

Supervised Learning

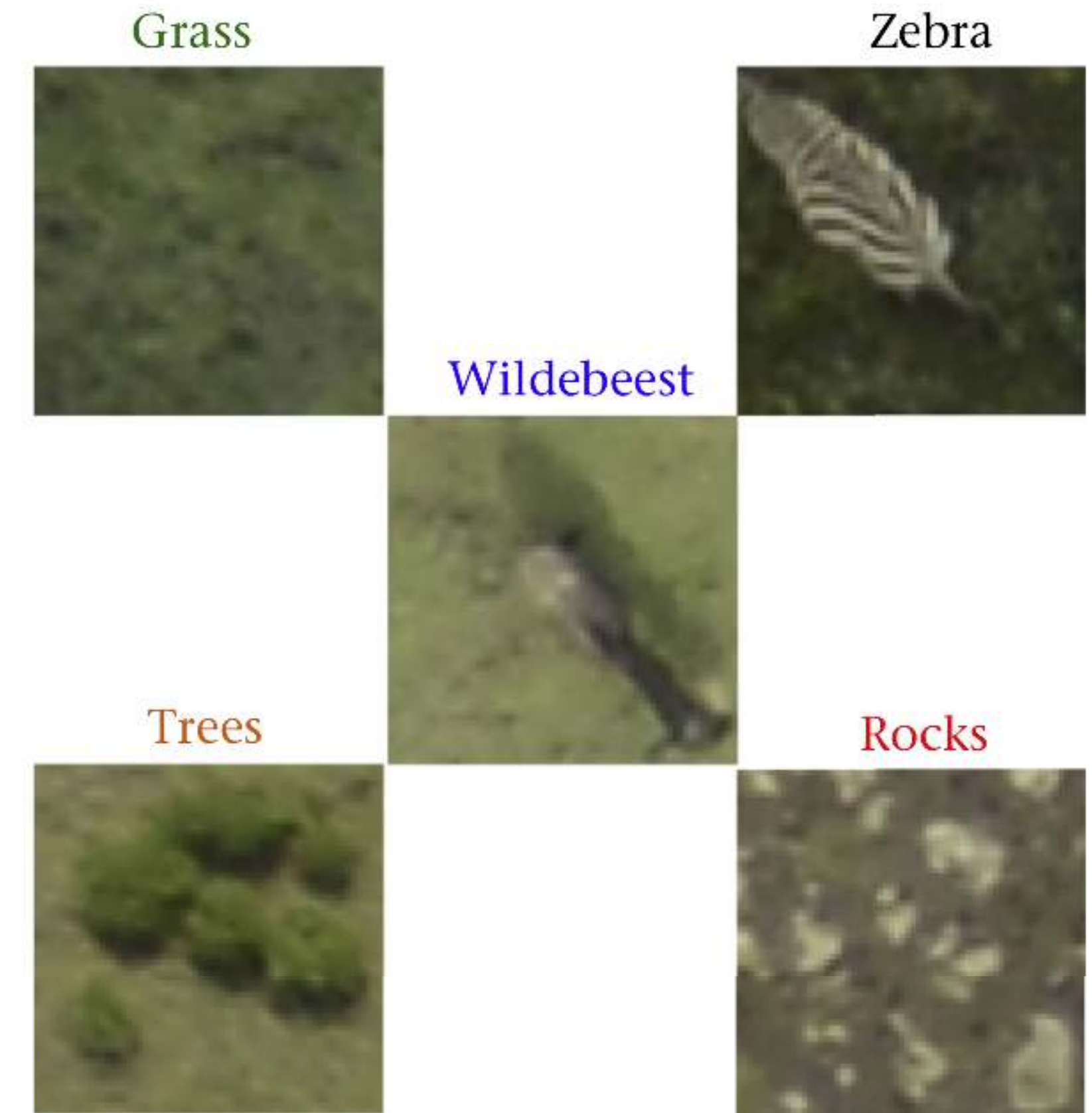
Mirco Schönfeld
University of Bayreuth

mirco.schoenfeld@uni-bayreuth.de
[@TWilyY29](https://twitter.com/TWilyY29)





Supervised
Machine
Learning





Supervised Machine Learning

Objective:

learn the *class label* y for each value of x , the *feature vector* consisting of multiple *features* (categorical or numerical)

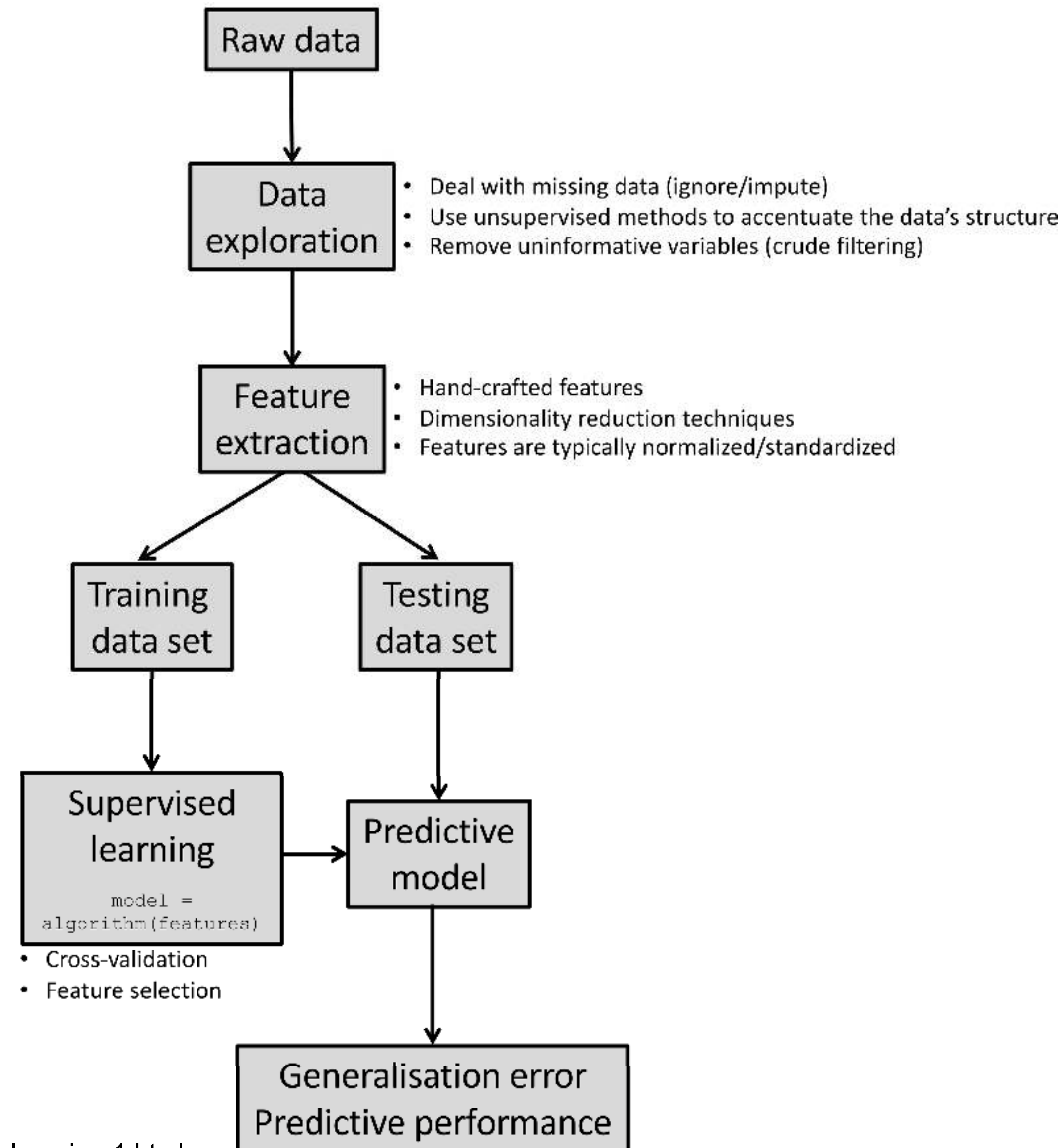
Result:

A function $f(x) = y$ that best predicts y for each value of x

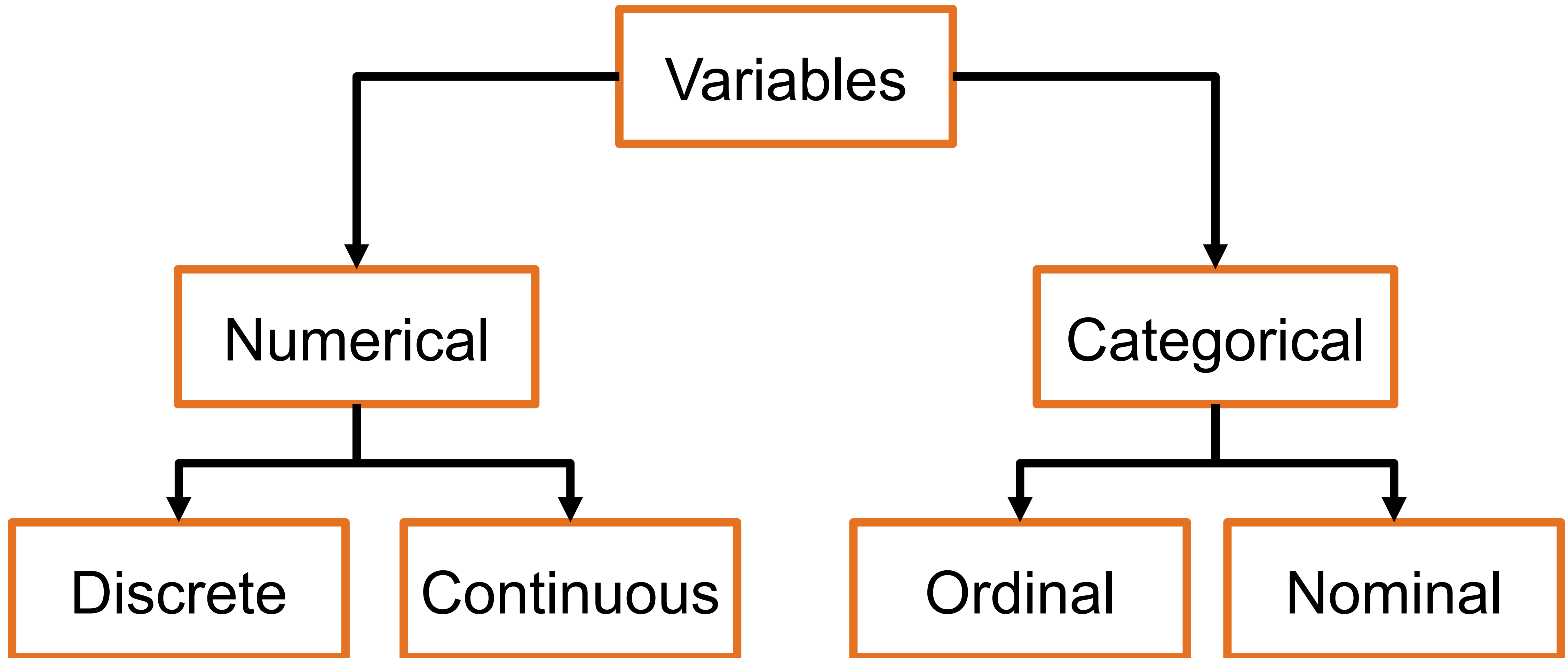
If y is

- a real number, a regression model is learned
- a Boolean value (true/false, +1/-1), we speak of *binary classification*
- a nominal value of some finite set, it is a *multiclass classification*

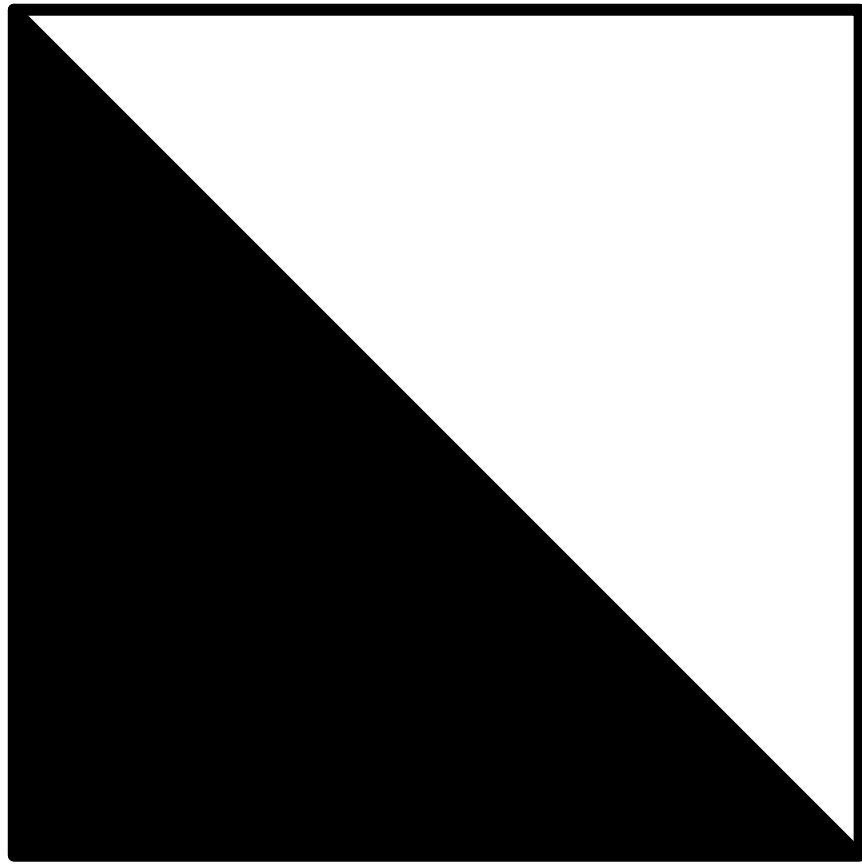
Supervised Machine Learning: Bird View



Hello Data



Discrete vs. Continuous



Dichotomous data:

Data points can only take up 2 values



Discrete data:

Data points can only take up values from a set of possible values.

Either finite or infinite but countable



Continuous data:

