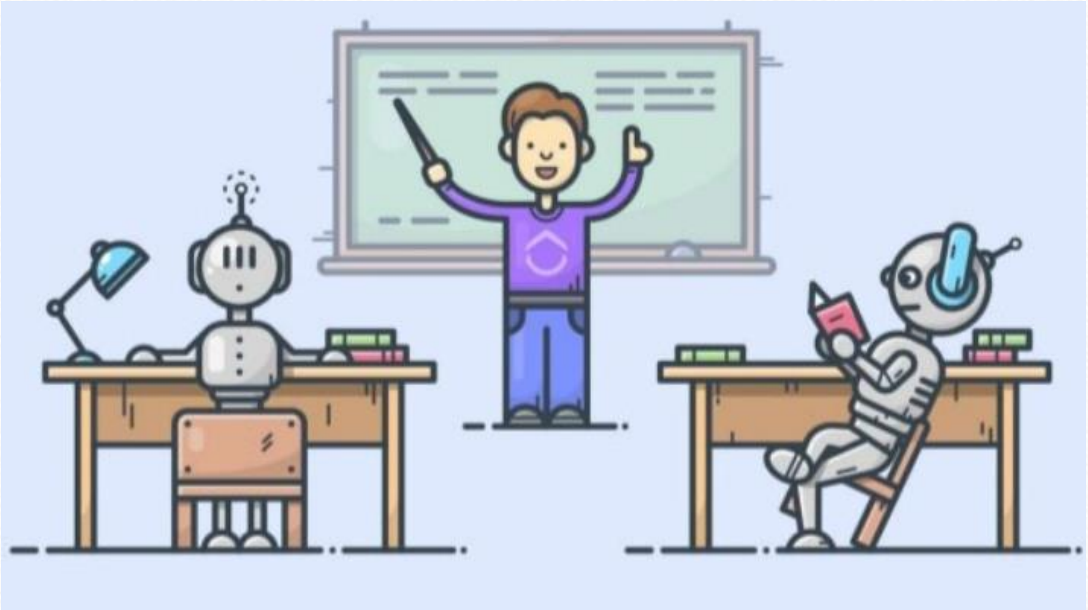
物聯網實務

(十一)

廖裕評

AI game

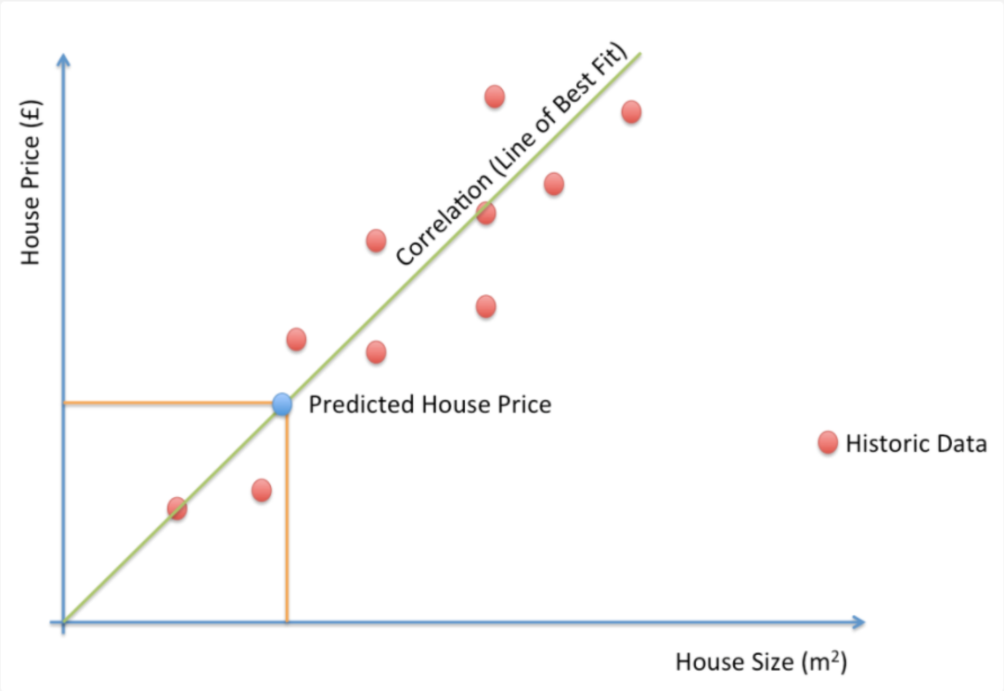
Teaching AI to Understand Our World



https://www.slideshare.net/sakhaglobal/unsupervised-learning-teaching-ai-to-understand-our-world

