Bayesian Decision Theory

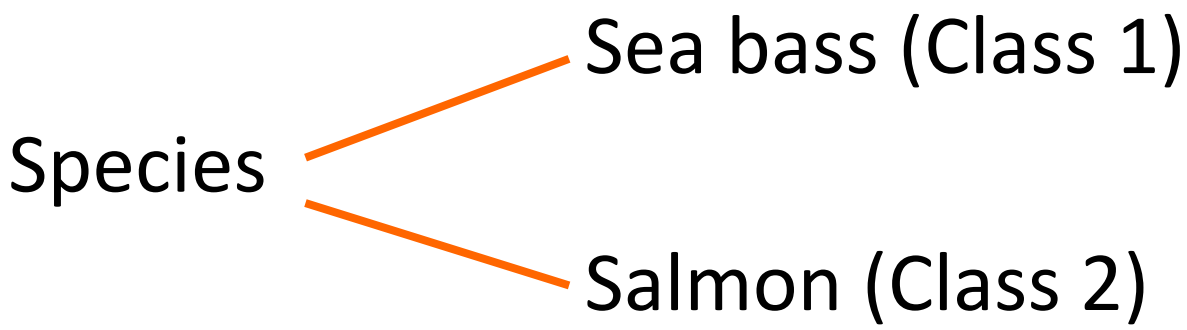
Primary source of reference: *Pattern Classification* – Duda and Hart

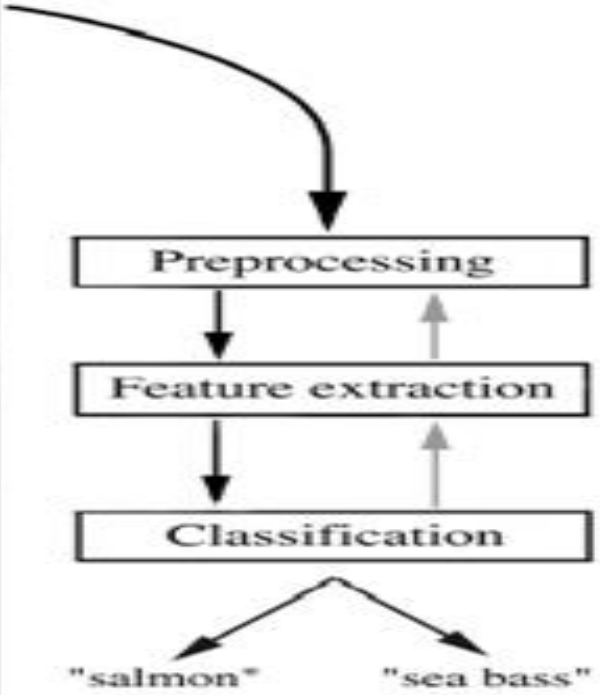
Introduction

Bayesian Decision Theory–Continuous Features

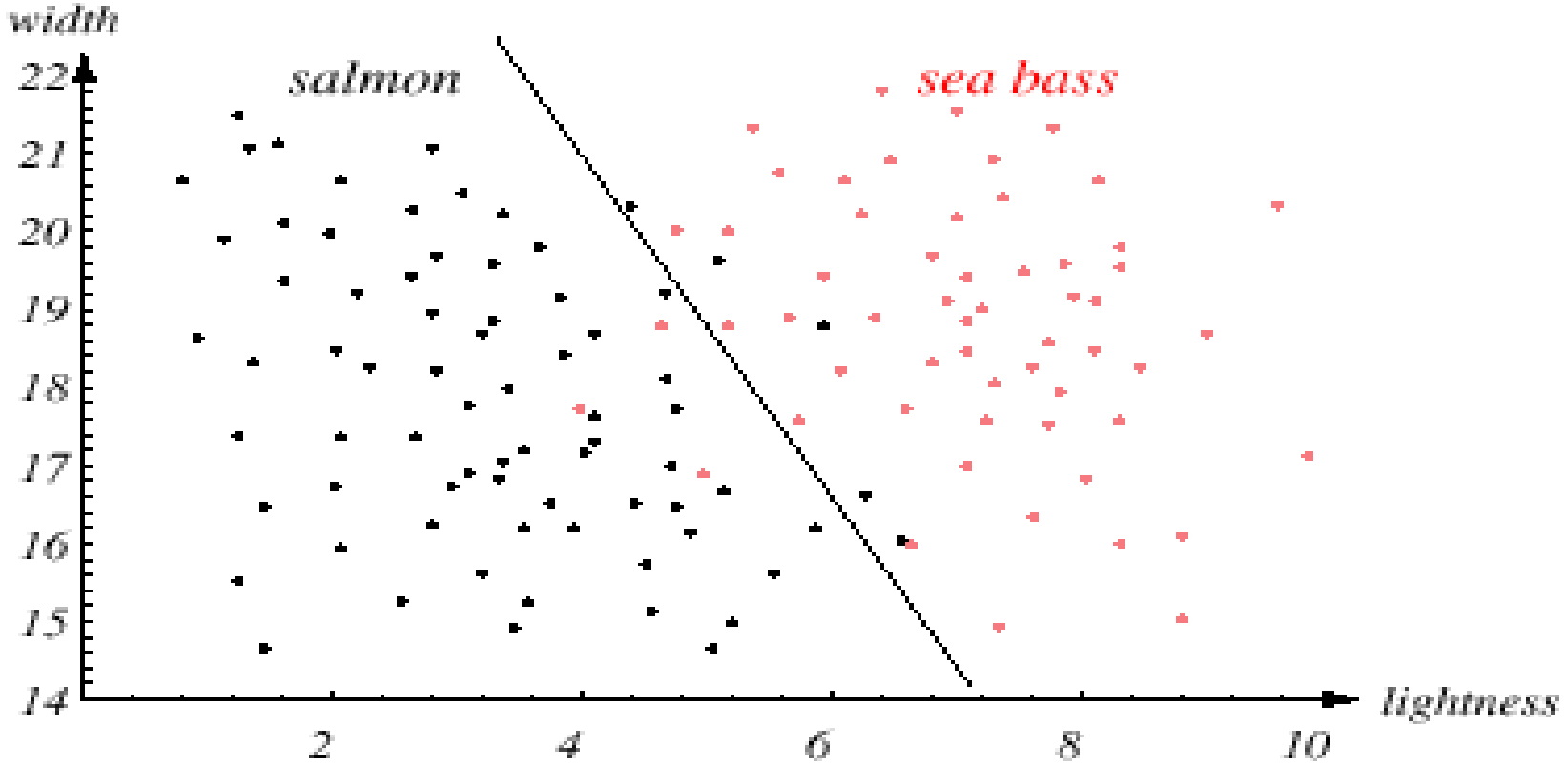
An Example

- "Sorting incoming Fish on a conveyor according to species using optical sensing"




Preprocessing
Feature extraction
Classification
"salmon"
"sea bass"

- Problem Analysis
  - Set up a camera and take some sample images to extract features like
    - Length of the fish
    - Lightness (based on the gray level)
    - Width of the fish


width
salmon
sea bass
lightness
22
21
20
19
18
17
16
15
14
2
4
6
8
10

Introduction

- The sea bass/salmon example (a two class problem)

- For example if we randomly catch 100 fishes and out of this if 75 are *sea bass* and 25 are *salmon*.
- Let the rule, in this case is: For any fish say its class is *sea bass*.
- What is the error rate of this rule?
- This information which is independent of feature values is called apriori knowledge.

- Let the two classes are ω_1 and ω_2
  - $P(\omega_1) + P(\omega_2) = 1$
  - State of nature (class) is a random variable
  - If $P(\omega_1) = P(\omega_2)$, we say it is of uniform priors
    - The catch of salmon and sea bass is equi-probable

- Decision rule with only the prior information
  - Decide ω_1 if $P(\omega_1) > P(\omega_2)$, otherwise decide ω_2
- *This is not a good classifier.*
- *We should take feature values into account !*
- *If x is the pattern we want to classify, then use the rule:*

If $P(\omega_1 \mid x) > P(\omega_2 \mid x)$ *then assign class* ω_1
Else assign class ω_2

- $P(\omega_1 \mid x)$ *is called posteriori probability of class* ω_1 *given that the pattern is x.*

Bayes rule

- From data it might be possible for us to estimate $p(x \mid \omega_j)$, *where i = 1 or 2.* These are called class-conditional distributions.
- Also it is easy to find apriori probabilities $P(\omega_1)$ and $P(\omega_2)$. How this can be done?
- Bayes rule combines apriori probability with class conditional distributions to find posteriori probabilities.

Bayes Rule

$$P(B|A) = \frac{P(A, B)}{P(A)} = \frac{P(A|B) * P(B)}{P(A)}$$

This is Bayes Rule

Bayes, Thomas (1763) An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, **53:370-418**

$$P(\omega_j \mid x) = \frac{p(x \mid \omega_j) \, . \, P(\omega_j)}{p(x)}$$

– Where in case of two categories

$$p(x) = \sum_{j=1}^{j=2} p(x \mid \omega_j) P(\omega_j)$$

– Posterior = $\frac{\text{Likelihood} \, . \, \text{Prior}}{\text{Evidence}}$

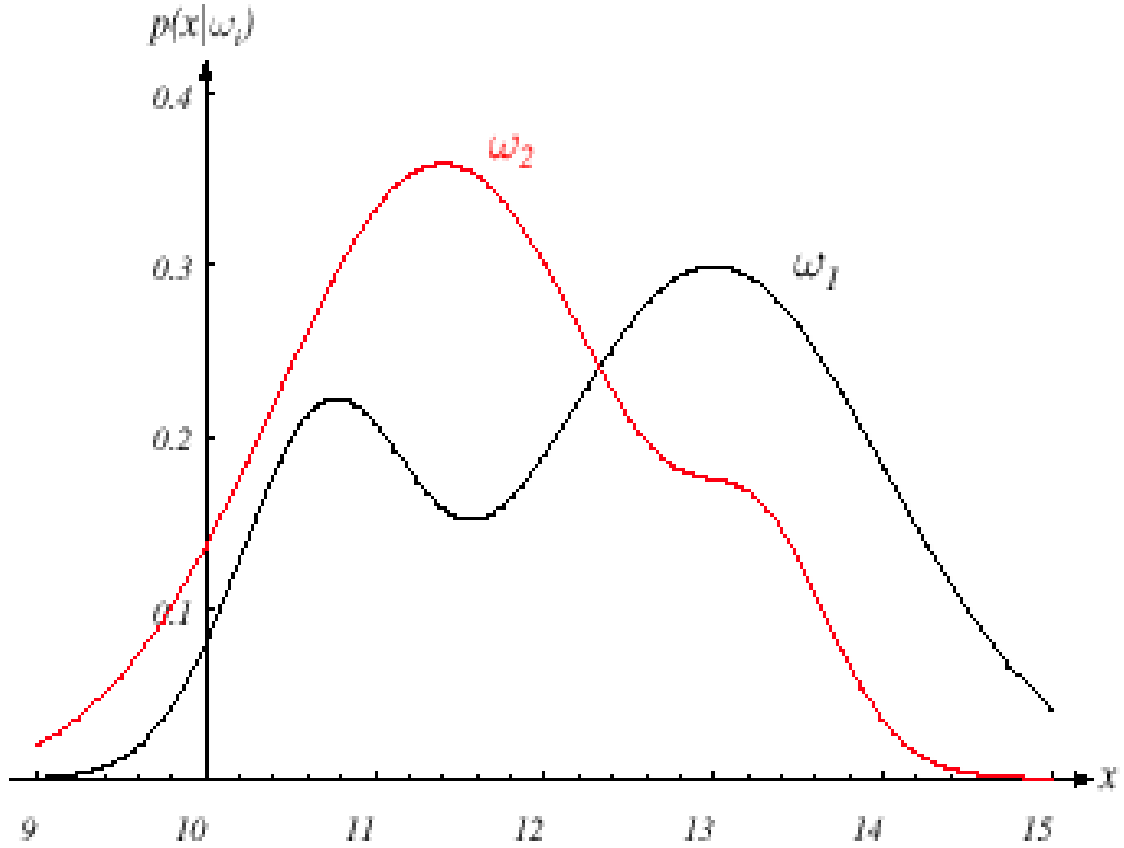


FIGURE 2.1. Hypothetical class-conditional probability density functions show the probability density of measuring a particular feature value x given the pattern is in category ω_j. If x represents the lightness of a fish, the two curves might describe the difference in lightness of populations of two types of fish. Density functions are normalized, and thus the area under each curve is 1.0. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

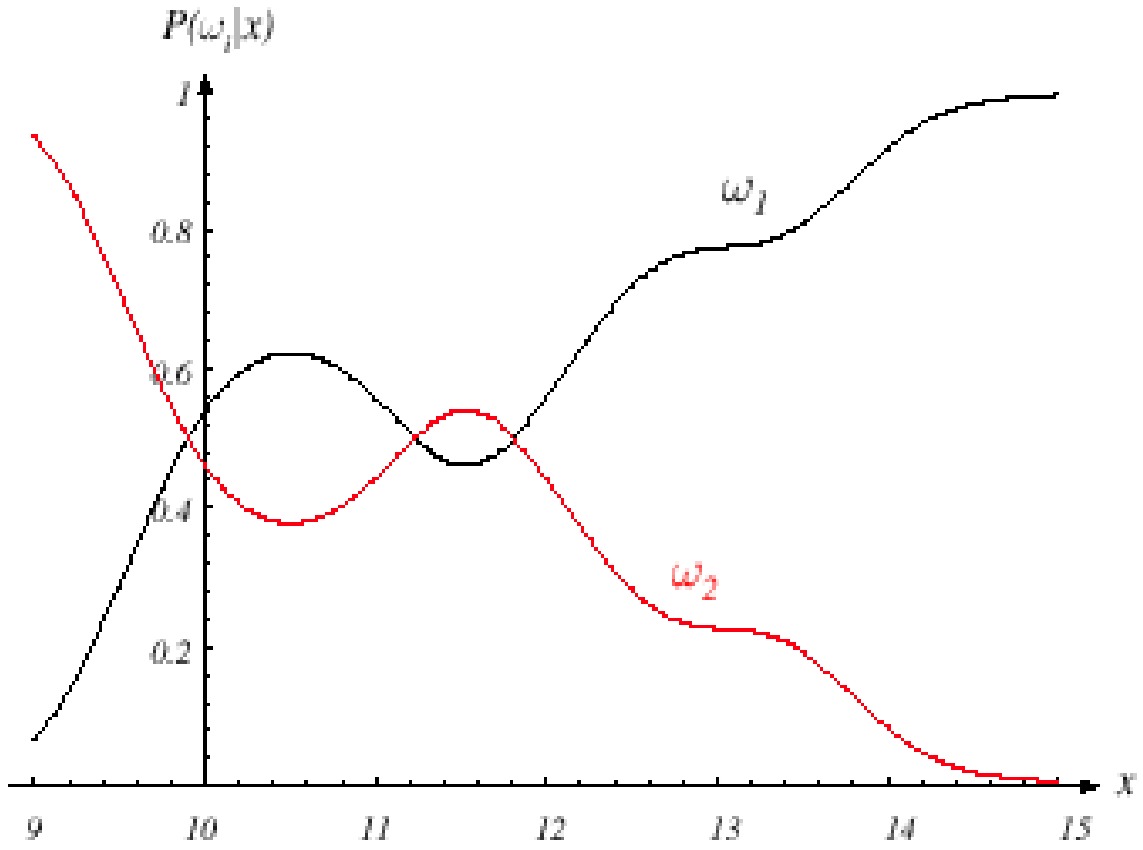


FIGURE 2.2. Posterior probabilities for the particular priors $P(\omega_1) = 2/3$ and $P(\omega_2) = 1/3$ for the class-conditional probability densities shown in Fig. 2.1. Thus in this case, given that a pattern is measured to have feature value $x = 14$, the probability it is in category ω_2 is roughly 0.08, and that it is in ω_1 is 0.92. At every x, the posteriors sum to 1.0. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

- Decision given the posterior probabilities

  X is an observation for which:

  if $P(\omega_1 \mid x) > P(\omega_2 \mid x)$ ⇒ True state of nature = ω_1
  if $P(\omega_1 \mid x) < P(\omega_2 \mid x)$ ⇒ True state of nature = ω_2

  Therefore:

  whenever we observe a particular x, the probability of error is :

  $P(error \mid x) = P(\omega_1 \mid x)$ *if we decide* ω_2
  $P(error \mid x) = P(\omega_2 \mid x)$ *if we decide* ω_1

- Minimizing the probability of error

- Decide ω_1 *if* $P(\omega_1 \mid x) > P(\omega_2 \mid x)$;
  otherwise decide ω_2

  Therefore:

$$P(\text{error} \mid x) = \min [P(\omega_1 \mid x), P(\omega_2 \mid x)]$$

(error of Bayes decision)

Average error rate

Average probability of error, P(error) is :

$$\int P(error \mid x)\, p(x)\, dx$$

This is the expected value of $P(error|x)$ $w.r.t.$ x ,

i.e., $E_x[P(error \mid x)]$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,$ $P(white \mid \omega_2) = 0.6,\quad P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,$ $P(white \mid \omega_2) = 0.6,\quad P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,\ P(white \mid \omega_2) = 0.6,\ P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.75 + 0.4 * 0.25 = 0.7$$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,\ P(white \mid \omega_2) = 0.6,\ P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.75 + 0.4 * 0.25 = 0.7$$

$$P(\omega_1|white) = \frac{P(white|\omega_1)P(\omega_1)}{P(white)} = \frac{0.2 * 0.75}{0.3} = 0.5$$

$$P(\omega_2|white) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)} = \frac{0.6 * 0.25}{0.3} = 0.5$$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,\ P(white \mid \omega_2) = 0.6,\ P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.75 + 0.4 * 0.25 = 0.7$$

$$P(\omega_1|white) = \frac{P(white|\omega_1)P(\omega_1)}{P(white)} = \frac{0.2 * 0.75}{0.3} = 0.5$$

$$P(\omega_2|white) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)} = \frac{0.6 * 0.25}{0.3} = 0.5$$

$$P(\omega_1|dark) = \frac{P(dark|\omega_1)P(\omega_1)}{P(dark)} = \frac{0.8 * 0.75}{0.7} = \frac{6}{7}$$

$$P(\omega_2|dark) = \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)} = \frac{0.4 * 0.25}{0.7} = \frac{1}{7}$$

$$P(error) = P(error|white)P(white) + P(error|dark)P(dark)$$

$$P(error) = 0.5 * 0.3 + \frac{1}{7} * 0.7 = 0.25$$

$$P(error) = P(error|white)P(white) + P(error|dark)P(dark)$$

$$P(error) = 0.5 * 0.3 + \frac{1}{7} * 0.7 = 0.25$$

- But, what is the error, if we use only apriori probabilities?

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

Since, $P(\omega_1) = 0.75$, $P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

$$P(error) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)}P(white) + \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)}P(dark)$$

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

$$P(error) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)}P(white) + \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)}P(dark)$$

$$P(error) = \big(P(white|\omega_2) + P(dark|\omega_2)\big)P(\omega_2)$$

$$P(error) = P(\omega_2) = 0.25$$

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

$$P(error) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)}P(white) + \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)}P(dark)$$

$$P(error) = \big(P(white|\omega_2) + P(dark|\omega_2)\big)P(\omega_2)$$

$$P(error) = P(\omega_2) = 0.25$$

- Same error? Where is the advantage?!

Consider $P(\omega_1) = 0.5,\ P(\omega_2) = 0.5$

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.5 + 0.6 * 0.5 = 0.4$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.5 + 0.4 * 0.5 = 0.6$$

$$P(\omega_1|white) = \frac{P(white|\omega_1)P(\omega_1)}{P(white)} = \frac{0.2 * 0.5}{0.4} = 0.25$$

$$P(\omega_2|white) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)} = \frac{0.6 * 0.5}{0.4} = 0.75$$

$$P(\omega_1|dark) = \frac{P(dark|\omega_1)P(\omega_1)}{P(dark)} = \frac{0.8 * 0.5}{0.6} = \frac{2}{3}$$

$$P(\omega_2|dark) = \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)} = \frac{0.4 * 0.5}{0.6} = \frac{1}{3}$$

$$P(error) = P(error|white)P(white) + P(error|dark)P(dark)$$

$$P(error) = 0.25 * 0.4 + \frac{1}{3} * 0.6 = 0.3$$

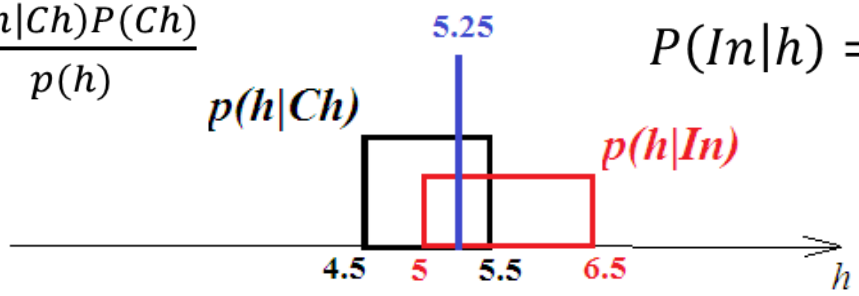
- But, P(error) based on apriori probabilities only is 0.5.
- Error based on the Bayes classifier is the lower bound.
  - Any classifier's error is greater than or equal to this.
- One can prove this!

2. We know that 30% of people in Sri City are from China. Remaining are Indians. We know that Chinese height is a uniformly distributed random variable with parameters 4.5 and 5.5. We also know that Indians height is also uniformly distributed with parameters 5 and 6.5. By measuring his/her height we want to classify the person as "Chinese" or "Indian" (We know that the person is living in Sri City). We follow the rule "if height is 5.25 or below classify the person as Chinese, otherwise classify the person as Indian". There are two types of mistakes in this classification, (1) Chinese being classified as Indians, (2) Indians being classified as Chinese. Find the probability of making each of these two types of mistakes.

-

$$P(Ch) = 0.3 \qquad P(In) = 0.7$$

$$P(Ch|h) = \frac{p(h|Ch)P(Ch)}{p(h)} \qquad P(In|h) = \frac{p(h|In)P(In)}{p(h)}$$



$$p(h) = p(h|Ch)P(Ch) + p(h|In)P(In)$$

$$= \begin{cases} 0.3, & \text{for } h \text{ in } [4.5, 5] \\ 1 * 0.3 + \frac{2}{3} * 0.7 = 0.767, & \text{for } h \text{ in } [5, 5.5] \\ \frac{2}{3} * 0.7 = 0.467, & \text{for } h \text{ in } [5.5, 6.5] \end{cases}$$

- For $h \in [4.5,\ 5]$

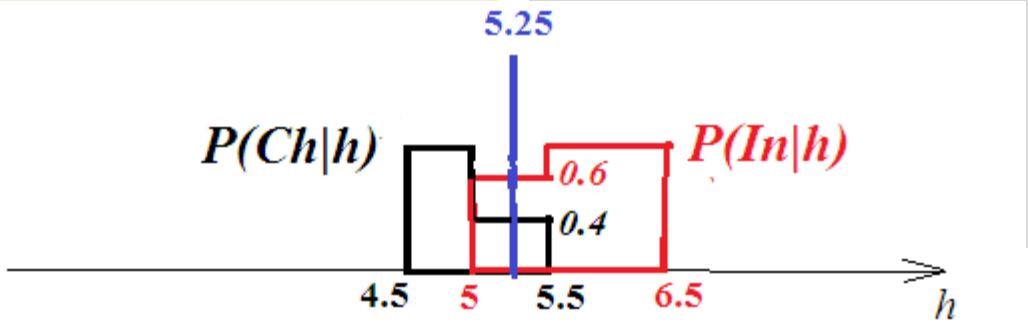
$P(Ch \mid h) = (1 * 0.3)/0.3 = 1$ $\quad P(In|h) = 0$

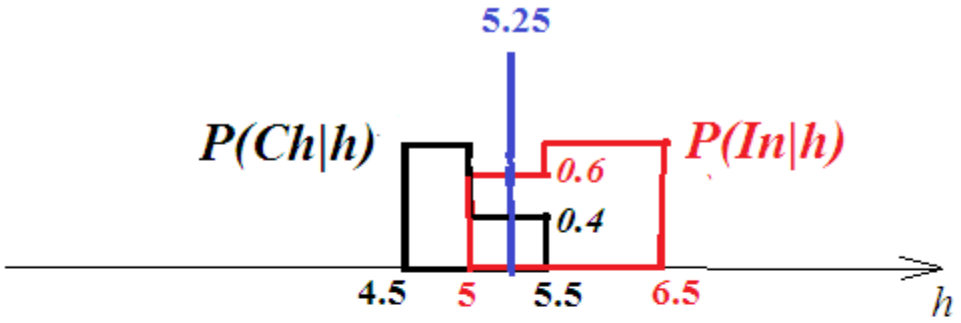
For $h \in [5,\ 5.5]$

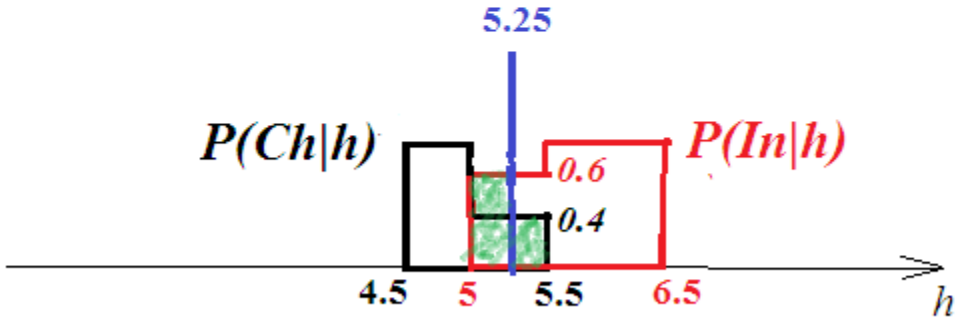
$P(Ch \mid h) = (1 * 0.3)/0.767 = 0.39$ $\quad P(In \mid h) = 0.67 * 0.7/0.767 = 0.61$

For $h \in [5,\ 6.5]$

$P(Ch \mid h) = 0$ $\quad P(In|h) = 1$







Error is due to this region

$$P(\text{error}) = \int_{5}^{5.25} 0.609\, p(h)dh + \int_{5.25}^{5.5} 0.391\, p(h)dh$$

$$= 0.117 + 0.075 = 0.192$$

- But, this is not the Bayes classifier.
- What does the Bayes classifier do?
- What is the error of the Bayes classifier?