Data points can be measured to an arbitrary level of exactness

Scales

Nominal

Values divide space of possible values into different segments

No order between segments

Example: gender, nationality

Ordinal

Allows for rank order – data can be sorted

No relative degree or relative difference between groups

Example: school grades

Interval

Order and difference between groups is defined

No natural „zero“ & ratio between points is undefined

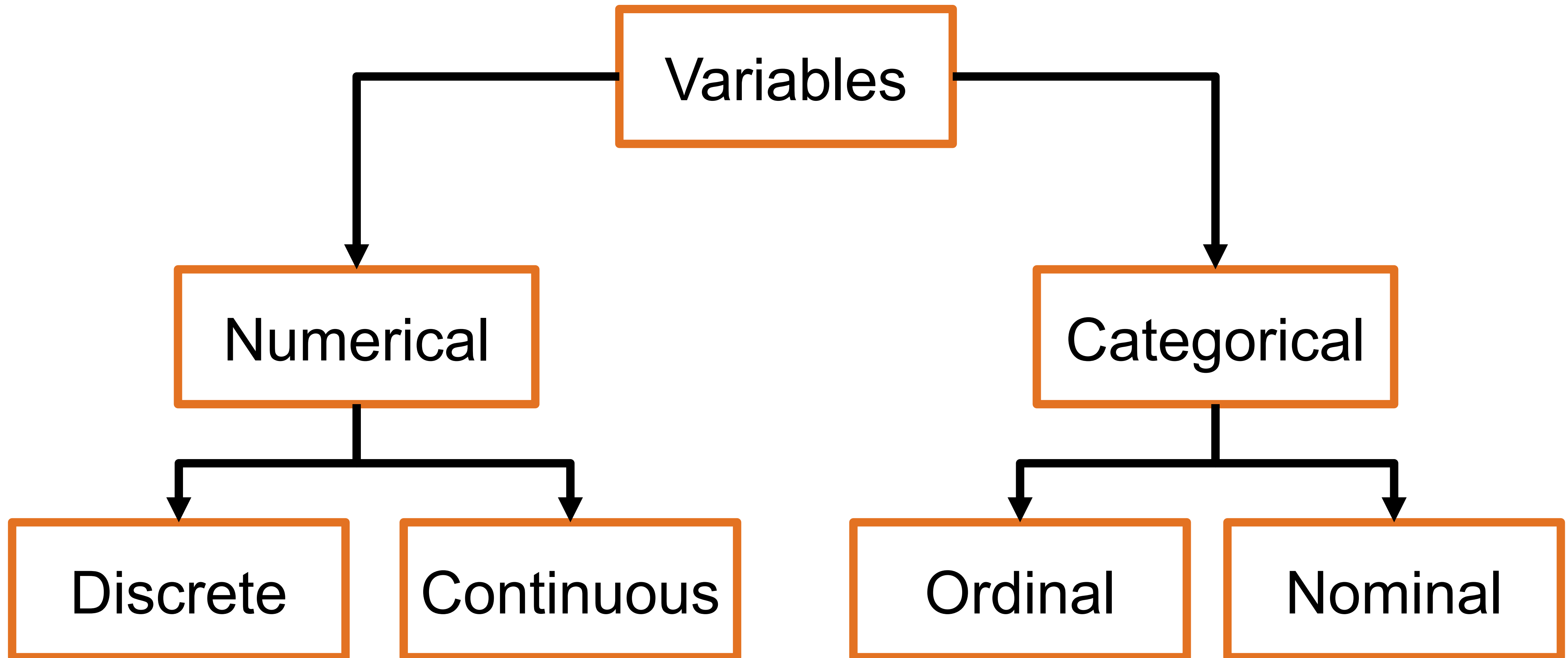
Example: temperature in Celsius

Ratio

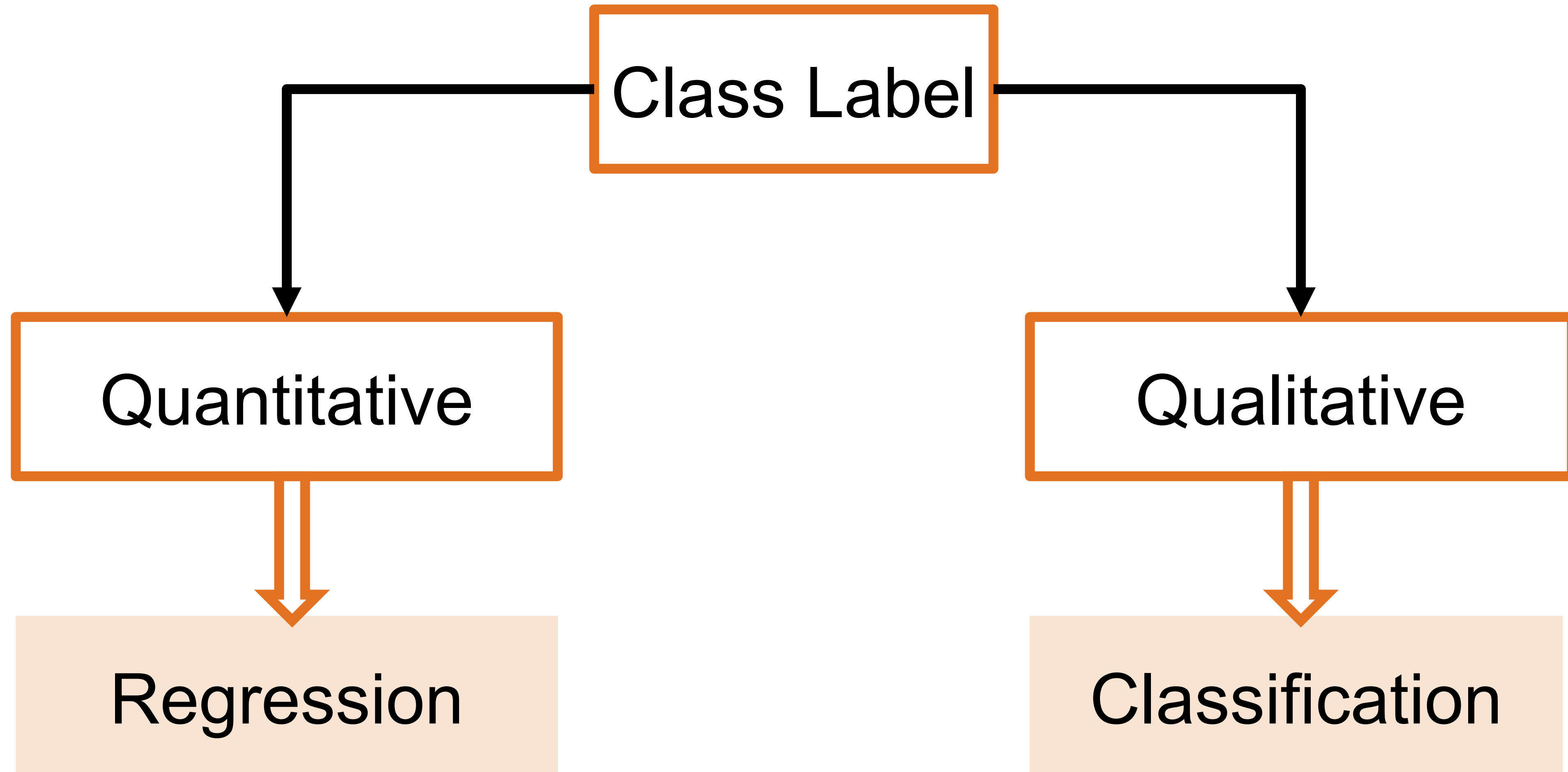
Meaningful zero, i.e. unique and non-arbitrary

Example: temperature in Kelvin, age, weight, height

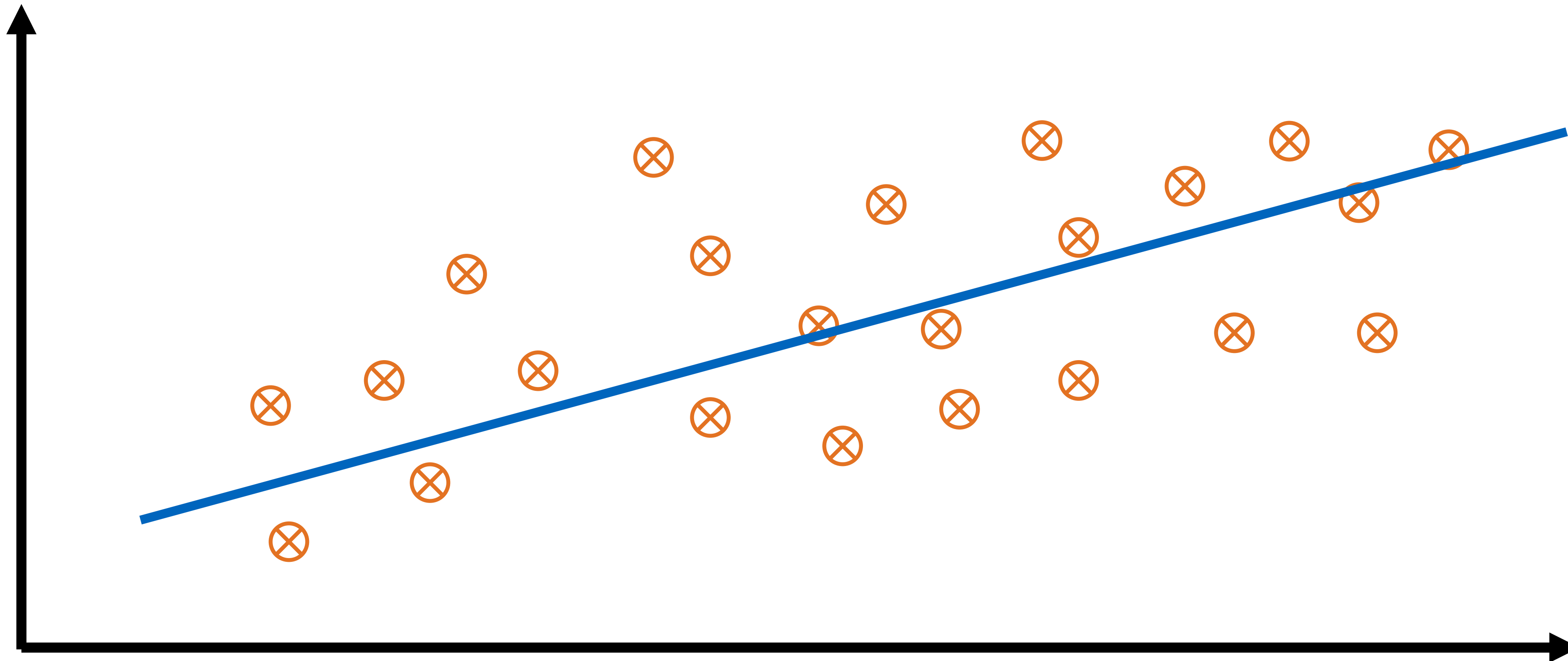
Hello Data



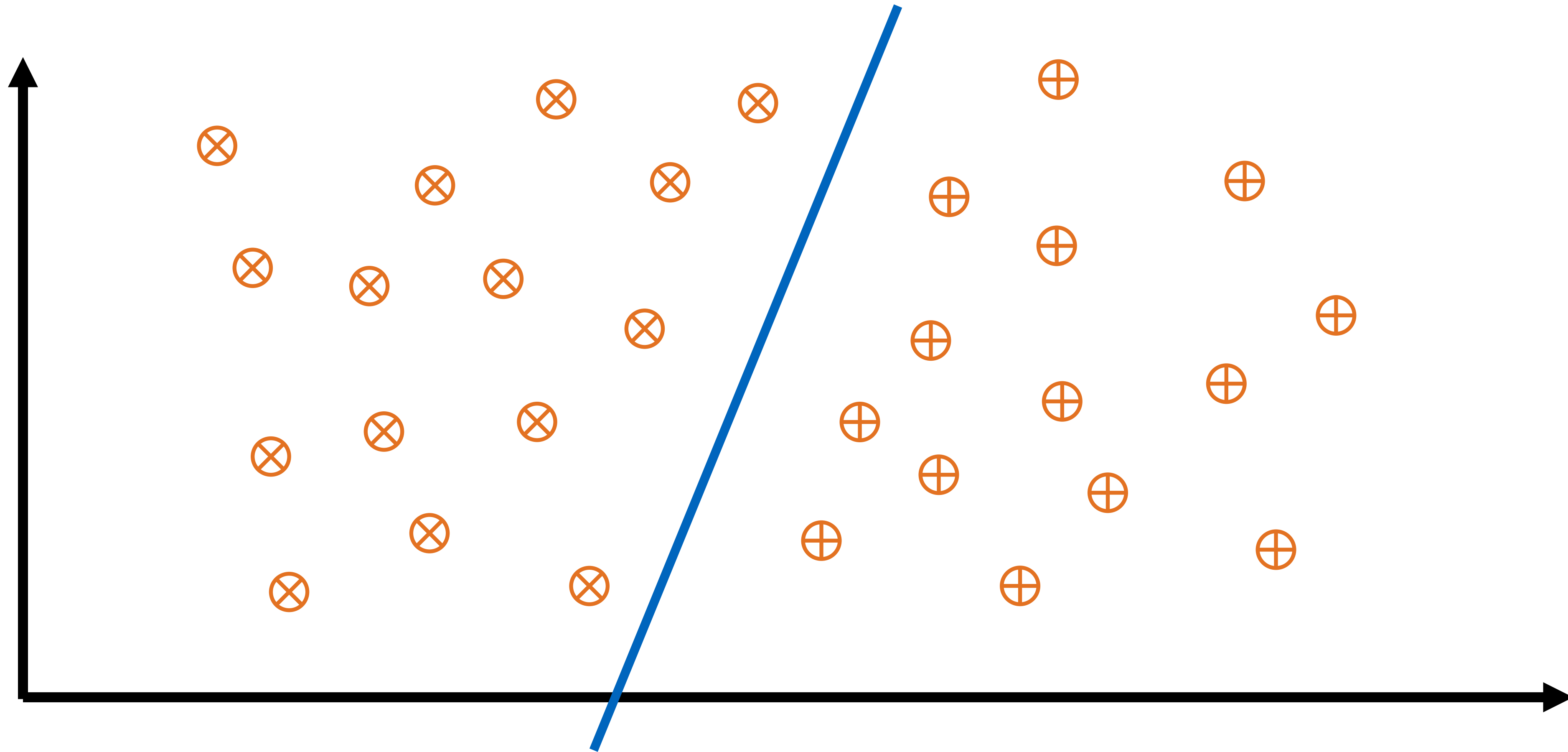
Nature of the Class Label



Regression



Classification





Supervised Machine Learning...in other words

Supervised Machine Learning aims at forecasting class labels for measured data

Correct class labels are known for training data

Training means to search for a good function mapping
measured artifacts to known class labels

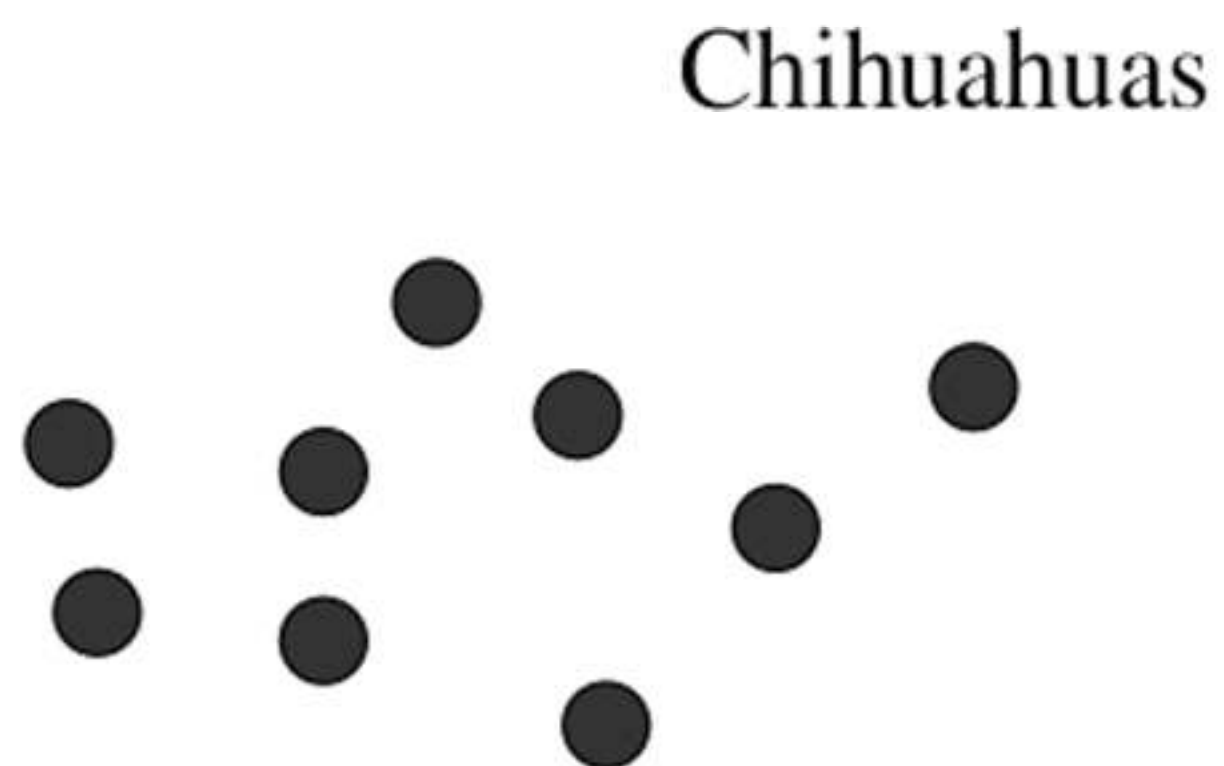
What you need:

- Classifier
- Measured data
- Class labels

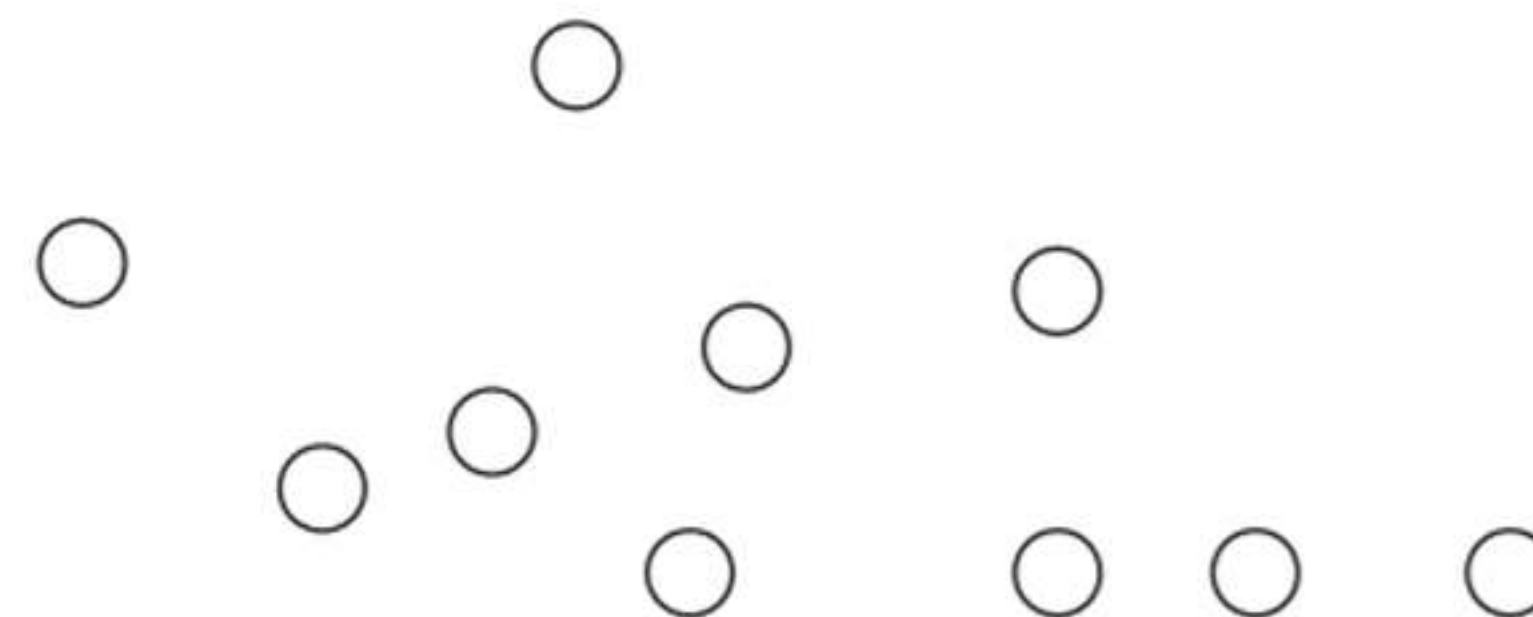
An Example



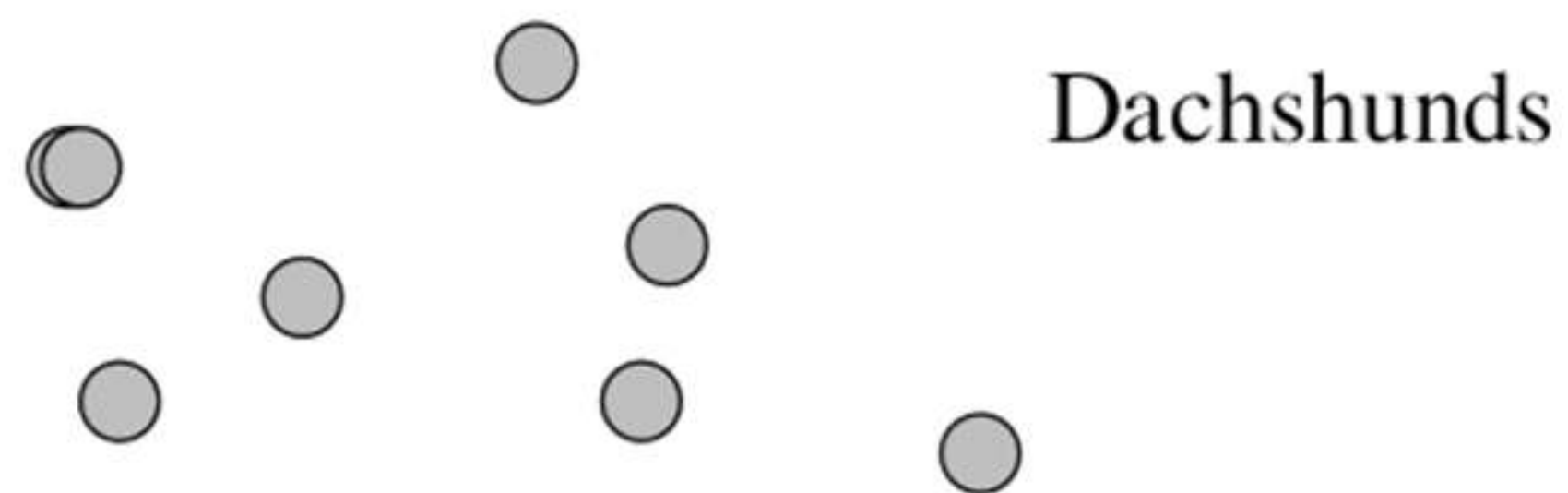
↑
Height



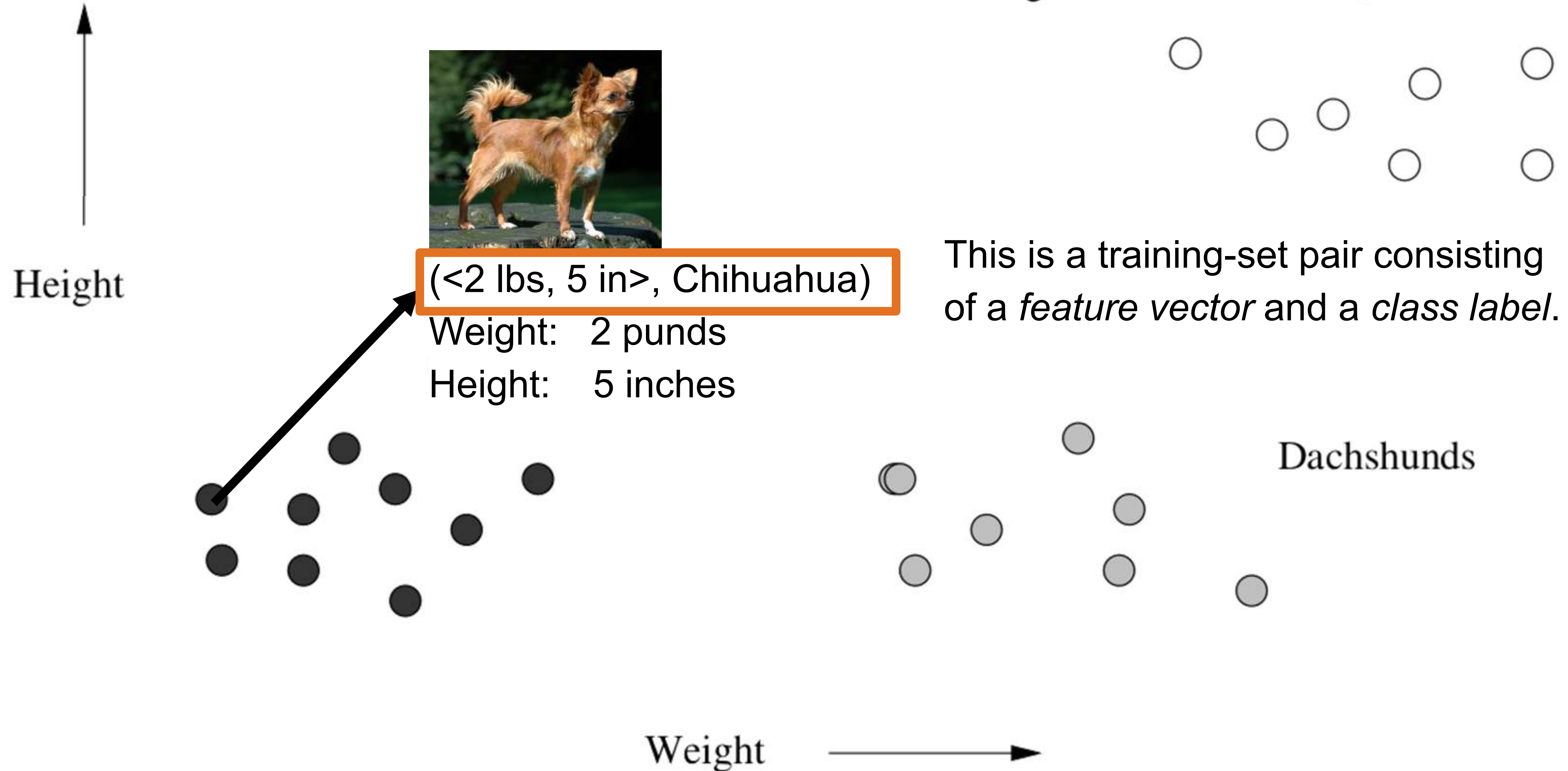
Beagles



Weight →



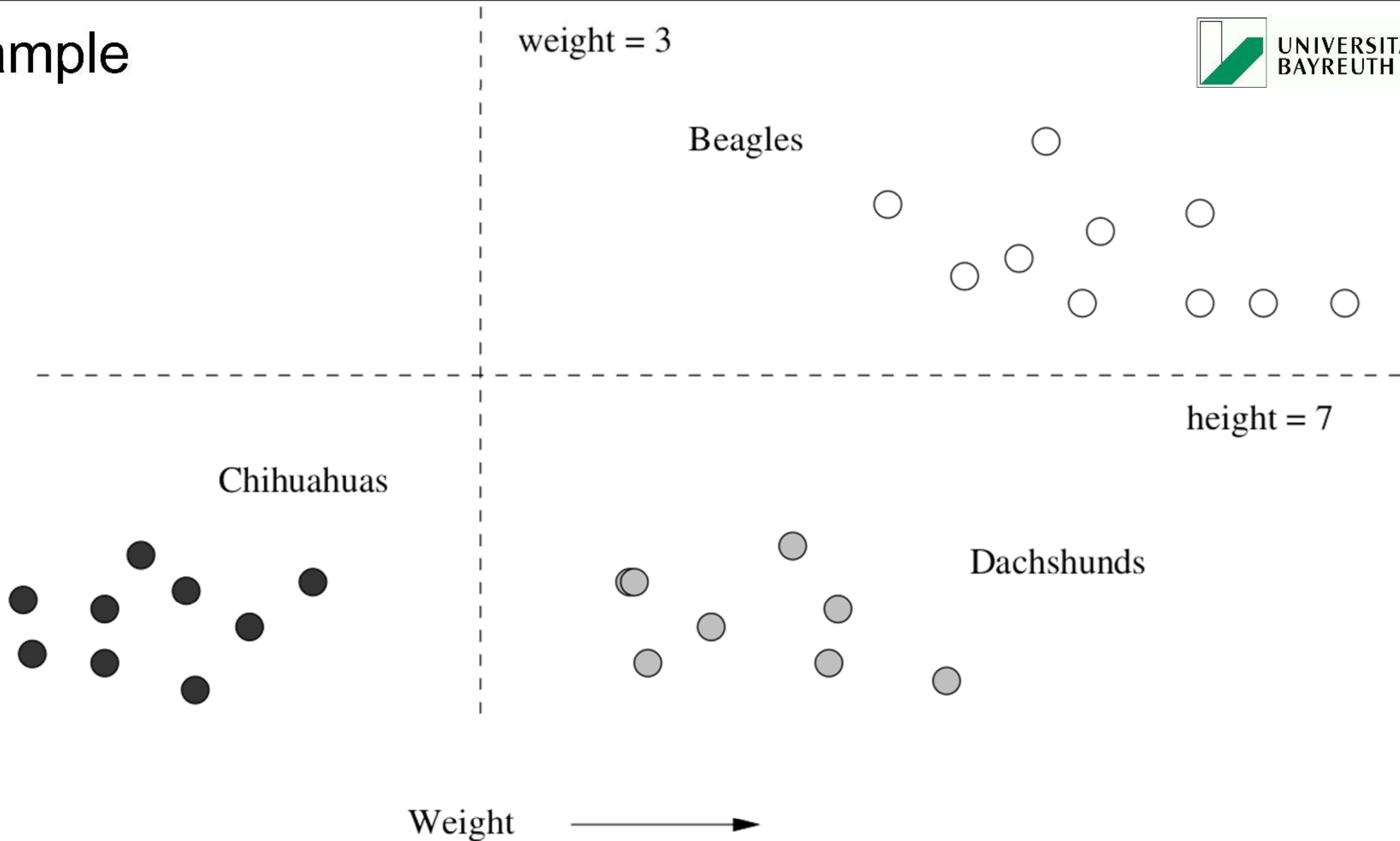
An Example



An Example



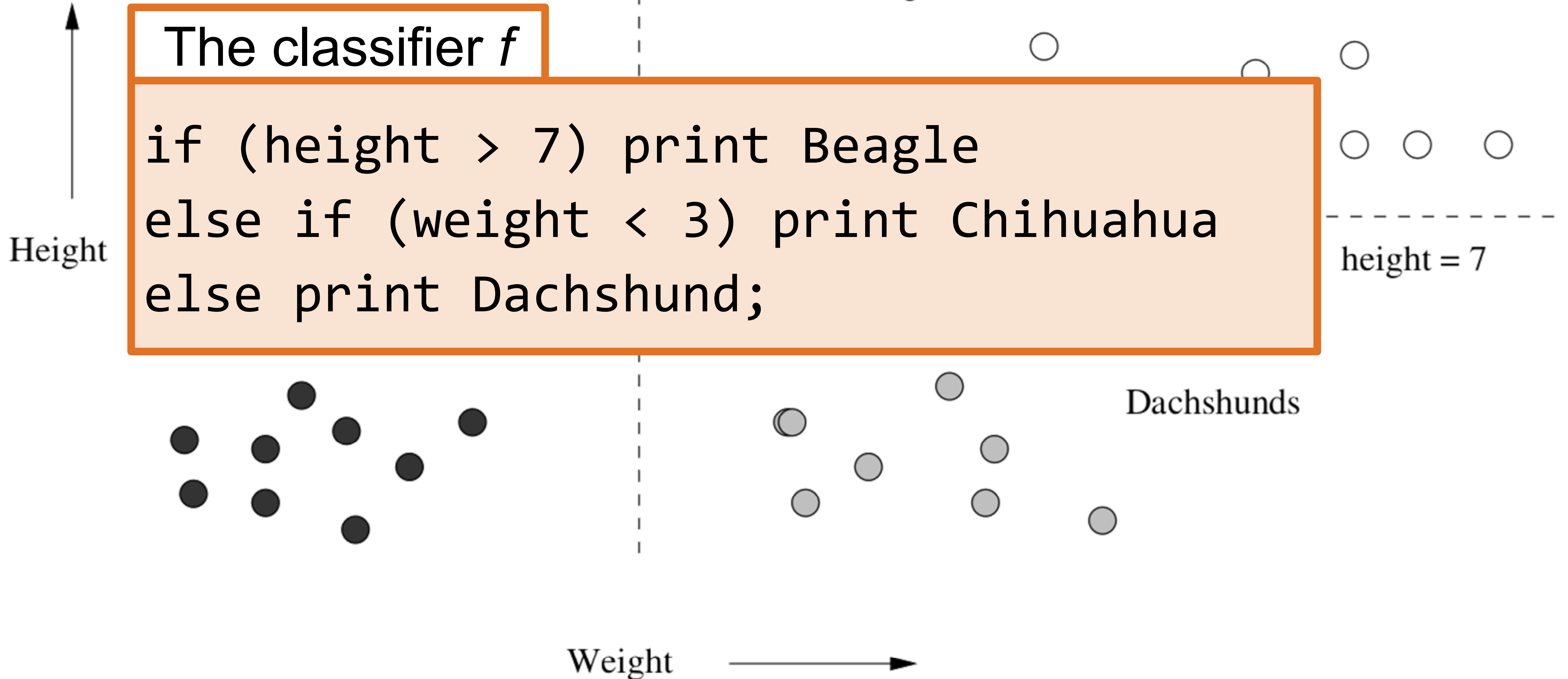
↑
Height



An Example

The classifier f

```
if (height > 7) print Beagle  
else if (weight < 3) print Chihuahua  
else print Dachshund;
```



Relation to Clustering

Classification:

- Class label is discrete
- Enough training data for *each class*

Regression:

- Target variable is numeric

Clustering:

- no class label / target variable
- no training – input data without pre-defined classes
- produces partitioning of data

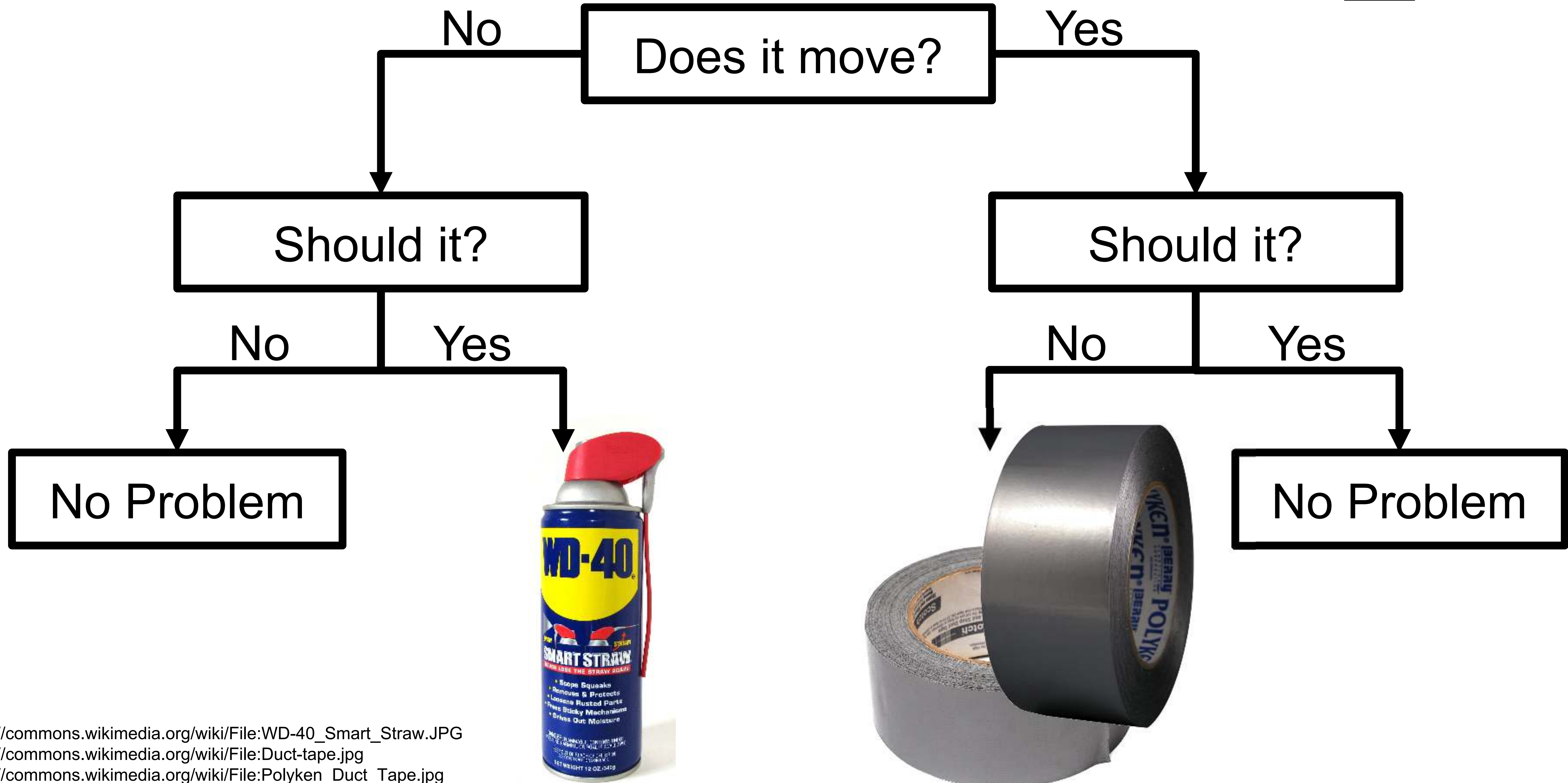
Clusters could be used to
identify new classes



