Application and exploration of machine learning in biological data analysis

TADAB WORKSHOP
Tools for Advanced Data Analyses in Biology

Sina Gholami

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

Manuel Tiburtini

UNIVERSITÀ DI PISA

Who are we?

Sina Gholami

Ph.D. Student in Computer Engineering at University of North Carolina, Charlotte, USA

- Application of deep learning in healthcare (ophthalmology)

What's my role in this module?

Manuel Tiburtini

PhD Student in Systematic Botany at University of Pisa, Italy

Why do I use ML?

- Because I'm fan of it
- to sort taxonomic data out

What's my role in this module?

We assume from you...

1. A basic knowledge of statistics

2. A basic knowledge of  programming language

Aims of this lecture

- Explaining machine learning modeling in the simplest way and with code instead of formulas
- Give the basic tools to start to train your own ML models
- Provide some reference and materials to deal with real data analyses

Overview of the lecture

- The meaning of statistical learning
- Supervised vs unsupervised learning
- Linear and nonlinear models
- How to improve the learning capability of a model
- An overview of tidymodels metapackage

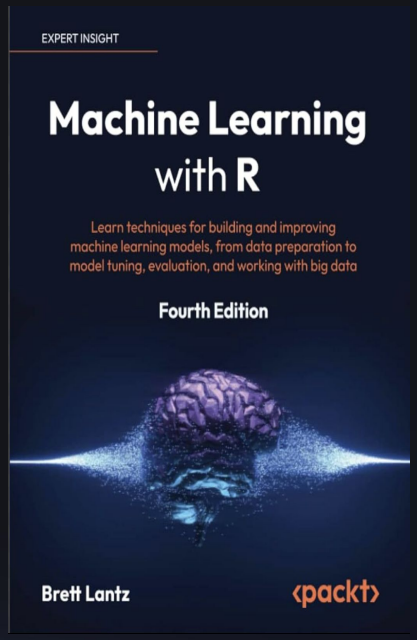
The course will be have a strong focus on classification task

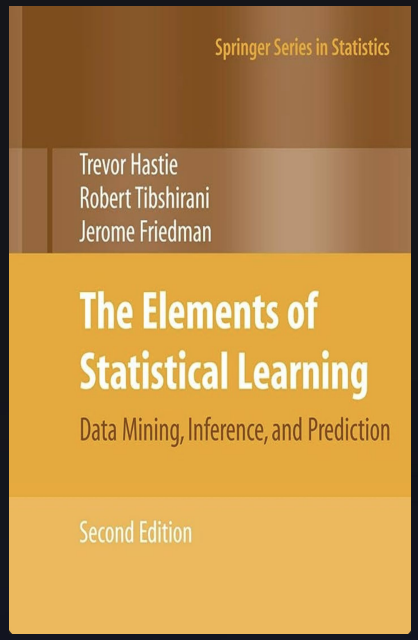
No financial interest or affiliation concerning material discussed in this presentation

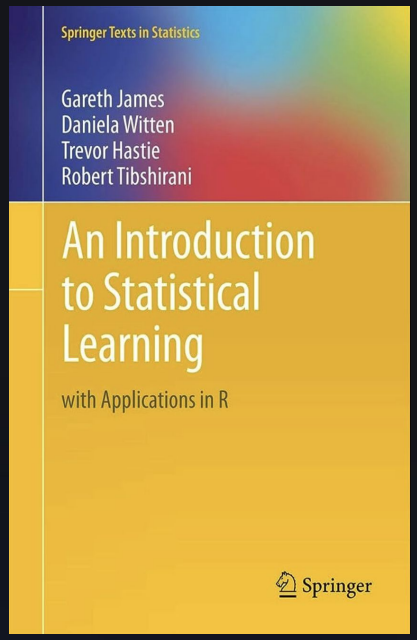
At the end of the lecture

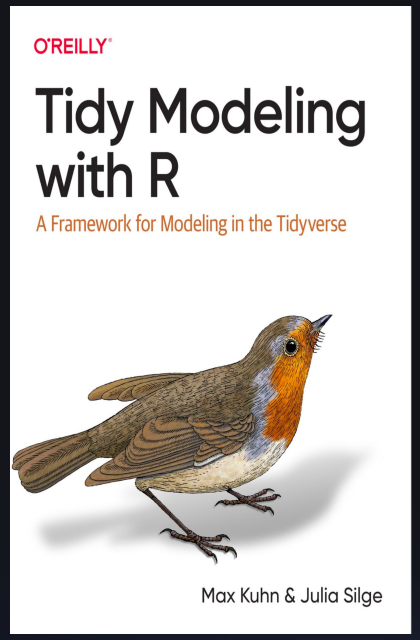
- You will have reference material to help you in your analyses
- You should be able to train effectively a model
- You have some clues on different models and their pro and cons
- Export your tuned model to be deployed (outside the scope of the course)

Reference books of the course

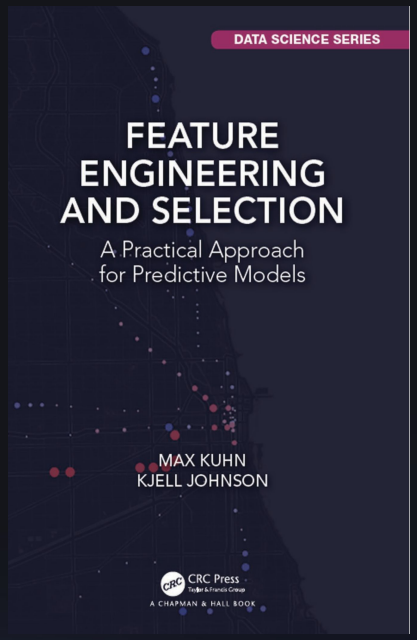


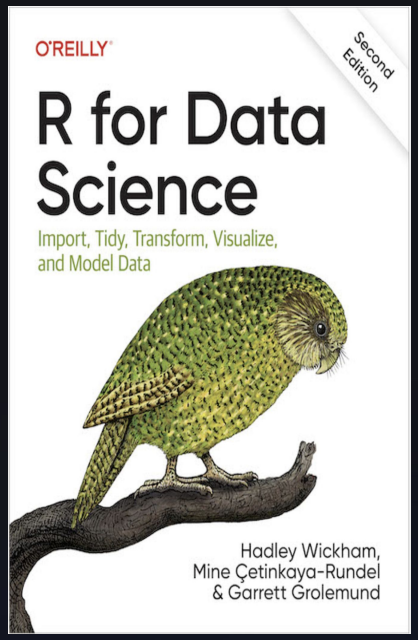


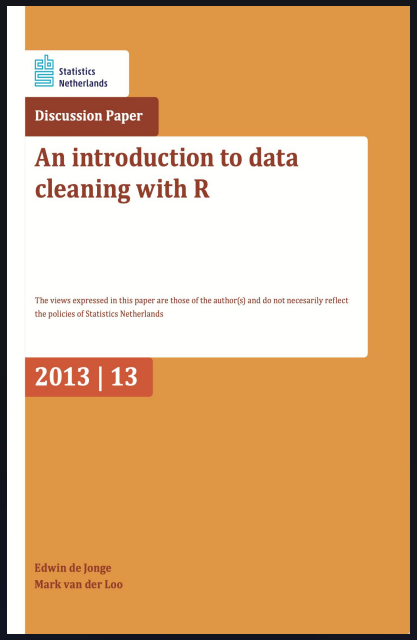


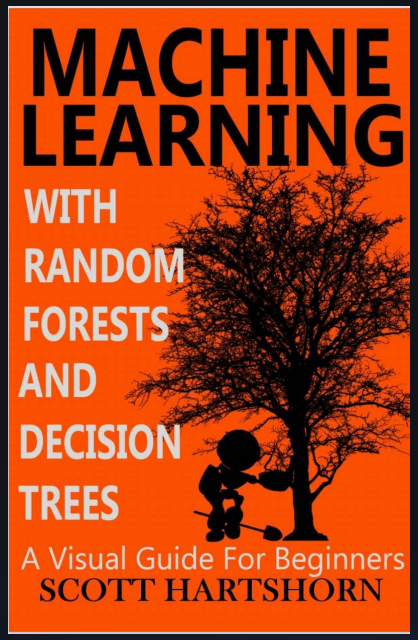


Other nice books

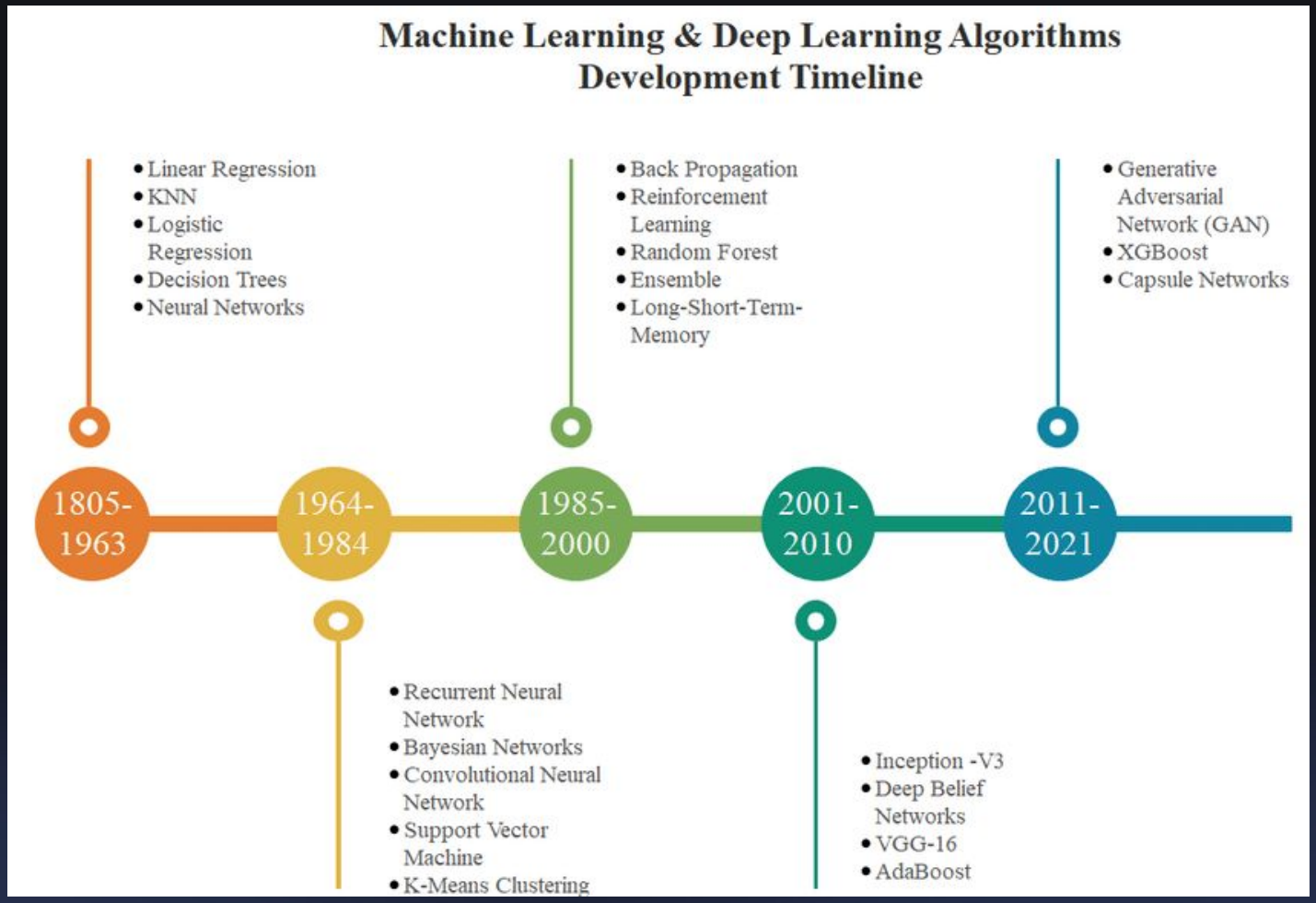








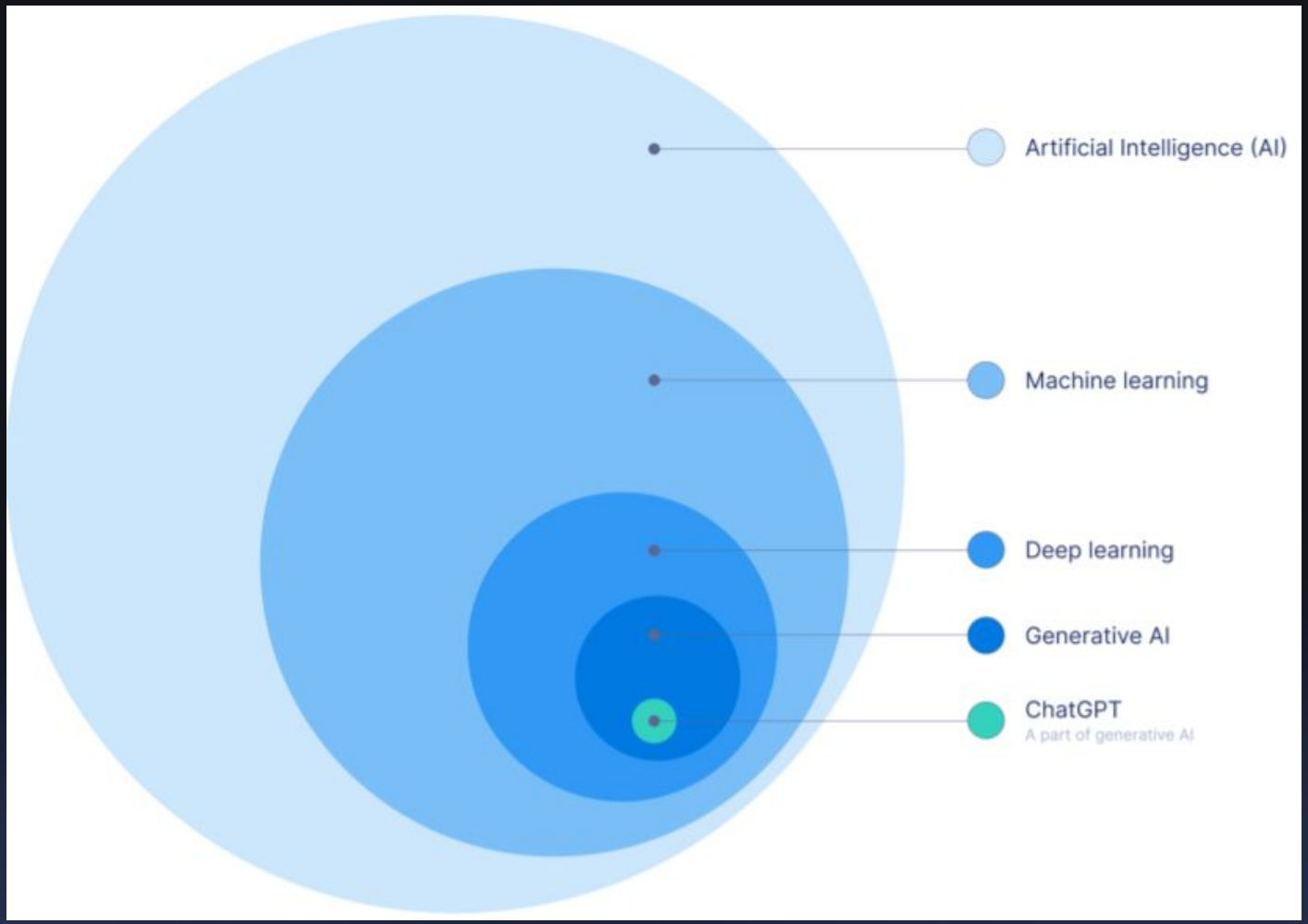
History of AI



Where AI is currently applied?

