

Recitation 1

Supervised learning and linear regression

CS-UY 4563: Introduction to Machine Learning

Daniel Enriquez

Fall 2026

Learning from examples

- You take the same MTA route to class each morning.
- After a few weeks, you have a sense of when to leave.
- An unexpected delay reveals the expectation you had formed.

Suppose you record each trip: departure time, service alerts, arrival time.

What information have you kept, and what have you left out?

A dataset and a prediction task

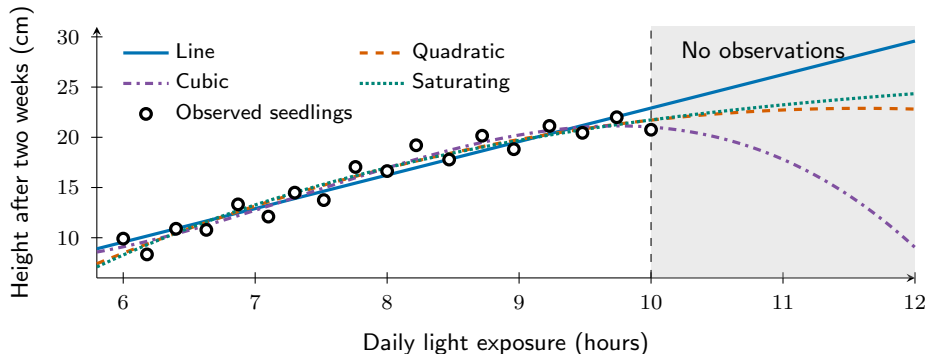
- A dataset is a collection of recorded observations.
- Features $x^{(i)}$: information available before the trip ends.
- Target $y^{(i)}$: the journey's duration.

$$\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^N, \quad \hat{y} = h(x).$$

- h is a **hypothesis function**. Its output $h(x)$ is the prediction.
- Supervised learning fits this function using input–target examples.
- Generalization means it remains useful on new observations.

Why choose a hypothesis class?

Eighteen noisy observations and four plausible hypothesis functions.



Synthetic data. Training RMSE: 0.90–1.15 cm. At 12 hours, predictions range from 9 to 30 cm.

Hypothesis class and inductive bias

The **hypothesis class** $\mathcal{H}$ is the set of functions we allow. With one feature, choosing straight lines means

$$\mathcal{H}_{\text{lin}} = \{h_w : h_w(x) = w_0 + w_1x, (w_0, w_1) \in \mathbb{R}^2\}.$$

- Fitting selects a function from this class using the observations.
- **Inductive priors:** assumptions and preferences that favor functions beyond what the training data alone justify. They shape our inductive bias.

What could justify preferring one curve when several fit the observations well?

Linear regression

Linear regression restricts its hypothesis class to linear models. With one feature, the graph is a straight line:

$$\hat{y} = h_w(x) = \underbrace{w_0}_{\text{intercept}} + \underbrace{w_1}_{\text{slope}} x.$$

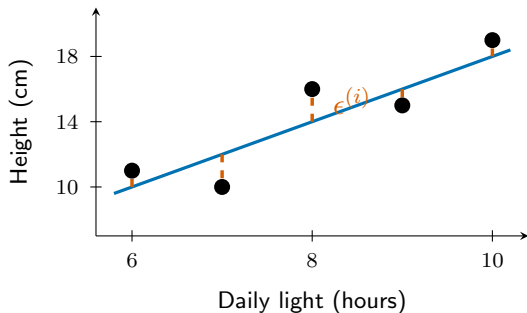
With several features, the contributions add:

$$h_w(x) = w_0 + w_1x_1 + \cdots + w_dx_d.$$

Two features give a plane. With d features, the graph is a hyperplane in $\mathbb{R}^{d+1}$.

With an intercept the function is affine in its inputs and linear in its coefficients.

Choosing a loss function



A **loss function** penalizes a prediction given its observed target.

$$\epsilon^{(i)} = y^{(i)} - h_w(x^{(i)})$$

For squared loss,

$$\mathcal{L}(y, \hat{y}) = (y - \hat{y})^2.$$

Twice the residual means four times the loss.

