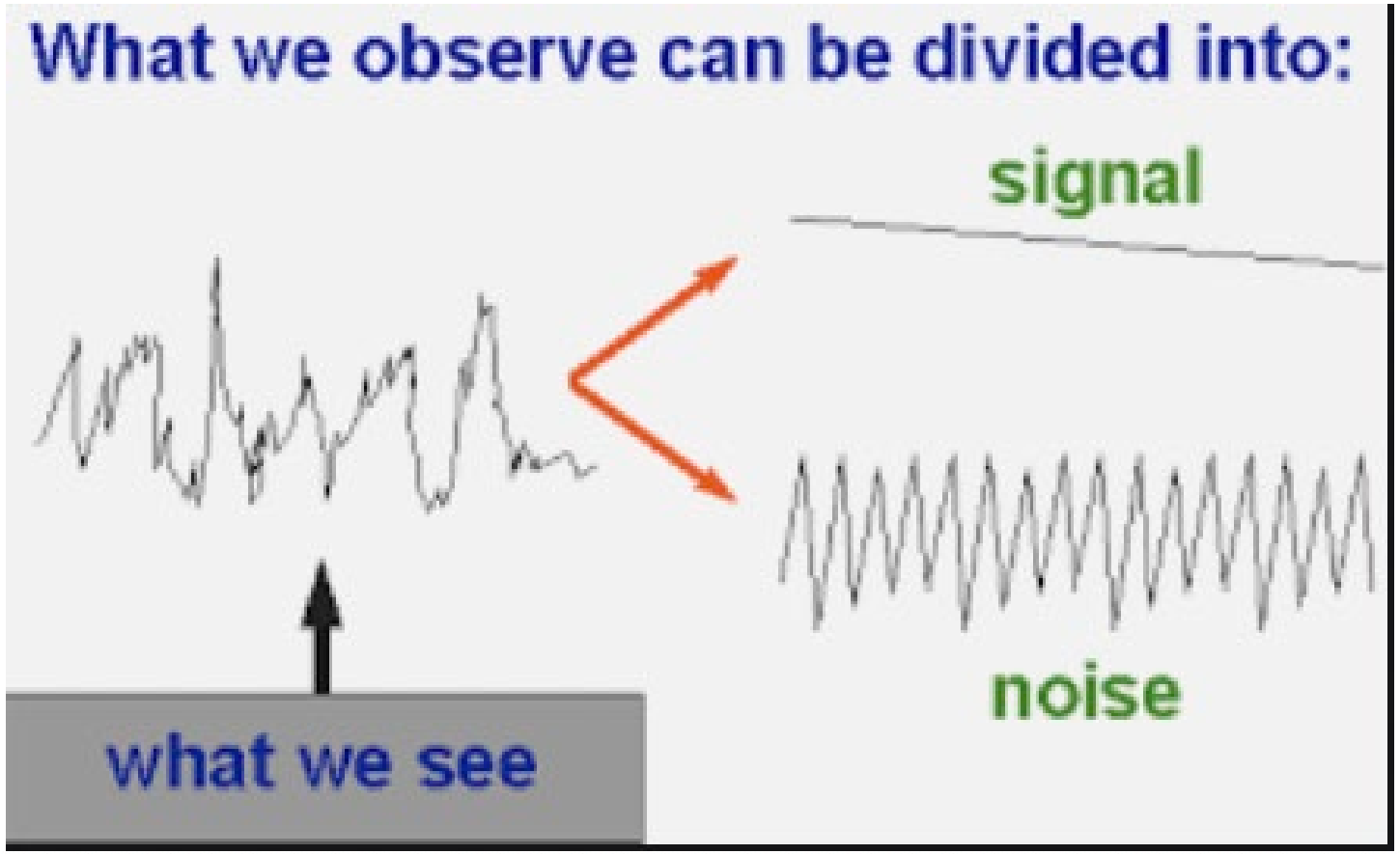
CM3015

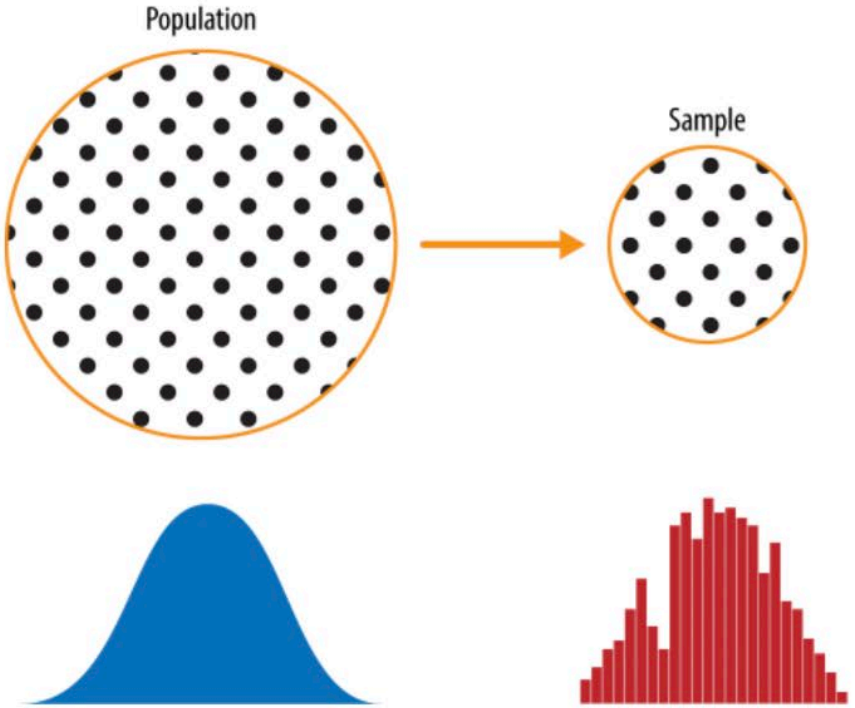
"I confess that I have been blind as a mole, but it is better to learn wisdom late than never to learn it at all."

Sherlock Holmes
The Man with the Twisted Lip

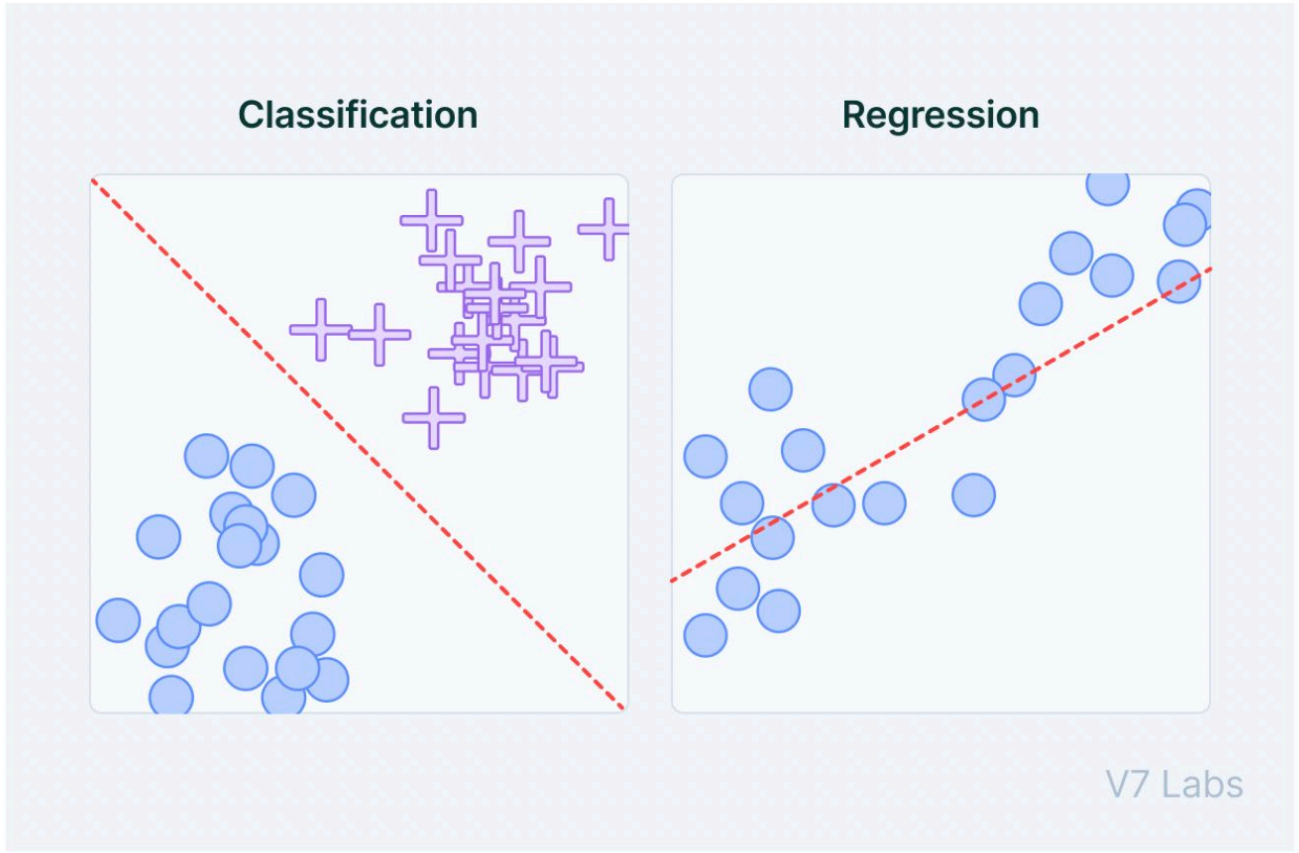


It is the goal of models to describe the signal, despite the noise.

A perfect model describes the signal exactly, and ignores all of the noise. If a model fails to capture all of the signal, that type of error is called **bias**. If a model captures of some of the noise, that type of error is called **variance**. Too much bias in our model means that it will perform poorly in all situations because it hasn't captured the signal well. You may also hear this called **underfitting**. This was the case



Recap: population and sample



Some of the most common algorithms in Supervised Learning include Support Vector Machines (SVM), Logistic Regression, Naive Bayes, Neural Networks, K-nearest neighbor (KNN), and Random Forest.

Classification = discrete/ categorical target variable

Regression = continuous/ numerical target variable

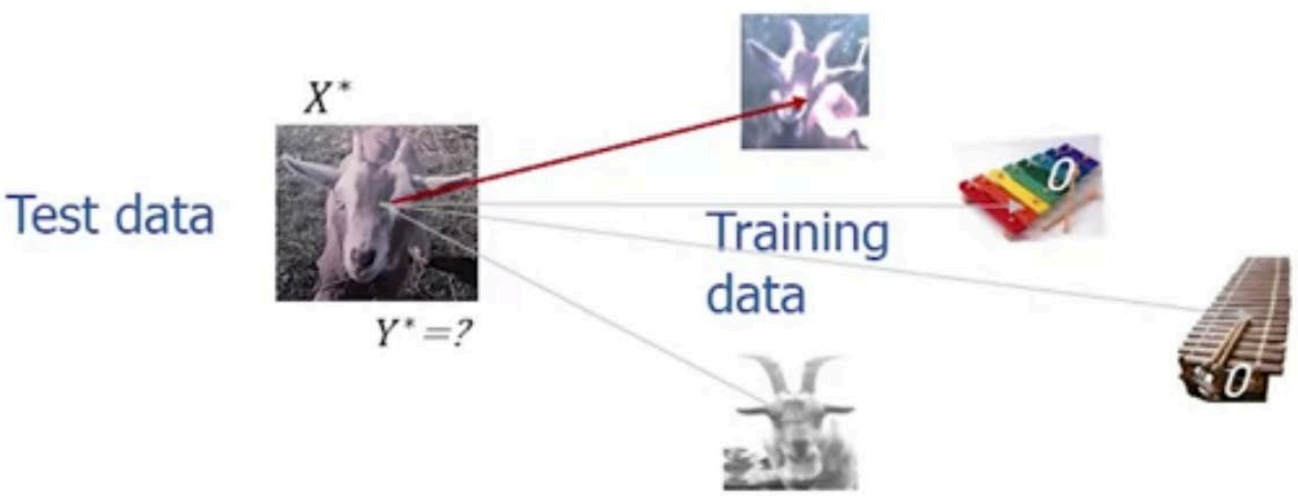
The Free Lunch Theorem

There is no one model that works best in every situation

Occam's Razor

- The optimal model describes the regularities of the data, which compresses the data the most, while keeping salient features

k-Nearest Neighbour Classification (k-NN)

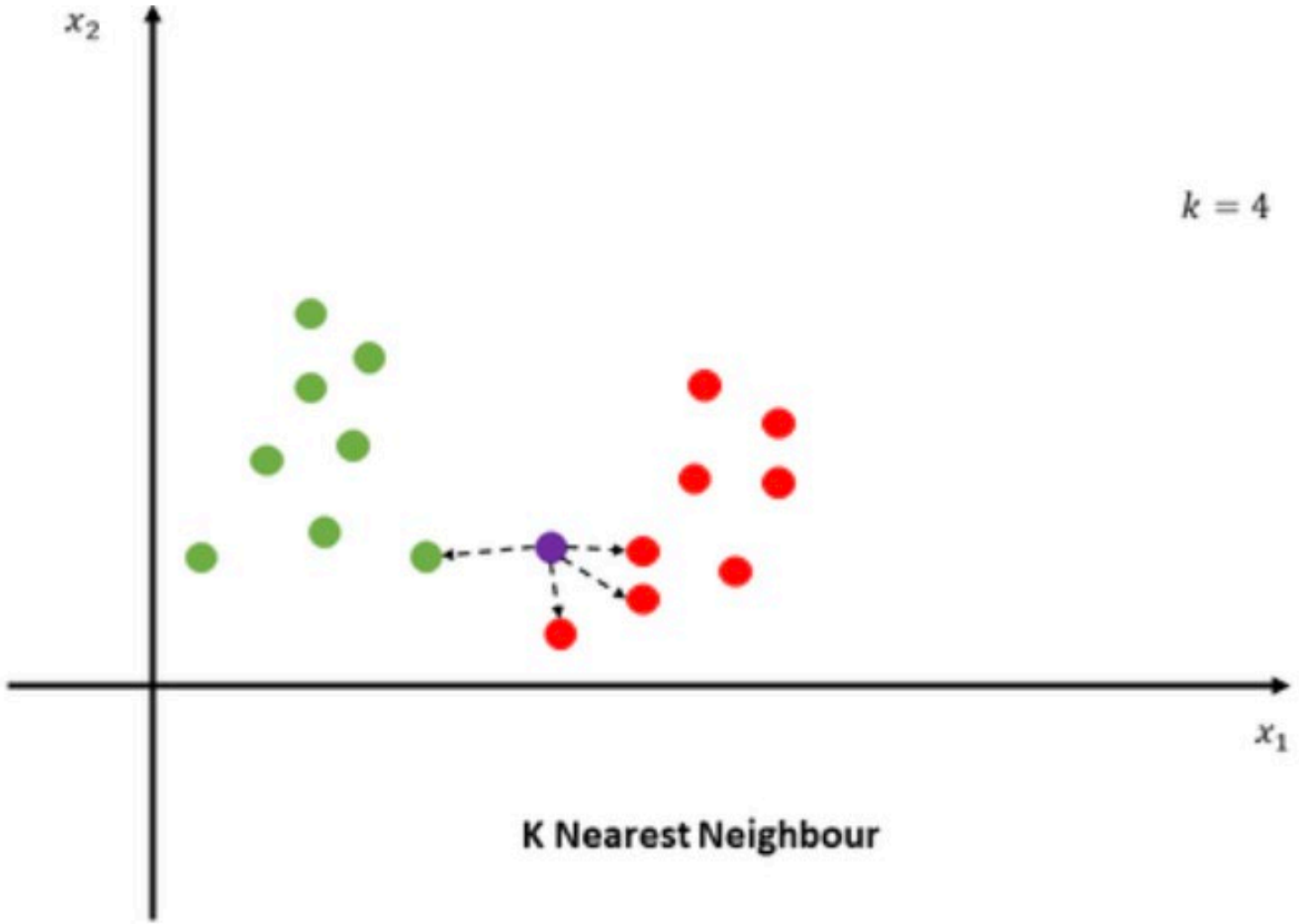


1. Measure distance from test image X^* to every image in training set X

$$\left(\text{image} - \text{image} \right)^2$$

2. Assign the label of the (k=1) "nearest neighbour" to test data X^*

K Nearest Neighbour = ‘lazy learner’