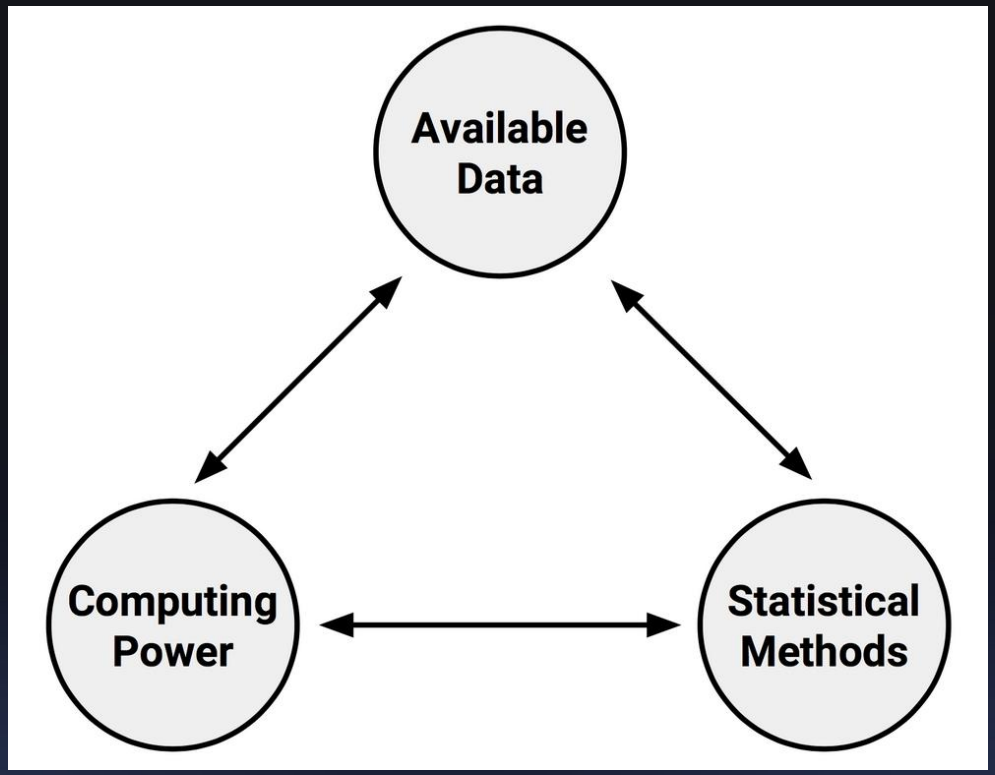
- Biology:
  - **Protein Folding**: DeepMind's AlphaFold is a prime example of how AI can revolutionize biological research. By predicting protein structures with remarkable accuracy, AlphaFold has the potential to accelerate drug discovery and understand various diseases better.
  - **Drug Discovery:** AI models are being used to predict potential drug candidates by analyzing vast chemical spaces. Companies like Atomwise and BenevolentAI are at the forefront of this.
  - **Microscopy Image Analysis**: AI is being used to analyze and interpret images from microscopes, identifying patterns and structures that might be difficult or time-consuming for humans to recognize.
- Genetics:
  - **Genome Sequencing:** AI is aiding in faster and more accurate genome sequencing. This has implications for personalized medicine and understanding genetic diseases.
  - **Gene Editing**: Tools like CRISPR have revolutionized genetics. AI can assist in predicting off-target effects and optimizing the design of guide RNAs.
  - **Genetic Trait Prediction:** AI models can predict phenotypic outcomes based on genetic information, which can be useful in agriculture and medicine.
- Botany:
  - **Plant Phenotyping:** AI-driven image analysis tools can automatically measure plant traits from images, helping in plant taxonomy.
  - **Biodiversity Analysis:** AI can assist in identifying and cataloging plant species, especially in biodiversity-rich regions.

The AI spectrum: Unveiling the layers of intelligent systems

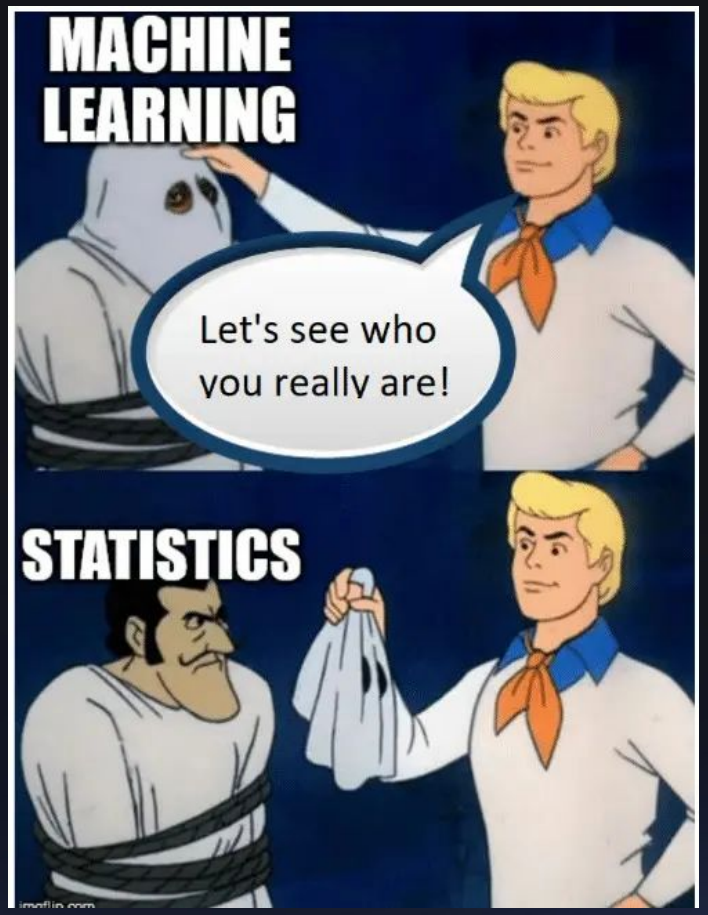


Introducing to Machine Learning (ML)

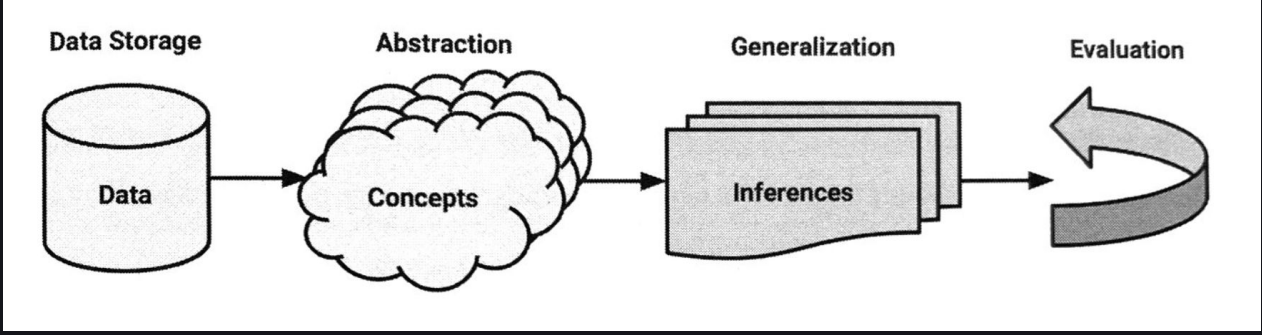
An ML model is a system that learns and improves from experience without being explicitly programmed for every instance, relying on data and algorithms to make predictions or decisions.



It’s clear that...



The steps in the learning process



1. **Data Input:** It utilizes observation, memory storage, and recall to provide a factual basis for further reasoning.

2. **Abstraction:** the translation of stored data in a model

3. **Generalization:** using the model to create knowledge and inferences that drive action in new contexts

4. **Evaluation:** provides feedback on the utility and potential improvements

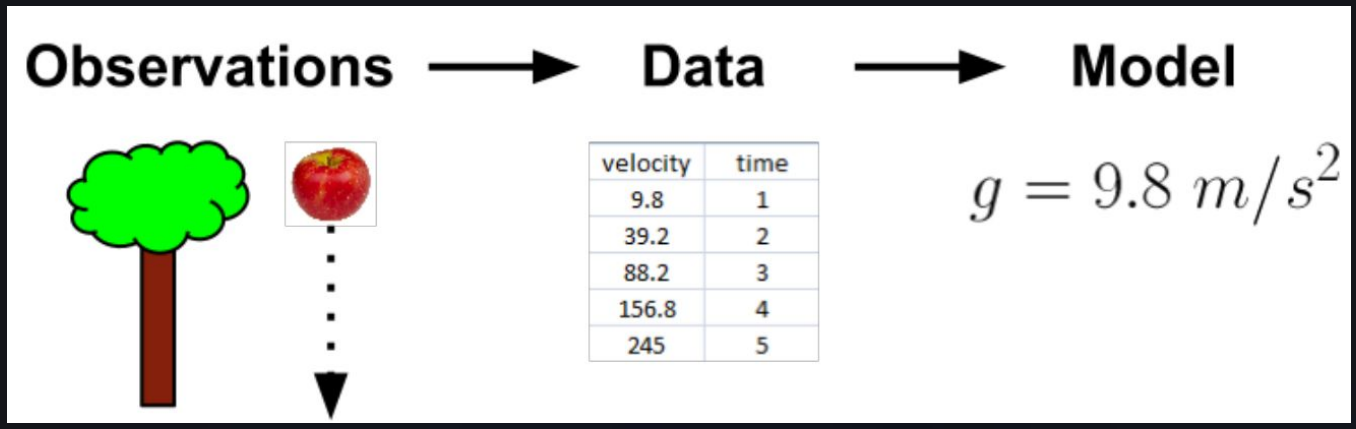
Wait!
Before hitting the ground running,
What is a model?

Models are simplification or approximation of reality and hence they will not capture all of reality.

“All models are wrong, but some are useful”
G.E.P. Box (1919–2013).

Training a model

1. During the process of knowledge representation, the computer summarizes raw inputs in a model

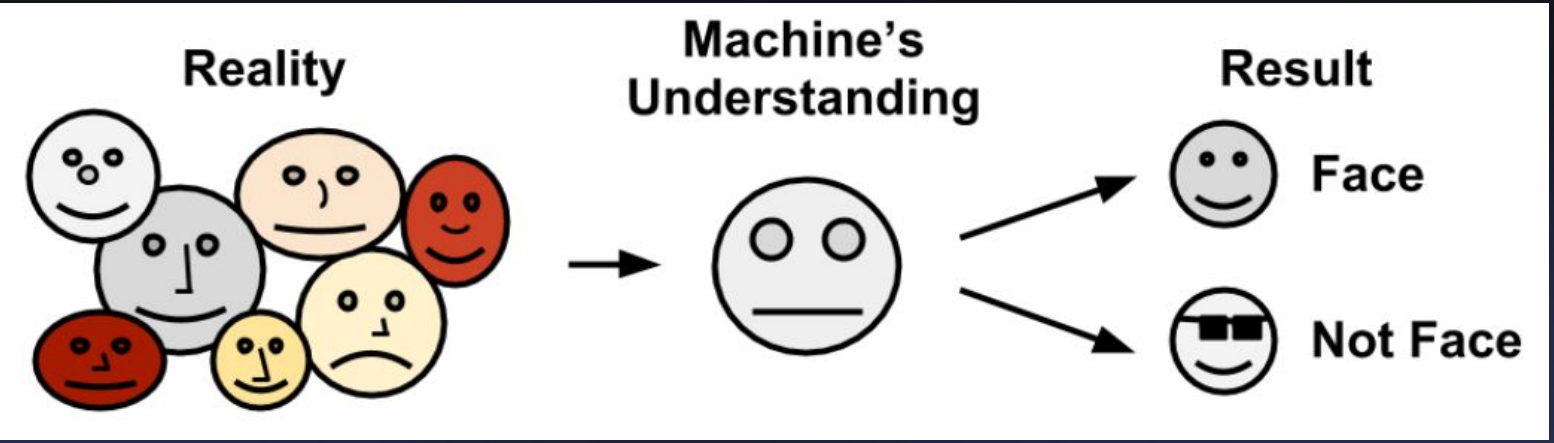


2. The process of fitting a particular model to a dataset is known as training: the data has been transformed into an abstract form that summarizes the original information.

Generalization

- Recall that the learning process is not complete until the learner is able to use its abstracted knowledge for future action.

- If the conclusions are systematically imprecise, the algorithm is said to have a **bias**. For example:



Assessing the success of learning

- **Bias** is a necessary evil associated with the abstraction and generalization process inherent in any machine learning task.

- Testing a trained model is a must!

- In part, the failure for models to perfectly generalize is due to the problem of **noise**, or unexplained variations in data.

Variance

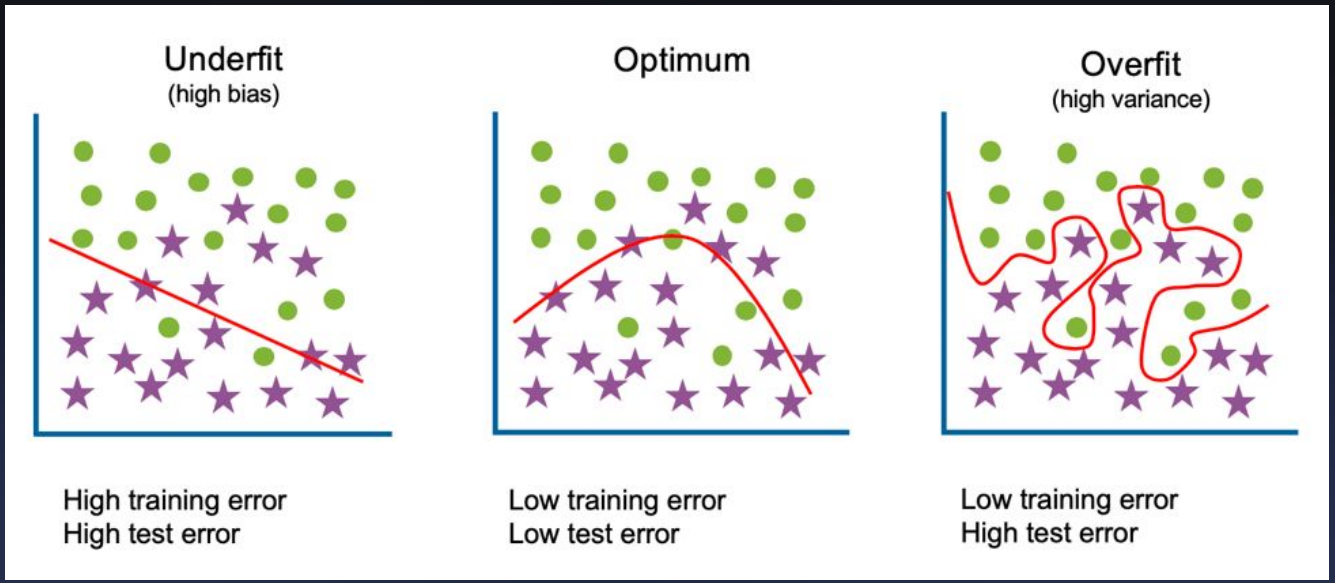
- How much the model's predictions change in response to changes in the training data.

- **Bias:** the error introduced by approximating a real-world problem with a simpler model. if the model is too simple, it might not capture important pattern in the data.

- **Noise:** Random error or variability in the data that cannot be accounted for by the moel. Due to measurement errors, irrelevant information.

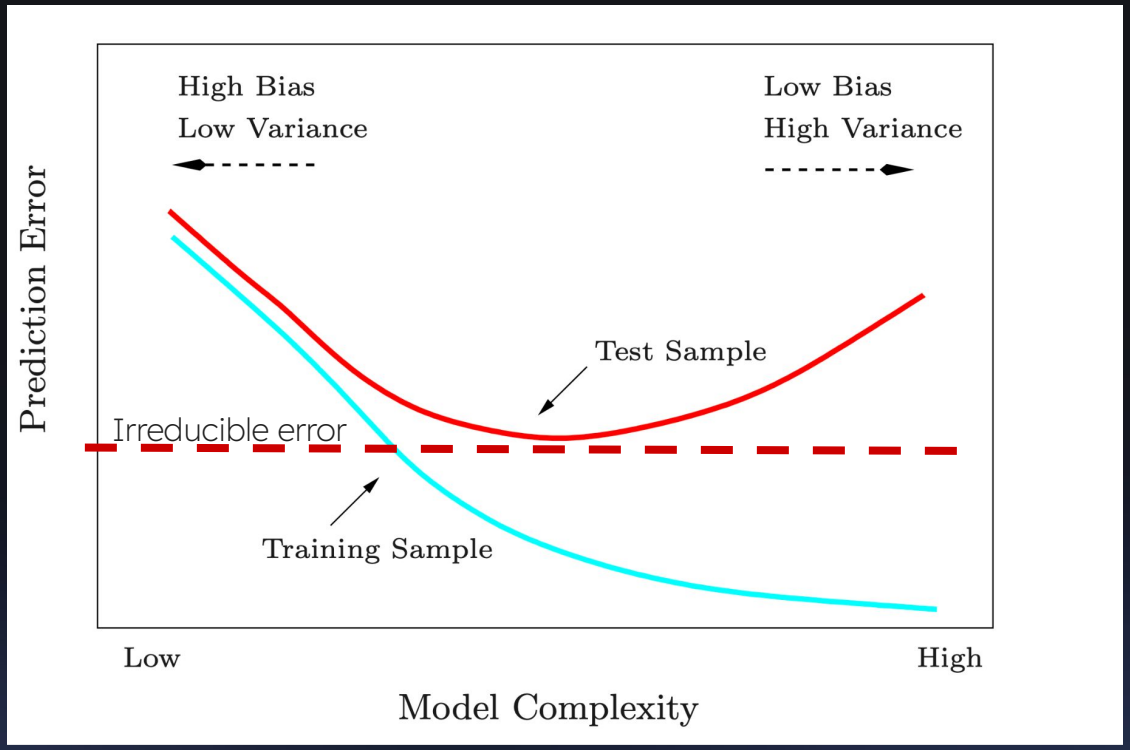
Overfitting

- Trying to model the noise in data is the basis of a problem called **overfitting.**
- Because noise is unexplainable by definition, attempting to explain the noise will result in erroneous conclusions that do not generalize well to new cases.



