COMP5121
Data Mining and Data Warehousing Applications

Week 9: Advanced Classification

Dr. Fengmei Jin

- Email: fengmei.jin@polyu.edu.hk
- Office: PQ747 (+852 3400 3327)
- Consultation Hours: 2.30-4.30 pm every Thursday

Outline

- Bayesian Belief Networks (BBN)
- Support Vector Machine (SVM)
- Neural Networks (NN)

BAYESIAN BELIEF NETWORKS (BBN)

Review of Naïve Bayesian Classifier

- A data tuple: $X = (x_1, x_2, \dots, x_n)$
- **Bayesian Theorem**

Priori probability *that the class Ci appears*

$$P(C_i \mid X) = \frac{P(X \mid C_i) \cdot P(C_i)}{P(X)}$$

Posterior probability *that X belongs to Ci after observing X*

- The probability of X occurring in the class C_i:
  - Assume all attributes are independent from each other.
  - $P(X \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \cdots \times P(x_n \mid C_i)$

Independence Assumption Can Be a Drawback!

- **Complexity of real-world problems**
  - "*age*" and "*student*" may collectively affect "*income*".
- **Overestimation of some evidence**
  - "*cloth_color*" and "*buys_computer*"
- **Underfitting importance evidence**
  - "*income*" and "*buys_computer*"

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*

Bayesian Belief Networks (BBN)

- An extension of Bayesian reasoning that:
  - Relaxes the independence assumption
  - Captures dependencies explicitly using a graphical structure

- **Example**:
  - Weather, Traffic, and Being late
    - “Weather” affects “traffic” (dependency)
    - “Traffic” affects “being late” (dependency)

Bayesian Belief Networks (BBN)

- Directed Acyclic Graph (DAG): model dependencies
  - Nodes: random variables
  - Edges: conditional dependency
    - $D \to A$ means A depends on D

$$\begin{tikzcd} D \arrow[d] & \\ A \arrow[dr] & B \arrow[d] \\ & C \end{tikzcd}$$

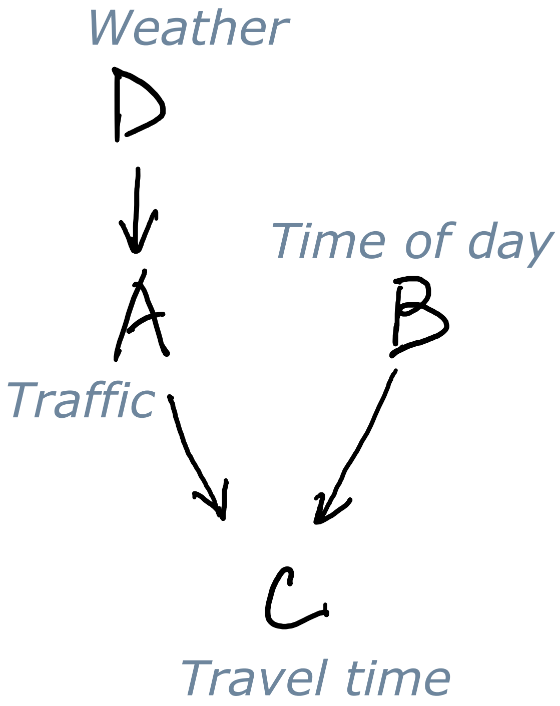
- Conditional Probability Tables (CPTs)
  - $A = \{0,1\}, B = \{0,1,2\}, C = \{0,1\}, D = \{0,1\}$
  - The probability of a random variable conditioned on its 'parents'

	A=0, B=0	A=0, B=1	A=0, B=2	A=1, B=0	A=1, B=1	A=1, B=2
C=0	0.7	0.6	0.2	0.9	0.75	0.3
C=1	0.3	0.4	0.8	0.1	0.25	0.7

Bayesian Belief Networks (BBN)

- $D \to A$ means A depends on D
  - D is called the parent of A.
  - A is called the descendant (child) of D.

- **Property**: Given its parents, a random variable is conditionally independent of its non-descendants
  - $P(B \mid A, D) = P(B)$
  - $P(C \mid A, B, D) = P(C \mid A, B)$
  - $P(A \mid C, D) \neq P(A \mid D)$



Classification Using BBN

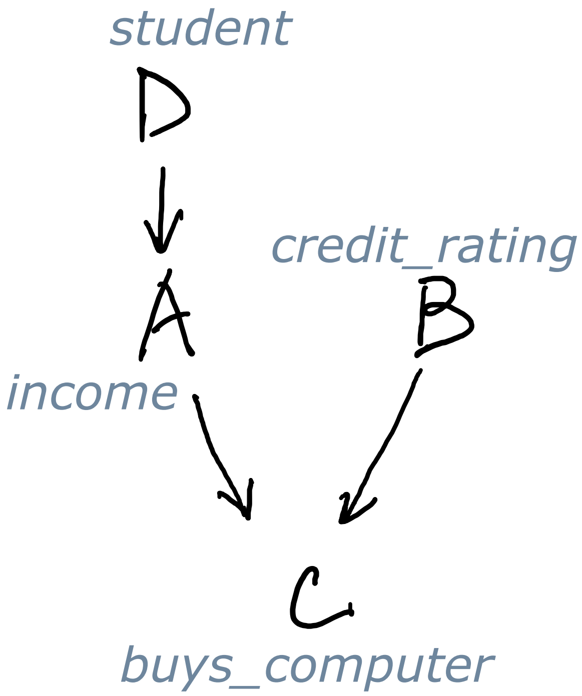
- Given an observation $X = (x_1, x_2, \dots, x_n)$, We calculate $P(C_i|X)$ for each class, and find C_i with the maximum $P(C_i|X)$
- For each class, apply Bayesian Theorem:

$$P(C_i \mid X) = \frac{P(X|C_i) \cdot P(C_i)}{P(X)} \propto P(X|C_i)P(C_i)$$

  - In Naïve Bayesian model, we assume $x_1, x_2, \dots, x_n$ are independent given C_i → $P(X \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \cdots \times P(x_n \mid C_i)$
  - In BBN, the relationship among $x_1, x_2, \dots, x_n$ and the calculation of $P(X \mid C_i)$ refer to the DAG and CPTs for (conditional) probabilities.

