Machine Learning

Revision

Definition of Machine Learning

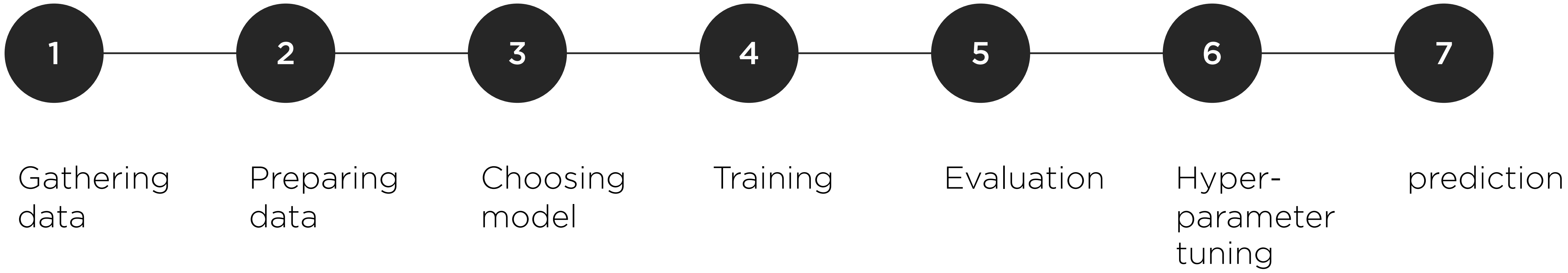
"A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."
-- Tom M. Mitchell, 1997

Machine learning is the study of algorithms that

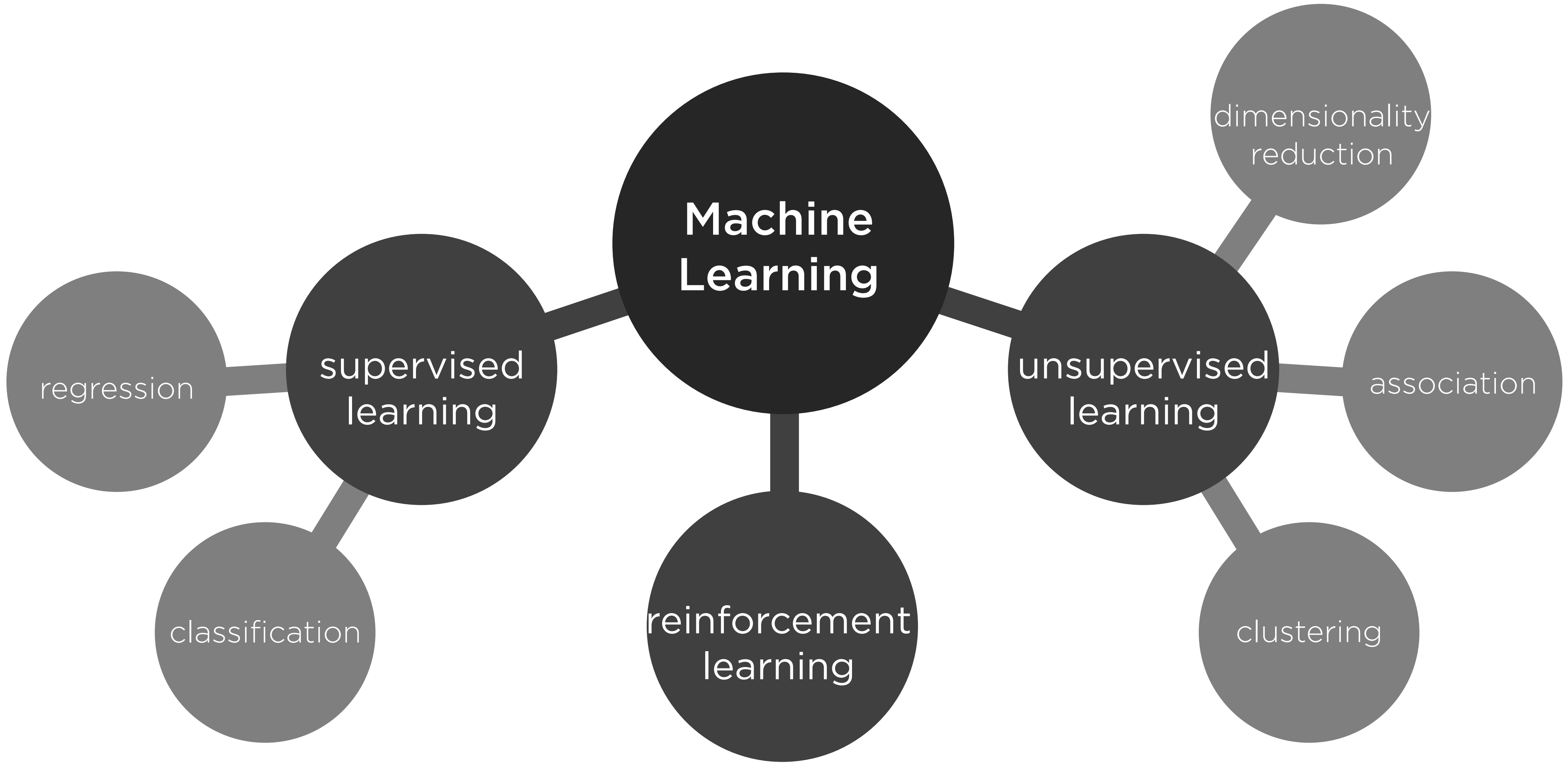
- improve their performance *P*
- at some task *T*
- with experience *E*

A well-defined learning task is given by *<P, T, E>*.

Machine Learning Lifecycle



Categories of Machine Learning



Key Terminologies

- **Label** is the variable that we are predicting
  typically represented by the variable **y**
- **Features** are input variables that describe our data
  typically represented by the variables $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_n\}$
- **Example** is a particular instance of data, $\mathbf{x}$
  - **Labelled example** has {features, label}: $(\mathbf{x}, \mathbf{y})$ used to train the model
  - **Unlabelled example** has {feature, ?}: $(\mathbf{x}, ?)$ used for making prediction on new data
- **Model** maps examples to predict labels: $\hat{y}$
  defined by internal parameters, which are learned
  - **Training** - creating or learning the model.
  - **Inference** - applying the trained model to unlabelled examples.

Linear Regression

A method to find the straight line or hyperplane that best fits a set of points.

$$\hat{y} = b + w_1 x_1$$

$\hat{\boldsymbol{y}}$	the predicted label (a desired output).
b	the bias (the y-intercept), sometimes referred to as w_0.
w_1	the weight of feature 1 (slope).
x_1	a feature (a known input).

Linear Regression

Training & Loss

Training a model: learning (determining) good values for all weights and the bias from labelled examples.