Bias – Variance Tradeoff

Generalization error = Bias^2 + Variance + irriducibile error



For those how wants more details

$$\mathbb{E}[(f(x) - \hat{f}(x))^2] = \mathbb{E}\left[\left((f(x) - \mathbb{E}[\hat{f}(x)]) - (\hat{f}(x) - \mathbb{E}[\hat{f}(x)])\right)^2\right] \tag{4}$$

$$= \mathbb{E}\left[\left(\mathbb{E}[\hat{f}(x)] - f(x)\right)^2\right] + \mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)^2\right]$$

$$- 2\mathbb{E}\left[\left(f(x) - \mathbb{E}[\hat{f}(x)]\right)\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)\right] \tag{5}$$

$$= \underbrace{(\mathbb{E}[\hat{f}(x)] - f(x))^2}_{=\text{bias}[\hat{f}(x)]} + \underbrace{\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)^2\right]}_{=\text{var}(\hat{f}(x))}$$

$$- 2\left(f(x) - \mathbb{E}[\hat{f}(x)]\right)\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)\right] \tag{6}$$

$$= \text{bias}[\hat{f}(x)]^2 + \text{var}(\hat{f}(x))$$

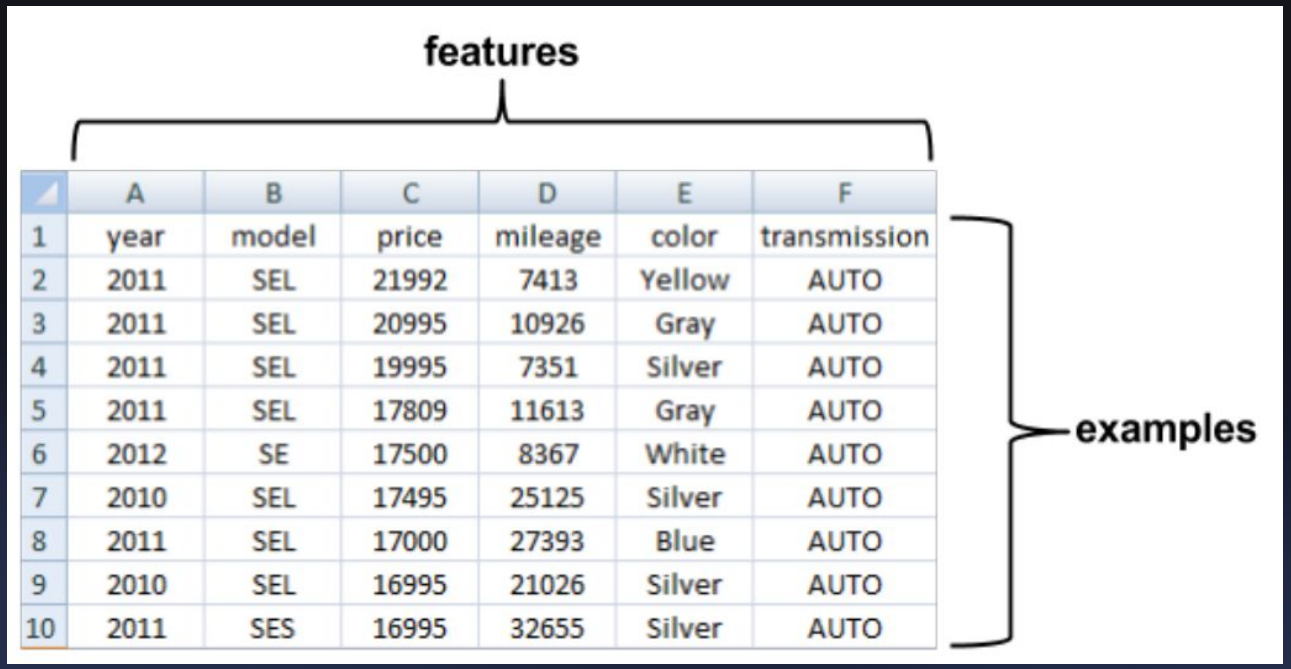
$$- 2\left(f(x) - \mathbb{E}[\hat{f}(x)]\right)\left(\mathbb{E}[\hat{f}(x)] - \mathbb{E}[\hat{f}(x)]\right) \tag{7}$$

$$= \text{bias}[\hat{f}(x)]^2 + \text{var}(\hat{f}(x)) \tag{8}$$

Steps to apply machine learning to your data

Features

Feature Engineering: It's the process of creating representations of data that increase the effectiveness of a model. Extracting most important characteristics of the data.



	A	B	C	D	E	F
1	year	model	price	mileage	color	transmission
2	2011	SEL	21992	7413	Yellow	AUTO
3	2011	SEL	20995	10926	Gray	AUTO
4	2011	SEL	19995	7351	Silver	AUTO
5	2011	SEL	17809	11613	Gray	AUTO
6	2012	SE	17500	8367	White	AUTO
7	2010	SEL	17495	25125	Silver	AUTO
8	2011	SEL	17000	27393	Blue	AUTO
9	2010	SEL	16995	21026	Silver	AUTO
10	2011	SES	16995	32655	Silver	AUTO

Data Preparation

- **Training vs Validation vs. Test data Distribution**: We assume that the training, validation and test examples are i.i.d. drawn from the same overall distribution of data.

- You have enough data for your data sets to be representative.

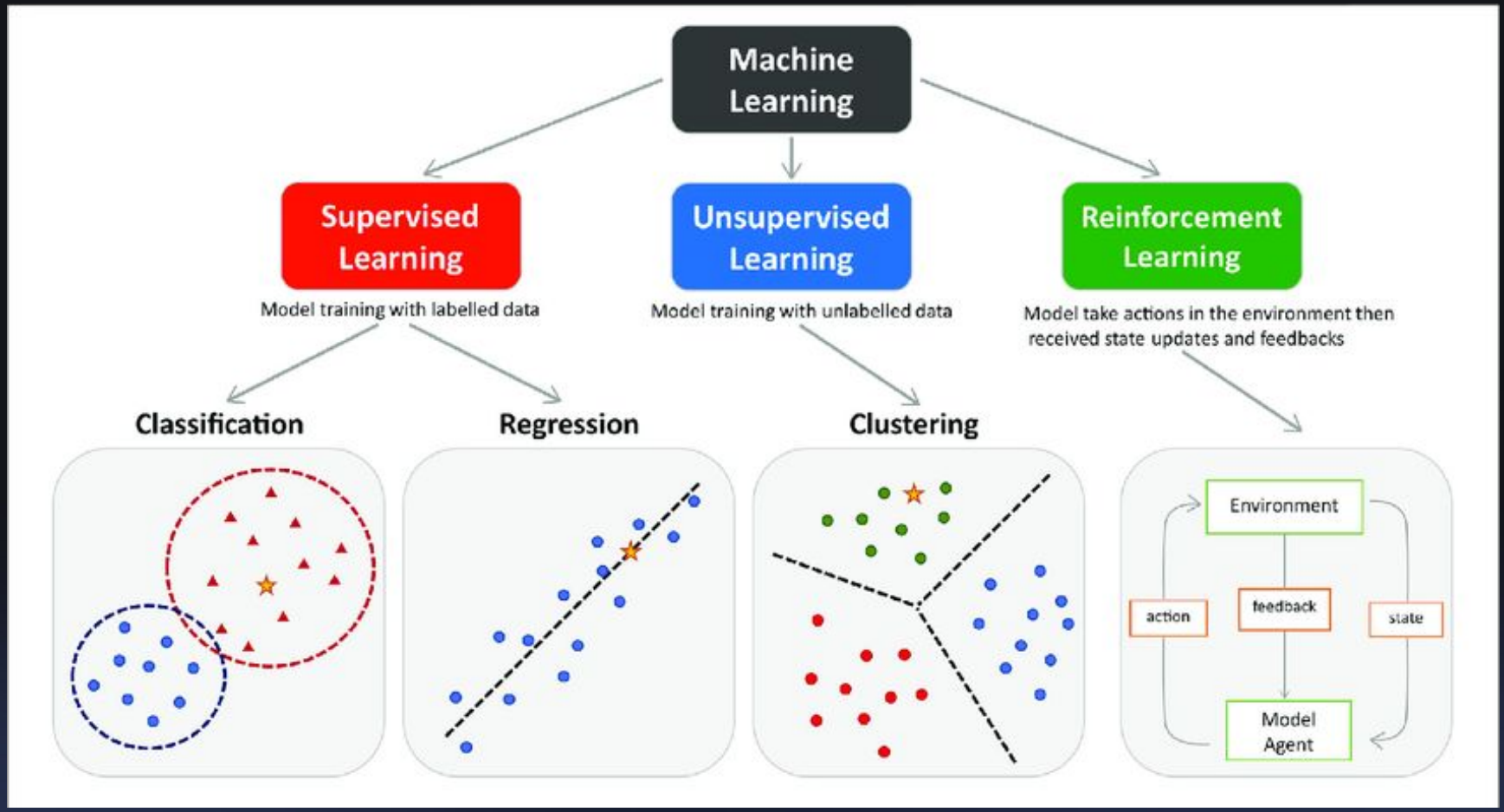
- Ideally, we seek to have a balance of the outcome in both training and testing data.

Datasets

There is No Free Lunch in ML! (and in life in general)

⚠️ Beware of data leakage ⚠️

Models


Machine Learning
Supervised Learning
Model training with labelled data
Unsupervised Learning
Model training with unlabelled data
Reinforcement Learning
Model take actions in the environment then received state updates and feedbacks
Classification
Regression
Clustering
Environment
action
feedback
state
Model Agent

Other fancy ML paradigms

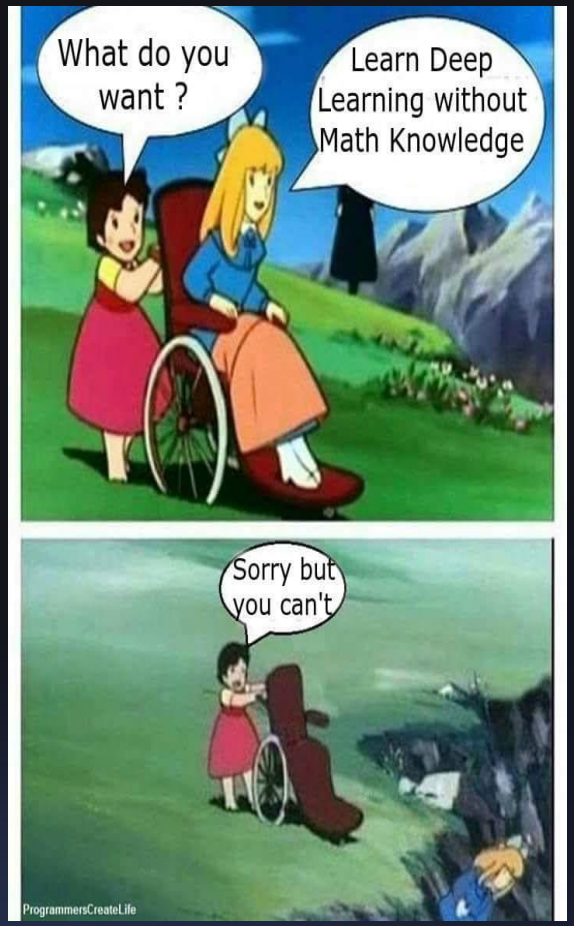
- Self - Supervised learning → Encourage a model to train from unlabeled data making new label to learn from them
- Meta Learning → Learning of how to learn more effectively (es. ensambles)
- Semi - Supervised learning → combine labelled and unlabelled data to improve the model performances
- Transfer learning → A model has been trained in one setting and is exploited to improve generalization in another setting.
- Lazy learning → ⚠️ is not a ML paradigm!

ML models

- Supervised Learning:
  - Linear Regression & Logistic Regression
  - Support Vector Machines (SVM)
  - K-Nearest Neighbor (KNN)
  - Decision Tree
  - Random Forest
  - Neural Networks (NN)

- Unsupervised Learning:
  - Principal Component Analysis (PCA)
  - K-means
  - Uniform Manifold Approximation and Projection (UMAP)
  - Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

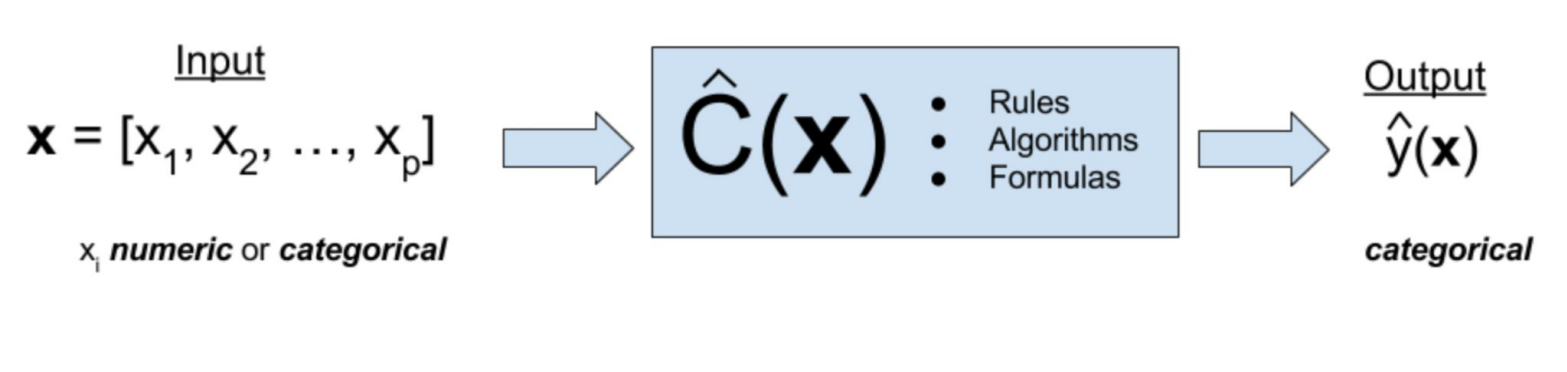
Before starting...!



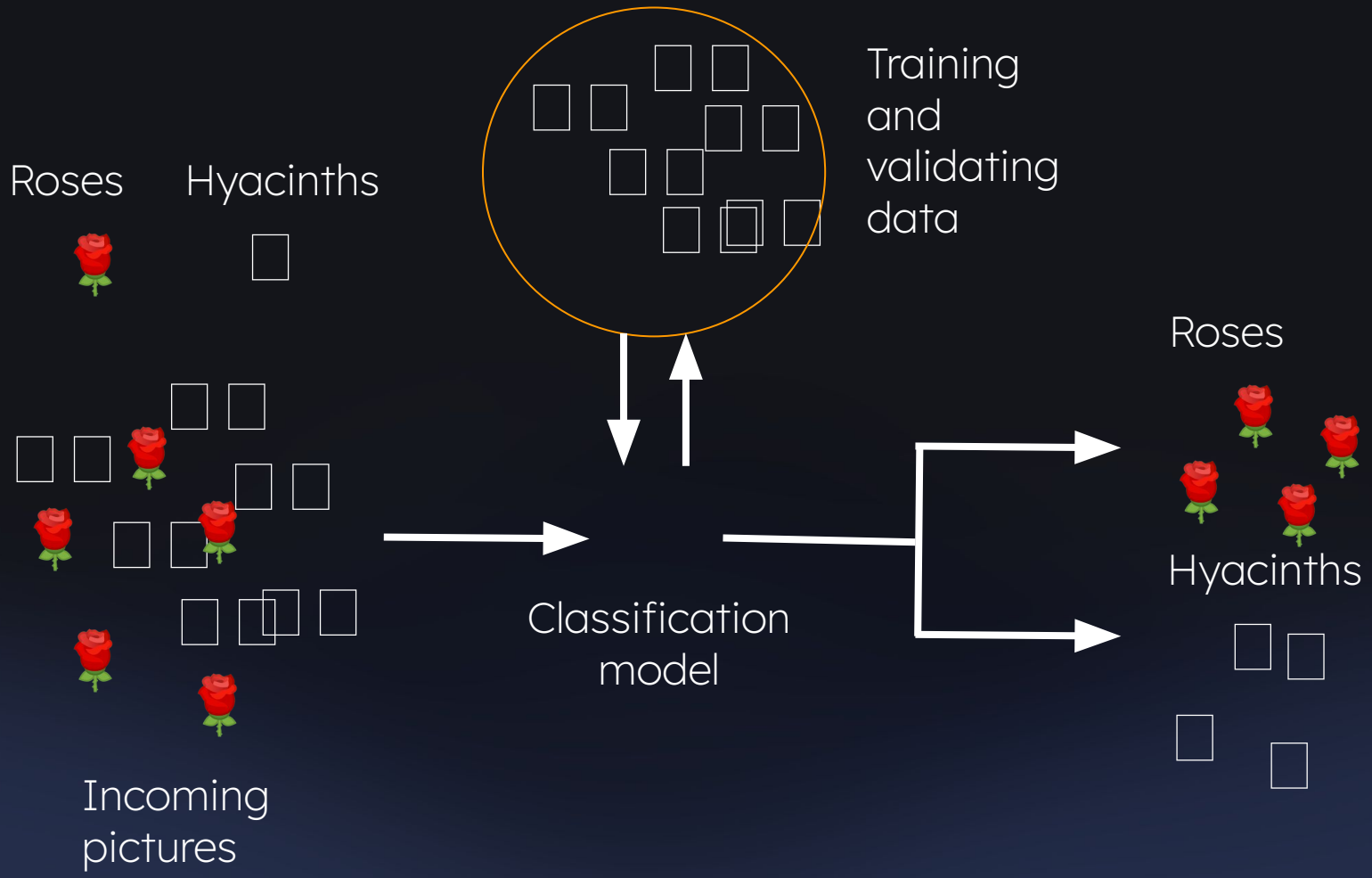
... but you can still learn something

Classification

A supervised learning technique where the model learns to categorize or classify data into predefined classes or labels.



