COMS 4771 Lecture 2

1. Classification problems
2. Classifiers via generative models
3. Evaluating classifiers

Classification problems

Terminology and notation

- Recall: $\mathcal{X}$ is the **input space**, and $\mathcal{Y}$ is the **output space**.

Terminology and notation

- Recall: $\mathcal{X}$ is the **input space**, and $\mathcal{Y}$ is the **output space**.
- In classification problems, output space $\mathcal{Y}$ comprised of K possible **classes** (or **categories**).

  For simplicity, we'll just call them $\mathcal{Y} := \{1, 2, \ldots, K\}$.

  (When $K = 2$, we typically use $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, +1\}$.)

Terminology and notation

- Recall: $\mathcal{X}$ is the **input space**, and $\mathcal{Y}$ is the **output space**.
- In classification problems, output space $\mathcal{Y}$ comprised of K possible **classes** (or **categories**).
  For simplicity, we'll just call them $\mathcal{Y} := \{1, 2, \ldots, K\}$.
  (When $K = 2$, we typically use $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, +1\}$.)
- **Labeled example**: $(x, y) \in \mathcal{X} \times \mathcal{Y}$.
  Interpretation:
  - x represents the description or measurements of an object;
  - y is the category to which that object belongs.

Terminology and notation

- Recall: $\mathcal{X}$ is the **input space**, and $\mathcal{Y}$ is the **output space**.
- In classification problems, output space $\mathcal{Y}$ comprised of K possible **classes** (or **categories**).

  For simplicity, we'll just call them $\mathcal{Y} := \{1, 2, \ldots, K\}$.

  (When $K = 2$, we typically use $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, +1\}$.)
- **Labeled example**: $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

  Interpretation:
  - x represents the description or measurements of an object;
  - y is the category to which that object belongs.
- **Task**: using **labeled examples** $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$, construct **classifier** $\hat{f} \colon \mathcal{X} \to \mathcal{Y}$ that usually predicts the correct class label.

Terminology and notation

- Recall: $\mathcal{X}$ is the **input space**, and $\mathcal{Y}$ is the **output space**.
- In classification problems, output space $\mathcal{Y}$ comprised of K possible **classes** (or **categories**).
  For simplicity, we'll just call them $\mathcal{Y} := \{1, 2, \ldots, K\}$.
  (When $K = 2$, we typically use $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, +1\}$.)
- **Labeled example**: $(x, y) \in \mathcal{X} \times \mathcal{Y}$.
  Interpretation:
  - x represents the description or measurements of an object;
  - y is the category to which that object belongs.
- **Task**: using **labeled examples** $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$, construct **classifier** $\hat{f} \colon \mathcal{X} \to \mathcal{Y}$ that usually predicts the correct class label.
- **Note**: possible to see both $(x, 1)$ and $(x, 2)$ for same input x.
  (Why is this realistic?)

Statistical setting

How do we say how good a classifier is?

- Assume there's a **distribution P over space of labeled examples** $\mathcal{X} \times \mathcal{Y}$.

How do we say how good a classifier is?

- Assume there's a **distribution P over space of labeled examples** $\mathcal{X} \times \mathcal{Y}$.
- P is *unknown* (e.g., we don't know its functional form), but it represents the population we care about.

Statistical setting

How do we say how good a classifier is?

- Assume there's a **distribution** P **over space of labeled examples** $\mathcal{X} \times \mathcal{Y}$.
- P is *unknown* (e.g., we don't know its functional form), but it represents the population we care about.
- For any classifier $f \colon \mathcal{X} \to \mathcal{Y}$, we care about its **prediction accuracy**:

$$\Pr[f(X) = Y].$$

where $(X, Y) \sim P$.
[**Notation**: "$(X, Y) \sim P$" means that inside the argument to $\Pr[\,\cdot\,]$, (X, Y) is a ($\mathcal{X} \times \mathcal{Y}$-valued) random variable with distribution P.]

Statistical setting

How do we say how good a classifier is?

- Assume there's a **distribution** P **over space of labeled examples** $\mathcal{X} \times \mathcal{Y}$.
- P is *unknown* (e.g., we don't know its functional form), but it represents the population we care about.
- For any classifier $f \colon \mathcal{X} \to \mathcal{Y}$, we care about its **prediction accuracy**:

$$\Pr[f(X) = Y].$$

  where $(X, Y) \sim P$.
  [**Notation**: "$(X, Y) \sim P$" means that inside the argument to $\Pr[\,\cdot\,]$, (X, Y) is a ($\mathcal{X} \times \mathcal{Y}$-valued) random variable with distribution P.]
- **Prediction error**:

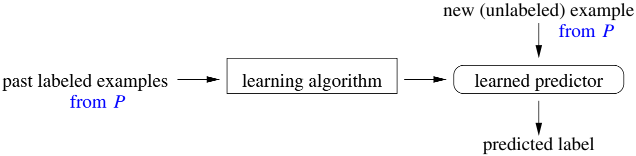
$$\mathrm{err}(f) := \Pr[f(X) \neq Y].$$

Statistical learning

- When is there any hope for finding a classifier with high accuracy?

