Basics of Machine Learning in Computer Vision

Kuk-Jin Yoon

Visual Intelligence Lab.
Department of Mechanical Engineering

KAIST

What do humans care about?

slide by Fei Fei, Fergus & Torralba

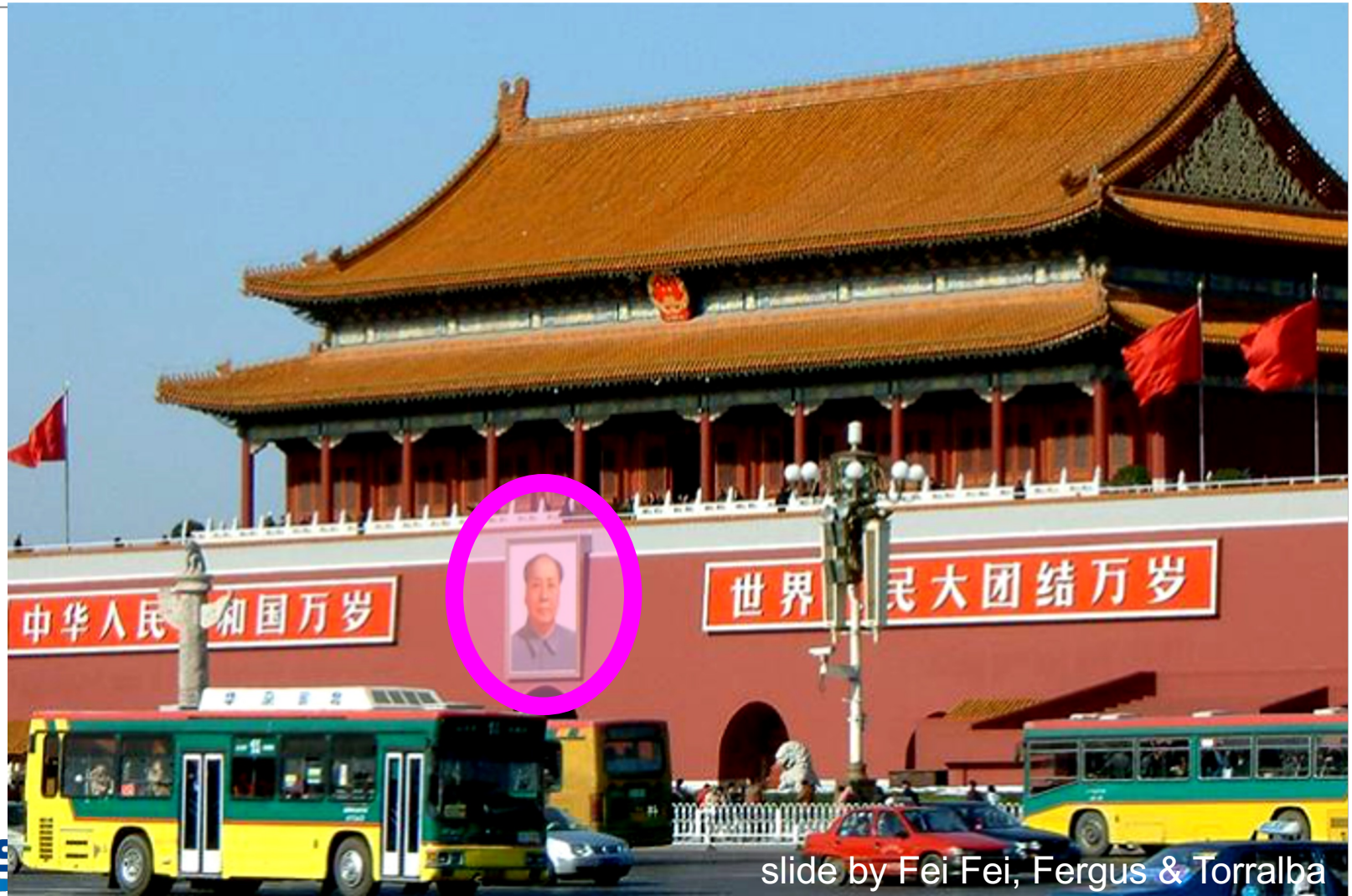
Verification: is that a bus?

slide by Fei Fei, Fergus & Torralba

Detection: are there cars?

slide by Fei Fei, Fergus & Torralba

Identification: is that a picture of Mao?

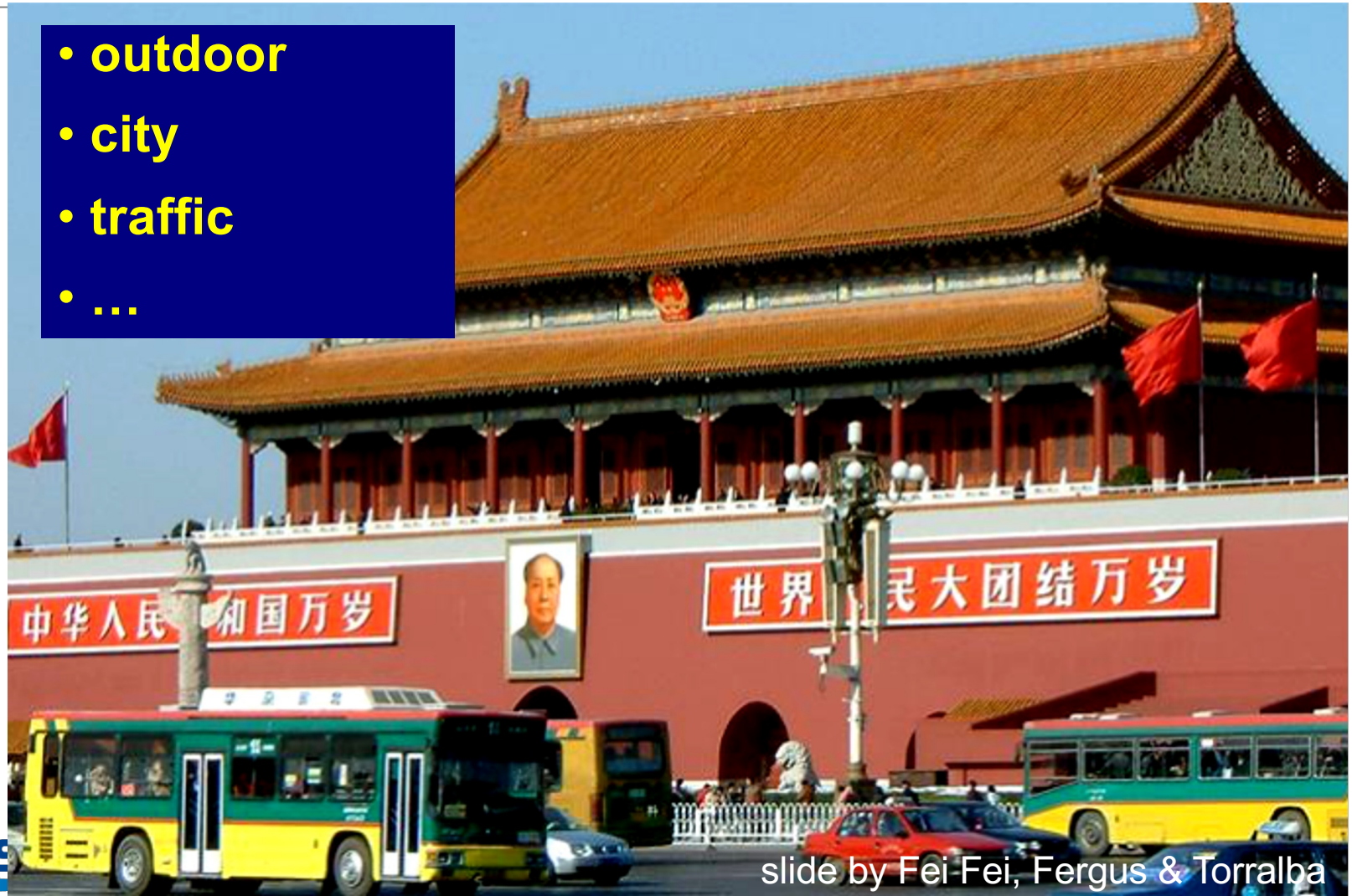


slide by Fei Fei, Fergus & Torralba

Object Categorization

Slide by Fei Fei, Fergus & Torralba

Scene and Context Categorization



slide by Fei Fei, Fergus & Torralba

Rough 3D Layout, Depth Ordering

slide by Fei Fei, Fergus & Torralba

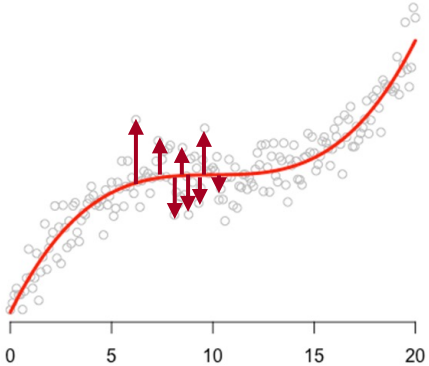
Why “Learn”?

