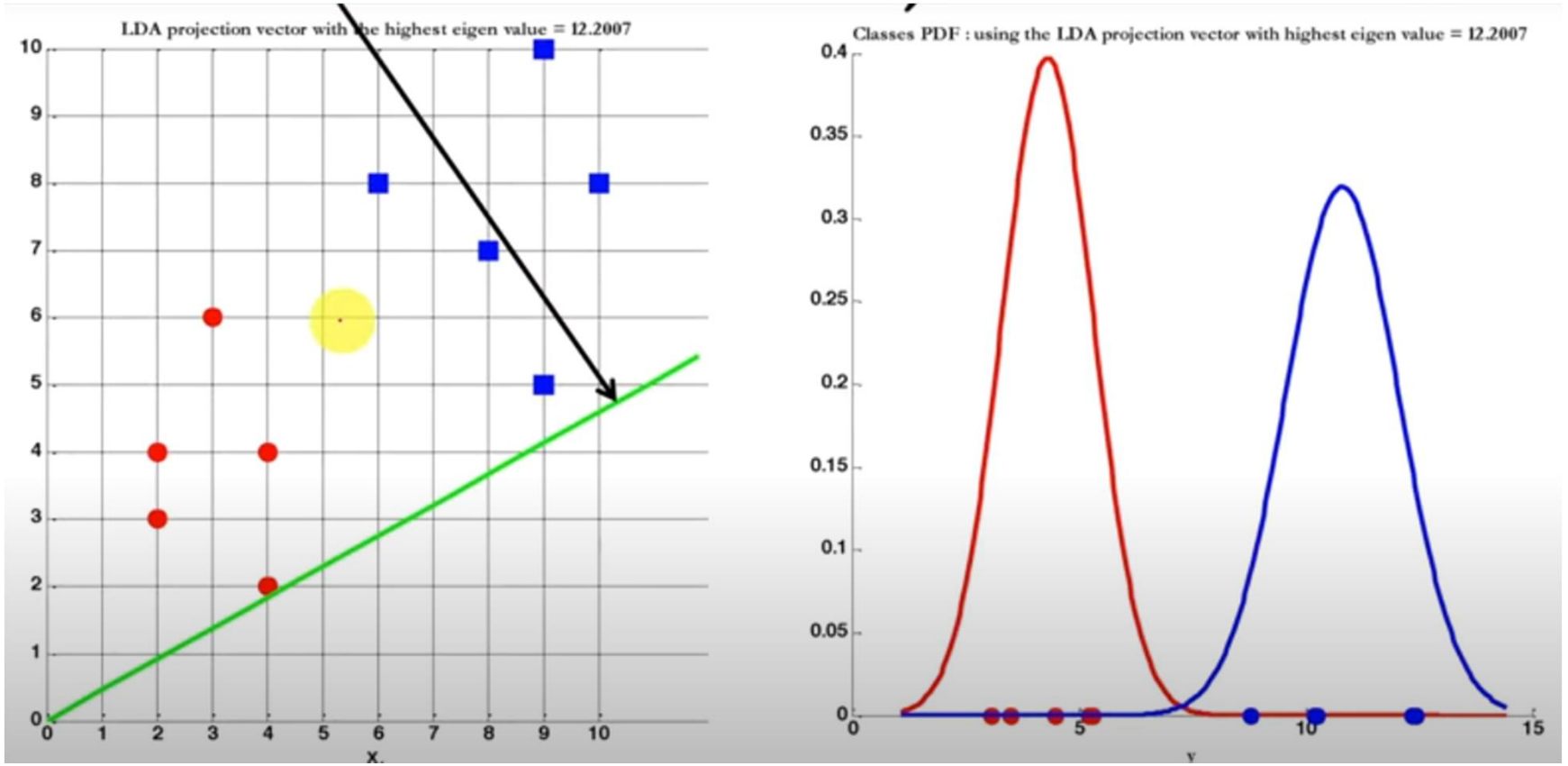
Find Eigen Vector

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

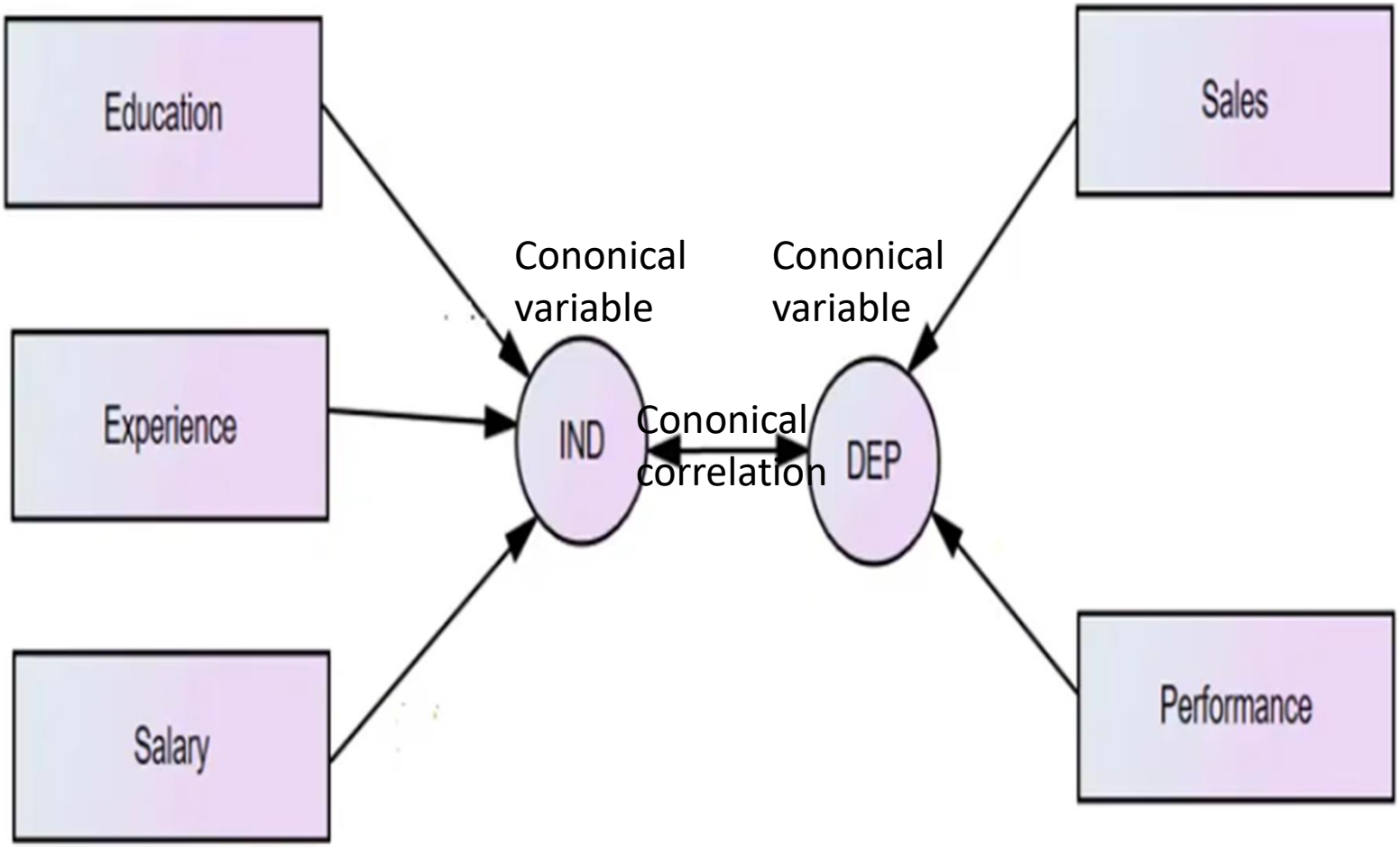
$$w_1 = \begin{pmatrix} -0.5755 \\ 0.8178 \end{pmatrix} \qquad w_2 = \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix} = w^*$$

Obtain LDA by taking dot product of best eigen vector w* and original data

X1	4	2	2	3	4	9	6	9	8	10
X2	2	4	3	6	4	10	8	5	7	8
1st LD	4.46	3.48	3.06	5.2	5.3	12.35	8.8	10.2	10.19	12.42


LDA projection vector with the highest eigen value = 12.2007
10
9
8
7
6
5
4
3
2
1
0
0 1 2 3 4 5 6 7 8 9 10
X1
Classes PDF : using the LDA projection vector with highest eigen value = 12.2007
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0 5 10 15
y

Cononical Correlation Analysis



Canonical: reduced to the simplest and most significant form possible, or to a criterion or standard of principle (canon)

Canonical Correlation Analysis

- Identifies and quantifies the relationships between two sets of variables
- Finds best combination of variables in each group to achieve high correlation
- Calculates canonical variables (to maximise correlation between sets) which are linear combination of original variables. Use SVD
- Applied in psychology, economics, medicine, marketing
- Sample data – set X: study habits

Multiview data

Student	Hours Studied (x_1)	Attendance (%) (x_2)
A	5	80
B	3	60
C	4	90

- Set Y: performance

Student	Exam Score (y_1)	Project Score (y_2)
A	75	80
B	50	60
C	85	90

Cont…

- Standardize the data (subtract mean and divide by standard deviation)

X = [[1, 0.22],
[-1, -1.09],
[0, 0.87]]

Y = [[0.28, 0.22],
[-1.11, -1.09],
[0.55, 0.87]]

- Compute covariance matrix (S_{xx}, S_{yy}, S_{xy}) – Since data is standardized, covariances are just correlations
- $S_{xy} = \frac{1}{n-1} X^T Y = \begin{bmatrix} 0.92 & 0.96 \\ 0.96 & 1.00 \end{bmatrix}$

Cont…

- Solve eigen value problem: $S_{xx}^{-1}S_{xy}S_{yy}^{-1}S_{yx}a=\lambda a$

$$b_i = S_{yy}^{-1}S_{yx}a_i/\sqrt{\lambda_i}$$

First canonical correlation (eigen value) ≈ **0.995** -> strong relationship between **study habits** and **performance**.

Canonical weights (eigen vectors):

a = [0.71, 0.71]

b = [0.70, 0.71]

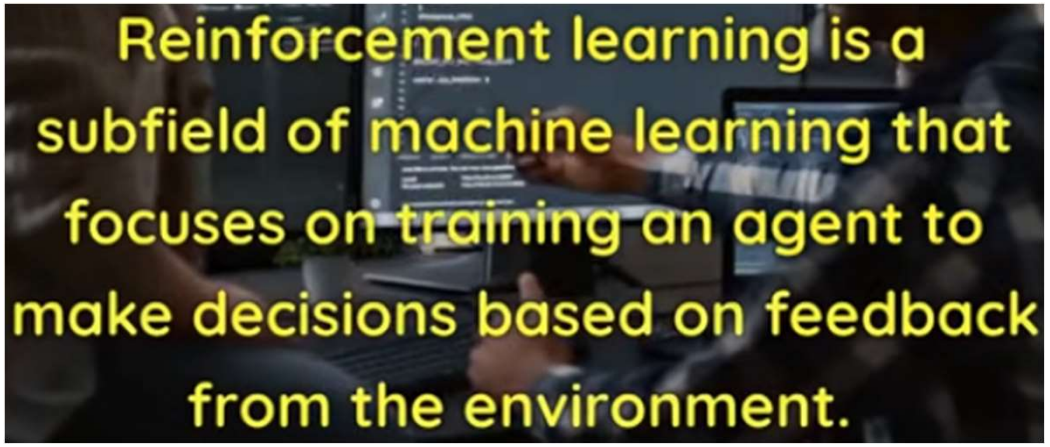
Show how much each original variable contributes to the relationship

- Compute canonical variables
- U = Xa
- V = Yb

U and V summarise X and Y

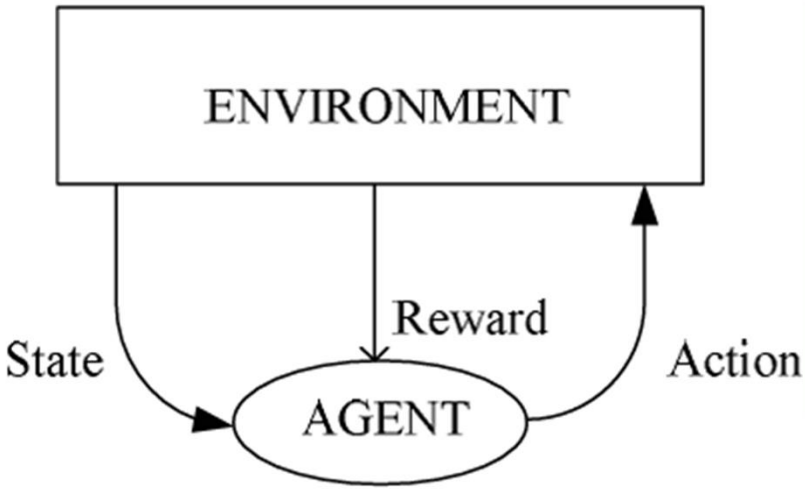
- Choose the combination of weights so that the correlation between **U** and **V** is maximized
- Both hours studied and attendance contribute equally
- Both exam and project scores are equally related to the study habits

Reinforcement Learning



Reinforcement Learning

- Learning with a critic - it does not tell us what to do in advance. E.g. Game
- Get the reward only when the complete sequence of actions is carried out
- Learning agent learns the best sequence of actions to solve a problem
- It learns to generate an internal value for the intermediate states or actions in terms of how good they are in leading us to the goal and getting the real reward
- Once such an internal reward mechanism is learned, the agent can just take the local actions to maximize it



For each correct action, the agent gets positive feedback, and for each incorrect action, the agent gets negative feedback or penalty.

`Cont...

- Since there is no labeled data, so the agent is bound to learn by its experience only.
- Reinforcement Learning is used to solve specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.
- There are two types of reinforcement learning - **positive and negative.**
- **Positive reinforcement learning** is a recurrence of behaviour due to positive rewards.
- Rewards increase strength and the frequency of a specific behaviour.
- This encourages to execute similar actions that yield maximum reward.
- Similarly, in **negative reinforcement learning**, negative rewards are used as a deterrent to weaken the behaviour and to avoid it.
- Rewards decreases strength and the frequency of a specific behaviour.

