Supervised Learning

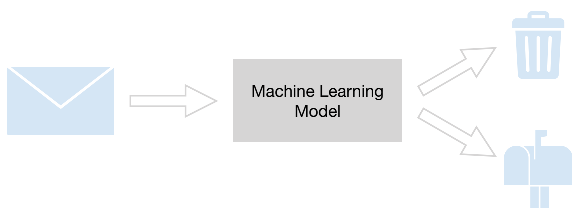
CSC 461: Machine Learning

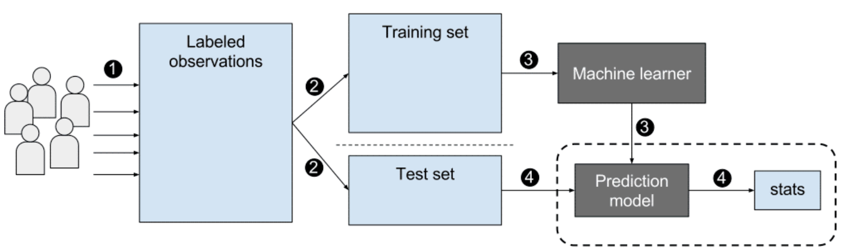
Fall 2021

Prof. Marco Alvarez
University of Rhode Island

Supervised Learning Setup

Spam filtering





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?

Components of (supervised) learning

- Input space $\mathcal{X}$
- Output space $\mathcal{Y}$
- Data instance $x \in \mathcal{X}, y \in \mathcal{Y}$
  - ✓ is a pair (x,y)
- Data $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$
  - ✓ is a set of data instances
- Hypothesis $g : \mathcal{X} \mapsto \mathcal{Y}, g \in \mathcal{H}$

Data

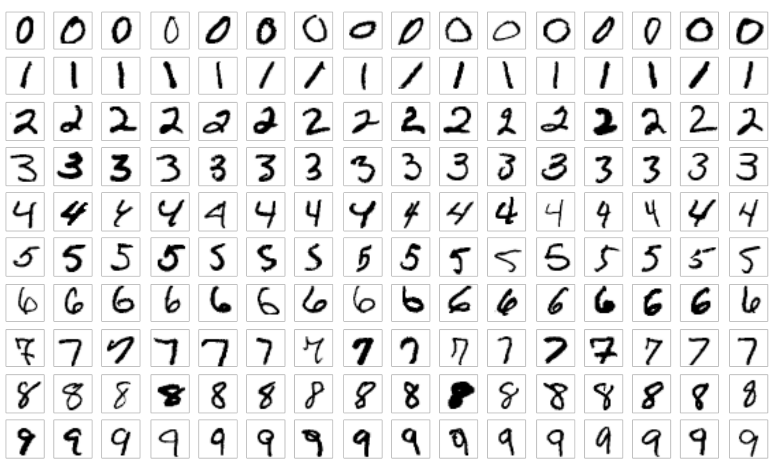
- Samples (data instances) are drawn from an **unknown distribution** $P(X, Y)$

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

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

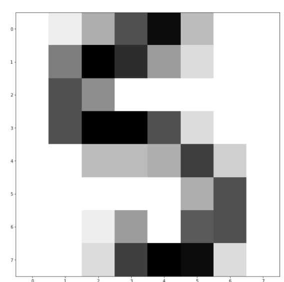
$$(x_i, y_i) \sim \underset{\text{unknown}}{P}$$