Classification Using BBN

income	student	credit_rating	Class: buys_computer



- ☐ Classify C given $A, B, D \Rightarrow$ Consider only A and B
- ☐ Classify A given $B, C, D \Rightarrow$ Consider only C and D
- ☐ How to classify D given A, B, C?

$$P(D \mid A, B, C) = \frac{P(A, B, C \mid D) \cdot P(D)}{P(A, B, C)}$$

- In Naïve Bayes Classifier,
  - ☐ $P(A, B, C|D) = P(A|D) \times P(B|D) \times P(C|D)$
- In Bayesian Belief Network,
  - ☐ $P(A, B, C|D)$
    $= P(A|D) \times P(B|A, D) \times P(C|A, B, D)$
    $= P(A|D) \times P(B) \times P(C|A, B)$

Chain rule for joint probability:

$P(A_1, A_2, A_3, \dots, A_n)$
$= P(A_1) \times P(A_2|A_1) \times P(A_3|A_1, A_2) \times \cdots$
$\times P(A_n|A_1, A_2, A_3, \dots, A_{n-1})$

An Example

- Given a person (X) who buys a computer, and whose income is medium and credit_rating is fair, decide if X is a student.
  - $X(income = \text{medium}, credit\_rating = \text{fair}, buys\_computer = \text{yes})$

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

An Example: Naïve Bayesian Classifier

$$\boldsymbol{P(student = yes \mid income = medium, credit\_rating = fair, buys\_computer = yes)}$$

$$\propto P(income = medium, credit\_rating = fair, buys\_computer = yes \mid student = yes)$$

$$\times\ P(student = yes)$$

$$= P(income = medium \mid student = yes) \times P(credit\_rating = fair \mid student = yes)$$

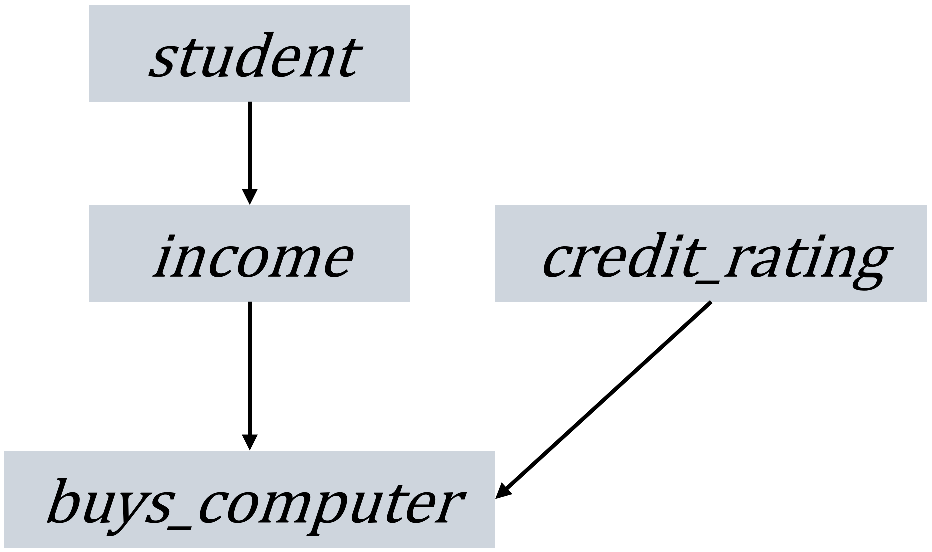
$$\times P(buys\_computer = yes \mid student = yes)\ \times\ P(student = yes)$$

$$= \frac{2}{7} \times \frac{4}{7} \times \frac{6}{7} \times \frac{1}{2} = \frac{24}{343} \approx 0.07$$

Similarly, $P(income = medium, credit\_rating = fair, buys\_computer = yes | student = no)\ \times$ $P(student = no) = \frac{4}{7} \times \frac{4}{7} \times \frac{3}{7} \times \frac{1}{2} = \frac{24}{343} \approx 0.07$

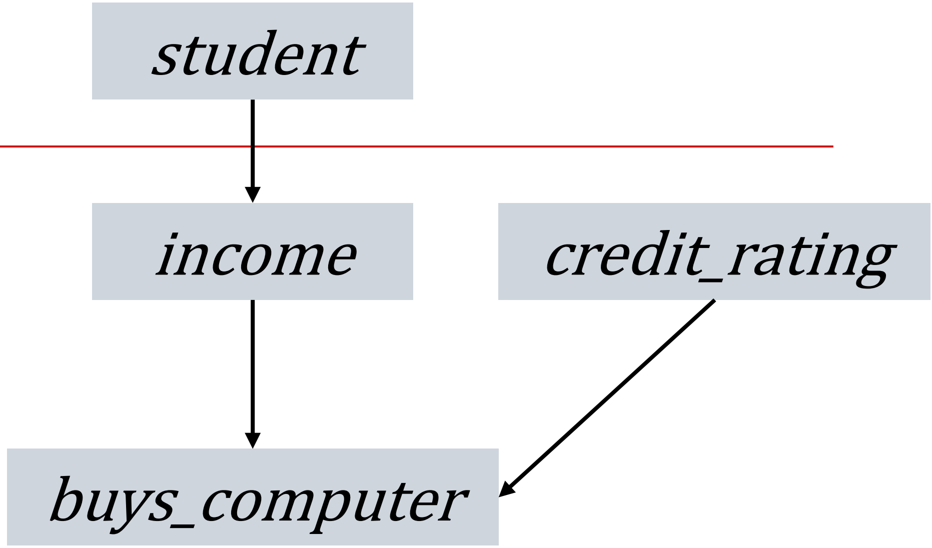
An Example: Using BBN

- Directed Acyclic Graph (DAG)



RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

An Example: Using BBN



- Conditional Probability Tables (CPTs)

income (i)	*student* = yes	*student* = no
low	4/7	0.0
medium	2/7	4/7
high	1/7	3/7

buys_computer	i=h, c=excellent	i=h, c=fair	i=m, c=e	i=m, c=f	i=l, c=e	i=l, c=f
yes	0.0	2/3	2/3	2/3	1/2	1.0
no	1.0	1/3	1/3	1/3	1/2	0.0

An Example: Using BBN

$$\boldsymbol{P(student = yes|income = medium, credit\_rating = fair, buys\_computer = yes)}$$

$$\propto P(income = medium, credit\_rating = fair, buys\_computer = yes|student = yes)$$

$$\times\ P(student = yes)$$

$$= P(income = medium|student = yes) \times P(credit\_rating = fair|student = yes, income = medium)$$

$$\times P(buys\_computer = yes|student = yes, income = medium, credit\_rating = fair) \times\ P(student = yes)$$

$$= P(income = medium|student = yes) \times P(credit\_rating = fair)$$

$$\times P(buys\_computer = yes|income = medium, credit\_rating = fair)\ \times\ P(student = yes)$$

$$= \frac{2}{7} \times \frac{8}{14} \times \frac{2}{3} \times \frac{1}{2} = \frac{8}{147} \approx 0.054$$

Similarly, $P(income = medium, credit\_rating = fair, buys\_computer = yes|student = no)\ \times\ P(student = no) = \frac{4}{7} \times \frac{8}{14} \times \frac{1}{3} \times \frac{1}{2} = \frac{8}{147} \approx 0.054$

SUPPORT VECTOR MACHINES (SVM)

A Binary Classification Problem

- The dataset contains a set of points: $D = \{X_1, X_2, \dots, X_{|D|}\}$
  - Each point X is represented as a d-dimensional vector, i.e., $X = (x_1, x_2, \dots, x_d)$
  - Each point X is associated with a label y_i, where $y_i \in \{+1, -1\}$

