Mathematics of Image and Data Analysis
Math 5467

Lecture 19: Intro to Machine Learning

Instructor: Jeff Calder
Email: jcalder@umn.edu

http://www-users.math.umn.edu/~jwcalder/5467S21

Last time

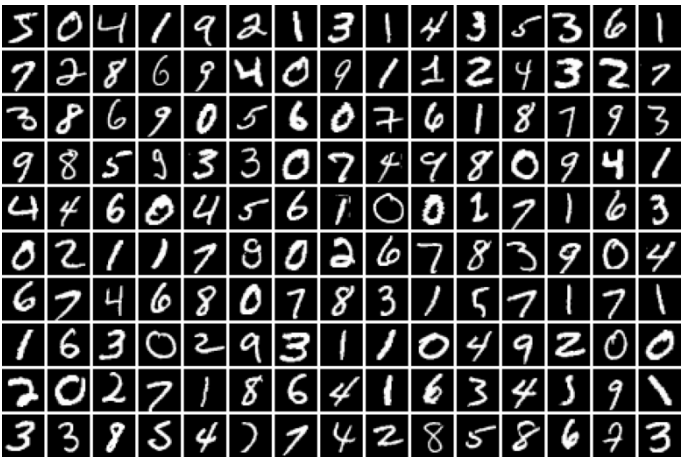
- Wavelets-based image classification
- General discrete wavelet transformations

Today

- Intro to Machine Learning

Machine learning

Machine learning refers to a class of algorithms that learn to complete tasks (like image classification or face recognition) by example, and are not explicitly programmed.



Data

Let $x_1, x_2, \ldots, x_m \in \mathbb{R}^n$ be a collection of datapoints, and let $y_1, y_2, \ldots, y_m \in \mathbb{R}^k$ be the corresponding labels. For example

- x_i can be images
- y_i can be labels, captions, segmentations, denoisings, deblurring, etc.

Normally the labels y_i are chosen from the one-hot vectors $e_1, e_2, \ldots, e_k$, with e_i representing the i^{th} class.

Types of machine learning

1. Fully supervied: Uses only training data $(x_1, y_1), \ldots, (x_m, y_m)$.
2. Semi-supervised: Uses training data and additional unlabeled data.
3. Unsupervised: Does not use any label information (e.g., clustering, PCA, dimension reduction, etc.)

Fully-supervised learning

In fully supervised learning, the algorithm uses the *training data* pairs given by $(x_1, y_1), \ldots, (x_m, y_m)$ to learn a mapping

$$f : \mathbb{R}^n \to \mathbb{R}^k \tag{1}$$

that generalizes the rule $f(x_i) = y_i$.
The learning is acheived by minimizing a loss function of the form

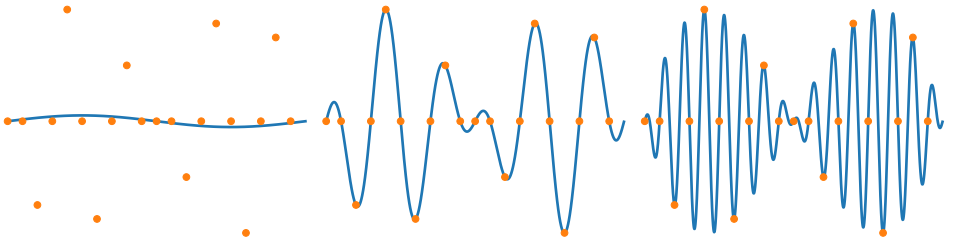
$$\mathcal{L}(\omega) = \frac{1}{m} \sum_{i=1}^{m} \ell(f(x_i; \omega), y_i), \tag{2}$$

where $\ell : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$ is a loss function. Possible choices for ℓ

- L^2 loss $\ell(x, y) = |x - y|^2$
- Cross-entropy loss

$$\ell(x, y) = -\sum_{j=1}^{k} x(j) \log(y(j)).$$

Over/under/correct fitting



(a) Underfitting (b) Correct fit (?) (c) Overfitting

- The goal is not to exactly fit the training data, but to learn a function that generalizes well to new data it has not seen before.
- The correct fit depends on the amount of noise in the data and labels.

Testing/training split

In practice, we measure generalization error by splitting our dataset into training and testing subsets.

- The model is trained using the training data
- Evaluated on the held out test data.

The peformance on testing data gives us an idea of how the algorithm will generalize to new data.

