Einführung in die Datenanalyse

Introduction to Data Science

Max Heimel, MSc

Prof. Dr. Volker Markl

Fachgebiet Datenbanksysteme und Informationsmanagement
Technische Universität Berlin

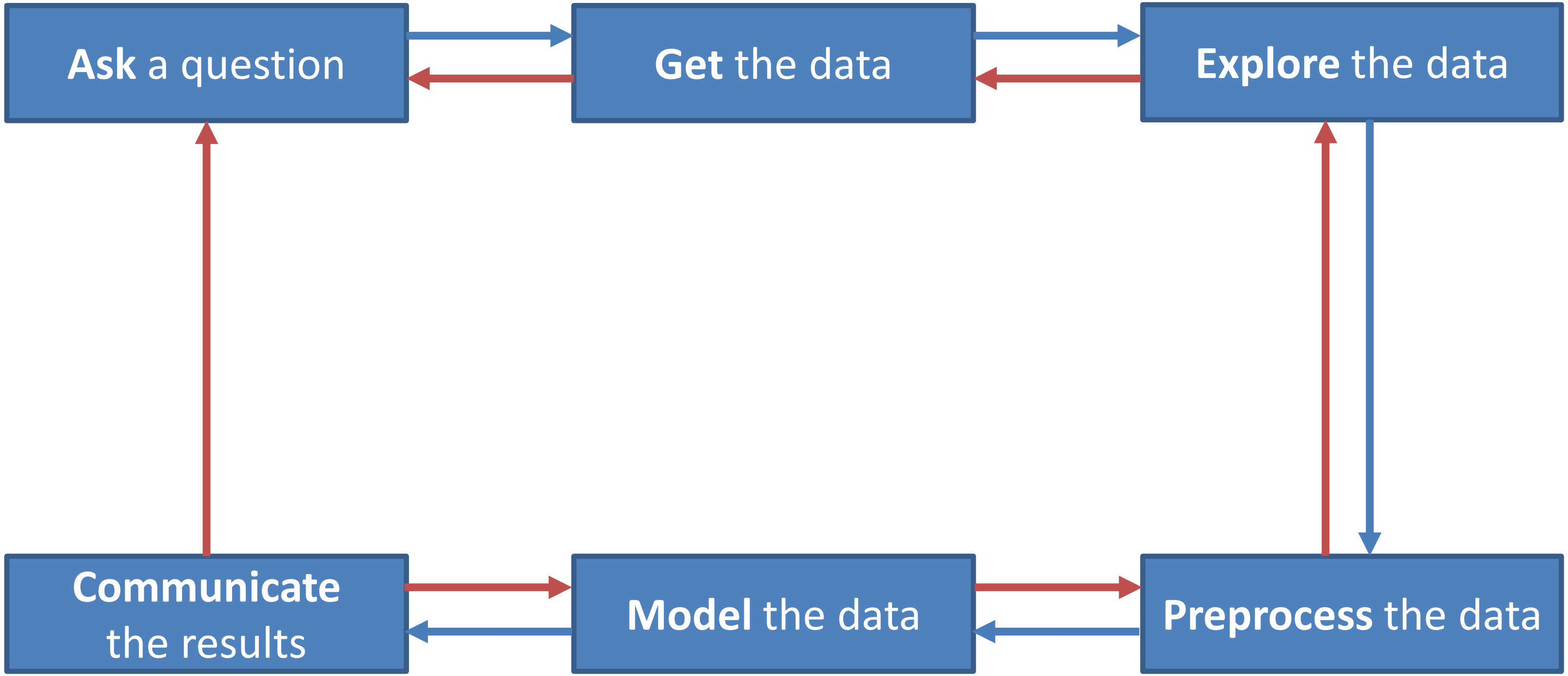
http://www.dima.tu-berlin.de/

Last Week.

- How to classify Data:
  - Structure (Structured vs. Unstructured).
  - Dimensionality (Univariate vs. Bivariate vs. Multivariate).
  - Variable Types (Qualitative vs. Quantitative).

- Exploratory Data Analysis:
  - First Step of the Data Analysis Process:
    - „Listen to the data“: Inspect data using tools from statistics and data visualization.
    - Used to identify interesting data aspects & important attributes.
  - Descriptive Statistics:
    - Central Tendency: Mean, Mode, Median.
    - Variability: Range, Interquartile Distance, Variance.
    - Correlation.
  - Visualization Methods:
    - Visualize Distributions: Histograms, Boxplots.
    - Visualize Correlation: Scatterplots, Scatter Matrices.

The Data Analysis Process.

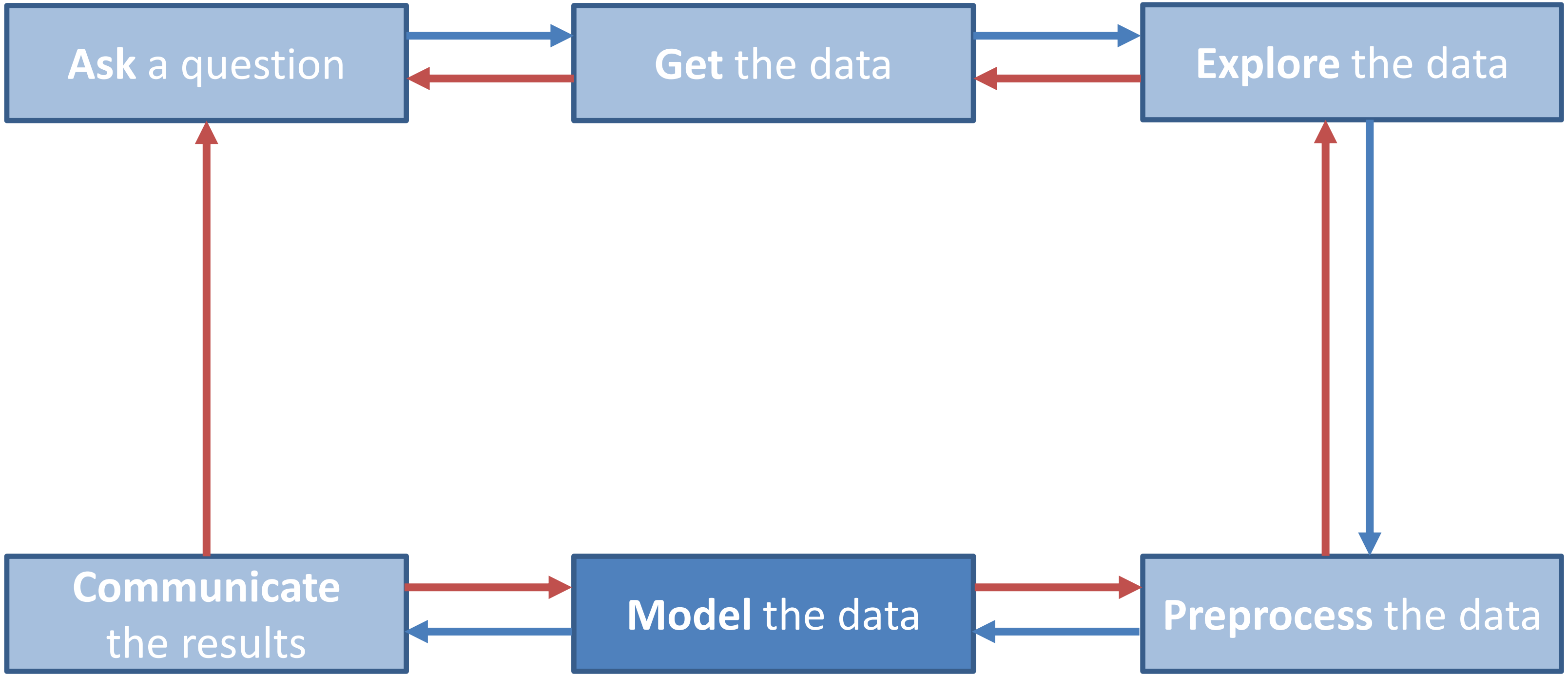


1. **What is Machine Learning?**

2. Supervised Learning

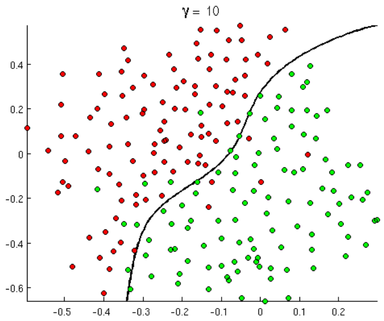
3. Unsupervised Learning

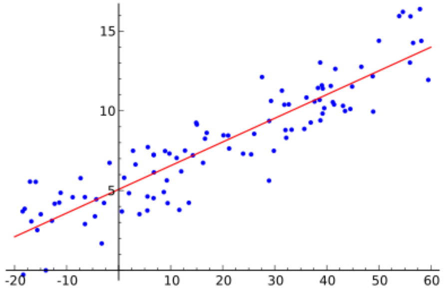
Today: Modelling Data.

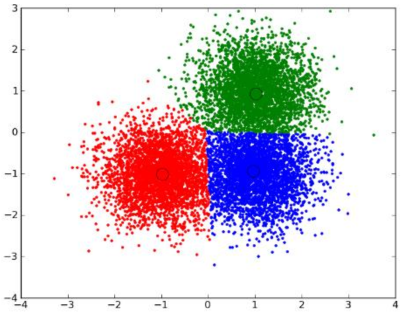


What is a model?

- A mathematical representation of “interesting” data aspects.
  - Typically some (mathematical) formula, configured by a parameter vector $\vec{\theta}$.
  - The actual shape of the formula / parameter vector depends on the model.







  - **Note:** Our focus is on statistical models, not data models (ER, MD)!
    - However, there are also analysis methods to induce data models ☺

- Modelling („Fitting“): Find the model parameters that best describe the data.

What is Machine Learning?

- Wikipedia: *"Machine Learning explores the construction and study of algorithms that can learn from and make predictions on data."*
  - For us: Methods to find the optimal model configuration(parameter vector) $\hat{\theta}$.
  - Data points are (typically) assumed to be numeric vectors, i.e. from $\mathbb{R}^d$.
  - "Applied (statistical) optimization theory"

Find the optimal parameter vector $\hat{\theta}$ …

… by picking the θ that minimizes …

… that depends on the dataset D.

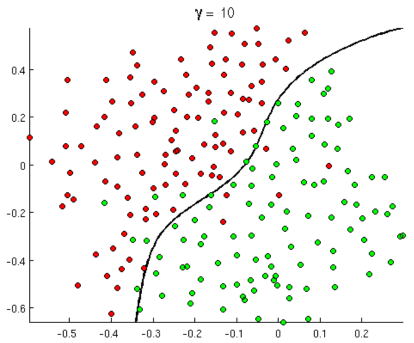
$$\hat{\theta} = argmin_{\theta}\, \mathcal{L}(D, \vec{\theta}, \dots)$$

… a given error function $\mathcal{L}$ …

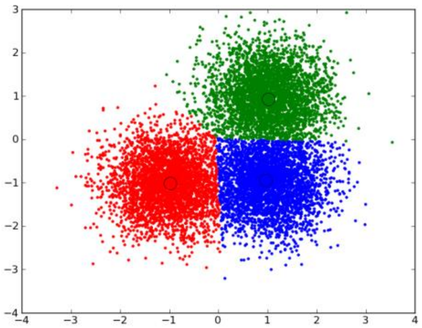
Fehlerfunktion: Wie gut beschreibt das Modell die Daten?

Supervised vs. Unsupervised Learning

- Supervised Learning:
  - Data is labeled: $D = \{(\vec{x}_1, y_1), \ \ldots \ ,(\vec{x}_n, y_n)\}$.
  - **Goal:** Predict the labels of unseen data points.
  - Examples:
    - Spam Classification (Label: *Is this Email message Spam?).*
    - Credit Score Prediction (Label: *Credit Score of the applicant*).
    - Voice Recognition (Label: *Word that is currently being spoken*).
    - Face Detection (Label: *Does this pixel belong to a face or not?*)



- Unsupervised Learning:
  - Data is unlabeled: $D = \{\vec{x}_1 \ \ldots \ ,\vec{x}_n\}$.
  - **Goal:** Describe and model the intrinsic structure of the data.
  - Examples:
    - Identify groups of similar customers.
    - Which products were frequently bought together?
    - (Lossy) compression of data / models.



Machine Learning Problems

	Supervised Learning	Unsupervised Learning
Quantitative Data	**Regression**	**Clustering** Dimensionality Reduction
Qualitative Data	**Classification** Recommendation	Association Analysis Sequence Mining

Course Overview

1. What is Machine Learning?

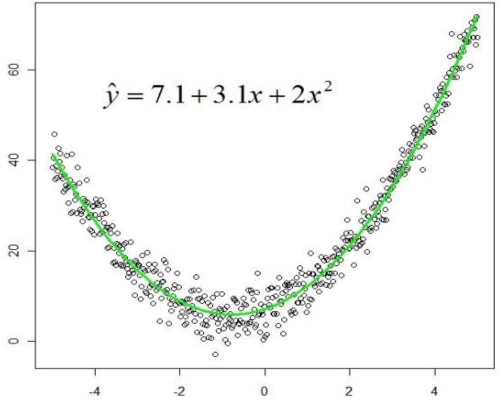
2. **Supervised Learning**

3. Unsupervised Learning

Regression

- Assume that labels are (continuous) numeric quantities: $\forall i: y_i \in \mathbb{R}$.
  - **Note:** Labels could also be vectors, but we'll assume scalars for simplicity.