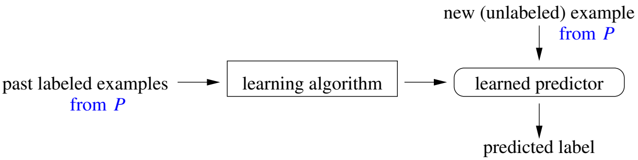
Statistical learning

- When is there any hope for finding a classifier with high accuracy?
- **Key assumption**: Data $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$ are i.i.d. random labeled examples with distribution P—i.e., an i.i.d. sample from P.



This assumption is the connection between what we've seen in the past to what we expect to see in the future.

- When is there any hope for finding a classifier with high accuracy?
- **Key assumption**: Data $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$ are i.i.d. random labeled examples with distribution P—i.e., an i.i.d. sample from P.



This assumption is the connection between what we've seen in the past to what we expect to see in the future.

What's next:

- What does an accurate classifier look like?
- How do we exploit the key assumption to construct an accurate classifier?

What is the optimal classifier?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.

What is the optimal classifier?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.
- Nature draws $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ according to the above probability distribution.
- We only observe $X \in \mathcal{X}$ (i.e. cloudy or sunny)
- Given X, want to predict Y (i.e. rain or no rain) most accurately.
- Most accurately $=$ least probability of error

What is the optimal classifier?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.
- Nature draws $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ according to the above probability distribution.
- We only observe $X \in \mathcal{X}$ (i.e. cloudy or sunny)
- Given X, want to predict Y (i.e. rain or no rain) most accurately.
- Most accurately $=$ least probability of error
- Question: what is the optimal prediction if it is cloudy?

What is the optimal classifier?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.
- Nature draws $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ according to the above probability distribution.
- We only observe $X \in \mathcal{X}$ (i.e. cloudy or sunny)
- Given X, want to predict Y (i.e. rain or no rain) most accurately.
- Most accurately $=$ least probability of error
- Question: what is the optimal prediction if it is cloudy? rain

What is the optimal classifier?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.
- Nature draws $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ according to the above probability distribution.
- We only observe $X \in \mathcal{X}$ (i.e. cloudy or sunny)
- Given X, want to predict Y (i.e. rain or no rain) most accurately.
- Most accurately $=$ least probability of error
- Question: what is the optimal prediction if it is cloudy? rain
- Question: what is the optimal prediction if it is sunny?

WHAT IS THE OPTIMAL CLASSIFIER?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.
- Nature draws $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ according to the above probability distribution.
- We only observe $X \in \mathcal{X}$ (i.e. cloudy or sunny)
- Given X, want to predict Y (i.e. rain or no rain) most accurately.
- Most accurately $=$ least probability of error
- Question: what is the optimal prediction if it is cloudy? rain
- Question: what is the optimal prediction if it is sunny? no rain

What is the optimal classifier?

- Example: $\mathcal{X} = \{\text{cloudy}, \text{sunny}\}$ is state of the sky, and $\mathcal{Y} = \{\text{rain}, \text{no rain}\}$ is the weather
- If it is cloudy, probability of rain is 0.7.
- If it is sunny, probability of rain is 0.05.
- Nature draws $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ according to the above probability distribution.
- We only observe $X \in \mathcal{X}$ (i.e. cloudy or sunny)
- Given X, want to predict Y (i.e. rain or no rain) most accurately.
- Most accurately $=$ least probability of error
- Question: what is the optimal prediction if it is cloudy? rain
- Question: what is the optimal prediction if it is sunny? no rain
- Optimal prediction rule:
  for each $\boldsymbol{x} \in \mathcal{X}$, chose $y \in \mathcal{Y}$ with maximum $\Pr[Y = y \mid X = \boldsymbol{x}]$.

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes $\Pr[f(X) \neq Y]$

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes $\Pr[f(X) \neq Y] = \sum_{\boldsymbol{x} \in \mathcal{X}} \Pr[f(X) \neq Y | X = \boldsymbol{x}] \cdot \Pr[X = \boldsymbol{x}]$

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes $\Pr[f(X) \neq Y] = \sum_{\boldsymbol{x} \in \mathcal{X}} \Pr[f(X) \neq Y | X = \boldsymbol{x}] \cdot \Pr[X = \boldsymbol{x}]$
- For every $\boldsymbol{x} \in \mathcal{X}$, set $f(\boldsymbol{x})$ to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$.