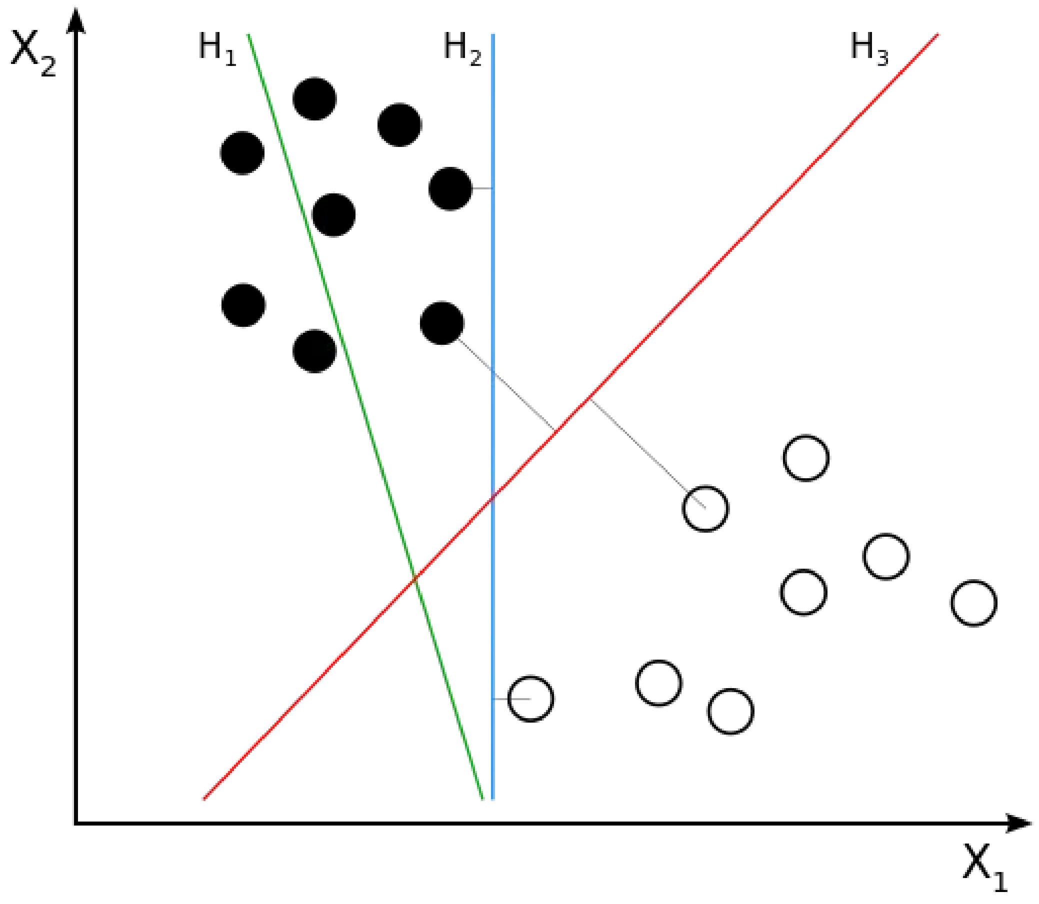
- A hyperplane H can separate these points into two parts:
  - $H\colon W \cdot X + b = 0$

    where $W = (w_1, w_2, \dots, w_d)$ and $W \cdot X = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d$

- We call H a decision boundary in classification problems.

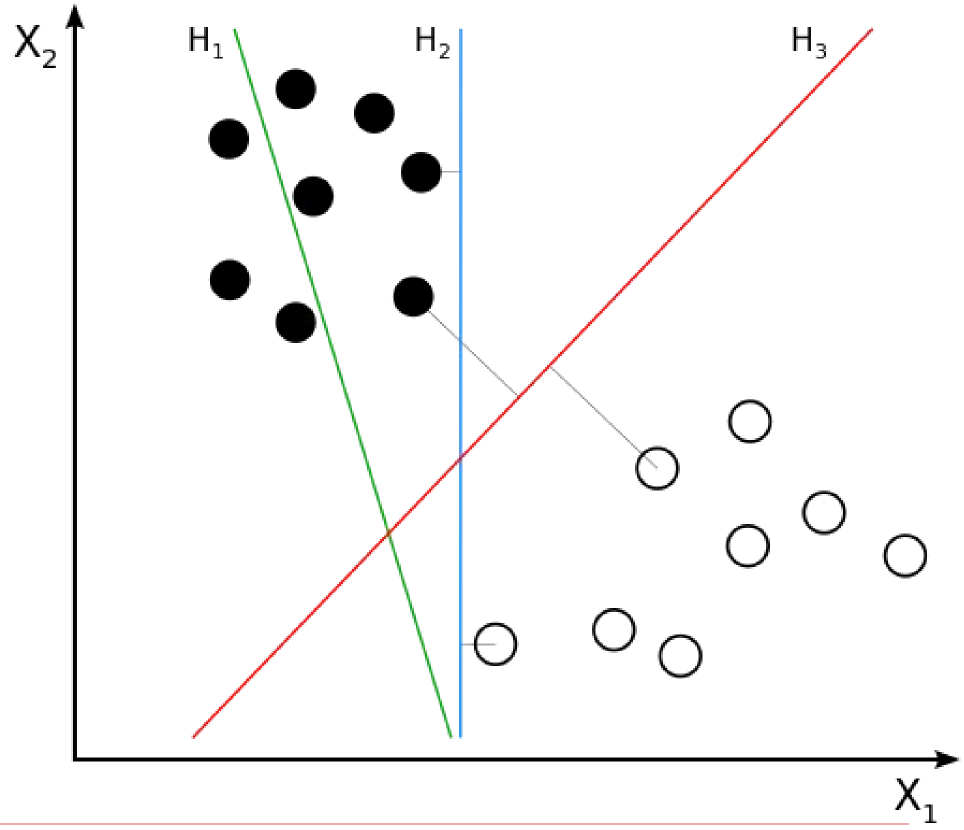
What is a good classifier?