Regression

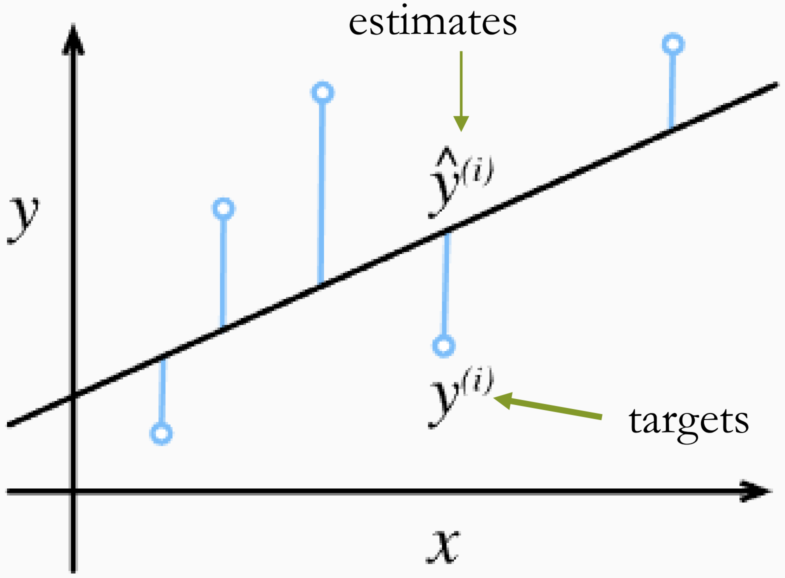
- *Regression* refers to a set of methods for modeling the relationship between one or more independent variables and a dependent variable.
- *Prediction:* predicting prices (of homes, stocks, etc.), predicting length of stay (for patients in the hospital), demand forecasting (for retail sales)



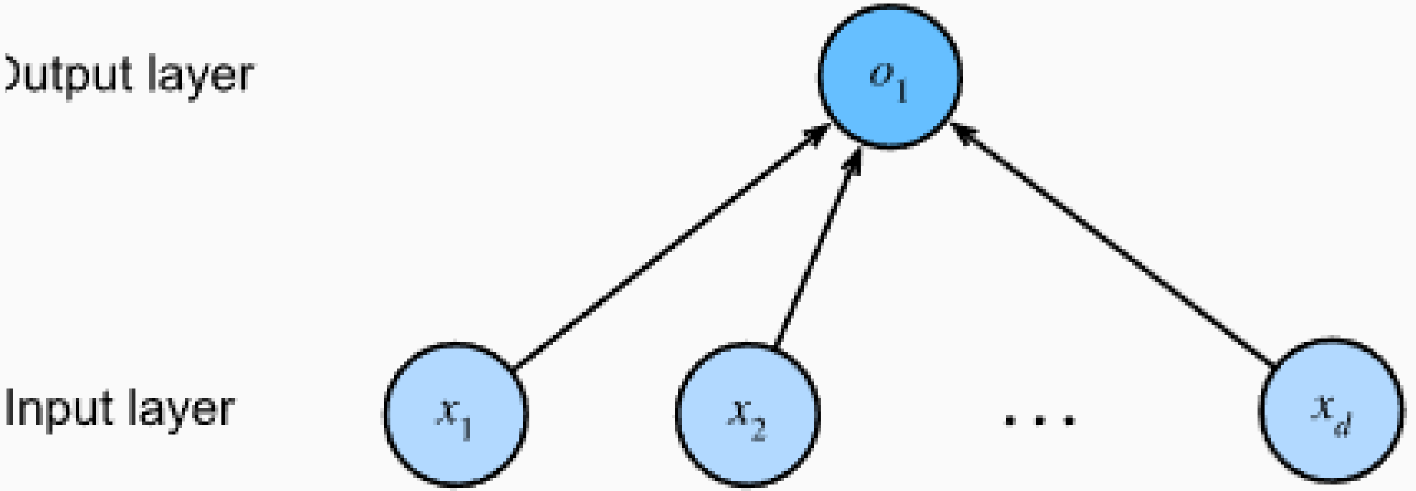
https://davidgildeh.com/2013/09/16/the-future-of-enterprise-machine-learning/

