Supervised and Unsupervised Learning

Machine Learning Course, Day 2

Plan - Day 2

1. Supervised Learning
    a. Linear Regression
    b. Logistic Regression
    c. Tree-Based Methods
        i. Random Forest
        ii. XGBoost
2. Unsupervised Learning
    a. K-means
    b. DBSCAN
    c. Hierarchical Clustering

→ Hands-on: Training and Testing Supervised and Unsupervised Models

→ Discussion: Challenges in Machine Learning

About Us

Barbara Füzi
Postdoc

Drug Discovery
Toxicology
Synthetic Data

barbara.fuzi@bsc.es

Miquel Mayor
Research Engineer

Factor Analysis
Multilayer Network

miquel.mayor@bsc.es

Guillermo Prol
PhD student

Cancer
Synthetic Data

guillermo.prolcastelo@bsc.es

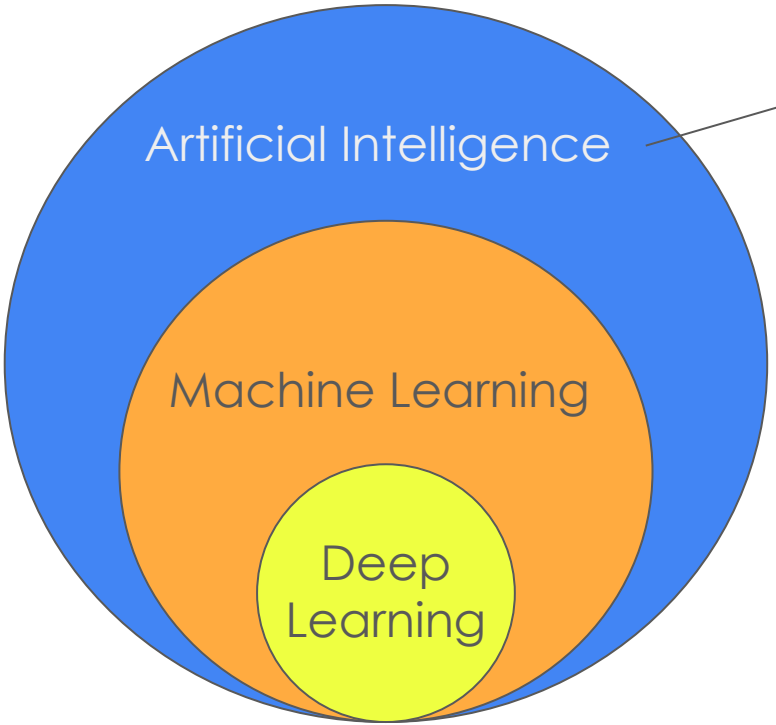
Before we begin...

- Open you ML Course Day 2 notebook
- Install the ucimlrepo package

  (Breast Cancer Wisconsin dataset)
- Import required libraries

  (Scikit-kit, Numpy, Panda, Matplotlib, Seaborn)
- Load the dataset

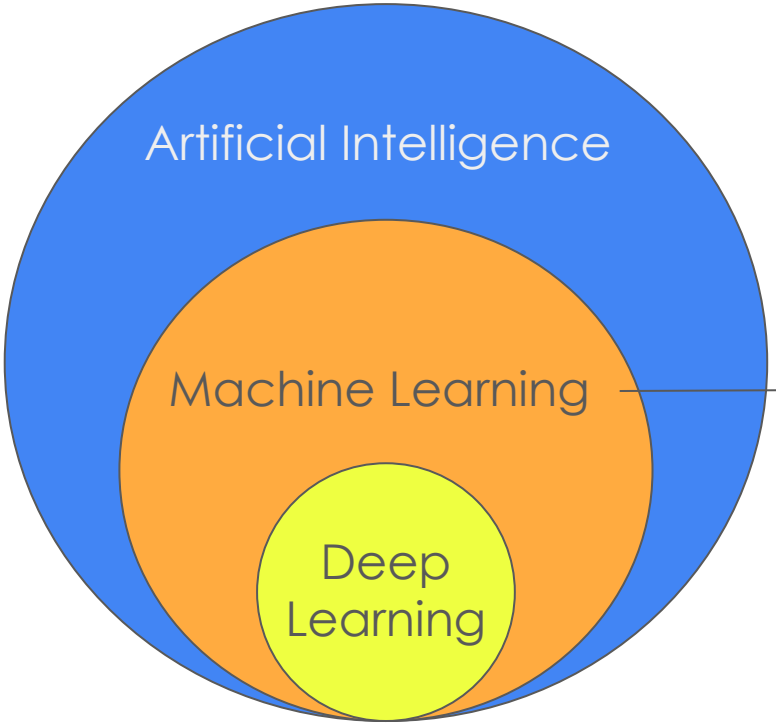
What is AI?



Artificial Intelligence (AI): attempt to mimic human problem-solving and decision-making abilities

Alan Turing: "Can machines think?"
→ Turing test (1950)

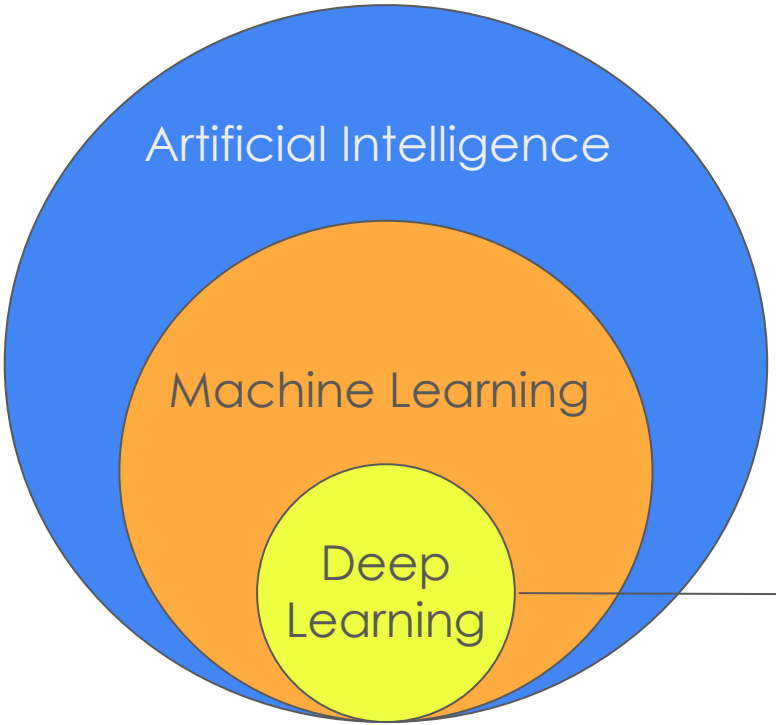
What is AI?

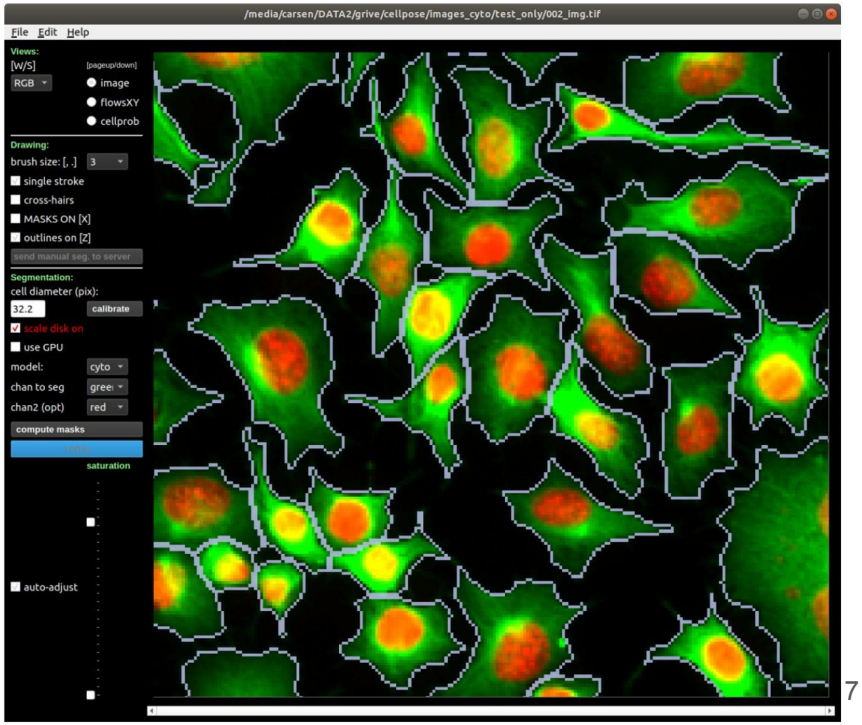


Machine Learning (ML): AI that gradually improves its accuracy—i.e., the AI that learns

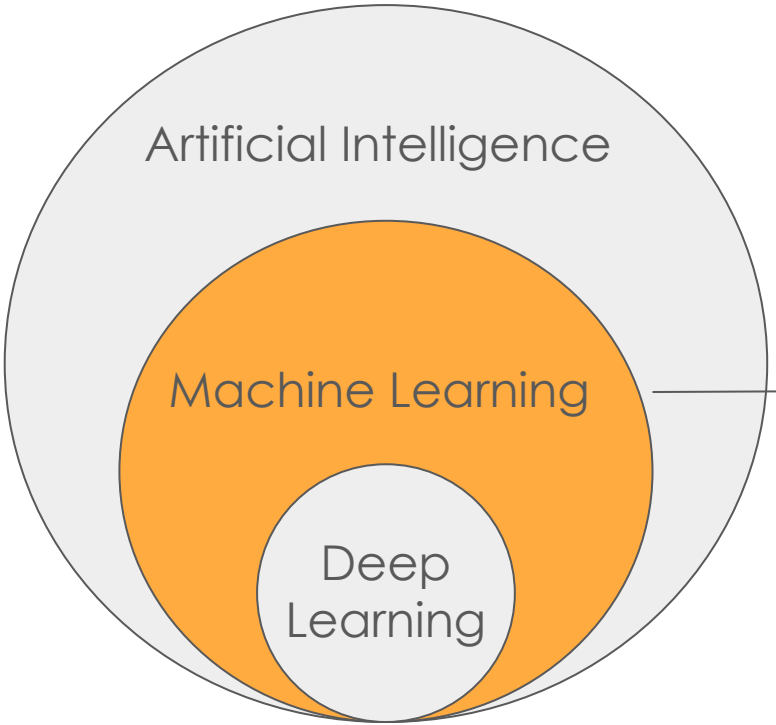
What is AI?

