Data Mining & Informatics Lecture #3

Amin Noroozi
University of Wolverhampton
✉ a.noroozifakhabi@wlv.ac.uk
in https://www.linkedin.com/in/amin-n-148350218/

NOTE

- **Extra lessons**

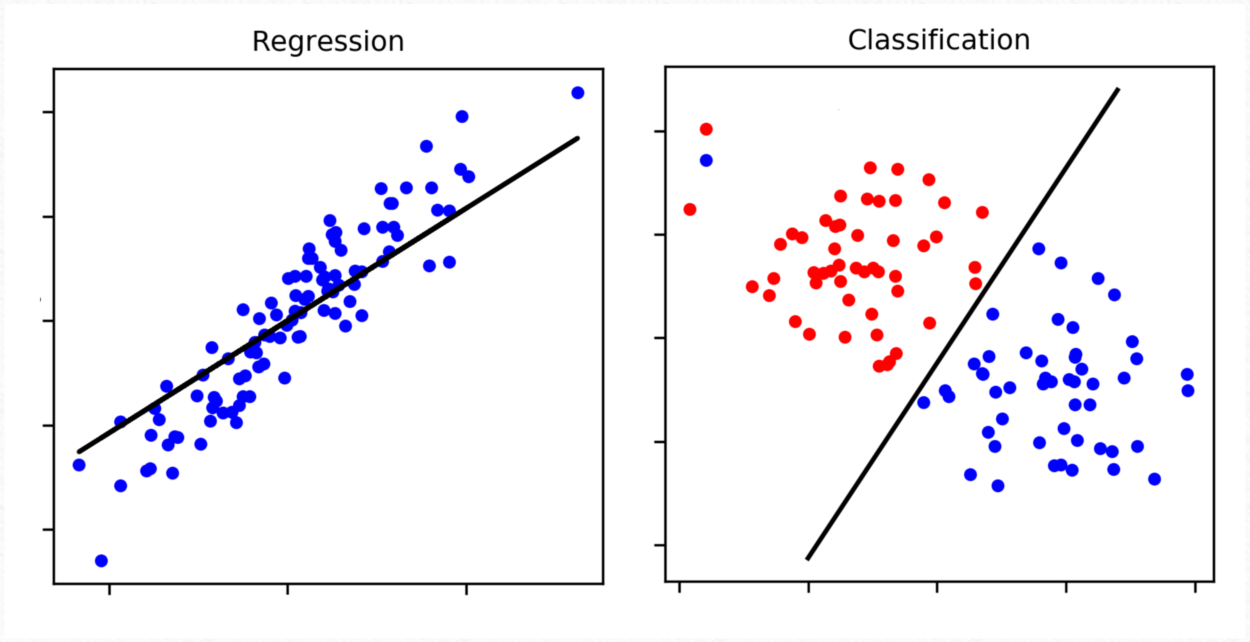
- Extra lessons will be on Tuesday 5-7 pm

- The first class will be next week and will be dedicated to Python basics

Classification

Supervised learning

- **Introduction**
- There are two main supervised learning methods: Classification and Regression



Supervised learning

- **Introduction**
- In all supervised learning methods there are 3 steps:

- Defining the model: In this step, the model is defined and hyperparameters are set.

- Fitting the model (training): In this step, the training data are used to find the optimum model parameters using a learning algorithm (for example gradient descent in neural network)

- Testing the model (prediction): after setting the model hyperparameter and finding the model's parameter in the training state, in this step, test data are used to check how accurate the model's predictions are.

Supervised learning

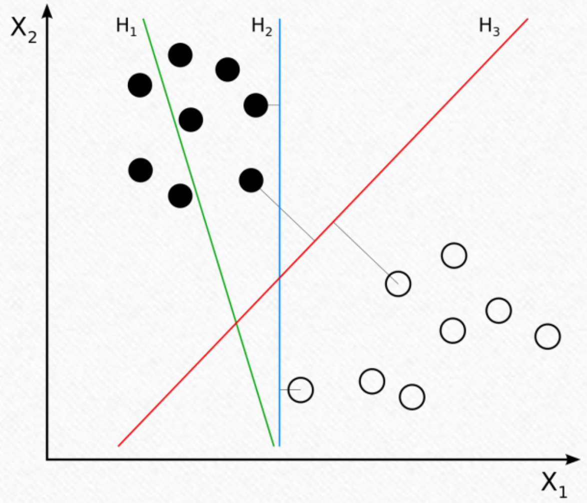
- **Classification**
- Classification is a supervised learning method in which the target variables (the variables that need to be predicted) are categorical.

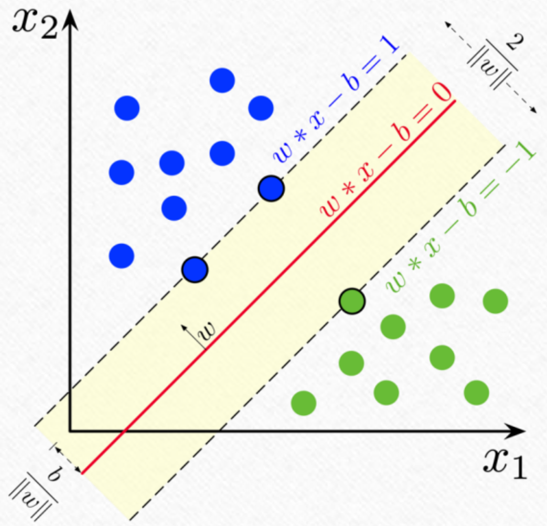
- If there are only two categories, it is called binary classification. If there are more than two categories, it is called multiclass classification.

- There are different classification methods. Here we discuss support vector machine (SVM), decision tree (DT), random forest (RF), and K-nearest neighbor (KNN) classifiers.

Supervised learning

- **Classification - SVM**
- The objective is to find a hyperplane in an n-dimensional space that separates the data points to their potential classes. The hyperplane should be positioned with the maximum distance to the data points
- The data points with the minimum distance to the hyperplane are called Support Vectors





Supervised learning

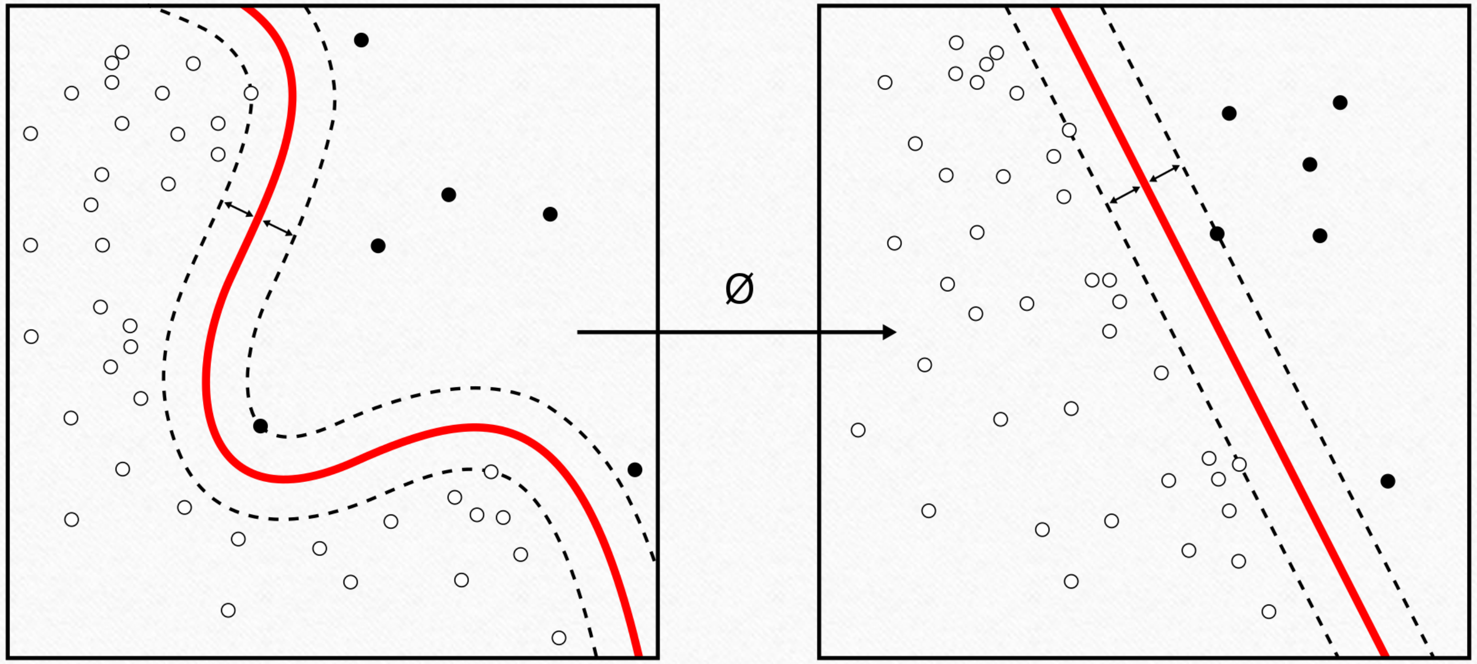
- **Classification - SVM**
- The original maximum-margin hyperplane algorithm proposed by Vapnik in 1963 constructed a linear classifier that works when data are linearly separable (using a line or hyperplane)

- For data that are not linearly separable, we need to use a kernel function and transform data to another space where they are linearly separable. This is called the kernel trick.

- There are different kernel functions including linear function, Polynomial function, Radial basis function (RBF), and Sigmoid function.

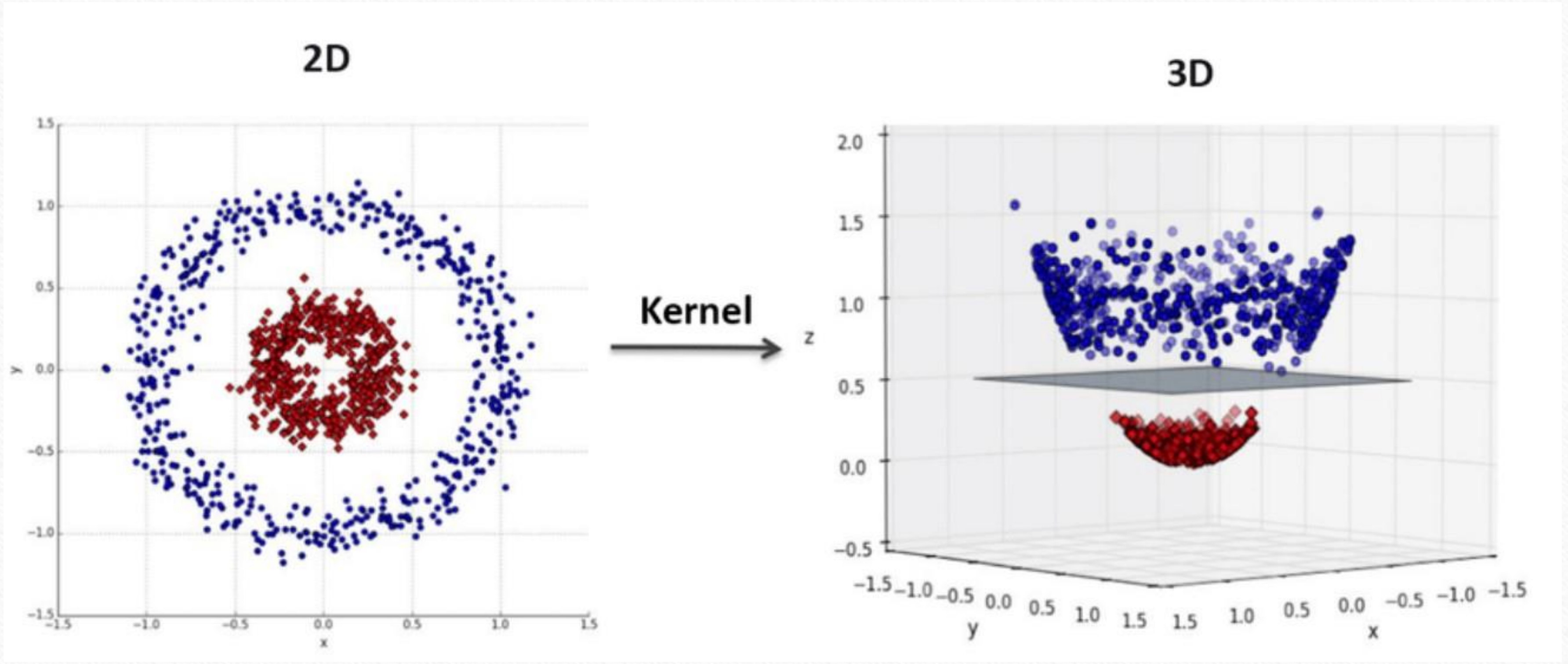
Supervised learning

- **Classification - SVM**



Supervised learning

- **Classification - SVM**



Supervised learning

- **Classification - SVM**

- There are different kernel functions, including linear function, Polynomial function, Radial basis function (RBF), and Sigmoid function.

- The kernel type is a hyperparameter and should be selected using a trial and error procedure.

- The SVM classification is used for binary classification by definition.