Example of Classification



Classification metrics

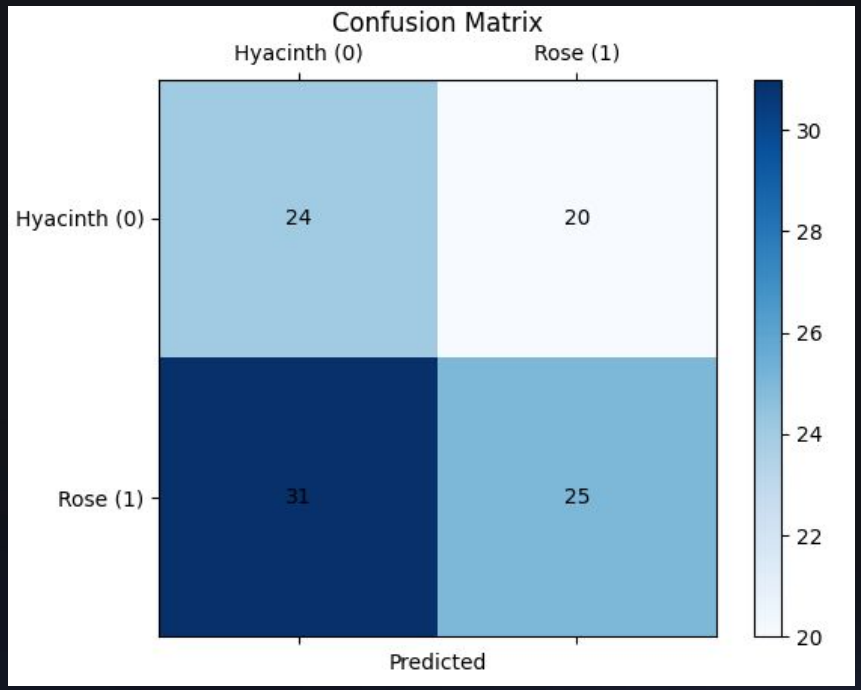
The standard metric is the **classification error rate**

ER = (FP + FN)/ (TP + TN + FP + FN)

	Predicted Positive	Predicted Negative	
Actual Positive	TP *True Positive*	FN *False Negative*	Sensitivity $\frac{TP}{(TP+FN)}$
Actual Negative	FP *False Positive*	TN *True Negative*	Specificity $\frac{TN}{(TN+FP)}$
	Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

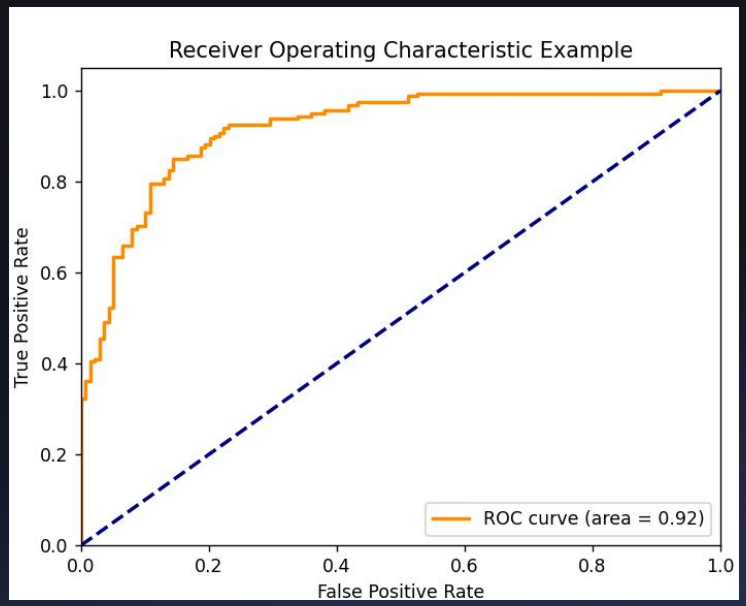
Example of classification metrics

- 24 hyacinths were correctly classified (True Negatives)
- 20 roses were incorrectly classified as hyacinths (False Positives)
- 31 hyacinths were incorrectly classified as roses (False Negative)
- 25 roses were correctly classified (True Positives)



ROC Analysis

- **Definition:** The ROC curve is a plot that illustrates the diagnostic ability of a binary classification system as its discrimination threshold is varied. It's used to assess the trade-offs between true positive rate (sensitivity) and false positive rate (1-specificity).
- **Curve Analysis**
- **Interpreting the Curve:**
  - Area Under the Curve (AUC)
  - Higher AUC



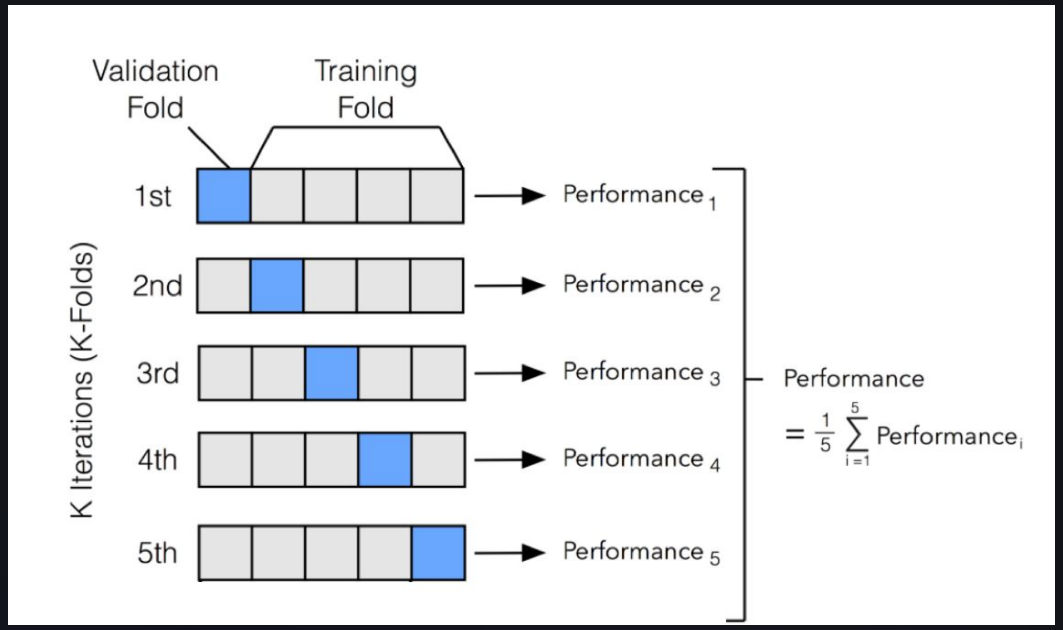
Parameters vs hyperparameters

- A **model parameter** is a configuration variable that is internal to the model and whose value can be estimated from data.

- A **model hyperparameter** is a configuration that is external to the model and whose value cannot be estimated from data.

K-fold cross-validation

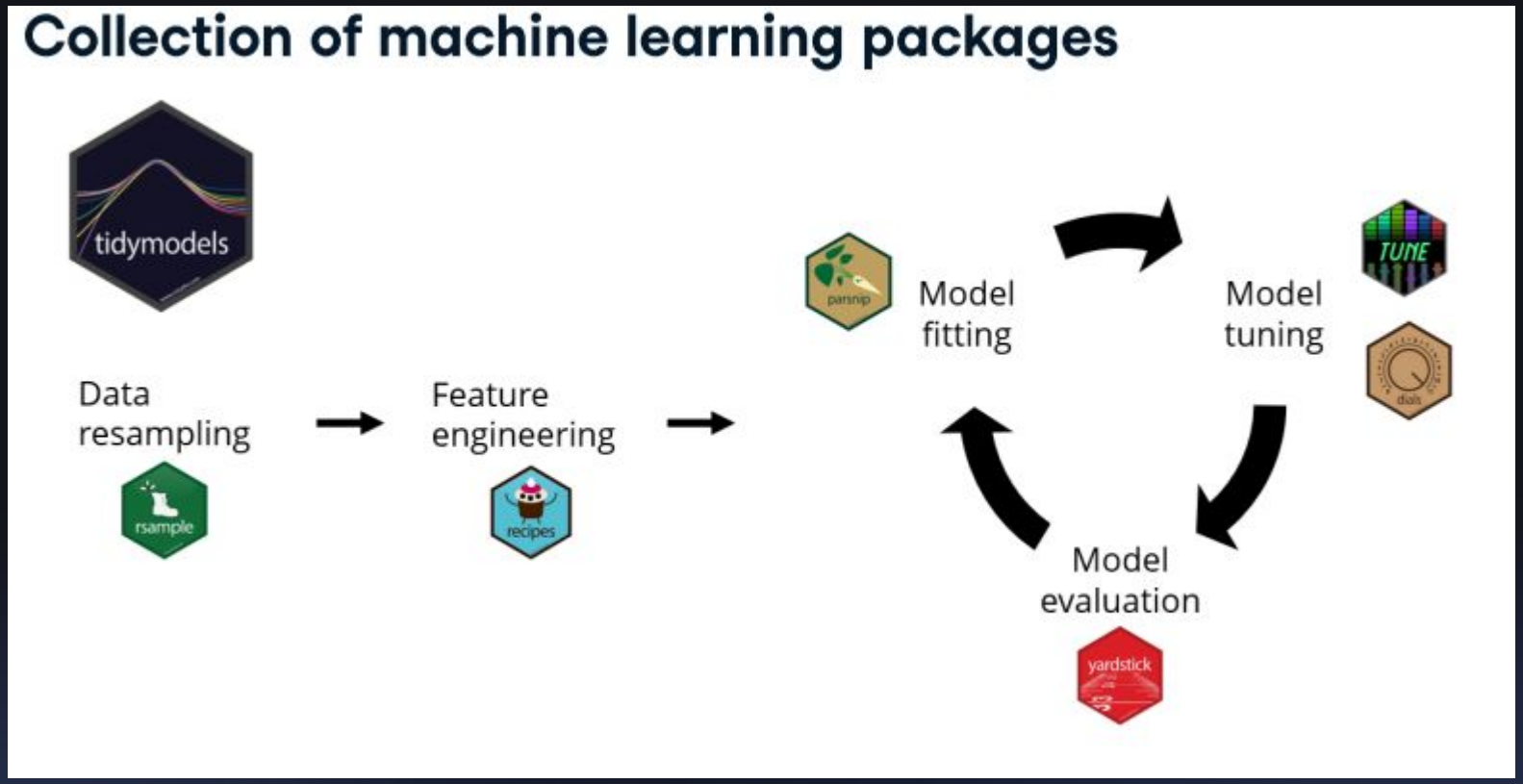
- Advantages
  - Reduced Bias
  - More Accurate Assessment
- Disadvantages
  - Computational Cost



  - Not Ideal for Very Large Datasets: Due to the increased computational cost, it might not be practical for very large datasets.
  - Time-Consuming: Especially with models that have longer training times.

Question & Answering

The Tidymodel metapackage



https://workshops.tidymodels.org/

https://emilhvitfeldt.github.io/ISLR-tidymodels-labs/12-unsupervised-learning.html

Why Tidymodels? *Consistency*

With `lm()`:

```
1 model <-
2   lm(mpg ~ ., mtcars)
```

With tidymodels:

```
1 model <-
2   linear_reg() %>%
3   set_engine("lm") %>%
4   fit(mpg ~ ., mtcars)
```

Why Tidymodels? *Consistency*

With glmnet:

```
model <-
  glmnet(
    as.matrix(mtcars[2:11]),
    mtcars$mpg
  )
```

With tidymodels:

```
model <-
  linear_reg() %>%
  set_engine("glmnet") %>%
  fit(mpg ~ ., mtcars)
```

Why Tidymodels? *Consistency*

With h2o:

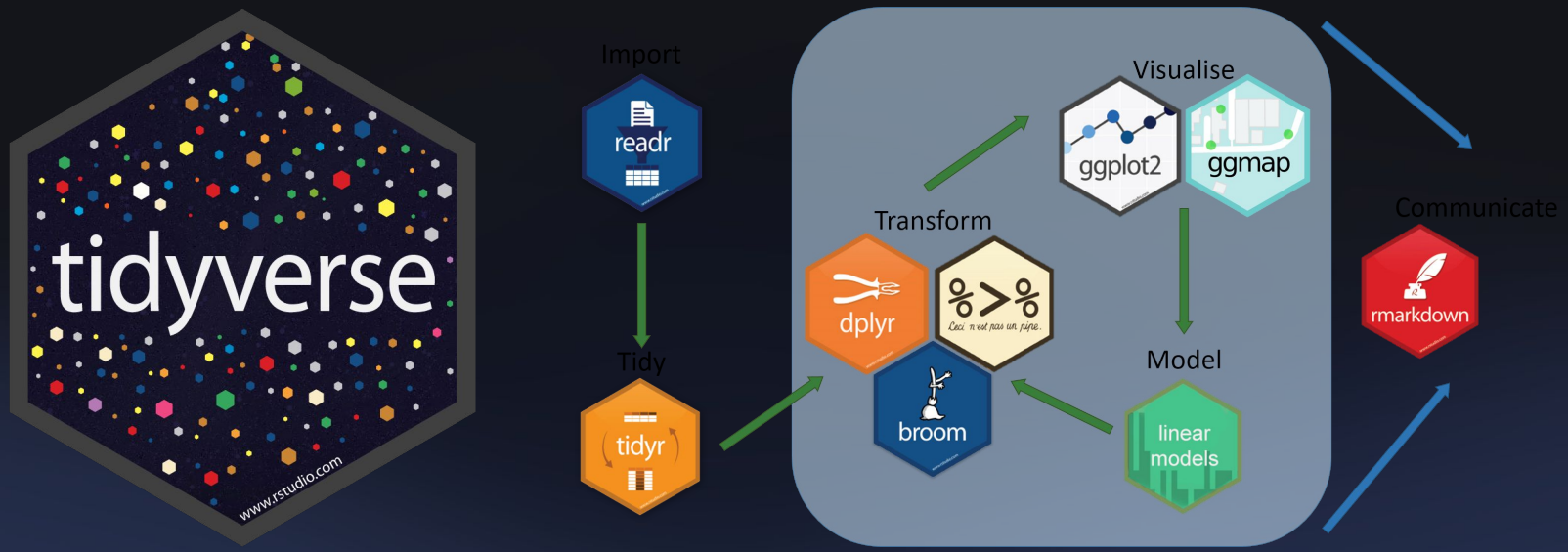
```
h2o::h2o.init()
as.h2o(mtcars, "mtcars")

model <-
  h2o.glm(
    x = colnames(mtcars[2:11]),
    y = "mpg",
    "mtcars"
  )
```

With tidymodels:

```
model <-
  linear_reg() %>%
  set_engine("h2o") %>%
  fit(mpg ~ ., mtcars)
```

Why Tidymodels? *Clarity*

- It's designed for functional programming on the tidy principle.
- Ensure code cleanness and readability, especially for