Deep Learning (DL): ML that uses multi-layered (deep) neural networks to learn





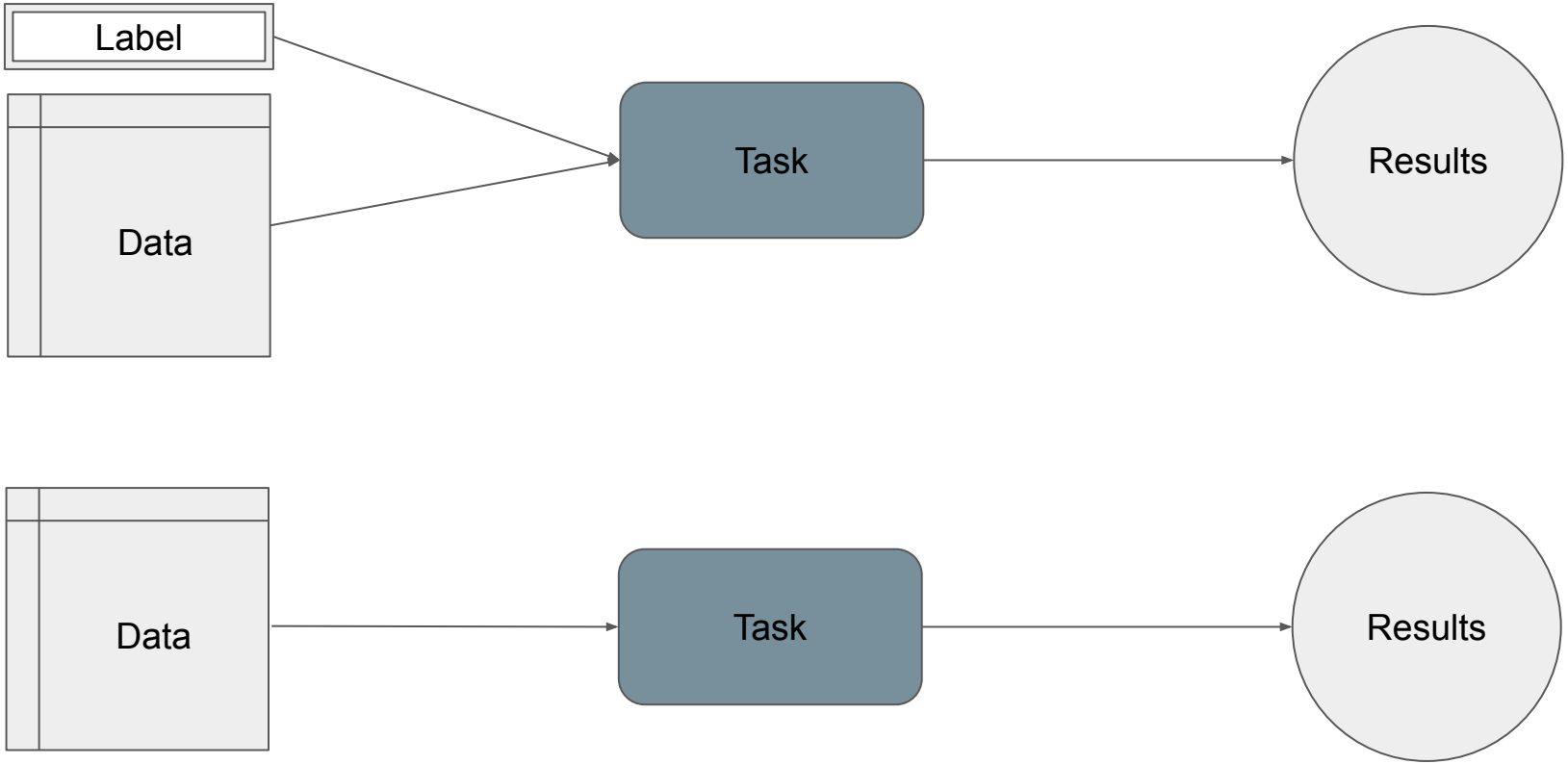
What is this class about?



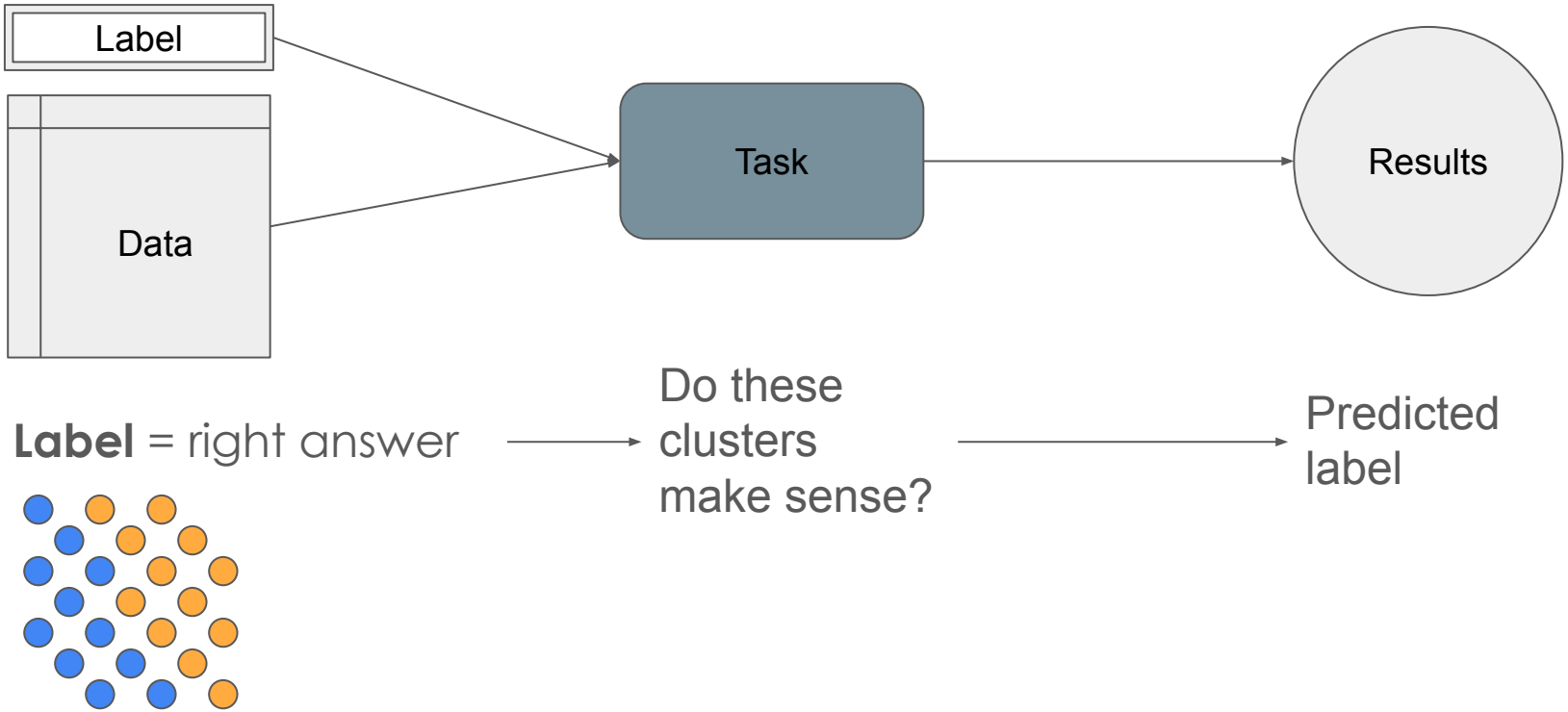
Machine Learning (ML): AI that gradually improves its accuracy—i.e., the AI that *learns*

- Supervised
- Unsupervised

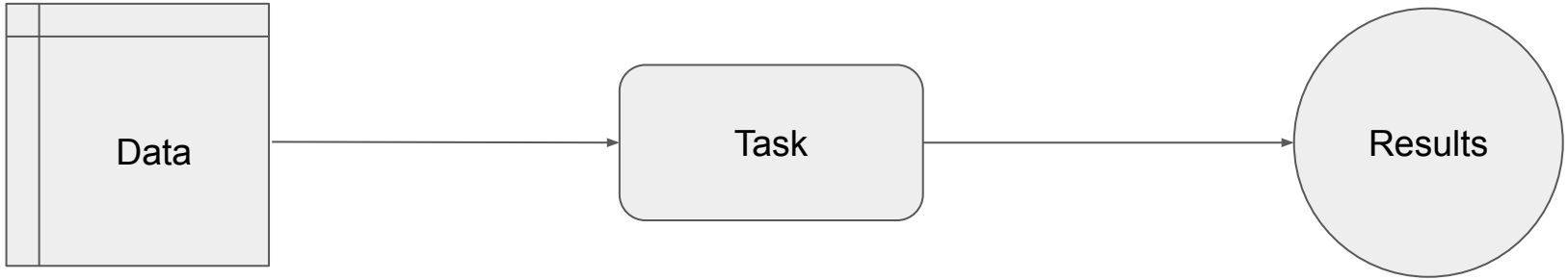
Supervised vs Unsupervised Learning (I)

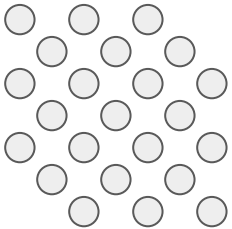


Supervised vs Unsupervised Learning (II)



Supervised vs **Unsupervised** Learning (III)





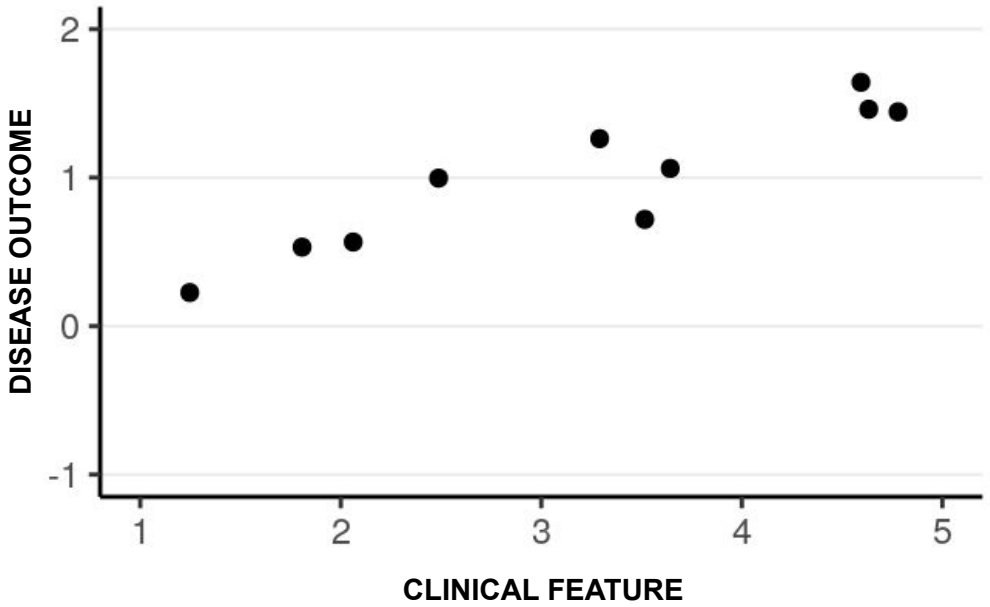
How many clusters are there?

