COIT20277 Introduction to Artificial Intelligence

Week 2 - Lecture

- Machine Learning Overview
- Supervised Learning: Classification

BE WHAT YOU WANT TO BE
cqu.edu.au

Acknowledgement of Country

I respectfully acknowledge the Traditional Custodians of the land on which we live, work and learn. I pay my respects to the First Nations people and their Elders, past, present and future

CQUniversity AUSTRALIA

BE WHAT YOU WANT TO BE
cqu.edu.au

Acknowledgment

The content of this lecture has been adopted from the following book:

- Artificial Intelligence Programming with Python - From Zero to Hero, 2022, Perry Xiao, *John Wiley & Sons, Inc.*, ISBN 978-1-119-82086-4.

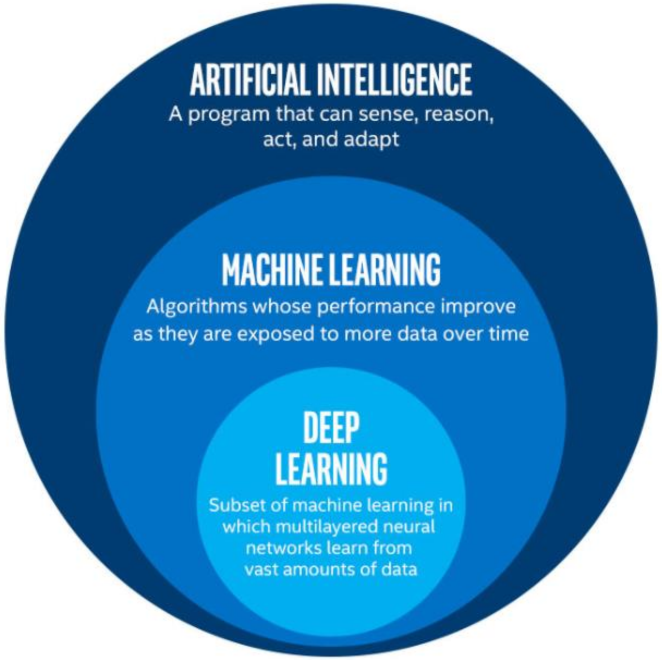
- Chapter 3 (Sections 3.1 and 3.2)

Outline

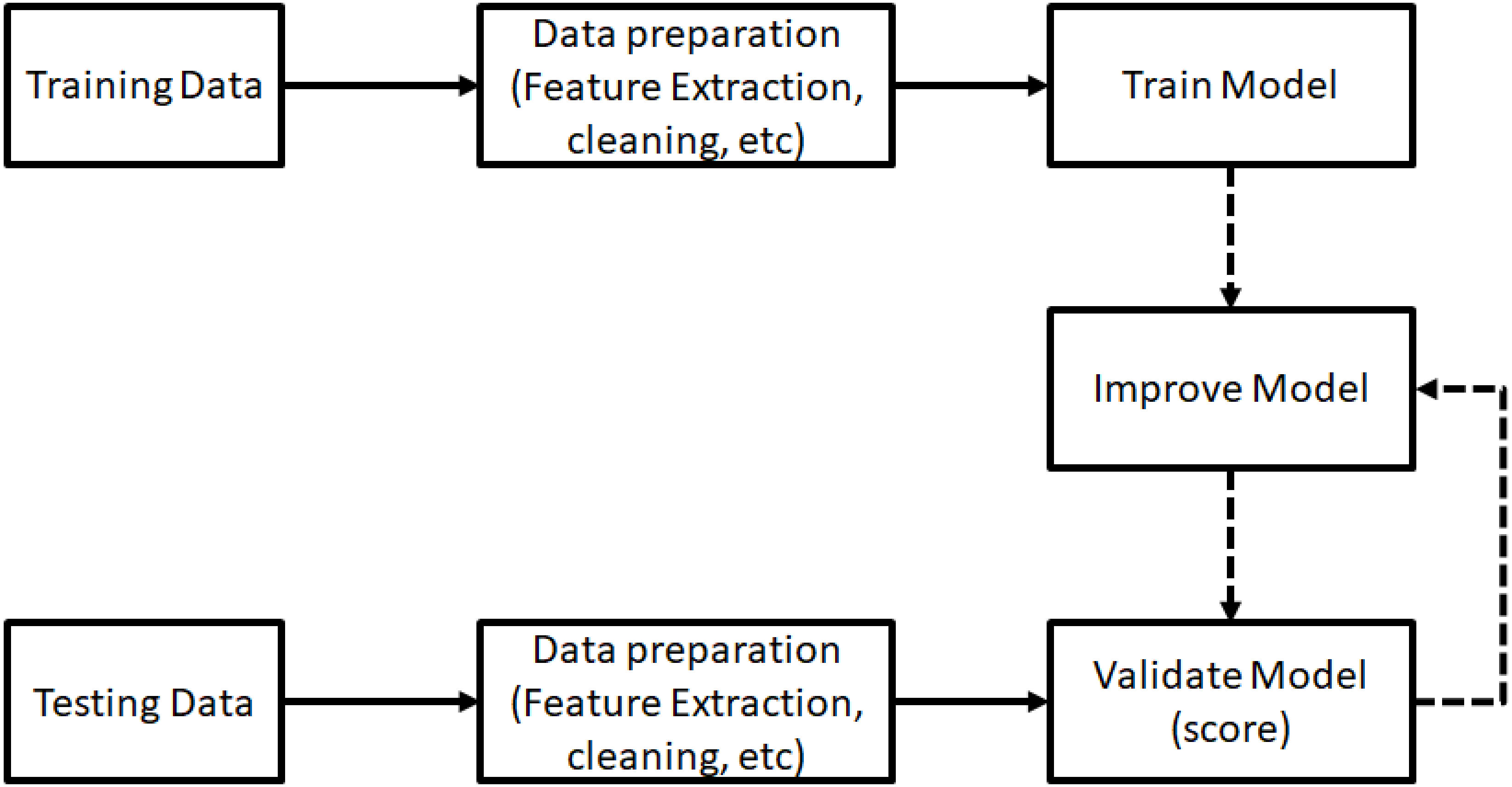
- What is Machine Learning?
- History of Machine Learning
- Types of Machine Learning
- Applications of Machine Learning
- Supervised Learning: Classifications
- Popular supervised learning algorithms
- Ready-made datasets in Scikit-Learn
- Support Vector Machines (SVM)
- Naïve Bayes
- Decision Trees and Random Forests
- K-Nearest Neighbors (K-NN)

What is Machine Learning (ML)?