WHAT IS THE OPTIMAL CLASSIFIER?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes
  $\Pr[f(X) \neq Y] = \sum_{\boldsymbol{x} \in \mathcal{X}} \Pr[f(X) \neq Y | X = \boldsymbol{x}] \cdot \Pr[X = \boldsymbol{x}]$
- For every $\boldsymbol{x} \in \mathcal{X}$, set $f(\boldsymbol{x})$ to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$.
- $\Pr[f(X) \neq Y | X = \boldsymbol{x}] = 1 - \Pr[f(X) = Y | X = \boldsymbol{x}]$

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes $\Pr[f(X) \neq Y] = \sum_{\boldsymbol{x} \in \mathcal{X}} \Pr[f(X) \neq Y | X = \boldsymbol{x}] \cdot \Pr[X = \boldsymbol{x}]$
- For every $\boldsymbol{x} \in \mathcal{X}$, set $f(\boldsymbol{x})$ to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$.
- $\Pr[f(X) \neq Y | X = \boldsymbol{x}] = 1 - \Pr[f(X) = Y | X = \boldsymbol{x}]$
- So, to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$, maximize $\Pr[f(X) = Y | X = \boldsymbol{x}]$.

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes $\Pr[f(X) \neq Y] = \sum_{\boldsymbol{x} \in \mathcal{X}} \Pr[f(X) \neq Y | X = \boldsymbol{x}] \cdot \Pr[X = \boldsymbol{x}]$
- For every $\boldsymbol{x} \in \mathcal{X}$, set $f(\boldsymbol{x})$ to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$.
- $\Pr[f(X) \neq Y | X = \boldsymbol{x}] = 1 - \Pr[f(X) = Y | X = \boldsymbol{x}]$
- So, to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$, maximize $\Pr[f(X) = Y | X = \boldsymbol{x}]$.
- I.e. choose $f(\boldsymbol{x}) = \arg\max_{y \in Y} \Pr[Y = y | X = \boldsymbol{x}]$.

What is the optimal classifier?

- Assume $\mathcal{X}$ is discrete for now.
- Suppose $(X, Y) \sim P$.
- Want to find $f : \mathcal{X} \to \mathcal{Y}$ that minimizes $\Pr[f(X) \neq Y] = \sum_{\boldsymbol{x} \in \mathcal{X}} \Pr[f(X) \neq Y | X = \boldsymbol{x}] \cdot \Pr[X = \boldsymbol{x}]$
- For every $\boldsymbol{x} \in \mathcal{X}$, set $f(\boldsymbol{x})$ to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$.
- $\Pr[f(X) \neq Y | X = \boldsymbol{x}] = 1 - \Pr[f(X) = Y | X = \boldsymbol{x}]$
- So, to minimize $\Pr[f(X) \neq Y | X = \boldsymbol{x}]$, maximize $\Pr[f(X) = Y | X = \boldsymbol{x}]$.
- I.e. choose $f(\boldsymbol{x}) = \arg\max_{y \in Y} \Pr[Y = y | X = \boldsymbol{x}]$.

The classifier f defined above is called the **Bayes classifier**.

Structure of the Bayes classifier

By Bayes' rule:

$$\Pr\big[Y = y \,\big|\, X = x\big] = \frac{\Pr[Y = y] \cdot \Pr\big[X = x \,\big|\, Y = y\big]}{\Pr[X = x]}.$$

Structure of the Bayes classifier

By Bayes' rule:

$$\Pr\big[Y = y \,\big|\, X = x\big] = \frac{\Pr[Y = y] \cdot \Pr\big[X = x \,\big|\, Y = y\big]}{\Pr[X = x]}.$$

Since $\Pr[X = x]$ does not depend on y, the Bayes classifier is

$$f^\star(x) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \; \Pr[Y = y] \cdot \Pr\big[X = x \,\big|\, Y = y\big].$$

Structure of the Bayes classifier

By Bayes' rule:

$$\Pr\left[Y = y \,\middle|\, X = x\right] = \frac{\Pr[Y = y] \cdot \Pr\left[X = x \,\middle|\, Y = y\right]}{\Pr[X = x]}.$$

Since $\Pr[X = x]$ does not depend on y, the Bayes classifier is

$$f^\star(x) = \underset{y \in \mathcal{Y}}{\arg\max}\ \Pr[Y = y] \cdot \Pr\left[X = x \,\middle|\, Y = y\right].$$

- $\Pr[Y = \cdot]$ (i.e., the marginal distribution of Y) is called the **class prior**.
- $\Pr[X = \cdot \mid Y = y]$ is called the **class conditional distribution** of X (for class y).