Goal of training: to find a set of weights and biases that have low loss, on average, across all examples.

Loss: the penalty for a bad prediction.

Empirical Risk Minimisation: the process of examining many examples and attempting to find a model that minimise loss.

Linear Regression

Training & Loss

$$Mean\ Square\ Error\ (MSE) = \frac{1}{N}\sum_{(x,y)\in D}(y - prediction(x))^2$$

- (x, y) is an example where
  - x is the set of features used by the model to make predictions.
  - y is the example’s label
- $\mathrm{prediction}(\mathrm{x})$ is a function of the weights & bias in combination with the set of features x.
- D is a dataset containing many labelled examples - (x, y) pairs.
- N is the number of examples in D.

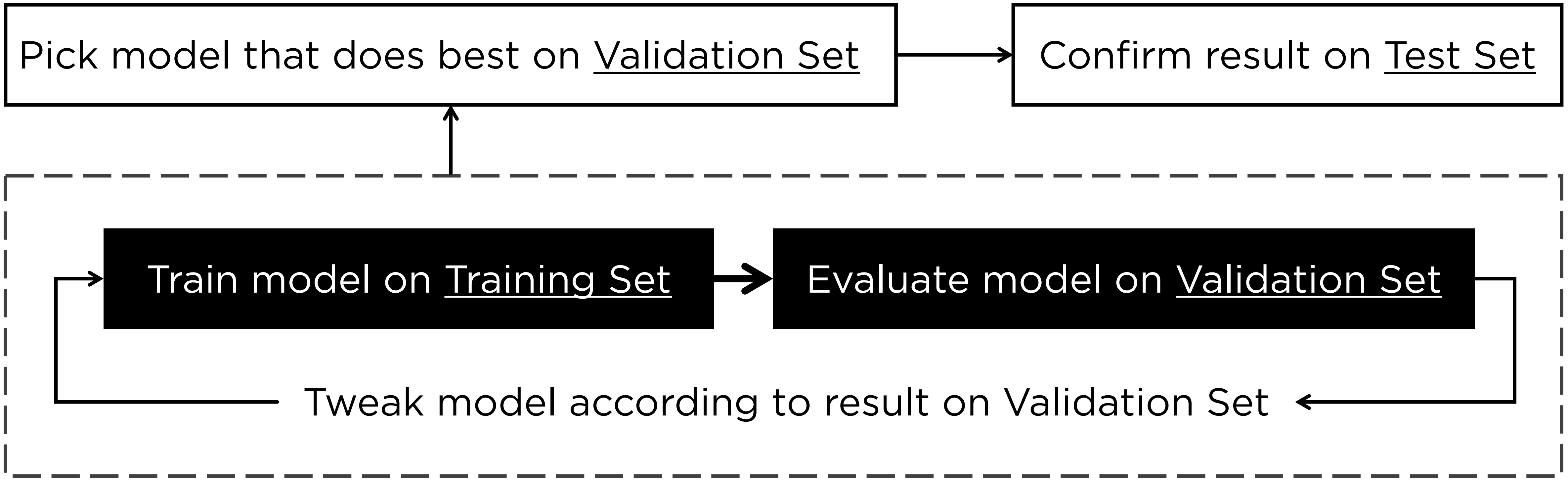
Linear Regression

Reducing Loss

- Hyperparameters are the configuration settings used to tune how the model is trained.
- Derivative of $(y- \hat{y})^2$ with respect to the weights and biases tells us how loss changes for a given example
  - Simple to compute and convex
- So we repeatedly take small steps in the direction that minimises loss
  - We call these Gradient Steps
  - This strategy is called **Gradient Descent**
- **Learning rate** $\frac{1}{f(x)''}$ is one of the hyperparameters of the training process.

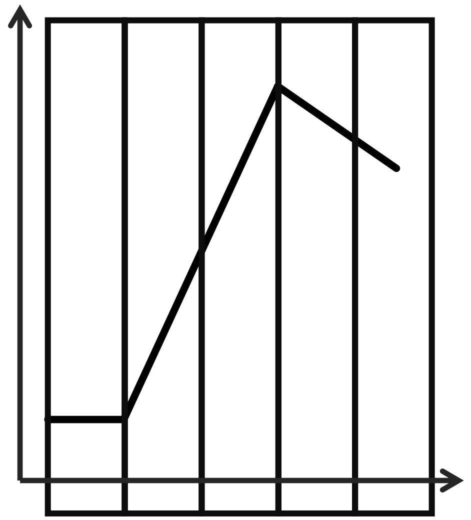
Training & Test Set

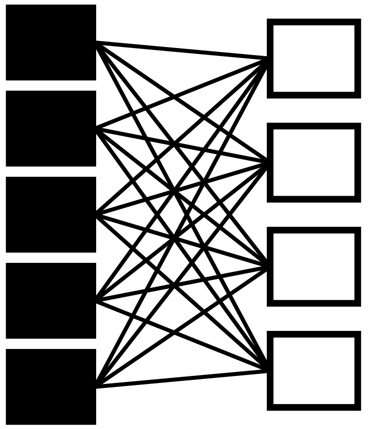
Better Workflow: Use a Validation Set

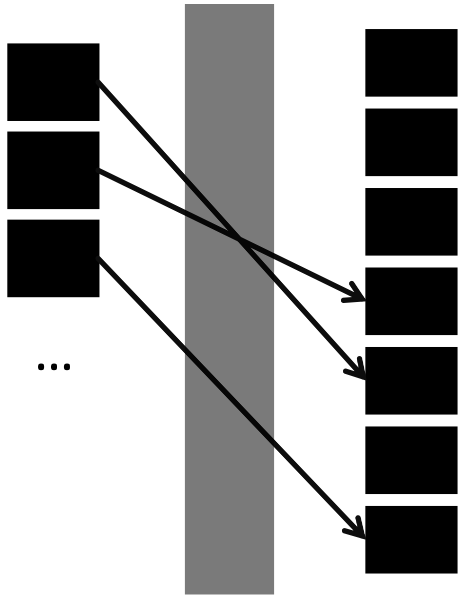

Pick model that does best on Validation Set
Confirm result on Test Set
Train model on Training Set
Evaluate model on Validation Set
Tweak model according to result on Validation Set

- Keeping the **test data** way off to the side (completely unused).
- Pick the model that does best on the **validation set**.
- Double-check that model against the **test set**.

