Motivation for Statistical Models
Motivation

Statistical perspective on prediction

- Make intuitions about supervised learning more precise
- Offer guidance in design and analysis of ML methods
Predicting the Outcome of a Coin Toss
Question: How can you predict the outcome of a coin toss?
ON THE SENSITIVE DYNAMICAL SYSTEM AND THE TRANSITION FROM THE APPARENTLY DETERMINISTIC PROCESS TO THE COMPLETELY RANDOM PROCESS

Yue Zeng-yuan (岳曾元)  Zhang Bin (张彬)
(Department of Geophysics, Peking University, Beijing)

(Received Feb. 1, 1984)

Abstract
The detailed analysis of the dynamical process of coin tossing is made. Through calculations, it is illustrated how and why the result is extremely sensitive to the initial conditions. It is also shown that, as the initial height of the mass center of the coin increases, the final configuration, i.e. "head" or "tail", becomes more and more sensitive to the initial parameters (the initial velocity, angular velocity, and the initial orientation), the coefficient of the air drag, and the energy absorption factor of the surface on which the coin bounces. If we keep the "head" upward initially but allow a small range for the change of some other initial parameters, the frequency that the final configuration is "head", would be 1 if the initial height of the mass center is sufficiently small, and would be close to 1/2 if it is sufficiently large. An interesting question is how this frequency changes continuously from
THE PROBABILITY OF HEADS*

JOSEPH B. KELLER

Departments of Mathematics and Mechanical Engineering, Stanford University, Stanford, CA 94305

2. Mechanics of a tossed coin. Let us consider a circular coin of radius $a$ and negligible thickness, one side of which is marked heads and the other side tails. We assume that the center of gravity of the coin is at its geometrical center, the height of which we denote $y(t)$ at time $t$. Then Newton’s equation for the vertical motion of the center of gravity of the coin is

\[
\frac{d^2 y(t)}{dt^2} = -g.
\]

Here the positive constant $g$ is the acceleration of gravity. To supplement (1) we suppose that initially, at time $t = 0$, the center of the coin is at height $a$ and that it has an upward velocity $u$. Thus we have the initial conditions

\[
y(0) = a, \quad \frac{dy(0)}{dt} = u.
\]
Dynamical Bias in the Coin Toss

Persi Diaconis, Susan Holmes, and Richard Montgomery
https://doi.org/10.1137/S0036144504446436

We analyze the natural process of flipping a coin which is caught in the hand. We show that vigorously flipped coins tend to come up the same way they started. The limiting chance of coming up this way depends on a single parameter, the angle between the normal to the coin and the angular momentum vector. Measurements of this parameter based on high-speed photography are reported. For natural flips, the chance of coming up as started is about .51.
Models for binary outcomes

Models for the outcome of a coin toss:

- Physics: Yue & Zhang (1985); Keller (1986); Diaconis, Holmes, & Montgomery (2007), etc.
- Statistics:
  - Outcome is a random variable
  - Statistical model for coin toss:
    \[ Y \sim \text{Bernoulli}(\theta) \]
    - Random variable \( Y \) follows the Bernoulli distribution with success probability parameter \( \theta \)
    - Parameter \( \theta \) is a real number between 0 and 1
    - \( \Pr(Y = 1) = \theta \) (probability of heads)
    - \( \Pr(Y = 0) = 1 - \theta \) (probability of tails)
    - Mean: \( \mathbb{E}(Y) = \theta \)
    - Standard deviation: \( \text{var}(Y) = \theta (1 - \theta) \)
Models for binary outcomes

Models for the outcome of a coin toss:

- **Physics:** Yue & Zhang (1985); Keller (1986); Diaconis, Holmes, & Montgomery (2007), etc.
Models for binary outcomes

Models for the outcome of a coin toss:

- **Physics**: Yue & Zhang (1985); Keller (1986); Diaconis, Holmes, & Montgomery (2007), etc.
- **Statistics**: Outcome is a *random variable*
Models for binary outcomes

Models for the outcome of a coin toss:

▶ **Physics:** Yue & Zhang (1985); Keller (1986); Diaconis, Holmes, & Montgomery (2007), etc.
▶ **Statistics:** Outcome is a *random variable*

---

**Statistical model for coin toss:** Outcome $Y$ is a random variable,

$$Y \sim \text{Bernoulli}(\theta)$$

“Random variable $Y$ follows the **Bernoulli distribution** with success probability parameter $\theta$”

▶ Parameter $\theta$ is a real number between 0 and 1
▶ $\Pr(Y = 1) = \theta$ (probability of heads)
▶ $\Pr(Y = 0) = 1 - \theta$ (probability of tails)
▶ **Mean:** $\mathbb{E}(Y) = \theta$
▶ **Standard deviation:** $\sqrt{\text{var}(Y)} = \sqrt{\theta(1 - \theta)}$
Task: Predict the outcome of the coin toss, $Y \sim \text{Bernoulli}(\theta)$
**Task:** Predict the outcome of the coin toss, $Y \sim \text{Bernoulli}(\theta)$

<table>
<thead>
<tr>
<th>prediction</th>
<th>probability of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>heads (1)</td>
<td>$1 - \theta$</td>
</tr>
<tr>
<td>tails (0)</td>
<td>$\theta$</td>
</tr>
</tbody>
</table>
Optimal predictions in the statistical model

**Task:** Predict the outcome of the coin toss, $Y \sim \text{Bernoulli}(\theta)$

<table>
<thead>
<tr>
<th>prediction</th>
<th>probability of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>heads (1)</td>
<td>$1 - \theta$</td>
</tr>
<tr>
<td>tails (0)</td>
<td>$\theta$</td>
</tr>
</tbody>
</table>

So, optimal prediction (i.e., prediction with smallest error probability) is

$$\mathbb{1}\{\theta > 1/2\} = \begin{cases} 1 & \text{if } \theta > 1/2 \\ 0 & \text{if } \theta \leq 1/2 \end{cases}$$

and optimal error probability is

$$\min \{\theta, 1 - \theta\}$$
Optimal predictions in the statistical model

**Task:** Predict the outcome of the coin toss, $Y \sim \text{Bernoulli}(\theta)$

<table>
<thead>
<tr>
<th>prediction</th>
<th>probability of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>heads (1)</td>
<td>$1 - \theta$</td>
</tr>
<tr>
<td>tails (0)</td>
<td>$\theta$</td>
</tr>
</tbody>
</table>

So, optimal prediction (i.e., prediction with smallest error probability) is

$$\mathbb{1}\{\theta > 1/2\} = \begin{cases} 1 & \text{if } \theta > 1/2 \\ 0 & \text{if } \theta \leq 1/2 \end{cases}$$

and optimal error probability is

$$\min \{\theta, 1 - \theta\}$$

Optimal prediction depends on the distribution of $Y$ (through the parameter $\theta$)
**Task:** Predict the outcome of the coin toss, $Y \sim \text{Bernoulli}(\theta)$

<table>
<thead>
<tr>
<th>prediction</th>
<th>probability of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>heads (1)</td>
<td>$1 - \theta$</td>
</tr>
<tr>
<td>tails (0)</td>
<td>$\theta$</td>
</tr>
</tbody>
</table>

So, optimal prediction (i.e., prediction with smallest error probability) is

$$1 \{ \theta > 1/2 \} = \begin{cases} 1 & \text{if } \theta > 1/2 \\ 0 & \text{if } \theta \leq 1/2 \end{cases}$$

and optimal error probability is

$$\min \{ \theta, 1 - \theta \}$$

Optimal prediction depends on the distribution of $Y$ (through the parameter $\theta$)

**But what if $\theta$ is unknown?**
Learning from Data
Learning from data

How can we learn from data?

Suppose:

- We have data: Outcomes of previous coin tosses
  0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0

- Data is related to what we want to predict: It's the same coin!

Question: Why/how can we learn from this data?
How can we learn from data?

Suppose:

- **We have data:** Outcomes of previous coin tosses

  0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1
How can we learn from data?