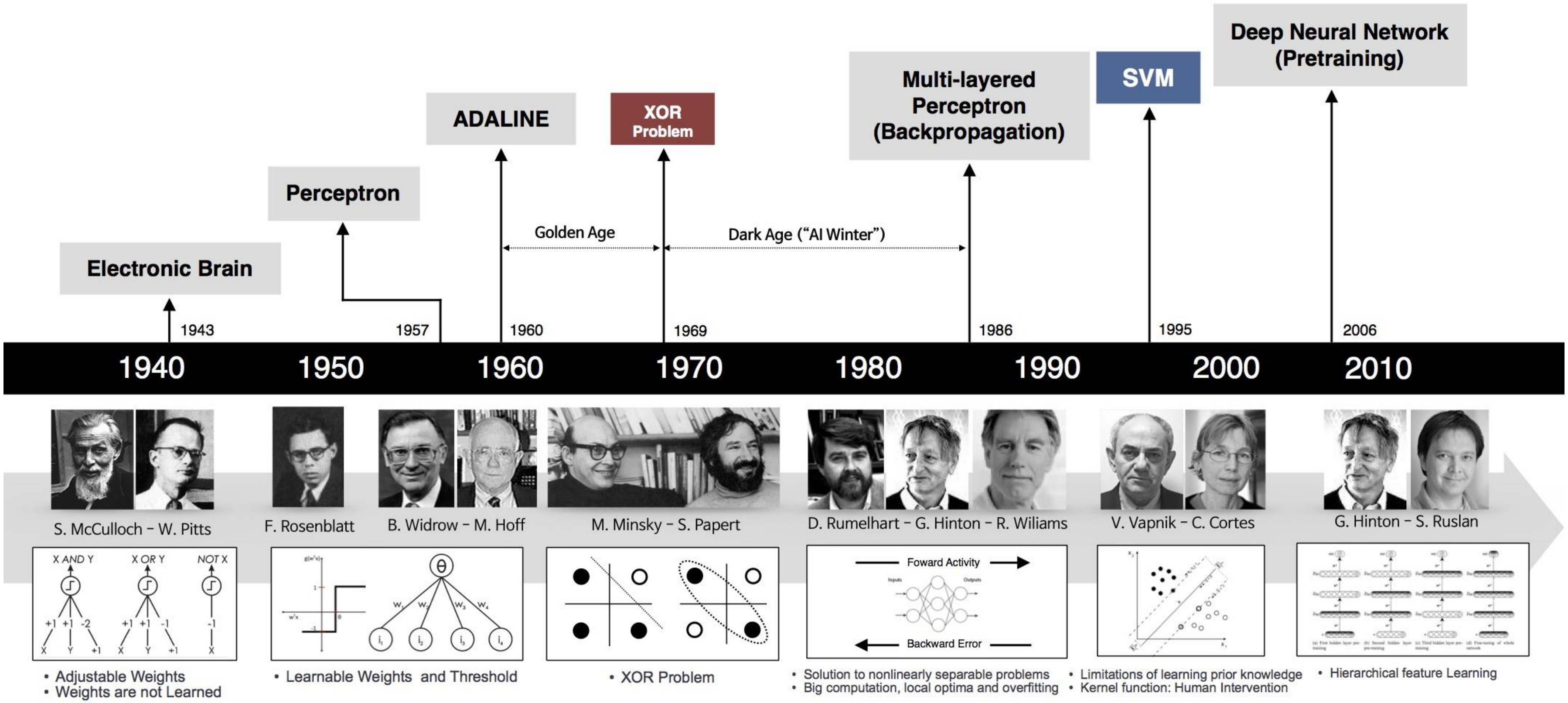
- Machine learning is a subset of AI.
- It involves teaching computers to learn from data and analyze data automatically, without human intervention.
- It includes a set of mathematical algorithms that can make decisions or predict results for a given set of data.



