Supervised Learning

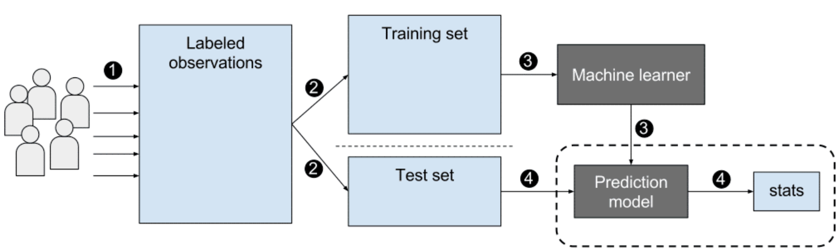
CSC 461: Machine Learning

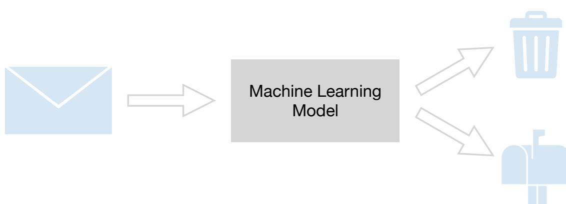
Fall 2022

Prof. Marco Alvarez
University of Rhode Island

Supervised Learning

Example: spam filtering





Components of supervised learning

- Data instance $(x, y), \quad x \in \mathcal{X}$ and $y \in \mathcal{Y}$
- Input space $\mathcal{X}$
- Output space $\mathcal{Y}$
- Data $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$
- Hypothesis $h : \mathcal{X} \mapsto \mathcal{Y}, h \in \mathcal{H}$

Example: spam filtering

- Problem
  - ✓ automatically tagging email messages as **spam** (1) or **ham** (0)
- Input space
  - ✓ assume every email is represented as a fixed-length vector of 10 features
- Output space?

Input space

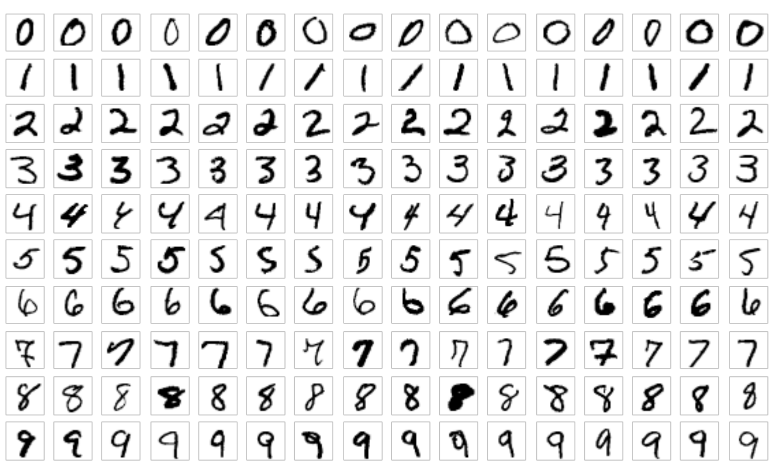
- Samples (data instances) are drawn from an **unknown distribution** $P(X, Y)$

$$\mathscr{D} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$$

in general $\mathscr{X} = \mathbb{R}^d$

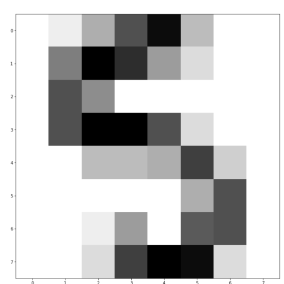
$$(\mathbf{x_i}, y_i) \sim \underset{\text{unknown}}{P}$$