Optimal clusters

Supervised Learning

Linear Regression

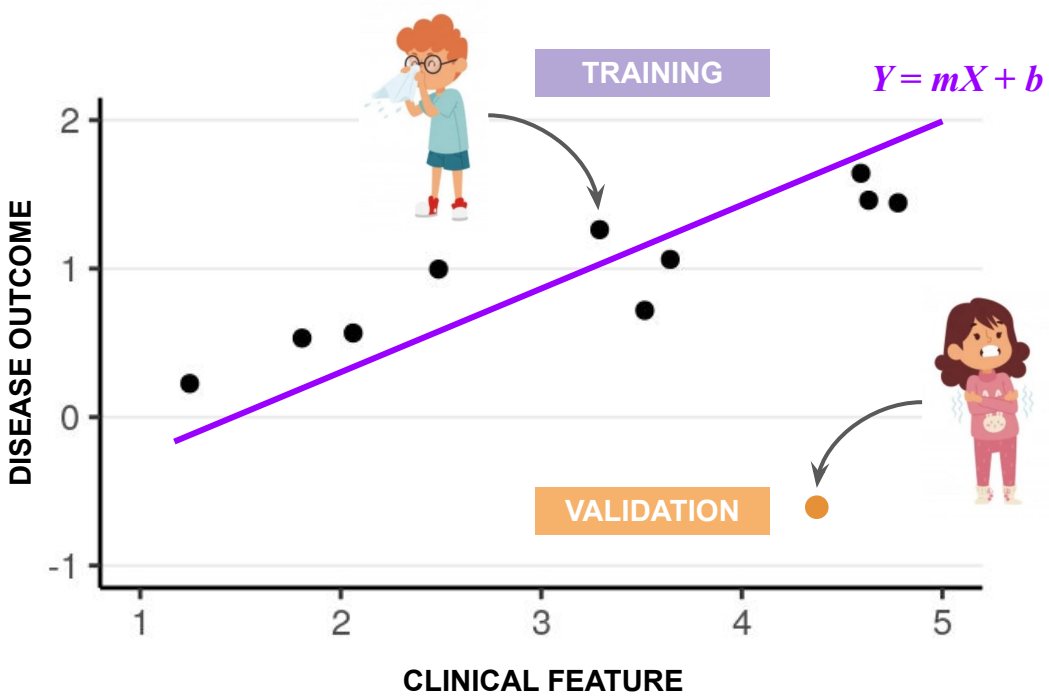
Linear Regression

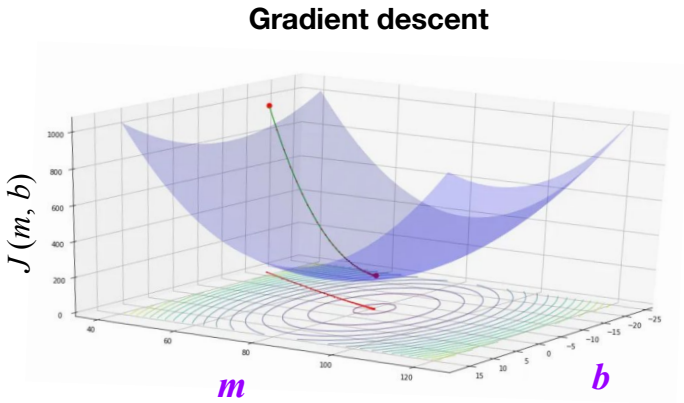


$y(x) = mx + b$,

- x: independent variable
- $y(x)$: dependent variable
- m: slope
- b: intersect

Gradient Descent





$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

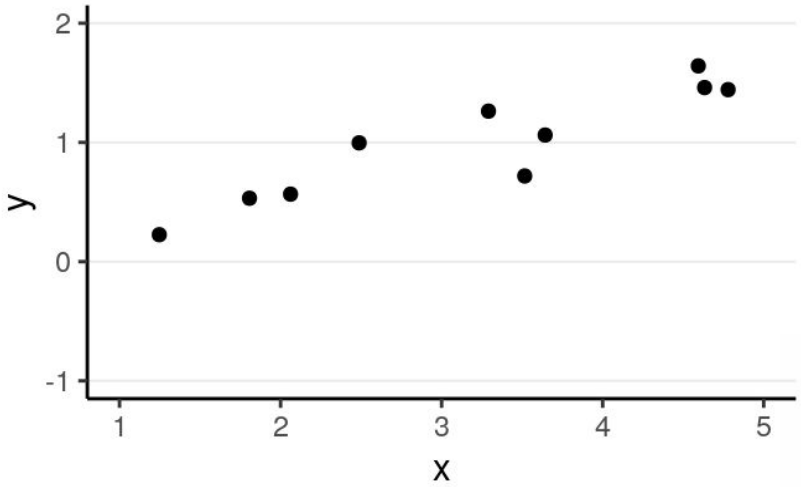
MSE = mean squared error

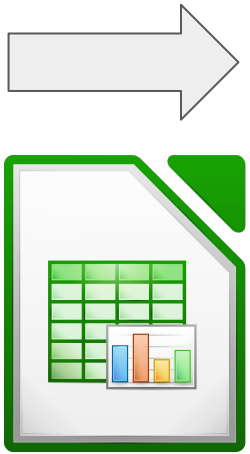
n = number of data points

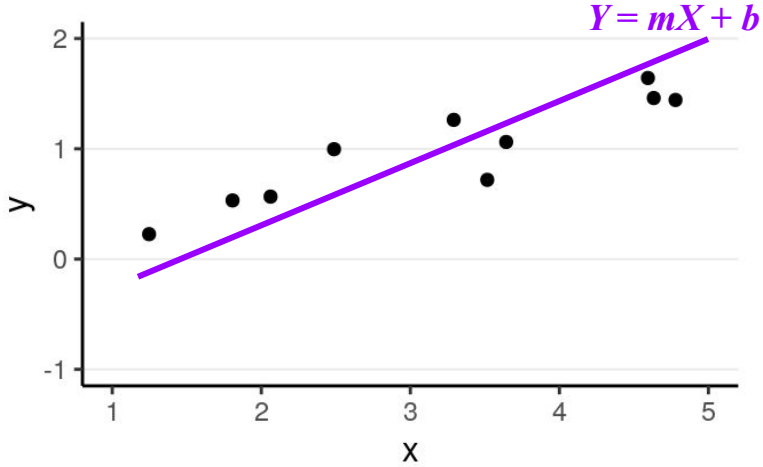
Y_i = observed values

$\hat{Y}_i$ = predicted values

Linear Regression on Spreadsheet-Software







Black-box model!

ML scheme of thought

1. Your problem implies an **estimation** → *mx* + *b*
2. **Minimize** the estimation **error**/loss → observed — predicted
3. Quality **control** → **R^2**

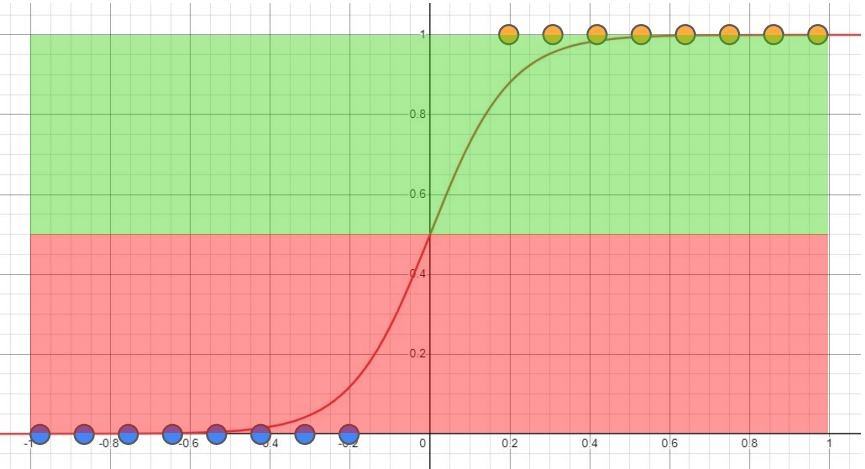
Supervised Learning

Logistic Regression

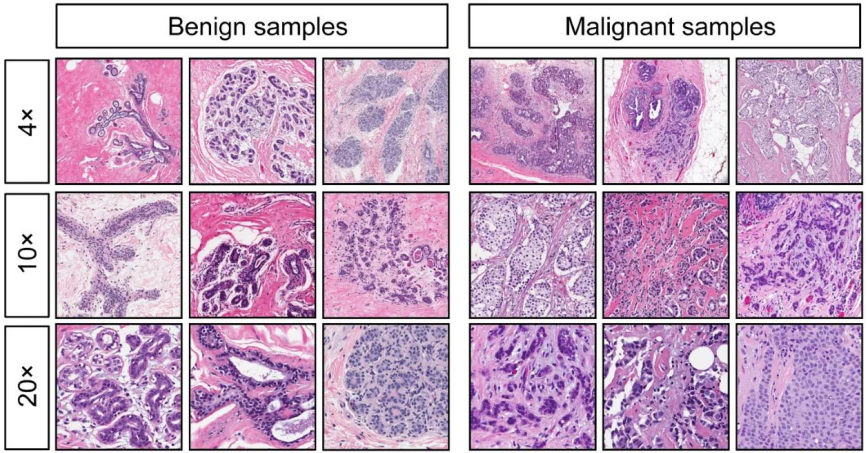
Logistic Regression

Breast Cancer Diagnosis

$$p(x) = \frac{1}{1 + e^{-(ax+b)}}$$



More complicated formula!
→ Need for Machine Learning



→ benign
→ malignant

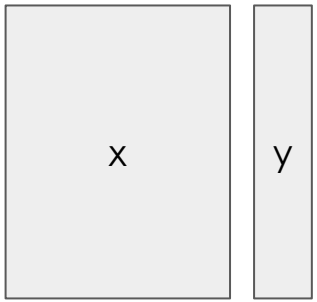
Training and Testing Sets (I)

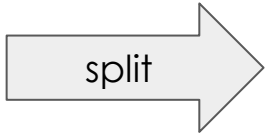
Training
Study for exam with practice questions

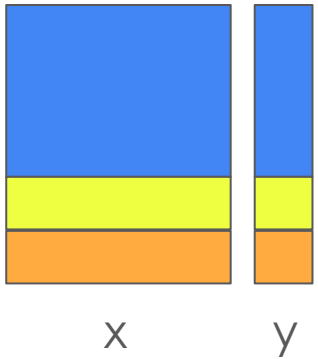
Objective: learn

Test
Unseen questions

Objective: show you have learned







Train (70-80%)

(Validation (~10%))

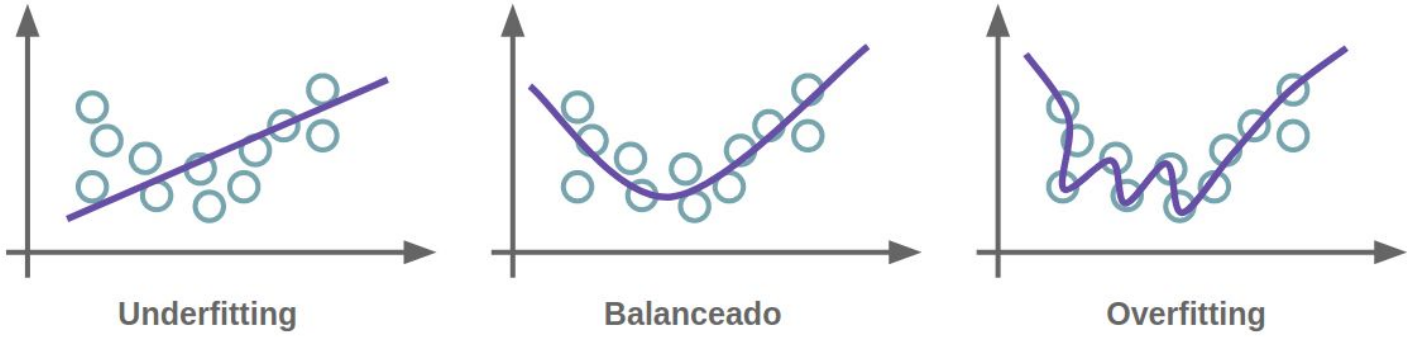
Test (20-30%)