Machine Learning Pipelines



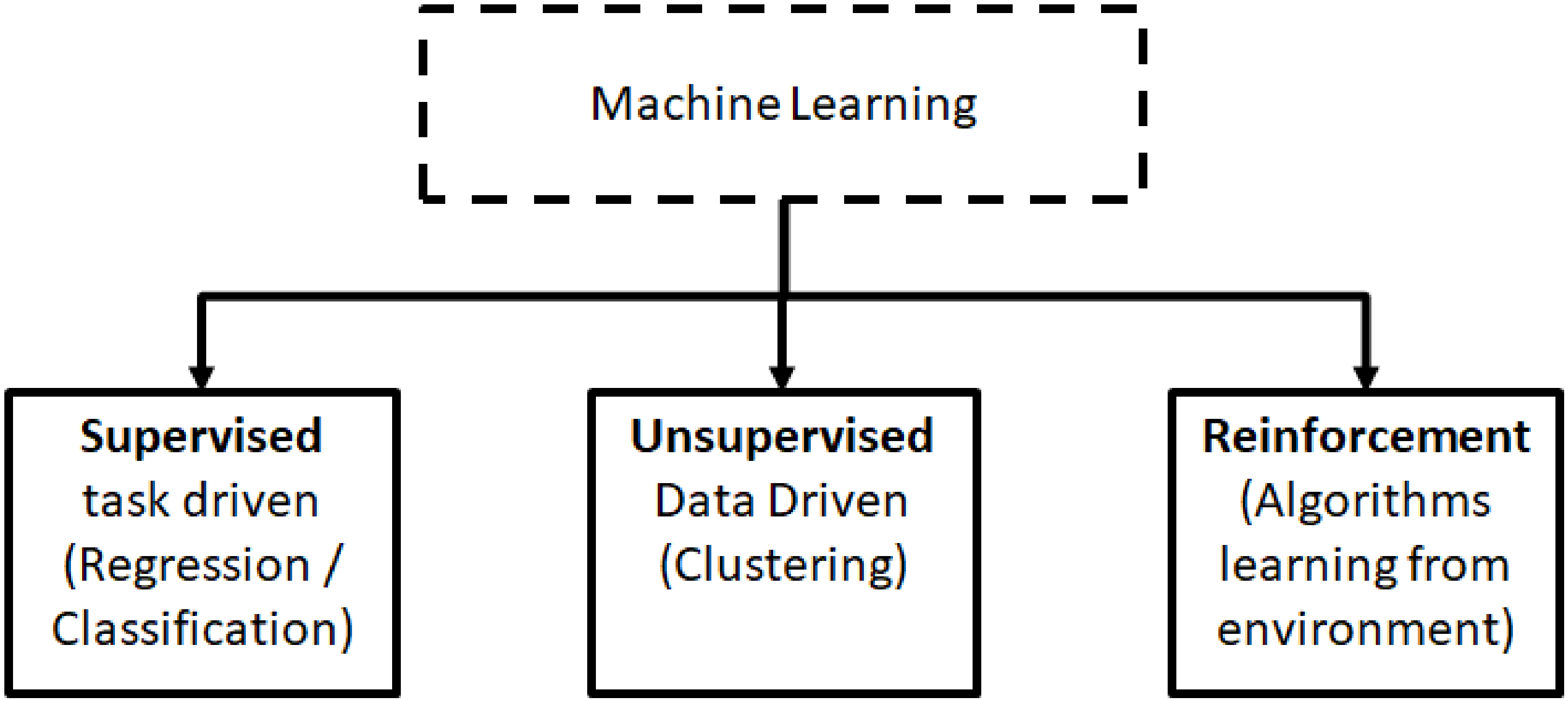
History of Machine Learning



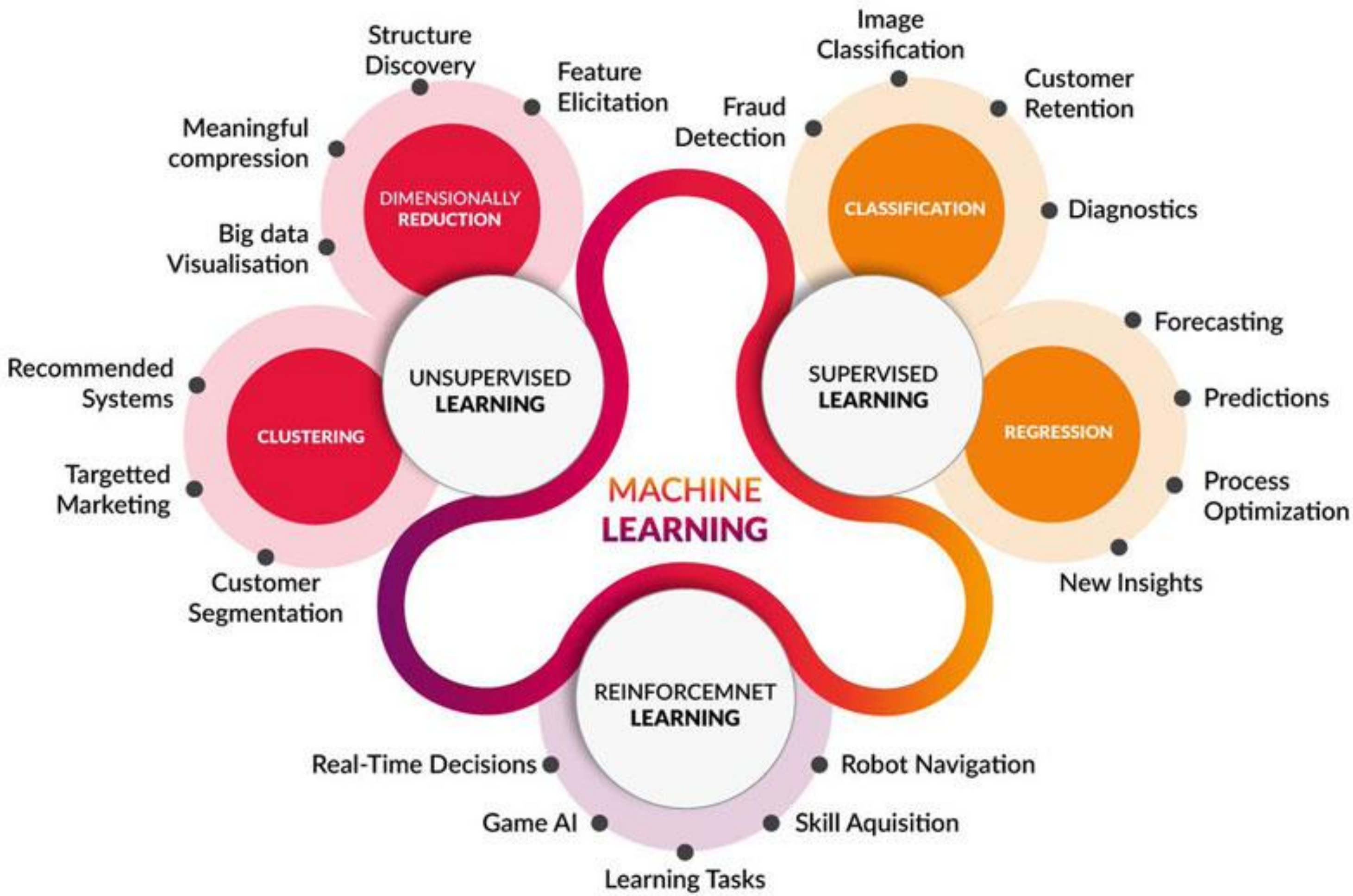
Current Landscape of Machine Learning

- ***Widely adopted across industries***: Machine learning is being used in diverse sectors like healthcare, finance, and retail to enhance decision-making and customer experience.
- ***Dominance of deep learning***: Techniques such as CNNs and RNNs are prevalent, especially in image recognition, natural language processing, and speech recognition.
- ***Ethical considerations***: Growing awareness of bias, fairness, privacy, and accountability is influencing the development and deployment of AI systems.
- ***Interdisciplinary collaboration***: Collaboration between experts from various fields is common, promoting holistic approaches to AI development and deployment.
- ***Democratization of AI***: Access to machine learning tools and platforms is becoming more widespread, enabling individuals and organizations to build AI solutions with less expertise.