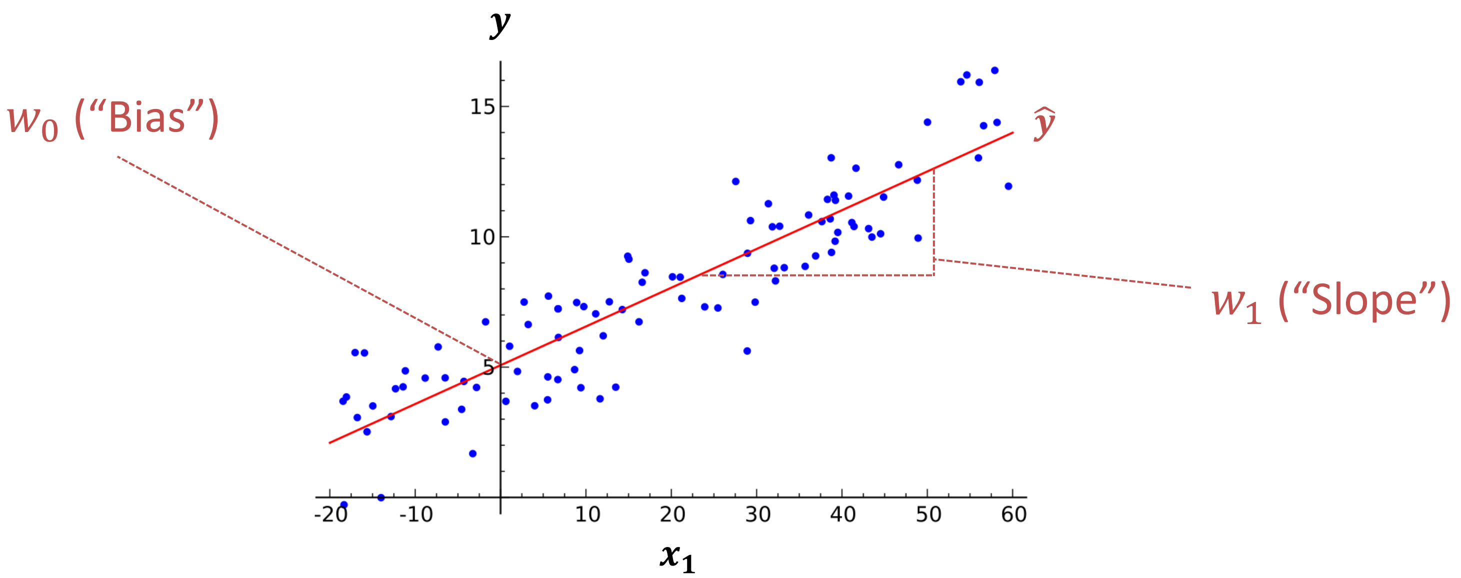
- The goal of regression analysis is to fit a *predictor function* $\hat{y}:\ \mathbb{R}^d \to \mathbb{R}$ to the data, minimizing the estimation error.
  - Also called “Line Fitting”.
  - Methods primarily differ in their choice for the shape of f.



- **Examples:**
  - Science: Gauss and Legendre used Regression Analysis to identify orbital parameters of comets from measurements of their positions.
  - Finance: Prediction of Housing / Share Prices.
  - Business Analysis: Resource & Demand Prediction.

Linear Regression

- One of the simplest regression models, assumes linear dependence.
  - $\hat{y} = w_0 + w_1 \cdot x_1 + \cdots + w_2 \cdot x_2$
  - Parameter vector: $\vec{\theta} = [w_0, \ldots, w_d]^T$.
  - Goal: Pick $w_0, \ldots, w_d$ that best describe the dataset.



Training Linear Regression Models

- Training a model means to find the parameter vector $\vec{\theta}$ that minimizes some error metric $\mathcal{L}: \mathbb{R}^2 \longrightarrow \mathbb{R}$ ("loss function") over the predictions.
  - Typically: Quadratic error $\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2$.

$$\begin{bmatrix} \hat{w}_0 \\ \hat{w}_1 \end{bmatrix} = \text{argmin}_{w_0, w_1} \underbrace{\frac{1}{n} \sum_{i=1}^{n} \overbrace{\left(w_0 + w_1 \cdot x_1^{(i)} - y^{(i)} \right)^2}^{\textit{Estimation error for the i-th data point.}}}_{\textit{Average estimation error across the dataset.}}$$

- How to solve this equation?
  - Closed-Form Solution: $\begin{bmatrix} \hat{w}_0 & \hat{w}_1 \end{bmatrix}^T = \left(X^T X \right)^{-1} X^T \vec{y}$
  - Gradient-Based Numerical Solvers: Gradient Descent, Conjugate Gradient, L-BFGS, ….
  - Rule-of-Thumb: High-dimensional → Numerical, Low-dimensional → Closed-Form.

(Non-)Linear Regression

- Despite the name, linear regression can also be used to train non-linear models:
  - Project data points into a higher-dimensional space by adding non-linear dimensions (“features”) during pre-processing.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \xrightarrow{\text{Projection}} \begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_2^2 & x_1 x_2 \end{bmatrix}^T$$

Original Data Point (2D) — *Expanded Data Point (6D)*

Trains — Trains

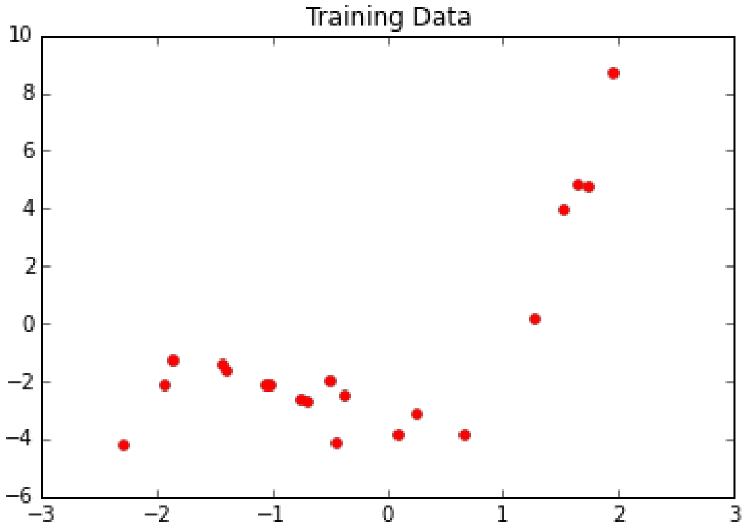
$$\hat{y} = w_1 \cdot x_1 + w_2 \cdot x_2$$

$$\hat{y} = w_0 + w_1 \cdot x_1 + w_2 \cdot x_1^2 + w_3 \cdot x_2 + w_4 \cdot x_2^2 + w_5 \cdot x_1 x_2$$

  - Typical model: Polynomials (weighted sum of exponentials of the original variables).
    - ➔ Allows to approximate arbitrary functions at high degrees (Taylor Expansion).

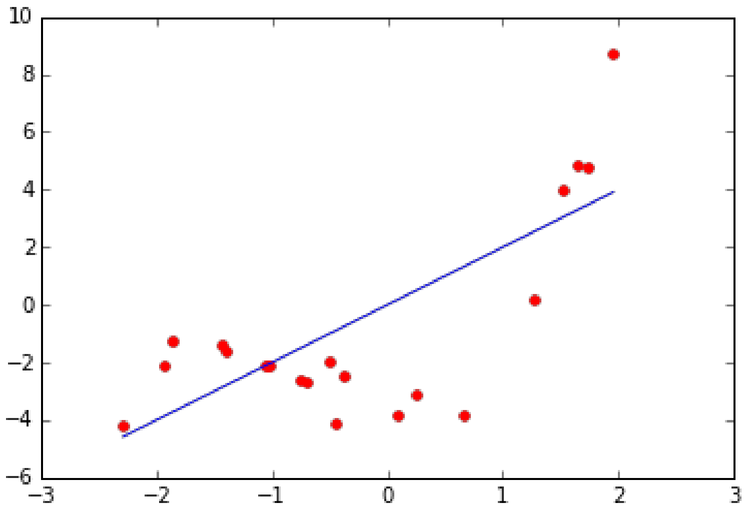
Example: Let's fit a polynomial!

- We generated labelled training data for: $y = x^3 + 2x^2 - x - 4$:
  1. Pick a random value for x from [-3:2].
  2. Use the polynomial to compute the label.
  3. Add some random (normal) noise to the label to simulate uncertain data.



Example: Let's fit a polynomial!

- Let's train a naïve linear regression model on the data!
  - Train directly on the X-values, don't perform any feature expansion.
  - $\hat{y} = w_0 \cdot x$

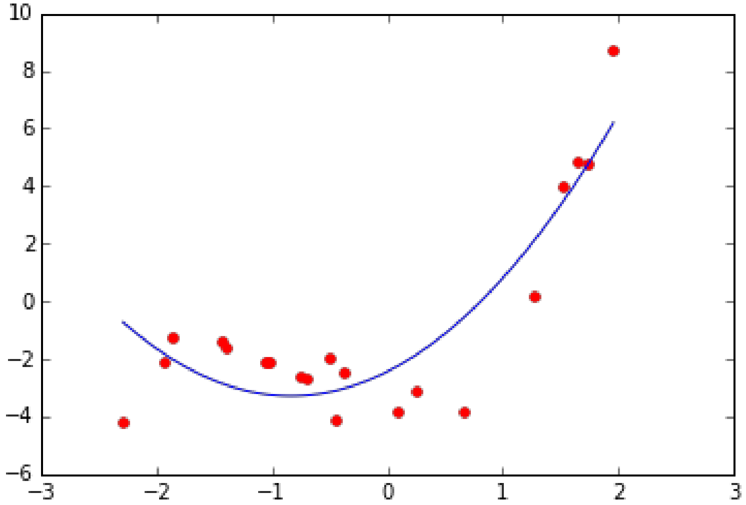


Training Error:
5.707

➔ No good match.
➔ Model **underfits**" data.

Example: Let's fit a polynomial!

- Alright, let's make our model more complex!
  - Train on X-Values and their squares (2nd degree polynomials).
  - $\hat{y} = w_0 + w_1 \cdot x + w_2 \cdot x^2$



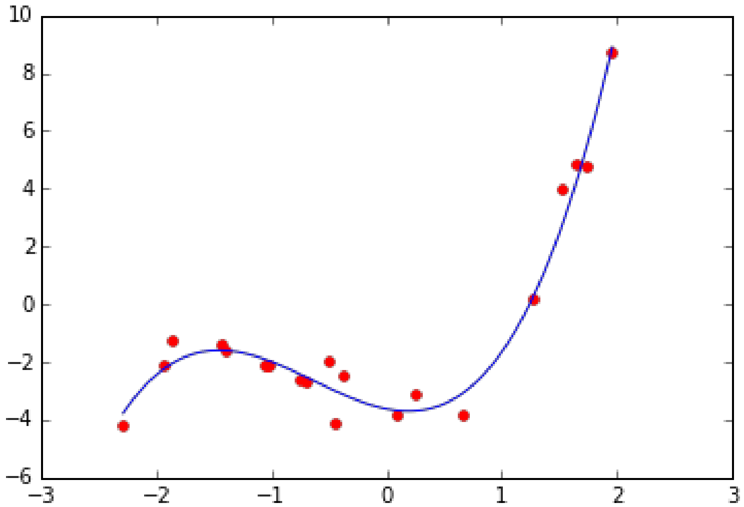
Training Error:
2.351

➔ Better, but let's keep going!

Example: Let's fit a polynomial!

- Let‘s match the data‘s inherent complexity.
  - Train on X-Values, their squares and their cubes (3rd degree polynomials).
  - $\hat{y} = w_0 + w_1 \cdot x + w_2 \cdot x^2 + w_3 \cdot x^3$

Training Error:
0.314



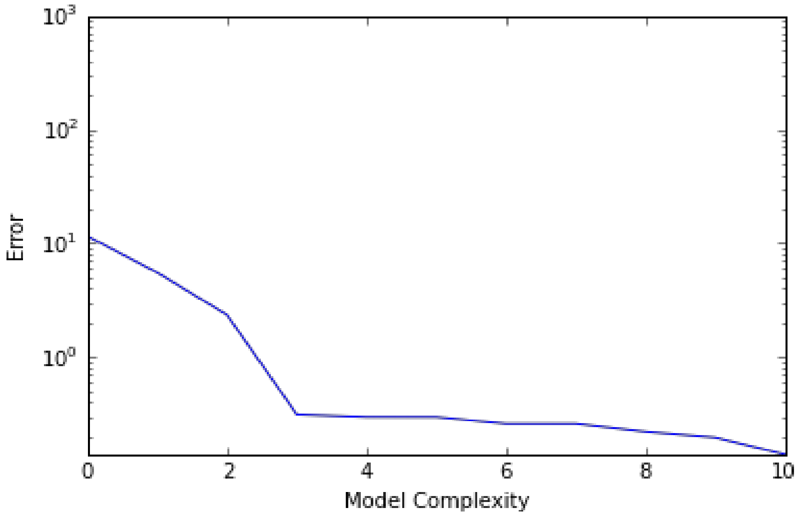
Trained Model:

$$\hat{y} = 1.09 \cdot x^3 + 2.14 \cdot x^2 - 1.29 \cdot x - 4.0$$

➔ Very good match ☺

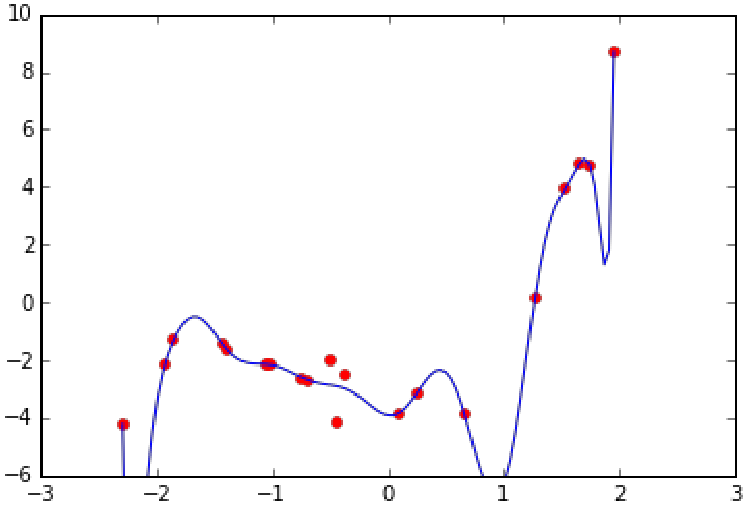
What if we keep increasing complexity?

- Increasing model complexity gives the training algorithm more „freedom“ (free parameters) to fit to the data.
  - ➔ Increasing model complexity generally decreases the training error.