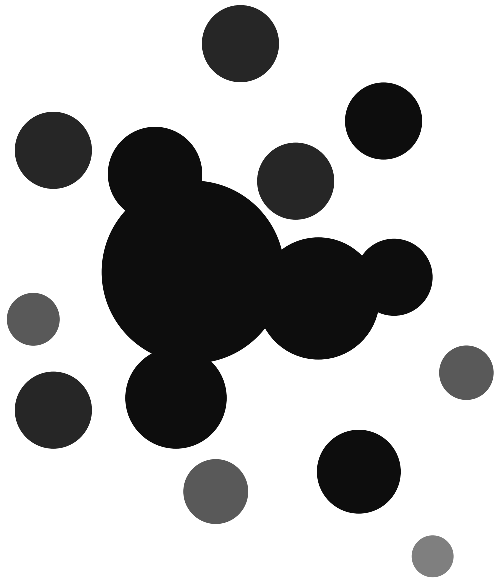
Representation


bucketing


crossing



hashing


embedding

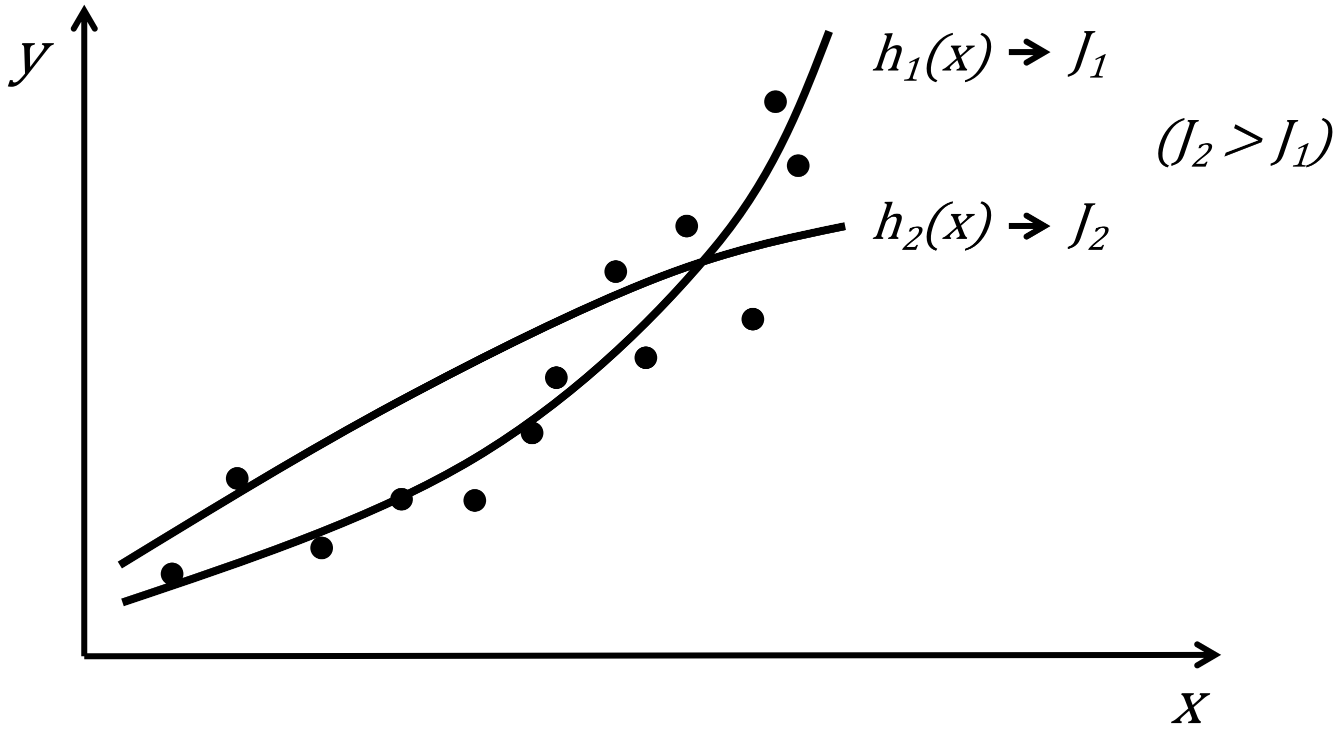
Numeric (Bucketing) / Categorical (Feature Crossing, Hashing, Embedding)

Cost Functions

- Describes how well the current response surface ***h(x)*** fits the available data (on a given data set):

$$J(y_i, h(x_i))$$

$\nearrow$ observed $\nwarrow$ predicted



- Smaller values of the cost function correspond to a better fit.
- Machine learning goal: construct ***h(x)*** such that *J* is minimised.
- In regression, ***h(x)*** is usually directly interpretable as predicted response.

Cost Functions

- Least Squares Deviation Cost $J(y_i, h(x_i)) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h(x_i))^{2}$

- Least Absolute Deviation Cost $J(y_i, h(x_i)) = \frac{1}{n}\sum_{i=1}^{n}|y_i - h(x_i)|$

- Huber-M Cost $J(y_i, h(x_i)) = \frac{1}{n}\sum_{i=1}^{n}\begin{cases}0.5(y_i - h(x_i))^2, if\,|y_i - h(x_i)| < \delta \\ \delta(|y_i - h(x_i)| - 0.5\delta), otherwise\end{cases}$

Binary Classifier

- Observed response $\mathbf{y}$ takes only two possible values **+** and **–**
- Define relationship between $\boldsymbol{h(x)}$ and $\boldsymbol{y}$
- Use the decision rule: $\hat{y} = \begin{cases} +, & h(x) \geq t \\ -, & otherwise \end{cases}$

