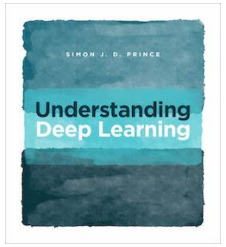
深度學習實作與應用
Deep learning and its applications

2. Basic Neural Networks:
From regressions to neural networks (1/3)

IM5062, Spring 2024

黃意婷



CH2

Outline

- **Supervised learning overview**
- **Regression**
  - Recall linear regression
  - Notation
  - 1D Linear regression example
  - Logistic Regression
  - Softmax Regression
- Summary

Supervised learning overview

- Supervised learning model = mapping from one or more inputs to one or more outputs
- Model is a mathematical equation
- Computing the inputs from the outputs = inference
- Example:
  - Input is age and milage of secondhand Toyota Prius
  - Output is estimated price of car

Supervised learning overview

- Supervised learning model = mapping from one or more inputs to one or more outputs
- Model is a mathematical equation

- Computing the inputs from the outputs = inference
- Model also includes parameters
- Parameters affect outcome of equation

Supervised learning overview

- Supervised learning model = mapping from one or more inputs to one or more outputs
- ~~Model is a mathematical equation~~

- Computing the inputs from the outputs = inference
- Model also includes parameters
- Parameters affect outcome of equation

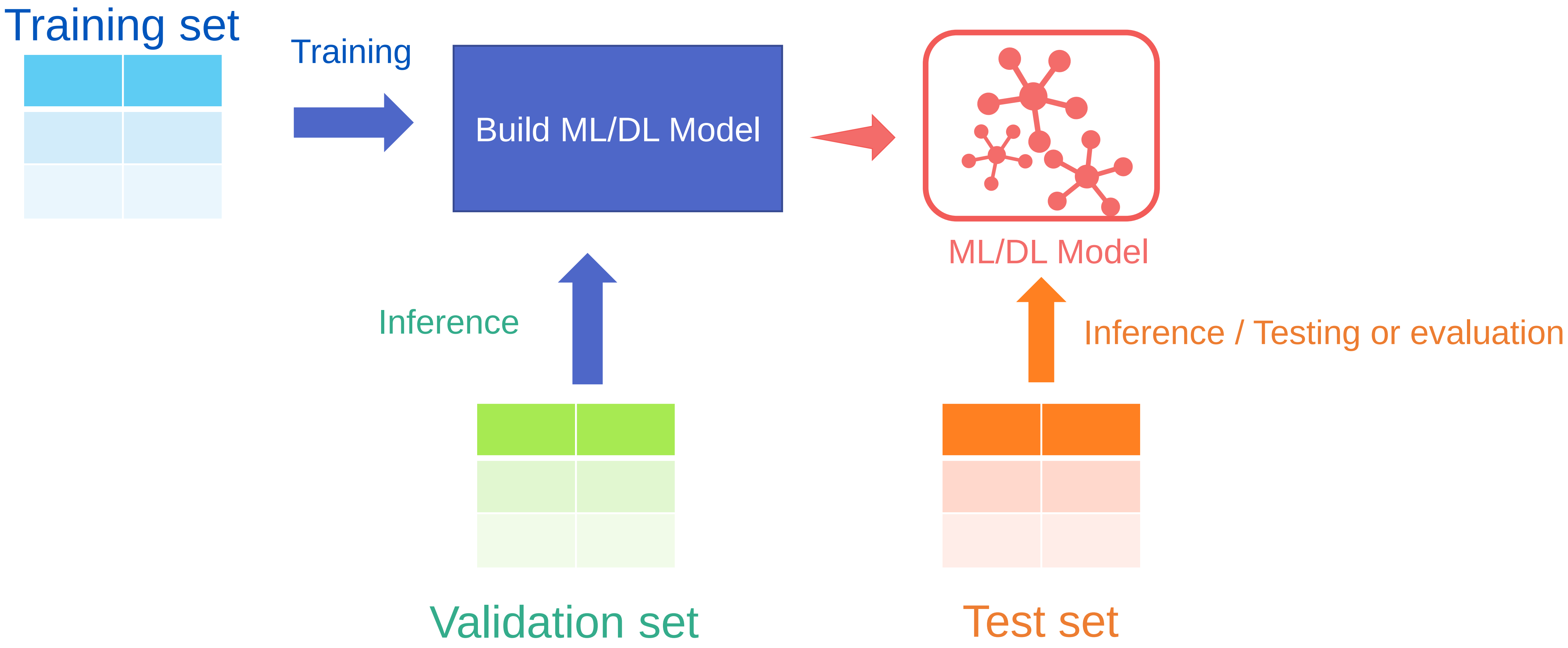
Supervised learning overview

- Supervised learning model = mapping from one or more inputs to one or more outputs
- ~~Model is a mathematical equation~~
- Model is a family of equations
  - The model equation describes a family of possible relationships between inputs and outputs
  - The parameters specify the particular relationship.

Supervised learning overview

- Training a model = finding parameters that predict outputs "well" from inputs for a training dataset of input/output pairs
- A learning algorithm takes a training set of input/output pairs and manipulates the parameters until the inputs predict their corresponding outputs as closely as possible.
- If the model works well for these training pairs, then we hope it will make good predictions for new inputs where the true output is unknown.

Learning process



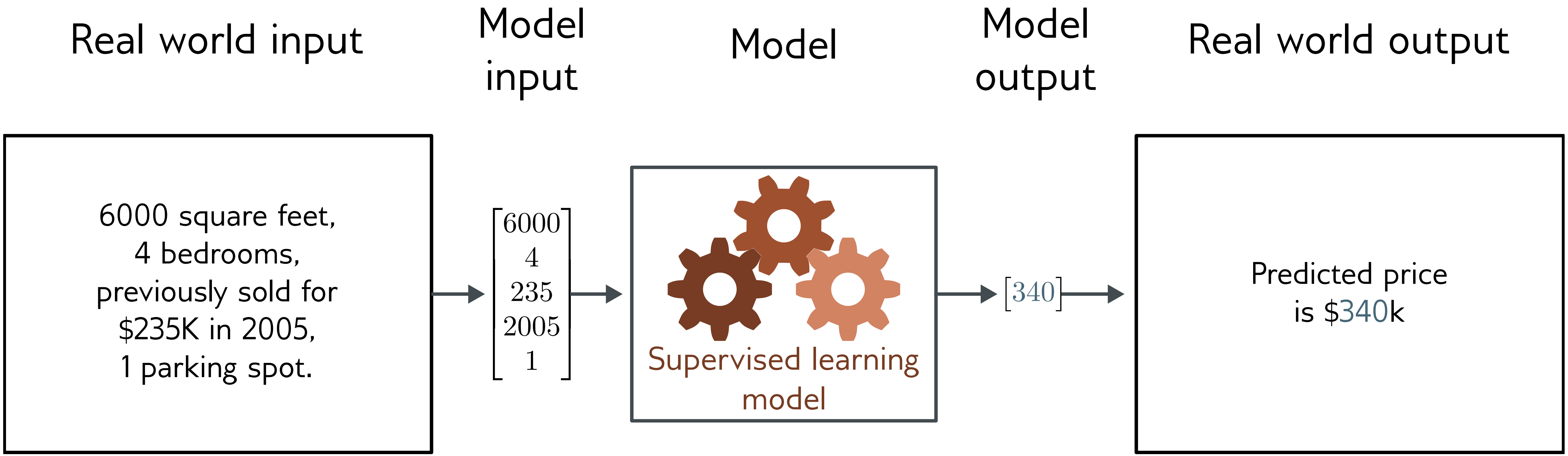
Dataset

- **Training set** is a set of examples used to fit the parameters (e.g. weights of connections between neurons) of the model.
- **Validation set** provides an unbiased evaluation of a model fit on the training data set while tuning the model's hyperparameters (e.g. the number of hidden units).
- **Test set** is a data set used to provide an unbiased evaluation of a final model fit on the training data set.
  - A test data set is a data set that is **independent** of the training data set, but that follows **the same probability distribution** as the training data set.

Outline

- **Supervised learning overview**
- **Regression**
  - Recall linear regression
  - Notation
  - 1D Linear regression example
  - Logistic Regression
  - Softmax Regression
- Summary

Regression



- Univariate regression problem (one output, real value)

House Buying

- Pick a house, take a tour.
- Estimate its price

Source: 信義房屋

BMI
Body Mass Index

u BMI值計算

BMI值計算公式: BMI = 體重(公斤) / 身高^2(公尺^2)

例如：一個52公斤的人，身高是155公分，則BMI為：

52(公斤)/1.55^2 (公尺^2)= 21.6

體重正常範圍為 BMI=18.5～24

快看看自己的BMI是否在理想範圍吧!

身　高: cm 開始計算

體　重: kg 清除重算

你的BMI為

	身體質量指數(BMI) (kg/m2)	腰圍 (cm)
體重過輕	BMI ＜ 18.5	-
正常範圍	18.5≦BMI＜24	-
異常範圍	過重：24≦BMI＜27 輕度肥胖：27≦BMI＜30 中度肥胖：30≦BMI＜35 重度肥胖：BMI≧35	男性：≧90公分 女性：≧80公分

Source: 亞東醫院

BMI
Body Mass Index

x_i	$x_{i,1}$	$x_{i,2}$	$x_{i,k}$	y_i
Subject ID	**Meal frequency**	**male**	**Age > 17**	**BMI**
A	4	0	1	27
B	7	1	1	29
C	6	1	0	23
D	2	0	0	20
E	3	0	1	21
etc	…	…	…	…


BMI值計算

BMI值計算公式: BMI = 體重(公斤) / 身高^2(公尺^2)

例如：一個52公斤的人，身高是155公分，則BMI為：

52(公斤)/1.55^2 (公尺^2)= 21.6

體重正常範圍為 BMI=18.5～24

快看看自己的BMI是否在理想範圍吧!

身　高: cm 開始計算

體　重: kg 清除重算

你的BMI為

	身體質量指數(BMI) (kg/m2)	腰圍 (cm)
體重過輕	BMI ＜ 18.5	-
正常範圍	18.5≦BMI＜24	-
異常範圍	過重：24≦BMI＜27 輕度肥胖：27≦BMI＜30 中度肥胖：30≦BMI＜35 重度肥胖：BMI≧35	男性：≥90公分 女性：≥80公分

Source: 亞東醫院

Linear Regression

- Given a series of input/output pairs: $(x_i \ , y_i \ )$
  - (house 1, price 1), (house 2, price 2) …
  - (subject 1, BMI 1), (subject 2, BMI 2) …

- For each observation
  - We represent x_i by a feature vector $[x_{i,1}, x_{i,2}, \dots, x_{i,n}]$
    - house 1, [0年, 14.45坪, 11樓, …]
    - House 2, [43.1年, 31.81坪, 5樓, …]
  - We compute an output: a continuous value $\hat{y}_i$
    - $\widehat{y_1}$: Price 1, y_1 : 2,095萬
    - $\hat{y}_2$: Price 2, y_2 : 2,836萬

Features in linear regression

- x_j = meal frequency: w_j = 1.5
- x_k = if male: w_k = 1.6
- x_l = if age>17: w_l = 4.2
- b = 18.0

Subject ID	Meal frequency	male	Age > 17	BMI
A	4	0	1	27
B	7	1	1	29
C	6	1	0	23
D	2	0	0	20
E	3	0	1	21
etc	…	…	…	…

- For feature x_j, weight w_j tells is how important is x_j

Linear Regression for one observation x

- Input observation: vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$
- Weights: one per feature: $\boldsymbol{w} = [w_1, w_2, \ldots, w_n]$
  - Sometimes we call the weights $\theta = [\theta_1, \theta_2, \ldots, \theta_n]$
  - bias = *b or* θ_0

- We'll sum up all the weighted features and the bias

$$\hat{y} = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

$$\hat{y} = \boldsymbol{w} \cdot \boldsymbol{x} + b$$

- Output: a continuous value $\hat{y}$

BMI example

Subject ID	Meal frequency	male	Age > 17	BMI
A	4	0	1	27
B	7	1	1	29
C	6	1	0	23
D	2	0	0	20
E	3	0	1	21
F	2	1	1	?

- Suppose w = [1.5, 1.6, 4.2], b = 18.0
- Predict BMI for an new and unseen input x_F = [2, 1, 1]
- BMI = bias + $w_1 \times x_1 + w_2 \times x_2 + w_3 \times x_3$

  = 18.0 + 1.5 (meal freq.) + 1.6 (male) + 4.2 (age>17)

  = $18.0 + 1.5 \times 2 + 1.6 \times 1 + 4.2 \times 1$

  = 26.8

Notation:

- Input:

$$\mathbf{x}$$

Normal = scalar
Bold = vector
Capital Bold = matrix

- Output:

$$\mathbf{y}$$

- Model: $\mathbf{y} = \mathbf{f}[\mathbf{x}]$

Functions always square brackets

Normal = returns scalar
Bold = returns vector
Capital Bold = returns matrix

Notation example:

- Input:

$$\mathbf{x} = \begin{bmatrix} \text{age} \\ \text{mileage} \end{bmatrix}$$

- Output:

$$y = \begin{bmatrix} \text{price} \end{bmatrix}$$

- Model:

$$y = \mathrm{f}[\mathbf{x}]$$

Structured or tabular data

Subject ID	Meal frequency	male	Age > 17	BMI
A	4	0	1	27
B	7	1	1	29
C	6	1	0	23
D	2	0	0	20
E	3	0	1	21
etc	…	…	…	…

Notation:

- Input:

$$\mathbf{x}$$

- Output:

$$\mathbf{y} \qquad \hat{y}$$

predicted value

- Model: $\mathbf{y} = \mathbf{f}[\mathbf{x}]$

BMI example

Subject ID	Meal frequency	male	Age > 17	BMI
A	4	0	1	27
B	7	1	1	29
C	6	1	0	23
D	2	0	0	20
E	3	0	1	21
F	2	1	1	?

- Suppose w = [1.5, 1.6, 4.2], b = 18.0
- Predict BMI for an new and unseen input x_F = [2, 1, 1]
- BMI = bias + $w_1 \times x_1 + w_2 \times x_2 + w_3 \times x_3$
  = 18.0 + 1.5 (meal freq.) + 1.6 (male) + 4.2 (age>17)
  = 18.0 + 1.5 × 2 + 1.6 × 1 + 4.2 × 1
  = 26.8

Model

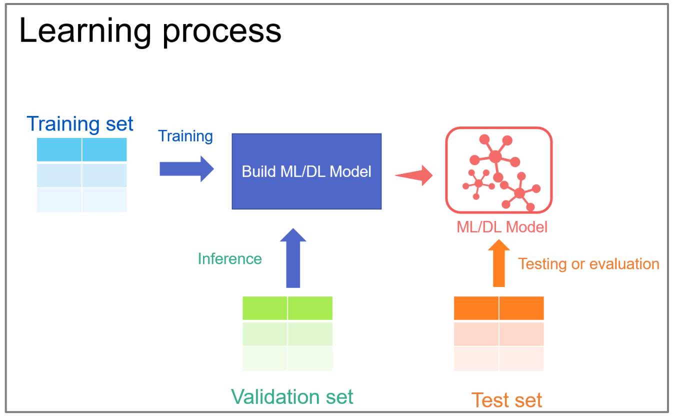
- Parameters:

$$\phi \qquad \theta \qquad w$$

- Model :

$$\mathbf{y} = \mathbf{f}[\mathbf{x}, \phi]$$

Loss function



- Training dataset of I pairs of input/output examples:

$$\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}$$

- Loss function or cost function measures how bad model is:
  - To quantify the degree of mismatch in this mapping with the loss L

$$L\left[\underbrace{\phi, \mathrm{f}[\mathbf{x}, \phi]}_{\text{model}}, \underbrace{\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}}_{\text{train data}}\right]$$

or for short:

$$L[\phi] \longleftarrow$$

Returns a scalar that is smaller when model maps inputs to outputs better

Training

- Loss function:

$$L\left[\phi\right]$$

Returns a scalar that is smaller when model maps inputs to outputs better

- Find the parameters that **minimize** the loss:

$$\hat{\phi} = \underset{\phi}{\operatorname{argmin}}\Big[\mathrm{L}\left[\phi\right]\Big]$$

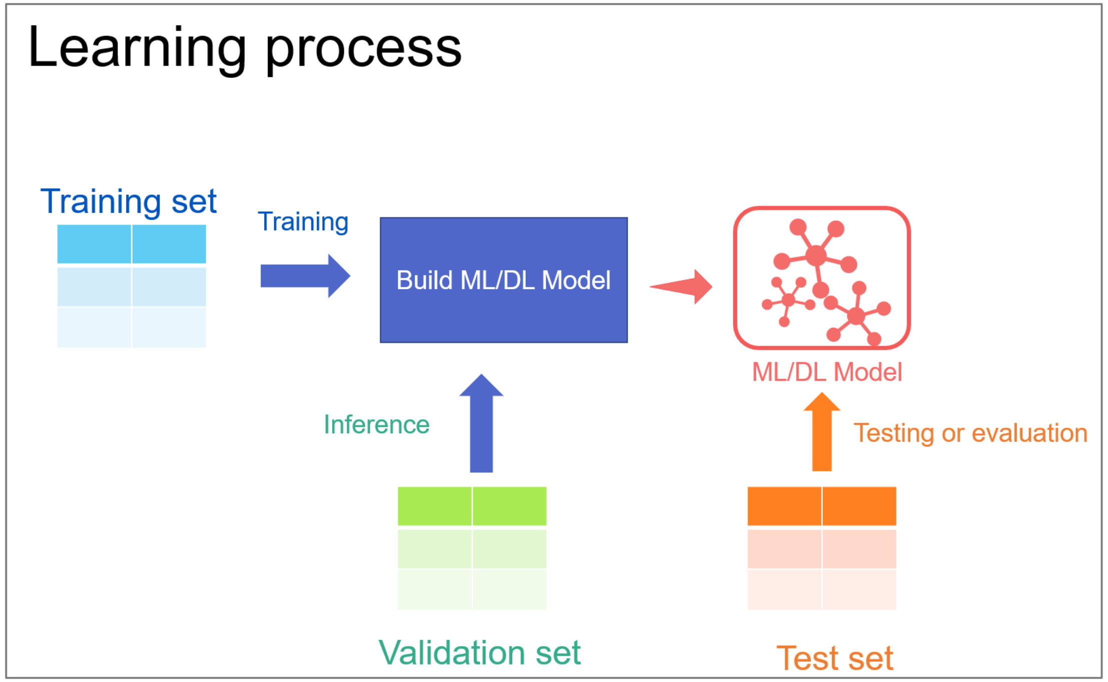
BMI example

Subject ID	Meal frequency	male	Age > 17	BMI
A	4	0	1	27
B	7	1	1	29
C	6	1	0	23
D	2	0	0	20
E	3	0	1	21
F	2	1	1	?