Generalization error

Theoretical machine learning:
Assume the training data $(x_1, y_1), \ldots, (x_m, y_m)$ are independent and identically distributed random variables sampled from a probability distribution μ on $\mathbb{R}^n \times \mathbb{R}^k$. We let $\hat{\omega}$ denote a minimizer of the loss function $\mathcal{L}$, so $x \mapsto f(x; \hat{\omega}) =: \hat{f}(x)$ is the learned function. The *generalization error* is defined as

$$\mathcal{G}(\hat{f}) = \int_{\mathbb{R}^n} \ell(\hat{f}(x), y) d\mu(x, y) - \underbrace{\frac{1}{m} \sum_{i=1}^{m} \ell(\hat{f}(x_i), y_i)}_{\mathcal{L}(\hat{\omega})}. \tag{3}$$

- In machine learning, one really wants to minimize the first term above–the *population loss*–which measures the expected loss of the model on a new random training data sample (x, y).
- Minimizing $\mathcal{L}$ is a surrogate problem, and if the generalization error $\mathcal{G}(\hat{f})$ is small, then we will have done a good job minimizing the population loss as well.

Generalization error

There is deep mathematical work in theoretical computer science that shows how to estimate and control the generlization error

$$\mathcal{G}(\hat{f}) = \int_{\mathbb{R}^n} \ell(\hat{f}(x), y) d\mu(x, y) - \underbrace{\frac{1}{m} \sum_{i=1}^{m} \ell(\hat{f}(x_i), y_i)}_{\mathcal{L}(\hat{\omega})}.$$

One way to do this is to estimate the worst case error over the set $\mathcal{F}$ of parameterized functions one is using in the machine learning model. That is, we estimate

$$\mathcal{G}(\hat{f}) \leq \sup_{f \in \mathcal{F}} \mathcal{G}(f).$$

If we can estimate the supremum on the right hand side, then we can control the generalization error. This is called the *hypothesis space complexity approach*, since bounds on $\sup_{f \in \mathcal{F}} \mathcal{G}(f)$ involve measuring the size of the hypothesis space $\mathcal{F}$ in an appropriate way (e.g., Radamacher complexity, VC dimension, etc.).

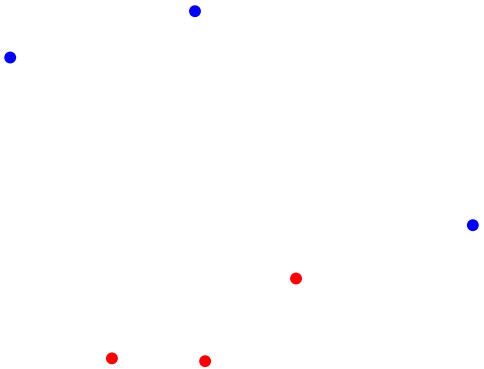
Semi-supervised learning

In semi-supervised learning, we have a, possibly ver small, amount of training data $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, and a large amount of unlabeled data denoted by $x_{m+1}, x_{m+2}, \ldots, x_N$, where $N \gg m$.

- The goal is to use the additional unlabeled data to train a better clasifier with limited labeled data.
- In many applications, like image classification or speech recognition, it is essentially free to get access to large amounts of unlabeled data (e.g., images and audio samples), so we may as well try to make use of this data.

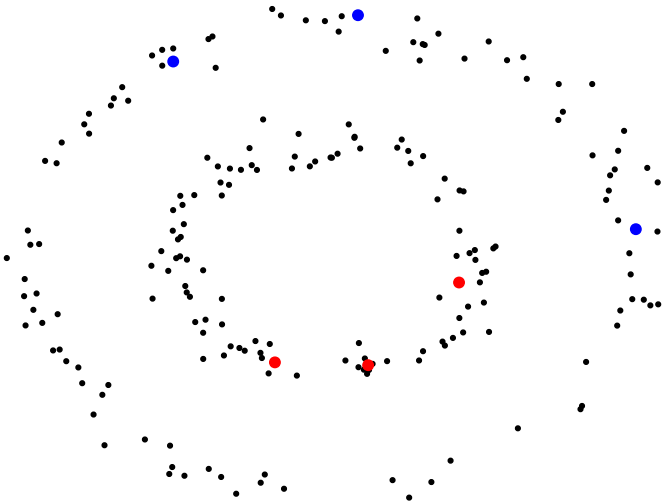
Example

To see why unlabeled data may be useful in classification, consider the figure below.



Example

To see why unlabeled data may be useful in classification, consider the figure below.



Transductive vs Inductive

Semi-supervised learning comes in 2 variations.