Well ...

- Let‘s take a look at the trained model for a polynomial of degree 16:
  - $\hat{y} = w_0 + w_1 \cdot x + w_2 \cdot x^2 + \ \dots\ + w_{15} \cdot x^{15} + w_{16} \cdot x^{16}$



Training Error:
0.122

- Complex models indeed fit training data very well (low training error).
- **However:** They typically do not generalize! They **overfit** the data.

How to detect Overfitting.

- **Never evaluate your model quality based on the training data!**
- Instead:
  - Split your data into two disjoint sets: Training Set & Test Set.
  - Train your model on the Training Set.
  - Then evaluate the model quality based on the Test Set (= Test Error).

High Training & Test Error
➔ **Underfitting**

Low Training & High Test Error
➔**Overfitting**

Low Training & Test Error
➔ **Optimal Model.**

Prevent Overfitting.

- There are two principle strategies:
  1. Pick the optimal model based on the test error.

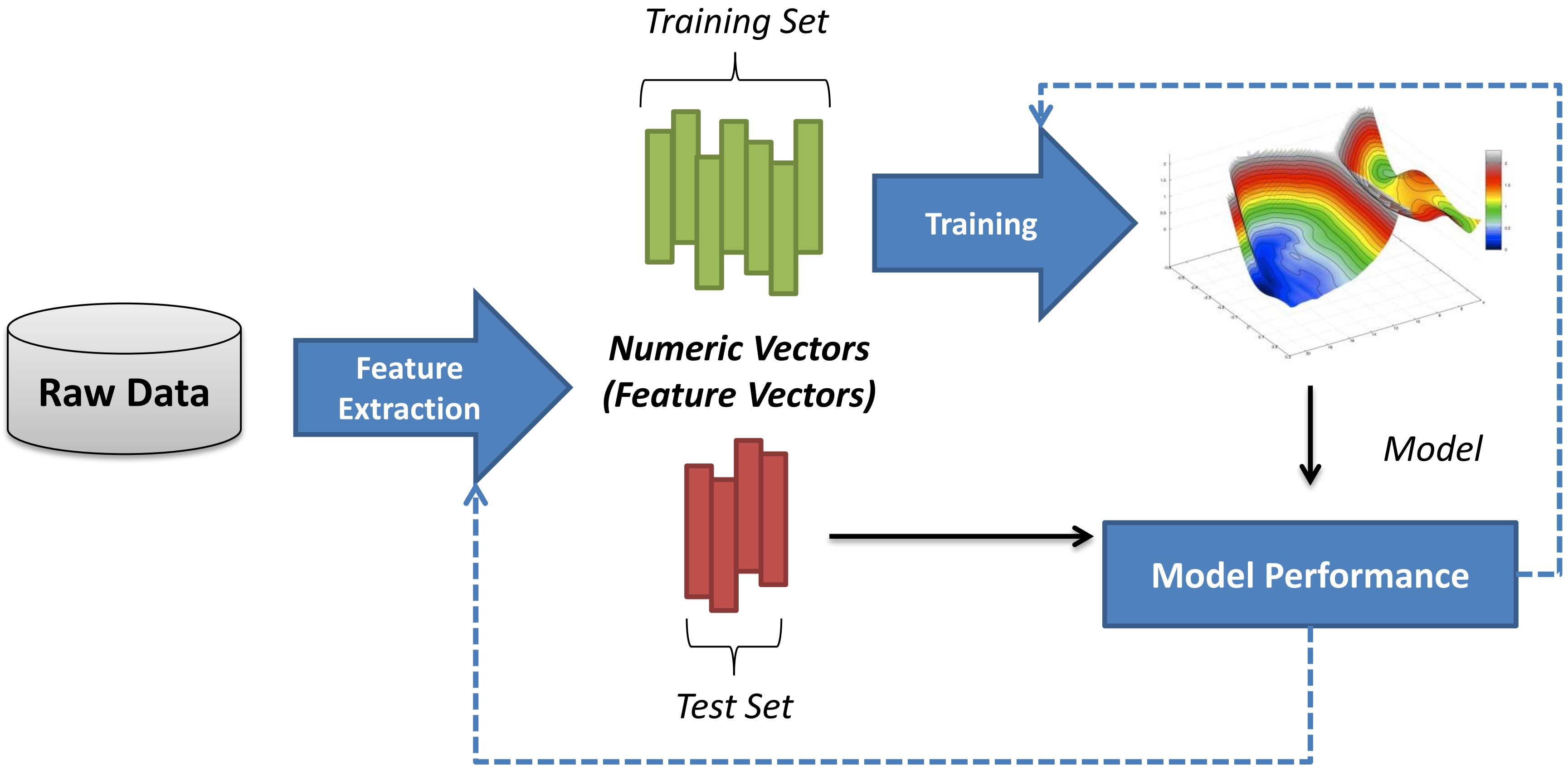
  2. Use model regularization:
     - Idea: Penalize “overly complex” models during model optimization.
     - For instance, LR with L2-Regularization („Ridge Regression“):

$$\hat{\theta} = \text{argmin}_{w_0, w_1} \left(\underbrace{\frac{1}{n} \sum_{i=1}^{n} \left(w_0 + w_1 \cdot x_1^{(i)} - y^{(i)} \right)^2}_{\textit{Linear Regression}} + \underbrace{\lambda \cdot \left|\left|\vec{\theta}\right|\right|_2^2}_{\textit{Regularization term}} \right)$$

     - Penalizes models with extreme parameters („long“ parameter vectors $\vec{\theta}$).

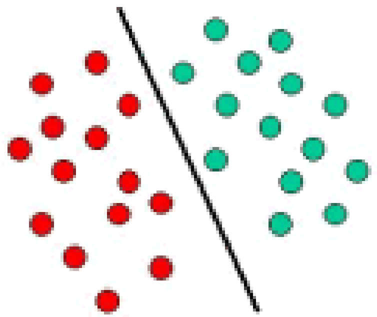
- In practice: Combine both strategies!

A Typical Supervised Machine Learning Pipeline



Classification

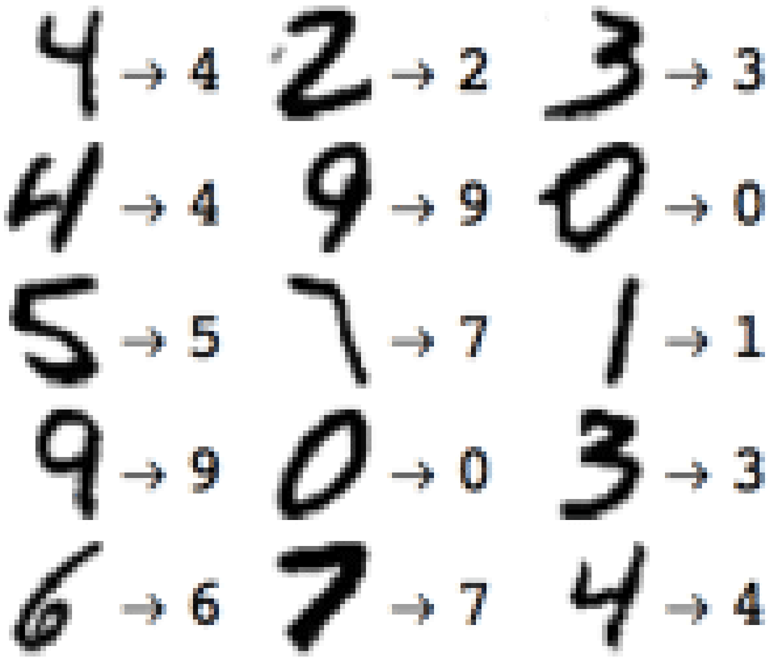
- Assume that labels are qualitative.
  - Labels indicate that points belong to a certain class.



- The goal of classification is to find a *predictor function* ("classifier") that can predict the class (label) for unseen data points.
  - Binary classifier: Separates two classes (Usual case).
  - Multi-class classifier: Separates between multiple classes.

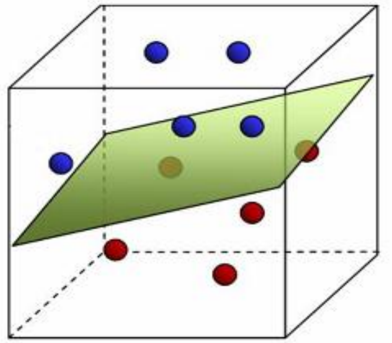
- **Examples:**
  - Optical Character Recognition: Identify which character a given image represents.
  - Medicine: Automatic analysis of medical samples to diagnose illnesses.
  - Video Analysis: Categorize videos according to their genre.
  - Security: Predict whether a given entity has malicious intent.

Classification Application: Handwritten Digits



Hyperplane Classifiers.

- Wide class of binary classification methods:
  - Typical assumption: Binary labels from $\{0,1\}$ (“positive” & “negative” class).
  - Idea: Find a separating Hyperplane („decision boundary “) between the two classes:
    - Hyperplane: Linear structure that separates a d-dimensional space into two half-spaces. (1D Data → Point, 2D Data → Line, 3D Data → Plane).
    - Data point lies left of the Hyperplane: Assign label 0. Data point lies right of the Hyperplane: Assign label 1.
  - In a d-dimensional space, a hyperplane is the set of points fulfilling the following equation:
    - $w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_d \cdot x_d - w_0 = 0$
    - Parameter vector: $\vec{\theta} = [w_0, w_1, \dots, w_d]^T$ („Normal vector“).



  - Since $\vec{\theta}$ is a normal vector of the hyperplane, the sign of $\vec{\theta}^T \vec{x}$ tells us on which side of the hyperplane a point lies: Negative - left, positive – right.

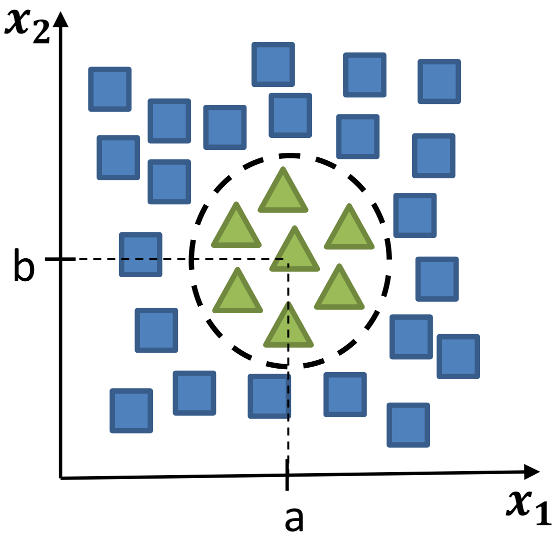
Non-Linear Hyperplanes

- As with linear regression, hyperplane classifiers are not limited to learning simple, planar decision boundaries.
  - Same idea as before: Add non-linear features to increase model complexity.

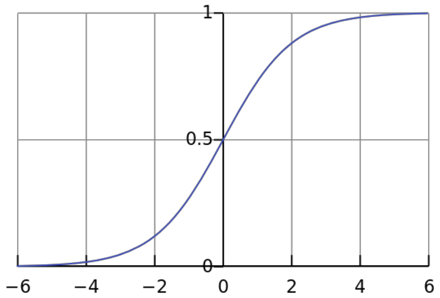
- Example: Enclosed data.
  - Decision boundary is a circle.
    - ➔ Cannot be represented as a 2D-hyperplane.
  - However, if we project our data as follows …

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_2^2 \end{bmatrix}^T$$

  - … our model can express “circular” hyperplanes!
    - Circle equation: $(x_1 - a)^2 + (x_2 - b)^2 = r^2$



- Widely used hyperplane classifier:
  - Assigns $\vec{x}$: To class 0 if: $h(\vec{\theta}^T\vec{x}) < 0.5$. To class 1 if: $h(\vec{\theta}^T\vec{x}) > 0.5$.
  - Where $h(x) = \frac{1}{1+e^{-x}}$ is the so-called sigmoid (or logistic) function.
    - Interpretation: $h(\vec{\theta}^T\vec{x})$ is the probability that $\vec{x}$ belongs to class 1.
    - Allows to assign a confidence to the classification.



- Training: Pick the hyperplane $\hat{\theta}$ that maximizes classification confidence.

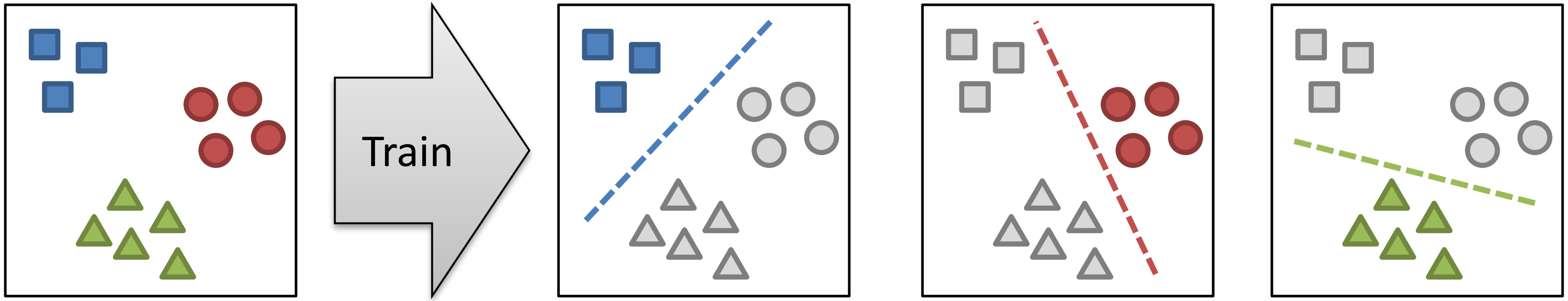
… for training points with y=1. *… for training points with y=0.*

$$\hat{\theta} = \operatorname{argmin}_{\vec{\theta}} -\frac{1}{n}\sum_{i=1}^{n}\left[\underbrace{y^{(i)} \cdot \log\left(\mathrm{h}\left(\vec{\theta}^T \vec{x}^{(i)}\right)\right)} + \left(1 - y^{(i)}\right) \cdot \log\left(1 - \mathrm{h}\left(\vec{\theta}^T \vec{x}^{(i)}\right)\right)\right]$$

fully correct classification (confidence = 1.0) → $-\infty$
unclear classification (confidence = 0.5) → ~ 0.3
fully incorrect classification (confidence = 0.0) → 0

  - No closed-form solution, but can be solved by (gradient-based) numeric solvers.