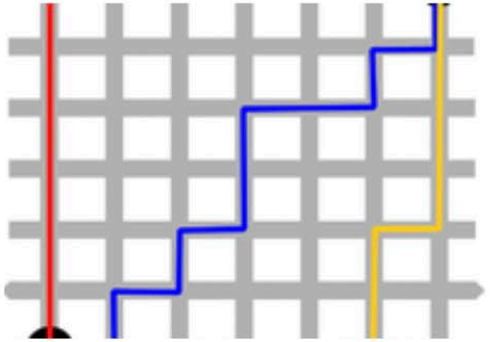
Euclidean Distance based on Pythagoras

Distance and Similarity

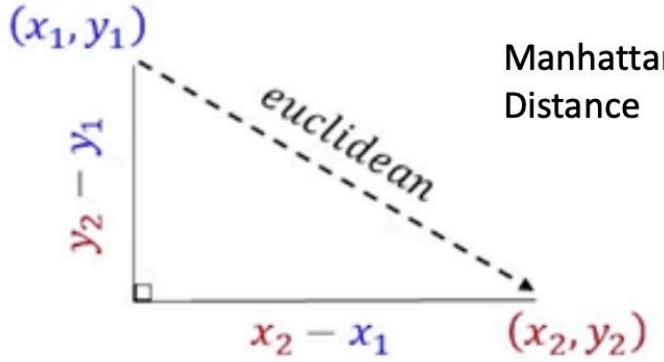
Distance from (x_1, y_1) *to* (x_2, y_2):

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

$$Manhattan = |x_2 - x_1| + |y_2 - y_1|$$

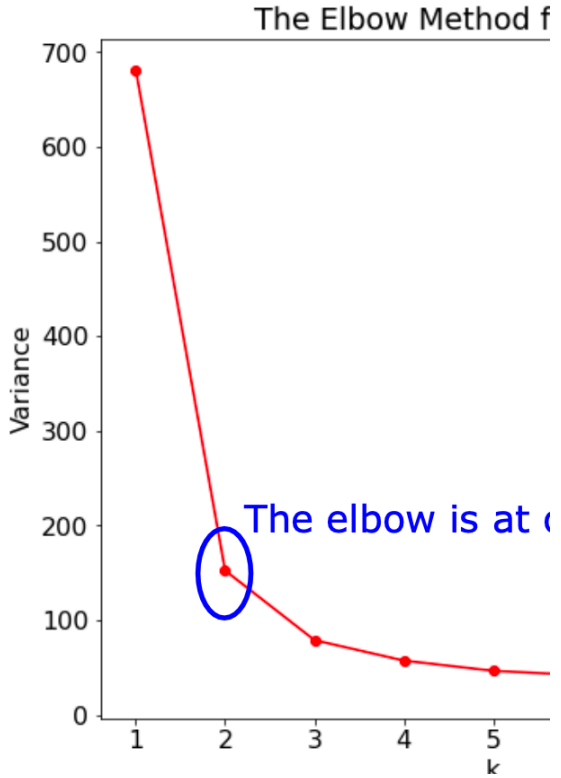


Manhattan Distance



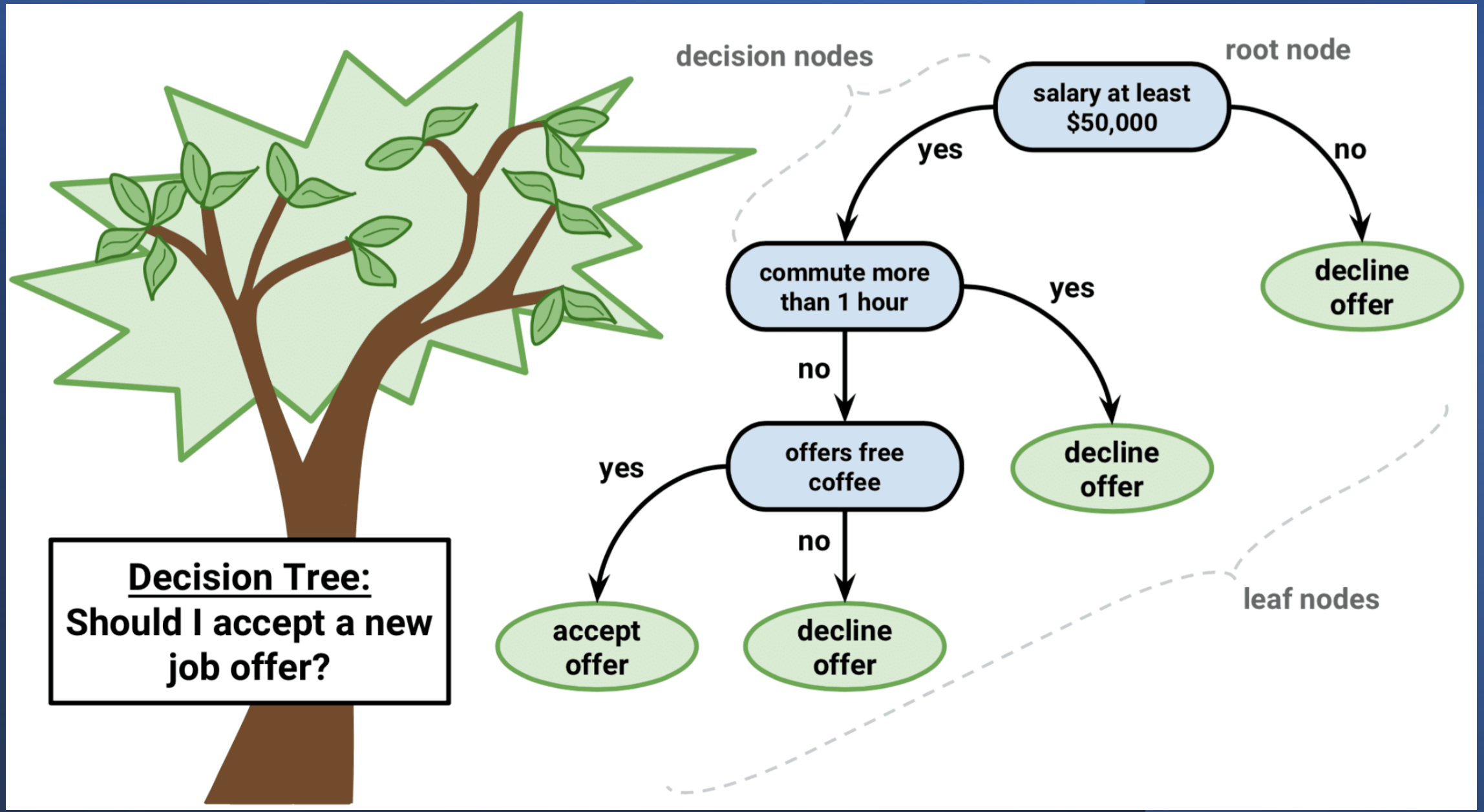
The elbow plot

- Try different values for k.
- Identify when the number of clusters explains ‘most’ of the variance in the data.
- The point of inflection on the curve is at the optimal value of *k*.
- No elbow is an indication that the data that does not have well-defined clusters.



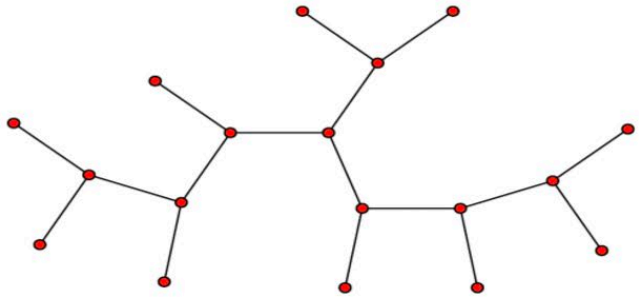
K-NN Elbow Plot

Decision Trees = white box algorithm

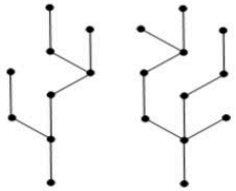


Decision Tree

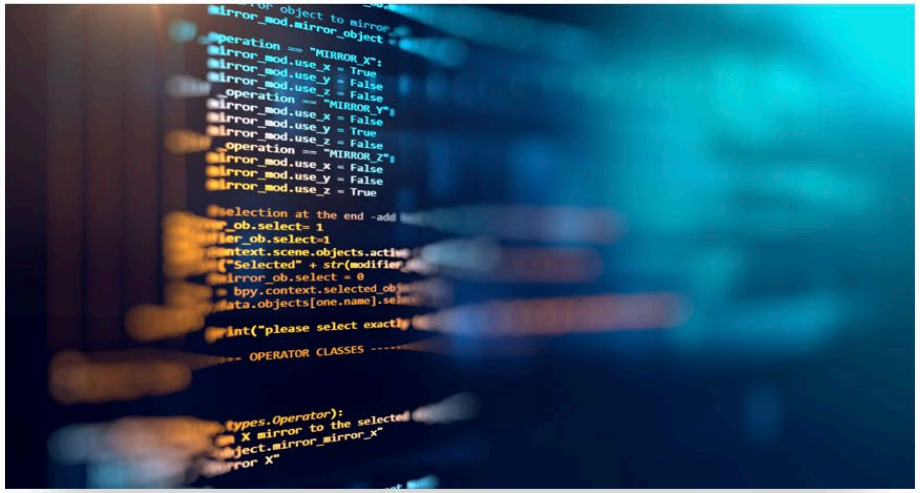
For example, here's a tree:



Here's a forest:



Mathematically, a tree T is a graph G(V, E) if between every pair of distinct vertices is a unique edge



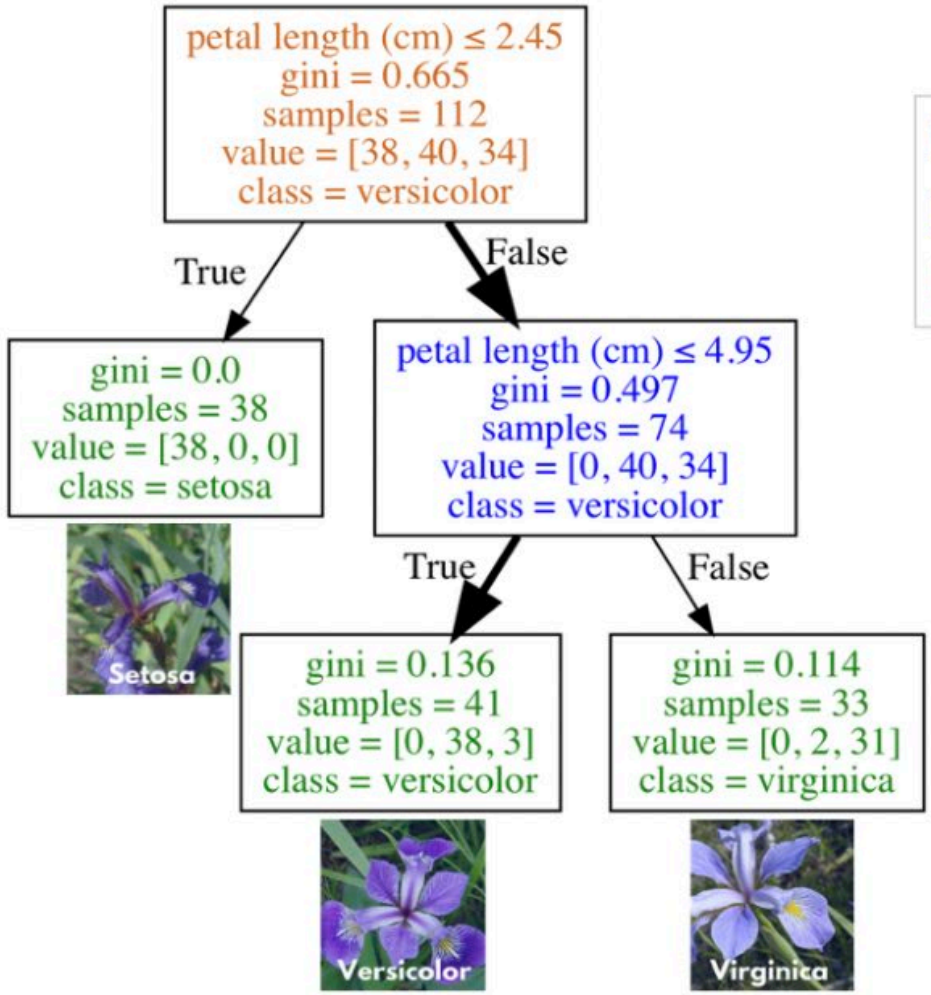
Based on greedy algorithm

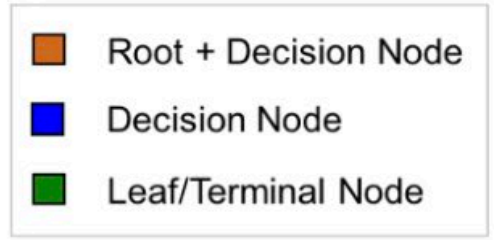
Greedy algorithm = heuristic for choosing locally optimal solution at each point

Anatomy of the Decision Tree or CART (Classification and Regression Tree)

What class (species) is a flower with the following feature?

