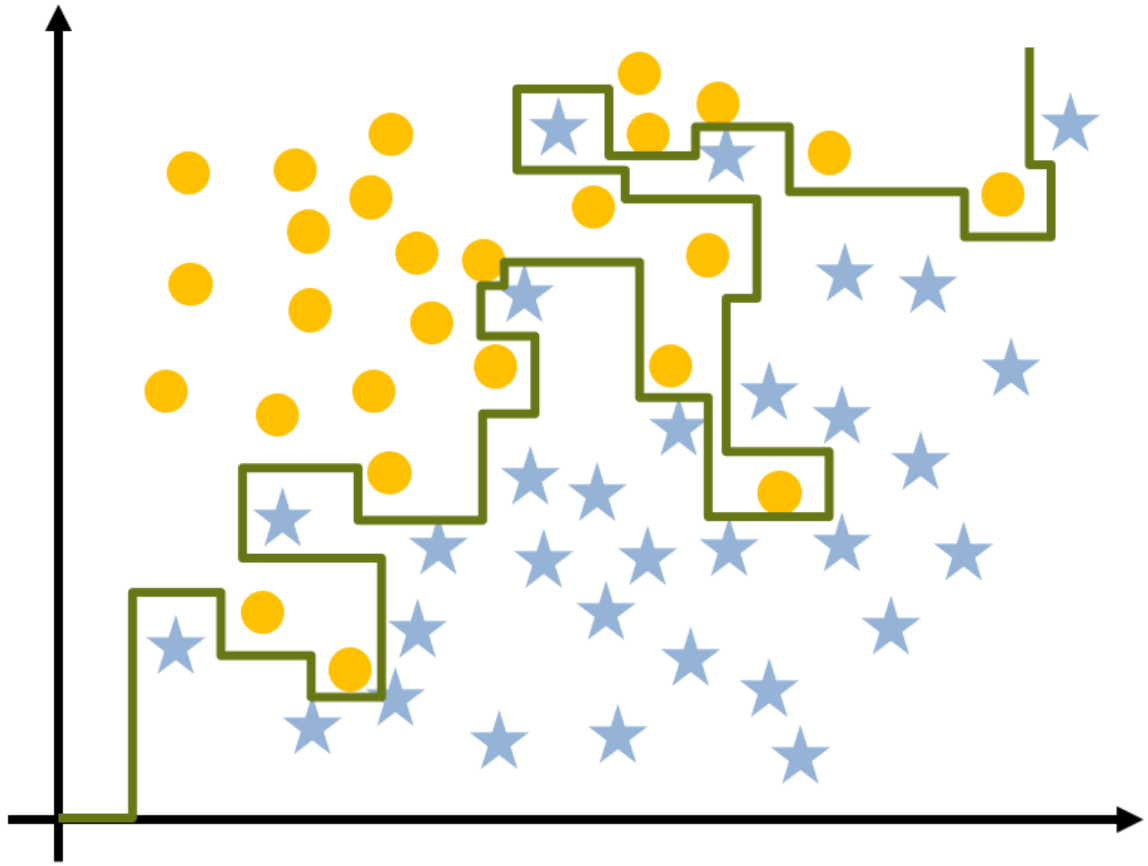
APPLIED SUPERVISED LEARNING (2)

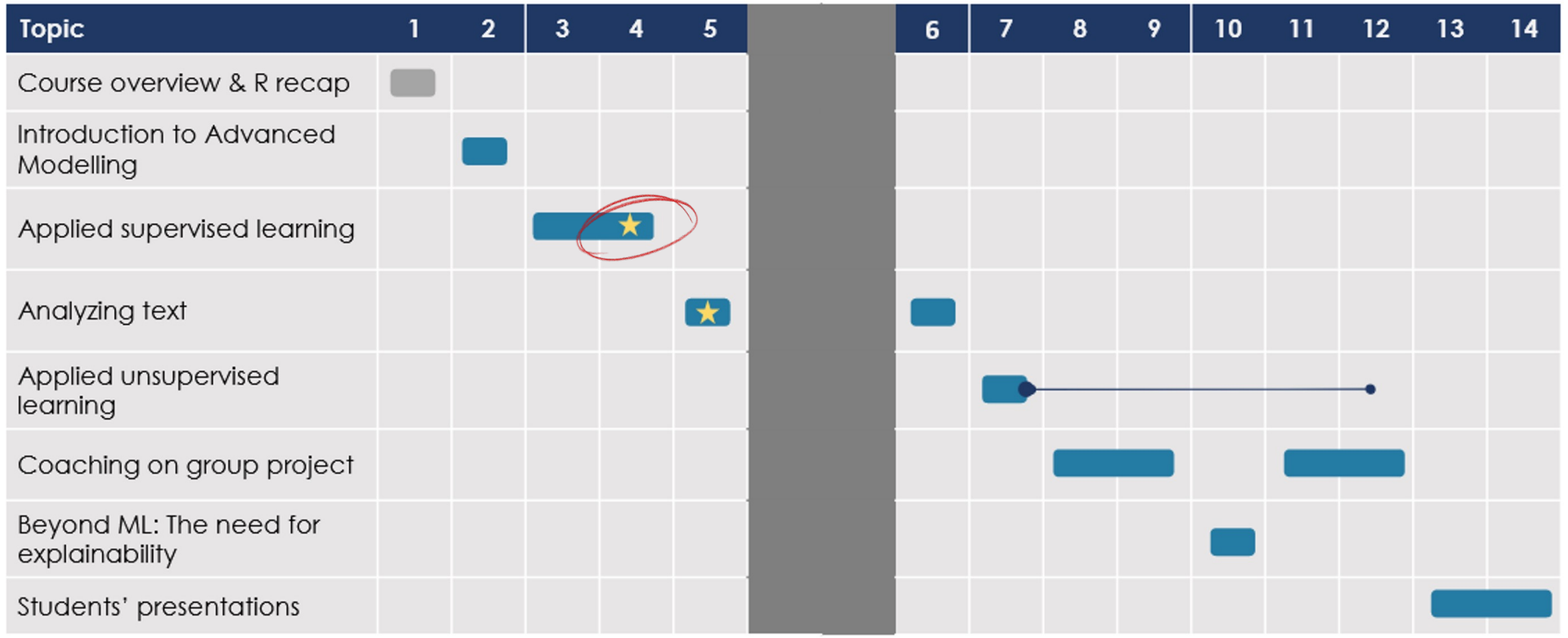
Branka Hadji Misheva &
Christoph Zangger



Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

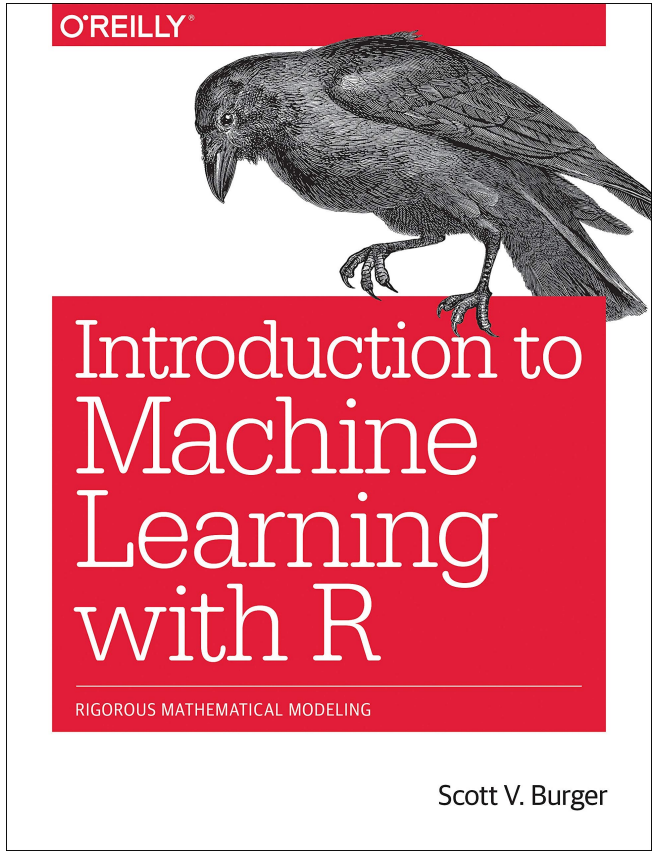
SBD3 – Course Overview

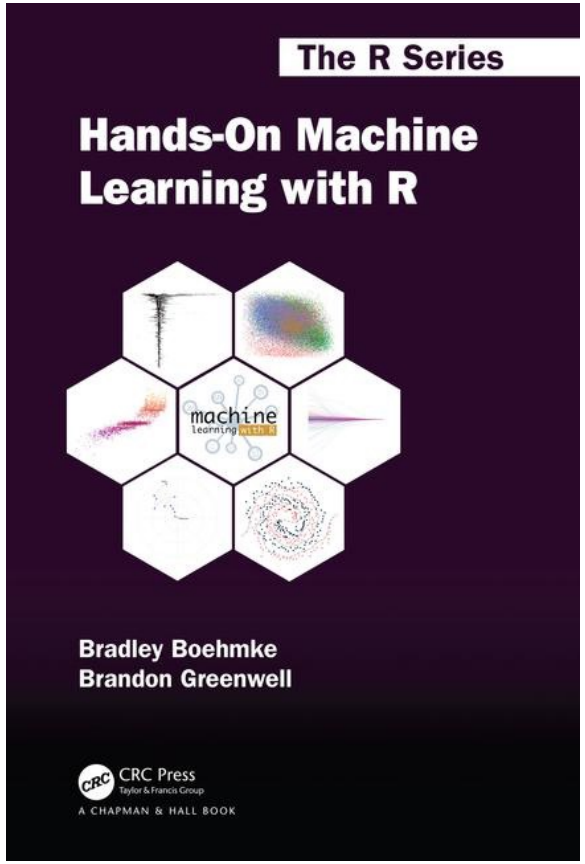
● - Homework handout
★ - Group project handout

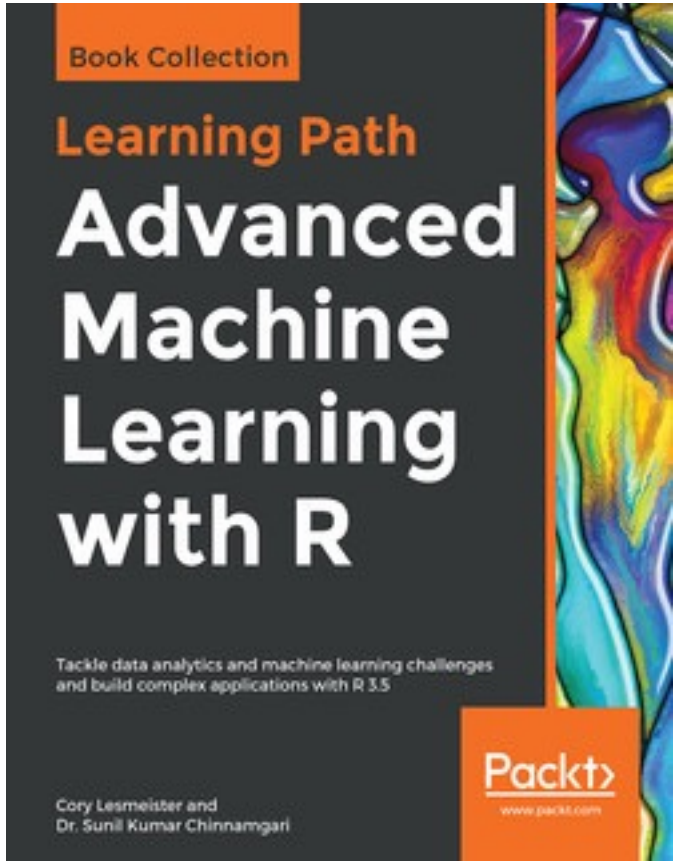


Notes:
- Easter break (no classes): 26th of March and 2nd of April

Literature


O'REILLY
Introduction to Machine Learning with R
RIGOROUS MATHEMATICAL MODELING
Scott V. Burger

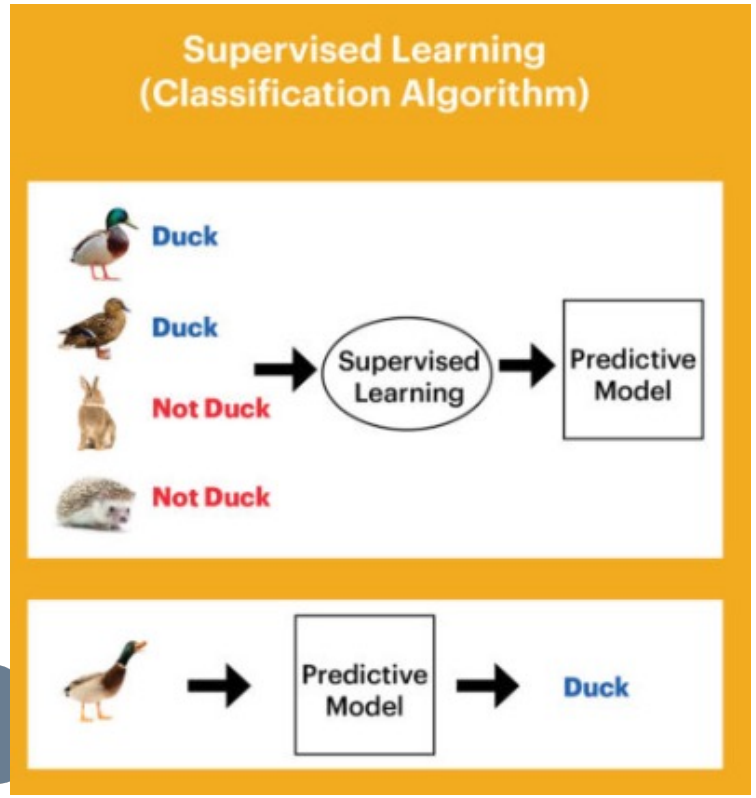

The R Series
Hands-On Machine Learning with R
machine learning with R
Bradley Boehmke
Brandon Greenwell
CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

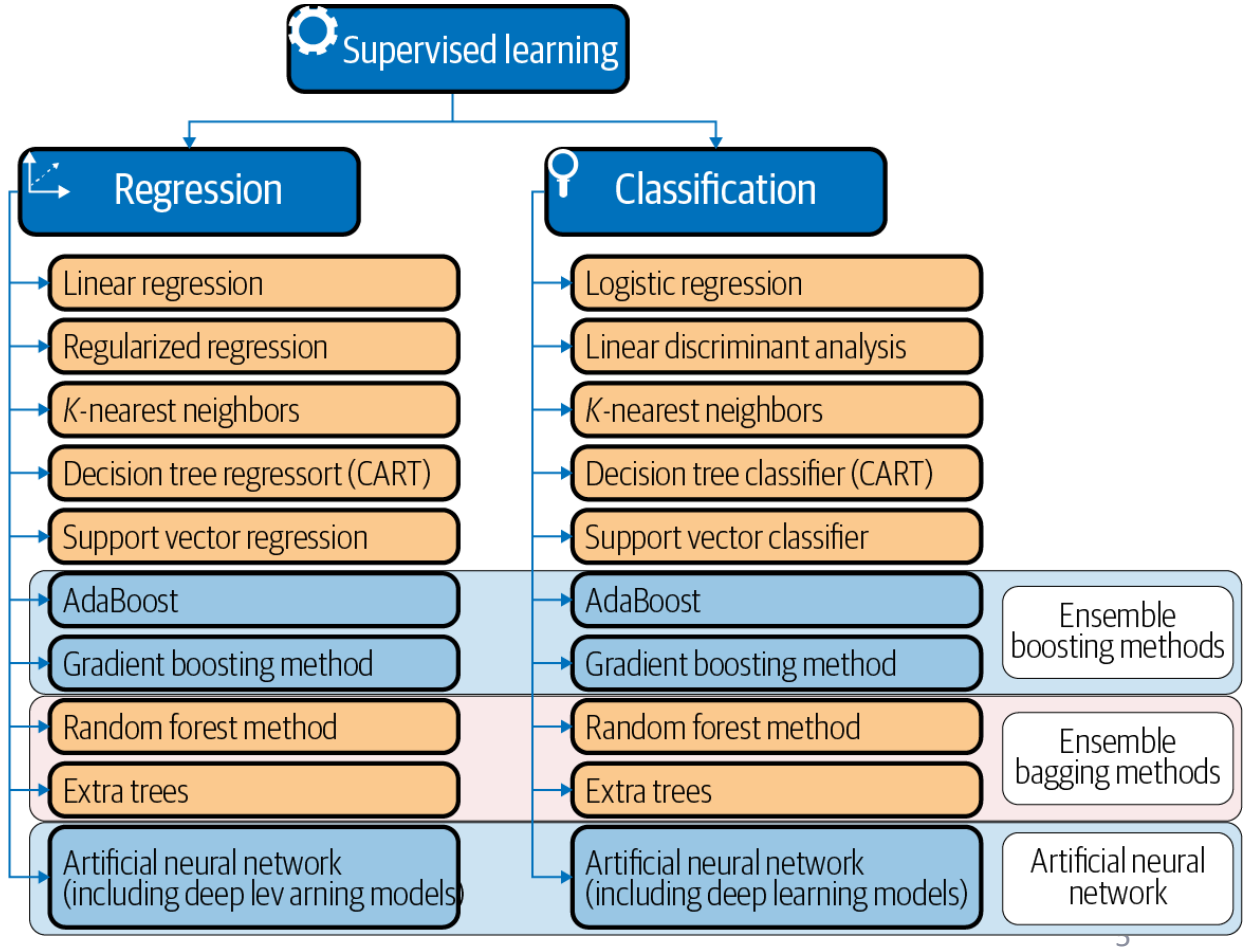

Book Collection
Learning Path
Advanced Machine Learning with R
Tackle data analytics and machine learning challenges and build complex applications with R 3.5
Cory Lesmeister and
Dr. Sunil Kumar Chinnamgari
Packt

Recap

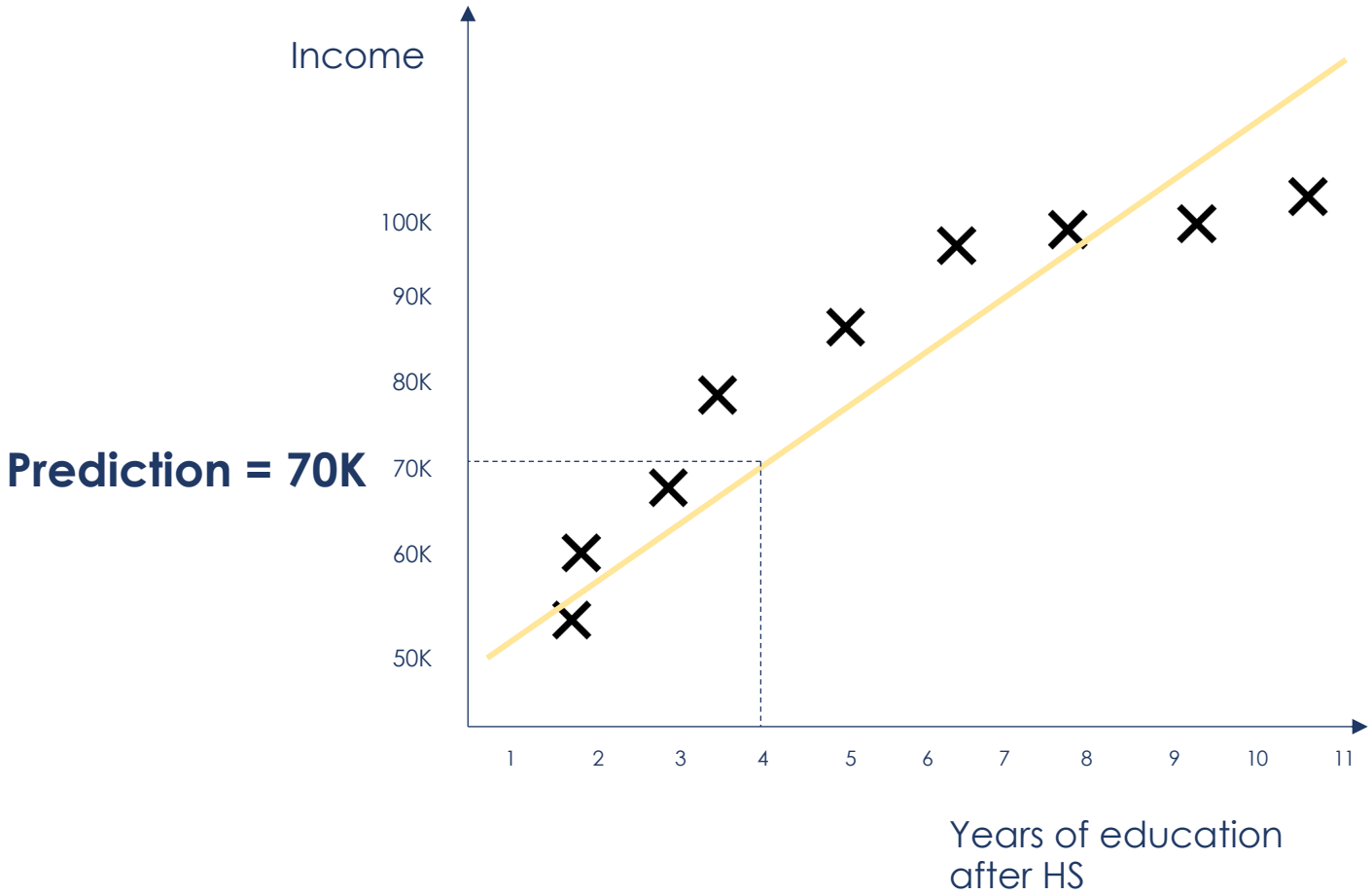
Supervised Learning

For problems where the available data consists of **labelled examples**





Linear Regression

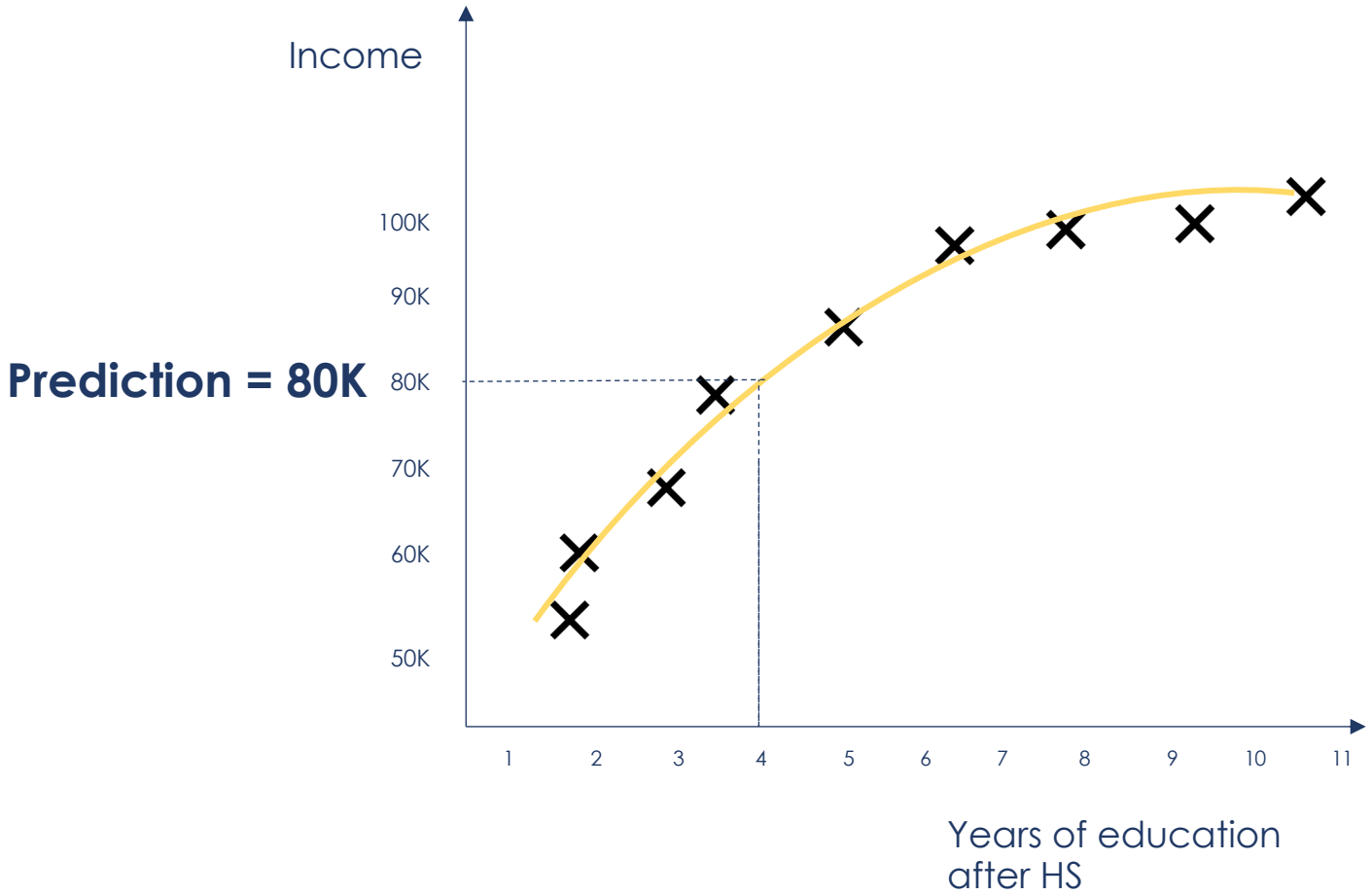


Prediction = 70K

New observation

Years of education = 4

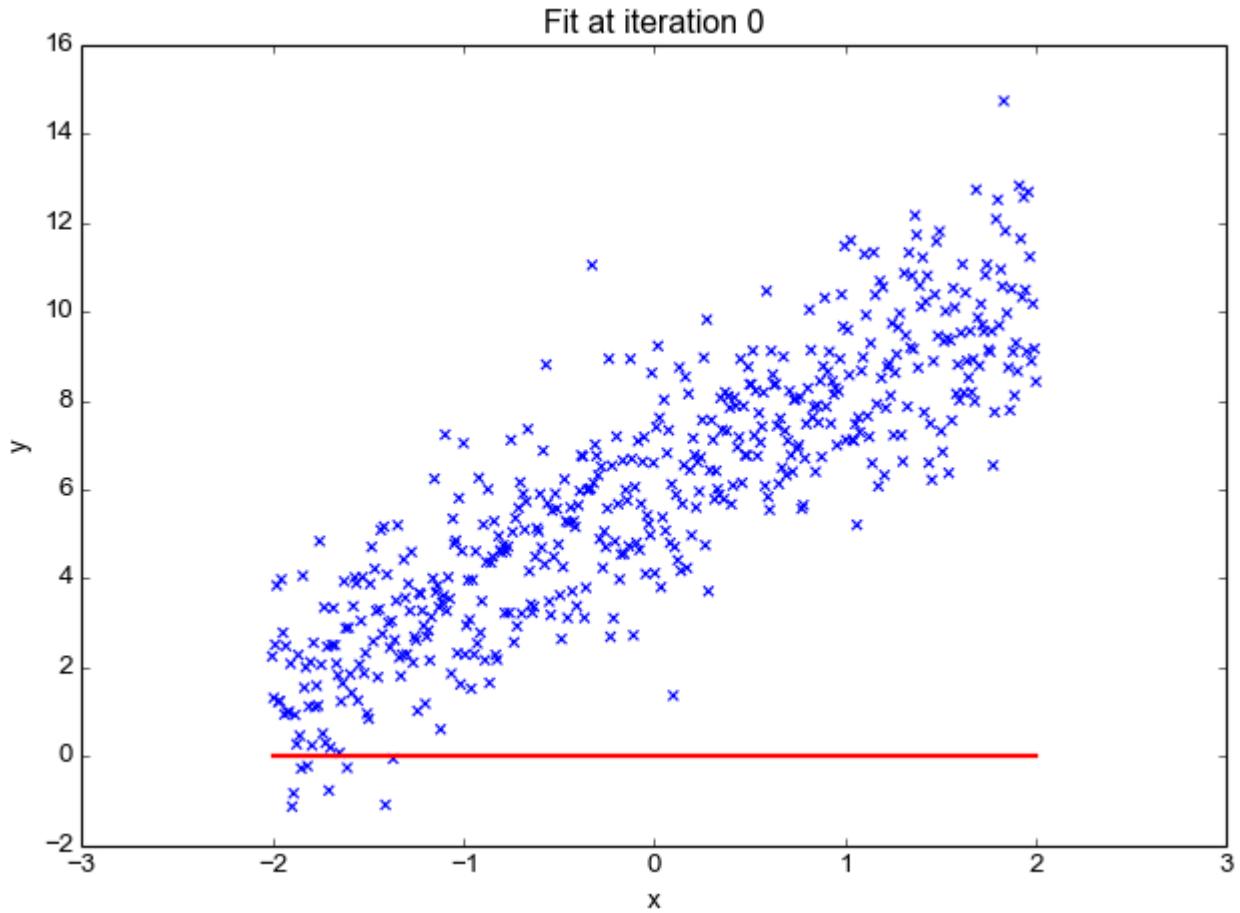
Linear Regression



New observation

Years of education = 4

How do we fit **the line**?



