Recap from previous lecture

- Learning is using past experience to improve future performance.
- Different types of learning:
  - supervised
  - unsupervised
  - reinforcement
  - active
  - online
  - ...
- For a machine, experience is encoded in data
Goal: teach a machine to distinguish handwritten 4's from handwritten 9's

\[ y = +1 \]

\[ y = -1 \]
**Supervised learning: example**

**Goal:** teach a machine to distinguish handwritten 4’s from handwritten 9’s

\[
y = +1 \\
\begin{array}{ccc}
4 & 4 & 0
\end{array}
\]

\[
y = -1 \\
\begin{array}{ccc}
9 & 9 & 9
\end{array}
\]

**Desired outcome:** a rule (function) for deciding whether a given \(10 \times 10\) pattern of black-and-white pixels is a 4 or a 9

\[
x \in \{0, 1\}^{100} \quad \rightarrow \quad \hat{y} \in \{-1, +1\}
\]
Supervised learning: example

**Goal:** teach a machine to distinguish handwritten 4’s from handwritten 9’s

\[
y = +1
\]

\[
y = -1
\]

**Desired outcome:** a rule (function) for deciding whether a given $10 \times 10$ pattern of black-and-white pixels is a 4 or a 9

\[
x \in \{0, 1\}^{100} \quad \longrightarrow \quad \hat{y} \in \{-1, +1\}
\]

**How to get there?** — assuming that the training data are representative of a stable probabilistic relationship between the input ($x$) and the output ($y$), do what you would do if you actually knew this relationship and could find an optimal predictor.

Figure taken from G. Shakhnarovich
Given:

\[
\begin{align*}
X & : \text{input (or feature)} \\
Y & : \text{output (or label)}
\end{align*}
\]

\[P_{XY} : \text{joint probability distribution}\]

Goal: find a function \( f \) that predicts \( Y \) given the value of \( X \)
Statistical decision theory: general framework

- **Given:**
  
  \[
  \begin{align*}
  &X & : & \text{input (or feature)} \\
  &Y & : & \text{output (or label)} \\
  &P_{XY} & : & \text{joint probability distribution}
  \end{align*}
  \]

- **Goal:** find a function \( f \) that predicts \( Y \) given the value of \( X \)

- **Evaluating predictors:** loss function

  \[
  \ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)
  \]
Statistical decision theory: general framework

- **Given:**
  \[
  \begin{aligned}
  &X : \text{input (or feature)} \\
  &Y : \text{output (or label)}
  \end{aligned}
  \quad P_{XY} : \text{joint probability distribution}
  \]

- **Goal:** find a function \( f \) that predicts \( Y \) given the value of \( X \)

- **Evaluating predictors:** loss function
  \[
  \ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)
  \]

- **Expected prediction error** of a candidate predictor \( f \):
  \[
  \mathcal{E}(f) = \mathbb{E}[\ell(Y, f(X))] \\
  = \int \ell(y, f(x)) P_{XY}(dx, dy)
  \]
Statistical decision theory: general framework

- **Given:**
  \[
  \begin{cases}
  X & \text{: input (or feature)} \\
  Y & \text{: output (or label)}
  \end{cases}
  \]
  \[P_{XY} : \text{joint probability distribution}\]

- **Goal:** find a function \( f \) that predicts \( Y \) given the value of \( X \)

- **Evaluating predictors:** loss function
  \[\ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)\]

- **Expected prediction error** of a candidate predictor \( f \):
  \[\mathcal{E}(f) = \mathbb{E}[\ell(Y, f(X))]\]
  \[= \int \ell(y, f(x)) P_{XY}(dx, dy)\]

- **Optimal predictor:**
  \[f^* = \arg\min_f \mathcal{E}(f), \quad \mathcal{E}^* \equiv \mathcal{E}(f^*)\]
Statistical decision theory: classification

- **Given:**

\[
\begin{cases}
X \in \mathbb{R}^p & \text{: feature} \\
Y \in \{0, 1\} & \text{: label} \\
\end{cases}
\]

\(P_{XY} \) : joint probability distribution

- **Goal:** find a function \( f : \mathbb{R}^p \rightarrow \{0, 1\} \) that accurately predicts the label \( Y \) given the feature \( X \).
Given:

\[
\begin{align*}
X \in \mathbb{R}^p &\quad : \text{feature} \\
Y \in \{0, 1\} &\quad : \text{label}
\end{align*}
\]

\[P_{XY} : \text{joint probability distribution}\]

Goal: find a function \( f : \mathbb{R}^p \rightarrow \{0, 1\} \) that accurately predicts the label \( Y \) given the feature \( X \)

Evaluating predictors: loss function

\[
\ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)
\]

For example, use 0-1 loss:

\[
\ell(Y, f(X)) = 1\{Y \neq f(X)\}
\]
Statistical decision theory: classification

Given:

\[
\begin{align*}
X \in \mathbb{R}^p & : \text{feature} \\
Y \in \{0, 1\} & : \text{label}
\end{align*}
\]

\( P_{XY} \) : joint probability distribution

Goal: find a function \( f : \mathbb{R}^p \rightarrow \{0, 1\} \) that accurately predicts the label \( Y \) given the feature \( X \)

Evaluating predictors: loss function

\[
\ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)
\]

For example, use 0-1 loss: \( \ell(Y, f(X)) = 1\{Y \neq f(X)\} \)

Expected prediction error of a candidate predictor \( f \):

\[
\mathcal{E}(f) = \mathbb{P}[Y \neq f(X)] = \int 1\{y \neq f(x)\} P_{XY}(dx, dy)
\]
How to pick $f$ to minimize $\mathcal{E}(f)$?
Minimizing expected classification error

How to pick $f$ to minimize $\mathcal{E}(f)$?