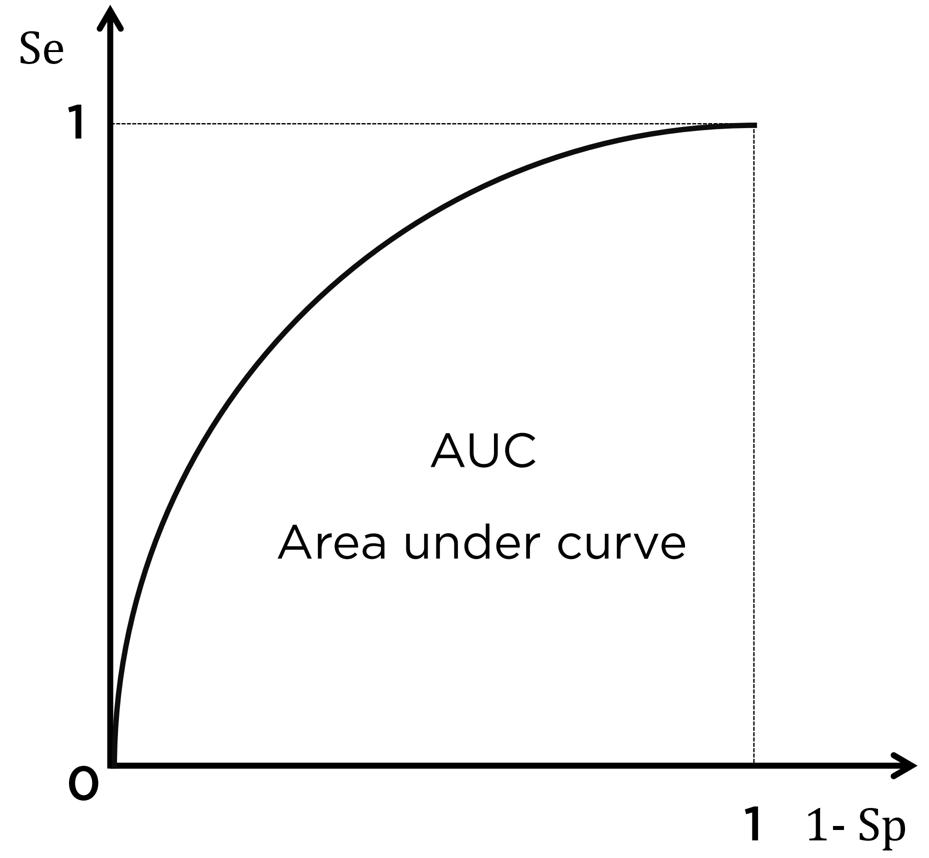
Performance Measures

Prediction Success (Confusion Matrix)

actual class

predicted class	+	−
+	true positives	false positives
−	false negatives	true negatives

ROC Curve



- Precision, Sensitivity (Recall), Specificity

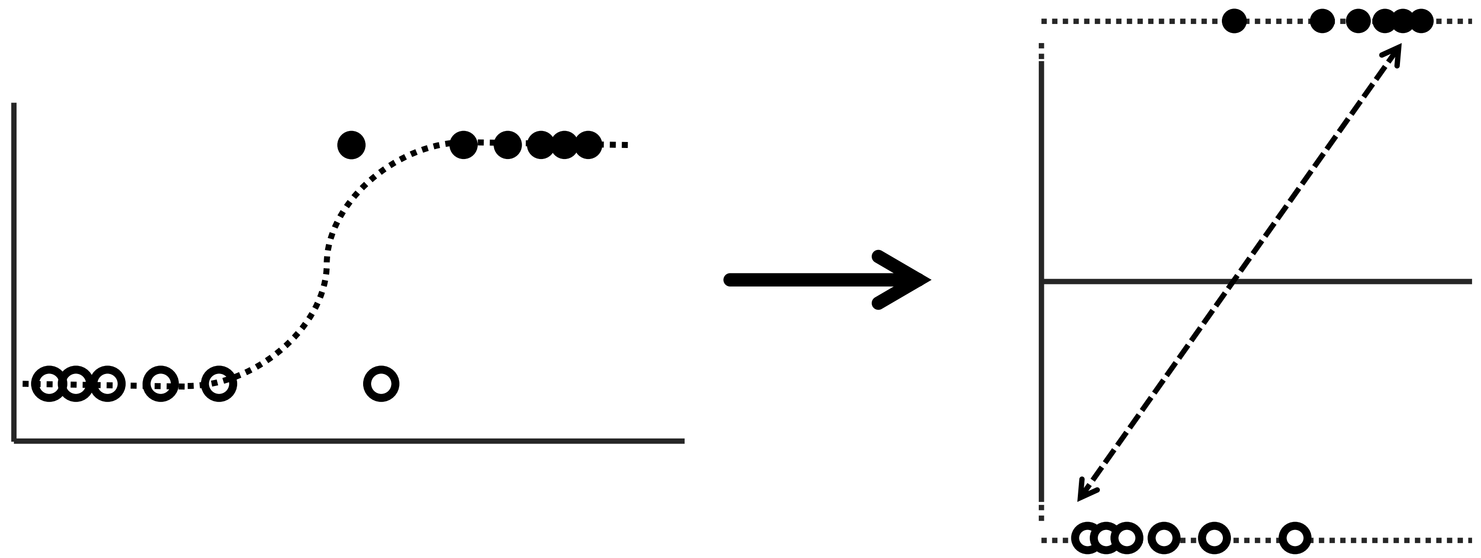
$$Pr = \frac{tp}{tp + fp} \qquad Se = \frac{tp}{tp + fn} \qquad Sp = \frac{tn}{tn + fp}$$

Logistic Regression

$$\log(\frac{p}{1-p}) = \log(odds)$$

Input: probability

Output: log(odds)