- H_1: can't separate data at all
- H_2: can separate data, but can't be generalized well
- H_3: can separate data, and can be generalized well



Support Vector Machines

- A supervised machine learning (ML) algorithm, aiming to find the best decision boundary (*hyperplane*) that separate data points into distinct classes

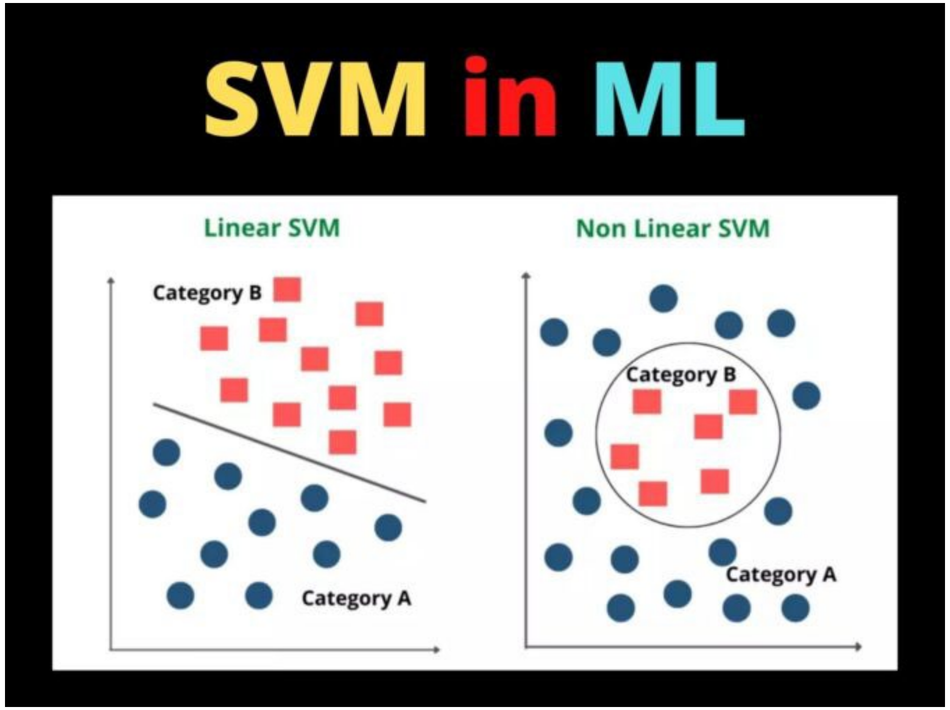


- **Hyperplane**: a line for 2D data; a plane for 3D data; a hyperplane for higher dimensional space

- **Margin**: the sum of distances between the hyperplane and the nearest data points from each class (called support vectors).
  - SVM maximizes this margin to ensure better generalization.

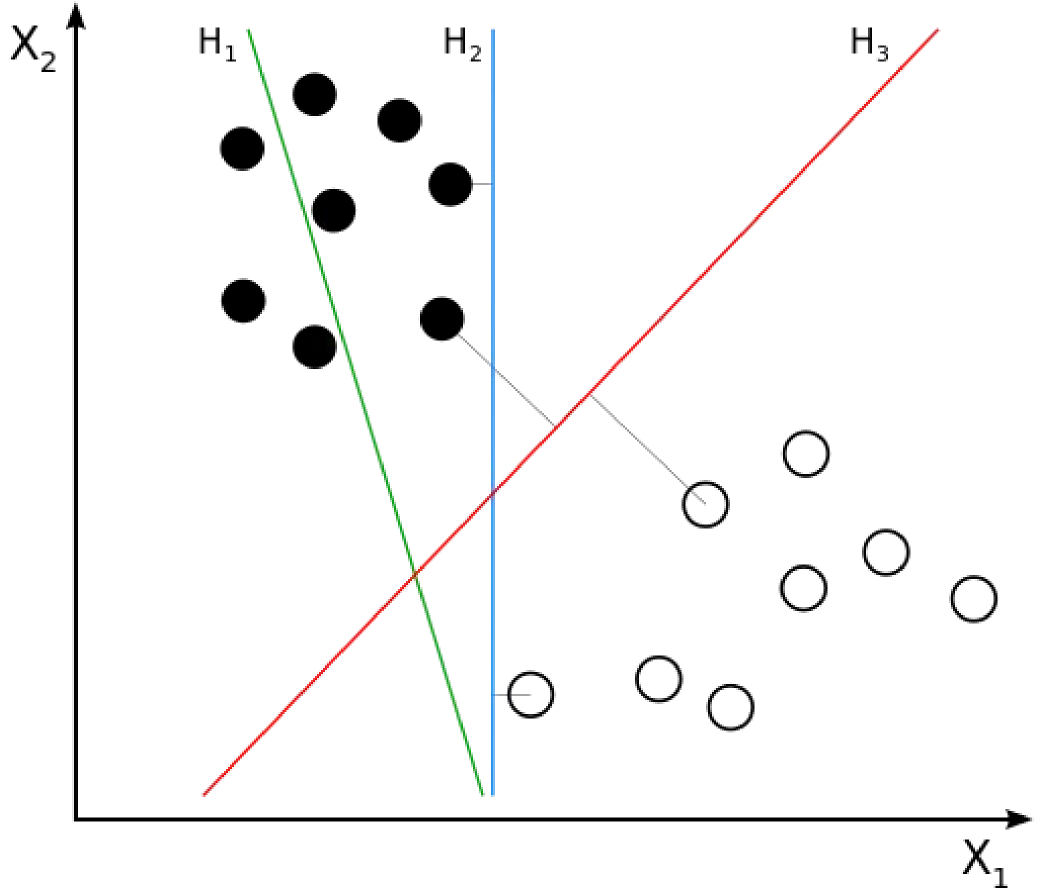
Why using SVM for classification?

- Better generalized on unseen data
  - Maximizes the margin between classes to ensure robust predictions
- Handle both linear (hyperplane) and non-linear (kernel) classification
- Robust to dimensionality and overfitting
  - Effective in high-dimensional spaces
  - Focus on critical data points (support vectors) to avoid overfitting