- There is no need to “learn” to calculate payroll
- Learning is used when:
  - Human expertise does not exist (navigating on Mars),
  - Humans are unable to explain their expertise (speech recognition)
  - Solution changes in time (routing on a computer network)
  - Solution needs to be adapted to particular cases (user biometrics)

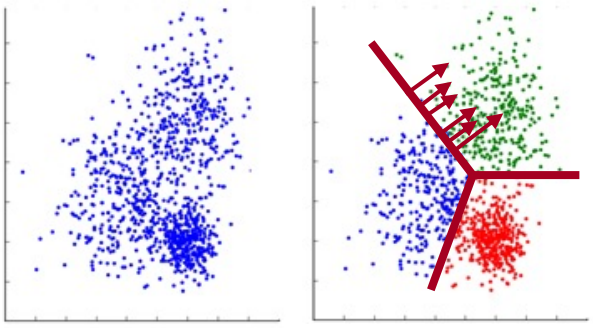
Machine Learning

- A computer program is said to learn from experience E with respect to some class of task T and performance measure P, if its performance at task in T, as measured by P, improves with experience E. (Mitchell, 1997)
- Task : classification, regression, clustering
- Performance (loss function): errors, distance
- Experience: data (labeled, unlabeled)

Rate of correct answers



Distance(=regression error) from model ($y = f(x^3)$)



Distance from lines (**Supporting Vector Machine**)

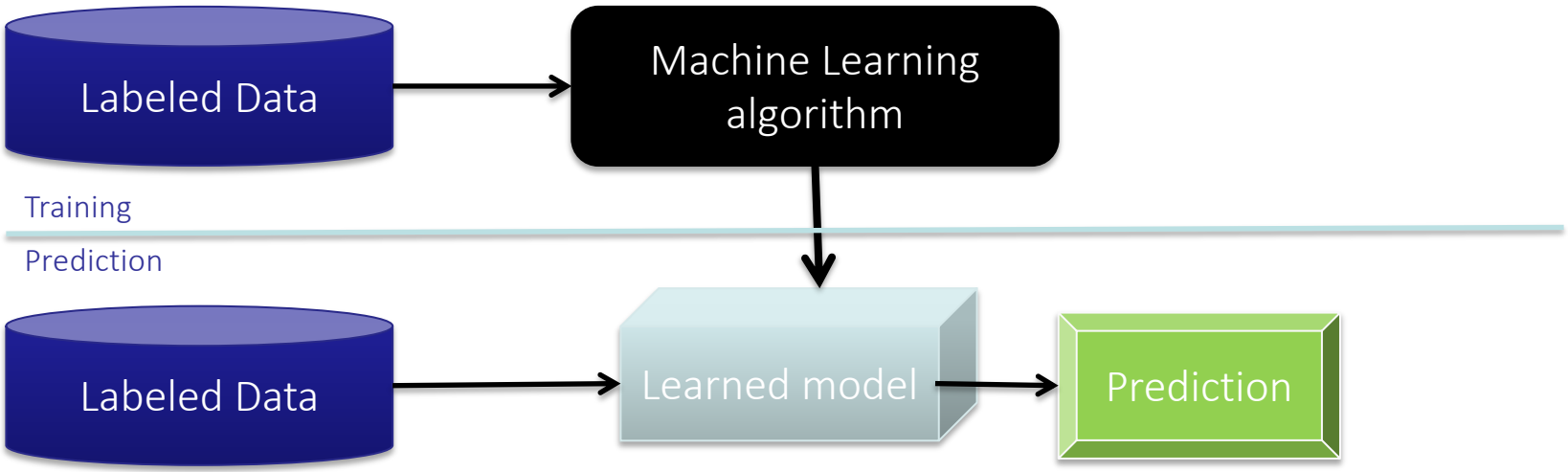
KAIST https://www.youtube.com/watch?v=j3za7nv7RfI, images from google

What is Machine Learning?

- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
  - Study of algorithms that
    - improve their performance
      - at some task
        - » with experience
- Optimize a performance criterion using example data or past experience.
- Role of Statistics: Inference from a sample
- Role of Computer science: Efficient algorithms to
  - Solve the optimization problem
  - Represent and evaluate the model for inference

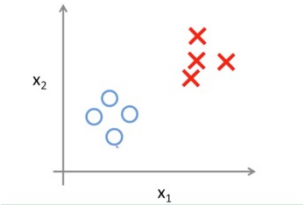
Machine Learning Basics

- Type of Artificial Intelligence that provides computers with the ability to **learn without being explicitly programmed**
- **Various techniques** can be used to for it learn make predictions based on data

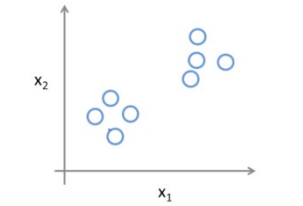


Types of Learning

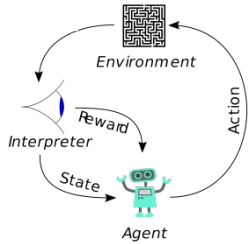
- **Supervised**: Learning with a labeled training set
  - Example: email *classification* with already labeled emails

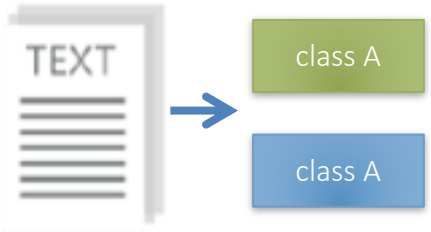


- **Unsupervised**: Discover patterns in unlabeled data
  - Example: *cluster* similar documents based on text

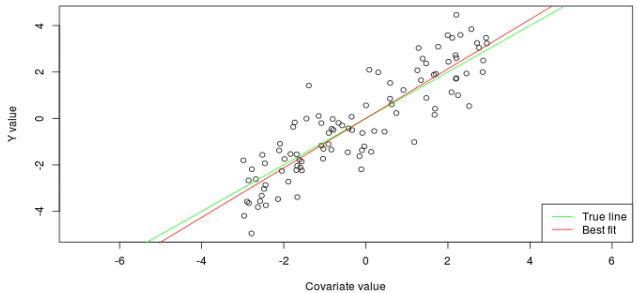


- **Reinforcement learning**: learn to act based on feedback/reward
  - Example: learn to play Go, reward: *win or lose*

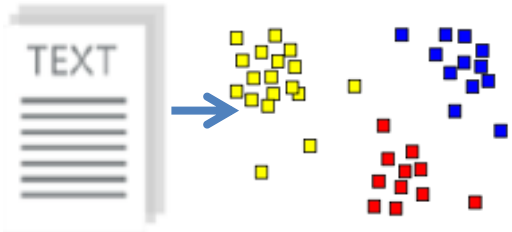




Classification



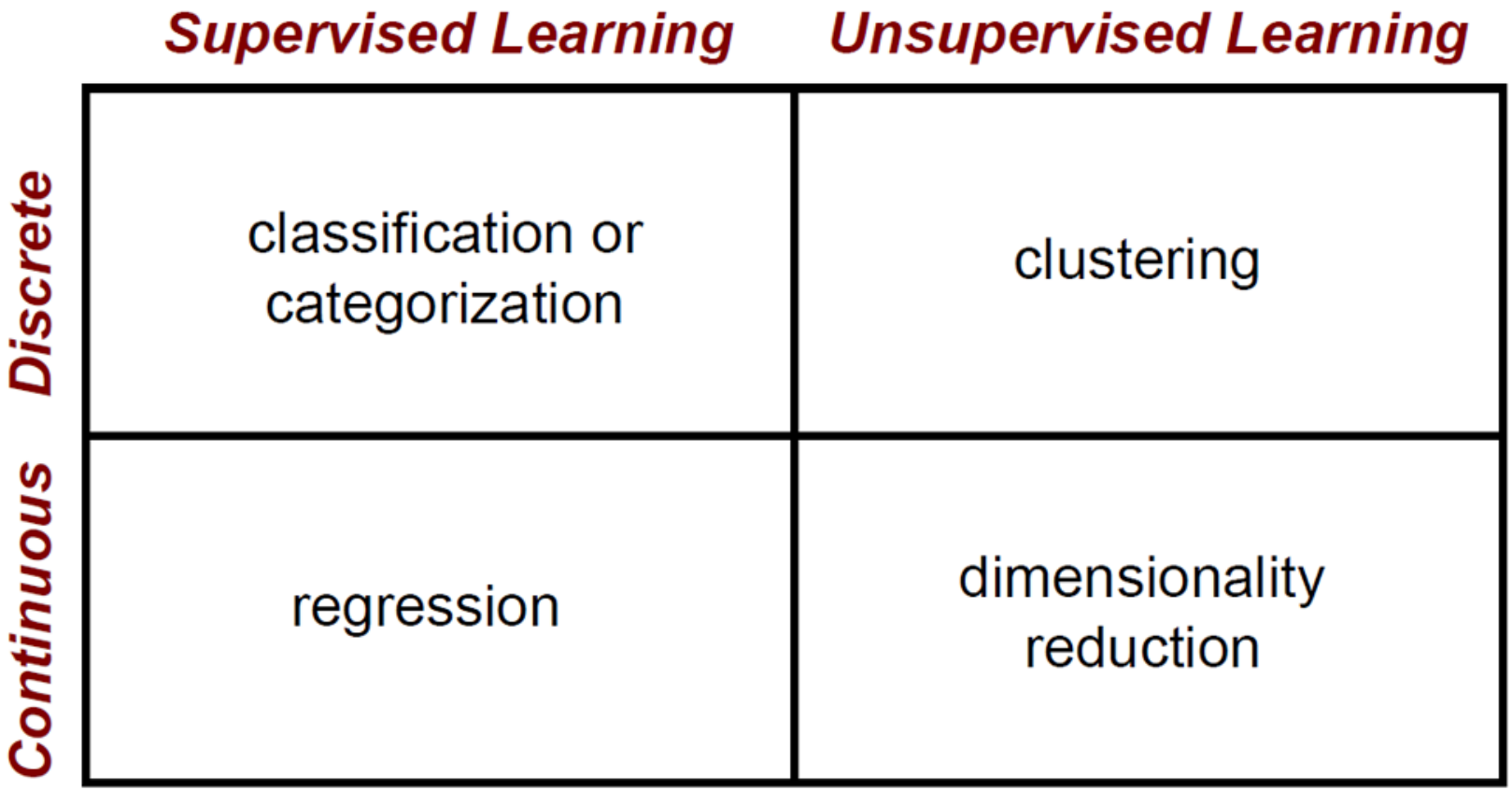
Regression



Clustering

http://mbjoseph.github.io/2013/11/27/measure.html

Machine Learning Problems



Supervised vs. Unsupervised

Given: Training data: $(x_1, y_1), \ldots, (x_n, y_n)$ / $x_i \in \mathbb{R}^d$ and y_i is the label.