Types of Machine Learning

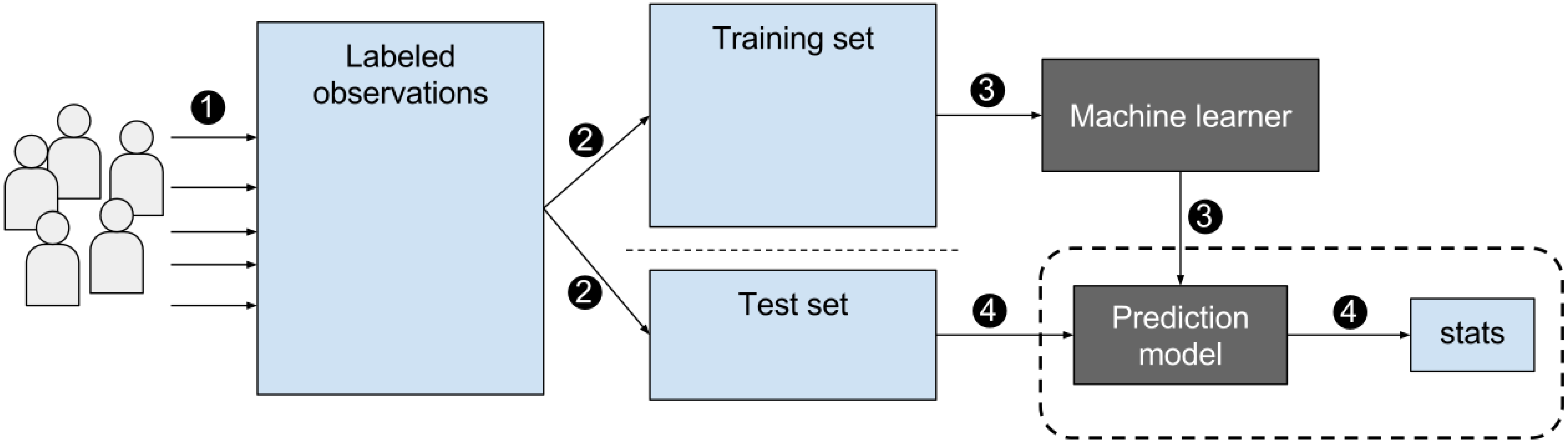


Applications of Machine Learning



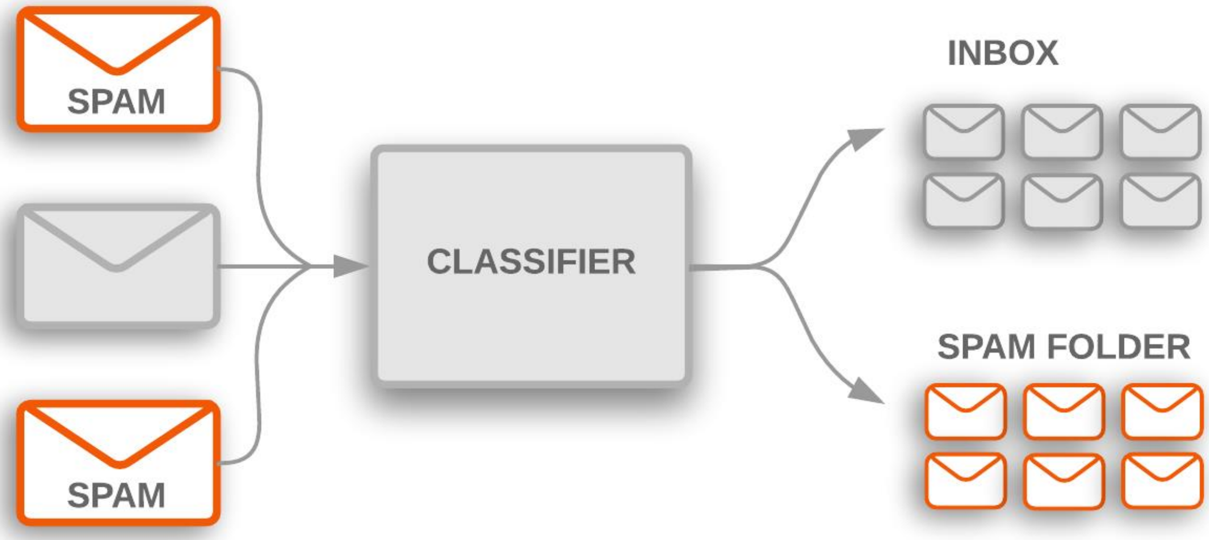
Supervised Learning

- Supervised learning is the most important part of machine learning, used for both classification and regression.
- Classification focuses on predicting the category a sample belongs to.
- Key terms: classes (categories), features (measurements), samples (data points), and parameters (model variables).