- Suppose w = [1.5, 1.6, 4.2], b = 18.0
- Predict BMI for an new and unseen input x_F = [2, 1, 1]
- BMI = bias + $w_1 \times x_1 + w_2 \times x_2 + w_3 \times x_3$
  = 18.0 + 1.5 (meal freq.) + 1.6 (male) + 4.2 (age>17)
  = 18.0 + 1.5 × 2 + 1.6 × 1 + 4.2 × 1
  = 26.8

Testing

- To test the model, run on a separate test dataset of input / output pairs
- See how well it generalizes to new data



Outline

- **Supervised learning overview**
- **Regression**
  - Recall linear regression
  - Notation
  - 1D Linear regression example
  - Logistic Regression
  - Softmax Regression
- Summary

Example: 1D Linear regression model

- Model:

$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 x
\end{aligned}
$$

- Parameters

$$
\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix}
\begin{array}{l} \longleftarrow \text{y-offset; y-intercept} \\ \longleftarrow \text{slope} \end{array}
$$

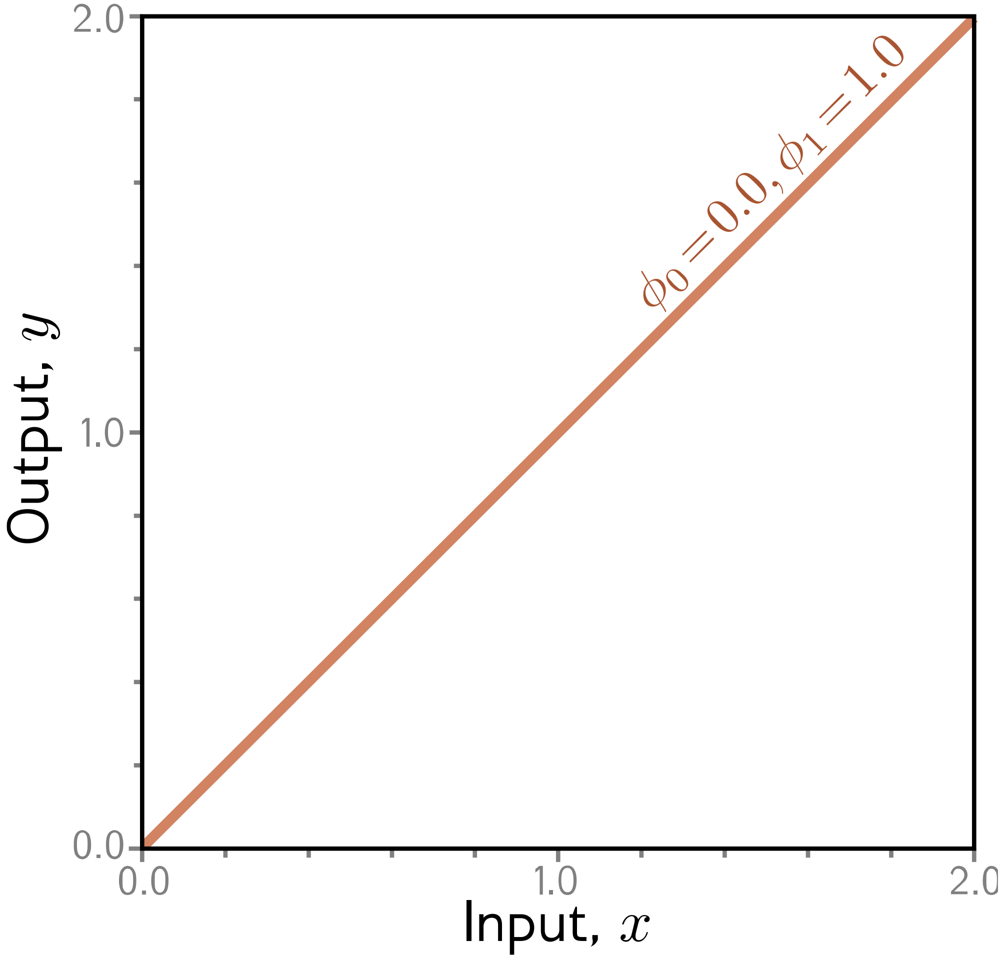
Example: 1D Linear regression model

- Model:

$$\begin{aligned} y &= \mathrm{f}[x, \boldsymbol{\phi}] \\ &= \phi_0 + \phi_1 x \end{aligned}$$

- Parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix} \begin{array}{l} \longleftarrow \text{y-offset} \\ \longleftarrow \text{slope} \end{array}$$



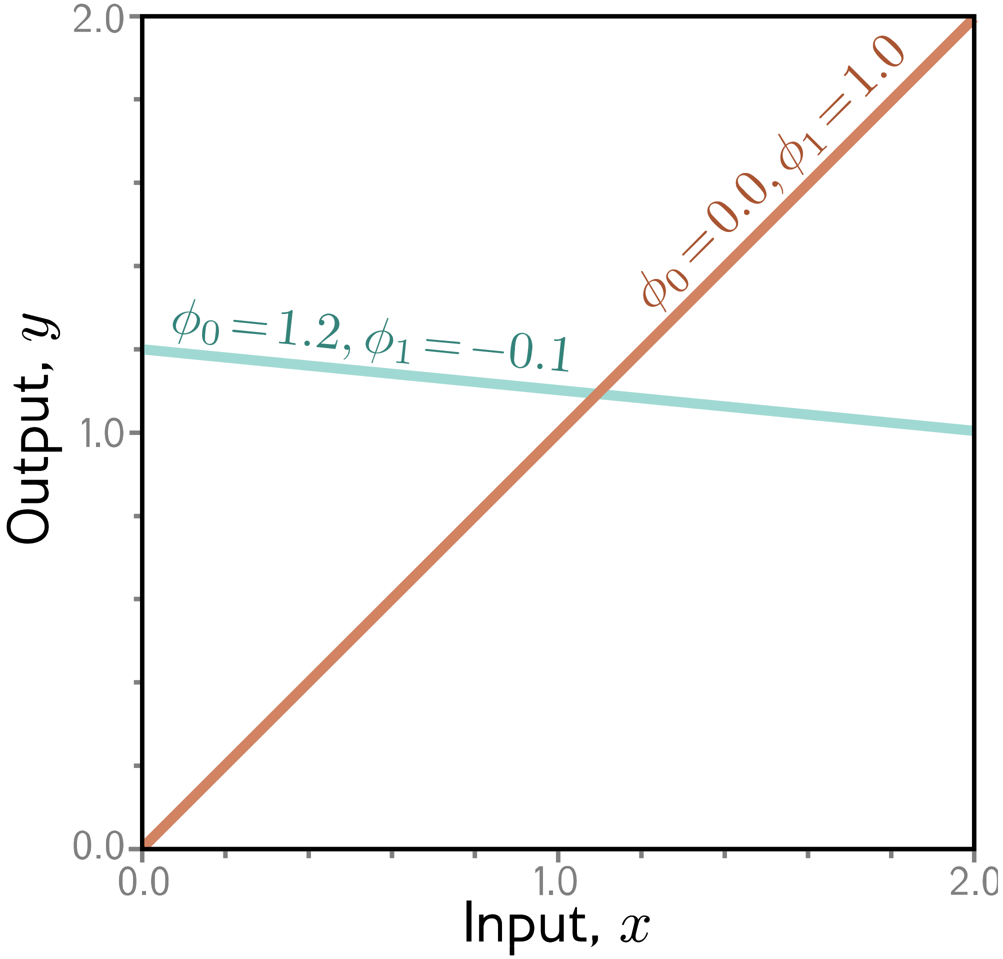
Example: 1D Linear regression model

- Model:

$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

- Parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix} \begin{array}{l} \longleftarrow \text{y-offset} \\ \longleftarrow \text{slope} \end{array}$$



Example: 1D Linear regression model

- Model:

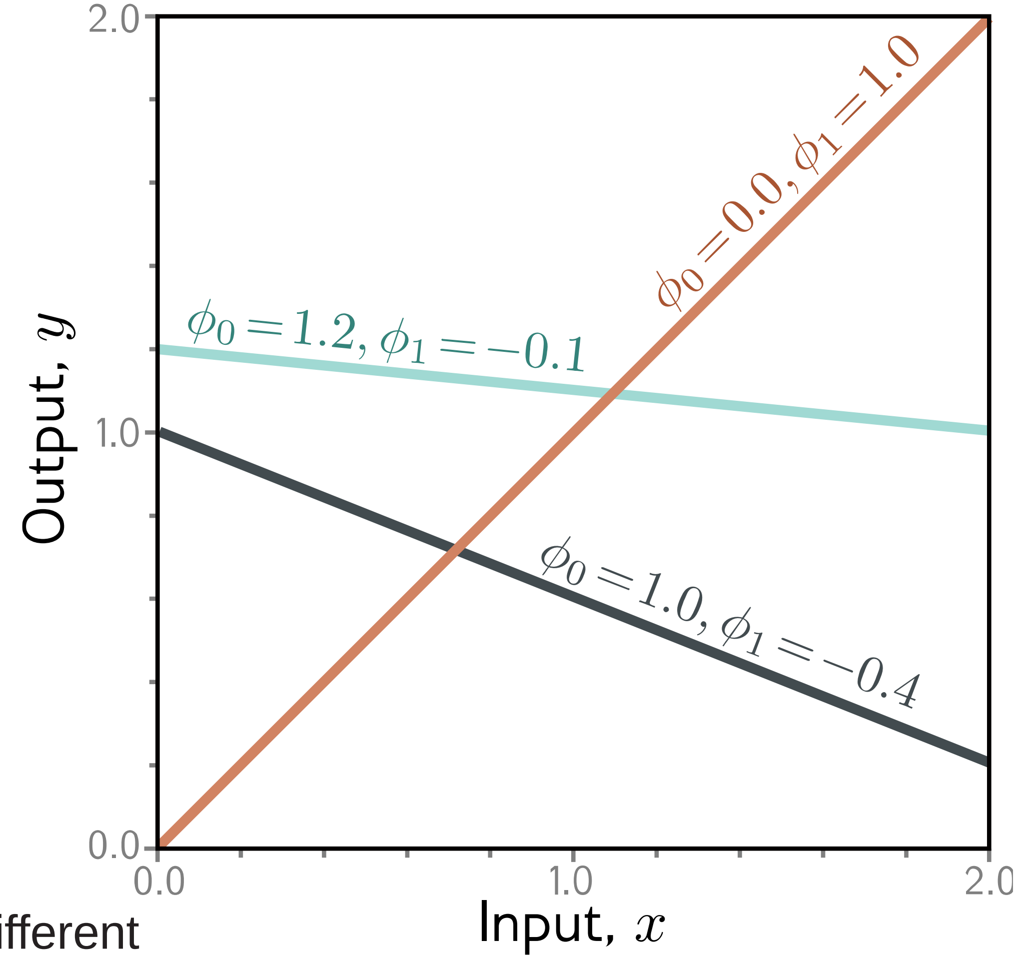
$$y = \mathrm{f}[x, \boldsymbol{\phi}]$$
$$= \phi_0 + \phi_1 x$$

- Parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \end{bmatrix}$$

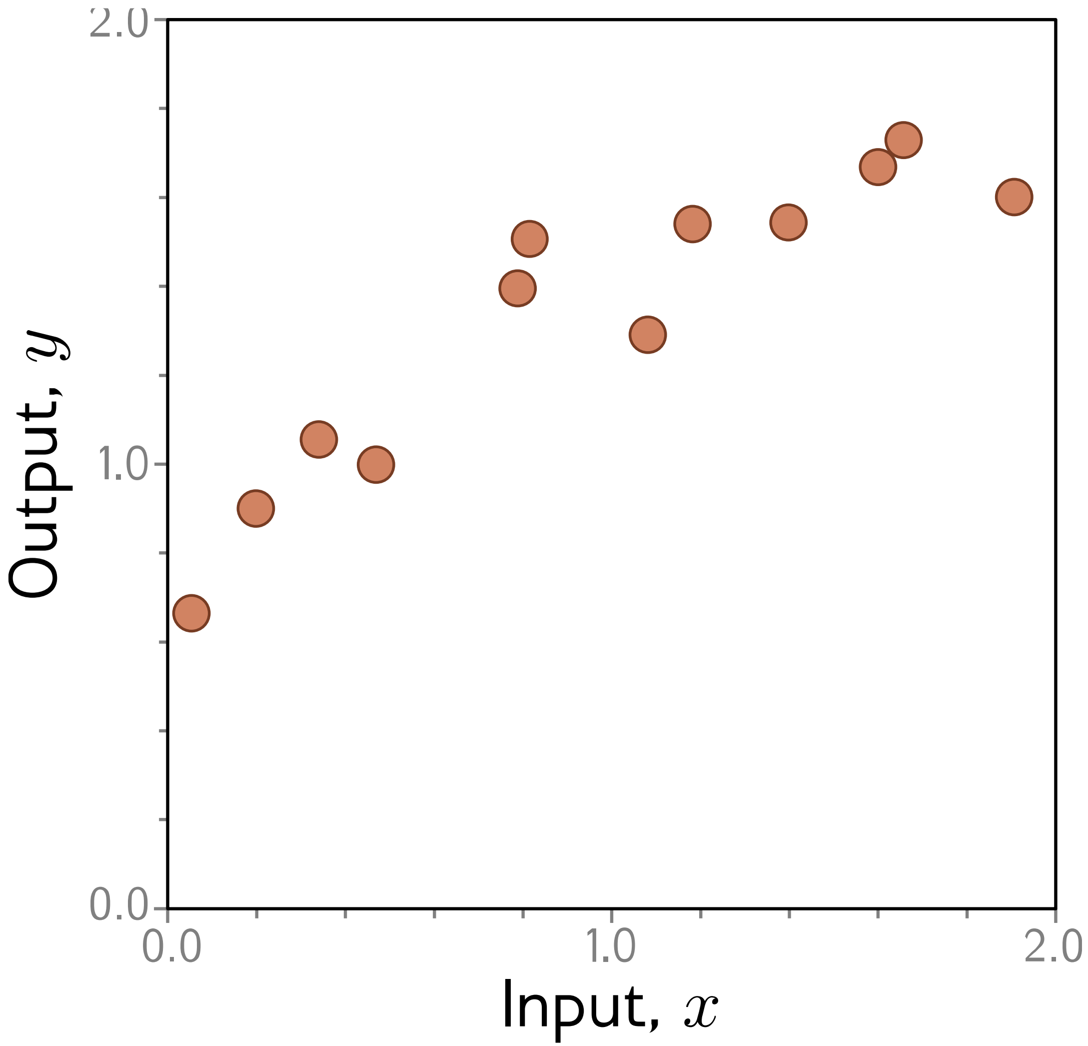
← y-offset

← slope



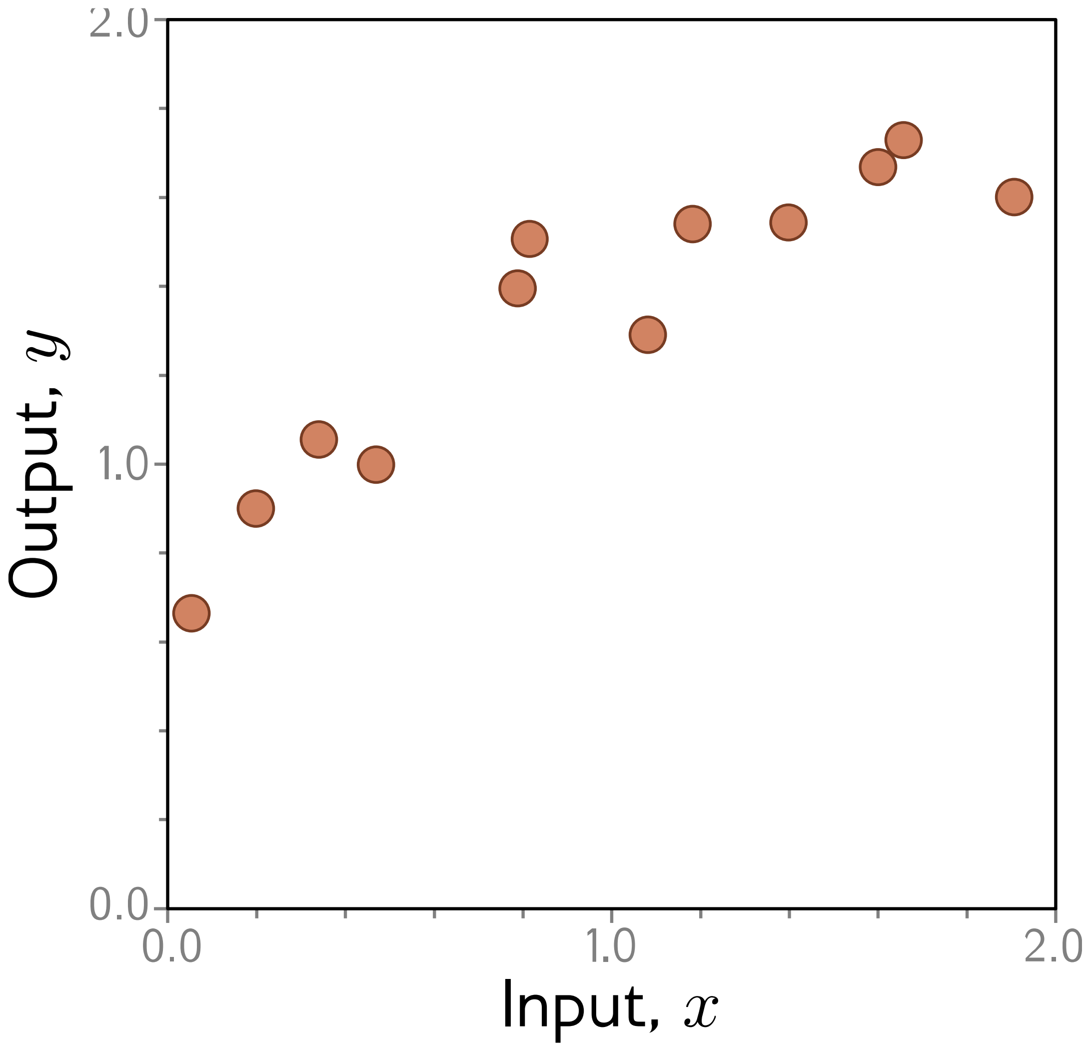
Different choices for the y-intercept and slope result in different relations between input and output

Example: 1D Linear regression training data



The training data (orange points) consist of $I = 12$ input/output pairs $\{x_i, y_i\}$

Example: 1D Linear regression training data



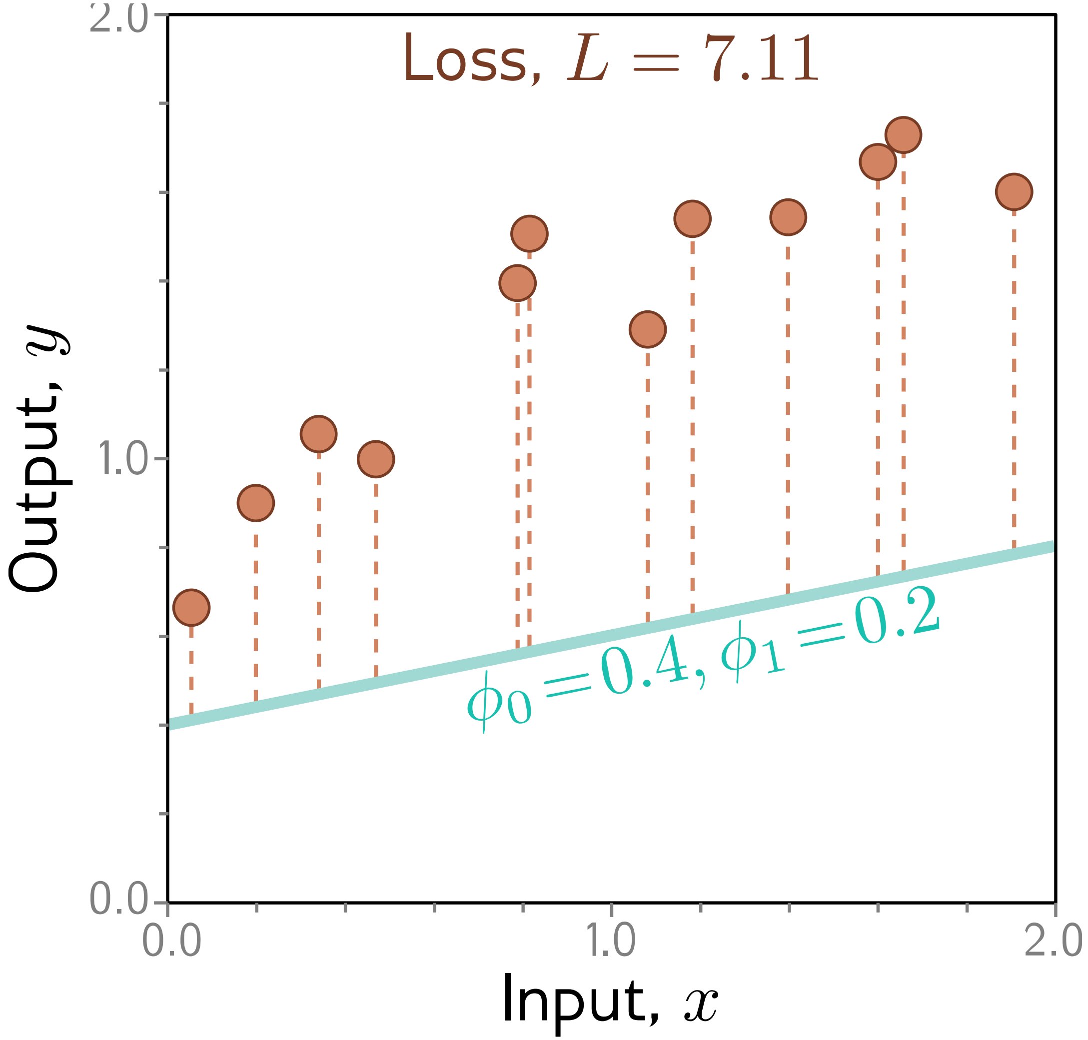
Loss function:

$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\underbrace{\mathrm{f}[x_i, \boldsymbol{\phi}]}_{\hat{y}_i} - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression loss function



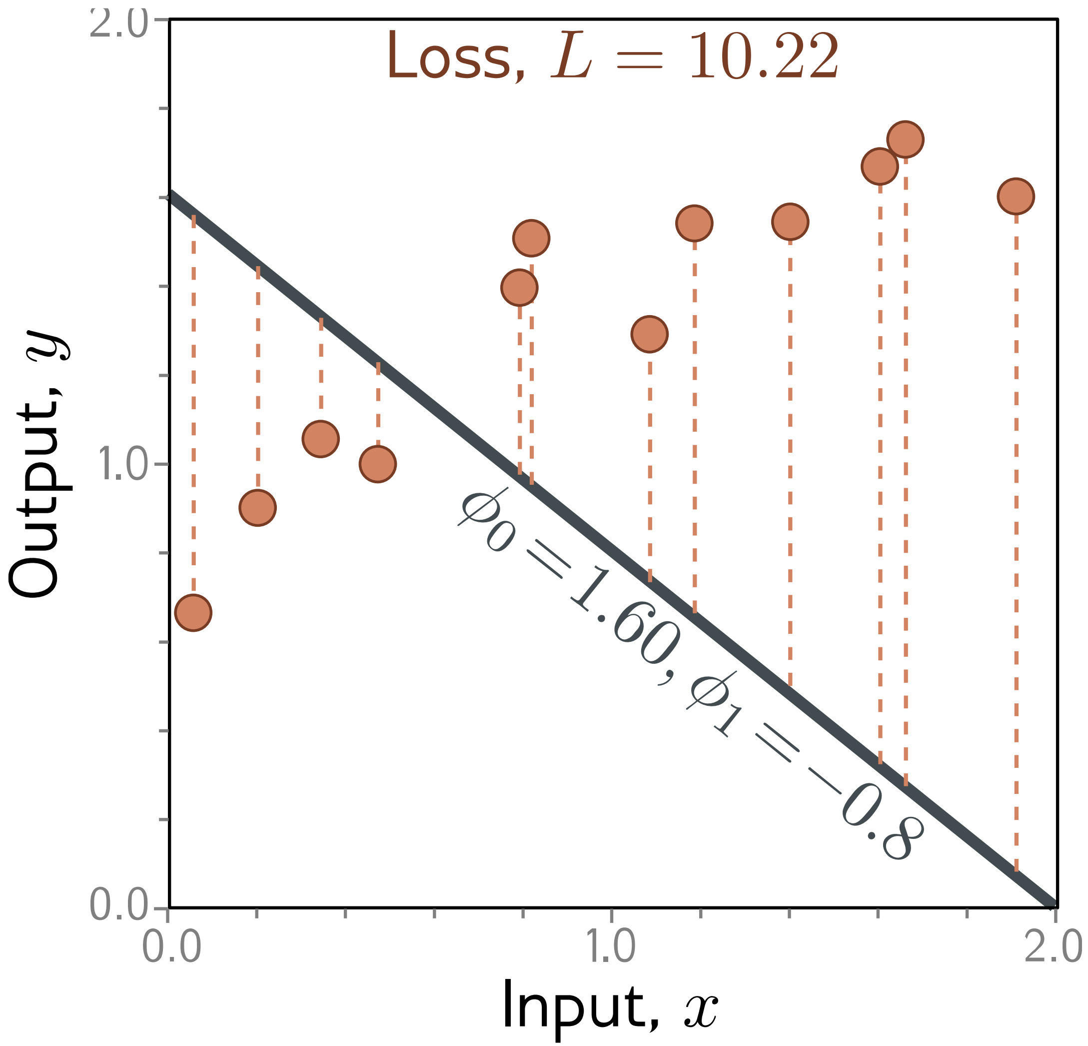
Loss function:

$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression loss function



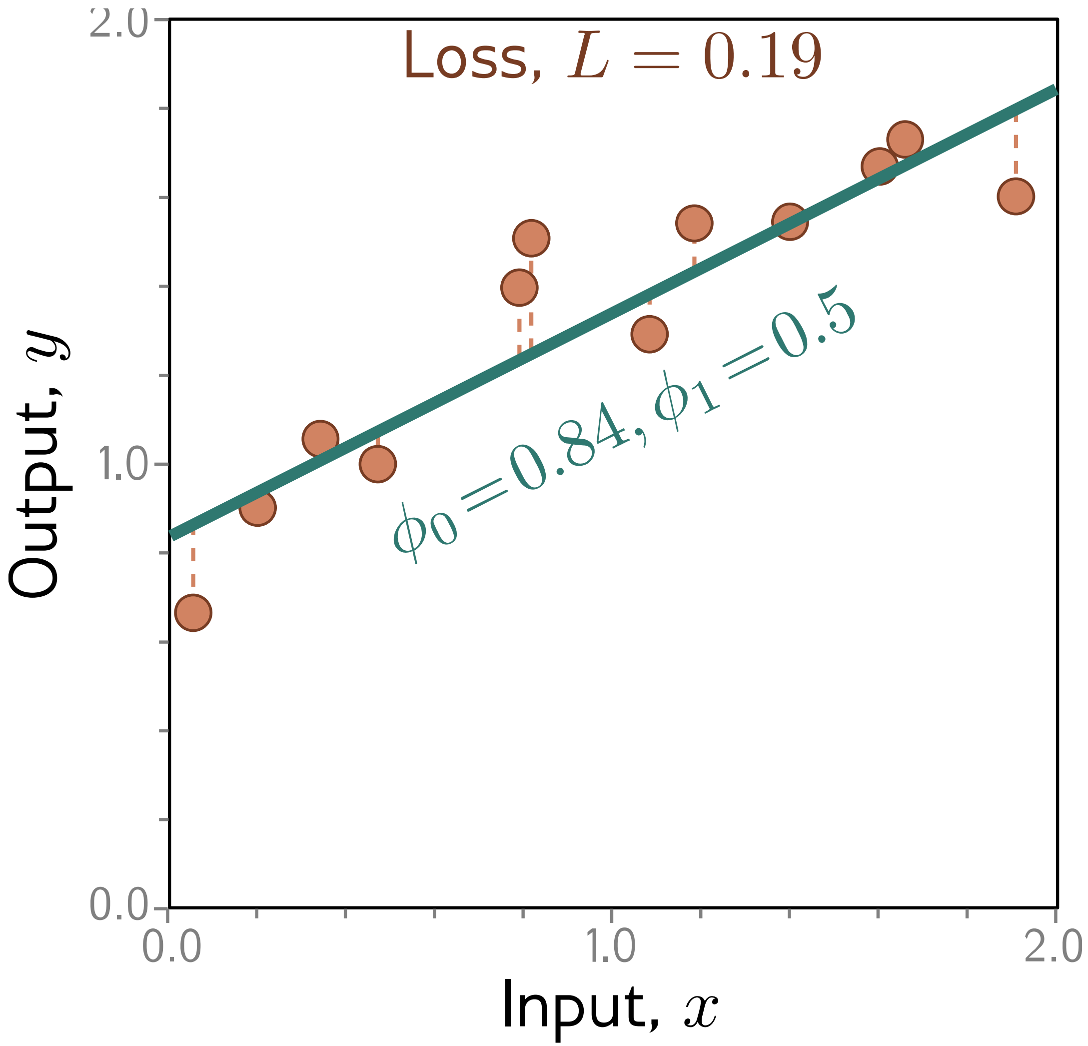
Loss function:

$$L[\phi] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \phi] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression loss function