Bayesian Decision Theory – Continuous Features

- Generalization of the preceding ideas
  - Use of more than one feature
  - Use more than two states (classes) of nature
  - Allowing actions (decisions) other than just classification.
  - Introduce a *loss function* which is more general than the probability of error.

- Allowing actions other than classification primarily allows the possibility of rejection
- Refusing to make a decision in close or bad cases!
- The loss function states how costly each action taken is

Problem setting

What is given to us:

Loss function : $\lambda(\alpha_i \mid \omega_j)$ is the loss of taking action α_i when the state of nature is ω_j

Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature (or "categories" or "classes")

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be the set of possible actions

For example : α_1 is the action *ring the alarm*

α_2 is the action *shutdown the system*

Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature (or "categories" or "classes")

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be the set of possible actions

For example : α_1 is the action *ring the alarm*

α_2 is the action *shutdown the system*

Objective: Given a pattern x, find the action to take.

That is, to find a function $\boldsymbol{\alpha}(x)$ which maps x to action.

How to find the best action?

Let $R(\alpha_i \mid x)$ is the risk of taking action α_i when the given pattern is x (this is called conditional risk).

We can take the action α_k provided $R(\alpha_k \mid x)$ is minimum in $\{ R(\alpha_1 \mid x), R(\alpha_2 \mid x), \ldots, R(\alpha_a \mid x) \}$

How to relate $R(\alpha_i \mid x)$ with $\lambda(\alpha_i \mid \omega_j)$ values.

$$R(\alpha_i \mid x) = \sum_{j=1}^{j=c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x)$$

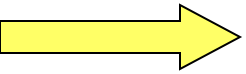
α(x) is given, how good is this?

- Overall risk of this rule can be found

$$R = \int R(\alpha(\mathbf{x})|\mathbf{x})p(\mathbf{x})\ d\mathbf{x},$$

- The smaller this quantity, better the decision rule.

Select the action α_i for which $R(\alpha_i \mid x)$ is minimum



R is minimum and R in this case is called the

Bayes risk = best performance that can be achieved!

- Two-category classification

α_1 : deciding ω_1

α_2 : deciding ω_2

$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$

loss incurred for deciding ω_i when the true state of nature is ω_j

Conditional risk:

$$R(\alpha_1 \mid x) = \lambda_{11}P(\omega_1 \mid x) + \lambda_{12}P(\omega_2 \mid x)$$
$$R(\alpha_2 \mid x) = \lambda_{21}P(\omega_1 \mid x) + \lambda_{22}P(\omega_2 \mid x)$$

Our rule is the following:

if $R(\alpha_1 \mid x) < R(\alpha_2 \mid x)$

action α_1: "decide ω_1" is taken

This results in the equivalent rule :

decide ω_1 if:

$$(\lambda_{21} - \lambda_{11})\, p(x \mid \omega_1)\, P(\omega_1) > (\lambda_{12} - \lambda_{22})\, p(x \mid \omega_2)\, P(\omega_2)$$

and decide ω_2 otherwise

An Example

- Let the two actions are:

$\alpha_1 \Rightarrow$ ***x is criminal***

$\alpha_2 \Rightarrow$ ***x is innocent***

Let $\lambda(\alpha_1 \mid \omega_1) = 0;$ $\lambda(\alpha_1 \mid \omega_2) = 10;$

$\lambda(\alpha_2 \mid \omega_1) = 1;$ $\lambda(\alpha_2 \mid \omega_2) = 0;$

Assume equal priors, and

Let $p(x \mid \omega_1) = 0.8,$ $p(x \mid \omega_2) = 0.6$

What action you will take?

ω_1 is class of criminals,

ω_2 is class of innocents.

Example: Contd…

- $P(\omega_j \mid x) = \dfrac{p(x \mid \omega_j) \,.\, P(\omega_j)}{p(x)}$

- $P(\omega_1 \mid x) = \dfrac{0.8(0.5)}{0.8(0.5) + 0.6(0.5)} = 8/14 = 0.57$

- $P(\omega_2 \mid x) = 0.43$

- *As per plain Bayes rule we declare the person to be a criminal.*

Example: Contd …

- $R(\alpha_1 \mid x) = \lambda(\alpha_1 \mid \omega_1) P(\omega_1 \mid x) + \lambda(\alpha_1 \mid \omega_2) P(\omega_2 \mid x)$
  $= 0 (0.57) + 10 (0.43)$
  $= 4.3$

- $R(\alpha_2 \mid x) = \lambda(\alpha_2 \mid \omega_1) P(\omega_1 \mid x) + \lambda(\alpha_2 \mid \omega_2) P(\omega_2 \mid x)$
  $= 1 (0.57) + 0 (0.43)$
  $= 0.57$

Action taken: x is innocent

Likelihood ratio:

The preceding rule is equivalent to the following rule:

$$if \ \frac{p(x \mid \omega_1)}{p(x \mid \omega_2)} > \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Then take action α_1 (decide ω_1)

Otherwise take action α_2 (decide ω_2)

Example 1

- Let $\Omega = \{\omega_1, \omega_2, \omega_3\}$, $A = \{\alpha_1, \alpha_2, \alpha_3\}$

  $P(\omega_1 | x) = \mathbf{0.1}$, $P(\omega_2 | x) = \mathbf{0.4}$, $P(\omega_3 | x) = \mathbf{0.5}$

  Loss function is

Loss	ω_1	ω_2	ω_3
α_1	0	1	2
α_2	1	0	2
α_3	3	10	0

Find $R(\alpha_k | x)$ for $k = 1, 2, 3$.

Find what is the best action?

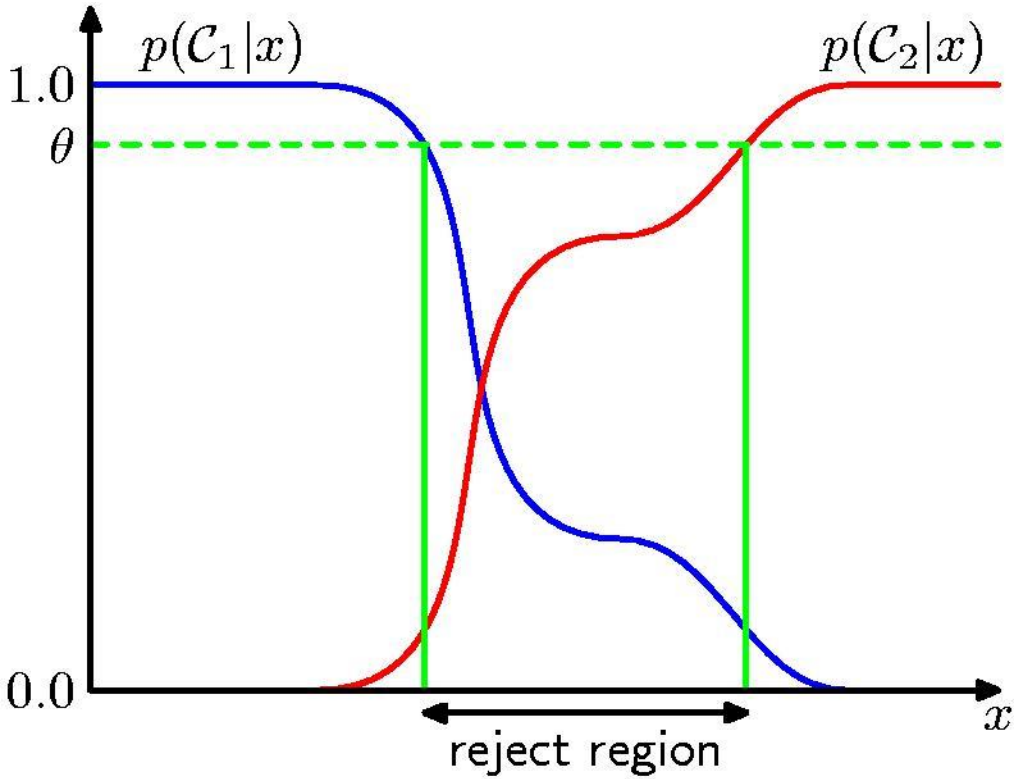
Reject option (I do not want to classify)

Actions are : Assign a class label or reject

Sometimes, when misclassification is costly, we can reject to classify it.

May be some other expert can look into it and can take appropriate action.

Reject Option



- Read Duda and Hart book and try to solve some problems related to this.

Naïve Bayes Classification

Introduction

- The Bayes Classifier requires probability structure of the problem to be known.
- Density estimation (using non-parametric or parametric methods) is one way to handle the problem.
- There are several problems ….

Problems with density estimation

- Large datasets are needed.
- Numeric valued features are required.

- In practice these two may not be satisfied.

How to overcome the problem

- One has to work with the given data set.
- So, Probability estimations needs to be done using the given data only.
- Often marginal probabilities can be better estimated than the joint probabilities.
- Also, marginal probabilities are easy to compute.

Play-tennis data

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

- P(<sunny,cool,high,false>|N) = 0
- But, P(sunny|N) = 3/5, P(cool|N) = 1/5,
  P(high|N) = 4/5, P(false|N) = 2/5.

- P(<sunny, cool, high, false>|N) = 0
- This may be because of the smaller dataset.
- If we increase the dataset size, this may become a positive number.

- This problem is often referred to as “the curse of dimensionality”.

Assumption

- Make the assumption that for a given class, features are independent of each other.

- In practice, this assumption holds very often.

- Then P(<sunny,cool,high,false>|N) = P(sunny|N) . P(cool|N) . P(high|N) . P(false|N) = 3/5 . 1/5 . 4/5 . 2/5 = 24/625.

Naïve Bayesian Classification

- Naïve assumption: for a given class, features are independent of each other

$$P(<x_1,\ldots,x_k>|C) = P(x_1|C)\cdot\ldots\cdot P(x_k|C)$$

- $P(x_i|C)$ is estimated as the relative freq of samples having value x_i as i-th attribute in class C
- It often makes the problem a feasible and easy one to solve.

Play-tennis example: estimating P(x|C)

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

P(p) = 9/14
P(n) = 5/14

outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) = 4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**

Play-tennis example: classifying X

- An unseen sample X = <rain, hot, high, false>

- P(X|p)·P(p) =

  P(rain|p)·P(hot|p)·P(high|p)·P(false|p)·P(p) = 3/9·2/9·3/9·6/9·9/14 = 0.010582

- P(X|n)·P(n) =

  P(rain|n)·P(hot|n)·P(high|n)·P(false|n)·P(n) = 2/5·2/5·4/5·2/5·5/14 = 0.018286

- Sample X is classified in class n (don't play)

With Continuous features

- In order to use the Naïve Bayes classifier, the features has to be discretized appropriately (otherwise what happens?)

With Continuous features

- In order to use the Naïve Bayes classifier, the features has to be discretized appropriately (otherwise what happens?)
- Height = 4.234 will not occur anywhere in that column; but 4.213, 4.285 may be occuring. If you discretize (eg., rounding) then frequency ratio's are meaningful.
- Clustering of feature values of a feature may be done to achieve a better discretization.

Maximum-likelihood Parameter Estimation

Parameter Estimation

- Bayes classifier is the best classifier.
- But, we should know about the prior probabilities $P(\omega_i)$ and class-conditional densities $p(X|\omega_i)$.
- That means the probabilistic structure of the problem should be known.
- In general, what is given to us is only a training set, not the probabilistic structure !
- Never-the-less we can assume some thing like: *the distribution is Normal* or so, based on the domain. And then, estimate its parameters based on the training set (eg: mean, covariance matrix can be estimated from the sample).

Density Estimation

- There are two broad ways in which the probability densities can be estimated from the training set.
- These are :
  1. *Parametric methods*
  2. *Non-parametric methods*

Parametric Methods

- We assume the form of the distribution (eg: Normal) and estimate its parameters (eg: mean and covariance matrix).

  Two broad parameter estimation methods are:

  1. maximum-likelihood estimation, and
  2. Bayesian estimation.

Non-parametric Methods

- We do not assume any thing about the form of the distribution, but we use the training examples directly to estimate the density at a given point.

  Two broad ways are:

  1. Parzen window based, and
  2. nearest neighbor based.

Maximum-likelihood method

- We study about maximum-likelihood parameter estimation.
- Here, we assume that the parameters are unknown but fixed.
- The other parameter estimation method, viz., Bayesian parameter estimation method assumes that *the parameters are unknown and random variables*.
- It is found that, both methods, frequently gives same results.
- Maximum-likelihood method is simpler than the Bayes method.

Maximum-likelihood method

- Training set is divided class-wise.
- We consider only one class's training set at a time.
- Let the parameters we are trying to estimate, for the class, be θ. For example $\theta = (\mu, \Sigma)^t$, if the distribution is assumed to be a Normal one.

Maximum-likelihood: General Principle

- Let $\mathcal{D}$ be the training set for the class.
- Let the patterns in $\mathcal{D}$ are independently and identically drawn(i.i.d).
- Suppose that $\mathcal{D}$ contains n samples, $X_1, \ldots, X_n$.
- Then,

$$p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(X_k|\theta).$$

Maximum-likelihood: General Principle

- $p(\mathcal{D}|\theta)$, when viewed as a function of θ, is called likelihood of θ with respect to the set of samples.
- The *maximum-likelihood estimate* of θ is, by definition, the value $\hat{\theta}$ that maximizes $p(\mathcal{D}|\theta)$.
- Intuitively, this estimate corresponds to the value of θ that in some sense best agrees with the training set.

To simplify analytically

- It is usually easier to work with the logarithm of the likelihood than with the likelihood itself.
- Because the logarithm is monotonically increasing, the $\hat{\theta}$ that maximizes the log-likelihood also maximizes the likelihood.
- If $p(\mathcal{D}|\theta)$ is well-behaved, differentiable function of θ, $\hat{\theta}$ can be found by the standard methods of differential calculus.

Maximum-likelihood ...

- Let the parameter vector $\theta = (\theta_1, \ldots, \theta_p)^t$. That is, there are p parameters to be estimated.
- Let ∇_θ be the gradient operator,

$$\nabla_\theta \equiv \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial}{\partial \theta_p} \end{bmatrix}$$

Maximum-likelihood ...

- We define $l(\theta)$ as the log-likelihood function

$$l(\theta) = \ln p(\mathcal{D}|\theta)$$

$$= \ln \left(\prod_{k=1}^{n} p(X_k|\theta)\right)$$

$$= \sum_{k=1}^{n} \ln p(X_k|\theta)$$

Maximum-likelihood ...

$$\nabla_\theta \, l = \begin{bmatrix} \frac{\partial l}{\partial \theta_1} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial l}{\partial \theta_p} \end{bmatrix}$$

- Thus, a set of necessary conditions for the maximum-likelihood estimate for θ can be obtained from a set of p equations

$$\nabla_\theta \, l = \mathbf{0}$$

Maximum-likelihood ...

- Let solution to $\nabla_\theta \, l = \mathbf{0}$ be $\hat{\theta}$.
- $\hat{\theta}$ could represent a true global maximum, a local maximum or minimum, or (rarely) an inflection point of $l(\theta)$.
- One must be careful regarding the above aspect. One remedy is, to find all solutions and findout from them which is the actual solution.
- In case of Normal distribution, we do not get these problems.

The Gaussian Case: Unknown μ

- Assume that, only μ is unknown, and we want to find the maximum-likelihood estimate for this.
- $\theta = [\mu]$.
-

$$\ln p(X_k|\mu) = -\frac{1}{2}\ln\left[(2\pi)^d|\Sigma|\right] - \frac{1}{2}(X_k-\mu)^t\Sigma^{-1}(X_k-\mu)$$

and

$$\nabla_\mu \ln p(X_k|\mu) = \Sigma^{-1}(X_k-\mu).$$

The Gaussian Case: Unknown μ

- The log-likelihood is,

$$l(\mu) = \sum_{k=1}^{n} \ln p(X_k|\mu)$$

Hence,

$$\nabla_\mu \, l \; = \; \sum_{k=1}^{n} \nabla_\mu \ln p(X_k|\mu) \; = \; \sum_{k=1}^{n} \Sigma^{-1}(X_k - \mu)$$

- When we equate the above to zero, we get

$$\hat{\mu} = \frac{1}{n} \sum_{k=1}^{n} X_k$$

The Gaussian Case: Unknown μ

- $$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} X_k$$

  is a very satisfying result.
- It says that the *sample mean* is the maximum-likelihood estimate for the mean.
- Sample mean is nothing but *centroid* of the set of patterns.

Unknown μ and σ^2

- Consider Univariate case.
- $\theta = (\theta_1, \theta_2)^t = (\mu, \sigma^2)^t$
  We know,

$$p(X_k|\theta) = \frac{1}{\sqrt{2\pi\theta_2}} \exp\left[-\frac{1}{2}\frac{(X_k - \theta_1)^2}{\theta_2}\right]$$

$$\ln p(X_k|\theta) = -\frac{1}{2}\ln 2\pi\theta_2 - \frac{1}{2\theta_2}(X_k - \theta_1)^2$$

Unknown μ and σ^2

$$\nabla_\theta \ln p(X_k|\theta) = \begin{bmatrix} \frac{\partial}{\partial\theta_1} \\ \frac{\partial}{\partial\theta_2} \end{bmatrix} \left[-\frac{1}{2} \ln 2\pi\theta_2 - \frac{1}{2\theta_2}(X_k - \theta_1)^2 \right]$$

$$= \begin{bmatrix} \frac{1}{\theta_2}(X_k - \theta_1) \\ -\frac{1}{2\theta_2} + \frac{(X_k - \theta_1)^2}{2\theta_2^2} \end{bmatrix}$$

Unknown μ and σ^2

- Maximum likelihood estimate for θ is obtained at

$$\sum_{k=1}^{n} \nabla_\theta \ln p(X_k|\theta) = \mathbf{0}$$

- That is,

$$\sum_{k=1}^{n} \frac{1}{\theta_2}(X_k - \theta_1) = 0 \qquad (1)$$

$$-\sum_{k=1}^{n} \frac{1}{\theta_2} + \sum_{k=1}^{n} \frac{(X_k - \theta_1)^2}{\theta_2^2} = 0 \qquad (2)$$

Unknown μ and σ^2

- We get,

$$\theta_1 = \hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} X_k$$

$$\theta_2 = \hat{\sigma^2} = \frac{1}{n}\sum_{k=1}^{n}(X_k - \hat{\mu})^2$$

Multivariate case: Unknown μ and Σ

- It can be found similar to univariate case.
- We get,

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} X_k$$

and

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(X_k - \hat{\mu})(X_k - \hat{\mu})^t$$

A problem

Consider univariate case. Let X have an exponential density

$$p(X|\theta) = \begin{cases} \theta e^{-\theta X} & X \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Given $\{X_1, \ldots, X_k\}$, the i.i.d drawn training set, find the maximum-likelihood estimate of θ.

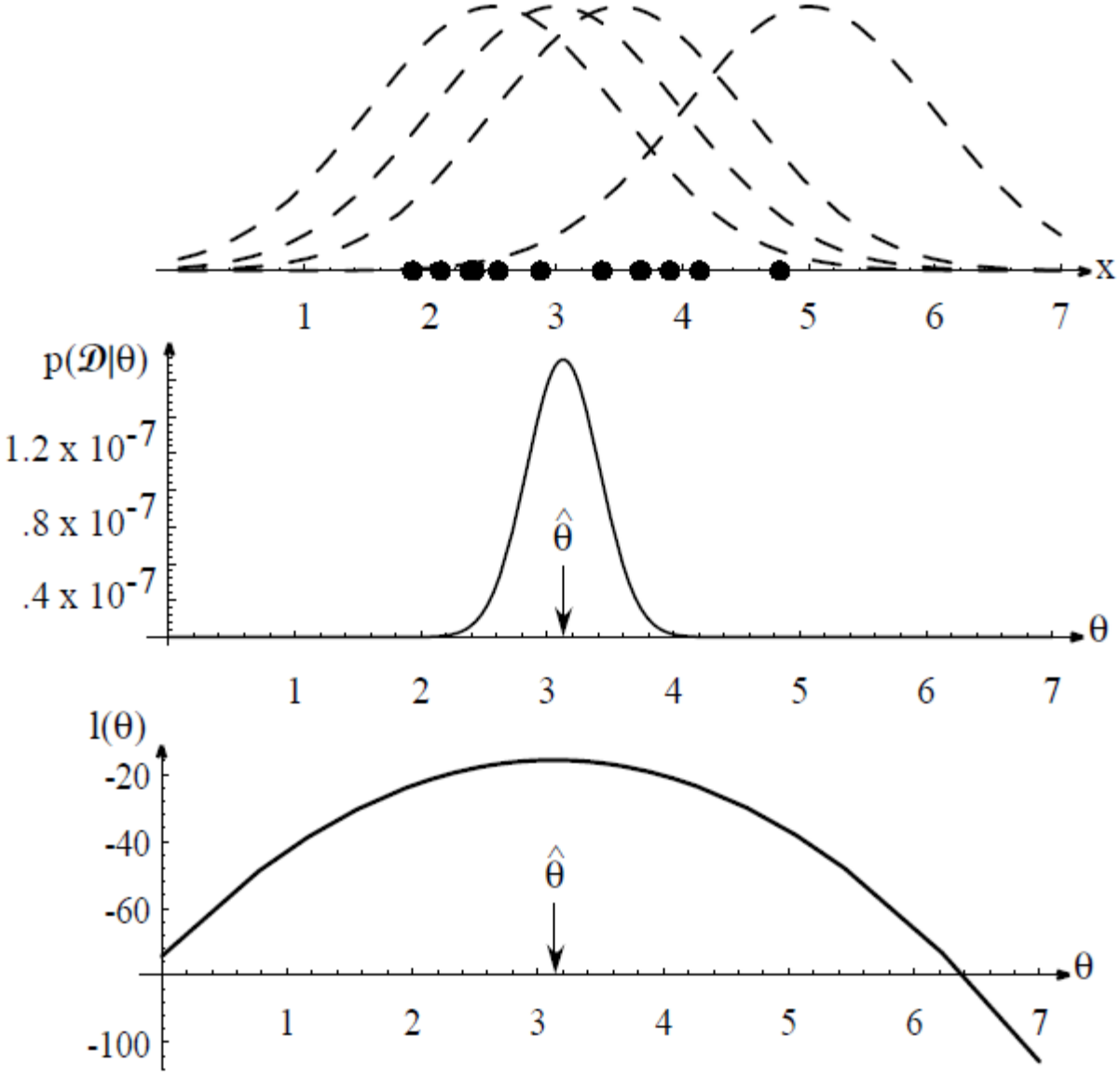
Gaussian Distribution

Univariate:

$$p(x) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma} \right)^2 \right],$$

Multivariate :

$$p(x) = \frac{1}{(2\pi)^{d/2}\left|\Sigma\right|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu) \right]$$


x
1
2
3
4
5
6
7
p(𝒟|θ)
1.2 x 10^-7
.8 x 10^-7
.4 x 10^-7
θ̂
θ
l(θ)
-20
-40
-60
-100

Maximum Likelihood Estimation of Gaussian Parameters

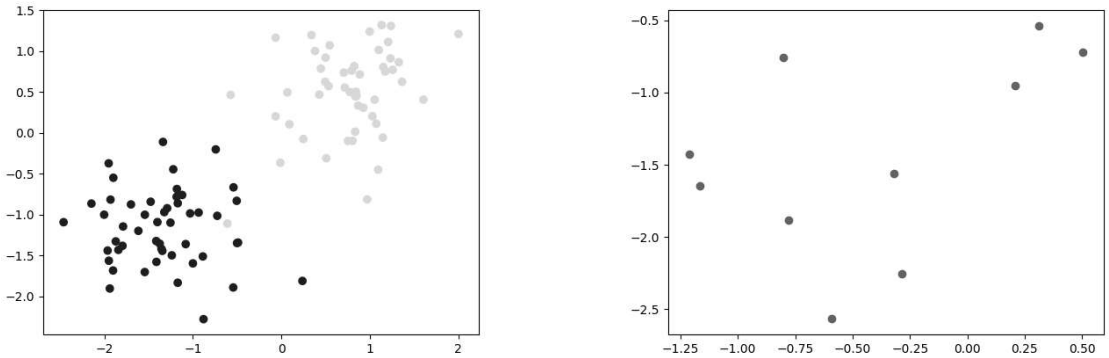
Aug 18, 2017

The Big Picture

Maximum Likelihood Estimation (MLE) is a tool we use in machine learning to acheive a *very* common goal. The goal is to create a statistical model, which is able to perform some task on *yet unseen data*.

The task might be classification, regression, or something else, so the nature of the task does not define MLE. The defining characteristic of MLE is that it uses *only existing data* to estimate parameters of the model. This is in contrast to approaches which exploit *prior knowledge* in addition to existing data.[1]

Today, we're talking about MLE for Gaussians, so this is going to be a classification task. That is, we have data with labels, and we want to take some new data, and classify it *using the labels from the old data*. In the below images, we see data with labels (left), and new, unlabeled data (right). We want to be able to categorize each point from the `new data` as belonging to either the `purple` group or the `yellow` group.



Labeled training data (**left**) and unlabeled new data (**right**)

Labeling dots as either `purple` or `yellow` sounds pretty boring, but the same idea applies to labeling emails as `spam` or `ham` or classifying audio clips as the vowel `[a]` or the vowel `[o]`.