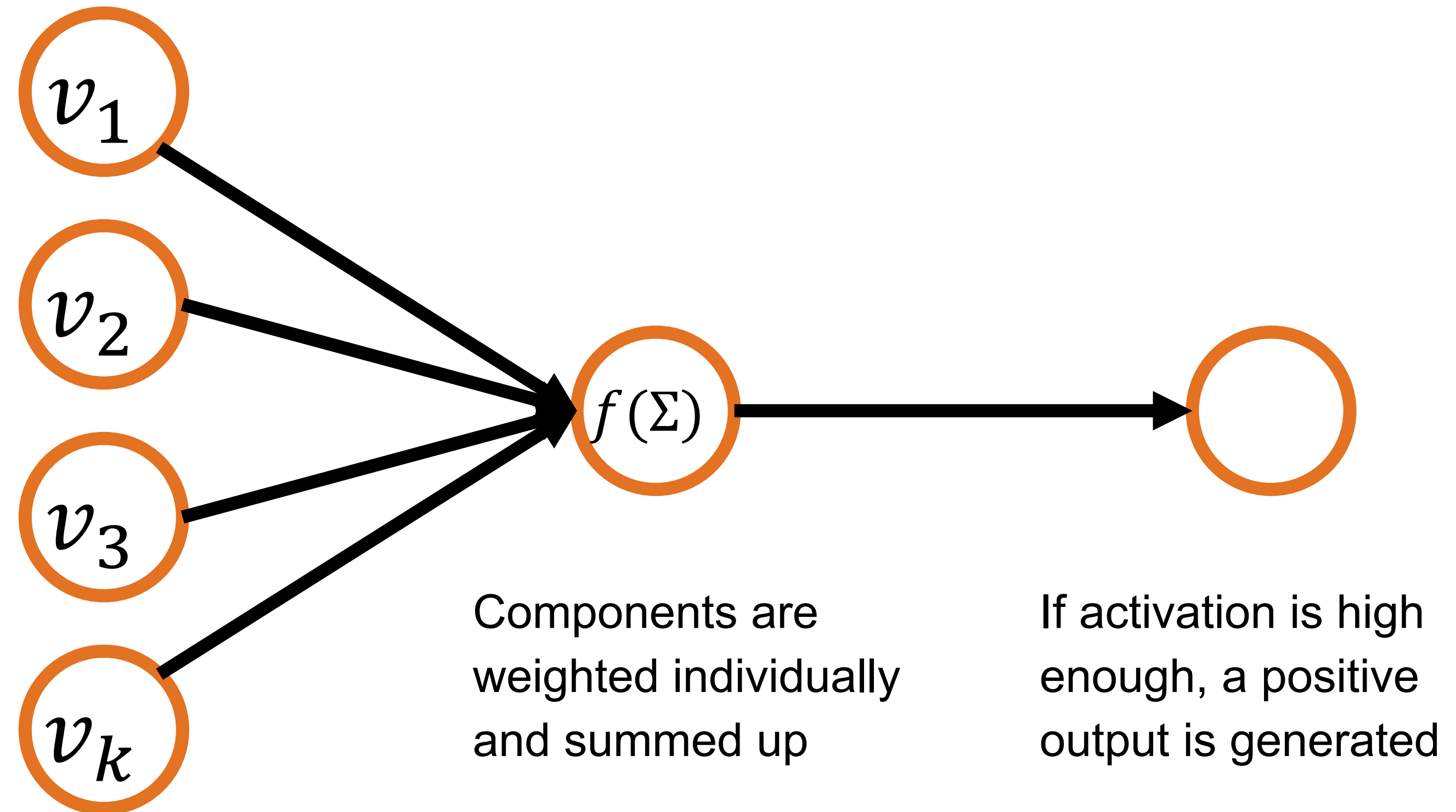
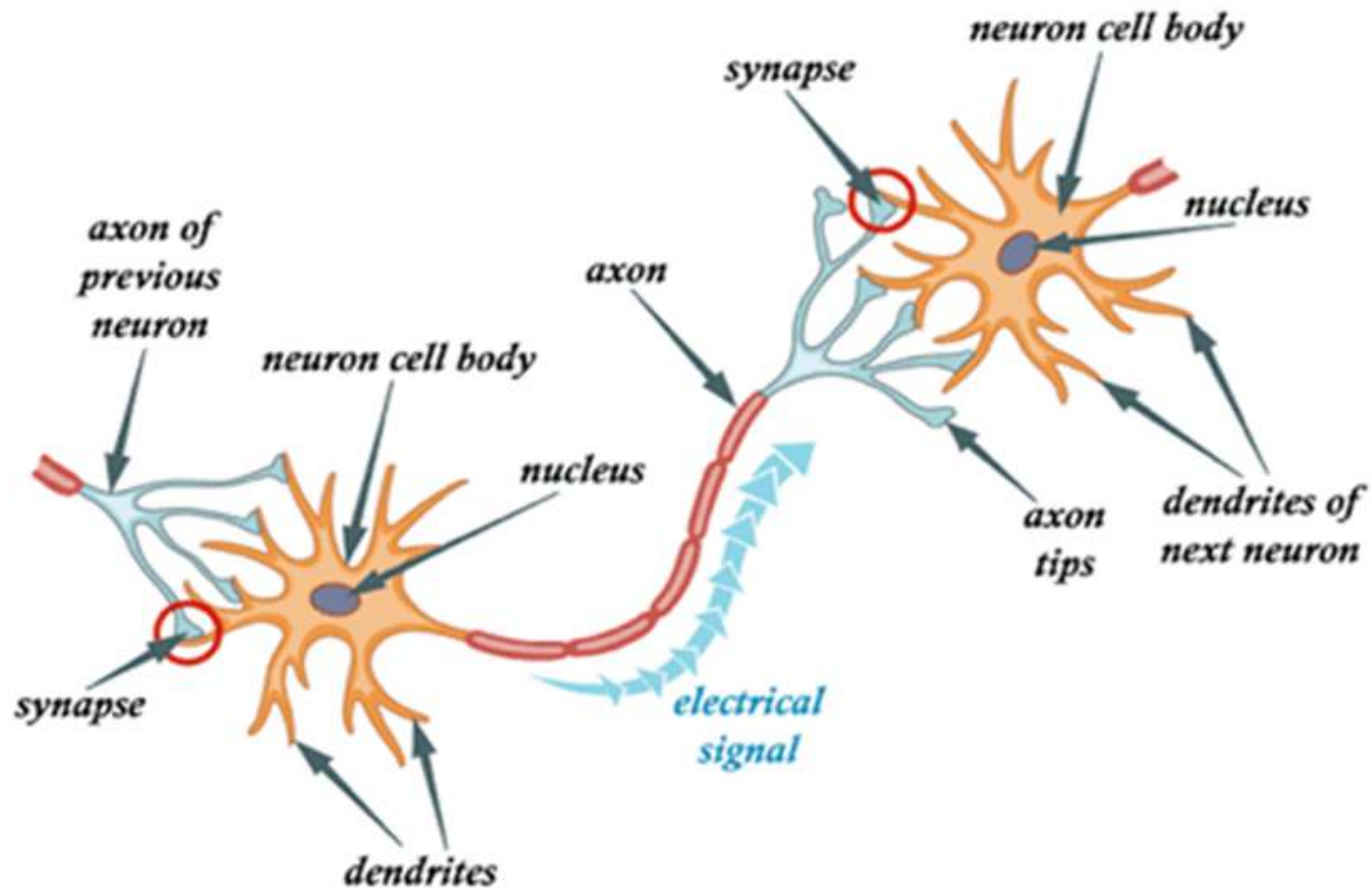
Major Classes of Classifiers

- **Decision trees**
Suitable for binary and multiclass classification with a limited number of features
- **Perceptrons**
Applies weights to components of vectors. Output +1 if sum exceeds a threshold, otherwise -1.
- **Neural networks**
Acyclic networks of perceptrons
- **Instance-based learning**
Compares instances of data to the entire training set. Example: k-nearest neighbor.
- **Support-vector machines**
Maps training examples to points in space so as to maximise the gap between categories

Decision Trees

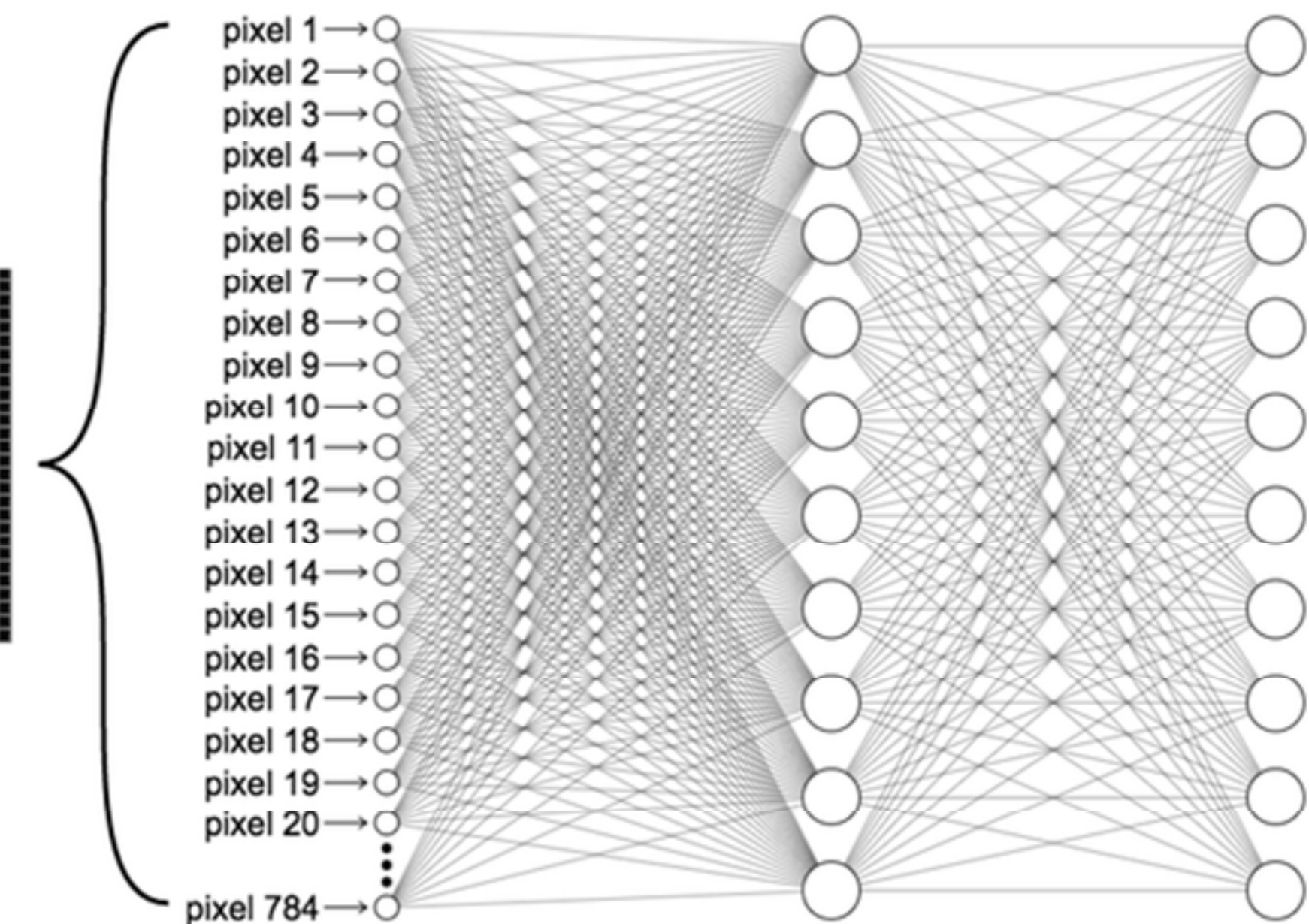
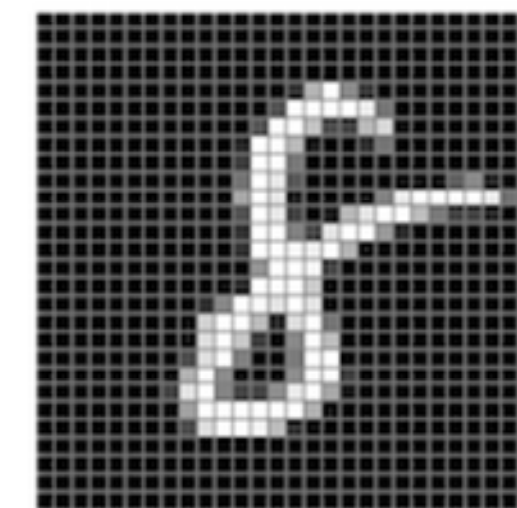
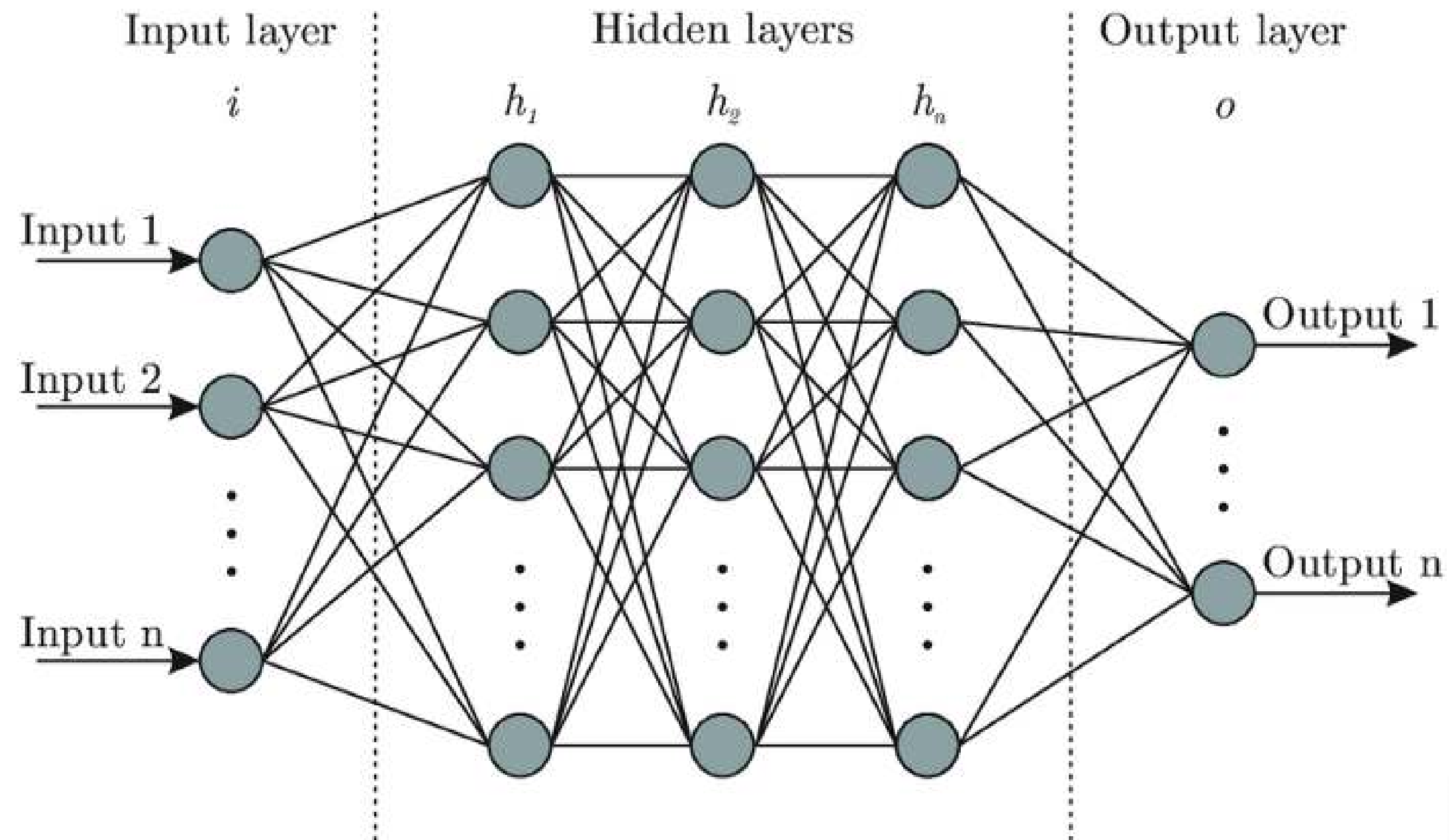