example $x_1 \rightarrow$	x_{11}	x_{12}	$\ldots$	x_{1d}	$y_1 \leftarrow$ label
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
example $x_i \rightarrow$	x_{i1}	x_{i2}	$\ldots$	x_{id}	$y_i \leftarrow$ label
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
example $x_n \rightarrow$	x_{n1}	x_{n2}	$\ldots$	x_{nd}	$y_n \leftarrow$ label

fruit	length	width	weight	label
fruit 1	165	38	172	Banana
fruit 2	218	39	230	Banana
fruit 3	76	80	145	Orange
fruit 4	145	35	150	Banana
fruit 5	90	88	160	Orange
...				
fruit n	...	...	...	...

Supervised vs. Unsupervised

fruit	length	width	weight	label
fruit 1	165	38	172	Banana
fruit 2	218	39	230	Banana
fruit 3	76	80	145	Orange
fruit 4	145	35	150	Banana
fruit 5	90	88	160	Orange
...				
fruit n	...	...	...	...

Unsupervised learning:

Learning a model from **unlabeled** data.

Supervised learning:

Learning a model from **labeled** data.

Unsupervised Learning

Training data: "examples" x.

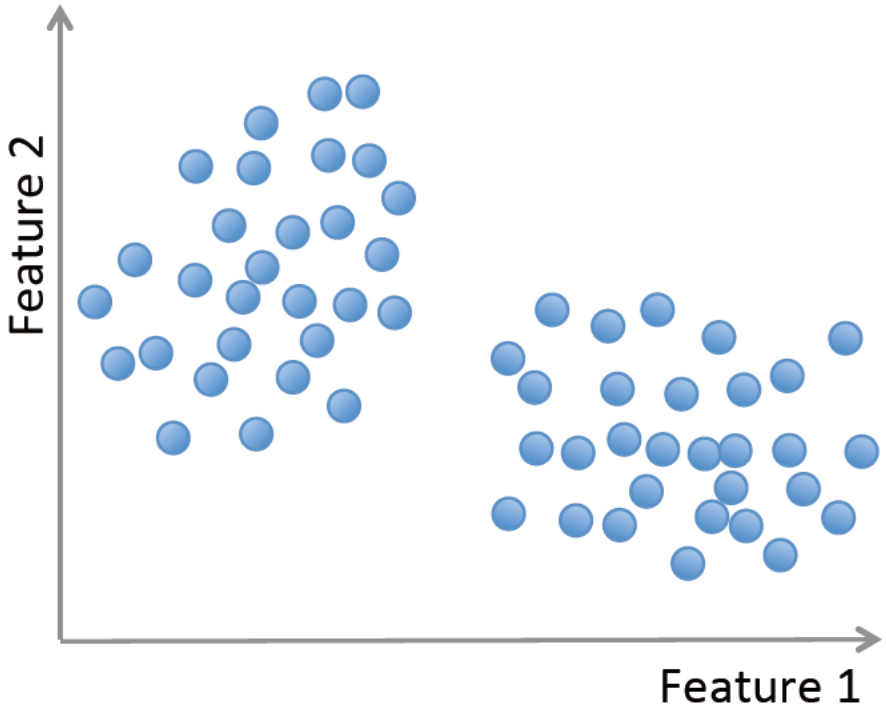
$$x_1, \ldots, x_n, \; x_i \in X \subset \mathbb{R}^n$$

- **Clustering/segmentation:**

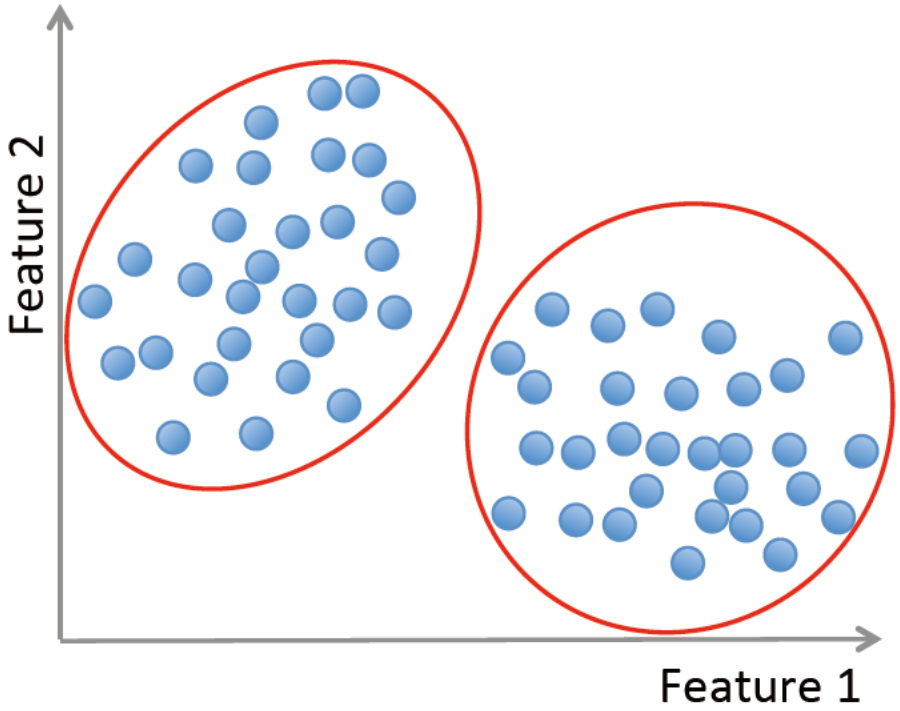
$$f : \mathbb{R}^d \longrightarrow \{C_1, \ldots C_k\} \text{ (set of clusters).}$$

Example: Find clusters in the population, fruits, species.

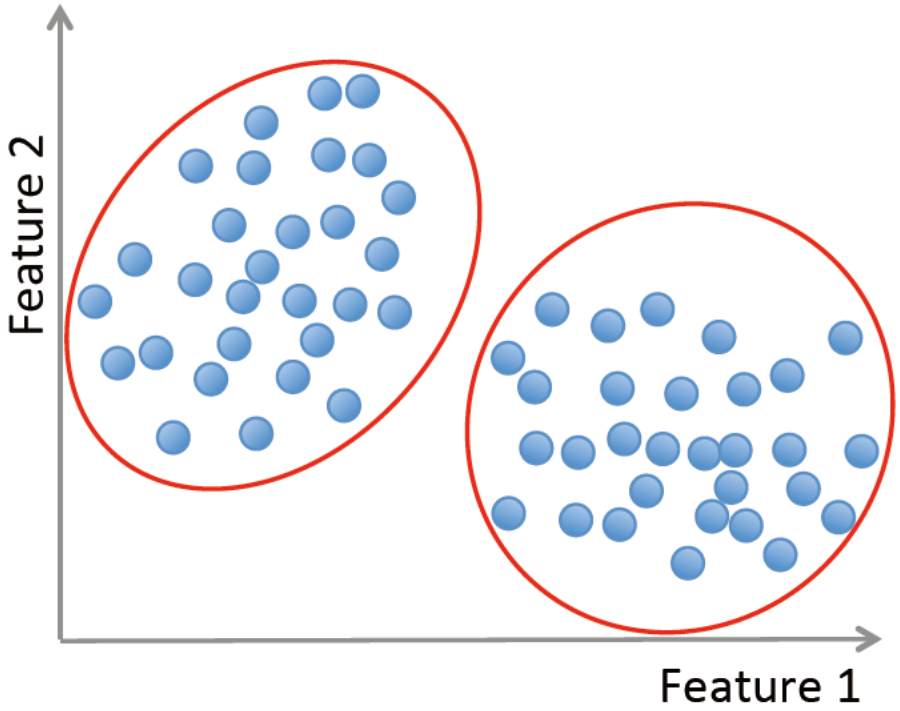
Unsupervised Learning



Unsupervised Learning



Unsupervised Learning



Methods: K-means, gaussian mixtures, hierarchical clustering, spectral clustering, etc.

Supervised Learning

Training data: "examples" x with "labels" y.