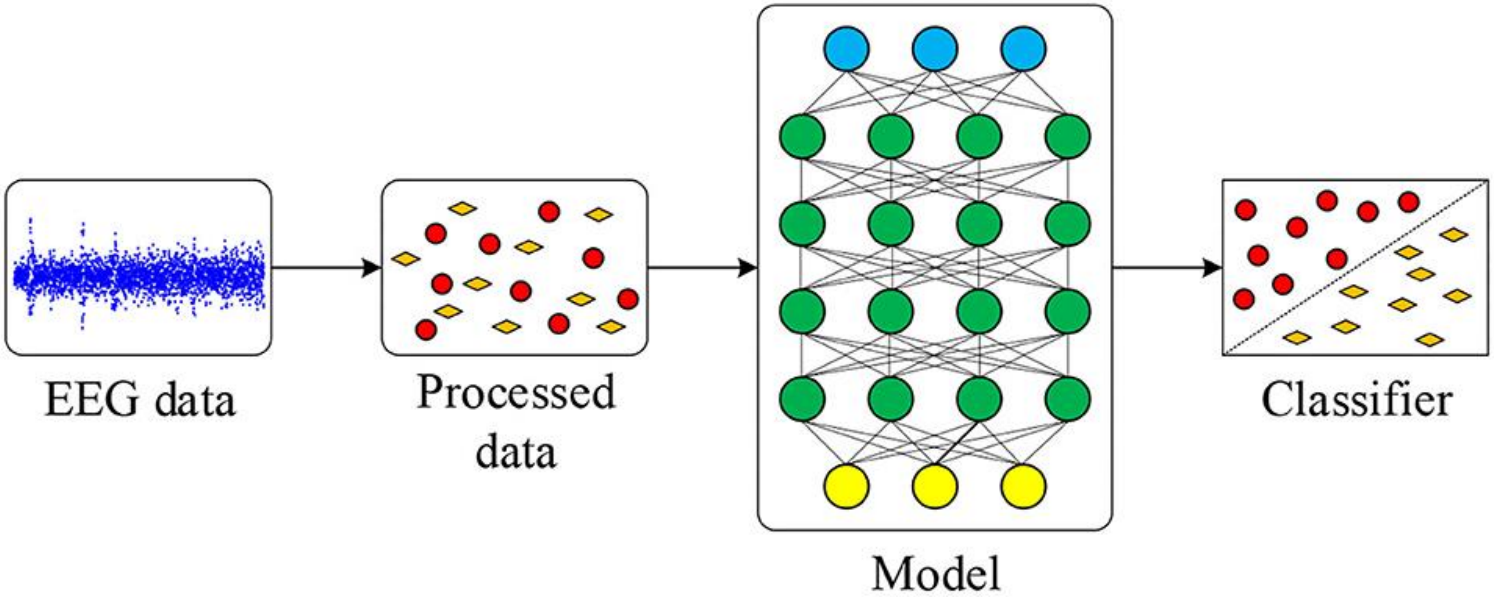
Classification Example

- **Spam email filtering** utilizes supervised learning to categorize emails into spam or non-spam categories.
- Data is labeled as either spam or non-spam, serving as the training set for the classification model.



Classification Example (cont…)

- **EEG signals classification** involves categorizing brain wave patterns recorded through electroencephalography (EEG) into specific classes.
- Using supervised learning, EEG signals are associated with corresponding outputs (class labels).



Popular Algorithms

- Support Vector Machines (SVM)
- Naïve Bayes
- Linear Discriminant Analysis (LDA)
- Principal Component Analysis (PCA)
- Decision Trees
- Random Forest
- K-Nearest Neighbors (K-NN)
- Artificial Neural Networks (Multilayer Perceptron)
- Find more details and examples using Scikit-Learn: https://scikit-learn.org/stable/supervised_learning.html

What is Scikit-learn?

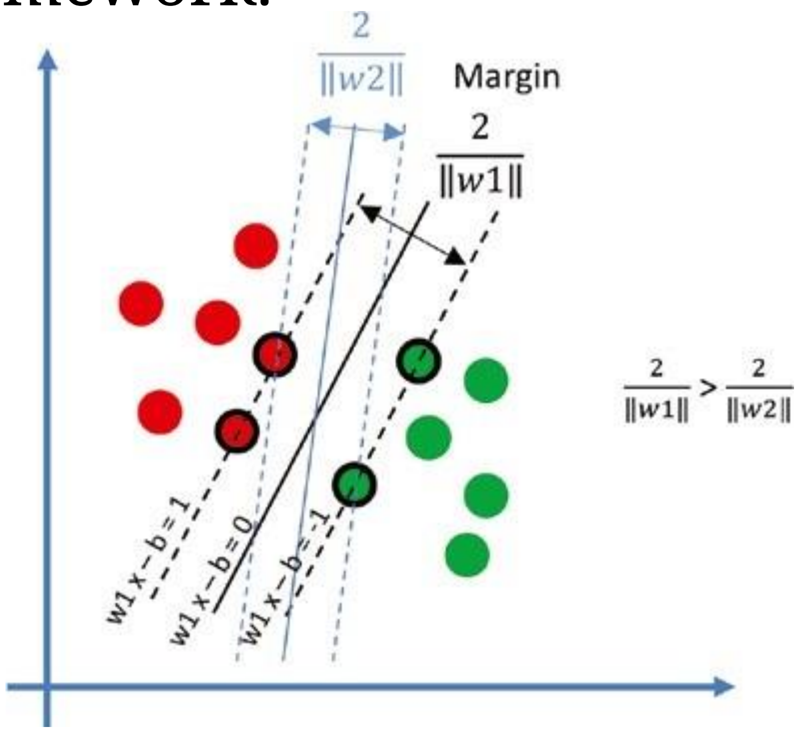
- Scikit-learn: Open-source Python library for machine learning.
- Features: Comprehensive, user-friendly, efficient, integrates with other Python libraries.
- Algorithms: Supervised and unsupervised learning algorithms included.
- Applications: Data preprocessing, model evaluation, real-world tasks like predictive modeling.
- Community: Active, with extensive documentation and tutorials available.

Scikit-Learn Datasets

- Ready-made datasets are available for easy access:
  - Toy datasets (e.g., iris flowers, breast cancer).
  - Real-world datasets (e.g., forest cover types).
  - Generated datasets.
  - Find more details: https://scikit-learn.org/stable/datasets.html
- Things to keep in mind:
  - Toy datasets may not capture all complexities of real-world problems.
  - Real-world datasets may require further preprocessing and exploration.
  - Always consider the specific research question and data availability when choosing datasets.