Elements

- Policy: mapping from perceived states of the environment to actions to be taken. It defines the behaviour of learning agent at a given time
- Reward signal: It is a numerical score received to the agent by the environment. It defines the goal of the problem
- Value function: total amount of reward an agent can expect to accumulate over the future. It specifies what is good in the long run
- Model: It is used for planning

K-armed (Multi-armed) Bandit Problem

- Gambler stands in front of a slot machine with *K* levers
- Action: pull one of the levers
- Reward: win a certain amount of money
- Task: which lever to pull to maximize the reward
- Q(*a*): value of actions *a*. Initially, Q(a)=0 for all *a*
- On action *a*, we get reward $r_a \geq 0$. If rewards are deterministic, we always get the same r_a. In such case $Q(a) = r_a$. To *exploit* we can keep choosing it. To *explore*, we can choose different actions

choose $a*$ if $Q(a*) = \max_a Q(a)$

Exploration vs Exploitation

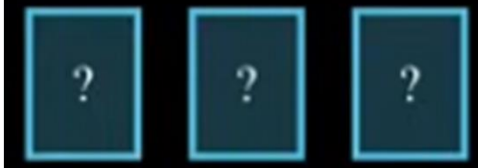
Explore

Try different options
to learn more

VS

Exploit

Stick with the best
known option



Too much exploration → Wasted opportunities

Too much exploitation → Miss better options

We need the right balance!

Cont…

- If rewards are stochastic, we get different rewards for the same action at each time
- p(r|a): the amount of the reward, $Q_t(a)$: average of all rewards received for the action *a* before time *t*
- Updated value

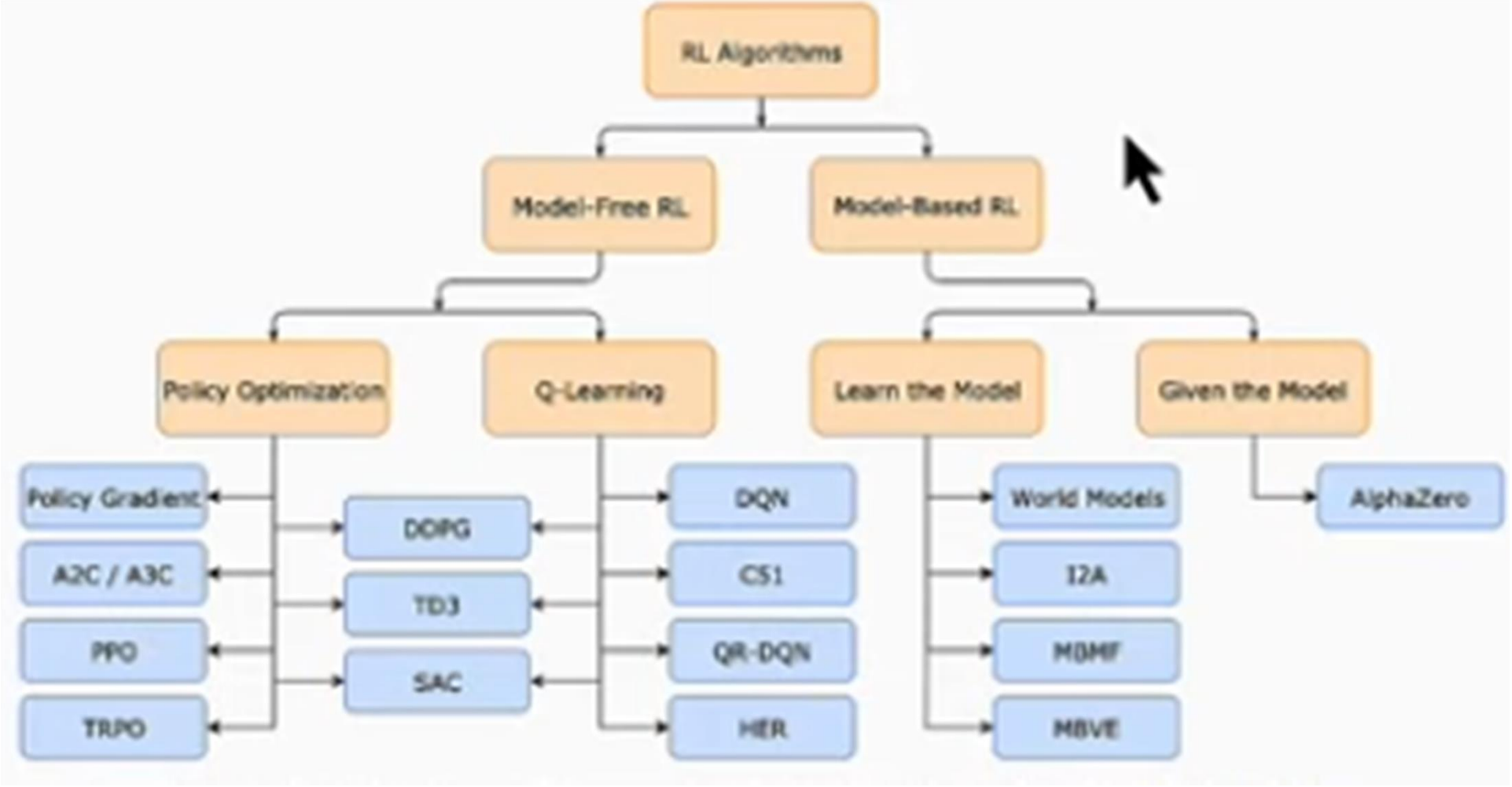
$$Q_{t+1}(a) \leftarrow Q_t(a) + \eta[r_{t+1}(a) - Q_t(a)]$$

η: learning parameter

Applications

- Online advertising: balancing the exploration of new ads with the exploitation of ads that have shown high hit count
- Clinical trials: allocate patients to different treatment arms by efficiently learning which treatments are most effective
- Recommender systems: to suggest products, movies, or content to users by continuously learning and adapting to user preferences

Reinforcement Learning Algorithms

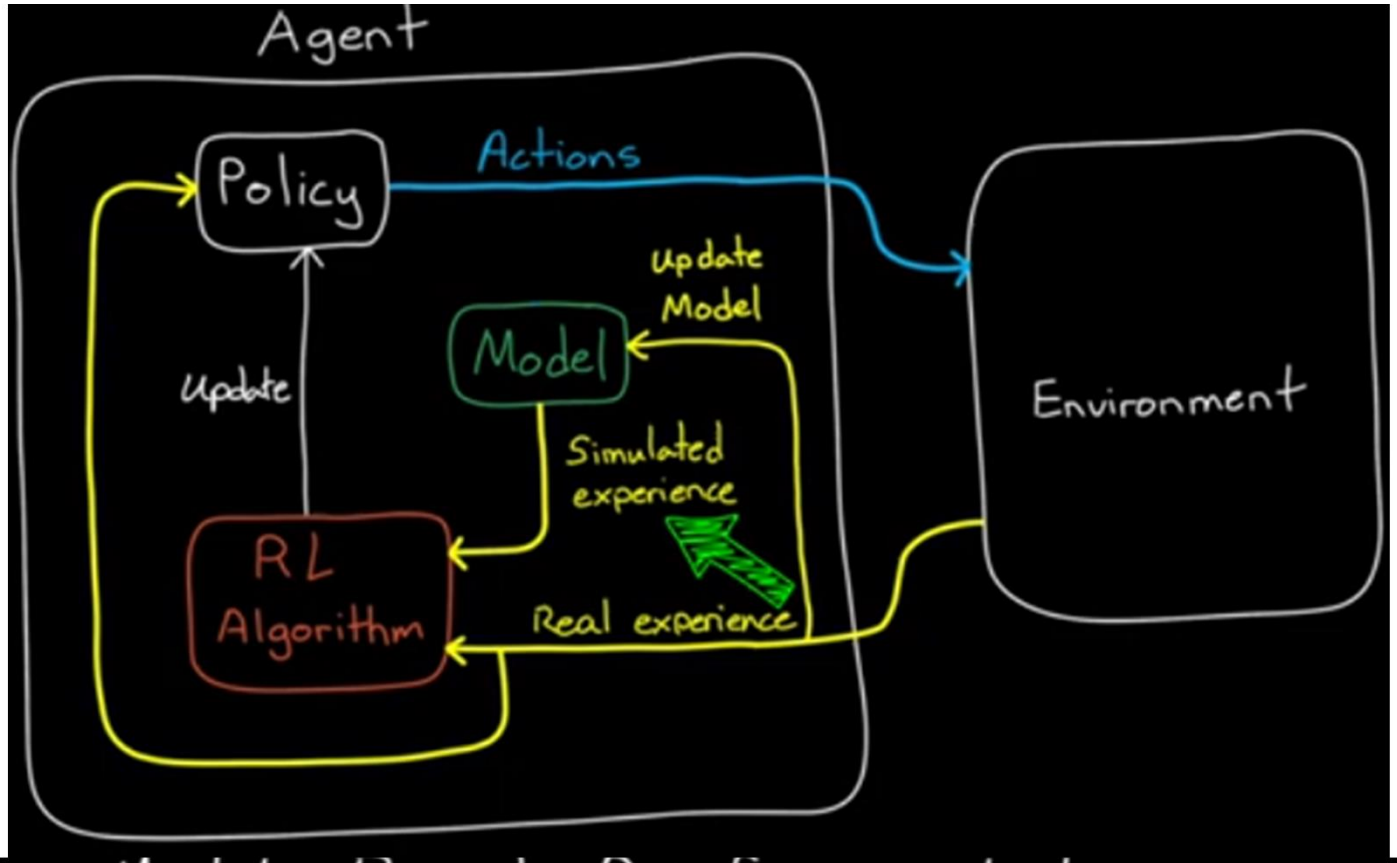


The system learns and predict using a model.

- The system uses a model
- It learns when the model is trained with data
- The model performance is evaluated
- Uses the model to make predictions

Model-Based Learning

- The model can be stored/saved
- There is no need use all data
- It takes less space than all data
- The model is faster making predictions
- Needs to trains and evaluate a model