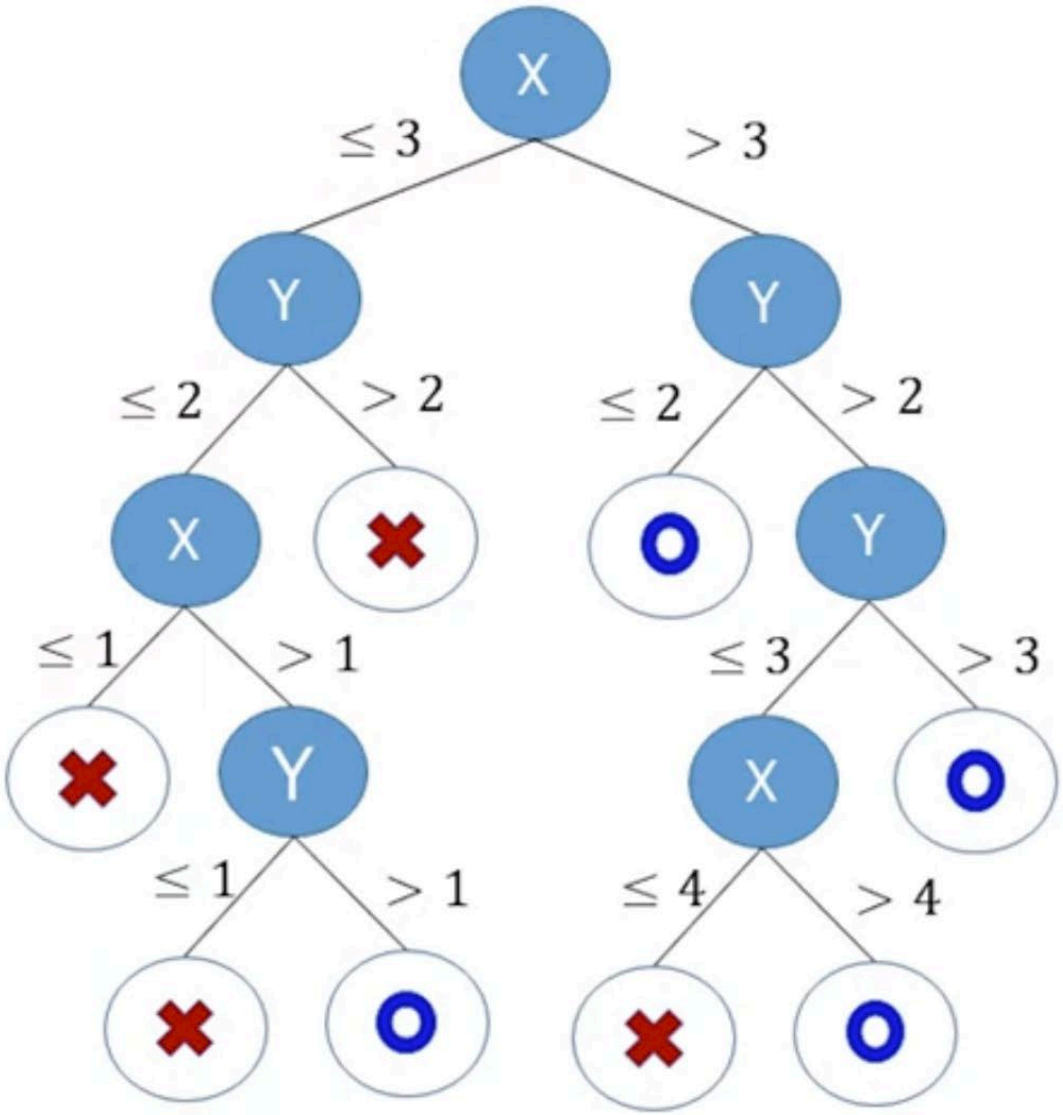
petal length (cm): 4.5

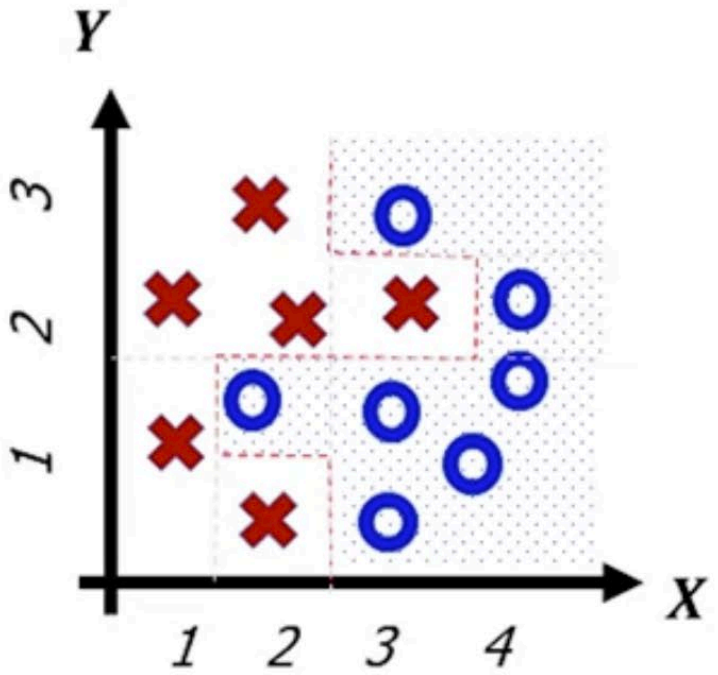




Species counts are: setosa=0, versicolor=38, virginica=3

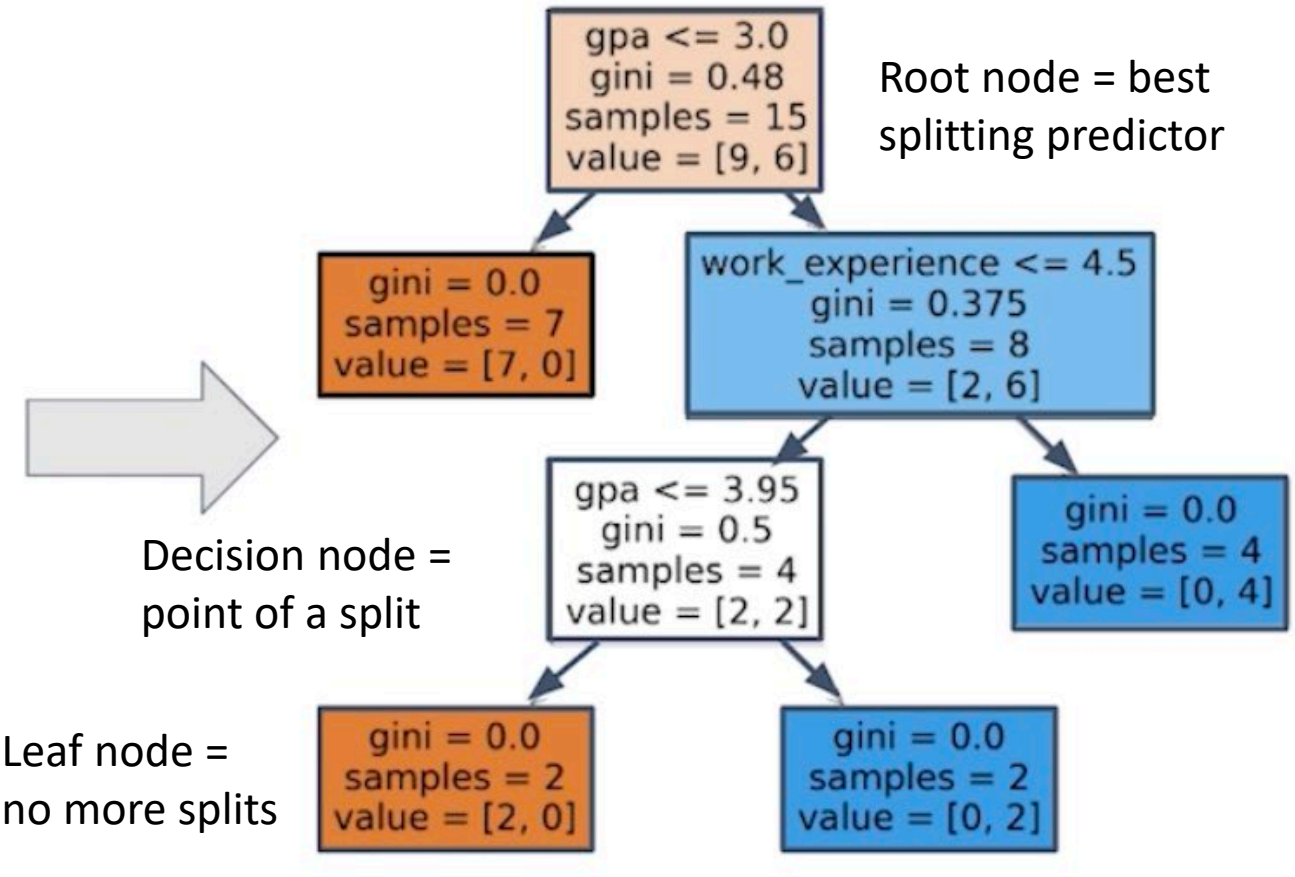
Prediction is **versicolor** as it is the majority class


X
≤ 3
> 3
Y
Y
≤ 2
> 2
≤ 2
> 2
X
Y
≤ 1
> 1
≤ 3
> 3
Y
X
≤ 1
> 1
≤ 4
> 4

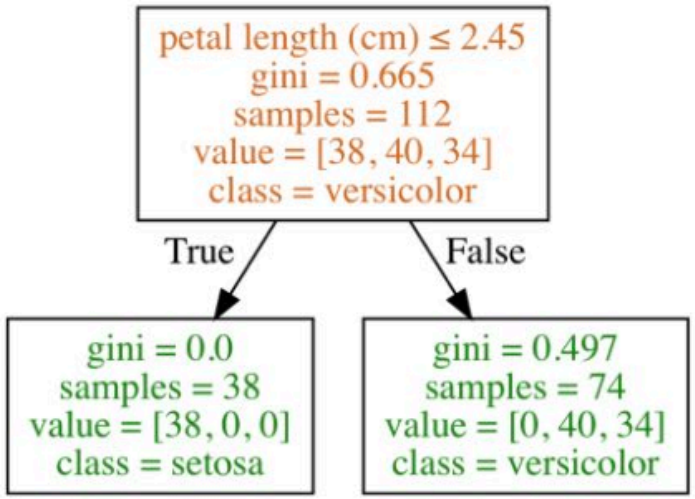

Y
3
2
1
1
2
3
4
X

Decision tree example

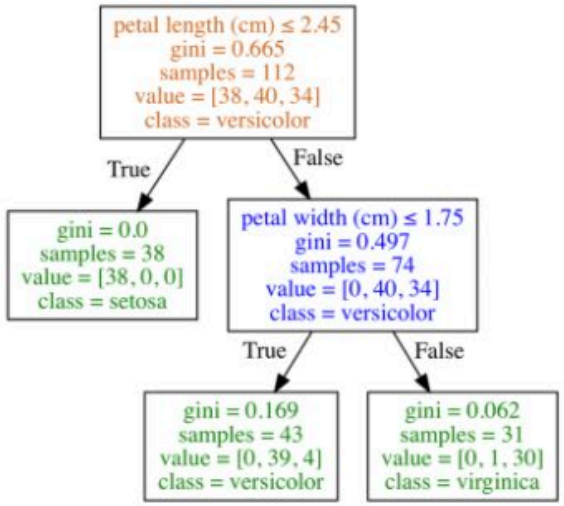
gmat	gpa	work_experience	admitted
580	2.7	4	0
660	3.3	6	1
740	3.3	5	1
590	1.7	4	0
660	4.0	4	1
540	2.7	2	0
690	2.3	1	0
550	2.7	1	0
580	2.3	2	0
620	2.7	2	0
710	3.7	5	1
660	3.3	5	1
780	4.0	3	1
680	3.3	4	0



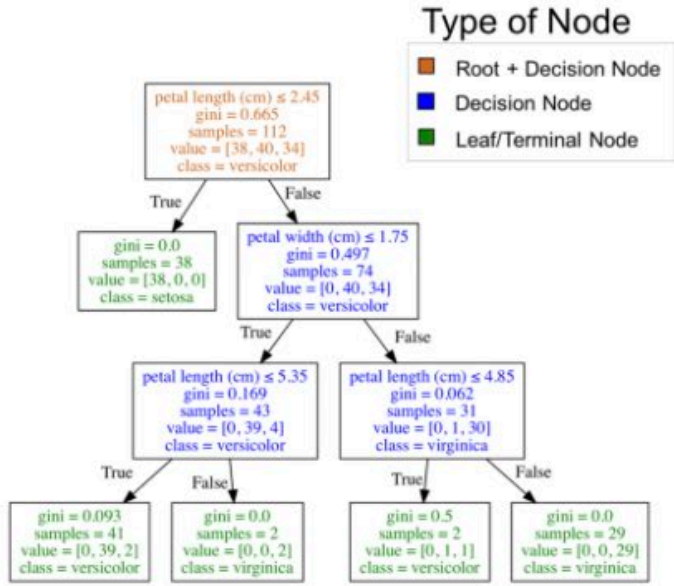
Depth of Classification Trees



Depth = 1

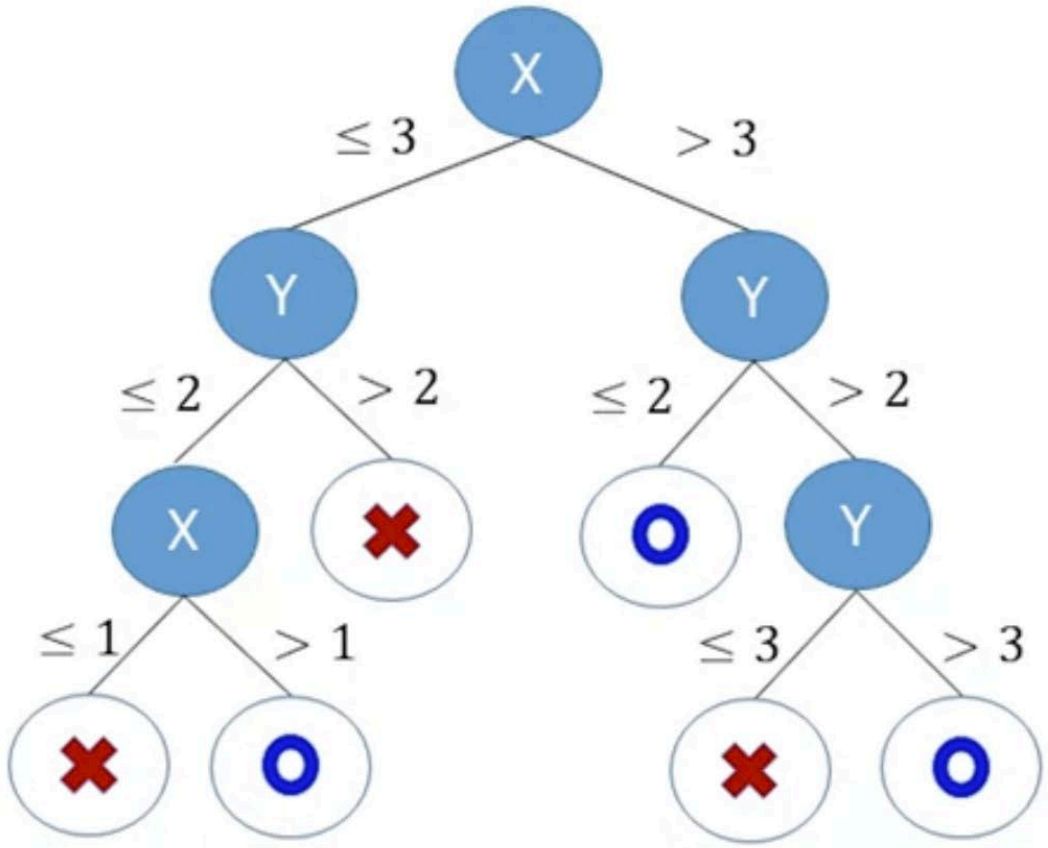


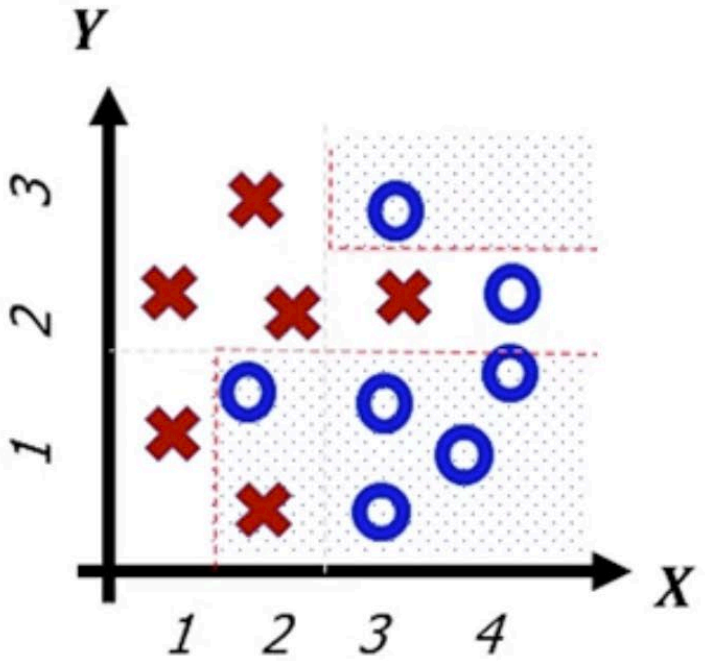
Depth = 2



Depth = 3

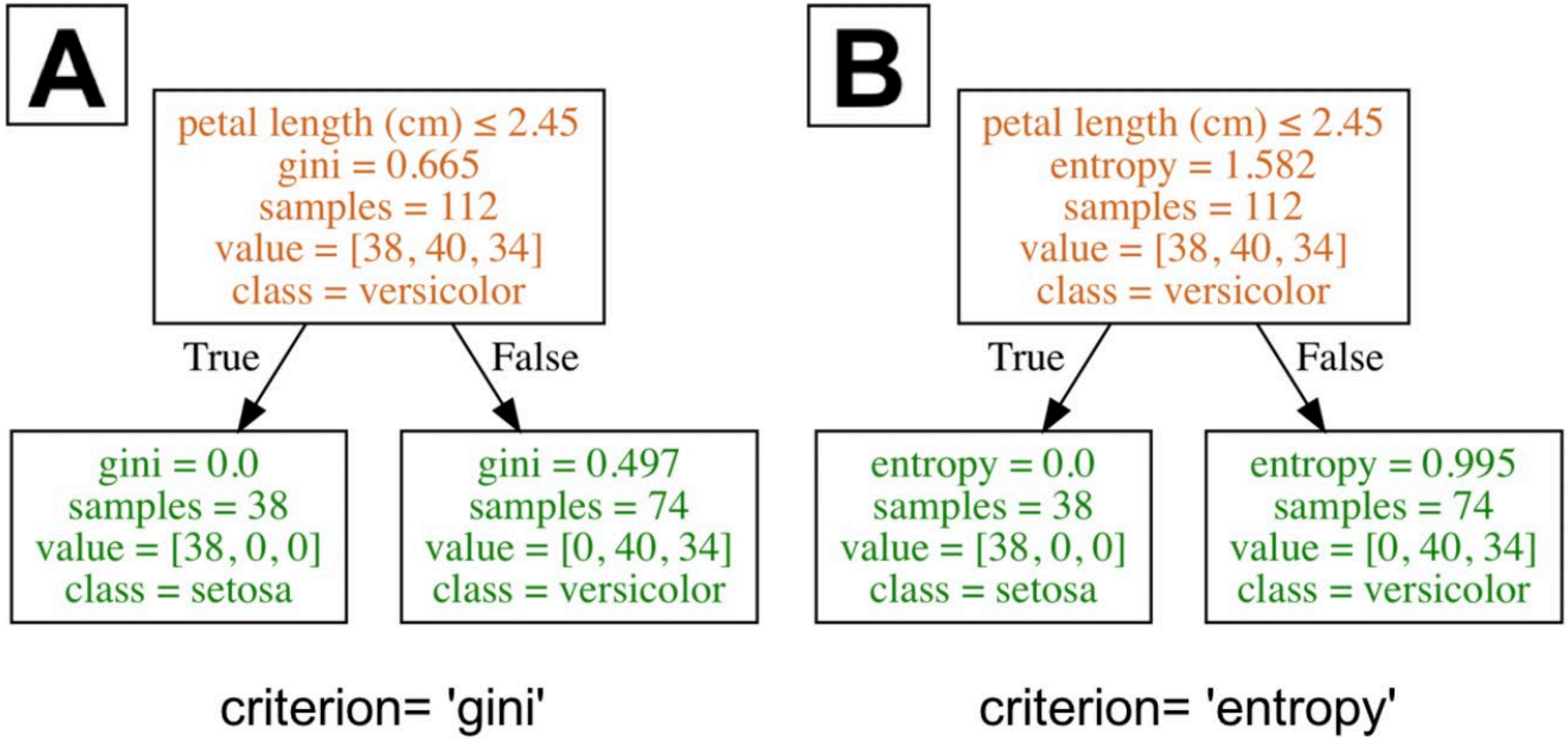
Classification trees of different depths fit on the IRIS dataset.