**Conditioning:**

$$\mathcal{E}(f) = \int 1_{\{y \neq f(x)\}} P_{XY}(dx, dy)$$

Maximize the inner integral for each fixed $x$ by setting $f^*(x) = \arg\max_{y \in \{0, 1\}} P[Y = y | X = x]$.

This is known as the Bayes classifier: classify to the most probable label given the observation $X = x$.

$\mathcal{E}(f^*)$ is known as the Bayes rate.
How to pick $f$ to minimize $\mathcal{E}(f)$?

**Conditioning:**

$$\mathcal{E}(f) = \int 1_{\{y \neq f(x)\}} P_{XY}(dx, dy)$$

$$= \int \left( \int 1_{\{y \neq f(x)\}} P_{Y|X}(dy|x) \right) P_X(dx)$$
Minimizing expected classification error

How to pick \( f \) to minimize \( \mathcal{E}(f) \)?

**Conditioning:**

\[
\mathcal{E}(f) = \int 1\{y \neq f(x)\} P_{XY}(dx, dy)
\]

\[
= \int \left( \int 1\{y \neq f(x)\} P_{Y|X}(dy|x) \right) P_X(dx)
\]

\[
= 1 - \int \left( \int 1\{y = f(x)\} P_{Y|X}(dy|x) \right) P_X(dx)
\]

Maximize the inner integral for each fixed \( x \) by setting \( f^*(x) = \text{arg max}_{y \in \{0, 1\}} P[Y = y | X = x] \).

This is known as the Bayes classifier: classify to the most probable label given the observation \( X = x \).

\( \mathcal{E}(f^*) \) is known as the Bayes rate.
Minimizing expected classification error

How to pick \( f \) to minimize \( \mathcal{E}(f) \)?

**Conditioning:**

\[
\mathcal{E}(f) = \int 1_{\{y \neq f(x)\}} P_{XY}(dx, dy) \\
= \int \left( \int 1_{\{y \neq f(x)\}} P_{Y|X}(dy|x) \right) P_X(dx) \\
= 1 - \int \left( \int 1_{\{y = f(x)\}} P_{Y|X}(dy|x) \right) P_X(dx) \\
= 1 - \int \left( \mathbb{P}[Y = 0|X = x] \cdot 1_{\{f(x) = 0\}} + \mathbb{P}[Y = 1|X = x] \cdot 1_{\{f(x) = 1\}} \right) P_X(dx)
\]
How to pick $f$ to minimize $\mathcal{E}(f)$?

**Conditioning:**

$$\mathcal{E}(f) = \int 1_{\{y \neq f(x)\}} P_{XY}(dx, dy)$$

$$= \int \left( \int 1_{\{y \neq f(x)\}} P_{Y|X}(dy|x) \right) P_X(dx)$$

$$= 1 - \int \left( \int 1_{\{y = f(x)\}} P_{Y|X}(dy|x) \right) P_X(dx)$$

$$= 1 - \int \left( \mathbb{P}[Y = 0|X = x] \cdot 1_{\{f(x) = 0\}} + \mathbb{P}[Y = 1|X = x] \cdot 1_{\{f(x) = 1\}} \right) P_X(dx)$$

Maximize the inner integral for each fixed $x$ by setting

$$f^*(x) = \arg\max_{y \in \{0, 1\}} \mathbb{P}[Y = y|X = x]$$

This is known as the **Bayes classifier**: classify to the most probable label given the observation $X = x$. $\mathcal{E}(f^*)$ is known as the **Bayes rate**.
The Bayes classifier

Classify to the most probable label:

\[ f^*(x) = \arg \max_{y \in \{0, 1\}} P[Y = y | X = x] \]

Define the regression function:

\[ \eta(x) \triangleq P[Y = 1 | X = x]. \]
The Bayes classifier

Classify to the most probable label:

\[ f^*(x) = \arg \max_{y \in \{0, 1\}} P[Y = y | X = x] \]

Define the **regression function**:

\[ \eta(x) \triangleq P[Y = 1 | X = x] \]

This leads to the following:

\[ f^*(x) = \begin{cases} 
1, & \text{if } \eta(x) \geq 1/2 \\
0, & \text{if } \eta(x) < 1/2 
\end{cases} \]

\[ E^* = \frac{1}{2} - \frac{1}{2} \int |1 - 2\eta(x)| P_X(dx) \]
Statistical decision theory: regression

- **Given:**
  \[ \begin{align*} 
  X \in \mathbb{R}^p & : \text{input variable} \\
  Y \in \mathbb{R} & : \text{output variable} 
  \end{align*} \]