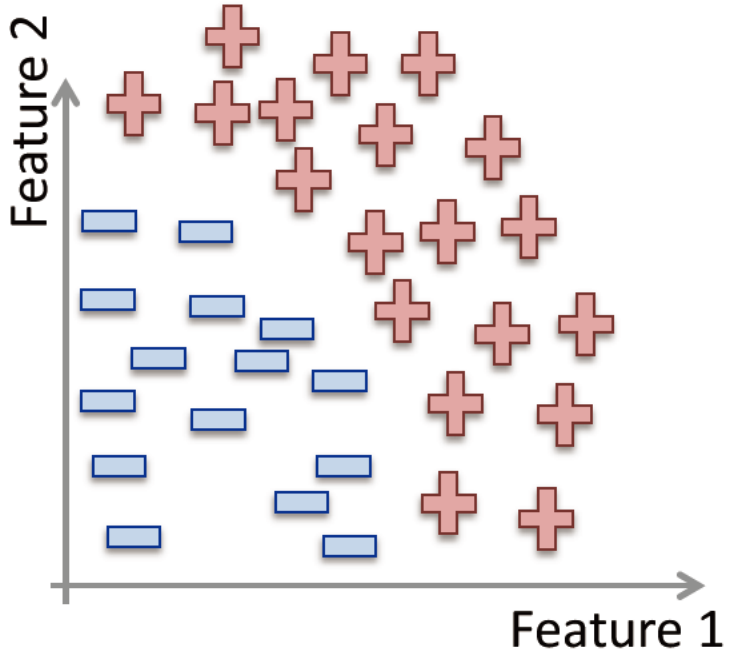
$$(x_1, y_1), \ldots, (x_n, y_n) \ / \ x_i \in \mathbb{R}^d$$

- **Classification:** y is discrete. To simplify, $y \in \{-1, +1\}$

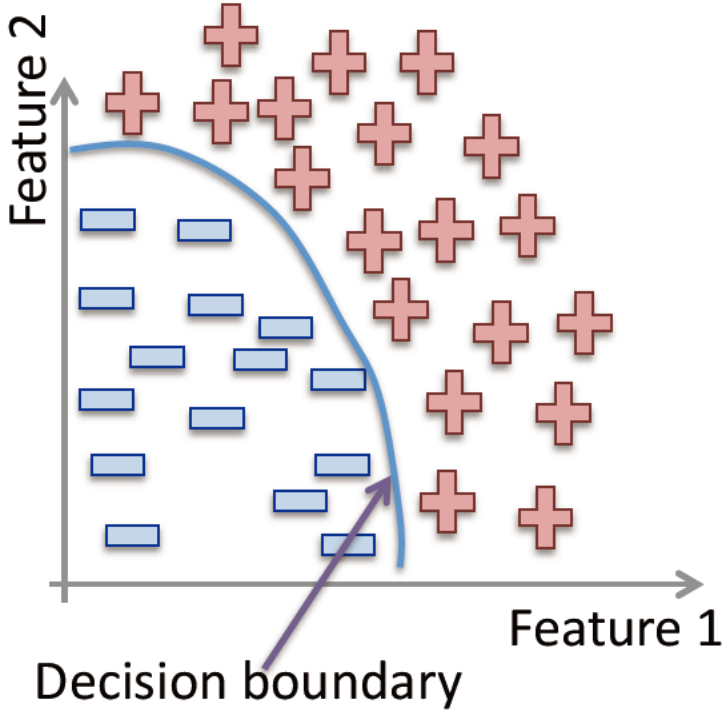
$$f : \mathbb{R}^d \longrightarrow \{-1, +1\}$$ f is called a **binary classifier**.

Example: Approve credit yes/no, spam/ham, banana/orange.

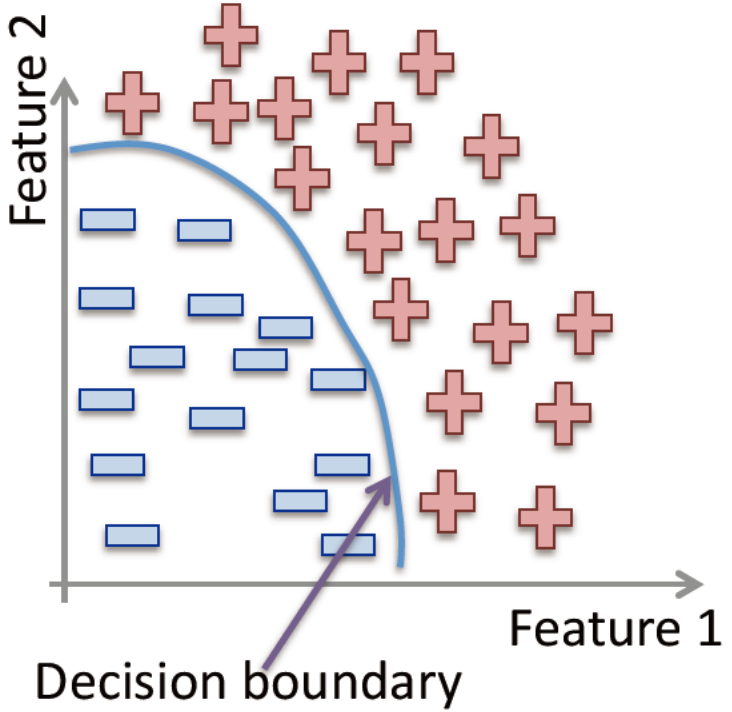
Supervised Learning



Supervised Learning



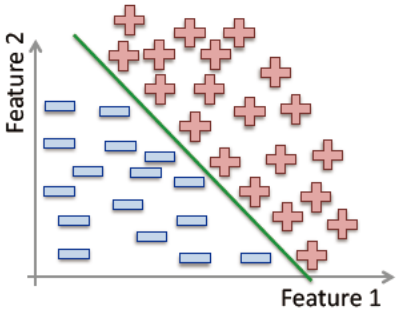
Supervised Learning

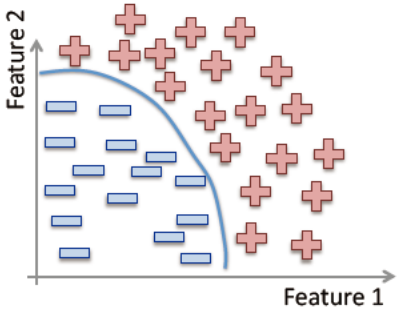


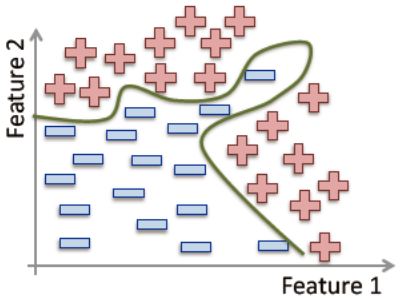
Methods: Support Vector Machines, neural networks, decision trees, K-nearest neighbors, naive Bayes, etc.

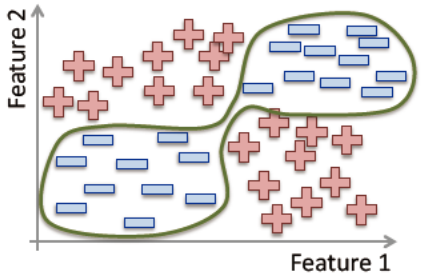
Supervised Learning

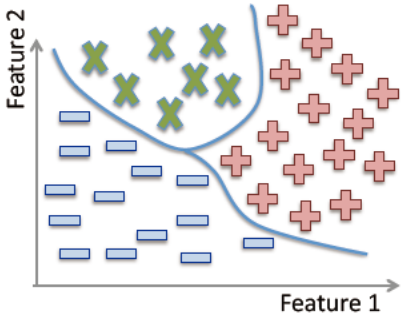
Classification:





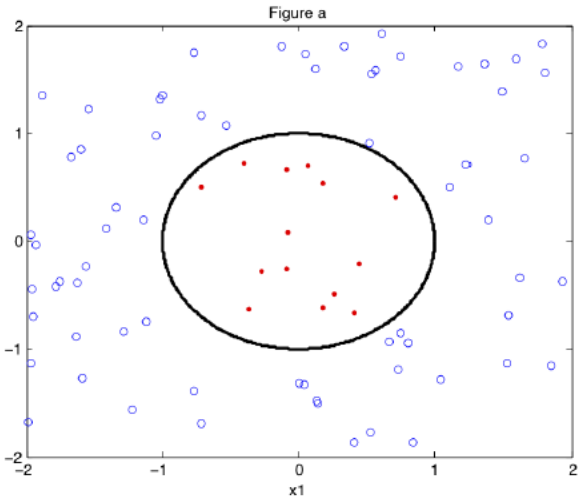


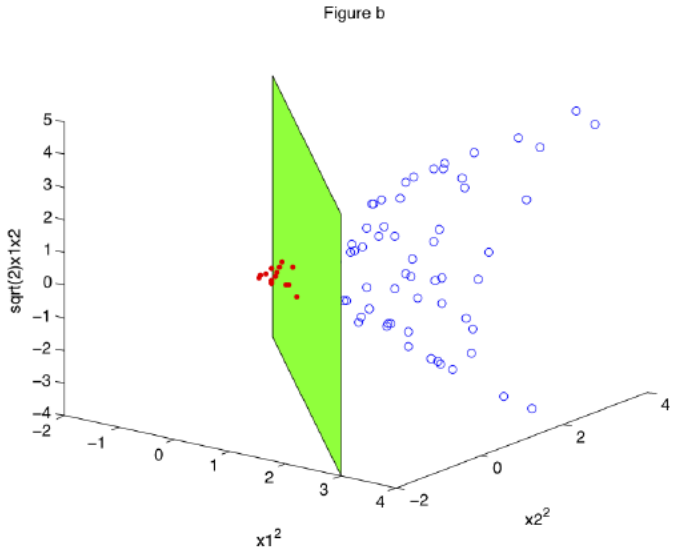




Supervised Learning

Non linear classification





Supervised Learning

Training data: "examples" x with "labels" y.

$$(x_1, y_1), \ldots, (x_n, y_n) \;/\; x_i \in \mathbb{R}^d$$

- **Regression:** y is a real value, $y \in \mathbb{R}$

  $f : \mathbb{R}^d \longrightarrow \mathbb{R}$ f is called a **regressor**.