- Most classification methods only support binary classification.
  - Luckily, we can easily turn any binary classifier into a multi-class one.
- One-vs-all Classification:
  - For a dataset with k classes, train k classifiers.
  - Each classifier trains one class against all other classes.



  - To classify a data point, simply run all k classifier and assign the class of the one that gives the most confident positive assignment.

Classifier Evaluation – Accuracy.

- Straightforward evaluation metric: Classifier Accuracy.
  - Fraction of correct classifications in the test set:

$$Accuracy = \frac{\#\ Correctly\ predicted\ points}{\#\ Points\ in\ test\ set}$$

- Problematic metric if the label distribution is skewed!
  - **Example:** Classifier to identify whether a patient has cancer.
    - Luckily, cancer is rare (Skewed distribution).
    - → Test set contains 996 cancer-free patients and 4 cancer patients.
  - **Stupid Classifier:** Always returns „No Cancer“:
    - Will classify all 1,000 patients in the test set as „No Cancer“ → Correct for 996 patients.
    - → Accuracy is 99,6% … even though the classifier is completely useless (and dangerous)!
- → You should (typically) avoid reporting classification accuracy!

Classifier Evaluation – Precision & Recall.

- Compute the "Confusion Matrix" for the test set (count for each field):

		Actual Label 0	*Actual Label* 1
Predicted Label	0	True Negative (TN)	False Negative (FN)
	1	False Positive (FP)	True Positive (TP)

- Based on this, we can compute more meaningful quality metrics:
  - Precision: $\frac{TP}{TP+FP}$ *„How often was a predicted 1-label actually correct?"*
  - Recall: $\frac{TP}{TP+FN}$ *„What fraction of test data points with a 1-label was discovered?"*

- Good classifiers have to offer both high precision & high recall.
  - A good evaluation metric is the F1-Score: $2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$

Course Overview

1. What is Machine Learning?
2. Supervised Learning
3. **Unsupervised Learning**

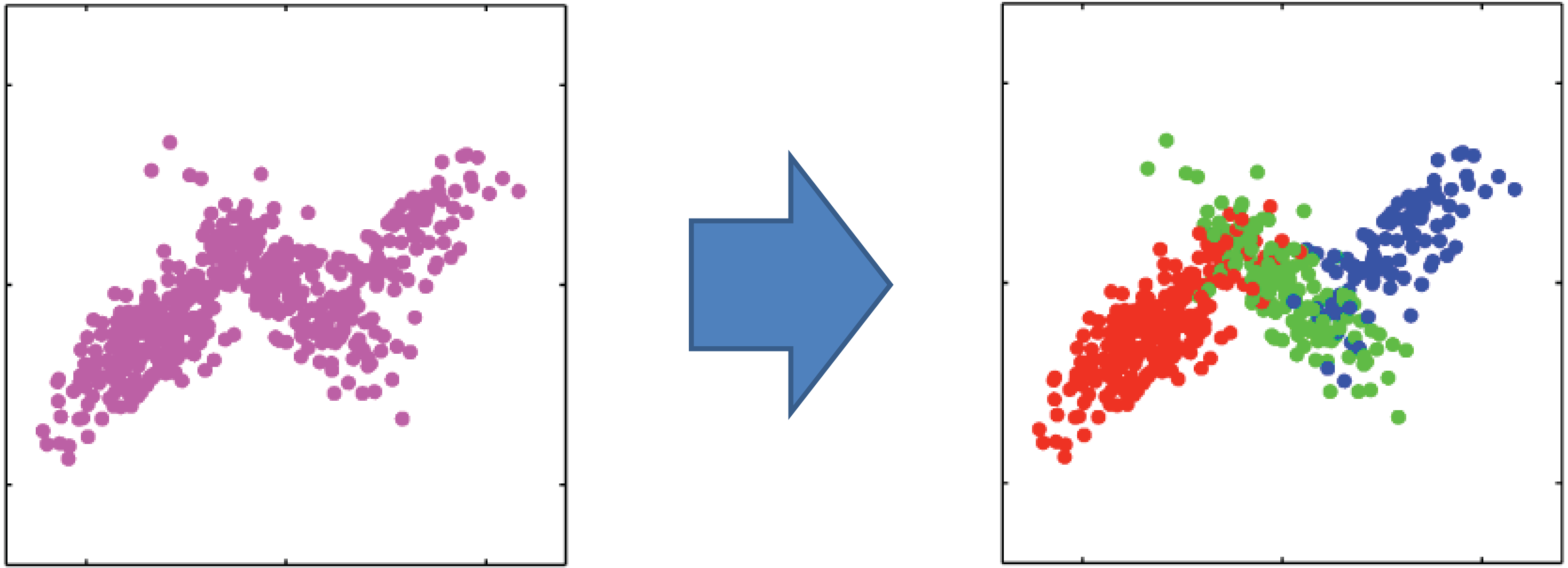
Clustering

- Problem Definition:
  - Given a set of points, with a notion of distance between points, partition the points into some number of clusters, so that:
    - Members of a cluster are close/similar to each other.
    - Members of different clusters are dissimilar.

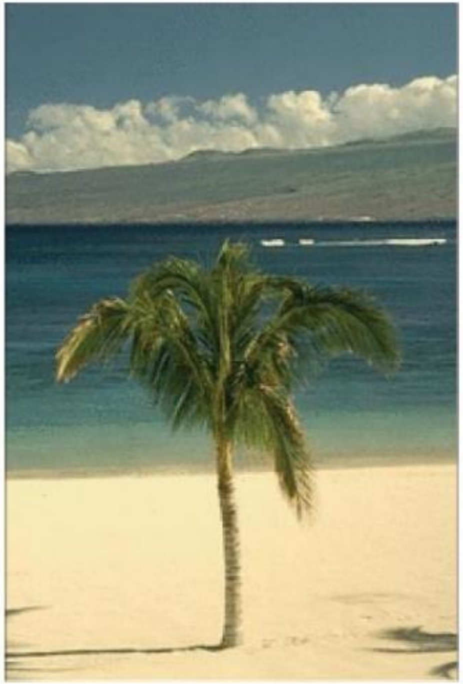
- Data is typically assumed to be numeric vectors, i.e. from $\mathbb{R}^d$.
  - If not: Feature extraction or custom distance metric.

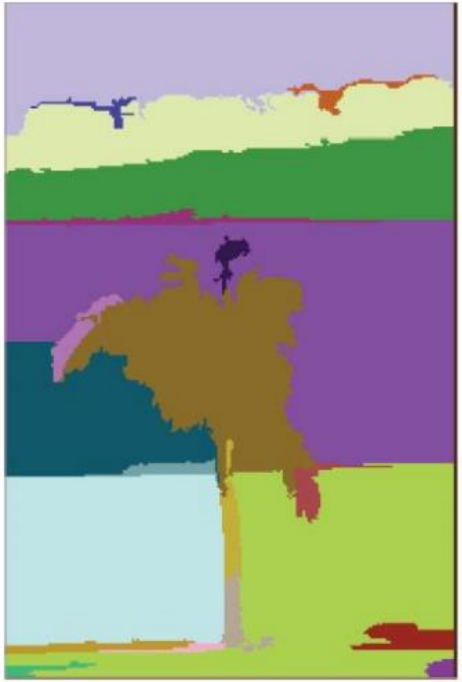
- Applications:
  - Visualize / Discover the internal structure of the data.
  - Preprocessing step for other methods (e.g. select good features for a classifier).
  - Actual Data Analysis Task (e.g.: Detect Market Segments, Compute Ideal Antenna Placements, Identify Similar Documents or Articles, …).

Clustering



Clustering Application: Image Segmentation.





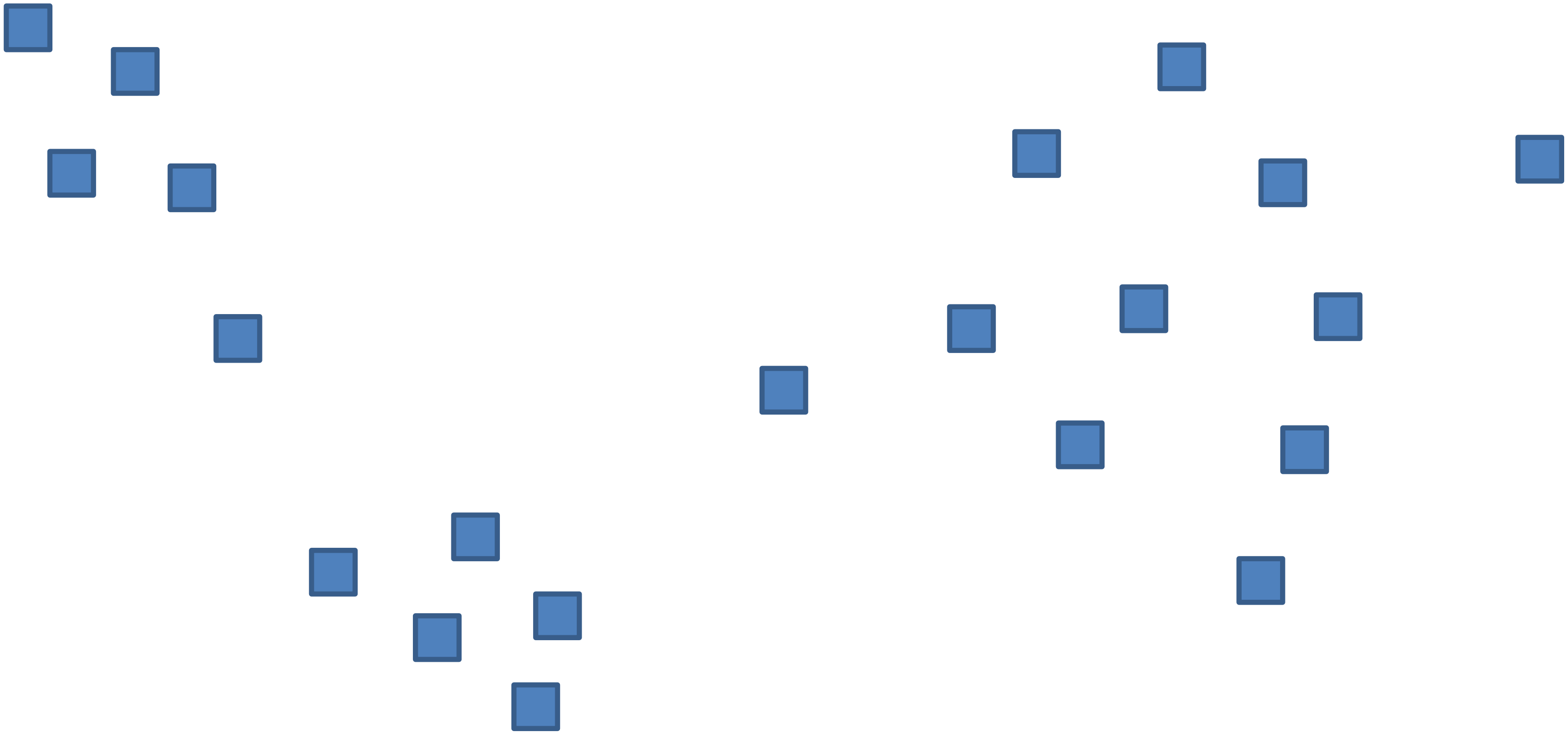
K-Means Clustering

- Iterative algorithm to partition a set of (numeric) data points $\mathrm{D} = \{\vec{x}_1, \dots, \vec{x}_n\}$ into a pre-defined number (k) of clusters $\{S_1, \dots, S_k\}$.
  - Each Cluster S_i is defined as a subset of the points from D.
    - Clusters are disjoint, i.e. each point is assigned to exactly one cluster.
  - Each Cluster S_i has a centroid $\vec{\mu}_1$, which is the "average" of the points in the cluster.
  - The parameter vector $\vec{\theta}$ of a K-Means model are the cluster assignments.

- Goal is to find cluster assignments that minimize the total distance of the data points to their nearest cluster centroid:
  - $\vec{\theta} = argmin_{S_1,\dots,S_k} \sum_{i=1}^{k} \sum_{\vec{x} \in S_i} \|\vec{x} - \mu_i\|^2$

- K-Means does not have a unique solution, there are local minima!
  - Different initialization strategies lead to different clustering results.