- **Goal:** find a function \( f : \mathbb{R}^p \to \mathbb{R} \) that predicts \( Y \) given the value of \( X \)

- \( P_{XY} : \text{joint probability distribution} \)
Given:

\[
\begin{align*}
X & \in \mathbb{R}^p : \text{input variable} \\
Y & \in \mathbb{R} : \text{output variable}
\end{align*}
\]

\(P_{XY} : \text{joint probability distribution}\)

Goal: find a function \(f : \mathbb{R}^p \rightarrow \mathbb{R}\) that predicts \(Y\) given the value of \(X\)

Evaluating predictors: loss function

\[\ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)\]

For example, use squared error loss: \(\ell(Y, f(X)) = (Y - f(X))^2\)
Given:

\[
\begin{align*}
X & \in \mathbb{R}^p : \text{input variable} \\
Y & \in \mathbb{R} : \text{output variable}
\end{align*}
\]

\[P_{XY} : \text{joint probability distribution}\]

Goal: find a function \( f : \mathbb{R}^p \rightarrow \mathbb{R} \) that predicts \( Y \) given the value of \( X \)

Evaluating predictors: loss function

\[
\ell(Y, f(X)) = \text{loss (or penalty) for predicting the true } Y \text{ with } \hat{Y} = f(X)
\]

For example, use squared error loss: \(
\ell(Y, f(X)) = (Y - f(X))^2
\)

Expected prediction error of a candidate predictor \( f \):

\[
\mathcal{E}(f) = \mathbb{E} [(Y - f(X))^2] = \int (y - f(x))^2 P_{XY}(dx, dy)
\]
Minimizing mean squared error

How to pick $f$ to minimize $\mathcal{E}(f)$?
Minimizing mean squared error

How to pick $f$ to minimize $\mathcal{E}(f)$?

Conditioning:

$$
\mathcal{E}(f) = \int (y - f(x))^2 P_{XY}(dx, dy)
$$

$$
= \int \left( \int (y - f(x))^2 P_{Y|X}(dy|x) \right) P_X(dx)
$$

$$
\equiv \mathbb{E}[(Y - f(X))^2 | X = x]
$$
Minimizing mean squared error

How to pick $f$ to minimize $\mathcal{E}(f)$?

Conditioning:

$$
\mathcal{E}(f) = \int (y - f(x))^2 P_{XY}(dx, dy)
$$

$$
= \int \left( \int (y - f(x))^2 P_{Y|X}(dy|x) \right) P_X(dx)
\equiv \mathbb{E}[(Y - f(X))^2 | X = x]
$$

Minimize the inner integral for each fixed $x$ by setting

$$
f^*(x) = \arg \min_c \mathbb{E}[(Y - c)^2 | X = x]
$$

The solution is the regression function:

$$
f^*(x) = \mathbb{E}[Y | X = x]
$$
Proof that regression function is optimal

For any predictor $f$, write

$$E(f) = E[(Y - f(X))^2]$$

$$= E[(Y - f^*(X) + f^*(X) - f(X))^2]$$

$$= E[(Y - f^*(X))^2] + 2E[(Y - f^*(X))(f^*(X) - f(X))] + E[(f(X) - f^*(X))^2]$$

We now show that the middle term is zero:

$$E[(Y - f^*(X))(f^*(X) - f(X))] = E\hat{E}[Y - f^*(X)(f^*(X) - f(X))]$$

$$\equiv E(Y\mid X)\cdot (f^*(X) - f(X))$$

$$\equiv 0$$

Therefore, for any $f$ we have

$$E(f) = E(f^*) + E[(f(X) - f^*(X))^2]$$

and the term $E[(f(X) - f^*(X))^2]$ = 0 if and only if $f = f^*$. 

COMP 875  
Machine learning techniques in image analysis
Proof that regression function is optimal

For any predictor \( f \), write

\[
\mathcal{E}(f) = \mathbb{E}[(Y - f(X))^2]
\]

\[
= \mathbb{E}[(Y - f^*(X) + f^*(X) - f(X))^2]
\]

\[
= \mathbb{E}[(Y - f^*(X))^2] + 2 \mathbb{E}[(Y - f^*(X))(f^*(X) - f(X))] + \mathbb{E}[(f(X) - f^*(X))^2]
\]

We now show that the middle term is zero:

\[
\mathbb{E}[(Y - f^*(X))(f^*(X) - f(X))] = \mathbb{E} \left[ \mathbb{E} \left[ (Y - f^*(X))(f^*(X) - f(X)) | X \right] \right]
\]

\[
= \mathbb{E} \left[ \mathbb{E} \left[ Y - f^*(X) | X \right] \cdot (f^*(X) - f(X)) \right]
\]

\[
= \mathbb{E} \left[ \left( \mathbb{E}[Y | X] - f^*(X) \right) \cdot (f^*(X) - f(X)) \right]
\]

\[
\equiv 0
\]

Therefore, for any \( f \) we have

\[
\mathbb{E}(f) = \mathbb{E}(f^*) + \mathbb{E}[(f(X) - f^*(X))^2] \geq \mathbb{E}(f^*),
\]

and the term \( \mathbb{E}[(f(X) - f^*(X))^2] = 0 \) if and only if \( f = f^* \).
Proof that regression function is optimal