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.]
```

Feature vector

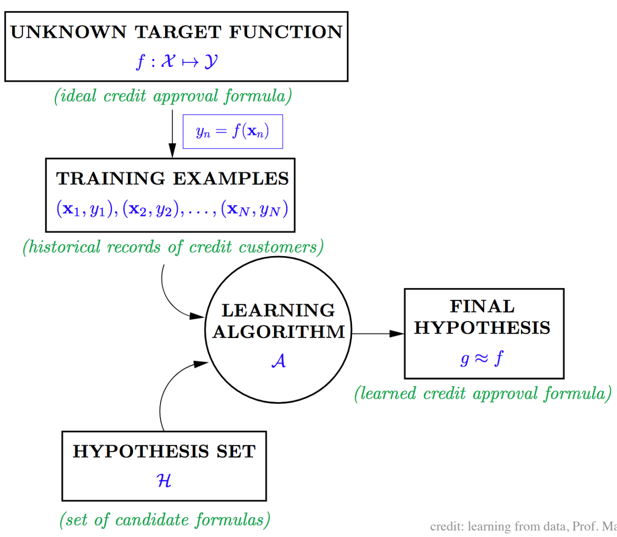
Feature Vectors

```
[[ 0.  0.  7.  16.  14.  13.  10.  0.  0.  0.  10.  12.  10.  16.  4.  0.  0.  0.  15.  5.  8.  13.  0.
0.  0.  1.  7.  1.  16.  3.  0.  0.  0.  2.  11.  13.  16.  12.  6.  0.  0.  4.  12.  15.  14.  11.  2.
0.  0.  0.  3.  16.  3.  0.  0.  0.  0.  0.  9.  13.  0.  0.  0.  0.]
[ 0.  0.  9.  16.  16.  16.  7.  0.  0.  3.  16.  11.  4.  4.  1.  0.  0.  6.  16.  1.  0.  0.  0.  0.
0.  9.  16.  9.  4.  0.  0.  0.  0.  0.  6.  10.  16.  8.  0.  0.  0.  0.  2.  0.  8.  14.  0.  0.  0.
0.  13.  7.  8.  14.  0.  0.  0.  0.  10.  16.  16.  4.  0.  0.]
[ 0.  0.  4.  15.  16.  16.  5.  0.  0.  0.  6.  9.  11.  16.  11.  0.  0.  0.  0.  0.  3.  16.  5.  0.
0.  0.  0.  3.  14.  16.  10.  0.  0.  0.  7.  16.  16.  11.  3.  0.  0.  0.  8.  15.  13.  0.  0.  0.
0.  0.  5.  16.  7.  0.  0.  0.  0.  0.  7.  14.  2.  0.  0.  0.]
[ 0.  1.  12.  16.  16.  16.  12.  0.  0.  9.  16.  13.  6.  8.  5.  0.  0.  8.  16.  15.  3.  0.  0.  0.
0.  0.  4.  14.  11.  0.  0.  0.  0.  0.  0.  12.  12.  0.  0.  0.  0.  0.  0.  12.  13.  0.  0.  0.  0.
0.  3.  15.  11.  0.  0.  0.  0.  0.  12.  13.  2.  0.  0.  0.]
[ 0.  0.  0.  16.  11.  0.  0.  0.  0.  0.  6.  16.  10.  0.  0.  0.  0.  0.  11.  11.  0.  0.  0.  0.
0.  0.  12.  15.  11.  5.  0.  0.  0.  0.  14.  15.  12.  15.  11.  0.  0.  0.  12.  13.  0.  0.  16.  5.
0.  0.  6.  15.  4.  11.  16.  4.  0.  0.  0.  13.  16.  14.  9.  0.]
[ 0.  0.  0.  12.  13.  5.  0.  0.  0.  0.  0.  11.  16.  9.  0.  0.  0.  0.  3.  15.  16.  6.  0.  0.
0.  7.  15.  16.  16.  2.  0.  0.  0.  0.  1.  16.  16.  3.  0.  0.  0.  0.  1.  16.  16.  6.  0.  0.  0.
0.  1.  16.  16.  6.  0.  0.  0.  0.  0.  11.  16.  10.  0.  0.]
[ 0.  0.  12.  10.  0.  0.  0.  0.  0.  0.  14.  16.  16.  14.  0.  0.  0.  0.  13.  16.  15.  10.  1.
0.  0.  0.  11.  16.  16.  7.  0.  0.  0.  0.  0.  4.  7.  16.  7.  0.  0.  0.  0.  0.  4.  16.  9.  0.
0.  0.  5.  4.  12.  16.  4.  0.  0.  0.  9.  16.  16.  10.  0.  0.]
[ 0.  0.  9.  15.  14.  2.  0.  0.  0.  0.  9.  3.  9.  8.  0.  0.  0.  0.  0.  0.  6.  10.  0.  0.  0.