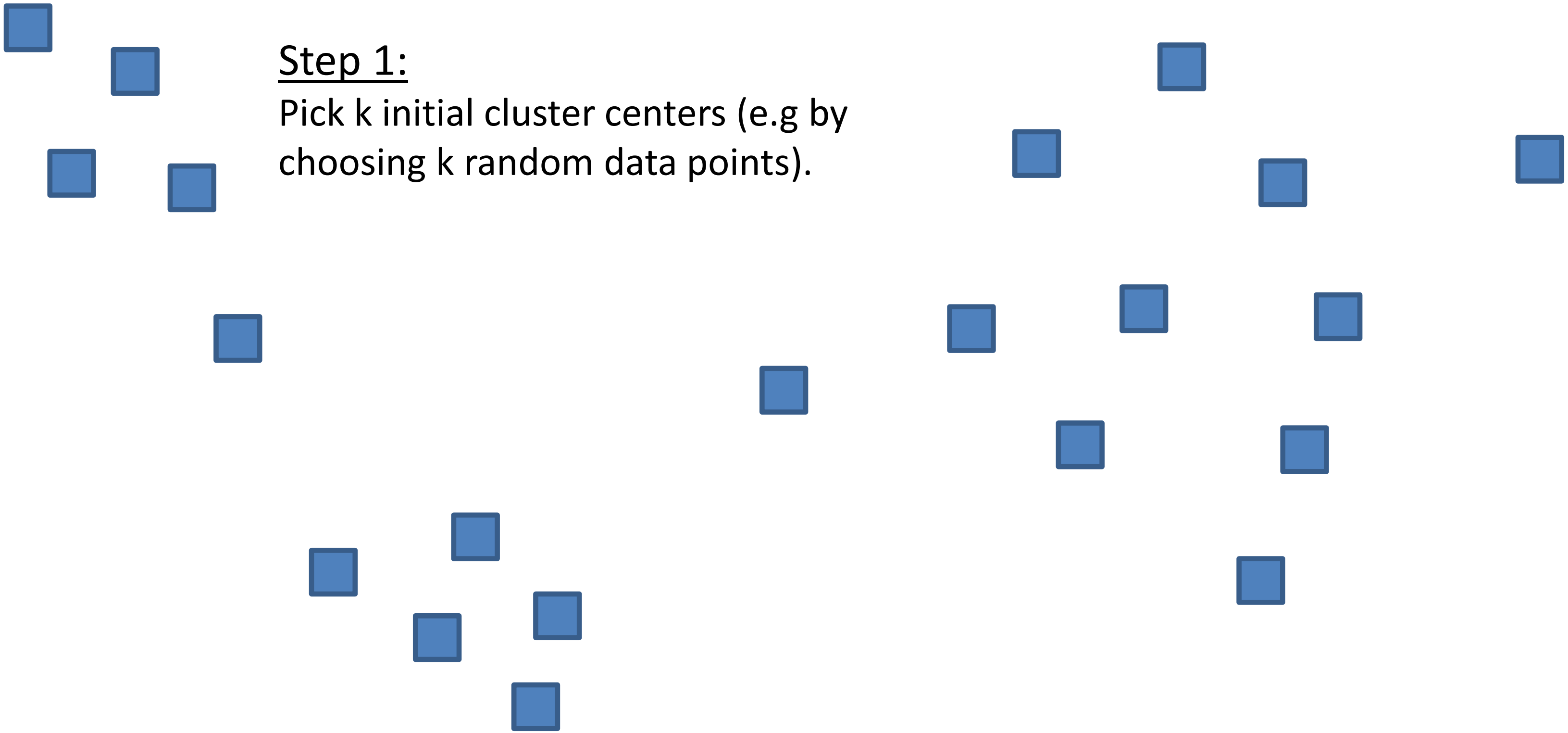
Example: K-Means Clustering (k=3)



Example: K-Means Clustering (k=3)

<u>Step 1:</u>

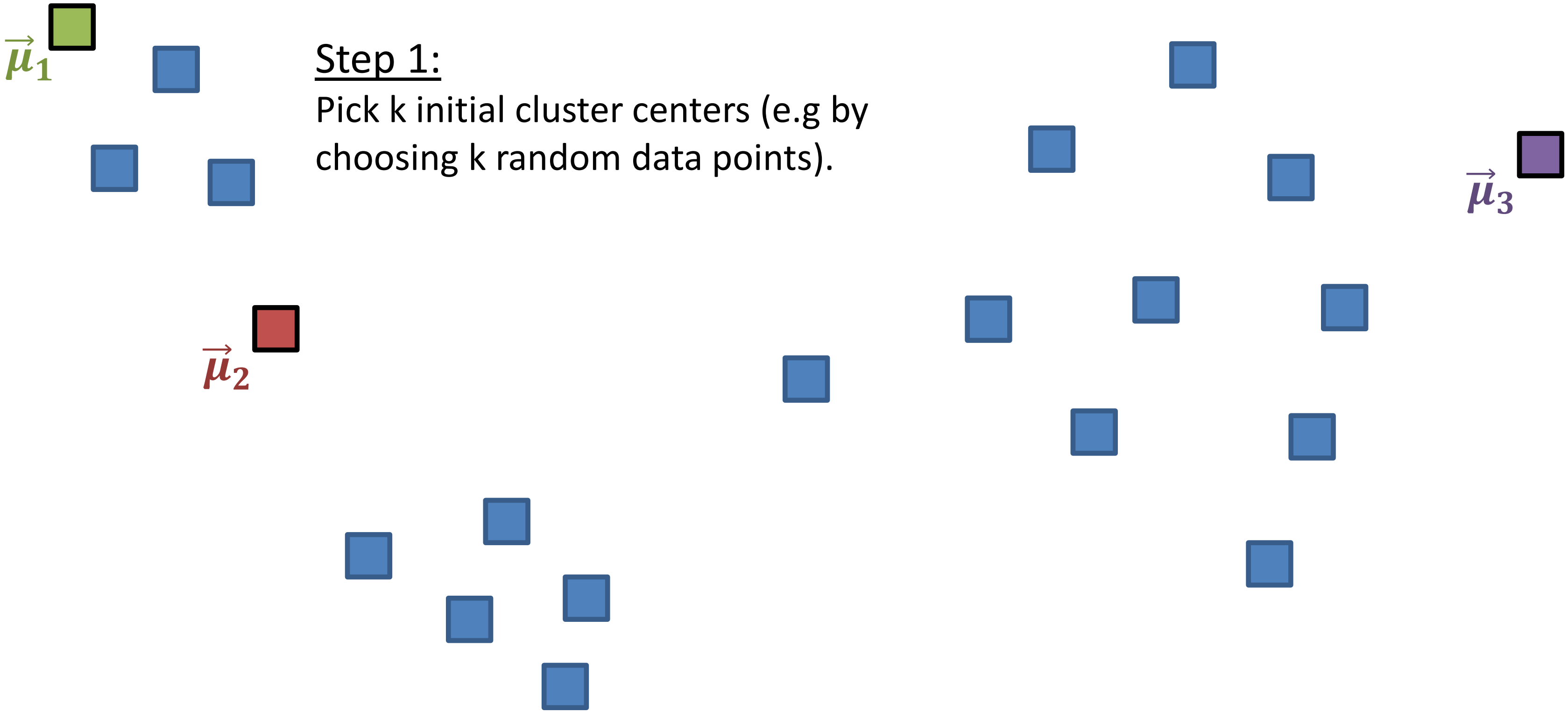
Pick k initial cluster centers (e.g by choosing k random data points).



Example: K-Means Clustering (k=3)

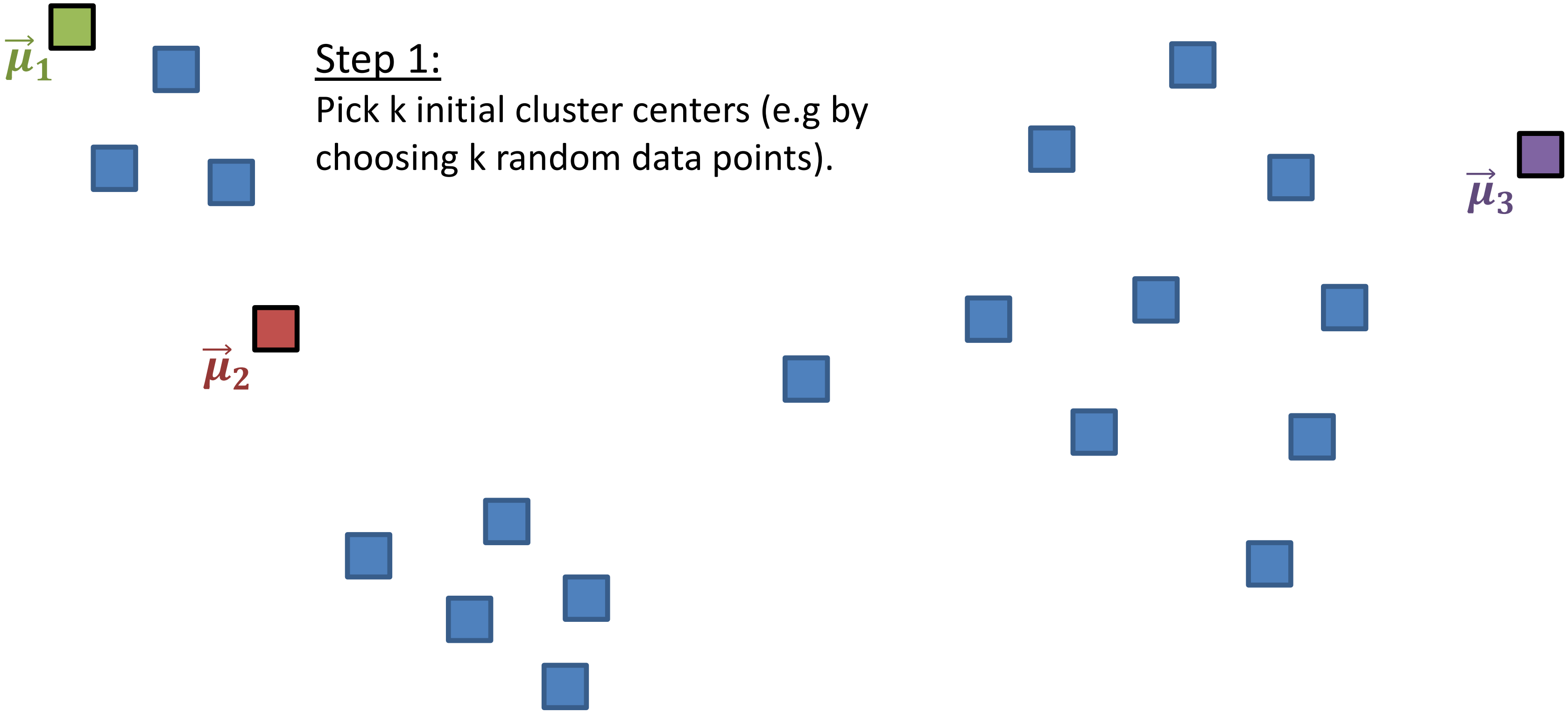
<u>Step 1:</u>

Pick k initial cluster centers (e.g by choosing k random data points).



Step 1:

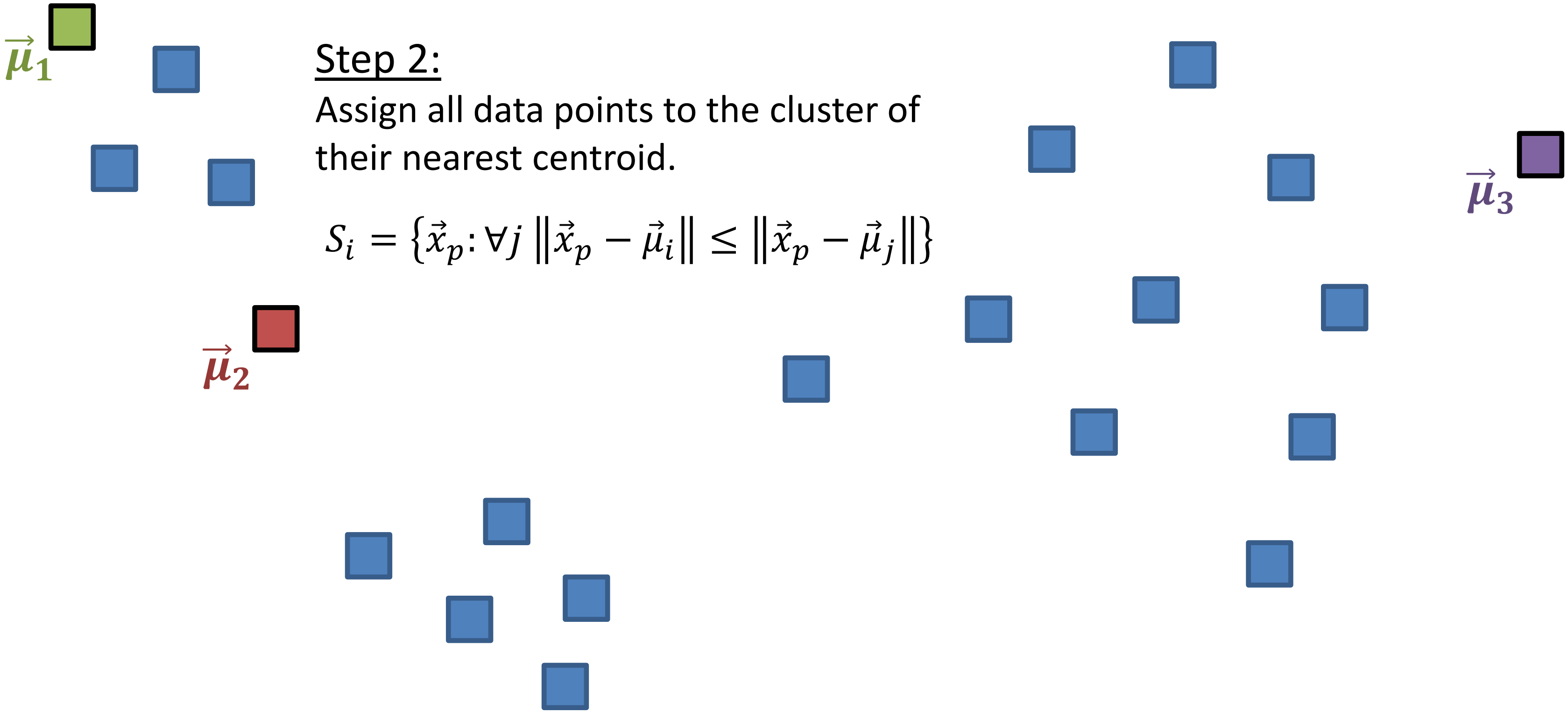
Pick k initial cluster centers (e.g by choosing k random data points).