For any predictor $f$, write

$$\mathcal{E}(f) = \mathbb{E}[(Y - f(X))^2]$$

$$= \mathbb{E}[(Y - f^*(X) + f^*(X) - f(X))^2]$$

$$= \mathbb{E}[(Y - f^*(X))^2] + 2 \mathbb{E}[(Y - f^*(X))(f^*(X) - f(X))] + \mathbb{E}[(f(X) - f^*(X))^2]$$

We now show that the middle term is zero:

$$\mathbb{E}[(Y - f^*(X))(f^*(X) - f(X))] = \mathbb{E} \left[ \mathbb{E} \left[ (Y - f^*(X))(f^*(X) - f(X)) | X \right] \right]$$

$$= \mathbb{E} \left[ \mathbb{E} \left[ (Y - f^*(X)) | X \right] \cdot (f^*(X) - f(X)) \right]$$

$$= \mathbb{E} \left[ \left( \mathbb{E}[Y|X] - f^*(X) \right) \cdot (f^*(X) - f(X)) \right]$$

$$\equiv 0$$

Therefore, for any $f$ we have

$$\mathcal{E}(f) = \mathcal{E}(f^*) + \mathbb{E}[(f(X) - f^*(X))^2] \geq \mathcal{E}(f^*),$$

and the term $\mathbb{E}[(f(X) - f^*(X))^2] = 0$ if and only if $f = f^*$. 
Restricting the predictor

Sometimes, even if we know the distribution $P_{XY}$, computing the optimal regression function $f^*$ may not be worth it. In such cases, we may want to settle for the best predictor in some restricted class. This is one way to introduce inductive bias.

**Example: best linear predictor**

Suppose we are only interested in predictors of the form

$$f(x) = \beta^T x = \sum_{i=1}^{p} \beta(i) x(i)$$

where $\beta \in \mathbb{R}^p$ is some fixed vector of weights.

Then


Optimize by differentiating w.r.t. $\beta$ to get the optimal linear predictor:

$$\beta^* = (E[XX^T])^{-1} E[XY].$$
Restricting the predictor

Sometimes, even if we know the distribution $P_{XY}$, computing the optimal regression function $f^*$ may not be worth it. In such cases, we may want to settle for the best predictor in some restricted class. This is one way to introduce inductive bias.

**Example: best linear predictor**

Suppose we are only interested in predictors of the form

$$f(x) = \beta^T x = \sum_{i=1}^{p} \beta(i)x(i)$$

where $\beta \in \mathbb{R}^p$ is some fixed vector of weights.
Restricting the predictor

Sometimes, even if we know the distribution $P_{XY}$, computing the optimal regression function $f^*$ may not be worth it. In such cases, we may want to settle for the best predictor in some restricted class. This is one way to introduce inductive bias.

**Example: best linear predictor**

Suppose we are only interested in predictors of the form

$$f(x) = \beta^T x = \sum_{i=1}^{p} \beta(i)x(i)$$

where $\beta \in \mathbb{R}^p$ is some fixed vector of weights.

Then

$$\mathcal{E}(f) = \mathcal{E}(\beta) = \mathbb{E}[(Y - \beta^T X)^2]$$

$$= \mathbb{E}[Y^2] - 2\beta^T \mathbb{E}[YX] + \mathbb{E}[(\beta^T X)^2].$$

Optimize by differentiating w.r.t. $\beta$ to get the optimal linear predictor:

$$\beta^* = (\mathbb{E}[XX^T])^{-1} \mathbb{E}[XY]$$
Decomposition of the expected prediction error

What happens when we use restricted predictors?

Let $F$ be a restricted class of candidate predictors (e.g., linear or quadratic ones); define the best prediction error over $F$:

$$E^*_{F} = \min_{f \in F} E(f).$$

Then for any $f \in F$ we have

$$E(f) = E(f) - E^* + E^*_{\{z\}} = E(f) - E^*_{F} + E^*_{\{z\}} = T_1 + E^*_{F} - E^*_F$$

where:

- $T_1$ can be set to zero by letting $f$ achieve the minimum
- $T_2$ is large or small depending on how well the predictors in $F$ can do
What happens when we use restricted predictors?

Let $\mathcal{F}$ be a restricted class of candidate predictors (e.g., linear or quadratic ones); define the best prediction error over $\mathcal{F}$:

$$
\mathcal{E}^*_\mathcal{F} = \min_{f \in \mathcal{F}} \mathcal{E}(f).
$$

Then for any $f \in \mathcal{F}$ we have

$$
\mathcal{E}(f) = \mathcal{E}(f) - \mathcal{E}^*_\mathcal{F} + \mathcal{E}^*_\mathcal{F} |\{z\} = T_1 + \mathcal{E}^*_\mathcal{F} - \mathcal{E}^*_\mathcal{F} |\{z\} = T_2 + \mathcal{E}^*_\mathcal{F}
$$

where:

$T_1$ can be set to zero by letting $f$ achieve the minimum $\min_{f \in \mathcal{F}} \mathcal{E}(f)$;

$T_2$ is large or small depending on how well the predictors in $\mathcal{F}$ can do.
Decomposition of the expected prediction error

What happens when we use restricted predictors?