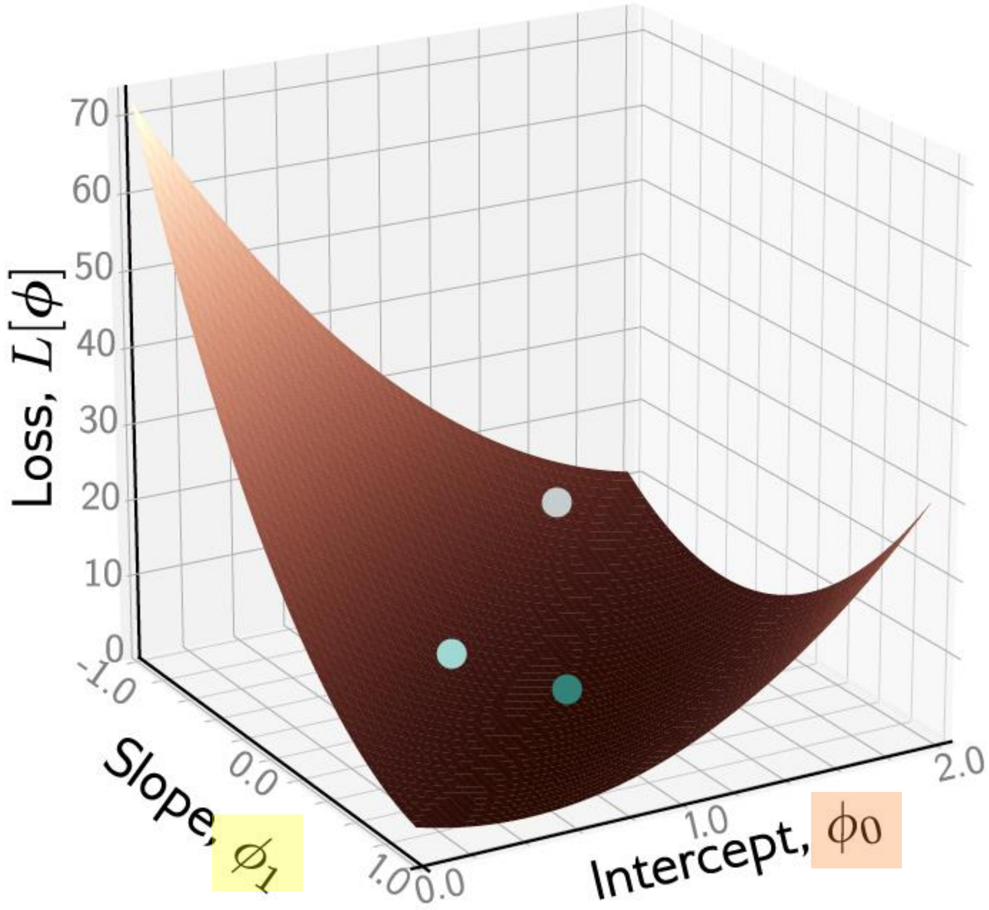
Loss function:

$$L[\phi] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \phi] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

“Least squares loss function”

Example: 1D Linear regression loss function

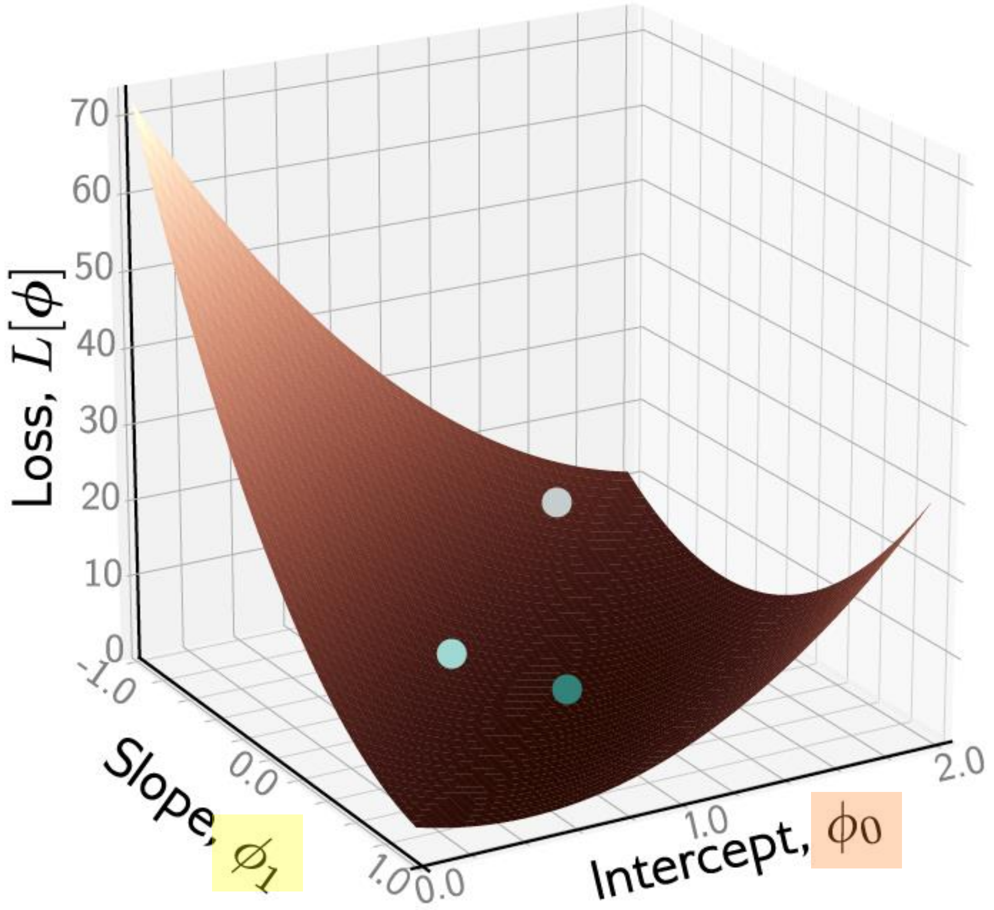


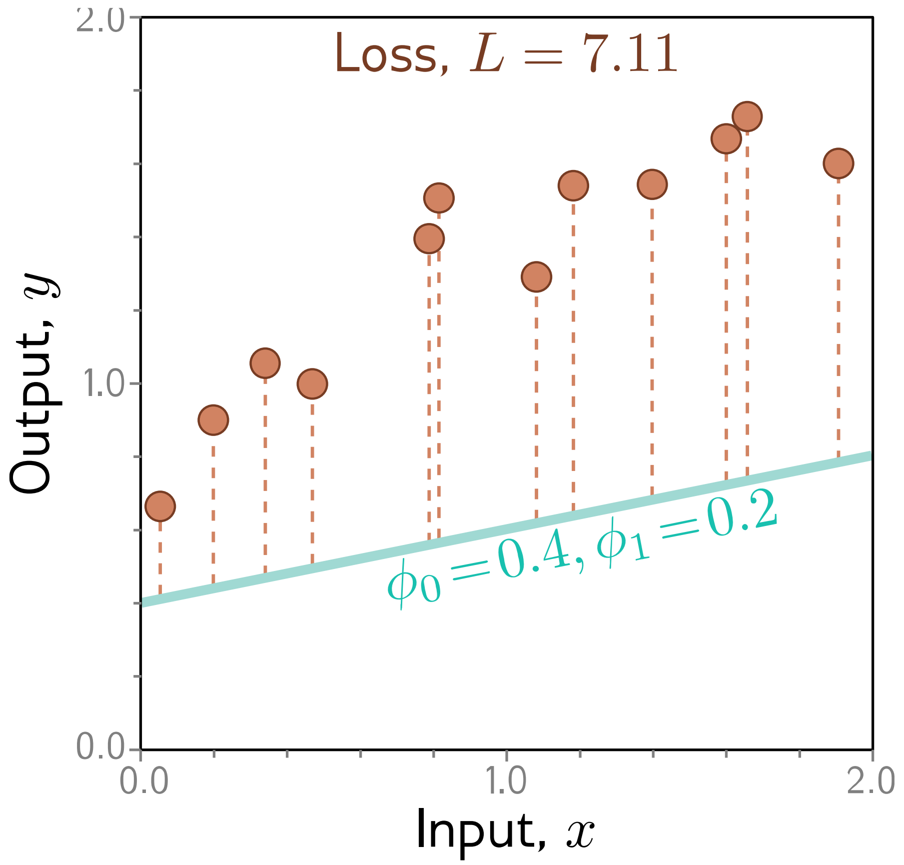
We can calculate the loss for every combination of values and visualize the loss function as a surface.

$$L[\phi] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \phi] - y_i)^2$$
$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

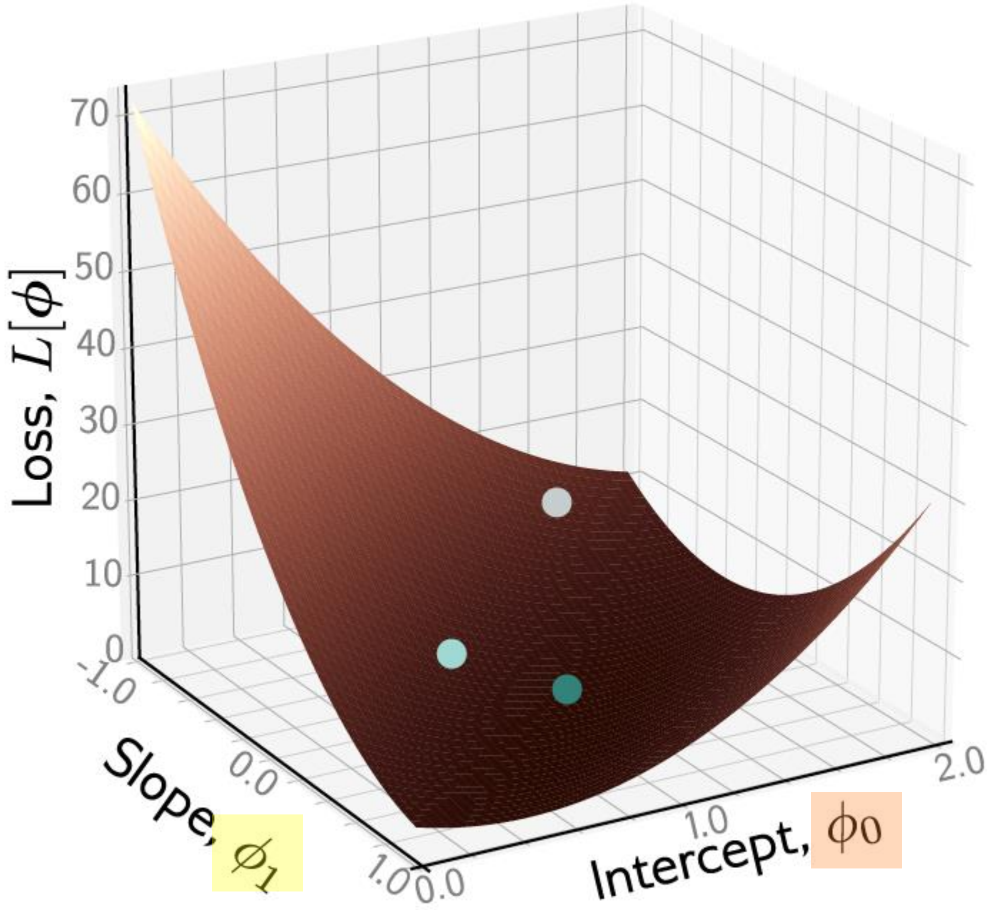
Example: 1D Linear regression loss function

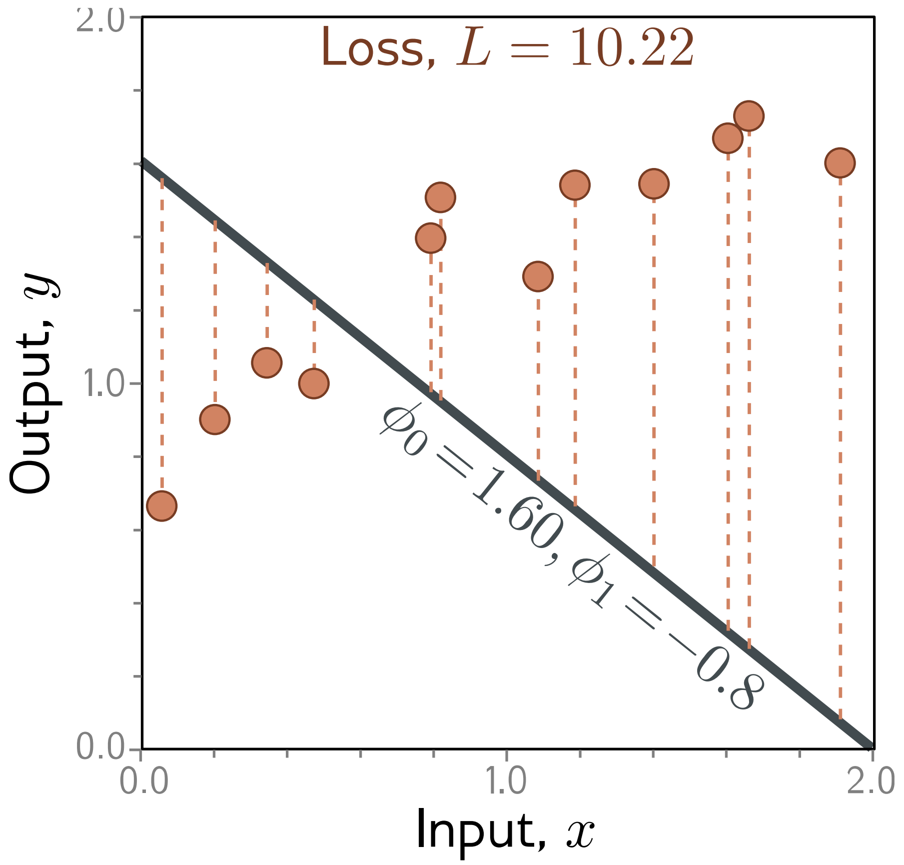




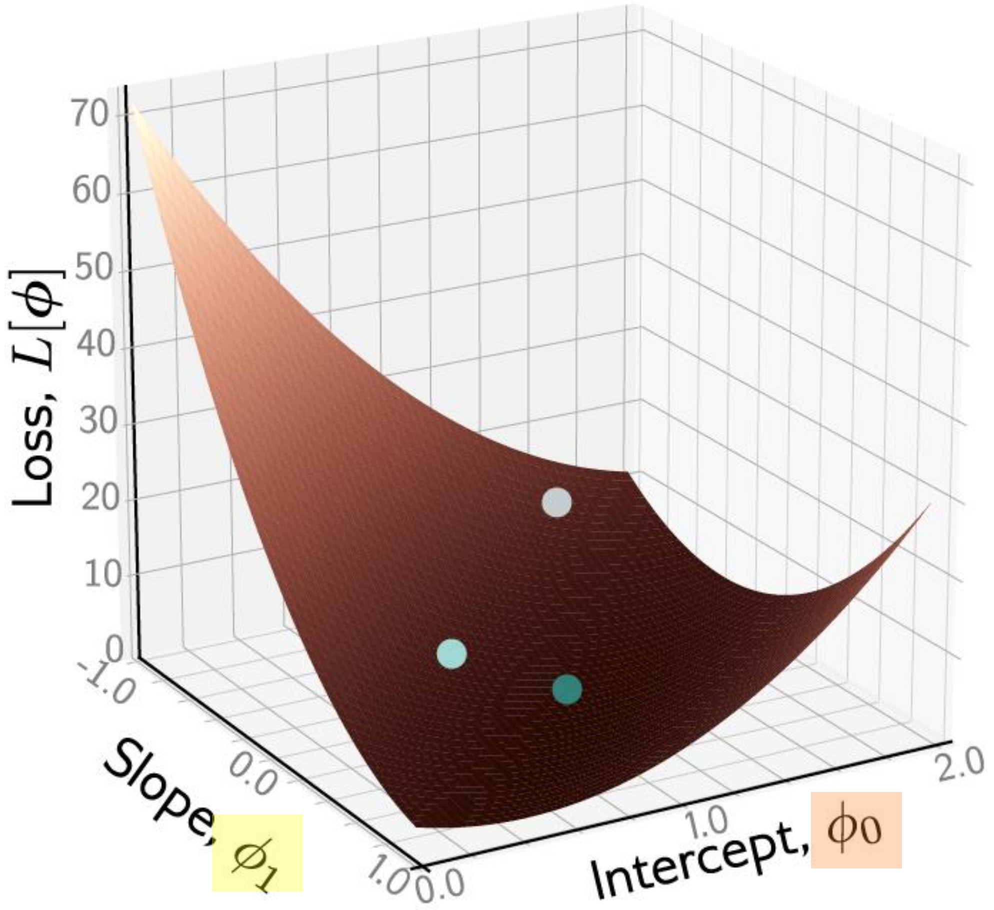
Source: Prof. Simon Prince

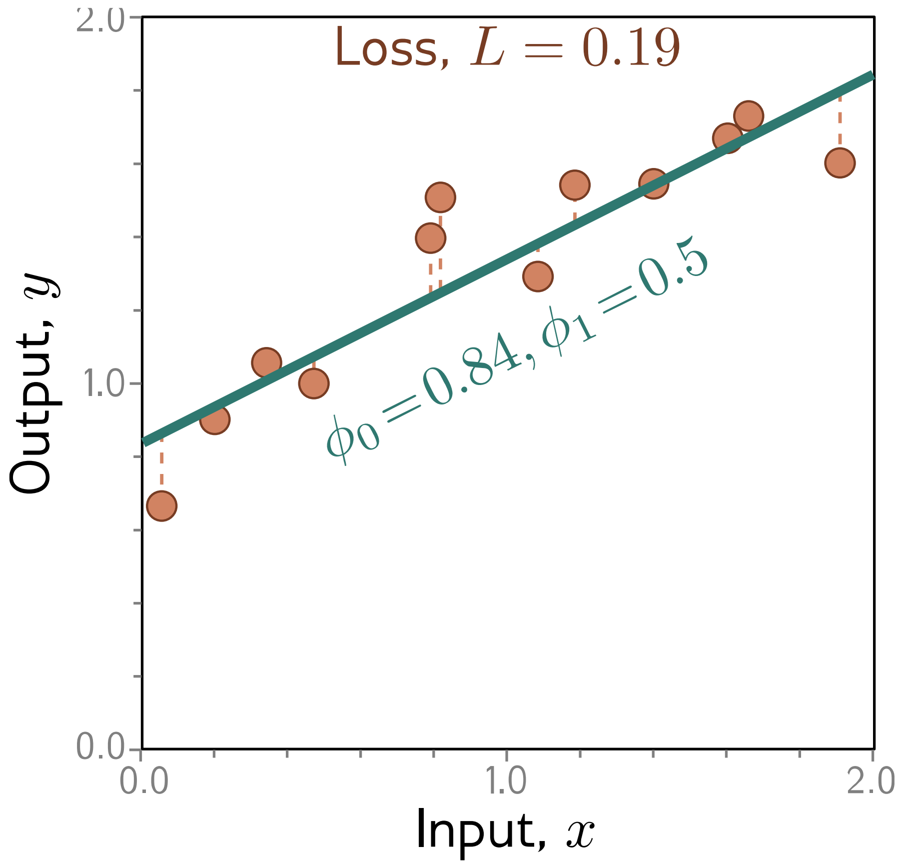
Example: 1D Linear regression loss function