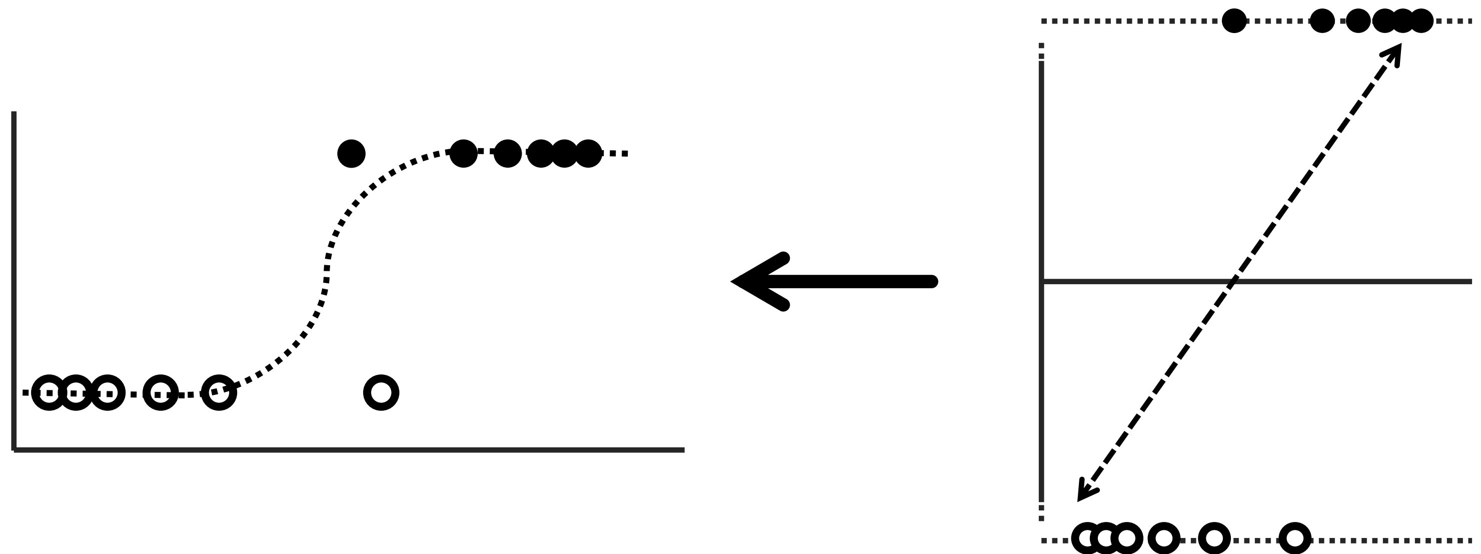


to

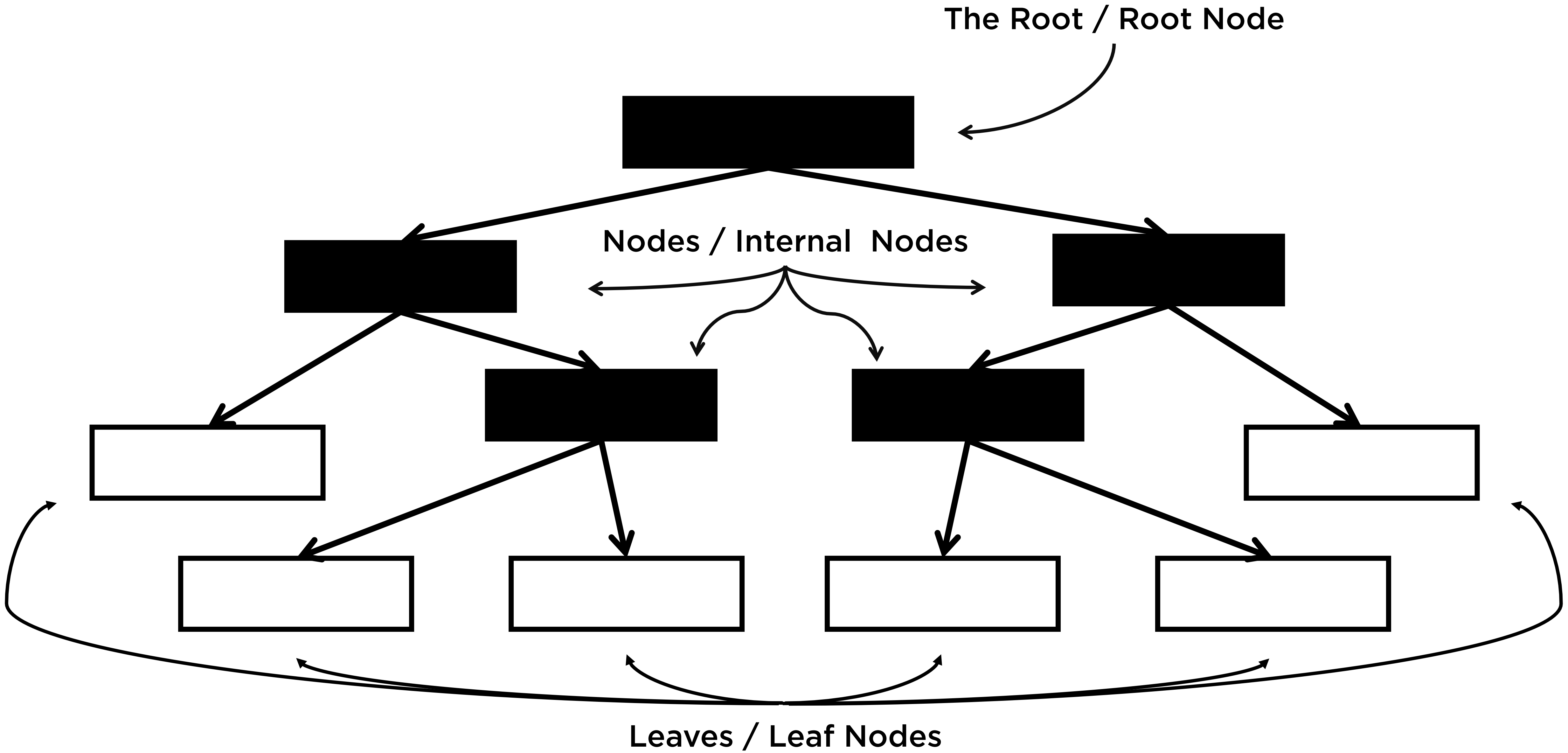
$$p = \frac{e^{\log(odds)}}{1+e^{\log(odds)}}$$

Input: log(odds)

Output: probability



Decision Tree



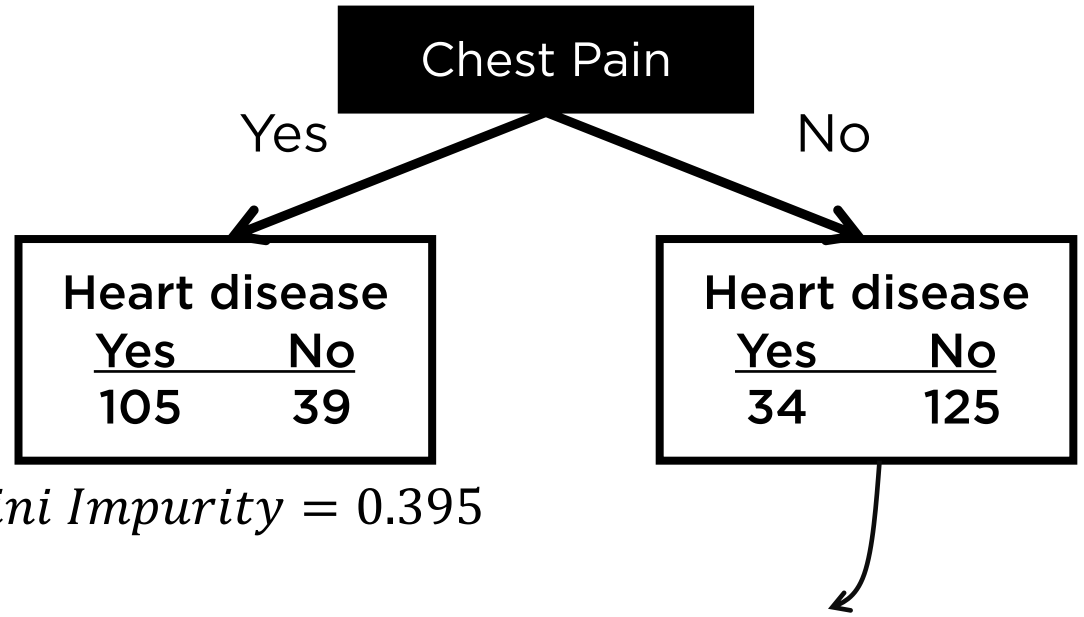
Decision Tree

Building a Decision Tree

1. Calculate all of the Gini impurity values.
2. If the node itself has the lowest value, leave it as a Leaf node.
3. If separating the data results in an improvement, then pick the separation with the lowest Gini impurity value.