Let $\mathcal{F}$ be a restricted class of candidate predictors (e.g., linear or quadratic ones); define the best prediction error over $\mathcal{F}$:

$$
\mathcal{E}_\mathcal{F}^* = \min_{f \in \mathcal{F}} \mathcal{E}(f).
$$

Then for any $f \in \mathcal{F}$ we have

$$
\mathcal{E}(f) = \mathcal{E}(f) - \mathcal{E}_\mathcal{F}^* + \mathcal{E}_\mathcal{F}^*
= \mathcal{E}(f) - \mathcal{E}_\mathcal{F}^* + \mathcal{E}_\mathcal{F}^* - \mathcal{E}_\mathcal{F}^* + \mathcal{E}_\mathcal{F}^* + \mathcal{E}^*
= T_1 + T_2
$$

where:

- $T_1$ can be set to zero by letting $f$ achieve the minimum $\min_{f \in \mathcal{F}} \mathcal{E}(f)$
- $T_2$ is large or small depending on how well the predictors in $\mathcal{F}$ can do
To compute the best predictor $f^*$, you need to know the underlying distribution $P_{XY}$!

What should we do when we don’t know $P_{XY}$?
To compute the best predictor $f^*$, you need to know the underlying distribution $P_{XY}$!

What should we do when we don’t know $P_{XY}$?

**Empirical risk minimization** — If we can draw a large number $n$ of independent samples $(X_1, Y_1), \ldots, (X_n, Y_n)$ from $P_{XY}$, then we can find $f$ that minimizes the empirical prediction error

$$\hat{\mathcal{E}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i))$$

Why does it work? — Without getting into details, the reason is the Law of Large Numbers: for any $f$, $\hat{\mathcal{E}}(f) \approx \mathbb{E}(f)$ with high probability. It can then be proved that if $f^*$ minimizes the empirical prediction error, then

$$\mathbb{E}(f^*) \approx \mathbb{E}(f)$$

with high probability, where $f^*$ is the “true” optimal predictor (for $P_{XY}$).
To compute the best predictor $f^*$, you need to know the underlying distribution $P_{XY}$!

What should we do when we don't know $P_{XY}$?

**Empirical risk minimization** — If we can draw a large number $n$ of independent samples $(X_1, Y_1), \ldots, (X_n, Y_n)$ from $P_{XY}$, then we can find $f$ that minimizes the empirical prediction error

$$\hat{\mathcal{E}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i))$$

**Why does it work?** — Without getting into details, the reason is the Law of Large Numbers: for any $f$,

$$\hat{\mathcal{E}}(f) \approx \mathcal{E}(f) \quad \text{with high probability}$$
To compute the best predictor \( f^* \), you need to know the underlying distribution \( P_{XY} \)!

What should we do when we don’t know \( P_{XY} \)?

**Empirical risk minimization** — If we can draw a large number \( n \) of independent samples \((X_1, Y_1), \ldots, (X_n, Y_n)\) from \( P_{XY} \), then we can find \( f \) that minimizes the empirical prediction error

\[
\hat{\mathcal{E}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i))
\]

**Why does it work?** — Without getting into details, the reason is the Law of Large Numbers: for any \( f \),

\[
\hat{\mathcal{E}}(f) \approx \mathcal{E}(f) \quad \text{with high probability}
\]

It can then be proved that if \( \hat{f}^* \) minimizes the empirical prediction error, then

\[
\mathcal{E}(\hat{f}^*) \approx \mathcal{E}(f^*) \quad \text{with high probability}
\]

where \( f^* \) is the “true” optimal predictor (for \( P_{XY} \))
Example: least-squares linear regression

Given training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), find \(\hat{\beta}^* \in \mathbb{R}^p\) to minimize

\[
\hat{\mathcal{E}}(\beta) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \beta^T X_i)^2
\]
Example: least-squares linear regression

Given training data $(X_1, Y_1), \ldots, (X_n, Y_n)$, find $\beta^* \in \mathbb{R}^p$ to minimize

$$\hat{E}(\beta) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \beta^T X_i)^2$$

**Least-squares linear regression**

Define the $n \times p$ design matrix $X = [X_{ij}]$, $X_{ij} =$ $j$th component of $X_i$, and the response vector $Y = [Y_1 \ Y_2 \ldots \ Y_n]^T$. Then

$$\hat{E}(\beta) = \frac{1}{n} (Y - X\beta)^T (Y - X\beta)$$
Example: least-squares linear regression

Given training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), find \(\hat{\beta}^* \in \mathbb{R}^p\) to minimize

\[
\hat{E}(\beta) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \beta^T X_i)^2
\]

Least-squares linear regression

Define the \(n \times p\) design matrix \(X = [X_{ij}]\), \(X_{ij}\) = \(j\)th component of \(X_i\), and the response vector \(Y = [Y_1, Y_2, \ldots, Y_n]^T\). Then

\[
\hat{E}(\beta) = \frac{1}{n} (Y - X\beta)^T (Y - X\beta)
\]