Suppose:

- **We have data:** Outcomes of previous coin tosses
  
  0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1

- **Data is related to what we want to predict:** It's the same coin!
How can we learn from data?

Suppose:

- **We have data:** Outcomes of previous coin tosses
  
  0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0

- **Data is related to what we want to predict:** It’s the same coin!

**Question:** Why/how can we learn from this data?
IID model of coin toss data:

- Regard data $Y_1, \ldots, Y_n$ (in addition to $Y$) as random variables.
IID model

IID model of coin toss data:

- Regard data $Y_1, \ldots, Y_n$ (in addition to $Y$) as random variables
- Assume data are independent
  - Did not (say) just accidentally write down outcome of one toss 20 times
IID model

IID model of coin toss data:

- Regard data $Y_1, \ldots, Y_n$ (in addition to $Y$) as random variables
- Assume data are independent
  - Did not (say) just accidentally write down outcome of one toss 20 times
- Assume data are identically distributed
  - Outcomes are from tossing the same coin

"Random variables $Y_1, \ldots, Y_n, Y$ are independent & identically distributed according to $\text{Bernoulli}(\theta)$"
IID model

IID model of coin toss data:

▶ Regard data $Y_1, \ldots, Y_n$ (in addition to $Y$) as random variables
▶ Assume data are independent
  ▶ Did not (say) just accidentally write down outcome of one toss 20 times
▶ Assume data are identically distributed
  ▶ Outcomes are from tossing the same coin
▶ Also assume $Y$ is independent of and identically distributed as $Y_1, \ldots, Y_n$
  ▶ $Y$ is really the outcome of a new coin toss, not just (say) recording of previous coin toss

Since $Y \sim \text{Bernoulli}(\theta)$, we can summarize above by: IID model for coin toss data:

$Y_1, \ldots, Y_n, Y \sim \text{iid Bernoulli}(\theta)$

“Random variables $Y_1, \ldots, Y_n, Y$ are independent & identically distributed according to Bernoulli($\theta$)”
IID model

IID model of coin toss data:

- Regard data $Y_1, \ldots, Y_n$ (in addition to $Y$) as random variables
- Assume data are independent
  - Did not (say) just accidentally write down outcome of one toss 20 times
- Assume data are identically distributed
  - Outcomes are from tossing the same coin
- Also assume $Y$ is independent of and identically distributed as $Y_1, \ldots, Y_n$
  - $Y$ is really the outcome of a new coin toss, not just (say) recording of previous coin toss

Since $Y \sim \text{Bernoulli}(\theta)$, we can summarize above by:

IID model for coin toss data:

$$Y_1, \ldots, Y_n, Y \sim_{iid} \text{Bernoulli}(\theta)$$

“Random variables $Y_1, \ldots, Y_n, Y$ are independent & identically distributed according to $\text{Bernoulli}(\theta)$”
Natural prediction learning strategy

Look at outcomes of \( n \) previous coin tosses \( Y_1, \ldots, Y_n \), and then predict with the majority outcome:

- If \( Y_1 + \cdots + Y_n > \frac{n}{2} \), then predict heads (1)
- Else, predict tails (0)

Theorem. If we assume \( Y_1, \ldots, Y_n, Y \sim \text{iid Bernoulli} (\theta) \), then using the above strategy, the probability of error is at most

\[
\min\{\theta, 1 - \theta\} + 2 \cdot \theta - 1 \cdot \frac{1}{2} \cdot \exp^{-2n\theta - 1/2} \\
\]

- First term \( \min\{\theta, 1 - \theta\} \): optimal error probability, even if we knew \( \theta \)
- Second term: the price of not knowing \( \theta \)
  - If \( \theta = \frac{1}{2} \), it is zero
  - If \( \theta \neq \frac{1}{2} \), it is exponentially small in \( n \)

Upshot: Under the IID model, data are very likely to reveal optimal prediction!
Natural prediction learning strategy

Look at outcomes of $n$ previous coin tosses $Y_1, \ldots, Y_n$, and then predict with the majority outcome:

- If $Y_1 + \cdots + Y_n > \frac{n}{2}$, then predict heads (1)
- Else, predict tails (0)

**Theorem.** If we assume $Y_1, \ldots, Y_n, Y \sim_{iid} \text{Bernoulli}(\theta)$, then using the above strategy, the probability of error is at most

$$\min\{\theta, 1 - \theta\} + 2 \left| \theta - \frac{1}{2} \right| \exp \left( -2n \left( \theta - \frac{1}{2} \right)^2 \right)$$
Natural prediction learning strategy

Look at outcomes of $n$ previous coin tosses $Y_1, \ldots, Y_n$, and then predict with the majority outcome:

- If $Y_1 + \cdots + Y_n > \frac{n}{2}$, then predict heads ($1$)
- Else, predict tails ($0$)

**Theorem.** If we assume $Y_1, \ldots, Y_n, Y \sim_{iid} \text{Bernoulli}(\theta)$, then using the above strategy, the probability of error is at most

$$\min \{\theta, 1 - \theta\} + 2 \cdot \left|\theta - \frac{1}{2}\right| \cdot \exp \left(-2n \left(\theta - \frac{1}{2}\right)^2\right)$$

- First term $\min \{\theta, 1 - \theta\}$: optimal error probability, even if we knew $\theta$
Natural prediction learning strategy

Look at outcomes of $n$ previous coin tosses $Y_1, \ldots, Y_n$, and then predict with the majority outcome:

- If $Y_1 + \cdots + Y_n > \frac{n}{2}$, then predict heads (1)
- Else, predict tails (0)

**Theorem.** If we assume $Y_1, \ldots, Y_n, Y \sim_{iid} \text{Bernoulli}(\theta)$, then using the above strategy, the probability of error is at most

$$\min \{\theta, 1 - \theta\} + 2 \cdot \left|\theta - \frac{1}{2}\right| \cdot \exp \left(-2n \left(\theta - \frac{1}{2}\right)^2\right)$$

- First term $\min \{\theta, 1 - \theta\}$: optimal error probability, even if we knew $\theta$
- Second term: the price of not knowing $\theta$
  - If $\theta = 1/2$, it is zero
  - If $\theta \neq 1/2$, it is exponentially small in $n$
Natural prediction learning strategy

Look at outcomes of \( n \) previous coin tosses \( Y_1, \ldots, Y_n \), and then predict with the majority outcome:

- If \( Y_1 + \cdots + Y_n > \frac{n}{2} \), then predict heads (1)
- Else, predict tails (0)

**Theorem.** If we assume \( Y_1, \ldots, Y_n, Y \sim_{\text{iid}} \text{Bernoulli}(\theta) \), then using the above strategy, the probability of error is at most

\[
\min \{\theta, 1-\theta\} + 2 \cdot \left|\theta - \frac{1}{2}\right| \cdot \exp \left(-2n \left(\theta - \frac{1}{2}\right)^2\right)
\]

- First term \( \min \{\theta, 1-\theta\} \): optimal error probability, even if we knew \( \theta \)
- Second term: the price of not knowing \( \theta \)
  - If \( \theta = 1/2 \), it is zero
  - If \( \theta \neq 1/2 \), it is exponentially small in \( n \)

**Upshot:** Under the IID model, data are very likely to reveal optimal prediction!
Illustration

Let \( S := Y_1 + \cdots + Y_n \), where \( Y_1, \ldots, Y_n \sim_{\text{iid}} \text{Bernoulli}(\theta) \) and \( \theta = 0.4 \)

**Question:** What is the probability that \( S > \frac{n}{2} \)?
Let $S := Y_1 + \cdots + Y_n$, where $Y_1, \ldots, Y_n \sim_{iid} \text{Bernoulli} (\theta)$ and $\theta = 0.4$.

**Question:** What is the probability that $S > \frac{n}{2}$?
Let $S := Y_1 + \cdots + Y_n$, where $Y_1, \ldots, Y_n \sim_{\text{iid}} \text{Bernoulli}(\theta)$ and $\theta = 0.4$.