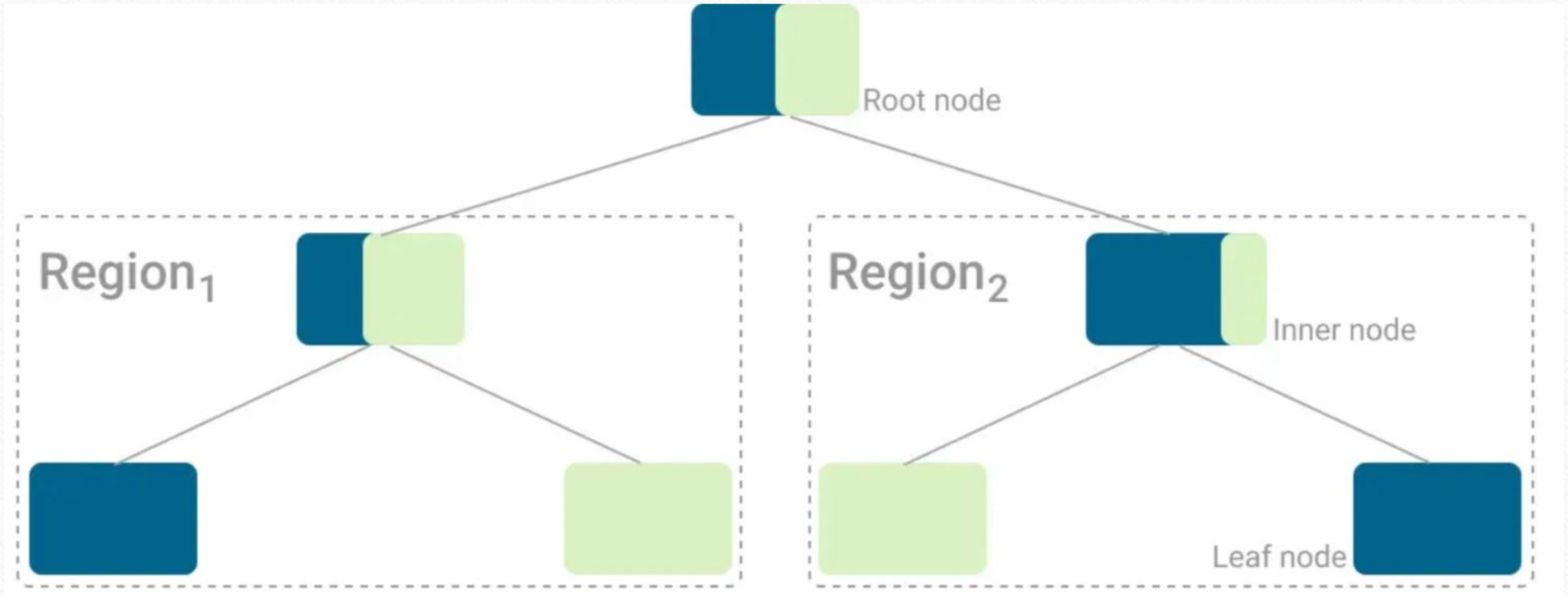
Supervised learning

- **Classification – Decision tree**

- Decision Tree is a Supervised Machine Learning Algorithm that uses a set of rules to make decisions, similar to how humans make decisions.
- The intuition behind Decision Trees is that you use the dataset features to create yes/no questions and continually split the dataset until you isolate all data points belonging to each class.
- Every time you ask a question you're adding a **node** to the tree. And the first node is called the **root node**. Each node is connected to other nodes using **branches**.
- If you decide to stop the process after a split, the last nodes created are called **leaf nodes**.
- Every time you answer a question, you're also creating branches and segmenting the feature space into disjoint **regions**.

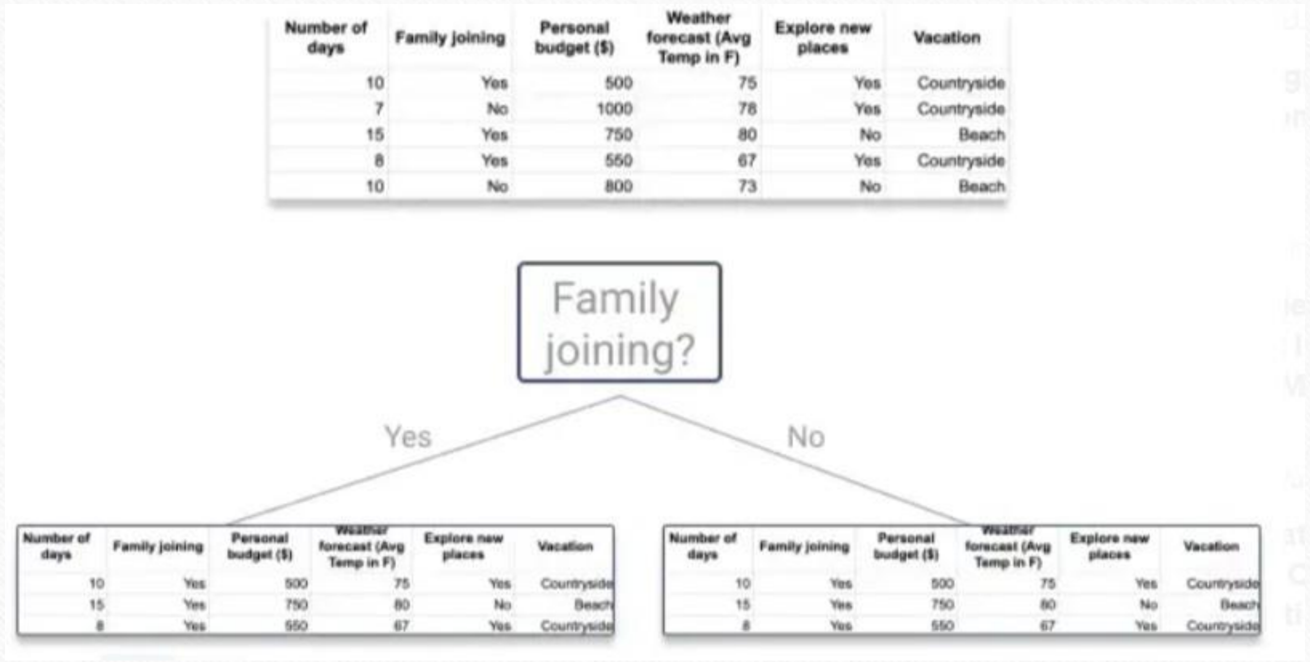
Supervised learning

- **Classification – Decision tree**



Supervised learning

- **Classification – Decision tree**



Number of days	Family joining	Personal budget ($)	Weather forecast (Avg Temp in F)	Explore new places	Vacation
10	Yes	500	75	Yes	Countryside
7	No	1000	78	Yes	Countryside
15	Yes	750	80	No	Beach
8	Yes	550	67	Yes	Countryside
10	No	800	73	No	Beach

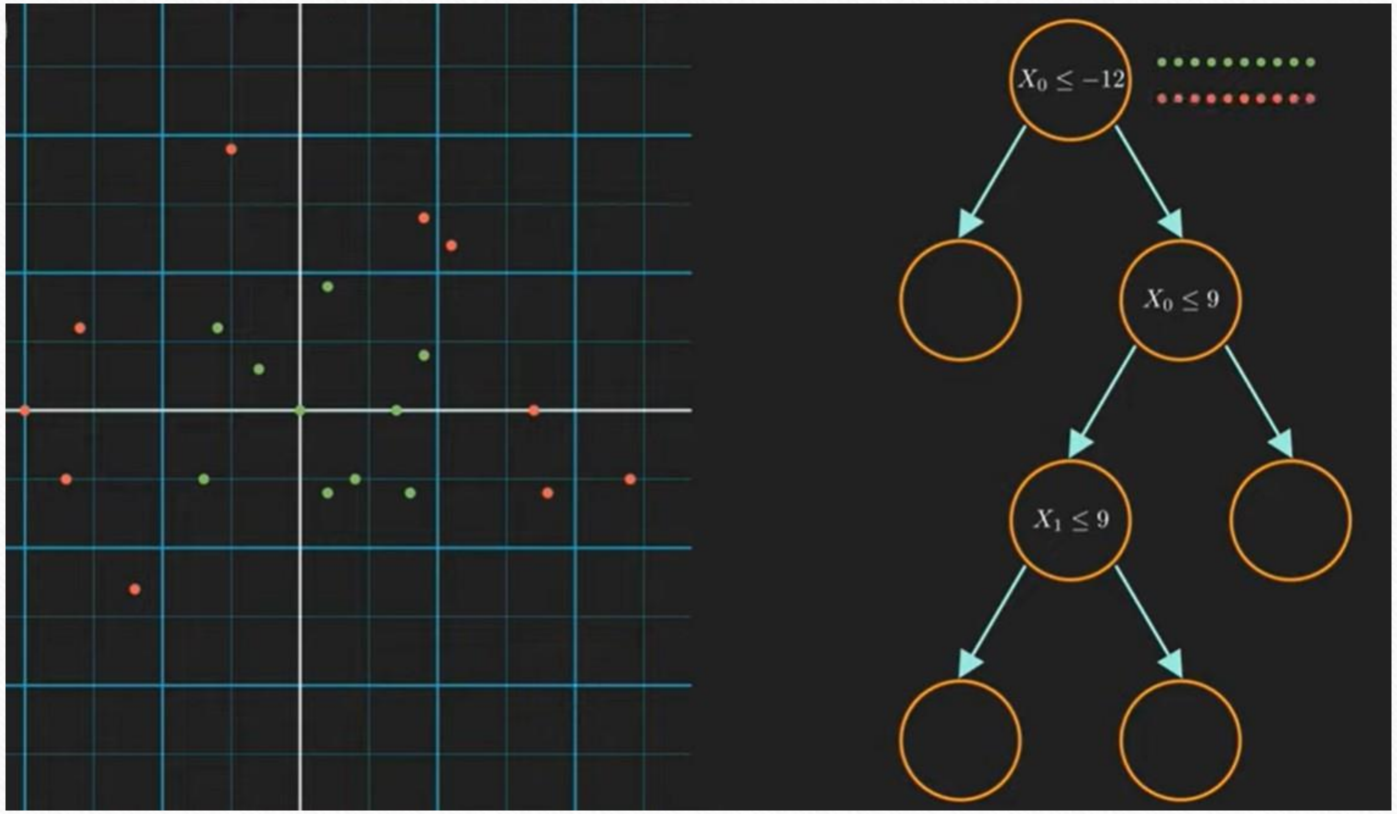
Number of days	Family joining	Personal budget ($)	Weather forecast (Avg Temp in F)	Explore new places	Vacation
10	Yes	500	75	Yes	Countryside
15	Yes	750	80	No	Beach
8	Yes	550	67	Yes	Countryside

Number of days	Family joining	Personal budget ($)	Weather forecast (Avg Temp in F)	Explore new places	Vacation
10	Yes	500	75	Yes	Countryside
15	Yes	750	80	No	Beach
8	Yes	550	67	Yes	Countryside