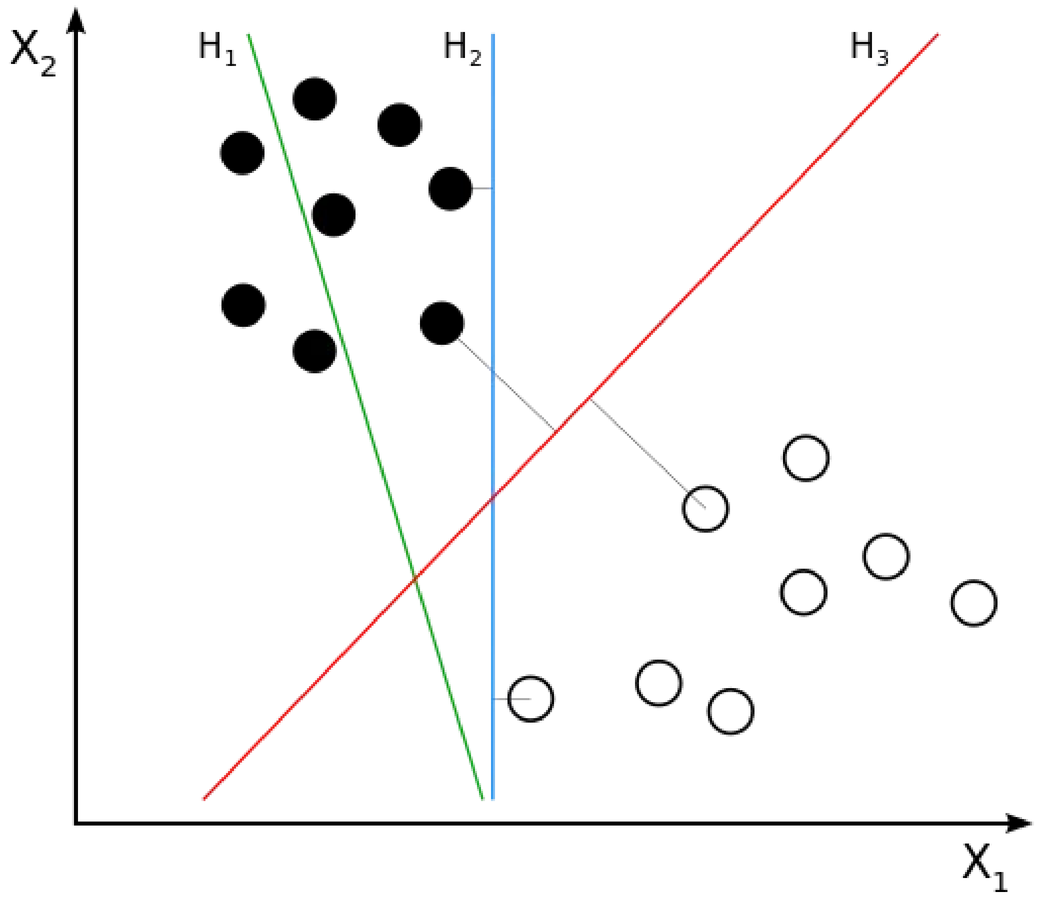
Support Vectors

- Distance between a point X and a hyperplane H:
  - $d(X,H) = \frac{|W \cdot X + b|}{\|W\|}$
  - $\|W\| = \sqrt{w_1^2 + w_2^2 + \cdots + w_d^2}$
- The points closest to the decision boundary in either class are support vectors



Support Vectors

- The sum of distances between the decision boundary and the closest data points from either class is called the margin
- The goal of SVM is to find the best decision boundary with maximum margin
  - always let the distances from the decision boundary to the support vectors to be equal



Finding the Optimal Decision Boundary

- Let the optimal optimal decision boundary $H: W \cdot X + b = 0$.
- Let two support vectors be X_1 and X_2.

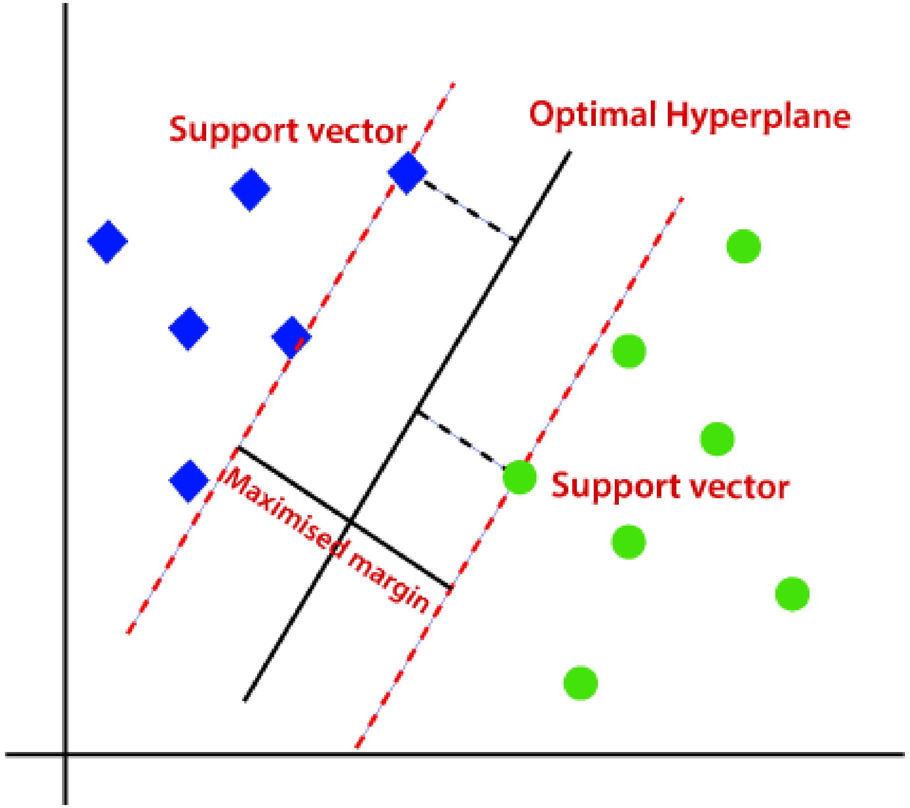
Define two hyperplanes parallel to H and passing the two support vectors

- $\boldsymbol{H_1: W \cdot X + b = 1}$
  - For X_1 , $W \cdot X_1 + b = 1$
  - For points above H_1, $W \cdot X + b \geq 1$
- $\boldsymbol{H_2: W \cdot X + b = -1}$
  - For X_2 , $W \cdot X_2 + b = -1$
  - points below H_2 with $W \cdot X + b \leq -1$

The margin can be calculated by:

$$m = \frac{|W \cdot X_1 + b| + |W \cdot X_2 + b|}{\|W\|} = \frac{2}{\|W\|}$$

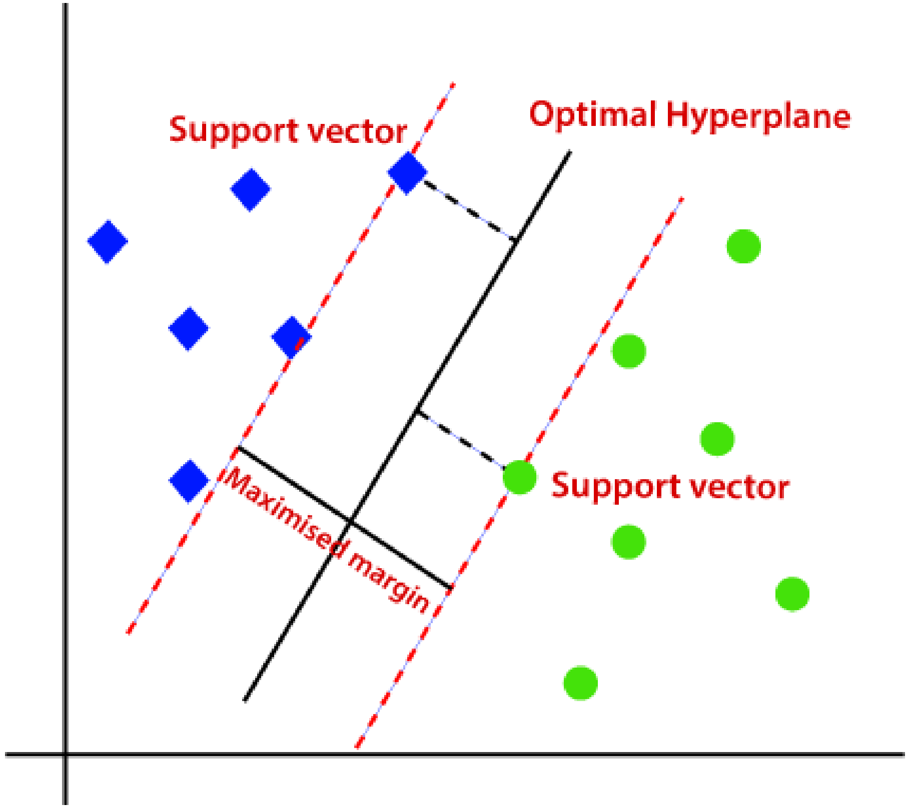
- So, W and b can be found by maximizing $\frac{2}{\|W\|}$