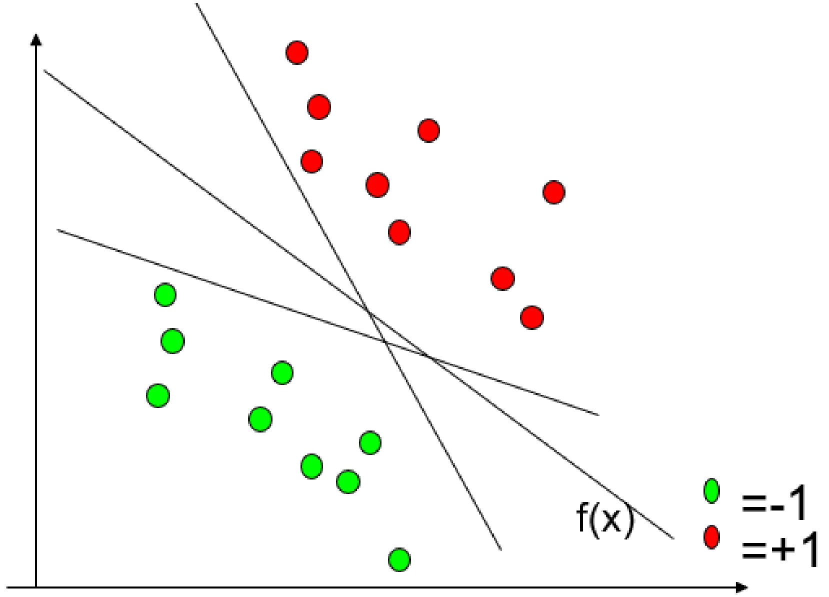
Support Vector Machine (SVM)

- SVM can work on both classification and regression problems.
- Proposed by Vapnik at AT&T Bell Labs in 1963, it uses a statistical learning framework.
- Two-category classification with SVM, separating data points using a hyperplane (straight line in 2-D).
- SVM adjusts the hyperplane to maximize the margin between data points.



Linear SVM

- All hyperplanes in R^d are parameterised by a vector (**w**) and a constant b.
- Can be expressed as **w•x**+b=0 (remember the equation for a hyperplane from algebra!)
- Our aim is to find such a hyperplane f(**x**)=sign(**w•x**+b), that correctly classify our data.



Data: $<\mathbf{x}_i, y_i>$, i=1,..,l
$\mathbf{x}_i \in R^d$
$y_i \in \{-1,+1\}$

Linear SVM (Definition)

- Define the hyperplane H such that:

  $\mathbf{w} \bullet \mathbf{x_i} + b \geq +1$ when $y_i = +1$

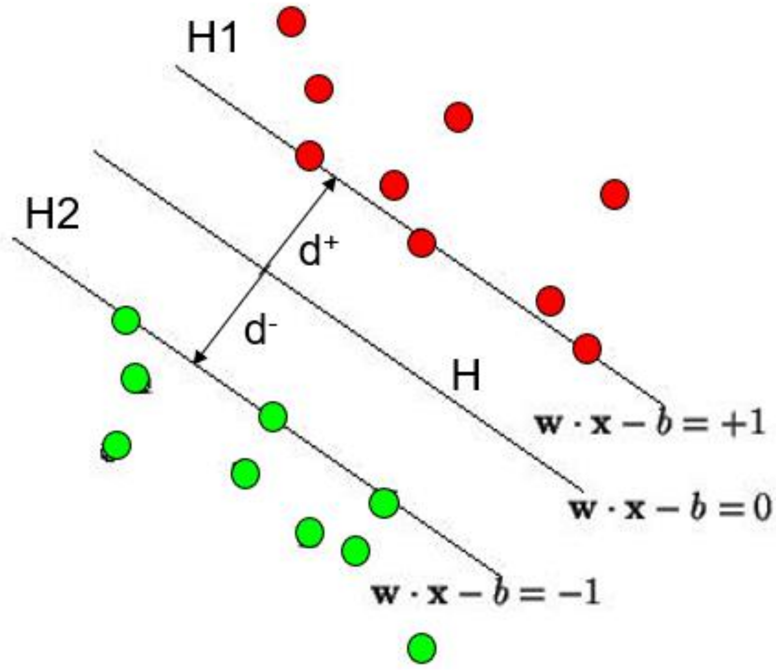
  $\mathbf{w} \bullet \mathbf{x_i} + b \leq -1$ when $y_i = -1$

- H1 and H2 are the planes:

  H1: $\mathbf{w} \bullet \mathbf{x_i} + b = +1$

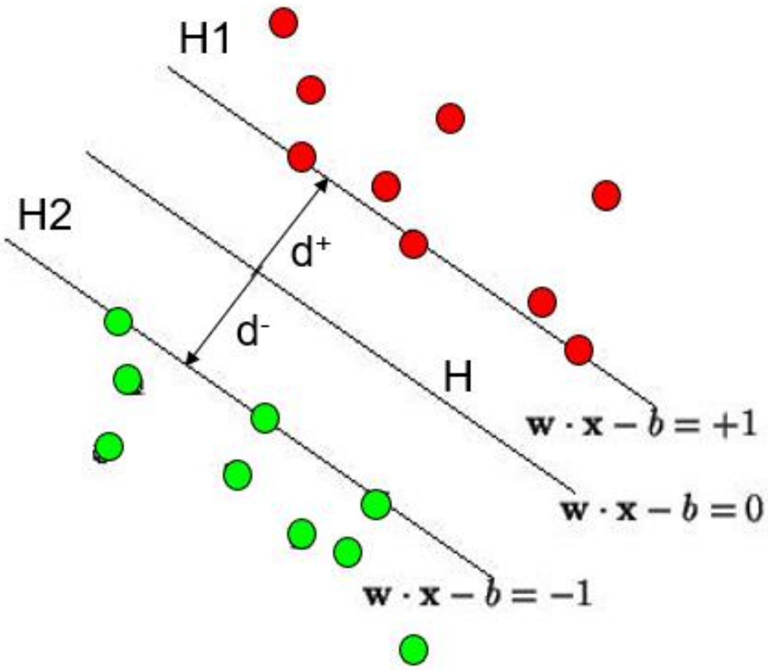
  H2: $\mathbf{w} \bullet \mathbf{x_i} + b = -1$

- The points on the planes H1 and H2 are the ***Support Vectors***.
- d+ = the shortest distance to the closest positive point, while d- = the shortest distance to the closest negative point.
- The ***margin*** of a separating hyperplane is $d^+ + d^-$



Linear SVM (Maximise the Margin)

- We want a classifier with as big a margin as possible.
- Recall the distance from a point(x_0,y_0) to a line of the form Ax+By+c = 0 is:
  $|Ax_0+By_0+c| / \mathrm{sqrt}(A^2+B^2)$
- The distance between H and H1 is: $|\mathbf{w}\bullet\mathbf{x}+b| / \mathrm{norm}(\mathbf{w}) = 1/\mathrm{norm}(\mathbf{w})$
- The distance between H1 and H2 is therefore $2/\mathrm{norm}(\mathbf{w})$
- In order to maximise the margin, we need to minimize norm(**w**).