Perceptrons

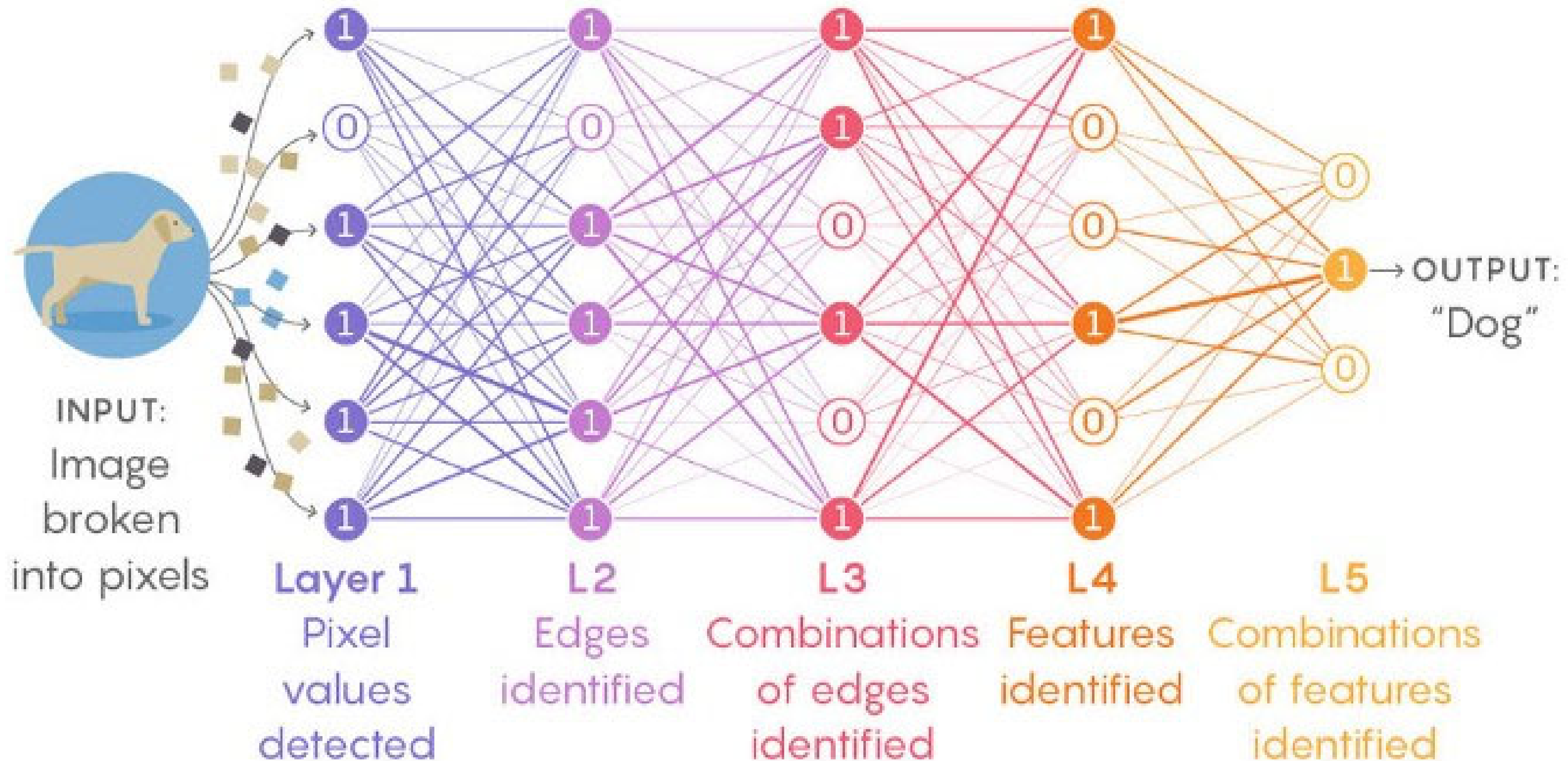


Components of instances

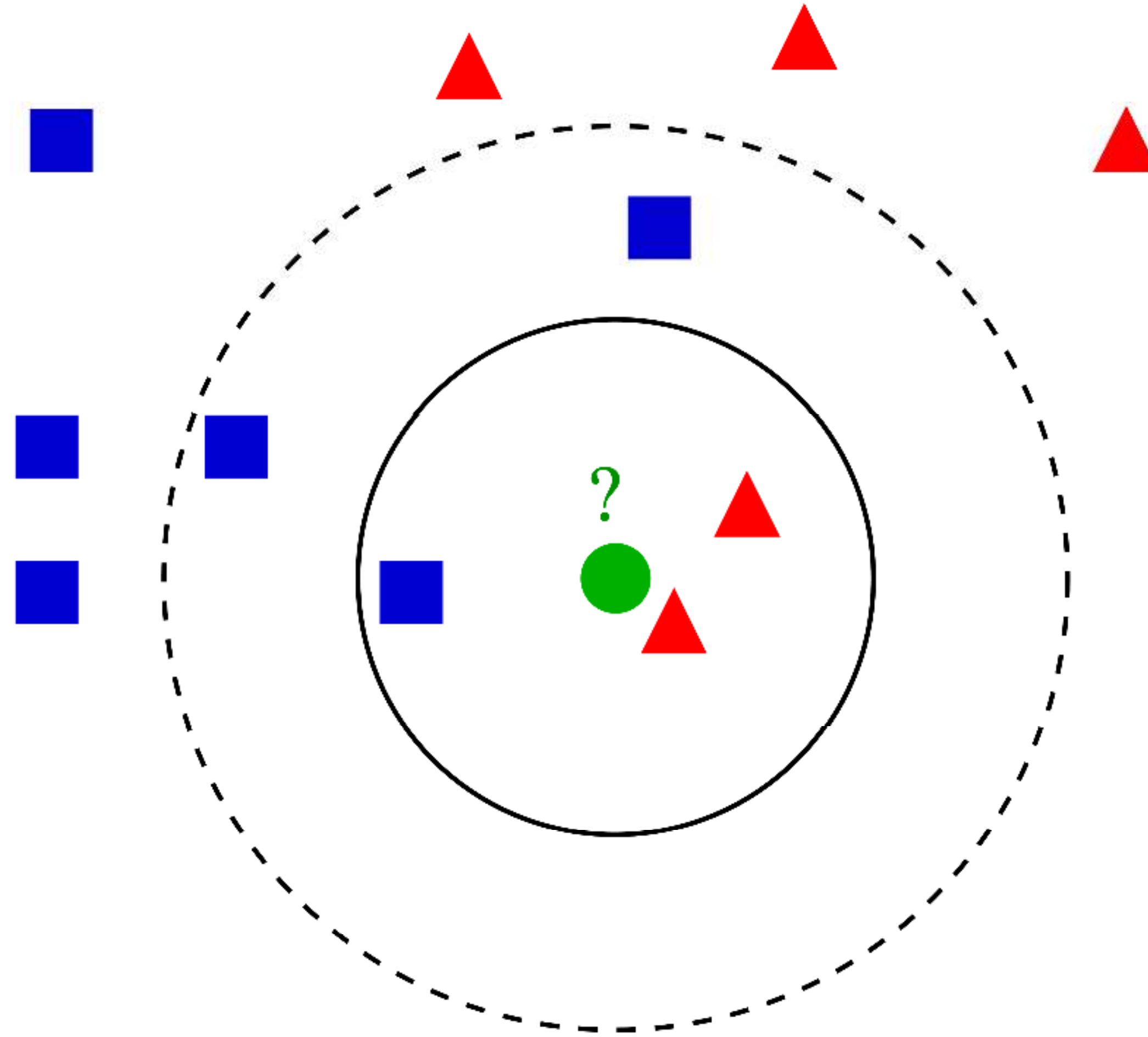
Neural Networks



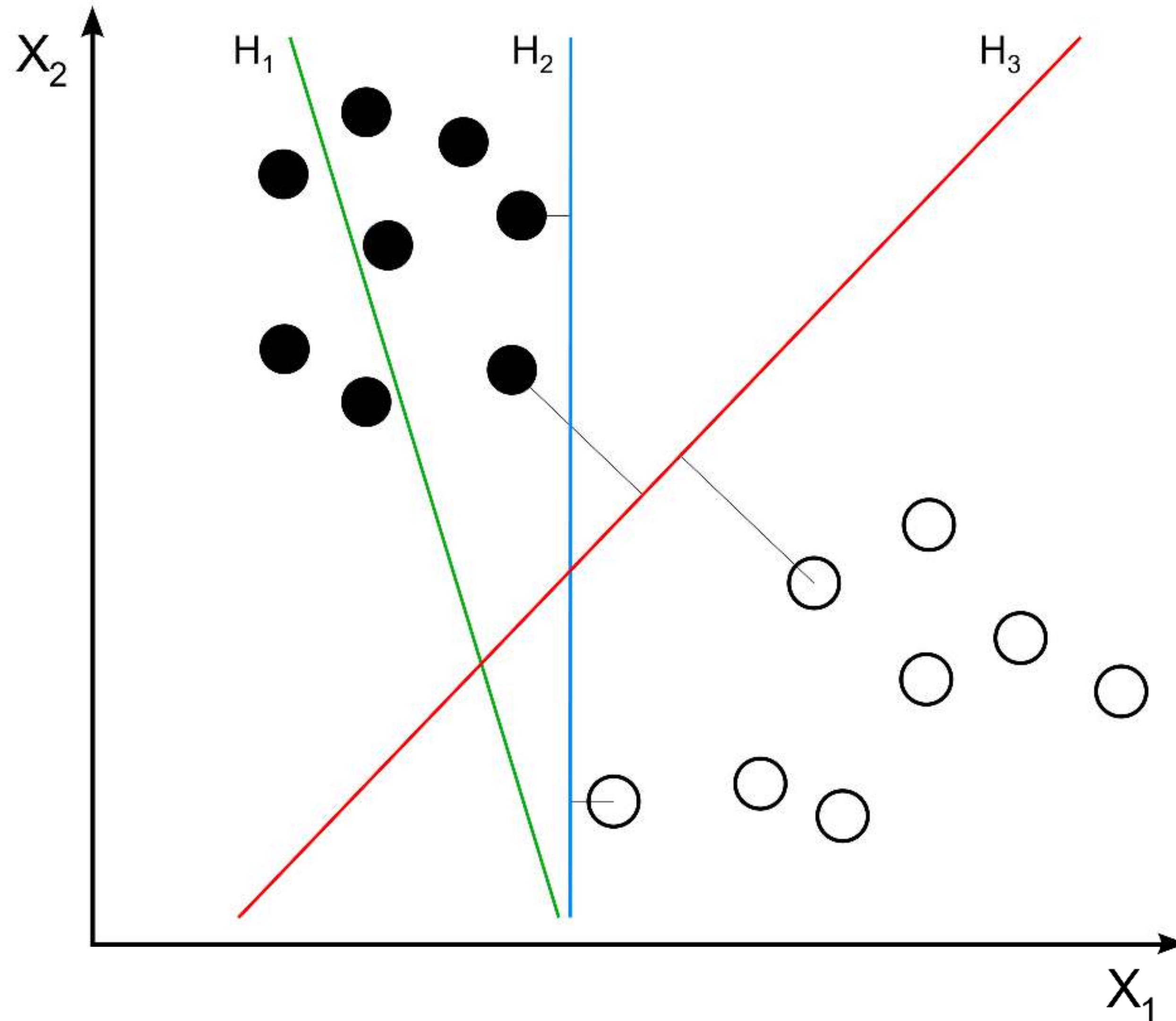
Neural Networks



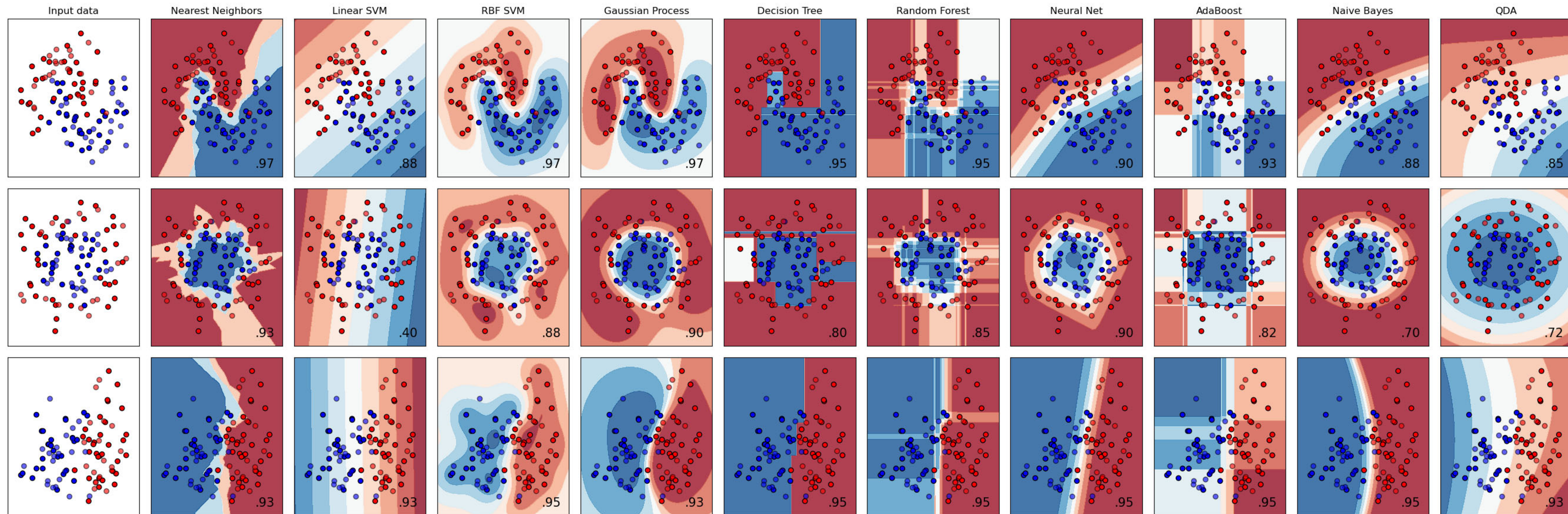
k -Nearest Neighbors



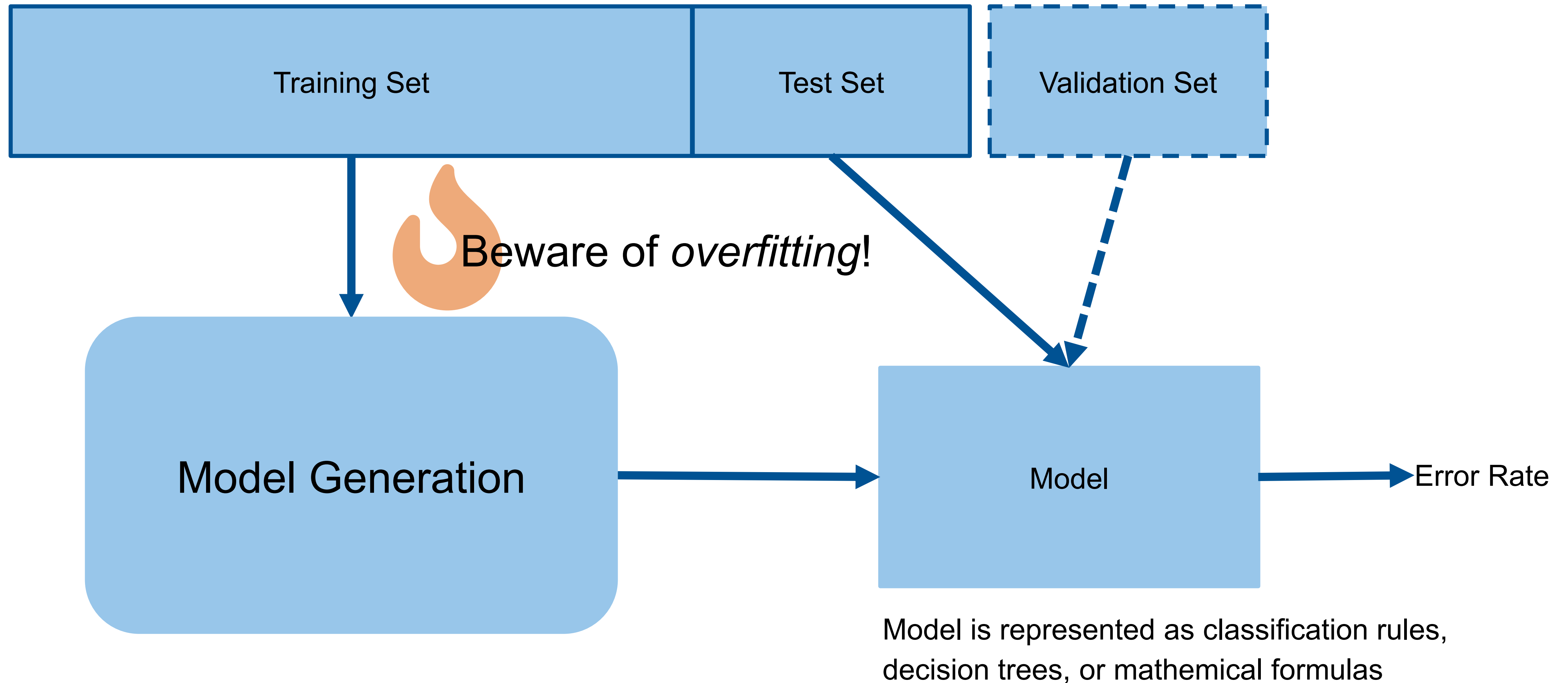
Support Vector Machines



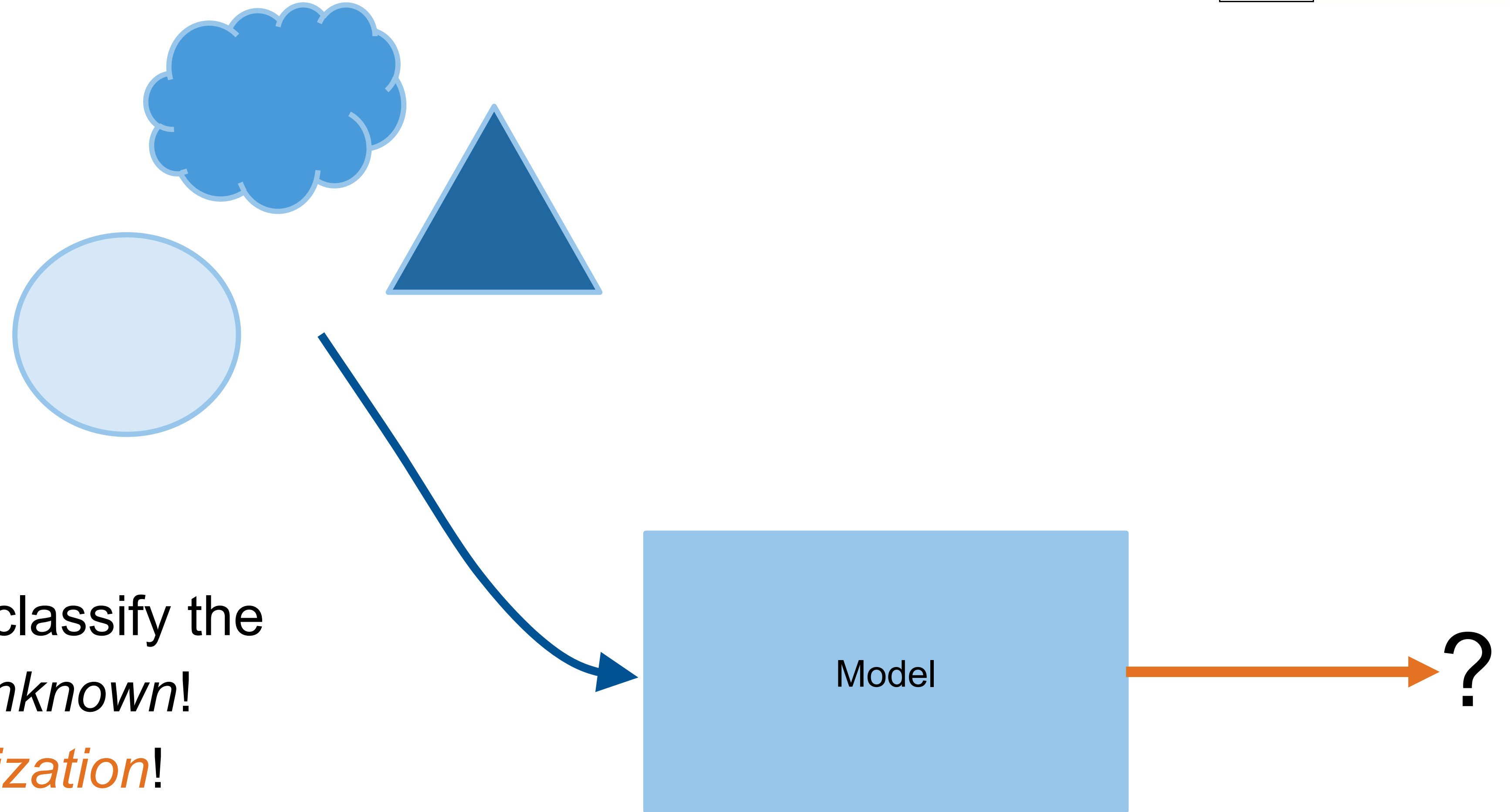
Some Classification Algorithms Compared



Supervised Learning: Training and Testing



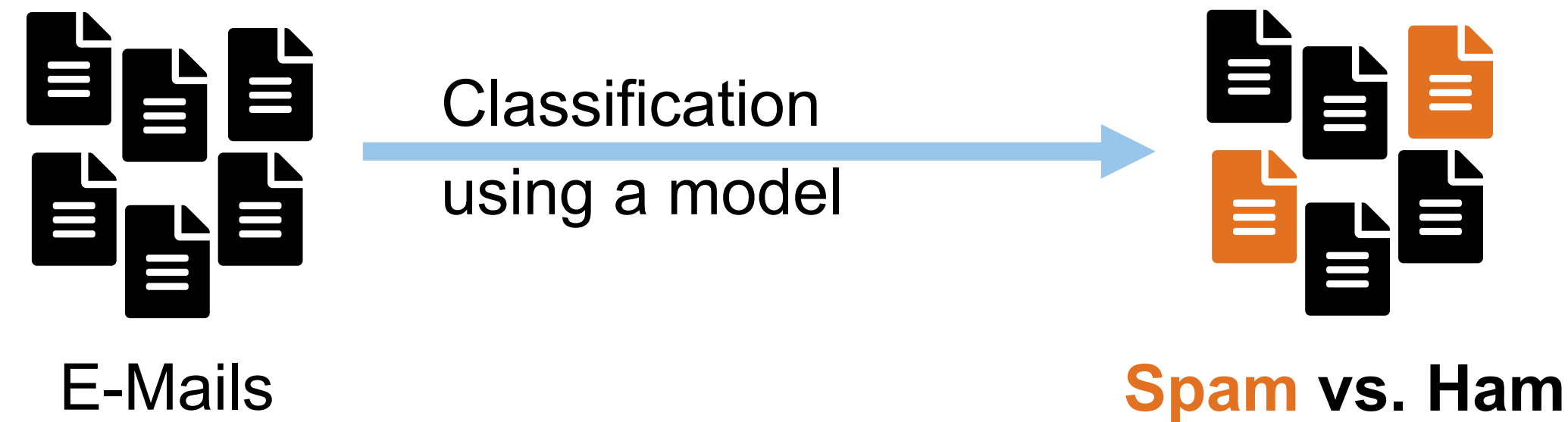
What the Model is for



The model is used to classify the data whose class is *unknown*!
Its purpose is *generalization*!

How to Design a Model: Feature Selection

Formulate characteristics that help distinguishing between classes.
For spam-detection: find words or combinations of words that indicate a mail being spam.



Spam: Wholesale Fashion Watches -57% today. Designer watches for cheap ...
 Spam: You can buy Viagra Fr\$1.85 All Medications at unbeatable prices! ...
 Spam: WE CAN TREAT ANYTHING YOU SUFFER FROM JUST TRUST US ...
 Spam: Sta.rt earn*ing the salary yo,u d-eserve by o'btaining the prope,r crede'ntials!

Ham: The practical significance of hypertree width in identifying more ...
 Ham: Abstract: We will motivate the problem of social identity clustering: ...
 Ham: Good to see you my friend. Hey Peter, It was good to hear from you. ...
 Ham: PDS implies convexity of the resulting optimization problem (Kernel Ridge ...