https://d2l.ai/chapter_linear-networks/linear-regression.html

Linear regression



y=wx+b

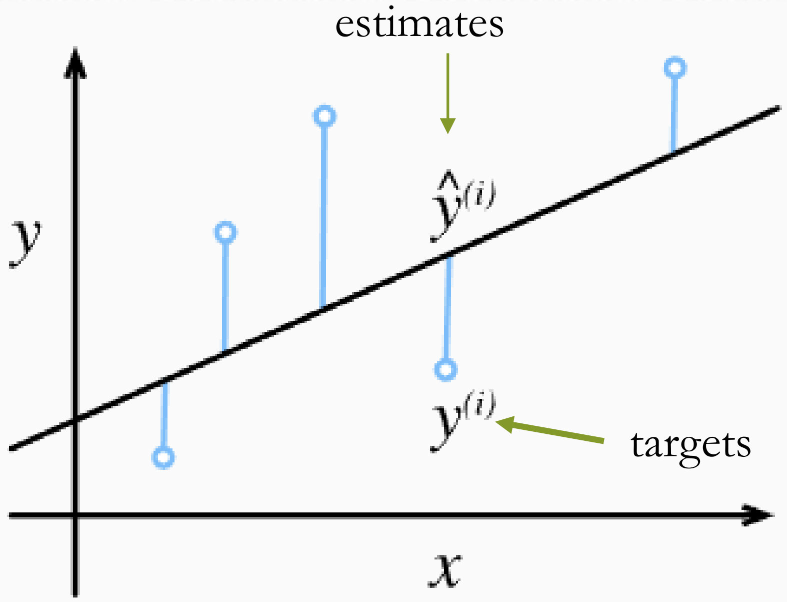


$$\hat{y} = w_1 x_1 + \ldots + w_d x_d + b$$

single-layer neural network

https://d2l.ai/chapter_linear-regression/linear-regression.html

Loss Function



y=wx+b

squared error $$l^{(i)}(\mathbf{w}, b) = \frac{1}{2}\left(\hat{y}^{(i)} - y^{(i)}\right)^2.$$

the losses on the training set $$L(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} l^{(i)}(\mathbf{w}, b)$$

training the model $$\mathbf{w}^*, b^* = \operatorname*{argmin}_{\mathbf{w}, b}\ L(\mathbf{w}, b).$$

our prediction for an example i is $\hat{y}^{(i)}$ and the corresponding true label is $y^{(i)}$

https://d2l.ai/chapter_linear-regression/linear-regression.html

Regression

- Typically, we will use n to denote the number of examples in our dataset. We index the data examples by i, denoting each input as $x^{(i)} = [x_1^{(i)}, x_2^{(i)}]^{\mathrm{T}}$ and the corresponding label as $y^{(i)}$.

weights

$$\text{price} = w_{\text{area}} \cdot \text{area} + w_{\text{age}} \cdot \text{age} + b.$$

x_1 x_2 *bias*

$$y^{(i)} = w_{area} \cdot x_1^{(i)} + w_{age} \cdot x_2^{(i)} + b$$

$$y^{(i)} = w_1 \cdot x_1^{(i)} + w_2 \cdot x_2^{(i)} + b$$

https://d2l.ai/chapter_linear-networks/linear-regression.html

Regression

- When our inputs consist of d d features, we express our prediction $\hat{y}$ (in general the "hat" symbol denotes estimates) as

$$\hat{y} = w_1 x_1 + \ldots + w_d x_d + b.$$

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b.$$

a vector $\mathbf{w} \in \mathbb{R}^d$

a vector $\mathbf{x} \in \mathbb{R}^d$

$$x = [x_1, x_2, \ldots, x_d]^{\mathrm{T}} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_d \end{bmatrix}$$

$$w = [w_1, w_2, \ldots, w_d]^{\mathrm{T}} = \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_d \end{bmatrix}$$

$$w^T = [w_1, w_2, \ldots, w_d]$$

$$\hat{y} = [w_1, w_2, \ldots, w_d] \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_d \end{bmatrix}$$

