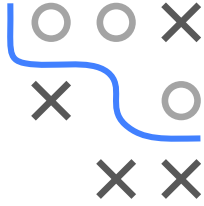
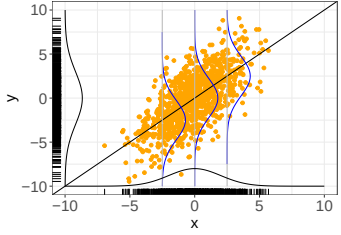
Introduction to Machine Learning



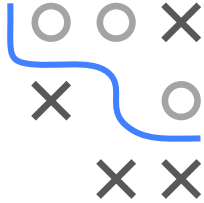
ML-Basics
Data



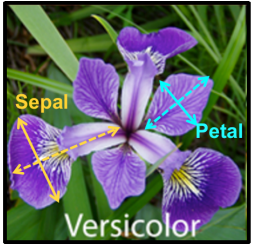
Learning goals

- Understand structure of tabular data in ML
- Understand difference between target and features
- Understand difference between labeled and unlabeled data
- Know concept of data-generating process

IRIS DATA SET



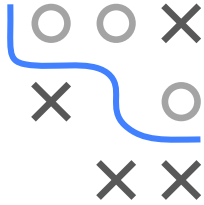
- Introduced by statistician Fisher, frequently used toy example
- Classify iris subspecies based on flower measurements
- 150 iris flowers: 50 versicolor, 50 virginica, 50 setosa
- Sepal length / width and petal length / width in [cm]



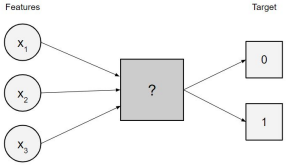
▸ Click for source

- Word of warning: “iris” is a small, clean, low-dim data set, very easy to classify; this is not necessarily true in the wild

DATA IN SL

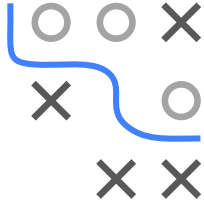


- Measurements on different aspects of *n* objects:
  - **Target**: output variable / goal of prediction
  - **Features**: properties that describe the object
- We assume some relationship between features and target, and want to discover that predictive rule.



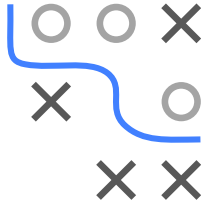
Features x				Target y
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
4.3	3.0	1.1	0.1	setosa
5.0	3.3	1.4	0.2	setosa
7.7	3.8	6.7	2.2	virginica
5.5	2.5	4.0	1.3	versicolor

DATA TYPES



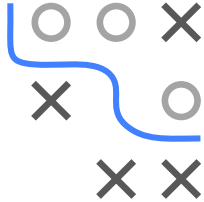
- Features and target may be of different data types
  - **Numerical** variables: $\in \mathbb{R}$
  - **Integer** variables: $\in \mathbb{Z}$
  - **Categorical** variables: $\in \{C_1, ..., C_g\}$
  - **Binary** variables: $\in \{0, 1\}$
- For the **target** variable, this results in different tasks of supervised learning: *regression* and *classification*
- Most learning algorithms can only deal with numerical features, although there are some exceptions (e.g., trees can use categoricals without problems). For other types we usually have to pick an appropriate **encoding** to cast them to numerical values.
- If not stated otherwise we assume numerical features

ENCODING FOR CATEGORICALS



- We expand the representation of a variable x with k mutually exclusive categories from a scalar to a length-$\tilde{k}$ vector with at most one element being 1, and 0 otherwise: $\boldsymbol{o}(x)_j = \mathbb{I}(x = j) \in \{0, 1\}$
- Each entry of $\boldsymbol{o}(x) \in \{0, 1\}^{\tilde{k}}$ is treated as a separate feature
- Two popular ways to do this are
  - **One-hot encoding**: $\tilde{k} = k$ dummies, so *exactly one* element is 1 ("hot"). E.g., $x \in \{a, b, c\} \mapsto \boldsymbol{o}(x) = (x_a, x_b, x_c)$, with $x_a = x_b = 0, x_c = 1$ and $\boldsymbol{o}(x) = (0, 0, 1)$ for $x = c$.
  - **Dummy encoding**: $\tilde{k} = k - 1$ dummies, so *at most one* element is 1, removing the redundancy of one-hot encoding (necessary for learners that require non-singular input matrices, such as in linear regression). E.g., $x \in \{a, b, c\} \mapsto \boldsymbol{o}(x) = (x_a, x_b)$ for reference category c, with $x_a = x_b = 0$ and $\boldsymbol{o}(x) = (0, 0)$ for $x = c$.
- We also often use this encoding for categorical targets
- For features with a natural **order** in their categories we resort to encodings that reflect this ordinality, e.g., a sequence of integer values