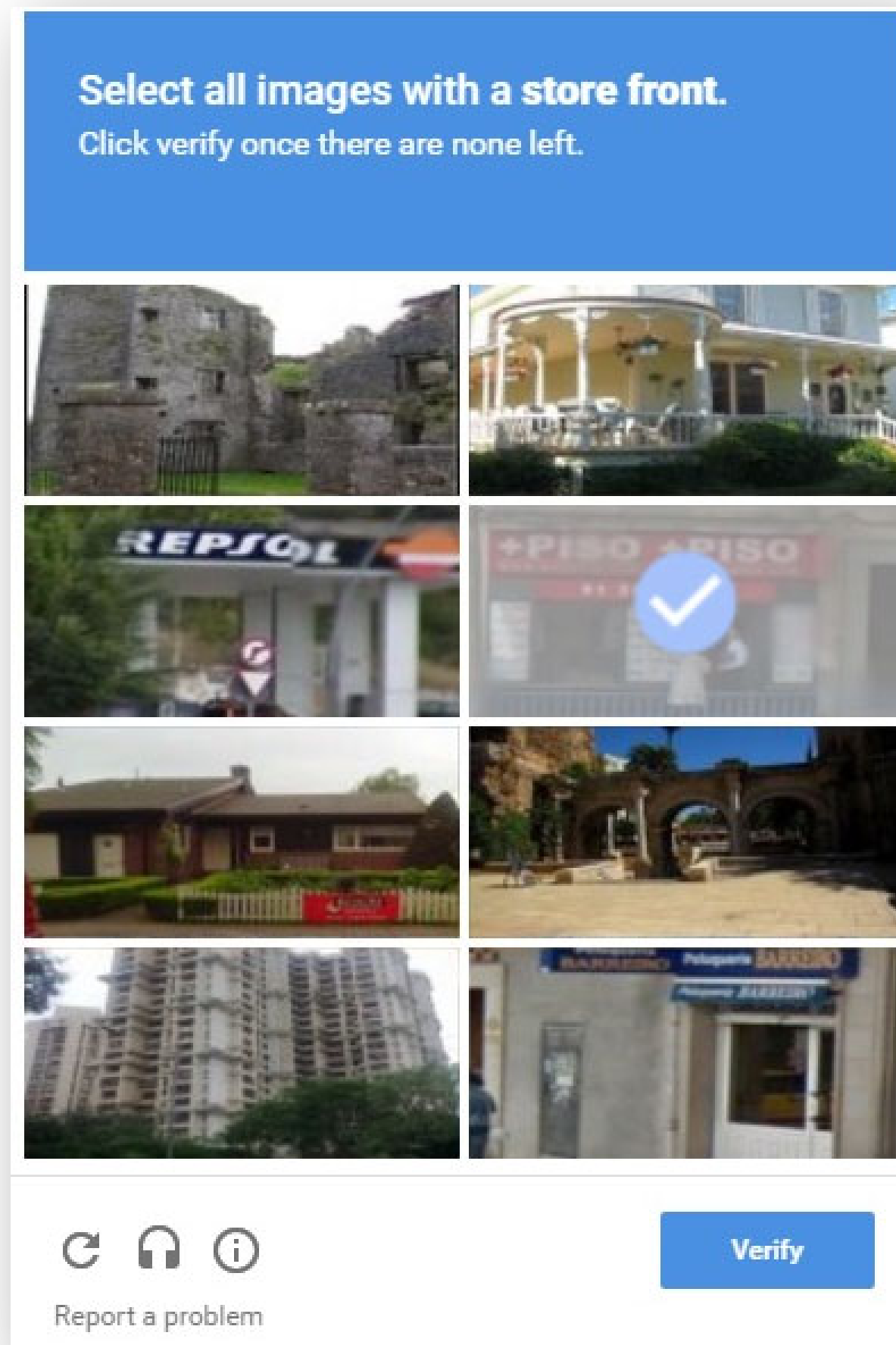
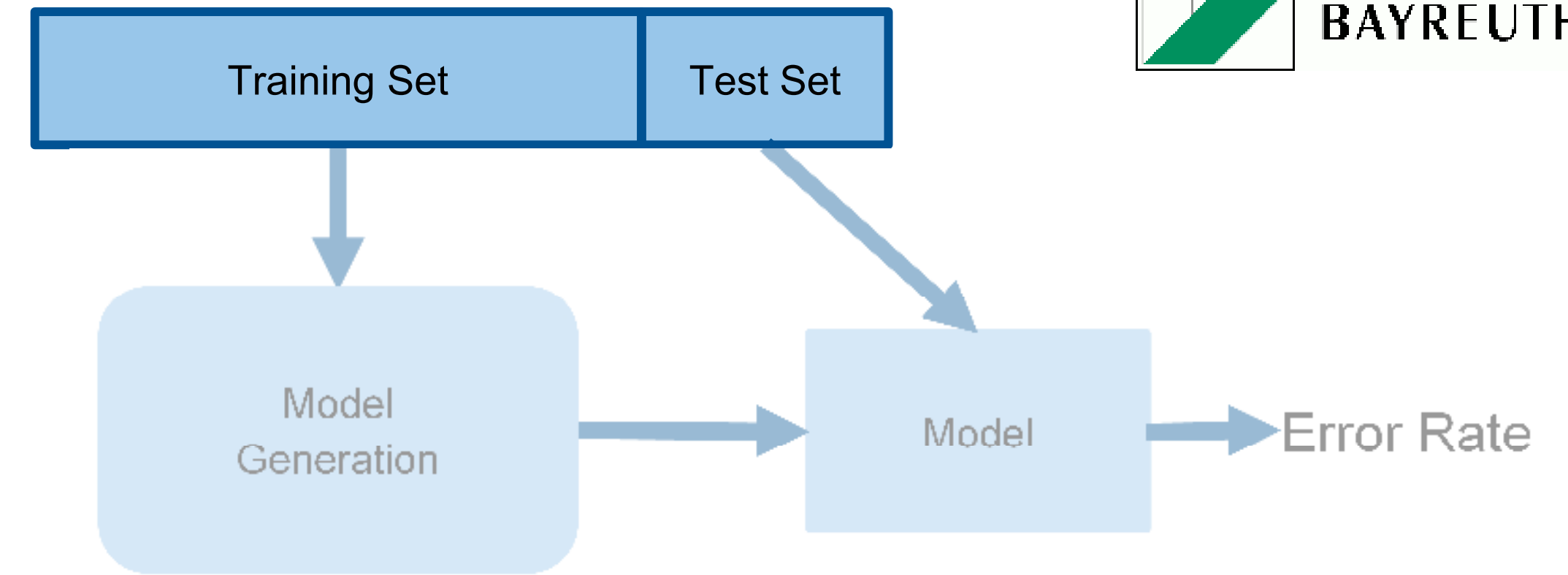
Curse of Dimensionality:

Including more features will improve classification *conceptually* but will render computation increasingly difficult.

/kæp.tʃə/ or Creating a Training Set

Label Data Manually?


Often time consuming and costly process for large training sets



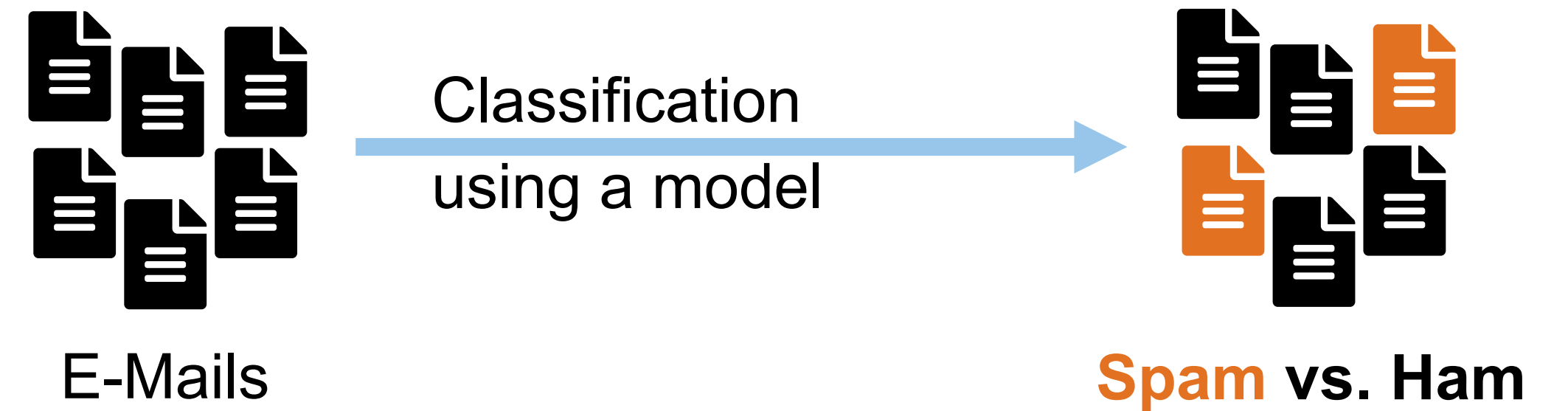
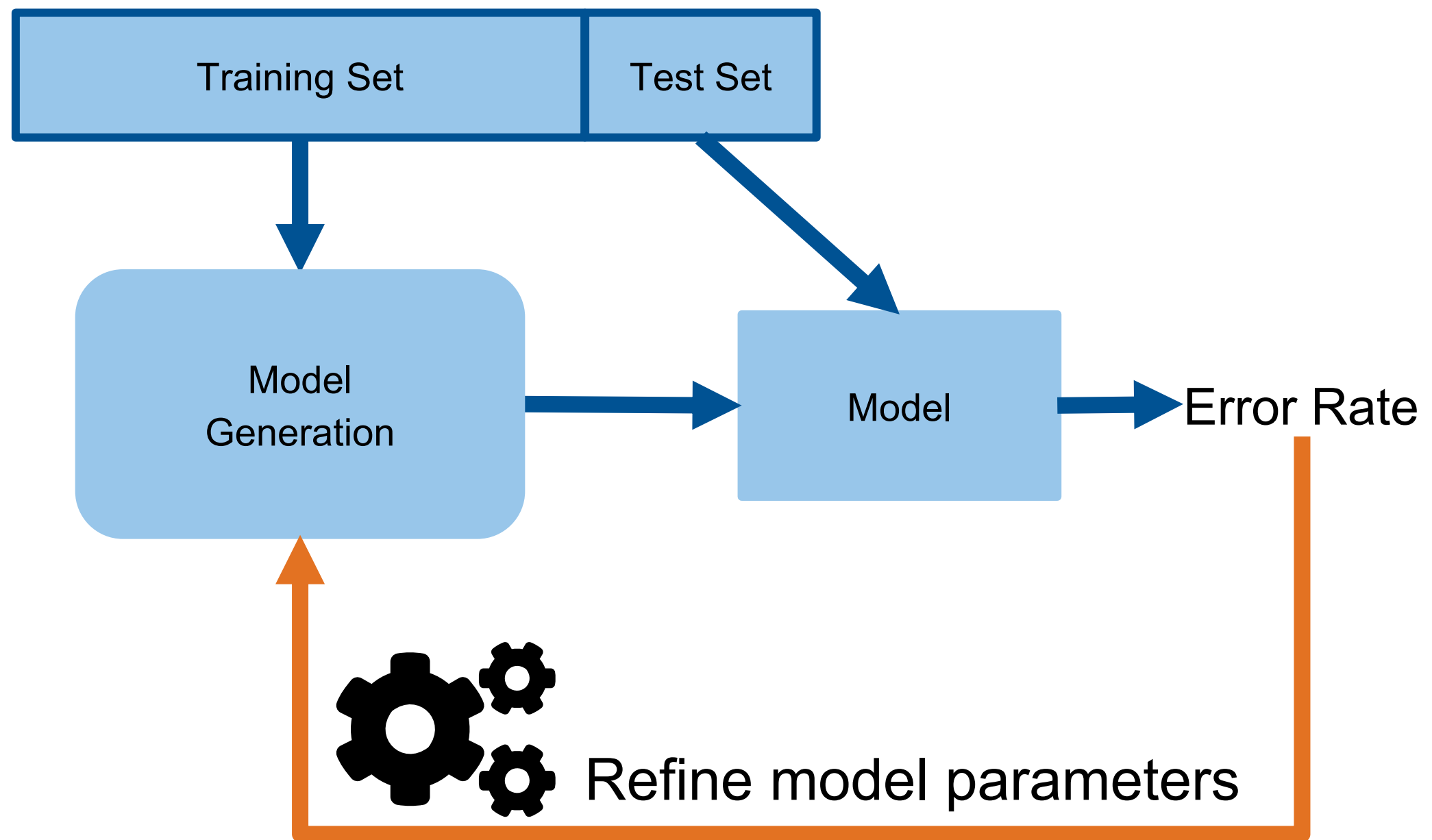
Active learning:

kick start a classifier only with some training examples, but leave it primarily with unclassified data, which it must classify. If the classifier is unsure of the classification (e.g., the newly arrived example is very close to the boundary), then the classifier can ask for ground truth at some significant cost

How to Choose a Model

 Minimize error rate

 Maximize *utility*



Model A:

		Correct class	
		Ham	Spam
Spam-Filter reports	Ham	200	38
	Spam	0	762

Model B:

		Correct class	
		Ham	Spam
Spam-Filter reports	Ham	189	1
	Spam	11	799



Beware of Overfitting

Two types of classification errors:

1. Training error – misclassification on training data
2. Generalization error – expected error on *unseen* data

Overfitting:

Good results on training data (low training error) and
bad results with test/validation data (high generalization error)

Error significantly *underestimated* – severe problem in application scenarios

Detecting overfitting:

Evaluation of training with *new* data – NOT using training data!