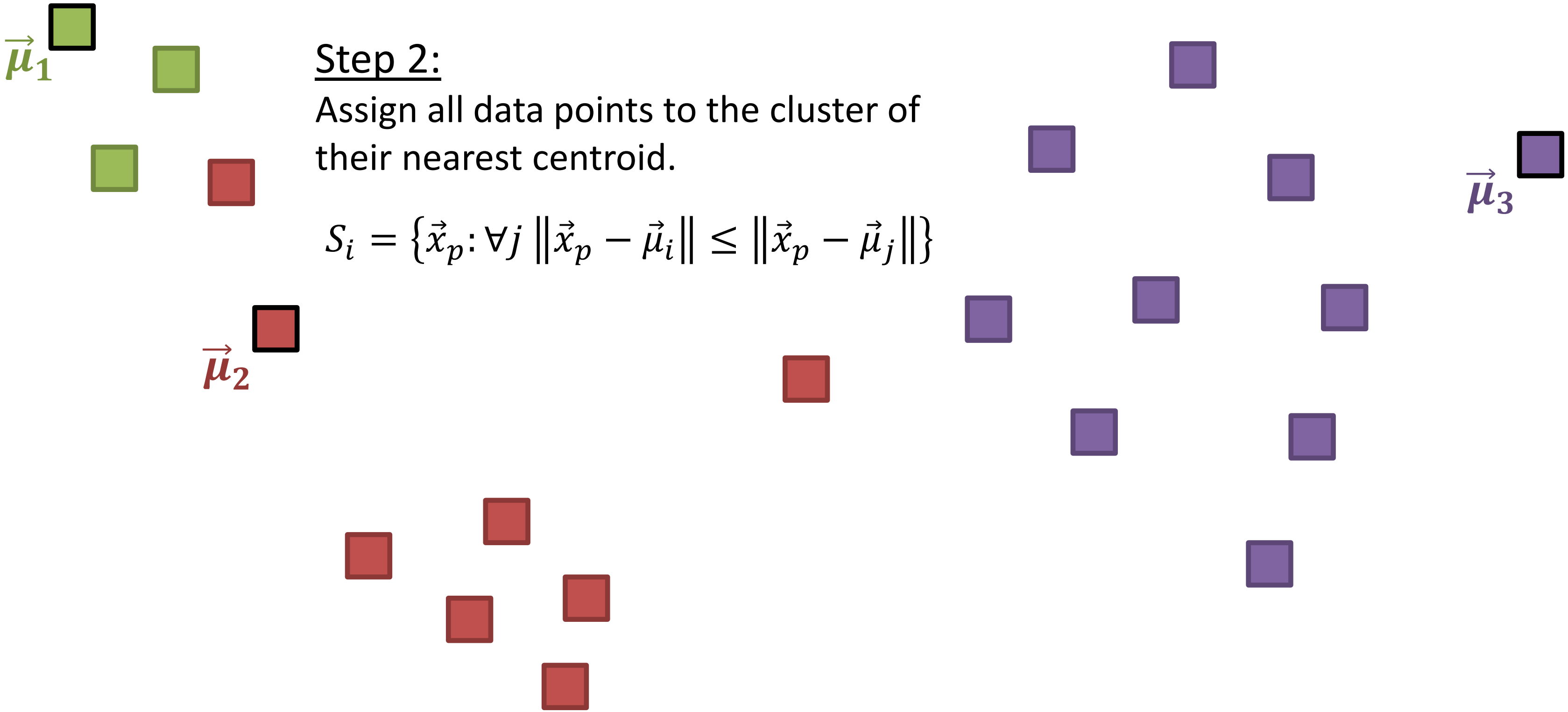
Example: K-Means Clustering (k=3)

<u>Step 2:</u>
Assign all data points to the cluster of their nearest centroid.

$$S_i = \{\vec{x}_p : \forall j \ \|\vec{x}_p - \vec{\mu}_i\| \leq \|\vec{x}_p - \vec{\mu}_j\|\}$$



Example: K-Means Clustering (k=3)



Step 2:

Assign all data points to the cluster of their nearest centroid.

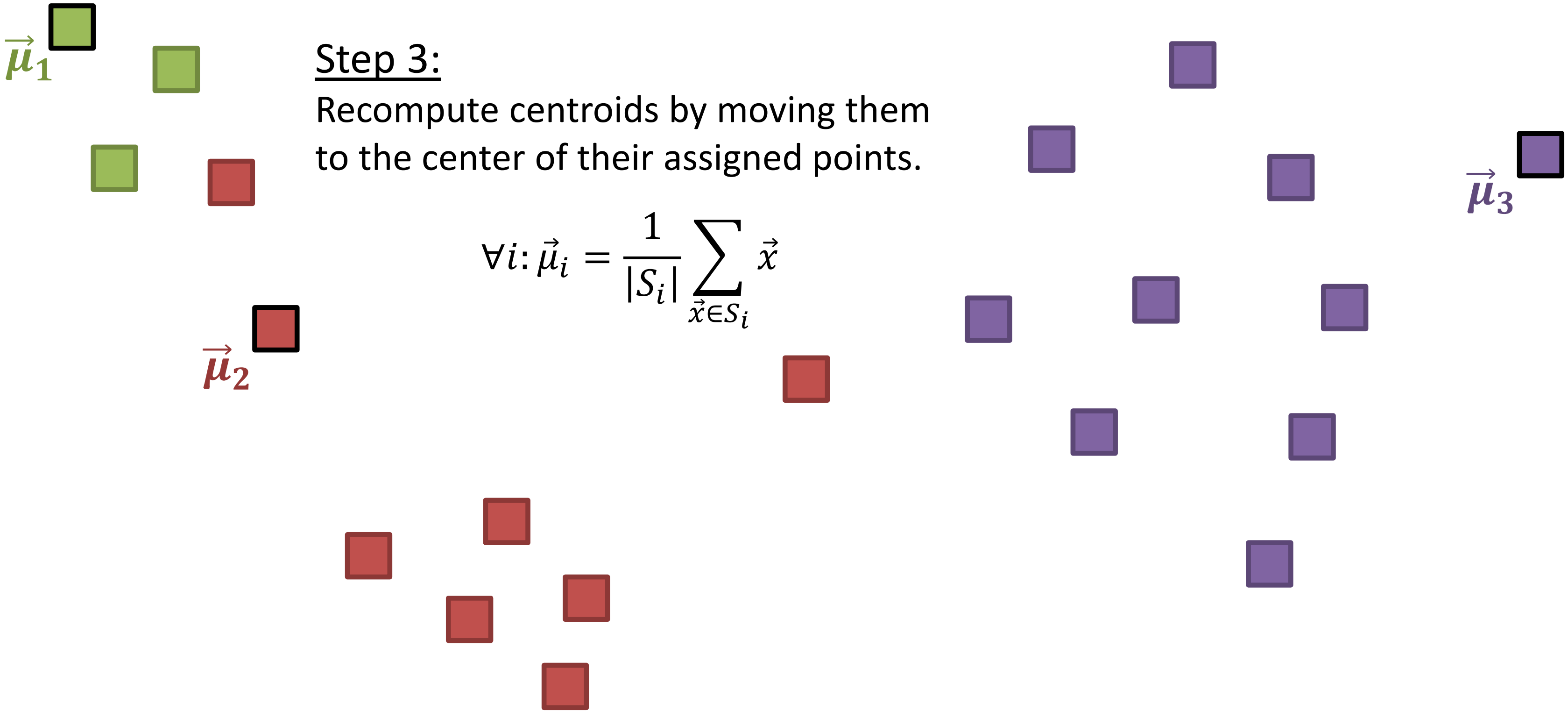
$$S_i = \{\vec{x}_p : \forall j \; \|\vec{x}_p - \vec{\mu}_i\| \leq \|\vec{x}_p - \vec{\mu}_j\|\}$$

Example: K-Means Clustering (k=3)

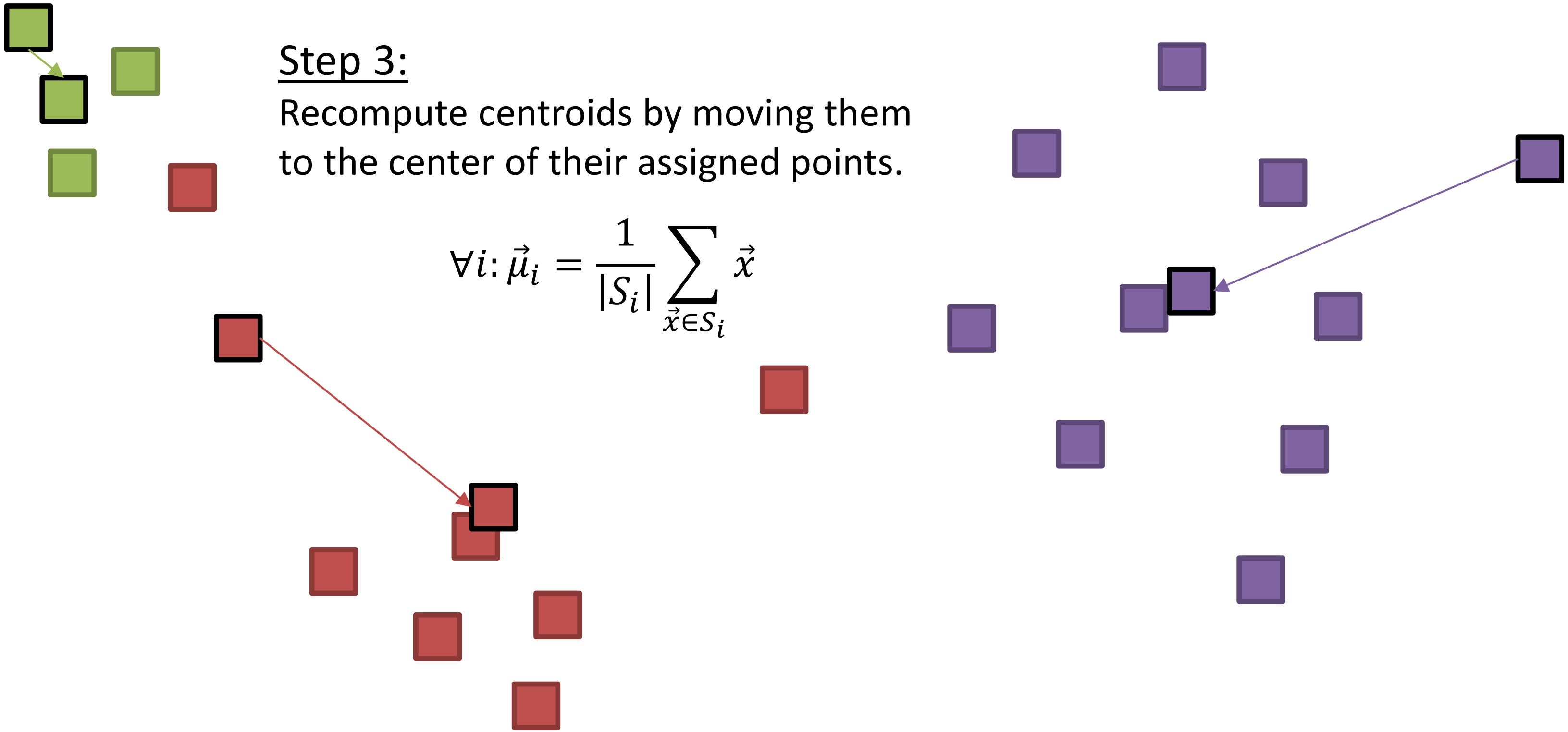
<u>Step 3:</u>

Recompute centroids by moving them to the center of their assigned points.

$$\forall i: \vec{\mu}_i = \frac{1}{|S_i|} \sum_{\vec{x} \in S_i} \vec{x}$$



Example: K-Means Clustering (k=3)

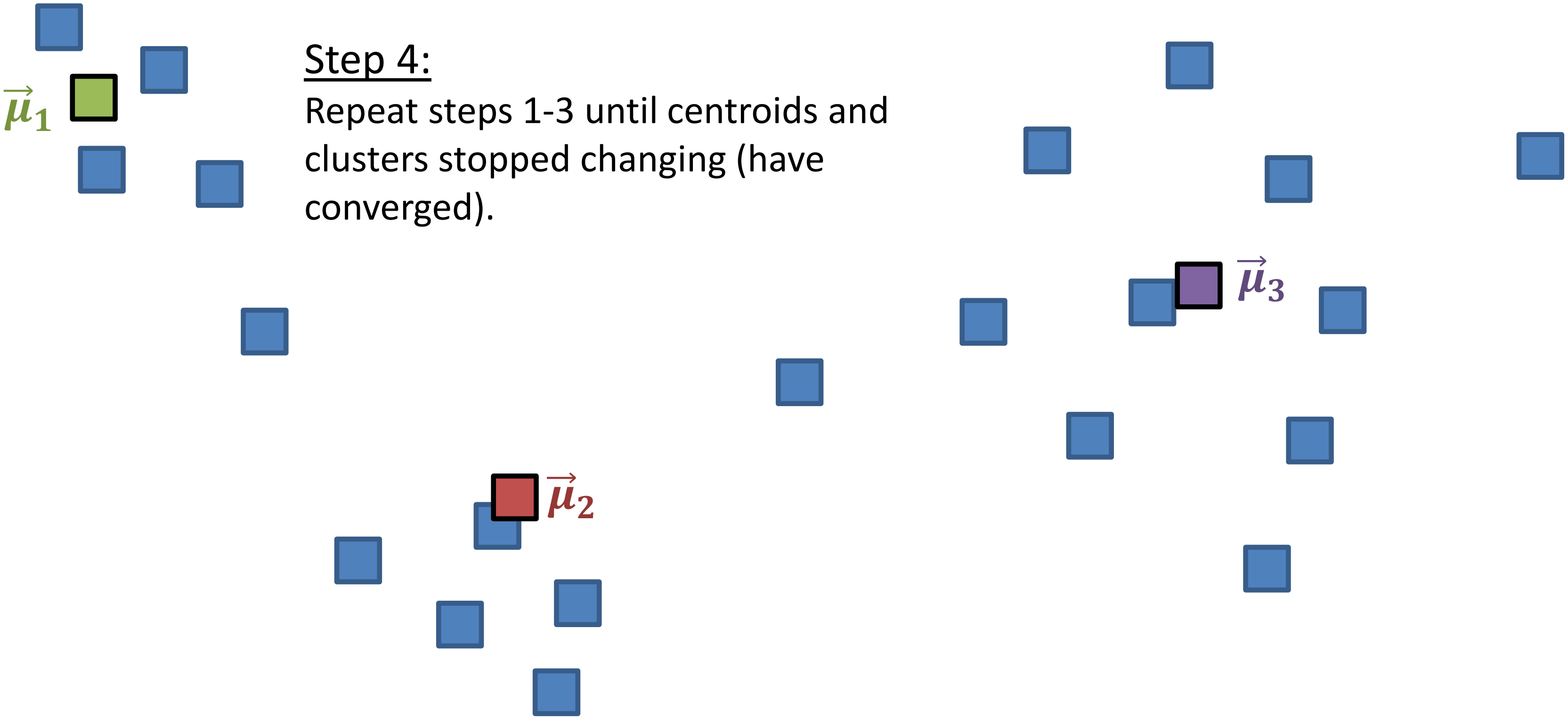


Step 3:

Recompute centroids by moving them to the center of their assigned points.

$$\forall i: \vec{\mu}_i = \frac{1}{|S_i|} \sum_{\vec{x} \in S_i} \vec{x}$$

Example: K-Means Clustering (k=3)



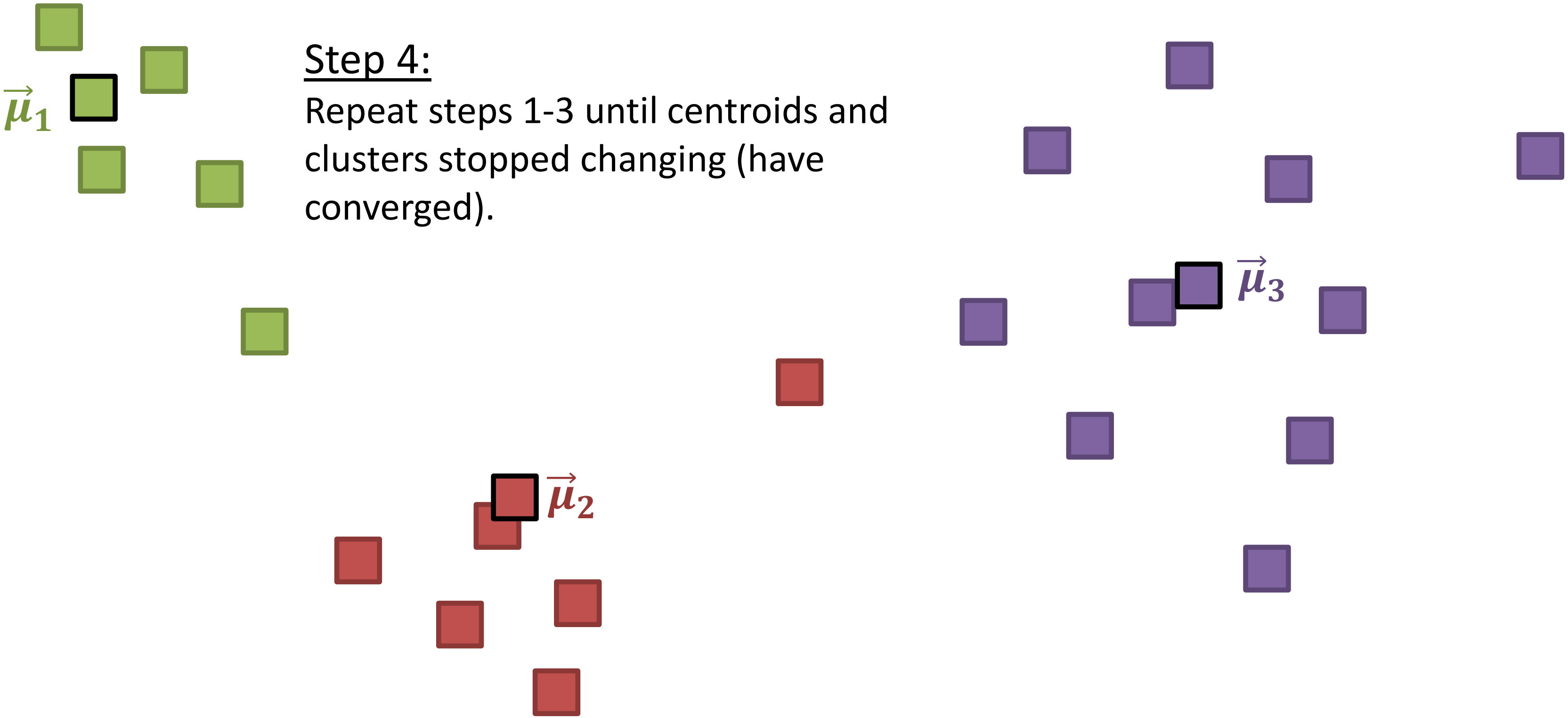
Step 4:

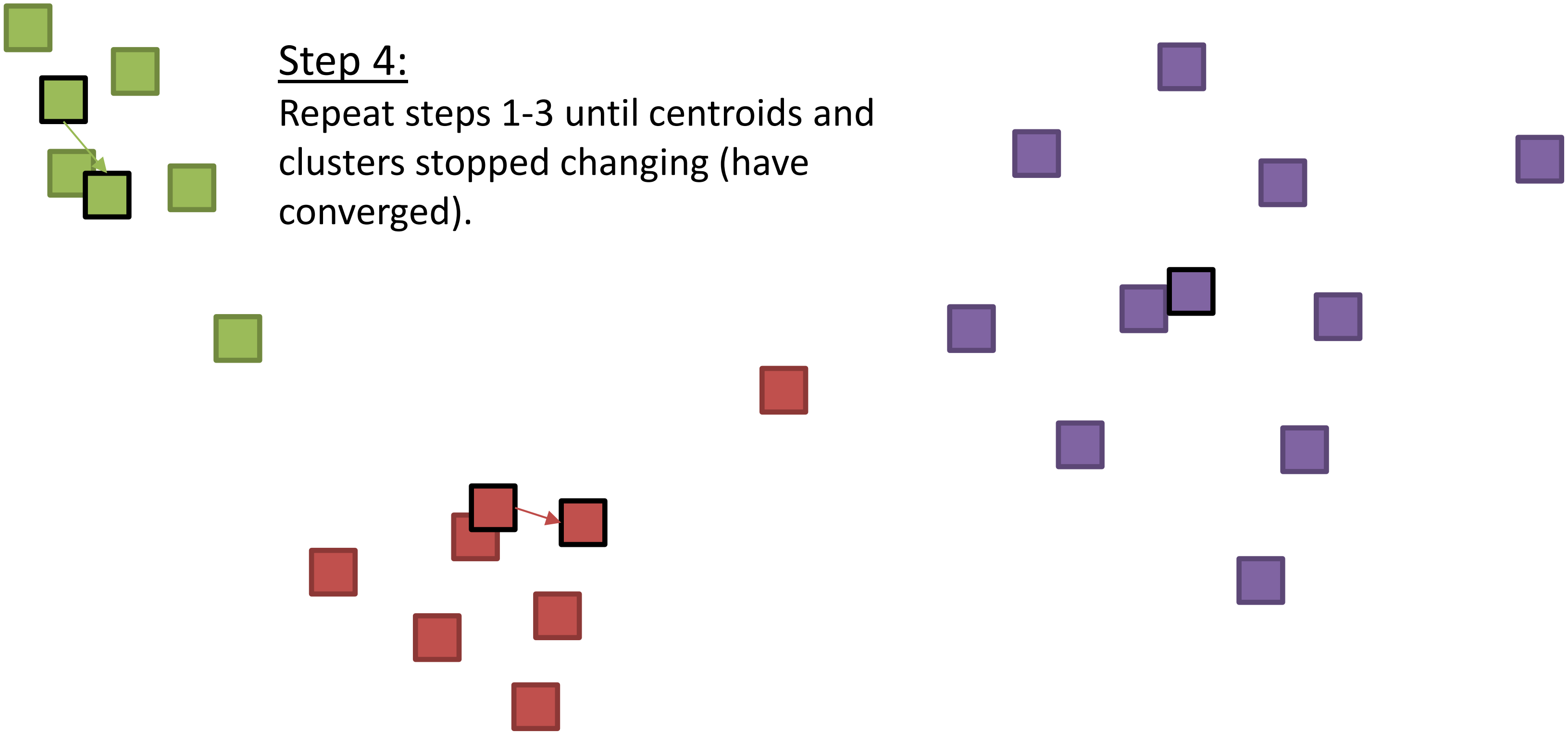
Repeat steps 1-3 until centroids and clusters stopped changing (have converged).

Example: K-Means Clustering (k=3)

<u>Step 4:</u>

Repeat steps 1-3 until centroids and clusters stopped changing (have converged).





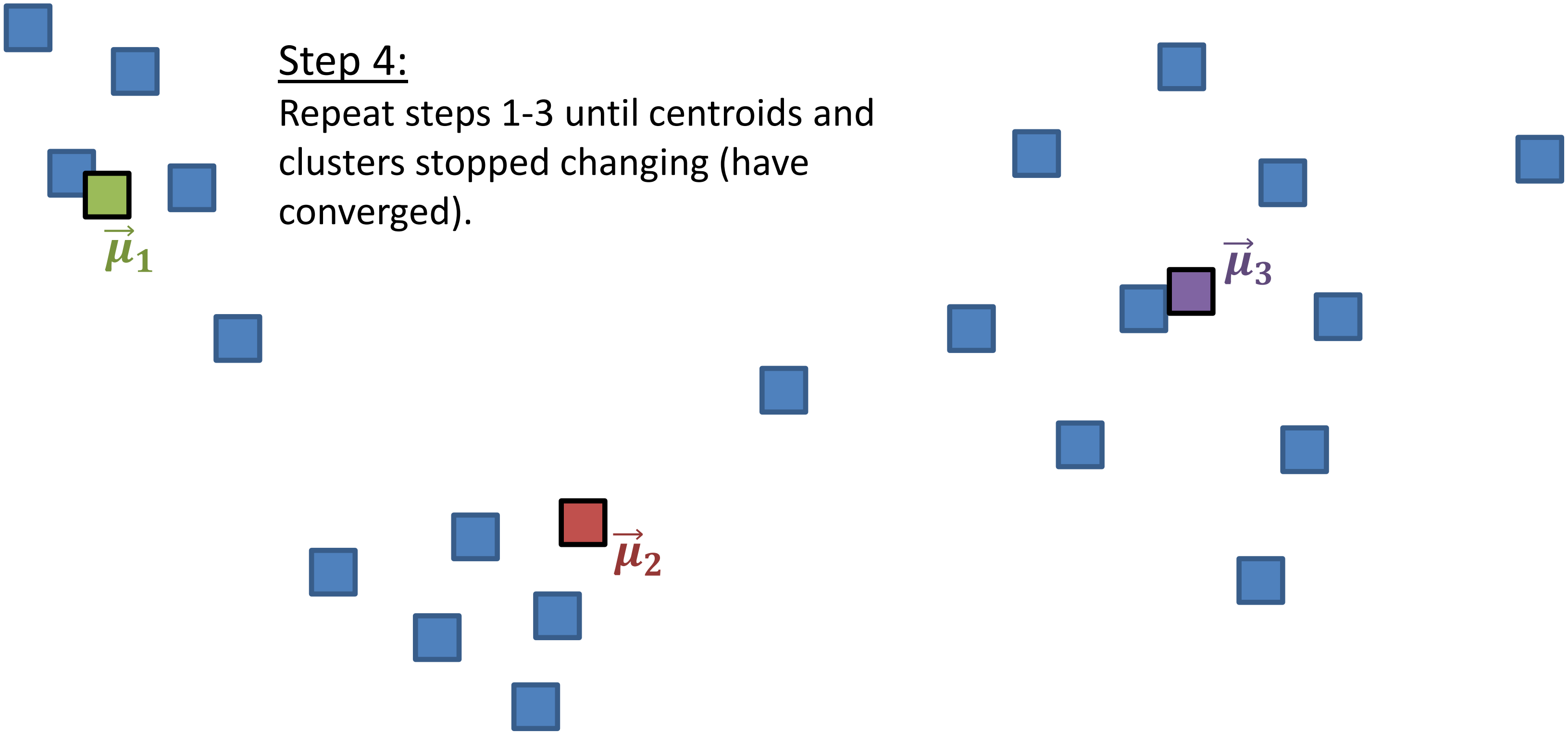
Step 4:

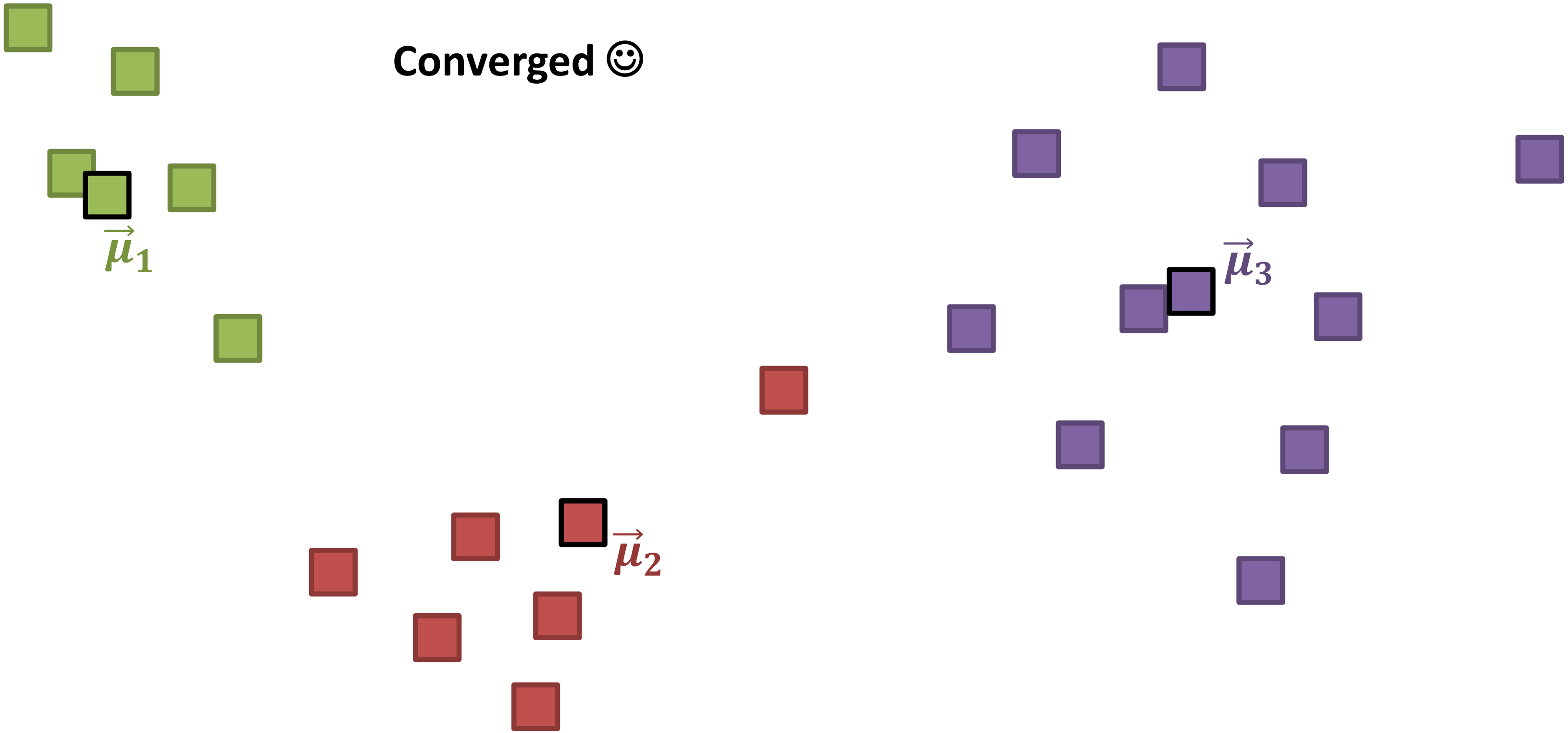
Repeat steps 1-3 until centroids and clusters stopped changing (have converged).