- **Inductive**: We still learn a general rule $u : \mathbb{R}^n \to \mathbb{R}^k$ that aims to generalize the training data, while using properties of the unlabeled data.
- **Transductive**: We only learn labels for the additional unlabeled datapoints $x_{m+1}, \ldots, x_N$

Graph-based learning, which we will discuss soon, falls into the transductive class of semi-supervised learning.

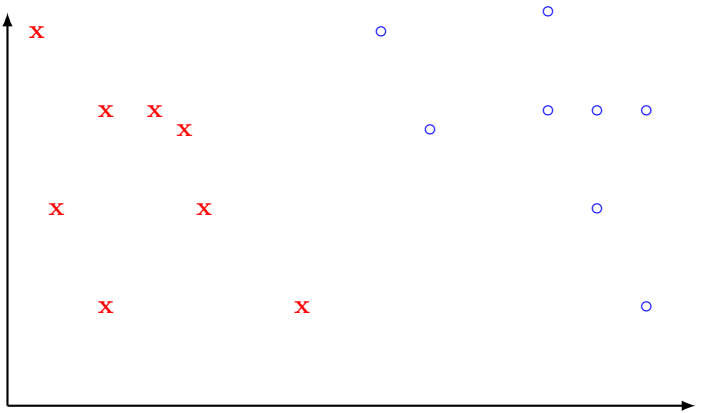
Unsupervised learning

Unsupervised learning algorithms use only the unlabeled data $x_1, x_2, \ldots, x_m$ for learning. Common tasks include

- Clustering (like k-means clustering and spectral clustering)
- Dimension reduction algorithms, like principal component analysis (PCA) and the spectral and t-SNE embeddings (covered soon).

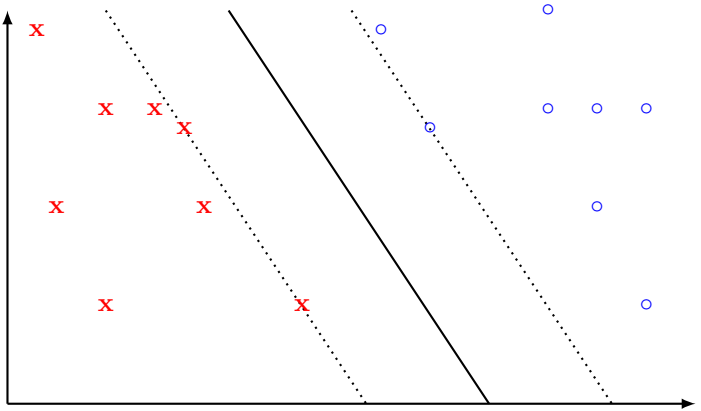
Support Vector Machines

Big Idea: Separate data with lines (and hyperplanes in higher dimensions)



Support Vector Machines

A *maximum margin classifier* is a support vector machine that classifies the data with the largest margin.

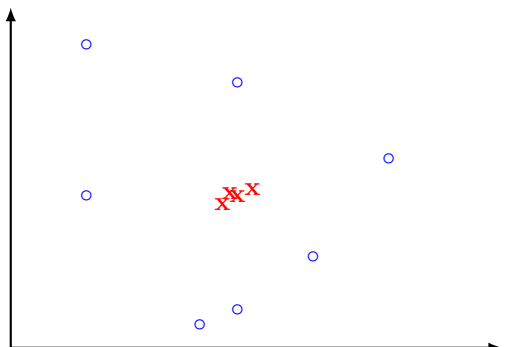


Nonlinear decision boundaries

The *kernel trick* allows for nonlinear decision boundaries by adding additional dimensions

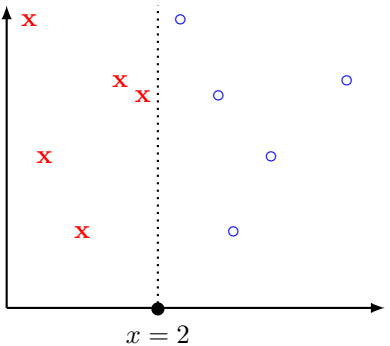
$$(x, y) \text{ becomes } (x, y, f(x, y), g(x, y), \dots).$$

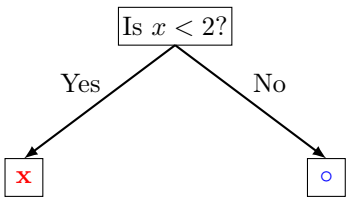
Often f and g are polynomials.