  Example: amount of credit, weight of fruit.

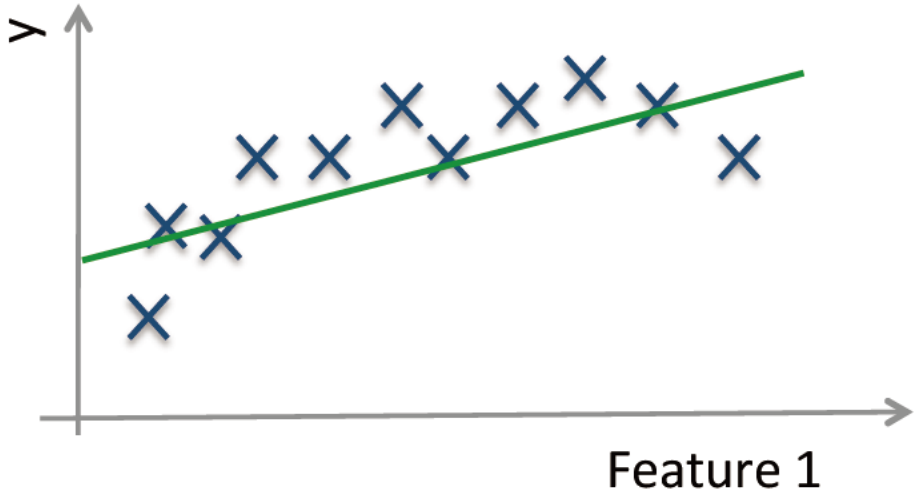
Supervised Learning

Regression:

Example: Income in function of age, weight of the fruit in function of its length.

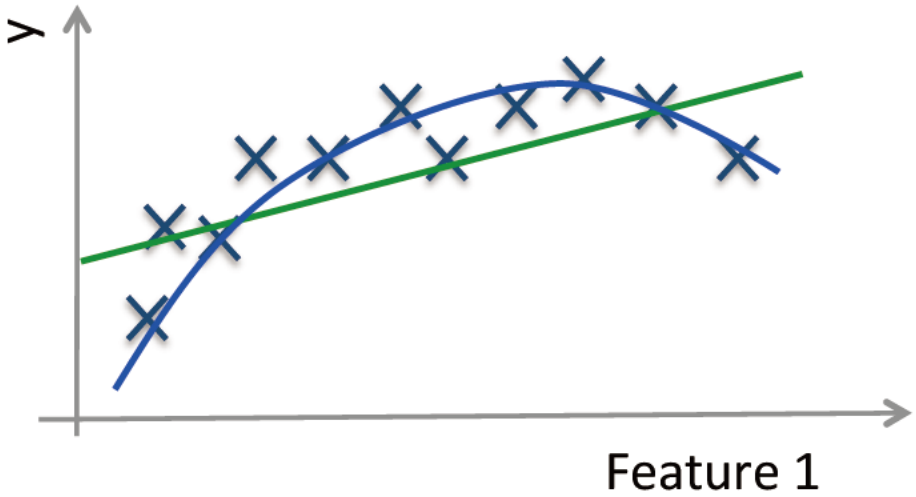
Supervised Learning

Regression:



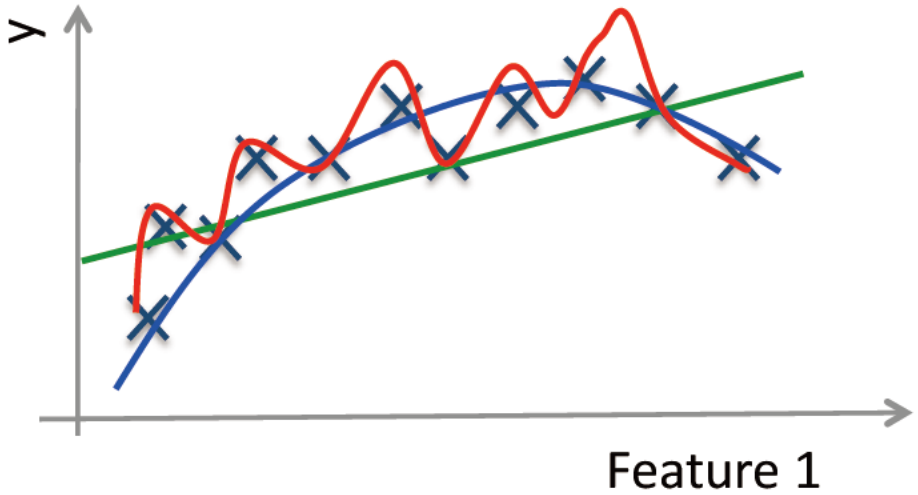
Supervised Learning

Regression:

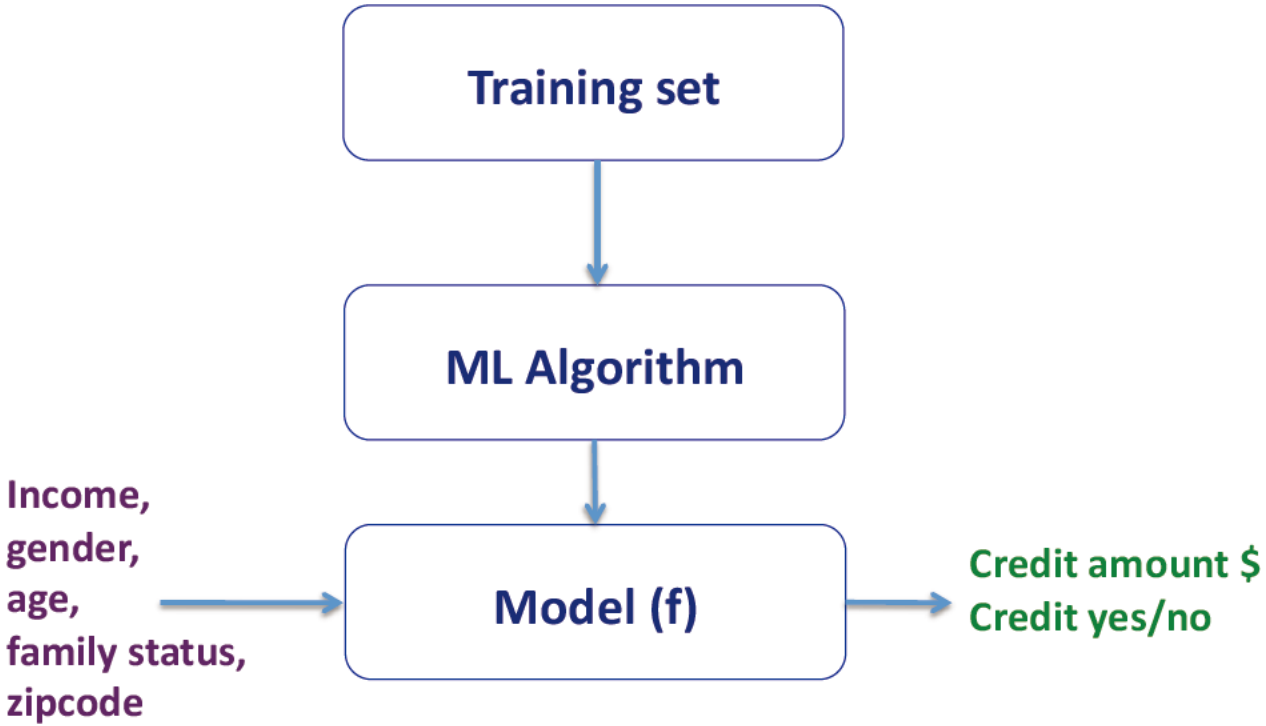


Supervised Learning

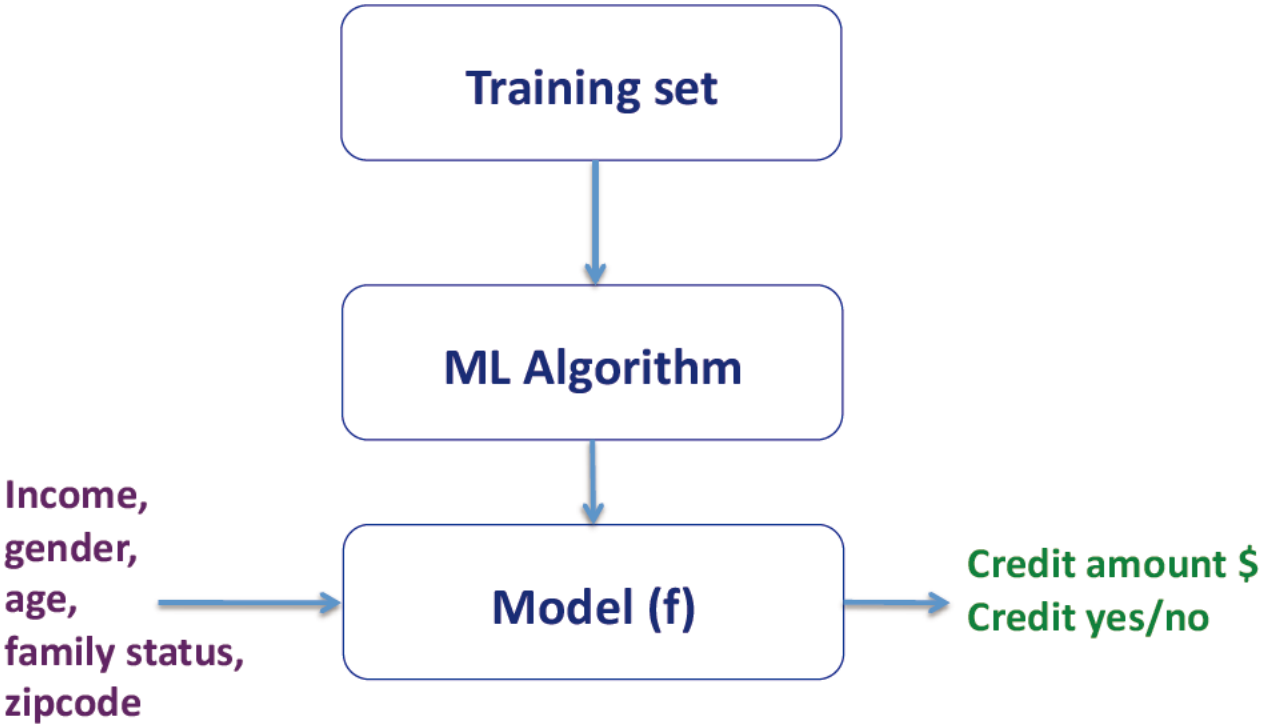
Regression:



Training and Testing



Training and Testing



Question: How can we be confident about f?