Example: K-Means Clustering (k=3)

<u>Step 4:</u>

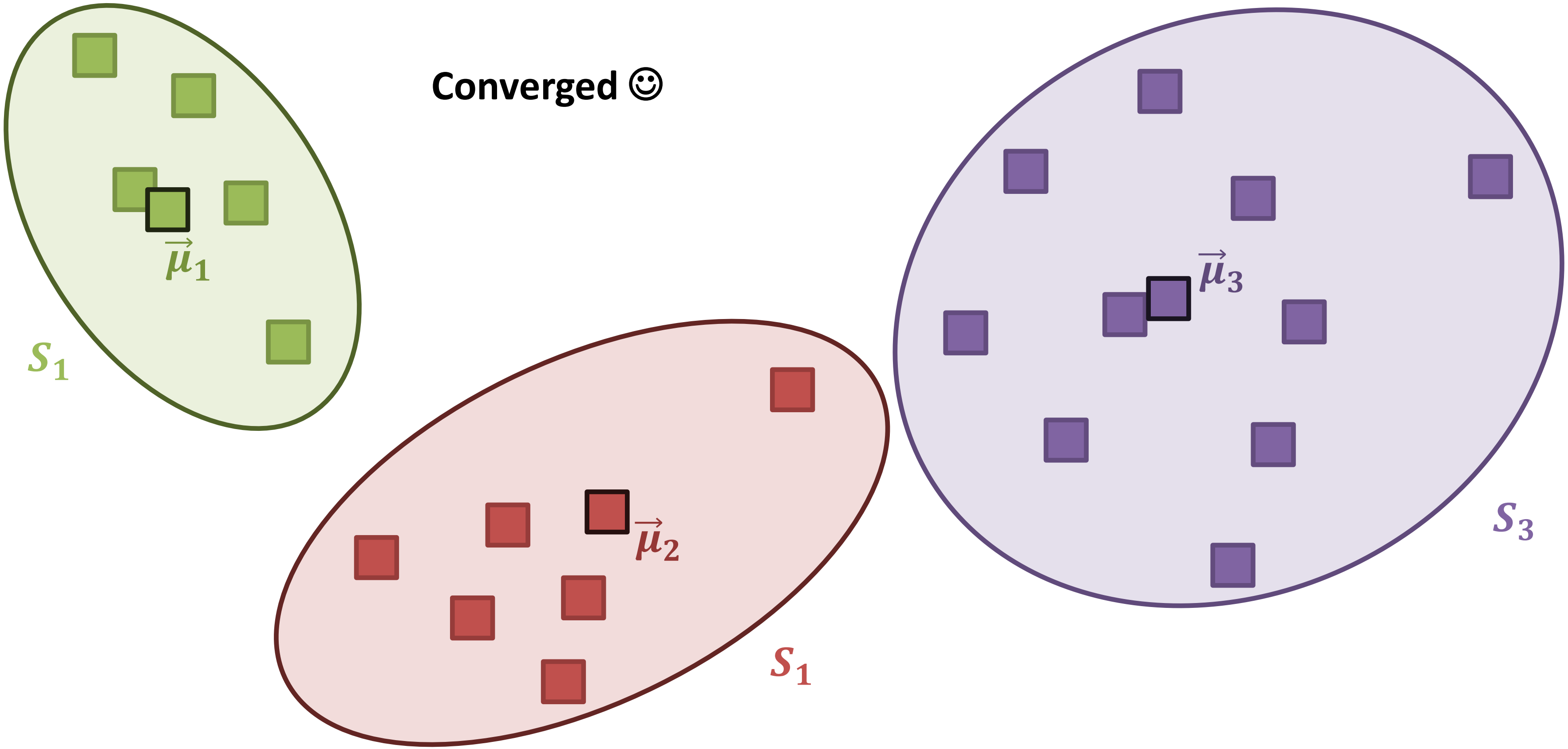
Repeat steps 1-3 until centroids and clusters stopped changing (have converged).





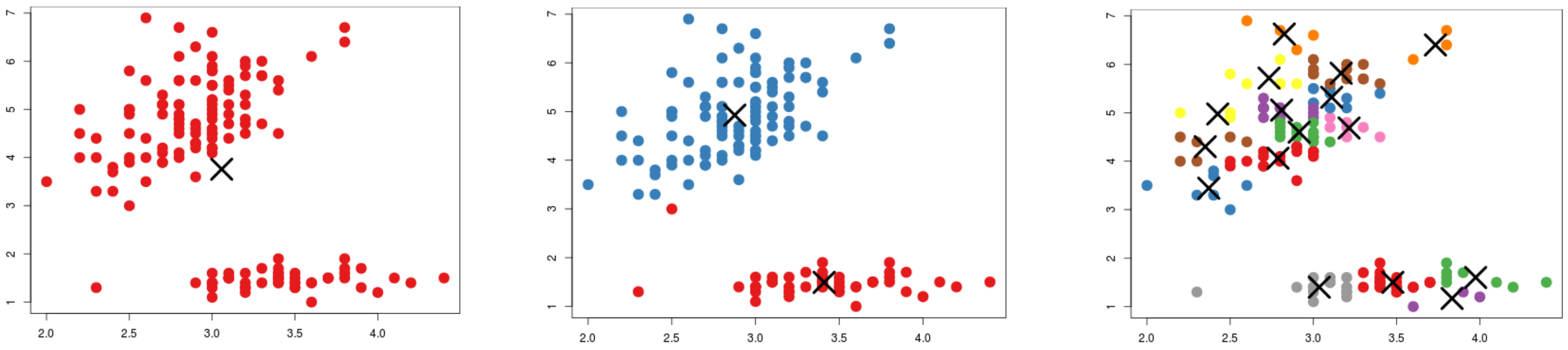
Converged ☺
$\vec{\mu}_1$
$\vec{\mu}_3$
$\vec{\mu}_2$

Example: K-Means Clustering (k=3)



How to choose k? (I)

- K-Means requires the user to provide the number of clusters.
  - Picking k has a huge impact on the clustering quality
  - Number too low: Underfitting.   Number too high: Overfitting.



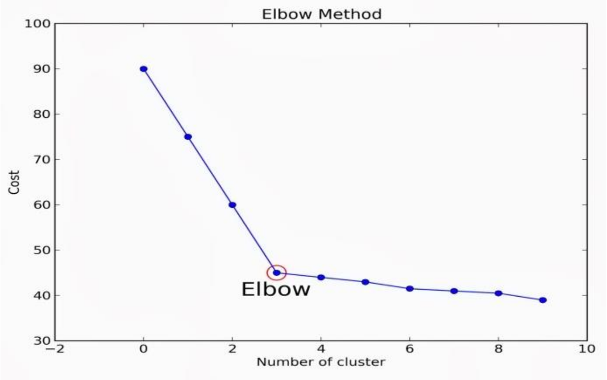
k = 1   k = 2   k = 16

- Problem: Unsupervised learning, we can’t use the test error.
  - ➔ No straight-forward way to pick k algorithmically.

How to choose k? (II)

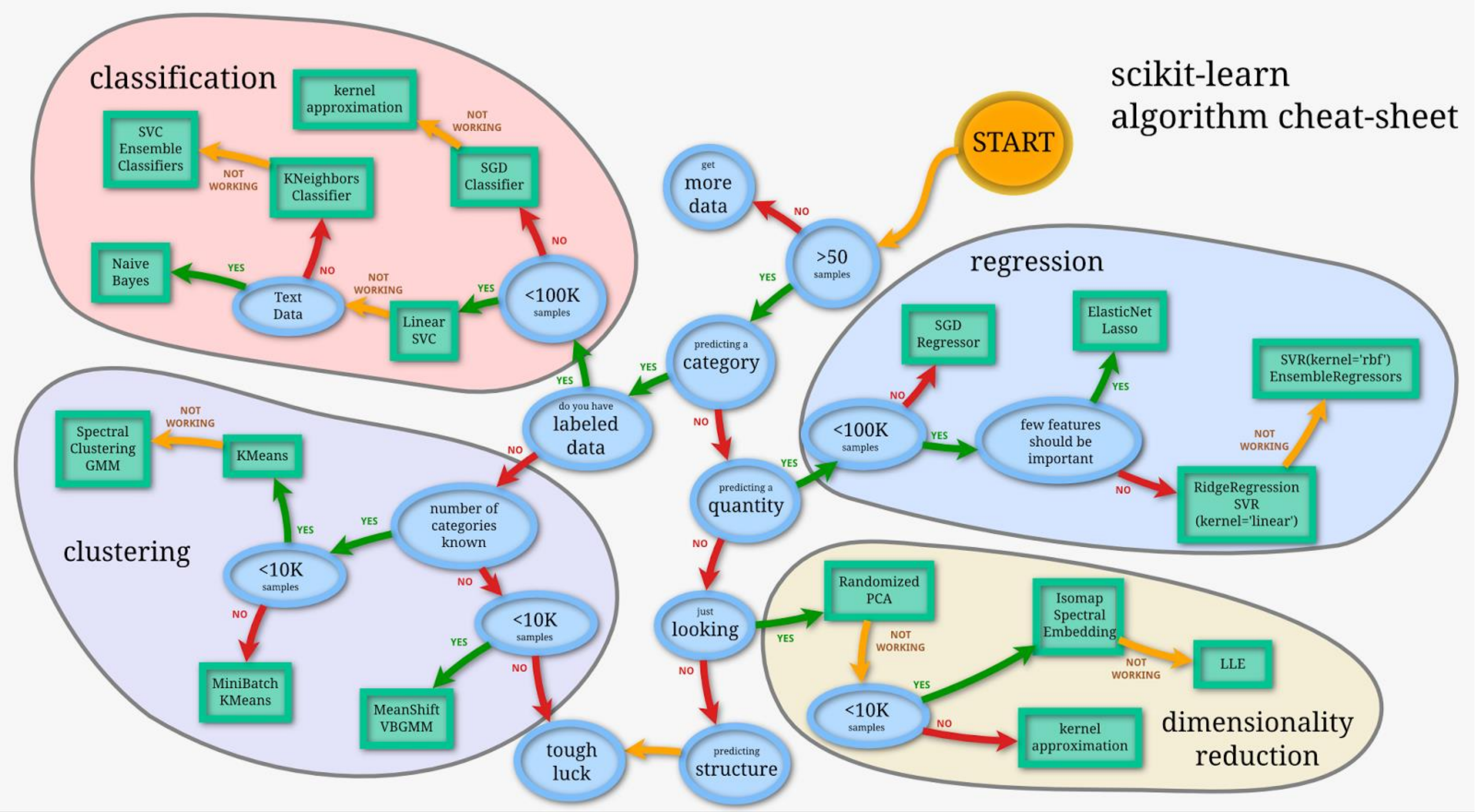
1. If you have prior knowledge, use it!
   - Sometimes, we know the inherent structure of the data (e.g.: Clustering two groups of persons in a social experiment), allowing us to plug in the actual value of k.

2. Use the “elbow”-method:
   - Run K-Means for multiple values of k, plotting the achieved cost for each clustering.
   - Then pick the “elbow”, i.e. the value of k at which the decrease in cost slows down.



3. Use the test error of dependent learning steps:
   - Clustering is often used as pre-processing for other supervised learning methods (e.g. to find discriminating features).
   - In this case, pick k based on how well it performs for the latter method.

This was just a very, very brief overview …



… so, if you want to learn more:

- Take one (or multiple) of the following Bachelor courses …
  - „Kognitive Algorithmen“ (Prof. Müller).
  - „Künstliche Intelligenz: Grundlagen & Anwendungen“ (Prof. Opper).
  - „Data Warehousing & Business Intelligence“ (Prof. Markl).

- … wait until you start your Master …
  - “Machine Learning” (Prof. Müller).
  - “Machine Intelligence” (Prof. Obermayer).
  - “Advanced Information Modelling 3” (Prof. Markl).

- … or take the (excellent) Online Course by Prof. Andrew Ng (Stanford):
  - https://www.coursera.org/learn/machine-learning/home/info

Outlook & Overview

- Today we discussed:
  - What is a Model?
  - What is Machine Learning?
  - What is the difference between Supervised & Unsupervised Learning?
  - What are Regression, Classification, Clustering & Association Rule Mining?
  - How do Linear & Logistic Regression work?
  - What to keep in mind when evaluating models?
  - How does K-Means Clustering work?

- Next week:
  - Feature Extraction – Analyzing qualitative data (text, images, videos).
  - Important tools for Data Scientists.