Decision Tree

Calculating **Gini Impurity**



$Gini\ Impurity = 0.395$

$$Gini\ Impurity = 1 - (the\ probability\ of\ Yes)^2 - (the\ probability\ of\ No)^2$$

$$= 1 - \left(\frac{34}{34+125}\right)^2 - \left(\frac{125}{34+125}\right)^2$$

$$= 0.336$$

Decision Tree

Gini Impurity vs Entropy

Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labelled if it was randomly labelled according to the distribution of labels in the subset. It measures how heterogeneous some value is over a set. In **decision trees**, we find criteria that make a set into more homogenous subsets. Information gain is often used to describe this difference that's being maximised though that term goes with **entropy** - entropy is the alternative to Gini impurity.

Gini: $Gini(E) = 1 - \sum_{j=1}^{c} p_j^2$

Entropy: $H(E) = -\sum_{j=1}^{c} p_j \log p_j$

Decision Tree

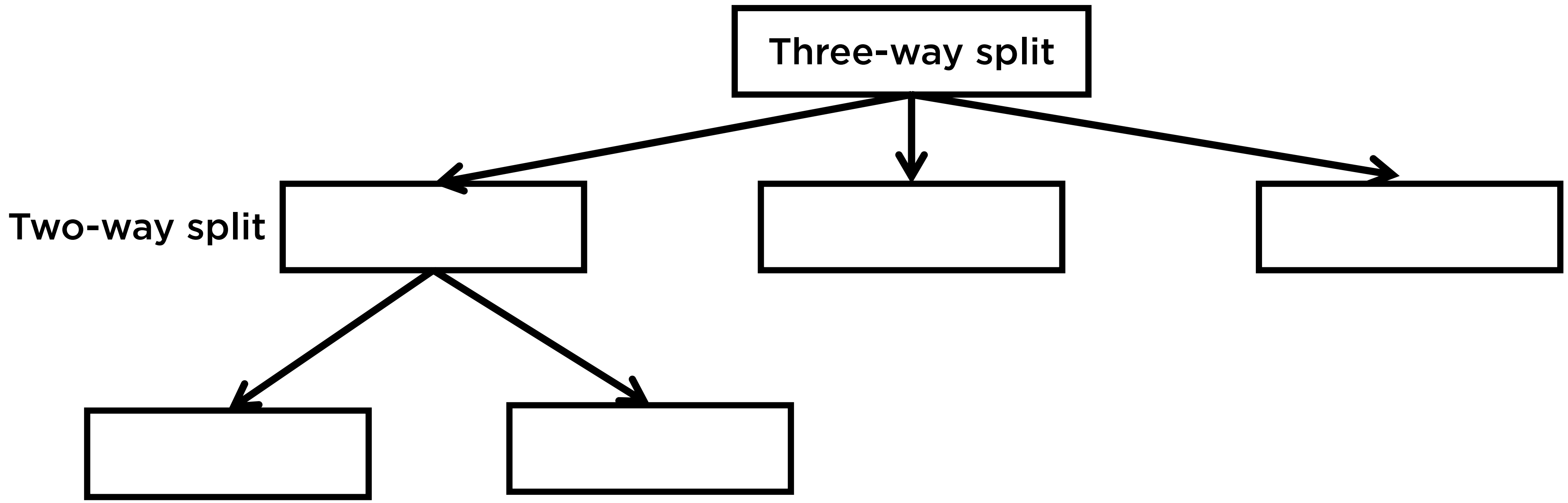
Building a Decision Tree from Numeric Data

1. Sort data
2. Calculate the average value for all adjacent data
3. Calculate the Gini/Entropy values for each average value.
4. Find the cutoff to split data

	Weight	Heart disease
	155	No
Gini/Entropy? ←	**167.5**	
	180	Yes
Gini/Entropy? ←	**185**	
	190	No
Gini/Entropy? ←	**205**	
	220	Yes
Gini/Entropy? ←	**222.5**	
	225	Yes

Decision Tree

A node can be a two-way (binary) split, a three-way split, and more.



Decision Tree

Advantages

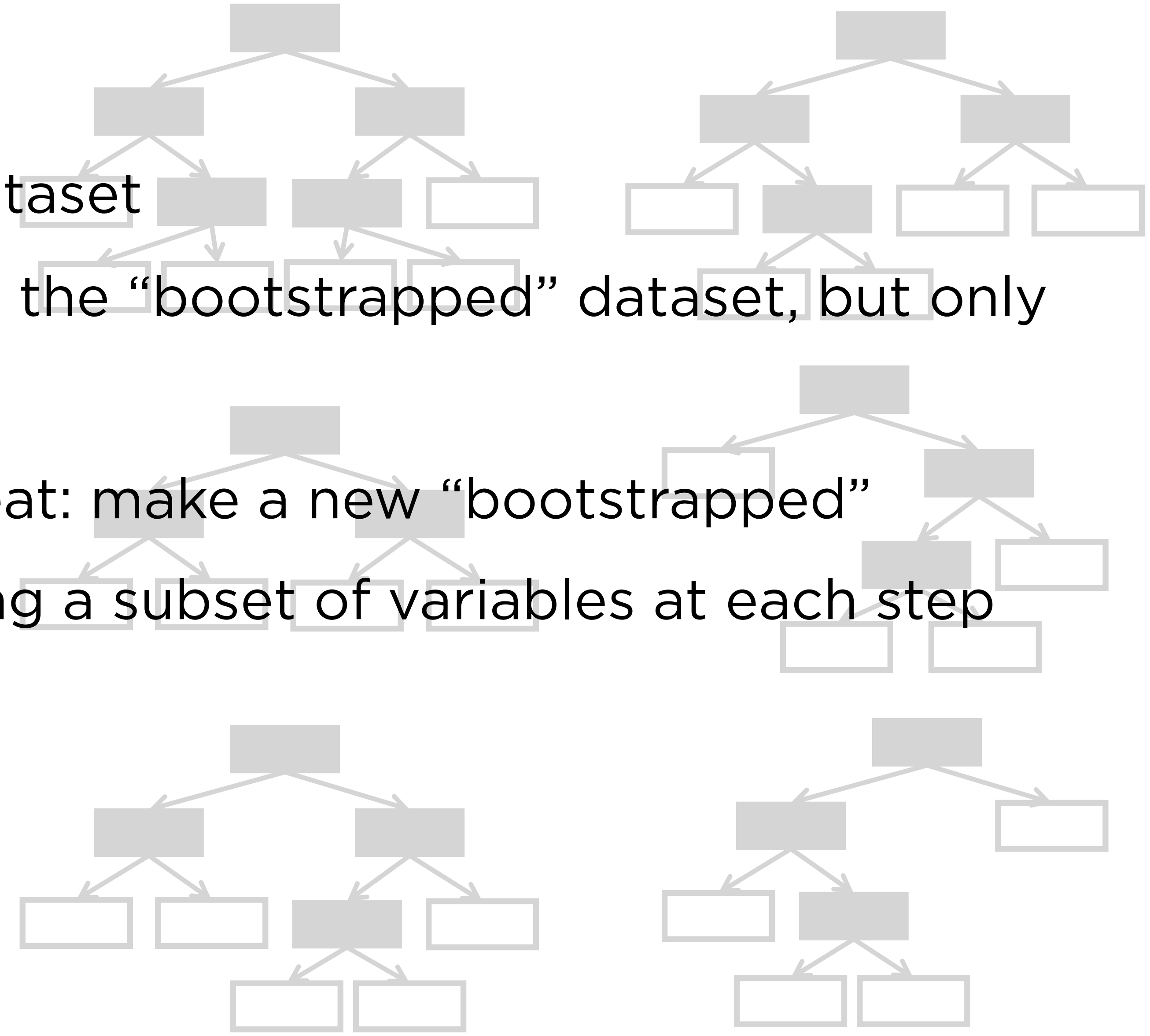
- Easy to understand
- Easy to generate rules
- Little effort for data preparation
- Less data cleaning required
- Data type is not constraint