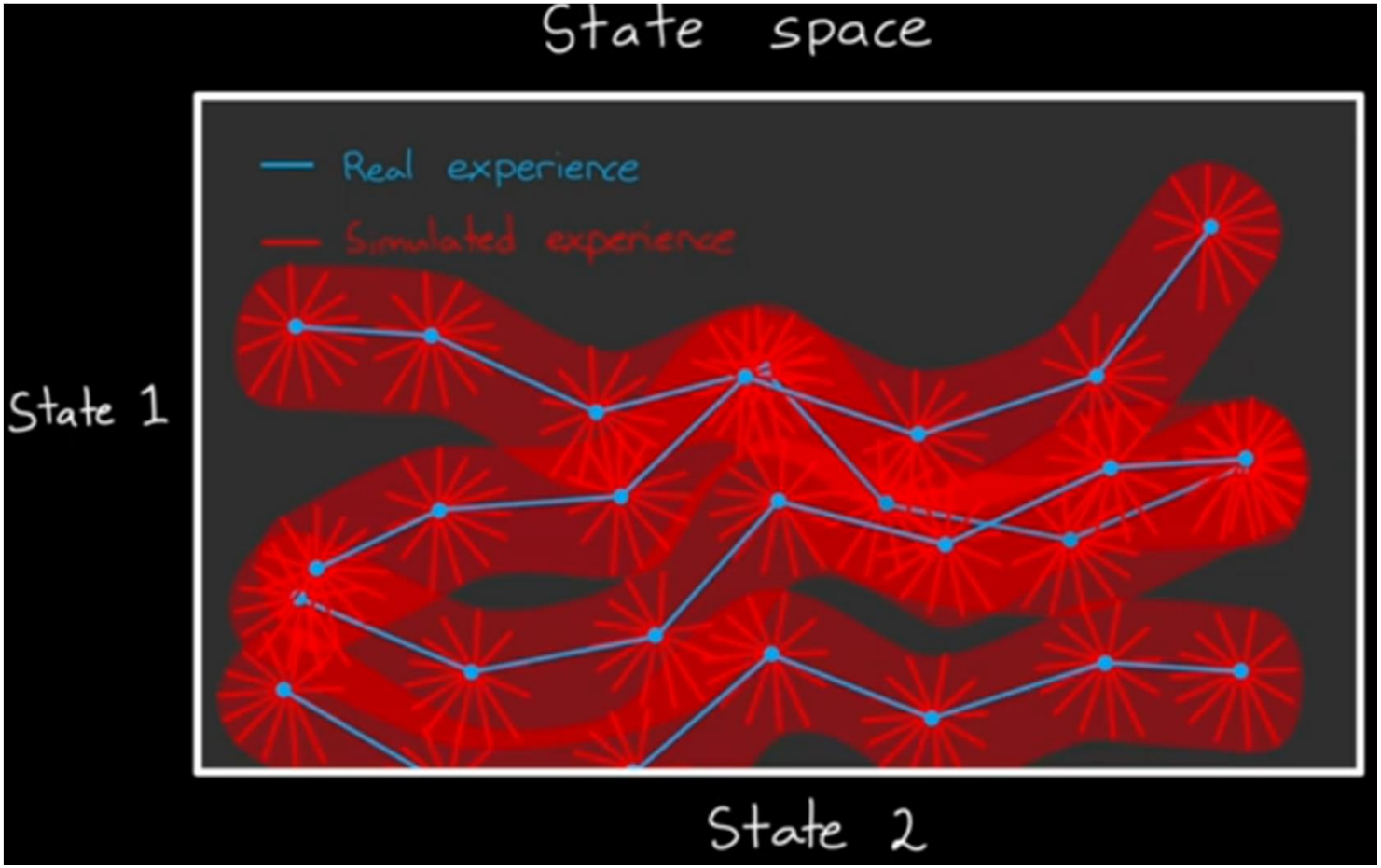


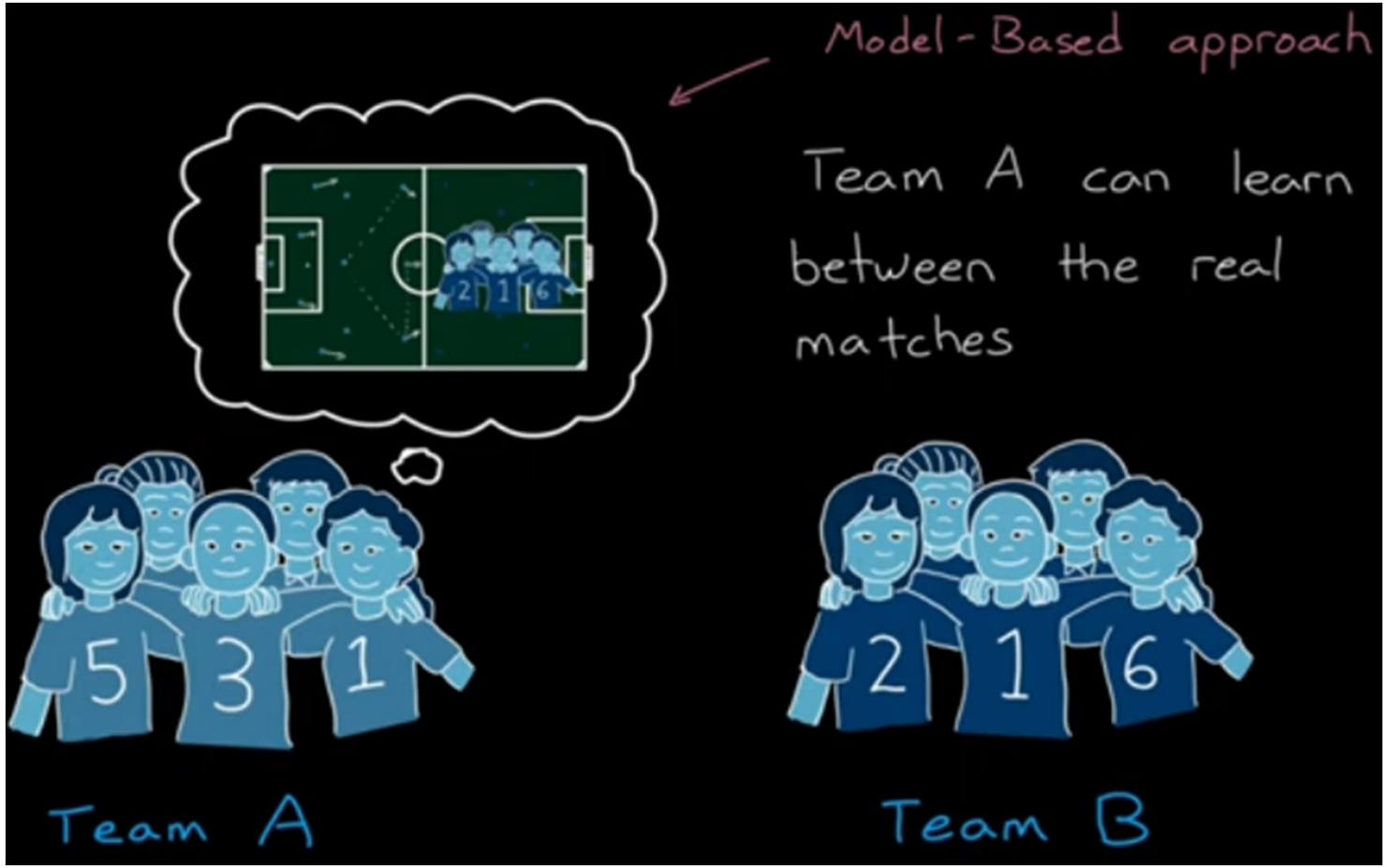
Model - Based Reinforcement Leaning

Create a model

often requires interacting with the environment

Learn the optimal policy


State space
Real experience
Simulated experience
State 1
State 2


Model-Based approach
Team A can learn between the real matches
5
3
1
2
1
6
Team A
Team B

Model Based Learning

- Agent constructs an internal model of the environment's dynamics and uses it to simulate future states, predict rewards and optimize actions efficiently

Components

- **Model of the Environment:** a predictive model which can forecast the next state and rewards based on the current state and action
- **Planning Algorithm:** Once a model is learned, a planning algorithm evaluates the model to decide the optimal sequence of actions

Process

- **Model Learning:** The agent collect experience by interacting with the environment and then use these experiences to learn a model and predict future states and rewards
- **Model based planning**: the agent uses the model to plan future steps without interacting with the real world. Use MCTS or dynamic programming to identify optimal actions
- **Policy Optimization:** The agent uses the results from planning to optimize its policy
- **Continuous Learning:** The model is updated regularly as the agent gathers new experiences

Cont…

Advantages

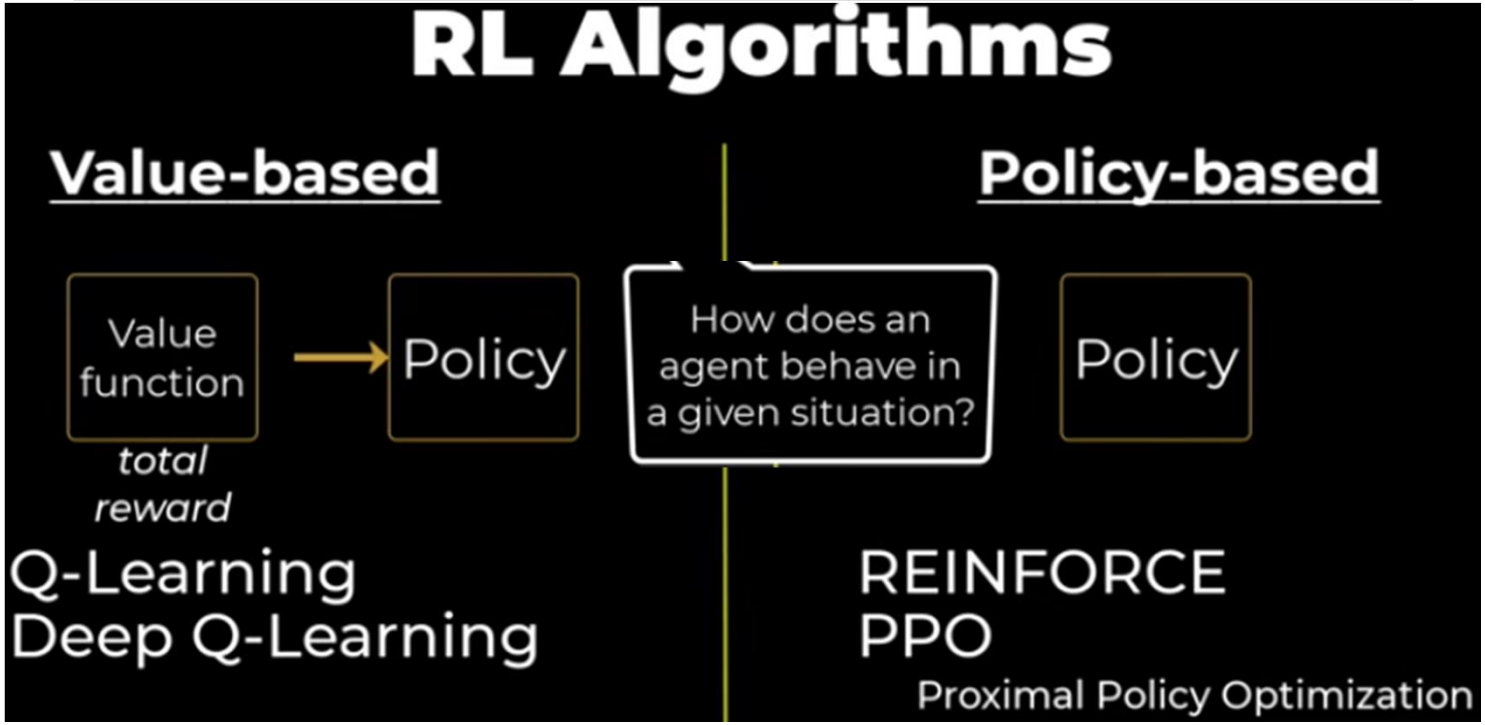
- Learn with fewer tries: it doesn’t need to try every action in the real world
- Handles new situations better
- It can imagine future steps and their results which helps in planning better

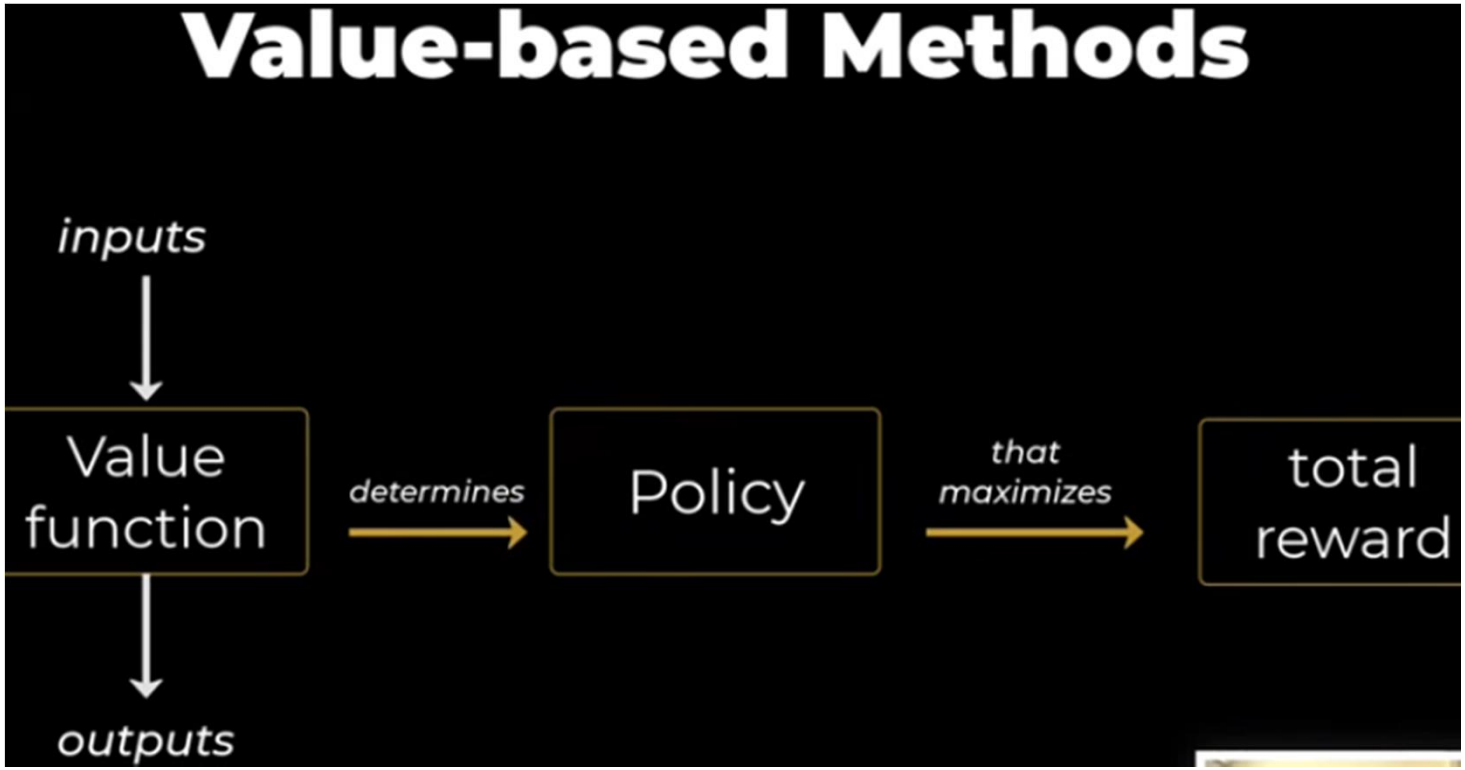
Challenges

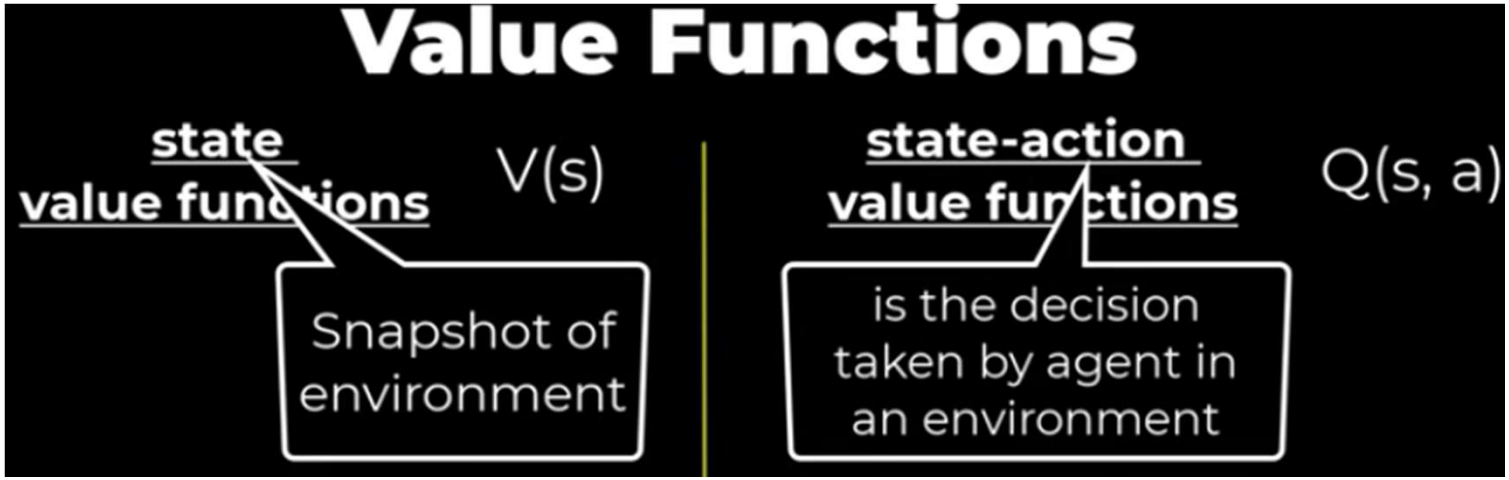
- Making a model that truly represents a complicated environment can be tough
- Takes a lot of computing power
- If it trusts its model too much, it might stop exploring better options and miss the best solution