Supervised learning

- **Classification – Decision tree**

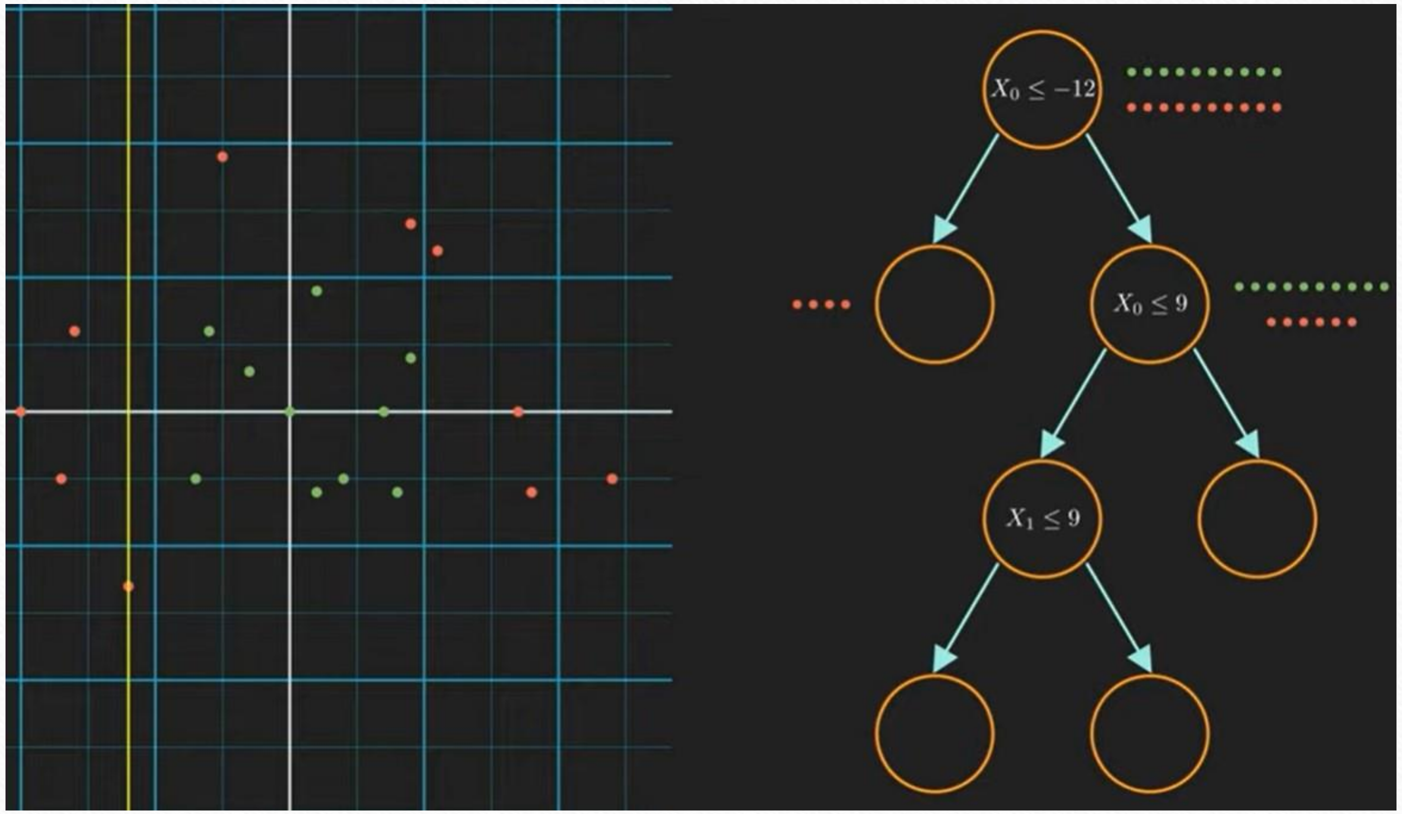
<u>Example</u>



Supervised learning

- **Classification – Decision tree**

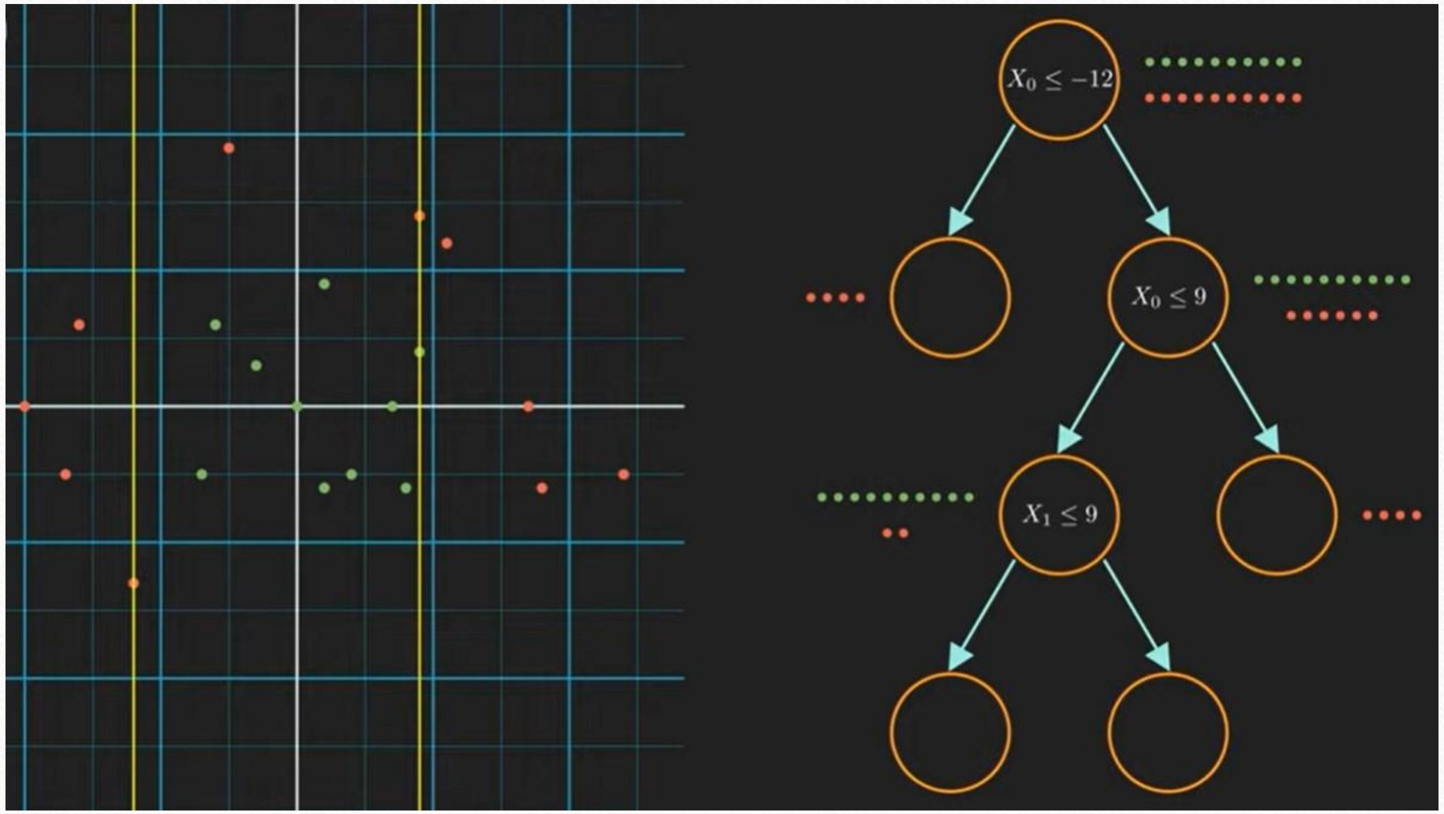
<u>Example</u>



Supervised learning

- **Classification – Decision tree**

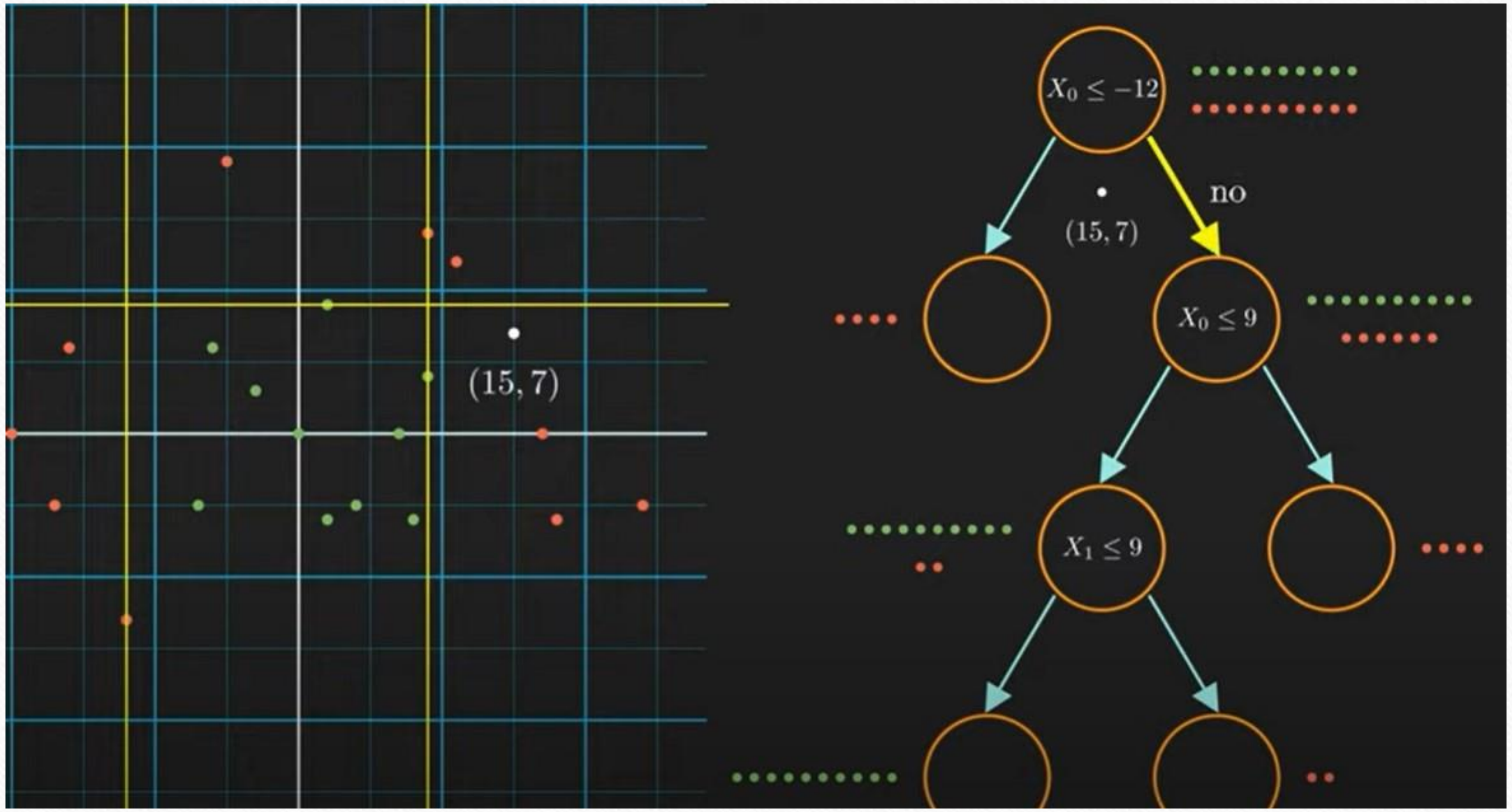
<u>Example</u>



Supervised learning

- **Classification – Decision tree**

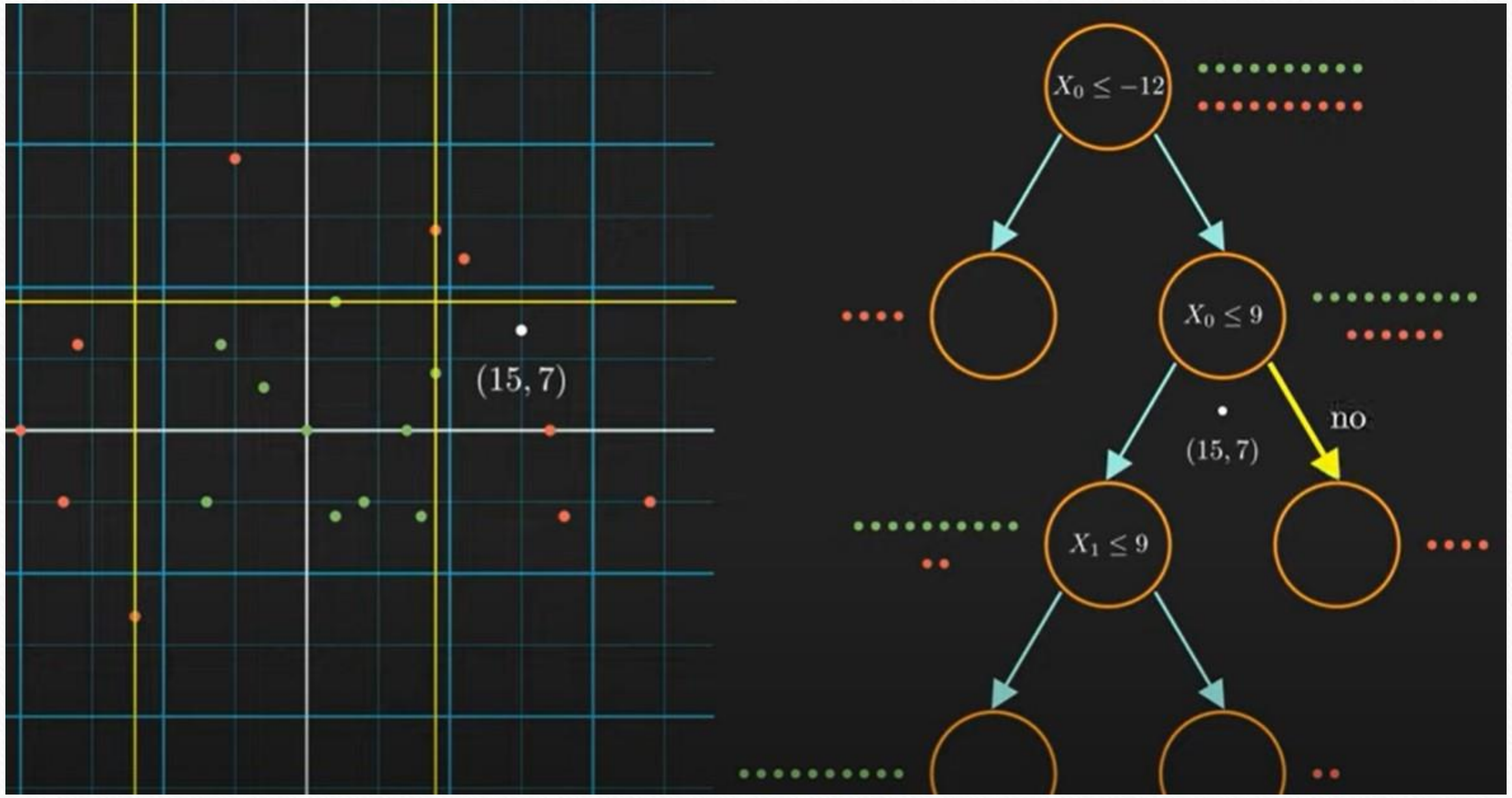
<u>Example</u>



Supervised learning

- **Classification – Decision tree**

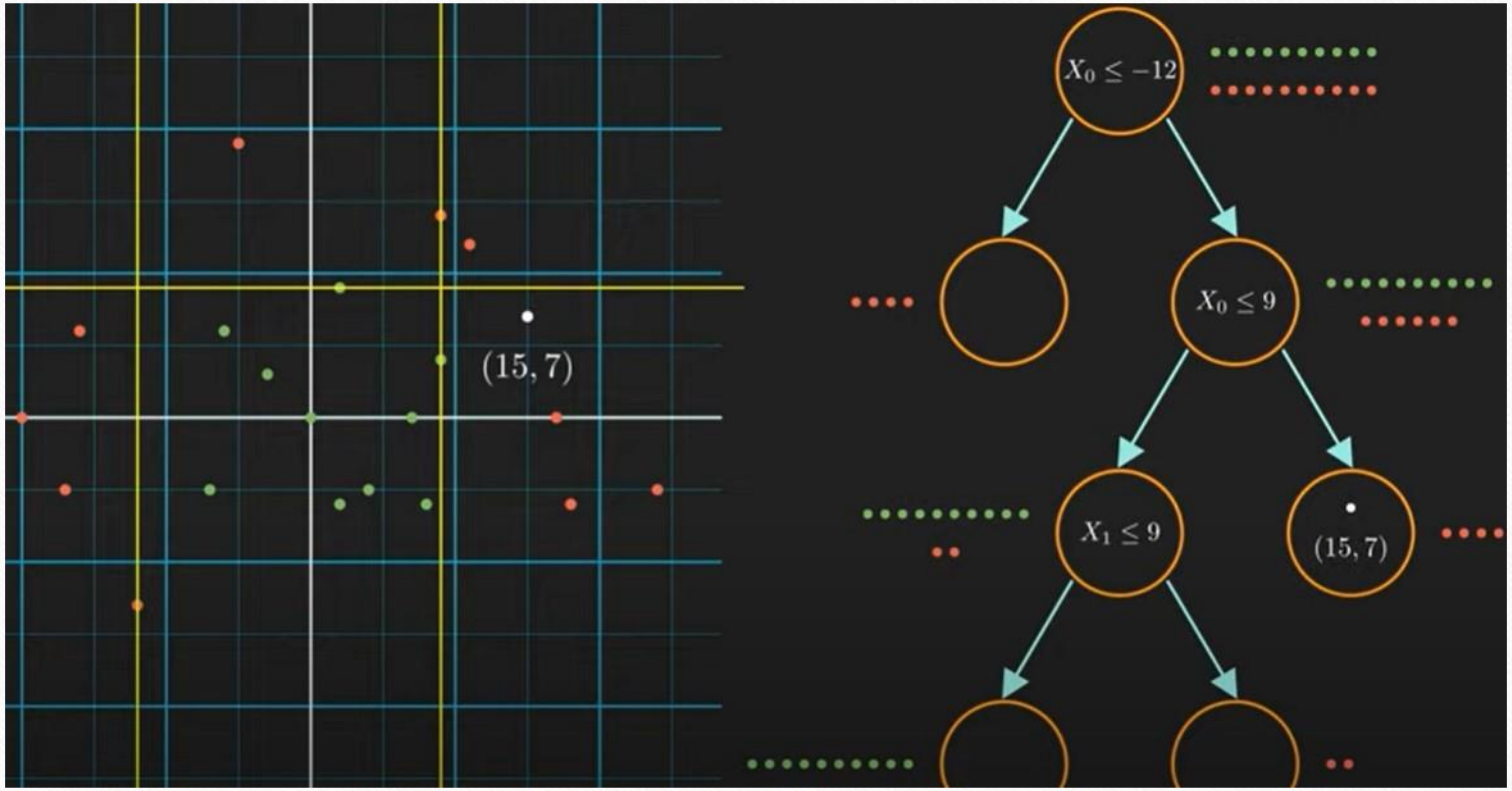
<u>Example</u>



Supervised learning

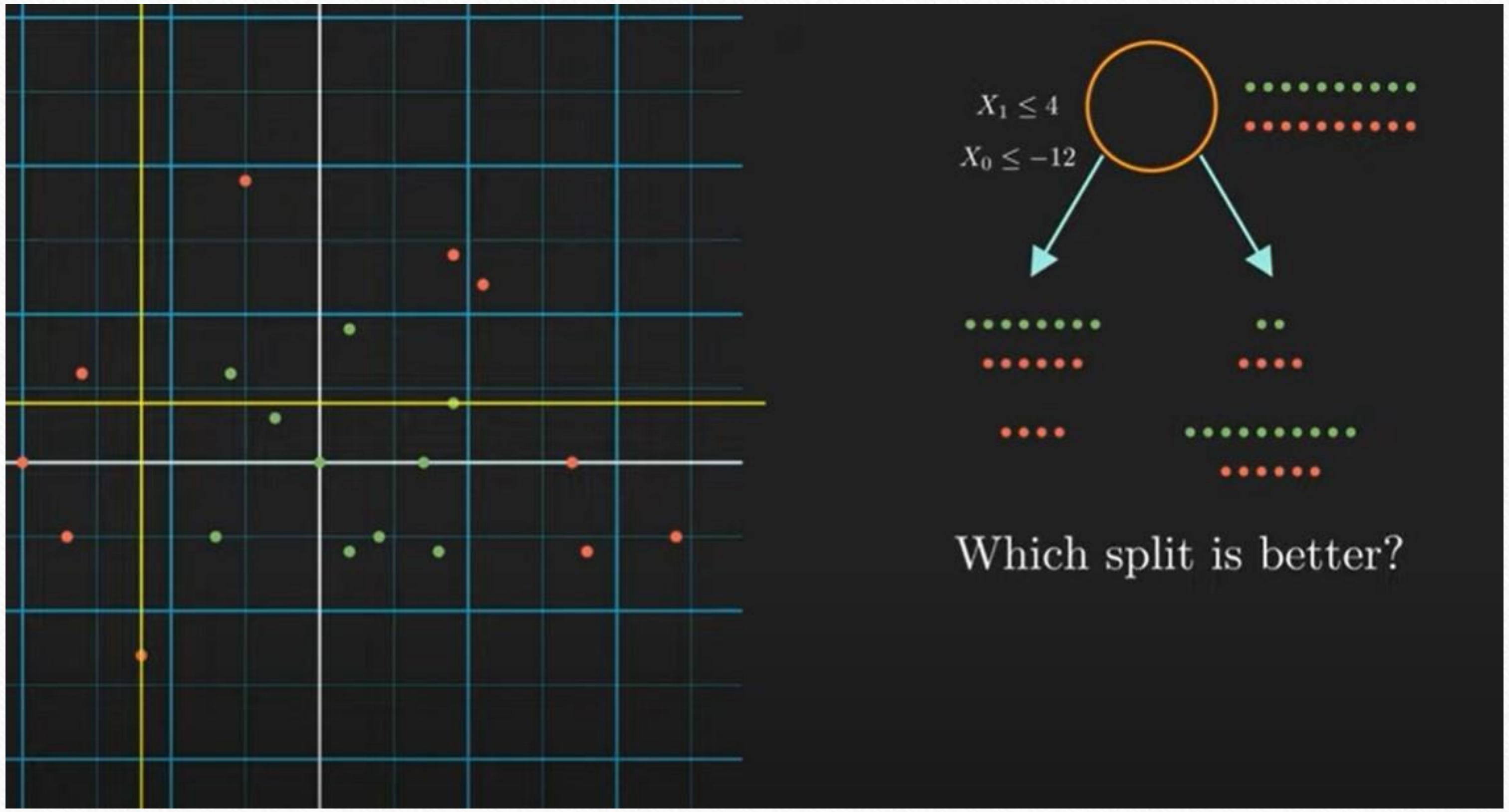
- **Classification – Decision tree**

<u>Example</u>



Supervised learning