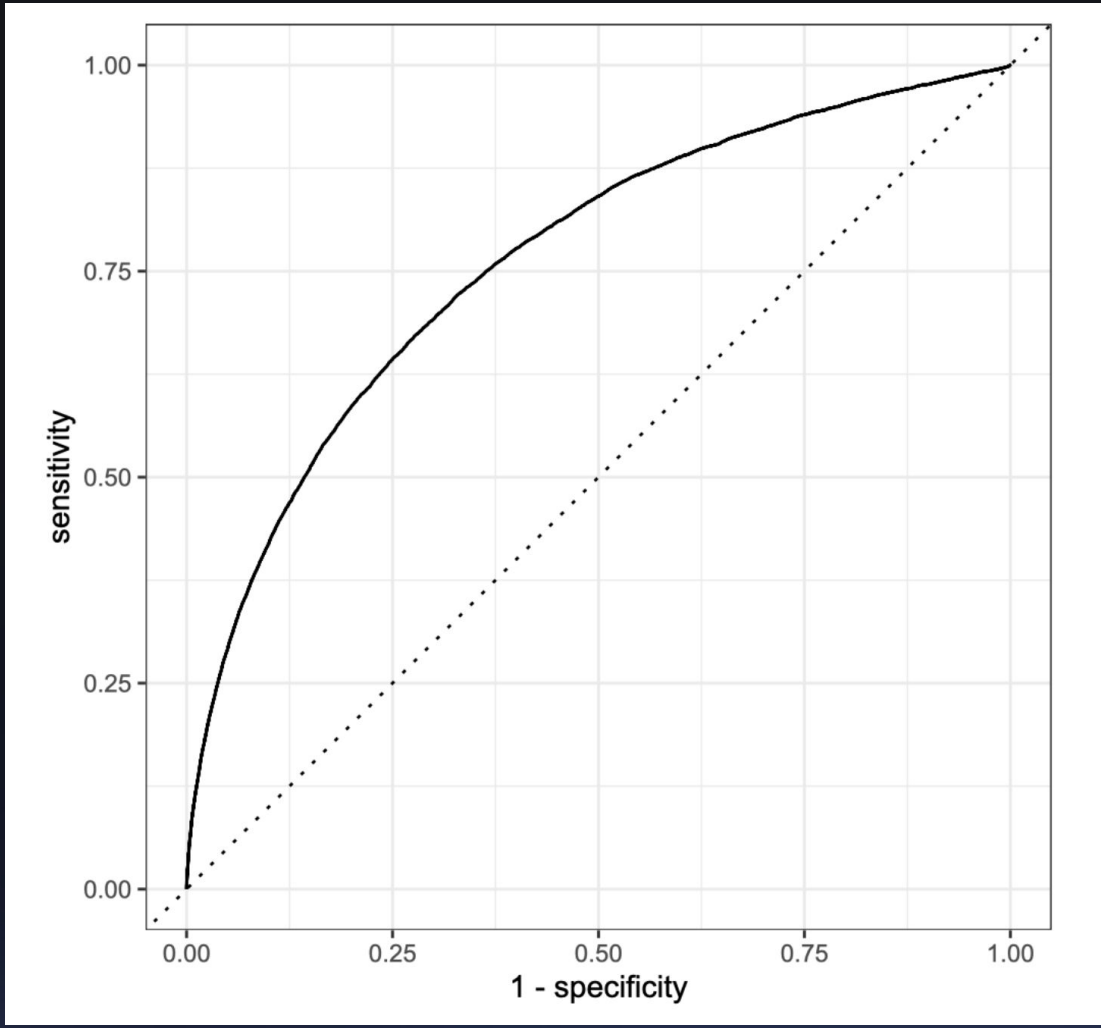
Why Tidymodels? *Safety*

- A 2023 review found data leakage to be “a widespread failure mode in machine-learning (ML)-based science.” Kapoor & Narayanan (2023); Patterns (NY) 4(9).
- Overfitting leads to analysts believing models are more performant than they actually are.
- Implementations of the same machine learning model give differing results, resulting in irreproducibility of modeling results.
- Give you a general path during modelling

Why Tidymodels? *Communicability*

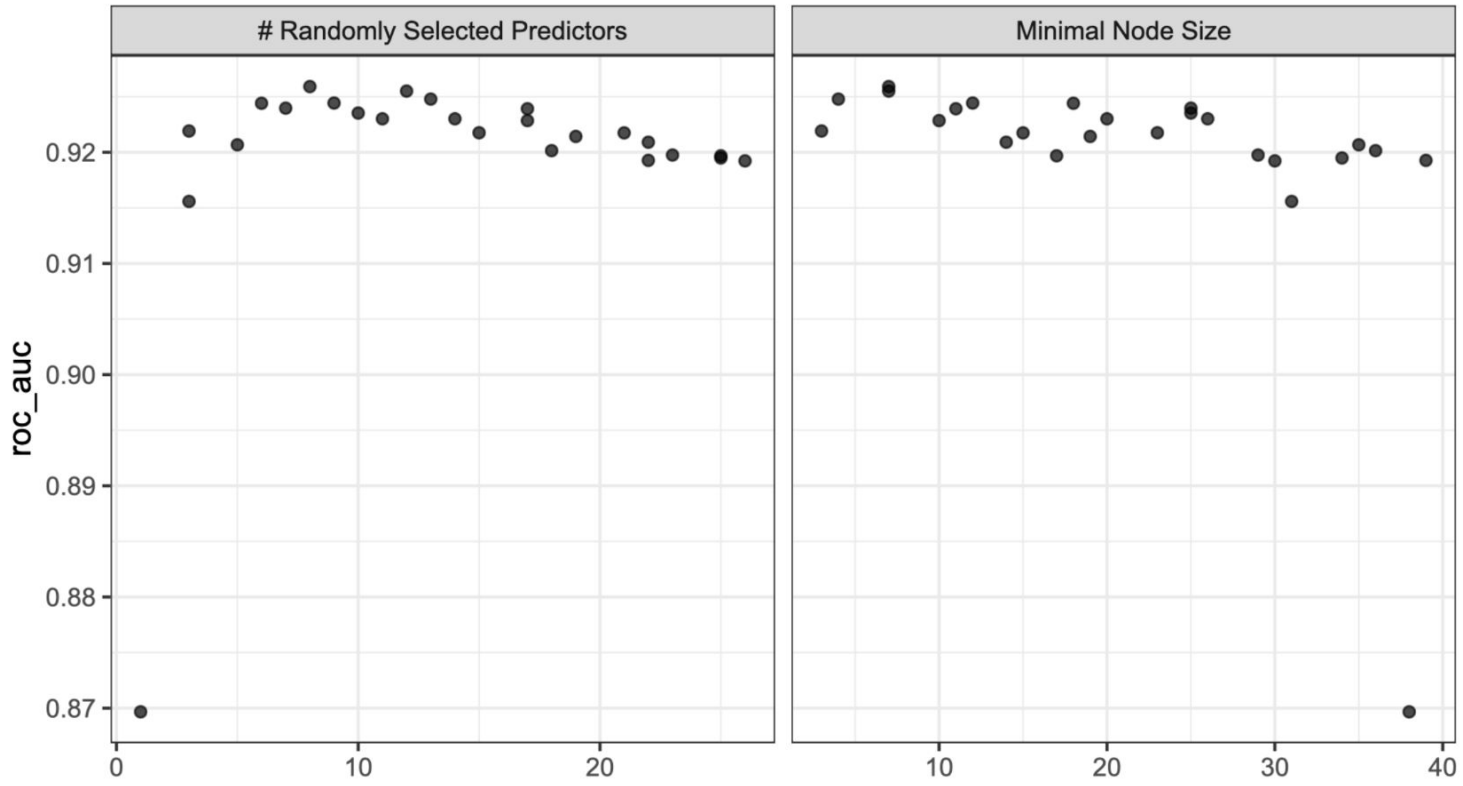
```
1  autoplot(roc_curve)
```

Tidymodels objects can easily be visualized



Why Tidymodels? *Communicability*

```
1 autoplot(random_forest_tuning_result)
```



Why Tidymodels? *Completeness*

Tidymodels support 43 models natively, up to 99 with extensions!

```
# A tibble: 43 × 2
   name             engine
   <chr>            <chr>
 1 null_model       parsnip
 2 multinom_reg     brulee
 3 multinom_reg     glmnet
 4 multinom_reg     keras
 5 multinom_reg     nnet
 6 multinom_reg     spark
 7 nearest_neighbor kknn
 8 svm_rbf          kernlab
 9 svm_rbf          liquidSVM
10 svm_poly         kernlab
# ℹ 33 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Why Tidymodels? *Completeness*

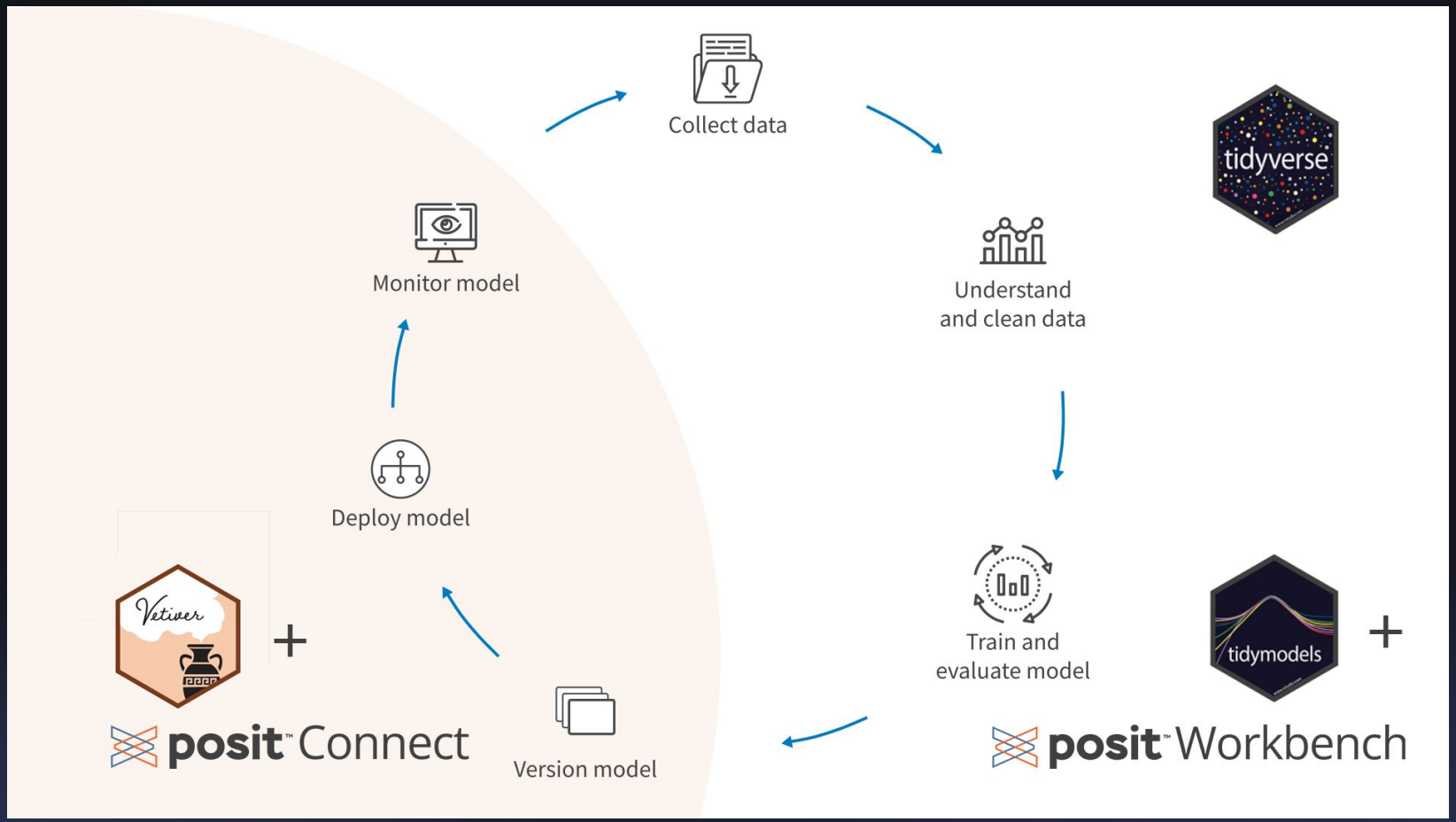
Built-in support for 102 data pre-processing techniques!

```
# A tibble: 102 × 1
   name
   <chr>
 1 step_relu
 2 step_lincomb
 3 step_count
 4 step_novel
 5 step_inverse
 6 step_corr
 7 step_relevel
 8 step_sqrt
 9 step_depth
10 step_kpca
# ℹ 92 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Why Tidymodels? *Extensibility*

- If the model you are looking for is missing, you can integrate yourself!
- It support the various types of modelling, not just ML
- Tightly integrated with other tidy packages

Why Tidymodels? *Deployability*



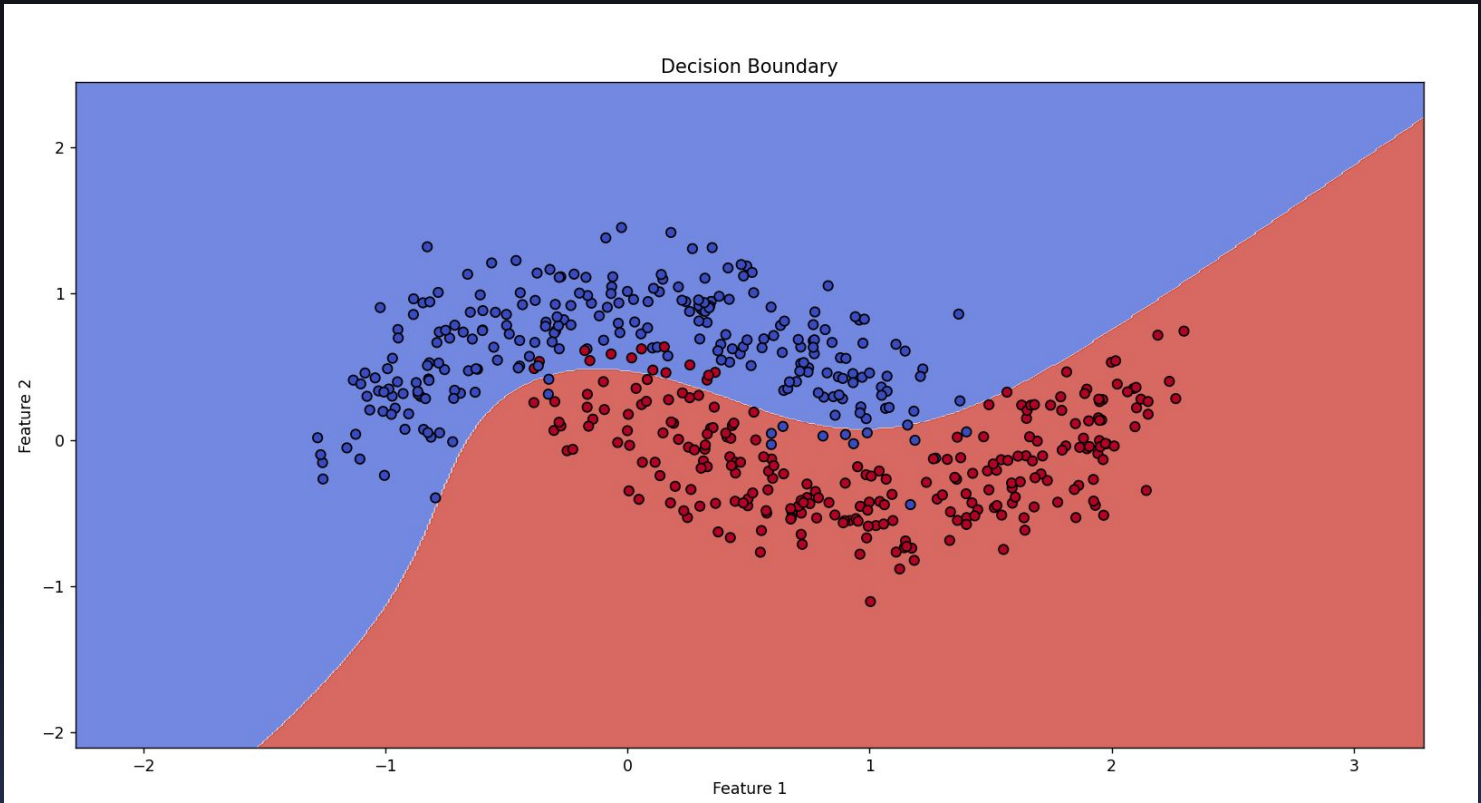
Logistic Regression

Logistic Regression is a widely-used statistical method for binary classification.