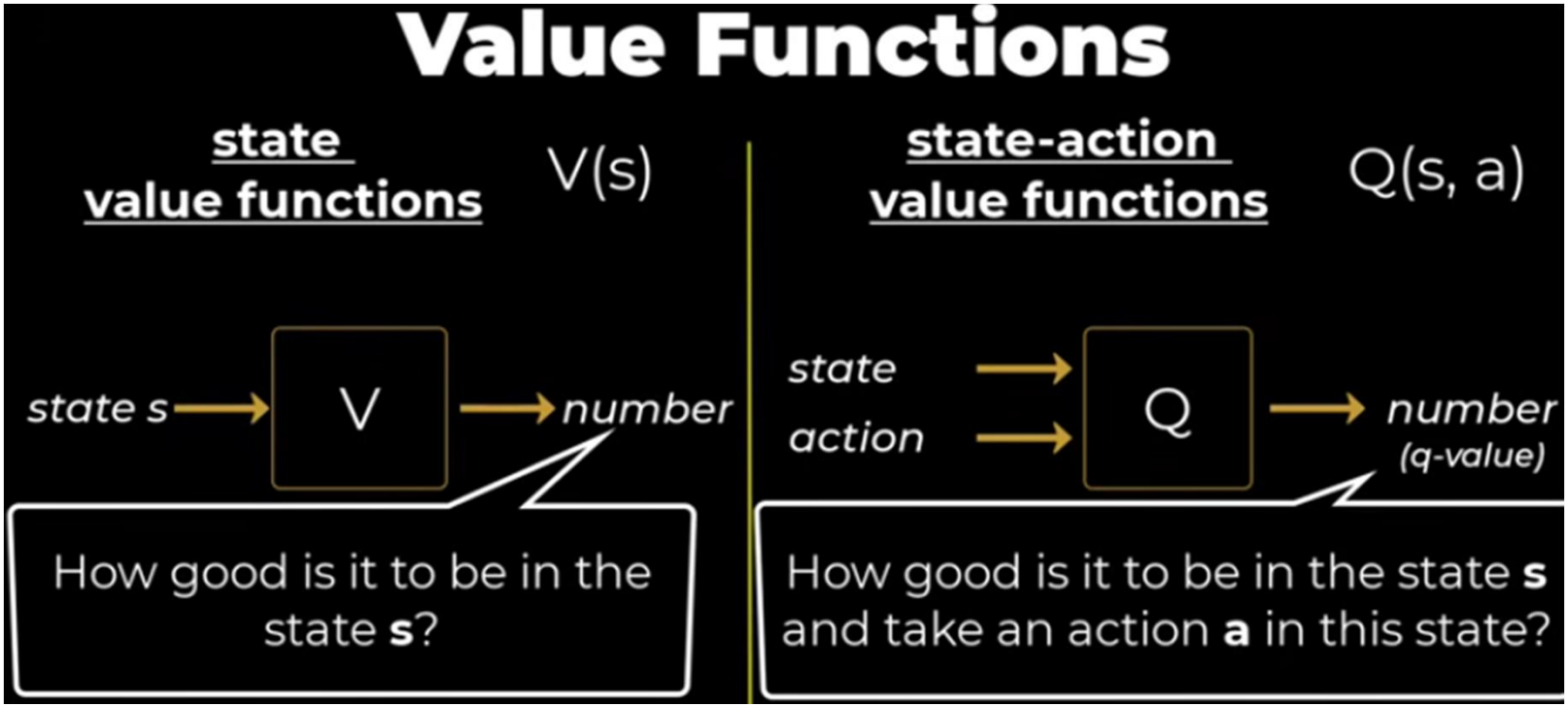
Temporal Difference Learning

Temporal Difference Learning is a method that value-based reinforcement learning algorithms, like Q-learning, use to iteratively learn state value functions or state-action value functions

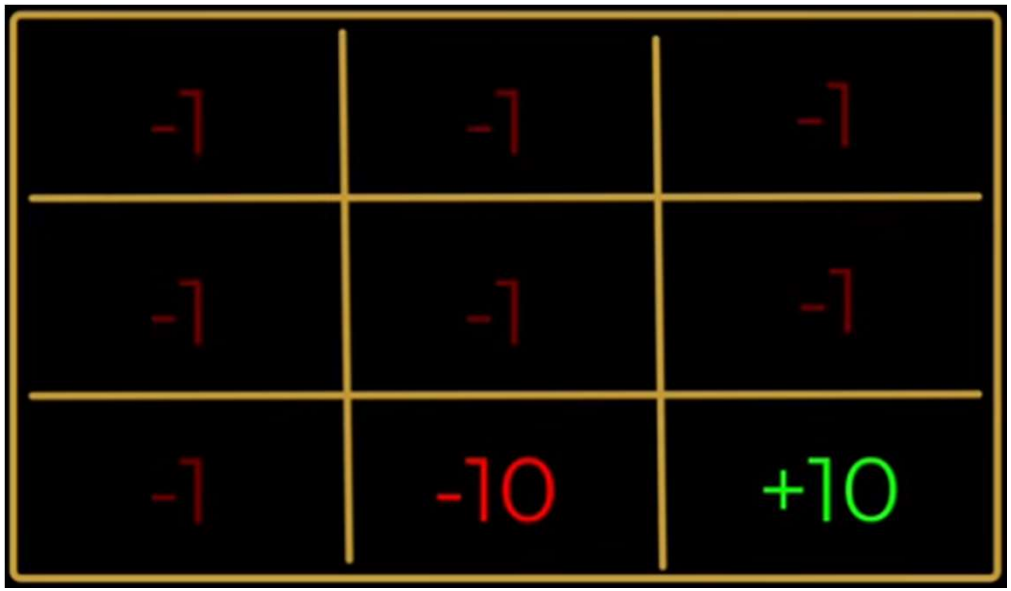



Value-based Methods
inputs
Value function
determines
Policy
that maximizes
total reward
outputs


Value Functions
state value functions
V(s)
Snapshot of environment
state-action value functions
Q(s, a)
is the decision taken by agent in an environment


Value Functions
state value functions
V(s)
state s
V
number
How good is it to be in the state s?
state-action value functions
Q(s, a)
state
action
Q
number
(q-value)
How good is it to be in the state s and take an action a in this state?

Example



We want the agent to learn the optimal policy

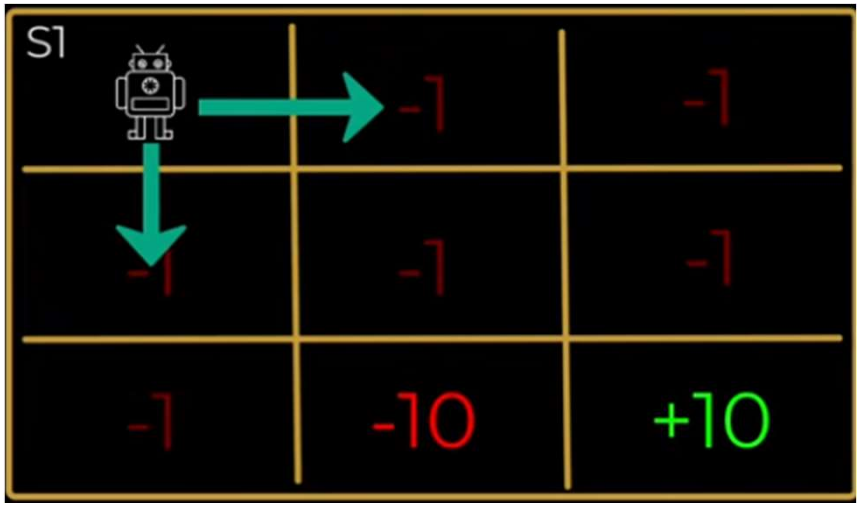
The sequence of actions —from a starting state to a terminal state—is called an **episode**

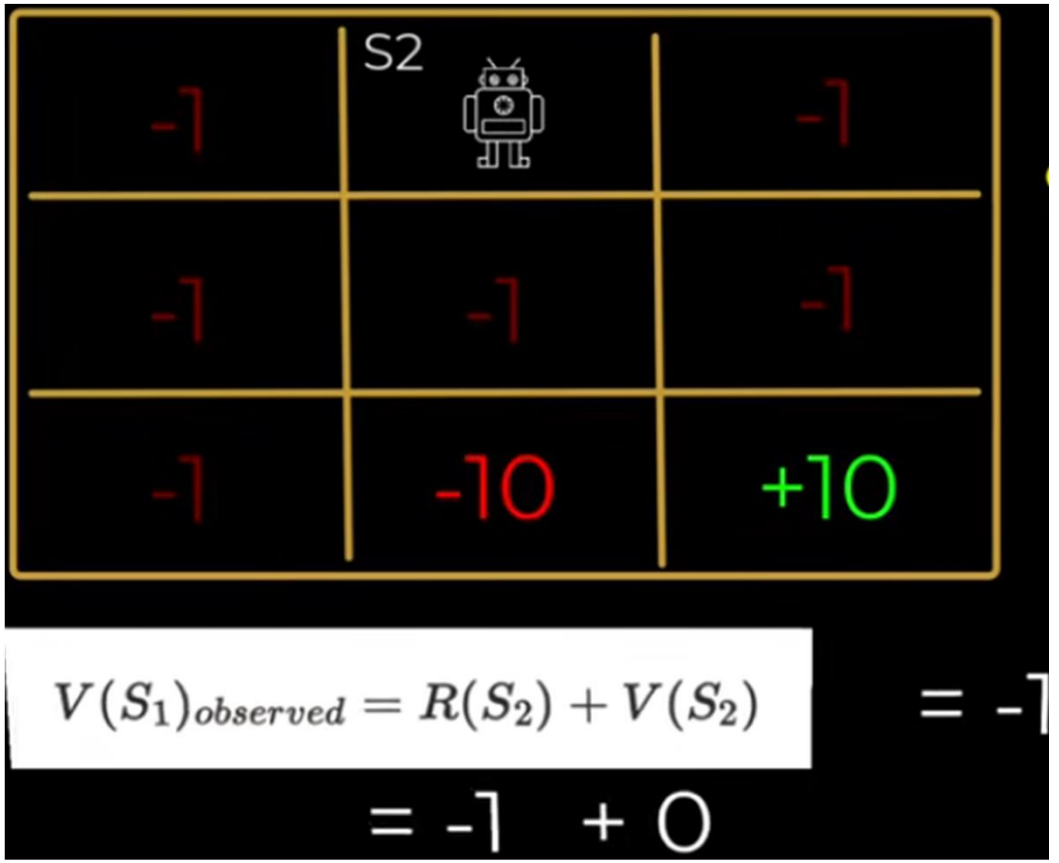
.1

s	V(s)
S1	0
S2	0
S3	0
.	.
.	.
Sn	0

How good is it to be in the state **s**?

What is the total reward I will see being in this state **s**?





s	V(s)
S1	-0.1
S2	0
S3	0
.	.
.	.
Sn	0

$$V(S_1)_{observed} = R(S_2) + V(S_2)$$

= -1

= -1 + 0

Temporal Difference Error $= V(S_1)_{\text{observed}} - V(S_1)_{\text{expected}}$

= -1 - 0 = -1

$V(S_1) = V(S_1) + \alpha \times$ Temporal Difference Error

= 0 + 0.1 x -1 = -0.1

Temporal Difference Learning

- A model-free reinforcement learning method in which an agent learns directly from **experience**, updating its predictions based on the **difference between consecutive predictions** — not waiting for the final outcome
- Monte Carlo + dynamic programming

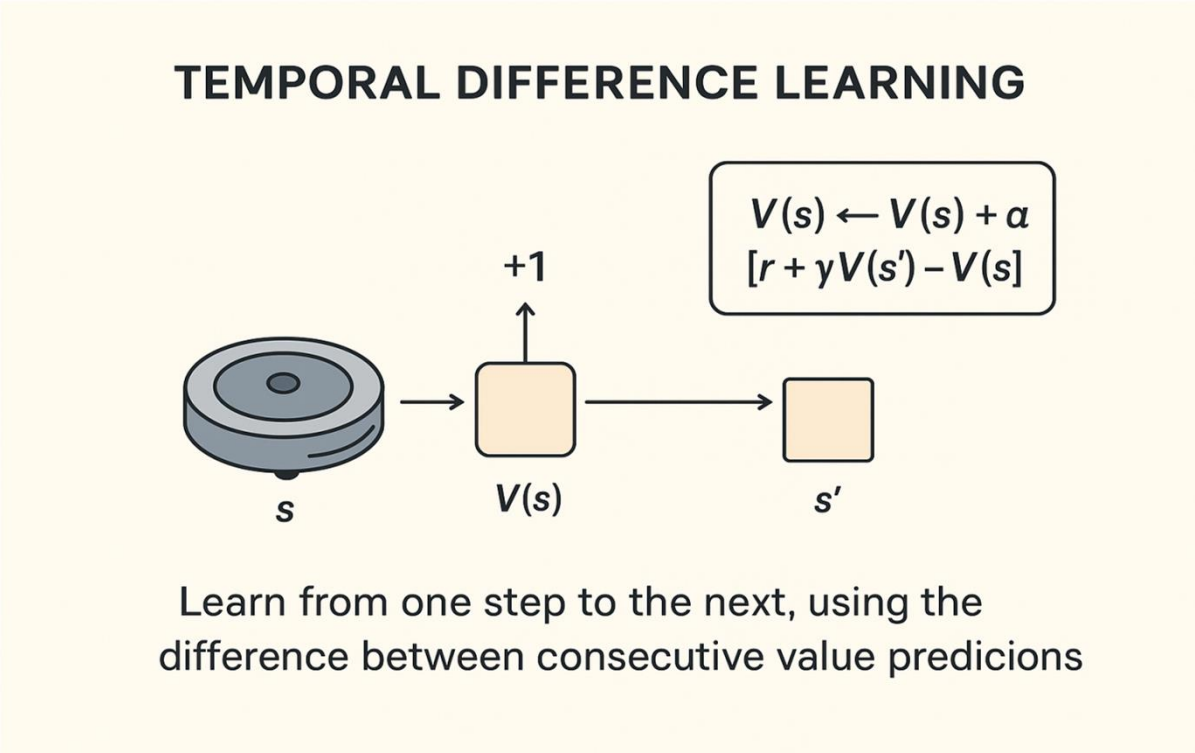
Algorithm

- Initialize value V(s) for all states
- For each step:
  - Take action a, observe reward *r* and next state s′
  - $V(s) = V(s) + \alpha[r + \gamma V(s') - V(s)]$
  - Set s = s′

TD error

where γ = discount factor (future importance), α = learning rate

Cont…



Characteristics

- Model-free: it can learn from experience (by updating values based on observed rewards and subsequent predictions) without knowing the environment's transition probabilities or reward functions
- It can learn and update its predictions in real-time
- It can be more efficient than Monte Carlo methods
- Applications: Robotics, Gaming, Finance

Generalization

- Agent to perform well in new or unseen situations
- Ability of an agent to **apply what it has learned** in one set of situations (training environment) to **slightly different situations** (testing environment)
- Agent doesn't memorize, instead it *understands patterns*
- Learn **robust policies** that transfer between tasks

Cont…

Method	Description
Function Approximation	Agent uses a neural network to approximate the value function or policy
Feature Representation	Using key features (like distance to obstacle, direction to goal) to improve generalization
Regularization	Techniques like dropout or weight decay prevent overfitting
Domain Randomization	Training the agent in varied simulated environments so that it learns robust behaviors
Transfer Learning	Using knowledge gained in one task to perform better on another related task

Partially Observable States

- Agent interacts with an environment by: observing its states – taking an action – receiving a reward
- Agent makes an optimal decision after seeing everything
- In **partially observable** environments, agent sees only part of the true state. Use POMDP
- Agent receives observations that give limited or noisy information about the true state
- Solution: Maintain memory or belief of what's unseen