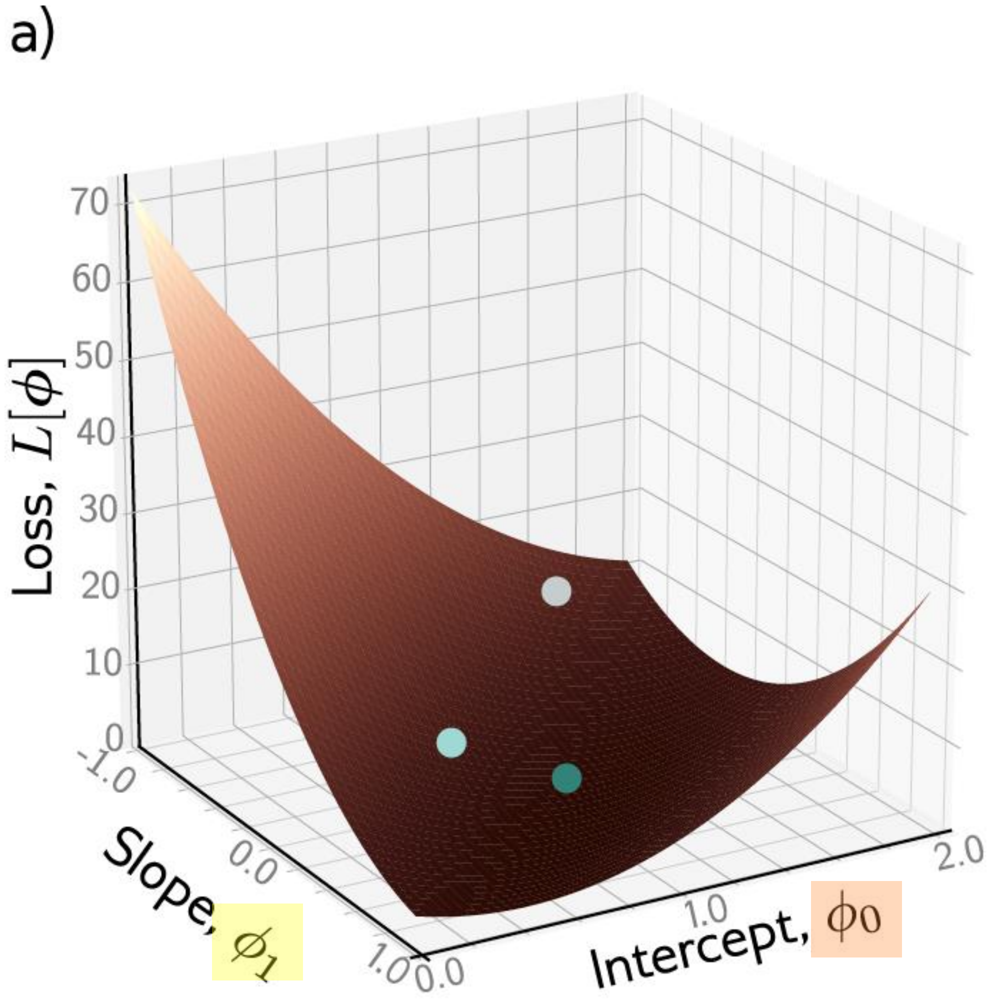


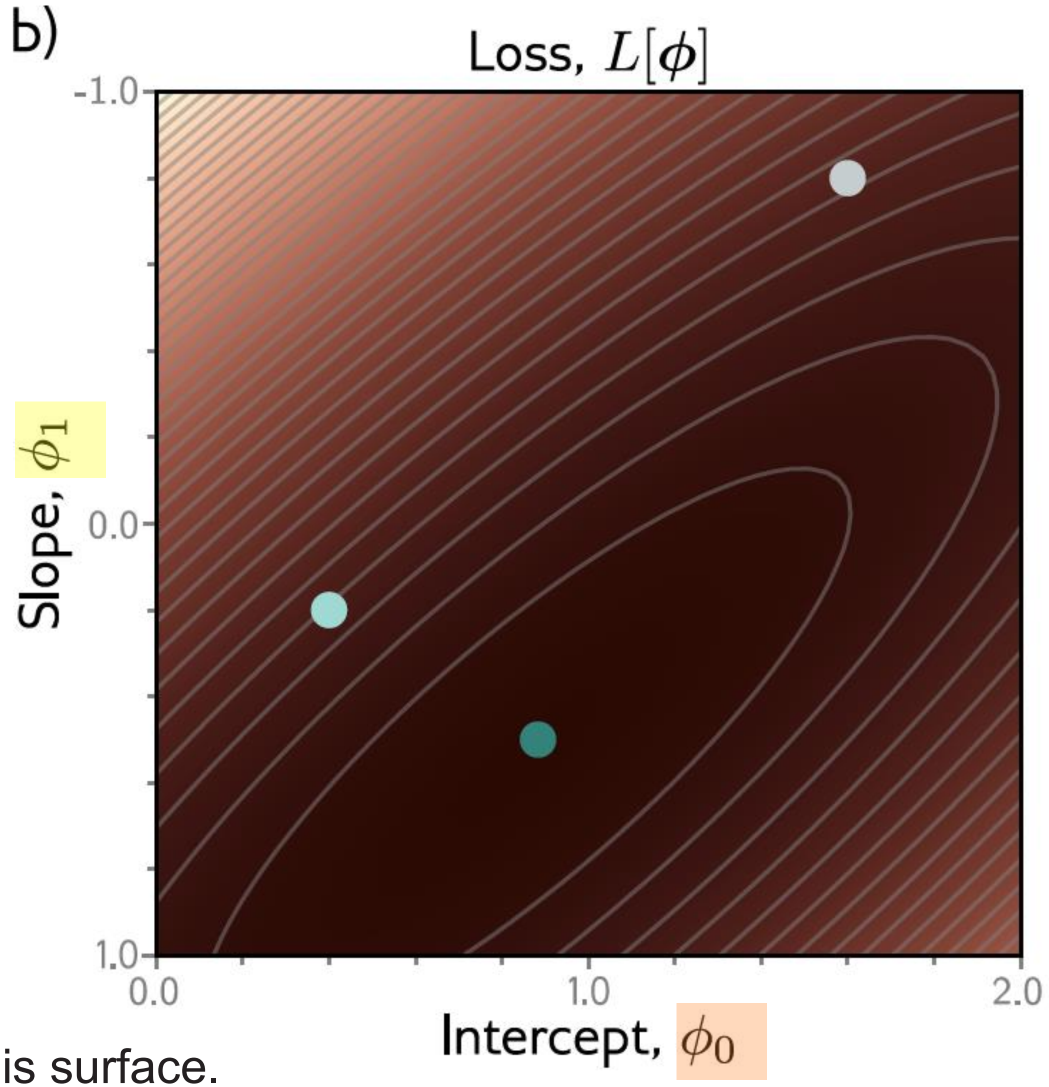
Example: 1D Linear regression loss function





Example: 1D Linear regression loss function

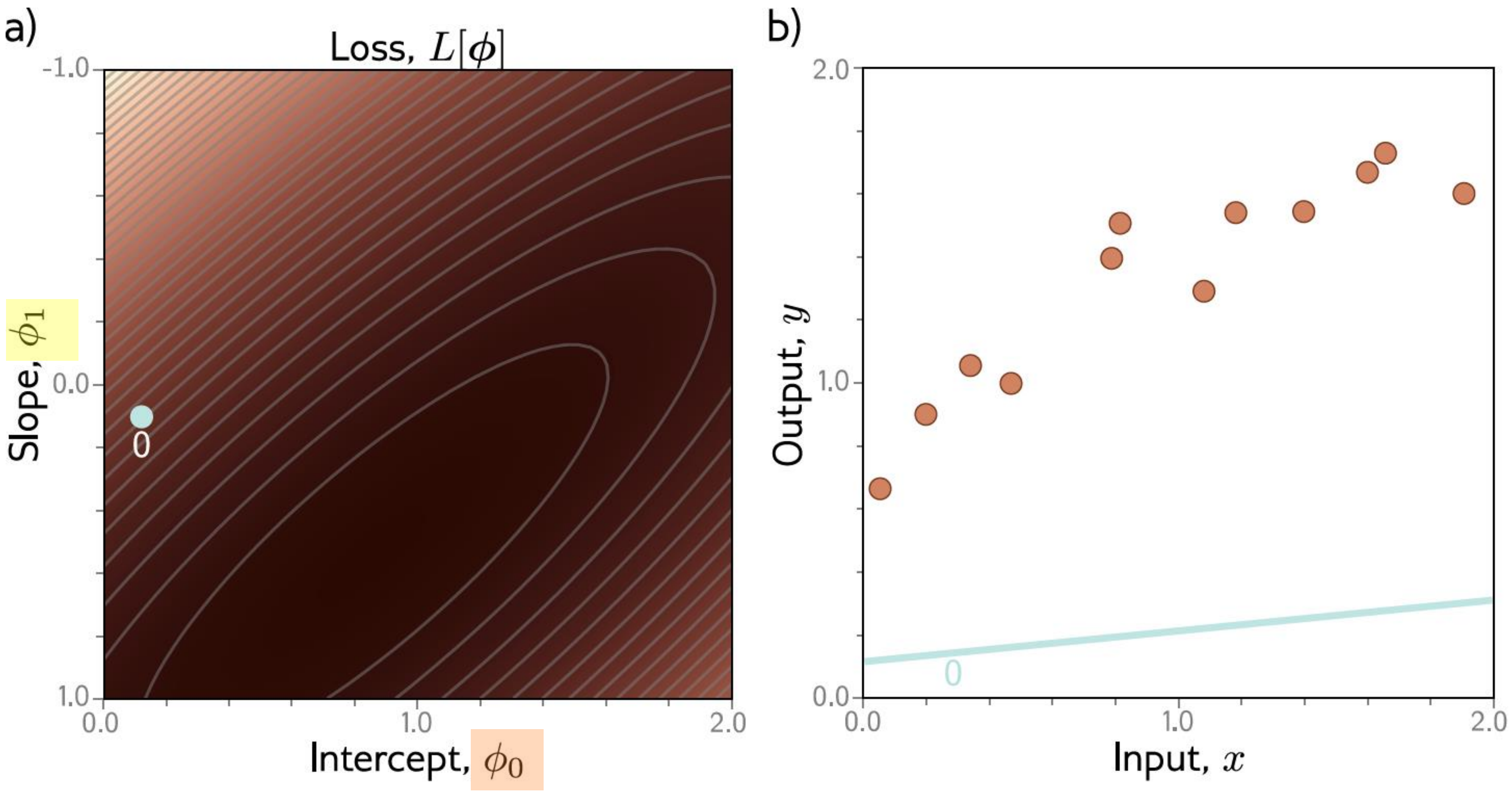




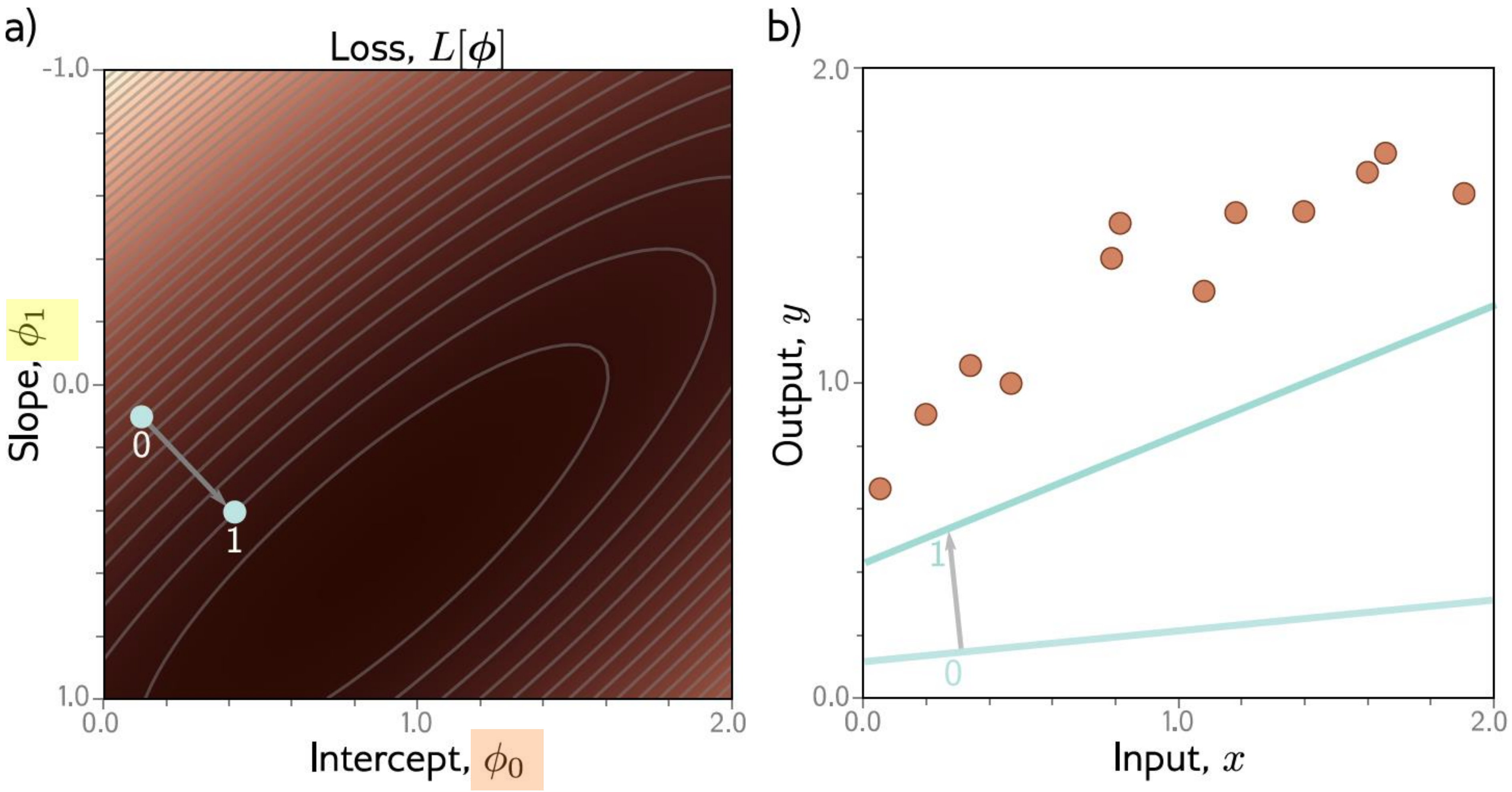
The “best” parameters are at the minimum of this surface.