- **Classification – Decision tree**
- But how can we determine the optimal split and condition?



Supervised learning

- **Classification – Decision tree**
- We can decide the best split by calculating the information gain resulting from each split using the following formula (it is log_2):

$$Entropy = \sum - p_i \ \log(p_i)$$

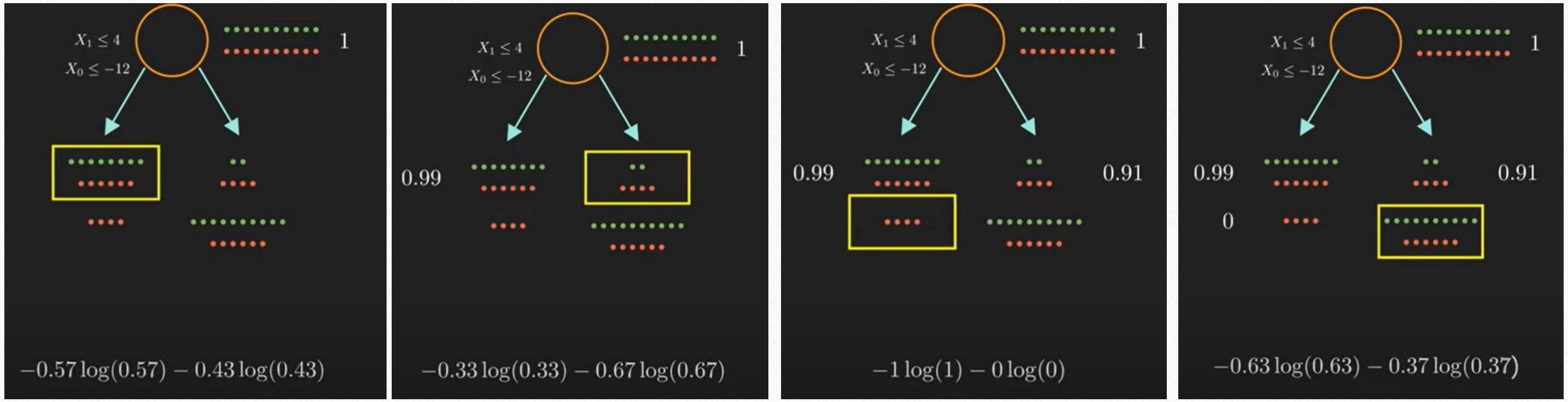
$$p_i = \text{probability of class i}$$

$$\text{IG} = E(parent) - \sum w_i \ E(child_i)$$

Supervised learning

- **Classification – Decision tree**

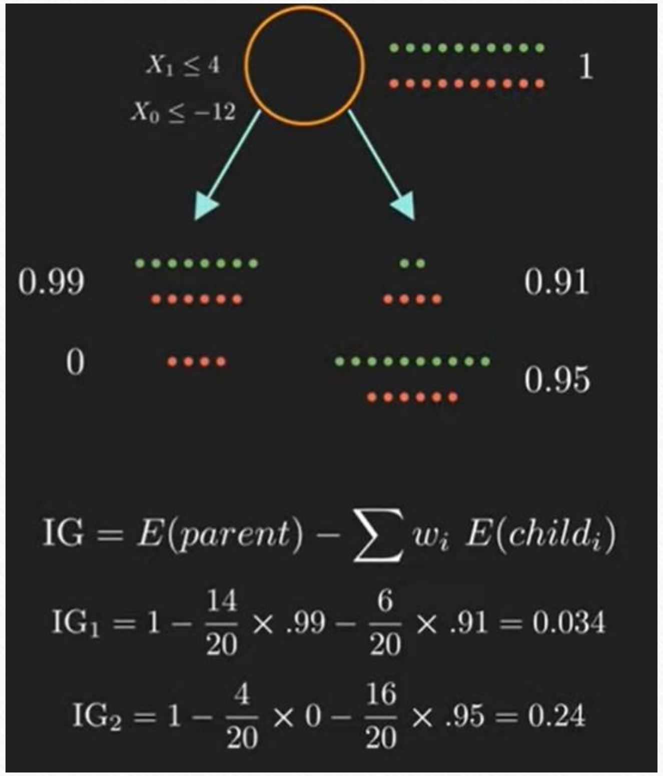
<u>Example</u>


$X_1 \leq 4$
$X_0 \leq -12$
1
0.99
0.91
0
$-0.57\log(0.57) - 0.43\log(0.43)$
$-0.33\log(0.33) - 0.67\log(0.67)$
$-1\log(1) - 0\log(0)$
$-0.63\log(0.63) - 0.37\log(0.37)$