Validation for hyperparameter tuning, e.g., optimal learning rate

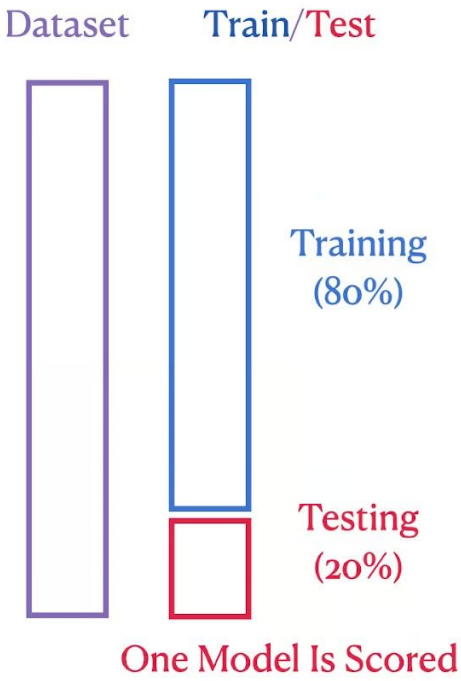
Training and Testing Sets (II)

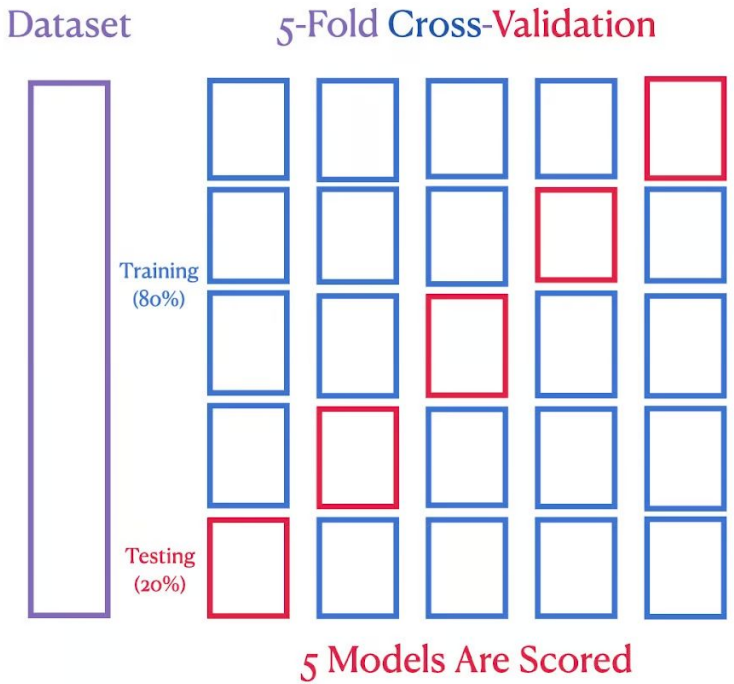
Possible outcomes:

- Did not study enough → underfitting
- Realized the patterns in the materials → learning
- Memorize a lot without understanding → **overfitting**



Cross Validation (CV)





Cost Functions *C(x)*

Patient	Real Value	Prediction
Low	0	0.5
Medium	1	0.5
High	0	0.5

The difference between our prediction and the real value is the **loss**

- Mean Squared Error (MSE)
  - "How far off are my predictions?" (for numbers)

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

- Binary Cross Entropy (BCE):
  - "How well do my probabilities match reality?" (for yes/no).

$$\mathrm{BCE} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

y_i → True label

$\hat{y}_i$ → Prediction

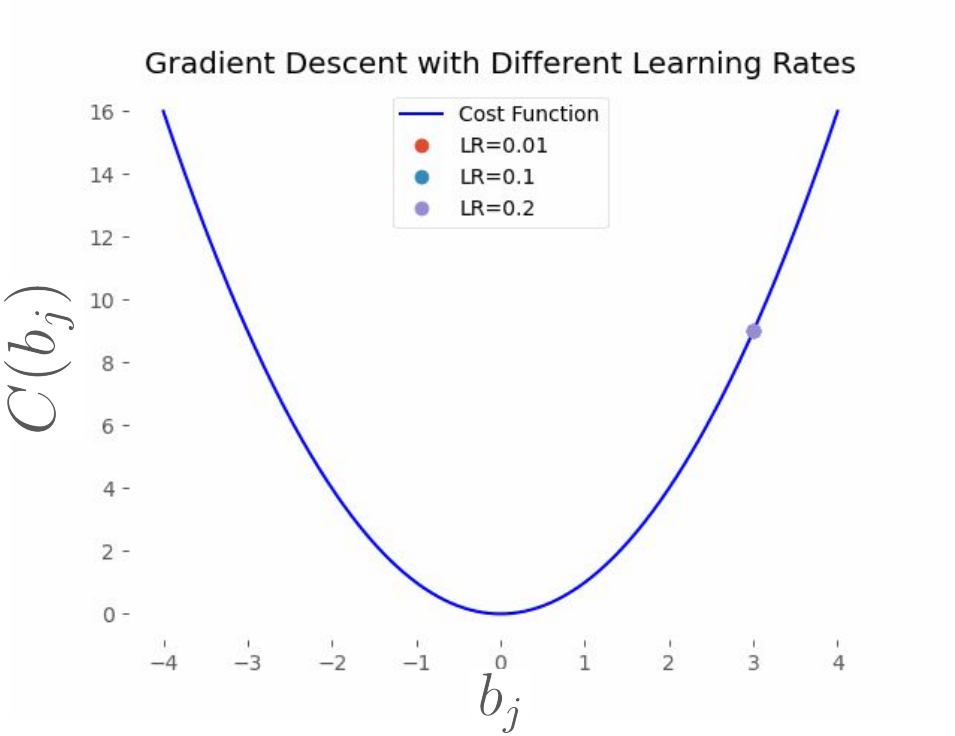
Backpropagation and Gradient Descent

1. Initialize w_i and b_j
2. Minimize loss function $C(x)$
3. **Gradient descent**:

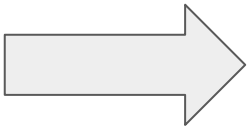
$$b_j = b_j - \alpha \frac{\partial \text{Loss}}{\partial b_j}$$

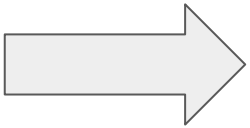
where α is the learning rate

Repeat a number of times when loss is below a threshold



Implementation





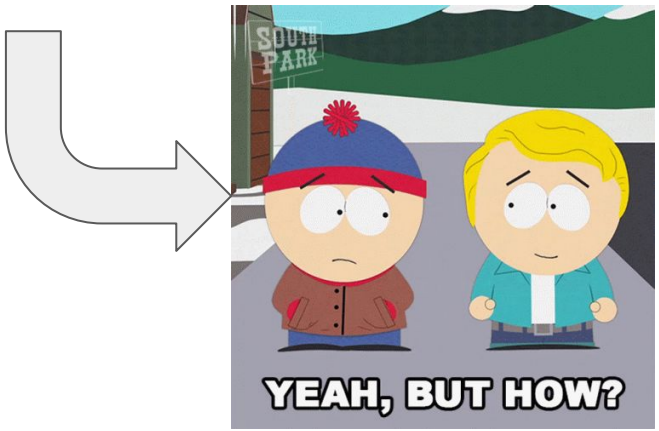
caret

ML scheme of thought

1. Your problem implies an **estimation** → $$p_i = \frac{1}{1 + e^{-(b_0 + b_1 x_{i1} + \cdots + b_n x_{in})}}$$

2. **Minimize** the estimation **error**/loss → Train Data + Gradient Descent

3. Quality **control** → Test Data



Assess predictions

- Precision
- Accuracy

Accurate and precise

Inaccurate and precise

Accurate and imprecise

Inaccurate and imprecise

Metrics

Precision = $\frac{TP}{TP+FP}$

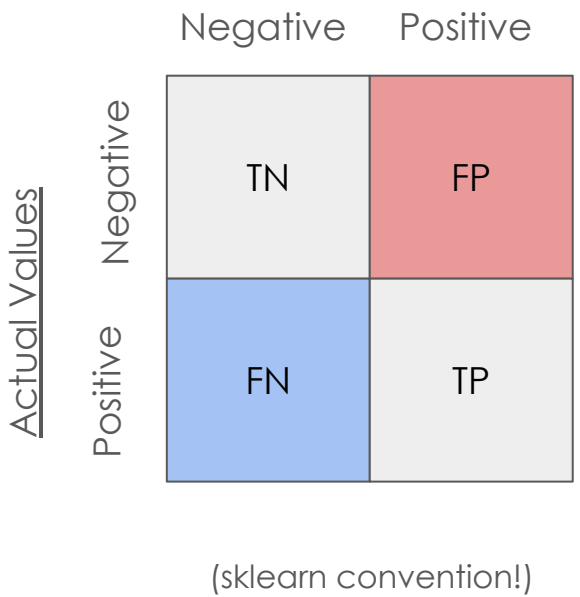
Recall or sensitivity = $\frac{TP}{TP+FN}$

Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

F1-score = $2\ \frac{recall \times precision}{recall+precision}$

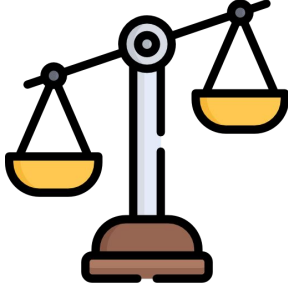
= $\frac{2TP}{2TP+FP+FN}$

Confusion Matrix



(sklearn convention!)

Unbalanced Data



One class is a lot more frequent than the other

- Example: 99% "healthy" vs. 1% "diseased" patients
- Risk: Models ignore rare classes ("Why bother with the “diseased” case if I can just guess ‘healthy’?").

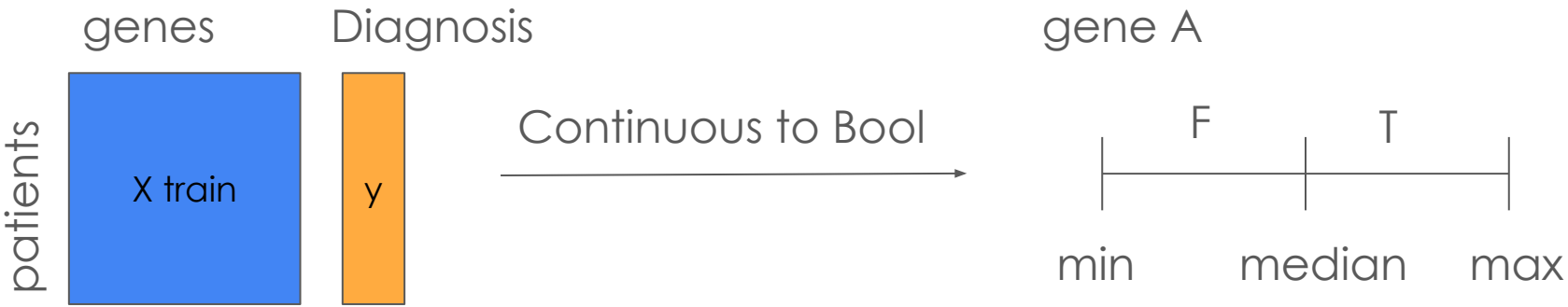
Fix:

Class weights: Tell the model, "Pay extra attention to rare classes!"