MAX_DEPTH=3

The Selection Criterion

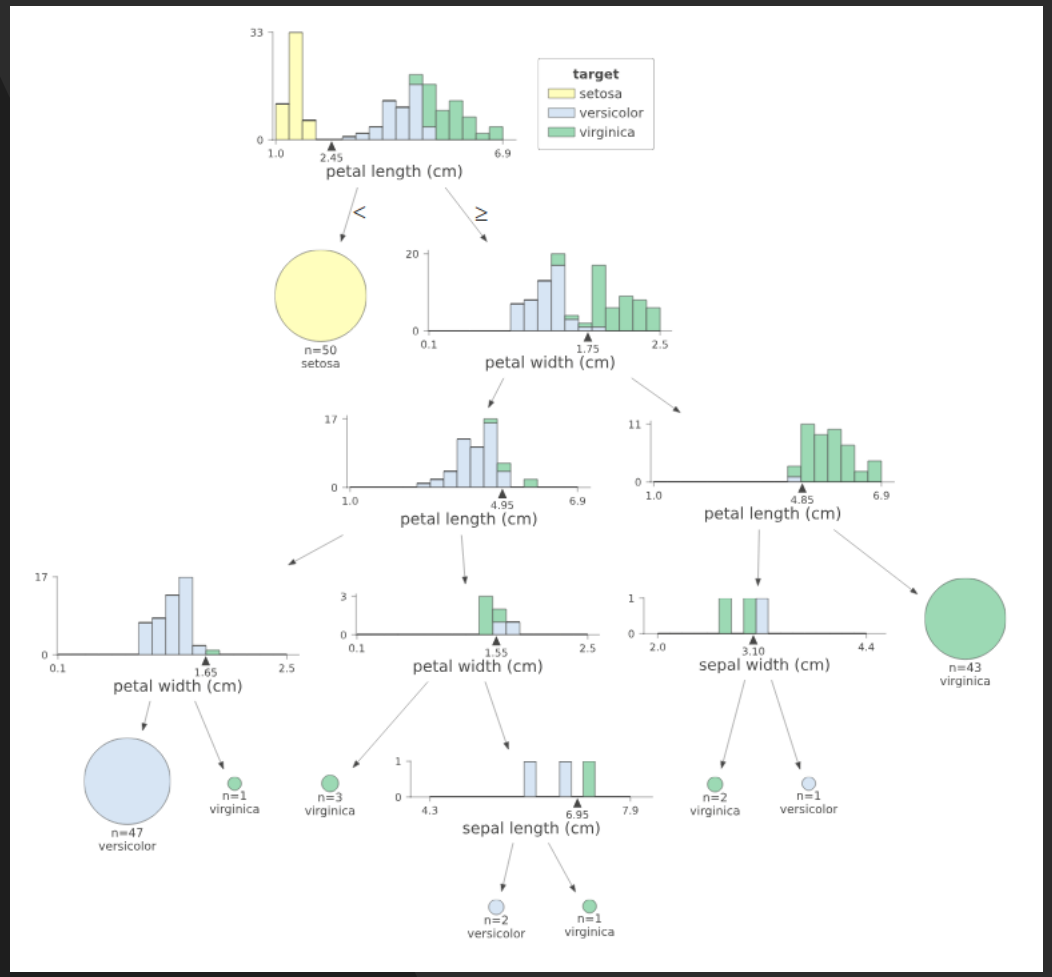


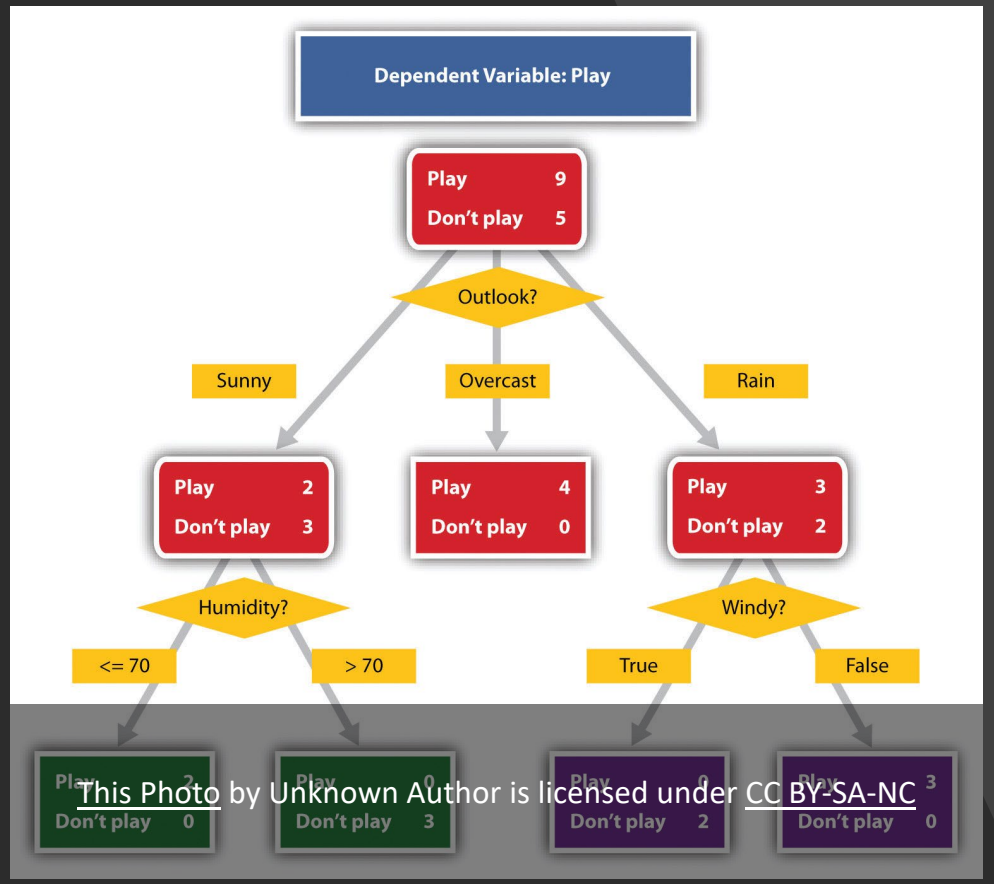
criterion= 'gini'

criterion= 'entropy'

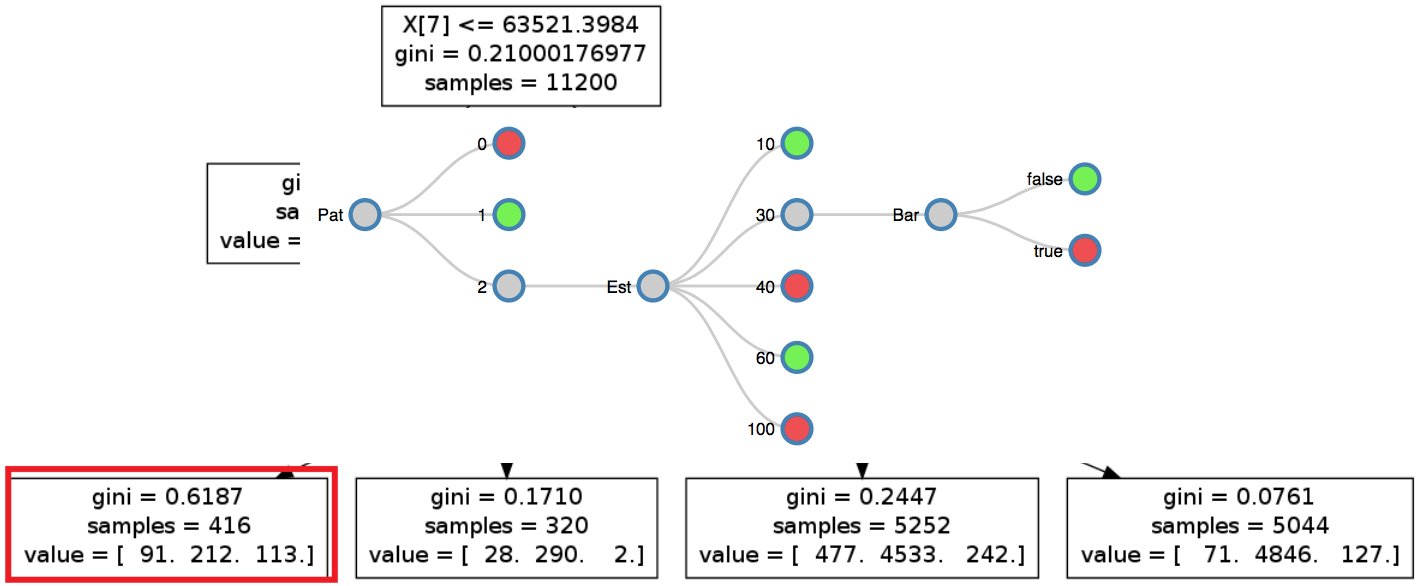
This section answers how information gain and two criterion gini and entropy are calculated.

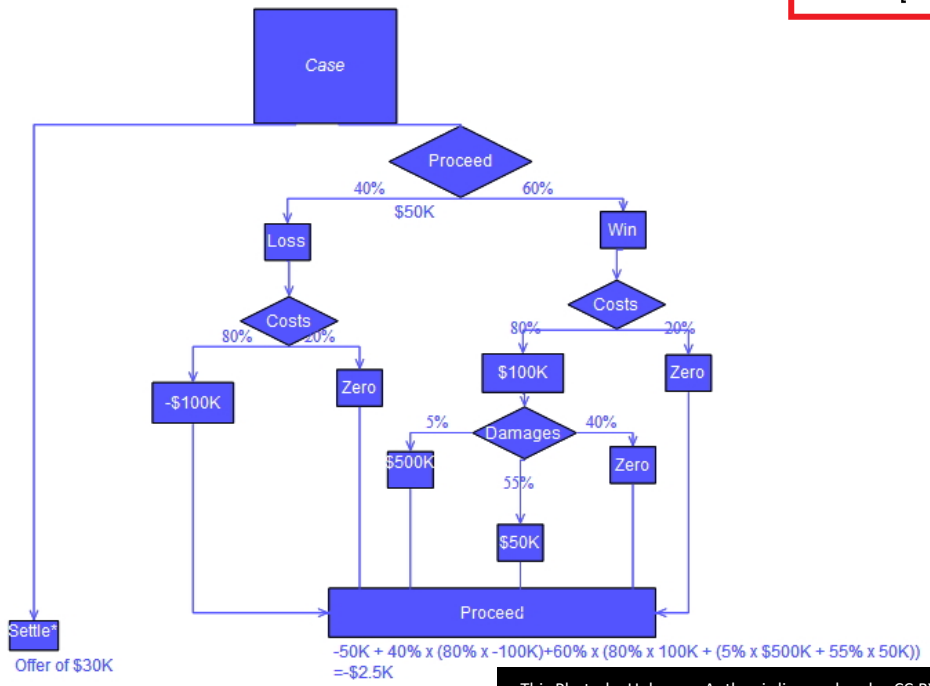
Decision Trees are Everywhere!

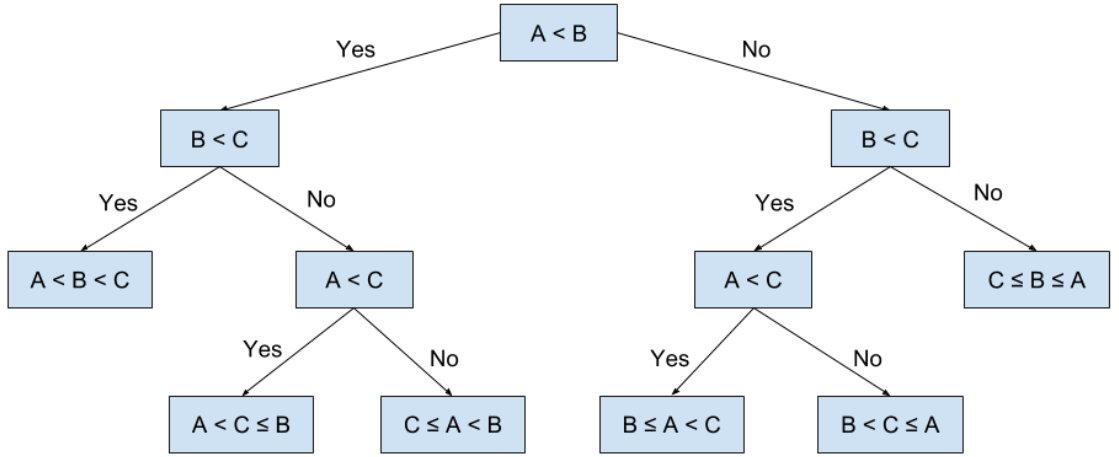


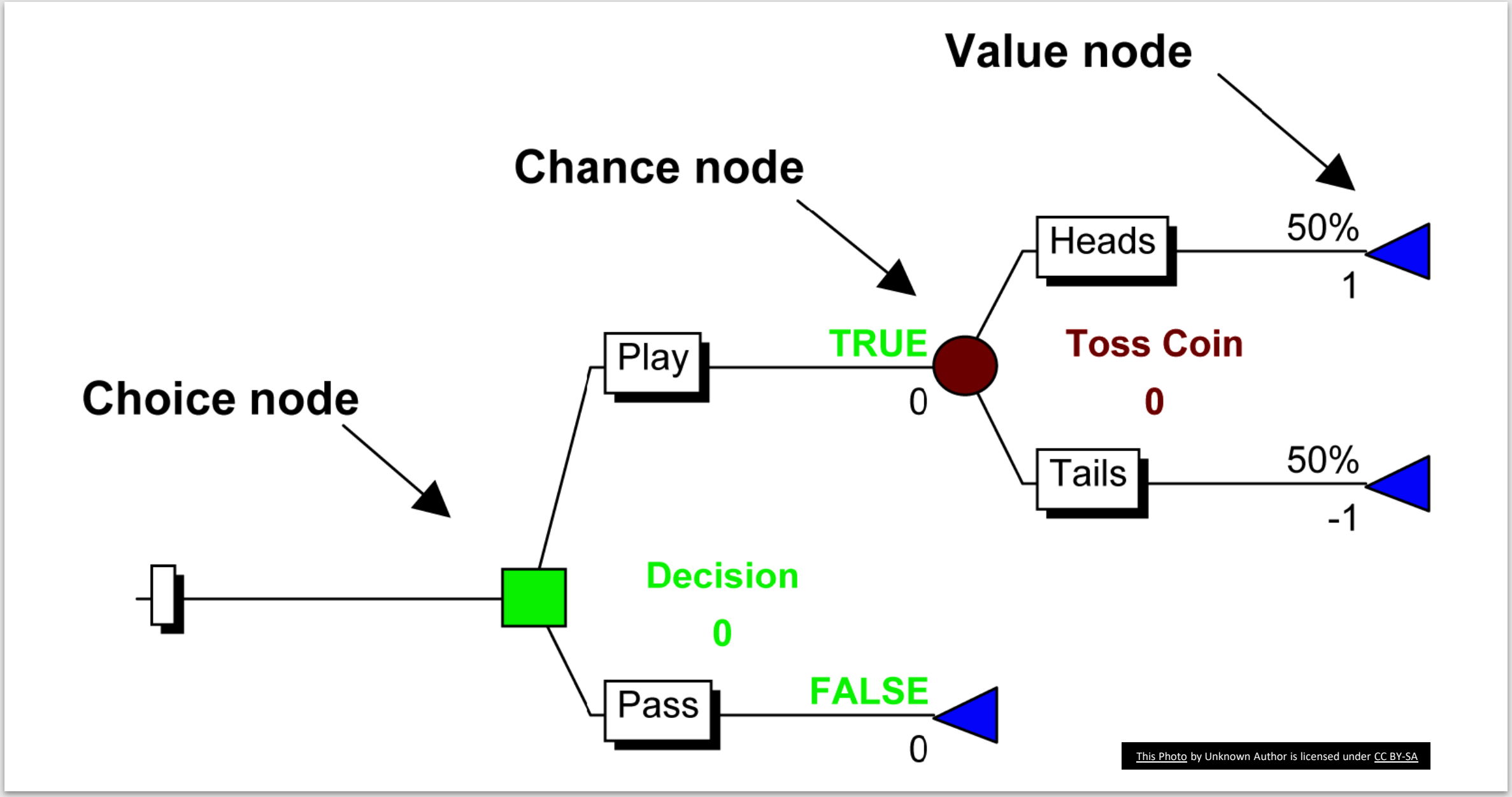


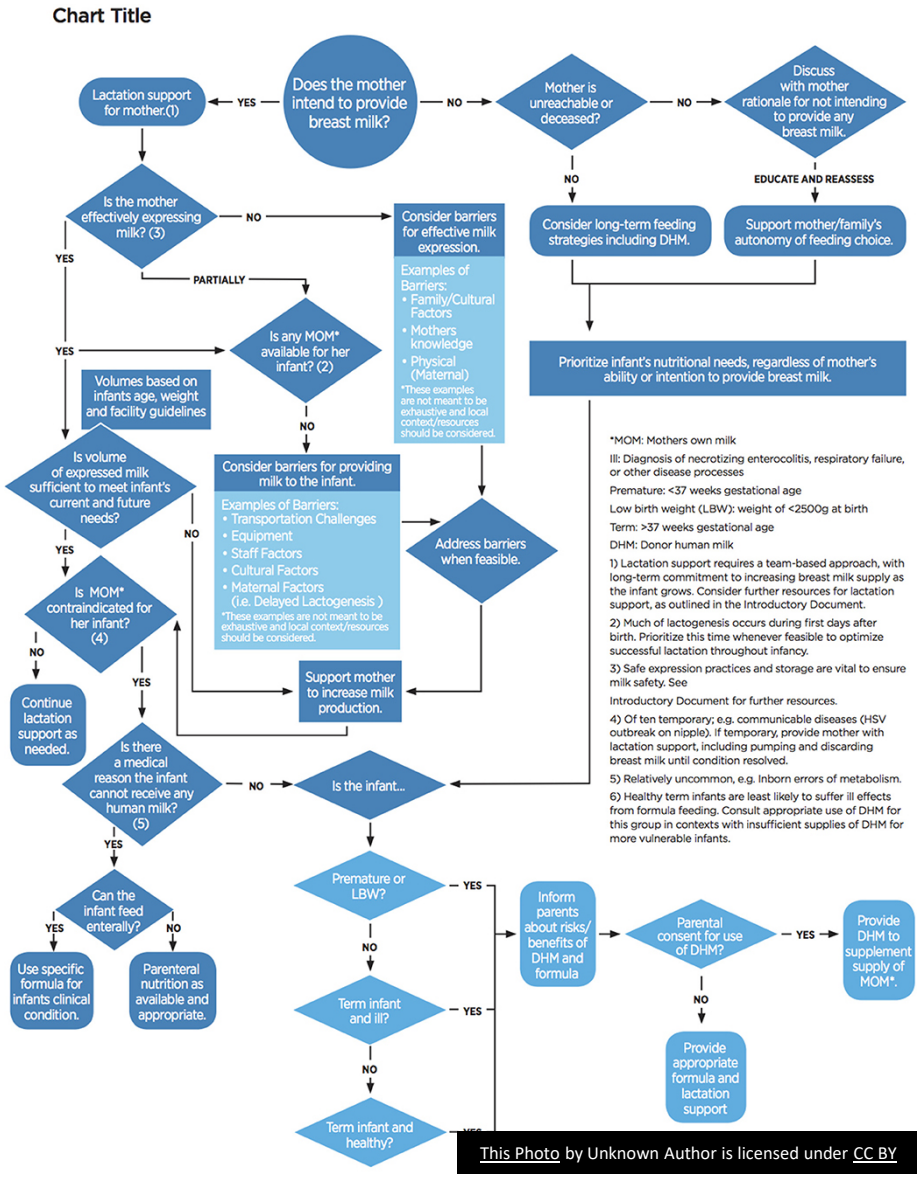
Can be horizontal or vertical











Decision Tree

Advantages:

1. Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
2. A decision tree does not require normalization of data.
3. A decision tree does not require scaling of data as well.
4. Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
5. A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.