Supervised learning

- **Classification – Decision tree**

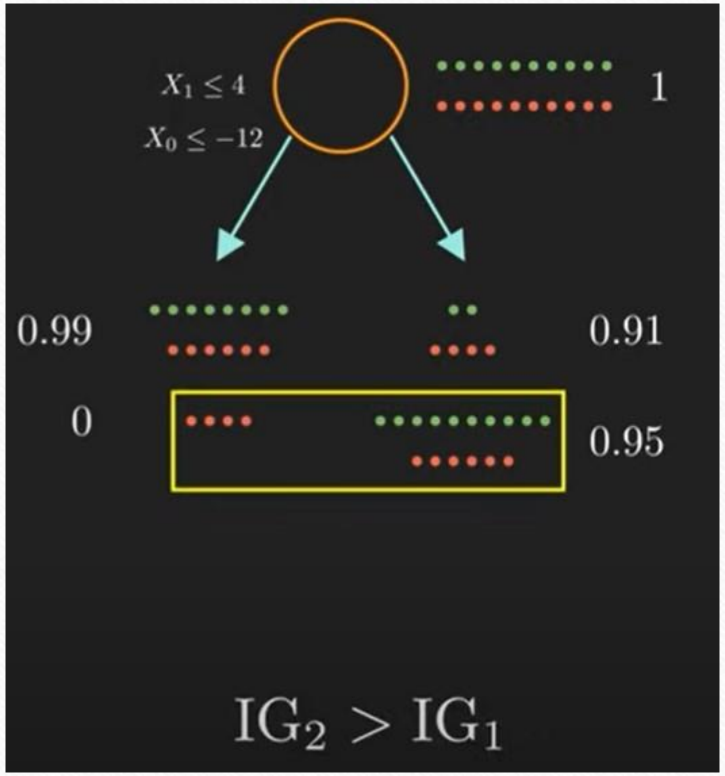
<u>Example</u>



$$\mathrm{IG} = E(parent) - \sum w_i \, E(child_i)$$

$$\mathrm{IG}_1 = 1 - \frac{14}{20} \times .99 - \frac{6}{20} \times .91 = 0.034$$

$$\mathrm{IG}_2 = 1 - \frac{4}{20} \times 0 - \frac{16}{20} \times .95 = 0.24$$



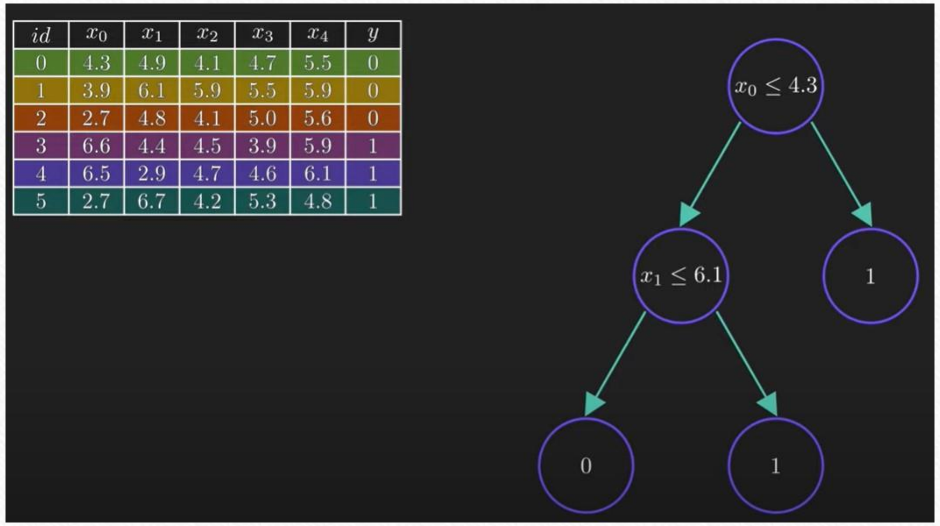
$$\mathrm{IG}_2 > \mathrm{IG}_1$$

Supervised learning

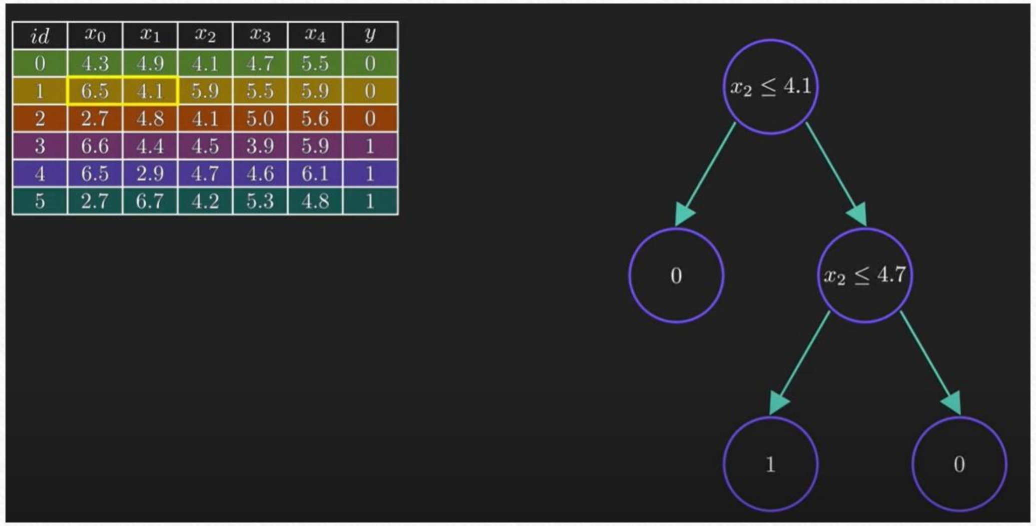
- **Classification – Decision tree**
- In reality, the model traverses through every possible feature and feature value to find the best split.
- A pure leaf node is a leaf node that contains data from only one class.
- Although it’s desirable to have data from only one class in each leaf node, this will not happen in most cases. If we have data from different classes in leaf nodes, we can use the **majority voting** technique to decide which class the leaf node belongs to.
- According to the majority voting, the class of the leaf node is the same class as the class of the majority of data inside the leaf.

Supervised learning

- **Classification – Random Forest**
- Decision trees are highly sensitive to the training data resulting in a high variance
- This problem will limit the generalization of the trained model



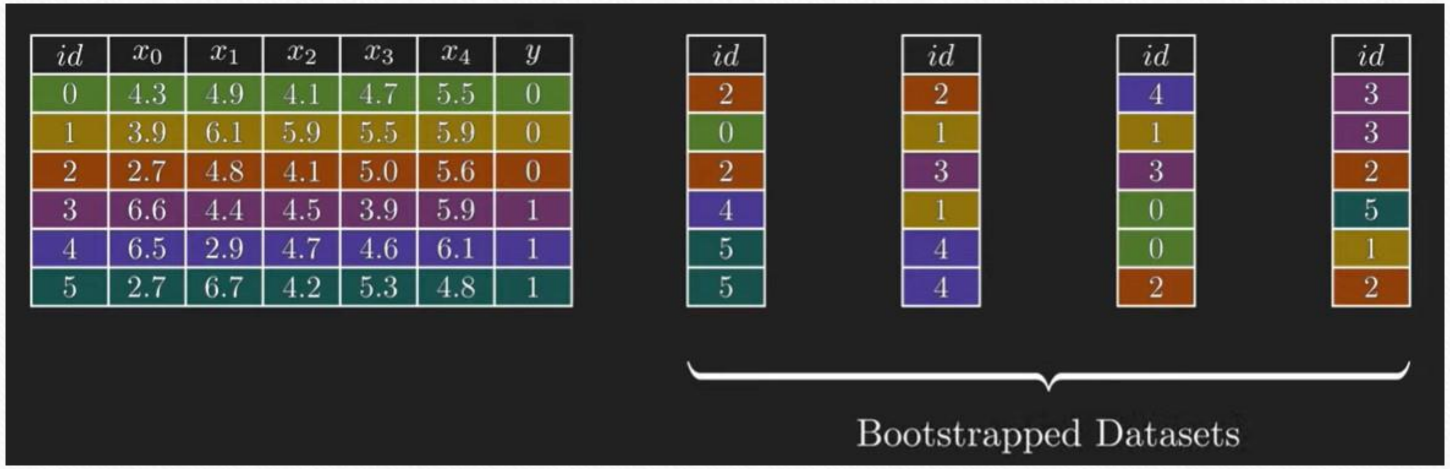
id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	3.9	6.1	5.9	5.5	5.9	0
2	2.7	4.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1



id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	6.5	4.1	5.9	5.5	5.9	0
2	2.7	4.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1

Supervised learning

- **Classification – Random Forest**
- Random forest is a collection of multiple random decision trees
- Random forest algorithm can be summarized as follows:
- **Step1**: Choose random subsets of the original dataset with replacement so that the total number of rows in the selected subset is the same as the original dataset. This process is called bootstrapping.

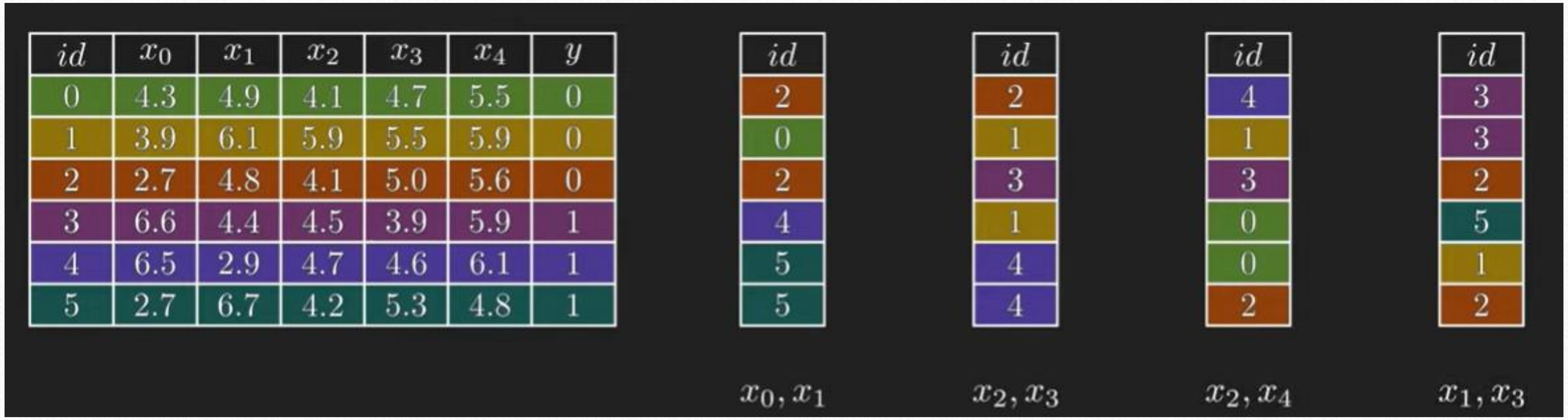


id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	3.9	6.1	5.9	5.5	5.9	0
2	2.7	4.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1

id	id	id	id
2	2	4	3
0	1	1	3
2	3	3	2
4	1	0	5
5	4	0	1
5	4	2	2

Supervised learning

- **Classification – Random Forest**
- NOTE: The number of subsets is a hyperparameter. Here we chose four subsets. However, it is usually set to 100.
- **Step2**: Randomly select a subset of features for each dataset. The number of selected features is a hyperparameter usually set to the square root or log of the total number of features.

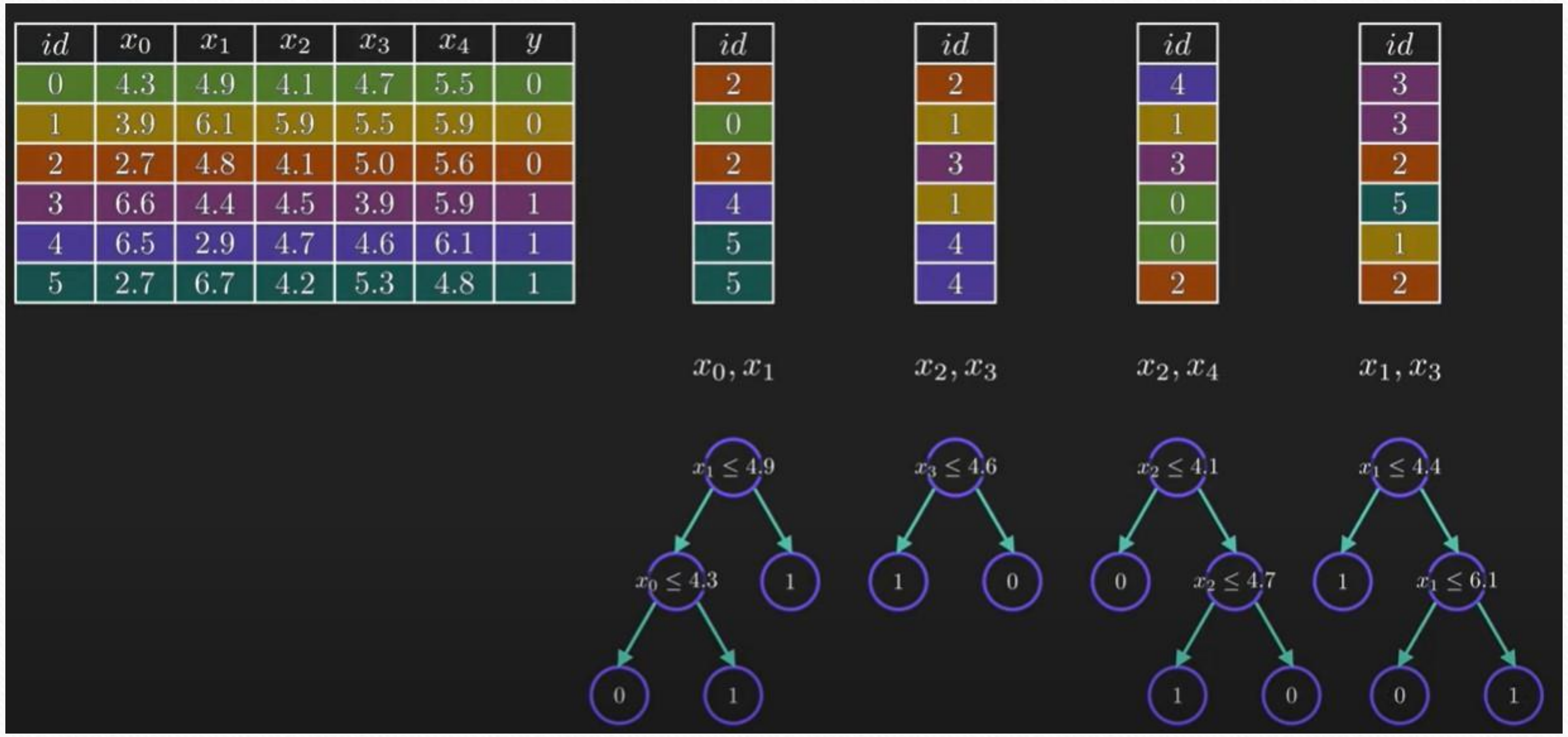


id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	3.9	6.1	5.9	5.5	5.9	0
2	2.7	4.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1

id	id	id	id
2	2	4	3
0	1	1	3
2	3	3	2
4	1	0	5
5	4	0	1
5	4	2	2
x_0, x_1	x_2, x_3	x_2, x_4	x_1, x_3

Supervised learning

- **Classification – Random Forest**
- **Step3**: Train one tree for each selected dataset and set of features.