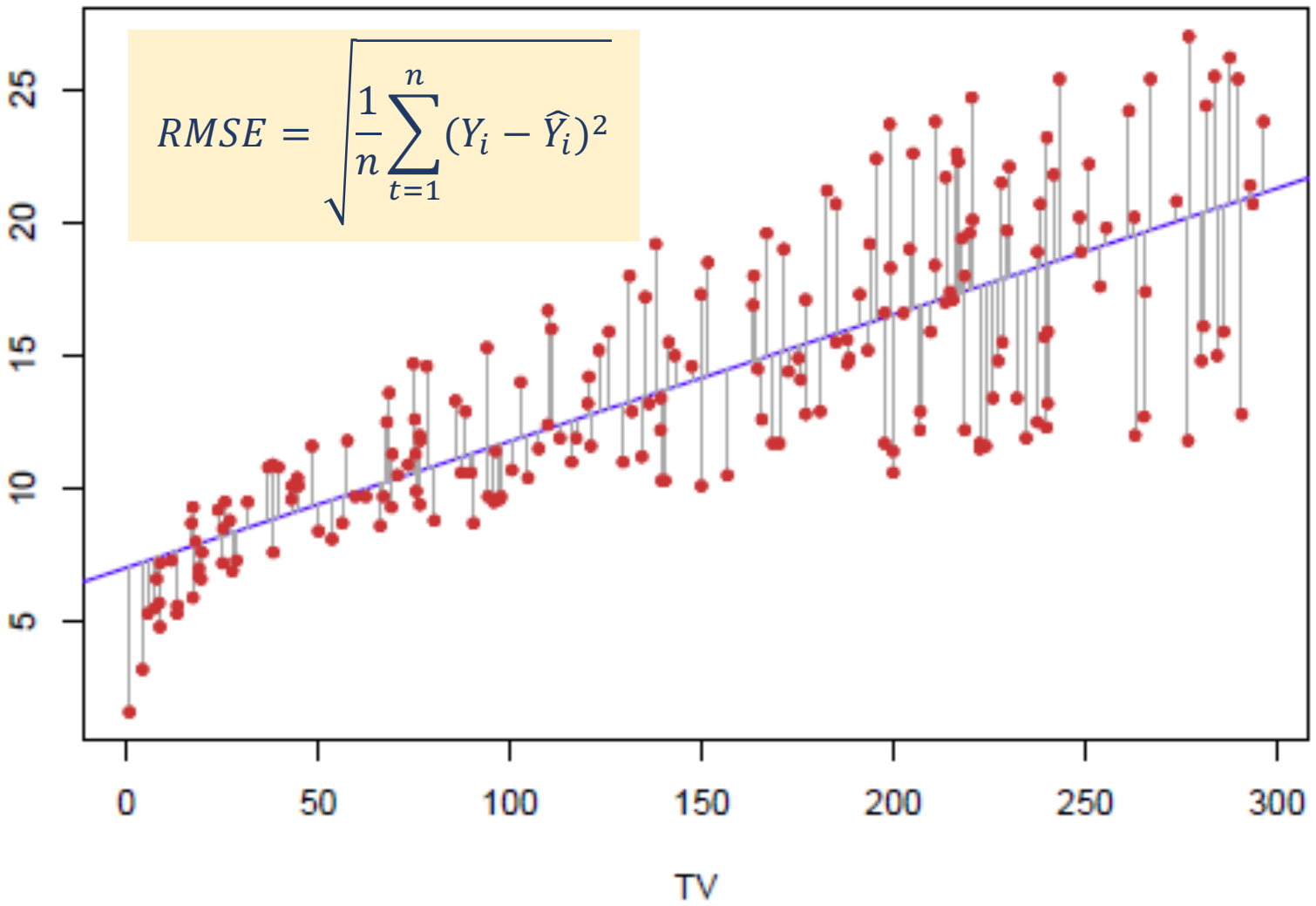
The Process **Flow**

With the model we are able to make predictions!

Lets imagine we want to predict the movement in the price of **Microsoft**

Date	Predicted Price
17.10.22	235.820007
18.10.22	243.240005
19.10.22	237.039993
20.10.22	235.770004
21.10.22	234.740005
24.10.22	243.759995
25.10.22	247.259995
26.10.22	231.169998
27.10.22	231.039993
28.10.22	226.240005
31.10.22	233.759995
01.11.22	234.600006
02.11.22	229.460007
03.11.22	220.089996
04.11.22	217.550003
07.11.22	221.990005
08.11.22	228.699997
09.11.22	227.369995
10.11.22	235.429993
11.11.22	242.990005
14.11.22	241.990005

How do I measure how good my model is?



Measuring **performance**

The **classification** problem

- The linear regression model assumes that the response variable y is **quantitative (metrical)**
  - In many situations, the response variable is instead qualitative (categorical)

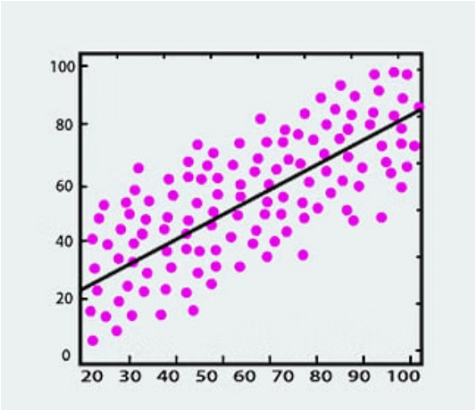
- Qualitative variables take values in an unordered set $C = \{"cat_1", "cat_2", \dots, "cat_n"\}$, such as:
  - Default ("yes", "no")
  - Churn ("yes", "no")
  - Risk category ("Risk-prone", "Risk-neutral", "Risk-averse")
  - ….

Metric vs Categorical

Metric data

- Describe a **quantity**
- An ordering is defined
- A distance is defined

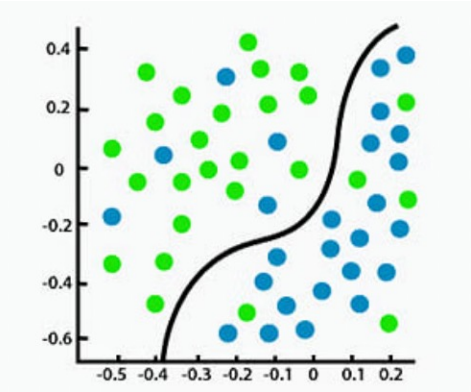
Eg. Wages, returns, profits, prices …



Categorical data

- Describe **membership** categories
- It is not meaningful to apply an ordering
- It is not meaningful to compute distances

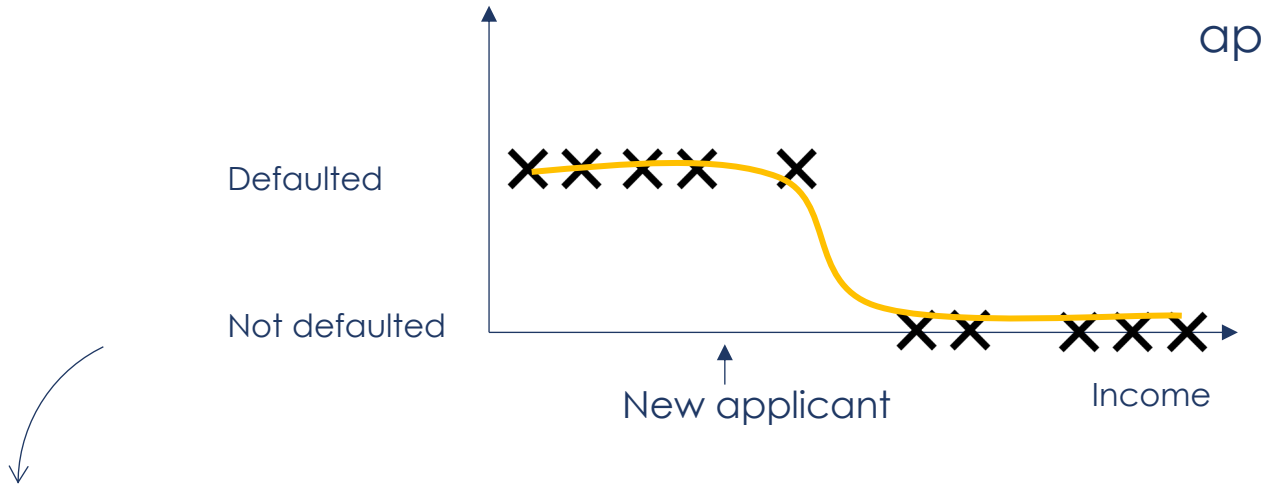
Eg. Default vs non-default; Churn vs not churn …



Can be a **discrete** response!

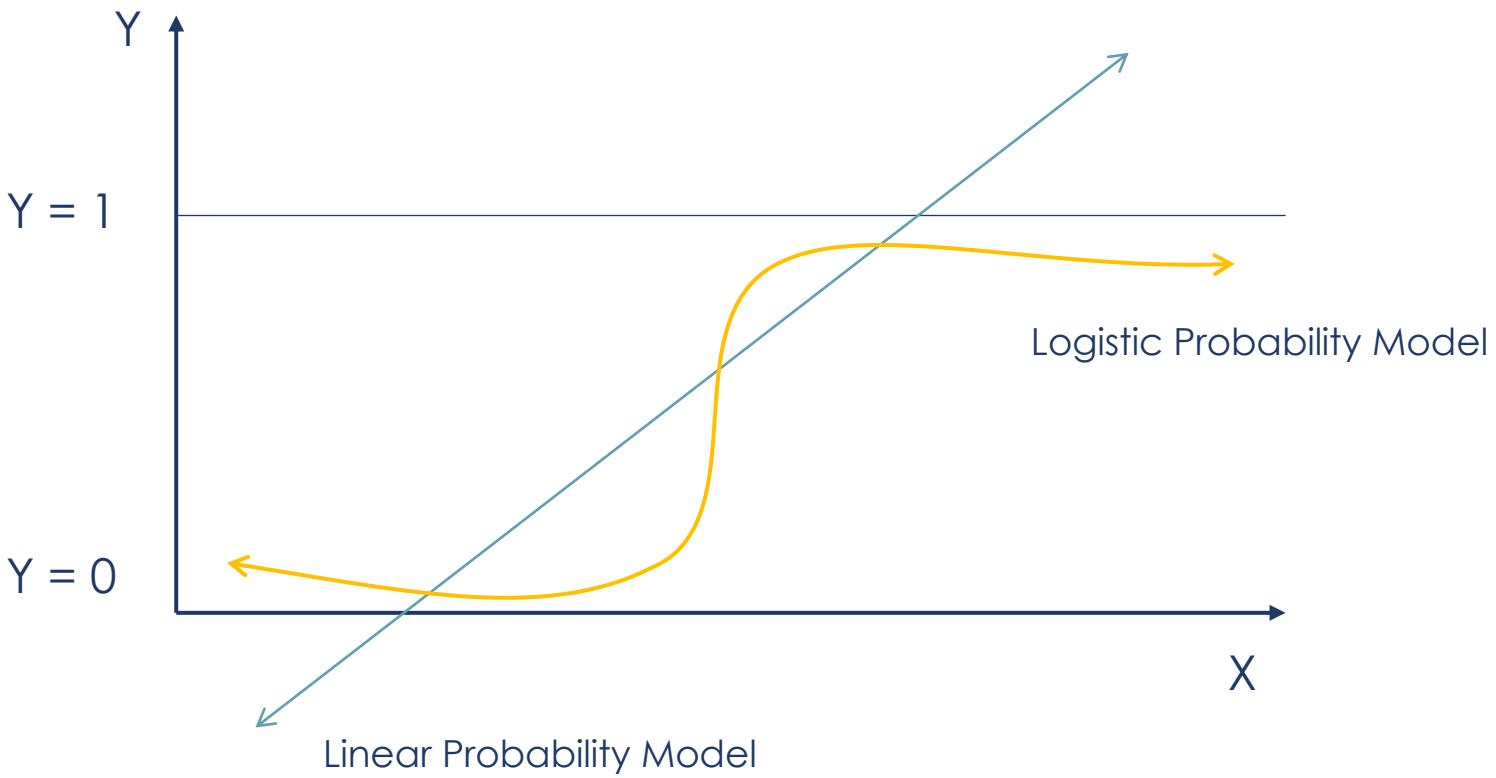
Classification task

Should a loan application get approved?



Estimate a **probability** that an instance belongs to a particular **class**

Logistic regression

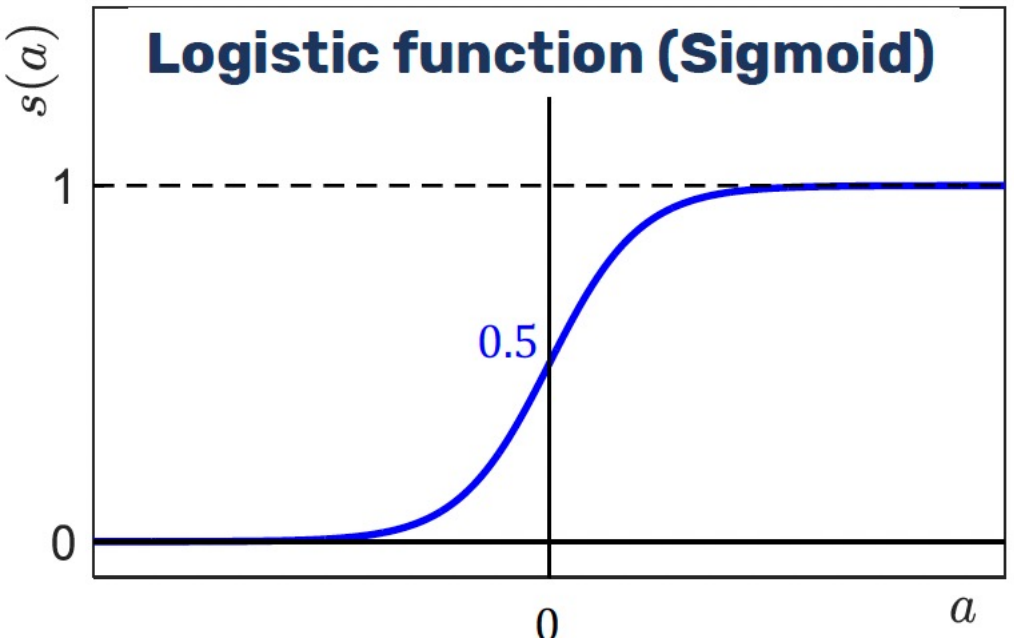


Logistic regression

- Purpose: Estimate the probability that a set of input regressors x belong to one of two classes $y \in 0, 1$
- Formally:

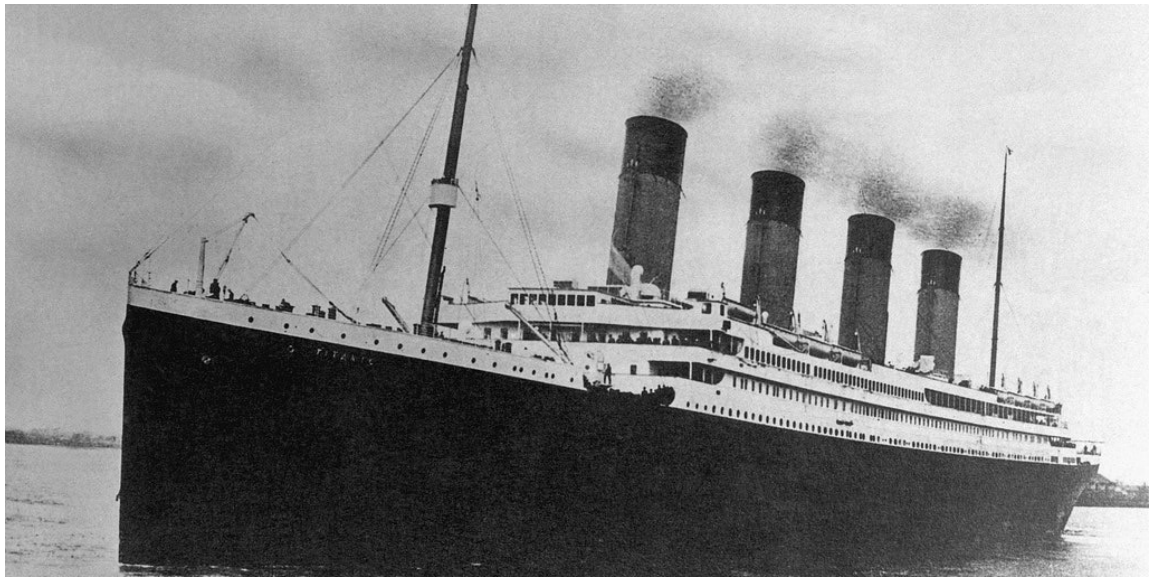
$$\ln\left(\frac{p}{1-p}\right) = \boldsymbol{\alpha} + \boldsymbol{\beta}\mathrm{x} + \boldsymbol{\varepsilon}$$

- Logistic regression models the log-odds (logit) of the probability of the event occurring.
- This transformation maps the probability from the (0, 1) range to the entire real number line.



What about **non-linear relationships**?

Let's think of the **Titanic**!

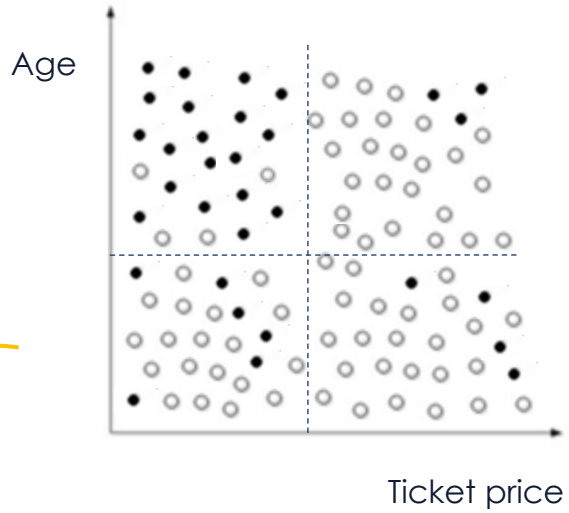


Could have Jack **survived**? Was there space on the door?

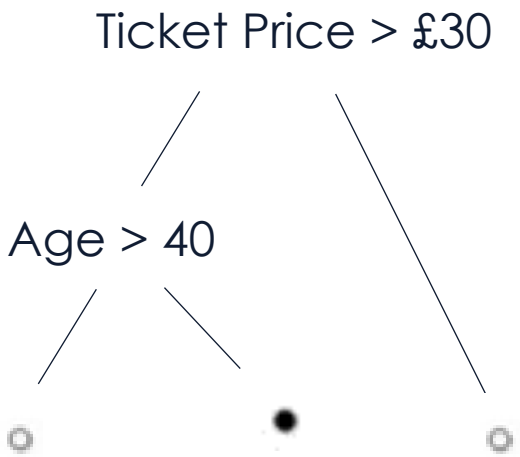
What about **non-linear relationships**?

● - Died ☹

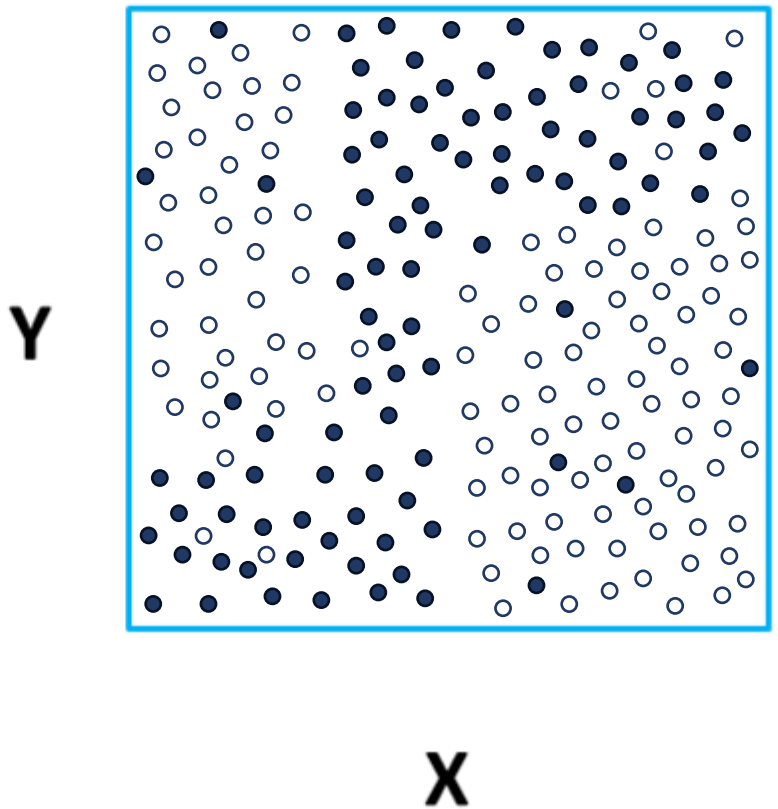
○ - Survived ☺



Scatterplot of the people on board based on the ticket price they paid and their age.

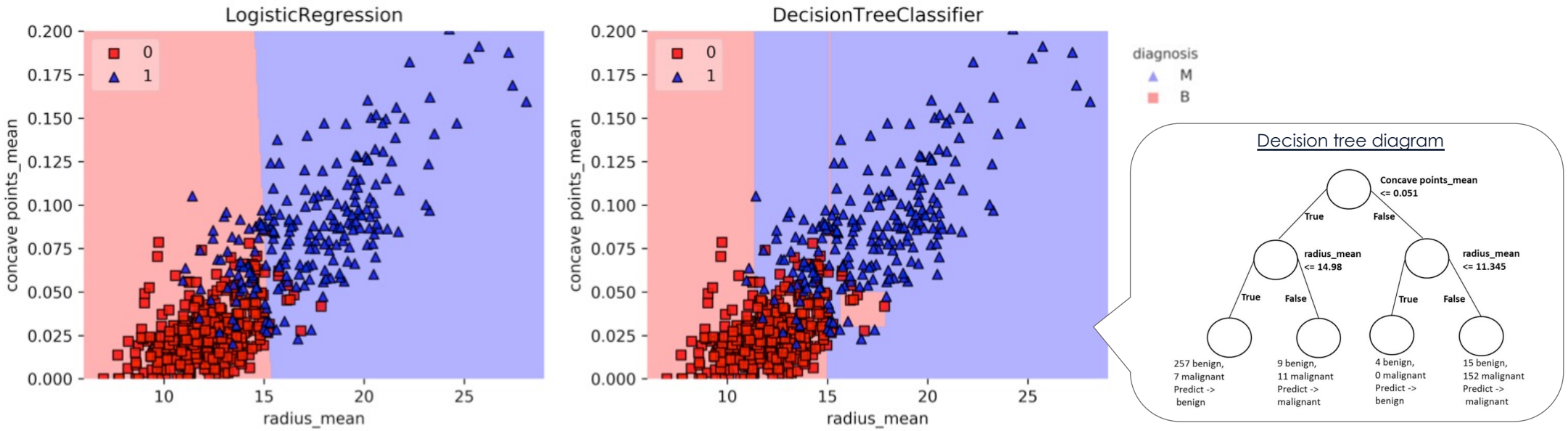


Decision trees



Logistic regression vs. decision tree

Cancer prediction: Benign vs Malignant Tumors



A classification tree produces rectangular decision regions in the feature space. This happens because at each split only one feature is involved. If this is beneficial or not depends on the data.

Random Forest

- What is a **random forest**?

Random Forest

- Individual decision trees can be very good at learning relationships, but they tend to **overfit** the data you use to train them on and **poorly generalize to new data**.

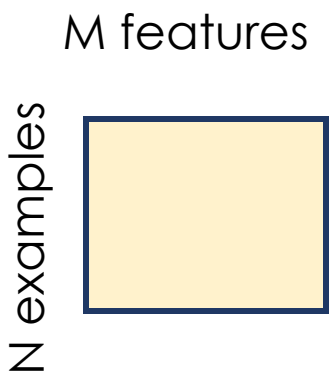
- Random Forest creates multiple **CART trees based on "bootstrapped" samples** of data and then combines the predictions.
  - Subset of data & subset of features

- Usually, the combination is an average of all the predictions from all CART models.