**Question:** What is the probability that $S > \frac{n}{2}$?
Let \( S := Y_1 + \cdots + Y_n \), where \( Y_1, \ldots, Y_n \sim_{\text{iid}} \text{Bernoulli}(\theta) \) and \( \theta = 0.4 \)

**Question:** What is the probability that \( S > \frac{n}{2} \)?
Let $S := Y_1 + \cdots + Y_n$, where $Y_1, \ldots, Y_n \sim \text{iid Bernoulli}(\theta)$ and $\theta = 0.4$

**Question:** What is the probability that $S > \frac{n}{2}$?
Let \( S := Y_1 + \cdots + Y_n \), where \( Y_1, \ldots, Y_n \sim_{iid} \text{Bernoulli}(\theta) \) and \( \theta = 0.4 \).

**Question:** What is the probability that \( S > \frac{n}{2} \)?
Plug-in principle

High-level principle at play: plug-in principle
Plug-in principle

High-level principle at play: **plug-in principle**

1. Consider a model for your problem, which may involve unknown quantities
   E.g., model parameter $\theta$

Example:
1. Assume IID model for coin toss data
2. Optimal prediction $\left\{ \theta > \frac{1}{2} \right\}$
3. Estimate $\theta$ using data $\to \theta_{\text{est}} = \frac{1}{n} \sum_{i=1}^{n} Y_i$; learned prediction is $\left\{ \theta_{\text{est}} > \frac{1}{2} \right\}$
Plug-in principle

High-level principle at play: **plug-in principle**

1. Consider a model for your problem, which may involve unknown quantities
   E.g., model parameter $\theta$

2. Assuming knowledge of all unknowns, identify a formula for desirable course of action
   E.g., form optimal prediction in terms of unknown quantities

Example:

1. Assume IID model for coin toss data
2. Optimal prediction $\{\theta > 1/2\}$
3. Estimate $\theta$ using data $\rightarrow \theta_{\text{est}} := \frac{1}{n} \sum_{i=1}^{n} Y_i$; learned prediction is $\{\theta_{\text{est}} > 1/2\}$
Plug-in principle

High-level principle at play: **plug-in principle**

1. Consider a model for your problem, which may involve unknown quantities
   
   E.g., model parameter $\theta$

2. Assuming knowledge of all unknowns, identify a formula for desirable course of action
   
   E.g., form optimal prediction in terms of unknown quantities

3. Use data to estimate the unknowns, and then “plug-in” to formula from previous step

Example:

1. Assume IID model for coin toss data
2. Optimal prediction $\{\theta > \frac{1}{2}\}$ which involves unknown parameter $\theta$
3. Estimate $\theta$ using data $\to \theta_{est} := \frac{1}{n} \sum_{i=1}^{n} Y_i$; learned prediction is $\{\theta_{est} > \frac{1}{2}\}$
Plug-in principle

High-level principle at play: **plug-in principle**

1. Consider a model for your problem, which may involve unknown quantities  
   E.g., model parameter $\theta$

2. Assuming knowledge of all unknowns, identify a formula for desirable course of action  
   E.g., form optimal prediction in terms of unknown quantities

3. Use data to estimate the unknowns, and then “plug-in” to formula from previous step

**Example:**

1. Assume IID model for coin toss data
High-level principle at play: **plug-in principle**

1. Consider a model for your problem, which may involve unknown quantities
   E.g., model parameter $\theta$

2. Assuming knowledge of all unknowns, identify a formula for desirable course of action
   E.g., form optimal prediction in terms of unknown quantities

3. Use data to estimate the unknowns, and then “plug-in” to formula from previous step

**Example:**

1. Assume IID model for coin toss data

2. Optimal prediction

   \[
   \mathbb{1}\{\theta > 1/2\}
   \]

   which involves unknown parameter $\theta$
Plug-in principle

High-level principle at play: **plug-in principle**

1. Consider a model for your problem, which may involve unknown quantities
   E.g., model parameter $\theta$

2. Assuming knowledge of all unknowns, identify a formula for desirable course of action
   E.g., form optimal prediction in terms of unknown quantities

3. Use data to estimate the unknowns, and then “plug-in” to formula from previous step

**Example:**

1. Assume IID model for coin toss data

2. Optimal prediction

   $$\mathbb{1}\{\theta > 1/2\}$$

   which involves unknown parameter $\theta$

3. Estimate $\theta$ using data $\rightarrow \theta_{\text{est}} := \frac{1}{n} \sum_{i=1}^{n} Y_i$; learned prediction is

   $$\mathbb{1}\{\theta_{\text{est}} > 1/2\}$$
Statistical Models for Binary Classification
Predicting a coin toss vs. filtering spam

The coin toss example seems far removed from ML applications (e.g., spam filtering). Why?

- Doesn't make sense to predict "spam" or "not spam" without looking at:
  - Text of email
  - Email address of sender
  - IP address of sender
  - And so on

I.e., features that are relevant (and available to use) for prediction

Received: from mx0a-00364e01.pphosted.com (mx0a-00364e01.pphosted.com. [148.163.135.74])
Date: Sat, 30 Jan 2021 15:57:16 +0530 (IST)
From: Mtar Tg <info@allalertsedu.com>
Reply-To: editor.infogain@gmail.com

Dear Sir / Madam,

For your research database, please find the complete IJAERS Nov 2020 edition (Volume 7, Issue 12):

https://ijaers.com/complete-issue/
Predicting a coin toss vs. filtering spam

The coin toss example seems far removed from ML applications (e.g., spam filtering). Why?

- Doesn't make sense to predict “spam” or “not spam” without looking at:
  - Text of email
  - Email address of sender
  - IP address of sender
  - And so on
Predicting a coin toss vs. filtering spam

The coin toss example seems far removed from ML applications (e.g., spam filtering). Why?

▶ Doesn’t make sense to predict “spam” or “not spam” without looking at:
  ▶ Text of email
  ▶ Email address of sender
  ▶ IP address of sender
  ▶ And so on

I.e., features that are relevant (and available to use) for prediction
Statistical model for binary classification:

- Outcome/label is a Bernoulli random variable $Y$.
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$.
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application.
Statistical model for binary classification:

- Outcome/label is a Bernoulli random variable $Y$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

Remarks:

- Vector of random variables (e.g., $\vec{X}$) is called a random vector; $(\vec{X}, Y)$ is a random example
Statistical model for binary classification

► Outcome/label is a Bernoulli random variable $Y$
► Feature vector is a vector of $d$ random variables $\mathbf{X} := (X_1, \ldots, X_d)$
► Joint distribution of $(\mathbf{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

Remarks:
► Vector of random variables (e.g., $\mathbf{X}$) is called a random vector; $(\mathbf{X}, Y)$ is a random example
► $\mathbf{X}$ and $Y$ may be dependent random variables
  ▶ If $\mathbf{X}$ and $Y$ are independent, then $\mathbf{X}$ is not useful for predicting $Y$!
Statistical model for binary classification:

- Outcome/label is a Bernoulli random variable $Y$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

Remarks:

- Vector of random variables (e.g., $\vec{X}$) is called a random vector; $(\vec{X}, Y)$ is a random example
- $\vec{X}$ and $Y$ may be dependent random variables
  - If $\vec{X}$ and $Y$ are independent, then $\vec{X}$ is not useful for predicting $Y$!
- Prediction of $Y$ is allowed to depend on $\vec{X}$
  - I.e., predict using a classifier $f$ that maps feature vectors to predictions
Statistical model for binary classification:

▶ Outcome/label is a Bernoulli random variable $Y$
▶ Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
▶ Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

Remarks:

▶ Vector of random variables (e.g., $\vec{X}$) is called a random vector; $(\vec{X}, Y)$ is a random example
▶ $\vec{X}$ and $Y$ may be dependent random variables
  ▶ If $\vec{X}$ and $Y$ are independent, then $\vec{X}$ is not useful for predicting $Y$!
▶ Prediction of $Y$ is allowed to depend on $\vec{X}$
  ▶ I.e., predict using a classifier $f$ that maps feature vectors to predictions
▶ Standard benchmark for classifier $f$: Error rate, defined to be
  \[ \Pr(f(\vec{X}) \neq Y) \]
  ▶ This probability is specific to joint distribution of $(\vec{X}, Y)$
Extend IID model to training data \((\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)\) for supervised learning
IID model for training data

Extend IID model to training data $(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)$ for supervised learning

**IID model for training data:**
Training data $(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)$ and “future” example $(\vec{X}, Y)$ are IID
Extend IID model to training data $(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)$ for supervised learning

**IID model for training data:**
Training data $(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)$ and “future” example $(\vec{X}, Y)$ are IID

Just as in problem of predicting a coin toss, can hope to use training data to learn a good classifier
**Binary predictions vs. classifiers**

<table>
<thead>
<tr>
<th>Learned binary prediction (heads or tails)</th>
<th>Learned binary classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Binary predictions vs. classifiers

<table>
<thead>
<tr>
<th>Learned binary prediction (heads or tails)</th>
<th>Learned binary classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant function that ignores $\vec{X}$</td>
<td>Possibly non-constant function that uses $\vec{X}$</td>
</tr>
</tbody>
</table>

Optimal error rate is $\min\{\Pr(Y = 1), \Pr(Y = 0)\}$.