When should a large error count disproportionately?

Synthetic measurements. The loss and hypothesis class are separate choices.

Empirical risk and the fitting objective

Empirical risk is the average training loss:

$$\hat{R}_{\mathcal{D}}(h) = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(y^{(i)}, h(x^{(i)})).$$

Choose a hypothesis function in our class that minimizes it:

$$\hat{h} \in \arg \min_{h \in \mathcal{H}} \hat{R}_{\mathcal{D}}(h).$$

For squared loss, **RSS (residual sum of squares)** adds the squared prediction errors. Its average is the mean squared error.

The objective is what we minimize during fitting. A score reports performance. Small training loss alone does not establish generalization.

The design matrix

Include a constant feature $x_0^{(i)} = 1$ for the intercept.

$$X = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \vdots & & \vdots \\ 1 & x_1^{(N)} & \cdots & x_d^{(N)} \end{bmatrix} \in \mathbb{R}^{N \times (d+1)}.$$

- Rows are examples; columns are features.
- y collects the targets; Xw collects the predictions.

$$J(w) = \frac{\text{RSS}(w)}{2N} = \frac{1}{2N} \|y - Xw\|_2^2.$$

RSS adds the squared entries of $y - Xw$. The factor $1/2$ leaves the minimizing coefficients unchanged.

Gradient descent

- The gradient $\nabla J(w)$ measures how the objective responds to small changes in each coefficient.
- Moving against it decreases the objective for a sufficiently small step.

$$w^{[t+1]} = w^{[t]} - \alpha \nabla J(w^{[t]}), \quad \alpha > 0.$$

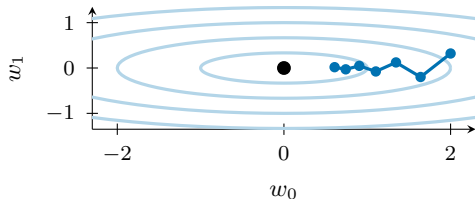
- Evaluate every coordinate using the same current parameter vector.
- The learning rate α controls the size of each update.

A direction is useful. How far should we move?

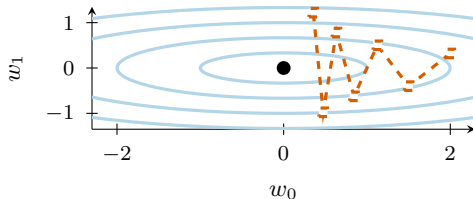
Curvature and the step size

A downhill direction can still lead to an overshoot.

Moderate step: $\alpha = 0.18$



Too large: $\alpha = 0.25$



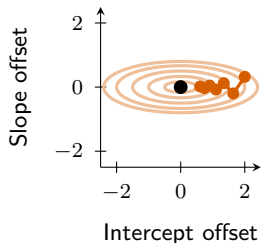
- A moderate step can zigzag toward the minimum.
- Too large a step can bounce farther away.

Contours join equal objective values. Same starting point, six updates. Center dot: minimum.

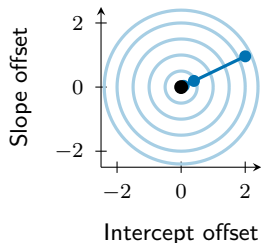
Why feature scaling helps

Feature scaling changes the geometry seen by gradient descent.

Before scaling: $\alpha = 0.18$



After scaling: $\alpha = 0.8$



Rounder contours let us take larger, more direct steps.

Same problem and starting predictions; six updates. Axes show offsets from each optimum.

Standardizing the features

Use each feature's training mean μ_j and standard deviation s_j :

$$\tilde{x}_j = \frac{x_j - \mu_j}{s_j}, \quad s_j > 0.$$

- Center the feature, then express deviations in standard-deviation units.
- With an intercept, rescaled coefficients preserve every prediction.
- Reuse the training μ_j and s_j for new observations.

Why can optimization change when the available predictions do not?

Standardization does not remove correlations, so it need not make every bowl circular.

What a prediction assumes

A new prediction depends on the data and our inductive priors.