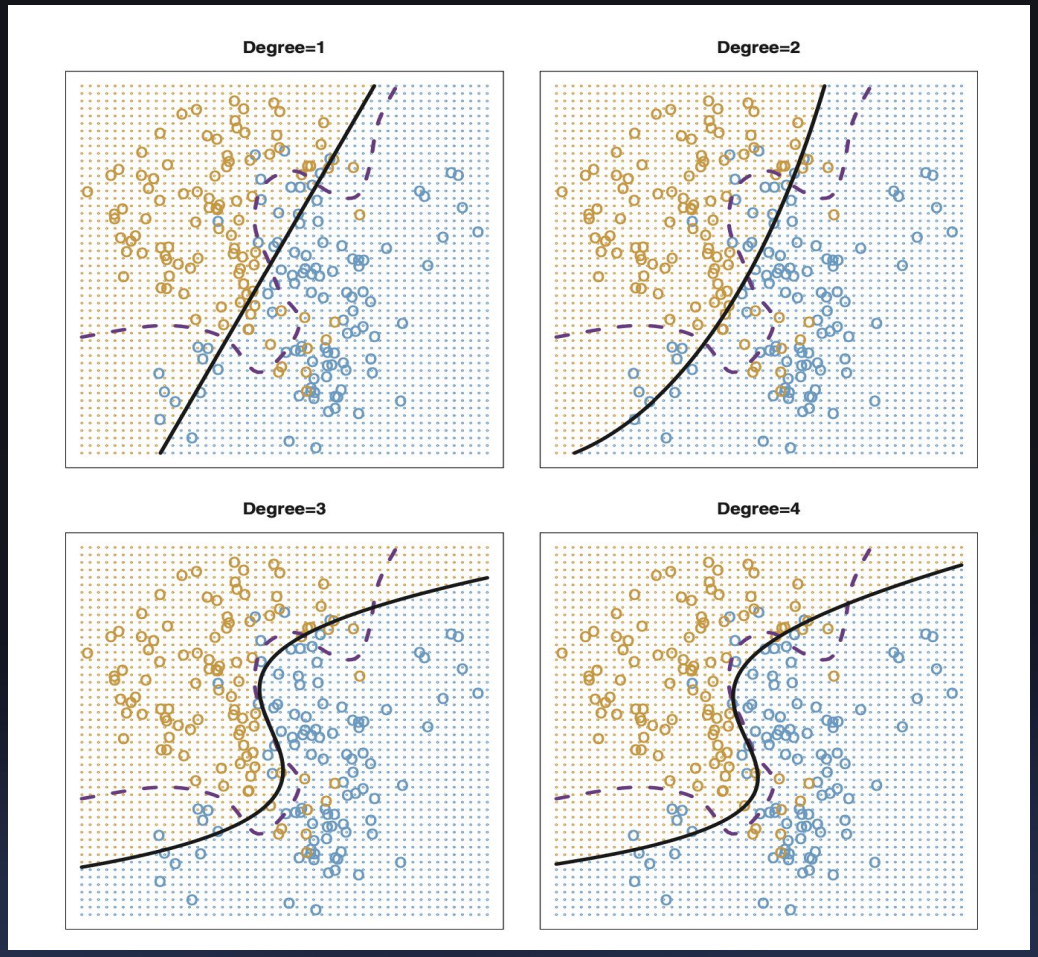
- **Binary Classification**.
- **Probability Estimation**: It outputs values between 0 and 1, interpreted as the likelihood of belonging to a certain class.
- **Sigmoid Function:** It utilizes the sigmoid (or logistic) function to model the probability.

- **Applications and Limitations:** Widely used in various fields like medicine, social sciences, and machine learning. However, it assumes a linear relationship between the independent variables and the logit of the dependent variable, and can struggle with complex relationships or high-dimensionality without regularization (High bias, low variance)

Example of Logistic Regression



$$y = \frac{e^{(b_0 + b_1 X)}}{1 + e^{(b_0 + b_1 X)}}$$



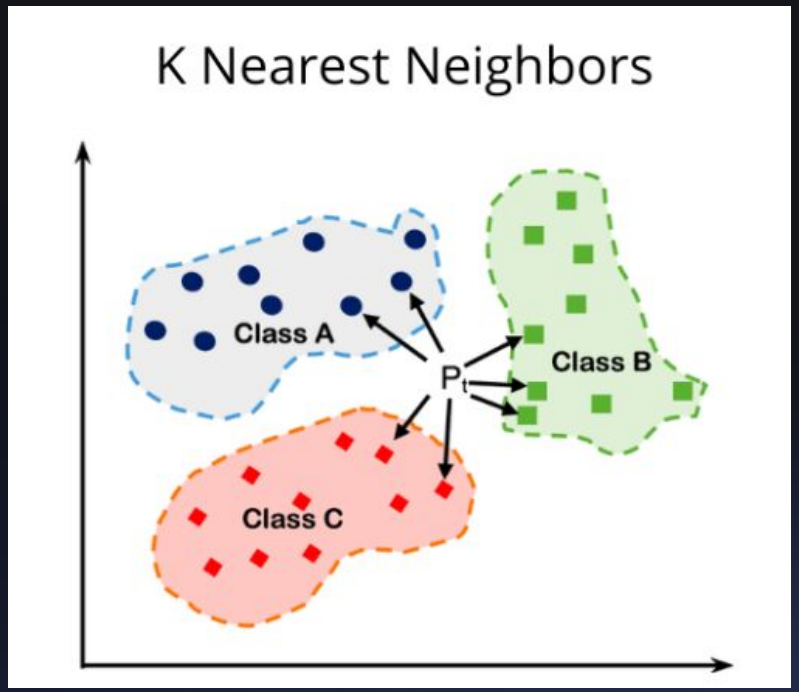
K-nearest neighbors (KNN)

It works by finding the 'k' closest training examples in the feature space and making predictions based on these neighbors.

- For Classification and Regression
- Choice of 'k'
- Distance Metrics

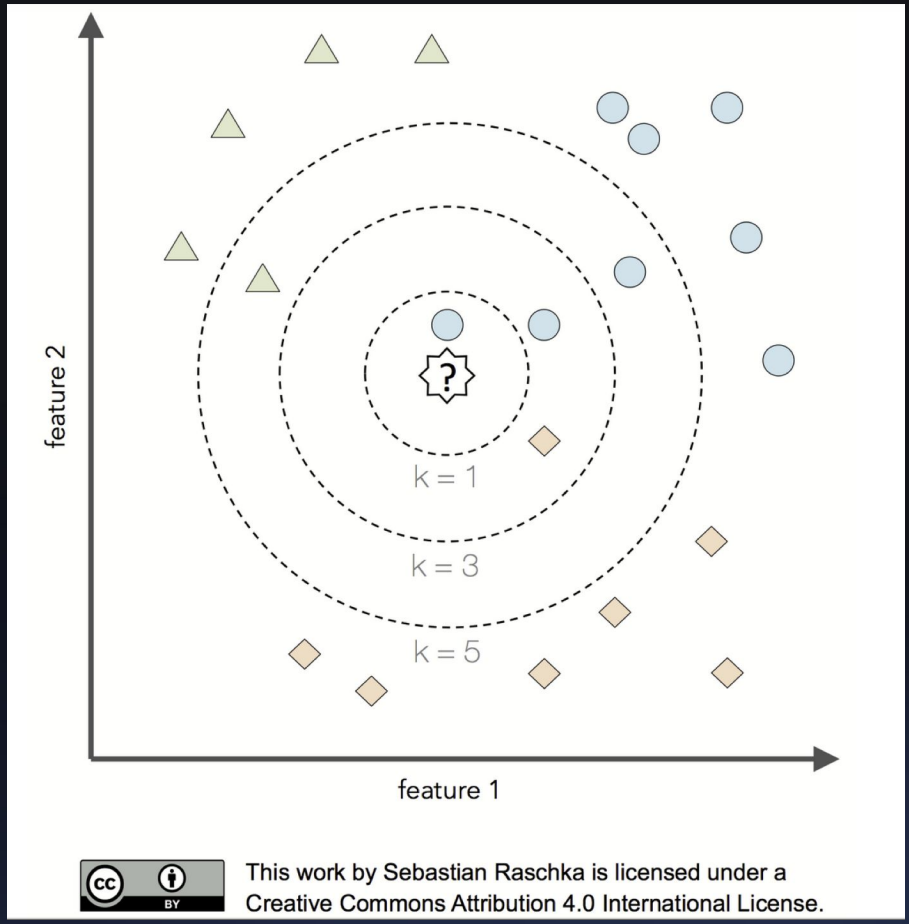
$$d(x,y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

- Feature Scaling



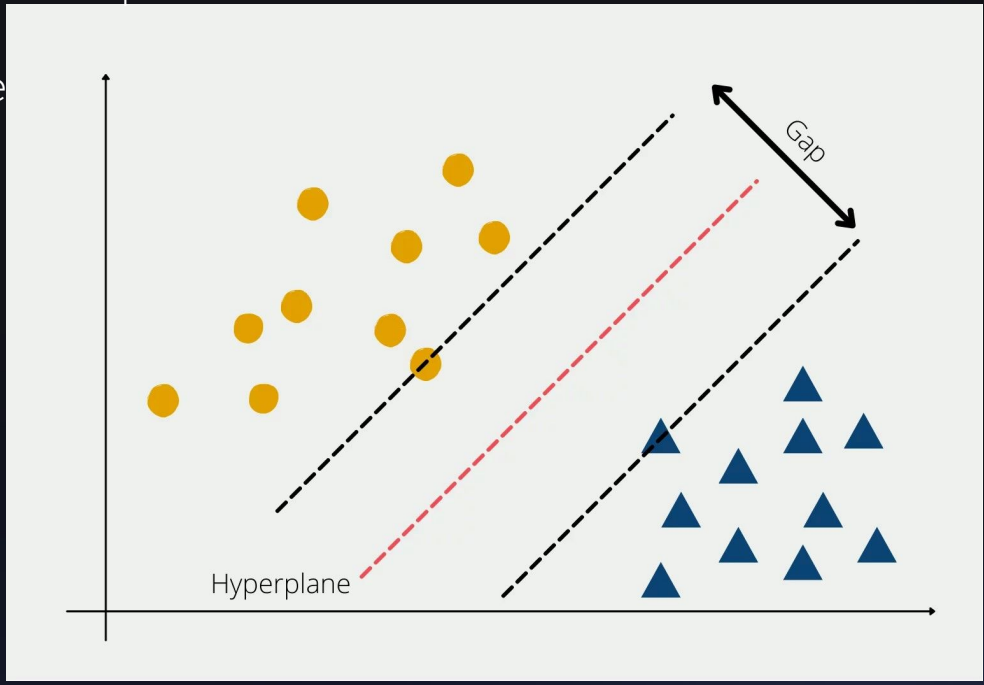
KNN

- **Pros:**
  - Simple and intuitive.
  - No assumption about the data distribution.
  - Adapts as we collect new training data.
- **Cons:**
  - Computationally expensive
  - Sensitive to irrelevant or redundant features
  - Poor performance on imbalanced datasets.
- Parameter Tuning



Support Vector Machines (SVM)

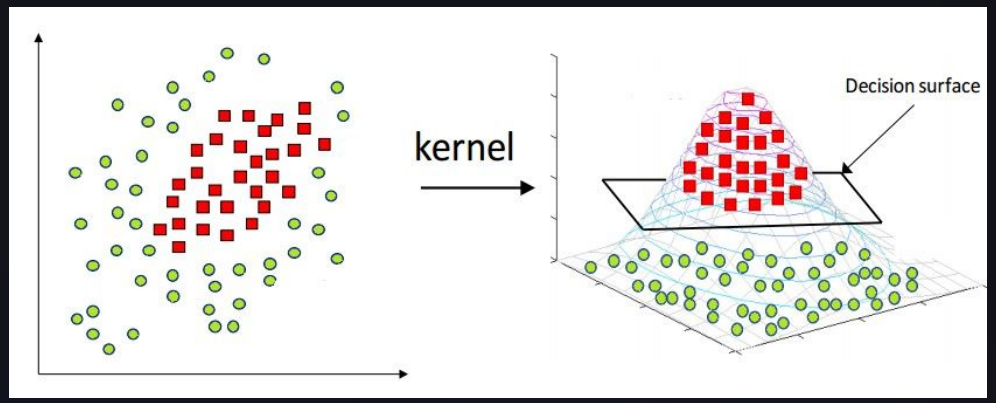
- **Goal:** find the optimal separating hyperplane which maximizes the margin Hyperplane: A decision boundary that separates different classes in the feature space.
- **Margin:** The distance between the hyperplane and the nearest data points.
- Maximizing the margin increases the model's ability to generalize to new data.
- **Advantages**
  - Effectiveness in high-dimensional spaces.
  - Memory efficiency due to the use of a subset of training points.
  - Versatility through different kernel functions.
- **Challenges and Limitations**
  - Choice of kernel and its parameters can be complex.
  - Not suitable for large datasets due to computational inefficiency.
  - Sensitive to noisy data and outliers.



SVM

- **Kernel Trick**

  Purpose: To transform linearly inseparable data to a higher dimension where it is linearly separable.



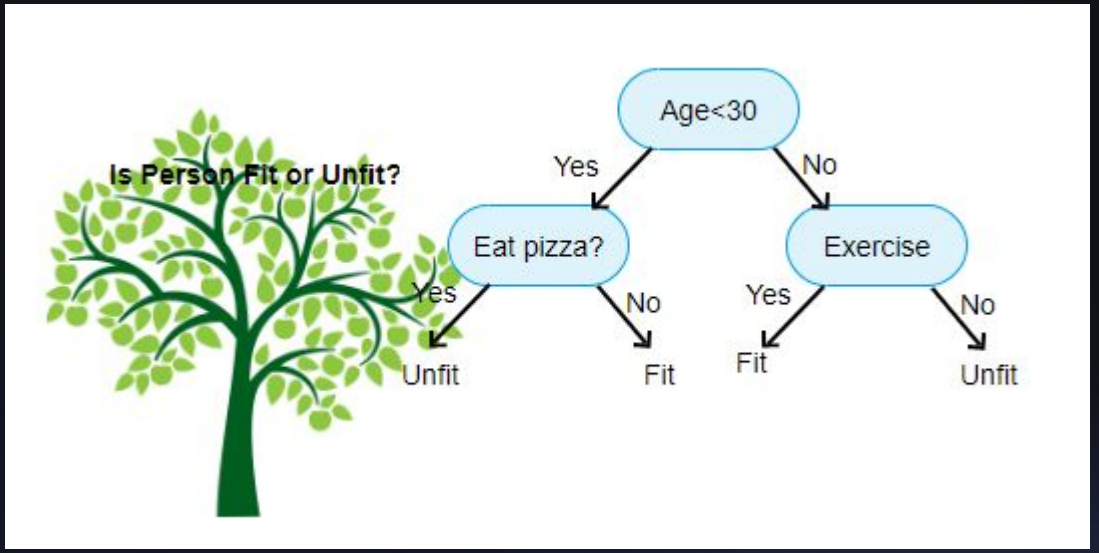
- **Types of Kernels**:
  - Linear,
  - Polynomial,
  - Radial Basis Function
- **SVM Variants**
  - Linear SVM: For linearly separable data.
  - Non-linear SVM: Uses kernel trick for non-linearly separable data.
- **SVM Optimization**
  - Methods like Cross-validation for parameter tuning.
  - Feature scaling for better performance.

Practical session on supervised learning

- Logistic Regression
- KNN
  - how to tune one hyperparameter
- SVMs – Linear, Polynomial, Radial Basis Function
  - Multiple Hyperparameter

Decision Tree

- A tree-like model used for classification and regression tasks in machine learning. Consists of nodes, branches, and leaves.
  - Root Node: From which the tree starts splitting.
  - Decision Nodes: Nodes that make a decision and split into further branches.
  - Leaf Nodes: Terminal nodes that represent a classification or decision.
- Splits the dataset into subsets based on an attribute value test.
- Types of Decision Trees:
  - Classification Trees.
  - Regression Trees.
- Splitting Criteria:
  - Gini Impurity: Measures the frequency at which any element of the dataset will be mislabeled.



- Entropy and Information Gain: Used in ID3, C4.5, and other algorithms to measure the best splitting point.

Random forest

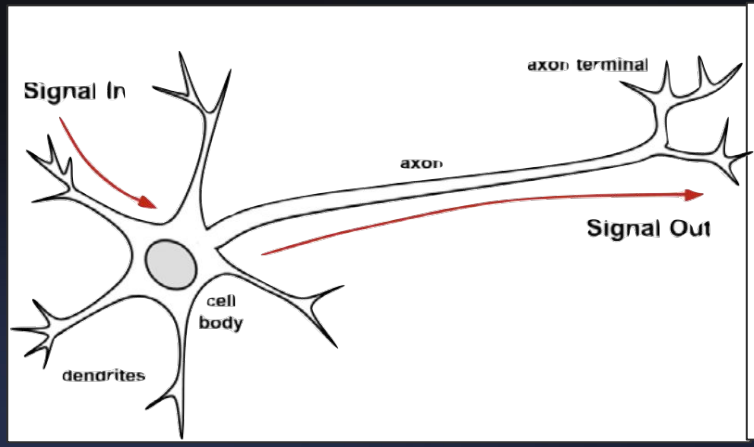
Random Forest is a popular and versatile machine learning algorithm that belongs to the family of ensemble methods.