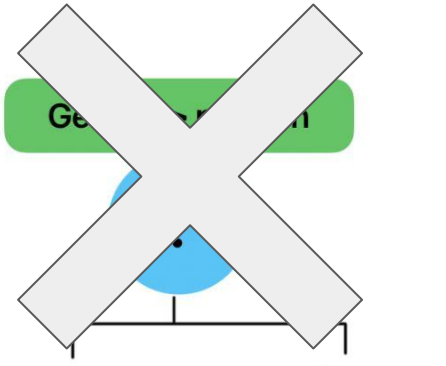
Supervised Learning

Random Forest

Decision Trees



“Planting” the Trees

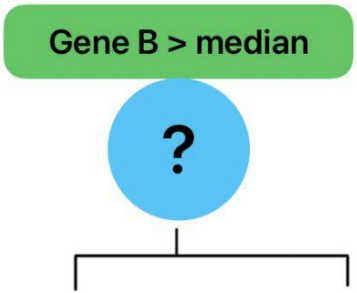


True

D	25
H	25

False

D	25
H	25

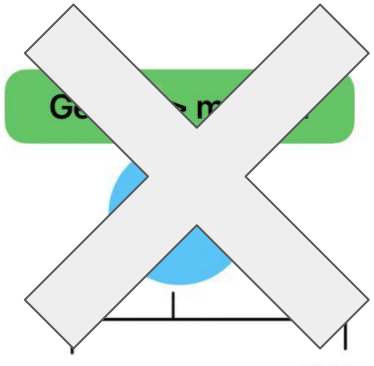


True

D	49
H	1

False

D	25
H	25



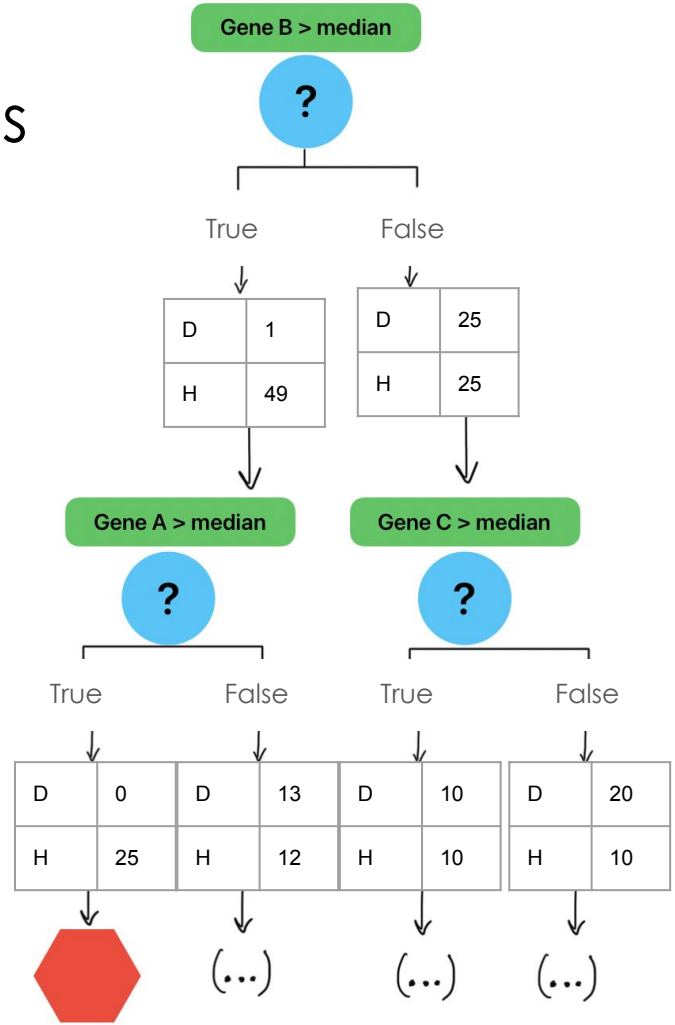
True

D	33
H	17

False

D	10
H	40

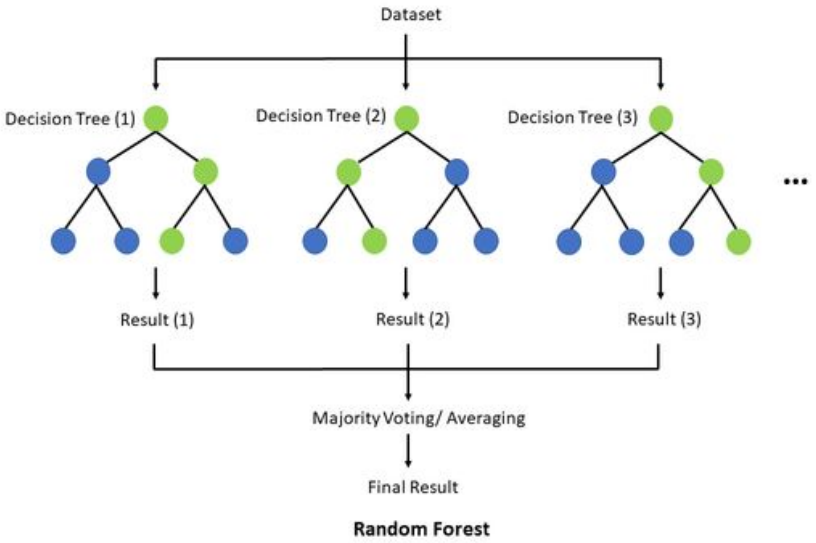
Growing the Trees



Max depth!

Random Forest

1. Build Many Decision Trees
    a. Bootstrapping → data subsample
    b. Random Seeds
2. Trees vote
    a. **Prediction** → Most popular class
3. Reduce overfitting and time
    a. "Crowd wisdom"



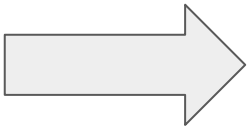
ML scheme of thought

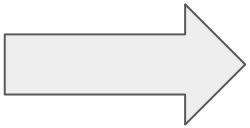
1. Your problem implies an **estimation** → **Forest of Trees**

2. **Minimize** the estimation **error**/loss → Grow Trees on Train Data (no gradient boosting!)