- Logic:
  - Weaker learners can be combined into a strong learner
  - Multiple models put together can perform much better as a group than they do alone

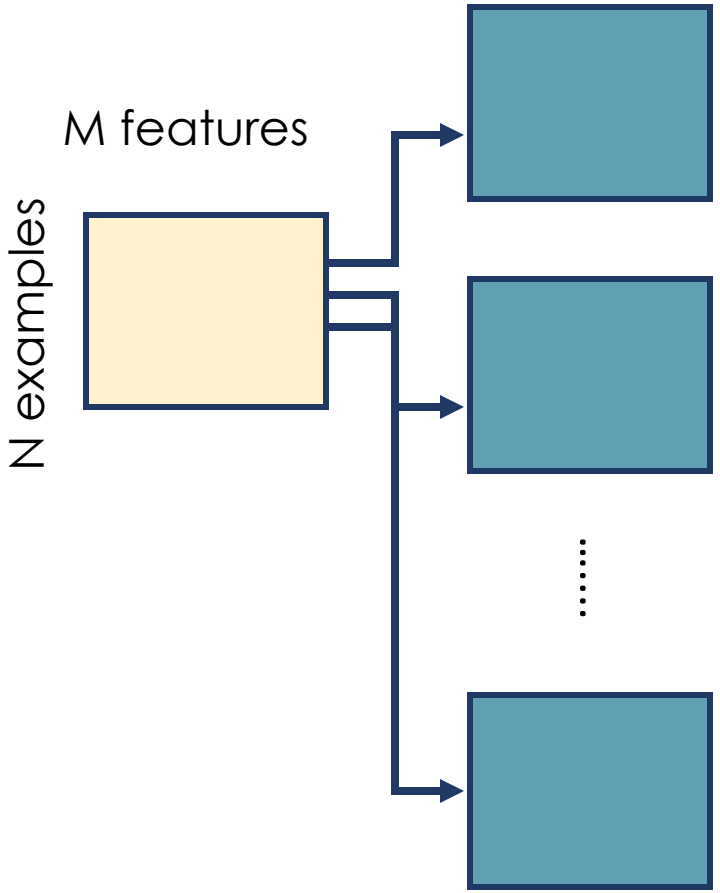
Random Forest

We start with the full data



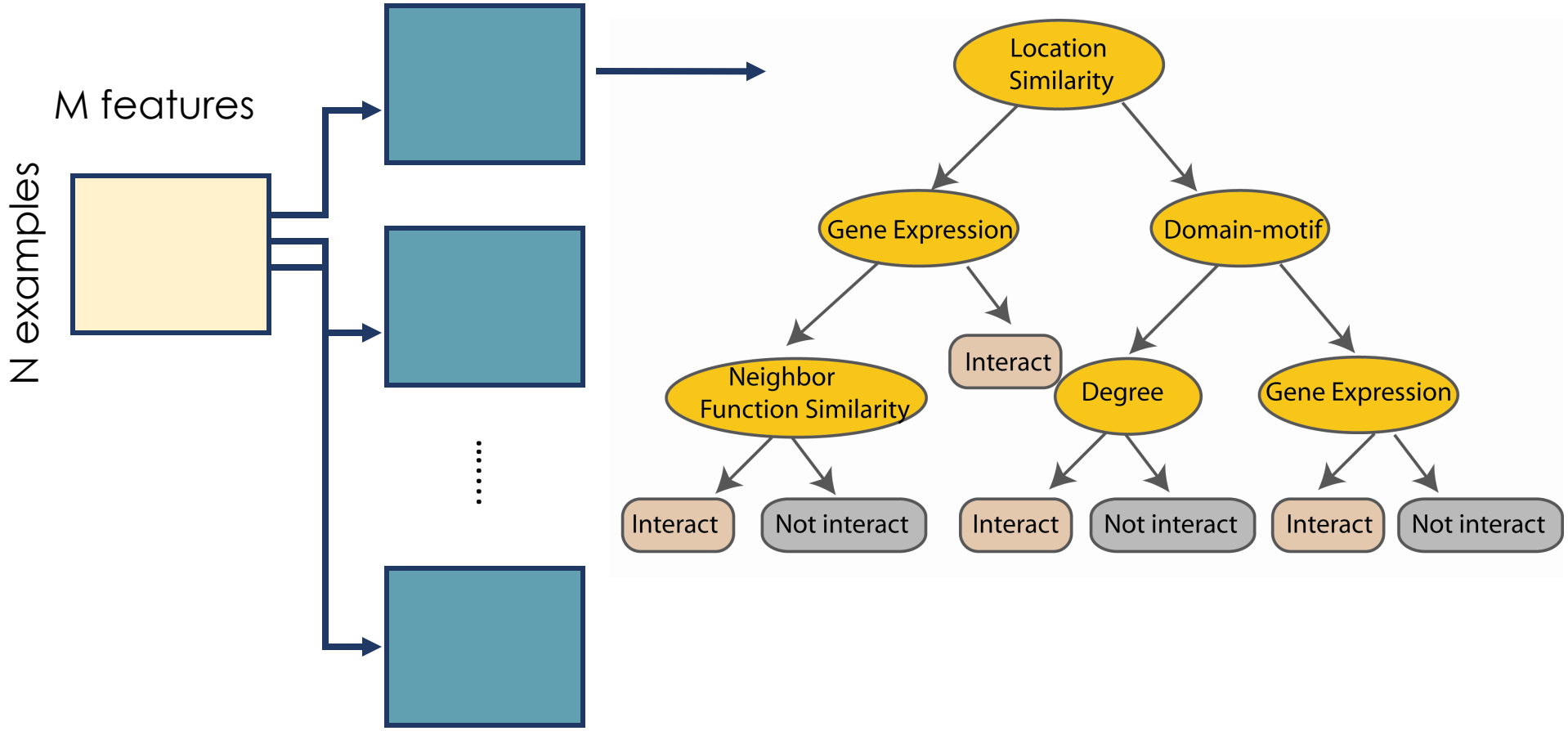
Random Forest

Next, we create bootstrap samples from the training data



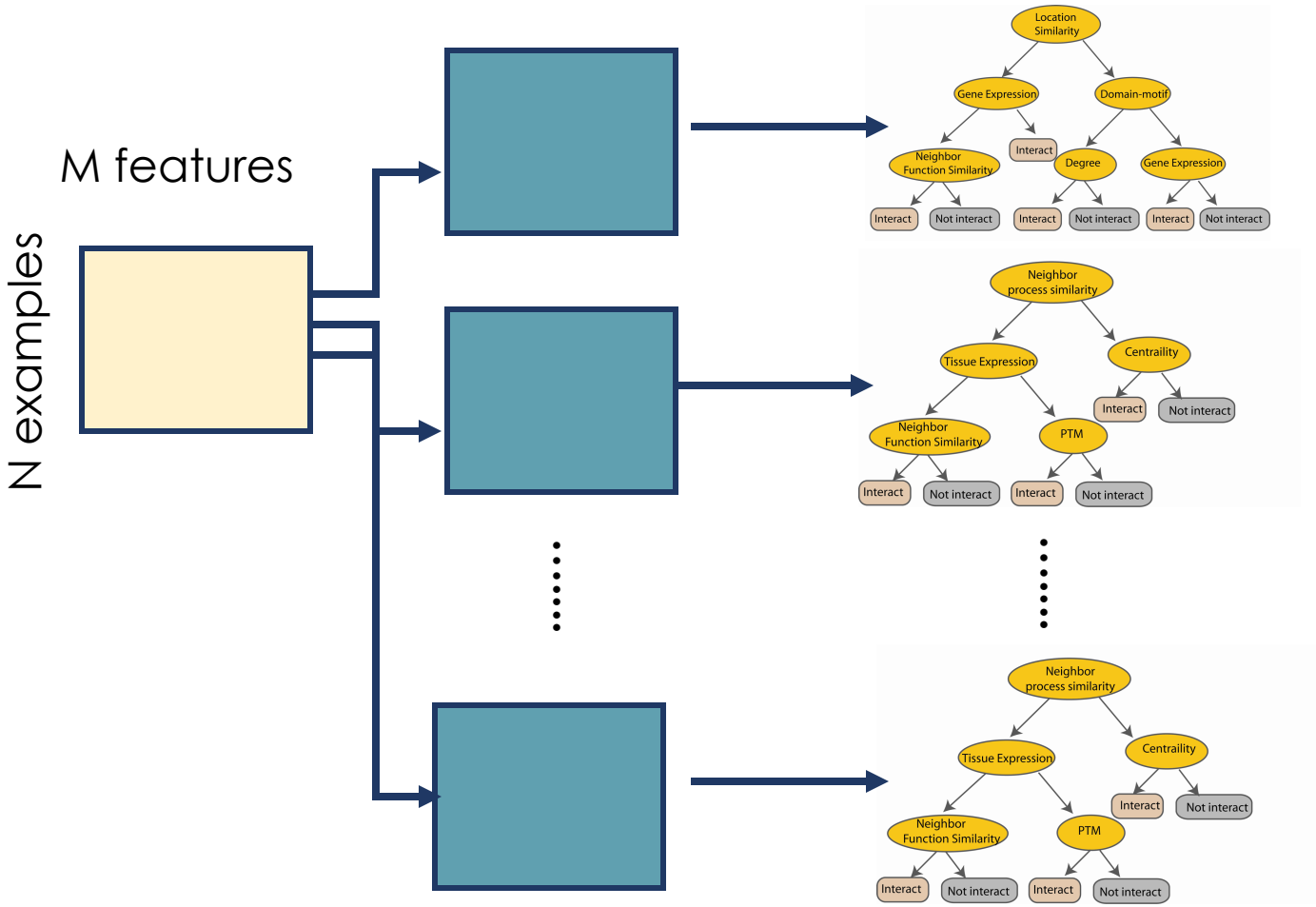
Random Forest

Next, from each subset we construct a decision tree

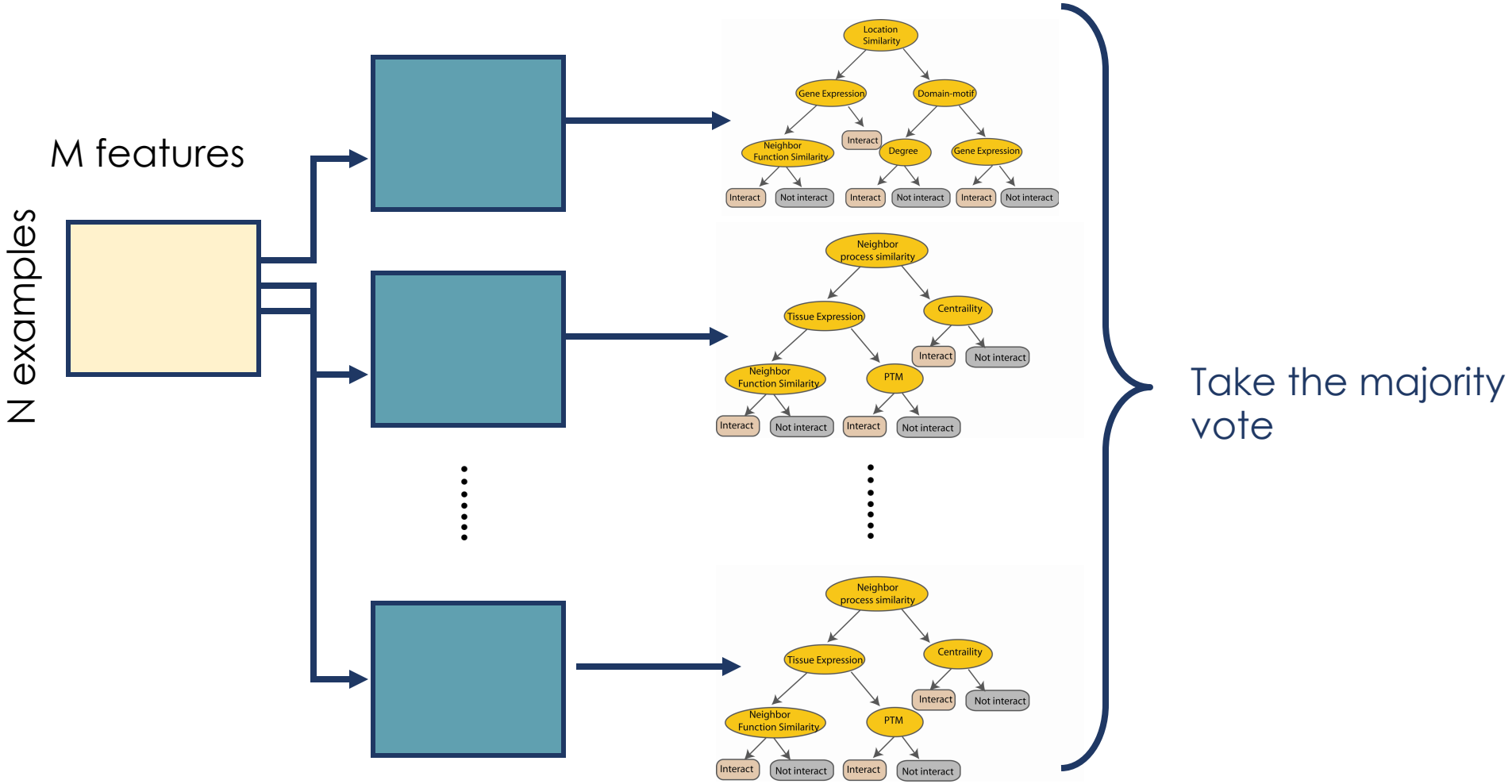


Random Forest

We do the same for each sample



Random Forest



Gradient boosting

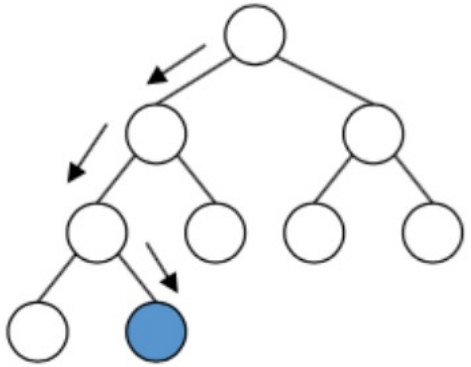
Gradient Boosting

- Remember: in supervised learning problems, there is an output variable and a vector of input variables and the goal is to find some function that best approximates the output variable from the input variables (loss function).

- What is a gradient boosting algorithm?
  - An **ensemble** of weak decision trees

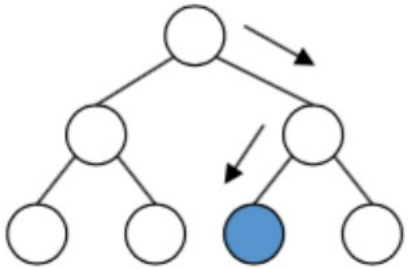
- Intuition → the best possiblle next model, when combined with previous models, minimizes the overall error.

Gradient Boosting: **How does it work?**



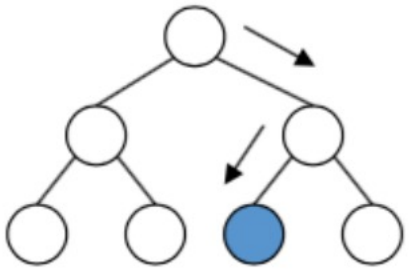
- **Goal:** to find a function which best approximates the data

- The model tries to find the best function **AND** the best parameters that minimise a certain loss



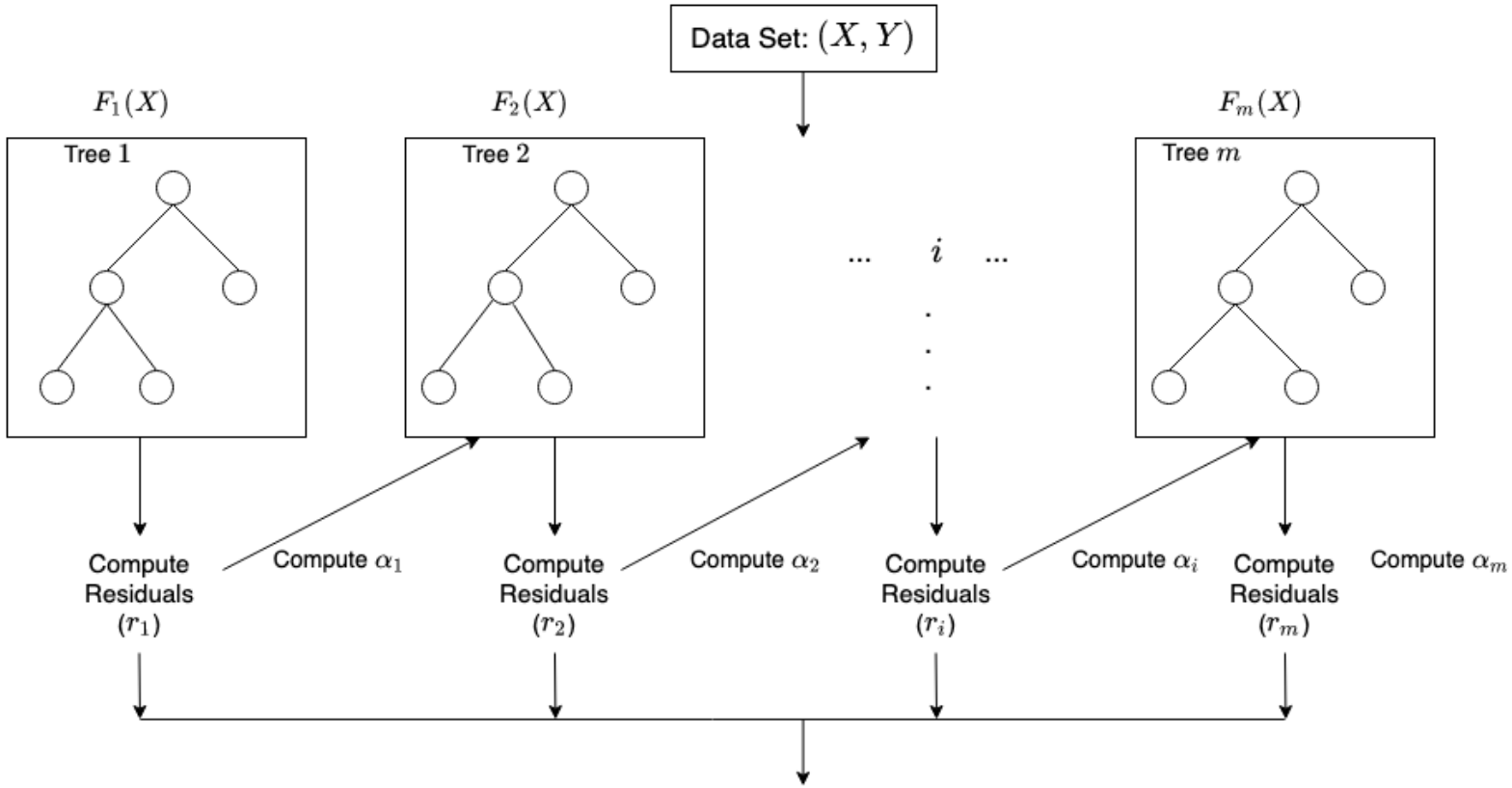
- For each consecutive model, we want to improve the errors of the first model i.e., we fit the next model on **the gradient of the error with respect to the prediction**

- Imagine, that for a data point y = 1, our first model gives $\hat{y} = 0.6$ hence the next model will be the gradient i.e. (y - $\hat{y}$) = 0.4.



- Last step: adding these models together!

Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

By training the second model on the gradient of the error with respect to the loss predictions of the first model, it has learned to correct the mistakes of the first model.

Random forest **vs** Gradient Boosting

- **How weaker learners are build?**
  - Unlike random forests, the decision trees in gradient boosting are built additively; in other words, each decision tree is built one after another.

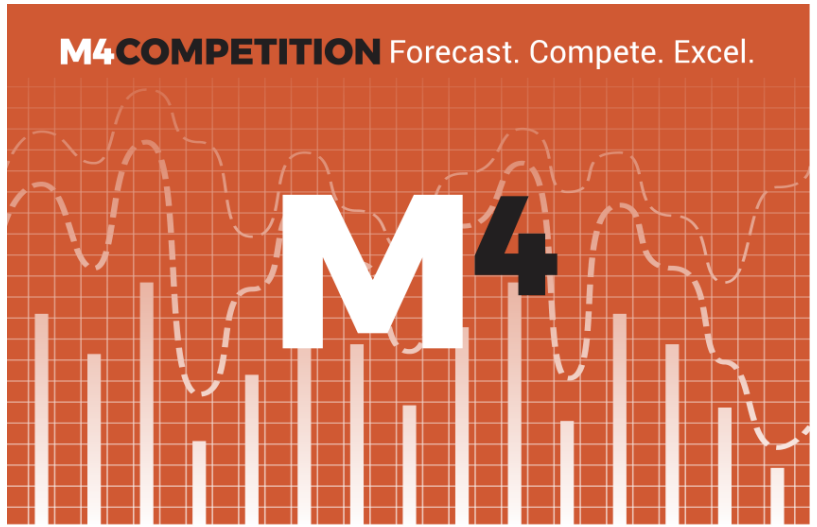
- **How results are aggregated?**
  - In random forests, the results of decision trees are aggregated at the end of the process. Gradient boosting does not do this and instead aggregates the results of each decision tree along the way to calculate the final result.

XGBoost

XGBoost Algorithm: Long May She Reign!

The new queen of Machine Learning algorithms taking over the world...

- **eXtreme Gradient Boosting**
- Fastest implementation of gradient boosting trees
- XGBoost is an incredibly popular machine learning library
- It was developed originally as a C++ command-line application
- After performing well in several popular machine learning competitions, the package started being adopted within the ML community
- As a result, now we have available implementations in many programming languages including **Python, R, Scala, and Julia.**



XGBoost: **Advantages**

- Speed:
  - it can harness all of the processing power of modern multi-core computers
  - it is parallelizable onto GPU's and across networks of computers
  - Hence: it is suited for training models on very large datasets (**hundreds of millions of training examples**)

- Performance:
  - It consistently outperforms almost all other single-algorithm methods in ML competitions
  - It achieves state-of-the-art performance on a variety of benchmark ML datasets

Putting **all together**

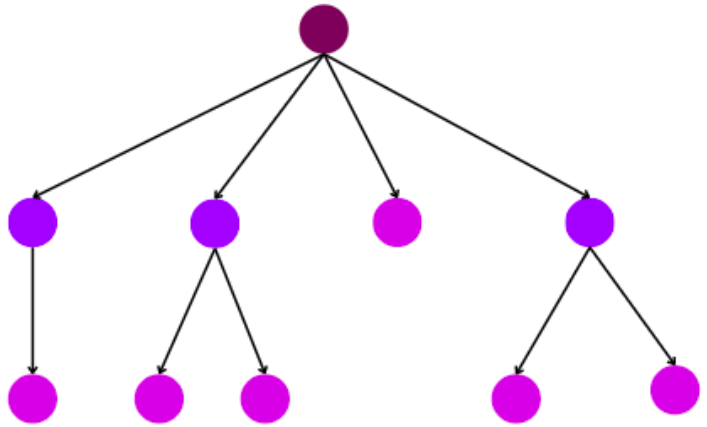
- Let's try to summarize though **an example**
- Imagine you are the manager of a company and you are deciding who, among a list of applicants, to hire.
- How would the decision happen under the different methods we discussed so far?

Putting **all together**

The manager has a set of criteria such as skills acquired, education level, number of year in industry, etc., on which it scores the candidates

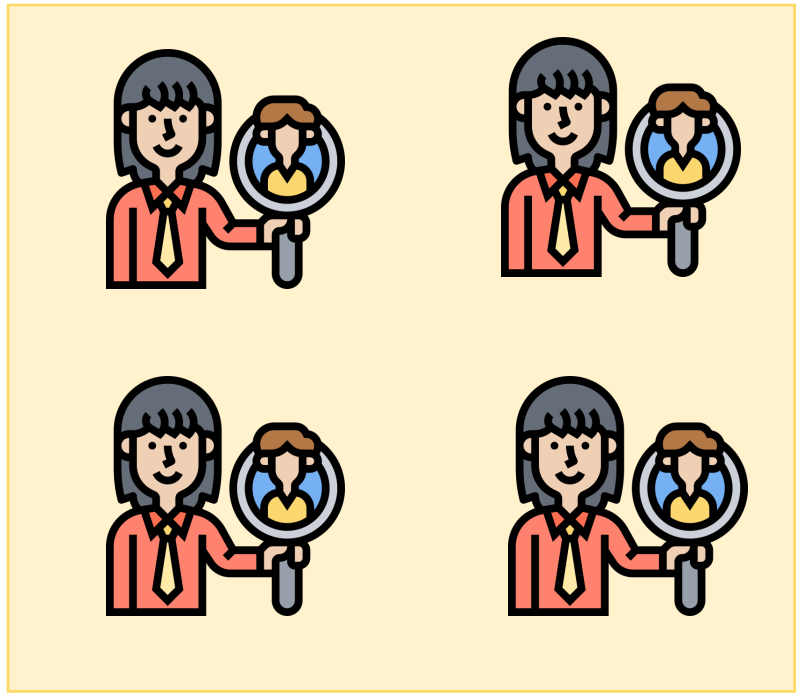
=

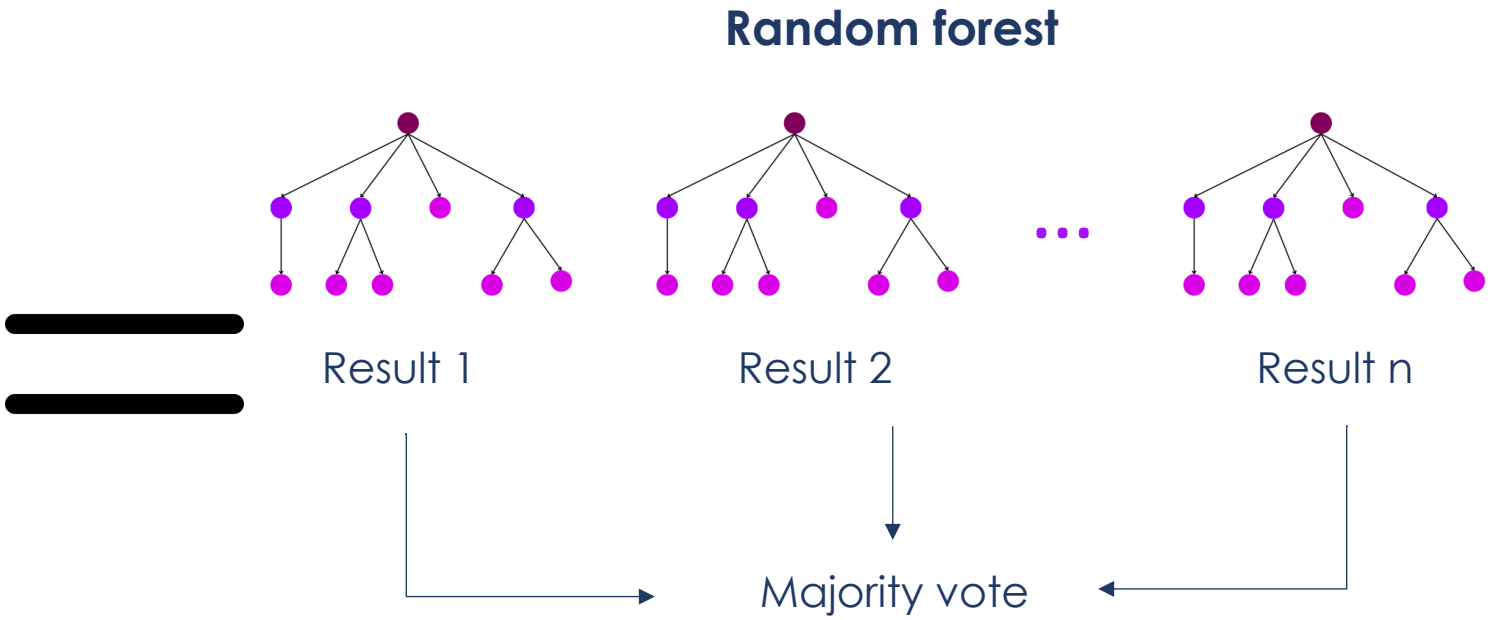
Single Decision Tree



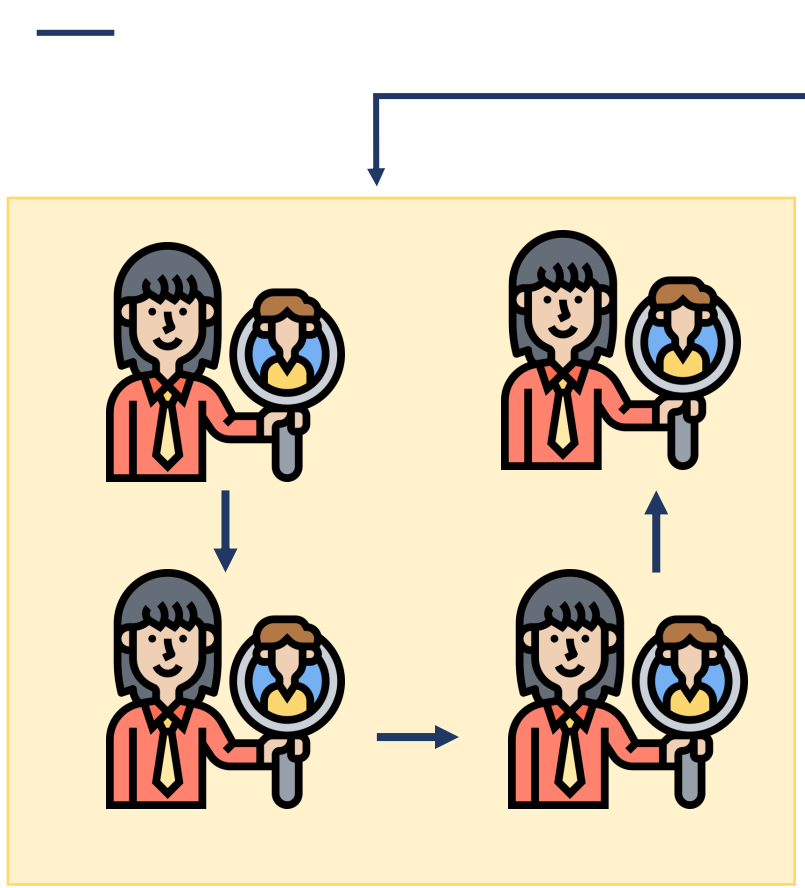
Putting **all together**

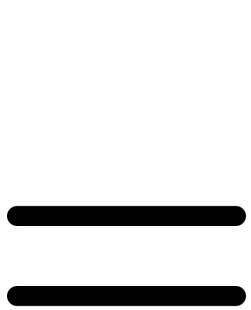
We now have **a panel** and each interviewer has a vote. Each interviewer will also score the applicants based on **a randomly selected subset of qualifications.**