Structure of the Bayes classifier

By Bayes' rule:

$$\Pr\bigl[Y = y \bigm| X = x\bigr] = \frac{\Pr[Y = y] \cdot \Pr\bigl[X = x \bigm| Y = y\bigr]}{\Pr[X = x]}.$$

Since $\Pr[X = x]$ does not depend on y, the Bayes classifier is

$$f^\star(x) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \; \Pr[Y = y] \cdot \Pr\bigl[X = x \bigm| Y = y\bigr].$$

- $\Pr[Y = \cdot]$ (i.e., the marginal distribution of Y) is called the **class prior**.
- $\Pr[X = \cdot \mid Y = y]$ is called the **class conditional distribution** of X (for class y).

If X has a probability density (rather than a probability mass function), replace $\Pr[X = \cdot \mid Y = y]$ with **class conditional density** $p_y(\cdot)$.

Example: Gaussian class conditional densities

Suppose $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{0, 1\}$, and the distribution P of (X, Y) is as follows.

- **Class prior**:

$$\Pr[Y = y] = \pi_y, \quad y \in \{0, 1\}$$

  for some real numbers $\pi_0, \pi_1 \in [0, 1]$ satisfying $\pi_0 + \pi_1 = 1$.

Example: Gaussian class conditional densities

Suppose $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{0, 1\}$, and the distribution P of (X, Y) is as follows.

- **Class prior**:

  $$\Pr[Y = y] = \pi_y, \quad y \in \{0, 1\}$$

  for some real numbers $\pi_0, \pi_1 \in [0, 1]$ satisfying $\pi_0 + \pi_1 = 1$.

- **Class conditional density** for class $y \in \{0, 1\}$:

  $$p_y(x) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x - \mu_y)^2}{2\sigma_y^2}\right)$$

  for some $\mu_y \in \mathbb{R}$ and $\sigma_y^2 > 0$ (i.e., $\mathrm{N}(\mu_y, \sigma_y^2)$).

Example: Gaussian class conditional densities

Suppose $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{0, 1\}$, and the distribution P of (X, Y) is as follows.

- **Class prior**:

$$\Pr[Y = y] = \pi_y, \quad y \in \{0, 1\}$$

  for some real numbers $\pi_0, \pi_1 \in [0, 1]$ satisfying $\pi_0 + \pi_1 = 1$.

- **Class conditional density** for class $y \in \{0, 1\}$:

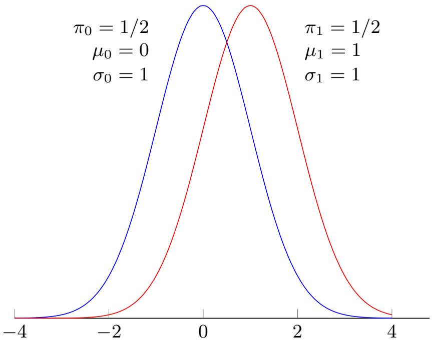
$$p_y(x) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x - \mu_y)^2}{2\sigma_y^2}\right)$$

  for some $\mu_y \in \mathbb{R}$ and $\sigma_y^2 > 0$ (i.e., $\mathrm{N}(\mu_y, \sigma_y^2)$).

- **Bayes classifier**:

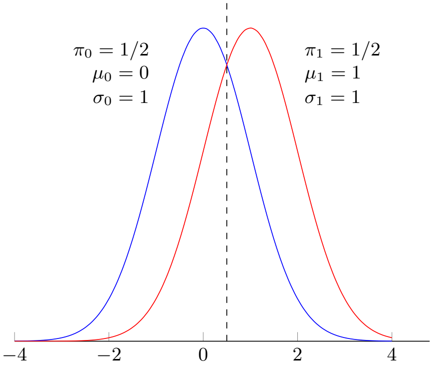
$$\begin{aligned} f^\star(x) &= \operatorname*{arg\,max}_{y \in \{0,1\}} \Pr\left[Y = y \,\middle|\, X = x\right] \\ &= \begin{cases} 1 & \text{if } \dfrac{\pi_1}{\sigma_1} \exp\left(-\dfrac{(x - \mu_1)^2}{2\sigma_1^2}\right) > \dfrac{\pi_0}{\sigma_0} \exp\left(-\dfrac{(x - \mu_0)^2}{2\sigma_0^2}\right); \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

Example: Gaussian class conditional densities



1/2 of x's from $\mathrm{N}(0,1)$ (w/ $y = 0$)
1/2 of x's from $\mathrm{N}(1,1)$ (w/ $y = 1$)