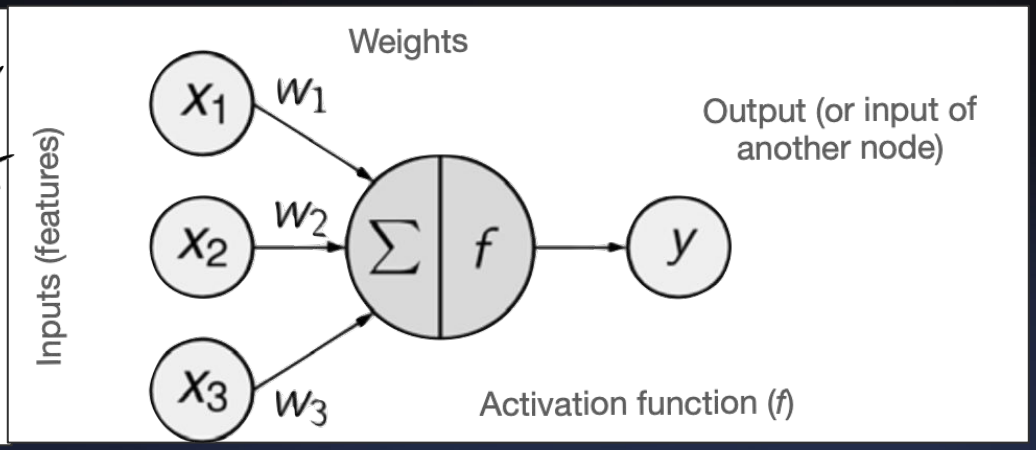
- **Ensemble of Decision Trees**
- **Bagging (Bootstrap Aggregating)**
- **Feature Randomness**
- **Handling Different Types of Data**
- **Overfitting:** One of the key advantages of Random Forest is its robustness to overfitting, especially compared to single decision trees.

  Variable Importance: An intrinsic benefit of Random Forest is its ability to rank the importance of variables in a regression or classification task.
- **Limitations:**
  - Interpretability
  - Computationally Intensive
  - Memory Usage

Artificial Neural Network (ANN)

- Artificial Neural Network (ANN) models the relationship between a set of input signals and an output signal using a model derived from our understanding of how a biological brain responds to stimuli from sensory inputs
- Black Boxes: because the underlying models are based on complex mathematical systems and the results are difficult to interpret
- Supervised, semi-supervised, unsupervised, self-supervised, reinforcement learning





Artificial Neural Network (ANN)

An ANN is generally constituted by;

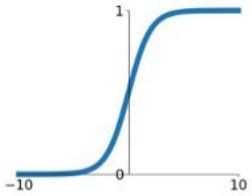
- An activation function, which transforms a neuron's net input signal into a single output signal to be broadcasted further in the network
- A network topology, which describes the number of neurons in the model as well as the number of layers and manner in which they are connected
- The training algorithm that specifies how connection weights are set in order to inhibit or excite neurons in proportion to the input signal.

Firing a neuron: activation functions

Activation Functions

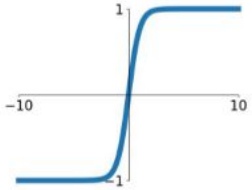
Sigmoid

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



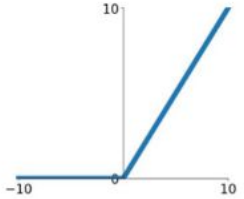
tanh

$$\tanh(x)$$



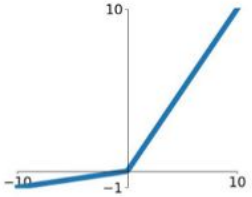
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

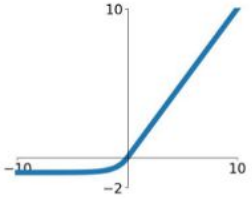


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

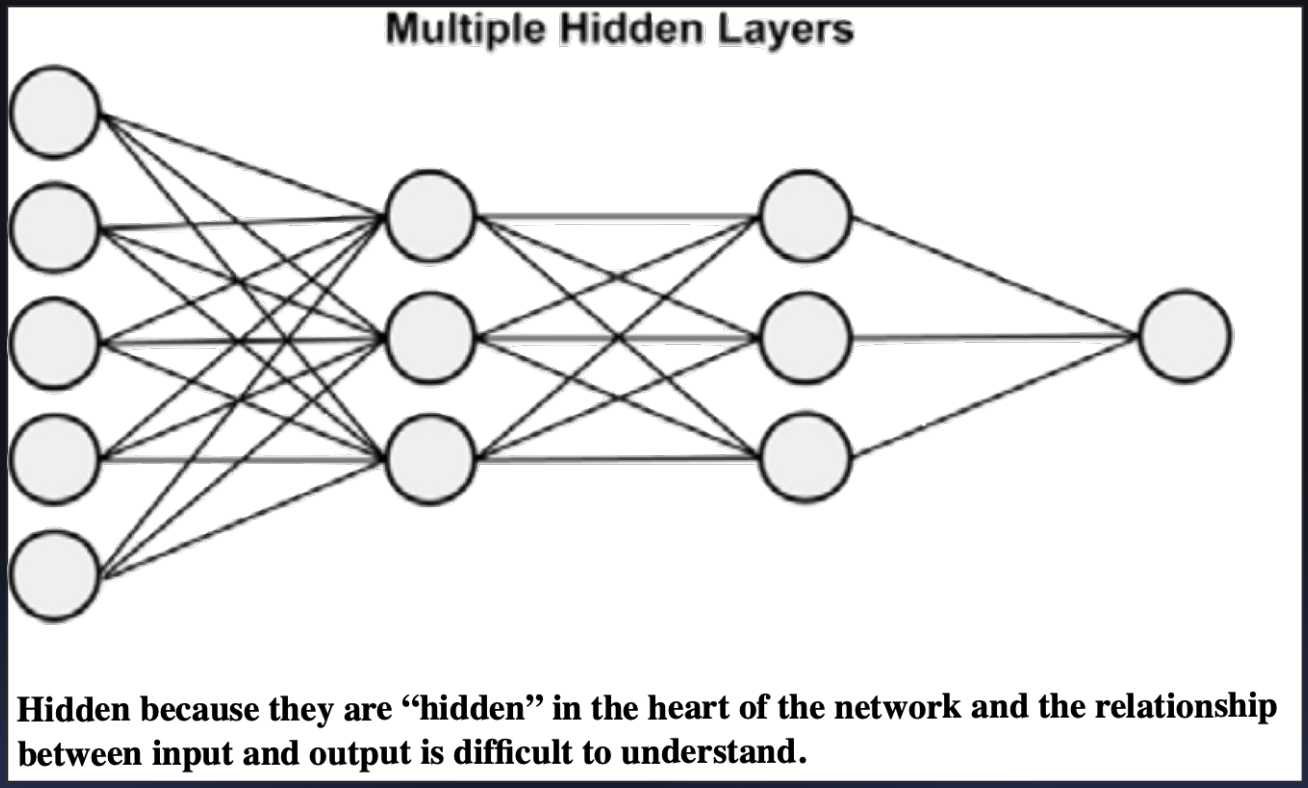
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Network topology

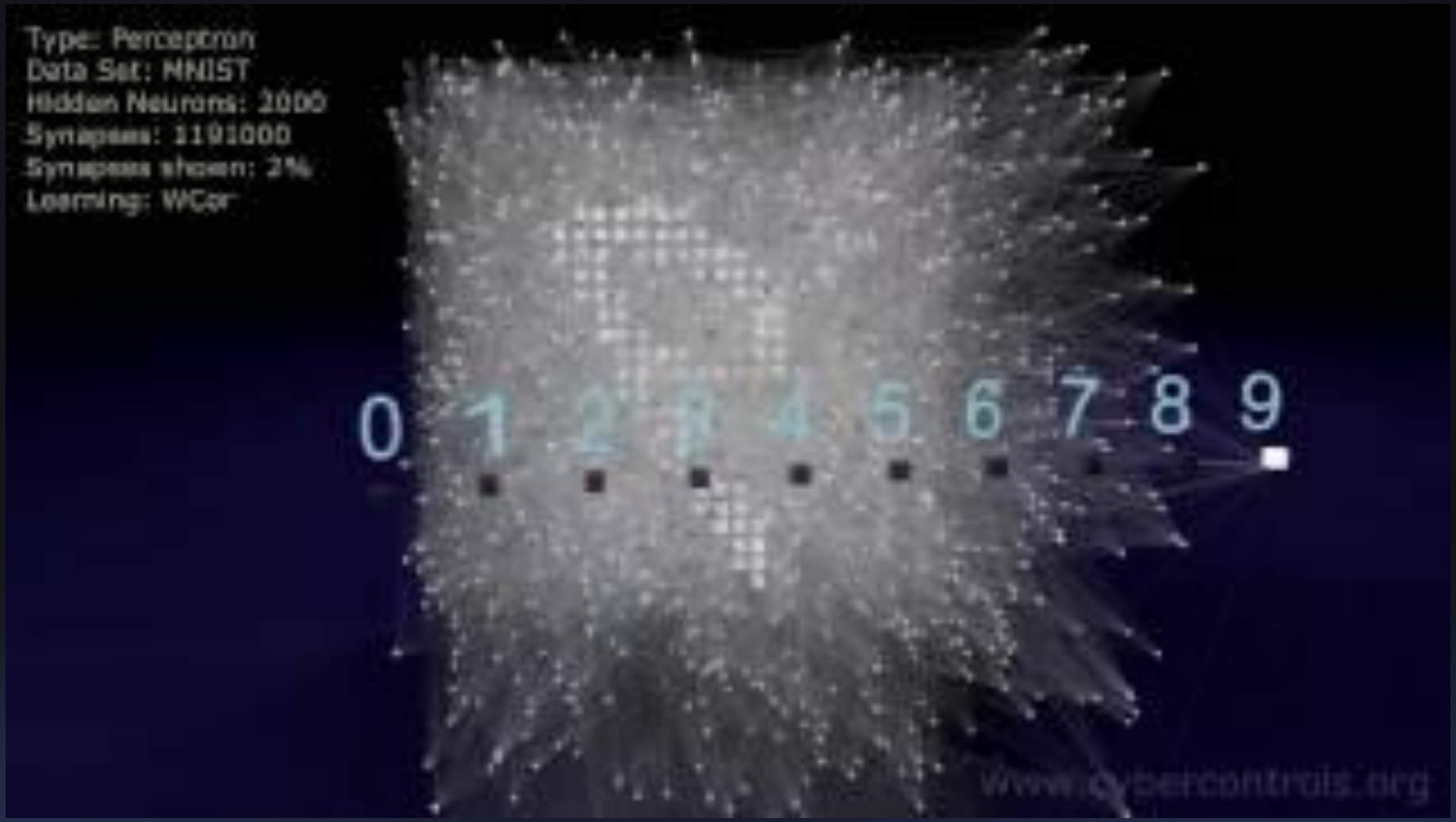
Multiple Hidden layers = Deep Learning



3. **The training algorithm:** How Does A Neural Network Learns?

- Adjusting the *weights* that connect nodes thanks to backpropagation
  - Forward fase
  - Backward phase (a loss functions penalize wrong answers)
- Iterates using epochs of learning adjusting weights according to
  Gradient Descent Optimization.
  - The algorithm will attempt to change the weights that result in the greatest reduction in error by an amount known as the learning rate.

https://playground.tensorflow.org/


Type: Perceptron
Data Set: MNIST
Hidden Neurons: 2000
Synapses: 1191000
Synapses shown: 2%
Learning: WCor
0 1 2 3 4 5 6 7 8 9
www.cybercontrols.org

Question & Answering

Supervised vs Unsupervised Learning

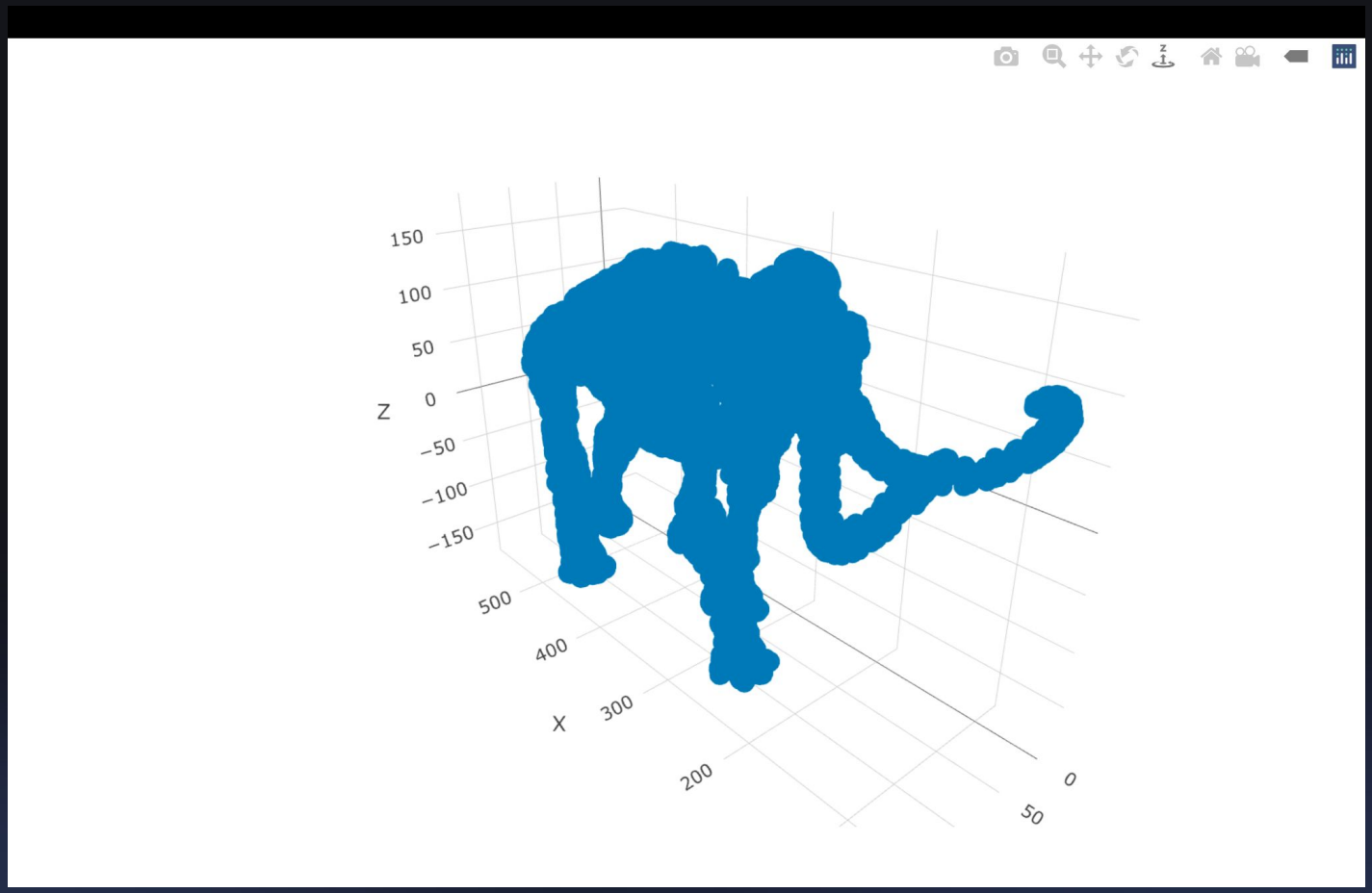
- **Unsupervised learning**
  - Input features X_i = {X1, X2, ..., X_N}, where X_i represents the features of the i-th data point
- **Output Labels :** none
- **Objective:** Discover hidden patterns, structure, or relationships in the data, often by clustering similar data points or reducing the dimensionality of the data.