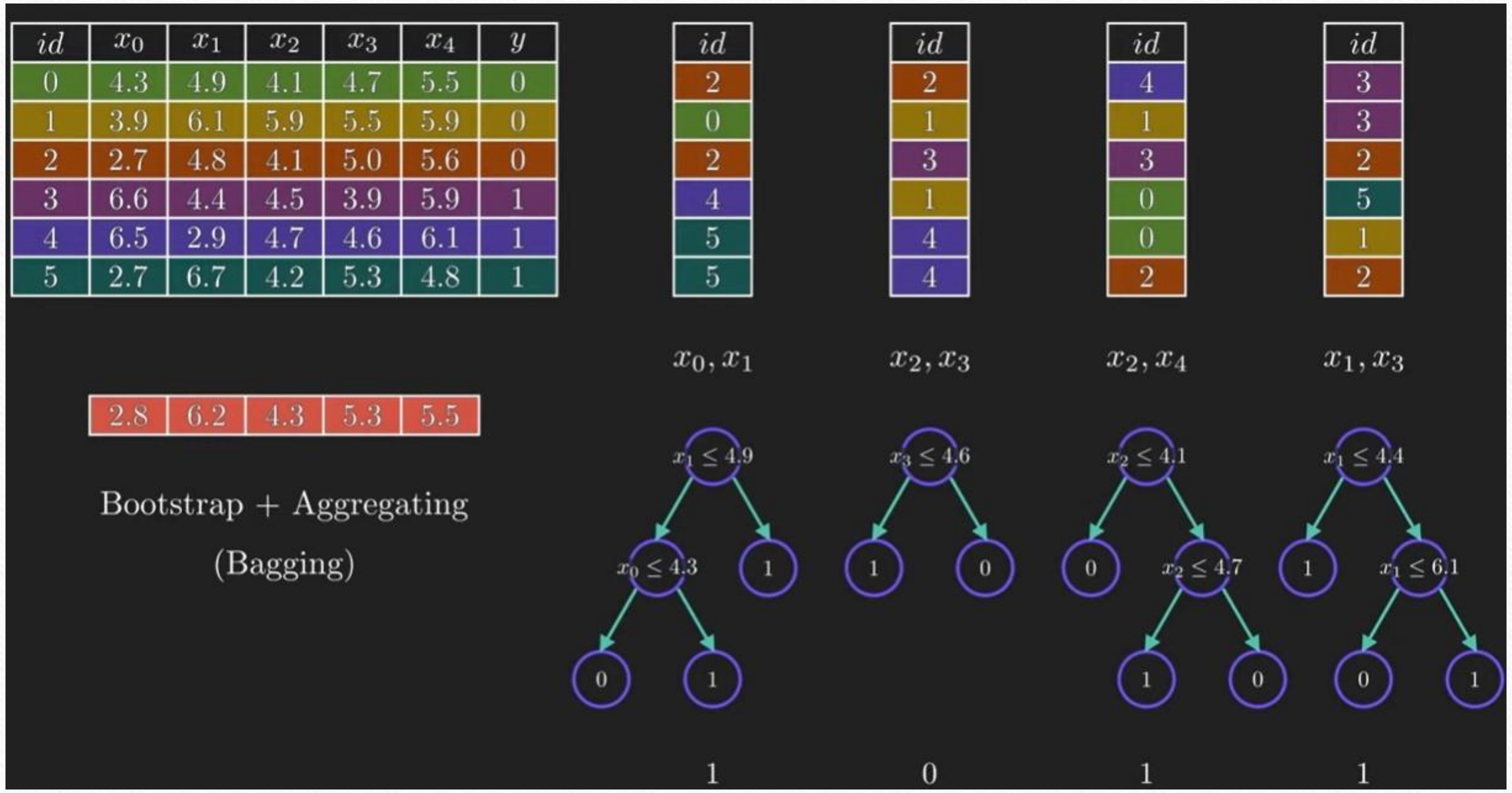


id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	3.9	6.1	5.9	5.5	5.9	0
2	2.7	4.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1

id	id	id	id
2	2	4	3
0	1	1	3
2	3	3	2
4	1	0	5
5	4	0	1
5	4	2	2

Supervised learning

- **Classification – Random Forest**
- **Step4**: For new unseen data, make predictions using the majority voting



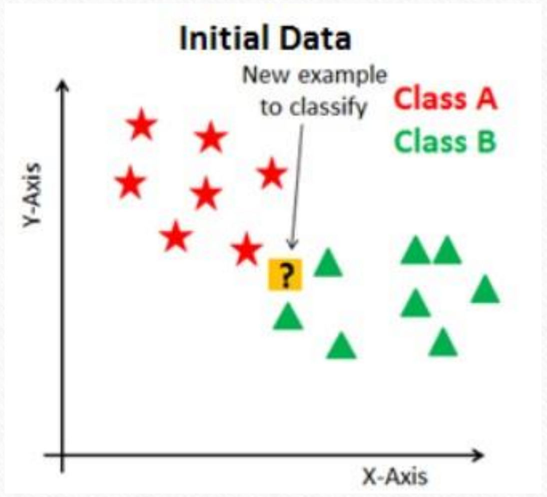
id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	3.9	6.1	5.9	5.5	5.9	0
2	2.7	4.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1

id	id	id	id
2	2	4	3
0	1	1	3
2	3	3	2
4	1	0	5
5	4	0	1
5	4	2	2
x_0, x_1	x_2, x_3	x_2, x_4	x_1, x_3

2.8	6.2	4.3	5.3	5.5

Supervised learning

- **Classification – K-nearest neighbors**
- Suppose P1 is the point, for which label needs to predict. First, you find the k closest point to P1 and then classify points by majority vote of its k neighbors. Each object votes for their class and the class with the most votes is taken as the prediction



Supervised learning

- **Classification – K-nearest neighbors (KNN)**
- For finding closest similar points, you find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance.

- KNN algorithm has the following three steps:

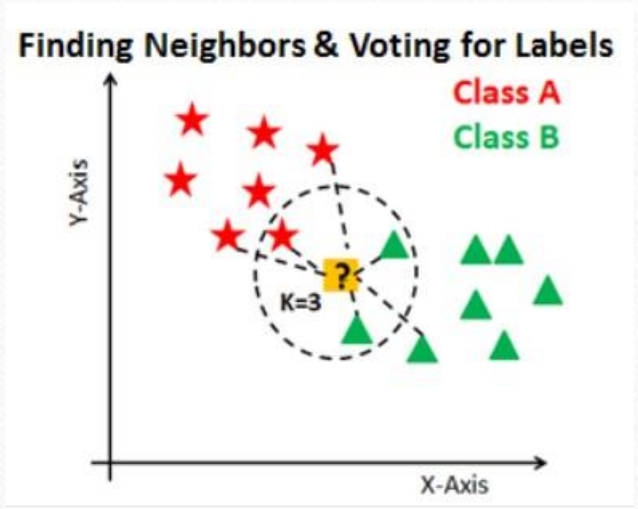
Step1: Calculate the distance between the new data point and K neighboring data

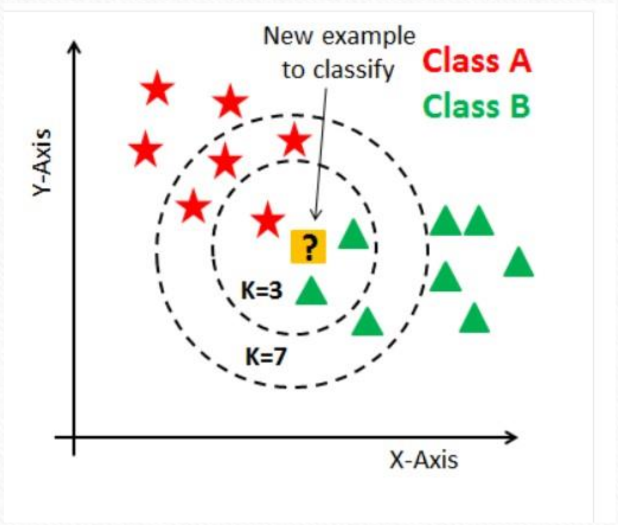
Step2: Find closest neighbors

Step3: Use majority voting to predict the class of the new data

Supervised learning

- **Classification – K-nearest neighbors (KNN)**





NOTE: K is a hyperparameter and is usually set to the square root of the total number of samples. You may get different results using different K.

Supervised learning

- **Classification – K-nearest neighbors (KNN)**

NOTE: KNN and RF can be used for multiclass classification

NOTE: There are also KNN and RF regression algorithms that use similar procedures to the classifiers but are used for regression.

Supervised learning

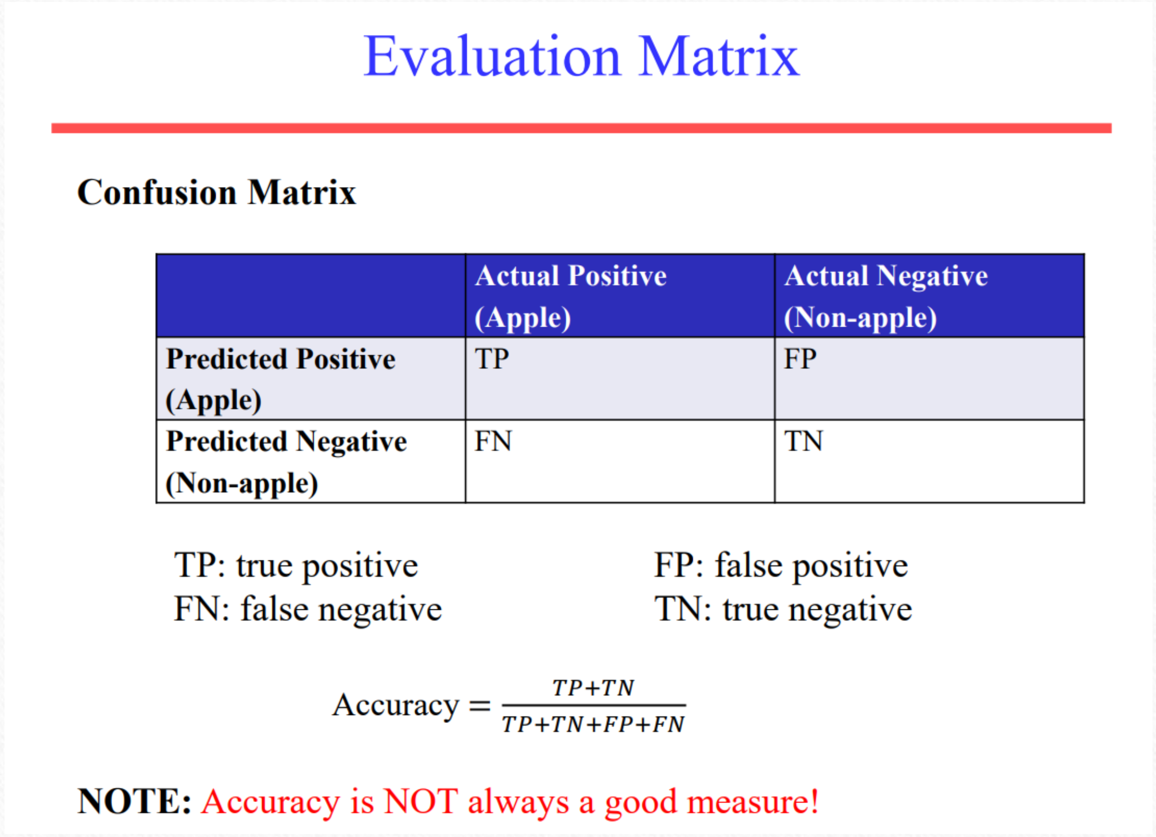
- **Classification – Evaluation metrics**

