

What is machine learning?

Trichotomy:

1. ML = statistics
2. ML = optimization
3. ML = modeling

Three components to any machine learning problem:

1. Task \longrightarrow model
2. Experience \longrightarrow data
3. Reward \longrightarrow loss

Together, these are fed into a learning algorithm which produces a solution to the task.

Examples of machine learning problems:

1. Chess \longleftarrow reinforcement learning
2. Medical diagnosis \longleftarrow classification
3. Real-estate pricing \longleftarrow regression
4. Machine translation \longleftarrow structured prediction
5. Document clustering \longleftarrow unsupervised learning

Important realization: how to map *your* problem into a known machine learning paradigm.

1. Reinforcement Learning

- (a) Agent, world, reward
- (b) Finite horizon: $\sum_{t=1}^T R(t)$
- (c) Infinite horizon: $\sum_{t=1}^{\infty} \gamma^t R(t)$, where $\gamma < 1$
- (d) Examples: block world, chess, robotic manipulation, taxi driving, elevator control, etc.

2. Classification (supervised learning)

- (a) Input / discrete output
- (b) Input is typically \mathbb{R}^D
- (c) Output is either $2 = \{0, 1\}$, $K = \{0, 1, \dots, K - 1\}$ or $\{-1, +1\}$ (for convenience)

- (d) Loss is usually $1[\hat{y} \neq y] = \delta_{\hat{y} \neq y} = \delta_{\hat{y} - y}$, where \hat{y} is predicted and y is truth
- (e) Examples: cancer prediction, document classification, image recognition, etc.

3. Regression (also supervised learning)

- (a) Input / *continuous* output
- (b) Input is typically \mathbb{R}^D
- (c) Output is either \mathbb{R}
- (d) Loss is usually half-squared loss: $\frac{1}{2}(y - \hat{y})^2$
- (e) Examples: Real-estate pricing, time prediction, weather prediction, stock prediction, etc.

4. Structured Prediction (also supervised learning)

- (a) Input / “large complex” output
- (b) Input is typically multiple \mathbb{R}^D s (can be considered just one)
- (c) Output is a (discrete) data structure
- (d) Loss is task specific (sometimes approximated by *Hamming* loss: $\sum_i 1[\hat{y}_i \neq y]$, with i ranging over “parts” of the structure)
- (e) Examples: sequence labeling, syntactic parsing, protein secondary-structure prediction, machine translation, etc.

5. Unsupervised Learning

- (a) Input is a data set (usually $(\mathbb{R}^D)^N$)
- (b) Output is a different representation of the input
- (c) Loss is task-specific
- (d) *No training data!*
- (e) Examples: document clustering, dimensionality reduction, feature selection (sort of), data visualization, language modeling, etc.

A simple classification example:

Length	Type
179cm	Ski
176cm	Ski
174cm	Ski
169cm	Ski
162cm	Ski
160cm	Ski
156cm	Ski
152cm	Ski
158cm	Snowboard
155cm	Snowboard
151cm	Snowboard
150cm	Snowboard
148cm	Snowboard
142cm	Snowboard
142cm	Snowboard
140cm	Snowboard

What “rule” would you use to make a prediction about type given length? Why not just memorize?

For an example like this, it makes sense to consider a *linear threshold predictor*:

$$\text{type} = \begin{cases} \text{ski} & \text{length} \geq \theta \\ \text{snowboard} & \text{length} < \theta \end{cases} \quad (1)$$

Task: choose θ . This is learning!

There’s no (theoretical) reason we couldn’t have used a *quadratic threshold predictor*:

$$\text{type} = \begin{cases} \text{ski} & \text{length} + \alpha \times (\text{length})^2 \geq \theta \\ \text{snowboard} & \text{length} + \alpha \times (\text{length})^2 < \theta \end{cases} \quad (2)$$

New task: choose θ and α . This is also learning!

Many other options are possible ...

A simple (perhaps unrealistic) regression example:

Square footage	Price
1200	\$120
1340	\$125
1390	\$105
1400	\$130
1420	\$135
1500	\$145
1550	\$160
1700	\$155
1900	\$140
2150	\$130
2300	\$135

Linear prediction model:

$$\text{price} \approx \theta_0 + \theta_1 \times \text{square footage} \quad (3)$$

Quadratic prediction model:

$$\text{price} \approx \theta_0 + \theta_1 \times (\text{sqft}) + \theta_2 \times (\text{sqft})^2 \quad (4)$$

K -th order prediction model:

$$\text{price} \approx \theta_0 + \theta_1 \times (\text{sqft}) + \dots + \theta_K \times (\text{sqft})^K \quad (5)$$

$$\approx \sum_{k=0}^K \theta_k \times (\text{sqft})^k \quad (6)$$

What K is preferable?

1. Too low K \leftarrow underfitting
2. Too high K \leftarrow overfitting

Regularization! Often tuned with cross-validation, development data, etc.

Learning theory: Can I guarantee how well I will do in prediction?

Learning algorithm \mathcal{A} takes training data $(x_1, y_1), \dots, (x_N, y_N)$ and makes a prediction \hat{y} about \hat{x} .

Can we make any guarantees about how well \mathcal{A} will do at this task?

Analyses are statistical: what is the probability \mathcal{A} will err? Or, what is its *expected error*?

Is there a “best” \mathcal{A} ?

What is this course all about?

1. How to identify learning problems. . .
2. How to specify a model. . .
3. How to find good values for parameters in the model. . .
4. How to select good features to enable learning. . .
5. How to evaluate the performance of learned systems. . .
6. How to transform learning problems. . .
7. How to evaluate the performance of learning algorithms. . .