Disadvantages

- May suffer from **overfitting**
- Does not easily handle nonnumeric data
- Can be quite large pruning is necessary

Random Forest

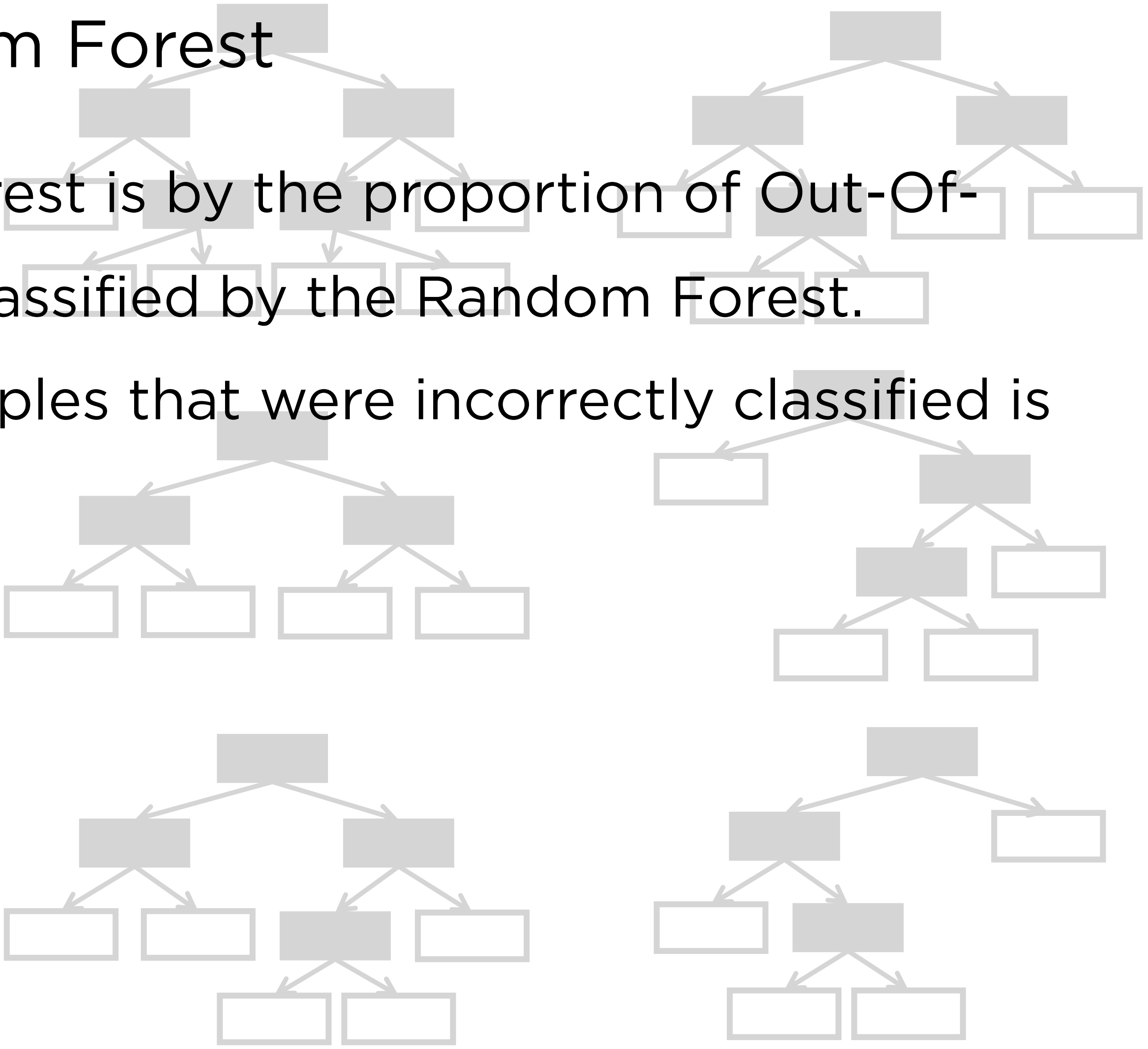
Building a Random Forest



- **Step 1**: create a “bootstrapped” dataset
- **Step 2**: build a Decision Tree using the “bootstrapped” dataset, but only use a random subset of variables
- **Step 3**: go back to Step 1 and repeat: make a new “bootstrapped” dataset and build a tree considering a subset of variables at each step (ideally 100’s of times)

Random Forest

Measuring accuracy of a Random Forest



- Measure **accuracy** of Random Forest is by the proportion of Out-Of-Bag samples that were correctly classified by the Random Forest.
- The proportion of Out-Of-Bag samples that were incorrectly classified is the "**Out-Of-Bag Error**"