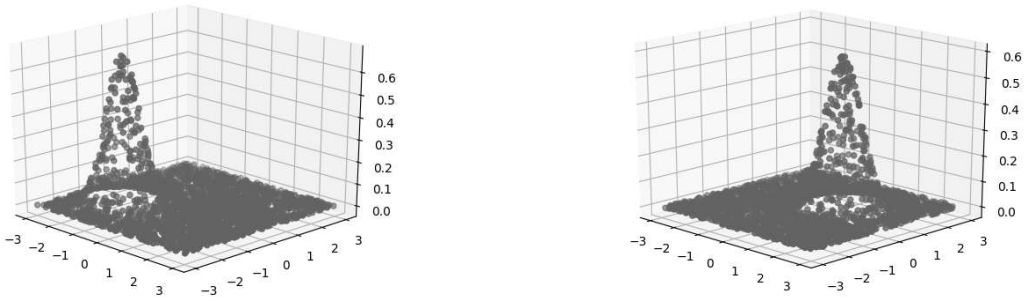
To make this post more tasty, let's pretend we're classifying `skittles` as `purple` or `yellow`.[2] We're classifying these skittles based on two dimensions `[x,y]`. Let's say the skittles have been rated by expert skittle-sommeliers on two traits: `x = aromatic lift` and `y = elegance`.

As you can see, `purple skittles` have bad ratings on both aromatic lift and elegance, whereas `yellow skittles` have been highly rated on both traits. Since these ratings are from expert skittle-sommeliers, they must be true.

To get our new, unlabeled data, we've given some new skittles to our expert sommeliers in a blind taste test. That is, the experts don't know what they ate, and neither do we. The only information available for each skittle is its rating on `aromatic lift` and `elegance`.

Now, we want to take ratings for each `mystery skittle` and figure out if it was a `purple skittle` or `yellow skittle`. To accomplish this task, we build a statistical model, learning its shape from the old ratings (i.e. the labeled data).

For the above data we can build two models (i.e. 2-D Gaussians), a `purple skittle model` and a `yellow skittle model`, and then see which model is more similar to a new rating on a `mystery skittle`. Another approach would be to build a single model (eg. a neural net) that distinguishes `purple skittles` from `yellow skittles`, and then see how it categorizes each `mystery skittle`.[3] Here, we're working with the former approach (build two models and see which one fits better).



Purple Skittle Model (**left**) Yellow Skittle Model (**right**)

We assume the data was in a sense *generated* by some process, and we're trying to model what that process was. This is called the generative approach. The model we're trying to learn is an approximation of the underlying process that created the data in the first place. So our data is just a sample from a process, and we want to learn the process.

At the end of the day, once we have our two models, we will use them to find which model was more likely to have *generated* the new data point. To take the leap from data $\rightarrow$ model, we need to not only estimate possible parameters of the model (eg. for Gaussians we need $[\mu, \Sigma]$), but we want the *best* model possible for our data. That's where MLE comes into play.

MLE as Parameter Estimation

MLE is one flavor of parameter estimation in machine learning, and in order to perform parameter estimation, we need:

1. some data $\mathbf{X}$
2. some hypothesized generating function of the data $f(\mathbf{X}, \theta)$
3. a set of parameters from that function θ
4. some evaluation of the goodness of our parameters (an objective function)

In MLE, the objective function (evaluation) we chose is the *likelihood* of the data given our model. This intuitively makes sense if you keep in mind that we don't get to change our data, and we have to make some assumption about the form of our model, but we *can* adjust the parameterization of our model. So, we are limited to adjusting θ, and we might as well choose the best θ for our data. To find the best θ then, we need to find the θ which maximizes our evaluation function (the likelihood). Therefore, in its general form the MLE is:

$$\theta_{MLE} = \underset{\theta}{\operatorname{argmax}}\, p(\mathbf{X}|\theta)$$

Likelihood for a Gaussian

We assume the data we're working with was generated by an underlying Gaussian process in the real world. As such, the likelihood function ($\mathcal{L}$) is the Gaussian itself.

$$\begin{aligned}\mathcal{L} = p(\mathbf{X}|\theta) &= \mathcal{N}(\mathbf{X}|\theta) \\ &= \mathcal{N}(\mathbf{X}|\mu, \Sigma)\end{aligned}$$

Therefore, for MLE of a Gaussian model, we will need to find good estimates of both parameters: μ and Σ:

$$\begin{aligned}\mu_{MLE} &= \underset{\mu}{\operatorname{argmax}}\, \mathcal{N}(\mathbf{X}|\mu, \Sigma) \\ \Sigma_{MLE} &= \underset{\Sigma}{\operatorname{argmax}}\, \mathcal{N}(\mathbf{X}|\mu, \Sigma)\end{aligned}$$

Solving these two above equations to find the best μ and Σ is a job for our good old friends from calculus… partial derivatives!

Before we can get to the point where we can find our best μ and Σ, we need to do some algebra, and to make that algebra easier, instead of just using the likelihood function as our evaluation function, we're going to use the log likelihood. This makes the math easier and it doesn't run any risks of giving us worse results. That's because the $\log()$ function is monotonically increasing, and therefore

$$\underset{\theta}{\operatorname{argmax}} \log(f(\theta)) == \underset{\theta}{\operatorname{argmax}}\, f(\theta)$$

So now, we know that we want to get the best parameters $\theta = [\mu, \Sigma]$ for a dataset $\mathbf{X}$ evaluating on a normal, Gaussian distribution.

$$\theta_{MLE} = \underset{\theta}{\operatorname{argmax}} \log(\mathcal{N}(\mathbf{X}|\theta))$$

Since in reality our dataset $\mathbf{X}$ is a set of labeled data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \ldots \mathbf{x}_n]$, to evaluate our parameters on the entire dataset, we need to sum up the log likelihood for each data point.

$$\log(\mathcal{N}(\mathbf{X}|\theta)) = \sum_{n=1}^{N} \log(\mathcal{N}(\mathbf{x}_n|\theta))$$

Remember how that θ is a general catch-all for any set of parameters? Let's be more explicit with our Gaussian parameters $[\mu, \Sigma]$:

$$\sum_{n=1}^{N} \log(\mathcal{N}(\mathbf{x}_n|\theta)) = \sum_{n=1}^{N} \log(\mathcal{N}(\mathbf{x}_n|\mu, \Sigma))$$

Here we're going to make a big simplfying assumption (and in reality a pretty common one). We're going to assume that our Gaussians have diagonal covariance matrices. So the full covariance matrix $\mathbf{\Sigma}$ gets

replaced by a diagonal variance vector σ^2:

$$\sum_{n=1}^{N} \log(\mathcal{N}(\mathbf{x}_n|\mu, \Sigma)) = \sum_{n=1}^{N} \log(\mathcal{N}(\mathbf{x}_n|\mu, \sigma^2))$$

Now, with this simplification, we can take a look at our fully specified log likelihood function that we'll be working with from here on out.

$$\sum_{n=1}^{N} \log(\mathcal{N}(\mathbf{x}_n|\mu, \sigma^2)) = \sum_{n=1}^{N} \log\Big(\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp^{-\frac{1}{2}\left(\frac{(x_n-\mu)^2}{\sigma^2}\right)}\Big)$$

Now we have the likelihood as we want it (Gaussian, logged, diagonal covariance matrix). Let's not forget what our main goal is! We want to find the best parameters for our model given our data, so we're going to find the $\underset{\mu}{\operatorname{argmax}}$ and $\underset{\sigma^2}{\operatorname{argmax}}$. Before we can get to that point, we need to do some simplifications to the log likelihood to make it easier to work with (that is, since we will soon be doing some partial derivatives, the log likelihood in its current form it will lead to some messy math). In the following, $\mathcal{LL}$ means *log likelihood*.

The next first steps take advantage of our choice to use the log likelihood instead of the plain likelihood. Our first step will be to use the log product rule:

$$\begin{aligned}
\mathcal{LL} &= \sum_{n=1}^{N} \log\Big(\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp^{-\frac{1}{2}\left(\frac{(x_n-\mu)^2}{\sigma^2}\right)}\Big) \\
&= \sum_{n=1}^{N} \Big(\log\Big(\frac{1}{\sqrt{2\pi\sigma^2}}\Big) + \log\Big(\exp^{-\frac{1}{2}\left(\frac{(x_n-\mu)^2}{\sigma^2}\right)}\Big)\Big)
\end{aligned}$$

Now we will use the log quotient rule:

$$\begin{aligned}
\mathcal{LL} &= \sum_{n=1}^{N} \Big(\log\Big(\frac{1}{\sqrt{2\pi\sigma^2}}\Big) + \log\Big(\exp^{-\frac{1}{2}\left(\frac{(x_n-\mu)^2}{\sigma^2}\right)}\Big)\Big) \\
&= \sum_{n=1}^{N} \Big(\log(1) - \log\big(\sqrt{2\pi\sigma^2}\big) + \log\Big(\exp^{-\frac{1}{2}\left(\frac{(x_n-\mu)^2}{\sigma^2}\right)}\Big)\Big)
\end{aligned}$$

Now, we'll use the log power rule:

$$\begin{aligned}
\mathcal{LL} &= \sum_{n=1}^{N} \Big(\log(1) - \log\big(\sqrt{2\pi\sigma^2}\big) + \log\Big(\exp^{-\frac{1}{2}\left(\frac{(x_n-\mu)^2}{\sigma^2}\right)}\Big)\Big) \\
&= \sum_{n=1}^{N} \Big(\log(1) - \log\big(\sqrt{2\pi\sigma^2}\big) + \Big(-\frac{1}{2}\big(\frac{(x_n-\mu)^2}{\sigma^2}\big) \cdot \log(e)\Big)\Big)
\end{aligned}$$

We're now going to be explicit that the $\log()$ function we used was base e. This allows us to simplify $\log_e(e) = 1$ as well as $\log(1) = 0$ (regardless of base).

$$\begin{aligned}
\mathcal{LL} &= \sum_{n=1}^{N} \Big(\log(1) - \log\big(\sqrt{2\pi\sigma^2}\big) + \Big(-\frac{1}{2}\big(\frac{(x_n-\mu)^2}{\sigma^2}\big) \cdot \log(e)\Big)\Big) \\
&= \sum_{n=1}^{N} \Big(-\log\big(\sqrt{2\pi\sigma^2}\big) + \Big(-\frac{1}{2}\big(\frac{(x_n-\mu)^2}{\sigma^2}\big)\Big)\Big)
\end{aligned}$$

We can apply the power rule one more time (remember that $\sqrt{x} = x^{1/2}$).

$$\begin{aligned}\mathcal{LL} &= \sum_{n=1}^{N}\Big(-\log\big(\sqrt{2\pi\sigma^2}\big)+\Big(-\frac{1}{2}\big(\frac{(x_n-\mu)^2}{\sigma^2}\big)\Big)\Big)\\ &= \sum_{n=1}^{N}\Big(-\frac{1}{2}\cdot\log(2\pi\sigma^2)-\frac{1}{2}\Big(\frac{(x_n-\mu)^2}{\sigma^2}\Big)\Big)\end{aligned}$$

Now for some basic algebra simplification:

$$\begin{aligned}\mathcal{LL} &= \sum_{n=1}^{N}\Big(-\frac{1}{2}\log(2\pi\sigma^2)-\frac{1}{2}\Big(\frac{(x_n-\mu)^2}{\sigma^2}\Big)\Big)\\ &= -\frac{N}{2}\log(2\pi\sigma^2)+\sum_{n=1}^{N}-\frac{1}{2}\Big(\frac{(x_n-\mu)^2}{\sigma^2}\Big)\\ &= -\frac{N}{2}\log(2\pi\sigma^2)-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\end{aligned}$$

Now we have our log likelihood function ($\mathcal{LL}$) in a nice, easy to work with form. Now we need to take the ($\mathcal{LL}$) and *estimate* it's *maximum* for our parameters. We've got the $\mathcal{LL}$ ready for $\underset{\theta}{\operatorname{argmax}}(\mathcal{LL})$, now we need to do the $\underset{\theta}{\operatorname{argmax}}$ part. This is where we get a little help from our friends, partial derivatives. We need *partial* derivatives because our θ is really two variables $[\mu, \sigma^2]$, and we need the best value for each.

So, now we're going to solve the problem for each variable one-by-one:

$$\underset{\mu}{\operatorname{argmax}}\,\mathcal{LL}(X|\mu,\sigma^2)$$
$$\underset{\sigma^2}{\operatorname{argmax}}\,\mathcal{LL}(X|\mu,\sigma^2)$$

To get the argmax for each parameter we have to do two things. First, we must:

1. derive the partial derivative of the function with respect to that parameter, and then
2. set that partial derivative to zero, and solve for our parameter

By doing step (1), we get an equation for the change of the function, and we know that if the function isn't changing at a certain point, that point is a maximum or a minimum (or a saddle point).

In our case, our log-likelihood function is concave, so we know that at $\frac{\partial\mathcal{LL}}{\partial\mu}=0$ we get a maximum.

MLE of μ

First we'll work to solve for the mean of our Gaussian, μ. Remember we've got our likelihood function in a simple form:

$$\mathcal{LL} = -\frac{N}{2}\log(2\pi\sigma^2)-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2$$

and now we want to get the best μ for that function:

$$\underset{\mu}{\operatorname{argmax}}\,\mathcal{LL}(X|\mu,\sigma^2) := \frac{\partial\mathcal{LL}}{\partial\mu}=0$$

So, to get to the point where we can set the partial derivative to zero and solve, we need to first find the partial derivative with respect to μ:

$$\frac{\partial \mathcal{LL}}{\partial \mu} = \frac{\partial}{\partial \mu}\Big(-\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big)$$

Now let's start simplifying! First we can right off the bat get rid of the first term since it doesn't contain μ, and therefore is practically speaking a constant:

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \mu} &= \frac{\partial}{\partial \mu}\Big(-\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= \frac{\partial}{\partial \mu}\Big(-\frac{N}{2}\log(2\pi\sigma^2)\Big) + \frac{\partial}{\partial \mu}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= 0 + \frac{\partial}{\partial \mu}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= \frac{\partial}{\partial \mu}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big)\end{aligned}$$

Next, remember that the summation expression is just a convenient way to write a longer expression:

$$\sum_{n=1}^{N} f(x_n) = f(x_1) + f(x_2) + \ldots + f(x_N)$$

Also, We know from the summation rule that :

$$\frac{\partial}{\partial x}\big(f(x)+g(x)\big) = \frac{\partial}{\partial x}f(x) + \frac{\partial}{\partial x}g(x)$$

Therefore, when we take the derivative of a sum, we can reformulate it as a sum of derivatives:

$$\frac{\partial}{\partial x}\sum_{n=1}^{N} f(x_n) = \sum_{n=1}^{N}\frac{\partial}{\partial x}f(x_n)$$

Now, getting back to the problem at hand, we can move the derivative operator inside the summation term:

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \mu} &= \frac{\partial}{\partial \mu}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= \frac{\partial}{\partial \mu}\Big(\sum_{n=1}^{N} -\frac{1}{2\sigma^2}(x_n-\mu)^2\Big) \\ &= \sum_{n=1}^{N}\frac{\partial}{\partial \mu}\Big(-\frac{1}{2\sigma^2}(x_n-\mu)^2\Big)\end{aligned}$$

Now we can use the product rule:

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \mu} &= \sum_{n=1}^{N}\frac{\partial}{\partial \mu}\Big(-\frac{1}{2\sigma^2}\cdot(x_n-\mu)^2\Big) \\ &= \sum_{n=1}^{N}\Big(\frac{\partial}{\partial \mu}\big(-\frac{1}{2\sigma^2}\big)\cdot(x_n-\mu)^2 + \big(-\frac{1}{2\sigma^2}\big)\cdot\frac{\partial}{\partial \mu}(x_n-\mu)^2\Big)\end{aligned}$$

Now some terms will nicely drop out:

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \mu} &= \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \mu} \big(-\frac{1}{2\sigma^2} \big) \cdot (x_n - \mu)^2 + \big(-\frac{1}{2\sigma^2} \big) \cdot \frac{\partial}{\partial \mu} (x_n - \mu)^2 \Big) \\ &= \sum_{n=1}^{N} \Big(0 + \big(-\frac{1}{2\sigma^2} \big) \cdot \frac{\partial}{\partial \mu} (x_n - \mu)^2 \Big) \\ &= -\frac{1}{2\sigma^2} \sum_{n=1}^{N} \frac{\partial}{\partial \mu} (x_n - \mu)^2\end{aligned}$$

At this point we can use the chain rule, $\frac{\partial}{\partial x}\big(f(g(x))\big) = \frac{\partial}{\partial x} f(g(x)) \cdot \frac{\partial}{\partial x} g(x)$, with $g(x) = (x - \mu)$ and $f(x) = x^2$.

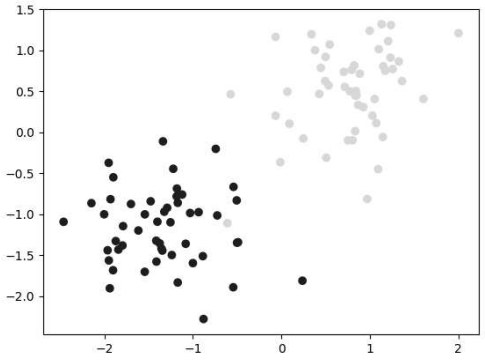
$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \mu} &= -\frac{1}{2\sigma^2} \sum_{n=1}^{N} \frac{\partial}{\partial \mu} (x_n - \mu)^2 \\ &= -\frac{1}{2\sigma^2} \sum_{n=1}^{N} 2(x_n - \mu) \cdot -1 \\ &= \frac{1}{\sigma^2} \sum_{n=1}^{N} (x_n - \mu)\end{aligned}$$

Yay! We've done as much simplifying as we can at this point, and gotten rid of all of our $\frac{\partial \mathcal{LL}}{\partial \mu}$ terms!

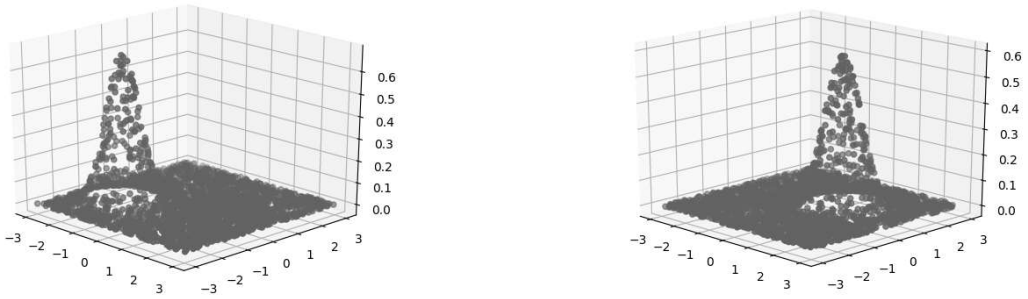
Now what we have is the simplest form of the partial derivative of our likelihood function with respect to μ. Now we want to use this equation to find the best μ, so we set it equal to zero, and solve for μ.

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \mu} &= \frac{1}{\sigma^2} \sum_{n=1}^{N} (x_n - \mu) \\ 0 &= \frac{1}{\sigma^2} \sum_{n=1}^{N} (x_n - \mu) \\ 0 &= \sum_{n=1}^{N} (x_n - \mu) \\ 0 &= \sum_{n=1}^{N} x_n - \sum_{n=1}^{N} \mu \\ 0 &= \sum_{n=1}^{N} x_n - N \cdot \mu \\ N \cdot \mu &= \sum_{n=1}^{N} x_n \\ \mu &= \frac{1}{N} \sum_{n=1}^{N} x_n\end{aligned}$$

Huzzah! We've reached the promised land! We now have a formula we can use to estimate one model parameter (μ) from our data ($\mathbf{X}$). Let's take a second to think about what this formula means. Remember that we started with a bunch of data points:



We want to take that data and make some models which we think represent that data well. In our case we are learning two Gaussian models, one for the `yellow skittle` data and one for the `purple skittle` data:



Purple Skittle Model (**left**) Yellow Skittle Model (**right**)

To make our Gaussians fit the data as well as we can, we can do two things: (1) move the center of the curve or (2) adjust the width of the peak. Right now, with μ we're only talking about the placement of the center of the curve, we're not talking at all about its width.

Where is the best place to put a bell curve to cover all our data? Well, how about the center of our data! Dead-center, bull's eye, whatever you call it, we're putting our Gaussian right in the middle of it all (the mean). We're taking all our points, summing them up, and dividing by the number of data points. This is the *average*, and it is (for the likelihood function) the best place to put the mean of our model.

Getting back to the skittles, the center of the curve $[\mu_x, \mu_y]$ for our `yellow skittle model` comes directly from our sommeliers' ratings (i.e. $[\mu_x]$ = average rating for `aromatic lift` and $[\mu_y]$ = average rating for `elegance`). Our `purple skittle model` was centered in the exact same way.

Sure enough, if you take a look at the data, you'll see that the `yellow skittle` data is grouped around the point `[-1,-1]` and that the `purple skittle` data points are all clustered around `[1,1]`. Now take a look at the models we've made. You'll see that the center of the peak for the `purple skittle model` is somewhere near `[-1,-1]` and that the peak of the `yellow skittle model` is around `[1,1]`.

MLE of σ^2

Now let's tackle the second parameter of our Gaussian model, the variance σ^2!

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= \frac{\partial}{\partial \sigma^2}\Big(-\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= \frac{\partial}{\partial \sigma^2}\Big(-\frac{N}{2}\log(2\pi\sigma^2)\Big) + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big)\end{aligned}$$

Let's start with the product rule for the lefthand term:

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= \frac{\partial}{\partial \sigma^2}\Big(-\frac{N}{2}\log(2\pi\sigma^2)\Big) + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= \frac{\partial}{\partial \sigma^2}\big(-\frac{N}{2}\big)\cdot\log(2\pi\sigma^2) + \big(-\frac{N}{2}\big)\cdot\frac{\partial}{\partial \sigma^2}\big(\log(2\pi\sigma^2)\big) + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= 0 + \big(-\frac{N}{2}\big)\cdot\frac{\partial}{\partial \sigma^2}\big(\log(2\pi\sigma^2)\big) + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= -\frac{N}{2}\cdot\frac{\partial}{\partial \sigma^2}\big(\log(2\pi\sigma^2)\big) + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big)\end{aligned}$$

Now we can use the chain rule for our term with the $\log$ operator,
$\frac{\partial}{\partial x}\big(f(g(x))\big) = \frac{\partial}{\partial x}f(g(x))\cdot\frac{\partial}{\partial x}g(x)$, with $g(x) = 2\pi x$ and $f(x) = log(x)$.

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= -\frac{N}{2}\cdot\frac{\partial}{\partial \sigma^2}\big(\log(2\pi\sigma^2)\big) + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= -\frac{N}{2}\cdot\frac{1}{2\pi\sigma^2}\cdot 2\pi + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= -\frac{N}{2}\cdot\frac{1}{\sigma^2} + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= -\frac{N}{2\sigma^2} + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big)\end{aligned}$$

Now, using the same logic as above with μ, we can move the derivative operator inside the summation operator:

$$\begin{aligned}\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= -\frac{N}{2\sigma^2} + \frac{\partial}{\partial \sigma^2}\Big(-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\Big) \\ &= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N}\Big(\frac{\partial}{\partial \sigma^2}\big(-\frac{1}{2\sigma^2}(x_n-\mu)^2\big)\Big)\end{aligned}$$

And again, the product rule:

$$
\begin{aligned}
\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2\sigma^2} (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2\sigma^2} \big) \cdot (x_n - \mu)^2 \big) + \big(-\frac{1}{2\sigma^2} \big) \cdot \frac{\partial}{\partial \sigma^2} (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2\sigma^2} \big) \cdot (x_n - \mu)^2 \big) + 0 \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2\sigma^2} \big) \cdot (x_n - \mu)^2 \big) \Big)
\end{aligned}
$$

Now let's be careful with our exponents, since we're taking the derivative of the function with respect to a squared variable σ^2:

$$
\begin{aligned}
\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2\sigma^2} \big) \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2} \cdot \sigma^{-2} \big) \cdot (x_n - \mu)^2 \big) \Big)
\end{aligned}
$$

Now it's obvious that we need the product rule:

$$
\begin{aligned}
\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2} \cdot \sigma^{-2} \big) \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{\partial}{\partial \sigma^2} \big(-\frac{1}{2} \big) \cdot \sigma^{-2} + \big(-\frac{1}{2} \cdot \frac{\partial}{\partial \sigma^2} \sigma^{-2} \big) \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(0 + \big(-\frac{1}{2} \cdot \frac{\partial}{\partial \sigma^2} \sigma^{-2} \big) \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(-\frac{1}{2} \cdot \frac{\partial}{\partial \sigma^2} \sigma^{-2} \cdot (x_n - \mu)^2 \big) \Big)
\end{aligned}
$$

Now, we're going to use the chain rule again, by first treating $\sigma^{-2} = (\sigma^2)^{-1}$, then we can see $f(g(x))$ with $f(x) = x^{-1}$ and then $g(x) = x^2$:

$$
\begin{aligned}
\frac{\partial \mathcal{LL}}{\partial \sigma^2} &= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(-\frac{1}{2} \cdot \frac{\partial}{\partial \sigma^2} \sigma^{-2} \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(-\frac{1}{2} \cdot \frac{\partial}{\partial \sigma^2} \big((\sigma^2)^{-1} \big) \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(-\frac{1}{2} \cdot -1 \cdot (\sigma^2)^{-2} \cdot 1 \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{1}{2} \cdot (\sigma^2)^{-2} \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \sum_{n=1}^{N} \Big(\frac{1}{2\sigma^4} \cdot (x_n - \mu)^2 \big) \Big) \\
&= -\frac{N}{2\sigma^2} + \frac{1}{2\sigma^4} \sum_{n=1}^{N} (x_n - \mu)^2 \\
&= \frac{1}{2\sigma^2} \Big(-N + \frac{1}{\sigma^2} \sum_{n=1}^{N} (x_n - \mu)^2 \Big)
\end{aligned}
$$

Huzzah! We've gotten our partial derivative for $\mathcal{LL}$ with respect to σ^2 as simplified as we can. Now let's find the best σ^2 for our data by setting the equation equal to zero and solving for σ^2.

$$0 = \frac{1}{2\sigma^2}\Big(-N + \frac{1}{\sigma^2}\sum_{n=1}^{N}(x_n - \mu)^2\Big)$$

$$0 = -N + \frac{1}{\sigma^2}\sum_{n=1}^{N}(x_n - \mu)^2$$

$$N\sigma^2 = \sum_{n=1}^{N}(x_n - \mu)^2$$

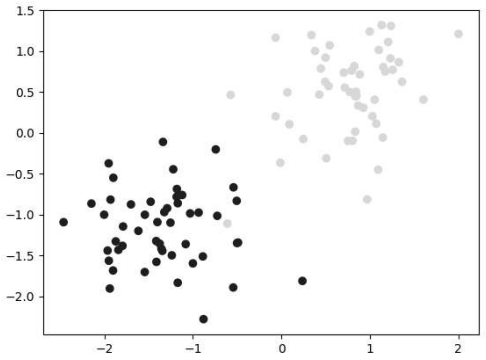
$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

And there we have it! All that work has boiled down to a simple equation do get the best σ^2 for our Gaussian given our data. Just like with μ if we take a second to look at the equation, we find it has a very intuitive interpretation.