- Evaluation metrics are used to measure how good the predictions of the system are

- The most important evaluation metrics for classification include **Accuracy, Precision, Recall, F1 Score, Specificity, and Sensitivity**

Supervised learning

- **Classification – Evaluation metrics**



Evaluation Matrix

Confusion Matrix

	Actual Positive (Apple)	Actual Negative (Non-apple)
Predicted Positive (Apple)	TP	FP
Predicted Negative (Non-apple)	FN	TN

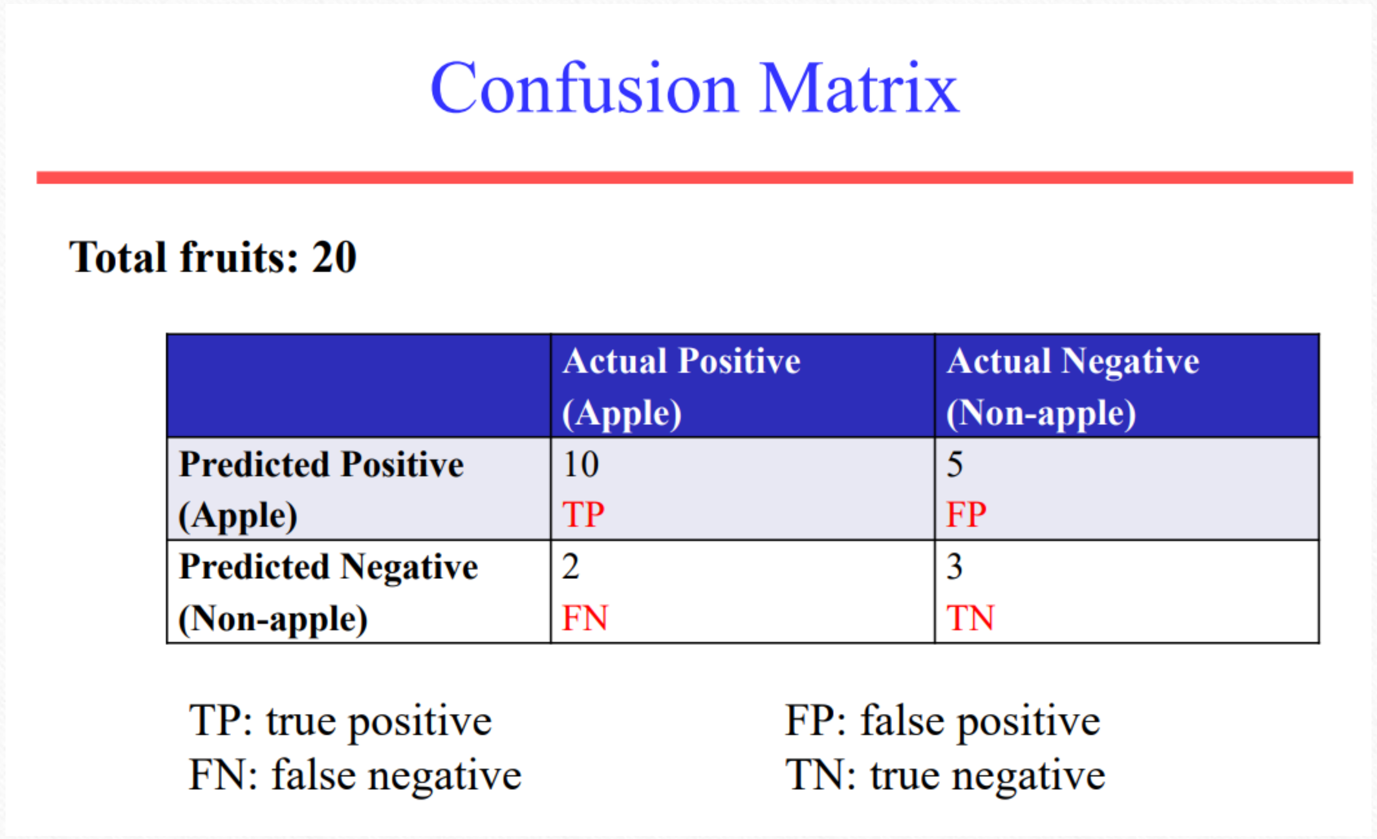
TP: true positive	FP: false positive
FN: false negative	TN: true negative

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

NOTE: Accuracy is NOT always a good measure!

Supervised learning

- **Classification – Evaluation metrics**



Confusion Matrix

Total fruits: 20

	Actual Positive (Apple)	**Actual Negative (Non-apple)**
Predicted Positive (Apple)	10 TP	5 FP
Predicted Negative (Non-apple)	2 FN	3 TN

TP: true positive
FN: false negative

FP: false positive
TN: true negative

Supervised learning

- **Classification – Evaluation metrics**

TP: The number of positive samples that are predicted correctly as positive (**T**ruly classified as **P**ositive)

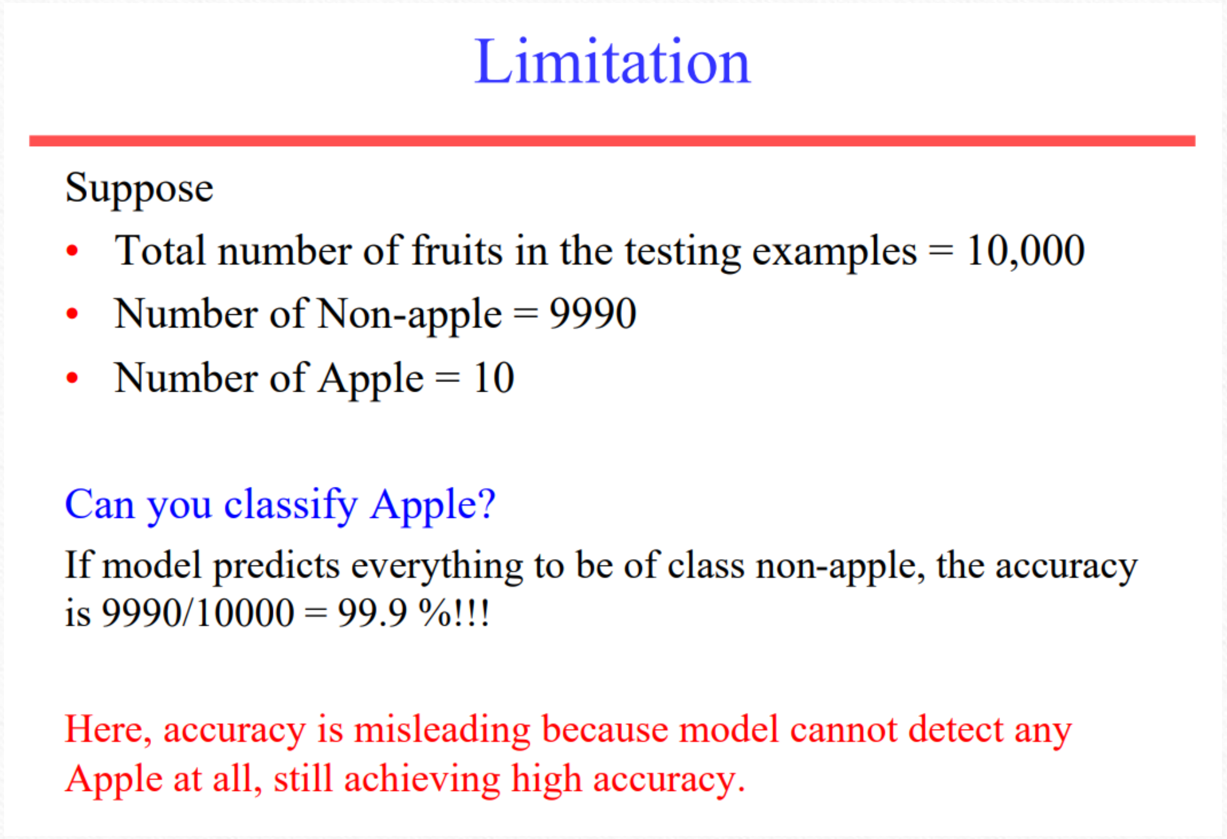
TN: The number of negative samples that are predicted correctly as negative (**T**ruly classified as **N**egative)

FP: The number of negative samples that are predicted incorrectly as positive (**F**alsely classified as **P**ositive)

FN: The number of positive samples that are predicted incorrectly as negative (**F**alsely classified as **N**egative)

Supervised learning

- **Classification – Evaluation metrics**



Limitation

Suppose

- Total number of fruits in the testing examples = 10,000
- Number of Non-apple = 9990
- Number of Apple = 10

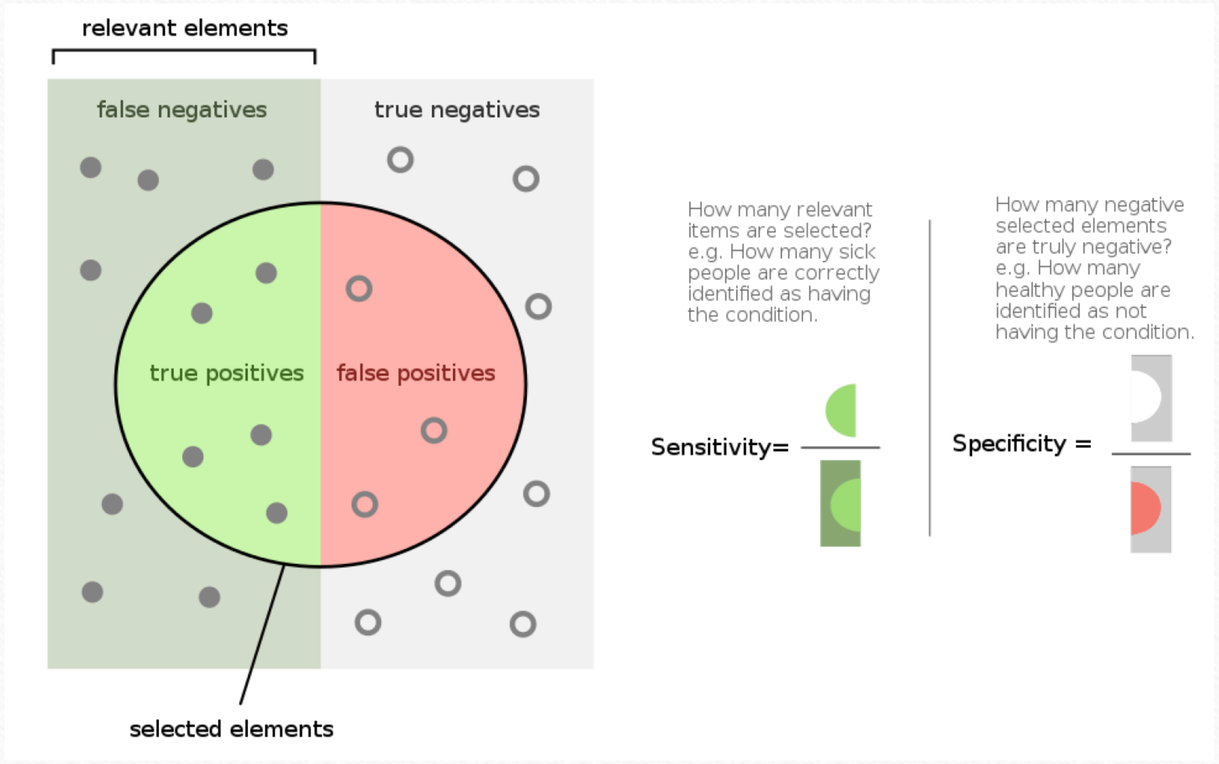
Can you classify Apple?

If model predicts everything to be of class non-apple, the accuracy is 9990/10000 = 99.9 %!!!

Here, accuracy is misleading because model cannot detect any Apple at all, still achieving high accuracy.