Example: Gaussian class conditional densities

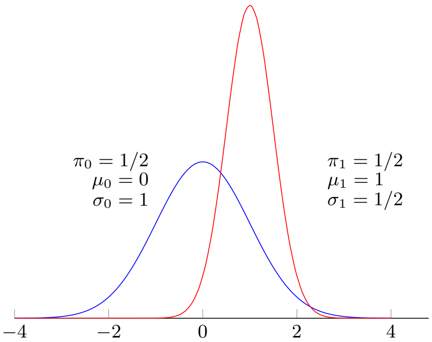


$1/2$ of x's from $\mathrm{N}(0,1)$ (w/ $y = 0$)
$1/2$ of x's from $\mathrm{N}(1,1)$ (w/ $y = 1$)

Bayes classifier:

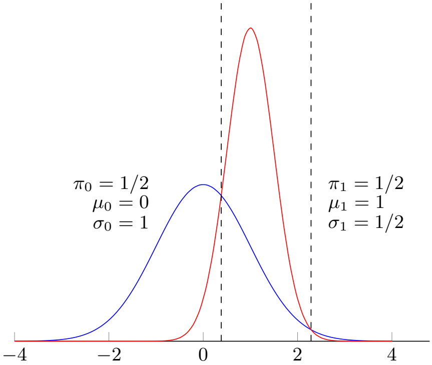
$$f^\star(x) = \begin{cases} 1 & \text{if } x > 1/2; \\ 0 & \text{otherwise.} \end{cases}$$

Example: Gaussian class conditional densities



$1/2$ of x's from $\mathrm{N}(0, 1)$ (w/ $y = 0$)
$1/2$ of x's from $\mathrm{N}(1, 1/2^2)$ (w/ $y = 1$)

Example: Gaussian class conditional densities



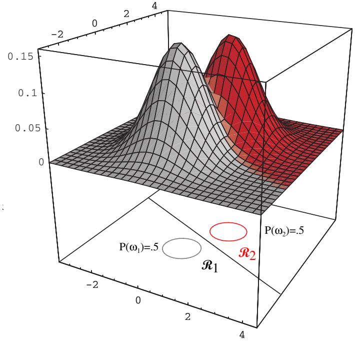
$1/2$ of x's from $\mathrm{N}(0,1)$ (w/ $y=0$)
$1/2$ of x's from $\mathrm{N}(1,1/2^2)$ (w/ $y=1$)

Bayes classifier:

$$f^\star(x) = \begin{cases} 1 & \text{if } x \in [0.38, 2.29]; \\ 0 & \text{otherwise.} \end{cases}$$

Example: multivariate Gaussians

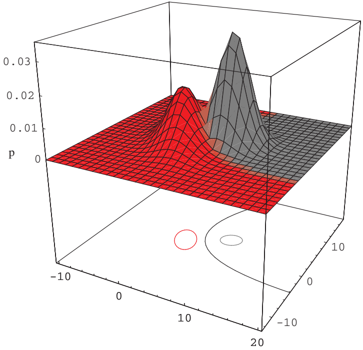
$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{0, 1\}$, class conditional densities are Gaussians in $\mathbb{R}^d$ $(d = 2)$.



$\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1$

Bayes classifier:

linear separator

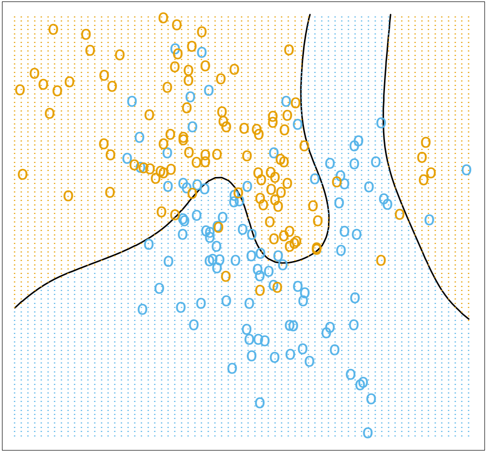


$\boldsymbol{\Sigma}_0 \neq \boldsymbol{\Sigma}_1$

Bayes classifier:

quadratic separator

Bayes classifier in general



In general, Bayes classifier may be rather complicated!

Classifiers via generative models

Bayes classifier

- Bayes classifier has smallest prediction error among *all* possible classifiers.

Bayes classifier

- Bayes classifier has smallest prediction error among *all* possible classifiers.
- But we can't construct the Bayes classifier without knowing $\Pr[Y = y|X = x]$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$!

  All we have are labeled examples drawn from the distribution.

Bayes classifier

- Bayes classifier has smallest prediction error among *all* possible classifiers.
- But we can't construct the Bayes classifier without knowing $\Pr[Y = y | X = x]$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$!

  All we have are labeled examples drawn from the distribution.

Plug-in classifiers

Using labeled examples, form an approximation to $\Pr[Y = y | X = x]$, then "plug-in" to the formula for Bayes classifier.