Training and Testing

- We calculate E^{train} the in-sample error (training error or empirical error/risk).

$$E^{train}(f) = \sum_{i=1}^{n} \ell oss(y_i, f(x_i))$$

- Examples of loss functions:
  - **Classification error:**

$$\ell oss(y_i, f(x_i)) = \begin{cases} 1 & \text{if } sign(y_i) \neq sign(f(x_i)) \\ 0 & \text{otherwise} \end{cases}$$

  - **Least square loss:**

$$\ell oss(y_i, f(x_i)) = (y_i - f(x_i))^2$$

Training and Testing

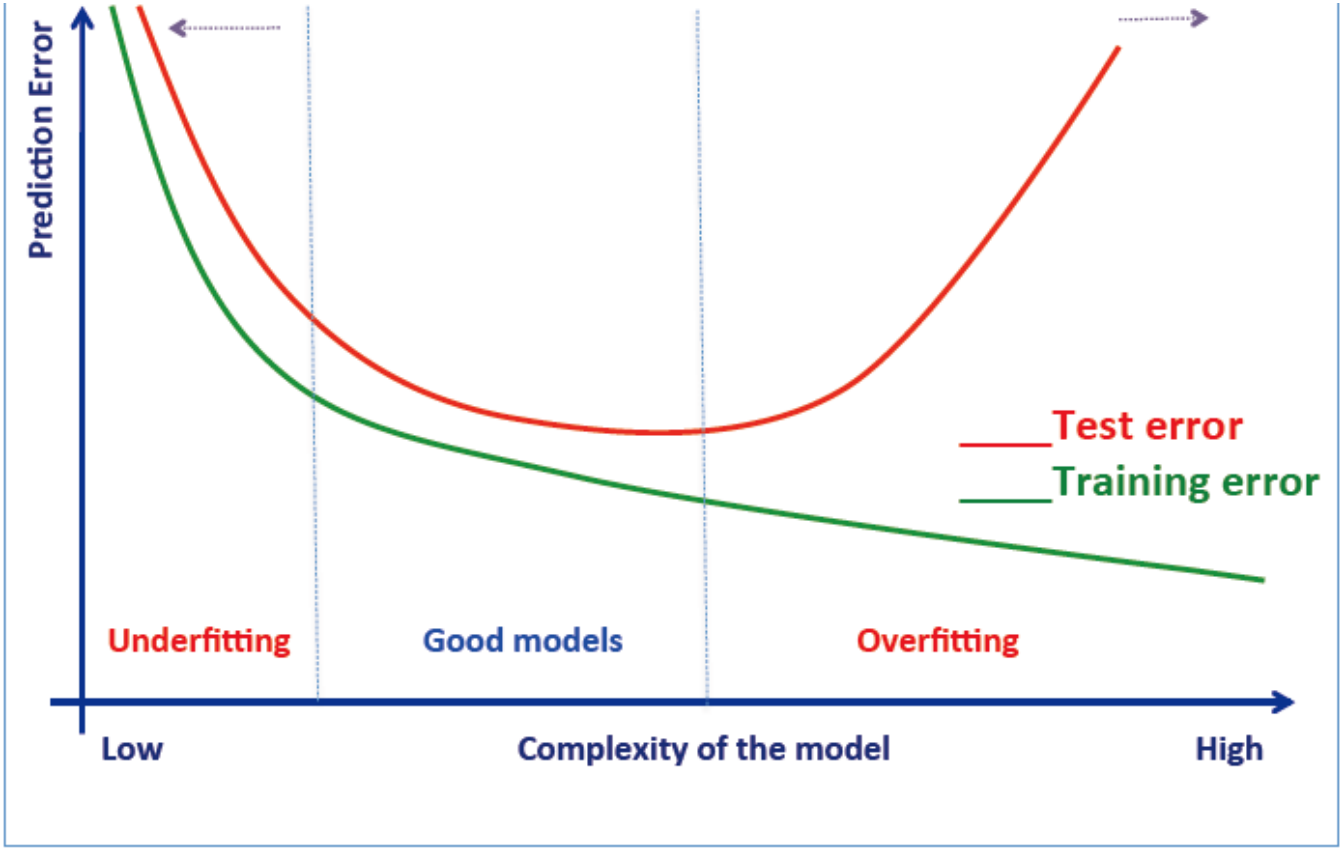
- We calculate E^{train} the in-sample error (training error or empirical error/risk).

$$E^{train}(f) = \sum_{i=1}^{n} \ell oss(y_i, f(x_i))$$

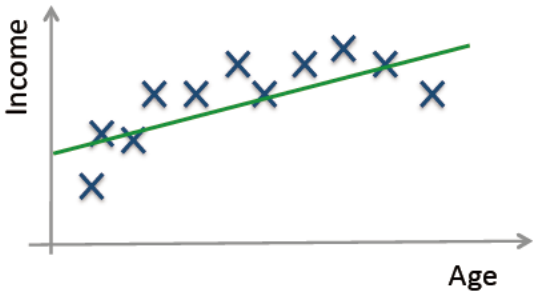
- We aim to have $E^{train}(f)$ small, i.e., minimize $E^{train}(f)$

- We hope that $E^{test}(f)$, the out-sample error (test/true error), will be small too.

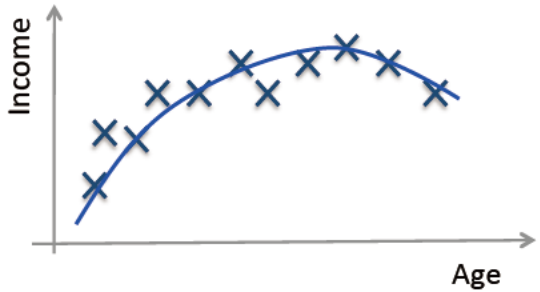
Structural Risk Minimization



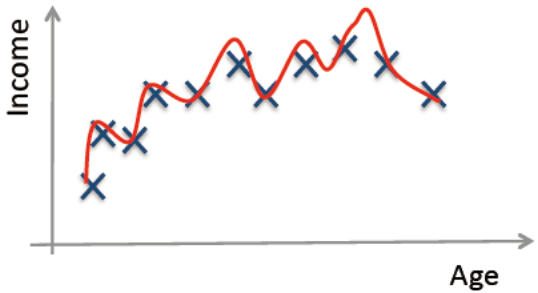
Underfitting and Overfitting



High bias (underfitting)



Just right!



High variance (overfitting)

Avoid Overfitting

- In general, use simple models!
  - **Reduce the number** of features manually or do feature selection.
  - Do a **model selection** (ML course).
  - Use **regularization** (keep the features but reduce their importance by setting small parameter values) (ML course).
  - Do a **cross-validation** to estimate the test error.

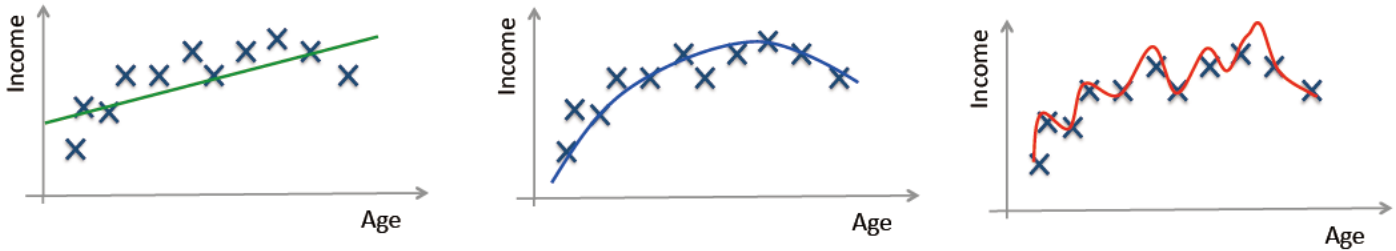
Regularization: Intuition

We want to minimize:

$$\text{Classification term} + C \times \text{Regularization term}$$

$$\sum_{i=1}^{n} \ell oss(y_i, f(x_i)) + C \times R(f)$$

Regularization: Intuition



$$f(x) = \lambda_0 + \lambda_1 x \ \ldots \ (1)$$

$$f(x) = \lambda_0 + \lambda_1 x + \lambda_2 x^2 \ \ldots \ (2)$$

$$f(x) = \lambda_0 + \lambda_1 x + \lambda_2 x^2 + \lambda_3 x^3 + \lambda_4 x^4 \ \ldots \ (3)$$

Hint: Avoid high-degree polynomials.