Supervised learning

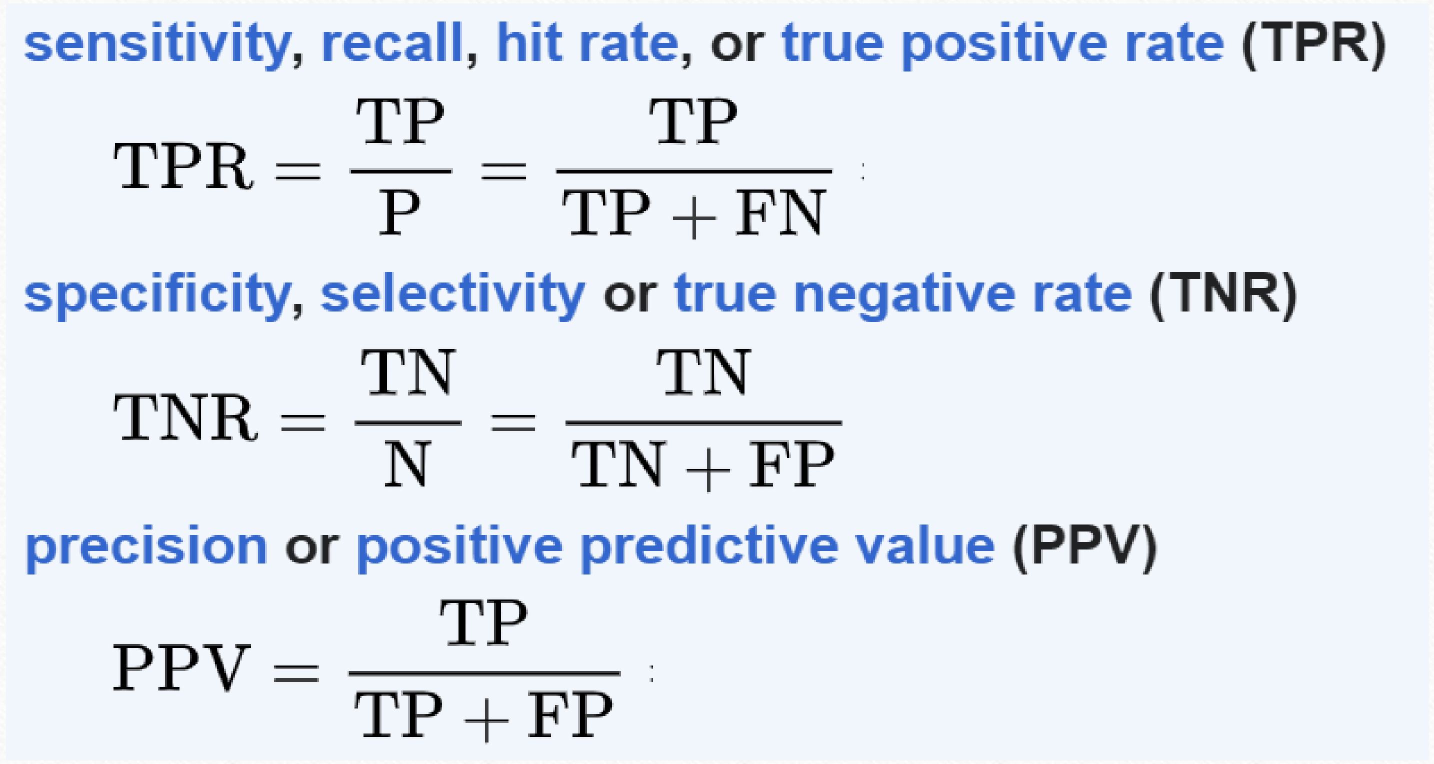
- **Classification – Evaluation metrics**


relevant elements
false negatives
true negatives
true positives
false positives
selected elements
How many relevant items are selected? e.g. How many sick people are correctly identified as having the condition.
How many negative selected elements are truly negative? e.g. How many healthy people are identified as not having the condition.
Sensitivity=
Specificity =

Sensitivity and specificity. (2024, January 24). In *Wikipedia*. https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Supervised learning

- **Classification – Evaluation metrics**



sensitivity, recall, hit rate, or true positive rate (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

specificity, selectivity or true negative rate (TNR)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

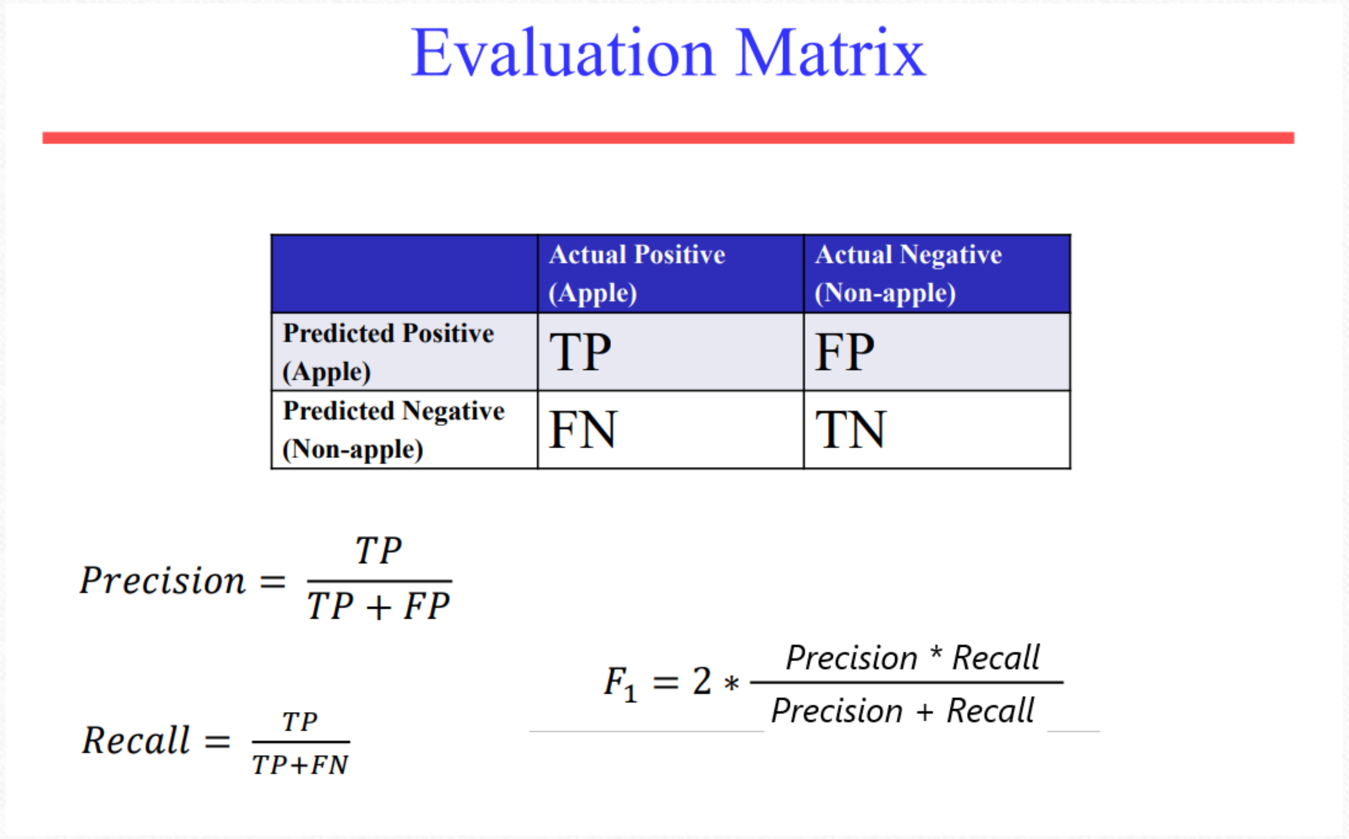
precision or positive predictive value (PPV)

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

Sensitivity and specificity. (2024, January 24). In *Wikipedia*. https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Supervised learning

- **Classification – Evaluation metrics**



Evaluation Matrix

	Actual Positive (Apple)	Actual Negative (Non-apple)
Predicted Positive (Apple)	TP	FP
Predicted Negative (Non-apple)	FN	TN

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

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

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

Supervised learning

- **Classification – Evaluation metrics**

NOTE 1: These metrics are defined for binary classification, but can be extended to multi-class classification

NOTE 2: It is optional to consider either of the classes as negative or positive. However, once you have defined a class as negative or positive, you should stick to your definition.

Supervised learning

- **Classification – Evaluation metrics**

NOTE 1: These metrics are defined for binary classification, but can be extended to multi-class classification

NOTE 2: It is optional to consider either of the classes as negative or positive. However, once you have defined a class as negative or positive, you should stick to your definition.

Classification

- **Evaluation metrics - example**

Question: Assume we have 1000 fruits, and we want to classify them into two classes as follows: Apple and Non-apple. Assume we have 10 apples in the dataset, and the rest of the fruits are not apples. If the classifier classifies all fruits as Non-apple, calculate the evaluation metrics for this classifier.

Solution: We define the Non-apple class as positive and the Apple class as negative.
TP= 990 (How many Non-apple fruits are classified as Non-apple)
TN= 0 (How many Apple fruits are classified as Apple)
FP= 10 (How many Apple fruits are classified as Non-apple)
FN= 0 (How many Non-apple fruits are classified as Apple)

Classification

- **Evaluation metrics - example**

Accuracy = (TP+TN)/(TP+TN+FP+FN)=(990+0)/(990+0+10+0)=0.99 (99%)
Interpretation: The model is prediction 99% of cases correctly.

Sensitivity= TP/(TP+FN)= 990/(990+0)=1 (100%)
Interpretation: The model is predicting 100% of Non-apple fruits correctly

Specificity= TN/(TN+FP)= 0/(0+10)=0 (0%)
Interpretation: The model is predicting 0% of Apple fruits correctly

Precision= TP/(TP+FP)= 990/(990+10)=0.99 (99%)
Interpretation: 99% of predicted Non-apple fruits are correct

Classification

- **Evaluation – Cross validation**

- In classification, we use a portion of data for training the model. This portion is called **training data**
- We use the rest of the data for resting the model. This portion is called **testing data**.
- There is a challenge. Which part of data should be used for training the model and which part should be used for testing?

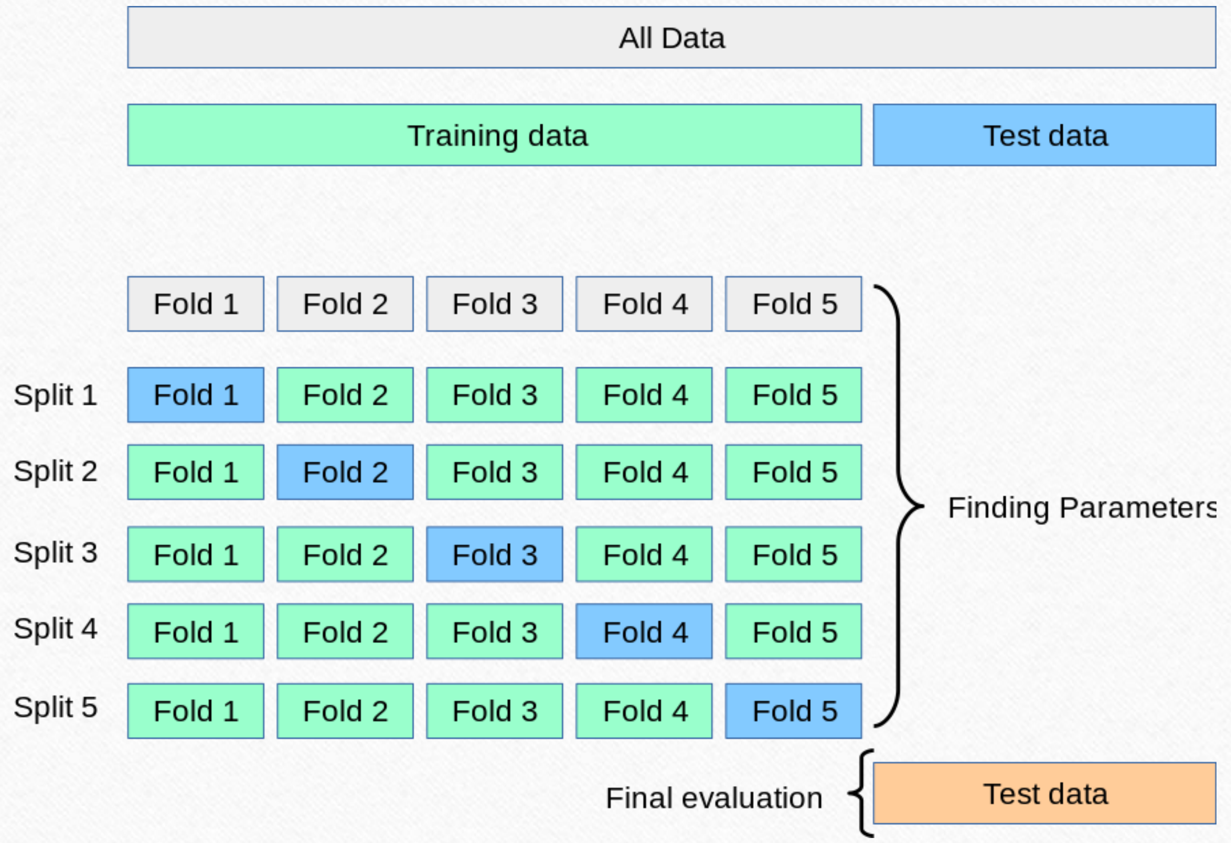
Classification

- **Evaluation – Cross validation**

- In the cross-validation (CV) method, we divide data into K folds. Then in each iteration, one of the folds is reserved for training the model and one portion is reserved for testing the model.
- We repeat this process until all folds have been used for testing at least once.
- This procedure is also called K-fold cross-validation.
- The accuracy of the model will be the average accuracy of all iterations.
- The best practice is to test the model on separate test data after the cross-validation

Classification

- **Evaluation – Cross validation**



Classification

- **Examples**

Please see Lecture #3-examples notebook for examples of classification models.