Source: Prof. Simon Prince

Example: 1D Linear regression training

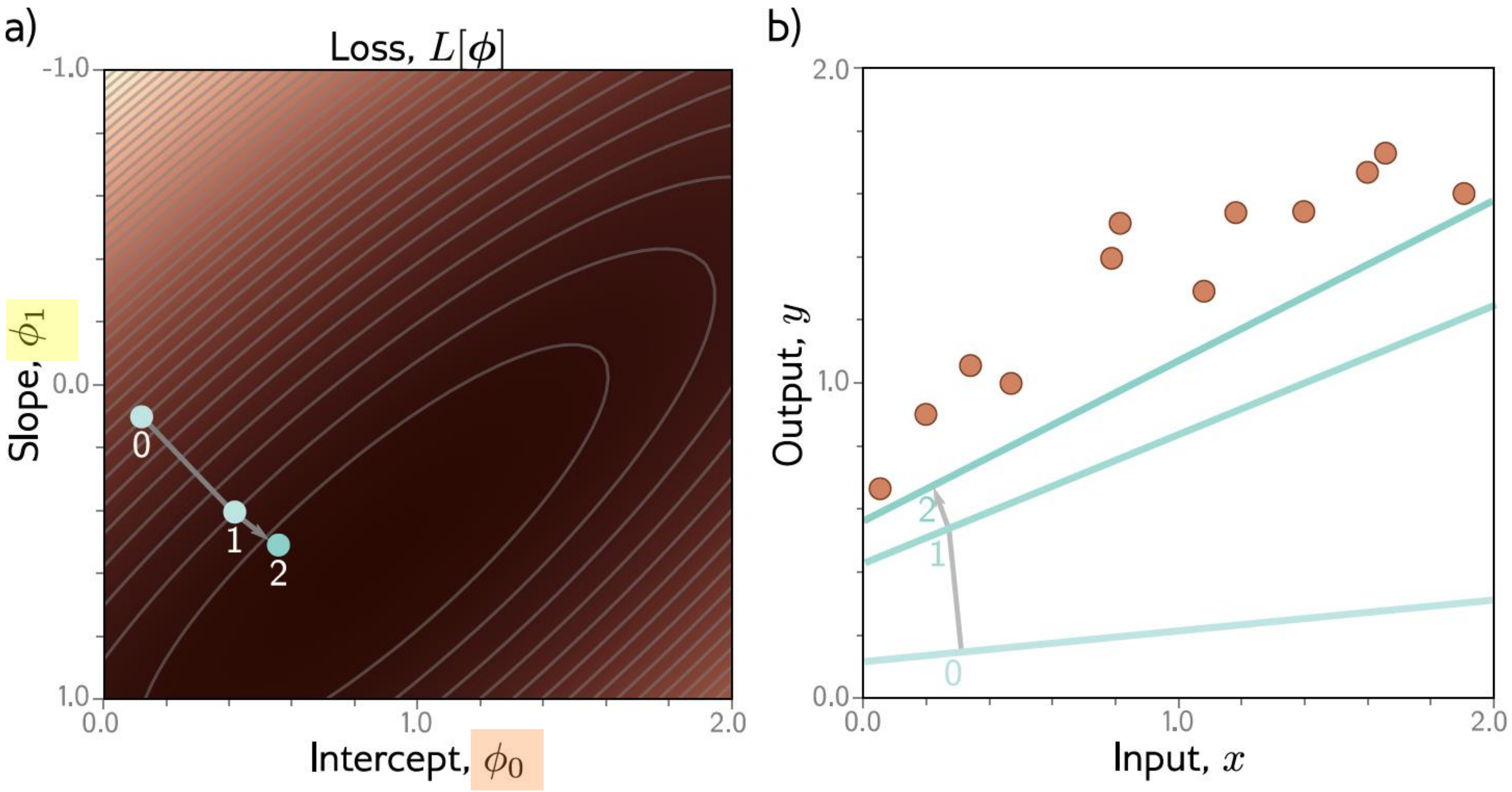


Example: 1D Linear regression training

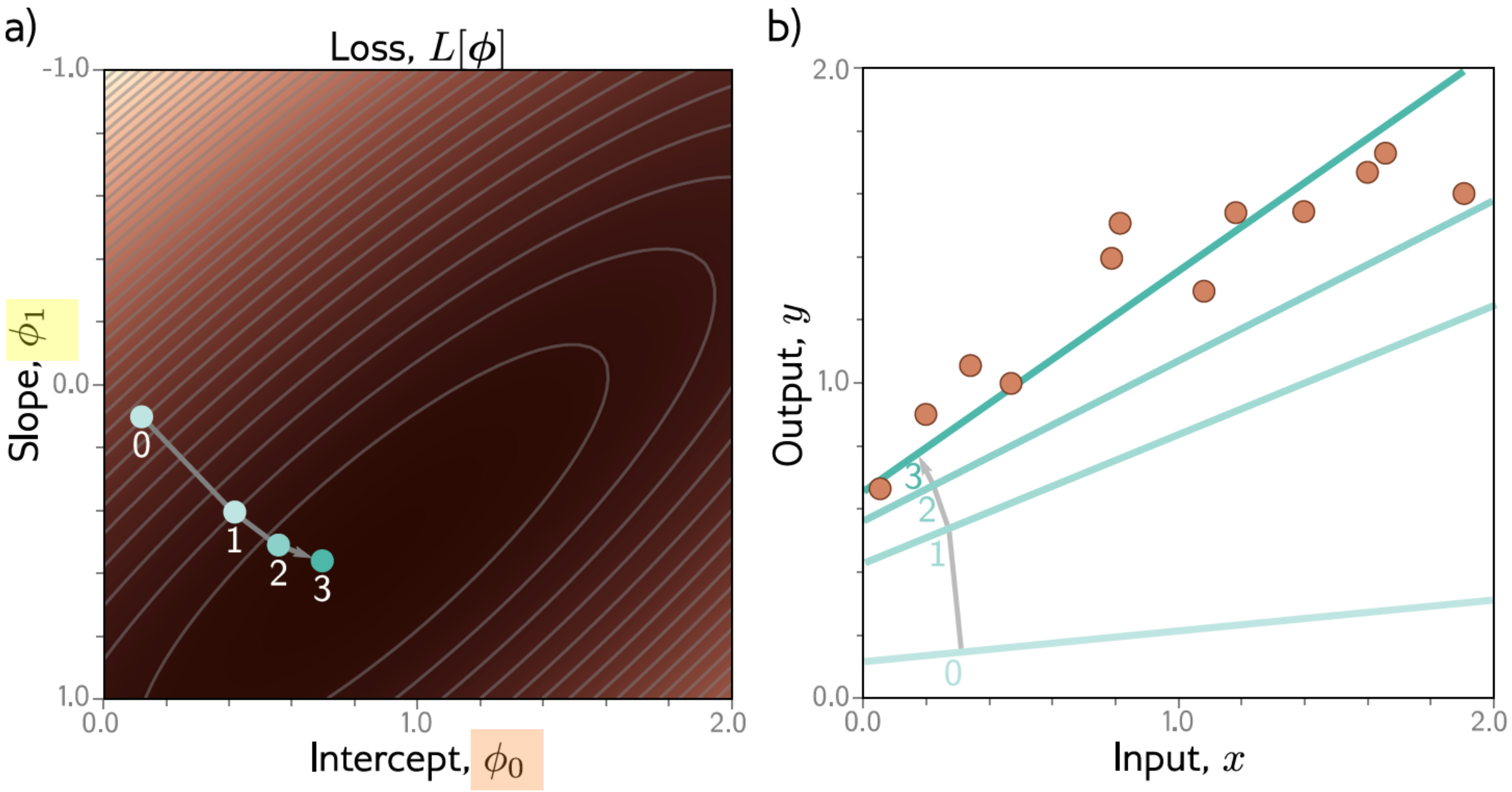


Source: Prof. Simon Prince

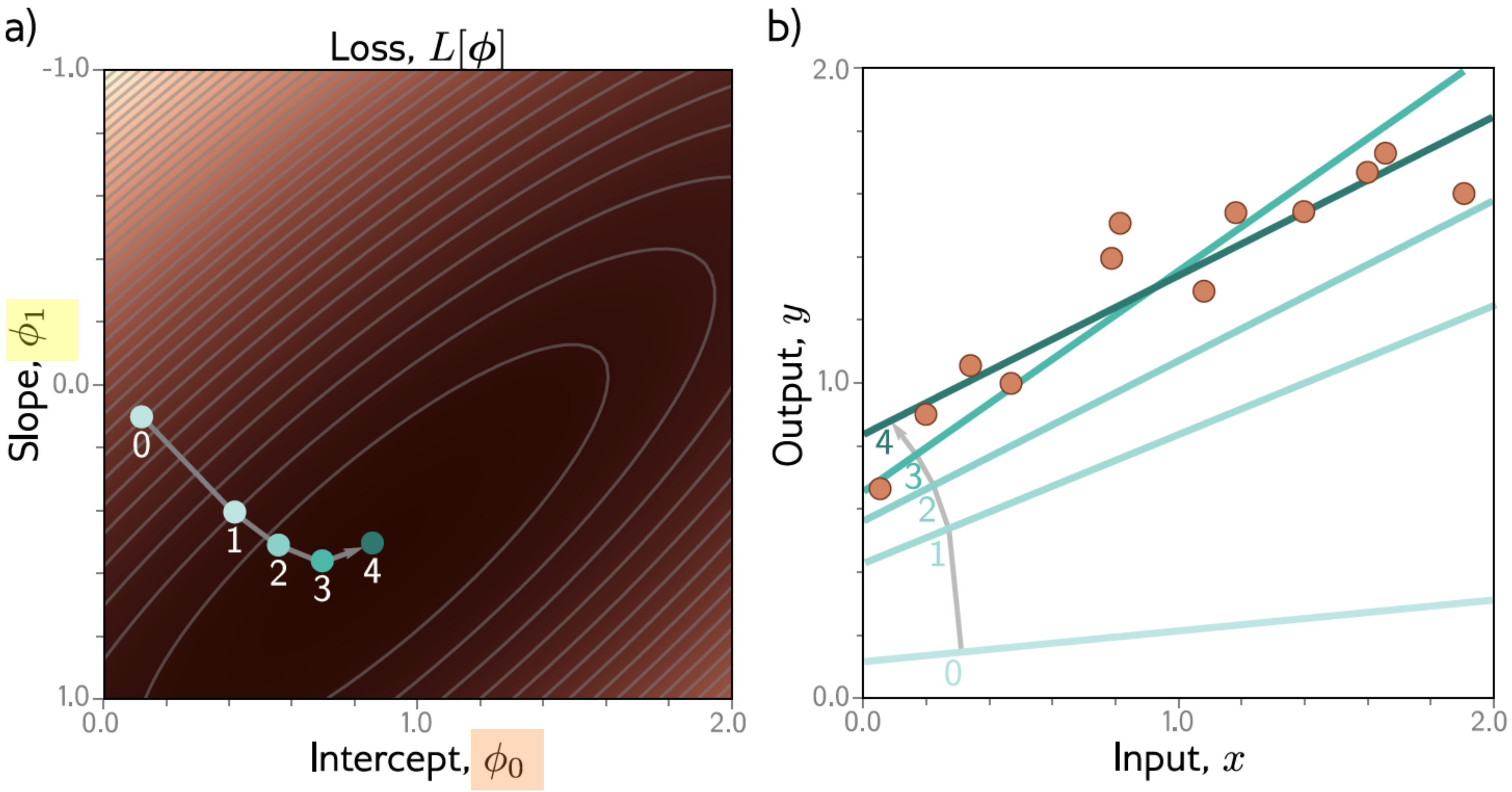
Example: 1D Linear regression training



Example: 1D Linear regression training



Example: 1D Linear regression training



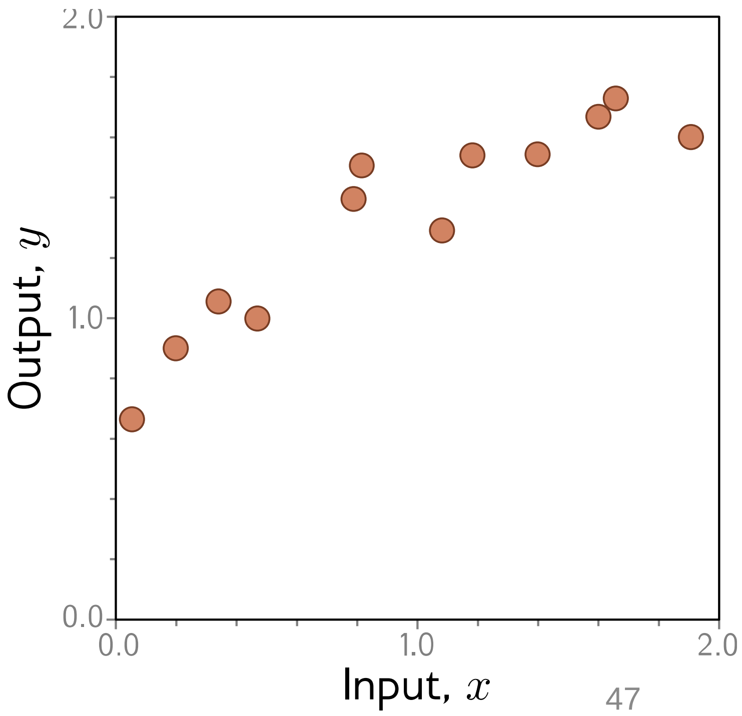
This technique is known as gradient descent

Possible objections

- But you can fit the line model in closed form!
  - Yes – but we won't be able to do this for more complex models
- But we could exhaustively try every slope and intercept combo!
  - Yes – but we won't be able to do this when there are a million parameters

Example: 1D Linear regression testing

- Test with different set of paired input/output data
  - Measure performance
  - Degree to which this is same as training = generalization
- Might not generalize well because
  - Model too simple
  - Model too complex
    - fits to statistical peculiarities of data
    - this is known as overfitting



Outline

- **Supervised learning overview**
- **Regression**
  - Recall linear regression
  - Notation
  - 1D Linear regression example
  - Logistic Regression
  - Softmax Regression
- Summary

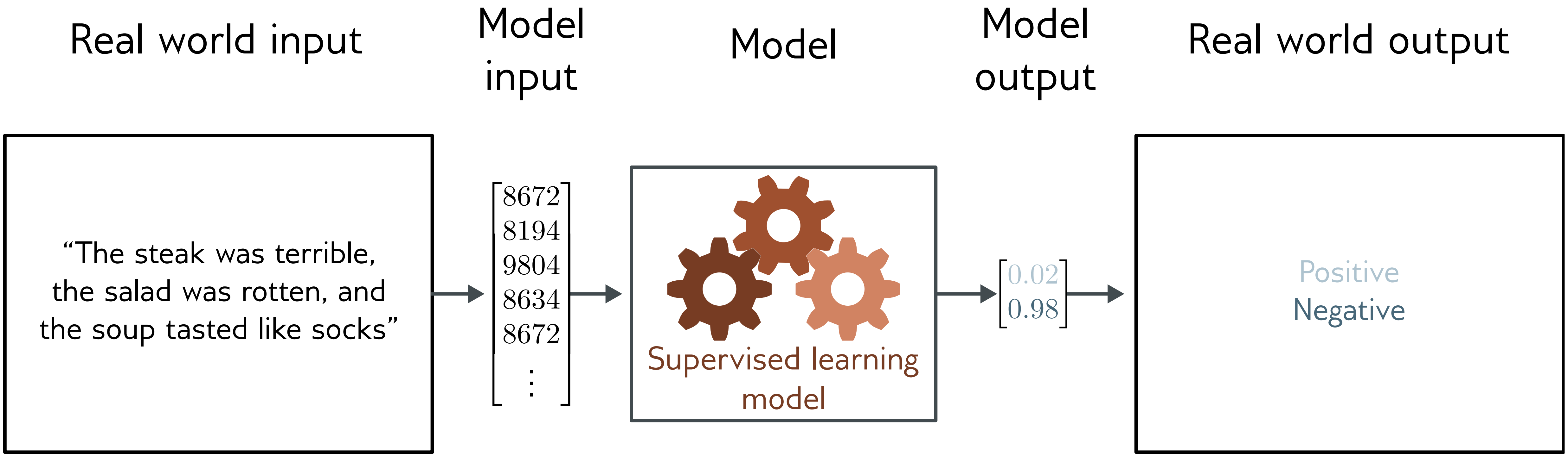
Shallow neural networks

- 1D regression model is obviously limited
  - Want to be able to describe input/output that are not lines
  - Want multiple inputs
  - Want multiple outputs

- Shallow neural networks
  - Flexible enough to describe arbitrarily complex input/output mappings
  - Can have as many inputs as we want
  - Can have as many outputs as we want

Source: Prof. Simon Prince

Text classification: receives a text string containing a restaurant review as input and predicts whether the review is positive or negative



- Binary classification problem (two discrete classes)
  - The model attempts to assign the input to one of two categories
  - The output vector contains the probabilities that the input belongs to each category

Regression vs. Classification

- Linear regression is to answer *how much?* or *how many?* questions.
  →Linear egression estimates a continuous value.