Differentiate to get the normal equations:

\[
X^T(Y - X\beta) = 0
\]
Example: least-squares linear regression

Given training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), find \(\hat{\beta}^* \in \mathbb{R}^p\) to minimize

\[
\hat{\mathcal{E}}(\beta) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \beta^T X_i)^2
\]

**Least-squares linear regression**

Define the \(n \times p\) design matrix \(X = [X_{ij}]\), \(X_{ij} = j\)th component of \(X_i\), and the response vector \(Y = [Y_1 \ Y_2 \ \ldots \ Y_n]^T\). Then

\[
\hat{\mathcal{E}}(\beta) = \frac{1}{n} (Y - X\beta)^T (Y - X\beta)
\]

Differentiate to get the normal equations:

\[
X^T (Y - X\beta) = 0
\]

If \(X^T X\) is invertible, solve for the optimum:

\[
\hat{\beta}^* = (X^T X)^{-1} X^T Y
\]
Example: least-squares linear regression

Given training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), find \(\hat{\beta}^* \in \mathbb{R}^p\) to minimize

\[
\hat{\mathcal{E}}(\beta) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \beta^T X_i)^2
\]

Least-squares linear regression

Define the \(n \times p\) design matrix \(X = [X_{ij}]\), \(X_{ij}\) = \(j\)th component of \(X_i\), and the response vector \(Y = [Y_1 Y_2 \ldots Y_n]^T\). Then

\[
\hat{\mathcal{E}}(\beta) = \frac{1}{n} (Y - X\beta)^T (Y - X\beta)
\]

Differentiate to get the normal equations:

\[
X^T (Y - X\beta) = 0
\]

If \(X^TX\) is invertible, solve for the optimum:

\[
\hat{\beta}^* = (X^TX)^{-1}X^T Y
\]

Figure taken from Hastie, Tibshirani and Friedman, 2nd ed.
If we have training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), where \(X_i \in \mathbb{R}^p\) are features and \(Y_i \in \{0, 1\}\) are binary labels, we can attempt to find the best linear classifier.
Classification using linear regression

If we have training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), where \(X_i \in \mathbb{R}^P\) are features and \(Y_i \in \{0, 1\}\) are binary labels, we can attempt to find the best linear classifier.

Classification via regression

Find the least-squares linear regressor:

\[
\hat{\beta}^* = (X^T X)^{-1} X^T Y
\]
Classification using linear regression

If we have training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), where \(X_i \in \mathbb{R}^p\) are features and \(Y_i \in \{0, 1\}\) are binary labels, we can attempt to find the best linear classifier.

**Classification via regression**

Find the least-squares linear regressor:

\[
\hat{\beta}^* = (X^T X)^{-1} X^T Y
\]

Use it to build the classifier

\[
\hat{Y}(x) = \begin{cases} 
1, & \text{if } \hat{\beta}^T x \geq 1/2 \\
0, & \text{if } \hat{\beta}^T x < 1/2 
\end{cases}
\]
Classification using linear regression

If we have training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), where \(X_i \in \mathbb{R}^p\) are features and \(Y_i \in \{0, 1\}\) are binary labels, we can attempt to find the best linear classifier.

**Classification via regression**

Find the least-squares linear regressor:

\[
\hat{\beta}^* = (X^T X)^{-1} X^T Y
\]

Use it to build the classifier

\[
\hat{Y}(x) = \begin{cases} 
1, & \text{if } \hat{\beta}^T x \geq 1/2 \\
0, & \text{if } \hat{\beta}^T x < 1/2 
\end{cases}
\]

Figure taken from Hastie, Tibshirani and Friedman, 2nd ed.
Classification using linear regression

If we have training data \((X_1, Y_1), \ldots, (X_n, Y_n)\), where \(X_i \in \mathbb{R}^p\) are features and \(Y_i \in \{0, 1\}\) are binary labels, we can attempt to find the best linear classifier.

Classification via regression

Find the least-squares linear regressor:

\[
\hat{\beta}^* = (X^T X)^{-1} X^T Y
\]

Use it to build the classifier

\[
\hat{Y}(x) = \begin{cases} 
1, & \text{if } \hat{\beta}^T x \geq 1/2 \\
0, & \text{if } \hat{\beta}^T x < 1/2 
\end{cases}
\]

This will work well if the two classes are linearly separable (or almost so).
Nearest-neighbor methods use training samples closest to the test point $x$ to form $\hat{Y}$. These methods work well if the regression function is continuous.
Nearest-neighbor methods use training samples closest to the test point \( x \) to form \( \hat{Y} \). These methods work well if the regression function is continuous.

### Nearest-neighbor methods

**\( k \)-nearest-neighbor regression:**

\[
\hat{Y}(x) = \frac{1}{k} \sum_{X_i \in N_k(x)} Y_i
\]

**\( k \)-NN classification:**

\[
\hat{Y}(x) = \begin{cases} 
1, & \text{if } \sum_{X_i \in N_k(x)} Y_i \geq k/2 \\
0, & \text{if } \sum_{X_i \in N_k(x)} Y_i < k/2
\end{cases}
\]

\( N_k(x) \): the set of \( k \) nearest neighbors of \( x \) in the training sample
Nearest-neighbor methods use training samples closest to the test point $x$ to form $\hat{Y}$. These methods work well if the regression function is continuous.