Classification Using SVM

- Given an unseen point X_q, decide its label y_q

- Calculate $W \cdot X_q + b$
  - If $W \cdot X_q + b > 0 \Rightarrow y_q = +1$
  - If $W \cdot X_q + b < 0 \Rightarrow y_q = -1$



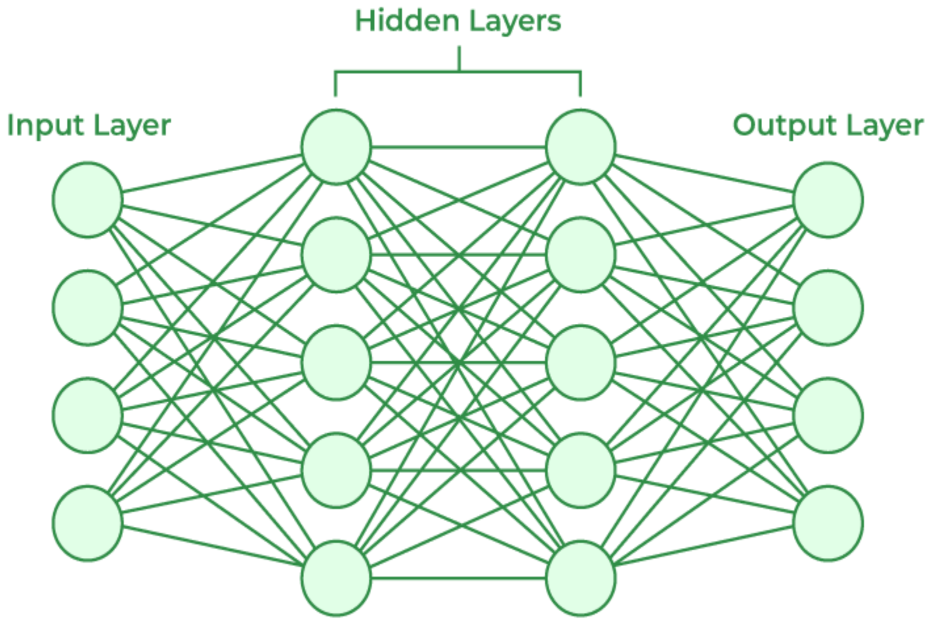
NEURAL NETWORKS (NN)

Neural Networks (NN)

- A machine learning model inspired by the human brain, consisting of interconnected layers of nodes (neurons) that process data and learn patterns.

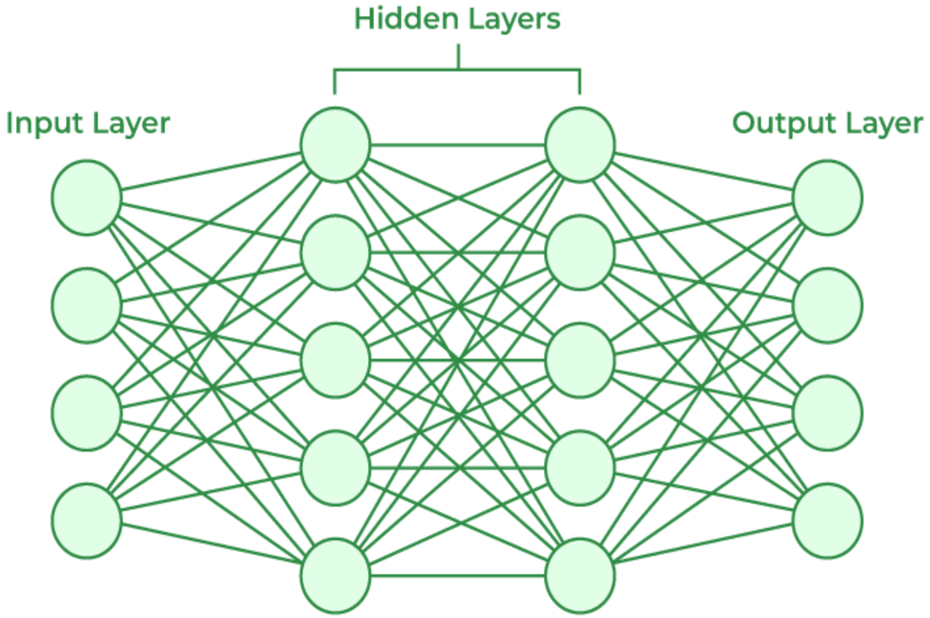
Structure of NN

- **Input Layer**: take in raw data features
- **Hidden Layer(s):** perform computations and learn representations through weights and bias
- **Output Layer**: produce final results



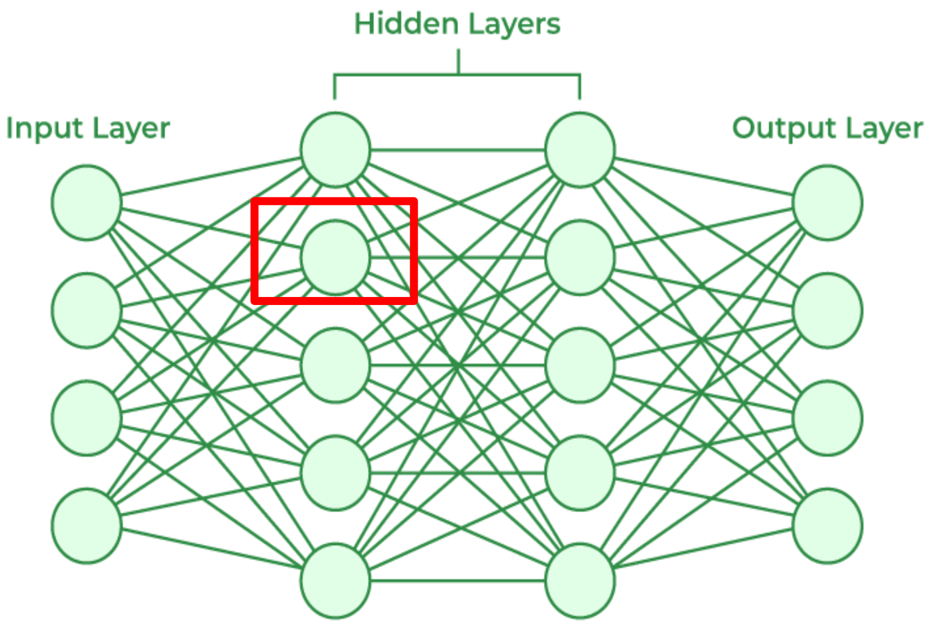
Classification Using NNs (I)