- A classification task to answer *which one?*
  →Classification predicts a discrete category.
  - Logistic Regression (binary classification)
  - Softmax Regression (multi-class classification)

Is this spam? [Y/N]

Act now, Action, Apply now, Apply online, Buy, Buy direct, Call, Call now, Click here, Do it today, Don't delete ...

Quarantined-Spam Notification 2021-09-11 16:30:03 +0800

寄件者:

收件者: "Yi-Ting Huang

CELLOPOINT Secure Your Email

Quarantined-Spam Notification

(Period: 2021-09-11 08:00:06 +0800 ~ 2021-09-11 16:30:03 +0800)

	NO.	Sender	Subject
	1	MaMa Rohita Rajkumar <kossi208emma@gmail.com>	(No Subject)

Click here to Login Mail Center

Help

Resend: Resend this email to your mailbox from Mail Center.

Not Spam&Resend: Resend this email to your mailbox from Mail Center, and report this mistakenly marked spam email to Global Antipam Center to reduce the false positive rate.

Add White&Resend: Resend this email to your mailbox and add sender email to personal white list on Mail Center.

Fake News? [Y/N]

台灣 衛福部長 陳時中提醒大家

再次強調：別出門，端午節(6月25日)過後，再看疫情控制情況！
警告：一旦染上，就算治癒了，後遺症也會拖累後半生！這場瘟疫比17年前的非典更嚴重，用的藥副作用更大。
如果出了特效藥，也只能保命，僅此而已！出門前想想你家人，別連累家人，能不出門就不出門，大家一起轉發吧！
這是一場戰役，不是兒戲，收起你盲目的自信和僥倖心理，也收起你事不關己高高掛起的態度，在這場戰役中沒有局外人！
在家！在家！在家！不要點贊！求轉發！
—— 陳時中

Source: NewTalk新聞

Fake News? [Y/N]

台灣 衛福部長 陳時中提醒大家

再次強調：別出門，端午節(6月25日)過後，再看疫情控制情況！
警告：一旦染上，就算治癒了，後遺症也會拖累後半生！這場瘟疫比17年前的非典更嚴重，用的藥副作用更大。如果出了特效藥，也只能保命，僅此而已！出門前想想你家人，別連累家人，能不出門就不出門，大家一起轉發吧！這是一場戰役，不是兒戲，收起你盲目的自信和僥倖心理，也收起你事不關己高高掛起的態度，在這場戰役中沒有局外人！
在家！在家！在家！不要點贊！求轉發！
—— 陳時中

Source: NewTalk新聞

?

Source: 衛服部

Positive or negative movie review? [P/N]

- ... characters and richly applied satire, and some great plot twists

- It was bad. The worst part about it was the fighting scenes... .

Source: Wikipedia

Positive or negative movie review? [P/N]

- (+): ... characters and richly applied satire, and some great plot twists
- (-): It was bad. The worst part about it was the fighting scenes... .

Source: Wikipedia

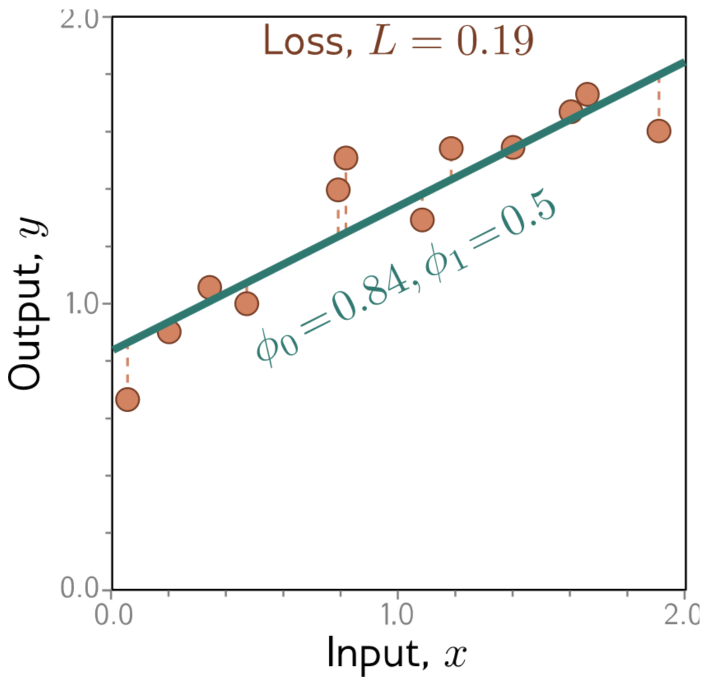
Binary classification in Logistic Regression

- Given a series of input/output pairs: $(x_i \;, y_i \;)$
  - (Email 1, spam), (Email 2, not spam), …
  - (News 1, not fake), (News 2, fake), …
  - (Review 1, positive), (Review 2, negative), …

- For each observation
  - We represent x_i by a feature vector $[x_{i,1}, x_{i,2}, \ldots, x_{i,n}]$
  - We compute an output: a predicted class $\hat{y_i} \in \{0, 1\}$
    - 0: is not spam, 1: is spam
    - 0: is not fake news, 1: is fake news
    - 0: negative review, 1: positive review

Features in logistic regression

- For feature x_i, weight w_i tells is how important is x_i
  - x_i = (0/1) review contains "great": w_i = +10
  - x_j = (0/1) review contains "worst": w_j = -10
  - x_k = (0/1) review contains "bad": w_k = -2
  - bias



How to do classification

- For each feature x_i, weight w_i tells is us important of x_i
  - Plus a bias b
- We'll sum up all the weighted features and the bias

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

$$z = \boldsymbol{w} \cdot \boldsymbol{x} + b$$

- If this sum is high, we say y = 1; if low, then y = 0

But we want a probabilistic classifier

$$z = w \cdot x + b$$

- The problem: $z \in$ [-∞, ∞] is not a probability, it is a number !
  - We need to formalize “sum is high”.

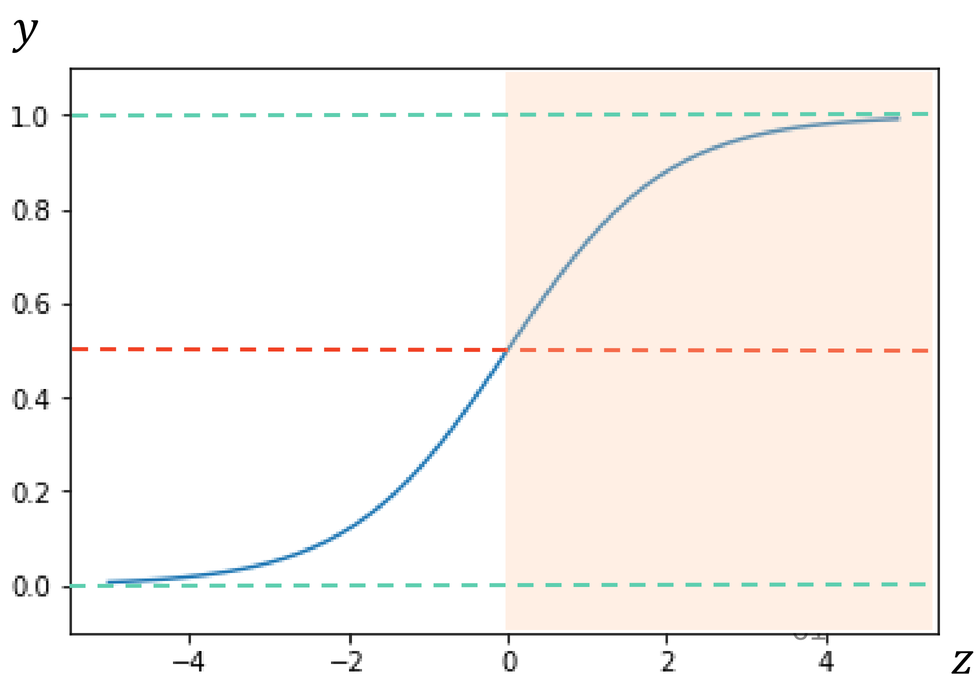
- We would like a principled classifier that gives us a probability [0, 1].

- We want a model that can tell us:
  - $p(y = 1|x; \theta)$
  - $p(y = 0|x; \theta)$

Solution: a **sigmoid** function

- Solution: use a **sigmoid** function (also called logistic function) of z that goes from 0 to 1.

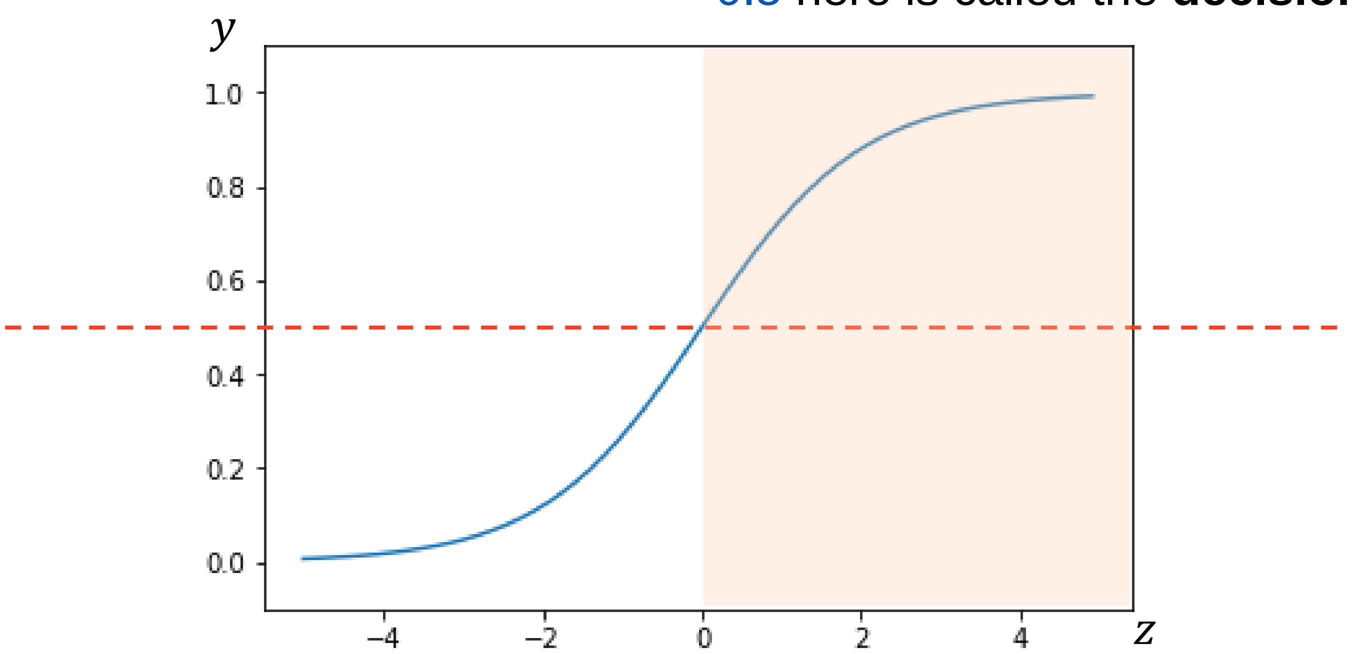
$$y = \sigma(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + \exp(-z)}$$



Turning a probability into a classifier

$$\hat{y} = \begin{cases} 1 & \text{if } P(y = 1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

0.5 here is called the **decision boundary**.



Example: does y=1 or y=0?

- It's hokey. There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ? For one thing , the cast is great . Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

Sentiment example: does y=1 or y=0?

- It's hokey. There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ? For one thing , the cast is great . Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

$x_1 = 3$: count(positive lexicon $\in$ doc)

$x_2 = 2$: count(negative lexicon $\in$ doc)

$$x_3 = 1: \begin{cases} 1 & \text{if } ''\text{no}'' \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$$

$x_4 = 3$: count(1st and 2nd pronouns $\in$ doc)

$$x_5 = 0: \begin{cases} 1 & \text{if } ''!'' \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$$

$x_6 = \ln(66) = 4.19$: log(word count of doc)

Classifying sentiment for input x

$x_1 = 3$: count(positive lexicon ∈ doc)

$x_2 = 2$: count(negative lexicon ∈ doc)

$x_3 = 1: \begin{cases} 1 & \text{if } ''\text{no}'' \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$

$x_4 = 3$: count(1st and 2nd pronouns ∈ doc)

$x_5 = 0: \begin{cases} 1 & \text{if } ''!'' \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$

$x_6 = \ln(66) = 4.19$: log(word count of doc)

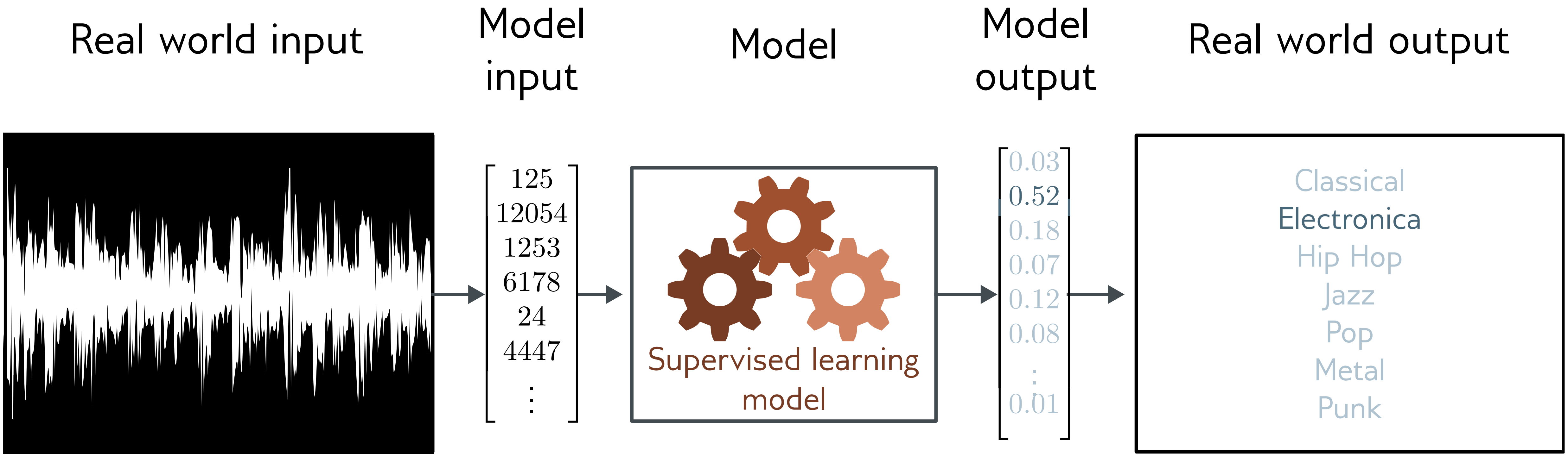
Suppose w = [2.5, -5.0, -1.2, 0.5, 2.0, 0.7], b=0.1

$$
\begin{aligned}
p(+|x) = P(Y = 1|x) \\
&= \sigma(w \cdot x + b) \\
&= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\
&= \sigma(.833) \\
&= 0.70
\end{aligned}
$$

Outline

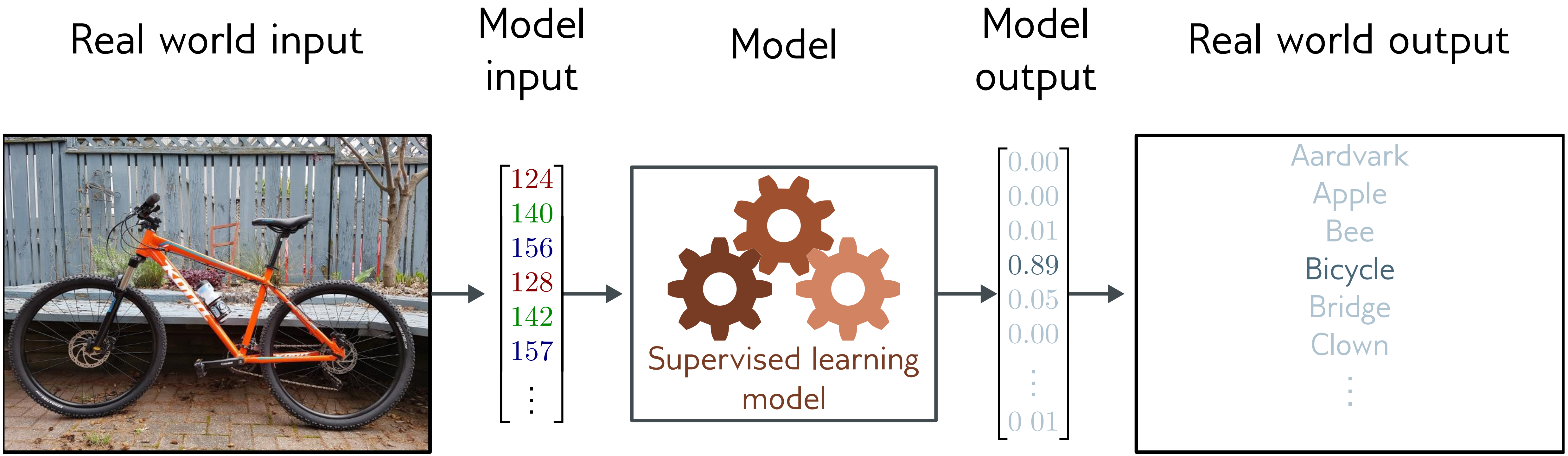
- **Supervised learning overview**
- **Regression**
  - Recall linear regression
  - Notation
  - 1D Linear regression example
  - Logistic Regression
  - Softmax Regression
- From regressions to shallow neural networks

Music genre classification: the input is an audio file, and the model predicts which genre of music it contains



- Multiclass classification problem (discrete classes, >2 possible values)
- Recurrent neural network (RNN)

Image classification: the input is an image, and the model predicts which object it contains



- Multiclass classification problem (discrete classes, >2 possible classes)
- Convolutional network

Multinomial Logistic Regression

MNIST database

- Often we need more than 2 classes
  - Positive/negative/neutral
  - Digit (0, 1, 2, 3,…9)
  - Parts of speech (noun, verb, adjective, adverb, etc.)
  - Types of malware (trojans, worms, ransomware, etc.)