Example: MNIST Dataset



https://en.wikipedia.org/wiki/MNIST_database

MNIST data instance



Image

```
[[ 0.   1.   5. 11. 15.   4.   0.   0.]
 [ 0.   8. 16. 13.   6.   2.   0.   0.]
 [ 0. 11.   7.   0.   0.   0.   0.   0.]
 [ 0. 11. 16. 16. 11.   2.   0.   0.]
 [ 0.   0.   4.   4.   5. 12.   3.   0.]
 [ 0.   0.   0.   0.   0.   5. 11.   0.]
 [ 0.   0.   1.   6.   0. 10. 11.   0.]
 [ 0.   0.   2. 12. 16. 15.   2.   0.]]
```

Matrix representation

```
[ 0.  1.  5. 11. 15.  4.  0.  0.  0.  8. 16. ……… 11.  0.  0.  0.  2. 12. 16. 15.  2.  0.]
```

Vector representation

Output space

Binary classification: $\mathscr{Y} = \{0,1\}$, $\mathscr{Y} = \{-1,+1\}$

Multiclass classification: $\mathscr{Y} = \{0,1,\ldots,k-1\}$

Regression: $\mathscr{Y} = \mathbb{R}$

Defining hypothesis spaces

- Hypotheses are functions that belong to a respective **hypothesis space**
  - ✓ space is defined by the machine learning technique, for example, decision trees, neural networks, support vector machines, etc.
- How to perform machine learning?
  - ✓ define the hypothesis space $\mathscr{H}$
  - ✓ find the best function within this space, $g \in \mathscr{H}$
    - ✓ a **loss function** is necessary to evaluate/compare hypotheses

Example

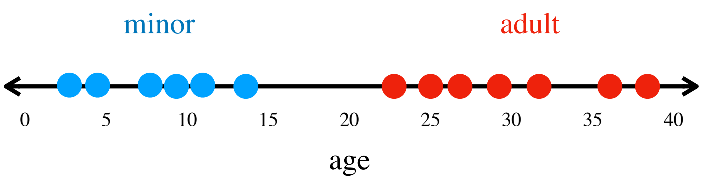
$h_1 \in \mathscr{H}$

$h_2 \in \mathscr{H}$

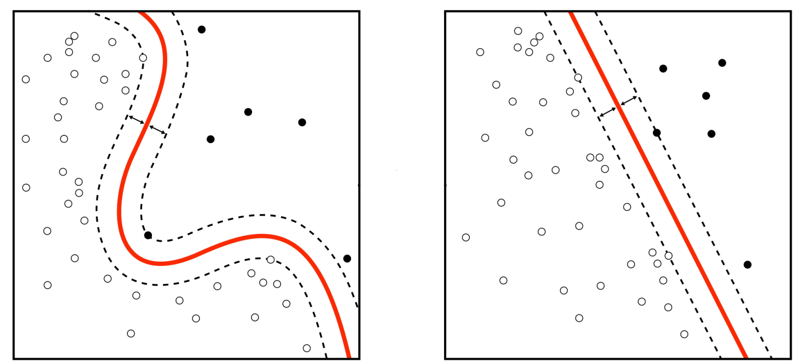
$\ldots$

can you define the hypothesis space?

how to pick a hypothesis that makes you happy?



Example of hypotheses



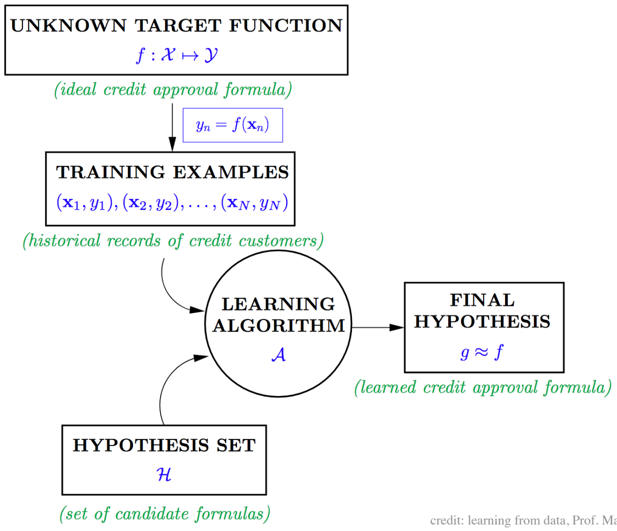
$h(\mathbf{x}) = \mathrm{sgn}\left(\mathbf{w}^T \mathrm{relu}\,(W\mathbf{x})\right)$