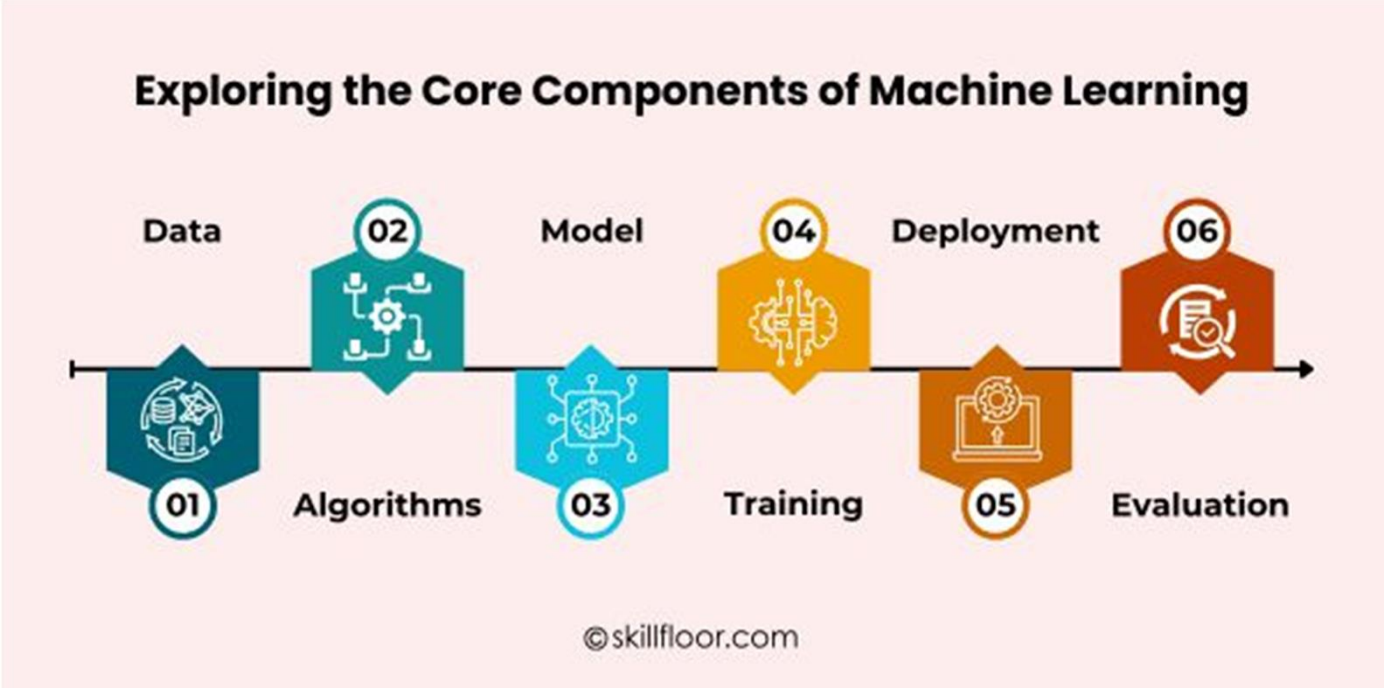
Cont…

- E.g. Autonomous driving

Method	Description
Belief State	Maintain a *probability distribution* over possible true states.
Recurrent Neural Networks (RNNs)	Use memory of past observations to infer hidden information.
Filtering Methods	Use algorithms like Kalman or Particle filters to estimate the current state.
Policy Search	Learn a direct mapping from observation history to actions.

Machine Learning Platform

- Providing an integrated space for tasks like data handling, algorithm selection, model training, validation, and deployment
- E.g. Amazon Sagemaker, Google Cloud Machine Learning Engine, Microsoft Azure Machine Learning, IBM Watson, TensorFlow



Data Preparation

- **Data collection**: Integrates with multiple sources - databases, APIs, and IoT devices
- data formats - structured, semi-structured, unstructured data
- Types of data - sounds, videos, images, text, or numbers
- larger datasets enhance accuracy and performance, smaller datasets may restrict learning
- **Data Transformation and Preprocessing**: normalizes data, encodes categorical variables, and applies necessary preprocessing to make data ML-ready
- **Data Versioning**: Maintains versions of datasets and data splits to ensure reproducibility in experiments
- **Data Validation**: Runs quality checks and validations to detect and remove anomalies/duplicates, missing values, or outliers, which ensures data integrity before model training

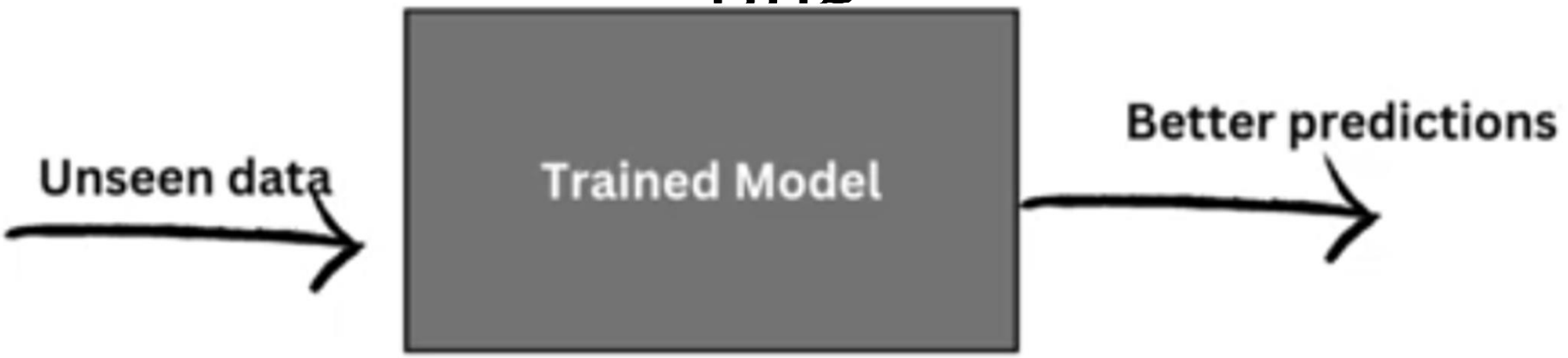
Algorithms

- **Learning methods**: rules-based or pattern-based approaches. System extracts valuable insights from the data
- **Choosing the Right Algorithm:** depending on goal and type of data
- **Types of algorithm**: reinforcement, supervised, unsupervised, and semi-supervised
- Tools: Orange, Weka

Model Training

- System uses the algorithm to accurately forecast or make decisions on fresh, unknown cases based on the knowledge it has learned
- Train the model on the preprocessed data to learn the underlying patterns and relationships.
- **Training Pipelines**: Creates pipelines for repetitive tasks like feature engineering, model training, and evaluation
- **Experiment Tracking and Logging**: Tracks model configurations, hyperparameters, and performance metrics across experiments
- **Distributed Training Support**: Enables large-scale data handling through TensorFlow, Apache Spark
- **Hyperparameter Tuning**: Using grid search, random search, or Bayesian optimization

ting



Bad predictions

Irrelevant data

HIgh Error Rate

Better predictions

Relevant data

Low Error Rate

High Generalizability

Validation and Testing

- Purpose of splitting the dataset into training, validation and testing sets is to assess the effectiveness of trained model in generalizing the new data
- **validation data:** to fine-tune learning rates or layer configurations to improve model accuracy
- To ensures that the model generalizes well beyond the training examples
- It serves as a checkpoint during the training process to assess whether the model is overfitting or underfitting
- **testing data:** evaluates the model's performance on new, unseen data
- Metrics such as accuracy, precision, recall, and F1-score are used to evaluate the model's performance
- If the model performs poorly on the testing data, it indicates that the adjustments made during validation were not sufficient, and the model may need to be revisited

Underfitting

Basics only

It fails to capture the underlying patterns in the data

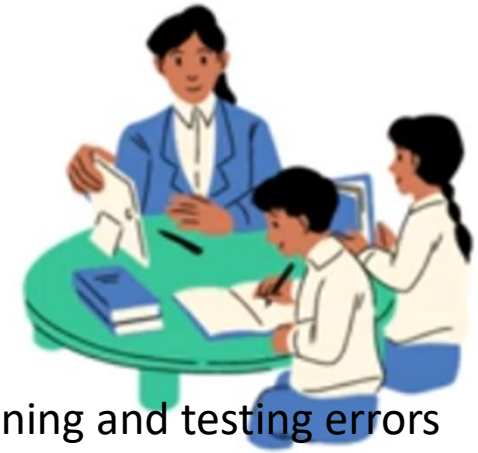
perfomance low

Overfitting

Knows everything but not able to create setences

Best fitting

Knows and able to build setences



Low training and testing errors

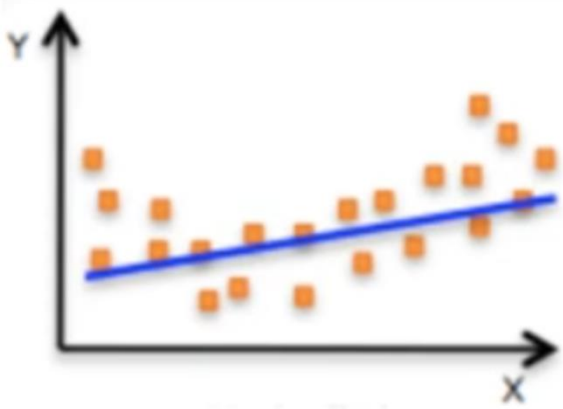
Best fitting is achieved when a model generalizes well from the training data to unseen data.

Reasons

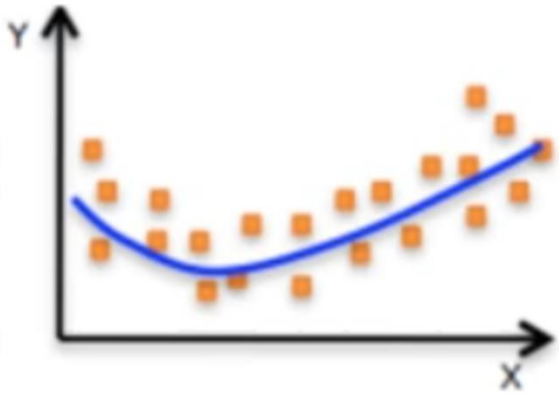
Overfitting happens when a model is too complex and captures noise in the training data rather than the underlying pattern. It performs well on the training data but poorly on the test data.

Model is Too Simple

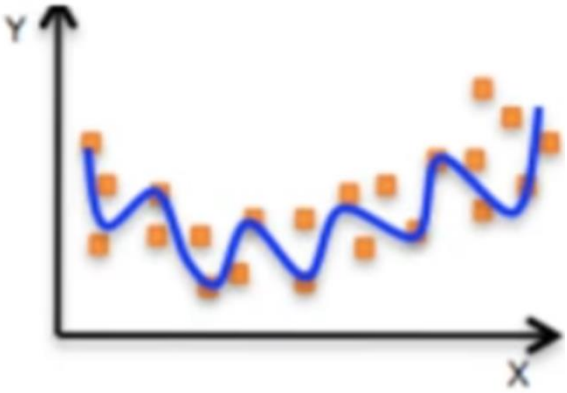
Insufficient Training Data



underfitting



Goodfit



Overfitting

Term	Description	Performance on Training Data	Performance on Unseen Data
Underfitting	Model is too simple	Poor	Poor
Overfitting	Model is too complex	Good	Poor
Best Fitting (Generalization)	Model captures essential patterns	Good	Good

Error Calculation

- Model performance good? Degrade its performance with new data

$$\text{Accuracy} = \frac{No\ of\ correct\ predictions}{Total\ prediction}$$

- Confusion metrics computes recall, precision, accuracy, ROC score, F1 score

$$\text{Recall} = TP/(TP + FN)$$

- how many of the actual positive cases were correctly identified (E.g., row-wise)

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Confusion matrix

Patient	Diagnosis	
	Sick	Healthy
Sick	8	20
Healthy	1	99

Cont…

Precision = TP / (TP + FP)

- Accuracy of positive predictions (E.g., column-wise)
- F1 score: Harmonic mean between precision and recall = $\frac{2ab}{a+b}$
- Receiver Operating Characteristic (ROC) score is the slope of a ROC plot
- It shows the performance of a *classification* model at all classification thresholds
- Two parameters for the ROC score - True Positive Rate and False Positive Rate

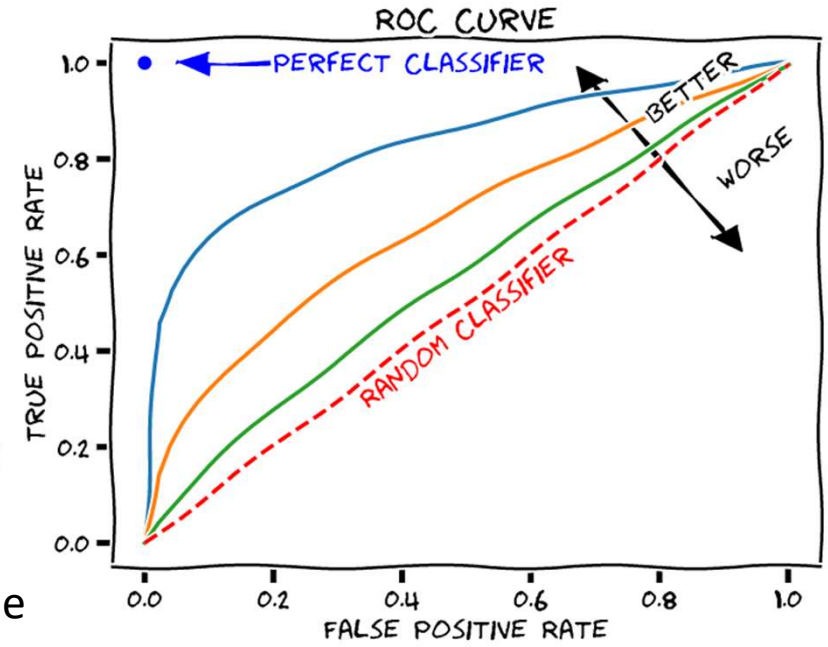
Regression

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

MAE (mean absolute error) will measure the average magnitude of errors, without considering the direction or sign of that er

$$MAE = \frac{1}{n}\sum|e_t|$$

where *e* is the error between the prediction and actual value



Cross-Validation

- Repeated use of the same data split differently
- Stratification: make sure that classes are represented in the right proportions when subsets of data are held out
- K-fold cross validation: dataset X is divided randomly into K equal sized parts
- keep one of the K parts out as the validation set and combine the remaining K −1 parts to form the training set. Repeated K times
- Leave-one-out: In a dataset of N instances, only one instance is left out as the validation set and training uses the N − 1 instances. Used in medical diagnosis
- 5 x 2 cross-validation: equal size training and validation sets. Swap the role of these sets after a training (first fold). Randomly shuffle the dataset and repeat the process four more times
- Bootstrap (resampling technique): From an original dataset, multiple new datasets are created by randomly selecting N data points with replacement. i.e., a single data point from the original dataset can appear multiple times, or not at all