We're iterating over each data point (x_n) and finding its particular deviation from the mean of all the data points μ. We sum up all the deviations, and then take the average!

Just as when we were finding the best μ for our Gaussian by setting it to the average (μ) of the data, we're now setting our standard deviation to be, well, the *standard* deviation of the data! Think of *standard* as being a synonym to *average*, and it becomes pretty clear.

Thinking back to the `skittles`, what we've done here is taken each skittle, one-by-one, and figured out how far it deviates from the mean on a certain rating. For example, we know that on average, our expert sommeliers rated `purple` skittles to have a `[-1]` score for `elegance`. However, not every `purple skittle` got a score of `[-1]`. Each skittle deviated from that average, and if we add up how much each skittle deviated (after squaring), we with the average deviation. Take a look again at our skittle data:



Specifically, look at the `purple skittles`. You see the purple skittle that got a rating of about `[0,-2]`? That skittle had pretty horrible `elegance` and better-than-average `aromatic lift`. This skittle obviously *deviated* from the normal rating. We take this skittle along with every other, calculate that deviation, and get our σ^2 for our model.

Conclusion

We did a lot of algebra, some calculus, and used some tricks with $\log_e$ to get to this point.

Along the way it's easy to get lost in the weeds, but if we keep in mind that all these equations have some interpretation, we can catch the big picture. In our case, the big picture is very clear:

> *When using Maximum Likelihood Estimation to estimate parameters of a Gaussian, set the mean of the Gaussian to be the mean of the data, and set the standard deviation of the Gaussian to be the standard deviation of the data.*

$$\mu_{MLE} = \frac{1}{N}\sum_{n=1}^{N} x_n$$
$$\sigma^2_{MLE} = \frac{1}{N}\sum_{n=1}^{N} (x_n - \mu)^2$$

I hope this was helpful or interesting! If you find errors or have comments, let me know!

1. For another approach to parameter estimation using not only information from the data, but a prior bias, see Maximum A Posteriori estimation. ↩
2. I have no sponsorship from Skittles, the Wrigley Company, or Mars, Inc. All the views expressed in this post are my own. However, if they would like to throw some cash my way, I would not be upset. ↩
3. This is an example of a discriminative model, as opposed to a generative model. The classification approach described here is the latter approach. ↩

Math 541: Statistical Theory II

Maximum Likelihood Estimation

Lecturer: Songfeng Zheng

1 Maximum Likelihood Estimation

Maximum likelihood is a relatively simple method of constructing an estimator for an unknown parameter θ. It was introduced by R. A. Fisher, a great English mathematical statistician, in 1912. Maximum likelihood estimation (MLE) can be applied in most problems, it has a strong intuitive appeal, and often yields a reasonable estimator of θ. Furthermore, if the sample is large, the method will yield an excellent estimator of θ. For these reasons, the method of maximum likelihood is probably the most widely used method of estimation in statistics.

Suppose that the random variables $X_1, \cdots, X_n$ form a random sample from a distribution $f(x|\theta)$; if X is continuous random variable, $f(x|\theta)$ is pdf, if X is discrete random variable, $f(x|\theta)$ is point mass function. We use the given symbol — to represent that the distribution also depends on a parameter θ, where θ could be a real-valued unknown parameter or a vector of parameters. For every observed random sample $x_1, \cdots, x_n$, we define

$$f(x_1, \cdots, x_n|\theta) = f(x_1|\theta) \cdots f(x_n|\theta) \tag{1}$$

If $f(x|\theta)$ is pdf, $f(x_1, \cdots, x_n|\theta)$ is the joint density function; if $f(x|\theta)$ is pmf, $f(x_1, \cdots, x_n|\theta)$ is the joint probability. Now we call $f(x_1, \cdots, x_n|\theta)$ as the *likelihood function.* As we can see, the likelihood function depends on the unknown parameter θ, and it is always denoted as $L(\theta)$.

Suppose, for the moment, that the observed random sample $x_1, \cdots, x_n$ came from a discrete distribution. If an estimate of θ must be selected, we would certainly not consider any value of θ for which it would have been impossible to obtain the data $x_1, \cdots, x_n$ that was actually observed. Furthermore, suppose that the probability $f(x_1, \cdots, x_n|\theta)$ of obtaining the actual observed data $x_1, \cdots, x_n$ is very high when θ has a particular value, say $\theta = \theta_0$, and is very small for every other value of θ. Then we would naturally estimate the value of θ to be θ_0. When the sample comes from a continuous distribution, it would again be natural to try to find a value of θ for which the probability density $f(x_1, \cdots, x_n|\theta)$ is large, and to use this value as an estimate of θ. For any given observed data $x_1, \cdots, x_n$, we are led by this reasoning to consider a value of θ for which the likelihood function $L(\theta)$ is a maximum and to use this value as an estimate of θ.

The meaning of maximum likelihood is as follows. We choose the parameter that makes the likelihood of having the obtained data at hand maximum. With discrete distributions, the likelihood is the same as the probability. We choose the parameter for the density that maximizes the probability of the data coming from it.

Theoretically, if we had no actual data, maximizing the likelihood function will give us a function of n random variables $X_1, \cdots, X_n$, which we shall call "maximum likelihood estimate" $\hat{\theta}$. When there are actual data, the estimate takes a particular numerical value, which will be the maximum likelihood estimator.

MLE requires us to maximum the likelihood function $L(\theta)$ with respect to the unknown parameter θ. From Eqn. 1, $L(\theta)$ is defined as a product of n terms, which is not easy to be maximized. Maximizing $L(\theta)$ is equivalent to maximizing $\log L(\theta)$ because log is a monotonic increasing function. We define $\log L(\theta)$ as *log likelihood function*, we denote it as $l(\theta)$, i.e.

$$l(\theta) = \log L(\theta) = \log \prod_{i=1}^{n} f(X_i|\theta) = \sum_{i=1}^{n} \log f(X_i|\theta)$$

Maximizing $l(\theta)$ with respect to θ will give us the MLE estimation.

2 Examples

Example 1: Suppose that X is a discrete random variable with the following probability mass function: where $0 \leq \theta \leq 1$ is a parameter. The following 10 independent observations

X	0	1	2	3
$P(X)$	$2\theta/3$	$\theta/3$	$2(1-\theta)/3$	$(1-\theta)/3$

were taken from such a distribution: (3,0,2,1,3,2,1,0,2,1). What is the maximum likelihood estimate of θ.

Solution: Since the sample is (3,0,2,1,3,2,1,0,2,1), the likelihood is

$$\begin{aligned} L(\theta) = & \quad P(X=3)P(X=0)P(X=2)P(X=1)P(X=3) \\ & \times \quad P(X=2)P(X=1)P(X=0)P(X=2)P(X=1) \end{aligned} \tag{2}$$

Substituting from the probability distribution given above, we have

$$L(\theta) = \prod_{i=1}^{n} P(X_i|\theta) = \left(\frac{2\theta}{3}\right)^2 \left(\frac{\theta}{3}\right)^3 \left(\frac{2(1-\theta)}{3}\right)^3 \left(\frac{1-\theta}{3}\right)^2$$

Clearly, the likelihood function $L(\theta)$ is not easy to maximize.

Let us look at the log likelihood function

$$\begin{aligned} l(\theta) &= \log L(\theta) = \sum_{i=1}^{n} \log P(X_i|\theta) \\ &= 2\left(\log\frac{2}{3} + \log\theta\right) + 3\left(\log\frac{1}{3} + \log\theta\right) + 3\left(\log\frac{2}{3} + \log(1-\theta)\right) + 2\left(\log\frac{1}{3} + \log(1-\theta)\right) \\ &= C + 5\log\theta + 5\log(1-\theta) \end{aligned}$$

where C is a constant which does not depend on θ. It can be seen that the log likelihood function is easier to maximize compared to the likelihood function.

Let the derivative of $l(\theta)$ with respect to θ be zero:

$$\frac{dl(\theta)}{d\theta} = \frac{5}{\theta} - \frac{5}{1-\theta} = 0$$

and the solution gives us the MLE, which is $\hat{\theta} = 0.5$. We remember that the method of moment estimation is $\hat{\theta} = 5/12$, which is different from MLE.

Example 2: Suppose $X_1, X_2, \cdots, X_n$ are i.i.d. random variables with density function $f(x|\sigma) = \frac{1}{2\sigma}\exp\left(-\frac{|x|}{\sigma}\right)$, please find the maximum likelihood estimate of σ.

Solution: The log-likelihood function is

$$l(\sigma) = \sum_{i=1}^{n}\left[-\log 2 - \log\sigma - \frac{|X_i|}{\sigma}\right]$$

Let the derivative with respect to θ be zero:

$$l'(\sigma) = \sum_{i=1}^{n}\left[-\frac{1}{\sigma} + \frac{|X_i|}{\sigma^2}\right] = -\frac{n}{\sigma} + \frac{\sum_{i=1}^{n}|X_i|}{\sigma^2} = 0$$

and this gives us the MLE for σ as

$$\hat{\sigma} = \frac{\sum_{i=1}^{n}|X_i|}{n}$$

Again this is different from the method of moment estimation which is

$$\hat{\sigma} = \sqrt{\frac{\sum_{i=1}^{n}X_i^2}{2n}}$$

Example 3: Use the method of moment to estimate the parameters μ and σ for the normal density

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\},$$

based on a random sample $X_1, \cdots, X_n$.

Solution: In this example, we have two unknown parameters, μ and σ, therefore the parameter $\theta = (\mu, \sigma)$ is a vector. We first write out the log likelihood function as

$$l(\mu, \sigma) = \sum_{i=1}^{n} \left[-\log \sigma - \frac{1}{2}\log 2\pi - \frac{1}{2\sigma^2}(X_i - \mu)^2 \right] = -n \log \sigma - \frac{n}{2}\log 2\pi - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(X_i - \mu)^2$$

Setting the partial derivative to be 0, we have

$$\frac{\partial l(\mu, \sigma)}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i - \mu) = 0$$

$$\frac{\partial l(\mu, \sigma)}{\partial \sigma} = -\frac{n}{\sigma} + \sigma^{-3}\sum_{i=1}^{n}(X_i - \mu)^2 = 0$$

Solving these equations will give us the MLE for μ and σ:

$$\hat{\mu} = \overline{X} \quad \text{and} \quad \hat{\sigma} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^2}$$

This time the MLE is the same as the result of method of moment.

From these examples, we can see that the maximum likelihood result may or may not be the same as the result of method of moment.

Example 4: The Pareto distribution has been used in economics as a model for a density function with a slowly decaying tail:

$$f(x|x_0, \theta) = \theta x_0^{\theta} x^{-\theta-1}, \quad x \geq x_0, \quad \theta > 1$$

Assume that $x_0 > 0$ is given and that $X_1, X_2, \cdots, X_n$ is an i.i.d. sample. Find the MLE of θ.

Solution: The log-likelihood function is

$$\begin{aligned} l(\theta) &= \sum_{i=1}^{n} \log f(X_i|\theta) = \sum_{i=1}^{n} (\log \theta + \theta \log x_0 - (\theta + 1)\log X_i) \\ &= n \log \theta + n\theta \log x_0 - (\theta + 1)\sum_{i=1}^{n} \log X_i \end{aligned}$$

Let the derivative with respect to θ be zero:

$$\frac{dl(\theta)}{d\theta} = \frac{n}{\theta} + n \log x_0 - \sum_{i=1}^{n} \log X_i = 0$$

Solving the equation yields the MLE of θ:

$$\hat{\theta}_{MLE} = \frac{1}{\overline{\log X} - \log x_0}$$

Example 5: Suppose that $X_1, \cdots, X_n$ form a random sample from a uniform distribution on the interval $(0, \theta)$, where of the parameter $\theta > 0$ but is unknown. Please find MLE of θ.

Solution: The pdf of each observation has the following form:

$$f(x|\theta) = \begin{cases} 1/\theta, & \text{for} \quad 0 \leq x \leq \theta \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Therefore, the likelihood function has the form

$$L(\theta) = \begin{cases} \frac{1}{\theta^n}, & \text{for} \quad 0 \leq x_i \leq \theta \quad (i = 1, \cdots, n) \\ 0, & \text{otherwise} \end{cases}$$

It can be seen that the MLE of θ must be a value of θ for which $\theta \geq x_i$ for $i = 1, \cdots, n$ and which maximizes $1/\theta^n$ among all such values. Since $1/\theta^n$ is a decreasing function of θ, the estimate will be the smallest possible value of θ such that $\theta \geq x_i$ for $i = 1, \cdots, n$. This value is $\theta = \max(x_1, \cdots, x_n)$, it follows that the MLE of θ is $\hat{\theta} = \max(X_1, \cdots, X_n)$.

It should be remarked that in this example, the MLE $\hat{\theta}$ does not seem to be a suitable estimator of θ. We know that $\max(X_1, \cdots, X_n) < \theta$ with probability 1, and therefore $\hat{\theta}$ surely underestimates the value of θ.

Example 6: Suppose again that $X_1, \cdots, X_n$ form a random sample from a uniform distribution on the interval $(0, \theta)$, where of the parameter $\theta > 0$ but is unknown. However, suppose now we write the density function as

$$f(x|\theta) = \begin{cases} 1/\theta, & \text{for} \quad 0 < x < \theta \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

We will prove that in this case, the MLE for θ does not exist.

Proof: The only difference between Eqn. 3 and Eqn. 4 is that the value of the pdf at the two endpints 0 and θ has been changed by replacing the weak inequalities in Eqn. 3 with strict inequalities in Eqn. 4. Either equation could be used as the pdf of the uniform distribution.

However, if Eqn. 4 is used as the pdf, then an MLE of θ will be a value of θ for which $\theta > x_i$ for $i = 1, \cdots, n$ and which maximizes $1/\theta^n$ among all such values. It should be noted that the possible values of θ no longer include the value $\theta = \max(x_1, \cdots, x_n)$, since θ must be *strictly* greater than each observed value x_i for $i = 1, \cdots, n$. Since θ can be chosen arbitrarily close to the value $\max(x_1, \cdots, x_n)$ but cannot be chosen equal to this value, it follows that the MLE of θ does not exist in this case.

Example 5 and 6 illustrate one shortcoming of the concept of an MLE. We know that it is irrelevant whether the pdf of the uniform distribution is chosen to be equal to $1/\theta$ over the open interval $0 < x < \theta$ or over the closed interval $0 \leq x \leq \theta$. Now, however, we see that the existence of an MLE depends on this typically irrelevant and unimportant choice. This difficulty is easily avoided in Example 5 by using the pdf given by Eqn. 3 rather than that given by Eqn. 4. In many other problems, as well, in which there is a difficulty of this type in regard to the existence of an MLE, the difficulty can be avoided simply by choosing one particular appropriate version of the pdf to represent the given distribution.

Example 7: Suppose that $X_1, \cdots, X_n$ form a random sample from a uniform distribution on the interval $(\theta, \theta + 1)$, where the value of the parameter θ is unknown $(-\infty < \theta < \infty)$. Clearly, the density function is

$$f(x|\theta) = \begin{cases} 1, & \text{for} \quad \theta \leq x \leq \theta + 1 \\ 0, & \text{otherwise} \end{cases}$$

We will see that the MLE for θ is not unique.

Proof: In this example, the likelihood function is

$$L(\theta) = \begin{cases} 1, & \text{for} \quad \theta \leq x_i \leq \theta + 1 \quad (i = 1, \cdots, n) \\ 0, & \text{otherwise} \end{cases}$$

The condition that $\theta \leq x_i$ for $i = 1, \cdots, n$ is equivalent to the condition that $\theta \leq \min(x_1, \cdots, x_n)$. Similarly, the condition that $x_i \leq \theta + 1$ for $i = 1, \cdots, n$ is equivalent to the condition that $\theta \geq \max(x_1, \cdots, x_n) - 1$. Therefore, we can rewrite the the likelihood function as

$$L(\theta) = \begin{cases} 1, & \text{for} \quad \max(x_1, \cdots, x_n) - 1 \leq \theta \leq \min(x_1, \cdots, x_n) \\ 0, & \text{otherwise} \end{cases}$$

Thus, we can select any value in the interval $[\max(x_1, \cdots, x_n) - 1, \min(x_1, \cdots, x_n)]$ as the MLE for θ. Therefore, the MLE is not uniquely specified in this example.

3 Exercises

Exercise 1: Let $X_1, \cdots, X_n$ be an i.i.d. sample from a Poisson distribution with parameter λ, i.e.,

$$P(X = x|\lambda) = \frac{\lambda^x e^{-\lambda}}{x!}.$$

Please find the MLE of the parameter λ.

Exercise 2: Let $X_1, \cdots, X_n$ be an i.i.d. sample from an exponential distribution with the density function

$$f(x|\beta) = \frac{1}{\beta} e^{-\frac{x}{\beta}}, \text{ with} \quad 0 \leq x < \infty.$$

Please find the MLE of the parameter β.

Exercise 3: Gamma distribution has a density function as

$$f(x|\alpha,\lambda)=\frac{1}{\Gamma(\alpha)}\lambda^{\alpha}x^{\alpha-1}e^{-\lambda x},\ \text{with}\quad 0\le x<\infty.$$

Suppose the parameter α is known, please find the MLE of λ based on an i.i.d. sample $X_1,\cdots,X_n$.

Exercise 4: Suppose that $X_1,\cdots,X_n$ form a random sample from a distribution for which the pdf $f(x|\theta)$ is as follows:

$$f(x|\theta)=\begin{cases}\theta x^{\theta-1}, & \text{for}\quad 0<x<1\\ 0, & \text{for}\quad x\le 0\end{cases}$$

Also suppose that the value of θ is unknown ($\theta>0$). Find the MLE of θ.

Exercise 5: Suppose that $X_1,\cdots,X_n$ form a random sample from a distribution for which the pdf $f(x|\theta)$ is as follows:

$$f(x|\theta)=\frac{1}{2}e^{-|x-\theta|}\quad \text{for}\quad -\infty<x<\infty$$

Also suppose that the value of θ is unknown ($-\infty<\theta<\infty$). Find the MLE of θ.

Exercise 6: Suppose that $X_1,\cdots,X_n$ form a random sample from a distribution for which the pdf $f(x|\theta)$ is as follows:

$$f(x|\theta)=\begin{cases}e^{\theta-x}, & \text{for}\quad x>\theta\\ 0, & \text{for}\quad x\le\theta\end{cases}$$

Also suppose that the value of θ is unknown ($-\infty<\theta<\infty$). a) Show that the MLE of θ does not exist. b) Determine another version of the pdf of this same distribution for which the MLE of θ will exist, and find this estimate.

Exercise 7: Suppose that $X_1,\cdots,X_n$ form a random sample from a uniform distribution on the interval (θ_1,θ_2), with the pdf as follows:

$$f(x|\theta)=\begin{cases}\frac{1}{\theta_2-\theta_1}, & \text{for}\quad \theta_1\le x\le\theta_2\\ 0, & \text{otherwise}\end{cases}$$

Also suppose that the values of θ_1 and θ_2 are unknown ($-\infty<\theta_1<\theta_2<\infty$). Find the MLE's of θ_1 and θ_2.

Machine Learning

Regression

Indian Institute of Information Technology
Sri City, Chittoor

Introduction to Regression

- Regression analysis is the part of statistics that investigates the relationship between two or more variables related in a nondeterministic fashion.

- Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data.

- One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable.

- Before attempting to fit a linear model to observed data, a modeler should first determine whether or not there is a relationship between the variables of interest.

Introduction to Linear Regression

- This does not necessarily imply that one variable causes the other (for example, ***higher SAT scores do not cause higher college grades***), but that there is some significant association between the two variables.

- A scatterplot can be a helpful tool in determining the strength of the relationship between two variables.

- If there appears to be no association between the proposed explanatory and dependent variables (i.e., the scatterplot does not indicate any increasing or decreasing trends), then fitting a linear regression model to the data probably will not provide a useful model.

Introduction to Linear Regression

- The simplest deterministic mathematical relationship between two variables x and y is a linear relationship.

$$y = b_0 + b_1 x$$

- The set of pairs (x, y) for which determines a straight line with **slope b_1 and y-intercept b_0**.

- More generally, the denoted by x and will be called the independent, predictor, or explanatory variable.

Introduction to Linear Regression

- Usually observations will be made for a number of settings of the independent variable.

- Let $x_1, x_2, \ldots, x_n$ denote values of the independent variable for which observations are made, and let Y_i and y_i , respectively, denote the random variable and observed value associated with.

- The available bivariate data then consists of the n pairs

$$(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_n, y_n).$$

- A picture of this data called a **scatter plot** gives preliminary impressions about the nature of any relationship.

- In such a plot, each (x_i, y_i) is represented as a point plotted on a two-dimensional coordinate system.

Scatter Plot Example: Linear Regression

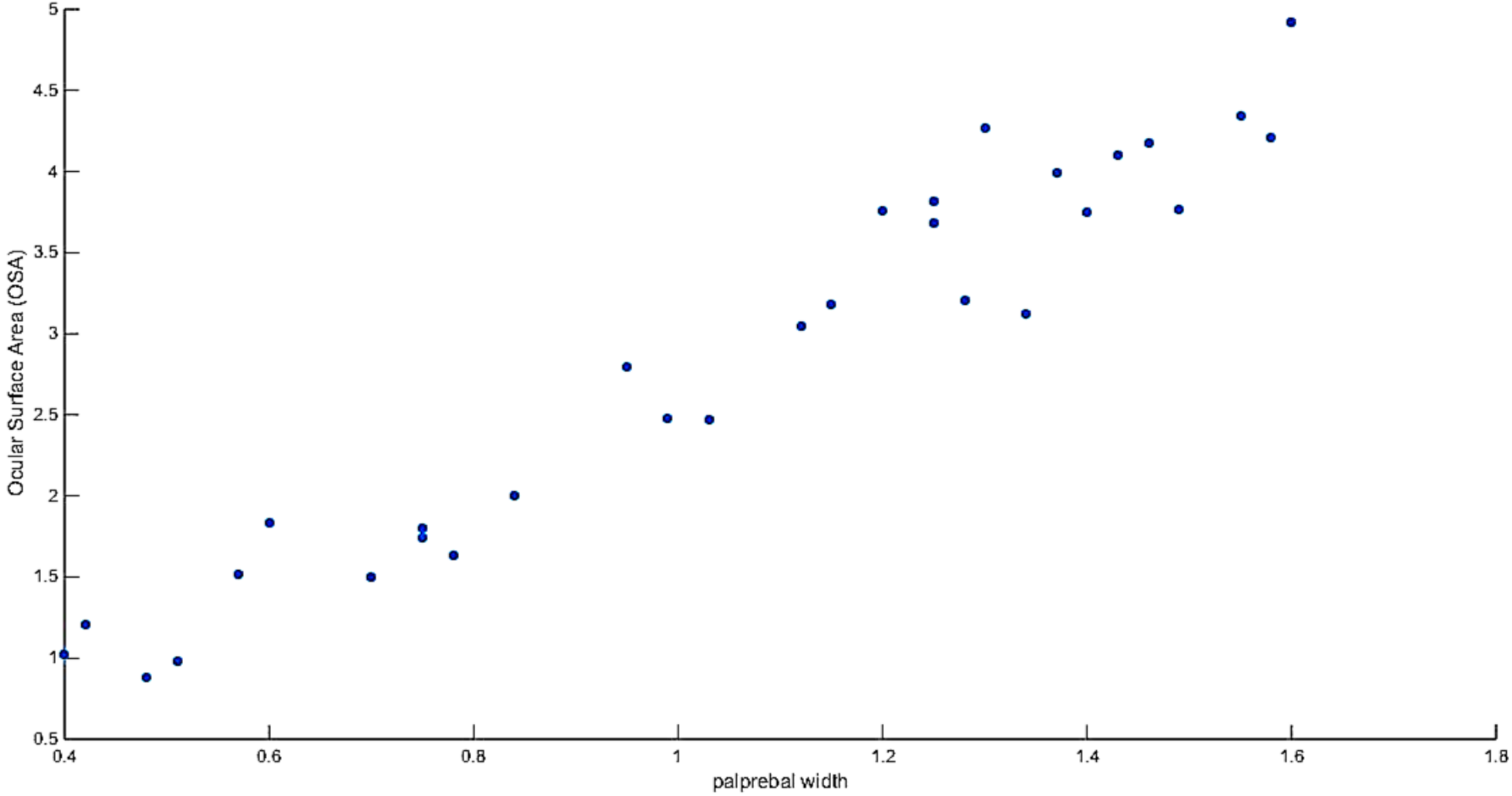
Visual and musculoskeletal problems associated with the use of visual display terminals (VDTs) have become rather common in recent years. Some researchers have focused on vertical gaze direction as a source of eye strain and irritation. This direction is known to be closely related to ocular surface area (OSA), so a method of measuring OSA is needed. The accompanying representative data on y = OSA (cm^2) and x = width of the palprebal fissure (i.e., the horizontal width of the eye opening, in cm) is from the article "Analysis of Ocular Surface Area for Comfortable VDT Workstation Layout" (*Ergonomics*, 1996: 877–884). The order in which observations were obtained was not given, so for convenience they are listed in increasing order of x values.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_i	.40	.42	.48	.51	.57	.60	.70	.75	.75	.78	.84	.95	.99	1.03	1.12
y_i	1.02	1.21	.88	.98	1.52	1.83	1.50	1.80	1.74	1.63	2.00	2.80	2.48	2.47	3.05

i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
x_i	1.15	1.20	1.25	1.25	1.28	1.30	1.34	1.37	1.40	1.43	1.46	1.49	1.55	1.58	1.60
y_i	3.18	3.76	3.68	3.82	3.21	4.27	3.12	3.99	3.75	4.10	4.18	3.77	4.34	4.21	4.92

Credits: Probability and Statistics for Engineering and the Science by Jan Devore

Scatter Plot Example



Scatter Plot Example: Linear Regression

- Several observations have identical x values yet different y values.
  - *e.g., x_8=x_9 =0 .75, but y_8 =1.80 and y_9 = 1.74). Thus the value of y is not determined solely by x but also by various other factors.*

- There is a strong tendency for y to increase as x increases.