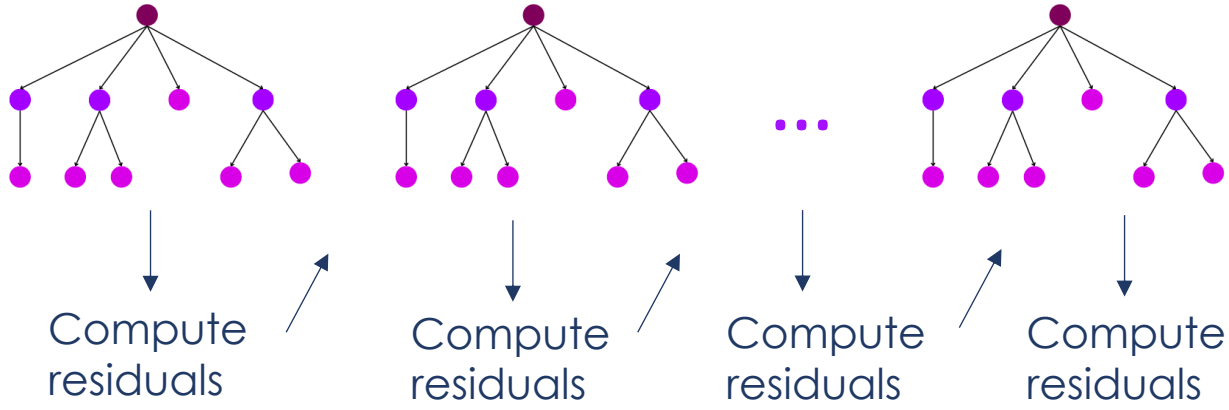


Putting **all together**





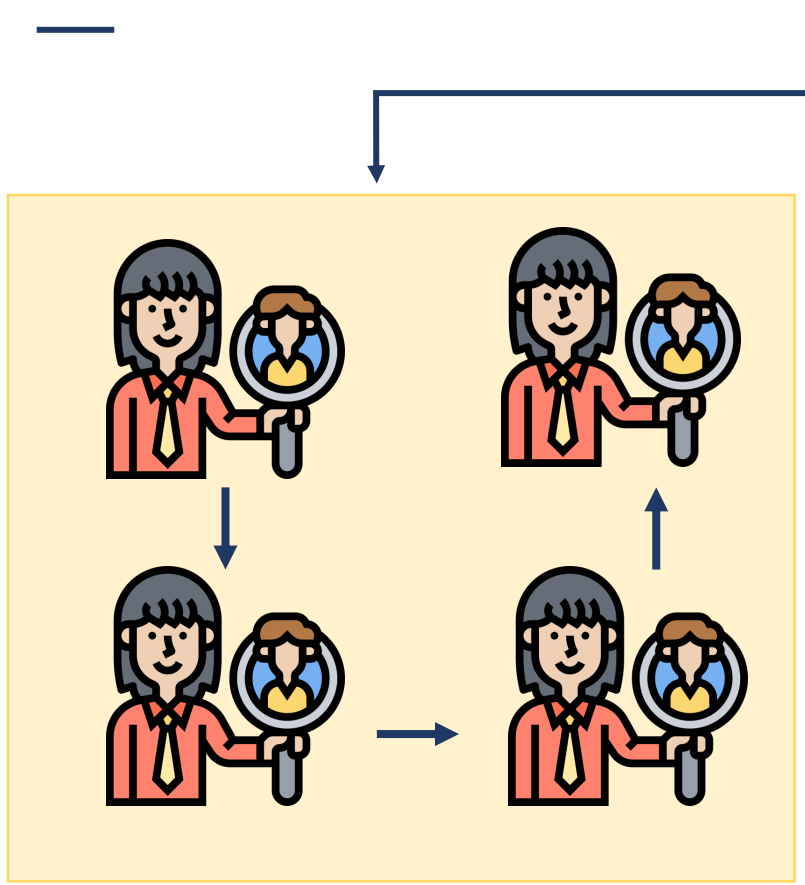
Gradient Boosting



Each interviewer alters the evaluation criteria based on the feedback of previous interviewers.

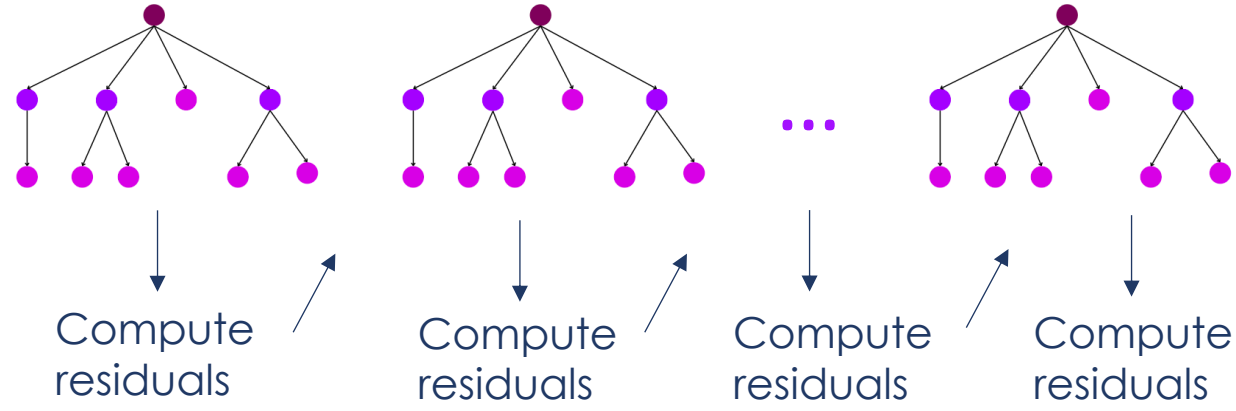
Plus strategy consulting firm steps in to weed out less qualified candidates

Putting **all together**



Enhanced
gradient boosting

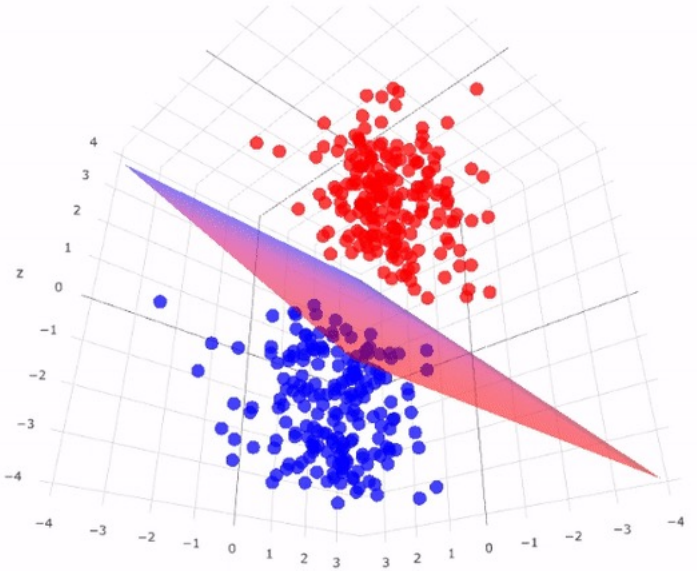
XGBoost



Software and hardware optimization techniques

Support Vector Machine

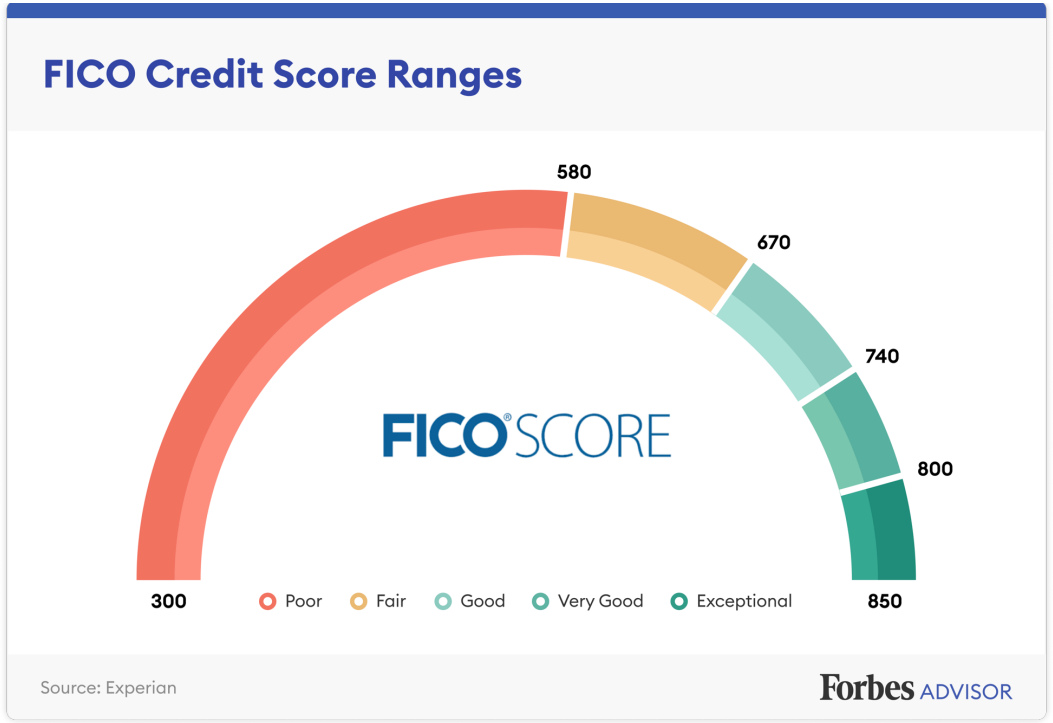
SVM

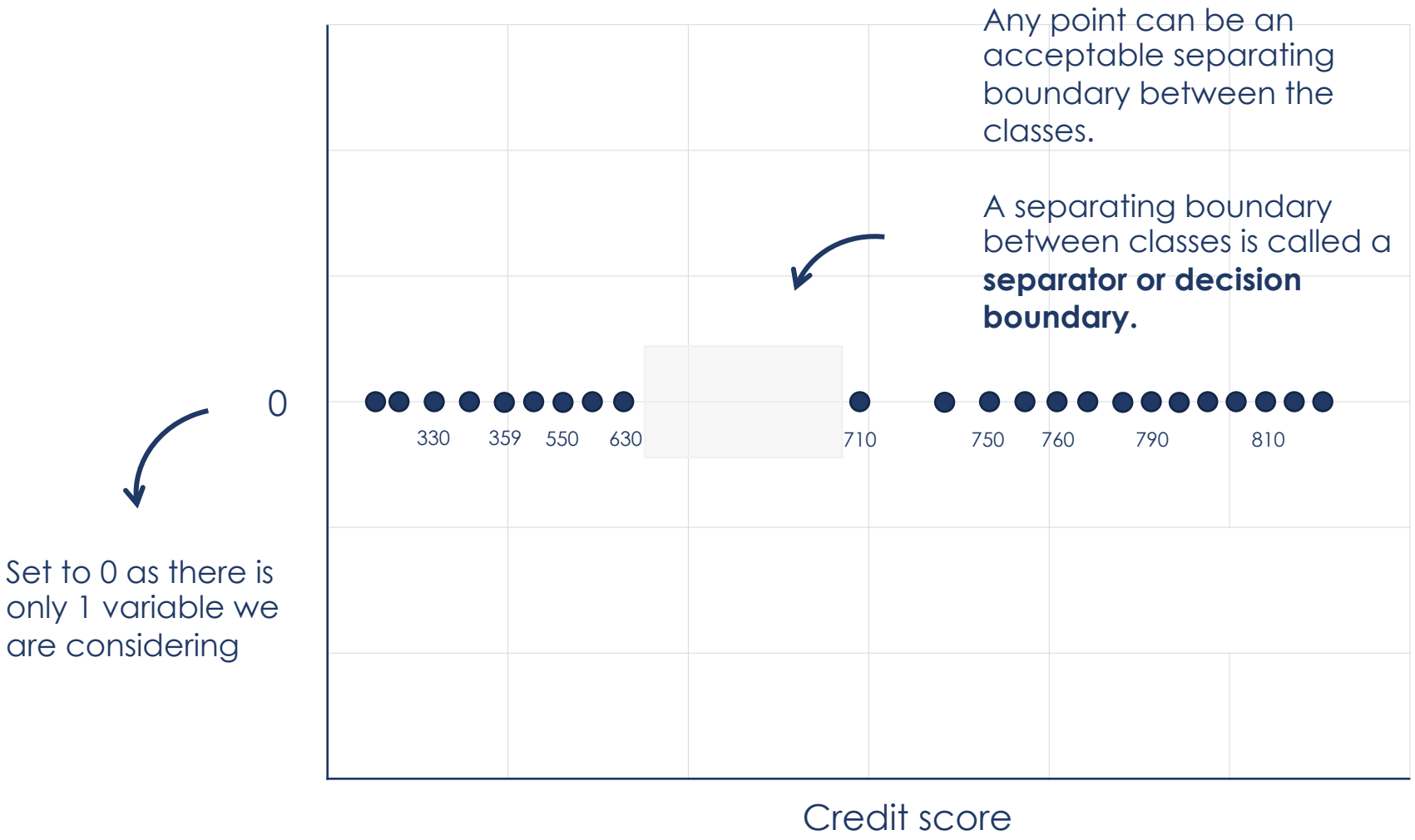


- SVM → Support Vector Machine
- Another type of supervised machine learning algorithm that are commonly used for **classification and regression.**
- SVMs try to find the hyperplane that best separates data into different classes, with the goal of maximizing the margin, or distance, between the hyperplane and the closest data points from each class.
- Despite their effectiveness, **SVMs can be computationally intensive** and require careful tuning of their parameters.

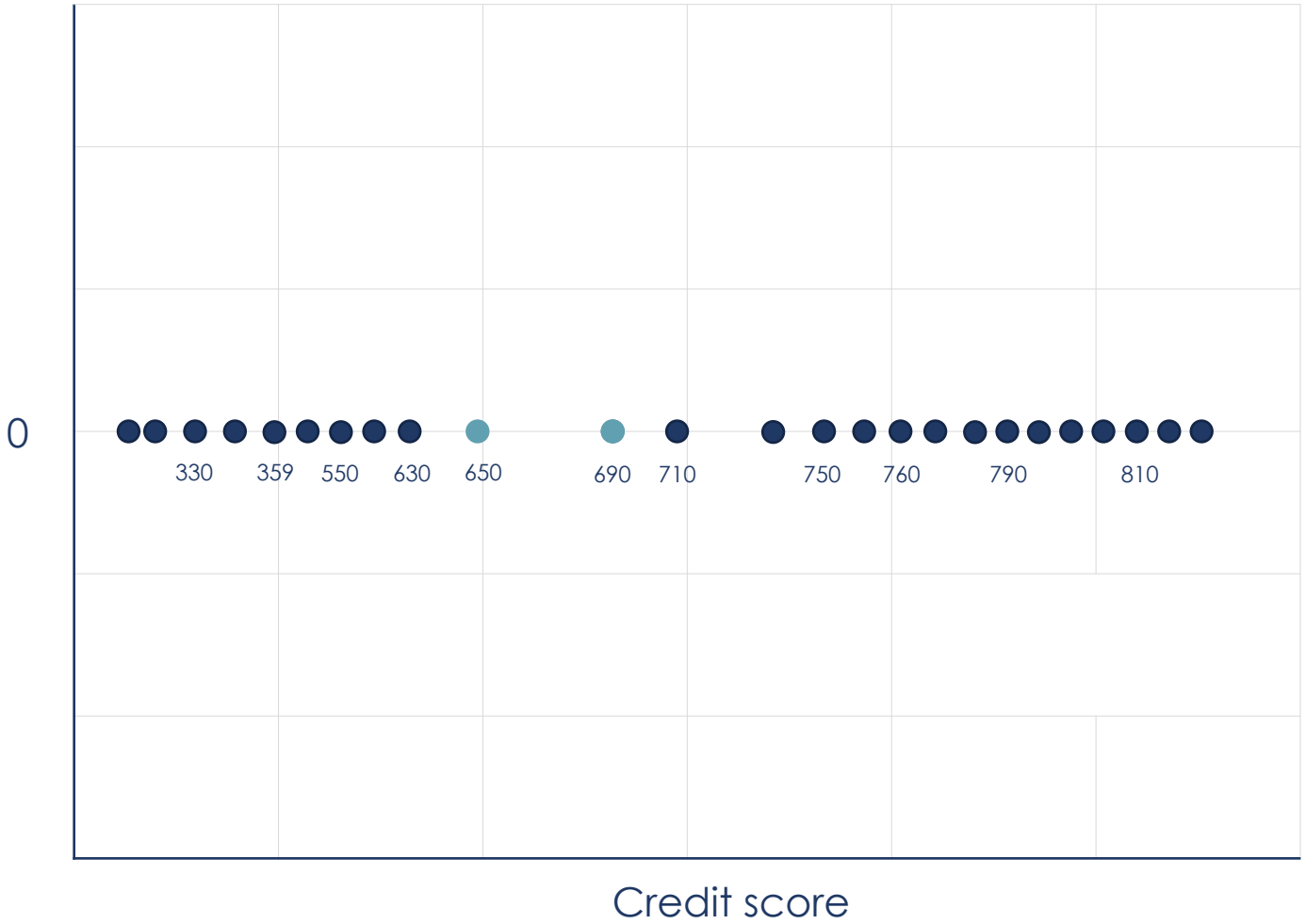
SVM: **Intuition**

- We start with a binary classification – problem that has two classes.
  - Loan status: **default or non-default**
  - Credit score: this is a feature that ranges from 300 to 850. Applicants that score around 740 are considered of good credit score while applicants that score around 550 are considered of poor credit score
  - Of course, in practice, the credit score of good vs bad applicants (i.e. non-defaulted vs defaulted loans) **varies quite a bit.**
  - Given 24 random samples, **our task is to determine a decision rule to distinguish between the two groups!**

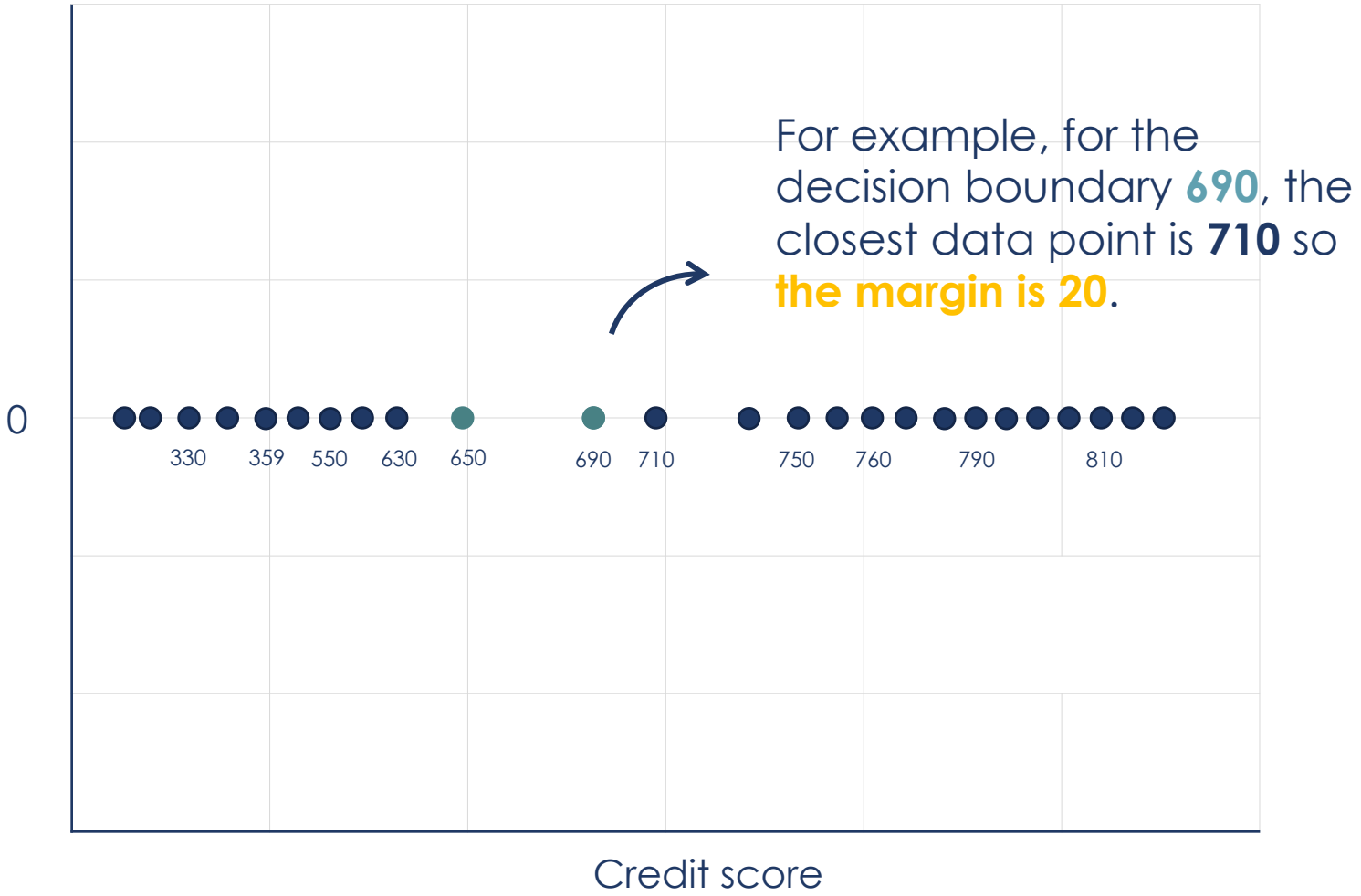


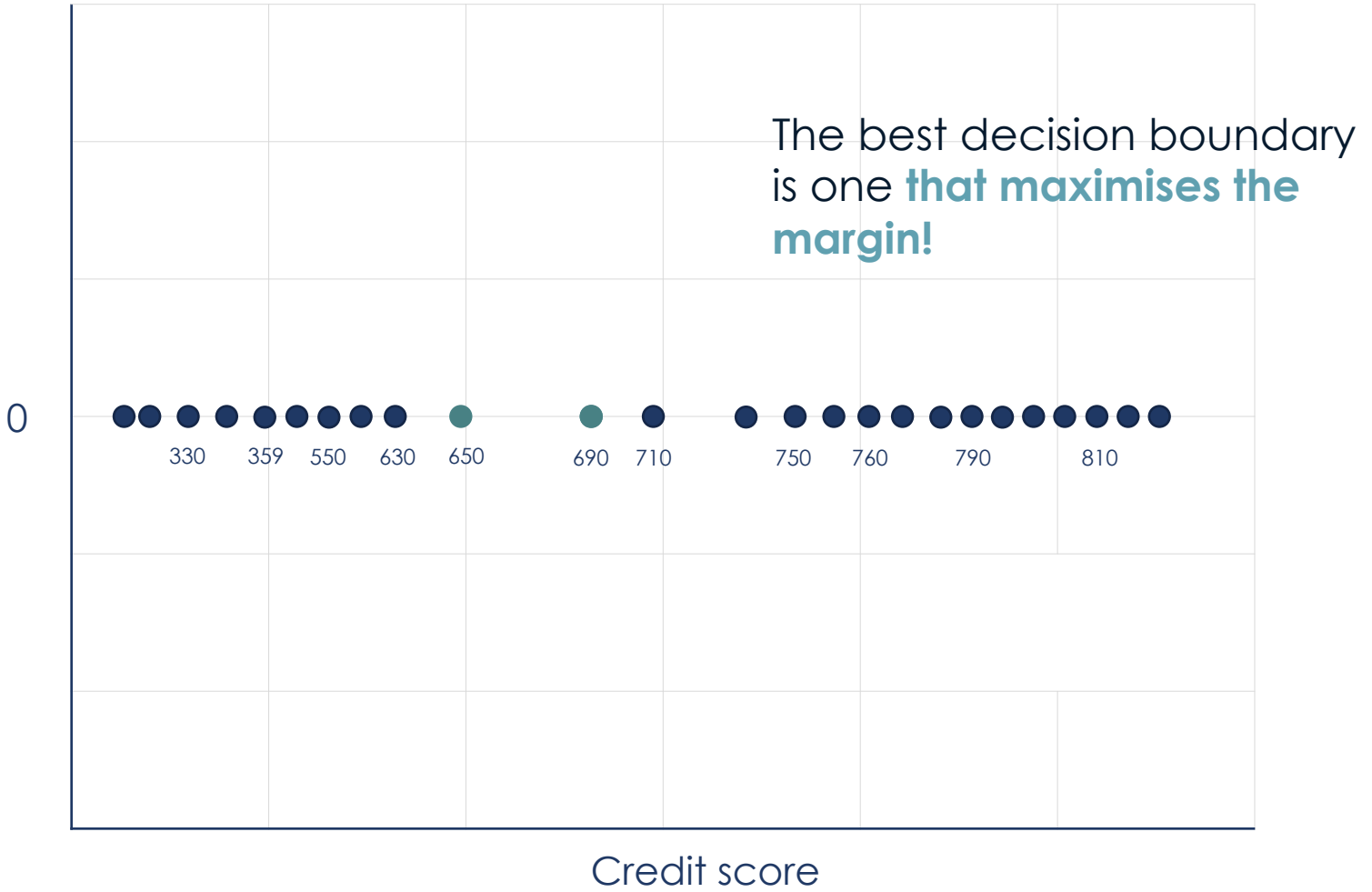

Any point can be an acceptable separating boundary between the classes.
A separating boundary between classes is called a **separator or decision boundary.**
0
330
359
550
630
710
750
760
790
810
Set to 0 as there is only 1 variable we are considering
Credit score