3. Quality **control** → Test Data

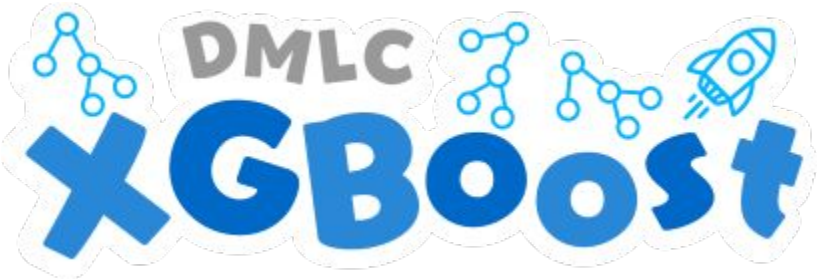
Implementation





caret

Supervised Learning



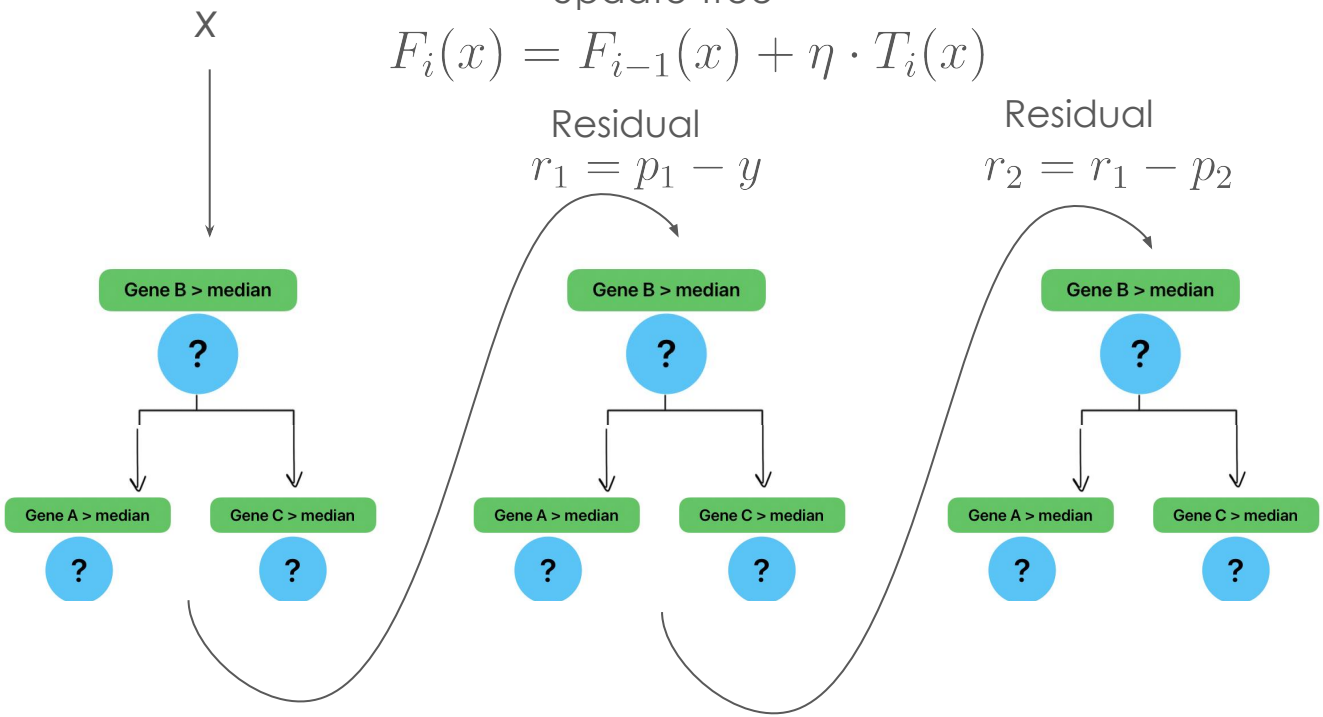
Boosted Trees

- Decision Trees → Decision Stump!
- + Gradient Descent

η learning rate

Update Tree

$$F_i(x) = F_{i-1}(x) + \eta \cdot T_i(x)$$



Boosted Ensemble

$$F_n(x) = F_0 + \eta \sum_{i=1}^{n} T_i(x)$$

F_0: initial predictions

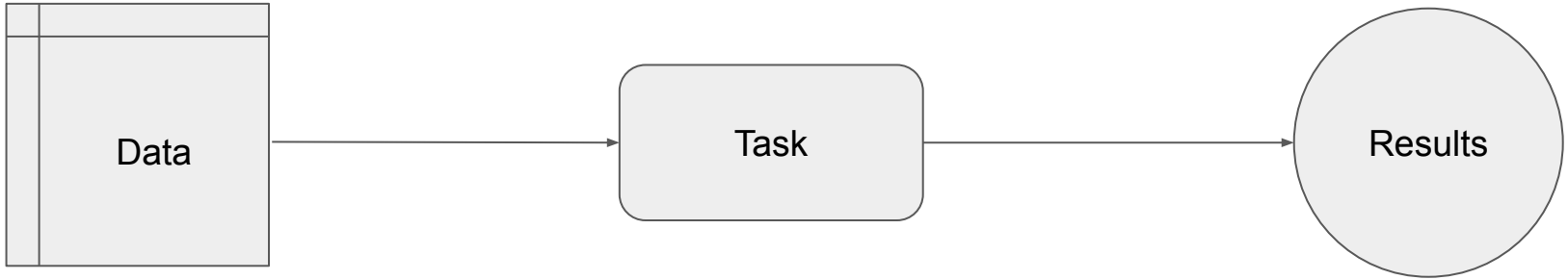
$T_1(x)$: Tree 1's predictions

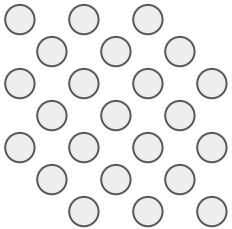
ML scheme of thought

1. Your problem implies an **estimation** → $F_n(x) = F_0 + \eta \sum_{i=1}^{n} T_i(x)$
2. **Minimize** the estimation **error**/loss → Train Data + Gradient Descent
3. Quality **control** → Test Data

Unsupervised Learning

Unsupervised Learning

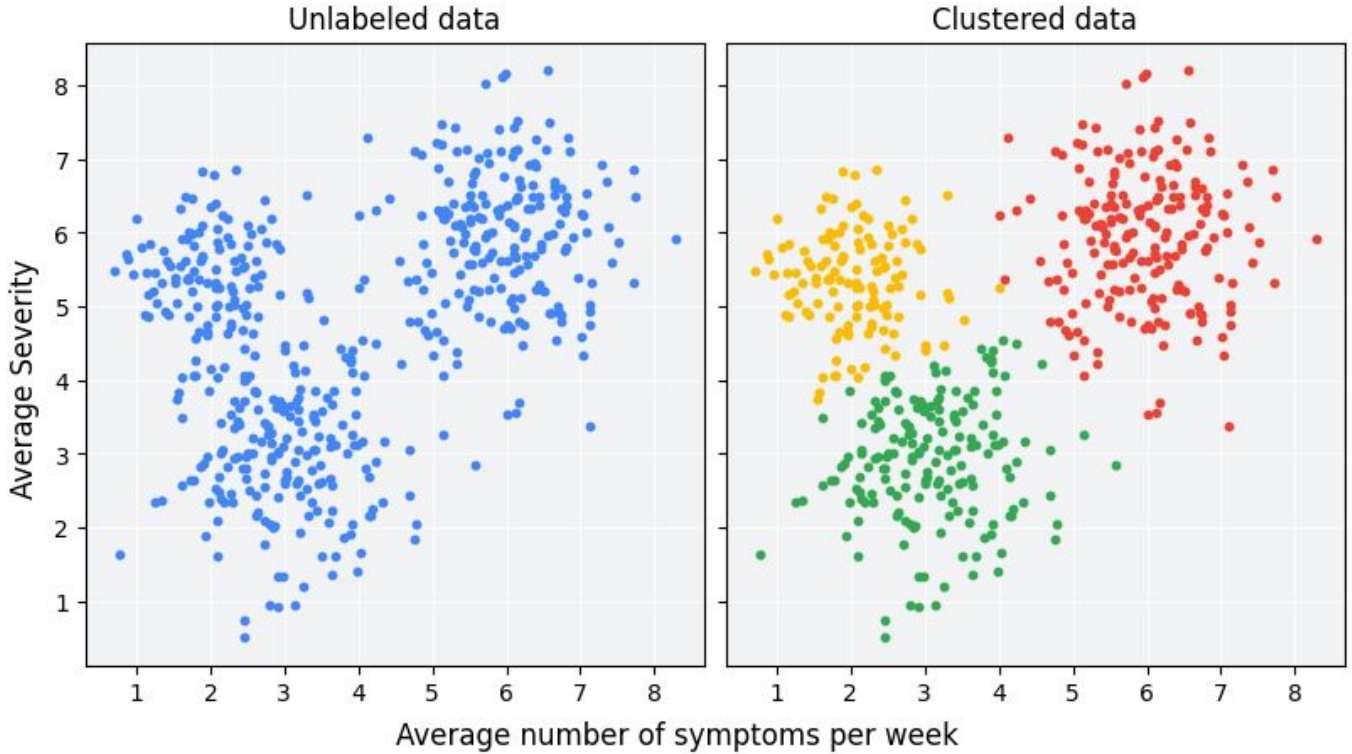




How many clusters are there?

Optimal clusters

Clustering

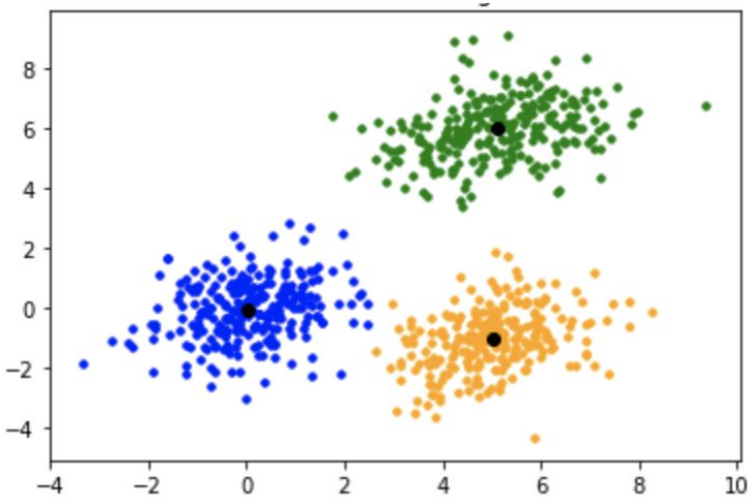

Unlabeled data
Clustered data
Average Severity
Average number of symptoms per week
1
2
3
4
5
6
7
8

K-Means Clustering

K-Means clustering is an unsupervised machine learning algorithm used for partitioning a dataset into **K** clusters.

Centroid:

- A centroid is the **center point** of a cluster in K-Means clustering.
- It represents the **average position** of all data points in a cluster.
- The centroid is **not necessarily** an actual data point
- It **minimizes** the sum of squared distances to all points in its cluster.
- It shifts position as the clustering algorithm iterates.



K-Means Steps

Step 1: Calculate the number of K (Clusters).

Step 2: Randomly select K data points as cluster center.

Step 3: Using the Euclidean distance formula measure the distance between each data point and each cluster center.

$$\text{Distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Step 4: Assign each data point to that cluster whose center is nearest to that data point.

Step 5: Re-compute the center of newly formed clusters. The center of a cluster is computed by taking the mean of all the data points contained in that cluster.

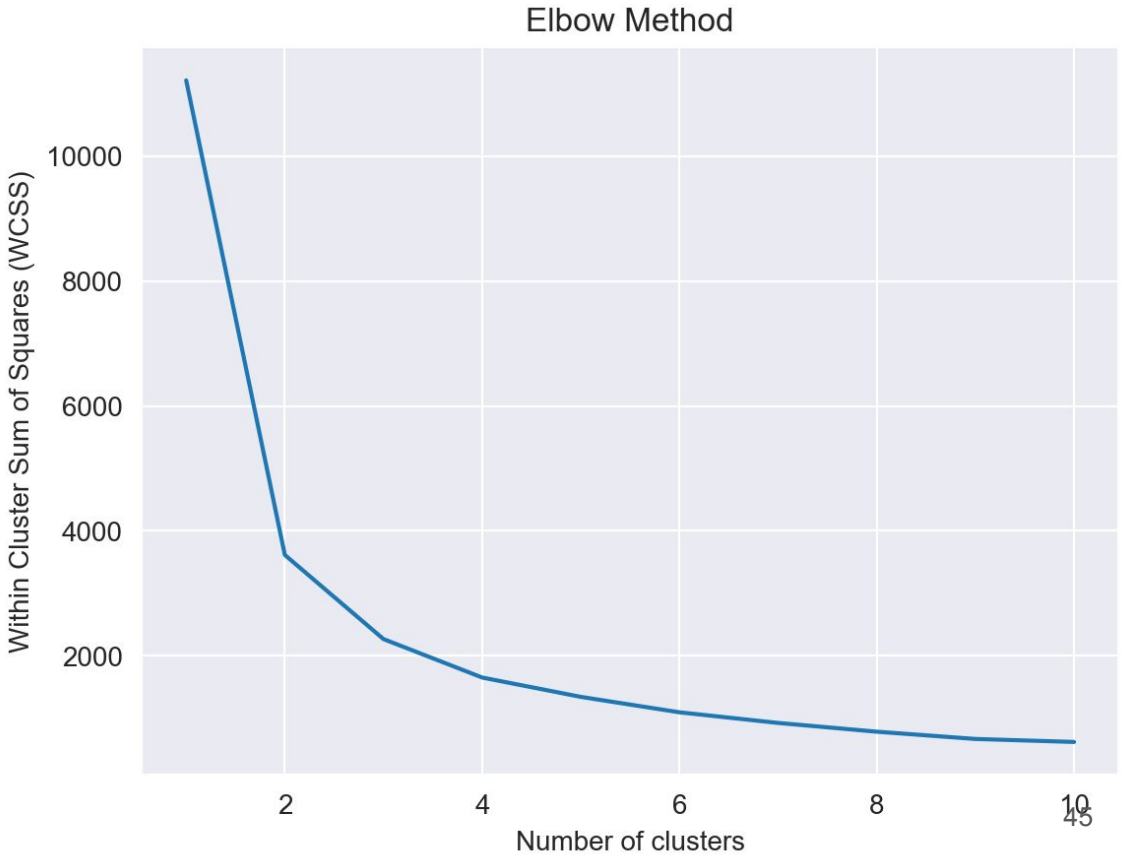
Minimizing Estimation Error

- K-Means minimizes the **sum of squared errors (SSE)**, also called the **inertia**.
- Each point is assigned to the nearest centroid, and the centroid is updated iteratively to reduce the total variance.
- **Quality Control**: If K is chosen poorly or centroids are initialized badly, the clusters may not reflect the true structure.

  Techniques like the **Elbow Method** or **Silhouette Score** help control quality.

How many K´s?

- Try different values of **K**
- Measure how well the data is grouped using **inertia** (how close points are to their cluster center).
- Plot **K vs. inertia** – it will look like a downward curve.
- Find the **"elbow"**—the point where the curve bends and adding more clusters **doesn't improve much**.