Train, Validation, and Test

Example: Split the data randomly into 60% for training, 20% for validation and 20% for testing.

Train, Validation, and Test

1. Training set is a set of examples used for learning a model (e.g., a classification model).
2. Validation set is a set of examples that cannot be used for learning the model but can help tune model parameters (e.g., selecting K in K-NN). Validation helps control overfitting.
3. Test set is used to assess the performance of the final model and provide an estimation of the test error.

Note: Never use the test set in any way to further tune the parameters or revise the model.

K-fold Cross Validation

A method for estimating test error using training data.

Algorithm:

Given a learning algorithm $\mathcal{A}$ and a dataset $\mathcal{D}$

Step 1: Randomly partition $\mathcal{D}$ into k equal-size subsets $\mathcal{D}_1, \ldots, \mathcal{D}_k$

Step 2:

For $j = 1$ to k

Train $\mathcal{A}$ on all $\mathcal{D}_i$, $i \in 1, \ldots k$ and $i \neq j$, and get f_j.

Apply f_j to $\mathcal{D}_j$ and compute $E^{\mathcal{D}_j}$

Step 3: Average error over all folds.

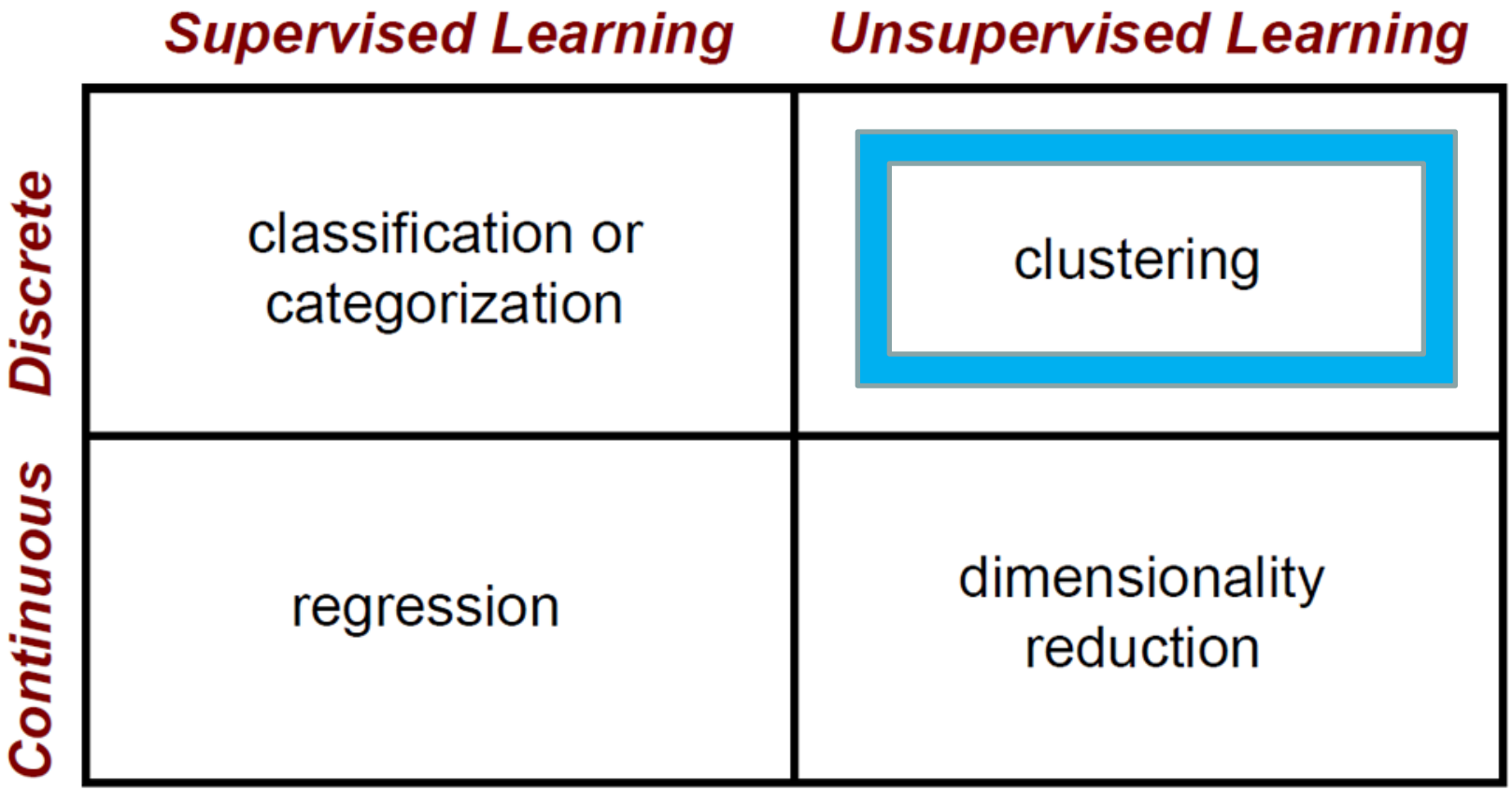
$$\sum_{j=1}^{k} (E^{\mathcal{D}_j})$$

Evaluation Metrics

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

Accuracy	(TP + TN) / (TP + TN + FP + FN)	The percentage of predictions that are correct
Precision	TP / (TP + FP)	The percentage of positive predictions that are correct
Sensitivity (Recall)	TP / (TP + FN)	The percentage of positive cases that were predicted as positive
Specificity	TN / (TN + FP)	The percentage of negative cases that were predicted as negative

Machine Learning Problems



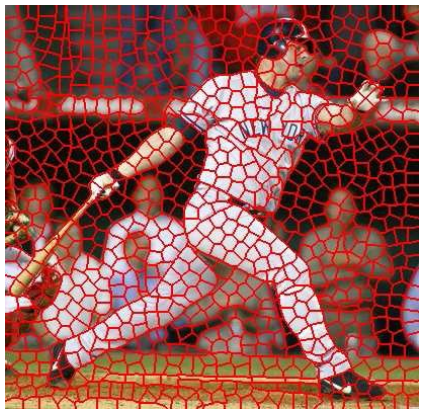
Clustering Example: Image Segmentation

- Goal: Break up the image into meaningful or perceptually similar regions

Segmentation for Feature Support or Efficiency

[Felzenszwalb and Huttenlocher 2004]

[Hoiem et al. 2005, Mori 2005]



[Shi and Malik 2001]

Slide: Derek Hoiem

Segmentation as a Result

Rother et al. 2004

Clustering

- Clustering: group together similar points and represent them with a single token

- Key Challenges:
  1. What makes two points/images/patches similar?
  2. How do we compute an overall grouping from pairwise similarities?

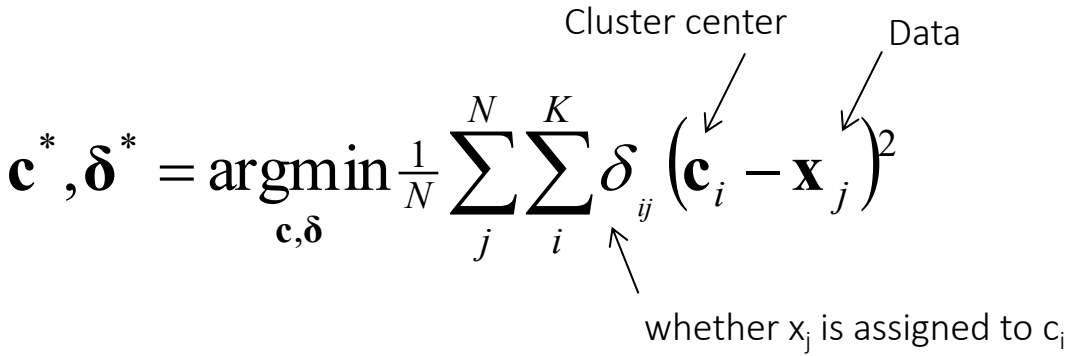
How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Clustering for Summarization

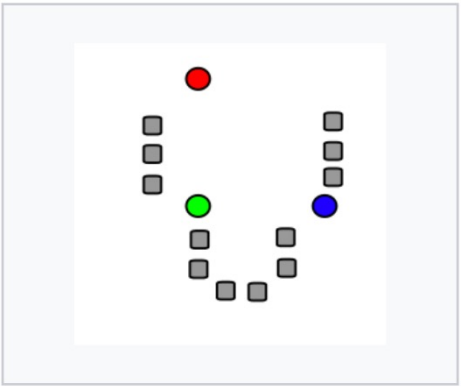
Goal: cluster to minimize variance in data given clusters