Optimal error rate can be much better!

Suppose each of the $d$ features takes only 2 possible values (say, 0 or 1).

Number of possible feature vectors $2^d$.

- $d = 0$: 1
- $d = 2$: 4
- $d = 5$: 32
- $d = 10$: 1024

Number of possible binary classifiers $2^{2^d}$.

- $d = 16 / 32$
### Binary predictions vs. classifiers

<table>
<thead>
<tr>
<th>Learned binary prediction (heads or tails)</th>
<th>Learned binary classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant function that ignores $\vec{X}$</td>
<td>Possibly non-constant function that uses $\vec{X}$</td>
</tr>
<tr>
<td>Optimal error rate is $\min{\Pr(Y = 1), \Pr(Y = 0)}$</td>
<td>Optimal error rate can be much better!</td>
</tr>
</tbody>
</table>

Suppose each of the $d$ features takes only 2 possible values (say, 0 or 1). The number of possible feature vectors is $2^d$. For $d = 0$, $d = 1$, $d = 2$, $d = 5$, $d = 10$, the number of possible binary classifiers is $2^{2^d}$, which is approximately $1.8 \times 10^{308}$. This demonstrates the potential for much better optimal error rates compared to simple predictions.
**Binary predictions vs. classifiers**

<table>
<thead>
<tr>
<th>Learned binary prediction (heads or tails)</th>
<th>Learned binary classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant function that ignores ( \vec{X} )</td>
<td>Possibly non-constant function that uses ( \vec{X} )</td>
</tr>
<tr>
<td>Optimal error rate is ( \min{\Pr(Y = 1), \Pr(Y = 0)} )</td>
<td>Optimal error rate can be much better!</td>
</tr>
</tbody>
</table>

Suppose each of the \( d \) features takes only 2 possible values (say, 0 or 1)

<table>
<thead>
<tr>
<th></th>
<th>( d = 0 )</th>
<th>( d = 2 )</th>
<th>( d = 5 )</th>
<th>( d = 10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of possible feature vectors</td>
<td>( 2^d )</td>
<td>1</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>Number of possible binary classifiers</td>
<td>( 2^{2^d} )</td>
<td>2</td>
<td>16</td>
<td>( \approx 4.3 \times 10^9 )</td>
</tr>
</tbody>
</table>
Learning Binary Classifiers
Structure of optimal binary classifiers

**Question:** What are the predictions made by the optimal classifier? (i.e., the classifier $f^*$ with smallest error rate)
Structure of optimal binary classifiers

Question: What are the predictions made by the optimal classifier?
(I.e., the classifier $f^*$ with smallest error rate)

- For each possible feature vector $\vec{x}$, conditional distribution of $Y$ given $\vec{X} = \vec{x}$ is Bernoulli with “success probability” parameter that may be specific to $\vec{x}$:

$$ (Y \mid \vec{X} = \vec{x}) \sim \text{Bernoulli}(\eta(\vec{x})) $$

The $\vec{x}$-specific parameter $\eta(\vec{x})$ is a number between 0 and 1
Structure of optimal binary classifiers

**Question:** What are the predictions made by the optimal classifier? (I.e., the classifier $f^*$ with smallest error rate)

- For each possible feature vector $\bar{x}$, conditional distribution of $Y$ given $\bar{X} = \bar{x}$ is Bernoulli with “success probability” parameter that may be specific to $\bar{x}$:

$$ (Y \mid \bar{X} = \bar{x}) \sim \text{Bernoulli}(\eta(\bar{x})) $$

The $\bar{x}$-specific parameter $\eta(\bar{x})$ is a number between 0 and 1

- Recall reasoning used to derive optimal prediction of a coin toss:

<table>
<thead>
<tr>
<th>prediction upon $\bar{X} = \bar{x}$</th>
<th>probability of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>heads (1)</td>
<td>$1 - \eta(\bar{x})$</td>
</tr>
<tr>
<td>tails (0)</td>
<td>$\eta(\bar{x})$</td>
</tr>
</tbody>
</table>
**Question:** What are the predictions made by the optimal classifier? (i.e., the classifier \( f^\star \) with smallest error rate)

- For each possible feature vector \( \vec{x} \), conditional distribution of \( Y \) given \( \vec{X} = \vec{x} \) is Bernoulli with “success probability” parameter that may be specific to \( \vec{x} \):

  \[
  (Y \mid \vec{X} = \vec{x}) \sim \text{Bernoulli}(\eta(\vec{x}))
  \]

  The \( \vec{x} \)-specific parameter \( \eta(\vec{x}) \) is a number between 0 and 1

- Recall reasoning used to derive optimal prediction of a coin toss:

<table>
<thead>
<tr>
<th>prediction upon ( \vec{X} = \vec{x} )</th>
<th>probability of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>heads (1)</td>
<td>1 - ( \eta(\vec{x}) )</td>
</tr>
<tr>
<td>tails (0)</td>
<td>( \eta(\vec{x}) )</td>
</tr>
</tbody>
</table>

- So, optimal prediction upon \( \vec{X} = \vec{x} \) is:

  \[
  f^\star(\vec{x}) := 1\{\eta(\vec{x}) > 1/2\} = \begin{cases} 
  1 & \text{if } \eta(\vec{x}) > 1/2 \\
  0 & \text{if } \eta(\vec{x}) \leq 1/2
  \end{cases}
  \]
Natural classifier learning strategy

Use training data $S$ to construct classifier $f$ so that $f(x) = f^*(x)$ for as many $x$ as possible!
Natural classifier learning strategy

Use training data $S$ to construct classifier $f$ so that $f(\vec{x}) = f^*(\vec{x})$ for as many $\vec{x}$ as possible!

- Suppose only $L$ possible feature vectors, for some finite number $L$
  (e.g., $L = 2^d$ if each of the $d$ features takes only 2 possible values)
Natural classifier learning strategy

Use training data $S$ to construct classifier $f$ so that $f(\vec{x}) = f^*(\vec{x})$ for as many $\vec{x}$ as possible!

- Suppose only $L$ possible feature vectors, for some finite number $L$
  (e.g., $L = 2^d$ if each of the $d$ features takes only 2 possible values)
- For each $\vec{x}$ among the $L$ possible feature vectors, let

  $S_{\vec{x}} := \text{training examples with feature vector } \vec{x}$

  $f(\vec{x}) := \text{majority label in } S_{\vec{x}}$

  (If $\vec{x}$ doesn’t appear among training data, use a default value for $f(\vec{x})$)

Just like learning predictions for coin toss, but:

- $L$ separate coins, one per possible feature vector
- Number of training data available per coin may be $n/L$ or smaller

(Also an instantiation of plug-in principle!)

This strategy won’t work if number of possible feature vectors $L$ is large or infinite!
Natural classifier learning strategy

Use training data $S$ to construct classifier $f$ so that $f(\vec{x}) = f^*(\vec{x})$ for as many $\vec{x}$ as possible!