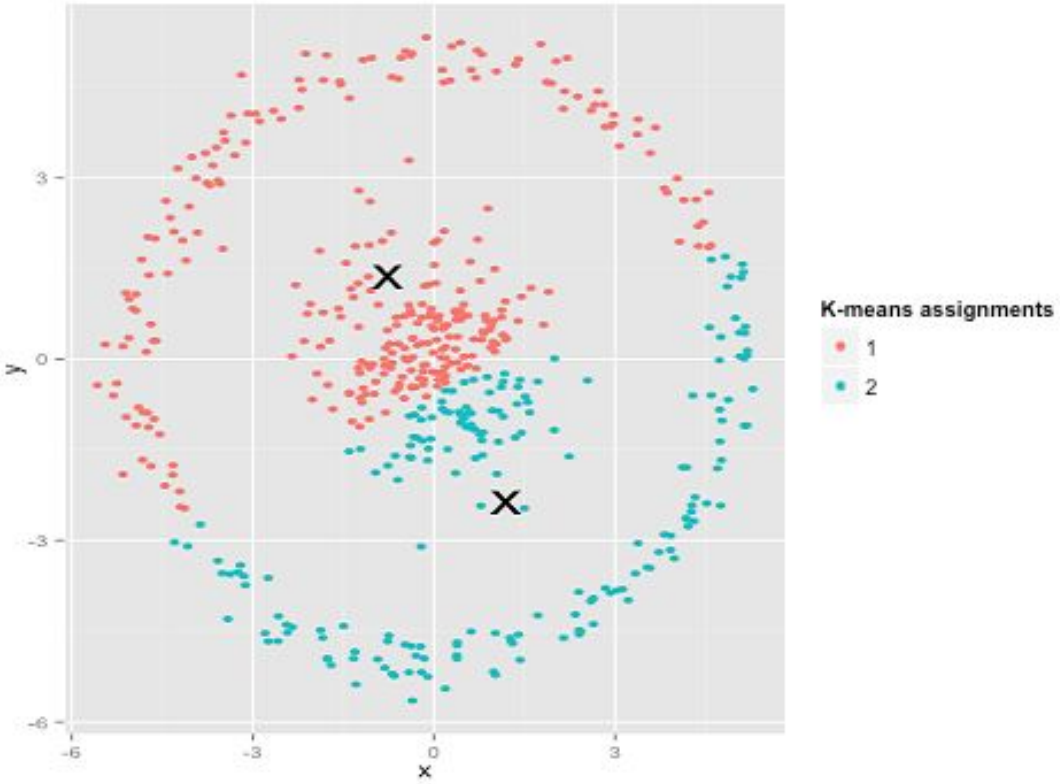
Silhouette scores

Silhouette scores are a method used to evaluate the quality of clustering, particularly for algorithms like k-means.

Measures how similar each point is to its own cluster compared to other clusters

- A score close to **+1** indicates that the data point is well-clustered (i.e., it is closer to points in its own cluster than to points in other clusters).
- A score close to **0** suggests that the data point is on or near the boundary between two clusters.
- A score close to **-1** suggests that the data point is likely in the wrong cluster.

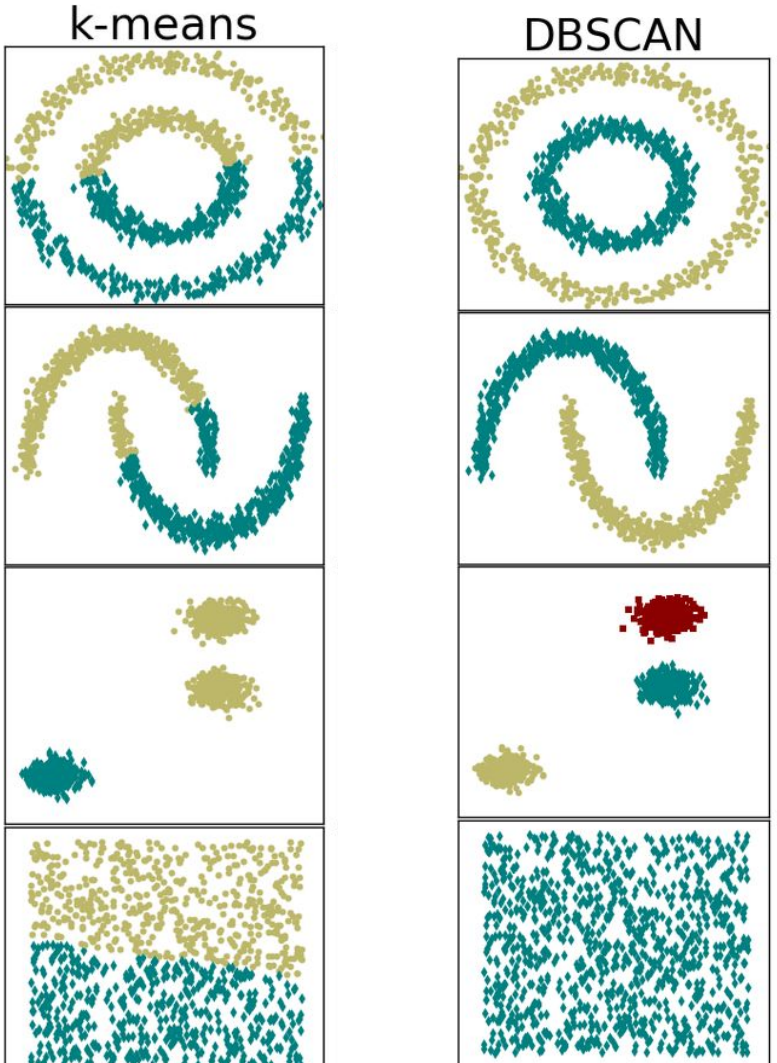
Where is the limitation for K-Means?



DBSCAN

Density-Based Spatial Clustering of Applications with Noise

- clustering algorithm that groups together closely packed points while marking outliers as noise



How does it work?

- **Core Points:** Points that have at least a minimum number of neighboring points within a specified radius (epsilon).
- **Border Points:** Points that have fewer than the required number of neighbors but are within the radius of a core point.
- **Noise (Outliers):** Points that do not belong to any cluster because they are not within the radius of any core point and do not have enough neighbors.
- **Epsilon (ε):** The maximum distance between two points for them to be considered neighbors.
- **MinPts:** The minimum number of points required to form a dense region (i.e., a cluster).

$$d((x_1, y_1), (x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

DBSCAN steps

Identify Core Points: DBSCAN starts by randomly selecting a point and checking if it has enough neighbors (≥ MinPts) within the epsilon radius. If so, it's labeled as a core point.

Expand the Cluster: All points within the epsilon radius of a core point are added to the cluster. Then, the algorithm checks each new point to see if it's also a core point and can expand the cluster further.

Assign Border Points: Points that are within the epsilon radius of a core point but don't meet the MinPts threshold are assigned to the cluster but aren't considered core points themselves.

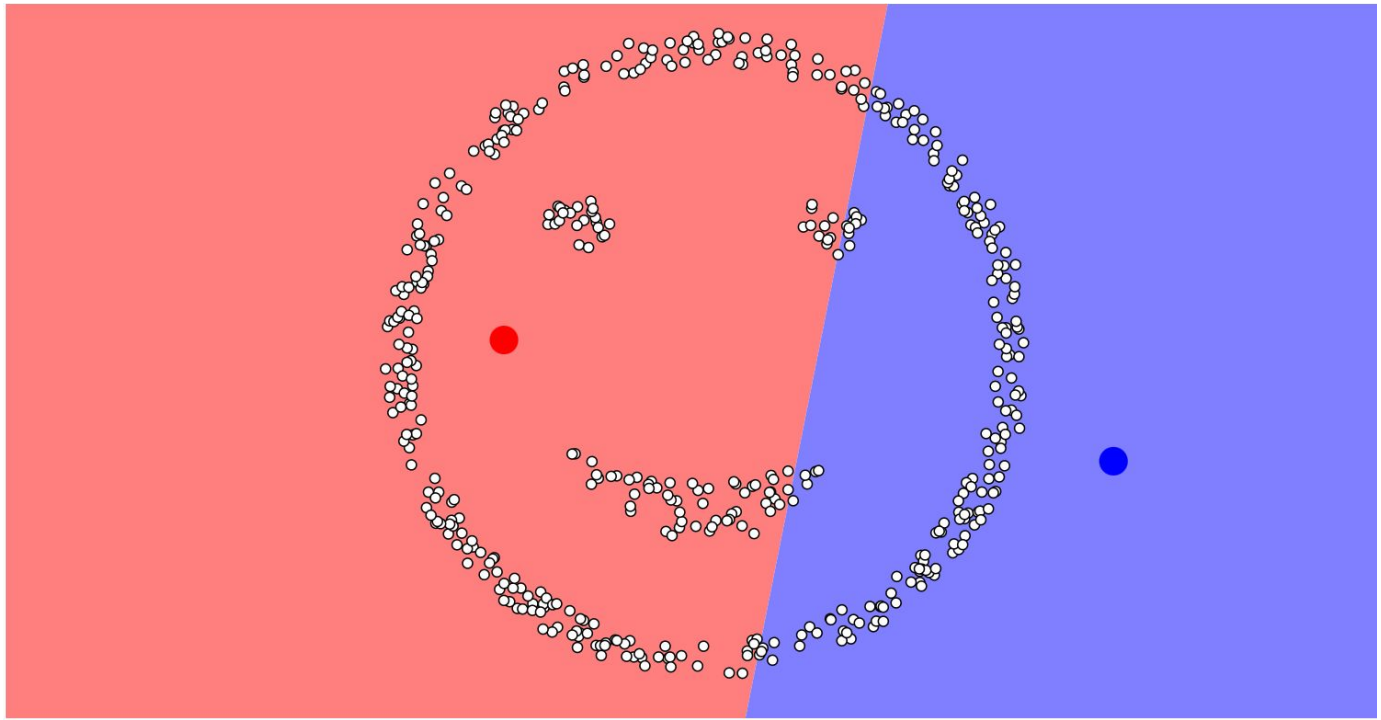
Label Noise: Points that do not meet any of the above criteria are labeled as noise (outliers)

Minimizing Noise and False Clustering

- DBSCAN estimates cluster density based on **ε (epsilon)** and **minPts**.
- It minimizes **false cluster assignments** by distinguishing **core points, border points, and noise**.
- **Quality Control**: If **ε** is too small, clusters break apart; if it's too large, clusters merge incorrectly. Tuning these parameters ensures robust clustering.

Playing around

https://www.naftaliharris.com/blog/visualizing-k-means-clustering/



How good is it?