Let's take two points in this area:
650 and 690

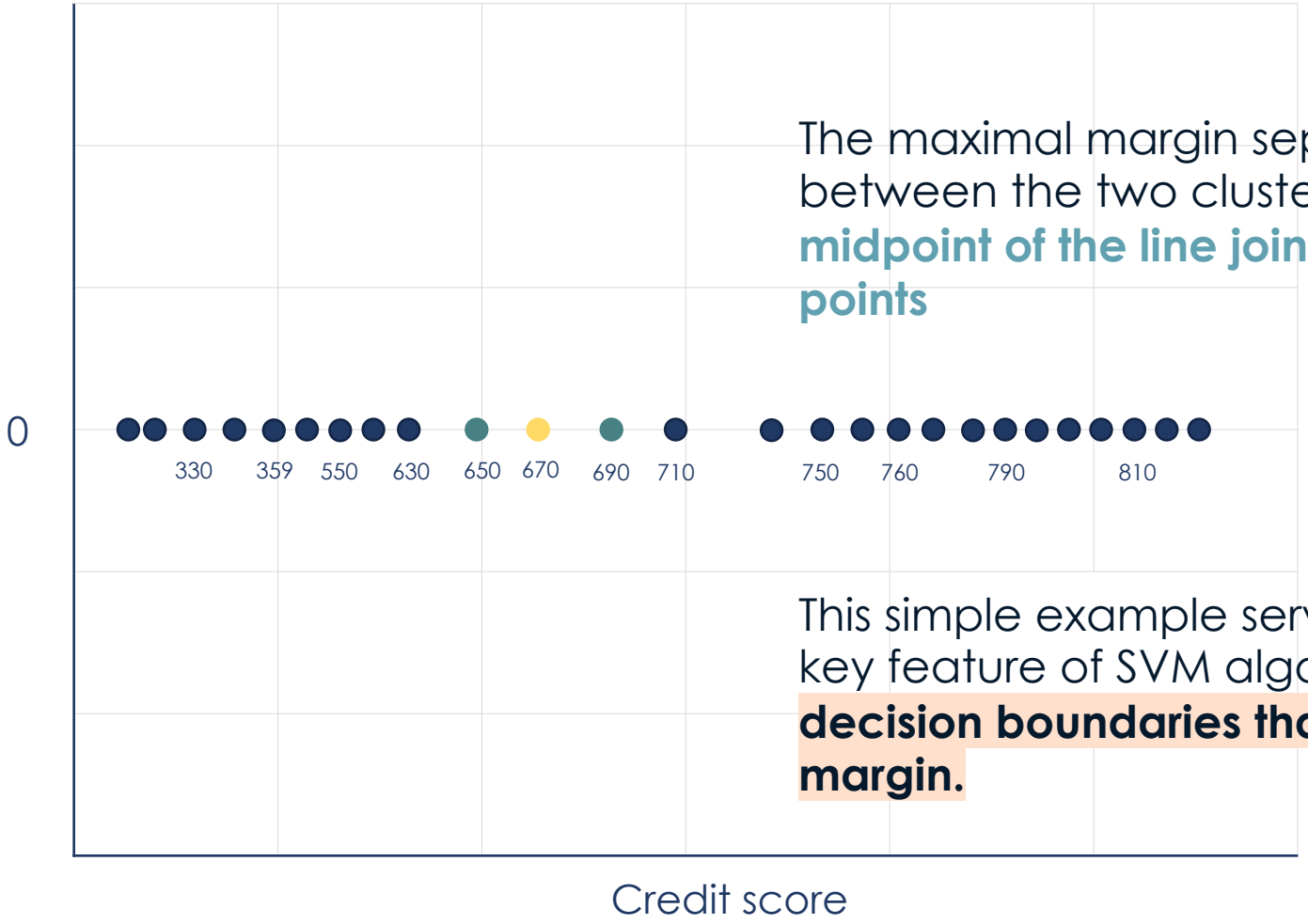


An important concept is that of the **margin**, which is the distance between the decision boundary and the closest data point


For example, for the decision boundary 690, the closest data point is 710 so the margin is 20.
0
330
359
550
630
650
690
710
750
760
790
810
Credit score


The best decision boundary is one that maximises the margin!
0
330
359
550
630
650
690
710
750
760
790
810
Credit score


Looking at the graph, which is the best decision boundary?
0
330
359
550
630
650
690
710
750
760
790
810
Credit score

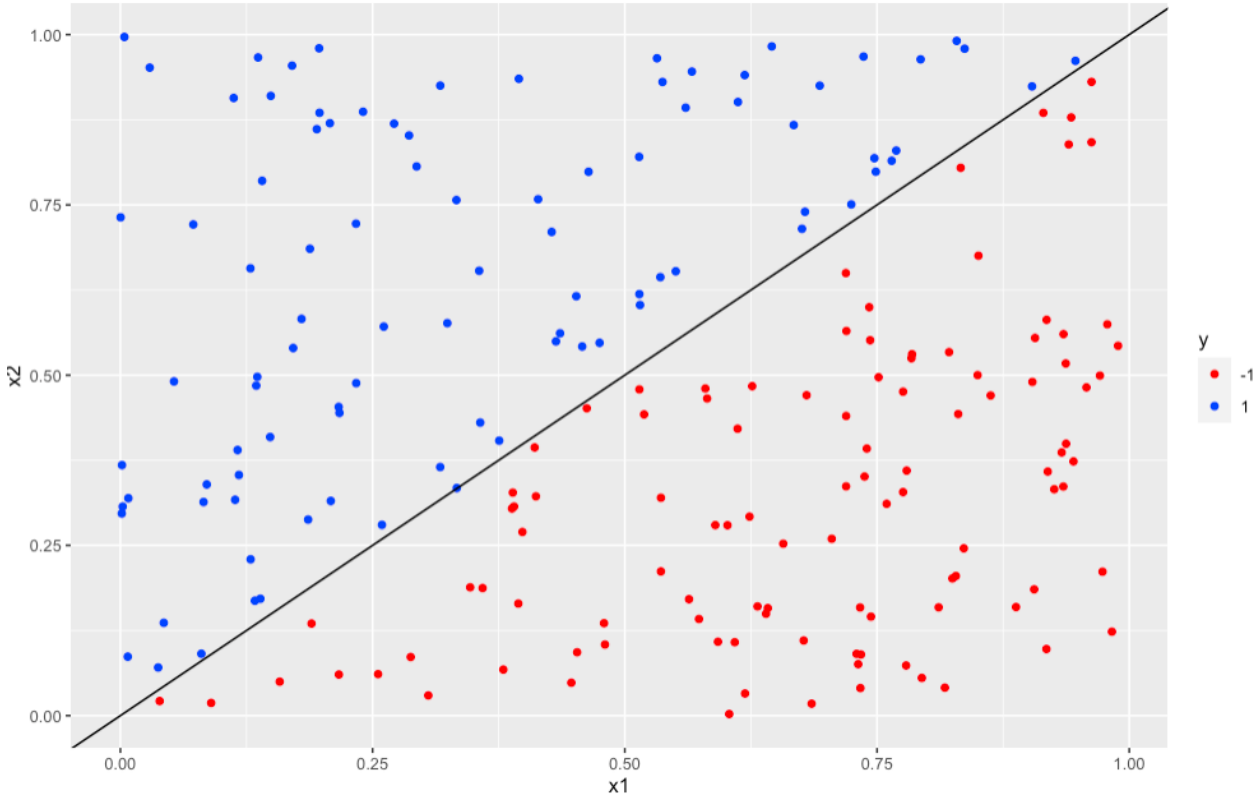


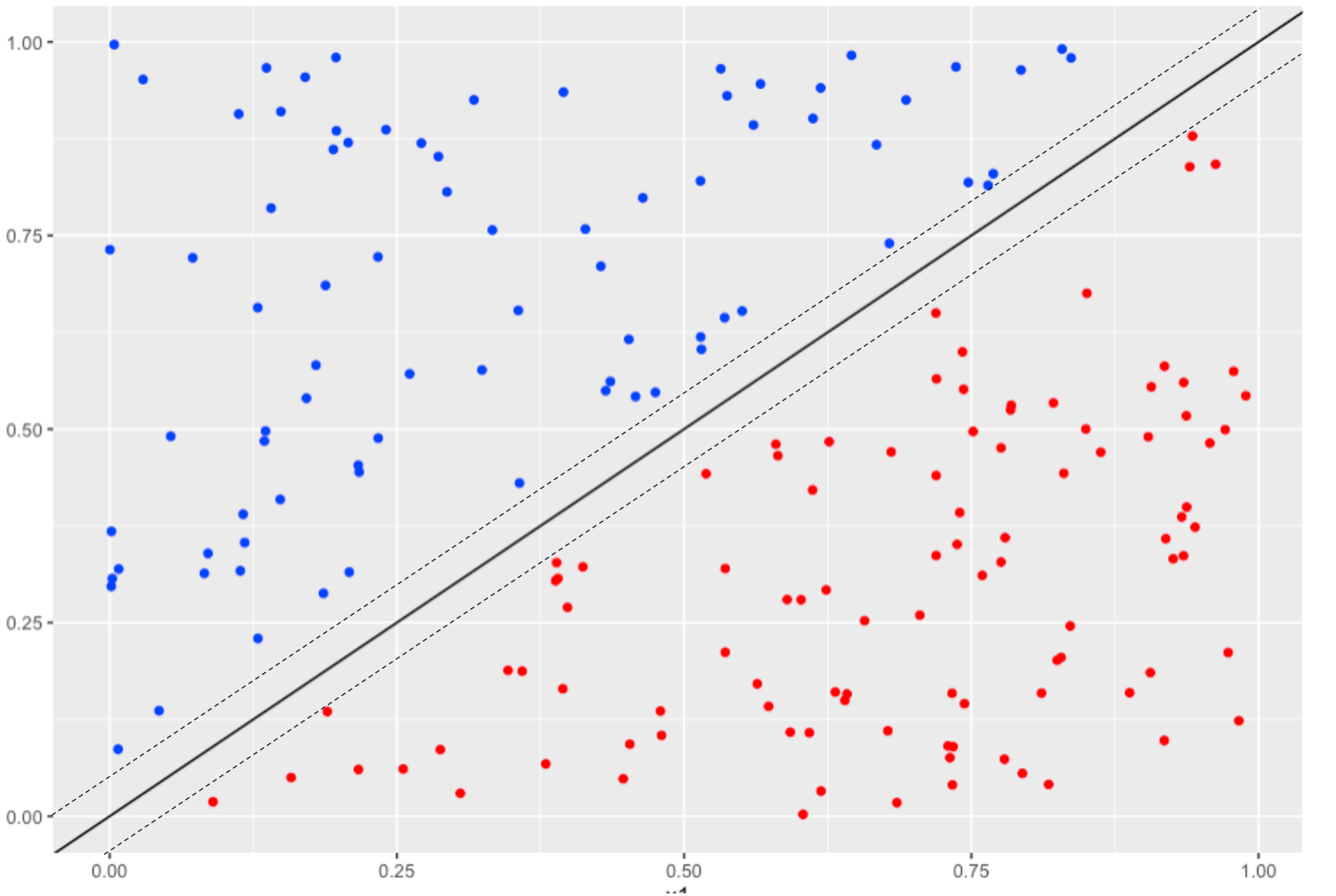
The maximal margin separator lies halfway between the two clusters. That is, at the **midpoint of the line joining the sample data points**

This simple example serves to illustrate a key feature of SVM algorithms → **they find decision boundaries that maximize the margin.**

SVM: **Intuition**

- 2 dimensions
- 200 random datapoints and 2 classes separated by a streight line (x1 = x2)
- The decision boundary that separates the classes passes through the origin and makes an angle of 45 degrees with the horizontal axis (slope = 1)
- **Notice:** the decision boundary has no margin.
- Let's introduce it!





Notice the empty space on either side of the decision boundary. **This is the margin.**

The decision boundary is the maximal margin separator because **it lies halfway between the margin boundaries.**

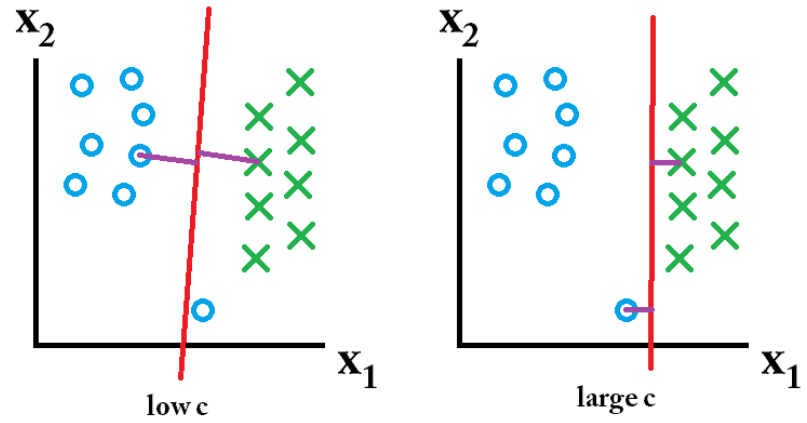
Note: C parametar tuning for SVM

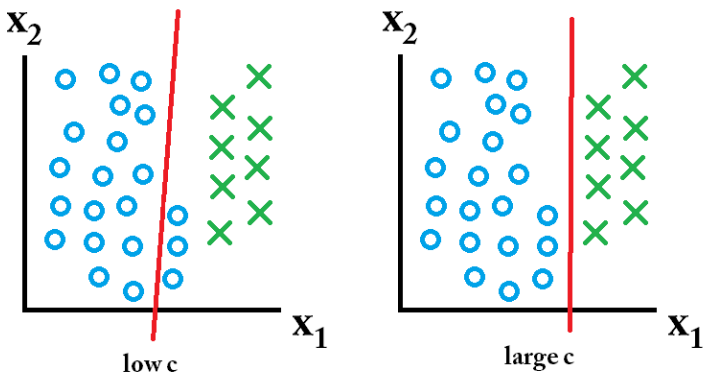
- In a SVM, you are searching for two things:
  - (i) a hyperplane with the largest margin,
  - (ii) a hyperplane that correctly separates as many instances as possible.

- It might not be always possible to **get both things**.

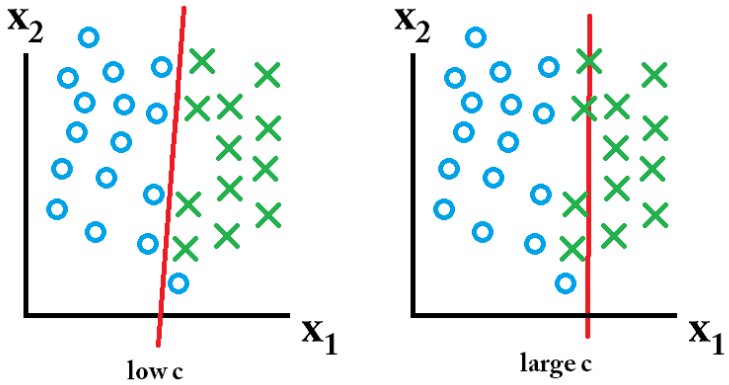
- The c parameter determines how much you favour (ii).

Training





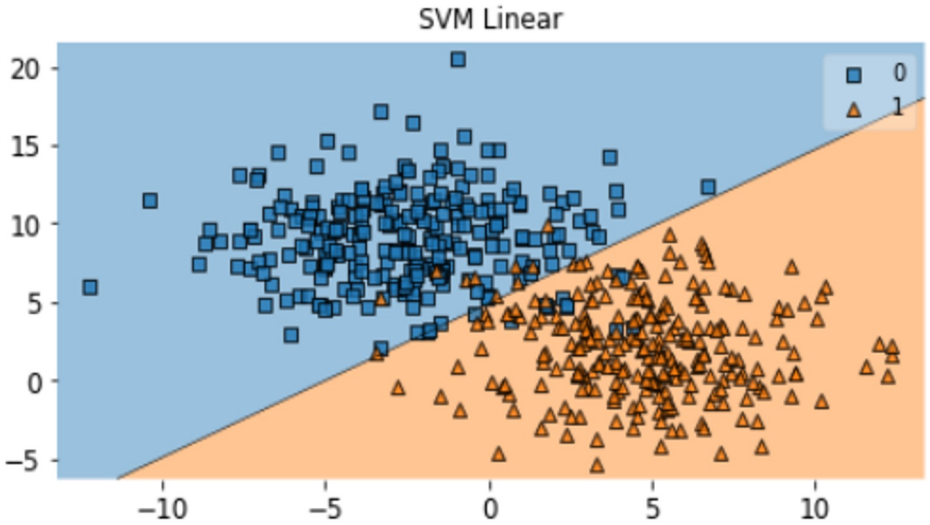
Possible future data set 1

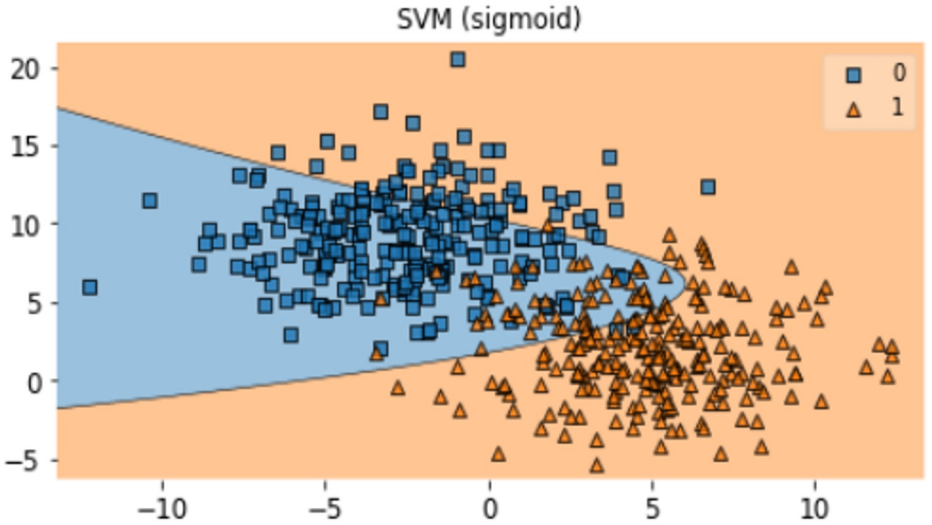


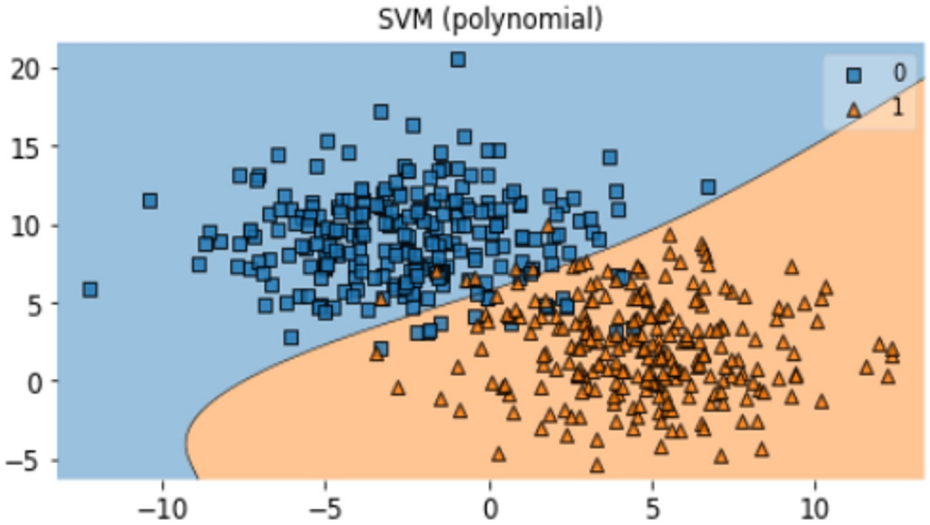
Possible future data set 2

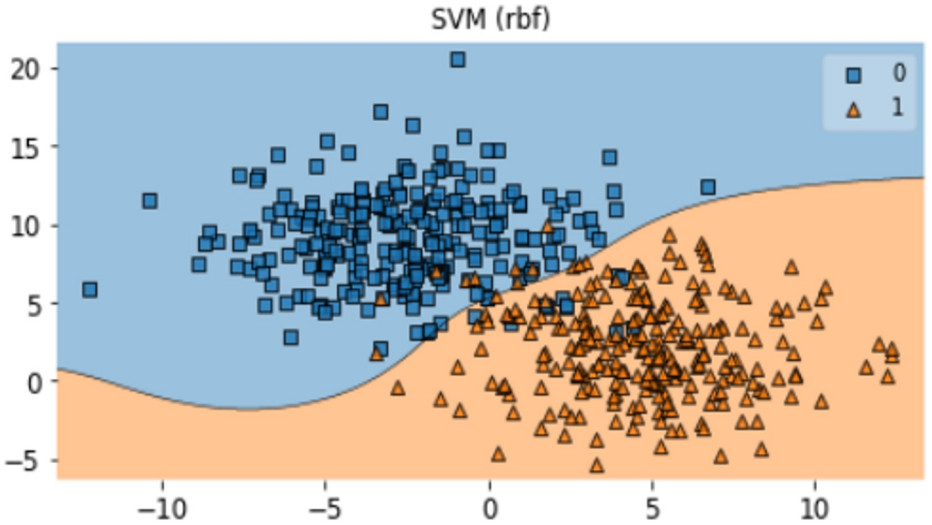
Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

SVM: Decision boundaries

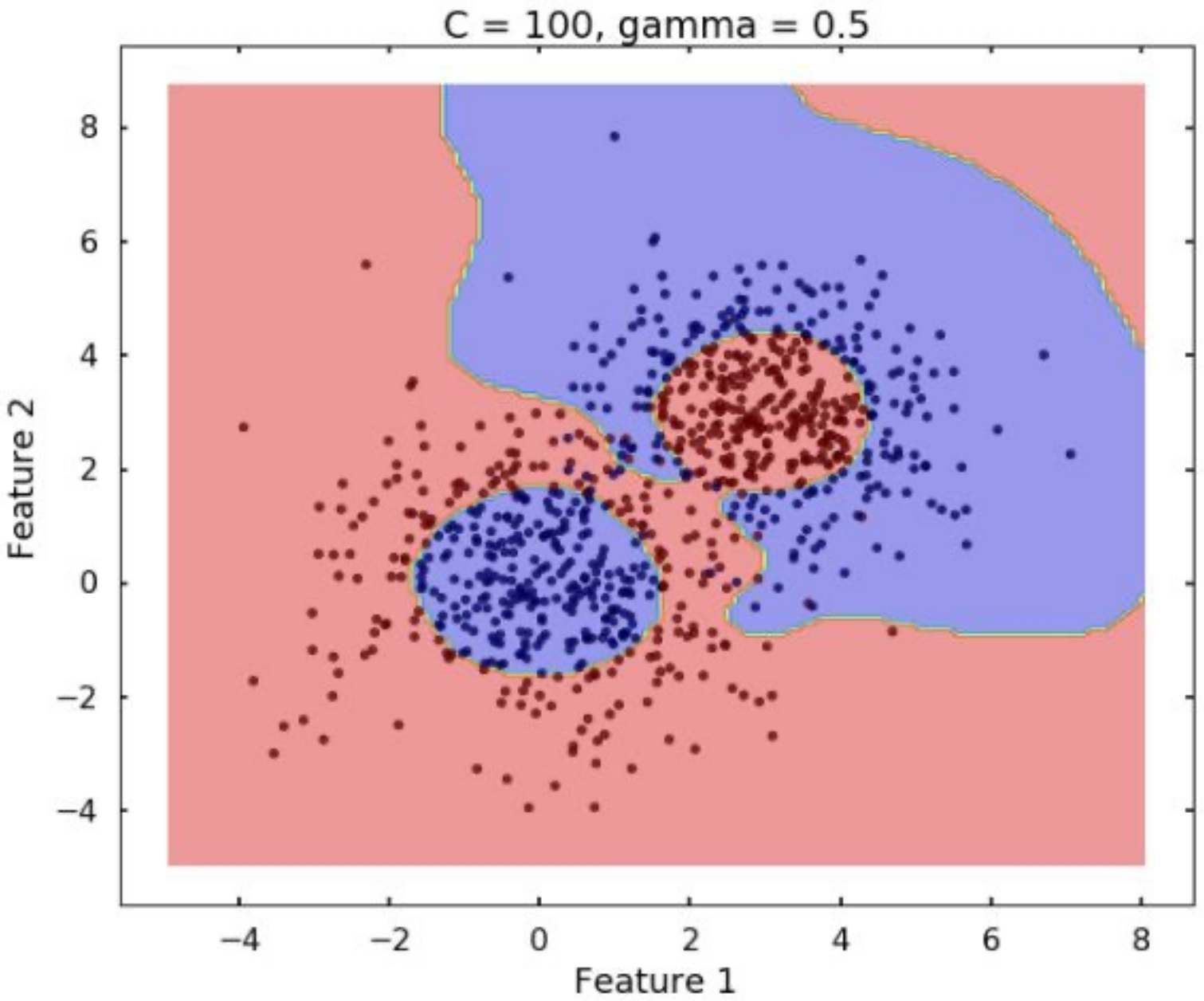








Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences



Radial basis function

Artificial Neural Networks

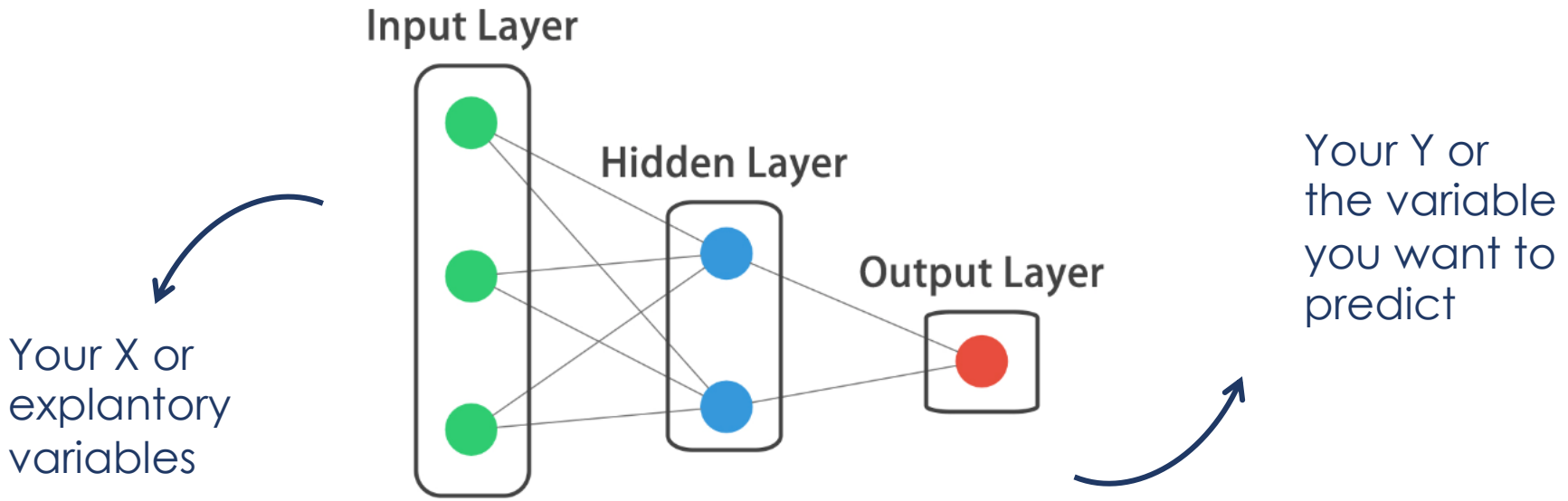
Starting arguments

- Artificial neural networks (ANNs) are a type of machine learning algorithm modeled after the **human brain's neural structure**.
- The idea of ANNs dates back to **the 1940s**, when Warren McCulloch and Walter Pitts proposed a simplified mathematical model of a neuron.
- However, the limited computational power available at the time made it difficult to train ANNs to handle more complex tasks.
- It wasn't until the 1980s, with the development of more powerful computers and new learning algorithms, that ANNs started to gain wider use and popularity.