- Disadvantages:
  - Tendency to overfit the data
  - You can choose in advance depth of tree and set tolerable level of impurity, but difficult to choose optimal solution
  - Or you can prune a fully-formed tree (regularization) but again difficult to find optimal
  - Choosing optimal tree NP-incomplete even for small datasets (computationally intractable)

Decision Tree

Disadvantage:

1. A small change in the data can cause a large change in the structure of the decision tree causing instability.
2. For a Decision tree sometimes calculation can go far more complex compared to other algorithms.
3. Decision tree often involves higher time to train the model.
4. Decision tree training is relatively expensive as the complexity and time has taken are more.
5. The Decision Tree algorithm is inadequate for applying regression and predicting continuous values.

Cost-sensitive decision tree

- In a DT we are looking for a clean separation of examples into groups where a group of examples of all 0 or all 1 class is the **purest**, and a 50–50 mixture of both classes is the **least pure.**
- The calculation of a purity measure involves calculating the probability of an example of a given class being misclassified by a split.
- Calculating these probabilities involves summing the number of examples in each class within each group.
- The splitting criterion can be updated to take the purity of the split into account as well as the importance of each class (**using weights**).

Cost-sensitive decision tree

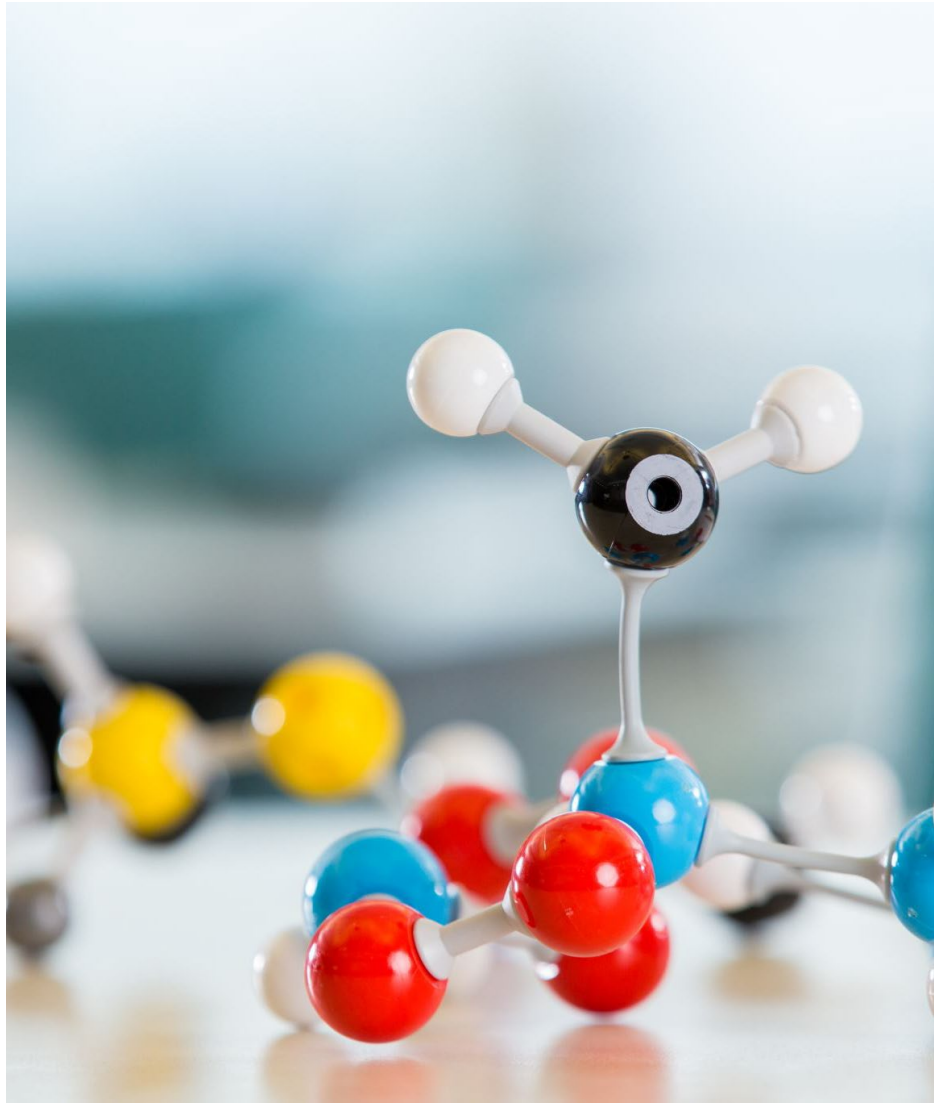
- **Cost-sensitive decision trees gained a great deal of attention.**
- The split points of the tree are chosen to best separate examples into two groups with minimum mixing.
- When both groups are dominated by examples from one class, the criterion used to select a split point will see good separation, when in fact, the examples from the minority class are being ignored.
- This can be overcome by modifying the criterion used to evaluate split points to take the importance of each class into account.

How to assign class weights

- Domain expertise, gained by speaking to domain experts.
- Tuning, determined by a hyperparameter search such as a grid search.
- Heuristic, specified using a general best practice.
- A common practice is to use the inverse of the class distribution present in the training dataset.
  - Example: 100 instances in class 0 and only 1 instance in class 1.
  - 1:100 = 0.5:50.
  - Class 0 can have a weight value of 0.5.
  - Class 1 can have a weight value of 50.

Simple Models vs Complex Models

In general, linear models are regarded as simple, non-linear models as complex.

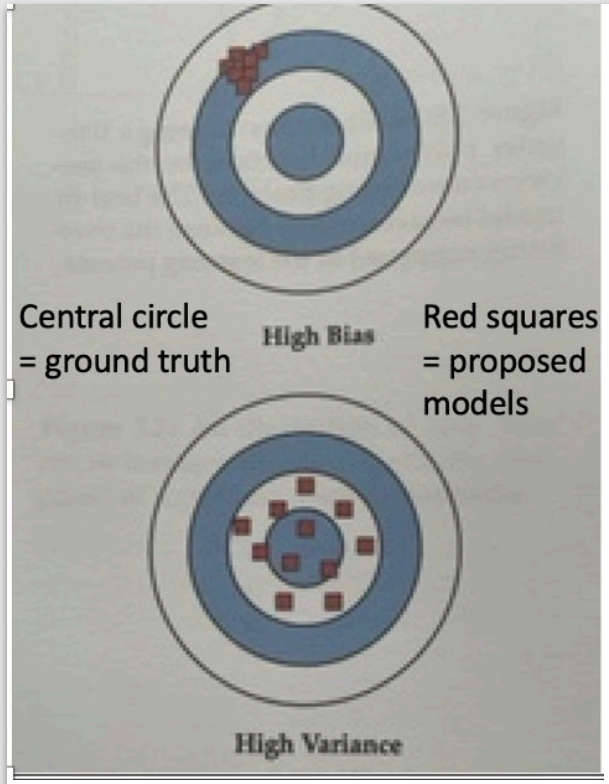


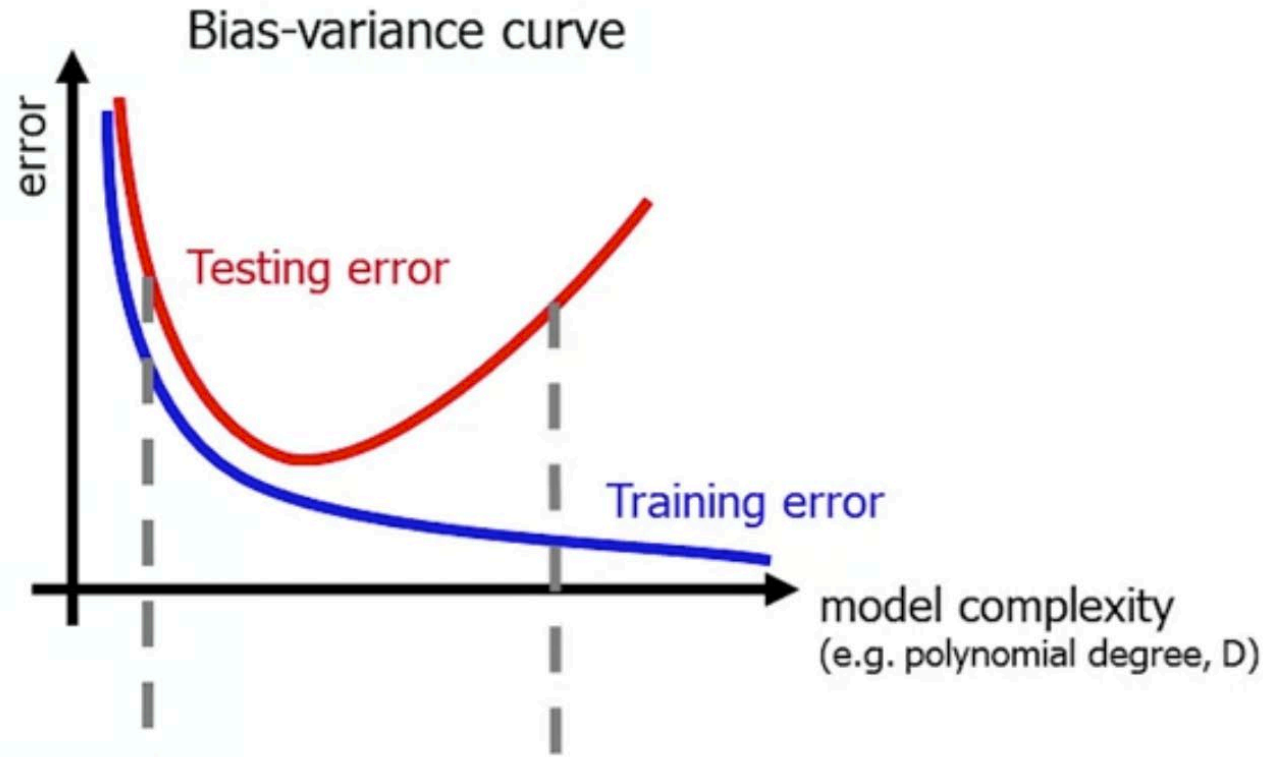
A Good Fit in Machine Learning

Ideally, you want to select a model at the sweet spot between underfitting and overfitting.

This is the goal, but is very difficult to do in practice.

To understand this goal, we can look at the performance of a machine learning algorithm over time as it is learning a training data. We can plot both the skill on the training data and the skill on a test dataset we have held back from the training process.





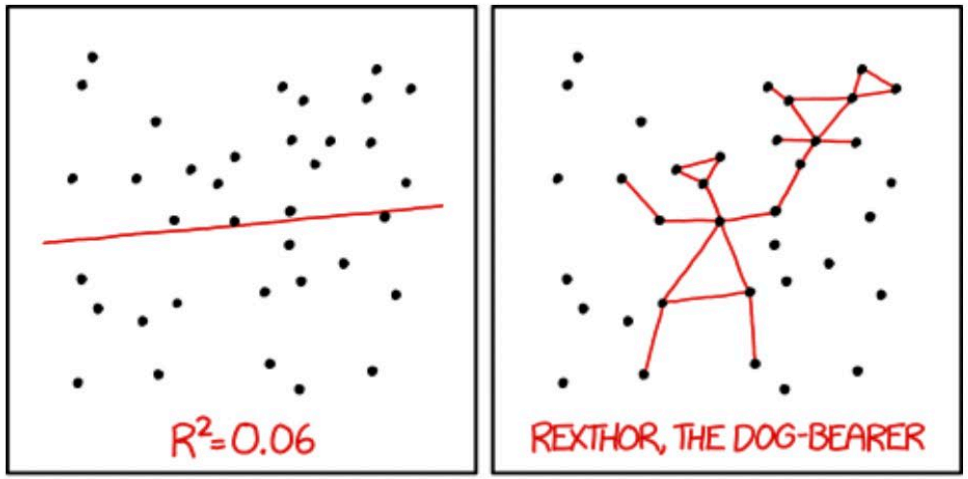
Underfitting (high bias)

- does not fit well
  - too simple
  - too few features

Overfitting (high variance)

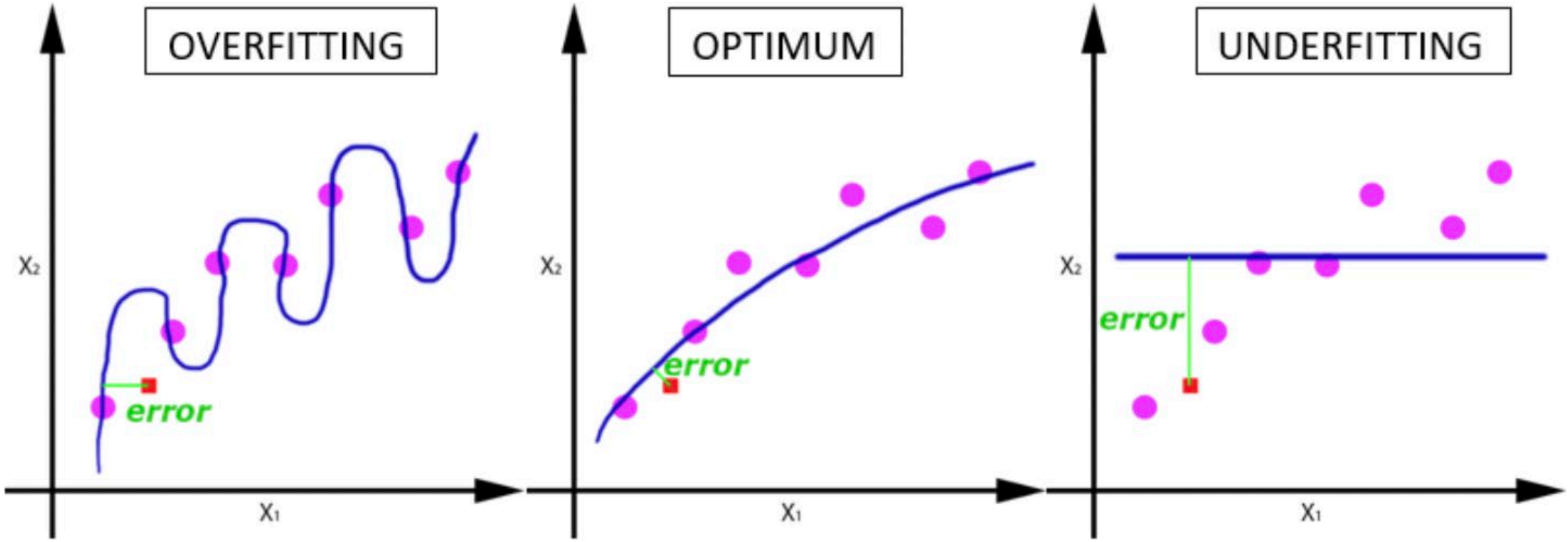
- fits training data, but does not generalise to unseen data
  - too complex
  - too many features