Binomial Test

- **Binary outcome:** The variable has only two categories (such as success/failure or yes/no).
- **Independence:** Each observation does not affect any other observation.
- **Fixed sample size:** The total number of trials (n) is decided before the test.
- **Constant probability:** The expected probability of success (p) stays the same for every trial
- **Null Hypothesis:** The probability of success is equal to a specified value, p_0.
- **Alternative Hypothesis;** The probability of success is different from (two-tailed), greater than (right-tailed), or less than (left-tailed) the specified value.
- **Two-tailed:** Tests whether the observed proportion is significantly different from p_0 in either direction.
- **Right-tailed:** Tests whether the observed proportion is significantly greater than p_0
- **Left-tailed:** Tests whether the observed proportion is significantly less than p_0

Binomial Test

- Flip a coin n=20 times and get x=14 heads. Test if coin is fair (p_0=0.5)
- Set up the hypothesis,

H_0: p=0.5 (coin is fair)
H_1: p≠0.5 (coin isn't fair)

- Compute the probability of each possible outcome under H_0

$$P(X = k) = \binom{n}{k} p_0^k (1 - p_0)^{n-k}$$

- Find the probability of observing 14 or more heads

$$P(X \geq 14) = \sum_{k=14}^{20} \binom{20}{k} 0.5^k 0.5^{20-} \quad = 0.05765$$

- Include outcomes equally extreme on the low side, P(X≤6)=P(X≥14)
- Compute total p-value, p=2×P(X≥14)=2×0.05765=0.1153
- At a 5% significance level (α=0.05): p=0.1153>0.05
- **Fail to reject** H_0 — no significant evidence that the coin is unfair

Approximate Normal Test

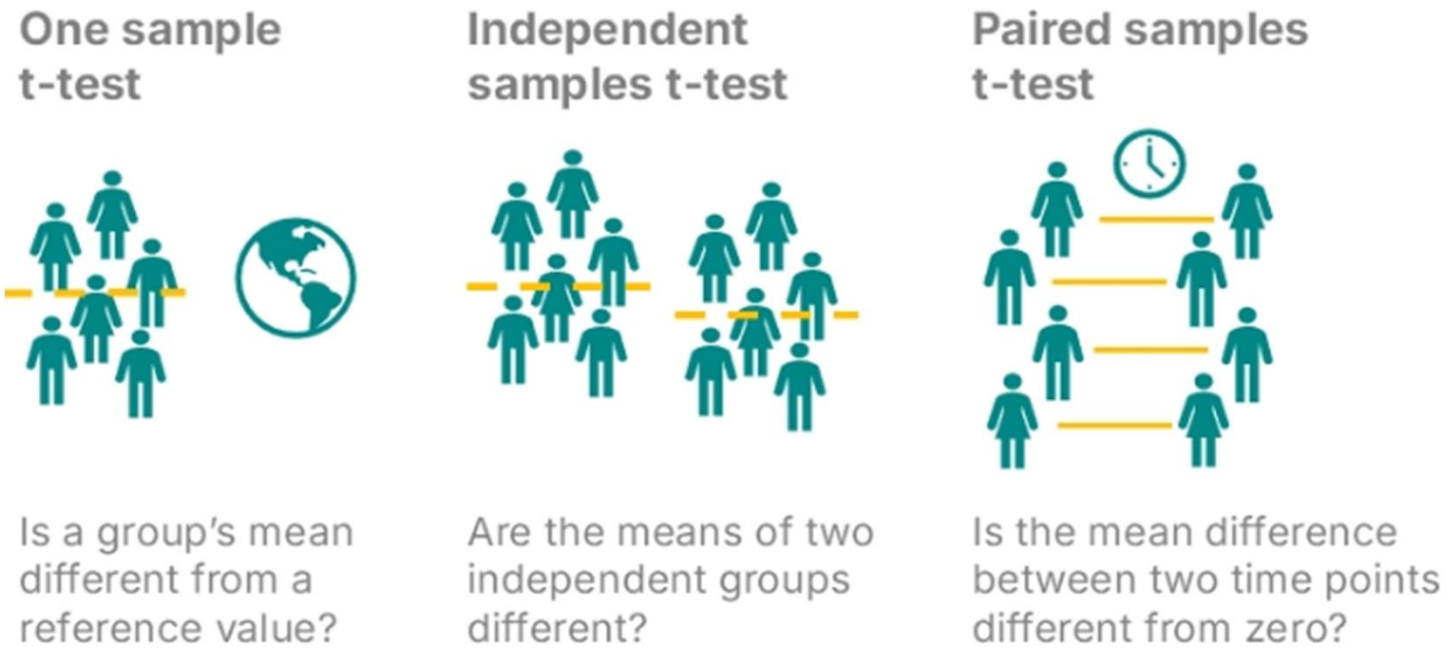
- Company's claim 60% of their customers are satisfied. Take a random sample of n=100 and find that 54 of them are satisfied. Is the true satisfaction level is < 60%?
- Define the hypothesis $H_o : p = 60$, $H_1 : p < 60$
- Compute sample proportion $\hat{p} = \frac{x}{n} = 0.54$
- Compute standard error under H_o
- $SE = \sqrt{\frac{p_0(1-p_0)}{n}} = \sqrt{\frac{0.6(1-0.6)}{100}} = 0.049$
- Compute test statistic $Z = \frac{\hat{p} - p_0}{SE} = -1.22$
- Determine the critical value: $z_{0.05} = -1.645$

At 5% significance level,

- Decision rule, Z=−1.22>−1.645; Fail to reject H_o
- There is **not enough evidence** that the true satisfaction rate is less than 60%.

t-Test

- Types



- degrees of freedom essentially tell the t-test how much "information" you have in your data
- **critical t-value**: For a two-tailed test, you reject the Null Hypothesis if the absolute t-value is greater than the critical value

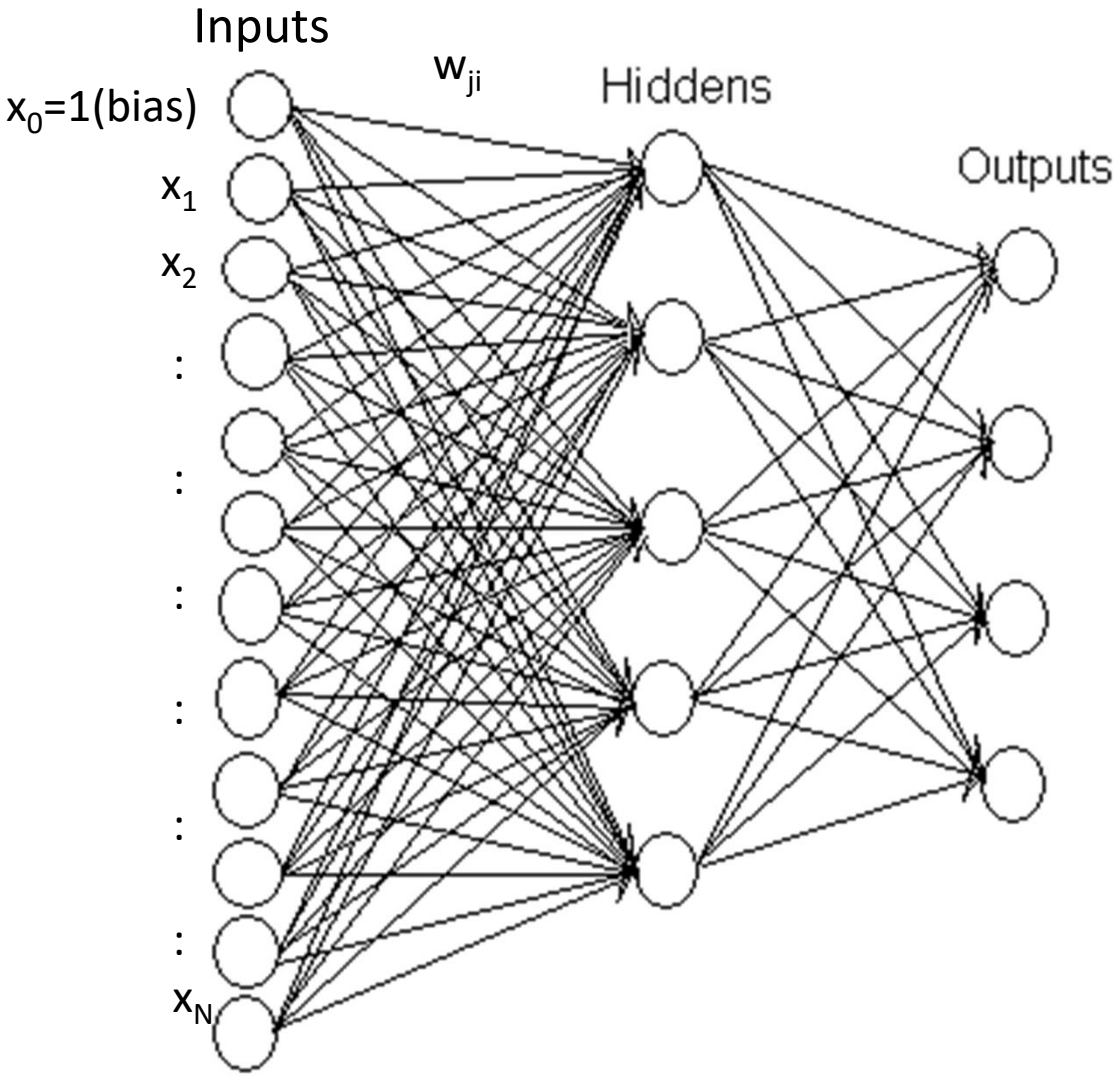
t Test

- Nutrition claims that a certain protein bar contains 20 g of protein. A random sample of 10 bars gives the contents: 18.5, 19.2, 20.1, 21.0, 19.8, 20.5, 19.9, 18.7, 19.5, 20.3. Test whether the mean protein content differs from 20 g
- State the hypothesis, H_0 : μ=20
  H_1 : μ≠20
- Compute $\bar{X} = 19.75$ σ=0.74
- Compute test statistic $t = \frac{\bar{X}-\mu_0}{\sigma/\sqrt{n}} = \frac{19.75-20}{0.74/\sqrt{10}} = -1.07$
- Find the critical value: degree of freedom = n-1 = 9;

Significance level, α=0.05; $t_{0.025,\ 9}$ = 2.262

- Decision rule, |t|=1.07<2.262
  Fail to reject H_0. There is **no statistically significant difference** between the sample mean and the claimed mean

Neural Network



Cont…

- Brain like computing device
- Neural networks are built up of model neurons
- Perceptron: an ANN model
- Real neuron is an immensely complex structure. It is difficult to model even a small fraction of it. So use a model that is adequate for our work
- The detailed interactions among many neurons are the key to the computation that neural network performs
- NN is defined as a computing structure consisting of a massively parallel interconnection of adaptive “neural” processors
- It is a dynamical system. i.e., a system that evolves with time
- The network is designed as multilayer feed forward network, using supervised mode of learning
- Types of network operations : training and testing

Important Property of Neural Networks

Results get better with

more data +
bigger models +
more computation

(Better algorithms, new insights and improved techniques always help, too!)

Back Propagation[1989]

- NN is used to make functional relationship between its i/p & o/p
- Generalised delta rule is the learning algorithm for the network
- i/p to the jth hidden node (weighted sum of inputs)

$$net_{pj}^{h} = \sum_{i=1}^{N} w_{ji}^{h} x_{pi} + \theta_{j}^{h}$$

- Convert i/p activation value for PE

o/p of PE $$g_{pj} = f_{j}^{h}\left(net_{pj}^{h}\right)$$

-> Multilayer feed forward network

- Equations for o/p nodes

$$net_{pk}^{o} = \sum_{j=1}^{L} w_{kj}^{o} g_{pj} + \theta_{k}^{0} \qquad o_{pk} = f_{k}^{o}\left(net_{pk}^{o}\right)$$