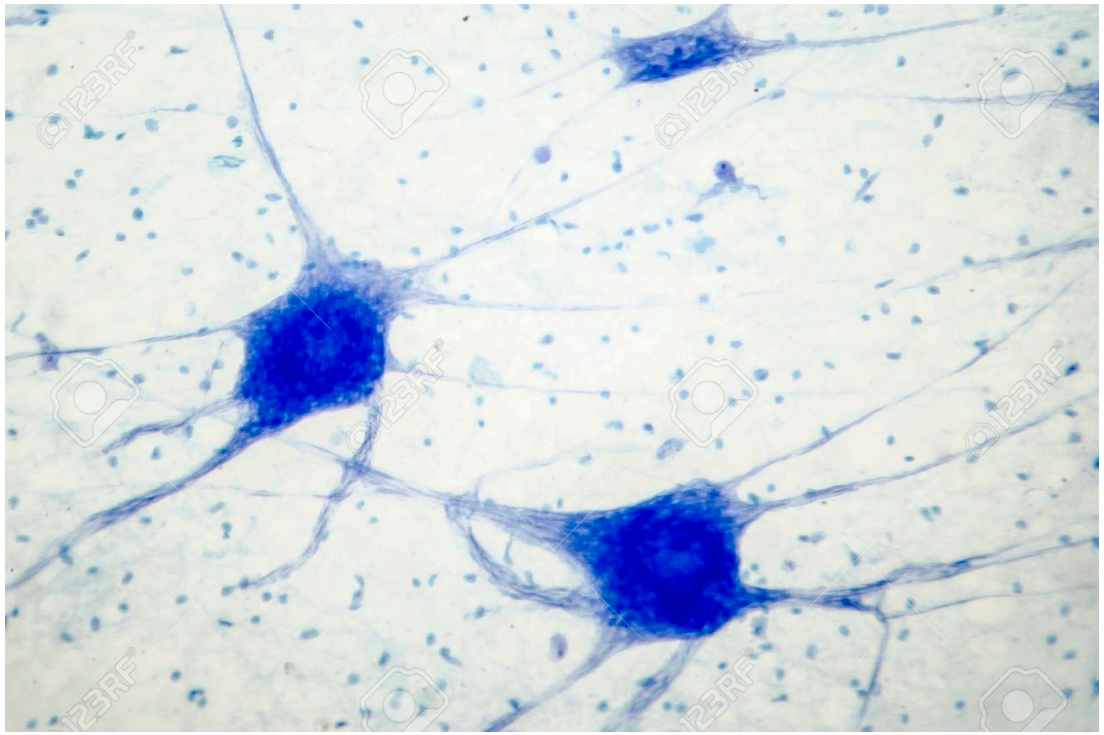
Artifical neural networks: **An Introduction**

—

- A neural network consists of an input layer, a hidden layer, and an output layer. The first layer receives raw input, it is processed by multiple hidden layers, and the last layer produces the result.

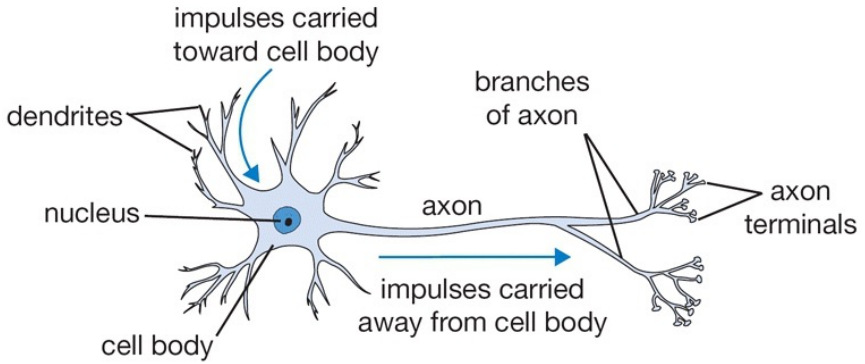


Neural networks: **An Introduction**



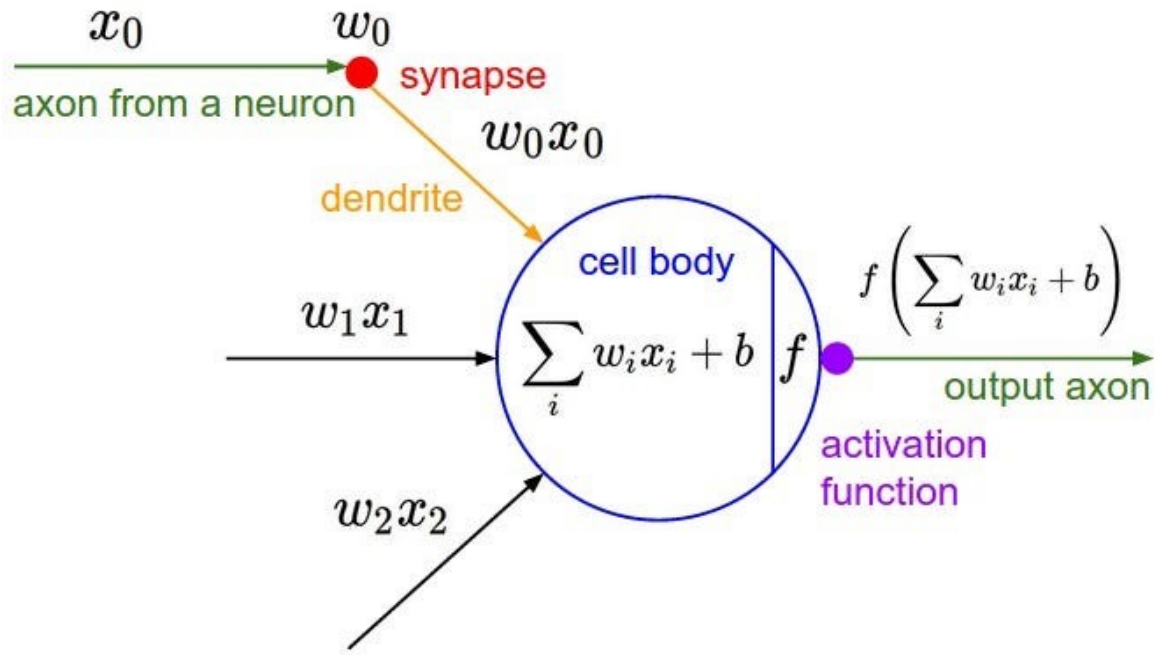
Approximately 86 billion neurons can be found in the human nervous system and they are connected with approximately 10^{14} — 10^{15} **synapses**

A neural network is **made of neurons, connected through synapses where information flows.**



When a neuron receives a stimulus with high enough voltage, it emits an action potential (aka, nerve impulse or spike). **It is said to fire.**

Further details



Matematical model

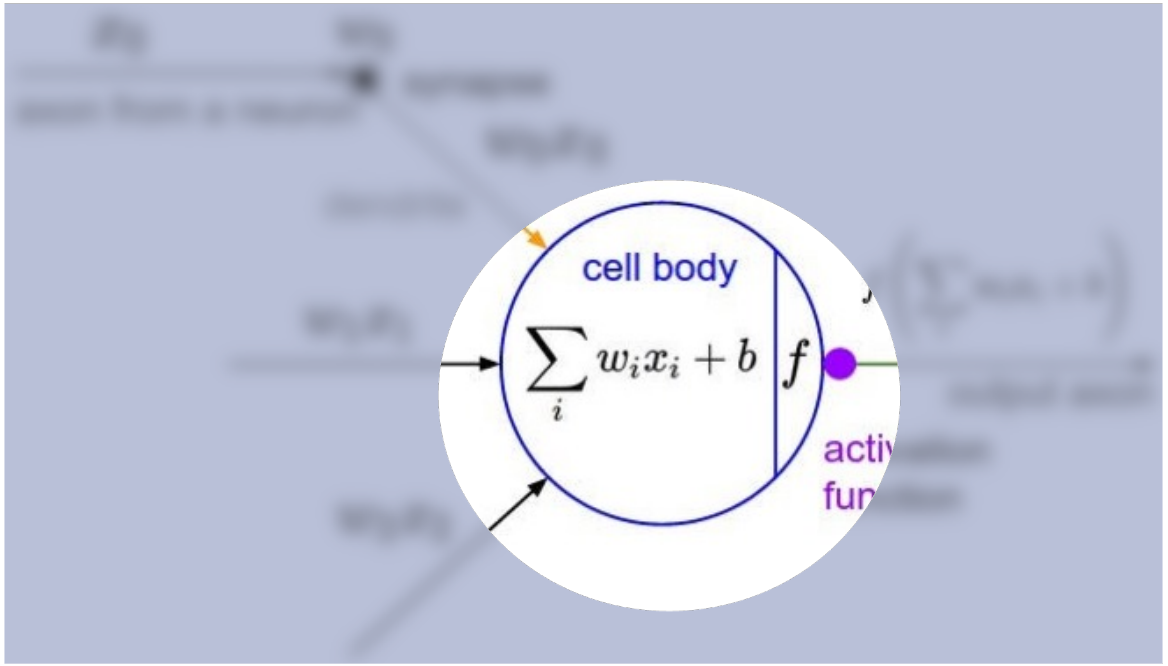
- The ANN mimics this activation effect. It fires when $\sum_i w_i x_i + b > 0$
- In other words, when we train a neural network, **we want the neurons to fire whenever they learn specific patterns** from the data.
- **We model the fire rate using an activation function**.

Further details

Input nodes:

- Input data
- No computation – information is passed to the next layer

Further details



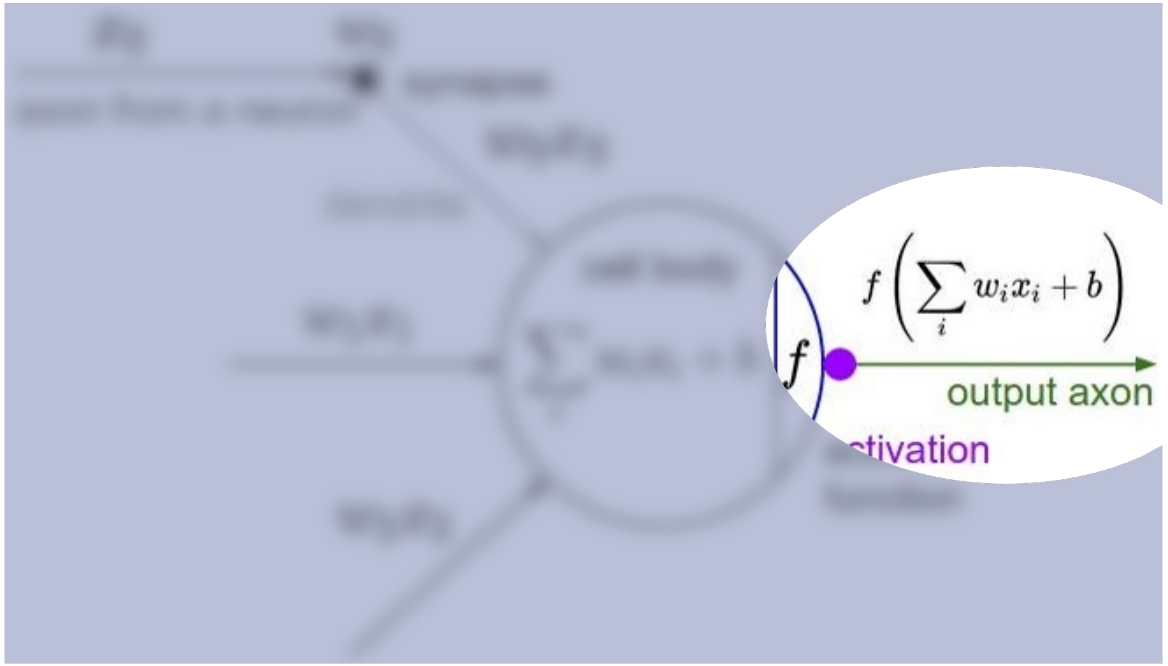
Input nodes:

- Input data
- No computation – information is passed to the next layer

Hiddent nodes:

- Intermediate processing
- Transfer of weights from the input to the following layer

Further details

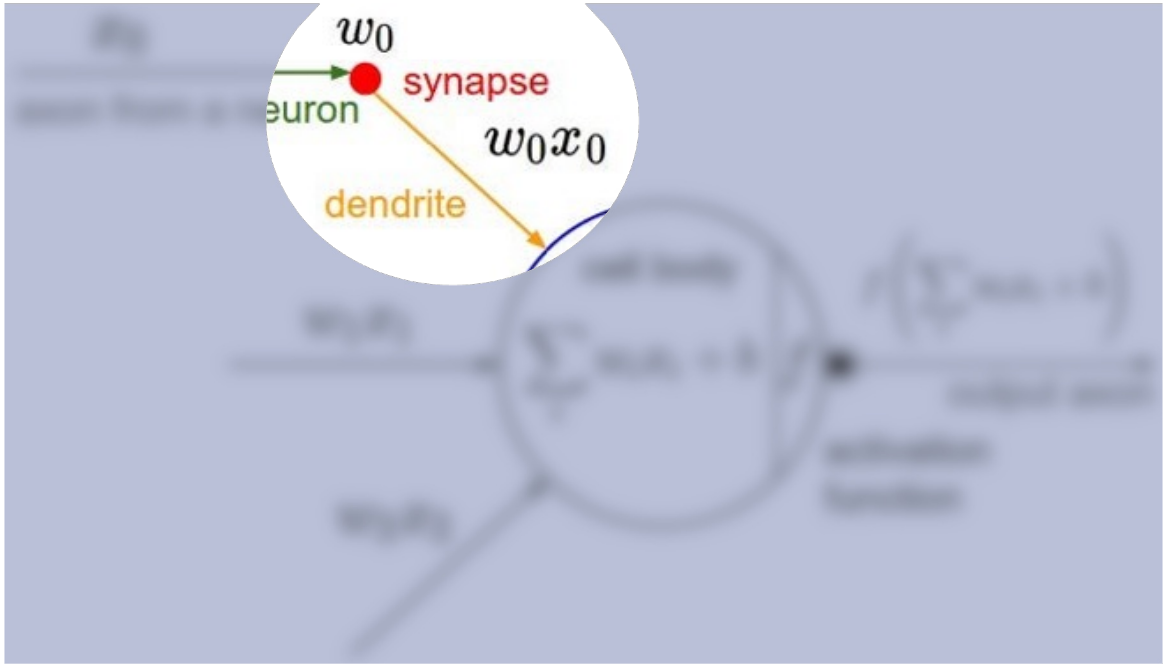


Input nodes:
- Input data
- No computation – information is passed to the next layer

Hiddent nodes:
- Intermediate processing
- Transfer of weights from the input to the following layer

Output nodes:
- The activation function maps to the output format

Further details



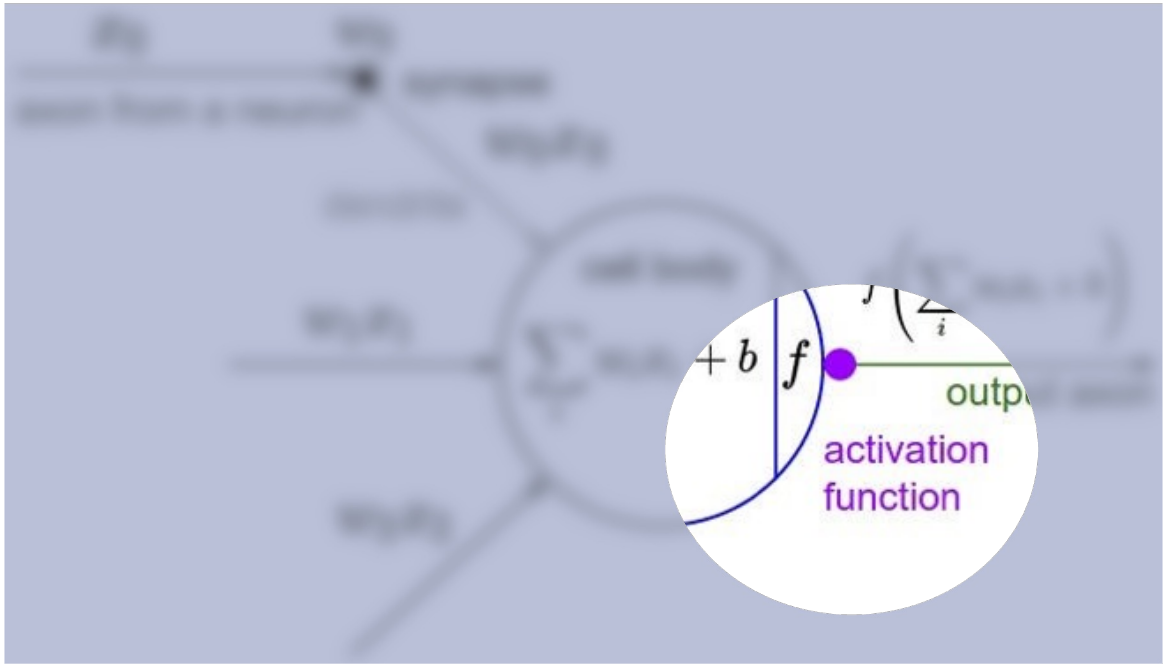
Input nodes:
- Input data
- No computation – information is passed to the next layer

Hiddent nodes:
- Intermediate processing
- Transfer of weights from the input to the following layer

Output nodes:
- The activation function maps to the output format

Connections and weights:
- Each connection transfers the output of one layer to another.

Further details



Input nodes:
- Input data
- No computation – information is passed to the next layer

Hiddent nodes:
- Intermediate processing
- Transfer of weights from the input to the following layer

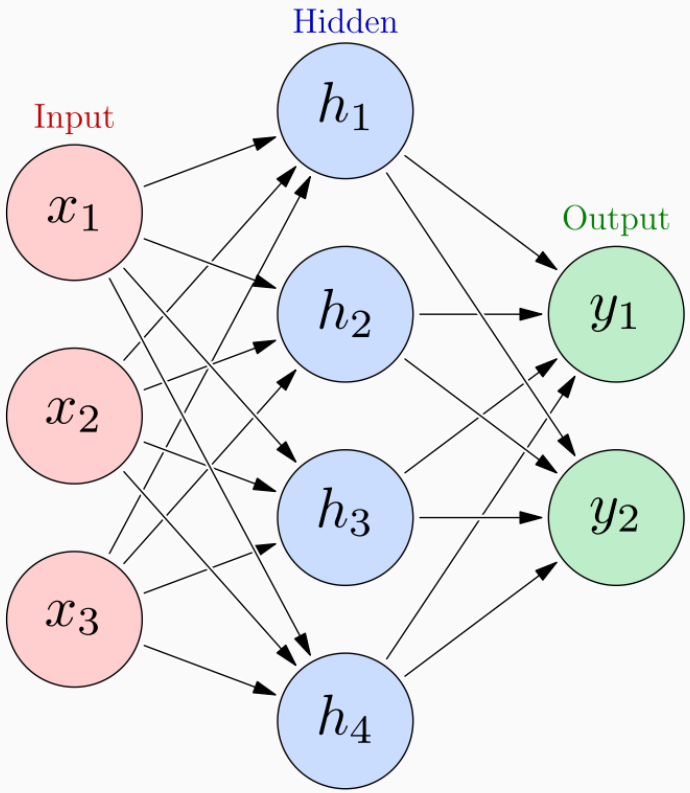
Output nodes:
- The activation function maps to the output format

Connections and weights:
- Each connection transfers the output of one layer to another.

Activation function:
- The function that defines the output of that node given an input (or set of inputs)

Further details

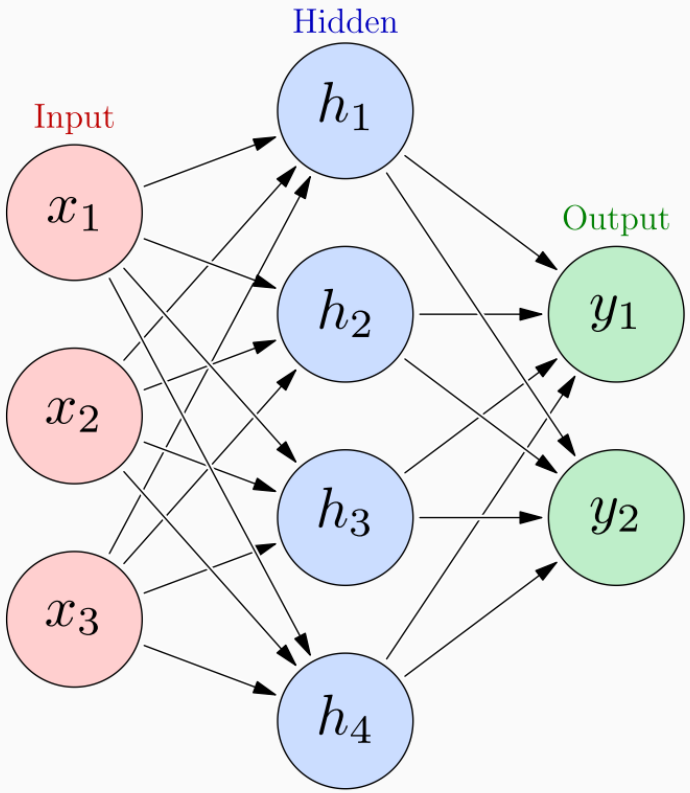
What does this look like to u?



$$h_1 = g_1\left(w_{11}^1 x_1 + w_{12}^1 x_2 + w_{13}^1 x_3 + b_1^1\right)$$
$$h_2 = g_1\left(w_{21}^1 x_1 + w_{22}^1 x_2 + w_{23}^1 x_3 + b_2^1\right)$$
$$h_3 = g_1\left(w_{31}^1 x_1 + w_{32}^1 x_2 + w_{33}^1 x_3 + b_3^1\right)$$
$$h_4 = g_1\left(w_{41}^1 x_1 + w_{42}^1 x_2 + w_{43}^1 x_3 + b_4^1\right)$$

$$y_1 = g_2\left(w_{11}^2 h_1 + w_{12}^2 h_2 + w_{13}^2 h_3 + w_{14}^2 h_4 + b_1^2\right)$$
$$y_2 = g_2\left(w_{21}^2 h_1 + w_{22}^2 h_2 + w_{23}^2 h_3 + w_{24}^2 h_4 + b_2^2\right)$$

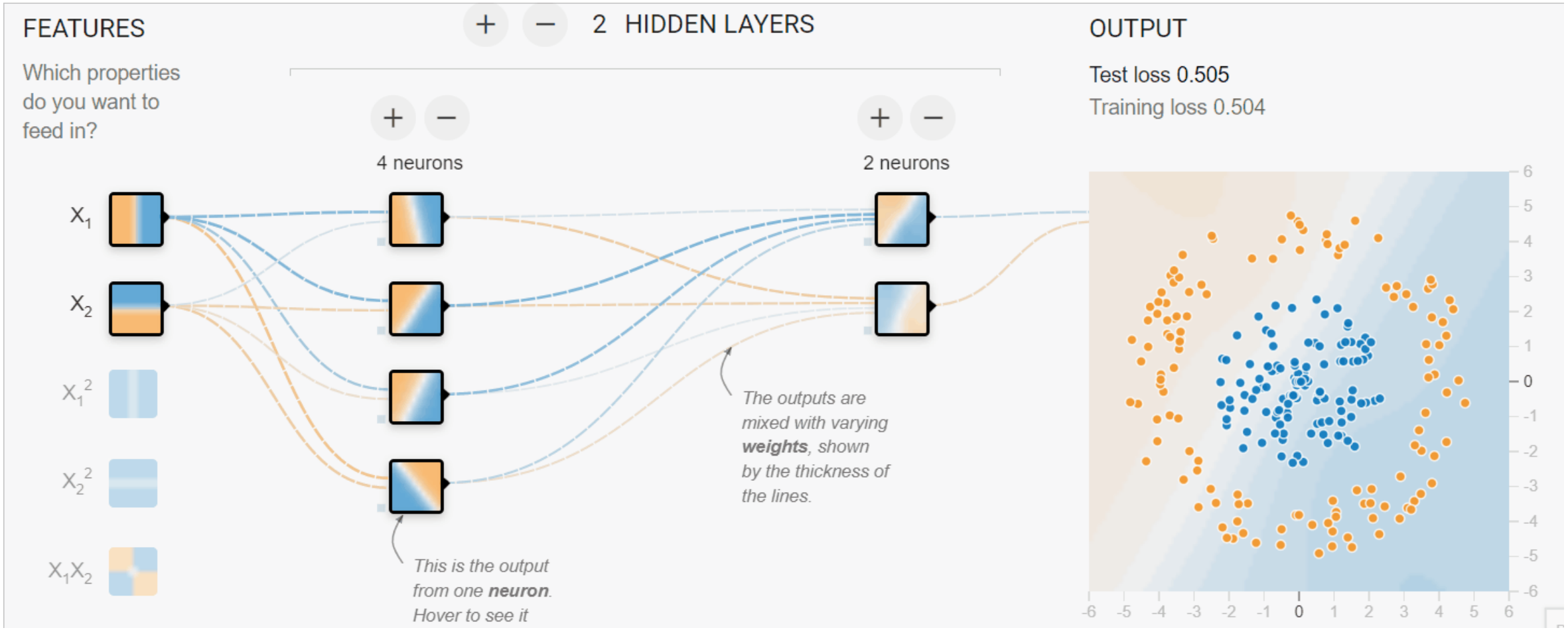
Further details



The NN computes with **the weights & biases (gs and bs)**

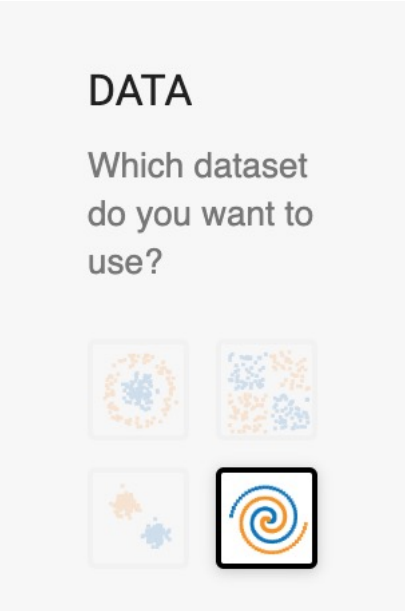
$$h_1 = g_1\left(w_{11}^1 x_1 + w_{12}^1 x_2 + w_{13}^1 x_3 + b_1^1\right)$$
$$h_2 = g_1\left(w_{21}^1 x_1 + w_{22}^1 x_2 + w_{23}^1 x_3 + b_2^1\right)$$
$$h_3 = g_1\left(w_{31}^1 x_1 + w_{32}^1 x_2 + w_{33}^1 x_3 + b_3^1\right)$$
$$h_4 = g_1\left(w_{41}^1 x_1 + w_{42}^1 x_2 + w_{43}^1 x_3 + b_4^1\right)$$