Regression

$\hat{\mathbf{y}} \in \mathbb{R}^n \qquad \mathbf{X} \in \mathbb{R}^{n \times d}.$

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w} + b,$$

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \cdots \\ \hat{y}_n \end{bmatrix} = \begin{bmatrix} x_1{}^{(1)} & x_2{}^{(1)} & & x_d{}^{(1)} \\ x_1{}^{(2)} , & x_2{}^{(2)} , & \dots , & x_d{}^{(2)} \\ & & & \\ x_1{}^{(n)} & x_2{}^{(n)} & & x_d{}^{(n)} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_d \end{bmatrix} + \mathrm{b}$$

$$x^{(i)} = [x_1{}^{(i)} , x_2{}^{(i)}, \dots , x_d{}^{(i)}]^{\mathrm{T}} = \begin{bmatrix} x_1{}^{(i)} \\ x_2{}^{(i)} \\ \cdots \\ x_d{}^{(i)} \end{bmatrix}$$

$$X = \begin{bmatrix} x_1{}^{(1)} & x_2{}^{(1)} & & x_d{}^{(1)} \\ x_1{}^{(2)} , & x_2{}^{(2)} , & \dots , & x_d{}^{(2)} \\ & & & \\ x_1{}^{(n)} & x_2{}^{(n)} & & x_d{}^{(n)} \end{bmatrix}$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_d \end{bmatrix}$$

Loss Function

The prediction for an example i is $\hat{y}_1{}^{(i)}$

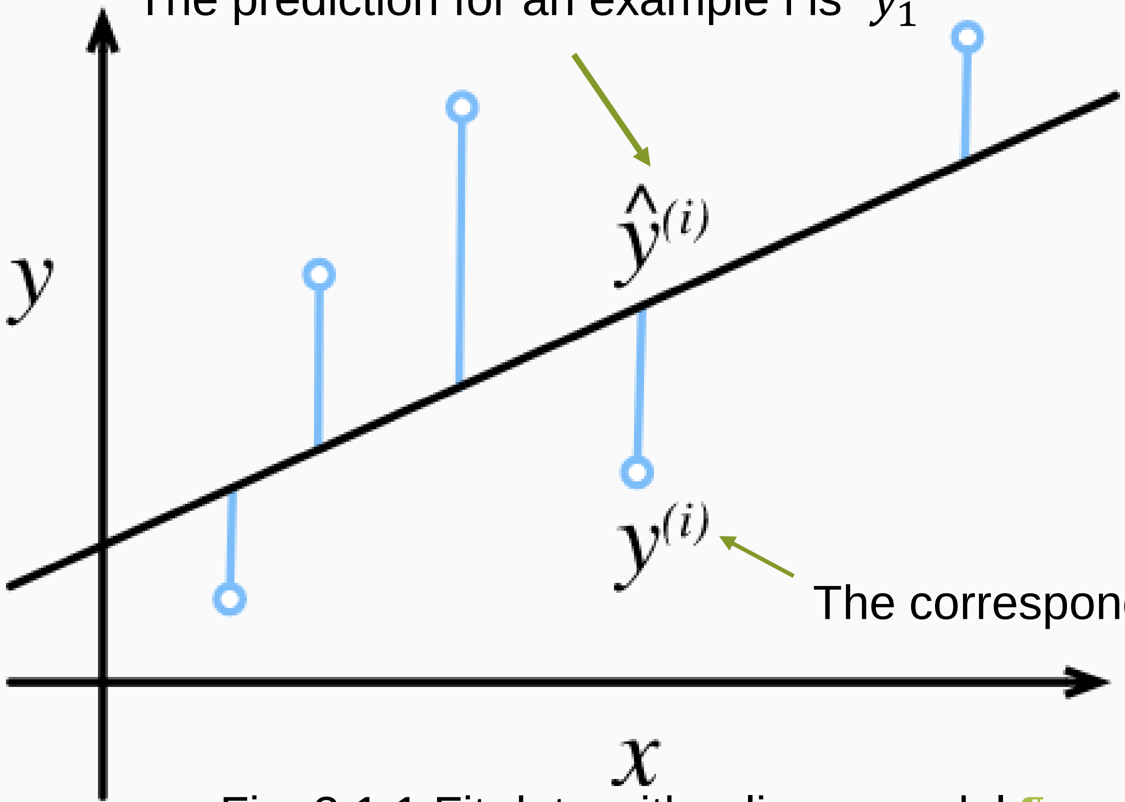


Fig. 3.1.1 Fit data with a linear model.¶

The corresponding true label is $y^{(i)}$

The loss function for an example i is:

$$l^{(i)}(\mathbf{w}, b) = \frac{1}{2}\left(\hat{y}^{(i)} - y^{(i)}\right)^2.$$

average loss

$$L(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} l^{(i)}(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\left(\mathbf{w}^\top\mathbf{x}^{(i)} + b - y^{(i)}\right)^2. \tag{3.1.6}$$

When training the model, we want to find parameters ($\mathbf{w}^*, b^*$) that minimize the total loss across all training examples:

$$\mathbf{w}^*, b^* = \operatorname*{argmin}_{\mathbf{w}, b}\ L(\mathbf{w}, b). \tag{3.1.7}$$

Making Predictions with the Learned Model

- Given the learned linear regression model $\hat{w}^T$x + $\hat{b}$, we can now estimate the price of a new house (not contained in the training data)

- given its area X_1 and age X_2. Estimating targets given features is commonly called *prediction* or *inference*.

From Linear Regression to Deep Networks

- Neural N

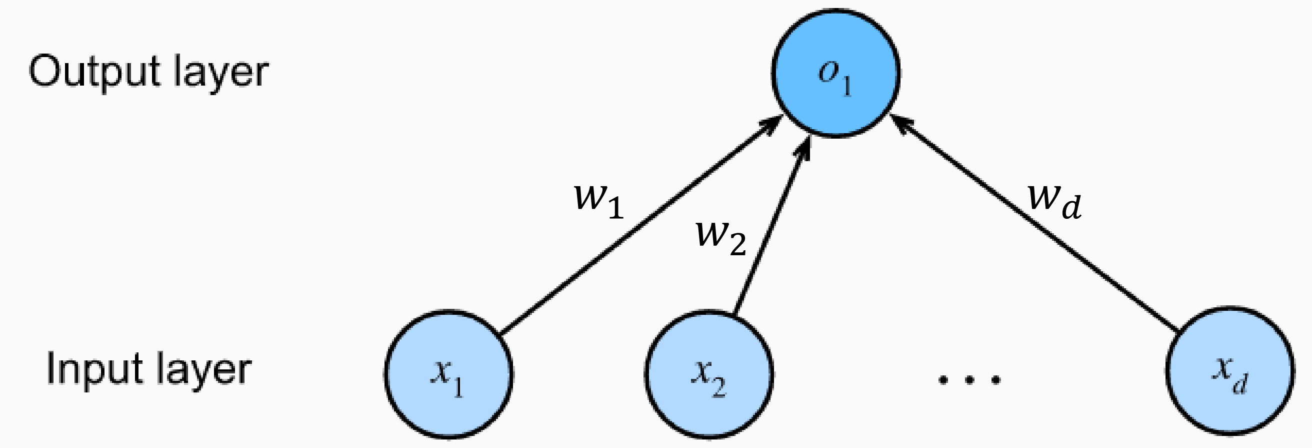
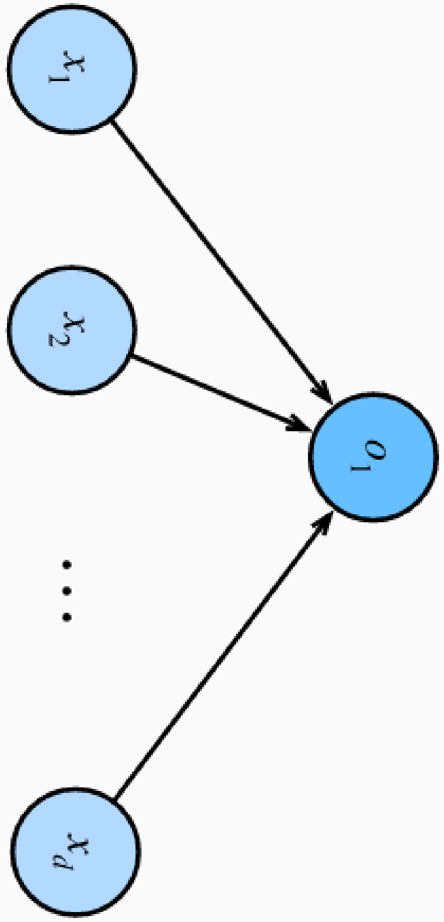
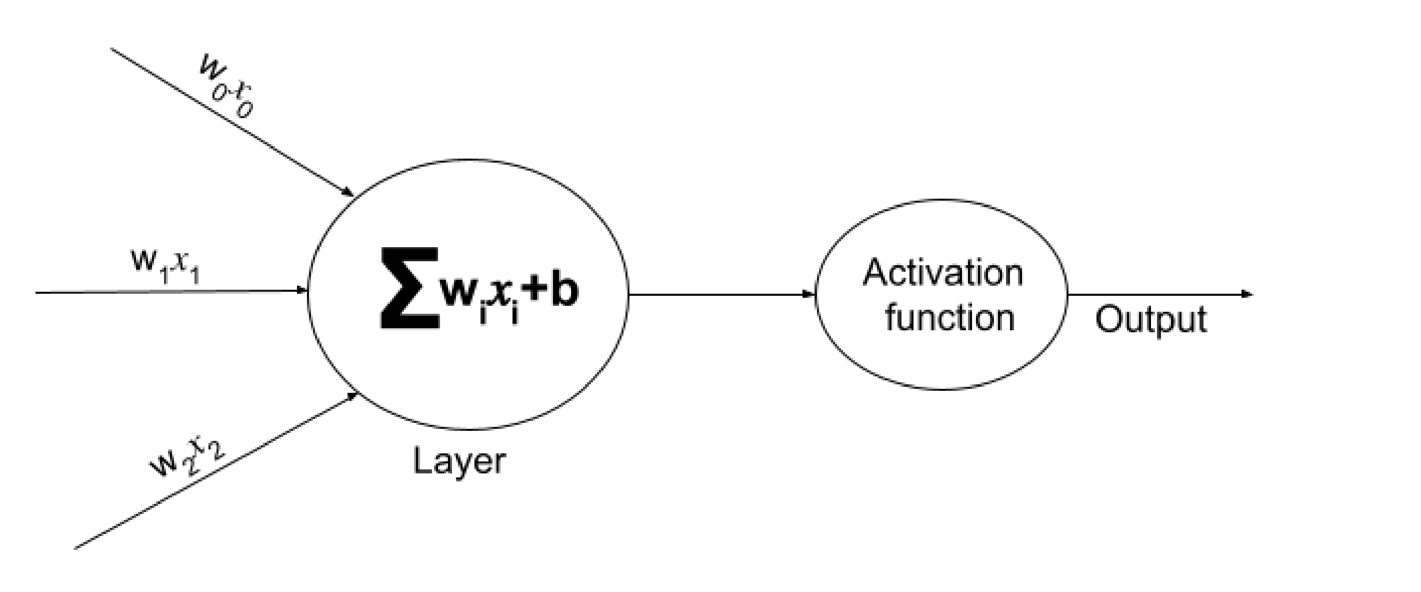


Fig. 3.1.2 Linear regression is a single-layer neural network.

Since for linear regression, every input is connected to every output (in this case there is only one output), we can regard this transformation (the output layer in Fig. 3.1.2) as a *fully-connected layer* or *dense layer*.