0.  0.  10.  15.  2.  0.  0.  0.  0.  2.  10.  11.  15.  2.  0.  0.  3.  1.  0.  0.  14.  4.  0.  0.  10.
13.  7.  2.  12.  4.  0.  0.  0.  7.  14.  16.  10.  0.  0.]
[ 0.  0.  0.  9.  9.  0.  0.  0.  0.  0.  3.  15.  4.  0.  0.  0.  0.  0.  10.  12.  0.  0.  0.  0.  0.
0.  12.  8.  4.  3.  0.  0.  0.  0.  14.  16.  12.  14.  5.  0.  0.  0.  12.  10.  0.  4.  13.  0.  0.
0.  9.  11.  0.  6.  16.  1.  0.  0.  0.  8.  14.  15.  8.  0.]
[ 0.  2.  15.  16.  15.  2.  0.  0.  0.  8.  14.  8.  14.  8.  0.  0.  0.  7.  5.  2.  16.  5.  0.  0.
0.  0.  0.  12.  13.  0.  0.  0.  0.  0.  8.  15.  1.  0.  0.  0.  0.  1.  15.  7.  0.  0.  0.  0.  0.
4.  16.  9.  8.  8.  2.  0.  0.  2.  15.  16.  16.  16.  13.  0.]]
```

Supervised learning

Binary classification	$\mathscr{Y} = \{0,1\}$ $\mathscr{Y} = \{-1,+1\}$
Multiclass classification	$\mathscr{Y} = \{0,1,\ldots,k-1\}$
Regression	$\mathscr{Y} = \mathbb{R}$

Learning setup



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

Example

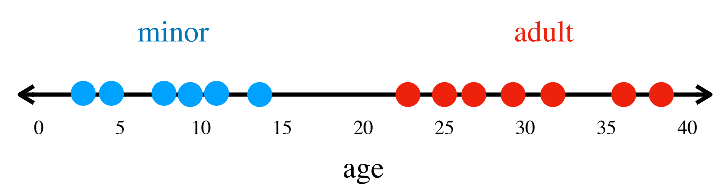
$h_1 \in \mathscr{H}$

$h_2 \in \mathscr{H}$

$\ldots$

can you define the hypothesis space?

how to pick a hypothesis that makes you happy?



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 learn?
  - ✓ define the hypothesis space $\mathscr{H}$
  - ✓ find the best function within this space, $h \in \mathscr{H}$
    - ✓ a **loss function** is necessary to evaluate/compare hypotheses

Loss Functions

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

What is the goal of (supervised) learning?

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

For $g \in \mathscr{H}$ 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

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

We cannot calculate this term, but we can approximate it

Approximating the expected loss?

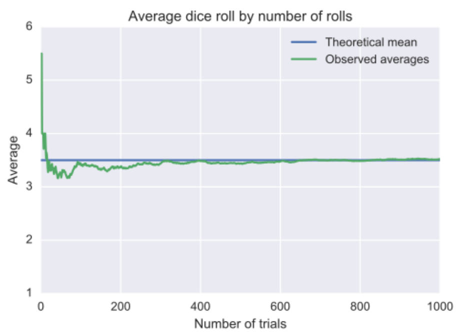
$$\mathbb{E}[l(g, (x_i, y_i))]_{(x_i, y_i) \sim P}$$

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

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

Generalization

- We can use a ML method to calculate:

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

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

Example using MNIST

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