- It appears that the value of y could be predicted from x by *finding a line that is reasonably close to the points in the plot* (the authors of the cited article superimposed such a line on their plot).

- In other words, there is evidence of a substantial (though not perfect) linear relationship between the two variables.

Simple Linear Regression Model

- For the deterministic model , the actual observed value of y is a linear function of x.

- The appropriate generalization of this to a probabilistic model assumes that the expected value of Y is a linear function of x, but that for fixed x the variable Y differs from its expected value by a random amount.

- In a simple regression problem (a single x and a single y), the form of the model would be:

$$y = b_0 + b_1 * x$$

Simple Linear Regression Model

- When a coefficient becomes zero, it effectively removes the influence of the input variable on the model.
- The values b_0 and b_1 must be chosen so that they minimize the error.
- If sum of squared error is taken as a metric to evaluate the model, then goal to obtain a line that best reduces the error.

$$\text{Error} = \sum_{i=1}^{n} (actual\_output - predicted\_output) ** 2$$

This error is called the Squared Error or Sum of Squared Error.

Simple Linear Regression Model

$y = b_0 + b_1 * x$

- For model with one predictor,

$$b_0 = \bar{y} - b_1 \bar{x}$$

and

$$b_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

- Exploring ‘b1’:
  - If b1 > 0, then x(predictor) and y (target) have a positive relationship. That is increase in x will increase y.
  - If b1 < 0, then x(predictor) and y (target) have a negative relationship. That is increase in x will decrease y.

Credits: **Linear Regression — Detailed View by** Saishruthi Swaminathan

Simple Linear Regression Model

- Exploring 'b_0'
  - If the model include x=0, then the prediction will become meaningless with only b_0 in some real world scenarios.
  - For example, we have a dataset that relates height(x) and weight(y). Taking x = 0 (that is height as 0), will make equation have only b_0 value which is completely meaningless as in real-time height and weight can never be zero
  - The value of b_0 guarantee that residual have mean zero. If there is no 'b_0' term, then regression will be forced to pass over the origin. Both the regression co-efficient b_1 and prediction y will be biased.

Credits: **Linear Regression — Detailed View by** Saishruthi Swaminathan

Formal problem setting

- Input: $x_i \in R^n$, $i = 1, \ldots, m$
- Output: $y_i \in R$ (regression task)
- Model Parameters: $\theta \in R$
- Predicted Output: $\hat{y}_i \in R$
  - » E.g.: $\hat{y}_i = \theta_1 \cdot x_i + \theta_2$
- For convenience, we define a function that maps inputs to feature vectors

$$\phi : R^n \rightarrow R^k$$

Hence, we can write:

$$\hat{y}_i = \sum_{j=1}^{k} \theta_j \cdot \phi_j(x_i) \equiv \theta^T \phi(x_i)$$

Loss functions

- Want a model that performs "well" on the data we have

$$\text{I.e., } \hat{y}_i \approx y_i, \quad \forall i$$

- We measure "closeness" of $\hat{y}_i$ and y_i using *loss function*

$$\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$$

- Example: squared loss

$$\ell(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

Finding model parameters, and optimization

- Want to find model parameters such that minimize sum of costs over all input/output pairs

$$J(\theta) = \sum_{i=1}^{m} \ell(\hat{y}_i, y_i) = \sum_{i=1}^{m} (\theta^T \phi(x_i) - y_i)^2$$

- Write our objective formally as

$$\underset{\theta}{\text{minimize}} \quad J(\theta)$$

Gradient descent

- Search algorithm: Start with an initial guess for θ. Keep changing θ (by a little bit) to reduce $J(\theta)$

$$J(\theta) = \sum_{i=1}^{m} \ell(\hat{y}_i, y_i) = \sum_{i=1}^{m} (\theta^T \phi(x_i) - y_i)^2$$

- Gradient descent: $\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$, for all j

$$\frac{\partial J}{\partial \theta_j} = \frac{\partial \sum_{i=1}^{m} (\theta^T \phi(x_i) - y_i)^2}{\partial \theta_j} = \sum_{i=1}^{m} \frac{\partial (\theta^T \phi(x_i) - y_i)^2}{\partial \theta_j}$$

$$= \sum_{i=1}^{m} 2(\theta^T \phi(x_i) - y_i) \frac{\partial (\theta^T \phi(x_i) - y_i)}{\partial \theta_j}$$

$$= \sum_{i=1}^{m} 2(\theta^T \phi(x_i) - y_i) \phi(x_i)_j$$

- Repeat until "convergence":

$$\theta_j = \theta_j - \alpha \sum_{i=1}^{m} 2(\theta^T \phi(x_i) - y_i) \phi(x_i)_j, \text{ for all } j$$

Error of Regression

- The expected squared error at a point x is given by,

$$Err(x) = E\left[(Y - \hat{f}\,(x))^2\right]$$

- The Err(x) can be further decomposed as expression for the expectation of the loss function:

$$Err(x) = \left(E[\hat{f}\,(x)] - f(x)\right)^2 + E\left[\left(\hat{f}\,(x) - E[\hat{f}\,(x)]\right)^2\right] + \sigma_e^2$$

$$Err(x) = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

- Err(x) is the sum of Bias^2, variance and the irreducible error.

- *Irreducible error is the error that can't be reduced by creating good models. It is a measure of the amount of noise in our data*.

Bias

- Given: dataset *D* with *m* samples.

- Learn: for different datasets D, you will get different functions f(x):
- Expected prediction (averaged over hypotheses): $E_D\left[f\left(x\right)\right]$

- Bias: difference between expected prediction and ground truth

– Measures how well you expect to represent true solution

– Decreases with more complex model

$$\text{Bias}^2 = \left(E[\hat{f}\left(x\right)] - f(x)\right)^2$$

Variance

- Given: dataset *D* with *m* samples.

- Learn: for different datasets D, you will get different functions f(x):
- Expected prediction (averaged over hypotheses): $E_D\,[f(x)]$

- Variance : difference between what you expect to learn and what you learn from a from a particular dataset.

– Measures how sensitive is learner is to the specific dataset

– Decreases with the simpler model.

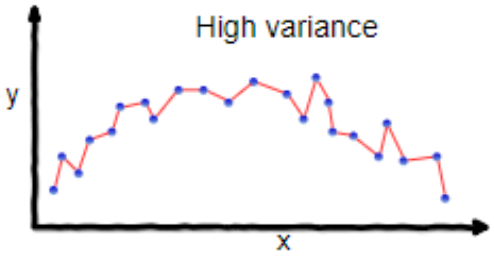
Variance= $E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)^2\right]$

Underfitting

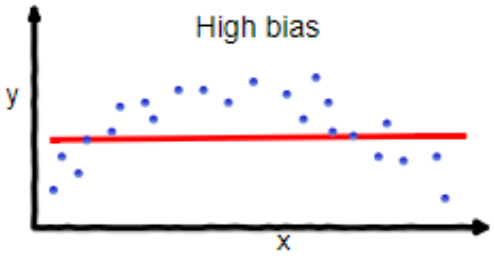
- In supervised learning, **underfitting** happens when a model unable to capture the underlying pattern of the data.
- These models usually have high bias and low variance.
- It happens when we have very less amount of data to build an accurate model or when we try to build a linear model with a nonlinear data.
- Also, these kind of models are very simple to capture the complex patterns in data like Linear and logistic regression.

Overfitting

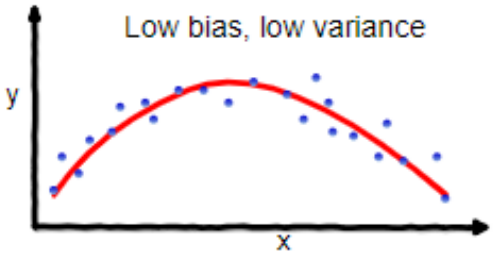
- In supervised learning, **overfitting** happens when our model captures the noise along with the underlying pattern in data.
- It happens when we train our model a lot over noisy dataset.
- These models have low bias and high variance.
- These models are very complex like Decision trees which are prone to overfitting.



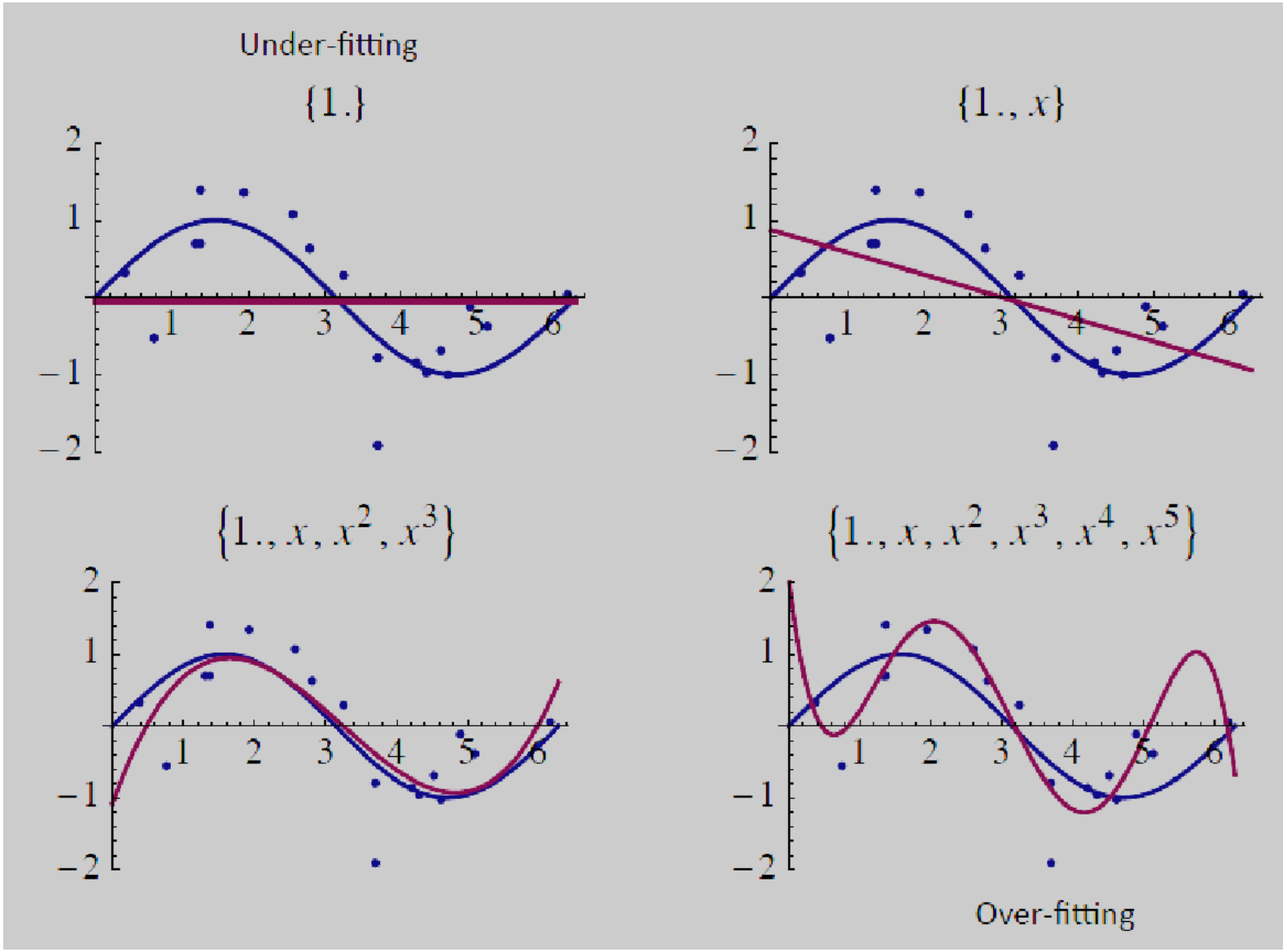
overfitting



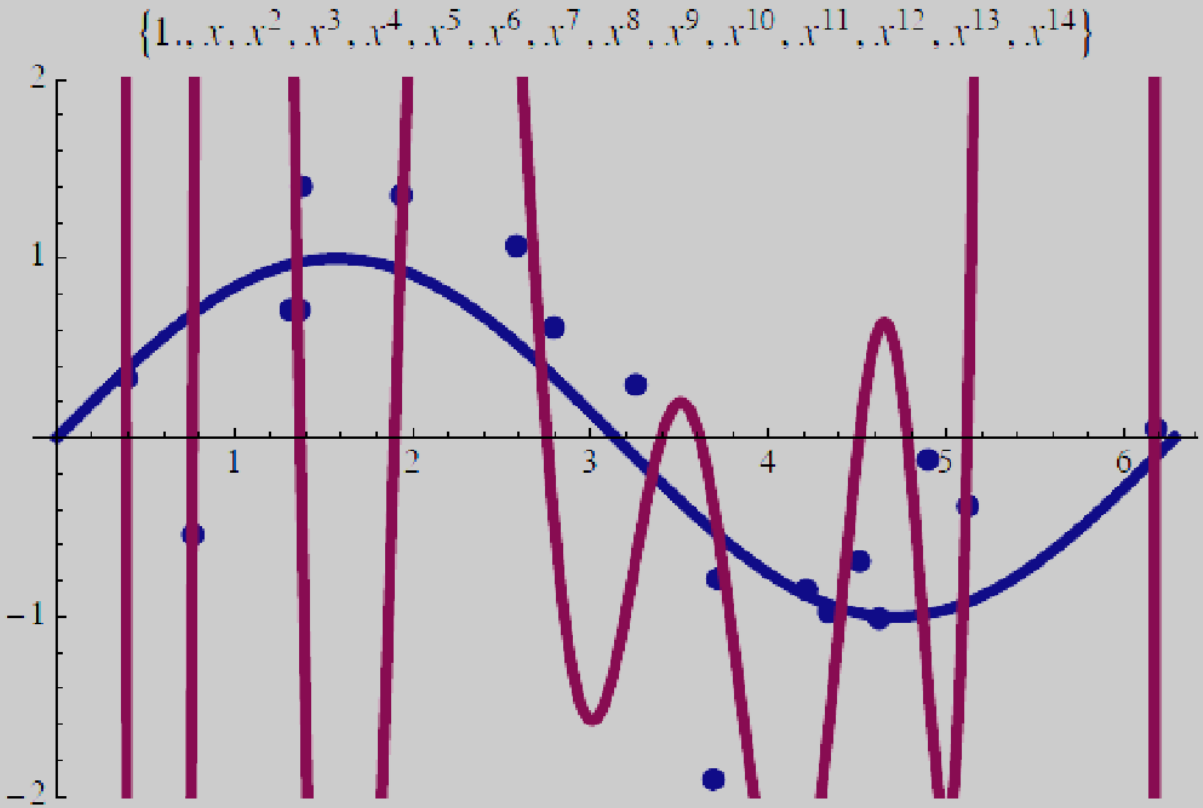
underfitting



Good balance

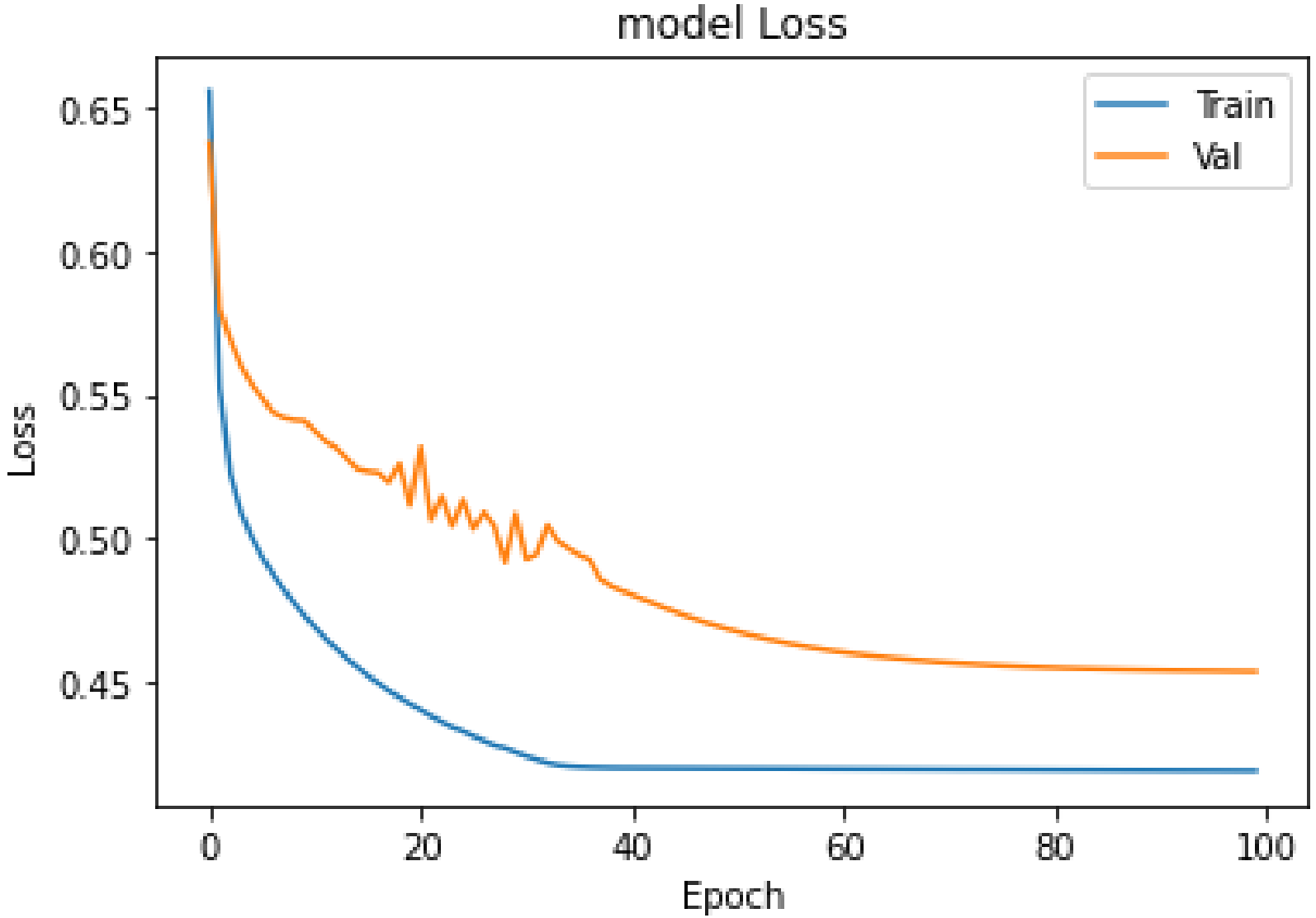

Under-fitting
{1.}
{1., x}
{1., x, x^2, x^3}
{1., x, x^2, x^3, x^4, x^5}
Over-fitting

Really Over-fitting!



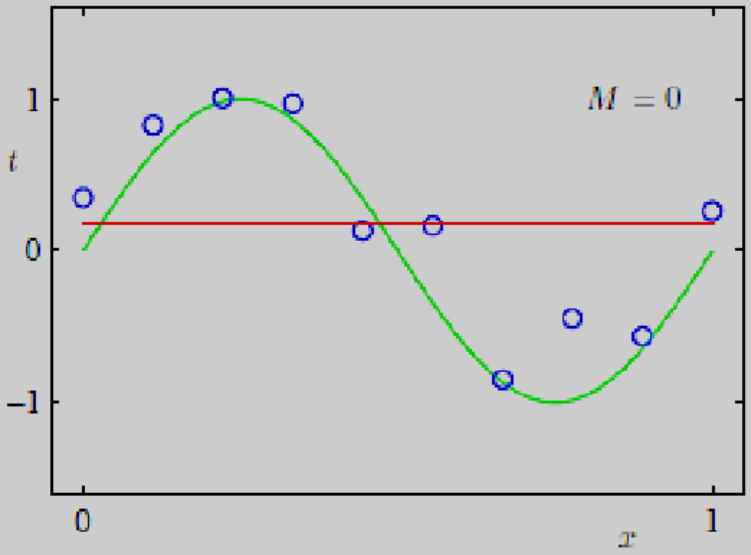
- Errors on training data are small
- But errors on new points are likely to be large

Generalization curve, which shows the loss for both the training set and validation set against the number of training iterations.

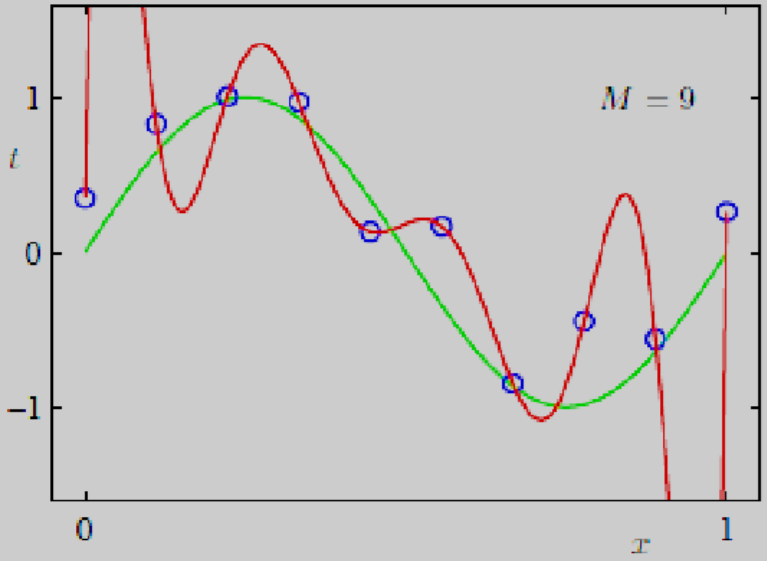


Bias Variance trade-off Intuition

- Model too simple: does not fit the data well
  - A *biased* solution



- Model too complex: small changes to the data, solution changes a lot
  - A *high-variance* solution



Regularization

- Regularization is a technique used to reduce the errors by fitting the function appropriately on the given training set and avoid overfitting.

- In context of machine learning, most regularization strategies are based on regularizing estimators. This is done through reducing variance at the expense of increasing the bias of the estimator.

- In other words, we could prevent overfitting by penalizing complex models, a principle called **regularization**.

Regularization

- In other words, instead of simply aiming to minimize loss (**empirical risk minimization**):

  minimize(Loss(Data|Model))

- we'll now minimize loss+complexity, which is called **structural risk minimization**:

  minimize(Loss(Data|Model)+complexity(model)

- Our training optimization algorithm is now a function of two terms: the **loss term**, which measures how well the model fits the data, and the **regularization term**, which measures model complexity.

https://developers.google.com/machine-learning/crash-course/regularization-for-simplicity/l2-regularization

Regularization

- In other words, instead of simply aiming to minimize loss (**empirical risk minimization**):

  minimize(Loss(Data|Model))

- we'll now minimize loss+complexity, which is called **structural risk minimization**:

  minimize(Loss(Data|Model)+complexity(model)

- Our training optimization algorithm is now a function of two terms: the **loss term**, which measures how well the model fits the data, and the **regularization term**, which measures model complexity.

https://developers.google.com/machine-learning/crash-course/regularization-for-simplicity/l2-regularization

Regularization types

- The commonly used regularization techniques are :

- L1 regularisation (Lasso Regression)

- L2 regularisation (Ridge regression)

Regularization

Regularization refers to the act of modifying a learning algorithm to favor “simpler” prediction rules to avoid overfitting.

Most commonly, regularization refers to modifying the loss function to **penalize** certain values of the weights you are learning.

- Specifically, penalize weights that are *large*.

Regularization

How do we define whether weights are *large*?

$$d(\mathbf{w}, \mathbf{0}) = \sqrt{\sum_{i=1}^{k} (w_i)^2} = ||\mathbf{w}||$$

This is called the **L2 norm** of **w**

- A norm is a measure of a vector's length
- Also called the Euclidean norm

Regularization

New goal for minimization:

$$L(\mathbf{w}) + \lambda \|\mathbf{w}\|^2$$

This is whatever loss function we are using

Regularization

New goal for minimization:

$$L(\mathbf{w}) + \lambda \|\mathbf{w}\|^2$$

By minimizing this, we prefer solutions where $\mathbf{w}$ is closer to $\mathbf{0}$.

Regularization

New goal for minimization:

$$L(\mathbf{w}) + \underbrace{\lambda \|\mathbf{w}\|^2}$$

Why squared? It eliminates the square root; easier to work with mathematically.

By minimizing this, we prefer solutions where $\mathbf{w}$ is closer to $\mathbf{0}$.

Regularization

New goal for minimization:

$$L(\mathbf{w}) + \underbrace{\lambda \|\mathbf{w}\|^2}$$

Why squared? It eliminates the square root; easier to work with mathematically.

By minimizing this, we prefer solutions where **w** is closer to **0**.

λ is a **hyperparameter** that adjusts the tradeoff between having low training loss and having low weights.

Regularization

More generally:

$$L(\mathbf{w}) + \lambda R(\mathbf{w})$$

This is called the **regularization term** or **regularizer** or **penalty**

- The squared L2 norm is one kind of penalty, but there are others

λ is called the regularization **strength**

L2 Regularization

When the regularizer is the squared L2 norm $||\mathbf{w}||^2$, this is called L2 regularization.

- This is the most common type of regularization
- When used with linear regression, this is called *Ridge regression*
- Logistic regression implementations usually use L2 regularization by default
  - L2 regularization can be added to other algorithms like perceptron (or any gradient descent algorithm)

L2 Regularization

The function $R(\mathbf{w}) = ||\mathbf{w}||^2$ is convex, so if it is added to a convex loss function, the combined function will still be convex.