Solve by a Constrained Optimisation Problem

- Minimise $\|\boldsymbol{w}\| = \langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle$ subject to $y_i(\langle \boldsymbol{x}_i \cdot \boldsymbol{w} \rangle + b) \geq 1$ for all i.
- Lagrangian method: max $\inf_{\boldsymbol{w}} L(\boldsymbol{w}, b, \alpha)$ where $L(\boldsymbol{w}, b, \alpha) = \frac{1}{2}\|\boldsymbol{w}\| - \sum_i \alpha_i[(y_i(\boldsymbol{x}_i \cdot \boldsymbol{w}) + b) - 1]$
- At the extremum, the partial derivative of L with respect to both $\boldsymbol{w}$ and b must be 0.
- Taking the derivatives, and setting them to 0, then substituting back into L and simplifying, yields:

Maximise $\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i \cdot \boldsymbol{x}_j \rangle$ subject to $\sum_i y_i \alpha_i = 0$ and $\alpha_i \geq 0$

- This is an instance of a positive, semi-definite programming problem which can be solved in $O(n \times \log n)$ time.

Naïve Bayes

- In 1763, Reverend Thomas Bayes, a mathematician and Presbyterian minister, posthumously published "An Essay towards solving a Problem in the Doctrine of Chances".

- This essay introduced the foundation of what we now call ***Bayes Theorem***, offering “a framework for updating beliefs based on new evidence”.

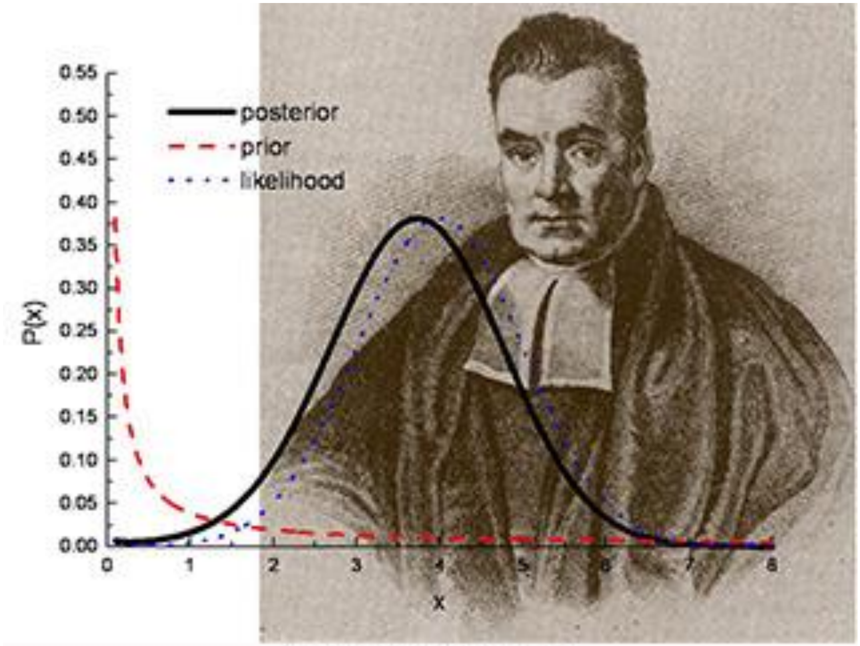


Image courtesy of LANL

The Bayes Classifier

- Problem statement:
  - Given features $X_1, X_2, \ldots, X_n$
  - Predict a label Y
- A good strategy is to predict:

$$\arg max_Y P(Y|X_1, X_2, \ldots, X_n)$$

- For example: *What is the probability that the image represents a dog given its pixels?*
- How do we compute that?

The Bayes Classifier (cont…)

- Use Bayes Theorem:

Likelihood Prior

Posterior

$$P(Y|X_1,\dots,X_n) = \frac{P(X_1,\dots,X_n|Y)P(Y)}{P(X_1,\dots,X_n)}$$

Normalisation Constant

- To classify if an image is a dog, we first compute the following two probabilities:

$$\boldsymbol{P(Y = Dog|X_1,\dots,X_n)} = \frac{P(X_1,\dots,X_n|Y = Dog)P(Y = Dog)}{P(X_1,\dots,X_n|Y = Dog)P(Y = Dog) + P(X_1,\dots,X_n|Y = \neg Dog)P(Y = \neg Dog)}$$

$$\boldsymbol{P(Y = \neg Dog|X_1,\dots,X_n)} = \frac{P(X_1,\dots,X_n|Y = \neg Dog)P(Y = \neg Dog)}{P(X_1,\dots,X_n|Y = Dog)P(Y = Dog) + P(X_1,\dots,X_n|Y = \neg Dog)P(Y = \neg Dog)}$$

- Classify the image is a dog if $P(Y = Dog|X_1,\dots,X_n) \geq P(Y = \neg Dog|X_1,\dots,X_n)$

The Bayes Classifier (cont…)

- For the Bayes Classifier, we need to learn two functions, the ***likelihood*** and the ***prior***.
- The problem with explicitly modelling $P(X_1, \ldots, X_n|Y)$ is that there are usually way too many parameters.
- We'll run out of space.
- We'll run out of time.
- And we'll need a lot of training data (which is usually not available).
- The solution lies in the Naïve Bayes Assumption.