**k-nearest-neighbor regression:**

$$\hat{Y}(x) = \frac{1}{k} \sum_{X_i \in N_k(x)} Y_i$$

**k-NN classification:**

$$\hat{Y}(x) = \begin{cases} 
1, & \text{if } \sum_{X_i \in N_k(x)} Y_i \geq k/2 \\
0, & \text{if } \sum_{X_i \in N_k(x)} Y_i < k/2 
\end{cases}$$

$N_k(x)$: the set of $k$ nearest neighbors of $x$ in the training sample.

Figure taken from Hastie, Tibshirani and Friedman, 2nd ed.
Issues and pitfalls

- **Curse of dimensionality**
  
  *Local methods (such as $k$-nearest-neighbors) require fairly dense training samples, which is hard to obtain when the problem dimension is very high.*

- **Bias-variance trade-off**
  
  *Using more complex models may lead to overfitting; using models that are overly simple may lead to underfitting. In both cases, we will have poor generalization.*
For nearest-neighbor methods, we need to capture a fixed fraction of the data (say, 10%) to form local averages. In high dimension, the data will be spread very thinly, and we will need to use almost the entire sample, which would conflict with the local nature of the method.
Bias and variance

Suppose the “true” model is

\[ Y = f^*(X) + \epsilon, \]

where \( \epsilon \) is noise, which is zero mean (\( \mathbb{E} \epsilon = 0 \)) and independent of \( X \). Let’s see how well a predictor \( \hat{f} \) learned from a training sample

\[ T = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \]

does on an independent test sample \( (X, Y) \):
Bias and variance

Suppose the “true” model is

\[ Y = f^*(X) + \epsilon, \]

where \( \epsilon \) is noise, which is zero mean (\( \mathbb{E} \epsilon = 0 \)) and independent of \( X \). Let’s see how well a predictor \( \hat{f} \) learned from a training sample

\[ T = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \]

does on an independent test sample \((X, Y)\):

\[
\mathbb{E}_T[(Y - \hat{f}(X))^2] = \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X) + \mathbb{E}_T \hat{f}(X) - \hat{f}(X))^2]
\]
Bias and variance

Suppose the “true” model is

\[ Y = f^*(X) + \epsilon, \]

where \( \epsilon \) is noise, which is zero mean (\( \mathbb{E} \epsilon = 0 \)) and independent of \( X \). Let’s see how well a predictor \( \hat{f} \) learned from a training sample

\[ T = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \]

does on an independent test sample \((X, Y)\):

\[
\mathbb{E}_T[(Y - \hat{f}(X))^2] = \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X) + \mathbb{E}_T \hat{f}(X) - \hat{f}(X))^2]
\]

\[
= \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X))^2] + 2 \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X)) (\mathbb{E}_T \hat{f}(X) - \hat{f}(X))]
\]

\[
= 0 + \mathbb{E}_T[(\hat{f}(X) - \mathbb{E}_T \hat{f}(X))^2]
\]

Bias: how well \( \hat{f} \) approximates the true regression function \( f^* \)

Variance: how much \( \hat{f}(X) \) wiggles around its mean
Bias and variance

Suppose the “true” model is

\[ Y = f^*(X) + \epsilon, \]

where \( \epsilon \) is noise, which is zero mean (\( \mathbb{E} \epsilon = 0 \)) and independent of \( X \). Let’s see how well a predictor \( \hat{f} \) learned from a training sample

\[ T = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \]

does on an independent test sample \((X, Y)\):

\[
\mathbb{E}_T[(Y - \hat{f}(X))^2] = \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X) + \mathbb{E}_T \hat{f}(X) - \hat{f}(X))^2] \\
= \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X))^2] + 2\mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X))(\mathbb{E}_T \hat{f}(X) - \hat{f}(X))] \\
+ \mathbb{E}_T[(\hat{f}(X) - \mathbb{E}_T \hat{f}(X))^2] \\
= [Y - \mathbb{E}_T \hat{f}(X)]^2 + \mathbb{E}_T [(\hat{f}(X) - \mathbb{E}_T \hat{f}(X))^2]
\]

**Bias:** how well \( \hat{f} \) approximates the true regression function \( f^* \)

**Variance:** how much \( \hat{f}(X) \) wiggles around its mean
Bias and variance

Suppose the “true” model is

\[ Y = f^*(X) + \epsilon, \]

where \( \epsilon \) is noise, which is zero mean (\( E \epsilon = 0 \)) and independent of \( X \). Let’s see how well a predictor \( \hat{f} \) learned from a training sample \( T = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \)

does on an independent test sample \((X, Y)\):

\[
E_T[(Y - \hat{f}(X))^2] = E_T[(Y - E_T \hat{f}(X) + E_T \hat{f}(X) - \hat{f}(X))^2] \\
= E_T[(Y - E_T \hat{f}(X))^2] + 2E_T[(Y - E_T \hat{f}(X))(E_T \hat{f}(X) - \hat{f}(X))] \\
= E_T[(Y - E_T \hat{f}(X))^2] + E_T[(\hat{f}(X) - E_T \hat{f}(X))^2] \\
= [Y - E_T \hat{f}(X)]^2 + E_T [(\hat{f}(X) - E_T \hat{f}(X))^2] \\
= O(\epsilon) + [f^*(X) - E_T \hat{f}(X)]^2 + E_T [(\hat{f}(X) - E_T \hat{f}(X))^2] \\
= \text{Bias}^2 + \text{Variance}
\]
Bias and variance