- Input of NN
  - A tensor $\boldsymbol{X}$ of size (n, d)
    - n is the number of points
    - d is the dimensionality of data
    - $\boldsymbol{X} = \begin{pmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{pmatrix}$



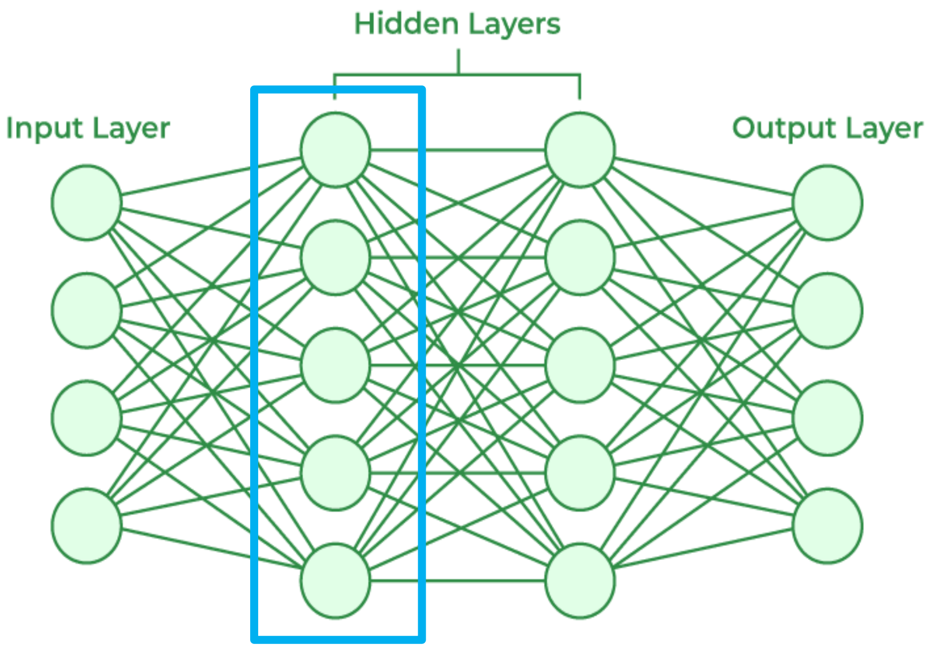
Classification Using NNs (II)

- Inside a neuron (unit)
  - Take the input from data or the output of other neurons as input
  - The output of a neuron for one point is computed as:
    - $w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + b$
    - weighted by $(w_1, w_2, \ldots, w_d)$
    - The bias b is to adjust outputs
  - The weights and bias are trainable parameters



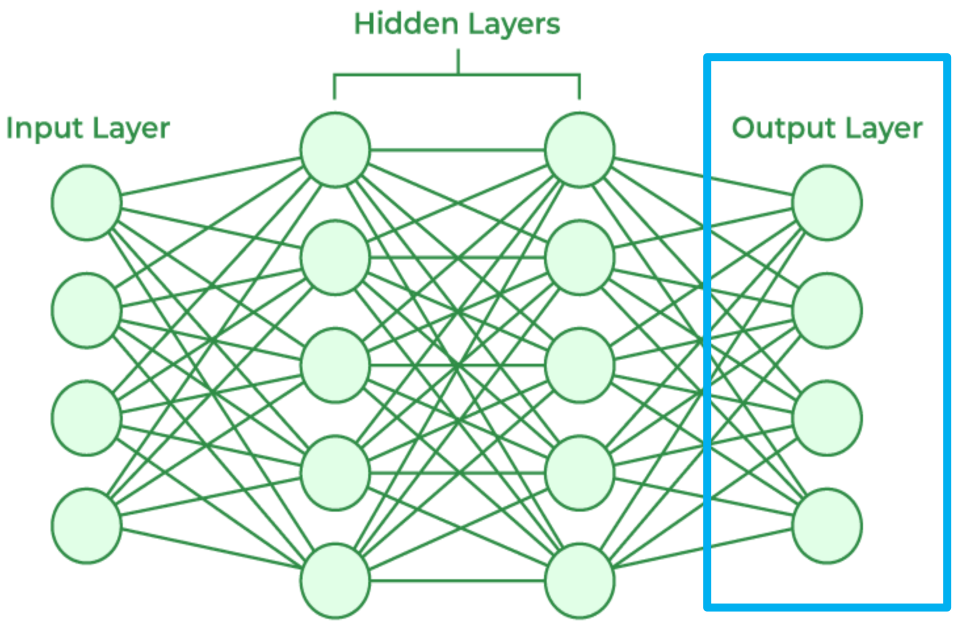
Classification Using NNs (III)

- Inside a hidden layer with m neurons
  - Weights: $\boldsymbol{W}$ of size (m, d)
    - $$\boldsymbol{W} = \begin{pmatrix} w_{11} & \cdots & w_{1d} \\ \vdots & \ddots & \vdots \\ w_{m1} & \cdots & w_{md} \end{pmatrix}$$
  - Bias: $\boldsymbol{b}$ of size $(m, 1)$
    - $$\boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$
  - The output of this hidden layer: a matrix of size (n, m), i.e., $\boldsymbol{X} \cdot \boldsymbol{W}^T + \boldsymbol{b}^T$



Classification Using NNs (IV)

- In the output layer with c neurons
  - $\boldsymbol{O}$: a tensor of size (n, c)
    - c is the number of classes
  - Generate the estimated labels $\widetilde{\boldsymbol{Y}}$
  - Calculate $Loss = f(\widetilde{\boldsymbol{Y}}, \boldsymbol{Y})$, where $\boldsymbol{Y}$ is the real labels
    - e.g., cross-entropy for classification
  - Backpropagate the loss and adjust the trainable parameters across all layers to minimize loss
    - using gradient descent or other optimization algorithms