Plug-in classifiers

Bayes classifier

- Bayes classifier has smallest prediction error among *all* possible classifiers.
- But we can't construct the Bayes classifier without knowing $\Pr[Y = y | X = x]$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$!

  All we have are labeled examples drawn from the distribution.

Plug-in classifiers

Using labeled examples, form an approximation to $\Pr[Y = y | X = x]$, then "plug-in" to the formula for Bayes classifier.

We'll use **"generative" statistical models** to estimate P, and then form approximation to $\Pr[Y = y | X = x]$.

Plug-in classifiers using generative models

- **Plug-in classifiers using "generative" statistical models**:

Plug-in classifiers using generative models

- **Plug-in classifiers using “generative” statistical models**:

  1. Use **training data** (labeled examples) to obtain approximations for each component in Bayes classifier formula:

     $$f^\star(x) = \arg\max_{y \in \mathcal{Y}} \Pr[Y = y] \cdot \Pr[X = x \mid Y = y]$$

     (i.e., class priors and class conditional distributions).

Plug-in classifiers using generative models

- **Plug-in classifiers using "generative" statistical models**:

  1. Use **training data** (labeled examples) to obtain approximations for each component in Bayes classifier formula:

     $$f^\star(x) = \arg\max_{y \in \mathcal{Y}} \Pr[Y = y] \cdot \Pr[X = x \mid Y = y]$$

     (i.e., class priors and class conditional distributions).

  2. **Plug-in** approximations to formula to form classifier $\hat{f}$.

Plug-in classifiers using generative models

- **Plug-in classifiers using "generative" statistical models**:
  1. Use **training data** (labeled examples) to obtain approximations for each component in Bayes classifier formula:

     $$f^\star(x) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \Pr[Y = y] \cdot \Pr[X = x \mid Y = y]$$

     (i.e., class priors and class conditional distributions).
  2. **Plug-in** approximations to formula to form classifier $\hat{f}$.
- Estimating class priors is easy: parameterized by $(\pi_y : y \in \mathcal{Y})$ where $\pi_y \geq 0$ for all $y \in \mathcal{Y}$ and $\sum_{y \in \mathcal{Y}} \pi_y = 1$. MLE is straightforward.

Plug-in classifiers using generative models

- **Plug-in classifiers using "generative" statistical models**:
  1. Use **training data** (labeled examples) to obtain approximations for each component in Bayes classifier formula:

     $$f^\star(x) = \arg\max_{y \in \mathcal{Y}} \Pr[Y = y] \cdot \Pr[X = x \mid Y = y]$$

     (i.e., class priors and class conditional distributions).
  2. **Plug-in** approximations to formula to form classifier $\hat{f}$.
- Estimating class priors is easy: parameterized by $(\pi_y : y \in \mathcal{Y})$ where $\pi_y \geq 0$ for all $y \in \mathcal{Y}$ and $\sum_{y \in \mathcal{Y}} \pi_y = 1$. MLE is straightforward.
- Estimating class conditional distributions hard in general.

Plug-in classifiers using generative models

- **Plug-in classifiers using "generative" statistical models**:
  1. Use **training data** (labeled examples) to obtain approximations for each component in Bayes classifier formula:

     $$f^\star(x) = \underset{y \in \mathcal{Y}}{\arg\max} \; \Pr[Y = y] \cdot \Pr[X = x \mid Y = y]$$

     (i.e., class priors and class conditional distributions).
  2. **Plug-in** approximations to formula to form classifier $\hat{f}$.
- Estimating class priors is easy: parameterized by $(\pi_y : y \in \mathcal{Y})$ where $\pi_y \geq 0$ for all $y \in \mathcal{Y}$ and $\sum_{y \in \mathcal{Y}} \pi_y = 1$. MLE is straightforward.
- Estimating class conditional distributions hard in general.

  Usually just use *simple* parametric models.

Example: Gaussian class conditional densities

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{1, 2, \ldots, K\}$

- **Class priors**: MLE estimate of π_y is

$$\hat{\pi}_y := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\}.$$

Class conditional density $\mathrm{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$: MLE estimate of $(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ is

$$\hat{\boldsymbol{\mu}}_y := \frac{1}{n\hat{\pi}_y} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\}\boldsymbol{x}_i,$$

$$\widehat{\boldsymbol{\Sigma}}_y := \frac{1}{n\hat{\pi}_y} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_y)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_y)^\top.$$