- Suppose only $L$ possible feature vectors, for some finite number $L$
  (e.g., $L = 2^d$ if each of the $d$ features takes only 2 possible values)
- For each $\vec{x}$ among the $L$ possible feature vectors, let
  
  $S_{\vec{x}} :=$ training examples with feature vector $\vec{x}$
  
  $f(\vec{x}) :=$ majority label in $S_{\vec{x}}$

  (If $\vec{x}$ doesn’t appear among training data, use a default value for $f(\vec{x})$)

Just like learning predictions for coin toss, but:

- $L$ separate coins, one per possible feature vector
- Number of training data available per coin may be $n/L$ or smaller

(Also an instantiation of plug-in principle!)
Natural classifier learning strategy

Use training data $S$ to construct classifier $f$ so that $f(\vec{x}) = f^*(\vec{x})$ for as many $\vec{x}$ as possible!

- Suppose only $L$ possible feature vectors, for some finite number $L$
  (e.g., $L = 2^d$ if each of the $d$ features takes only 2 possible values)
- For each $\vec{x}$ among the $L$ possible feature vectors, let

  $S_{\vec{x}} :=$ training examples with feature vector $\vec{x}$

  $f(\vec{x}) :=$ majority label in $S_{\vec{x}}$

  (If $\vec{x}$ doesn’t appear among training data, use a default value for $f(\vec{x})$)

Just like learning predictions for coin toss, but:

- $L$ separate coins, one per possible feature vector
- Number of training data available per coin may be $n/L$ or smaller

(Also an instantiation of plug-in principle!)

This strategy won’t work if number of possible feature vectors $L$ is large or infinite!
Greedy training heuristic for decision trees “coarsens” the space of possible feature vectors to some finite number $L$ of possible values (one per leaf node)
Greedy training heuristic for decision trees “coarsens” the space of possible feature vectors to some finite number $L$ of possible values (one per leaf node)

- Caveat: Same training data used for “coarsening” are also used to determine the “return values” (the predictions of the classifier)
1. Assume/leverage local regularity
   ▶ ⇒ prediction on $\vec{x}$ benefits from training examples $(\vec{X}_i, Y_i)$ where $\vec{X}_i$ is near $\vec{x}$
   (e.g., decision tree learning)
1. Assume/leverage *local regularity*
   ▶ ⇒ prediction on $\vec{x}$ benefits from training examples $(\vec{X}_i, Y_i)$ where $\vec{X}_i$ is near $\vec{x}$
   (e.g., decision tree learning)

2. Assume/leverage *global structure*
   ▶ ⇒ prediction on $\vec{x}$ benefits from *all* training examples $(\vec{X}_i, Y_i)$
   (e.g., linear regression, linear classification)
General approaches to learning

1. Assume/leverage local regularity
   ▶ ⇒ prediction on \( \vec{x} \) benefits from training examples \((\vec{X}_i, Y_i)\) where \( \vec{X}_i \) is near \( \vec{x} \)
   (e.g., decision tree learning)

2. Assume/leverage global structure
   ▶ ⇒ prediction on \( \vec{x} \) benefits from all training examples \((\vec{X}_i, Y_i)\)
   (e.g., linear regression, linear classification)

(Later, we'll see other ways to instantiate the plug-in principle)
Testing
IID model for training and test data

Extending the IID model to test data gives statistical justification for testing

Training data $(\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)$, test data $(\vec{X}_1', Y_1'), \ldots, (\vec{X}_m', Y_m')$, and “future” example $(\vec{X}, Y)$ are IID
IID model for training and test data

Extending the IID model to test data gives statistical justification for testing

Training data $((\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)$, test data $((\vec{X}_1', Y_1'), \ldots, (\vec{X}_m', Y_m'))$, and “future” example $(\vec{X}, Y)$ are IID

Key observations in this model:
- Training data are random, and hence so is classifier $f$ learned from training data
Extending the IID model to test data gives statistical justification for testing

Training data \((\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)\), test data \((\vec{X}_1', Y_1'), \ldots, (\vec{X}_m', Y_m')\), and “future” example \((\vec{X}, Y)\) are IID

Key observations in this model:

- Training data are random, and hence so is classifier \(f\) learned from training data
- If learned classifier \(f\) is based only on training data, then test data are independent of \(f\)
**IID model for training and test data**

Extending the IID model to test data gives statistical justification for testing

Training data \((\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)\), test data \((\vec{X}'_1, Y'_1), \ldots, (\vec{X}'_m, Y'_m)\), and “future” example \((\vec{X}, Y)\) are IID

**Key observations in this model:**

- Training data are random, and hence so is classifier \(f\) learned from training data
- If learned classifier \(f\) is based only on training data, then test data are independent of \(f\)

Conditional on \(f\), the learned classifier \(f\) has some probability of prediction error on \((\vec{X}, Y)\):

\[
\varepsilon := \Pr(f(\vec{X}) \neq Y \mid f)
\]
Extending the IID model to test data gives statistical justification for testing

Training data \((\vec{X}_1, Y_1), \ldots, (\vec{X}_n, Y_n)\), test data \((\vec{X}_1', Y_1'), \ldots, (\vec{X}_m', Y_m')\), and “future” example \((\vec{X}, Y)\) are IID

Key observations in this model:

- Training data are random, and hence so is classifier \(f\) learned from training data
- If learned classifier \(f\) is based only on training data, then test data are independent of \(f\)

Conditional on \(f\), the learned classifier \(f\) has some probability of prediction error on \((\vec{X}, Y)\):

\[
\varepsilon := \Pr(f(\vec{X}) \neq Y \mid f)
\]

Goal of testing: Estimate \(\varepsilon\) (conditional on \(f\))
The test error rate of $f$ is

$$\hat{\varepsilon} := \frac{Z_1 + \cdots + Z_m}{m}$$

where $Z_i := \begin{cases} 1 & \text{if } f(X_i') \neq Y_i \\ 0 & \text{otherwise} \end{cases}$ for all $i = 1, \ldots, m$.
Test error rate

The test error rate of $f$ is

$$\hat{\epsilon} := \frac{Z_1 + \cdots + Z_m}{m}$$

where

$$Z_i := \begin{cases} 
1 & \text{if } f(\vec{X}_i) \neq Y_i \\ 
0 & \text{otherwise}
\end{cases}$$

for all $i = 1, \ldots, m$

Recall that test data is independent of $f$ (since $f$ assumed to only be based on training data)
Test error rate

The test error rate of $f$ is

$$\hat{\varepsilon} := \frac{Z_1 + \cdots + Z_m}{m}$$

where

$$Z_i := \begin{cases} 1 & \text{if } f(\vec{X}_i') \neq Y_i \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } i = 1, \ldots, m$$

Recall that test data is independent of $f$ (since $f$ assumed to only be based on training data)

So conditional on $f$, random variables $(Z_1, \ldots, Z_m \mid f) \sim_{iid} \text{Bernoulli}(\theta)$ for

$$\theta = \mathbb{E}[Z_1 \mid f] = \Pr(Z_1 = 1 \mid f)$$
The test error rate of $f$ is

$$
\hat{\varepsilon} := \frac{Z_1 + \cdots + Z_m}{m}
$$

where

$$
Z_i := \begin{cases} 
1 & \text{if } f(\vec{X}_i') \neq Y_i \\
0 & \text{otherwise}
\end{cases}
$$

for all $i = 1, \ldots, m$.