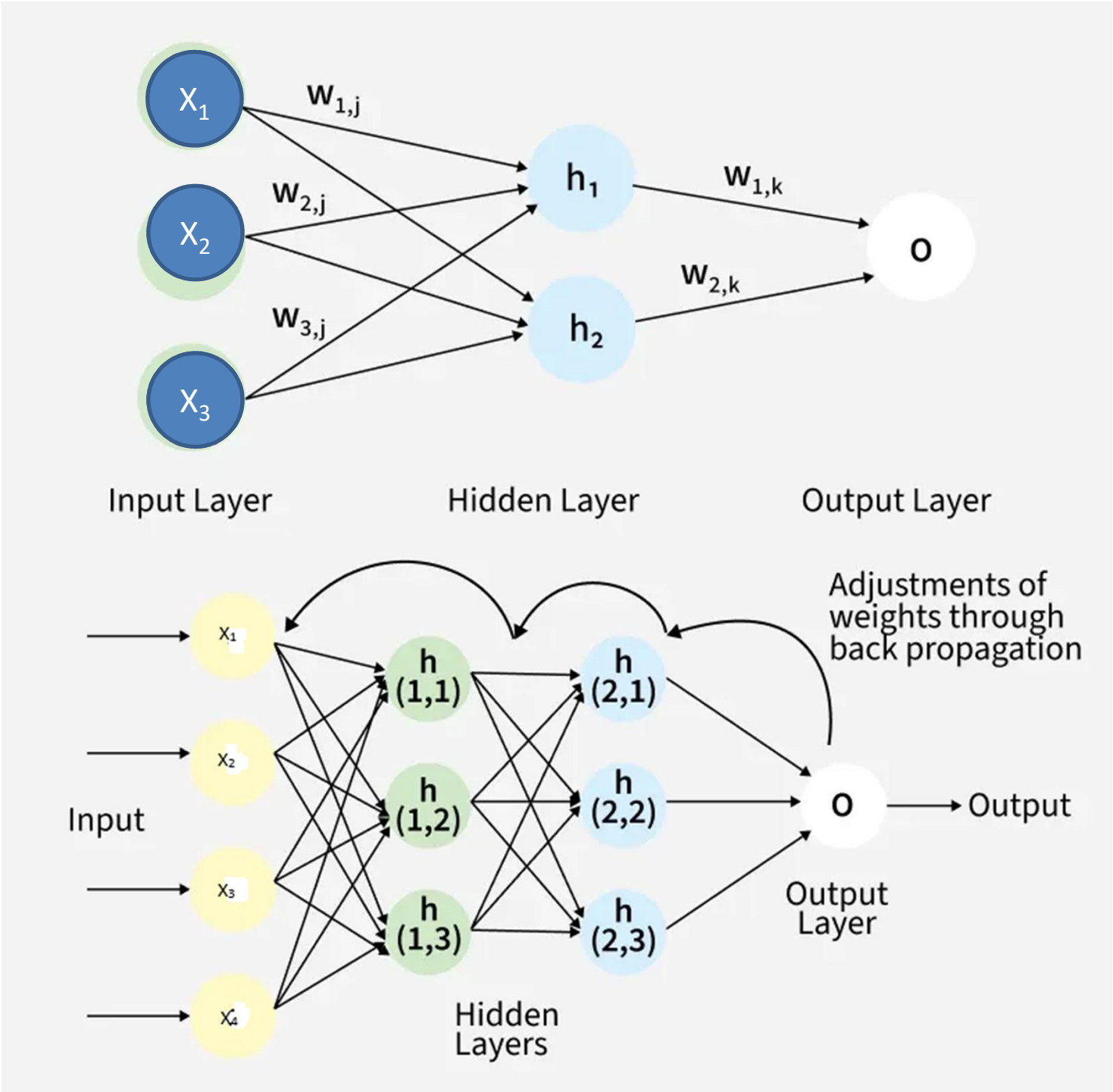
Cont…

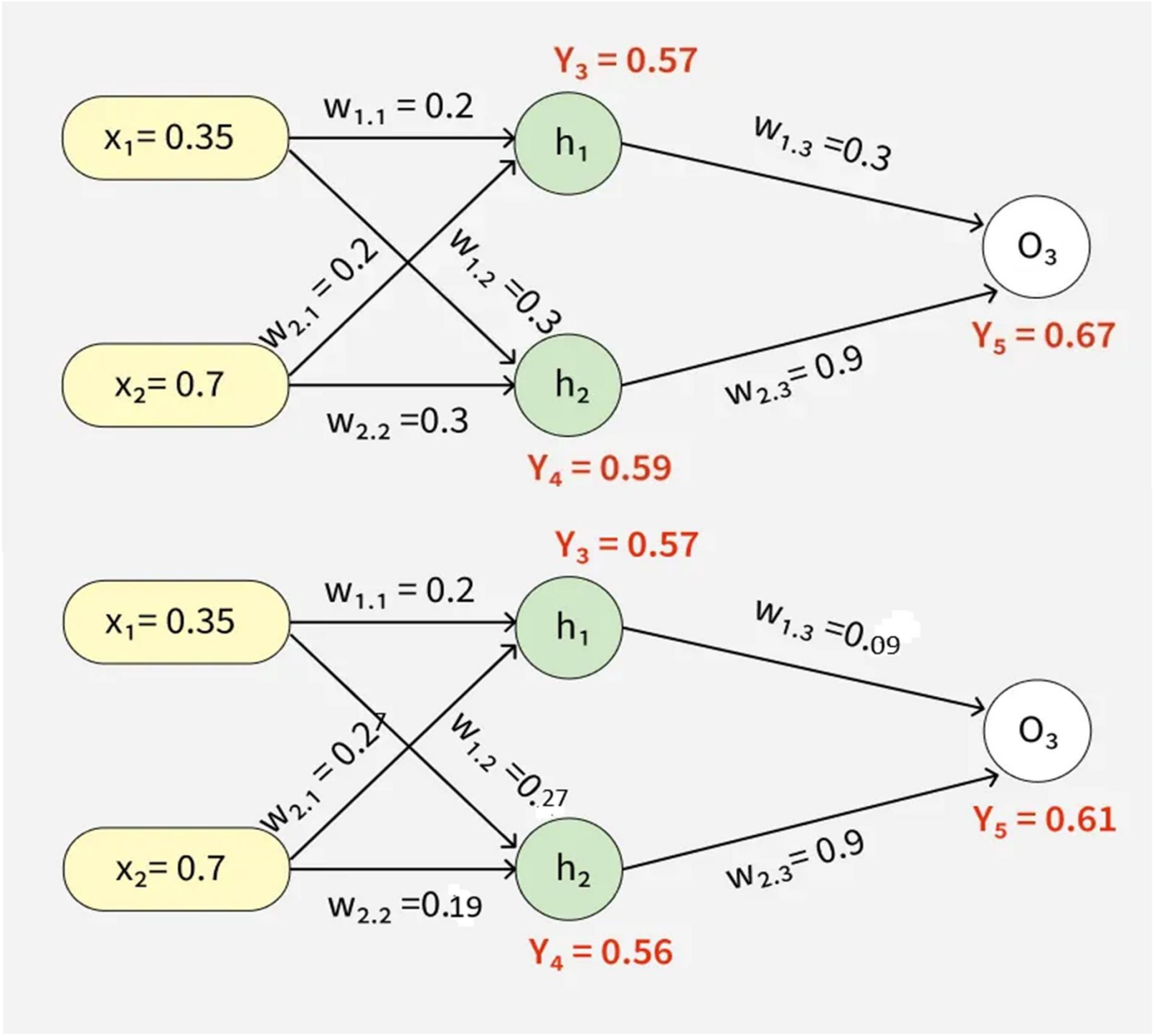
- O/P pattern is compared to the desired o/p and an error signal is transmitted from the o/p layer to each node in the intermediate layer

Cost function, MSE = (predicted o/p – desired o/p)^2

- As the network trains, different layers learn to recognise different features of the input space
- Iteratively updates the weights and biases. Level of adjustment is determined by the gradients of the cost function
- Reduce loss across epochs
- Update rule for single neuron

$$w = w\text{-}\eta(\hat{y} - y)f'(wx + b)x$$


X1
X2
X3
w1,j
w2,j
w3,j
h1
h2
w1,k
w2,k
O
Input Layer
Hidden Layer
Output Layer
Adjustments of weights through back propagation
x1
x2
x3
x4
Input
h (1,1)
h (1,2)
h (1,3)
h (2,1)
h (2,2)
h (2,3)
Hidden Layers
O
Output
Output Layer

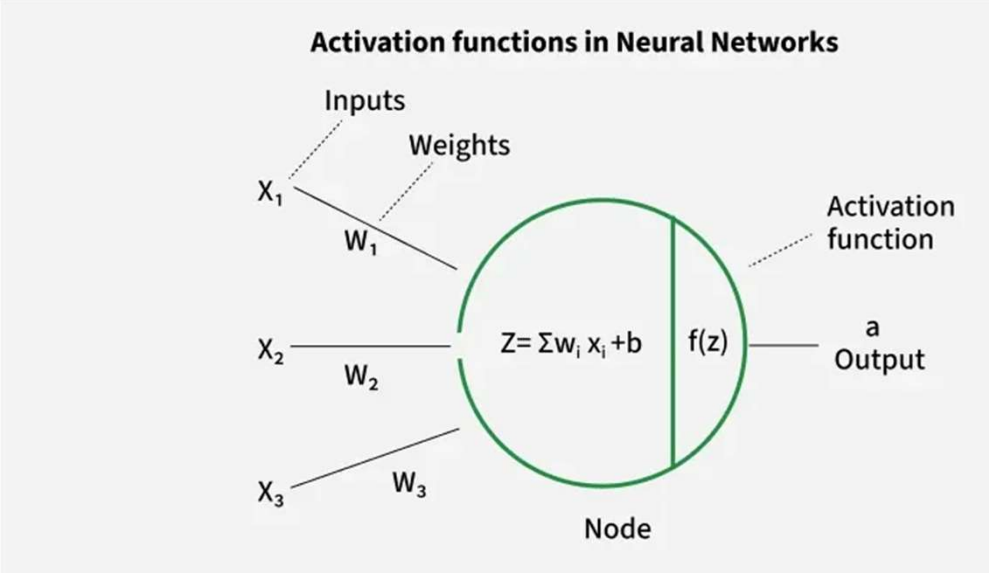

Y3 = 0.57
x1= 0.35
w1.1 = 0.2
h1
w1.3 =0.3
O3
w2.1 =0.2
w1.2 =0.3
x2= 0.7
h2
w2.3= 0.9
Y5 = 0.67
w2.2 =0.3
Y4 = 0.59
Y3 = 0.57
x1= 0.35
w1.1 = 0.2
h1
w1.3 =0.09
O3
w2.1 =0.2
w1.2 =0.27
x2= 0.7
h2
w2.3= 0.9
Y5 = 0.61
w2.2 =0.19
Y4 = 0.56

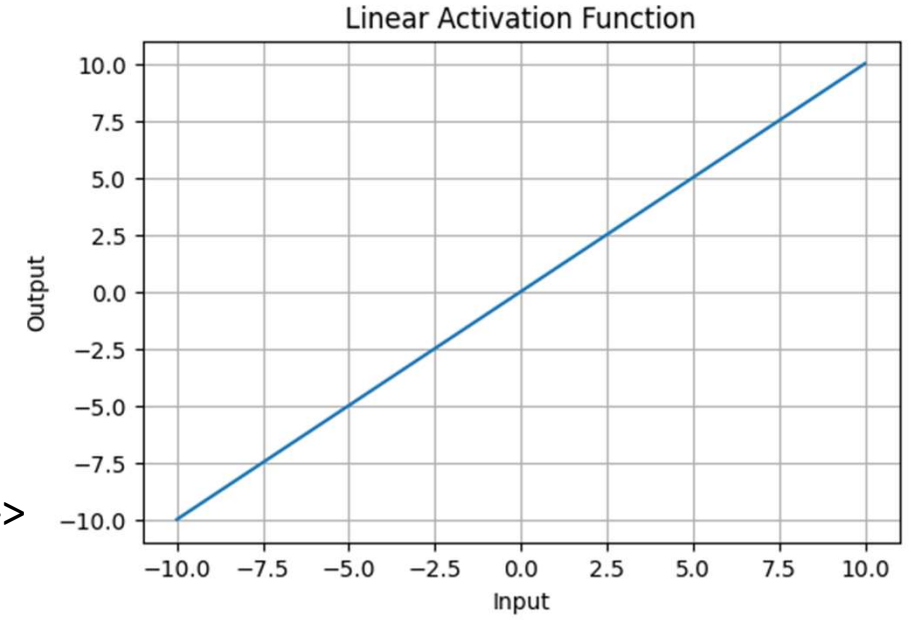
Linear Activation Function

$f(x) = x$

- A mathematical function applied to the output of a neuron
- passes the weighted sum of inputs directly to the next neuron without making any changes
- If you use a linear function in every layer, multiple hidden layers collapse into a single layer
- Output range (−∞ to +∞)
- Limited capability to learn complex patterns

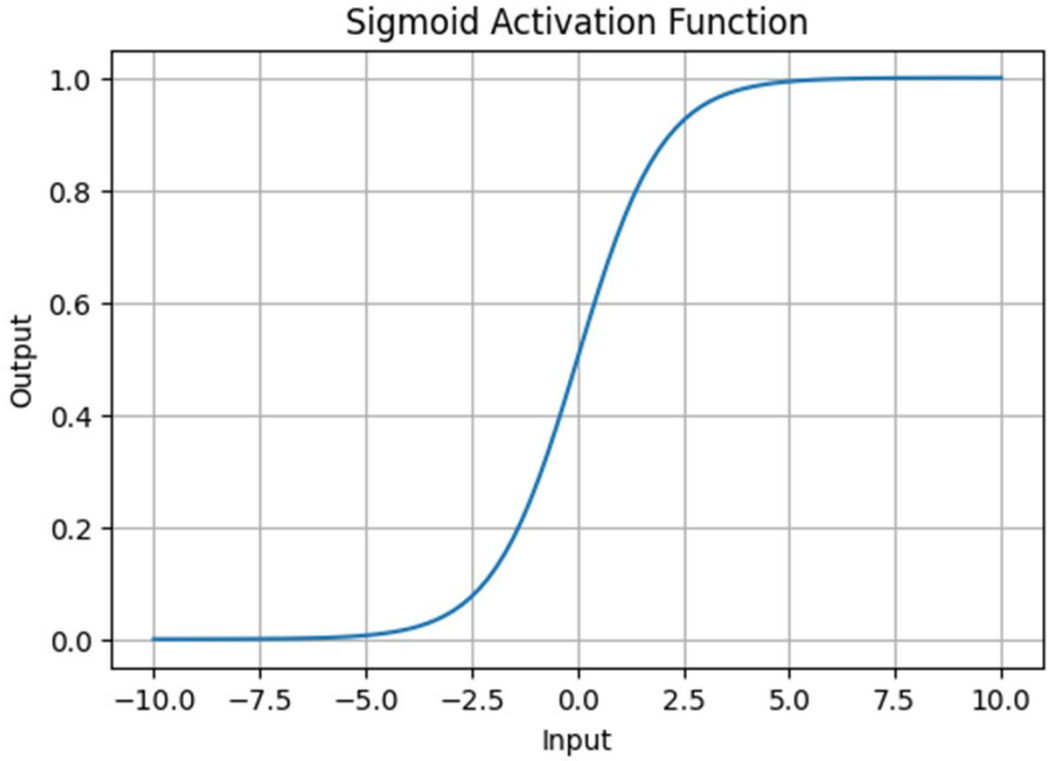
output is proportional to the input ->





Non-linear Activation Functions

- It allows neural networks to form curved decision boundaries, making them capable of handling complex patterns
- Output range 0 to 1. Widely used in output layer for binary classification