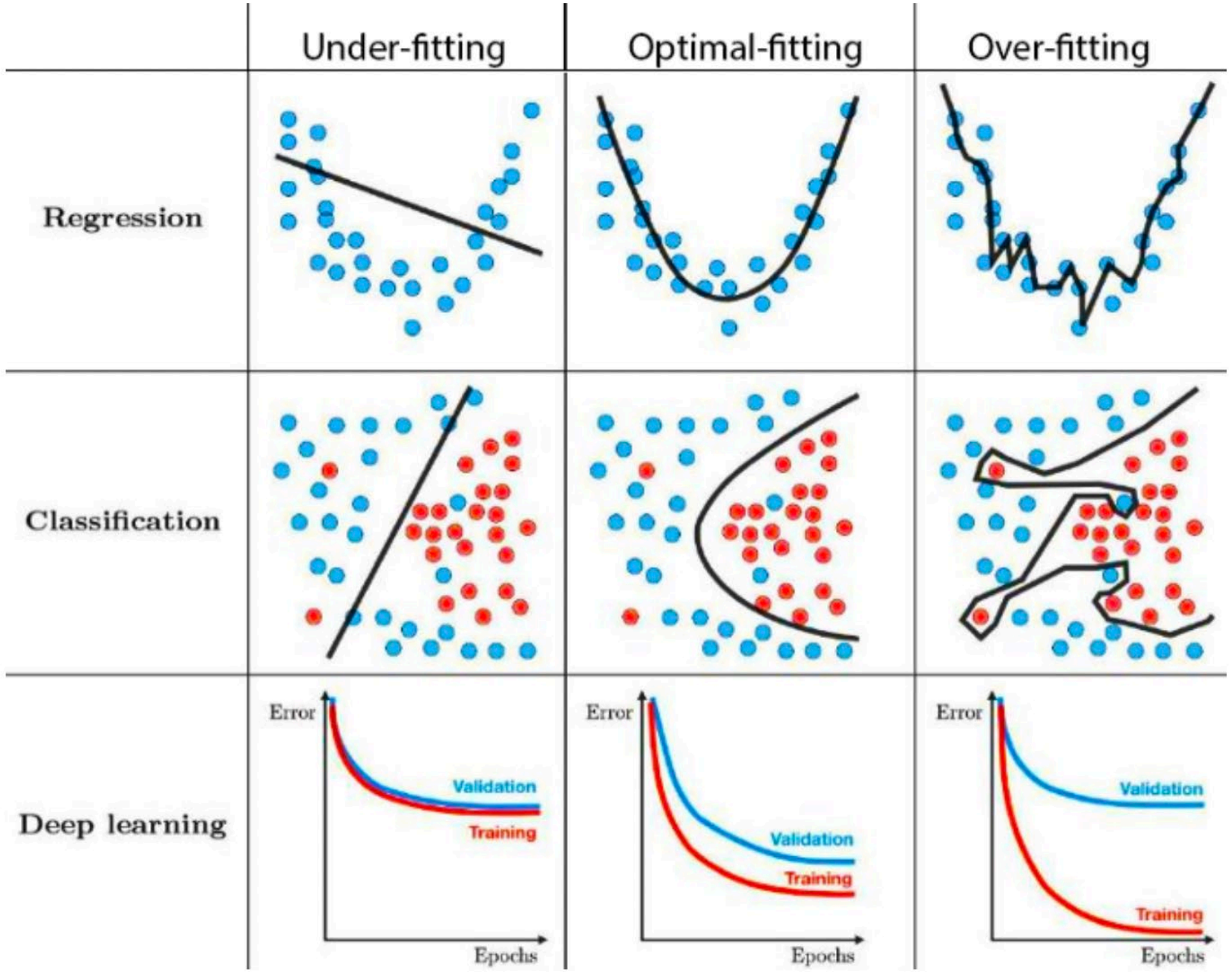
Too much variance in our model will also cause it to fail. It won't generalize well. Instead of capturing just the pattern we care about, it will also capture a lot of extraneous noise that we don't care about. The patterns in the noise will be different from situation to situation. When we try to generalize and apply or model to a new situation, it will have extra error. This is also called **overfitting**. The more complex our model, the greater the risk of overfitting. This was the case in the connect-the-dots



The 95% confidence interval suggests Rexthor's dog could also be a cat, or possibly a teapot.
(https://xkcd.com/1725/)

Goodness of Fit





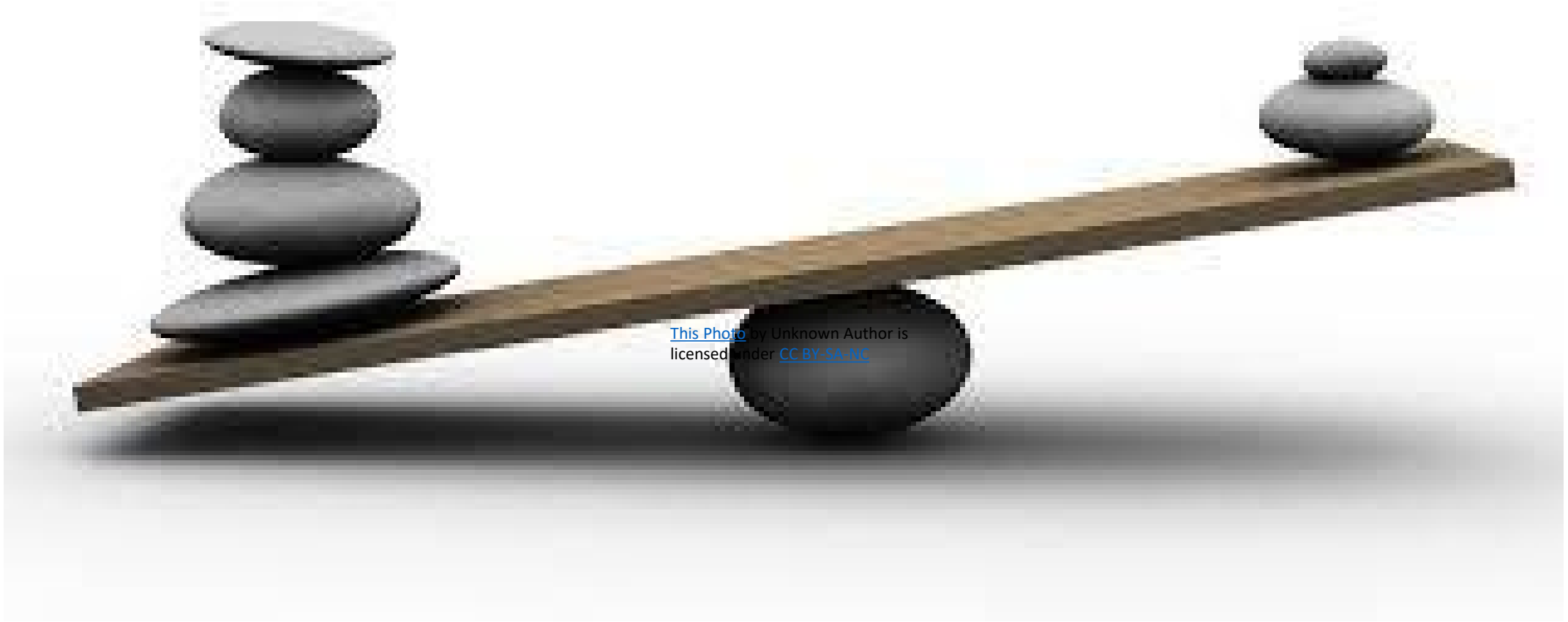
Source: Underfitting, Optimal-fitting and Overfitting for linear regression [1]

Fix against underfitting = try different models
Fix against overfitting = train/test split

The way to protect against underfitting is to try several different types of models. Each model has its own strengths and weaknesses, patterns that it finds more easily and more accurately, and other patterns that it struggles with. By trying a variety of models, we have the best chance at pulling out the patterns hidden in our data. Our best defense against bias is a rich pool of candidate models.

To protect against overfitting, we make sure to test how well each trained model generalizes. After training it on one set of observations, we then use it to predict the pattern in another set of observations. If the predictions are accurate, then we know our model is good, and our variance is low. If it makes much worse predictions on the test data set then on the training data set, then we know that the model overfitted the training data, that it captured a lot of noise, rather than just signal.

Accuracy is not a good measure for imbalanced datasets



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

Having an imbalanced dataset can lead to a large number of True Negatives where the majority class is predicted correctly. Thus, the F1- Score is used to find the harmonic mean between Precision and Recall metrics to determine the model's overall accuracy. It is represented with the following equation:

$$F1 = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall}\right)$$

Then we need to evaluate the model:-

Confusion Matrix

Predicted Values	Actual Values: Positive (1)	Actual Values: Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

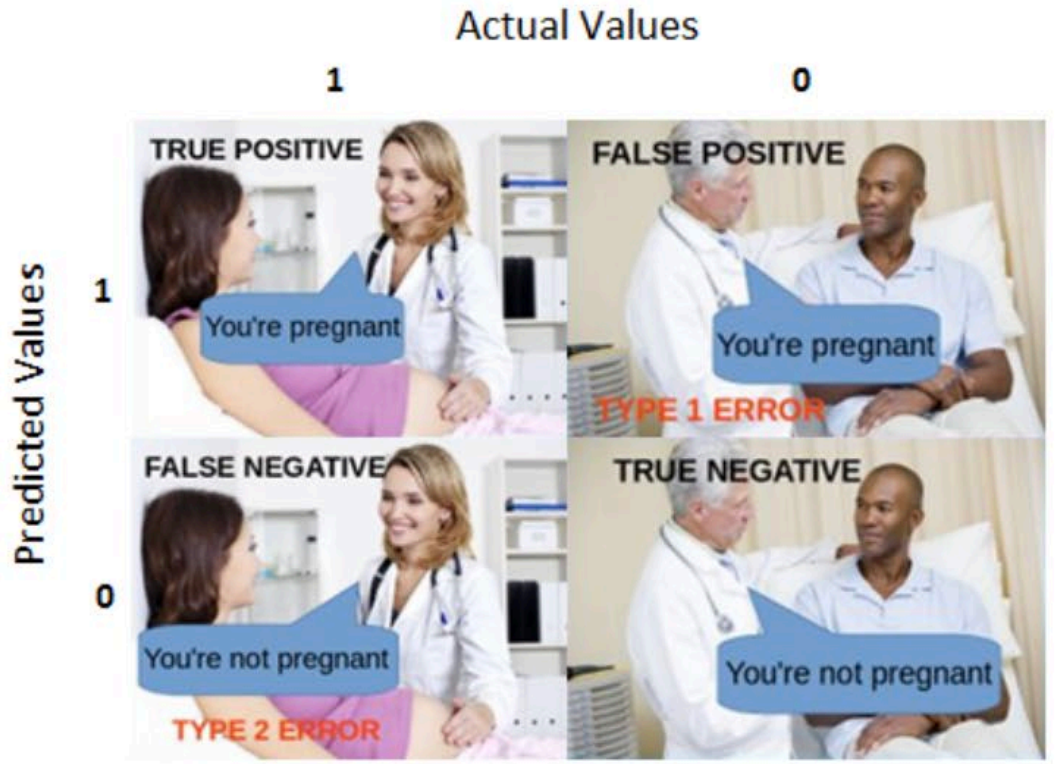


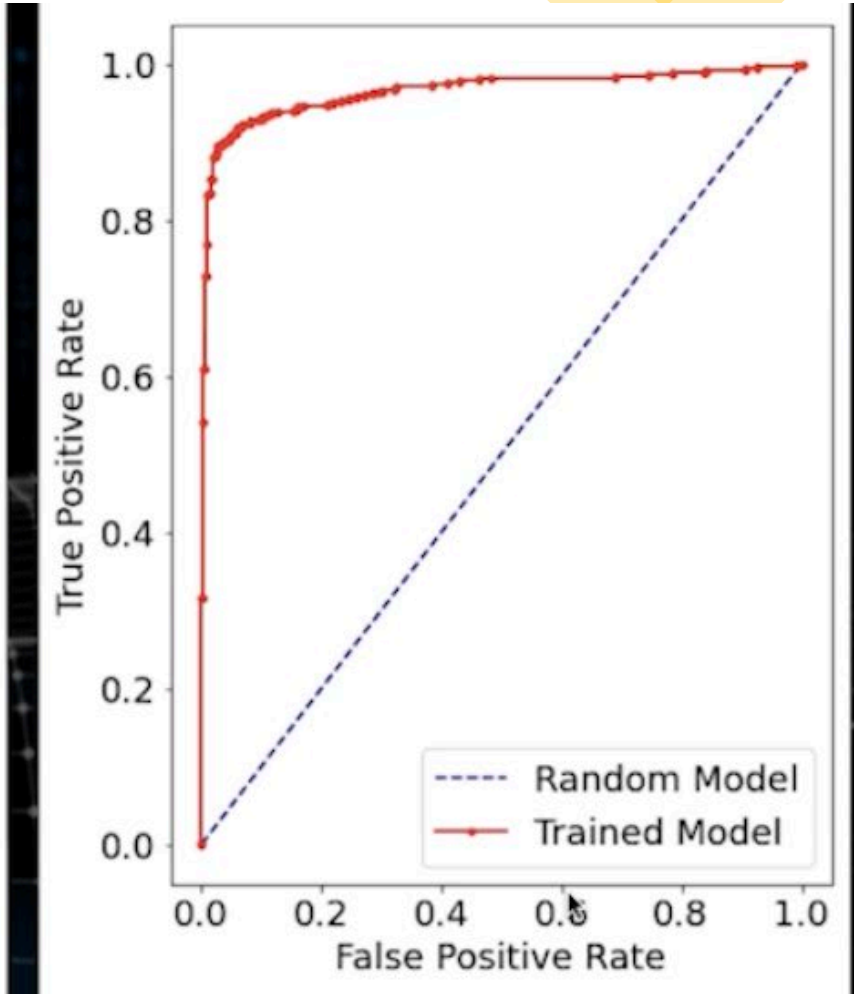
Figure 1 : Confusion Matrix

Predicted class

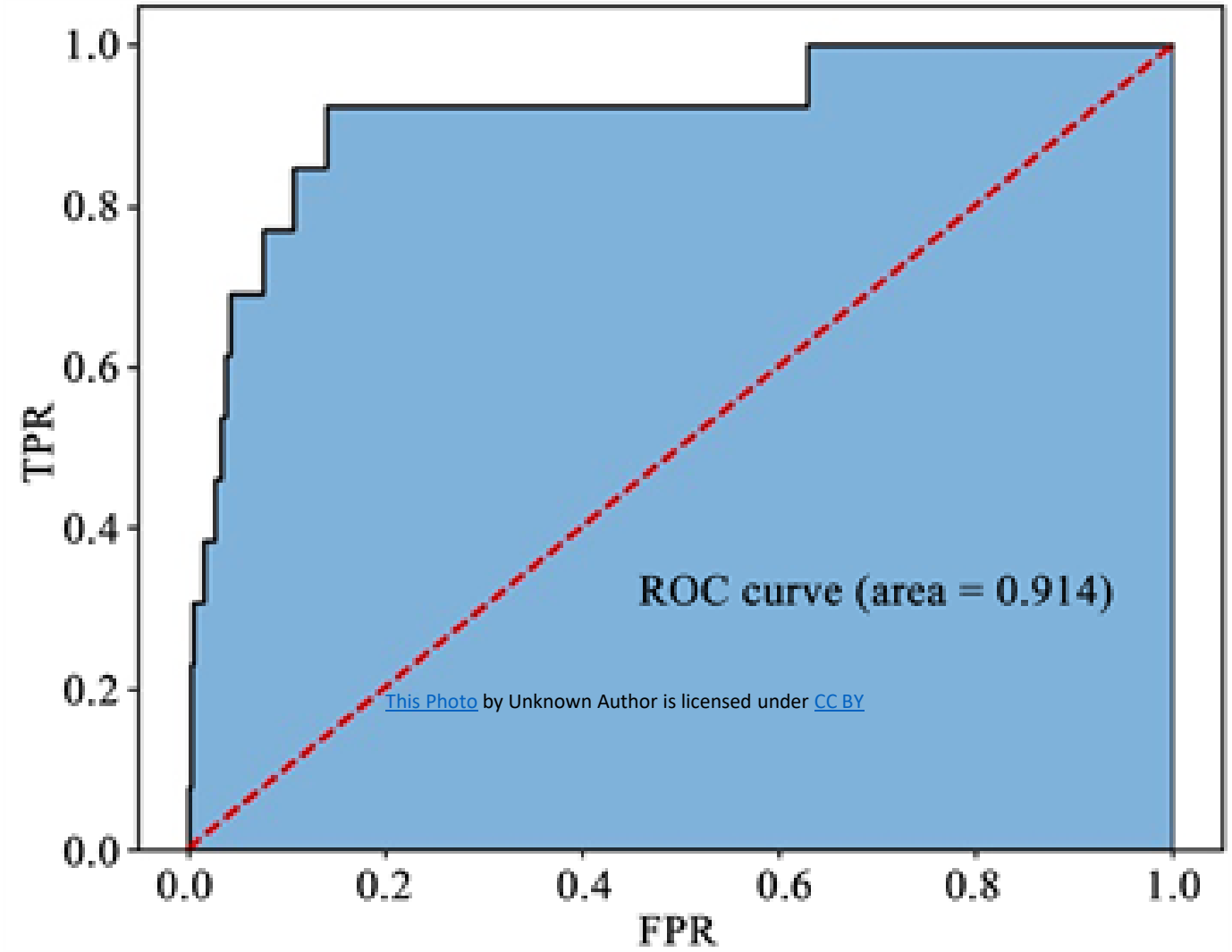
Actual Class	Positive	Negative	
Positive	**True Positive (TP)**	**False Negative (FN)** **Type II Error**	Sensitivity (i.e. Recall) $\frac{TP}{TP+FN}$
Negative	**False Positive (FP)** **Type I Error**	**True Negative (TN)**	Specificity $\frac{TN}{TN+FP}$
	Precision $\frac{TP}{TP+FP}$	Negative Predicted Value $\frac{TN}{TN+FN}$	Accuracy $\frac{TP+TN}{TP+FP+TN+FN}$

ROC Curve

- It is a plot of the false positive rate (x-axis) versus the true positive rate (y-axis) for a number of different candidate threshold values between 0.0 and 1.0.
    - It plots the false alarm rate versus the hit rate.
- True Positive Rate (Sensitivity) = True Positives / (True Positives + False Negatives).
- Specificity = True Negatives / (True Negatives + False Positives).
- False Positive Rate (1 - Specificity) = False Positives / (False Positives + True Negatives).