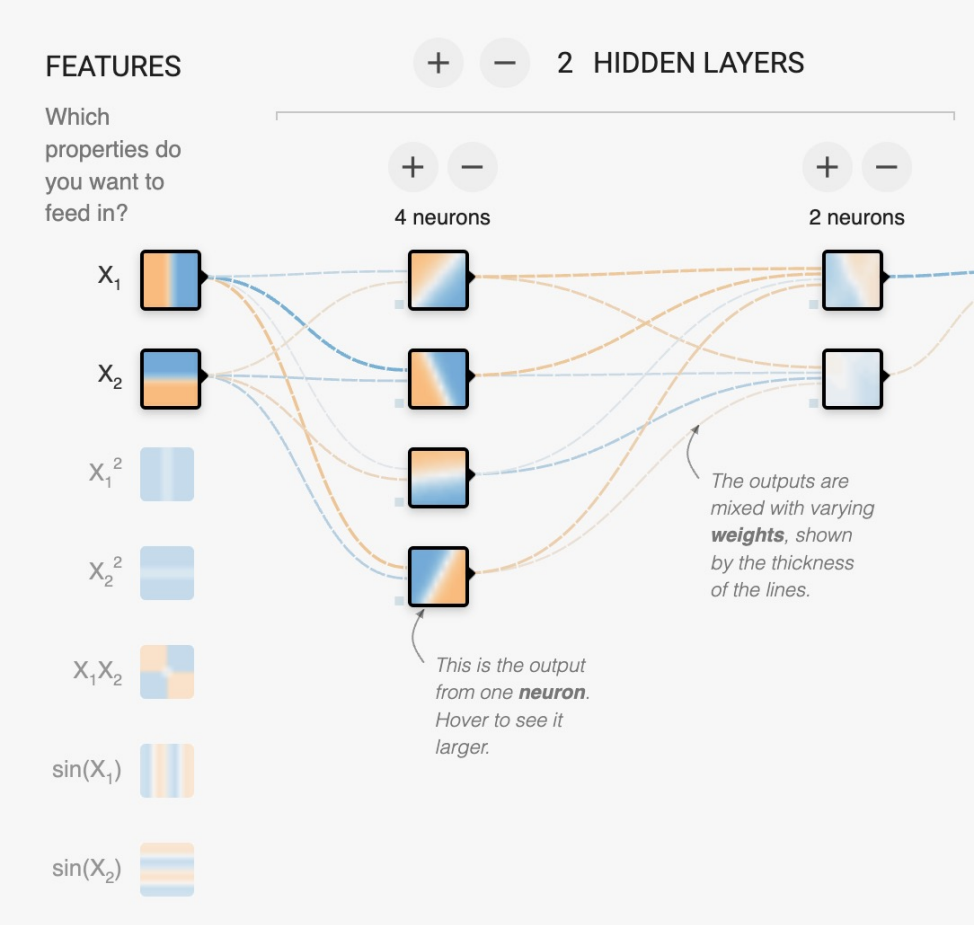
$$y_1 = g_2\left(w_{11}^2 h_1 + w_{12}^2 h_2 + w_{13}^2 h_3 + w_{14}^2 h_4 + b_1^2\right)$$
$$y_2 = g_2\left(w_{21}^2 h_1 + w_{22}^2 h_2 + w_{23}^2 h_3 + w_{24}^2 h_4 + b_2^2\right)$$

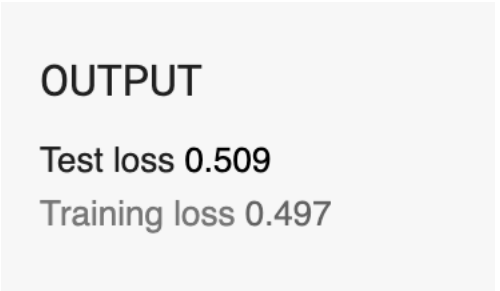

FEATURES
Which properties do you want to feed in?
2 HIDDEN LAYERS
4 neurons
2 neurons
X_1
X_2
X_1^2
X_2^2
X_1X_2
This is the output from one neuron. Hover to see it
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.505
Training loss 0.504

Now, you try ☺

https://playground.tensorflow.org/



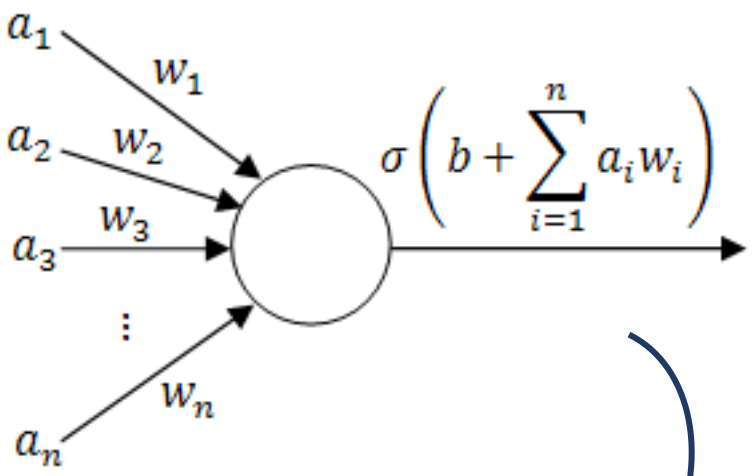




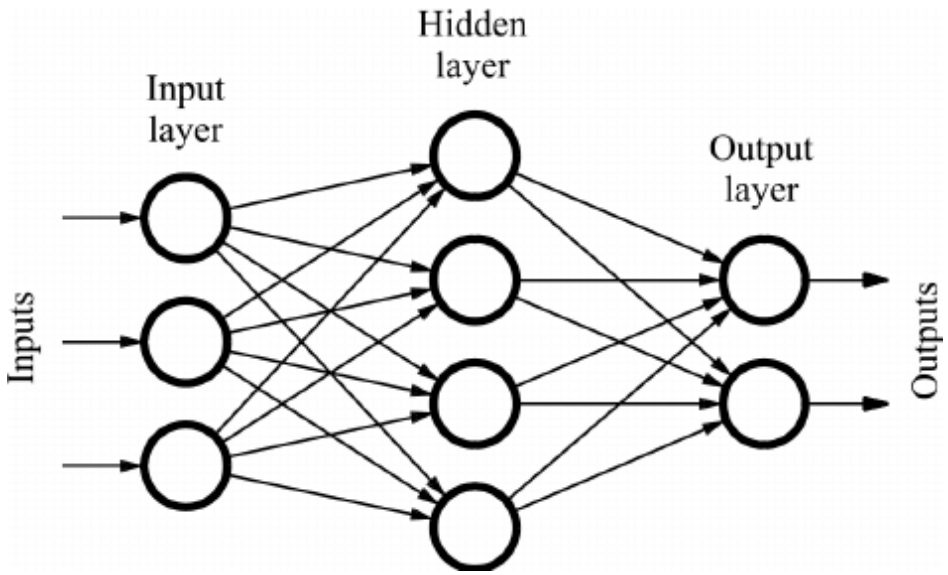
Lowest error **wins**!

Types of NN

- **Feedforward Neural Network** – no loop; information move in one direction

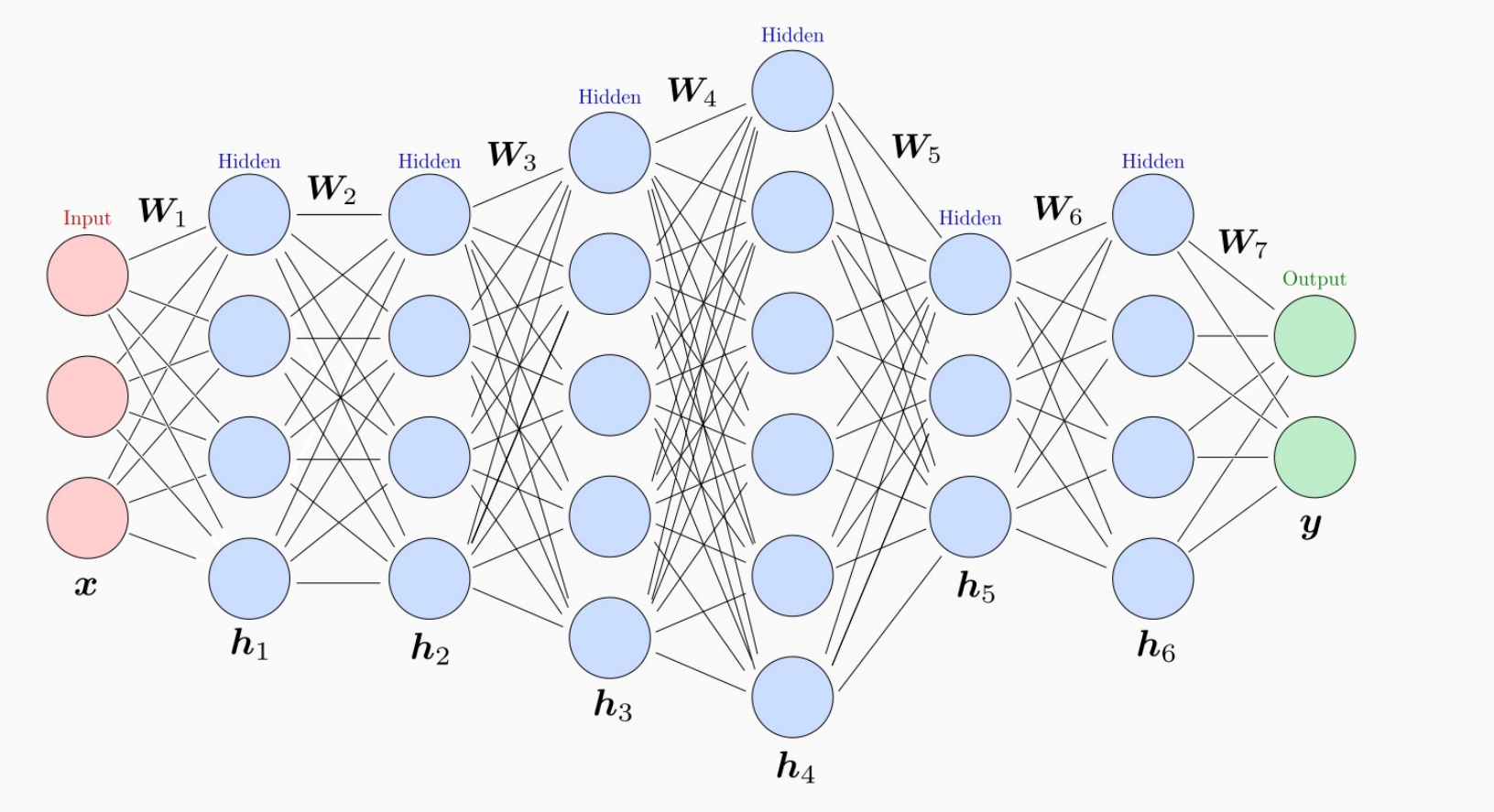


Single layer



Multi layer

Types of NN



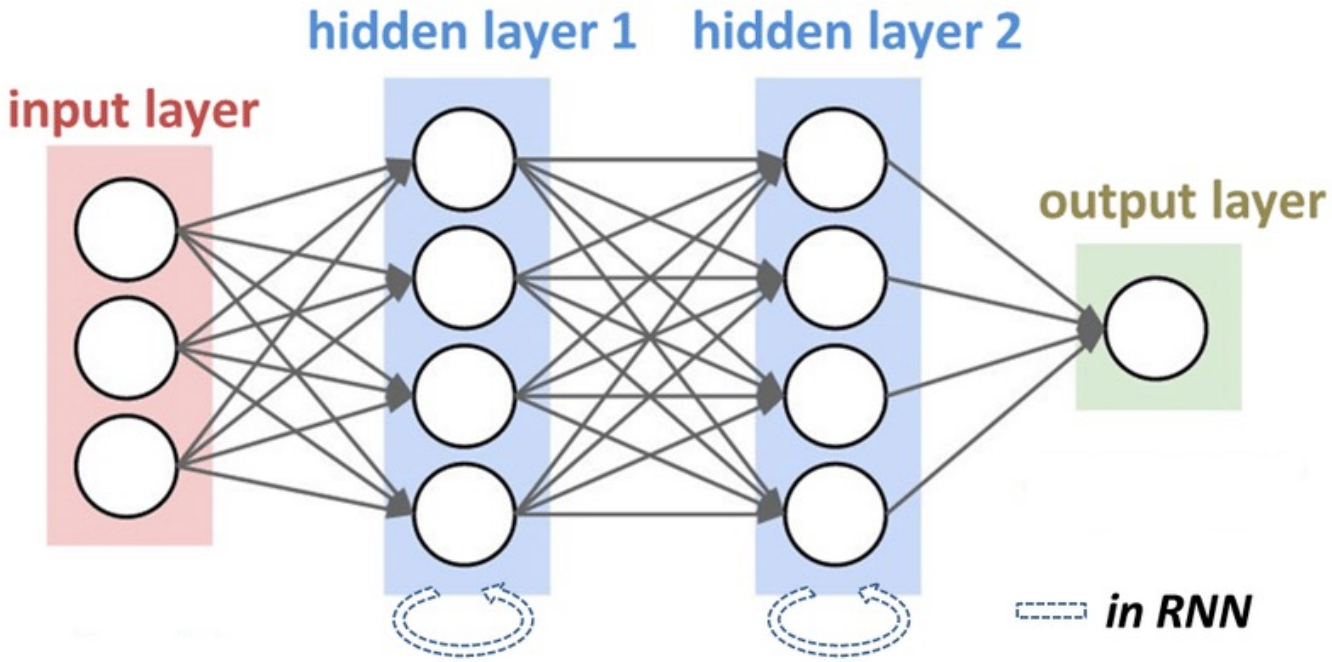
It can have 1 hidden layer only (**shallow network**) or many (**deep network**)

Each layer may have a **different size**

Hidden and output layers often have **different activation functions**

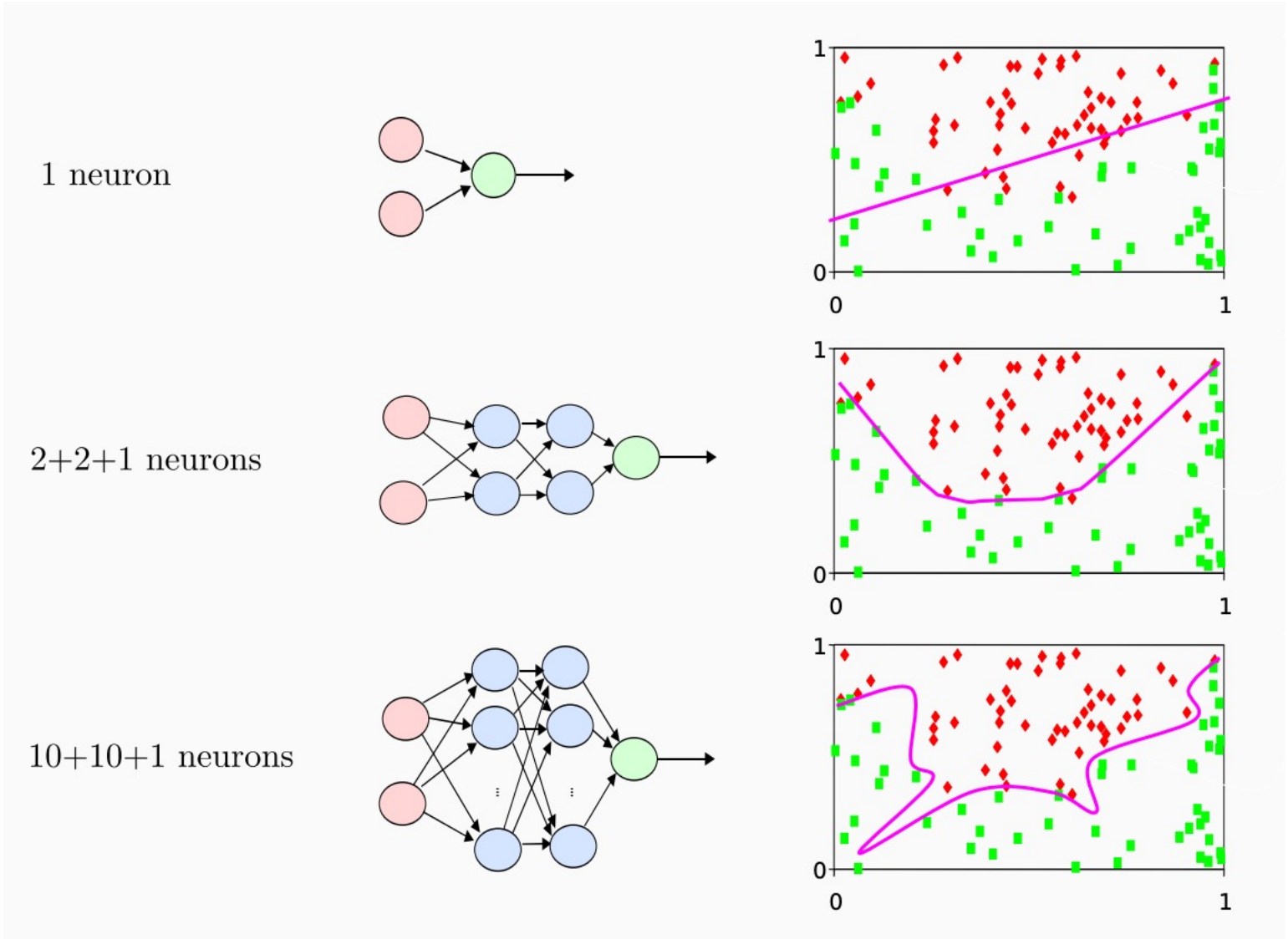
Types of NN

- **Recurrent neural networks** – loops; data propagates forward, but also backwards

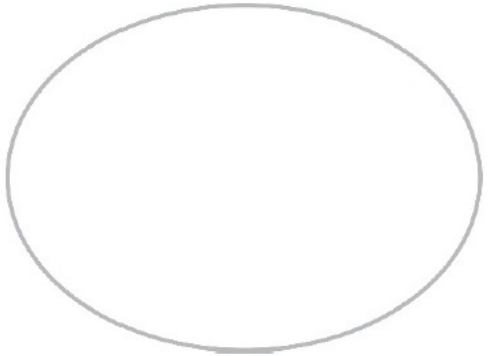
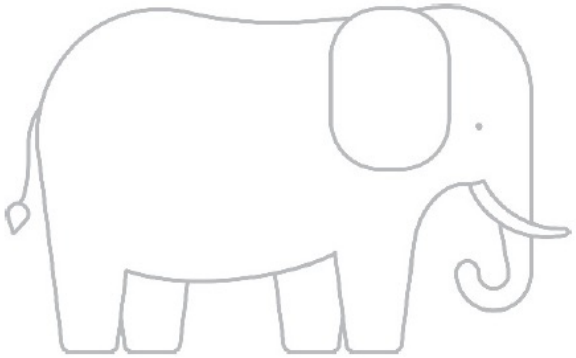
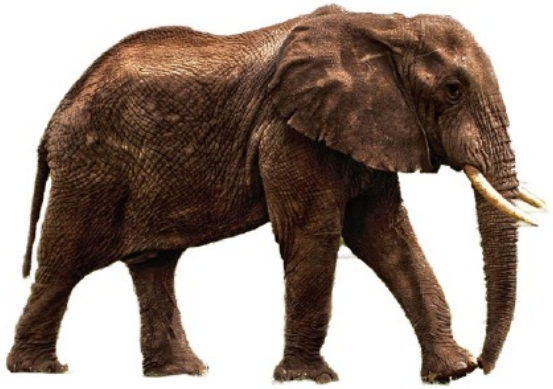


Architecture & relationships

- The architecture of the network defines the shape of the separator
- There is a trade off between generalization and overfitting

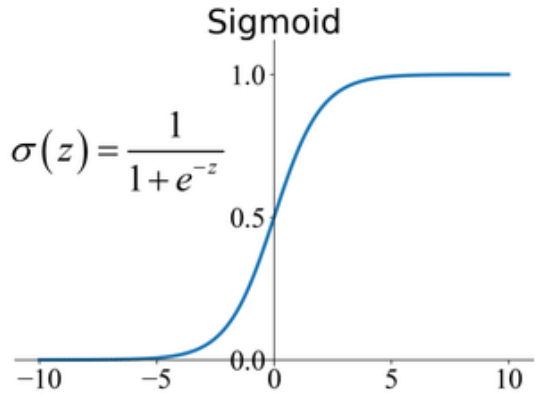


Recap: Generalizaton vs overfitting

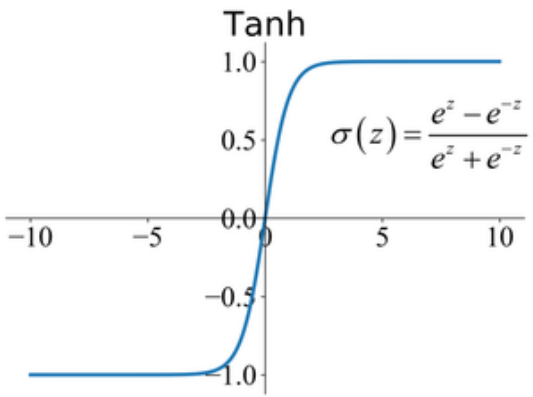
Underfitted	Generalized	Overfitted
		
Model overlooks underlying patterns in the training set	Model captures correlations in the training set	Model memorizes the training set rather then finding underlying patterns

Activation functions

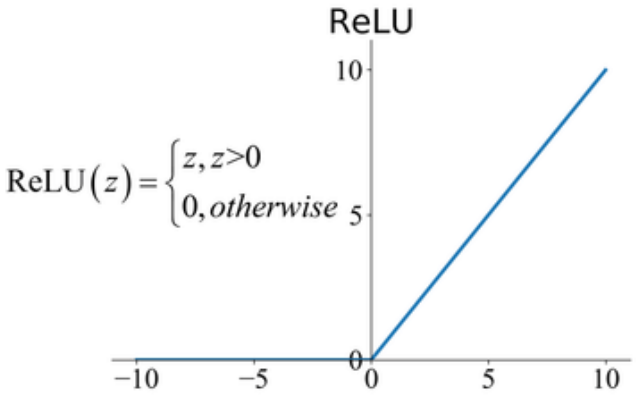
In general, **non-linearity is needed to learn complex (non-linear) representations of data**, otherwise the NN would be just a linear function.



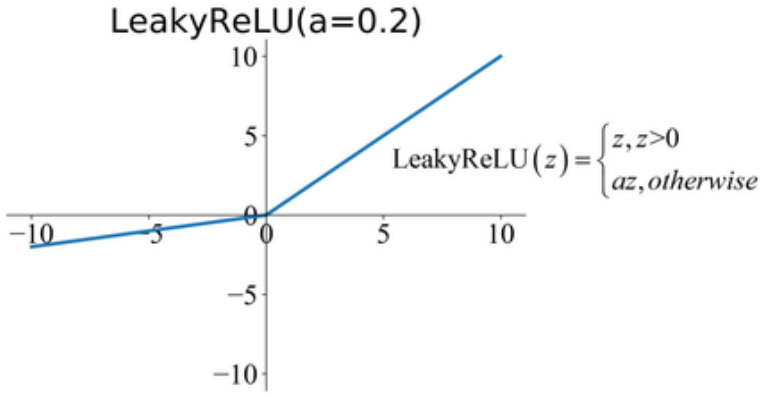
(a)



(b)



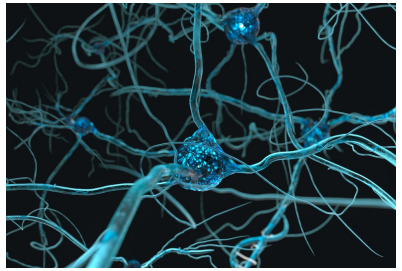
(c)



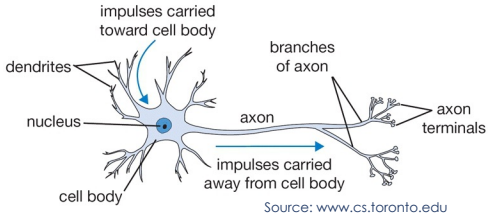
(d)

Timeline

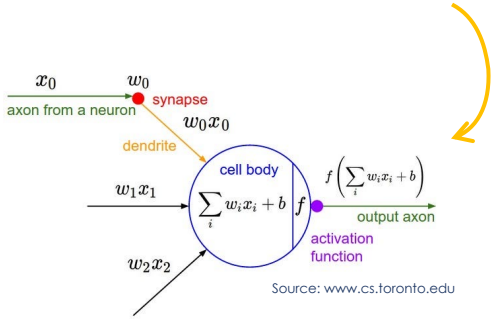
Source: The Birthplace of AI



A neural network is **made of neurons, connected through synapses where information flows.**

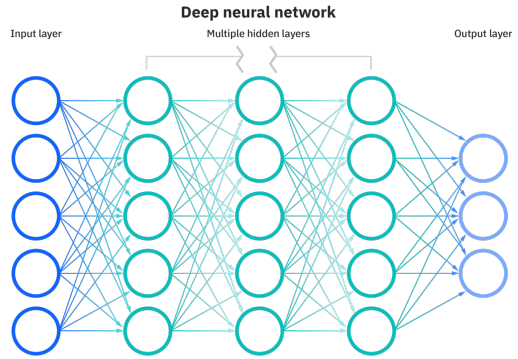


Source: www.cs.toronto.edu



Source: www.cs.toronto.edu

Can represent wide variaty of factions; can have many layers, thus learn patterns at different levels of abstraction; can have millions/billions adjustable parameters …

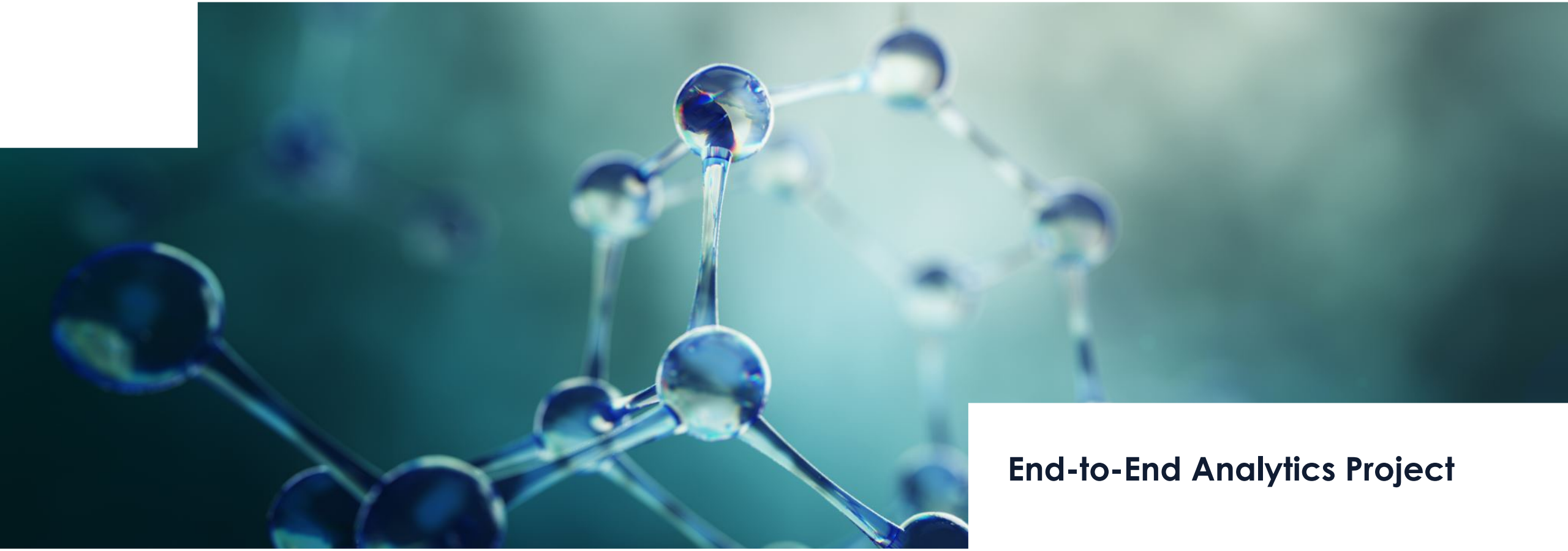


Source: https://www.ibm.com/topics/neural-networks

1950 – 1980s
Early Foundations and Symbolic AI

1980 – 2000s
The Rise of Neural Networks

2010s
Deep Learning Revolution



End-to-End Analytics Project

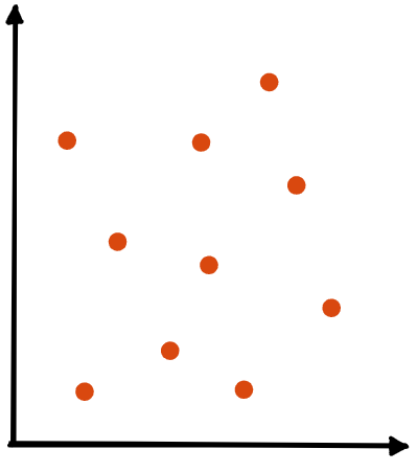
Step 1: **Data Exploration & Processing**

—

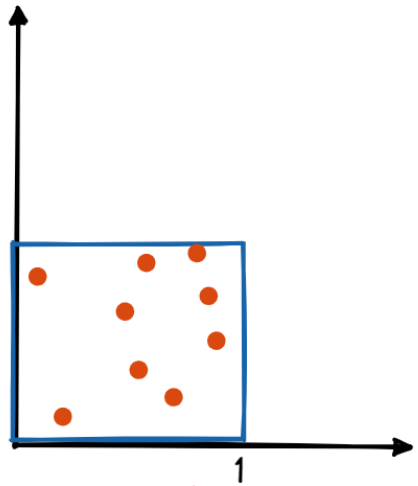
- **Describe the data:**
  - Which variable do we want to predict? **This is our Y**
  - Which variable(s) do we believe cause or impacts changes is our Y? **This is our X(s)**
  - Investigate numeric and categorical variables
    - Not all models work with all types of features – make sure that the data types are appropriate for the models you want to train
  - In the case in which we have a classification task: Check how represented each class is
  - Check for NAs
  - Check for outliers
- **Pre-processing:**
  - Define the appropriate data types
  - Deal with NAs
  - Deal with outliers (column-wise or row-wise; substitute)
  - Deal with class imbalances (undersampling vs oversampling)
  - Scale the data

Scaling

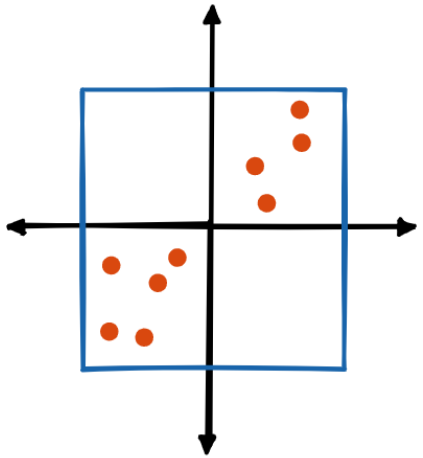
- Ensures that the features used in the model have a similar range of values.
- Without scaling, some features may dominate the model while others are ignored, **leading to biased or inaccurate results**.