- Preserve information

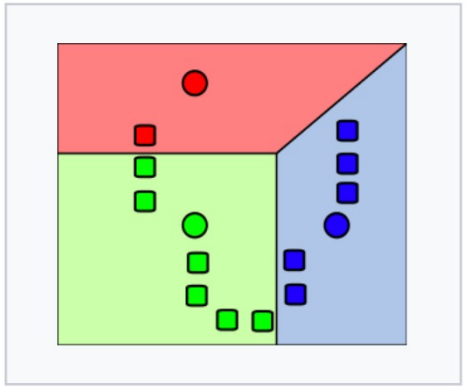


K-means Algorithm

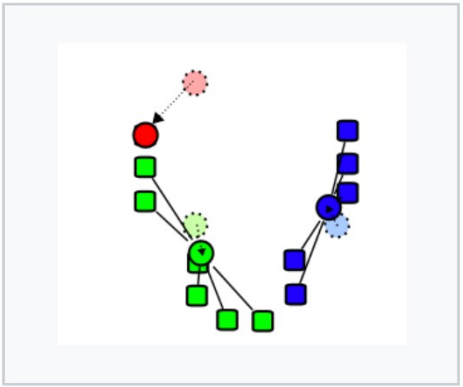
Demonstration of the standard algorithm



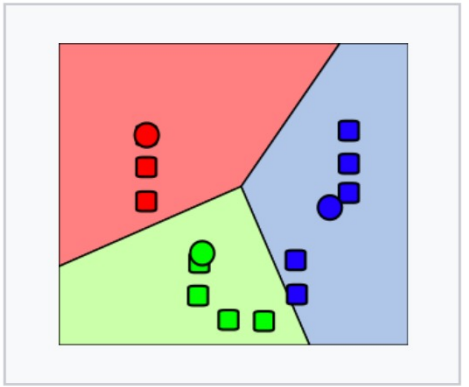
1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3. The centroid of each of the *k* clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means Algorithm

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

2. Assign each point to the closest center

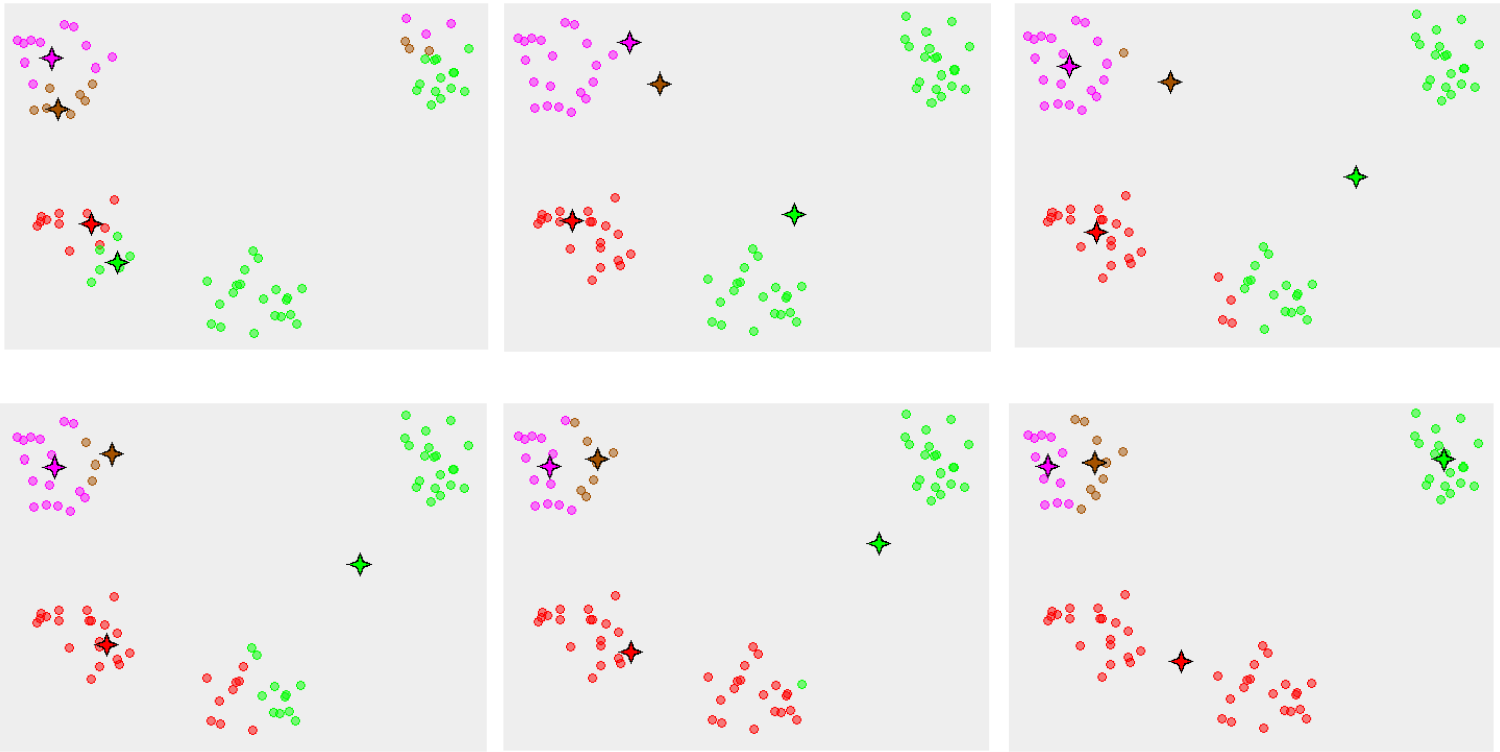
$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned (t=t+1)

K-means Converges to a Local Minimum



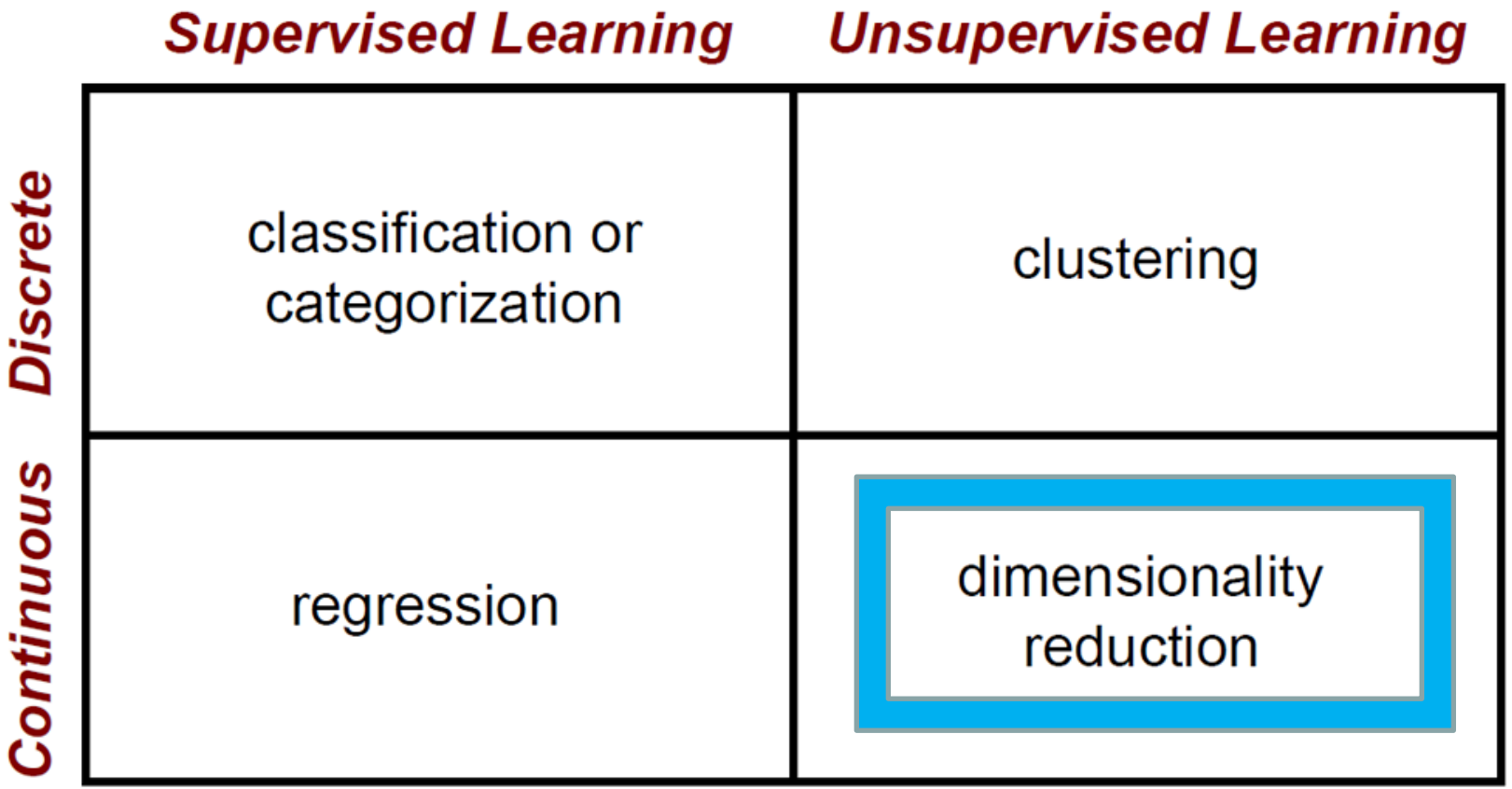
K-means: Design Choices

- Initialization
  - Randomly select K points as initial cluster center
  - Or greedily choose K points to minimize residual
- Distance measures
  - Traditionally Euclidean, could be others
- Optimization
  - Will converge to a *local minimum*
  - May want to perform multiple restarts

How to choose the number of clusters?

- Validation set
  - Try different numbers of clusters and look at performance
    - When building dictionaries (discussed later), more clusters typically work better

Machine Learning Problems



Dimensionality Reduction

- **PCA**, ICA, LLE, Isomap, *Autoencoder*
  - PCA is the most important technique to know. It takes advantage of correlations in data dimensions to produce the best possible lower dimensional representation based on linear projections (minimizes reconstruction error).
  - PCA should be used for dimensionality reduction, not for discovering patterns or making predictions. Don't try to assign semantic meaning to the bases.