L1 Regularization

Another common regularizer is the L1 norm:

$$||\mathbf{w}||_1 = \sum_{j=1}^{k} |w_j|$$

- When used with linear regression, this is called *Lasso*
- Often results in many weights being exactly 0 (while L2 just makes them small but nonzero)

L2+L1 Regularization

L2 and L1 regularization can be combined:

$$R(\mathbf{w}) = \lambda_2 ||\mathbf{w}||^2 + \lambda_1 ||\mathbf{w}||_1$$

- Also called *ElasticNet*
- Can work better than either type alone
- Can adjust hyperparameters to control which of the two penalties is more important

Additional Resources:

- https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86
- http://www.stat.yale.edu/Courses/1997-98/101/linreg.htm
- http://www.stat.cmu.edu/~roeder/stat707/lectures.pdf
- http://people.stern.nyu.edu/wgreene/MathStat/GreeneChapter4.pdf
- http://www.seas.ucla.edu/~vandenbe/103/lectures/qr.pdf

Linear Regression

Closed Form Solution

Notation

- Let the given data is $\mathrm{D} = \{(X_1, y_1), \ldots, (X_n, y_n)\}$.
- Let $X_i \in R^d$ and $y_i \in R$
- Further, $X_i = [x_{i1} \quad \ldots \quad x_{id}]^t$

We believe that

- $\theta_0 + \theta_1 x_{i1} + \cdots + \theta_d x_{id} = y_i$

We like to solve

- $$\theta_0 + \theta_1 x_{11} + \cdots + \theta_d x_{1d} = y_1$$
$$\theta_0 + \theta_1 x_{21} + \cdots + \theta_d x_{2d} = y_2$$
$$\vdots$$
$$\theta_0 + \theta_1 x_{n1} + \cdots + \theta_d x_{nd} = y_n$$

In matrix form $Z\Theta = Y$

Solve this to find Θ

Z may not be invertible !!

$$Z\Theta = Y$$

- $$\begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Closest Solution: by minimizing the squared error

- Obtained by minimizing the criterion

$$J(\Theta) = \|Z\Theta - Y\|^2 = (Z\Theta - Y)^t(Z\Theta - Y)$$

$$J(\Theta) = \Theta^t Z^t Z\Theta - 2(Z\Theta)^t Y - Y^t Y$$

$$J(\Theta) = \Theta^t Z^t Z\Theta - 2(Z\Theta)^t Y - Y^t Y$$

- Equating gradient of J to zero. So

$$\nabla_\Theta(J) = 2(Z^t Z)\Theta - 2Z^t Y = 0$$

We get $$\Theta = (Z^t Z)^{-1} Z^t Y$$

Example: 1D problem

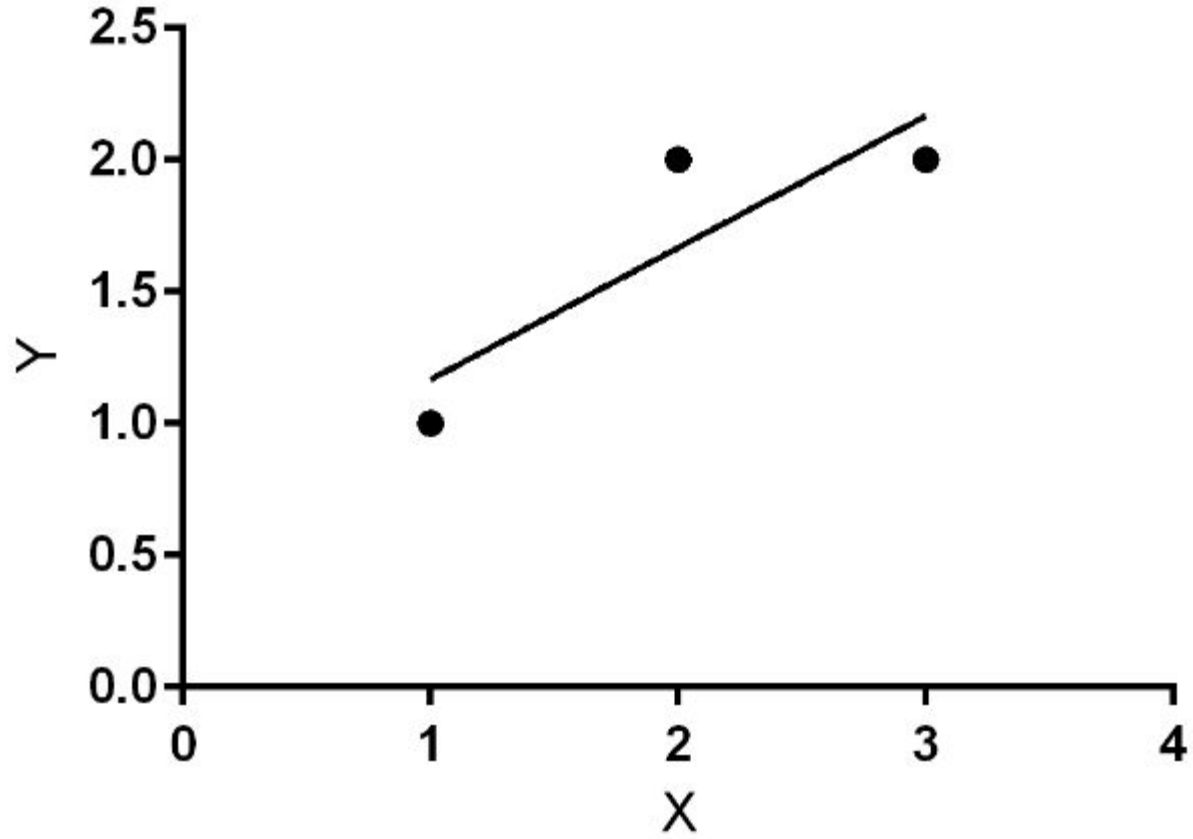
- $D = \{(1,1), (2,2), (3,2)\}$

- $Z = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}, \Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}, Y = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}$

- $Z^t Z = \begin{bmatrix} 3 & 6 \\ 6 & 14 \end{bmatrix} \quad (Z^t Z)^{-1} = \frac{1}{6}\begin{bmatrix} 14 & -6 \\ -6 & 3 \end{bmatrix}$

- $\Theta = (Z^t Z)^{-1} Z^t Y$
- $\Theta = \frac{1}{6}\begin{bmatrix} 14 & -6 \\ -6 & 3 \end{bmatrix}\begin{bmatrix} 5 \\ 11 \end{bmatrix} = \begin{bmatrix} 2/3 \\ 1/2 \end{bmatrix}$

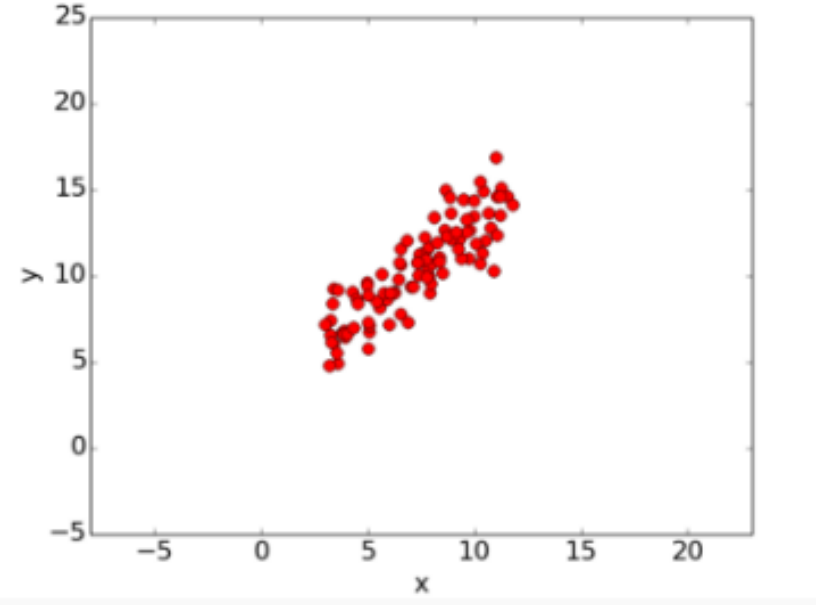
- So the line we fitted is $y = \frac{2}{3} + \frac{x}{2}$


2.5
2.0
1.5
1.0
0.5
0.0
Y
0
1
2
3
4
X

Linear Regression – Numerical Solution

Iterative solution

Linear regression – an example that uses gradient descent



We want to fit a straight line (in 2D case). The sample we are given with is $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$.

We believe the relation between x and y:

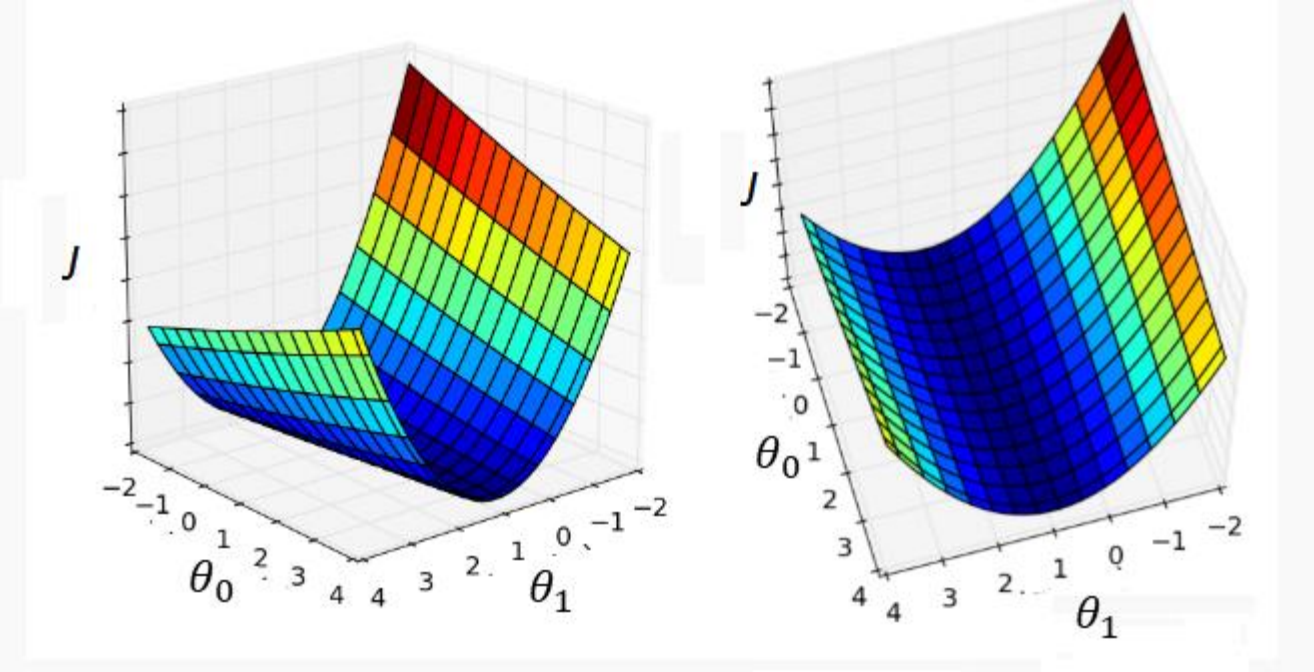
$$y = \theta_0 + \theta_1 x = h(x)$$

We want to find (θ_0, θ_1).

In the space (θ_0, θ_1). We want to find the best solution.

Sum of Squared Error = J(θ_0 , θ_1)

J(θ_0 , θ_1) = $\sum_{i=1}^{n}$(y_i−($\theta_0 + \theta_1 x_i$))^2



$$\nabla_{\binom{\theta_0}{\theta_1}} J = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix}$$

$$\frac{\partial J}{\partial \theta_0} = 2 \sum_{i=1}^{n} -(y_i - (\theta_0 + \theta_1 x_i))$$

$$\frac{\partial J}{\partial \theta_1} = 2 \sum_{i=1}^{n} -x_i (y_i - (\theta_0 + \theta_1 x_i))$$

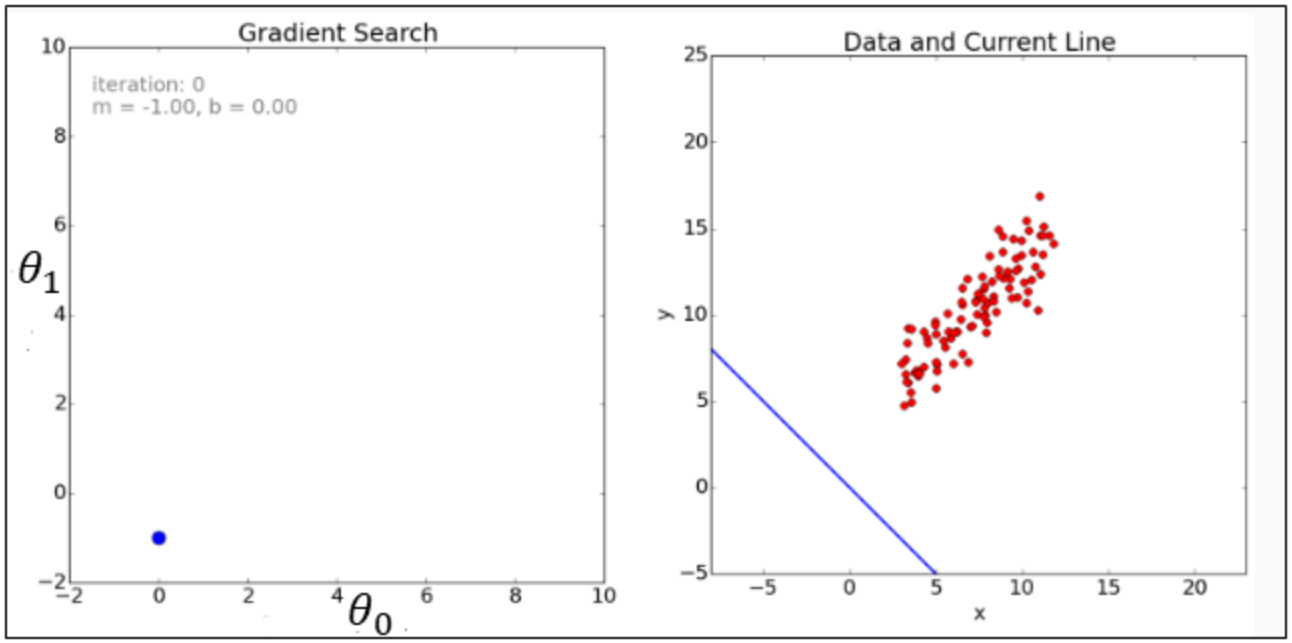
Gradient Descent Method:

Begin with random $\Theta = \binom{\theta_0}{\theta_1}$, we call this Θ_0

$\Theta_1 = \Theta_0 - \eta \nabla \mathrm{J}$

In general $\Theta_{k+1} = \Theta_k - \eta \nabla \mathrm{J}$

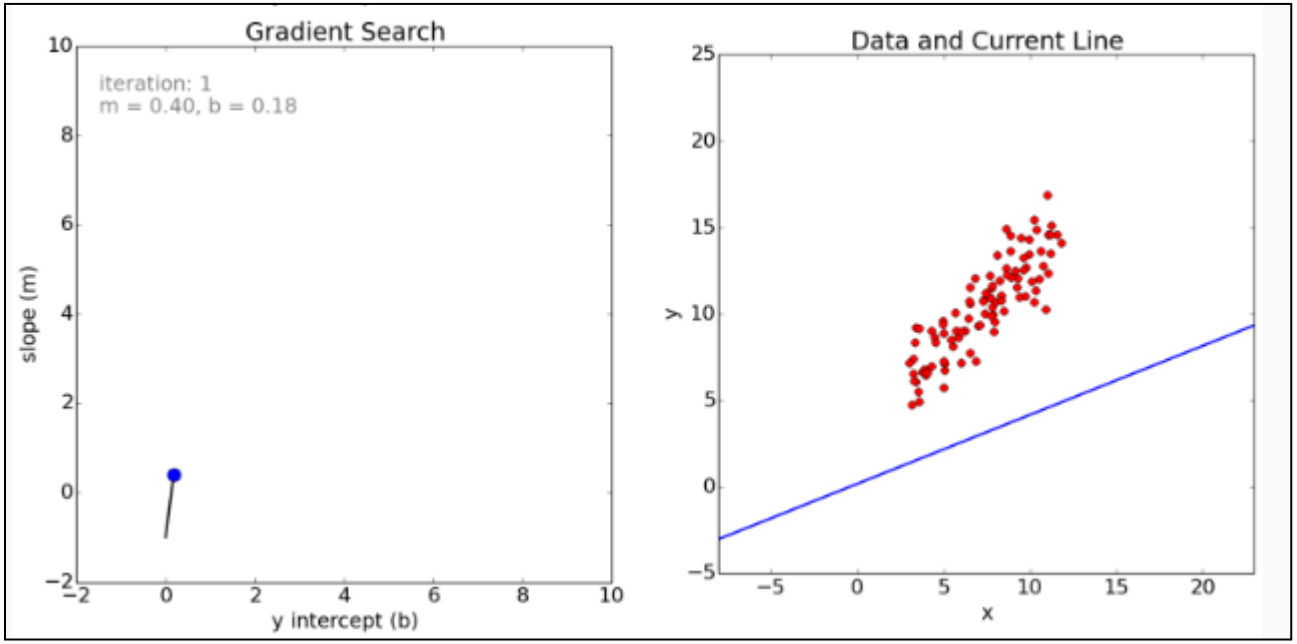
- $\Theta_0 = \begin{pmatrix} \theta_0 = 0 \\ \theta_1 = 1 \end{pmatrix}, \eta = 0.01$

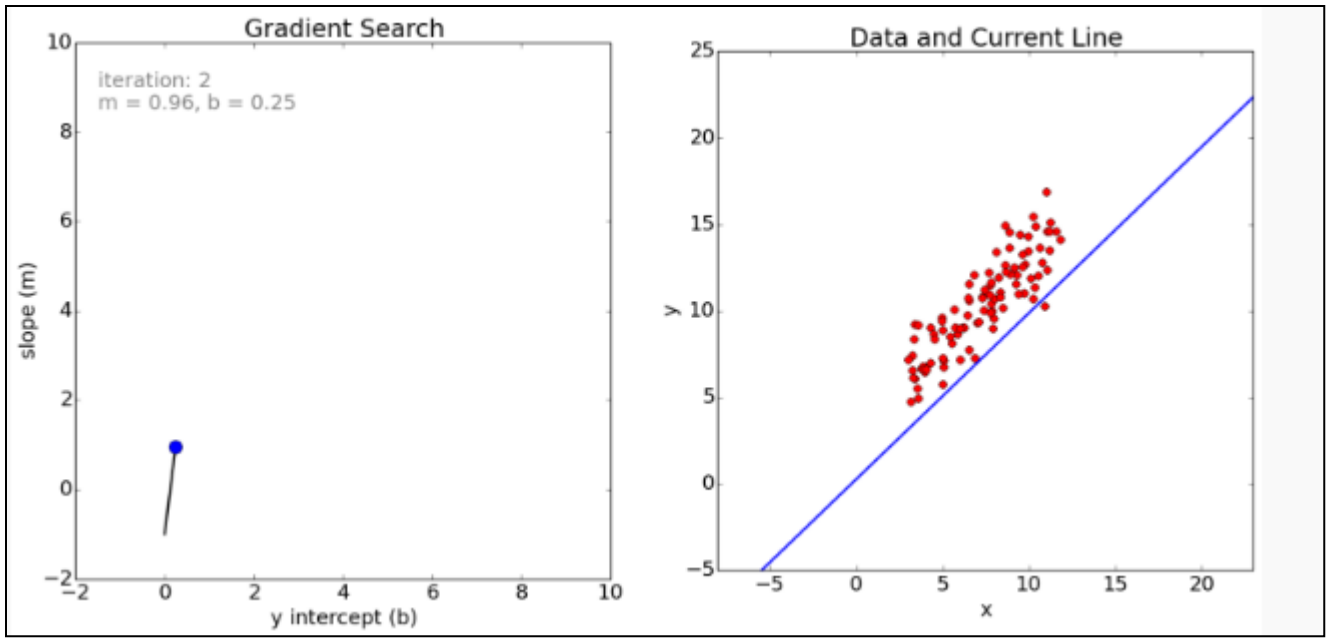


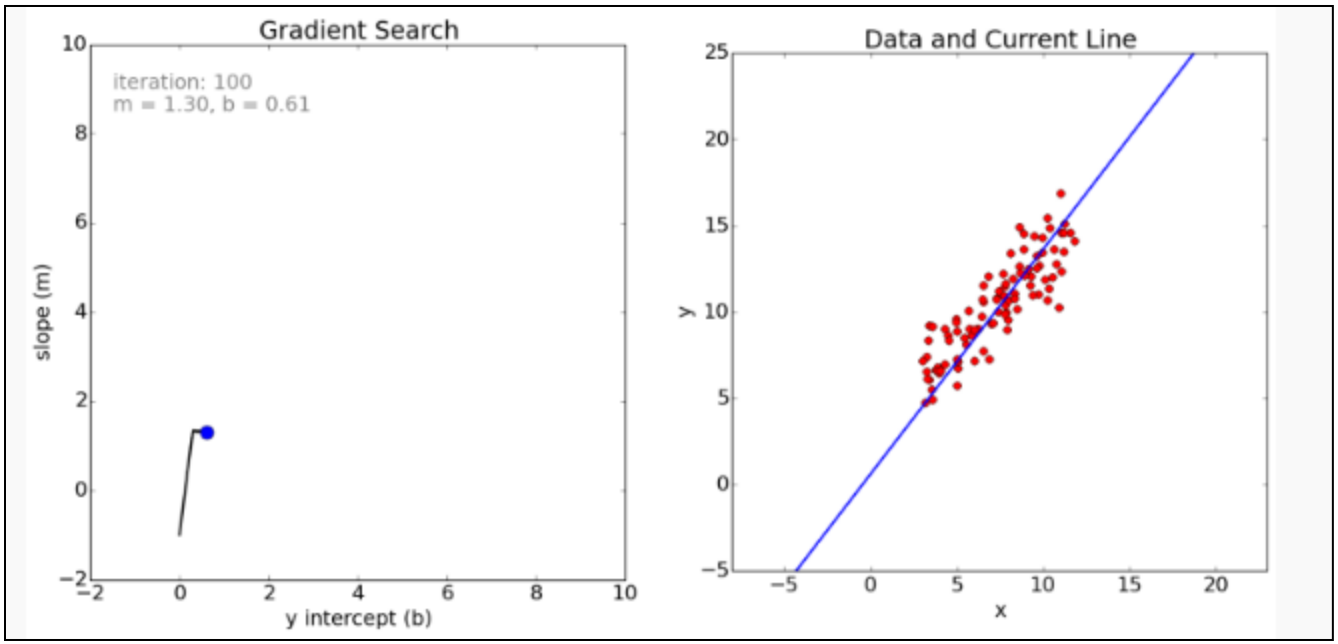
In 1D problem, often we call, $\theta_0 = \text{y-intercept} = b$

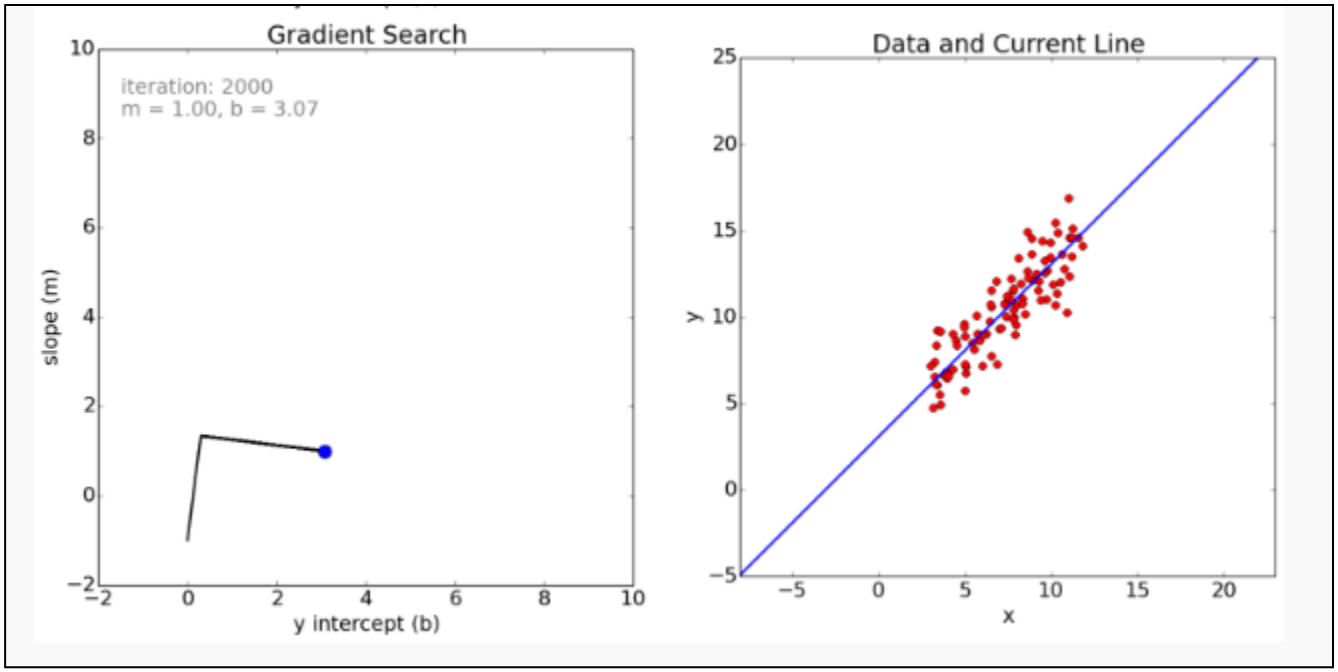
$\theta_1 = \text{slope} = m.$

Line equation is $y = mx + b$


Gradient Search
iteration: 1
m = 0.40, b = 0.18
slope (m)
y intercept (b)
Data and Current Line
y
x


Gradient Search
iteration: 2
m = 0.96, b = 0.25
slope (m)
y intercept (b)
Data and Current Line
y
x


Gradient Search
iteration: 100
m = 1.30, b = 0.61
slope (m)
y intercept (b)
Data and Current Line
y
x


Gradient Search
iteration: 2000
m = 1.00, b = 3.07
slope (m)
y intercept (b)
Data and Current Line
y
x

When to stop

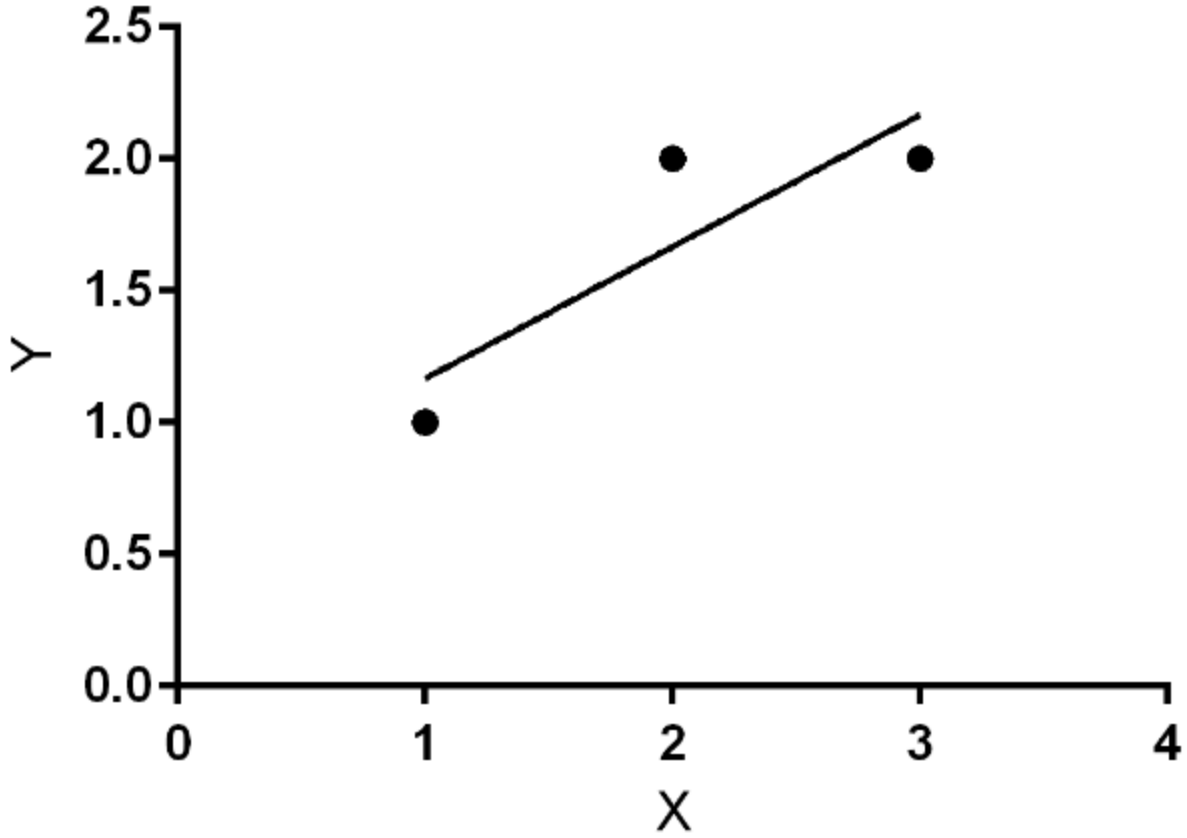
- Ideally when $\nabla J = 0$
- In practice, often when $\|\nabla J\| < \epsilon$ where ϵ is a small real number like 0.01 is chosen as the stopping condition.