Actual Data



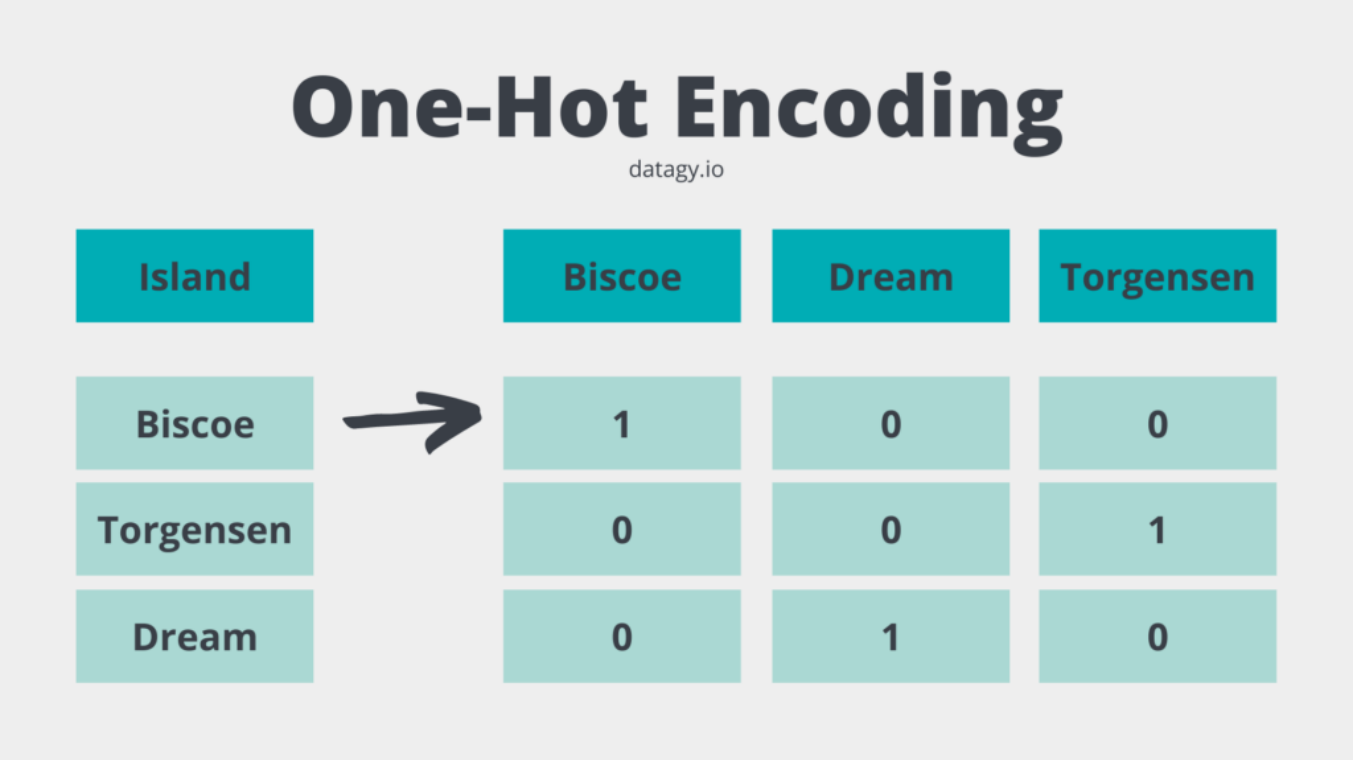
Normalization



Standardization

Dealing with **non-numeric** data types

- **Some models DO NOT work with non-numeric data**
- **One solution: One-hot encoding**
  - Important techniques that allows categorical data to be represented in a format that can be used by machine learning algorithms
  - It ensures that each category is treated as a separate feature, which can help to prevent bias in the model.
  - It can improve the performance of the model by reducing the impact of irrelevant or noisy features

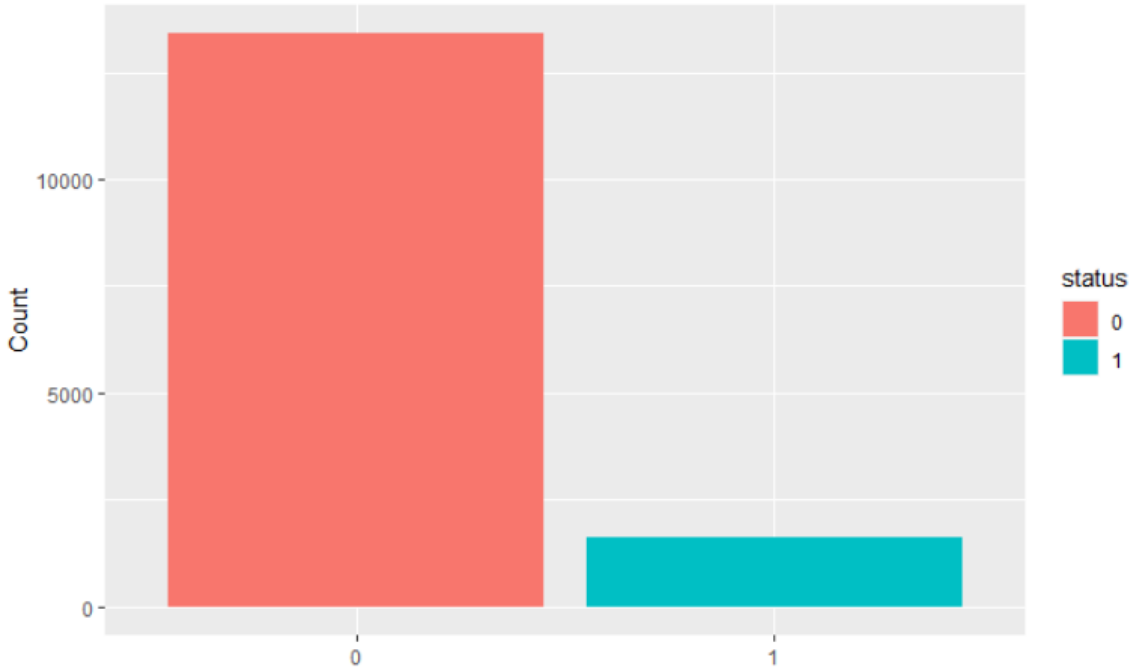


Class **imbalances**

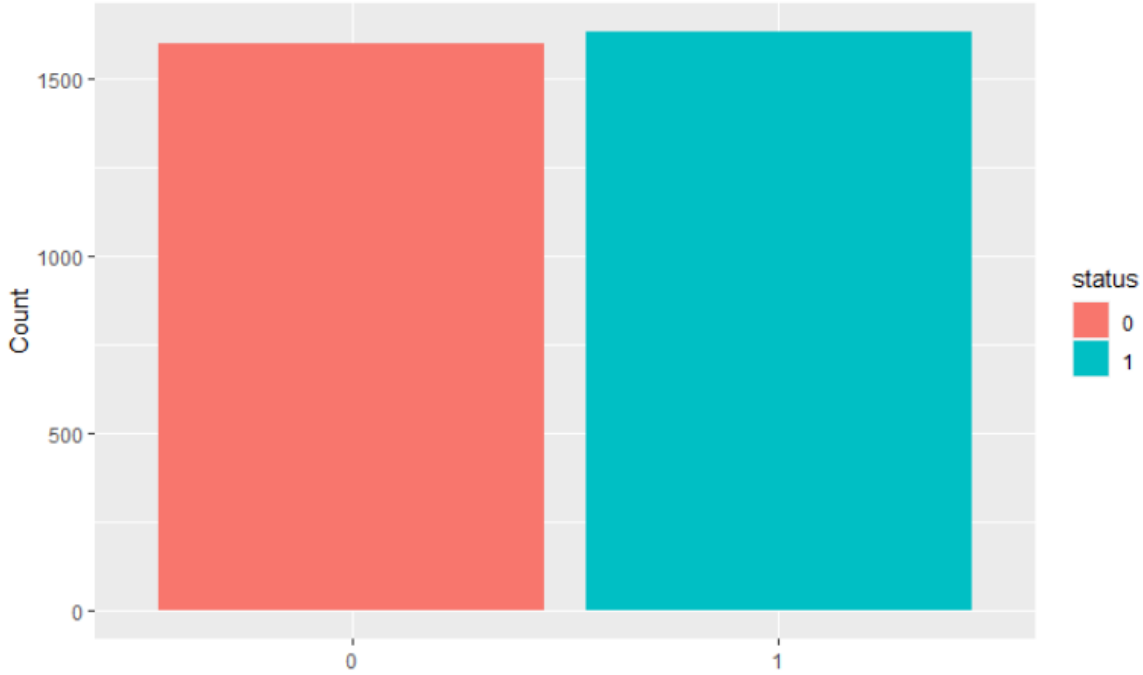
- Why is this an issue?
  - If I tell you that a model is **90% accurate**, would you be satisfied by that?
  - Now – what if the model just predicts the "**majority class**" for every new observation?
  - Considering the distribution to the left – this would lead to a 90% accurate model that cannot predict the "**default**" event.

Class **imbalances**

Imbalanced classes



Undersampling



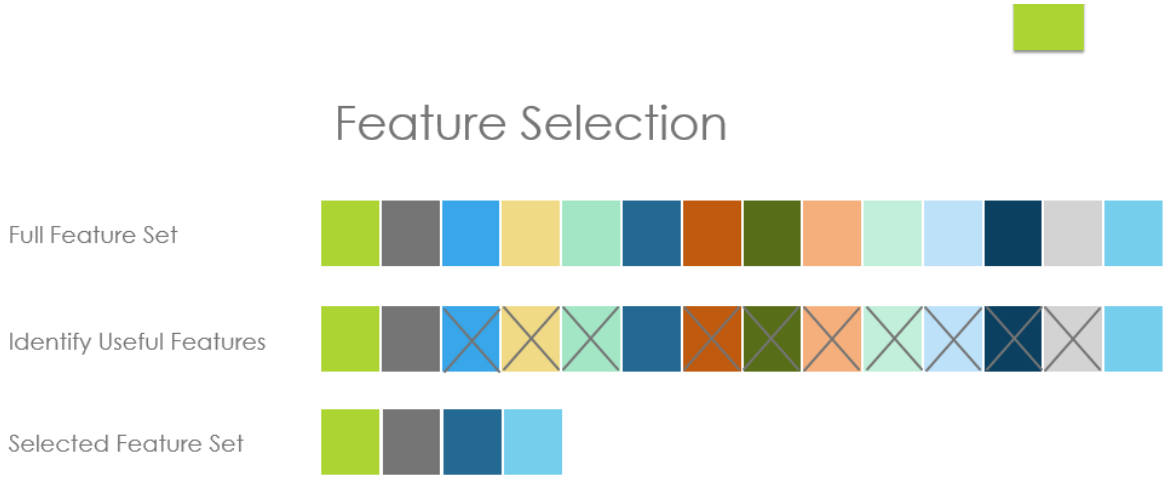
Feature selection

- **Feature selection** -- the process that chooses an optimal subset of features according to a certain criterion.

- **Why do we need it?**

  - To reduce the dimensionality and complexity of our data

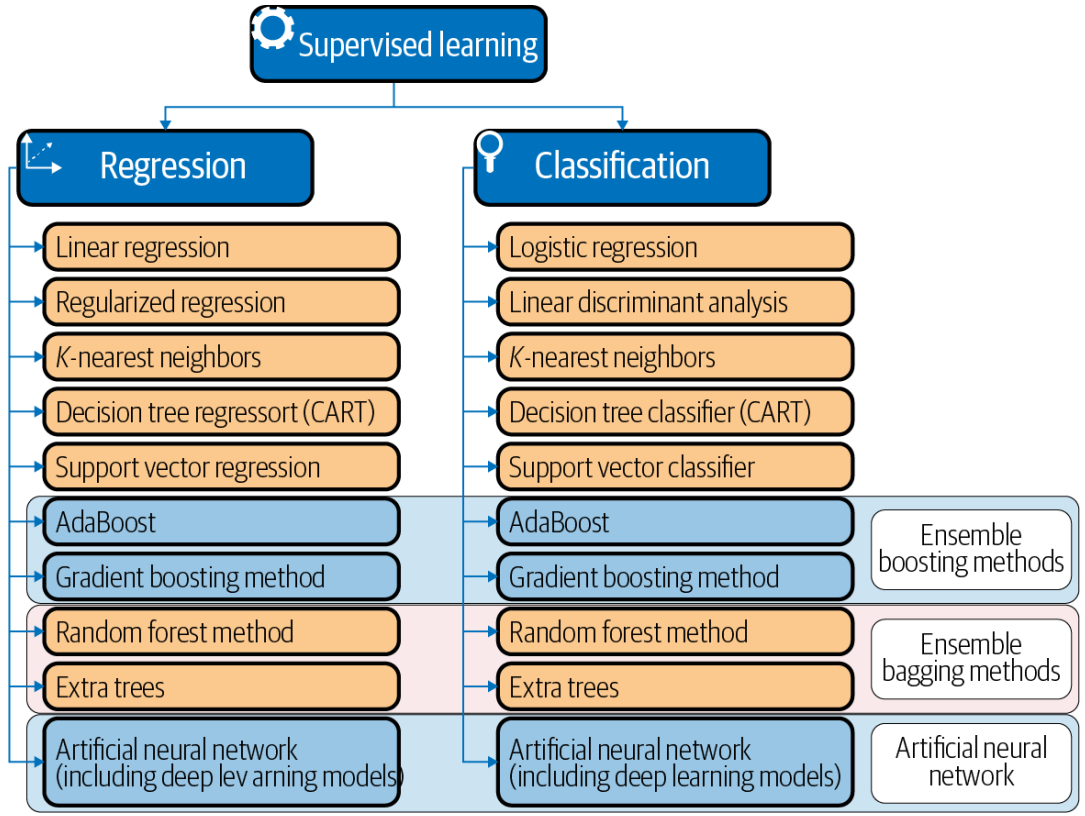
  - To improve performance (in terms of speed, predictive power, simplicity of the model).

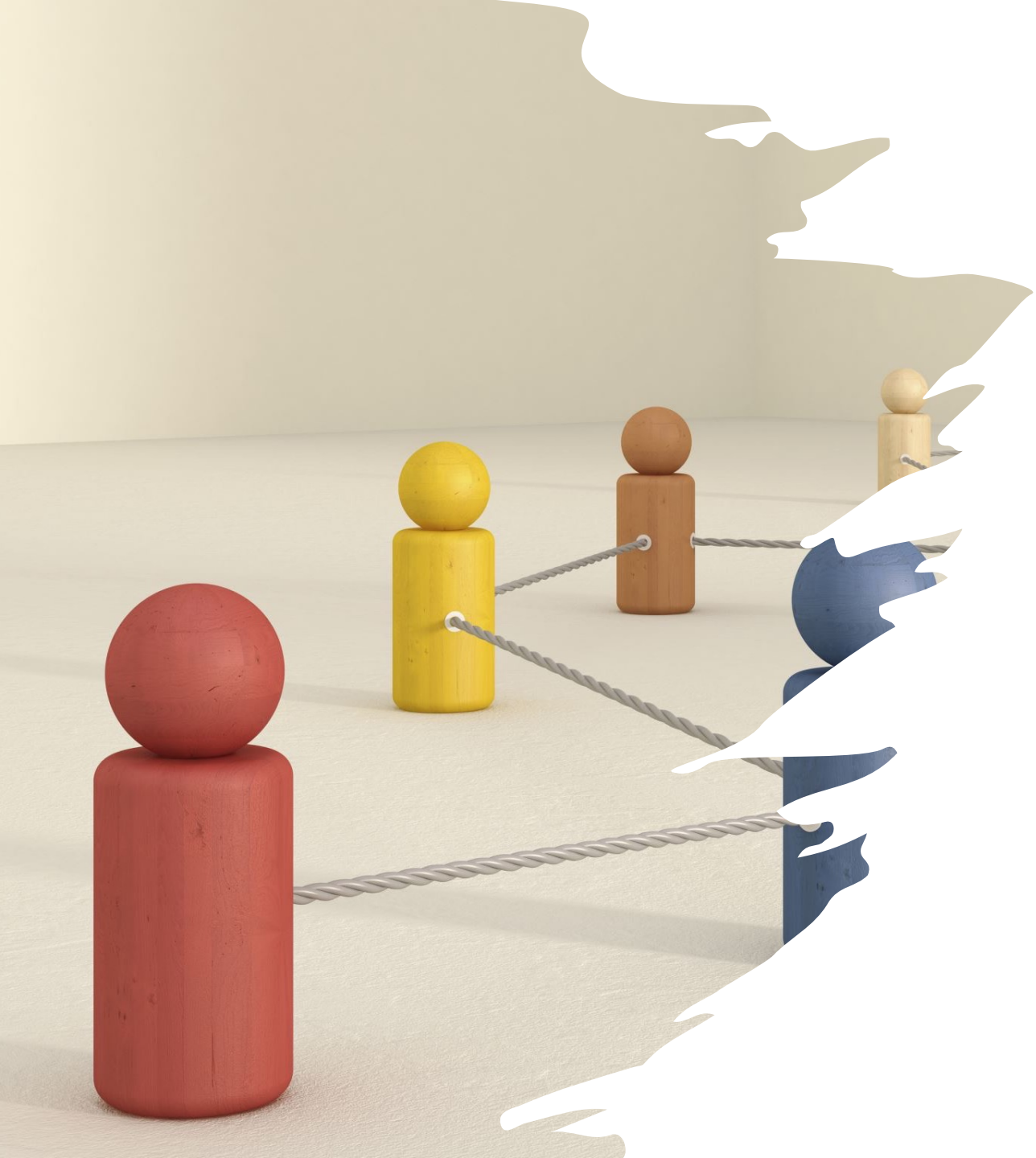


Step 2: **Model selection & Training**

- What models can you use for the task at hand?
  - Is this a **classiciation or regression task**?
  - What **type of a relationship do you expect between your Y and X**? Is it linear?
  - ….

- **Ultimtaly:** You select the model that performs best for your given problem set





Step 3: **Testing preformance**

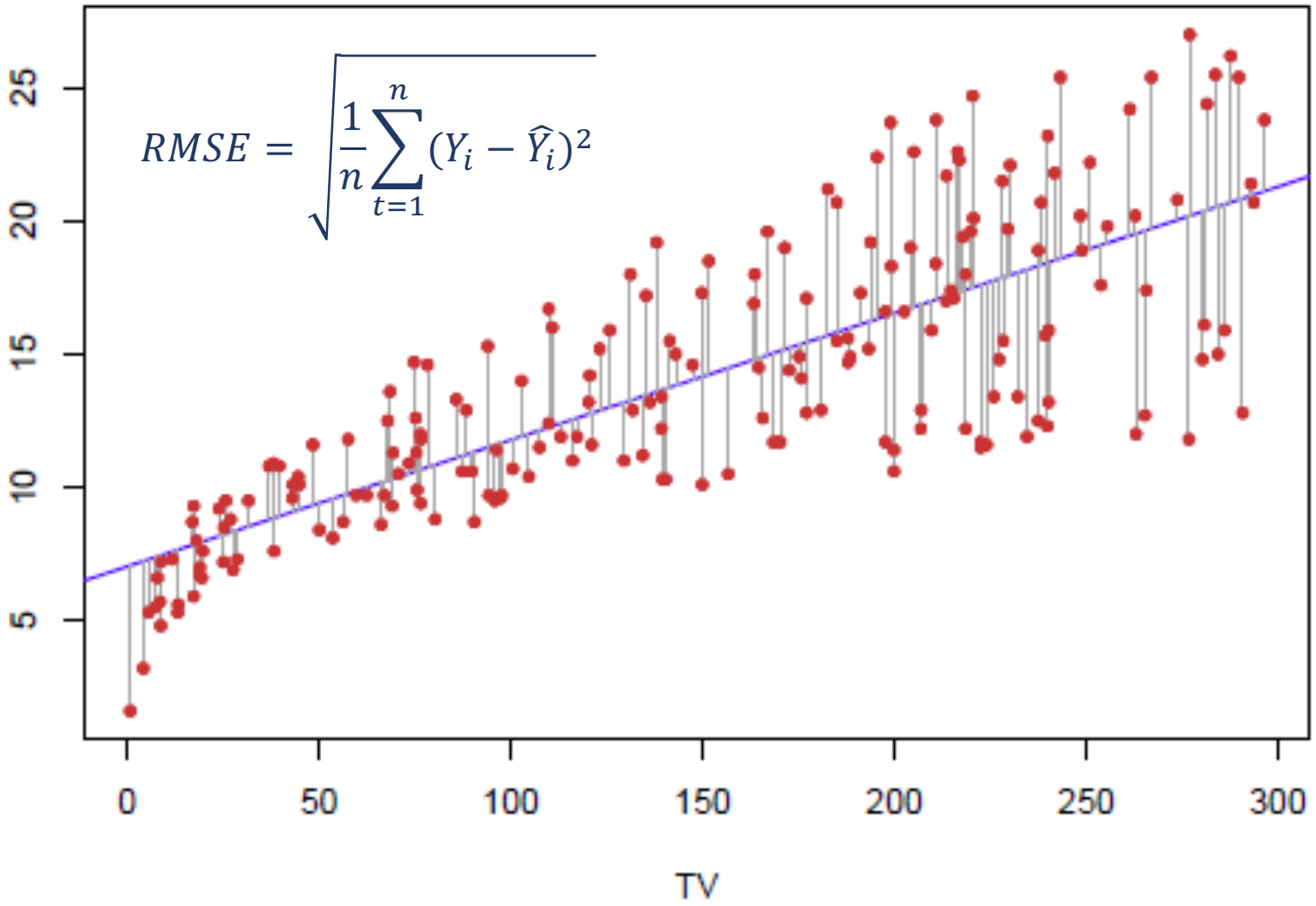
- **Why evaluation metrics?**
  - Quantify the power of a model
  - Compare model configurations and/or models, and select the best performing one
  - Obtain the expected performance of the model for new data

- **Different model evaluation techniques are available for:**
  - Classification/regression models
  - Imbalanced/balanced target class distributions

Testing preformance: **Regression models**

- Comparison between the **real and predicted values**

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(Y_i - \hat{Y}_i)^2}$$



Testing preformance: **Classification models**

- The **overall accuracy** of the model can be computed as:

$$\textbf{ACC} = TP + TN/TP + TN + FP + FN$$

Where:

- **True Positive (TP):** Actual and predicted class is positive
- **True Negative (TN):** Actual and predicted class is negative
- **False Negative (FN):** Actual class is positive and predicted negative
- **False Positive (FP):** Actual class is negative and predicted positive

Downsides:

- Only considers the performance in general and not for the different classes
- Therefore, not informative when the class distribution is unbalanced

Testing preformance: **Classification models**

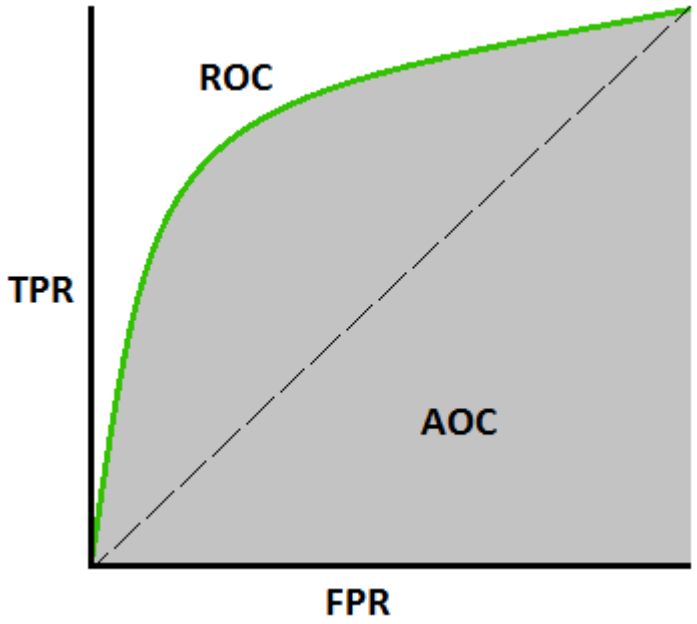
- Confusion matrix

Predicted Class \ Actual Class	**Defaulted**	**Non-Defaulted**
Defaulted	Correct call *True Positive (TP)*	False alarm *False Positive (FP)*
Non-Defaulted	Missed crisis *False Negative (FN)*	Correct silence *True Negative (TN)*

- **Rows** – predicted class values
- **Columns** – predicted class values
- **Numbers on main diagonal** – correctly classified samples
- Numbers off the main diagonal – misclassified samples

Testing preformance: **Classification models**

- **The ROC Curve** shows the false positive rate and true positive rate for different threshold values:
  - **False positive rate (FPR)**
    - negative events incorrectly classified as positive
  - **True positive rate (TPR)**
    - positive events correctly classified as positive
- **AUC** – Area under the ROC curve is a performance measurement for classification problem at various thresholds settings
  - Range 0 to 1
  - Closer to 1 it is, the better the classifier is at identifying 0s as 0s and 1s as 1s



True Positive Rate

$$\text{TPR} = \frac{TP}{TP+FN}$$

False Positive Rate

$$\text{FPR} = \frac{FP}{FP+TN}$$

Let’s SEE it!