Training Test Split

Split training data *at least* in training and testing (Popular splits: 2:1 / 90:10)

Recommended: Split data in training, testing *and validation*

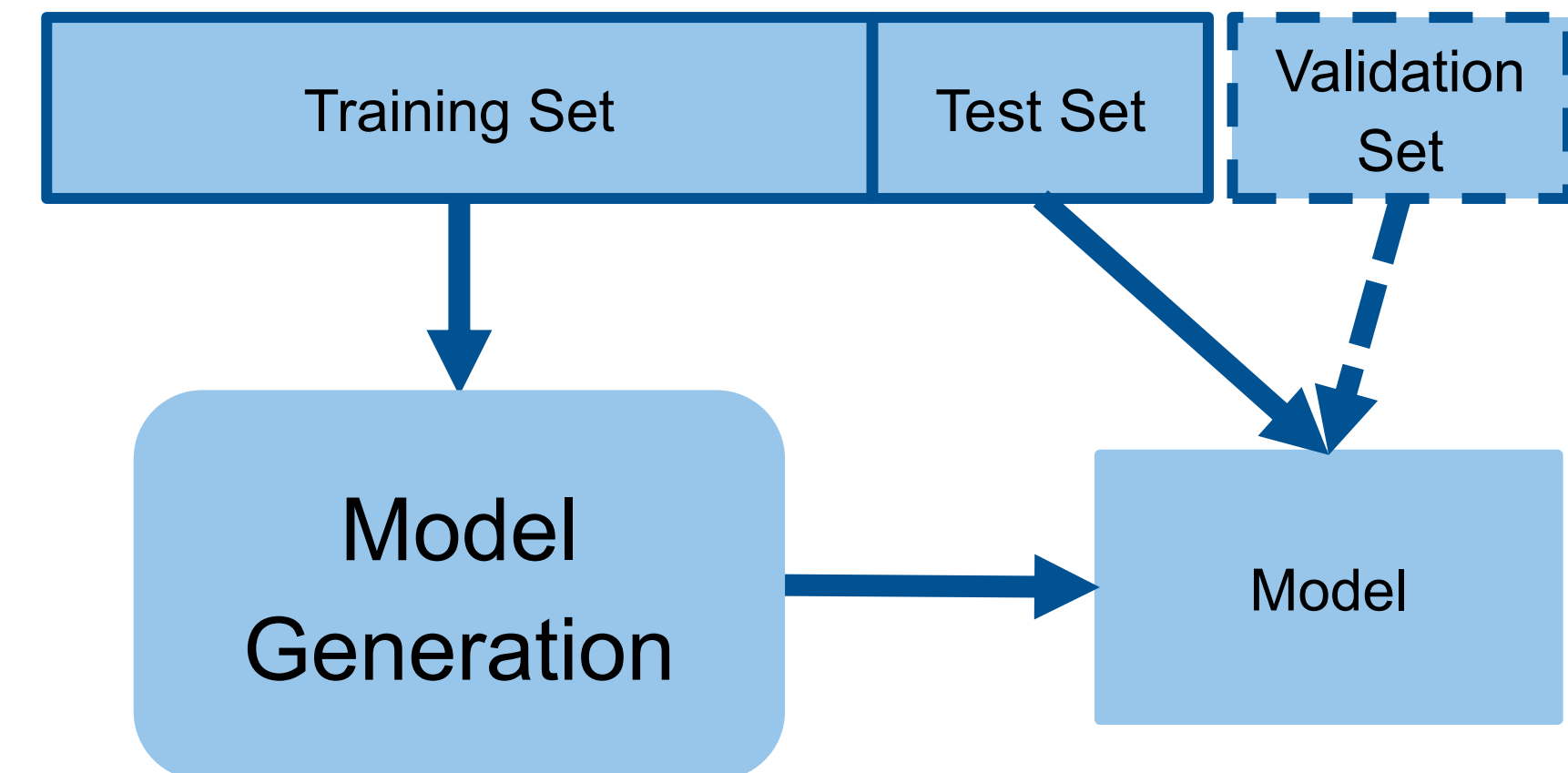
Splits: 80:10:10

Choose best classifier *only* on training and test data

Estimate accuracy & tune parameters of model

Keep validation-data *secret!* Use that only *once* to estimate the generalization power of the model!

Terminology of test and validation data is often mixed up.



What if...

Prepare data

Chose classifier

Train it

Test it

Validate it

Results do not look good

Repeat

What's the problem?

Repeated testing leads to overfitting

Once the validation data is used, do not go back to improve classification!

Cross-Validation

Cross-validation is a technique to help choosing classifier and optimal parameters

Partition data in k non-overlapping parts of equal size

During i th iteration, use data in partition D_i for validation, all other data as training data

Quality of classifier:
mean over all k iterations



Exhaustive Cross-Validation

Cross-validation methods which learn and test on all possible combinations to divide the original sample into training and test set.

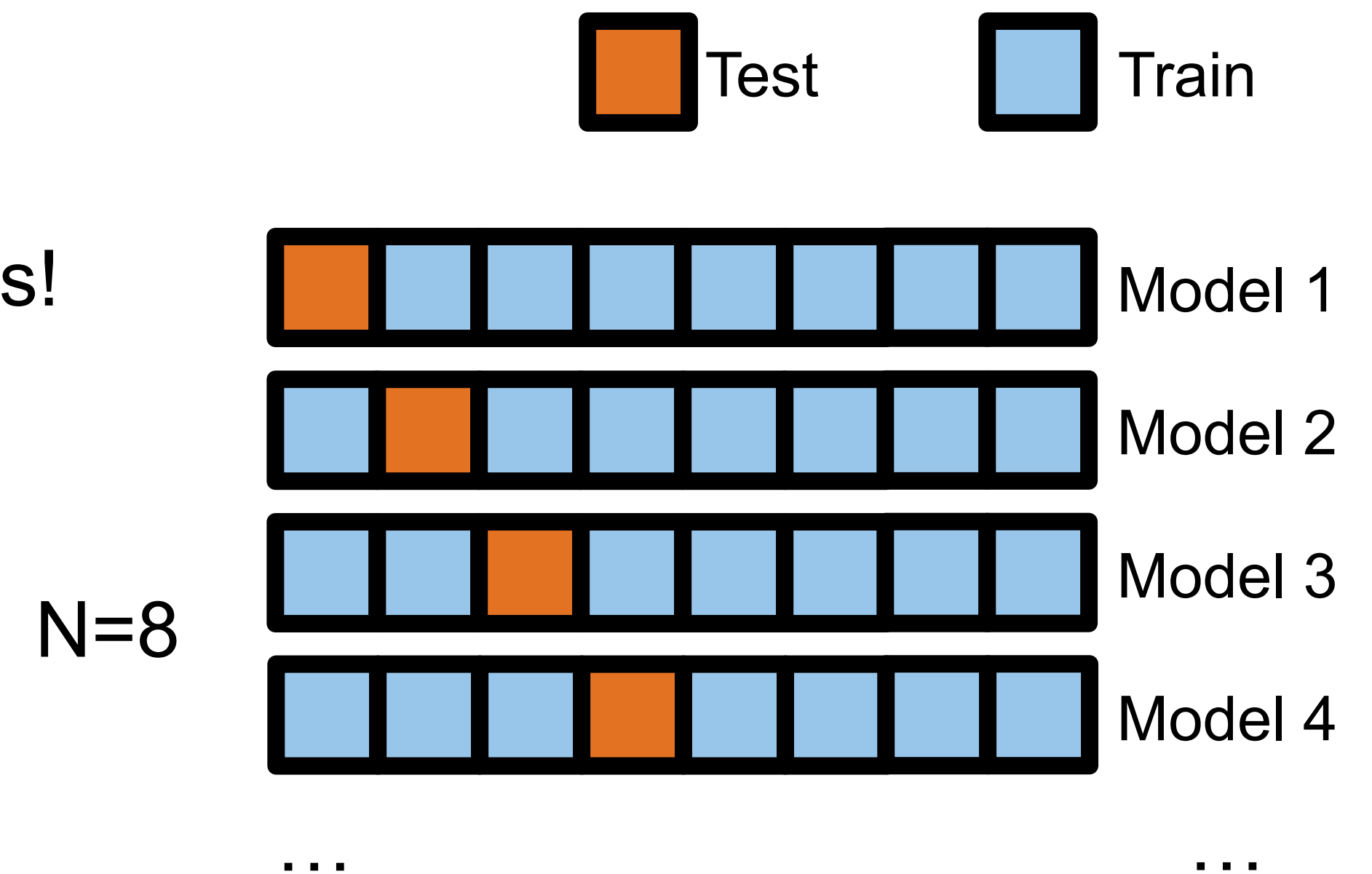
Leave-p-out cross-validation:

Use p observations as the test set and the remaining observations as training set.

Leave-one-out cross-validation:

Leave-p-out cross validation with $p=1$

Means finding one classifier for each instance – N classifiers!





Imbalanced data

Imbalance:

Number of samples of different classes are diverging significantly.

Often, collecting samples of a certain class is difficult because these are rare events.

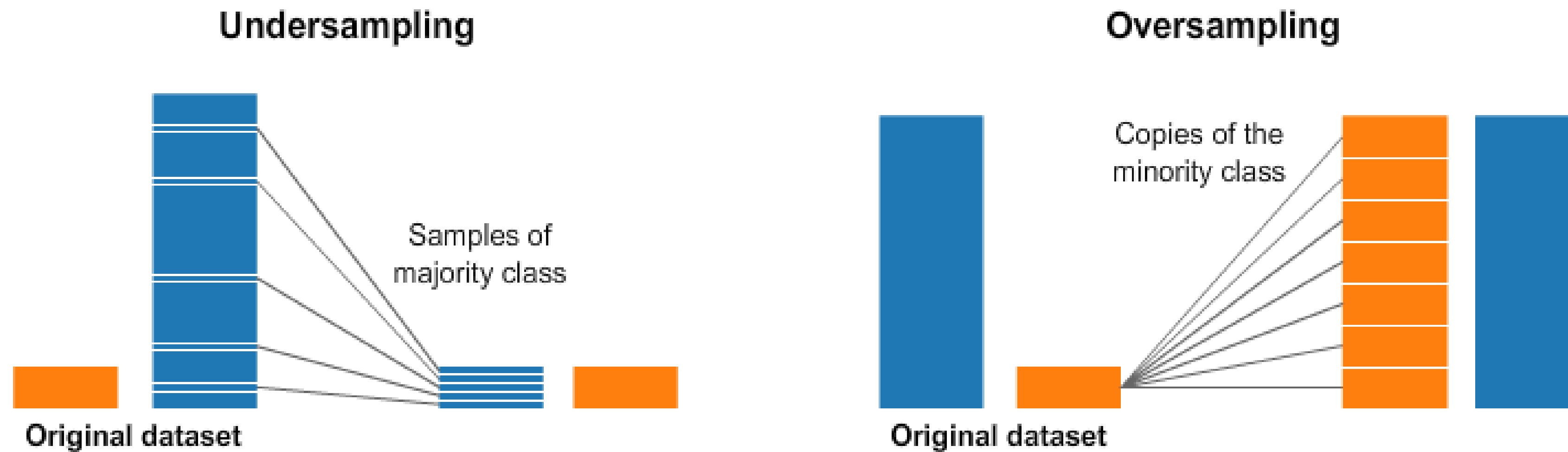
Consequences of building models using imbalanced data:

- Bias
 - Classifiers are more sensitive to detecting the majority class
- Optimization metrics
 - Metrics like accuracy may not report true performance

Has implications for sampling for cross validation!

Resampling

Balancing classes by removing samples from the majority class (under-sampling) and/or adding more examples from the minority class (over-sampling)



Various strategies, e.g. under-sampling by generating cluster-centroids, over-sampling by synthesizing elements (SMOTE), ...

Overfitting can occur on subtle ways

- Evaluation and task are adapted to solution
- Preprocessing of data tells something about solution
- Unbalanced data
- Little variety of data
- New observations
- Insufficient data



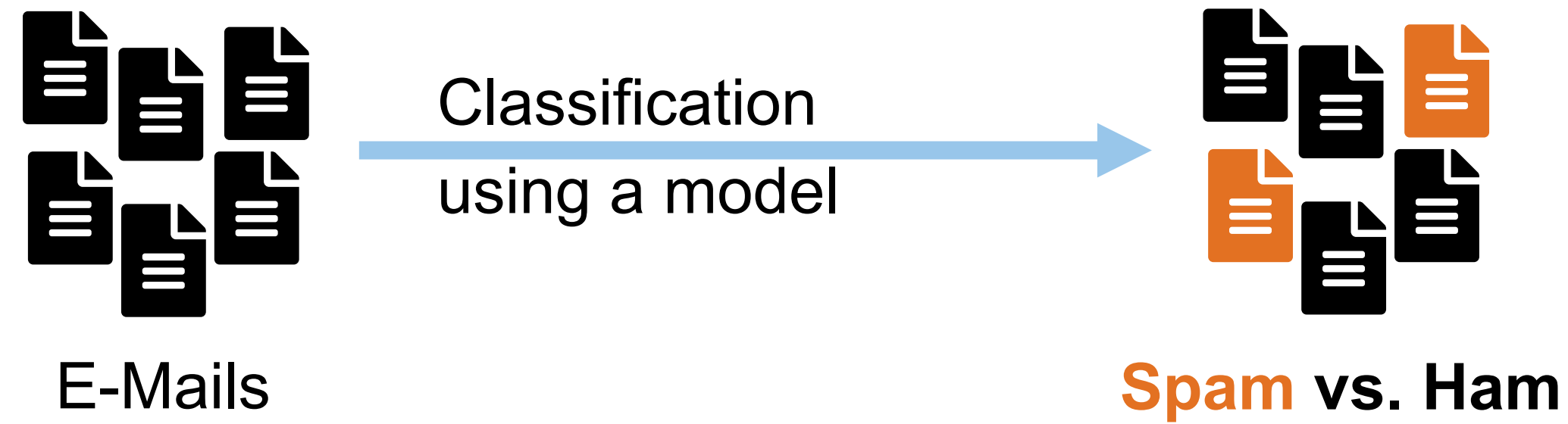
No Free Lunch Theorem

Choosing an appropriate algorithm requires making *assumptions*

With no assumptions, there will be no universal algorithm „better“ than random choice

“ [...] what an algorithm gains in performance on one class of problems is necessarily offset by its performance on the remaining problems;

Good Classifications: The Confusion Matrix



Model A:

		Correct class	
		Ham	Spam
Spam-Filter reports	Ham	200	38
	Spam	0	762

		Correct class	
		Ham	Spam
Spam-Filter reports	Ham	200	38
	Spam	0	762

Model B:

		Correct class	
		Ham	Spam
Spam-Filter reports	Ham	189	1
	Spam	11	799

		Correct class	
		Ham	Spam
Spam-Filter reports	Ham	189	1
	Spam	11	799

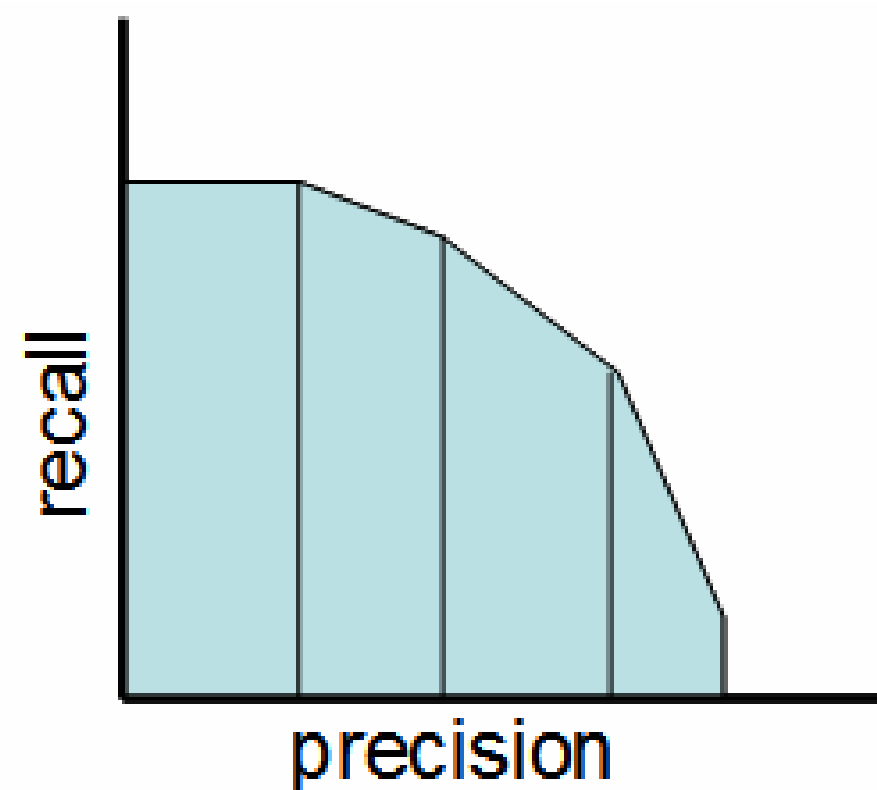
		Correct	
		Positive (P)	Negative (N)
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Measuring Goodness

- **Precision:** Proportion of predicted positives that are truly positive
good choice when we need to be very sure of prediction

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

- **Recall:** Proportion of actual positives that are correctly classified
good choice when as many positives as possible should be captured



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

		Correct	
		Positive (P)	Negative (N)
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Measuring Goodness

- **Accuracy:** Proportion of true results among total number of cases
good choice when classes are balanced

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

- **F_1 Score:** harmonic mean between precision & recall – a number between 0 and 1
good choice when we want a model with both good precision and recall

$$2 * \frac{\textit{precision} * \textit{recall}}{\textit{precision} + \textit{recall}}$$

Important variant F_β allows to apply a custom weight to precision & recall

		Correct	
		Positive (P)	Negative (N)
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Measuring Goodness & more

Prevalence

$$\frac{P}{P + N}$$

accuracy (ACC)

$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}$$

balanced accuracy (BA)

$$BA = \frac{TPR + TNR}{2}$$

F1 score

is the harmonic mean of precision and sensitivity:

$$F_1 = 2 \times \frac{PPV \times TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN}$$

phi coefficient (ϕ or r_ϕ) or Matthews correlation coefficient (MCC)

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Fowlkes-Mallows index (FM)

$$FM = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}} = \sqrt{PPV \times TPR}$$

informedness or bookmaker informedness (BM)

$$BM = TPR + TNR - 1$$

markedness (MK) or deltaP (Δp)

$$MK = PPV + NPV - 1$$

Diagnostic odds ratio (DOR)

$$DOR = \frac{LR+}{LR-}$$

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

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$$

specificity, selectivity or true negative rate (TNR)

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP} = 1 - FPR$$

precision or positive predictive value (PPV)

$$PPV = \frac{TP}{TP + FP} = 1 - FDR$$

negative predictive value (NPV)

$$NPV = \frac{TN}{TN + FN} = 1 - FOR$$

miss rate or false negative rate (FNR)

$$FNR = \frac{FN}{P} = \frac{FN}{FN + TP} = 1 - TPR$$

fall-out or false positive rate (FPR)

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} = 1 - TNR$$

false discovery rate (FDR)

$$FDR = \frac{FP}{FP + TP} = 1 - PPV$$

false omission rate (FOR)

$$FOR = \frac{FN}{FN + TN} = 1 - NPV$$

Positive likelihood ratio (LR+)

$$LR+ = \frac{TPR}{FPR}$$

Negative likelihood ratio (LR-)

$$LR- = \frac{FNR}{TNR}$$

prevalence threshold (PT)

$$PT = \frac{\sqrt{TPR(-TNR + 1)} + TNR - 1}{(TPR + TNR - 1)} = \frac{\sqrt{FPR}}{\sqrt{TPR} + \sqrt{FPR}}$$

threat score (TS) or critical success index (CSI)

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

		Correct	
		Positive (P)	Negative (N)
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Interpretability

Large project set out to evaluate ML application to problems in healthcare

Scenario: predicting pneumonia risk

Goal: predict probability of death for patients with pneumonia

The most accurate model of the study was a multitask neural net.

Outperformed other models by wide margin but was still dropped. Why?

One rule-based system learned the rule „patient has asthma → lower risk“

Reflected a true pattern in training data

The best model was the least intelligible one – was deemed to risky

No way of checking the features that were picked up

Goals of Interpretable Models

- **Trust: Identify and mitigate *bias***
Recognizing bias in a black-box algorithm is *very* hard
- **Causality: Account for context**
Helps you understand how the factors included in the model led to the prediction
- **Informativeness: Extract knowledge**
Helps you determine if patterns that appear to be present in the model are really there.
Rather learning from the model than evaluating it (compared to identifying bias)
- **Transferability: Generalize**
Models are trained on carefully collected datasets to solve narrowly defined problems. Interpretable models should help you determine if and how they can be generalized
- **Fair and Ethical Decision-Making**
algorithmic decision-making mediates more and more of our interactions. Need a way to make sure that decisions conform to ethical standards

Properties of Interpretable Models

1. Transparency

- **Simulatability**

Transparency at the level of the entire model

- **Decomposability / Intelligibility**

Transparency at the level of the individual components, e.g. parameters

- **Algorithmic Transparency**

Transparency at the level of the training algorithm

Properties of Interpretable Models

2. Post-hoc Interpretability

- Text Explanations
- Visualization
- Local Explanations
- Explanation by Example



Ensemble Models

Inspectable models ease debugging problems in data collection, feature engineering, etc

Ensemble models provide ways to restrict features to separate models that otherwise would interact in unintended ways

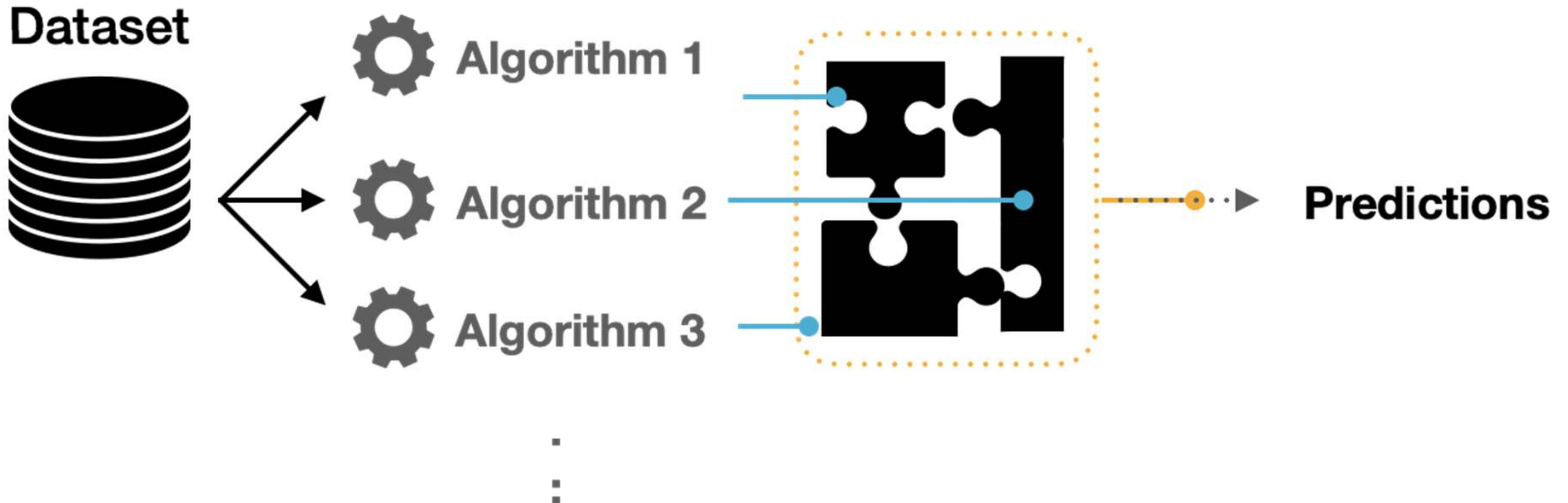
Helps overcoming

- high variance: single estimators are very sensitive to inputs to the learned features
- low accuracy: single models/algorithms might not be good enough
- features noise and bias: single estimators might rely heavily on one or few features

Various ways to combine models

Ensemble Models

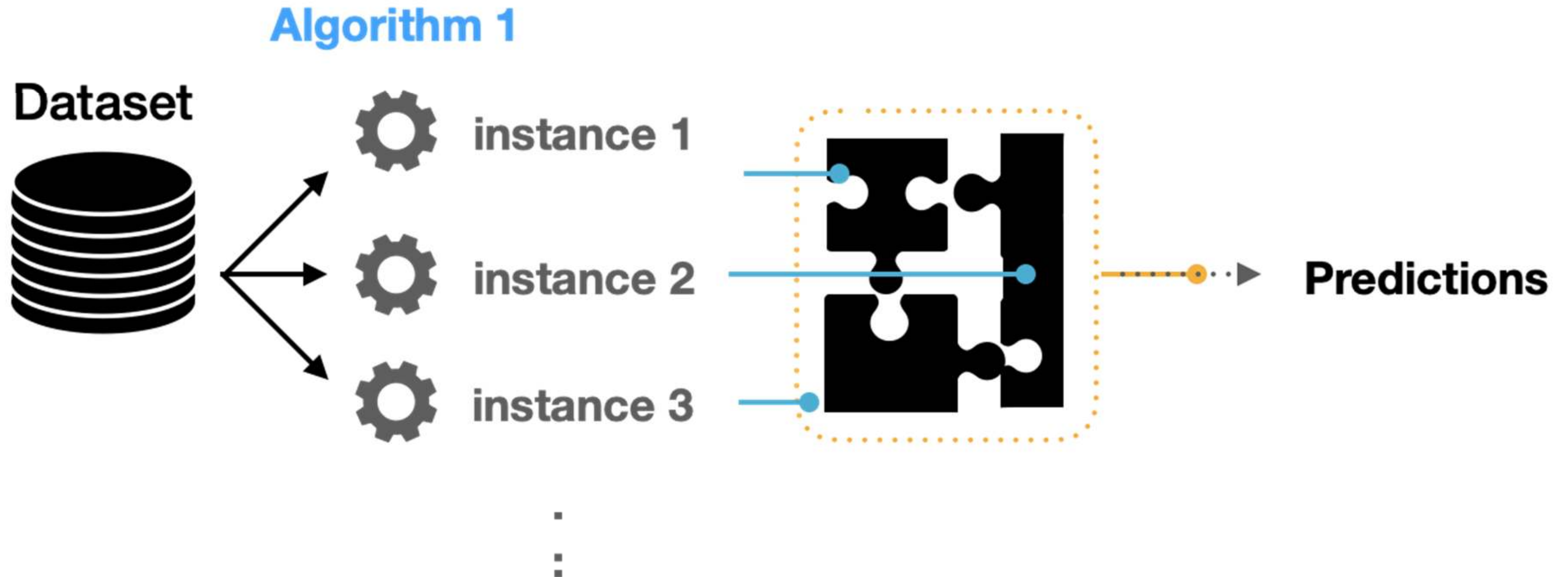
Using multiple algorithms to diversify model predictions



⋮

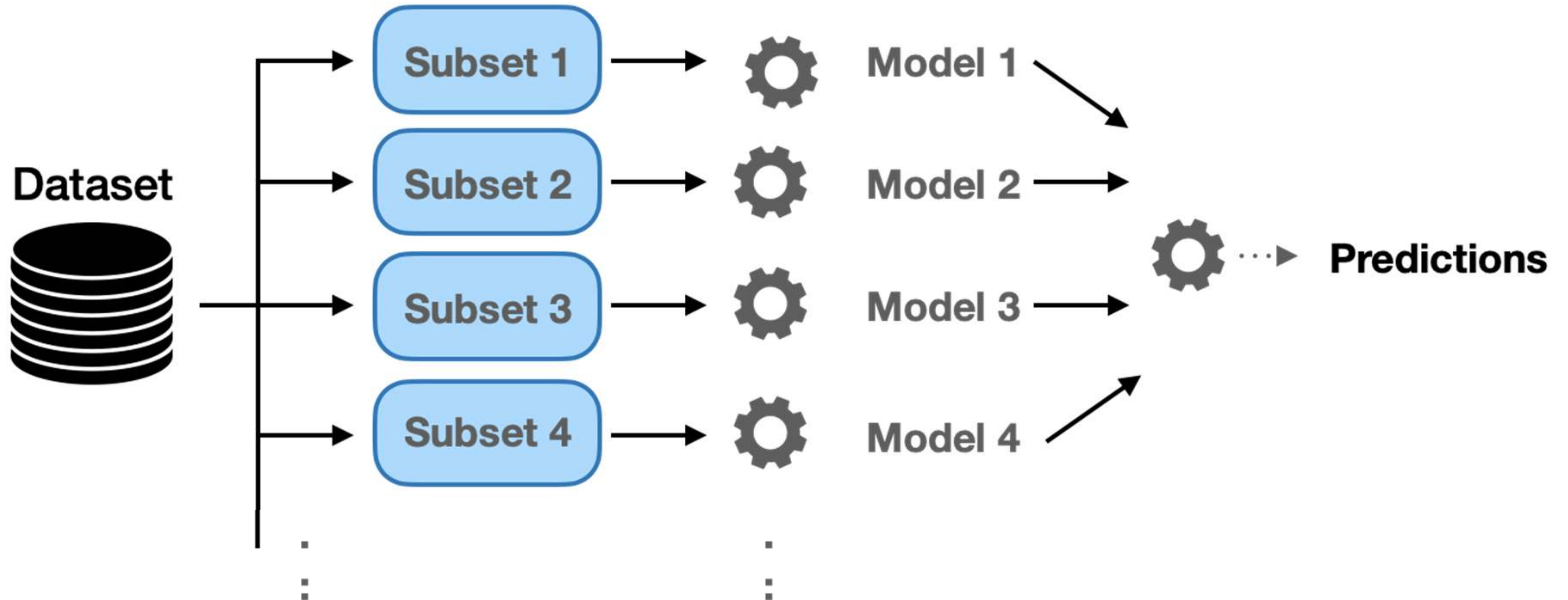
Ensemble Models

Using multiple weak instances of the same algorithm



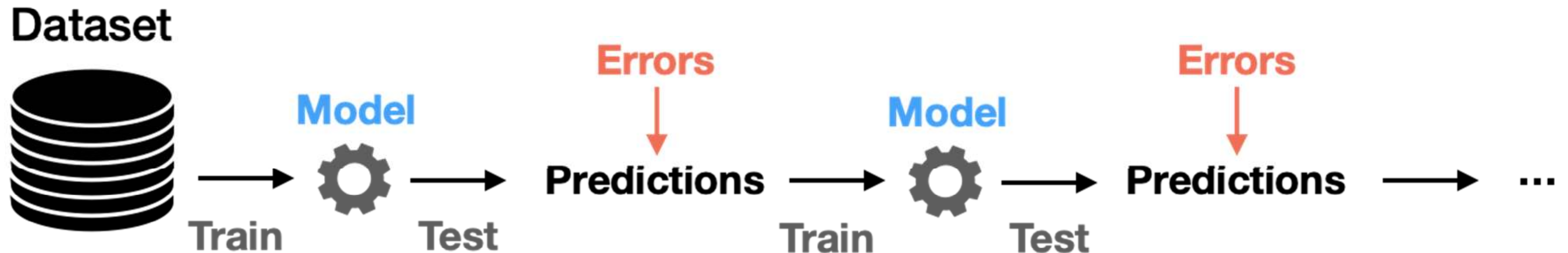
Ensemble Models: Bagging & Bootstrapping

Combine predictions from multiple models



Ensemble Models: Boosting

Ensemble of algorithms that builds models on top of several weak learners
Here: Sequential Adaptive Boosting (AdaBoost)



Ensemble Models: Stacking

Stacking intermediate predictions to make a final prediction

Algorithm 1



How to select a model?

- **Quality of predictions**
i.e. performance in terms of a quality metric
- **Speed**
i.e. training time, prediction time
- **Robustness**
i.e. handling noise or missing values and still classify correctly
- **Scalability**
i.e. computational efficiency
- **Interpretability**
subjective means
- **Other**



Thanks.

mirco.schoenfeld@uni-bayreuth.de

<https://xkcd.com/1838/>