- If >2 classes we use **multinomial logistic regression**

  = **Softmax regression**

- So “logistic regression” will just mean binary (2 output classes)

Multi-class classification in softmax regression

- Given a series of input/output pairs: $(x_i\ , y_i\ )$
  - (Review 1, positive), (Review 2, negative), (Review 3, neutral) …

- How to represent y?
  - $y \in \{1, 2, 3\}$ represents {positive, negative, neutral} respectively.

Multi-class classification in softmax regression

- Given a series of input/output pairs: (x^i, y^i)
  - (Review 1, positive), (Review 2, negative), (Review 3, neutral) …
- How to represent y?
  - $y \in \{1, 2, 3\}$ represents {positive, negative, neutral} respectively.
  - Not applicable when the classification problems do not come with natural ordering among the classes.
- One-hot encoding: $y \in \{(1,0,0), (0,1,0), (0,0,1)\}$
  - positive: {1, 0, 0}
  - negative: {0, 1, 0}
  - neutral: {0, 0, 1}

Weights in softmax regression

- Suppose 4 features x_1, x_2, x_3, x_4 and 3 class.

$$z_1 = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + b_1$$
$$z_2 = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + w_{24}x_4 + b_2$$
$$z_3 = w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + w_{34}x_4 + b_3$$

- Given 12 scalar ($\boldsymbol{W}_{3\times4}$) as weights w_{ij} and 3 scalars ($b_{3\times1}$) as biases b_i , we compute three logits z_1, z_2, z_3 for each input:

$$\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + b \qquad \boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + b$$
$$z_{|c|\times1} = \mathrm{W}_{|c|\times|f|}\, x_{|f|\times1} + b_{|c|\times1} \qquad z_{3\times1} = \mathrm{W}_{3\times4}\, x_{4\times1} + b_{3\times1}$$

The softmax function

- The probability of everything must still sum to 1.

$$P(c1|x) + P(c2|x) + P(c3|x) = 1$$

$$P(\text{positive}|\text{doc}) + P(\text{negative}|\text{doc}) + P(\text{neutral}|\text{doc}) = 1$$

- Need a generalization of the sigmoid called the **softmax**
  - Take a vector $\boldsymbol{z} = [z_1, z_2, \ldots, z_k]$ of k arbitrary values
  - Outputs a probability distribution
    - Each value in the range [0, 1]
    - All the values summing to 1

Features and weights in softmax regression

$$p(y=c|x)=\frac{\exp(\boldsymbol{w}_c\cdot\boldsymbol{x}+b_c)}{\sum_{j=1}^{k}\exp(\boldsymbol{w}_j\cdot\boldsymbol{x}+b_j)},c\in C$$

- Input is still the product between weight vector ***w*** and input vector ***x*** (plus a bias).
- But now we will need **separate weight vectors *W*** for each of the *c* classes.

The softmax function

- Turns a vector $z = [z_1, z_2, \ldots, z_k]$ of *k* arbitrary values into probabilities.

$$softmax(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \le i \le k$$

- Turns a vector $z = [z_1, z_2, \ldots, z_k]$ of *k* arbitrary values into probabilities.

$$softmax(z) = [\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)}]$$

The softmax function

- Turns a vector $z = [z_1, z_2, \dots, z_k]$ of *k* arbitrary values into probabilities.

$$z = [0.6, 1.1, -1.5, 1.2, 3.2, -1.1]$$

$$softmax(z) = [\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \dots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)}]$$

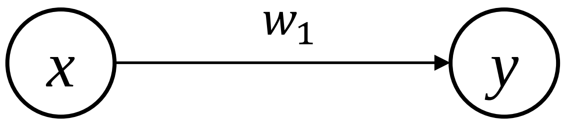
$$z = [0.055, 0.090, 0.006, 0.099, 0.74, 0.010]$$
$$\hat{y} = argmax(z)$$

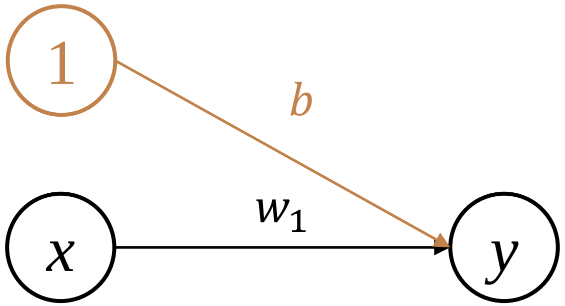
Outline

- **Supervised learning overview**
- **Regression**
  - Recall linear regression
  - Notation
  - 1D Linear regression example
  - Logistic Regression
  - Softmax Regression
- **Summary**

Linear regression

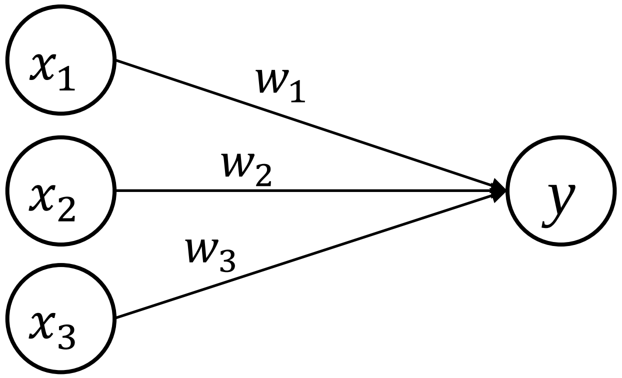
- 1D linear regression: 1 input, 1 output
  - $y = b + w_1 x_1$

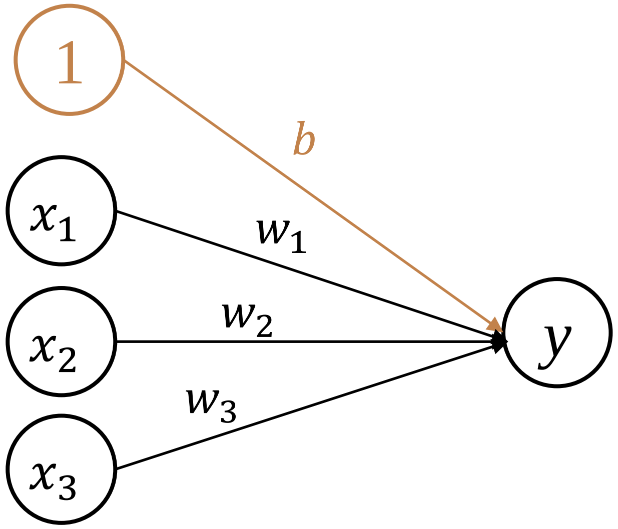




Linear regression

- Linear regression: 3 input (features), 1 output
  - $y = b + w_1 x_1 + w_2 x_2 + w_3 x_3$

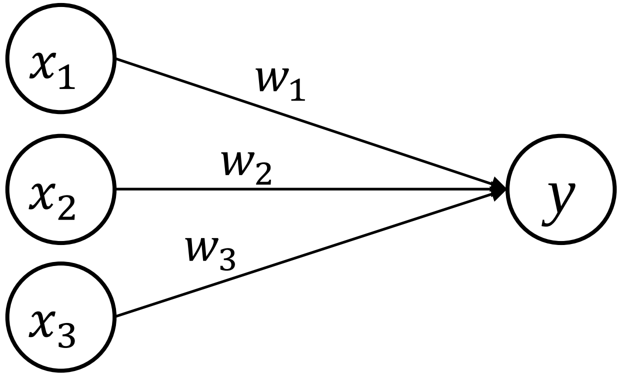


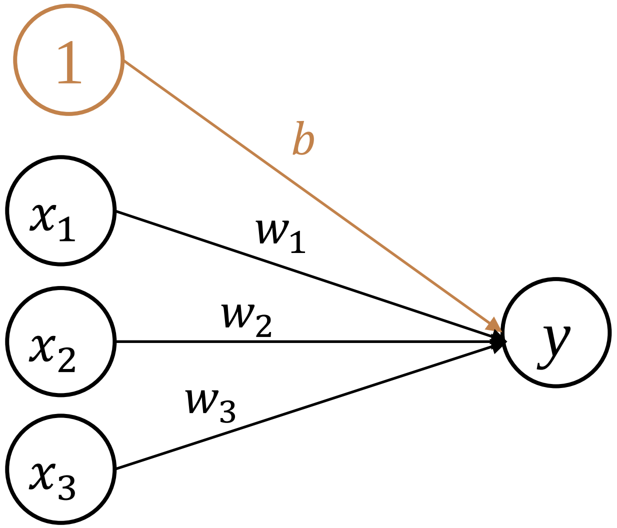


Linear regression

$$y = b + \sum_{i=1}^{n} w_i x_i$$

- Linear regression: 3 input (features), 1 output
  - $y = b + w_1 x_1 + w_2 x_2 + w_3 x_3$

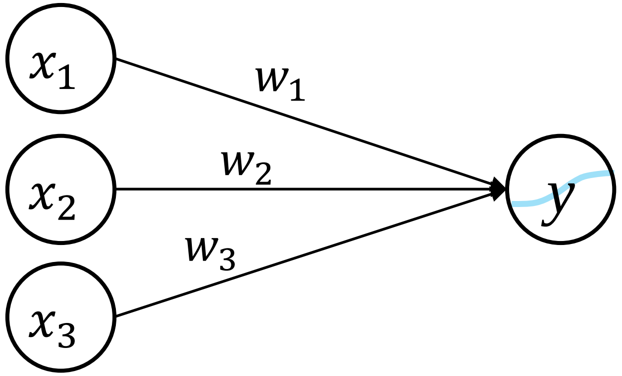


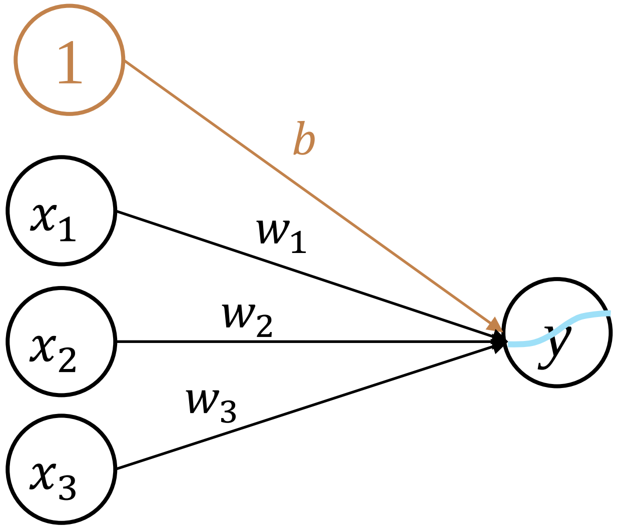


Logistic regression

$$y = sigmoid(\, b + \sum_{i=1}^{n} w_i x_i)$$

- Logistic regression: 3 input (features), 2 output
  - $y = b + w_1 x_1 + w_2 x_2 + w_3 x_3$

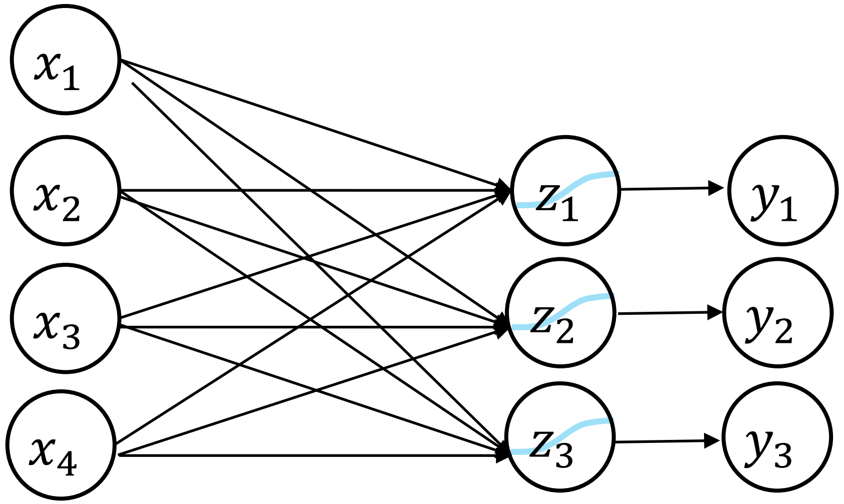


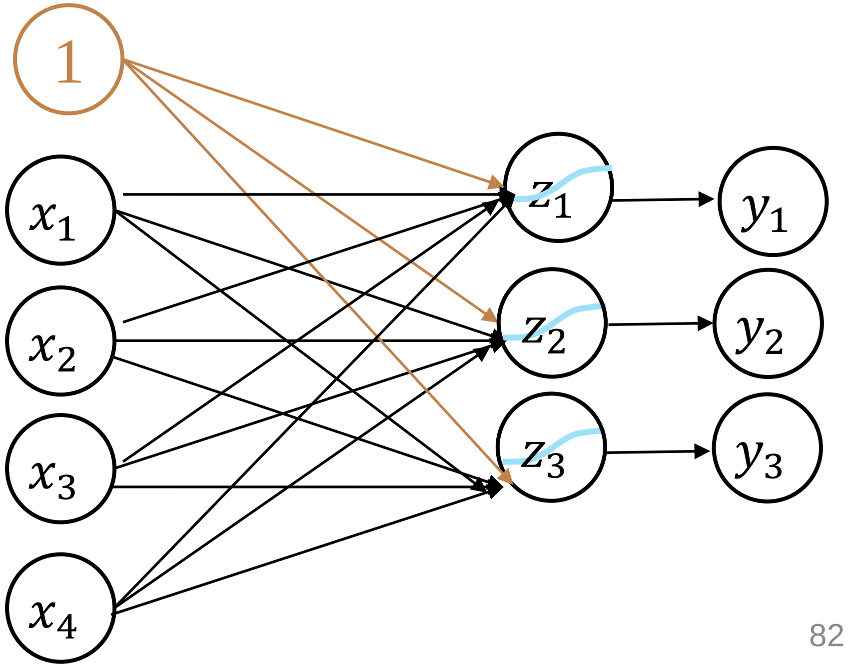


Softmax regression

$$\boldsymbol{y} = softmax(\,\boldsymbol{b} + \boldsymbol{W}\boldsymbol{x})$$

- Softmax regression: 4 input (features), 3 output
  - $y_1 = b + w_{1,1}x_1 + w_{1,2}x_2 + w_{1,3}x_3 + w_{1,4}x_4$
  - $y_2 = b + w_{2,1}x_1 + w_{2,2}x_2 + w_{2,3}x_3 + w_{2,4}x_4$
  - $y_3 = b + w_{3,1}x_1 + w_{3,2}x_2 + w_{3,3}x_3 + w_{3,4}x_4$





Summary

- Supervised learning process
- Forward propagation
- Linear/logistic/softmax regression