

Machine Learning

Textbook: Machine Learning Refined, by Watt, Borhani, and Katsaggelos, 2nd edition:
MLRefined.com

§1.1 thru 1.4 - What Is Machine Learning?

Usually: we give a computer a rule.

Example: convert Celsius to Fahrenheit.

$$F = \frac{9}{5}C + 32.$$

The rule is known. The computer evaluates it.

Machine learning: **what if the rule is not known?**

Learning From Examples

Suppose we observe:

Hours studied x	1	2	3	4	5
Exam score y	52	61	67	78	85

We suspect a relationship between x and y .

But we do not know the rule.

Can the data suggest one?

Data

A supervised data set consists of input-output pairs:

$$(x_1, y_1), (x_2, y_2), \dots, (x_P, y_P).$$

x_p = input / feature.

y_p = desired output / target.

In many problems x_p is a vector:

$$\mathbf{x}_p = [x_{p1}, x_{p2}, \dots, x_{pN}]^T.$$

One observation can have many features.

A Model

We choose a family of possible rules.

For the study-time data, try a line:

$$f(x) = w_0 + w_1x.$$

w_0, w_1 are **parameters** (or weights).

Different parameters $\Rightarrow$ different models.

Example: $f(x) = 45 + 8x$.
 $f(3) = 45 + 8(3) = 69$.

Actual value: 67.

Not perfect — but close.

How Good Is a Model?

Prediction error for observation p :

$$e_p = f(x_p) - y_p.$$

For $(x,y) = (3,67)$ and $f(x) = 45 + 8x$:
 $e = 69 - 67 = 2$.

Adding errors is not enough:
 $10 + (-10) = 0$.

Square them instead.

A Cost Function

Mean squared error:

$$g(w_0, w_1) = \frac{1}{P} \sum_{p=1}^P (w_0 + w_1 x_p - y_p)^2.$$

Small $g \Rightarrow$ model fits the data well.

Large $g \Rightarrow$ poor fit.

Example: $f(x) = 45 + 8x$.
Predictions: 53, 61, 69, 77, 85.
Errors: 1, 0, 2, -1, 0.
 $g = \frac{1}{5}(1^2 + 0^2 + 2^2 + (-1)^2 + 0^2) = 1.2$.

Where Is the Learning?

We chose the form of the model:
 $f(x) = w_0 + w_1 x$.

The machine chooses the parameters.

Goal:
$$\underset{w_0, w_1}{\text{minimize}} \quad g(w_0, w_1).$$

Equivalently:
$$\mathbf{w}^* = \underset{\mathbf{w}}{\text{argmin}} \quad g(\mathbf{w}).$$

Machine learning often becomes an **optimization problem**.

Three Kinds of Problems

Regression: predict a number.
house price, temperature, exam score.

Classification: predict a category.
spam / not spam, benign / malignant.

Unsupervised learning: no target y is supplied.
Look for structure in the x 's themselves.
clustering, principal component analysis.

Fitting Is Not the Final Goal

A complicated model may fit the observed data perfectly.

But what happens on **new** data?

Training data = data used to learn the model.

Test data = new data used to evaluate it.

Good machine learning requires **generalization**.

Excellent training fit + poor new predictions = **overfitting**.

Our First Python

Python will be learned as we need it.

Today: store data, do arithmetic, plot points.

Import numerical and plotting tools:

```
import numpy as np
import matplotlib.pyplot as plt
```

Enter the data:

```
x = np.array([1, 2, 3, 4, 5])
y = np.array([52, 61, 67, 78, 85])
```

Plot the data:

```
plt.scatter(x, y)
plt.xlabel("Hours studied")
plt.ylabel("Exam score")
plt.show()
```

A Model in Python

Mathematics:

$$f(x) = w_0 + w_1x.$$

Python:

```
def model(x, w0, w1):  
    return w0 + w1*x
```

Try $w_0 = 45, w_1 = 8$:

```
y_pred = model(x, 45, 8)  
y_pred
```

Output:

```
[53, 61, 69, 77, 85].
```

Python is doing the arithmetic — not replacing the mathematics.

Cost in Python

```
def mse(y, y_pred):  
    return np.mean((y_pred - y)**2)
```

Then:

```
mse(y, model(x, 45, 8))
```

1.2.

Same calculation as before.

Try: can you find w_0, w_1 giving a cost smaller than 1.2?

Guessing works here. What if there are 10^6 parameters?

What did we learn?

- ♦ Machine learning uses data to choose a model's parameters.
- ♦ Supervised learning uses input-output pairs (x_p, y_p) .
- ♦ A cost function measures how well a model fits the data.
- ♦ Learning often means minimizing that cost function.
- ♦ Python will implement and visualize the mathematics.



Next

- We can measure the quality of a model with $g(\mathbf{w})$.
- Now we need to find parameter values that make g small.
- **1.2 - Learning as an Optimization Problem**