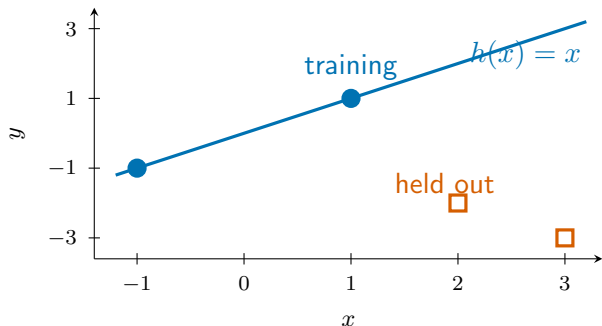
$$\hat{y}_* = \hat{h}(x_*), \quad \hat{h} \in \mathcal{H}.$$

- Features decide which distinctions the model can make.
- The linear class assumes additive effects with constant slopes.
- Fitting uses the data and loss to select a function within this class.

Which part came from the data, and which part did we assume?

Inductive priors can be structural assumptions; they need not be probability distributions.

Learning beyond the observed sample



- The training fit is exact.
- The held-out errors are large.
- New observations test whether the relationship persists.

Constructed example. Evaluation data should represent the cases on which the hypothesis function will be used.

Measuring the improvement: R^2

Compare with always predicting the evaluation set's mean target $\bar{y}$:

$$R^2 = 1 - \frac{\text{RSS}}{\text{TSS}}, \quad \text{TSS} = \sum_i (y^{(i)} - \bar{y})^2.$$

- $R^2 = 1$: perfect fit on this set.
- $R^2 = 0$: the same squared error as always predicting this set's mean.
- $R^2 < 0$: worse than that constant prediction.

RSS is the sum of squared residuals. Use one set throughout and require $\text{TSS} > 0$.

Constructing the columns

A new feature can make a curved relationship representable:

$$h_w(z) = w_0 + w_1 z + w_2 z^2, \quad X_{\text{quad}} = \begin{bmatrix} 1 & z^{(1)} & (z^{(1)})^2 \\ \vdots & \vdots & \vdots \\ 1 & z^{(N)} & (z^{(N)})^2 \end{bmatrix}.$$

The model is nonlinear in z but linear in its coefficients. Setting $w_2 = 0$ recovers every line, so

$$\mathcal{H}_{\text{lin}} \subseteq \mathcal{H}_{\text{quad}} \implies \min_{h \in \mathcal{H}_{\text{quad}}} \text{RSS}(h) \leq \min_{h \in \mathcal{H}_{\text{lin}}} \text{RSS}(h).$$

Training error cannot increase. Does performance on new data improve?

Beyond linear regression

Methods

Deep learning

Learn features with neural networks.

Self-supervised learning (SSL)

Predict missing or next parts of the data.

Reinforcement learning (RL)

Learn behavior using reward over time.

Model predictive control (MPC)

Plan with a model, act, then replan.

Applications

Genomics: predict gene activity from DNA.

Language models and protein representations.

Agents choosing sequences of actions.

Robotics, including control with learned dynamics.

Gradient-based learning: new models and objectives, a familiar update.

$$w \leftarrow w - \alpha \nabla J(w).$$

Course: neural networks, transformers, pretraining. Further directions: broader RL and MPC.

Discussion: can we learn without priors?

Suppose several hypothesis classes fit your recorded MTA trips equally well.

If the training data don't settle the choice, what should?

- Does a larger hypothesis class let us avoid assumptions?
- What prior does a linear model bring to this task?
- What evidence would make you revise that prior?

Discussion: what should the model get right?

An MTA travel-time model has low average squared loss, but often underestimates the longest trips.

Have we trained it to solve the problem we care about?

- Which errors should our loss penalize most?
- Could a different loss change the fit without changing $\mathcal{H}$?
- What should we report besides average error?

Discussion: which assumption would you change?

Your model predicts weekday trips well and fails on weekends.

Would you collect more data, change the features, or change $\mathcal{H}$?

- What are we assuming transfers from weekdays to weekends?
- What could more weekday data fix? What would it leave unresolved?
- What evidence would tell you which change to make?