Strengths:

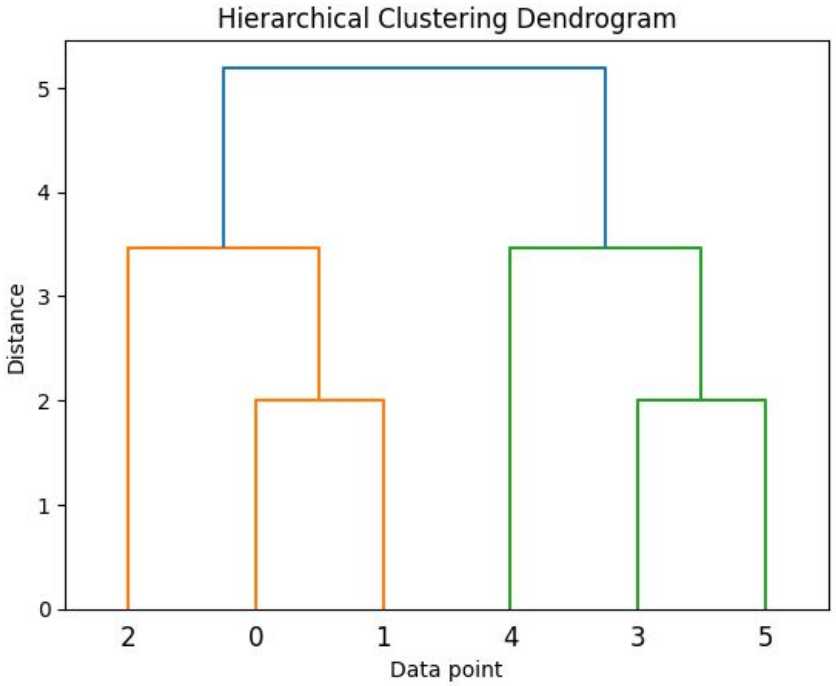
- Can find arbitrarily shaped clusters.
- Does not require the number of clusters to be specified in advance.
- Handles noise well.

Weaknesses:

- Performance can degrade in high-dimensional spaces (curse of dimensionality).
- Sensitive to the choice of epsilon and MinPts.
- Struggles with clusters of varying densities.

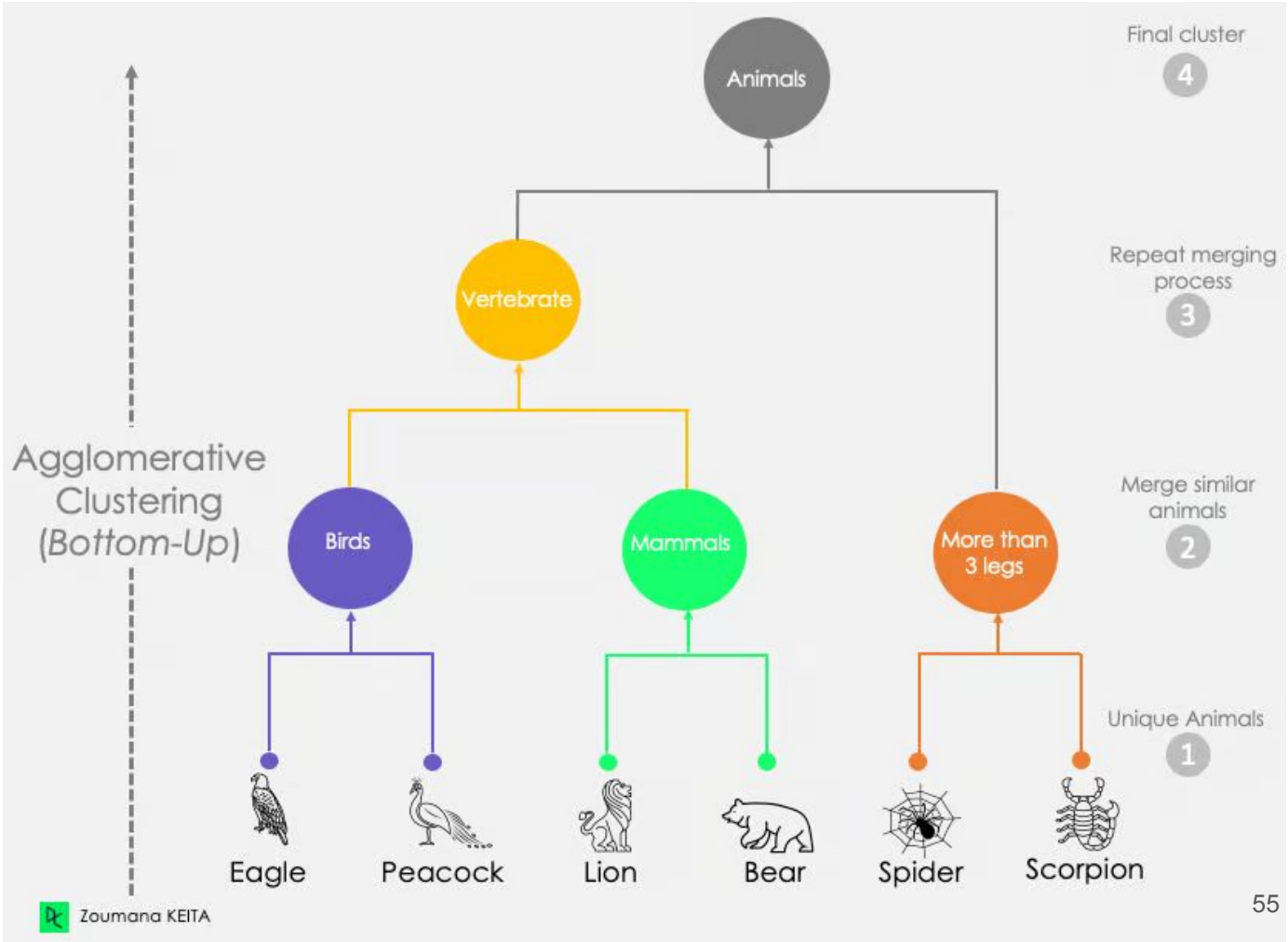
Hierachical clustering

A method to group similar items into a hierarchy of clusters.



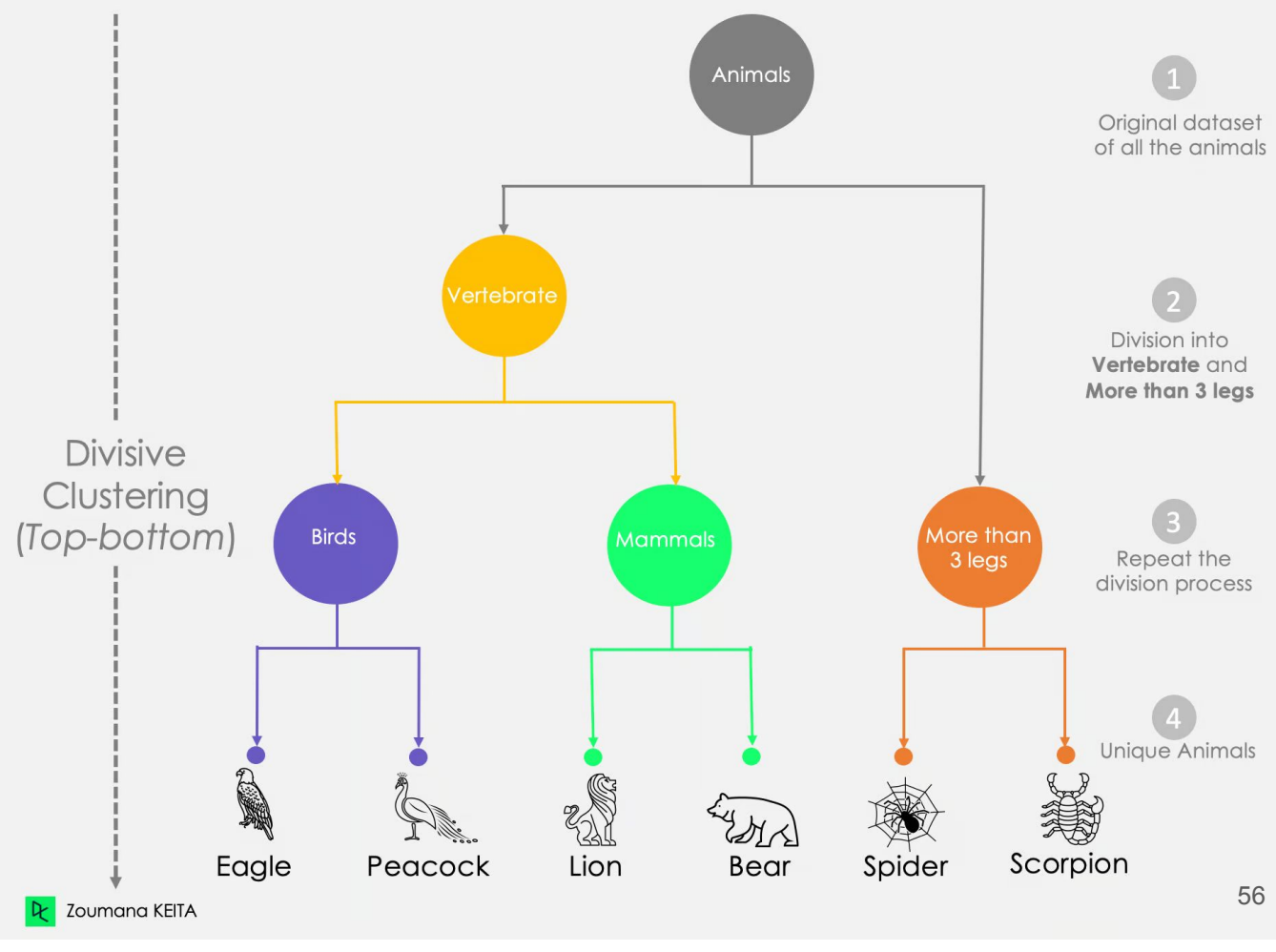
Bottom-Up

Agglomerative (Bottom-Up): Starts with each point as its own cluster, then merges them.



Top-Bottom

Divisive (Top-Down): Starts with all points in one cluster, then splits it.



How does it work?

- **Start** with individual points as their own clusters.
- **Find the closest clusters** using a distance measure (e.g., Euclidean).
- **Merge** the closest clusters.
- **Repeat** until all points are in one large cluster.
- Visualize the process with a **dendrogram** (tree-like diagram).

Pros and Cons

Advantages:

- No need to specify the number of clusters in advance.
- Hierarchical structure: Produces a dendrogram to visualize clusters at different levels.
- Flexible: Can handle various shapes of clusters.

Disadvantages:

- Computationally expensive: Slow for large datasets.
- Sensitive to noise and outliers.
- Does not scale well with large datasets.

a) K-means
b) DBSCAN
c) Hierarchical Clustering

Quiz clustering -> **menti**

1. You are analyzing gene expression data to identify groups of genes that behave similarly across different conditions. Which clustering method would you use?

2. You are conducting a study on species in an ecological dataset and want to group organisms based on environmental factors, such as temperature and humidity. The number of species is not known in advance. Which method is most appropriate?

3. You are analyzing the distribution of patients based on their health conditions (e.g., diabetes, hypertension) using their clinical data. Some patients may not fit into any of the common groups. Which clustering method will allow you to account for potential outliers?

Quiz clustering -> **menti**

a) K-means
b) DBSCAN
c) Hierarchical Clustering

4. You have a dataset with lots of noise and you're interested in discovering naturally dense areas of data (e.g., clusters of proteins in a biochemical pathway). You don't know the number of clusters. Which method would be ideal?

5. You are studying microbial communities in soil samples and want to cluster them based on various environmental measurements. The number of clusters is known to be around 5. Which method would you use?

6. You are analyzing single-cell RNA-seq data to group cells with similar expression profiles. The data may contain some cells that don't belong to any cluster. Which method is best for this?

7. You have a dataset of protein structures and want to classify them into a set number of categories (clusters). The number of categories is known beforehand. Which clustering method is most appropriate?