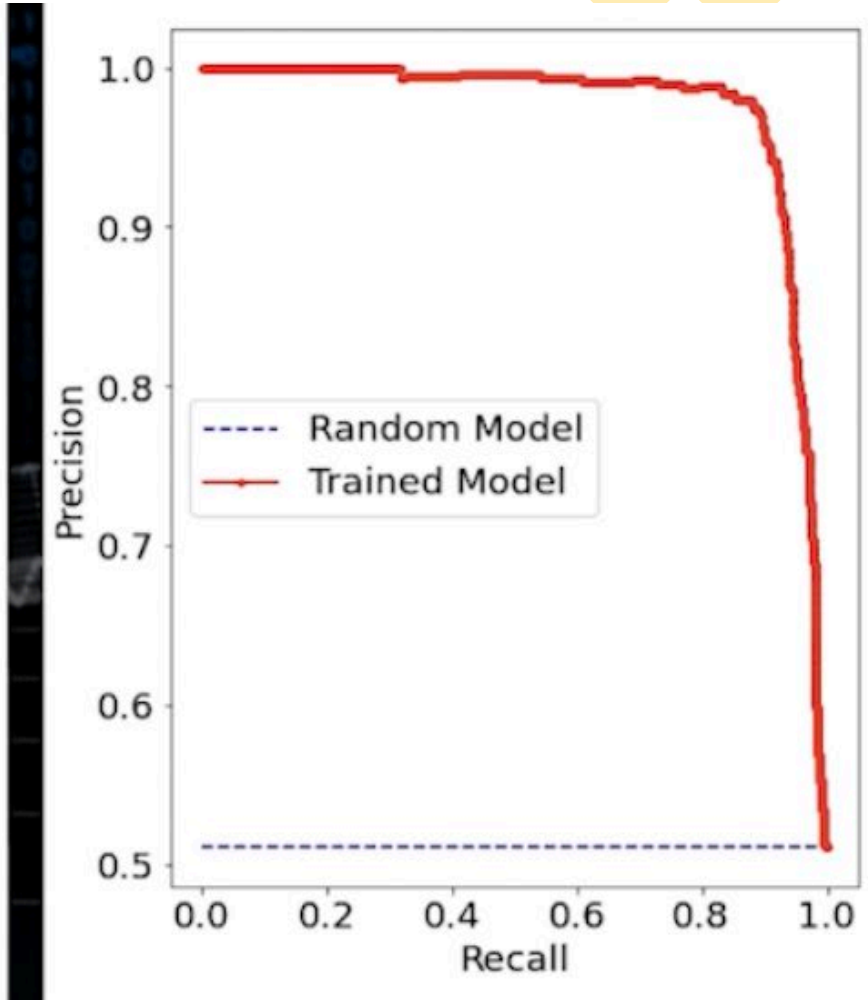


ROC curve example



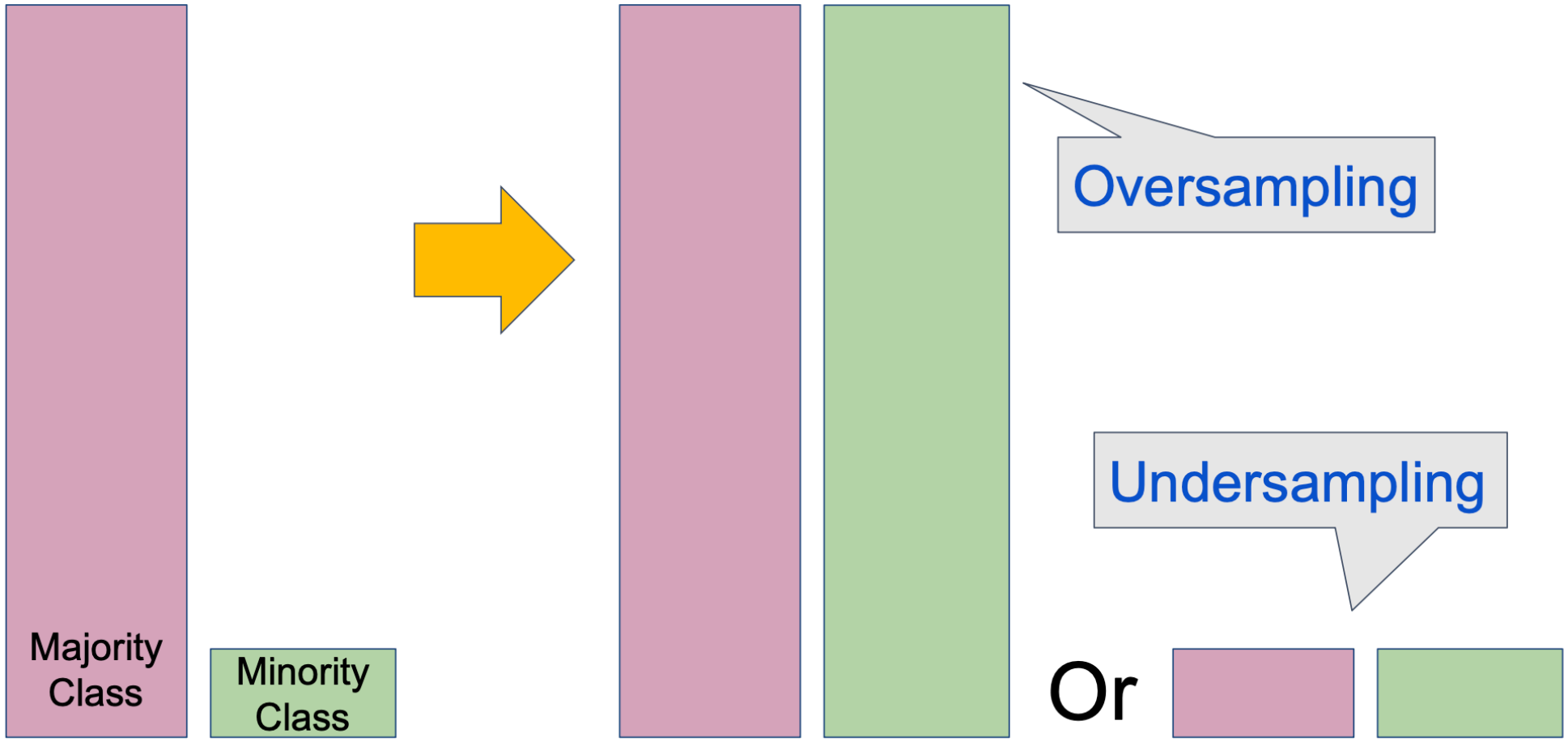
PRC Curve

- Positive Predictive Power (Precision) = True Positives / (True Positives + False Positives)
- Recall (Sensitivity) = True Positives / (True Positives + False Negatives)
- Because of the high class imbalance, we are less interested in the skill of the model at predicting class 0 correctly, e.g. high true negatives
- Key to the calculation of precision and recall is that the calculations **do not make use of the true negatives**
- It is only concerned with the correct prediction of the minority class, class 1



- An evaluation metric must be carefully selected.
- It should best capture what is **important about the model or predictions**
- Accuracy: the number of correct predictions divided by the total number of predictions
  - Accuracy = Correct Predictions / Total Predictions.
  - Value is usually [0,1] or as a percentage [0-100].
  - A model that predicts everything as the majority class would give a high accuracy on an imbalanced test dataset (which is misleading).
- Error Rate = Incorrect Predictions / Total Predictions.
- Accuracy = 1 – Error Rate Error Rate = 1 – Accuracy.

Data sampling



Classification errors

- **Majority class**: negative or no-event assigned the class label 0.
- **Minority class**: positive or event assigned the class label 1.
- In imbalanced classification, classifying a negative case as a positive case is typically far less of a problem than classifying a positive case as a negative case (**false positive vs false negative**).
- Remember: the goal of a classifier on imbalanced binary classification problems is to detect the positive cases correctly, and positive cases represent an exceptional event that we are most interested in.
- Predicting a positive case as a negative case is more harmful and more costly. More important to diagnose someone with a serious asymptomatic illness

Oversampling techniques

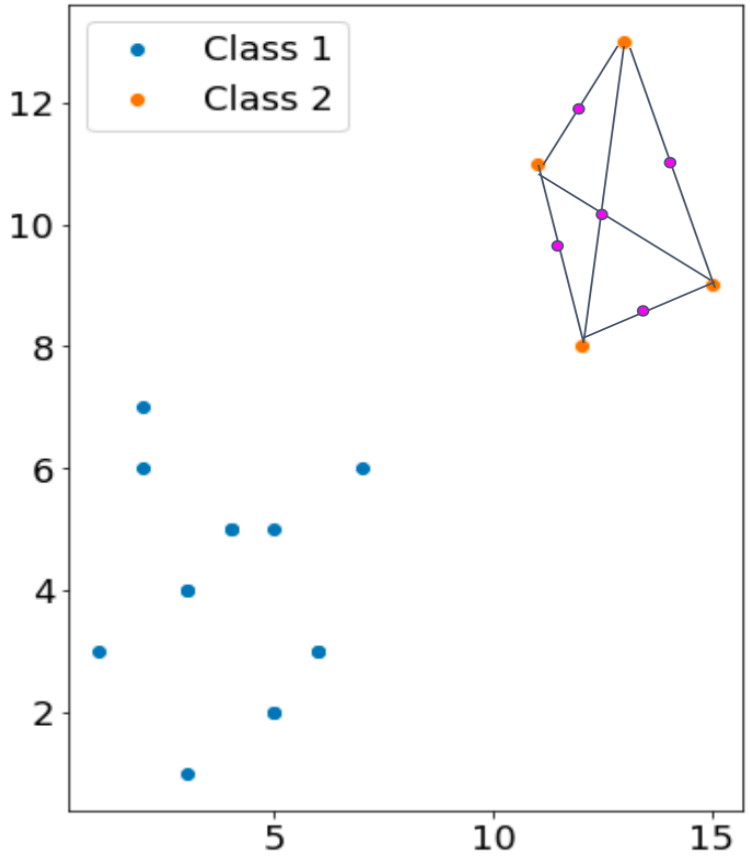
Oversampling methods increase the number of examples in the minority class by duplicating them or synthesising new examples from the original examples in the minority class.

Example methods:

- random oversampling: randomly duplicating examples from the minority class in the training dataset (i.e. sampling with replacement)
- synthetic minority oversampling technique (SMOTE)
- borderline-SMOTE
- borderline oversampling with SVM
- adaptive synthetic sampling (ADASYN).

SMOTE

- Generates new instances (not real) based on existing real instances.
- Needs at least two instances of the class to work.



Undersampling

Undersampling methods delete or select a subset of examples from the majority class.

Example methods:

- random undersampling
- near miss undersampling
- condensed nearest neighbour rule (CNN)
- Tomek links undersampling
- edited nearest neighbours rule (ENN)
- one-sided selection (OSS)
- neighbourhood cleaning rule (NCR).

Some undersampling methods

- **Random undersampling:** as the name suggests, here we randomly delete examples from the majority class in the training dataset until the data is balanced.
- **Near miss undersampling:** uses kNN to select examples from the majority class that have the smallest average distance to the X closest/furthest examples from the minority class.
- **Condensed nearest neighbour rule (CNN):** uses a 1 nearest neighbour rule to find the subset of examples that can correctly classify the entire original dataset.

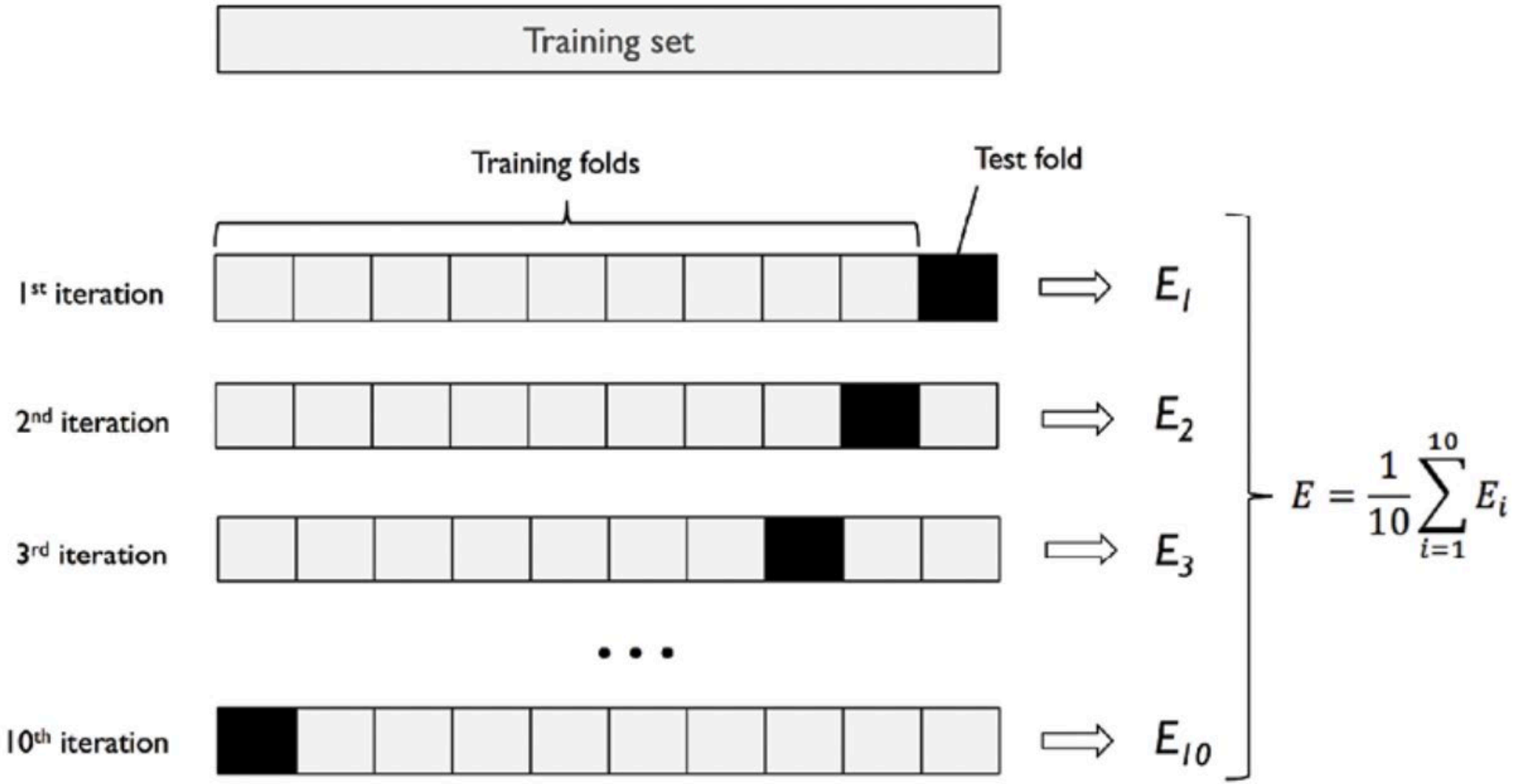
Combining over/undersampling methods

- Usually using one method or the other on the training dataset is effective.
- In some cases applying both types of techniques together can result in better overall performance of a model fit on the resulting transformed dataset.
- The purpose is to remove noisy points along the class boundary from both classes.
- Some common combinations:
  a. SMOTE and random undersampling
  b. SMOTE and Tomek links
  c. SMOTE and edited nearest neighbours rule.