Decision Trees

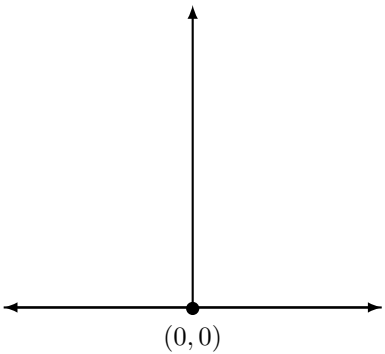
Big Idea: Ask as series of questions to separate data into groups.

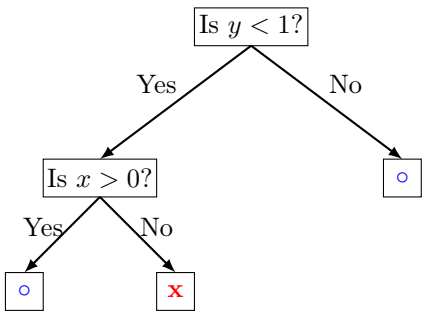




Decision Tree Exercise

Exercise: Consider the decision tree below. Sketch points x and ○ that would be classified correctly by the tree.





Classifying Superheros Exercise

Exercise:

1. Create a decision tree that correctly classifies the superheros as *good* or *evil*.
2. Use the tree to classify Batgirl and Riddler.

Superhero	**mask**	**cape**	**tie**	**ears**	**smokes**	**height**	**class**
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?

Table 1: Superhero Data

Data borrowed from https://hackr.io/blog/decision-tree-in-machine-learning.

Superhero tree: One possible solution

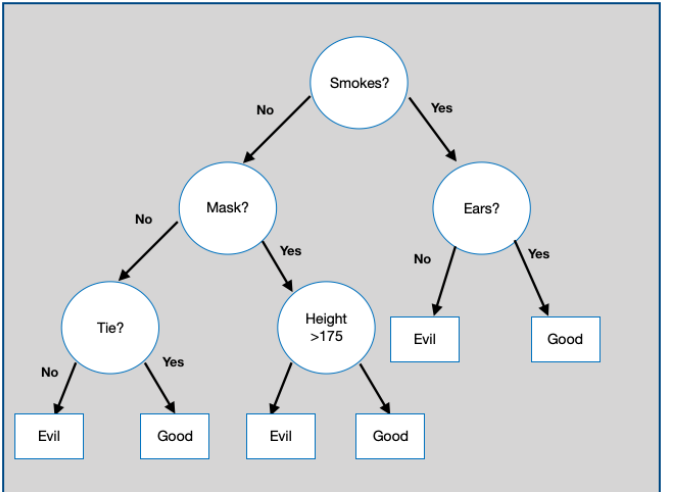
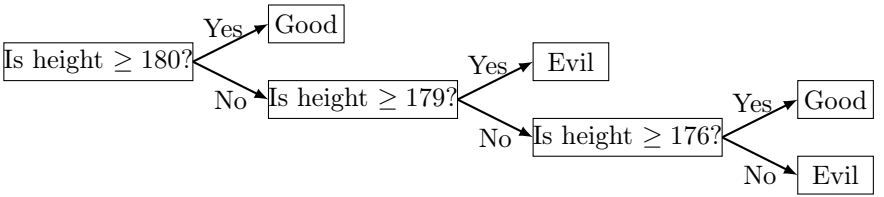


Image from https://hackr.io/blog/decision-tree-in-machine-learning.

Overfitting in decision trees

Question: Why is the tree below bad? Does it classify Riddler/Batgirl correctly?



Superhero	**mask**	**cape**	**tie**	**ears**	**smokes**	**height**	**class**
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?

Random forests

1. Random forests are ensembles of decision trees, and aim to prevent *overfitting*.
2. Each tree is trained on a random bootstrap of the training data and features.
3. Final classification is obtained by taking the most common label among all trees.

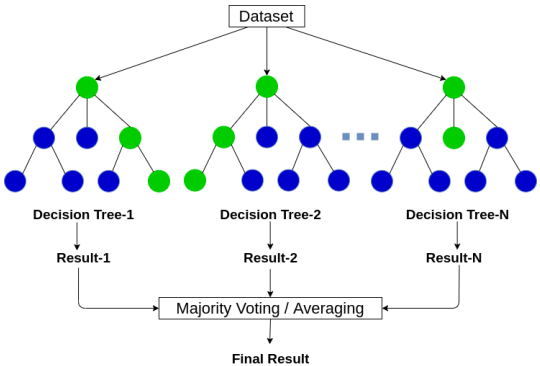


Image from https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/

Intro to Machine Learning (.ipynb)