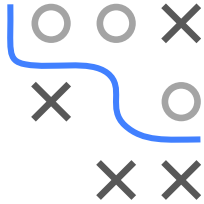
OBSERVATION LABELS



- We call the entries of the target column **labels**
- We distinguish two types of data:
  - **Labeled** data: target is observed
  - **Unlabeled** data: target is unknown

	Features x				Target y
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
labeled data	4.3	3.0	1.1	0.1	setosa
	5.0	3.3	1.4	0.2	setosa
	7.7	3.8	6.7	2.2	virginica
	5.5	2.5	4.0	1.3	versicolor
unlabeled data	5.9	3.0	5.1	1.8	?
	4.4	3.2	1.3	0.2	?

NOTATION FOR DATA



$$\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)} \right), \ldots, \left(\mathbf{x}^{(n)}, y^{(n)} \right) \right) \in (\mathcal{X} \times \mathcal{Y})^n$$

Where

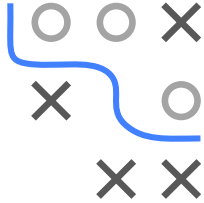
- $\mathcal{X}$: input space, usually $\mathcal{X} \subset \mathbb{R}^p$
- $\mathcal{Y}$: output / target space
- $\left(\mathbf{x}^{(i)}, y^{(i)} \right) \in \mathcal{X} \times \mathcal{Y}$: i-th observation
- $\mathbf{x}_j = \left(x_j^{(1)}, \ldots, x_j^{(n)} \right)^\top$: j-th feature vector

We also use

- $(\mathcal{X} \times \mathcal{Y})^n$ is the set of all data sets of size n, as $\mathbb{D}_n$
- $\bigcup_{n \in \mathbb{N}} (\mathcal{X} \times \mathcal{Y})^n$ is the set of all finite data sets, as $\mathbb{D}$

$\Rightarrow$ We observe n objects, described by p features

DATA-GENERATING PROCESS



- We assume the observed data $\mathcal{D}$ to be generated by a process that can be characterized by some probability distribution

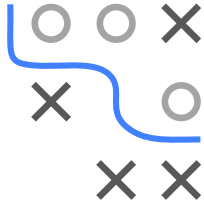
$$\mathbb{P}_{xy},$$

  defined on $\mathcal{X} \times \mathcal{Y}$

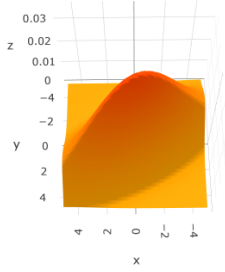
- We denote the random variables following this distribution by lowercase **x** and y

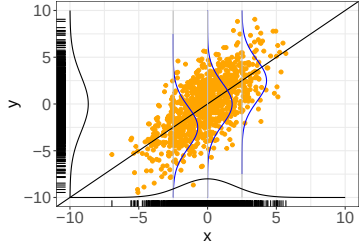
- We usually assume the true distribution to be **unknown**. Learning (part of) its structure to predict y is what ML is all about

DATA-GENERATING PROCESS

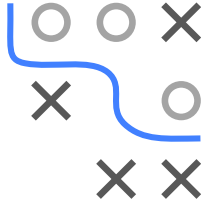


- We assume data to be drawn *i.i.d.* (**i**ndependent and **i**dentically **d**istributed) from $\mathbb{P}_{xy}$
- This means: We assume that all samples are drawn from the same distribution and are mutually independent – the i-th realization does not depend on the other $n - 1$ ones
- This is a strong assumption, not always realistic. More complex scenarios (e.g., time series) exist, but we focus on the i.i.d. scenario here





DATA-GENERATING PROCESS



Remarks:

- With a slight abuse of notation we write random variables, e.g., $\mathbf{x}$ and y, in lowercase, as normal variables or function arguments. The context will make clear what is meant.
- Often, distributions are characterized by a parameter vector $\boldsymbol{\theta} \in \Theta$. We then write $p(\mathbf{x}, y \mid \boldsymbol{\theta})$.
- This lecture mostly takes a frequentist perspective. Distribution parameters $\boldsymbol{\theta}$ appear behind the $\mid$ for improved legibility, not to imply that we condition on them in a probabilistic Bayesian sense. So, strictly speaking, $p(\mathbf{x}|\boldsymbol{\theta})$ should usually be understood to mean $p_{\boldsymbol{\theta}}(\mathbf{x})$ or $p(\mathbf{x}, \boldsymbol{\theta})$ or $p(\mathbf{x}; \boldsymbol{\theta})$. On the other hand, this notation makes it very easy to switch to a Bayesian view.