Example: Gaussian class conditional densities

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{1, 2, \ldots, K\}$

- **Class priors**: MLE estimate of π_y is

$$\hat{\pi}_y := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\}.$$

Class conditional density $\mathrm{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$: MLE estimate of $(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ is

$$\hat{\boldsymbol{\mu}}_y := \frac{1}{n\hat{\pi}_y} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\} \boldsymbol{x}_i,$$

$$\widehat{\boldsymbol{\Sigma}}_y := \frac{1}{n\hat{\pi}_y} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_y)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_y)^\top.$$

- **Plug-in classifier**:

$$\hat{f}(\boldsymbol{x}) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \frac{\hat{\pi}_y}{\det(\widehat{\boldsymbol{\Sigma}}_y)^{1/2}} \exp\left(-\frac{1}{2} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_y)^\top \widehat{\boldsymbol{\Sigma}}_y^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_y) \right).$$

Example: Gaussian class conditional densities

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{1, 2, \dots, K\}$

- **Class priors**: MLE estimate of π_y is

$$\hat{\pi}_y := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\}.$$

Class conditional density $\mathrm{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$: MLE estimate of $(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ is

$$\hat{\boldsymbol{\mu}}_y := \frac{1}{n\hat{\pi}_y} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\} \boldsymbol{x}_i,$$

$$\widehat{\boldsymbol{\Sigma}}_y := \frac{1}{n\hat{\pi}_y} \sum_{i=1}^{n} \mathbb{1}\{y_i = y\} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_y)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_y)^\top.$$

- **Plug-in classifier**:

$$\hat{f}(\boldsymbol{x}) = \operatorname*{arg\,max}_{y \in \mathcal{Y}} \frac{\hat{\pi}_y}{\det(\widehat{\boldsymbol{\Sigma}}_y)^{1/2}} \exp\left(-\frac{1}{2} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_y)^\top \widehat{\boldsymbol{\Sigma}}_y^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_y) \right).$$

Caveat: $\widehat{\boldsymbol{\Sigma}}_y$ could be singular!

Classifiers via generative models: recap

Advantages:

1. Very simple recipe, just need some domain expertise to pick the class conditional distributions.
2. Surprisingly effective in many applications.

Classifiers via generative models: recap

Advantages:

1. Very simple recipe, just need some domain expertise to pick the class conditional distributions.
2. Surprisingly effective in many applications.

Disadvantages:

1. Hard to justify unless class conditional distributions are close to the truth.
2. Effort modeling P away from decision boundary between classes is not necessary for good classification.

Classifiers via generative models: recap

Advantages:

1. Very simple recipe, just need some domain expertise to pick the class conditional distributions.
2. Surprisingly effective in many applications.

Disadvantages:

1. Hard to justify unless class conditional distributions are close to the truth.
2. Effort modeling P away from decision boundary between classes is not necessary for good classification.

Next time: methods for modeling decision boundary directly.

Evaluating classifiers

Evaluating classifiers

Let $S := ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ be the **training data** used to construct a classifier $\hat{f}$ (say, via the plug-in method).

Evaluating classifiers

Let $S := ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ be the **training data** used to construct a classifier $\hat{f}$ (say, via the plug-in method).

Question: Can we tell if $\hat{f}$ is any good?

- **True error**:

$$\operatorname{err}(\hat{f}) := \Pr\left[\hat{f}(X) \neq Y\right]$$

  where $(X, Y) \sim P$.

Evaluating classifiers

Let $S := ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ be the **training data** used to construct a classifier $\hat{f}$ (say, via the plug-in method).

Question: Can we tell if $\hat{f}$ is any good?

- **True error**:
$$\mathrm{err}(\hat{f}) := \Pr\left[\hat{f}(X) \neq Y\right]$$
where $(X, Y) \sim P$.

- **Training error**:
$$\mathrm{err}(\hat{f}, S) := \frac{1}{|S|} \sum_{(x,y) \in S} \mathbb{1}\{\hat{f}(x) \neq y\}.$$

Evaluating classifiers

Let $S := ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ be the **training data** used to construct a classifier $\hat{f}$ (say, via the plug-in method).

Question: Can we tell if $\hat{f}$ is any good?

- **True error**:
  $$\operatorname{err}(\hat{f}) := \Pr\left[\hat{f}(X) \neq Y\right]$$
  where $(X, Y) \sim P$.

  Unfortunately, we don't know P so this can't be computed.
- **Training error**:
  $$\operatorname{err}(\hat{f}, S) := \frac{1}{|S|} \sum_{(x,y) \in S} \mathbb{1}\{\hat{f}(x) \neq y\}.$$

Evaluating classifiers

Let $S := ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ be the **training data** used to construct a classifier $\hat{f}$ (say, via the plug-in method).

Question: Can we tell if $\hat{f}$ is any good?