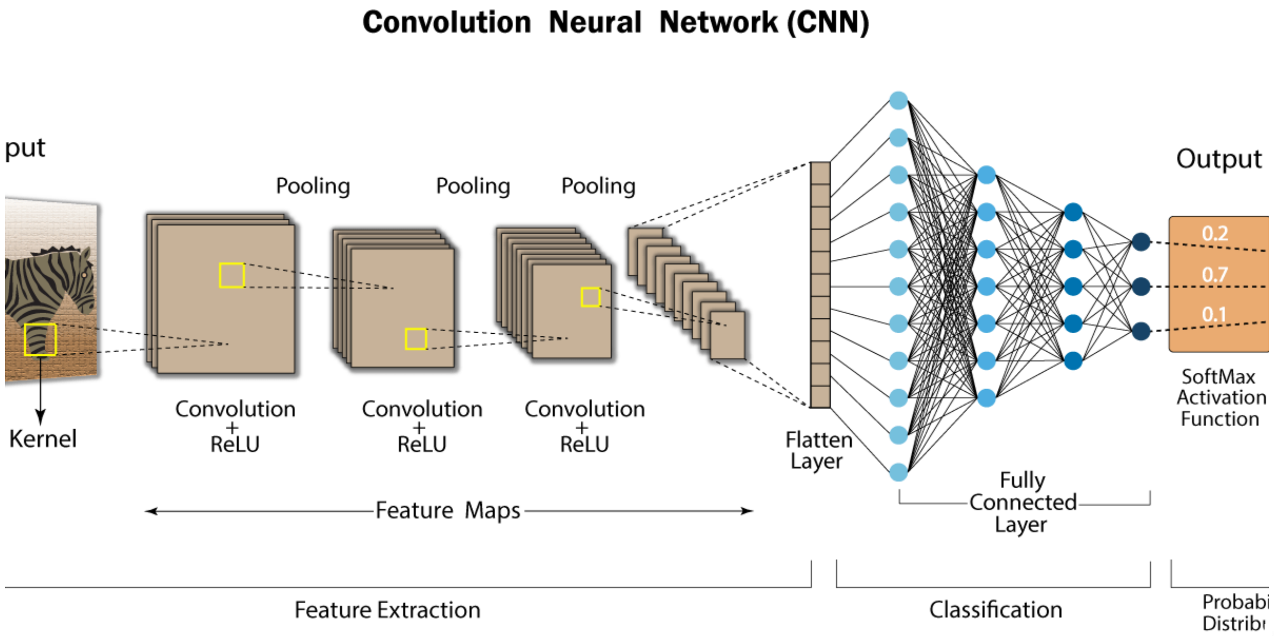


Some Further Discussions

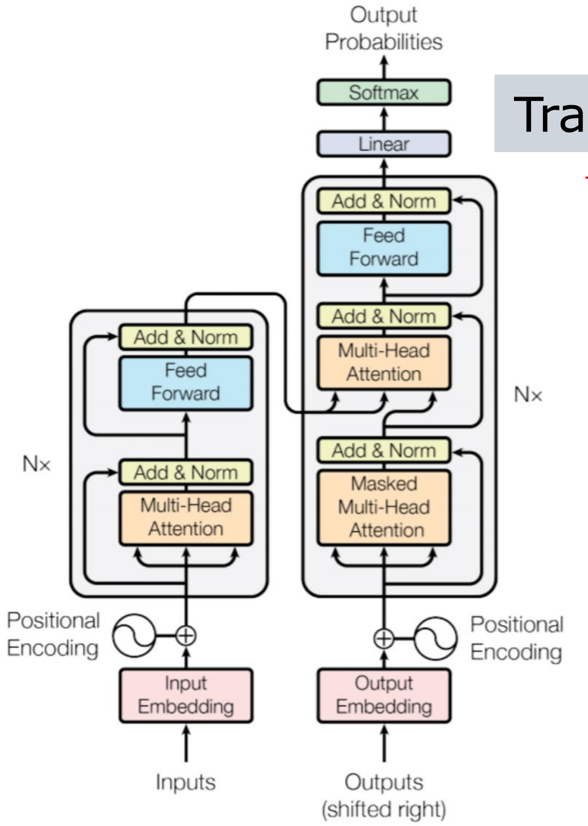
- **Why using NN for classification?**
  - High accuracy
  - Fast evaluation speed once trained
  - Robust to noises

- **Is NN enough for every task? No.**
  - Long training time
  - Can’t be generalized well
  - Highly dependent on the quality of training data
  - Not explainable

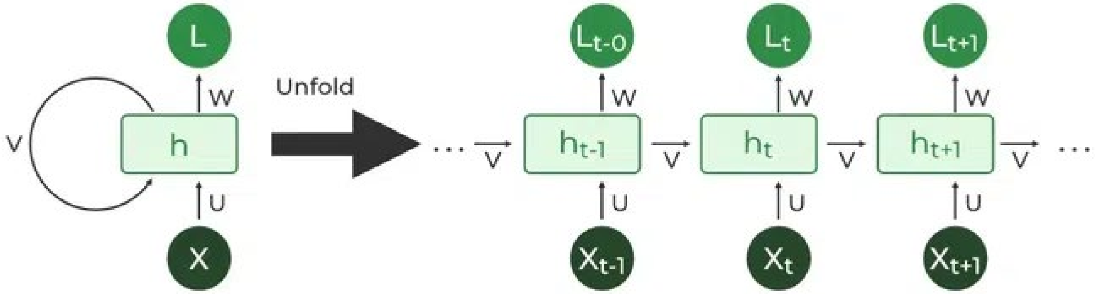
Other Advanced NNs



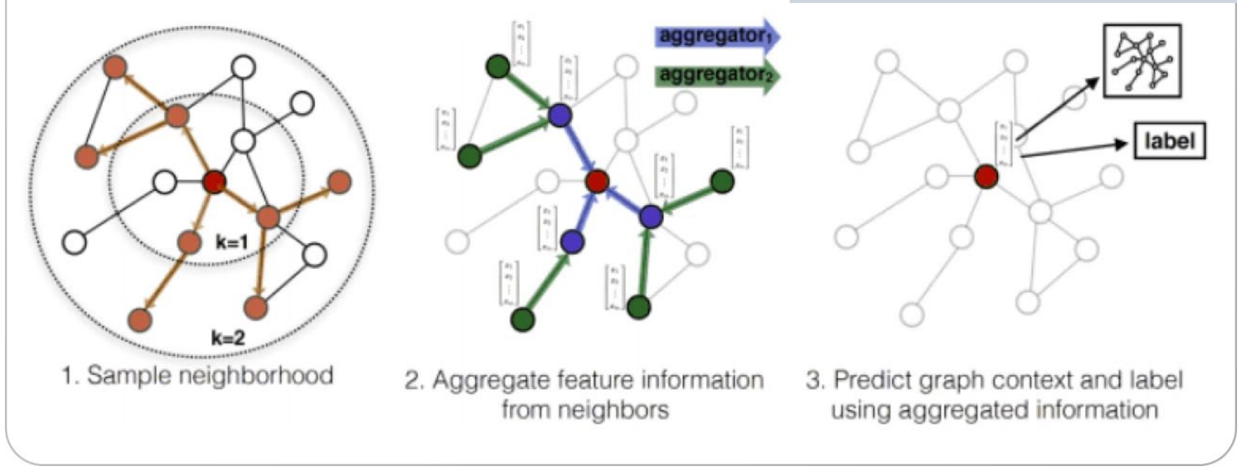
Transformer



Recurrent NN (RNN)



Graph NN (GNN)



Summary

- **Bayesian Belief Networks (BBN):** A probabilistic model that represents variables' dependencies using DAG and CPTs
  - Reasoning under uncertainty but computationally expensive
- **Support Vector Machines (SVM):** A supervised learning model that finds the optimal hyperplane to separate data
  - Optimization may struggle with very large datasets
- **Neural Networks (NN):** A machine learning model inspired by the human brain, consisting of interconnected layers of neurons that learn patterns in data
  - High accuracy, long training, "black box" model

Email: fengmei.jin@polyu.edu.hk

Office: PQ747

THANK YOU!