Example

- Given $D = \{(1,1), (2,2), (3,2)\}$
- Let us begin with $\Theta_0 = \begin{pmatrix} \theta_0 = 0 \\ \theta_1 = 1 \end{pmatrix}$,
- Let $\eta = 0.1$

- Can you do this?

- Solution is, $y = \frac{2}{3} + \frac{x}{2}$



GENERALIZING TO MULTIVARIATE DATA

Notation

- Let the given data is $\mathrm{D} = \{(X_1, y_1), \ldots, (X_n, y_n)\}$.
- Let $X_i \in R^d$ and $y_i \in R$
- Further, $X_i = [x_{i1} \quad \ldots \quad x_{id}]^t$
- We augment each X_i with 1 in order to simplify.
- The augmented vector we call Z_i

-
- $Z_i = [1 \;\; x_{i1} \cdots x_{id}]^t = \begin{bmatrix} 1 \\ x_{i1} \\ \vdots \\ x_{id} \end{bmatrix}$

We like to solve

- $$\theta_0 + \theta_1 x_{11} + \cdots + \theta_d x_{1d} = y_1$$
$$\theta_0 + \theta_1 x_{21} + \cdots + \theta_d x_{2d} = y_2$$
$$\vdots$$
$$\theta_0 + \theta_1 x_{n1} + \cdots + \theta_d x_{nd} = y_n$$

In matrix form $\boldsymbol{Z}\Theta = Y$

-
- $Z_i = \begin{bmatrix} 1 \\ x_{i1} \\ \vdots \\ x_{id} \end{bmatrix} = [1 \quad x_{i1} \quad \dots \quad x_{id}]^t$
- $\boldsymbol{Z} = \begin{bmatrix} 1 & x_{11} & \dots & x_{1d} \\ 1 & x_{21} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \dots & x_{nd} \end{bmatrix} = \begin{bmatrix} -Z_1 - \\ \vdots \\ -Z_n - \end{bmatrix} = \begin{bmatrix} Z_1^t \\ \vdots \\ Z_n^t \end{bmatrix}$

$$Z\Theta = Y$$

-
- $\boldsymbol{Z} = \begin{bmatrix} -Z_1 - \\ \vdots \\ -Z_n - \end{bmatrix} = \begin{bmatrix} Z_1^t \\ \vdots \\ Z_n^t \end{bmatrix}$
- $\Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}$
- $Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

Criterion J

- $J(\Theta) = \|\boldsymbol{Z}\Theta - Y\|^2 = (\boldsymbol{Z}\Theta - Y)^t(\boldsymbol{Z}\Theta - Y)$

$$J(\Theta) = \Theta^t \boldsymbol{Z}^t \boldsymbol{Z}\Theta - 2(\boldsymbol{Z}\Theta)^t Y + Y^t Y$$

$$J(\Theta) = \Theta^t \boldsymbol{Z}^t \boldsymbol{Z}\Theta - 2(\boldsymbol{Z}\Theta)^t Y + Y^t Y$$

- $$\nabla_\Theta(J) = 2(\boldsymbol{Z}^t \boldsymbol{Z})\Theta - 2\boldsymbol{Z}^{\mathrm{t}} Y$$

Recall by equating the above to 0, we got the closed form solution which is,

$$\Theta = (\boldsymbol{Z}^t \boldsymbol{Z})^{-1} \boldsymbol{Z}^t Y$$

Iterative solution

- $\Theta_{k+1} = \Theta_k - \eta \nabla(J)$
- Note, this Gradient is at Θ_k

Batch Learning

- In each iteration entire training set is considered.
- $\Theta_{k+1} = \Theta_k - \eta(2(\boldsymbol{Z}^t\boldsymbol{Z})\Theta_k - 2\boldsymbol{Z}^tY)$
  $= \Theta_k - 2\eta\boldsymbol{Z}^t(\boldsymbol{Z}\Theta_k - Y)$

- Each element of Θ can be updated as below.
- $\theta_j = \theta_j - 2\eta\sum_{i=1}^{n}(h(Z_i) - y_i)x_j$

Stochastic Gradient Descent

- This is single sample correction method
- Consider one example, viz., (X_i, y_i)
- For X_i, its augmented vector is Z_i
- $Z_i = \begin{bmatrix} 1 \\ X_i \end{bmatrix}$
- $\Theta_{k+1} = \Theta_k - \eta\left(2\left(Z_i Z_i^t\right)\Theta_k - 2y_i Z_i\right)$

Normally X_i is chosen randomly from the training set.

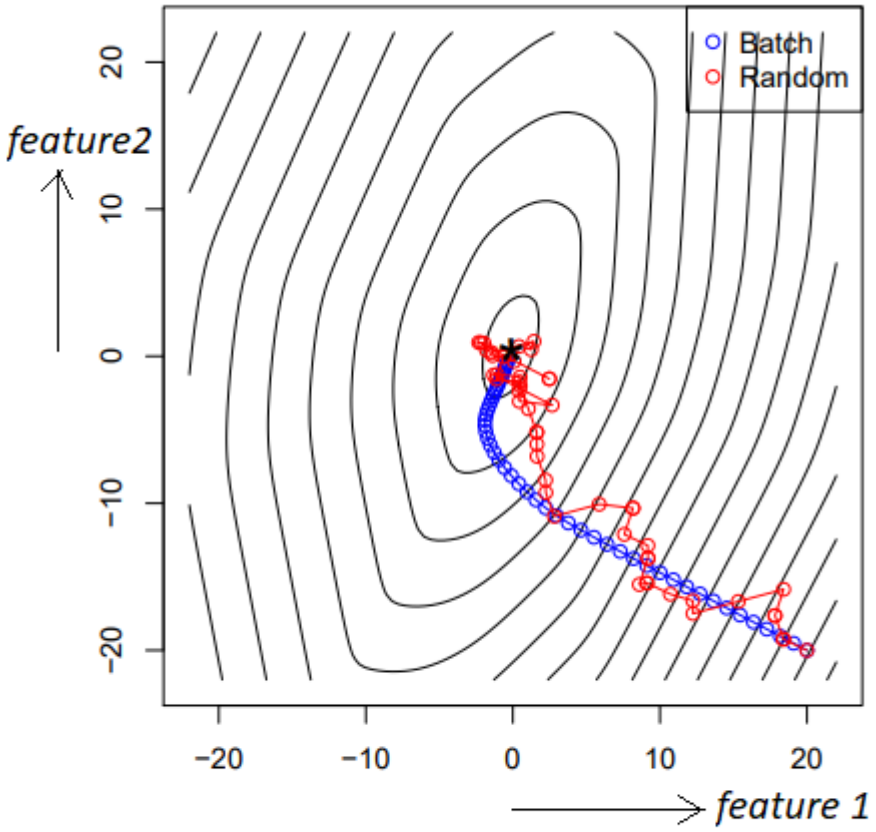
Stochastic Gradient Descent

- $\Theta_{k+1} = \Theta_k - \eta\left(2\left(Z_i Z_i^t\right)\Theta_k - 2y_i Z_i\right)$

$$= \Theta_k - 2\eta\left(Z_i^t\Theta_k - \mathrm{y}_i\right)Z_i$$

- Each element of Θ can be updated as below.
- $\theta_j = \theta_j - 2\eta(h(Z_i) - y_i)x_j$

Batch Vs Stochastic Convergence



Blue: batch steps,
Red: stochastic steps,

Stochastic is also known as random / single sample correction.

- Batch method smoothly converges
- Stochastic method strays away often from the optimal path.

Stopping Condition

- It is better to have a validation set (VS) and
  - keep track of error on the VS,
  - stop when the error on the VS is not decreasing considerably
- In practice we fix number of iterations.

Bias Variance Analysis

Ref: Domingos, Pedro. "A unified bias-variance decomposition." *Proceedings of 17th international conference on machine learning*. Morgan Kaufmann, 2000.

For Regression

Error

Error of a learning method can be decomposed into bias and variance.

Bias: Deviation from the true value (inherent weakness of the learning method)

Variance: Variance of the prediction over different training sets (variation in the prediction because the training set is varied).

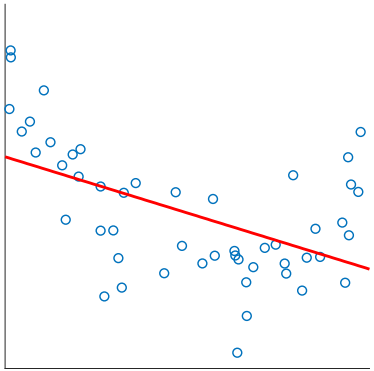
Error

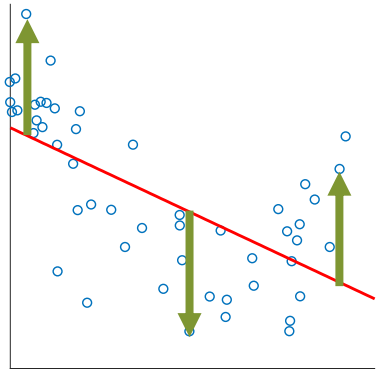
Error depends on the learning method (f) and on the training set (D_i).

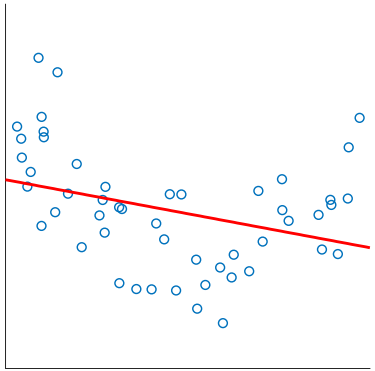
For the given test example X, the prediction is $f(X)$.

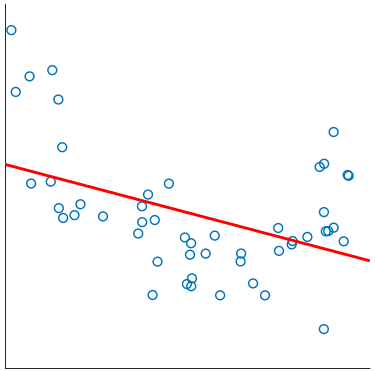
Actually since this depends on the training set D_i, we can write $f(X;\ D_i)$

Bias





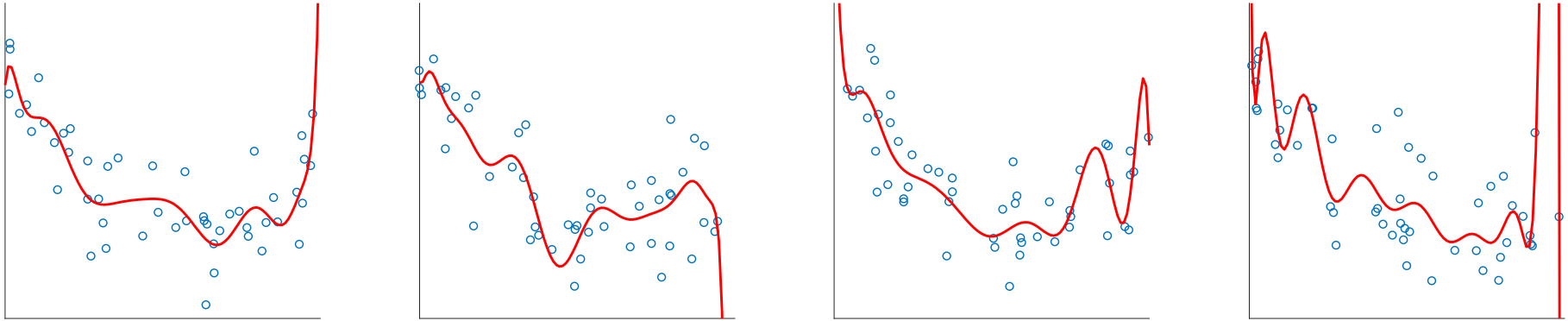




Regardless of training sample, or size of training sample, model will produce consistent errors

Actual relationship may be quadratic

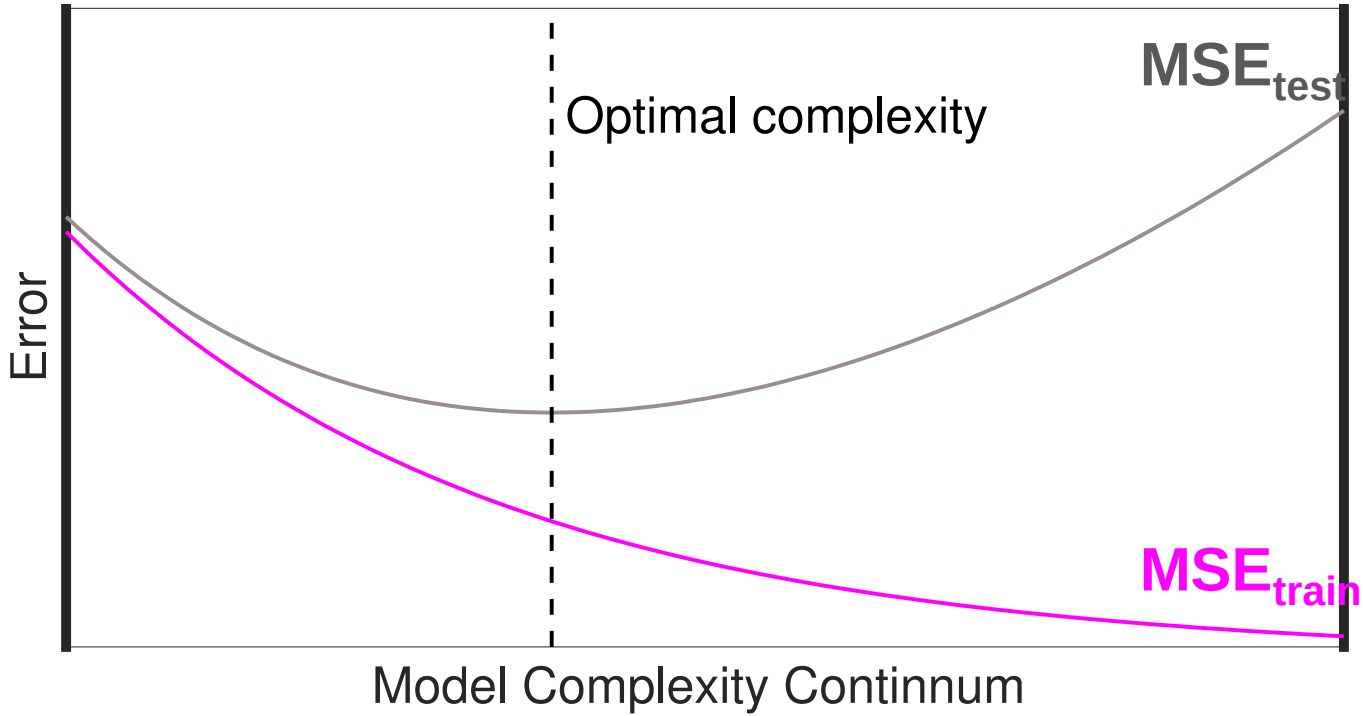
Variance



Different samples of training data yield different model fits

We are trying to fit degree 8 polynomial

Bias-Variance Trade Off Is Revealed Via Test Set Not Training Set



Bias increases ⟸ ⟹ **Variance increases**

Training sets

Let $D = \{D_1, \ldots, D_i, \ldots, D_{10}\}$, i.e., we are having 10 different training sets drawn from the same distribution.

Size of each D_i is the same. Let us say $|D_i| = n$.

Each training set will be like

$$D_i = \{(X_1, t_1), \ldots, (X_n, t_n)\}$$

Note, t is the target and $y = f(X; D_i)$ is the prediction

Mean prediction for x is called y_m

- $y_m = Mean\,\{f(X; D_1), f(X; D_2), \ldots, f(X;\ D_{10})\}$
- This is nothing but average prediction over the training sets.

Square Loss

$$L(y_i, y_j) = (y_i - y_j)^2$$

In regression we usually use the square loss.

Bias

- On average, deviation from the true prediction.
- For X, bias in the prediction is, $B(X) = L(E_t t, y_m) = (E_t t - y_m)^2$

$$= \left(E_t t - \frac{f(X; D_1) + \cdots + f(X; D_{10})}{10}\right)^2$$

Variance

- Variance in the prediction
- For X, variance in the prediction is, $V(X) = \frac{L(f(X; D_1), y_m) + \cdots + L(f(X; D_{10}), y_m)}{10}$

Note, for a single example (x,y) these are bias and variance (across the training sets)

Bias and Variance

- Bias and variance has to be found by averaging over the entire feature-space.
- Bias $= E_X[B(X)]$
- Variance $= E_X[V(X)]$

- In practice, we take average over the Test Set.

Test Set

- Let D_S be the test set.
- Let $D_s = \{(X_1, t_1), \ldots, (X_s, t_s)\}$
- Let $|D_s| = s$

- Bias $= \frac{1}{s}\sum_{k=1}^{s} B(X_k)$
- Variance $= \frac{1}{s}\sum_{k=1}^{s} V(X_k)$

- Note, this is the average Bias and Variance over the Test Set.
- The summation is over all Test Examples.

Distribution from which the training set is drawn

- We assume that the target which captures the correct relationship between X and t, is $t = \theta_0 + \theta_1 x_1 + \cdots + \theta_d x_d$
- Fix the values $\theta_0, \theta_1, \ldots, \theta_d$
- Generate X from the given distribution (Say Gaussian with given parameters)
- Calculate t, the true prediction for X.
- Generate noise ϵ from the Gaussian distribution $N(0,1)$.
- Update $t := t + \epsilon$
- Then the training example generated is (X, t)
- Generate n examples to create the training set D_i. One has to generate D_i for $i = 1, \ldots, 10$. Each of the D_i is generated independently.

Lab Exercise

- You need to generate 10 different training sets from the given source distribution, each of size n.
- You need to generate the test set of size s from the same distribution, but independent from the training set.
- Let us fix s to 100.
- But n can be varied from 100 to 1000. For simplicity let n take values 100, 200, …,1000.
- Find Bias and Variance for each of the n value.
- You can plot n vs Bias and n vs Variance.

Below are given at the start of the Lab exercise

- The learning method ie., the classifier
- The distributions
- You can use libraries (tool-box given) to generate data.

Note, For each test example, you need to find its mean prediction which is y_m which can change for each test example.

Formal Derivation

Formally (we introduce t …)

- Let D be the set of training sets (each of same size)
- Let t be the target (true value) for the given x.
- Note, t is a random variable. It depends on x. But is independent of any training set.
- y is the prediction which depends on the training set, hence we write $y = f(x, D_i)$ where D_i is the training set. y depends on D_i but is independent of t.

For the given x, let t be the target, and D be the set of training sets

- The expected loss incurred for the prediction of x is $E_{D,t}[L(t,y)] = E_{D,t}[(t-y)^2]$

$$= E_{D,t}(t - E_t t + E_t t - y)^2$$
$$= E_t(t - E_t t)^2 + E_{D,t}(E_t t - y)^2$$
$$= E_t(t - E_t t)^2 + E_{D,t}(E_t t - E_D y + E_D y - y)^2$$
$$= E_t\ (t - E_t t)^2 + (E_t t - E_D y)^2 + E_D(E_D y - y)^2$$
$$= N(x) + B(x) + V(x)$$

Here $N(x)$ is the noise, $B(x)$ is the Bias and $V(x)$ is the variance for x.

For Classification

0-1 Loss

$$L(y_i, y_j) = \begin{cases} 0, \text{ if } y_i = y_j \\ 1, \text{otherwise} \end{cases}$$

In classification we usually use 0-1 Loss

Main prediction for X is called y_m

- $y_m = Mode\ \{f(X; D_1), f(X; D_2), \ldots, f(X, D_{10})\}$
- This is nothing but majority (most frequent) prediction of f over the training sets.
  - In case of a Tie we break it randomly

The Bayes prediction for X

- y^* is the Bayes prediction for X
- Since we know the distributions, this can be found from the Bayes Classifier

Bias

- Deviation from the Bayes classifier.
- For X, bias in the prediction is, $B(X) = L(y_m, y^*)$

Variance

- Variance in the prediction
- For X, variance in the prediction is, $V(X) = \frac{L(f(X; D_1), y_m) + \cdots + L(f(X; D_{10}), y_m)}{10}$

Note, y_m , y^* are the main prediction and the Bayes prediction for the x, respectively.

Note

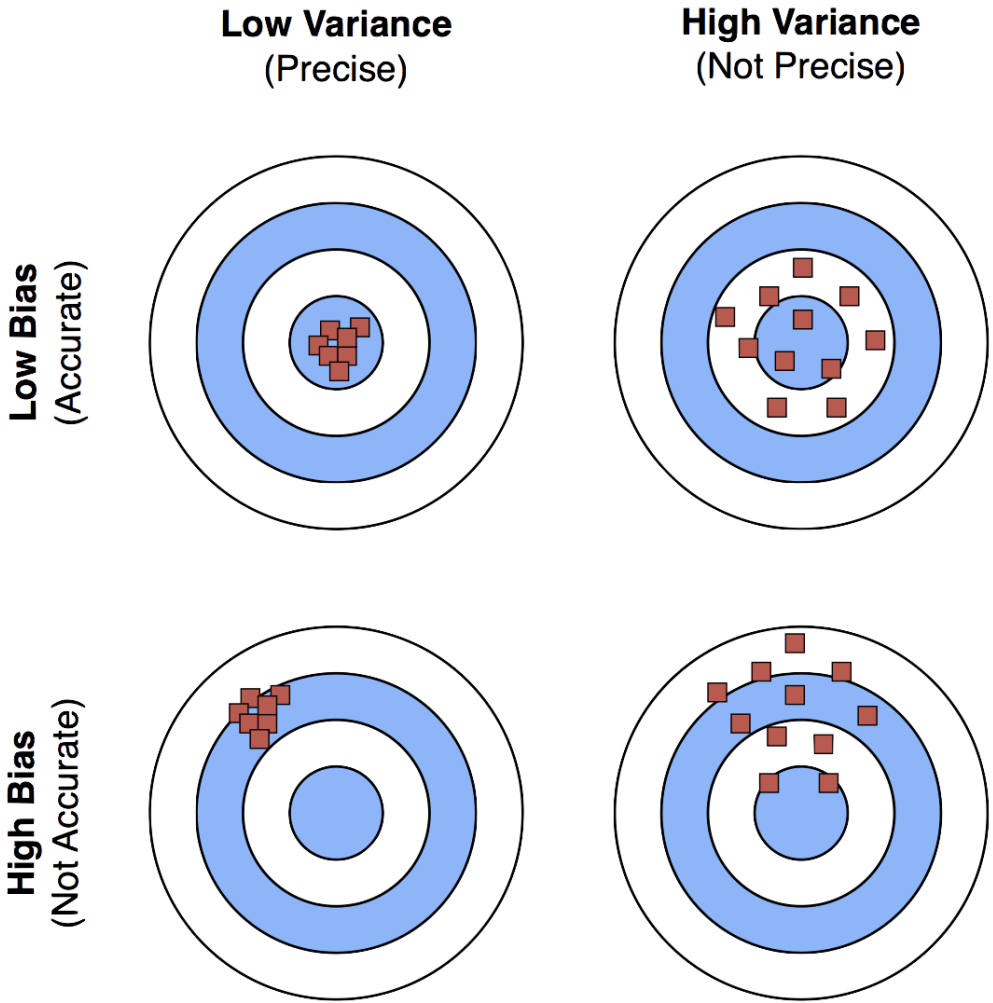
- We have taken 10 different training sets.
- In general, one may consider different number of training sets to estimate Bias and Variance.