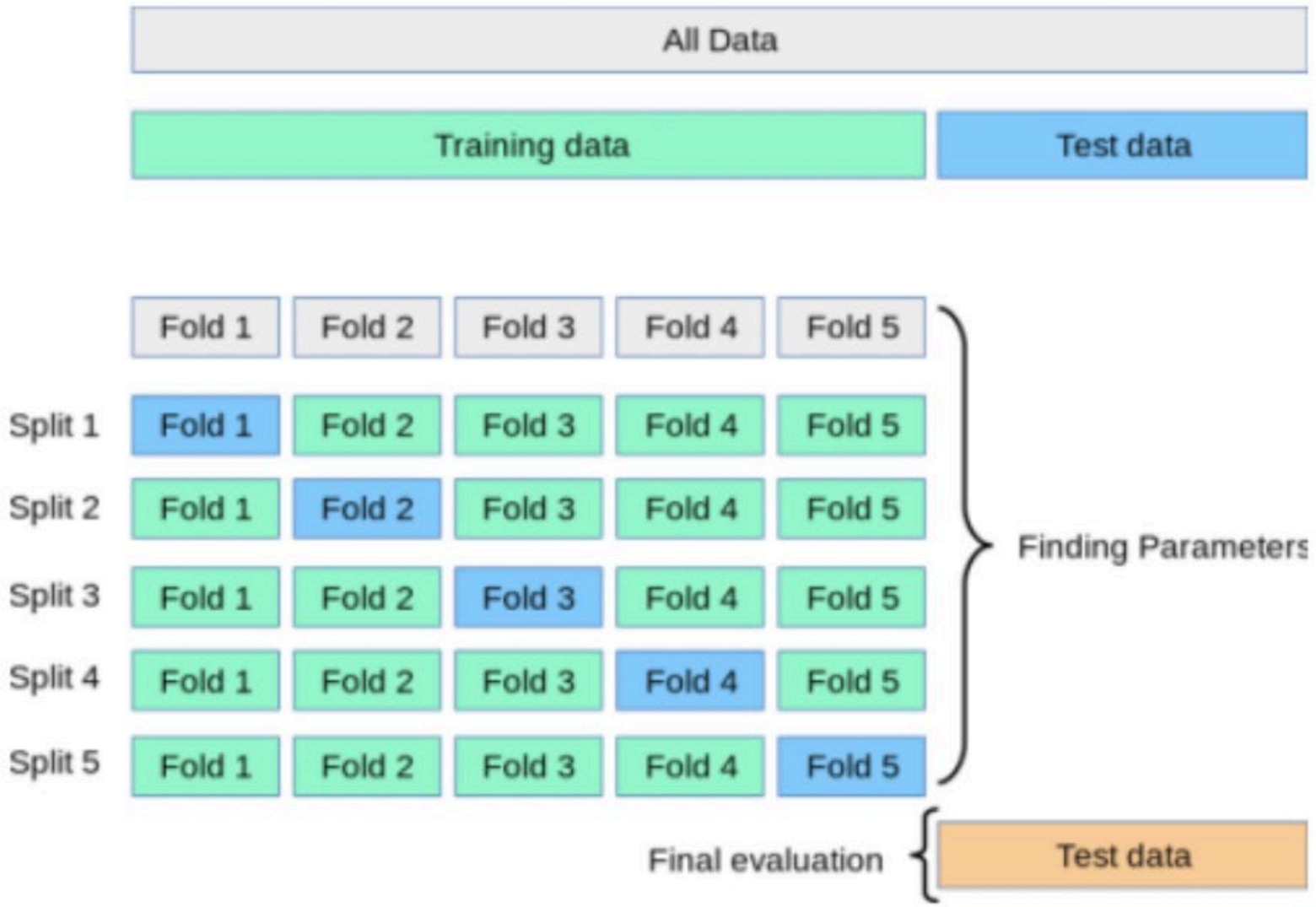
Then we can validate the model with Cross-validation:-

N-fold Cross-Validation/K-fold Cross-validation

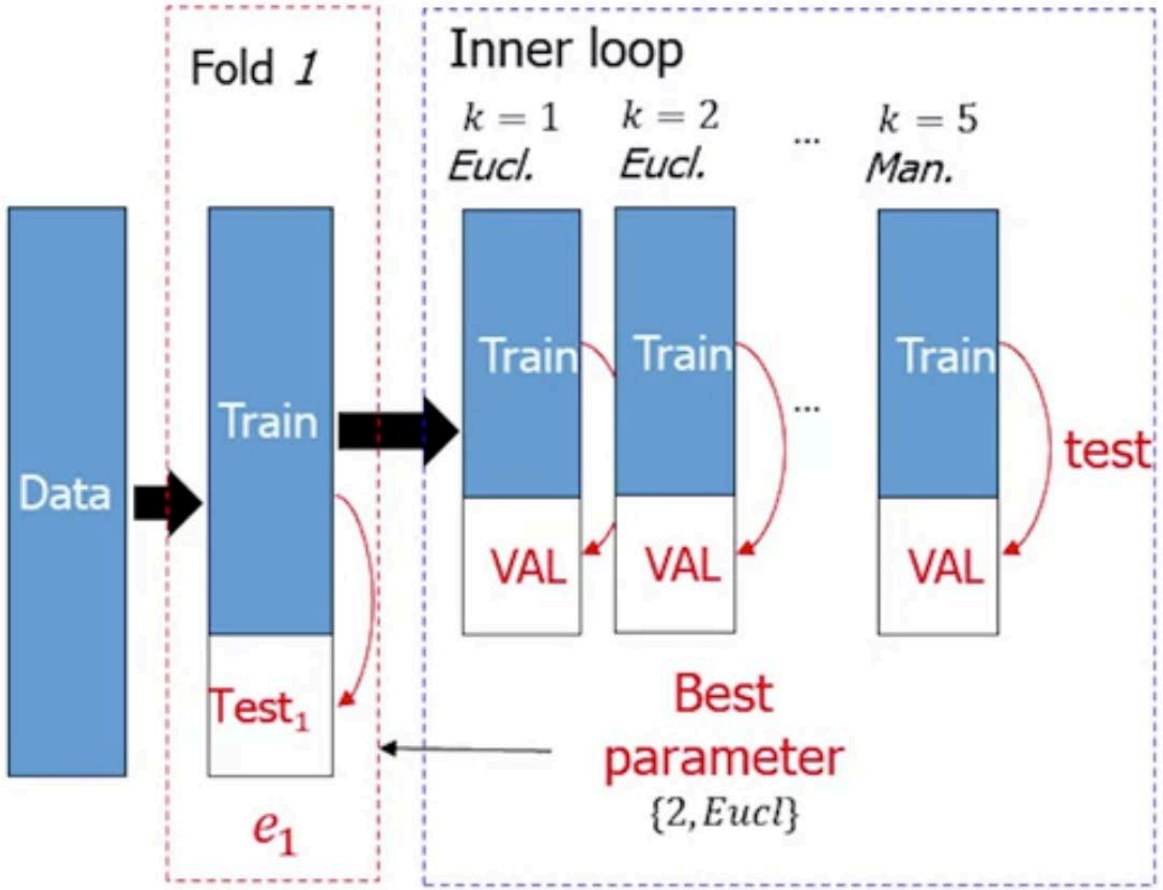
1. The dataset is split into training and test dataset.
2. The training dataset is then split into K-folds.
3. Out of the K-folds, (K-1) fold is used for training
4. 1 fold is used for validation
5. The model with specific hyperparameters is trained with training data (K-1 folds) and validation data as 1 fold. The performance of the model is recorded.
6. The above steps (step 3, step 4 and step 5) is repeated until each of the k-fold got used for validation purpose. This is why it is called k-fold cross validation.

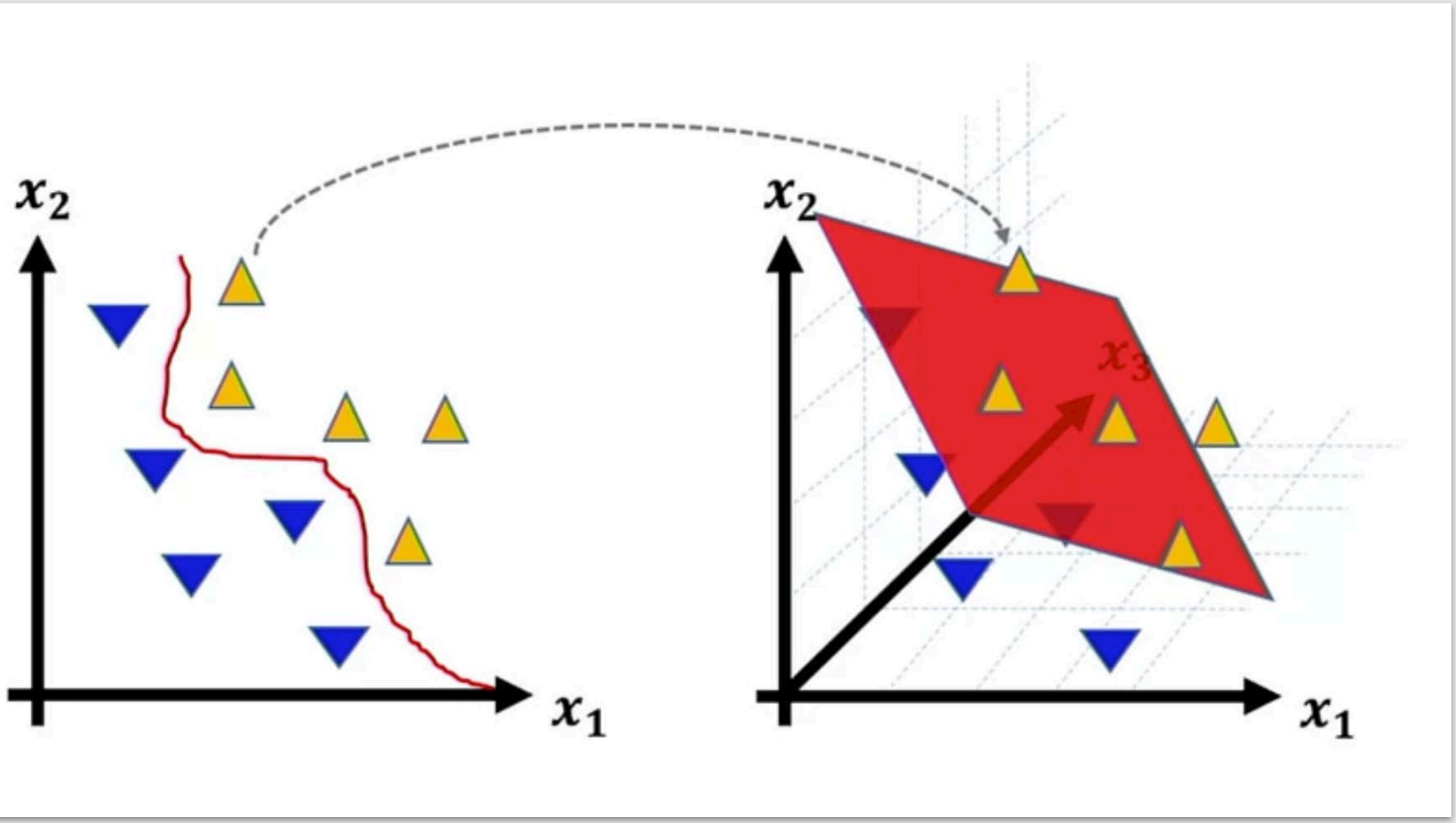

Training set
Training folds
Test fold
1st iteration
2nd iteration
3rd iteration
10th iteration
E_1
E_2
E_3
E_{10}
$E = \frac{1}{10}\sum_{i=1}^{10} E_i$

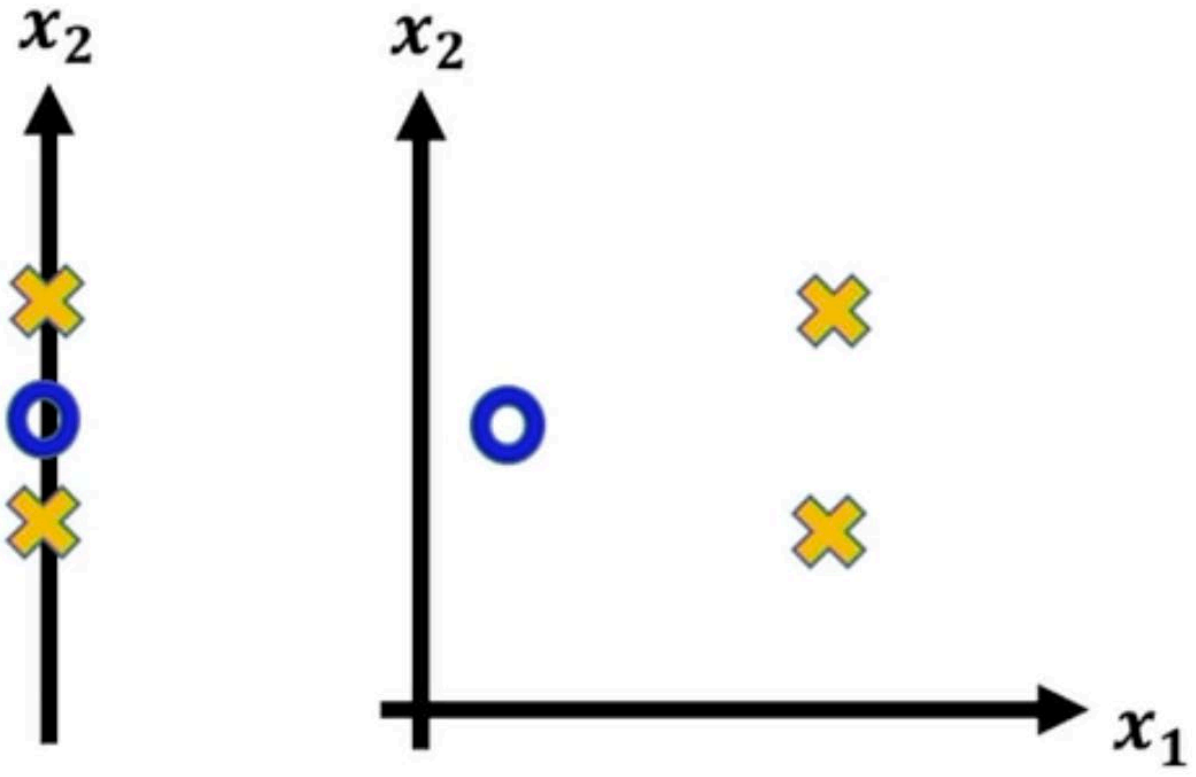
K-fold cross-validation

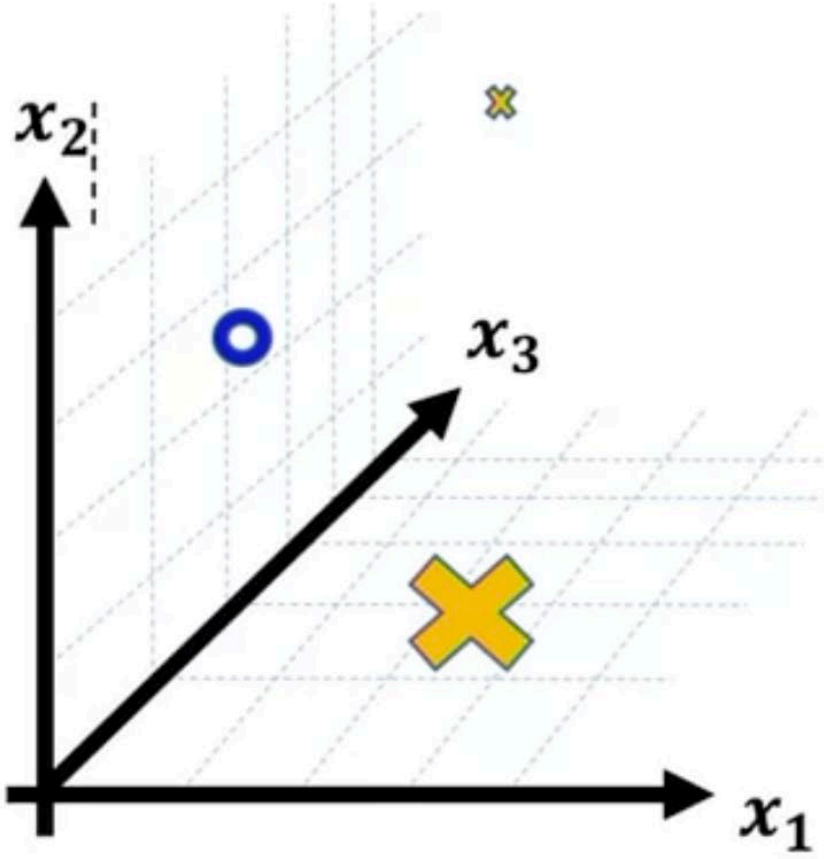


N-fold Nested Cross Validation




x_2
x_1
x_2
x_3
x_1


x_2
x_2
x_1


x_2
x_3
x_1

Skicit-learn Algorithm

BecomingHuman.AI