PCA

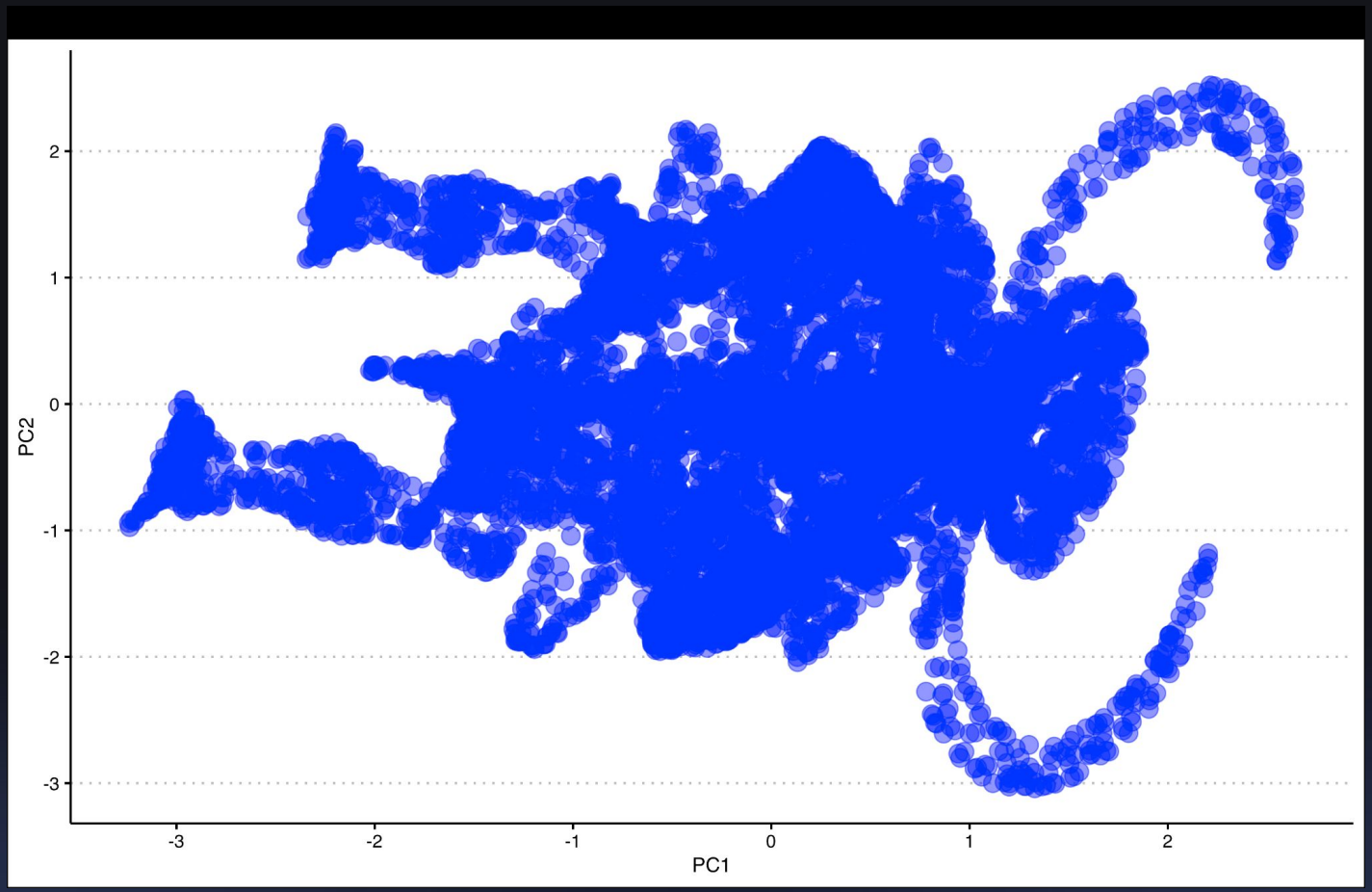
PCA, or Principal Component Analysis, is a statistical technique used in the field of data analysis and machine learning. Here are some key points about PCA:

- **Dimensionality Reduction**
- **Principal Components**
- **Steps**
  - Standardization
  - Covariance Matrix
  - Eigenvalues and Eigenvectors
  - Select top ‘k’ eigenvalues
- **Preprocessing in Machine Learning**
- PCA is powerful for pattern identification in complex datasets, but it's important to remember that it's a linear technique and may not work well with non-linear relationships in the data.

Visualizing the results of PCA



Visualizing the results of PCA



Uniform Manifold Approximation and Projection (UMAP)

- Efficient Non-linear Dimensionality Reduction
- Manifold Learning:
  - UMAP operates on the principle of manifold learning, meaning it assumes the data is distributed on a manifold within the high-dimensional space and seeks to understand the structure of this manifold.
- Preserving Data Structure:
- Flexibility and Versatility:
  - UMAP can be used for a variety of data types and is compatible with different metrics (distance measures), making it versatile for many types of analysis.
- Scalability and Performance:
  - UMAP often demonstrates superior performance and speed compared to other dimensionality reduction methods, especially on large datasets.

UMAP Example

Understanding UMAP

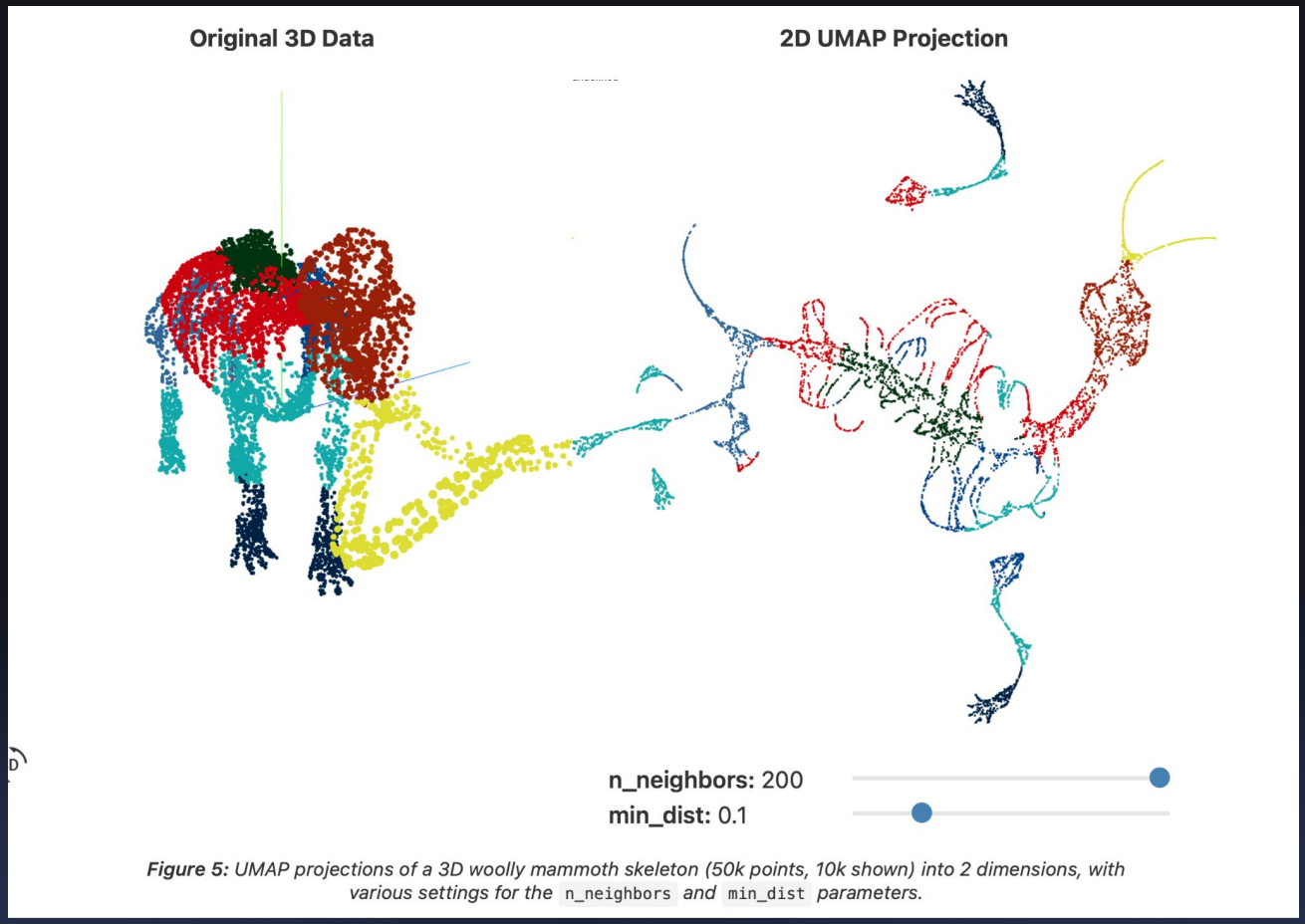


Figure 5: *UMAP projections of a 3D woolly mammoth skeleton (50k points, 10k shown) into 2 dimensions, with various settings for the* `n_neighbors` *and* `min_dist` *parameters.*

Practical session on Dimensionality reduction

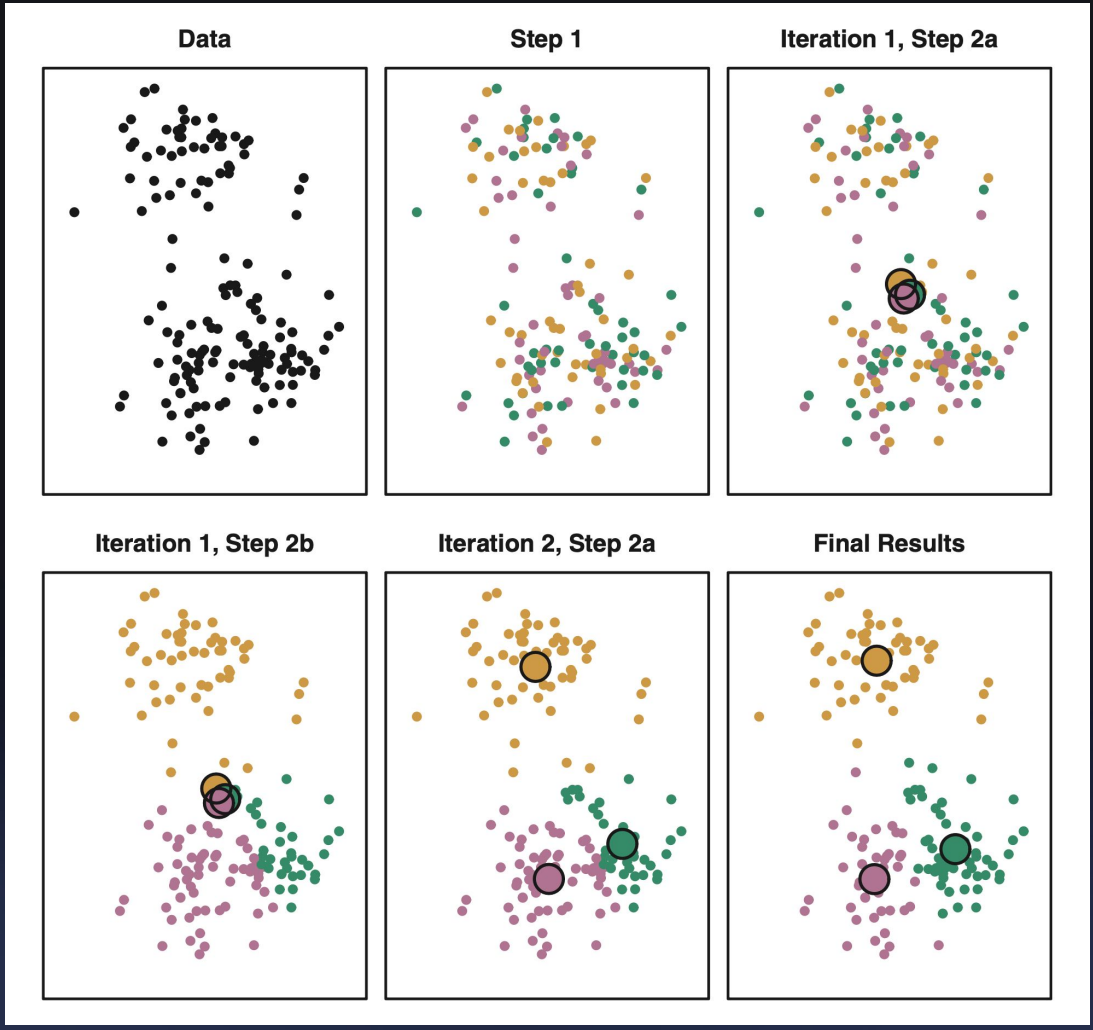
comparing PCA and UMAP

K-means

K-means is an unsupervised learning algorithm, meaning it is used to find patterns or groupings in data without reference to known, labeled outcomes.

- **Cluster Formation:** The algorithm partitions the data into K distinct, non-overlapping subgroups (clusters) based on similarities, where K is a predefined number of clusters.
- **Centroids:** Each cluster is characterized by its centroid, which is the mean of the points in the cluster. The algorithm aims to minimize the variance within each cluster.
- **Iterative Process**
- **Applications:** K-means is widely used in market segmentation, document clustering, image segmentation, and in many other fields requiring data partitioning into distinct groups.
- **Limitations:** The algorithm assumes spherical clusters and is sensitive to the initial choice of centroids.

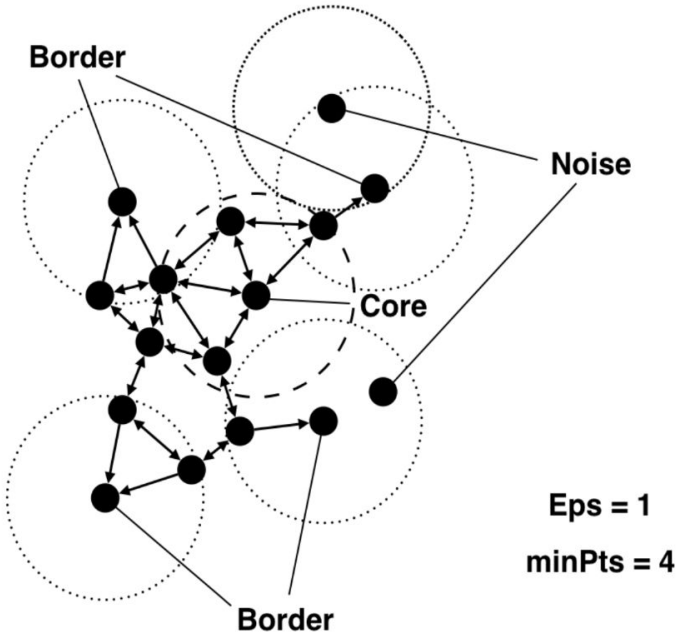
K-Mean: an iterative partition algorithm

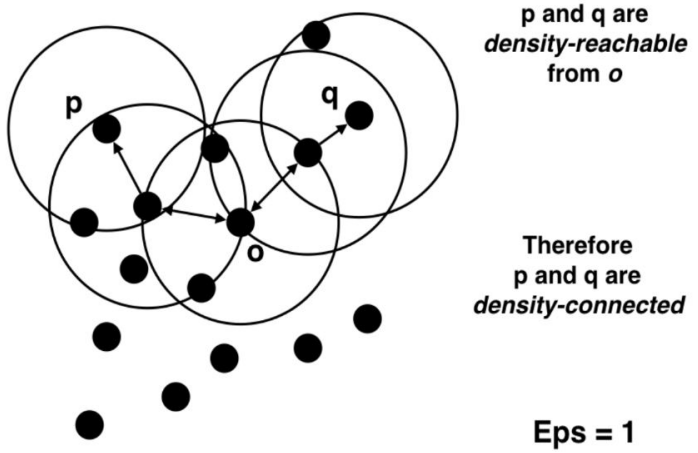


Density-based spatial clustering of applications with noise (DBSCAN)

- Density-Based Clustering
- No Need to Specify Number of Clusters
- Handles Noise
- Parameters: Epsilon, MinPts
- Points
  - Core Point
  - Border Point
  - Noise Point
- The algorithm follows these steps:
  - Find Core Points
  - Expand Clusters
  - Assign Border Points
  - Mark Noise Points

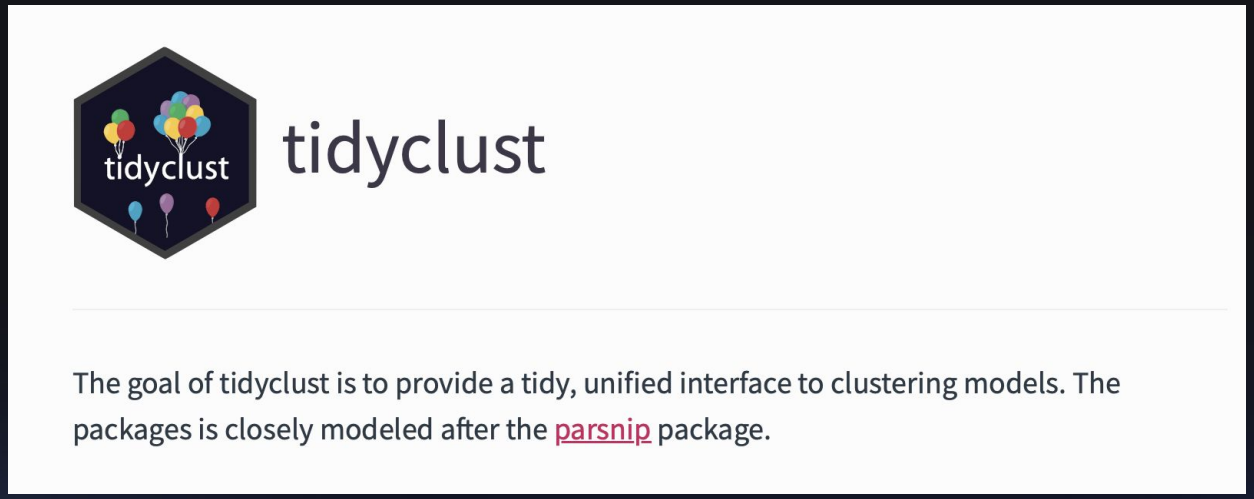
DBSCAN


Border
Noise
Core
Border
Eps = 1
minPts = 4


p
q
o
p and q are
density-reachable
from o
Therefore
p and q are
density-connected
Eps = 1
minPts = 4

Practical session on Clustering

Tidyclust metapackage



Practical session: Going Beyond

Try yourself

Assignment:

Take the *Juniperus* dataset and fit a Random Forest model using Tidymodels.


Thanks for you attention
Thank you for your attention
Thanks for your attention