In case of classification, the Bayes error forms the noise. One may refer to the paper

Domingos, Pedro. "A unified bias-variance decomposition." *Proceedings of 17th international conference on machine learning*. Morgan Kaufmann, 2000.

For formal derivations, refer to

- **Ref:** Domingos, Pedro. "A unified bias-variance decomposition." *Proceedings of 17th international conference on machine learning*. Morgan Kaufmann, 2000.


Low Variance
(Precise)
High Variance
(Not Precise)
Low Bias
(Accurate)
High Bias
(Not Accurate)

Bias Variance Analysis for Classification

For the Lab Exercise

Error

Error of a learning method can be decomposed into bias and variance.

For the Bayes Classifier, once the distributions are known we know its error.

Error

Error depends on the learning method (f) and on the training set (D_i).

For the given test example x, the prediction is $f(X)$.

Actually since this depends on the training set D_i, we can write $f(X;\ D_i)$

Training sets and the test set

Let $D = \{D_1, \ldots D_i \ldots, D_{10}\}$, i.e., we are having 10 different training sets drawn from the same distribution.

Size of each D_i is the same. Let us say $|D_i| = n$.

Main prediction for x is called y_m

- $y_m = Mode\ \{f(X; D_1), f(X; D_2), \ldots, f(X, D_{10})\}$
- This is nothing but majority (most frequent) prediction of f over the training sets.
  - In case of a Tie we break it randomly

The Bayes prediction for x

- y^* is the Bayes prediction for X
- Since we know the distributions, this can be found from the Bayes Classifier

Note, $y = f(X; D_i)$ is the prediction for X while using the training set D_i

0-1 Loss

$$L(y_i, y_j) = \begin{cases} 0, \text{ if } y_i = y_j \\ 1, \text{otherwise} \end{cases}$$

In classification we usually use 0-1 Loss

Bias

- Deviation from the Bayes classifier.
- For x, bias in the prediction is, $B(x) = L(y_m, y^*)$

Variance

- Variance in the prediction
- For x, variance in the prediction is, $V(x) = \frac{L(f(x; D_1), y_m) + \cdots + L(f(x; D_{10}), y_m)}{10}$

Note, y_m , y^* are the main prediction and the Bayes prediction for the x, respectively.

Bias and Variance

- Bias and variance has to be found by averaging over the entire feature-space.
- Bias $= E_X[B(x)]$
- Variance $= E_X[V(x)]$

- In practice, we take average over the Test Set.

Test Set

- Let D_S be the test set.
- Let $D_s = \{(X_1, t_1), \ldots, (X_s, t_s)\}$
- Let $|D_s| = s$

We are using the notation t for the target and y for the prediction.

- Bias $= \frac{1}{s}\sum_{k=1}^{s} B(x_k)$
- Variance $= \frac{1}{s}\sum_{k=1}^{s} V(x_k)$

- Note, this is the average Bias and Variance over the Test Set.
- The summation is over all Test Examples.

In this Lab Exercise

- You need to generate 10 different training sets from the given source distributions, each of size n.
- You need to generate the test set of size s from the same distribution, but independent from the training set.
- Let us fix s to 100.
- But n can be varied from 100 to 1000. For simplicity let n take values 100, 200, …,1000.
- Find Bias and Variance for each of the n value.
- You can plot n vs Bias and n vs Variance.

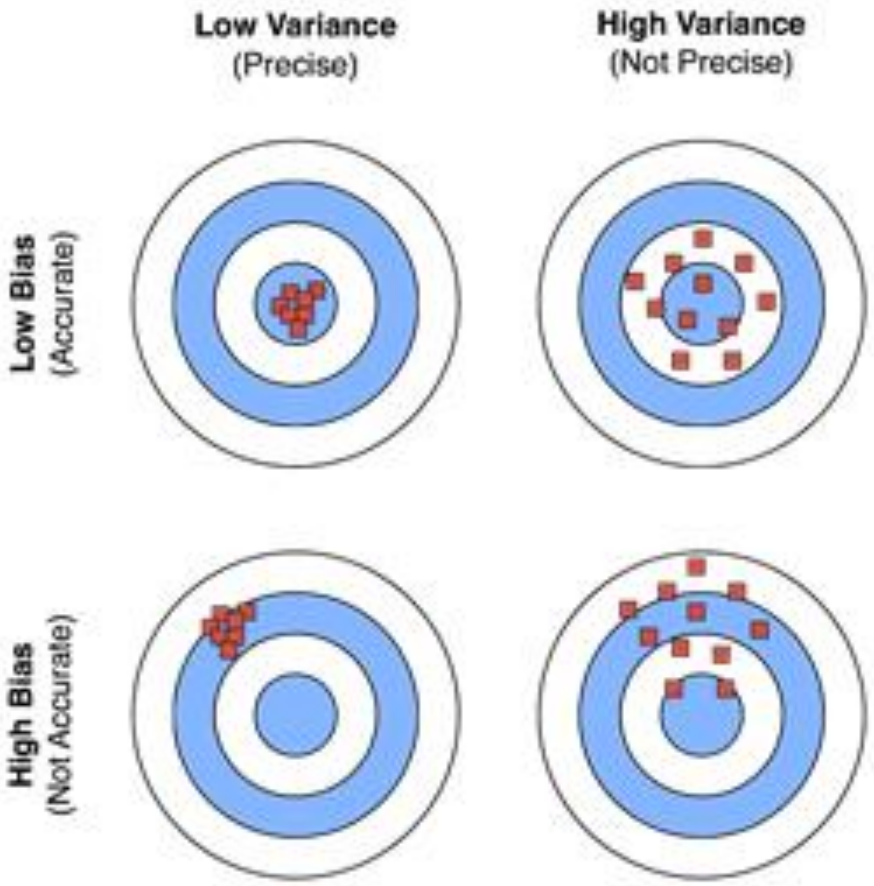
Below are given at the start of the Lab

- The learning method ie., the classifier
- The distributions (which includes apriori probabilities and class-conditional densities)
- You can use libraries (tool-box given) to generate data.

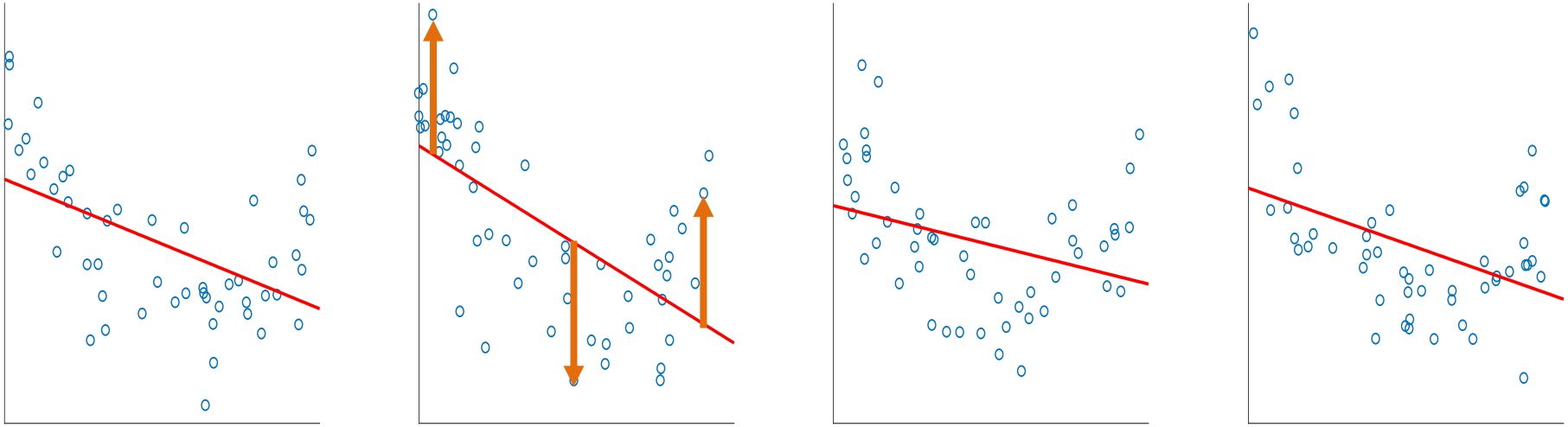
Note, For each test example, you need to find its main prediction which is y_m and y^* which is the Bayes prediction. For each test example, these predictions, changes (in general).

Regularization

A way to avoid overfitting


Low Variance
(Precise)
High Variance
(Not Precise)
Low Bias
(Accurate)
High Bias
(Not Accurate)

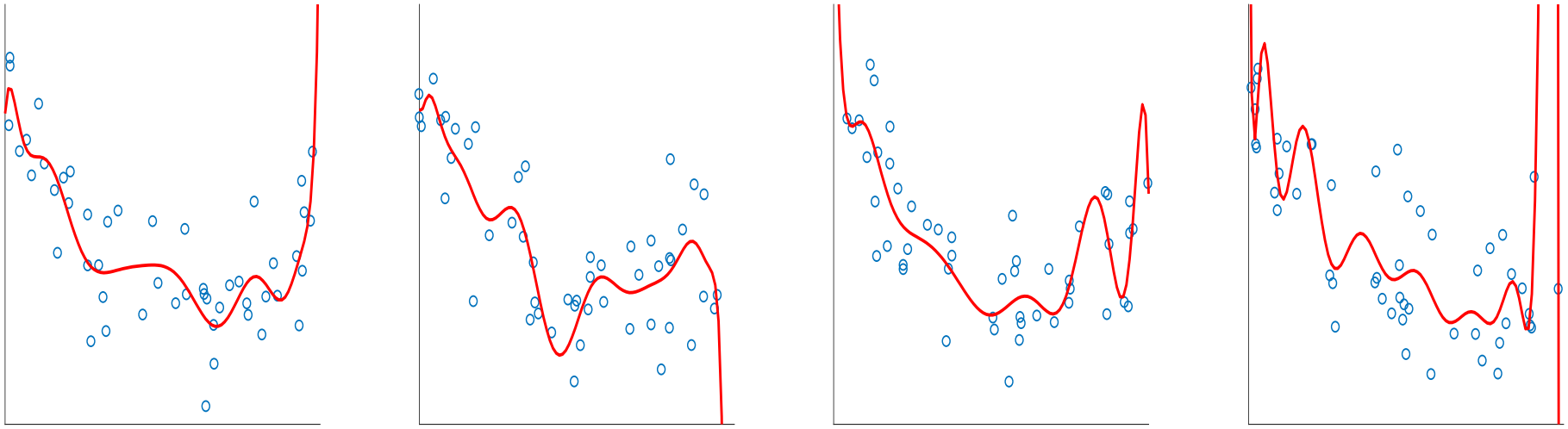
Bias



- Regardless of training sample, or size of training sample, model will produce consistent errors

Actual relationship may be quadratic

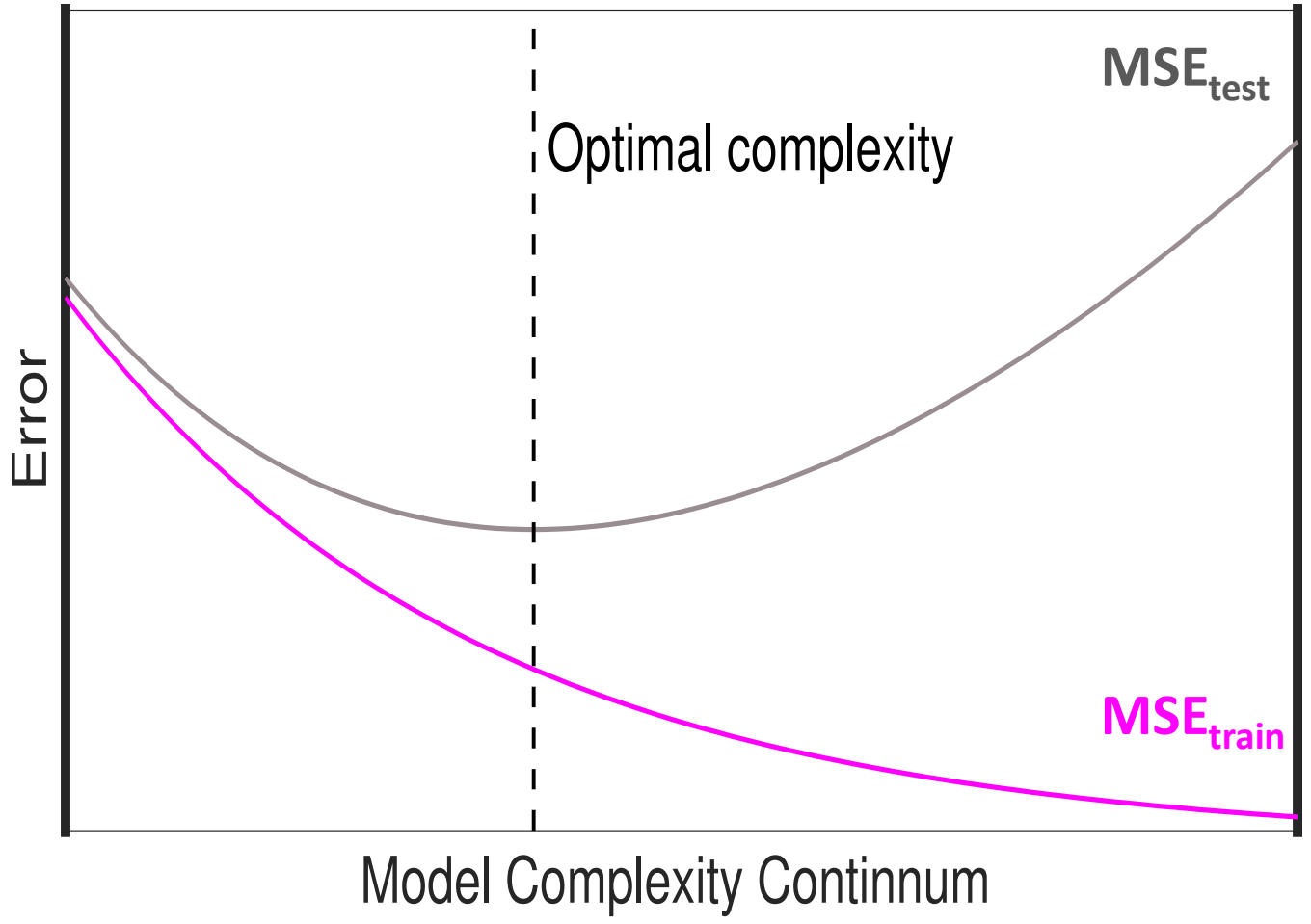
Variance



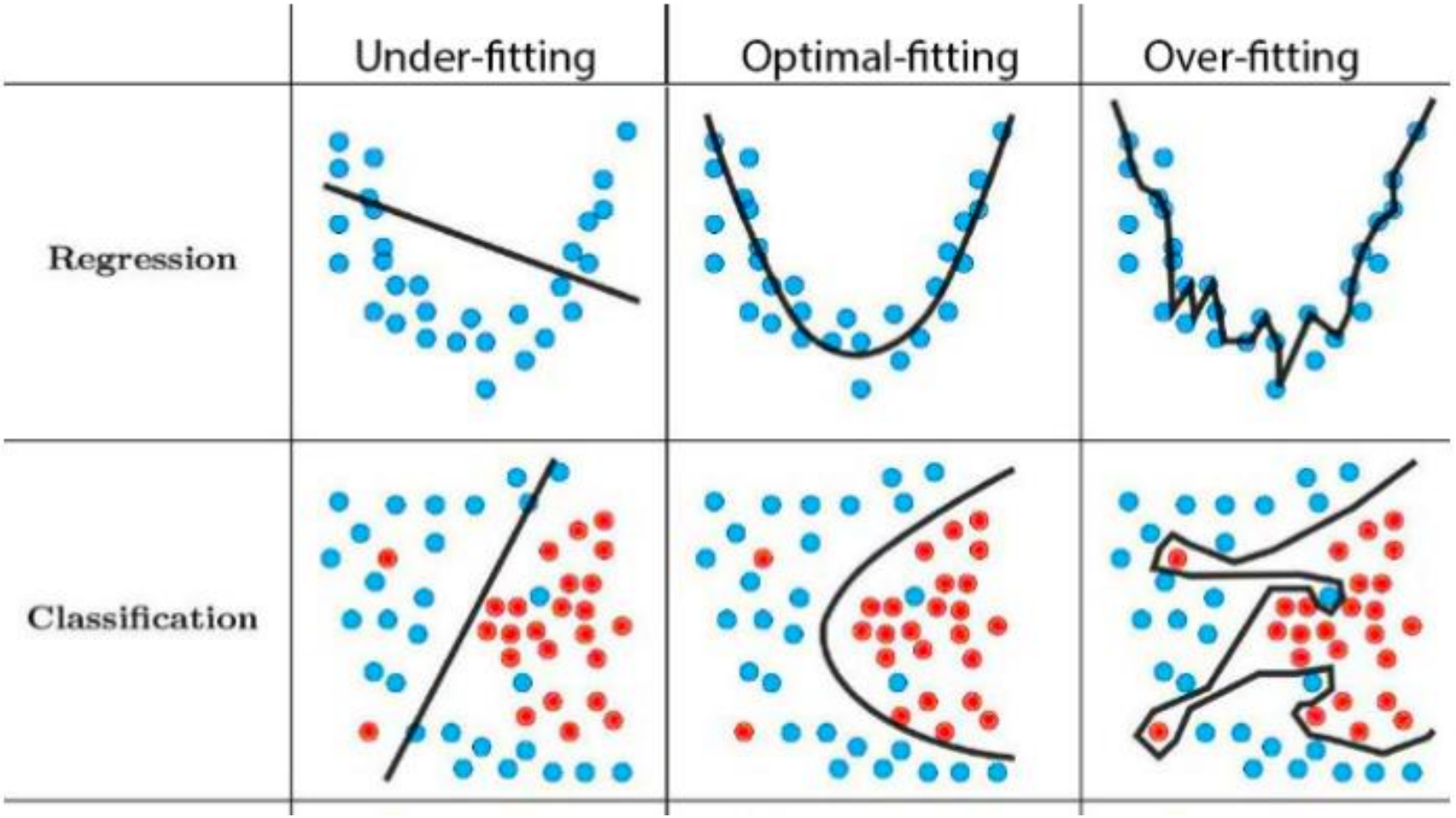
- Different samples of training data yield different model fits

We are trying to fit degree 8 polynomial

Bias-Variance Trade Off Is Revealed Via Test Set Not Training Set



Bias increases ⟸ ⟹ **Variance increases**

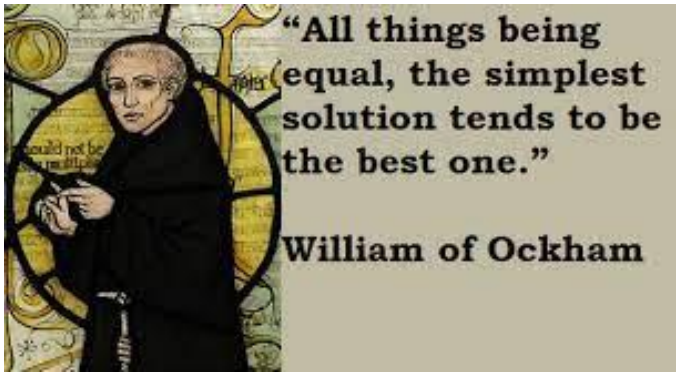


High bias ,
Low variance

Low bias ,
Low variance

Low bias ,
High variance

Regularization



- A controlled way of reducing the complexity of the fitted curve.
- We need to measure the complexity.
- We need to penalize the complex solutions.

Regularization: An Overview

- The idea of regularization revolves around modifying the criterion J; in particular, we add a regularization term that penalizes some specified properties of the model parameters
- $J_{reg}(\Theta) = J(\Theta) + \lambda R(\Theta)$
- where λ is a scalar that gives the weight (or importance) of the regularization term.
- Fitting the model using the modified loss function J_{reg} would result in model parameters with desirable properties (specified by *R*).

RIDGE and LASSO Regularizations

- In Ridge regularization,
- $J_{reg}(\Theta) = J(\Theta) + \lambda(\theta_1^2 + \cdots + \theta_d^2)$
- Note θ_0 is not used in the penalization

- In Lasso regularization,
- $J_{reg}(\Theta) = J(\Theta) + \lambda(|\theta_1| + \cdots + |\theta_d|)$
- Here also θ_0 is not penalized.

$J(\Theta)$: SSE, Ridge regression

- $J(\Theta) =$
  $\sum_{i=1}^{n} \left[\left(\sum_{j=1}^{d} x_{ij}\theta_j + \theta_0\right) - y_j\right]^2$ $J_{reg}(\Theta) =$
  $\sum_{i=1}^{n} \left[\left(\sum_{j=1}^{d} x_{ij}\theta_j + \theta_0\right) - y_j\right]^2 +$
  $\lambda\left(\sum_{j=1}^{d} \theta_j^2\right)$
- $\frac{\partial J_{reg}}{\partial \theta_j} =$
  $2\left[\sum_{i=1}^{n}\left[\left(\sum_{j=1}^{d} x_{ij}\theta_j + \theta_0\right) - y_j\right]x_{ij} + \lambda\theta_j\right]$

This is for j = 1 to d. For θ_0 there is no regularization penalty (see the next slide)

- $\frac{\partial J_{reg}}{\partial \theta_0} = 2\left[\sum_{i=1}^{n}\left[\left(\sum_{j=1}^{d} x_{ij}\theta_j + \theta_0\right) - y_j\right]\right]$

$J(\Theta)$: SSE, Lasso regression

- $J(\Theta) =$
  $\sum_{i=1}^{n} \left[\left(\sum_{j=1}^{d} x_{ij}\theta_j + \theta_0\right) - y_j\right]^2$ $J_{reg}(\Theta) =$
  $\sum_{i=1}^{n} \left[\left(\sum_{j=1}^{d} x_{ij}\theta_j + \theta_0\right) - y_j\right]^2 +$
  $\lambda\left(\sum_{j=1}^{d} |\theta_j|\right)$
- This is not differentiable, hence we cannot do gradient descent,
  - Quadratic programming (using Lagrange multipliers) can be employed.

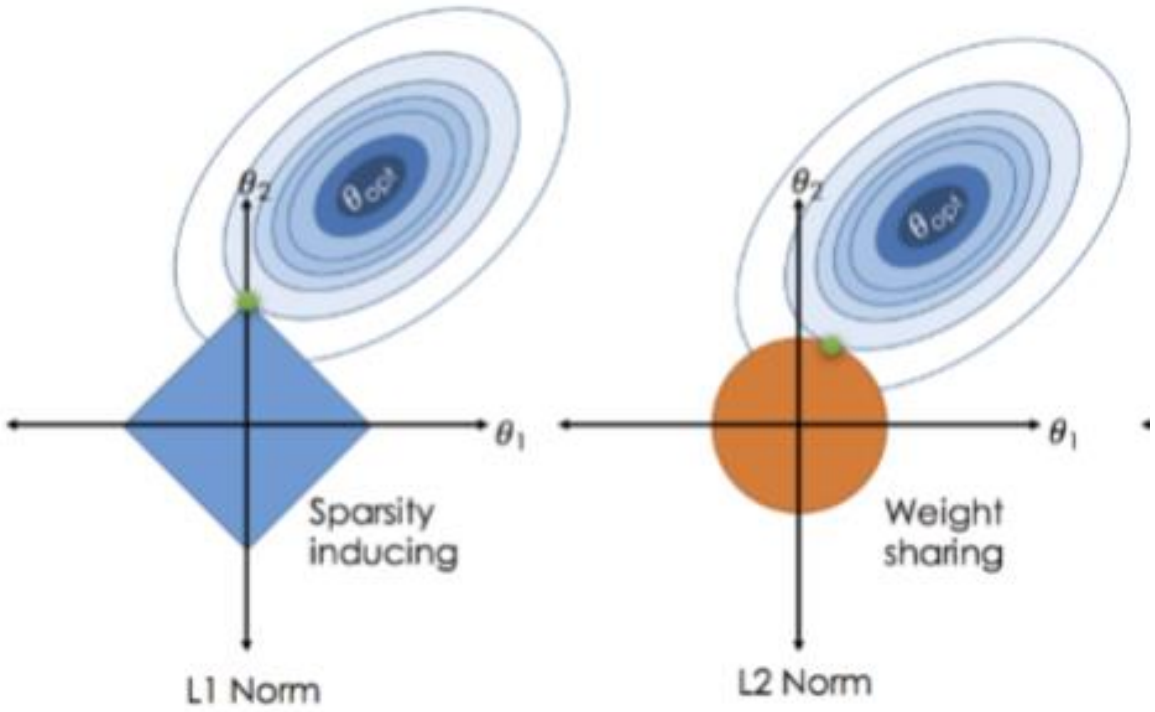
The gradient descent update rule is

$$\theta_j(k+1) = \theta_j(k) - \eta \frac{\partial J_{reg}}{\partial \theta_j}$$

Choosing λ

- In both ridge and LASSO regression, we see that the larger our choice of the **regularization parameter** λ, the more heavily we penalize large values in Θ,
- If λ is close to zero, we recover the SSE, i.e. ridge and LASSO regression is just ordinary regression.
- If λ is sufficiently large, the SSE term in the regularized loss function will be insignificant and the regularization term will force $\theta_1, \dots, \theta_d$ to be close to zero. {note, θ_0 can escape this}
- To avoid ad-hoc choices, we should select λ using cross-validation.

Geometric Interpretation



Lasso regularization

Ridge regularization