Recall that test data is independent of $f$ (since $f$ assumed to only be based on training data)

So conditional on $f$, random variables $(Z_1, \ldots, Z_m \mid f) \sim \text{iid Bernoulli}(\theta)$ for

$$
\theta = \mathbb{E}[Z_1 \mid f] = \Pr(Z_1 = 1 \mid f)
$$

$$
= \Pr(f(\vec{X}_1') \neq Y_1' \mid f)
$$
The test error rate of $f$ is

$$\hat{\varepsilon} := \frac{Z_1 + \cdots + Z_m}{m}$$

where

$$Z_i := \begin{cases} 1 & \text{if } f(\vec{X}_i') \neq Y_i \\ 0 & \text{otherwise} \end{cases}$$

for all $i = 1, \ldots, m$

Recall that test data is independent of $f$ (since $f$ assumed to only be based on training data)

So conditional on $f$, random variables $(Z_1, \ldots, Z_m \mid f) \sim \text{iid Bernoulli}(\theta)$ for

$$\theta = \mathbb{E}[Z_1 \mid f] = \Pr(Z_1 = 1 \mid f)$$

$$= \Pr(f(\vec{X}_1') \neq Y_1' \mid f)$$

$$= \varepsilon$$

(Uses assumption that $(\vec{X}_1', Y_1')$ & $(\vec{X}, Y)$ are identically distributed, even conditional on $f$)
The test error rate of $f$ is

$$
\hat{\epsilon} := \frac{Z_1 + \cdots + Z_m}{m}
$$

where

$$
Z_i := \begin{cases} 
1 & \text{if } f(\vec{X}_i') \neq Y_i \\
0 & \text{otherwise}
\end{cases}
$$

for all $i = 1, \ldots, m$.

Recall that test data is independent of $f$ (since $f$ assumed to only be based on training data).

So conditional on $f$, random variables $(Z_1, \ldots, Z_m | f) \sim_{iid} \text{Bernoulli}(\theta)$ for

$$
\theta = \mathbb{E}[Z_1 | f] = \Pr(Z_1 = 1 | f)
$$

$$
= \Pr(f(\vec{X}_1') \neq Y_1' | f)
$$

$$
= \epsilon
$$

(Uses assumption that $(\vec{X}_1', Y_1')$ & $(\vec{X}, Y)$ are identically distributed, even conditional on $f$)

Conditional on $f$, numerator in definition of test error rate $\hat{\epsilon}$:

- Sum of $m$ independent $\text{Bernoulli}(\epsilon)$ random variables $S := Z_1 + \cdots + Z_m$

- Conditional distribution of $S$ is called **binomial** with $m$ trials and success parameter $\epsilon$:

$$
(S | f) \sim \text{Binomial}(m, \epsilon)
$$
Quality of the test error rate

**Facts about binomial distribution** $\text{Binomial}(m, \varepsilon)$:

- Mean: $m\varepsilon$
- Standard deviation: $\sqrt{m\varepsilon(1-\varepsilon)}$

Therefore, (conditional) mean and standard deviation of test error rate $\hat{\varepsilon}$ are

$$E(\hat{\varepsilon} | f) = \varepsilon$$

$$\text{stddev}(\hat{\varepsilon} | f) = \sqrt{\varepsilon(1-\varepsilon)}$$

First statement says that test error rate is an unbiased estimator of the (actual) error rate.

Standard deviation measures spread of estimator's distribution; decreasing with $m$ (Explains role of size of test set!)
Quality of the test error rate

Facts about binomial distribution $\text{Binomial}(m, \varepsilon)$:

- Mean: $m\varepsilon$
- Standard deviation: $\sqrt{m\varepsilon(1 - \varepsilon)}$

Therefore, (conditional) mean and standard deviation of test error rate $\hat{\varepsilon} = \frac{S}{m}$ are

$$
\mathbb{E}(\hat{\varepsilon} \mid f) = \varepsilon
$$

$$
\text{stddev}(\hat{\varepsilon} \mid f) = \sqrt{\frac{\varepsilon(1 - \varepsilon)}{m}}
$$
Quality of the test error rate

**Facts about binomial distribution** Binomial$(m, \varepsilon)$:

- Mean: $m\varepsilon$
- Standard deviation: $\sqrt{m\varepsilon(1 - \varepsilon)}$

Therefore, (conditional) mean and standard deviation of test error rate $\hat{\varepsilon} = \frac{S}{m}$ are:

$$\mathbb{E}(\hat{\varepsilon} \mid f) = \varepsilon$$

$$\text{stddev}(\hat{\varepsilon} \mid f) = \sqrt{\frac{\varepsilon(1 - \varepsilon)}{m}}$$

- First statement says that test error rate is an **unbiased** estimator of the (actual) error rate
Facts about binomial distribution \text{Binomial}(m, \varepsilon): 

- **Mean:** \( m \varepsilon \)
- **Standard deviation:** \( \sqrt{m \varepsilon (1 - \varepsilon)} \)

Therefore, (conditional) mean and standard deviation of test error rate \( \hat{\varepsilon} = \frac{S}{m} \) are 

\[
\begin{align*}
\mathbb{E}(\hat{\varepsilon} \mid f) &= \varepsilon \\
\text{stddev}(\hat{\varepsilon} \mid f) &= \sqrt{\frac{\varepsilon (1 - \varepsilon)}{m}}
\end{align*}
\]

- First statement says that test error rate is an **unbiased** estimator of the (actual) error rate
- Standard deviation measures spread of estimator’s distribution; decreasing with \( m \) 
  (Explains role of size of test set!)
What goes wrong if we train on test data?

If classifier $f$ was trained also on test data:

- $Z_1, \ldots, Z_m$ not necessarily conditionally IID
- Conditional distribution of $(\vec{X}_1', Y_1')$ could differ from that of $(\vec{X}, Y)$ given $f$
- Entire previous analysis about test error rate would be invalid!
Multi-class Classification
Multi-class classification

Statistical model for multi-class classification:

- Outcome/label $Y$ is random variable taking values in a finite, unordered set $\{1, 2, \ldots, K\}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application
Multi-class classification

Statistical model for multi-class classification:

- Outcome/label $Y$ is random variable taking values in a finite, unordered set $\{1, 2, \ldots, K\}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

- Distribution of $Y$ models outcome of rolling a $K$-sided die
  (But values on die faces are ignored; may as well be “apple”, “banana”, “cantaloupe”, etc.)
Multi-class classification

**Statistical model for multi-class classification:**

- Outcome/label $Y$ is random variable taking values in a finite, unordered set $\{1, 2, \ldots, K\}$
- Feature vector is a vector of $d$ random variables $\mathbf{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\mathbf{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

- Distribution of $Y$ models outcome of rolling a $K$-sided die
  (But values on die faces are ignored; may as well be “apple”, “banana”, “cantaloupe”, etc.)
- Standard benchmark: Error rate (same as in binary classification)
Regression
Regression

Statistical model for regression:

- Outcome/label $Y$ is real-valued random variable taking values in (a subset of) $\mathbb{R}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

Ordering and structure of real line is important in the distribution of $Y$

Prediction that is close to $Y$ is better than one that is far from $Y$

Common benchmark for predictor $f$: mean squared error (MSE)

$$E\left[ (f(\vec{X}) - Y)^2 \right]$$

If labels always take values 1 or 0, and predictions of $f$ are also always 1 or 0, then MSE is same as probability of error
Regression

Statistical model for regression:

- Outcome/label $Y$ is real-valued random variable taking values in (a subset of) $\mathbb{R}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

- Ordering and structure of real line is important in the distribution of $Y$
Regression

**Statistical model for regression:**

- Outcome/label $Y$ is real-valued random variable taking values in (a subset of) $\mathbb{R}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

- Ordering and structure of real line is important in the distribution of $Y$
- Prediction that is close to $Y$ is better than one that is far from $Y$
Regression

Statistical model for regression:

- Outcome/label $Y$ is real-valued random variable taking values in (a subset of) $\mathbb{R}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

- Ordering and structure of real line is important in the distribution of $Y$
- Prediction that is close to $Y$ is better than one that is far from $Y$
- Common benchmark for predictor $f$: **mean squared error (MSE)**

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

Statistical model for regression:

- Outcome/label $Y$ is real-valued random variable taking values in (a subset of) $\mathbb{R}$
- Feature vector is a vector of $d$ random variables $\vec{X} := (X_1, \ldots, X_d)$
- Joint distribution of $(\vec{X}, Y)$ reflects the population of examples we anticipate encountering in the future for the present application