Support Vector Machine

Definition

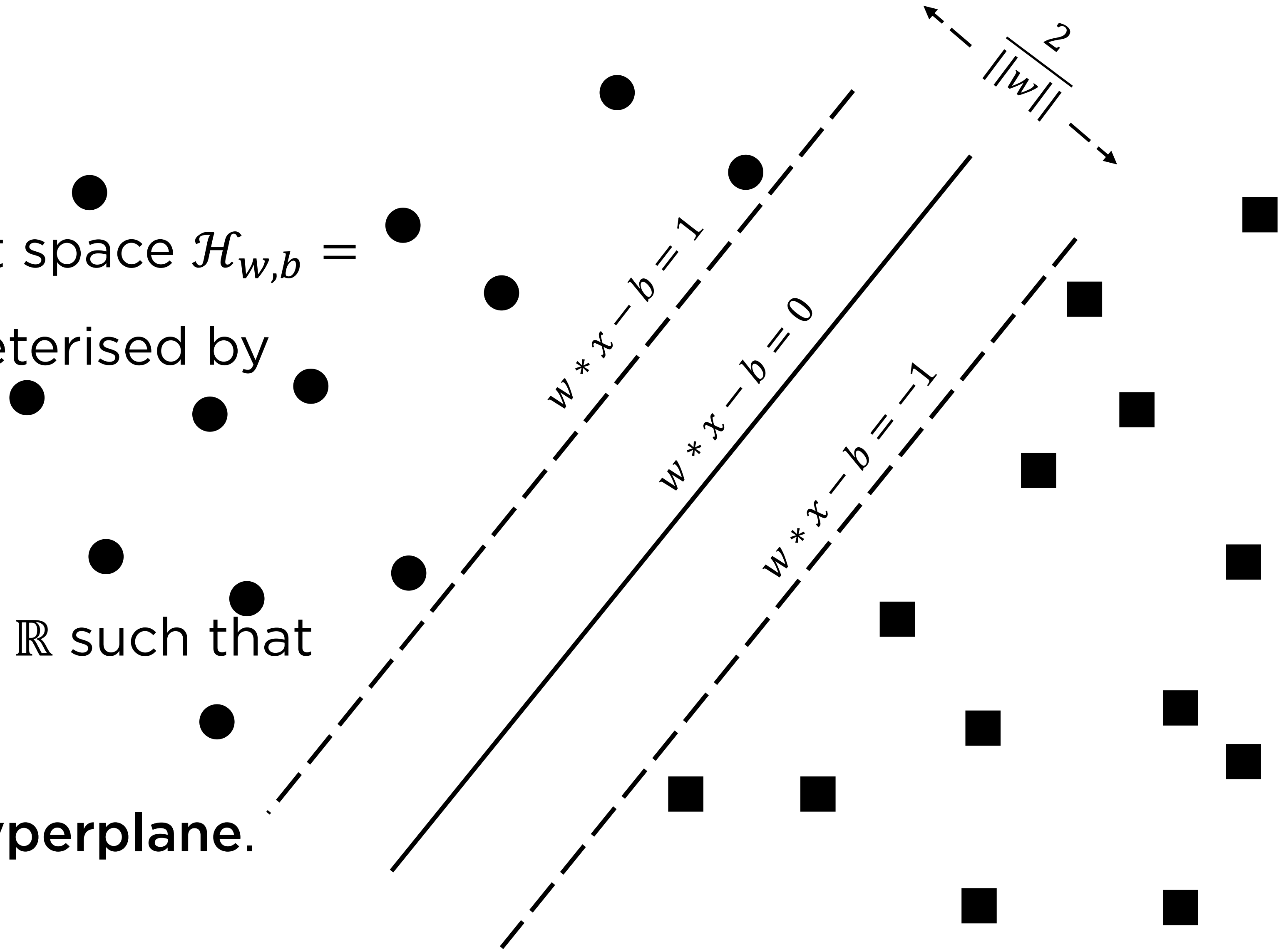
- More formally, a Support Vector Machine constructs a **hyperplane** or set of hyperplanes in a high- or infinite-dimensional space, that can be used for **classification**, **regression**, or other tasks like **outliers detection**.
- Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin, the lower the generalization error of the classifier.

Support Vector Machine

Support Vector Machine

Linear Separable SVM

- **Hyperplane**: a set of Hilbert space $\mathcal{H}_{w,b} = \{x \in \mathbb{R}^d : w^T x + b = 0\}$ parameterised by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$.
- Data are **linearly separable**, if there exist $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ such that $y_i(w^T x_i + b) > 0,\ i = 1, \dots, m$
- Then, $\mathcal{H}_{w,b}$ is a **separating hyperplane**.



Support Vector Machine

Linear Separable SVM

The **distance** $\rho_x(w,b)$ of a point x from a hyperplane $\mathcal{H}_{w,b}$ is:

$$\rho_x(w,b) = \frac{|w^T x + b|}{||w||}$$

If $\mathcal{H}_{w,b}$ separates the training set S we define its **margin** as:

$$\rho_x(w,b) = \min_{i=1:m} \rho_{x_i}(w,b)$$

If $\mathcal{H}_{w,b}$ is a hyperplane (separating or not) we also define the margin of a point x as $w^T x + b$ (note that this can be positive)

Support Vector Machine

Linear Separable SVM – Hard Margin SVM

- If the training data is linearly separable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is **as large as possible**.
- The region bounded by these two hyperplanes is called the "margin", and the **maximum-margin hyperplane** is the hyperplane that lies halfway between them.

Support Vector Machine

Linear Separable SVM – Soft Margin SVM

If the data is not linearly separable:

Minimise $\frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i$

Subject to $y_i(w^T x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, \dots, m$

The idea is to introduce the **slack variable** ξ_i to relax the separation constraints ($\xi_i > 0$)

Support Vector Machine

Linear Separable SVM – Soft Margin SVM

The role of the parameter C (**slack penalty**, a regularisation parameter)

- Small C allows constraints to be easily ignored, hence results in large margin.
- Large C makes constraints hard to ignore, hence results in narrow margin.
- When $C = \infty$, it enforces all constraints to become hard margin problem.

Clustering

Motivation

- Explores the unknown natures of the data that are integrated with little or no prior information
- Saves effort of data labelling, which can be extremely expensive and time consuming
- Provides a compressed representation of the data and is useful in large-scale data analysis

Clustering

K-means

Optimisation algorithm:

1. Initialise $\{\boldsymbol{\mu}_k\}$, $\mathrm{k}\epsilon 1, \ldots, K$, then keep $\boldsymbol{\mu}_k$ fixed and minimise E with respect to r_{nj}
2. Assign r_{nj} to each data point $\boldsymbol{x}_\mathrm{n}$ based on equation (2), then keep r_{nj} fixed and minimise $\boldsymbol{J}$ with respect to $\boldsymbol{\mu}_k$
3. Repeat step 1 and 2 until **convergence**

Clustering

K-means

Estimating the number of clusters – Silhouette analysis

- Study the separation distance between the resulting clusters
- The silhouette plot displays a measure of how close each point in one cluster is to points in the neighbouring clusters
- This measure has a range of [-1, 1].

Clustering

K-means

- Initialise cluster means and assign data points to clusters
- Iteratively repeat the two phases computation:
    - Re-assigning data points to clusters
    - Re-computing the cluster means
- Until there is no further change in the assignments (or until some maximum number of iterations is exceeded)

Clustering

K-means

Issues

- Convergence
- The number of K
- Robustness
- Extension of the Definition of Means

Good Luck!