Suppose the "true" model is

\[ Y = f^*(X) + \epsilon, \]

where \( \epsilon \) is noise, which is zero mean (\( \mathbb{E} \epsilon = 0 \)) and independent of \( X \). Let's see how well a predictor \( \hat{f} \) learned from a training sample

\[ T = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \]

does on an independent test sample \( (X, Y) \):

\[
\mathbb{E}_T[(Y - \hat{f}(X))^2] = \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X) + \mathbb{E}_T \hat{f}(X) - \hat{f}(X))^2] \\
= \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X))^2] + 2 \mathbb{E}_T[(Y - \mathbb{E}_T \hat{f}(X)) (\mathbb{E}_T \hat{f}(X) - \hat{f}(X))] \\
= \mathbb{E}_T[(\hat{f}(X) - \mathbb{E}_T \hat{f}(X))^2] \\
= [Y - \mathbb{E}_T \hat{f}(X)]^2 + \mathbb{E}_T [(\hat{f}(X) - \mathbb{E}_T \hat{f}(X))^2] \\
= O(\epsilon) + [f^*(X) - \mathbb{E}_T \hat{f}(X)]^2 + \mathbb{E}_T [(\hat{f}(X) - \mathbb{E}_T \hat{f}(X))^2] \\
= \text{Bias}^2 + \text{Variance}
\]

- **Bias**: how well \( \hat{f} \) approximates the true regression function \( f^* \)
- **Variance**: how much \( \hat{f}(X) \) wiggles around its mean
There is a trade-off between the bias and the variance:

- “Complex” predictors will generally have low bias and high variance
- “Simple” predictors will generally have high bias and low variance

Figure taken from Hastie, Tibshirani and Friedman, 2nd ed.
One way to mitigate the bias-variance trade-off is to explicitly penalize predictors that are too complex.

\[
\text{minimize the complexity-regularized empirical prediction error}
\]

\[
\minimize \ E_{\text{cr}}(f) = E(f) + \lambda J(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i)) + \lambda J(f)
\]

where:
- The regularizer \( J(f) \) is high for complicated predictors \( f \) and low for simple predictors \( f \).
- The regularization constant \( \lambda > 0 \) controls the balance between the data fit term \( E(f) \) and the complexity term \( J(f) \).
One way to mitigate the bias-variance trade-off is to explicitly penalize predictors that are too complex.

Instead of finding the minimizer of the empirical prediction error

$$\hat{E}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i)),$$

minimize the complexity-regularized empirical prediction error

$$\hat{E}_{cr}(f) = \hat{E}(f) + \lambda J(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i)) + \lambda J(f),$$

where:

- The regularizer $J(f)$ is high for complicated predictors $f$ and low for simple predictors $f$
- The regularization constant $\lambda > 0$ controls the balance between the data fit term $\hat{E}(f)$ and the complexity term $J(f)$
Examples of regularization

- **Roughness penalty:** $\mathcal{F}$ is the class of twice differentiable functions, and

  $$ J(f) = \int |f''(x)|^2 \, dx $$

  This penalizes functions that “wiggle around” too much. The complexity-regularized solution is the **smoothing spline**.
Examples of regularization

- **Roughness penalty**: $\mathcal{F}$ is the class of twice differentiable functions, and

  $$J(f) = \int |f''(x)|^2 dx$$

  This penalizes functions that “wiggle around” too much. The complexity-regularized solution is the smoothing spline.

- **Subset selection**: $\mathcal{F}$ is the class of linear regression functions $f(x) = \beta^T x$, and

  $$J(\beta) = \|\beta\|_0 = \text{the number of nonzero components of } \beta$$

  This is effective, but may be computationally expensive.
Examples of regularization

- **Roughness penalty:** $\mathcal{F}$ is the class of twice differentiable functions, and

  $$J(f) = \int |f''(x)|^2 dx$$

  This penalizes functions that “wiggle around” too much. The complexity-regularized solution is the smoothing spline.

- **Subset selection:** $\mathcal{F}$ is the class of linear regression functions $f(x) = \beta^T x$, and

  $$J(\beta) = \|\beta\|_0 = \text{the number of nonzero components of } \beta$$

  This is effective, but may be computationally expensive.

- **$\ell_1$-penalty:** $\mathcal{F}$ is again the class of linear regression functions, and

  $$J(\beta) = \|\beta\|_1 = \sum_{i=1}^{p} |\beta(i)|$$

  This often serves as a reasonable tractable relaxation of the subset selection penalty. This approach is known as the LASSO, and is very popular right now.
Next time, we will focus on linear methods for classification:

- Logistic regression
- Separating hyperplanes and the Perceptron