- Ordering and structure of real line is important in the distribution of $Y$
- Prediction that is close to $Y$ is better than one that is far from $Y$
- Common benchmark for predictor $f$: mean squared error (MSE)

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

- If labels always take values 1 or 0, and predictions of $f$ are also always 1 or 0, then MSE is same as probability of error
Aside: Useful property of mean squared error

Consider any real-valued random variable $Z$
Aside: Useful property of mean squared error

Consider any real-valued random variable $Z$

- **Question:** For what $t \in \mathbb{R}$ is $E[(Z - t)^2]$ as small as possible?
Aside: Useful property of mean squared error

Consider any real-valued random variable $Z$

- **Question:** For what $t \in \mathbb{R}$ is $\mathbb{E}[(Z - t)^2]$ as small as possible?
- **Answer:** $t = \mathbb{E}(Z)$, because for any $t \in \mathbb{R}$,

$$\mathbb{E}[(Z - t)^2] = (t - \mathbb{E}(Z))^2 + \mathbb{E}[(Z - \mathbb{E}(Z))^2]$$

Called bias-variance decomposition of MSE.
Aside: Useful property of mean squared error

Consider any real-valued random variable $Z$

- **Question:** For what $t \in \mathbb{R}$ is $\mathbb{E}[(Z - t)^2]$ as small as possible?
- **Answer:** $t = \mathbb{E}(Z)$, because for any $t \in \mathbb{R},$

$$
\mathbb{E}[(Z - t)^2] = (t - \mathbb{E}(Z))^2 + \mathbb{E}[(Z - \mathbb{E}(Z))^2]
$$

Called **bias-variance decomposition** of MSE
Structure of optimal predictors

**Question:** What are the predictions made by the optimal predictor? (I.e., the predictor \( f^* \) with smallest MSE)

- For each possible feature vector \( \vec{x} \), let \( \eta(\vec{x}) \) be the conditional mean of \( Y \) given \( \vec{X} = \vec{x} \):
  \[
  \eta(\vec{x}) := E[Y | \vec{X} = \vec{x}]
  \]

- By bias-variance decomposition, for any \( t \in \mathbb{R} \),
  \[
  E[(Y - t)^2 | \vec{X} = \vec{x}] = (t - \eta(\vec{x}))^2 + E[(Y - \eta(\vec{x}))^2 | \vec{X} = \vec{x}]
  \]

- Minimized when \( t = \eta(\vec{x}) \)

- So, optimal prediction upon \( \vec{X} = \vec{x} \) is:
  \[
  f^*(\vec{x}) := \eta(\vec{x})
  \]

Function \( \eta \) defined above is called the regression function for \( (\vec{X}, Y) \).
Question: What are the predictions made by the optimal predictor? (i.e., the predictor $f^\star$ with smallest MSE)

- For each possible feature vector $\vec{x}$, let $\eta(\vec{x})$ be the conditional mean of $Y$ given $\vec{X} = \vec{x}$:

$$\eta(\vec{x}) := \mathbb{E}[Y \mid \vec{X} = \vec{x}]$$
Question: What are the predictions made by the optimal predictor? (i.e., the predictor \( f^* \) with smallest MSE)

- For each possible feature vector \( \vec{x} \), let \( \eta(\vec{x}) \) be the conditional mean of \( Y \) given \( \vec{X} = \vec{x} \):

\[
\eta(\vec{x}) : = \mathbb{E}[Y \mid \vec{X} = \vec{x}]
\]

- By bias-variance decomposition, for any \( t \in \mathbb{R} \),

\[
\mathbb{E}[(Y - t)^2 \mid \vec{X} = \vec{x}] = (t - \eta(\vec{x}))^2 + \mathbb{E}[(Y - \eta(\vec{x}))^2 \mid \vec{X} = \vec{x}]
\]

Minimized when \( t = \eta(\vec{x}) \)
Structure of optimal predictors

**Question:** What are the predictions made by the optimal predictor? (i.e., the predictor $f^*$ with smallest MSE)

- For each possible feature vector $\vec{x}$, let $\eta(\vec{x})$ be the conditional mean of $Y$ given $\vec{X} = \vec{x}$:
  
  $$\eta(\vec{x}) := \mathbb{E}[Y \mid \vec{X} = \vec{x}]$$

- By bias-variance decomposition, for any $t \in \mathbb{R}$,
  
  $$\mathbb{E}[(Y - t)^2 \mid \vec{X} = \vec{x}] = (t - \eta(\vec{x}))^2 + \mathbb{E}[(Y - \eta(\vec{x}))^2 \mid \vec{X} = \vec{x}]$$

  Minimized when $t = \eta(\vec{x})$

- So, optimal prediction upon $\vec{X} = \vec{x}$ is:
  
  $$f^*(\vec{x}) := \eta(\vec{x})$$
**Structure of optimal predictors**

**Question:** What are the predictions made by the optimal predictor? (i.e., the predictor $f^*$ with smallest MSE)

- For each possible feature vector $\vec{x}$, let $\eta(\vec{x})$ be the conditional mean of $Y$ given $\vec{X} = \vec{x}$:

$$
\eta(\vec{x}) := \mathbb{E}[Y \mid \vec{X} = \vec{x}]
$$

- By bias-variance decomposition, for any $t \in \mathbb{R}$,

$$
\mathbb{E}[(Y - t)^2 \mid \vec{X} = \vec{x}] = (t - \eta(\vec{x}))^2 + \mathbb{E}[(Y - \eta(\vec{x}))^2 \mid \vec{X} = \vec{x}]
$$

Minimized when $t = \eta(\vec{x})$

- So, optimal prediction upon $\vec{X} = \vec{x}$ is:

$$
f^*(\vec{x}) := \eta(\vec{x})
$$

Function $\eta$ defined above is called the **regression function** for $(\vec{X}, Y)$
Decision trees for regression

Decision trees for regression: regression trees

Just three changes to greedy training heuristic:

▶ Objective function:
  \[ \text{sse}(T; S) := \sum_{(\vec{x},y) \in S} (f_T(\vec{x}) - y)^2 \]  
  (instead of number of classification mistakes)

▶ Return value at leaf node \( \ell \):
  sample average of labels in training examples \( S \) routed to \( \ell \) by \( T \):
  \[ \text{return-value}(T)(\ell) := \frac{1}{|S_\ell|} \sum_{(\vec{x},y) \in S_\ell} y \]  
  (instead of plurality label)

▶ Uncertainty of data set \( S \)′:
  sample variance of labels in \( S \)′:
  \[ \text{uncertainty}(S_\prime) := \frac{1}{|S_\prime|} \sum_{(\vec{x},y) \in S_\prime} y^2 - \left( \frac{1}{|S_\prime|} \sum_{(\vec{x},y) \in S_\prime} y \right)^2 \]
Decision trees for regression: regression trees

Just three changes to greedy training heuristic:

- Objective function: sum of squared errors

\[
\text{sse}(T; S) := \sum_{(\vec{x}, y) \in S} (f_T(\vec{x}) - y)^2
\]

(instead of number of classification mistakes)
Decision trees for regression: regression trees

Just three changes to greedy training heuristic:

▶ Objective function: sum of squared errors

\[
\text{sse}(T;S) := \sum_{(\vec{x},y) \in S} (f_T(\vec{x}) - y)^2
\]

(instead of number of classification mistakes)

▶ Return value at leaf node \(\ell\): sample average of labels in training examples \(S_\ell\) routed to \(\ell\) by \(T\)

\[
\text{return-value}_T(\ell) := \frac{1}{|S_\ell|} \sum_{(\vec{x},y) \in S_\ell} y
\]

(instead of plurality label)
Decision trees for regression: regression trees

Just three changes to greedy training heuristic:

- **Objective function:** sum of squared errors

  \[
  \text{sse}(T; S) := \sum_{(\vec{x}, y) \in S} (f_T(\vec{x}) - y)^2
  \]

  (instead of number of classification mistakes)