The Naïve Bayes Model

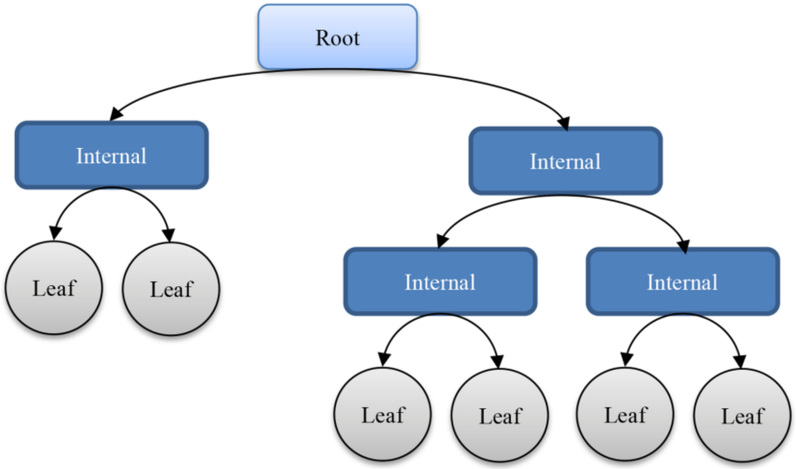
- The Naïve Bayes Assumption: Assume that all features $X_1, X_2, \ldots, X_n$ are ***independent*** given the class label Y.
- With this assumption, the following equation holds:

$$P(X_1, \ldots, X_n | Y) = \prod_{i=1}^{n} P(X_i | Y)$$

- Without the ***independent*** assumption, the number of parameters for modelling $P(X_1, \ldots, X_n | Y)$ is $\mathbf{2(2^{n} - 1)}$.
- With the ***independent*** assumption, the number of parameters is reduced to $\mathbf{2n}$.

Decision Trees

- Tree-like structures where each node represents a feature (question) and branches represent possible answers.
- Each branch leads to a new node, asking another question or reaching a leaf node containing the final prediction.
- They are easy to understand and visualise.
- Used for both classification (predicting categories) and regression (predicting continuous values).



Decision Trees (cont…)

- Decision trees are nonparametric, supervised learning methods.
- Represented as an upside-down tree, where derived rules are used to guide decisions.
- Nodes represent query variables, edges represent their values, branches represent if-then rules.
- Deeper trees yield more complex rules and models.
- ***Goal***: Training algorithm creates decision tree based on dataset, produces rules for prediction.

Strengths of Decision Trees

- ***Interpretability***: We can easily understand the thought process behind each prediction due to the tree structure.
- ***Ability to handle mixed data types***: Decision trees can handle both numerical and categorical features without complex preprocessing.
- ***Robustness to missing data***: They can impute missing values by following the most common branch for that feature.

Weaknesses of Decision Trees

- ***Overfitting***: Decision trees can become too specific to the training data, leading to poor performance on unseen data.
- ***Instability***: Small changes in the data can lead to significant changes in the tree structure and predictions.
- ***High variance***: Individual trees can be sensitive to changes in the training data, leading to inconsistent predictions.

Random Forest

- Random forests are ensembles of decision trees, combining multiple trees for improved accuracy and stability.
- Overfitting: Random forest reduces overfitting and improves performance by combining output of individual decision trees.
- Each tree in the forest is trained on a different bootstrap sample of the data and uses a subset of features randomly selected without replacement.
- The final prediction is based on the majority vote (for classification) or the average (for regression) of the individual tree predictions.

Strengths of Random Forest

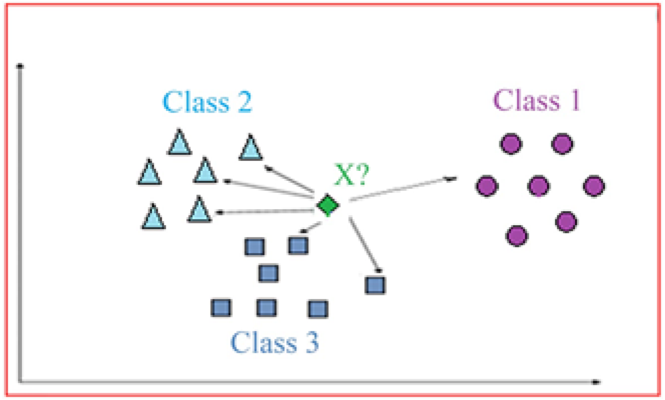
- ***Improved accuracy***: Random forests often outperform individual decision trees due to reduced overfitting and variance.
- ***Robustness to noise and outliers***: By averaging predictions from multiple trees, random forests are less sensitive to outliers and noise in the data.
- ***Ability to handle high-dimensional data***: They can effectively deal with datasets with many features without significant performance degradation.

K-Nearest Neighbors (K-NN)

- K-NN Algorithm: Utilises K nearest points to determine classification or regression.
- K-nearest neighbors are different from K-means, which is for clustering.
- K-NN classifies data points based on their "neighborhood" in the feature space.
- Imagine data points as colored dots on a map. Their colors represent their class (e.g., red for apples, green for oranges).
- When encountering a new data point, K-NN finds its K nearest neighbors (say, 3 closest dots).
- The new point's class is assigned based on the majority vote of its neighbors (e.g., if 2 neighbors are red, the new one is likely red too).

Choosing the Right K: A Balancing Act

- K, the number of neighbors, significantly impacts K-NN's performance.
- K too low (e.g., K=1) can be sensitive to noise and outliers, leading to overfitting.
- K too high (e.g., K=20) can smooth out details, causing underfitting.
- Finding the optimal K often involves trial and error or techniques like cross-validation.



Strengths and Applications of K-NN

- Strengths:
  - Simple to understand and implement.
  - Effective for small datasets and high-dimensional data.
  - Can handle mixed data types (numerical and categorical).
- Applications:
  - Image classification (recognizing handwritten digits, categorizing photos).
  - Customer segmentation (grouping customers based on purchase history).
  - Fraud detection (identifying unusual transactions).
  - Spam filtering (classifying emails as spam or not spam).

THANK YOU

TIME FOR DISCUSSION & QUESTIONS