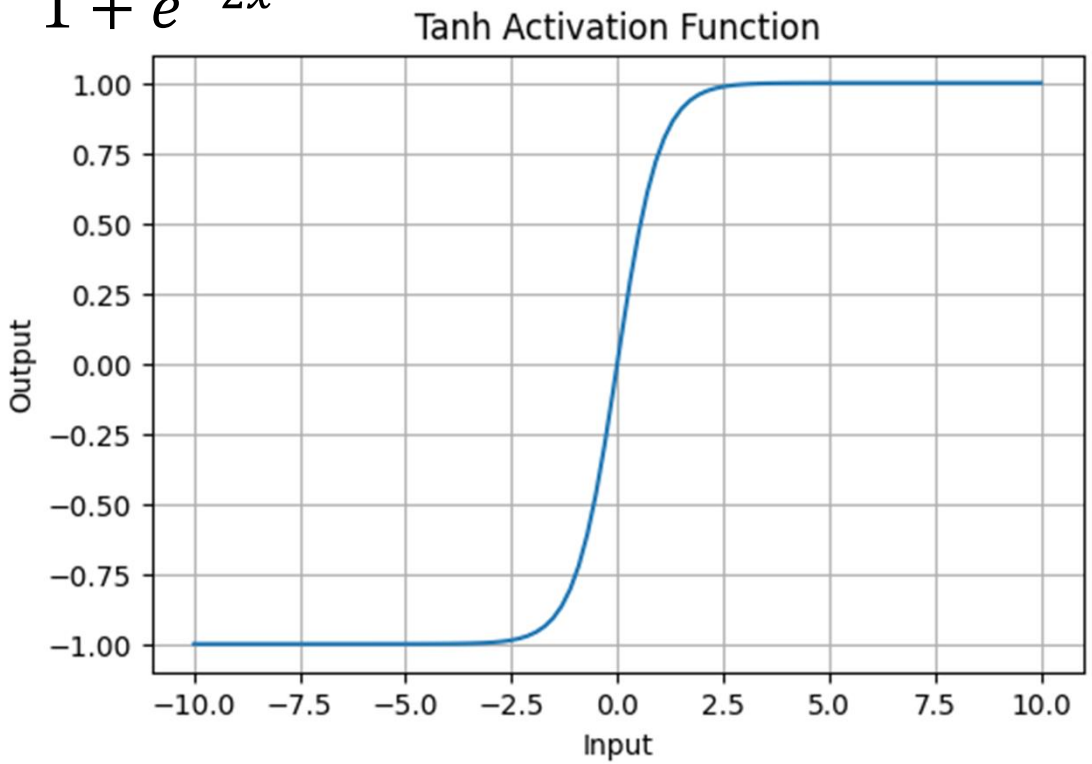


$$A = \frac{1}{1 + e^{-x}}$$

Cont…

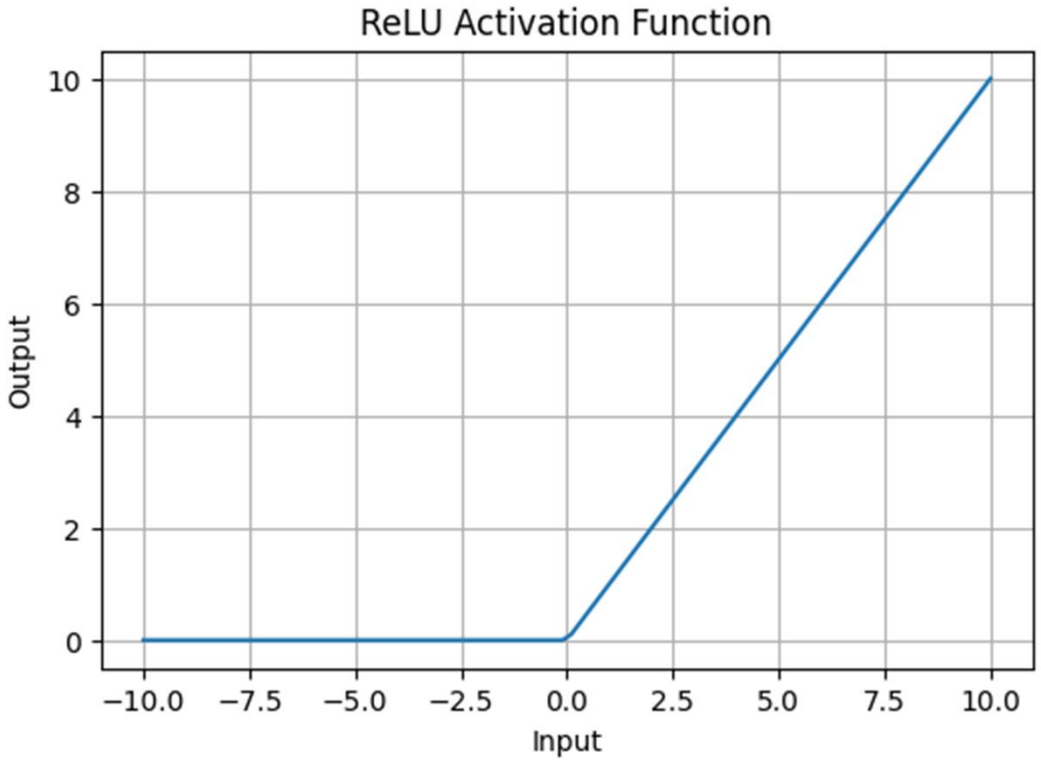
- Output range -1 to 1
- Commonly used in hidden layers

$$\tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$



Cont…

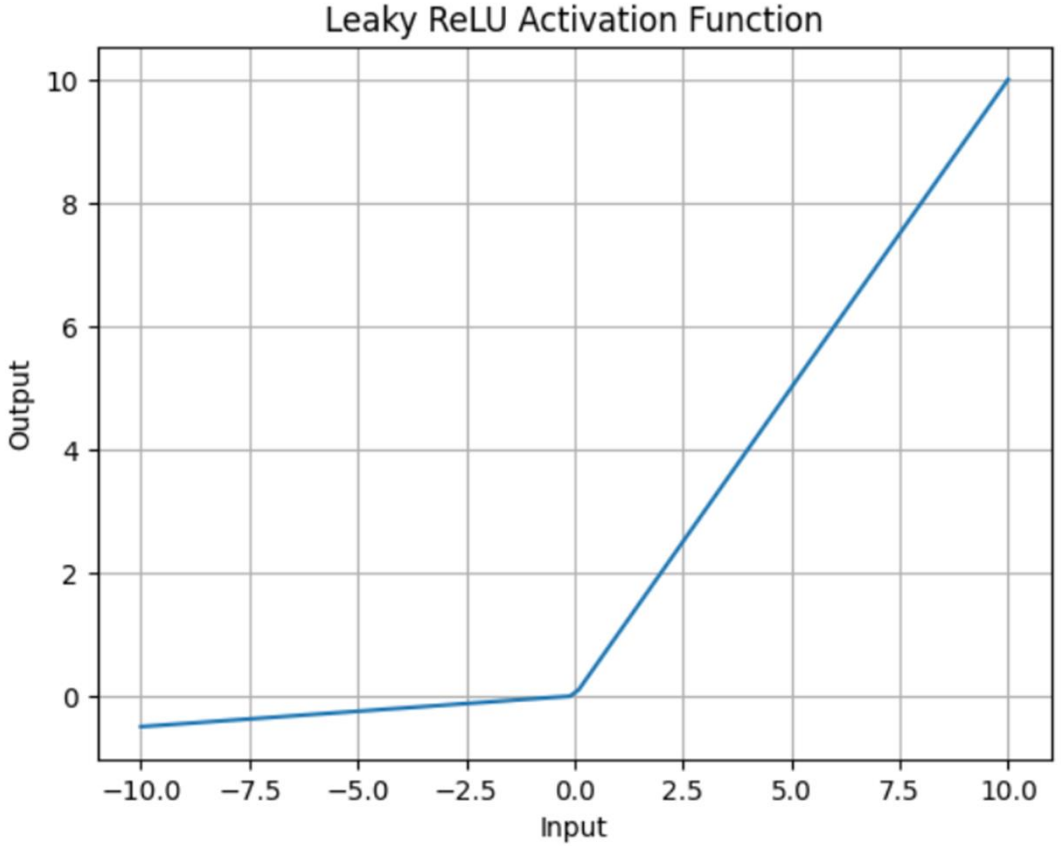
- outputs the input directly if it is positive and returns zero for any negative input
- A(x) = max(0, x). Output range (0, ∞)
- Less computationally expensive. At a time only a few neurons are activated making the network sparse



Cont…

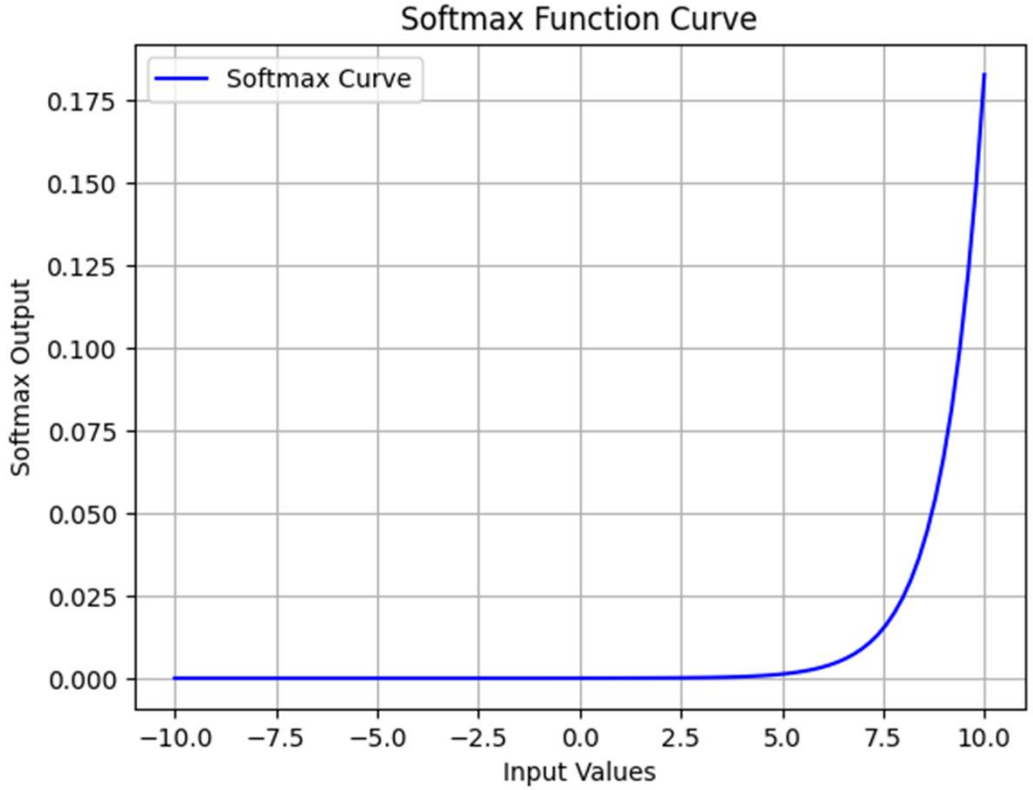
- Similar to ReLU but allows a small negative slope (α, e.g., 0.01)
- Output range $(-\infty, \infty)$

$$f(x) = \begin{cases} x & x > 0 \\ \alpha x & x \leq 0 \end{cases}$$



Cont…

- For multiclass classification. Used in the output layer
- It ensures that each class is assigned a probability. Sum of all probabilities equals 1
- For a vector *z* with items z_i, the softmax of item *i* is

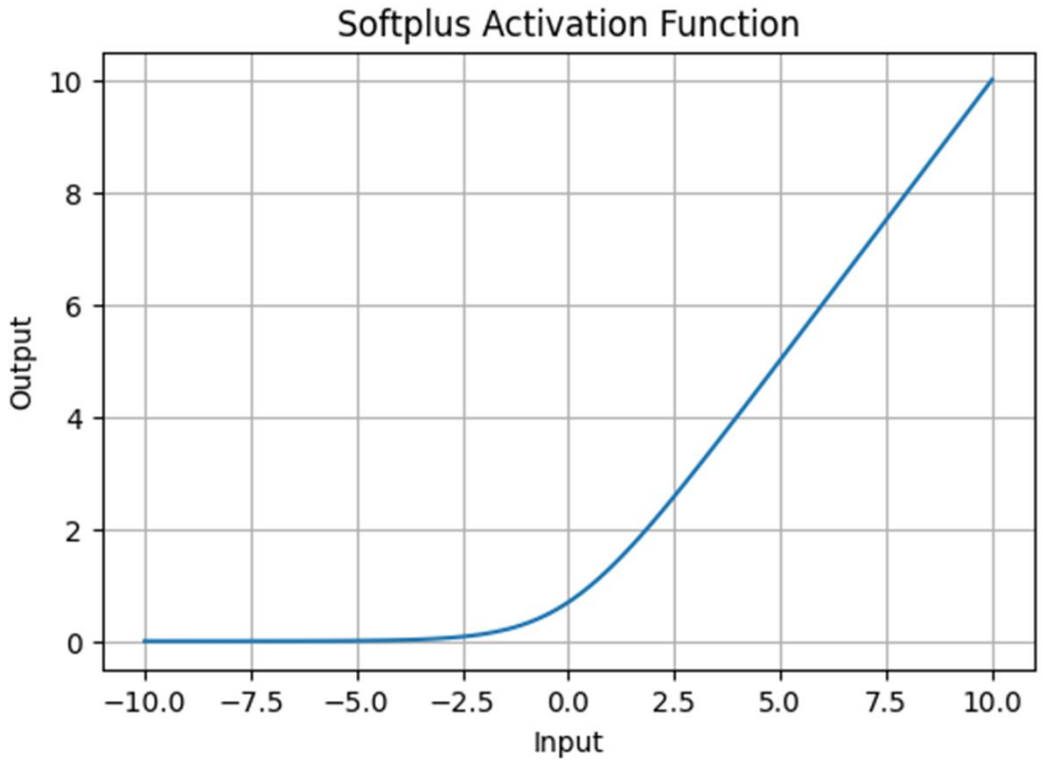


$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Cont…

$A(x)=\log(1+e^{x})$.

- This equation ensures that the output is always positive
- It is a smooth, continuous function
- **Smoothness:** Unlike ReLU, which has a sharp corner at x = 0, softplus is continuously differentiable everywhere



Cont…

Impact of activation functions

- ReLU allows *faster* training while Sigmoid and Tanh can *slow* down convergence in deep networks
- ReLU ensures better *gradient flow*, helping deeper layers learn effectively. In contrast Sigmoid can lead to small gradients, hindering learning in deep layers
- ReLU or Leaky ReLU are used for basic layers

Applications

Computer Vision

- **Facial Recognition:** Used in smartphones and security systems to identify individuals (PimEyes)
- **Medical Imaging:** Analyzing X-rays, CT scans, and ultrasounds to detect diseases
- **Autonomous Vehicles:** Recognizing road signs, obstacles, and pedestrians for self-driving cars
- **Image and Video Analysis:** Identifying objects, scenes, and activities in images and videos (Gemini)

Natural Language Processing (NLP)

- **Chatbots and Virtual Assistants:** Understanding and responding to user queries, as seen in customer service applications
- **Machine Translation:** Translating text from one language to another
- **Text Summarization:** Condensing long documents into shorter summaries (SciSpace)
- **Sentiment Analysis:** Determining the emotional tone of text

Cont…

Forecasting and Prediction

- **Stock Market Prediction:** Analyzing financial data to forecast future trends (capitalize.ai)
- **Weather Forecasting:** Predicting future weather conditions (WeatherNext)
- **Demand Prediction:** Forecasting consumer demand for products and services (c3.ai)

Healthcare

- **Disease Diagnosis:** Assisting doctors in identifying diseases from medical data.
- **Personalized Treatment Plans:** Developing customized treatment strategies based on a patient's data (Tempus)
- **Drug Discovery:** Analyzing complex data to identify potential new drugs (AIDDISON)

Cont…

Finance

- **Fraud Detection:** Identifying fraudulent transactions by detecting unusual patterns (Feedzai)
- **Risk Management:** Assessing and managing financial risks (Predict360)
- **Trading:** Analyzing trading volume and other market data to make predictions (StockHero)

Other Applications

- **Recommendation Systems:** Suggesting products, movies, or music based on user preferences (Netflix, Amazon).
- **Gaming:** Developing AI agents that can play games (AlphaGo).
- **Data Mining:** Extracting meaningful information from large datasets (Scikit-learn, Pandas)
- **Speech Recognition:** Transcribing spoken words into text and understanding the tone of voice (Google AI)