Hypothesis space: all possible single-layer neural nets

$h(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^T\mathbf{x})$

Hypothesis space: all possible linear functions

Learning setup



credit: learning from data, Prof. Malik Magdon-Ismail

Loss Functions

What is the goal of (**supervised**) learning?

- Finding a **hypothesis** (**classifier/regressor**) that best approximates the **target** function

For $g \in \mathcal{H}$ and $\forall (x_i, y_i) \sim P$, we want $g(x) \approx f(x)$

ML uses **search** and **optimization** (to **minimize expected loss**)

Expected Loss

$$\mathbb{E}\left[l(g, (x_i, y_i))\right]_{(x_i, y_i) \sim P}$$

We cannot calculate this term, but we can approximate it

Approximating the expected loss?

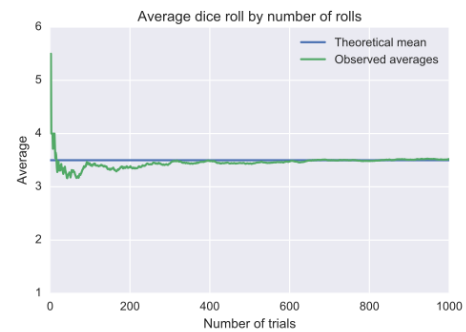
$$\mathbb{E}\left[l(g,(x_i,y_i))\right]_{(x_i,y_i)\sim P}$$

$$\approx \frac{1}{n}\sum_{i=1}^{n} l\left(g,(x_i,y_i)\right)$$

the **law of large numbers** states that the arithmetic mean of the values almost surely converges to the expected value as the number of repetitions approaches infinity

Law of large numbers

$$Pr\left(\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}x_n=\mathbb{E}[x]\right)=1$$



credit: wikipedia

0/1 loss

$$L_{0/1}(h,\mathscr{D})=\frac{1}{n}\sum_{(x_i,y_i)\in\mathscr{D}} I(h(x_i)\neq y_i)$$

indicator function

Prediction	Target
5	5
1	9
2	2
7	7
8	0
0	0
0	8
3	3
6	6
4	4

Squared loss

$$L_{sq}(h,\mathscr{D})=\frac{1}{n}\sum_{(x_i,y_i)\in\mathscr{D}}(h(x_i)-y_i)^2$$

positive loss and penalizes big mistakes

Prediction	Target
1.2	1.4
2.3	2.3
1.1	1.2
3.4	4.1
2.3	2.5
1.1	1.1
2.5	2.6
3.1	3.2
1.7	1.8
2.3	2.3

Absolute loss

$$L_{abs}(h, \mathscr{D}) = \frac{1}{n} \sum_{(x_i, y_i) \in \mathscr{D}} |h(x_i) - y_i|$$

Prediction	Target
1.2	1.4
2.3	2.3
1.1	1.2
3.4	4.1
2.3	2.5
1.1	1.1
2.5	2.6
3.1	3.2
1.7	1.8
2.3	2.3

Learning

- We can use a ML method to calculate:

$$g = \arg\min_{h \in \mathscr{H}} L(g, \mathscr{D})$$

- Problem: it may **overfit** the training data $\mathscr{D}$
- Solution: split your data in train, validation, test
  - ✓ use train and validation to select the best hypothesis
  - ✓ use test for final evaluation and report

Train, Validation, Test

Example using MNIST

https://colab.research.google.com/drive/1m_h-c2sSC4fNhRRNR2q-Dfk2ji5V6ILQ?usp=sharing