- **Return value at leaf node \( \ell \): sample average of labels in training examples \( S_\ell \) routed to \( \ell \) by \( T \)

  \[
  \text{return-value}_T(\ell) := \frac{1}{|S_\ell|} \sum_{(\vec{x}, y) \in S_\ell} y
  \]

  (instead of plurality label)

- **Uncertainty of data set \( S' \): sample variance of labels in \( S' \)

  \[
  \text{uncertainty}(S') := \left( \frac{1}{|S'|} \sum_{(\vec{x}, y) \in S'} y^2 \right) - \left( \frac{1}{|S'|} \sum_{(\vec{x}, y) \in S'} y \right)^2
  \]
Approximation error and variability

Recall, in context of IID model for training data:

- Predictor $f$ learned from training data is random (because training data is random)
Approximation error and variability

Recall, in context of IID model for training data:

- Predictor $f$ learned from training data is random (because training data is random)
- Therefore, for any feature vector $\vec{x}$, prediction $f(\vec{x})$ is a random variable

Theorem. Assume the IID model of training data (with regression function $\eta$), and assume predictor $f$ depends only on the training data. Then for any feature vector $\vec{x}$:

$$E\left[ (f(\vec{x}) - Y)^2 \mid \vec{X} = \vec{x} \right] = E\left[ (Y - \eta(\vec{x}))^2 \mid \vec{X} = \vec{x} \right] (\text{error of optimal predictor}) + \left( E[f(\vec{x})] - \eta(\vec{x}) \right)^2 (\text{approximation error}) + \text{var}(f(\vec{x})) (\text{variability})$$
Approximation error and variability

Recall, in context of IID model for training data:

▶ Predictor $f$ learned from training data is random (because training data is random)
▶ Therefore, for any feature vector $\vec{x}$, prediction $f(\vec{x})$ is a random variable

**Theorem.** Assume the IID model of training data (with regression function $\eta$), and assume predictor $f$ depends only on the training data. Then for any feature vector $\vec{x}$:

$$
\mathbb{E}[(f(\vec{x}) - Y)^2 | \vec{X} = \vec{x}] = \mathbb{E}[(Y - \eta(\vec{x}))^2 | \vec{X} = \vec{x}] + (\mathbb{E}[f(\vec{x})] - \eta(\vec{x}))^2 + \text{var}(f(\vec{x}))
$$

(error of optimal predictor)

(approximation error)

(variability)
Approximation error and variability

Recall, in context of IID model for training data:

- Predictor $f$ learned from training data is random (because training data is random)
- Therefore, for any feature vector $\vec{x}$, prediction $f(\vec{x})$ is a random variable

**Theorem.** Assume the IID model of training data (with regression function $\eta$), and assume predictor $f$ depends only on the training data. Then for any feature vector $\vec{x}$:

\[
\mathbb{E}[(f(\vec{x}) - Y)^2 | \vec{X} = \vec{x}] = \mathbb{E}[(Y - \eta(\vec{x}))^2 | \vec{X} = \vec{x}] + (\mathbb{E}[f(\vec{x})] - \eta(\vec{x}))^2 + \text{var}(f(\vec{x}))
\]

(error of optimal predictor) (approximation error) (variability)

Approximation error and variability are typically in tension with each other:
**Approximation error and variability**

Recall, in context of IID model for training data:

- Predictor $f$ learned from training data is random (because training data is random)
- Therefore, for any feature vector $\vec{x}$, prediction $f(\vec{x})$ is a random variable

**Theorem.** Assume the IID model of training data (with regression function $\eta$), and assume predictor $f$ depends only on the training data. Then for any feature vector $\vec{x}$:

$$
\mathbb{E}[(f(\vec{x}) - Y)^2 \mid \vec{X} = \vec{x}] = \mathbb{E}[(Y - \eta(\vec{x}))^2 \mid \vec{X} = \vec{x}] \\
+ (\mathbb{E}[f(\vec{x})] - \eta(\vec{x}))^2 \\
+ \text{var}(f(\vec{x}))
$$

(error of optimal predictor)
(approximation error)
(variability)

Approximation error and variability are typically in tension with each other:

- Flexible models (e.g., large decision trees) are capable of achieving lower approximation error, but have higher variability due to ability to fit any data set
Approximation error and variability

Recall, in context of IID model for training data:

- Predictor $f$ learned from training data is random (because training data is random)
- Therefore, for any feature vector $\vec{x}$, prediction $f(\vec{x})$ is a random variable

**Theorem.** Assume the IID model of training data (with regression function $\eta$), and assume predictor $f$ depends only on the training data. Then for any feature vector $\vec{x}$:

$$
E[(f(\vec{x}) - Y)^2 \mid \vec{X} = \vec{x}] = E[(Y - \eta(\vec{x}))^2 \mid \vec{X} = \vec{x}] \\
+ (E[f(\vec{x})] - \eta(\vec{x}))^2 \\
+ \text{var}(f(\vec{x}))
$$

(error of optimal predictor)

(approximation error)

(variability)

Approximation error and variability are typically in tension with each other:

- Flexible models (e.g., large decision trees) are capable of achieving lower approximation error, but have higher variability due to ability to fit any data set
- Simpler models (e.g., small decision trees) may incur higher approximation error, but restricted flexibility ensures a good fit to training data will “generalize” to distribution of $(\vec{X}, Y)$
Illustration of approximation error and variability

$\vec{X}$ uniformly distributed over $[-1, +1]^2$

$$\mathbb{E}[Y \mid \vec{X} = \vec{x}] = \frac{x_1 + x_2}{2}$$

Learn size-$L$ regression tree with $n = 100$ training data; then examine prediction at $\vec{x}^* = (0.5, 0.5)$

(Note: $\mathbb{E}[Y \mid \vec{X} = \vec{x}^*] = 0.5$)
Illustration of approximation error and variability

$\vec{X}$ uniformly distributed over $[-1, +1]^2$

$$\mathbb{E}[Y \mid \vec{X} = \vec{x}] = \frac{x_1 + x_2}{2}$$

Learn size-$L$ regression tree with $n = 100$ training data; then examine prediction at $\vec{x}^* = (0.5, 0.5)$

(Note: $\mathbb{E}[Y \mid \vec{X} = \vec{x}^*] = 0.5$)

Repeat experiment 1000 times (new training data each time):

<table>
<thead>
<tr>
<th>Regression tree size</th>
<th>$L = 2$</th>
<th>$L = 3$</th>
<th>$L = 12$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. of predictions at $\vec{x}^*$</td>
<td>0.30</td>
<td>0.42</td>
<td>0.50</td>
</tr>
<tr>
<td>Std. dev. of predictions at $\vec{x}^*$</td>
<td>0.17</td>
<td>0.21</td>
<td>0.32</td>
</tr>
</tbody>
</table>
Illustration of approximation error and variability

\[ \vec{X} \text{ uniformly distributed over } [-1, +1]^2 \]

\[ \mathbb{E}[Y \mid \vec{X} = \vec{x}] = \frac{x_1 + x_2}{2} \]

Learn size-$L$ regression tree with $n = 100$ training data; then examine prediction at $\vec{x}^* = (0.5, 0.5)$

(Note: $\mathbb{E}[Y \mid \vec{X} = \vec{x}^*] = 0.5$)

Repeat experiment 1000 times (new training data each time):

<table>
<thead>
<tr>
<th>regression tree size</th>
<th>$L = 2$</th>
<th>$L = 3$</th>
<th>$L = 12$</th>
</tr>
</thead>
<tbody>
<tr>
<td>avg. of predictions at $\vec{x}^*$</td>
<td>0.30</td>
<td>0.42</td>
<td>0.50</td>
</tr>
<tr>
<td>std. dev. of predictions at $\vec{x}^*$</td>
<td>0.17</td>
<td>0.21</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Predictions of size-3 regression trees
Recap

- Many roles for statistical models
  - Formalize prediction problem & goal of supervised learning
  - Explain role of training data
  - Motivate “testing” & clarify what it can (and cannot) do
- Plug-in principle (more later!)
- Some types of prediction problems: classification, regression
- Bias-variance decomposition, approximation error versus variability