- **True error**:
$$\mathrm{err}(\hat{f}) := \Pr\left[\hat{f}(X) \neq Y\right]$$
where $(X, Y) \sim P$.
Unfortunately, we don't know P so this can't be computed.
- **Training error**:
$$\mathrm{err}(\hat{f}, S) := \frac{1}{|S|} \sum_{(x,y) \in S} \mathbb{1}\{\hat{f}(x) \neq y\}.$$
This often **under-estimates** the true error $\mathrm{err}(\hat{f})$.

Test error

General methodology
Given your pile of labeled examples, (randomly) split into two disjoint groups:

1. **training data** S;
2. **test data** T.

Test error

General methodology
Given your pile of labeled examples, (randomly) split into two disjoint groups:

1. **training data** S;
2. **test data** T.

Assuming you only use S to build the classifier $\hat{f}$, the **test error**

$$\operatorname{err}(\hat{f}, T) := \frac{1}{|T|} \sum_{(x,y) \in T} \mathbb{1}\{\hat{f}(x) \neq y\}$$

is a good estimate of the true error of $\hat{f}$.

Test error

General methodology
Given your pile of labeled examples, (randomly) split into two disjoint groups:

1. **training data** S;
2. **test data** T.

Assuming you only use S to build the classifier $\hat{f}$, the **test error**

$$\mathrm{err}(\hat{f}, T) := \frac{1}{|T|} \sum_{(x,y) \in T} \mathbb{1}\{\hat{f}(x) \neq y\}$$

is a good estimate of the true error of $\hat{f}$.

Assuming T is an i.i.d. sample from P:

- The test error is an unbiased estimate of $\mathrm{err}(\hat{f})$: $\mathbb{E}[\mathrm{err}(\hat{f}, T) \mid S] = \mathrm{err}(\hat{f})$.
  [Expectation is over random draw of T.]
- The standard deviation of $\mathrm{err}(\hat{f}, T)$ is $\sqrt{\dfrac{\mathrm{err}(\hat{f})(1 - \mathrm{err}(\hat{f}))}{|T|}}$.

Example: OCR

- **Task**: classify images of handwritten digits

EXAMPLE: OCR

- **Task**: classify images of handwritten digits
- **Data**: 30000 grayscale 28×28 images (treated as vectors in $\mathbb{R}^{784}$), with labels indicating the digit they represent.

  An example from each class:

Example: OCR

- **Task**: classify images of handwritten digits
- **Data**: 30000 grayscale 28×28 images (treated as vectors in $\mathbb{R}^{784}$), with labels indicating the digit they represent.

  An example from each class:

- Split into training data S and test data T using random $(2/3, 1/3)$-split.

Example: OCR

- **Task**: classify images of handwritten digits
- **Data**: 30000 grayscale 28×28 images (treated as vectors in $\mathbb{R}^{784}$), with labels indicating the digit they represent.

  An example from each class:

- Split into training data S and test data T using random $(2/3, 1/3)$-split.
- Using S, trained plug-in classifer $\hat{f}$ with Gaussian class conditional densities (plus one other trick).

  MLE $\hat{\boldsymbol{\mu}}_y$ for $y \in \mathcal{Y}$:

Example: OCR

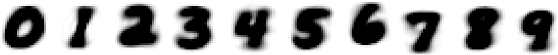
- **Task**: classify images of handwritten digits
- **Data**: 30000 grayscale 28×28 images (treated as vectors in $\mathbb{R}^{784}$), with labels indicating the digit they represent.

  An example from each class:

- Split into training data S and test data T using random $(2/3, 1/3)$-split.
- Using S, trained plug-in classifer $\hat{f}$ with Gaussian class conditional densities (plus one other trick).

  MLE $\hat{\boldsymbol{\mu}}_y$ for $y \in \mathcal{Y}$:

- **Training error**: $\operatorname{err}(\hat{f}, S) = 0.0346$

  Test error: $\operatorname{err}(\hat{f}, T) = 0.0415$

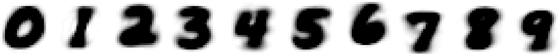
EXAMPLE: OCR

- **Task**: classify images of handwritten digits
- **Data**: 30000 grayscale 28×28 images (treated as vectors in $\mathbb{R}^{784}$), with labels indicating the digit they represent.

  An example from each class:

- Split into training data S and test data T using random $(2/3, 1/3)$-split.
- Using S, trained plug-in classifer $\hat{f}$ with Gaussian class conditional densities (plus one other trick).

  MLE $\hat{\boldsymbol{\mu}}_y$ for $y \in \mathcal{Y}$:



- **Training error**: $\mathrm{err}(\hat{f}, S) = 0.0346$

  Test error: $\mathrm{err}(\hat{f}, T) = 0.0415$
- **True error**? Unknown, but perhaps $0.039 \leq \mathrm{err}(\hat{f}) \leq 0.044$ or so.