Activation function

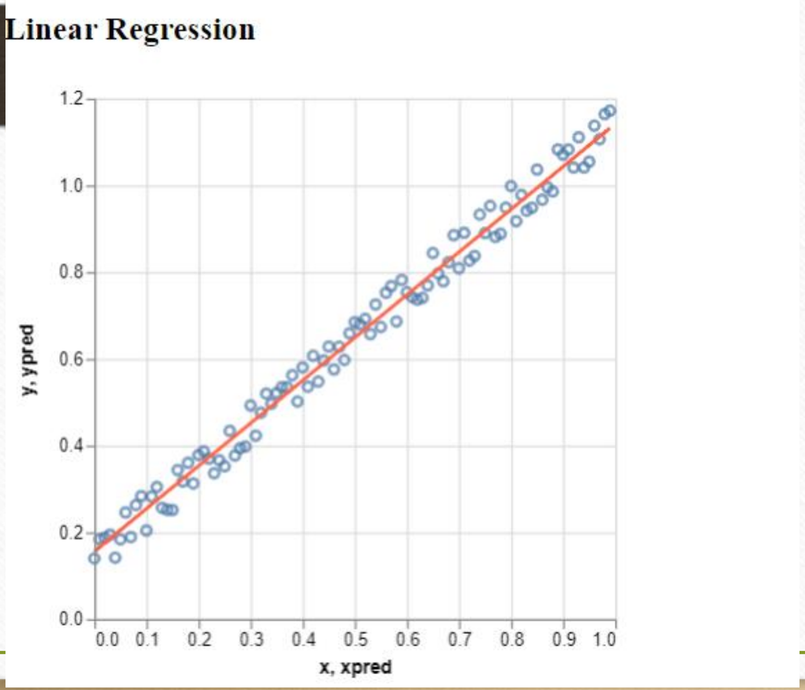


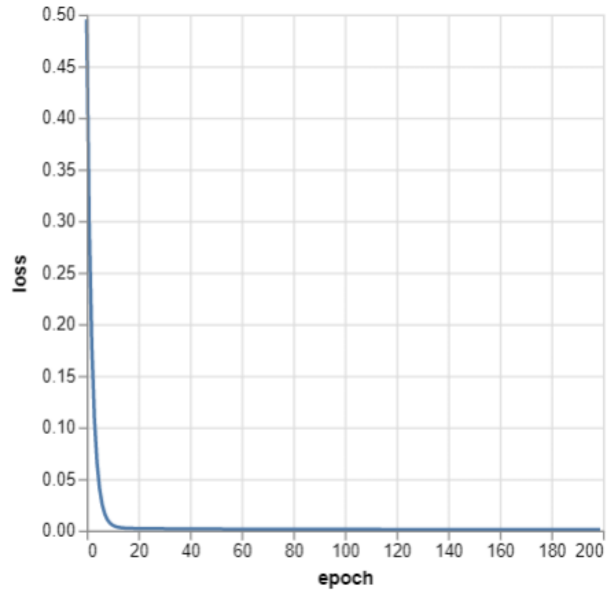


https://androidkt.com/advantages-relu-tanh-sigmoid-activation-function-deep-neural-networks/

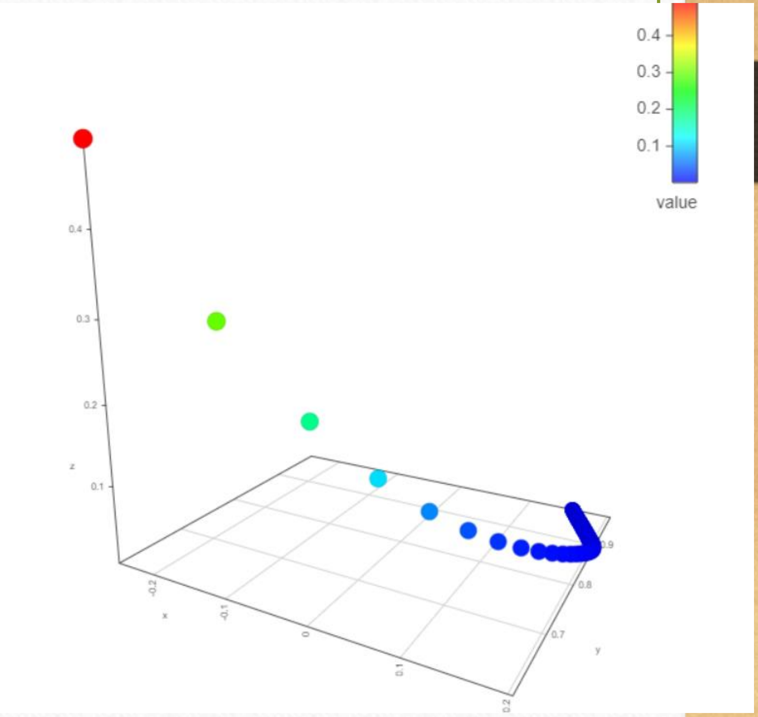
Exercise 11-1

- ex11_29_1.html (copy from ex11_29_1.txt)

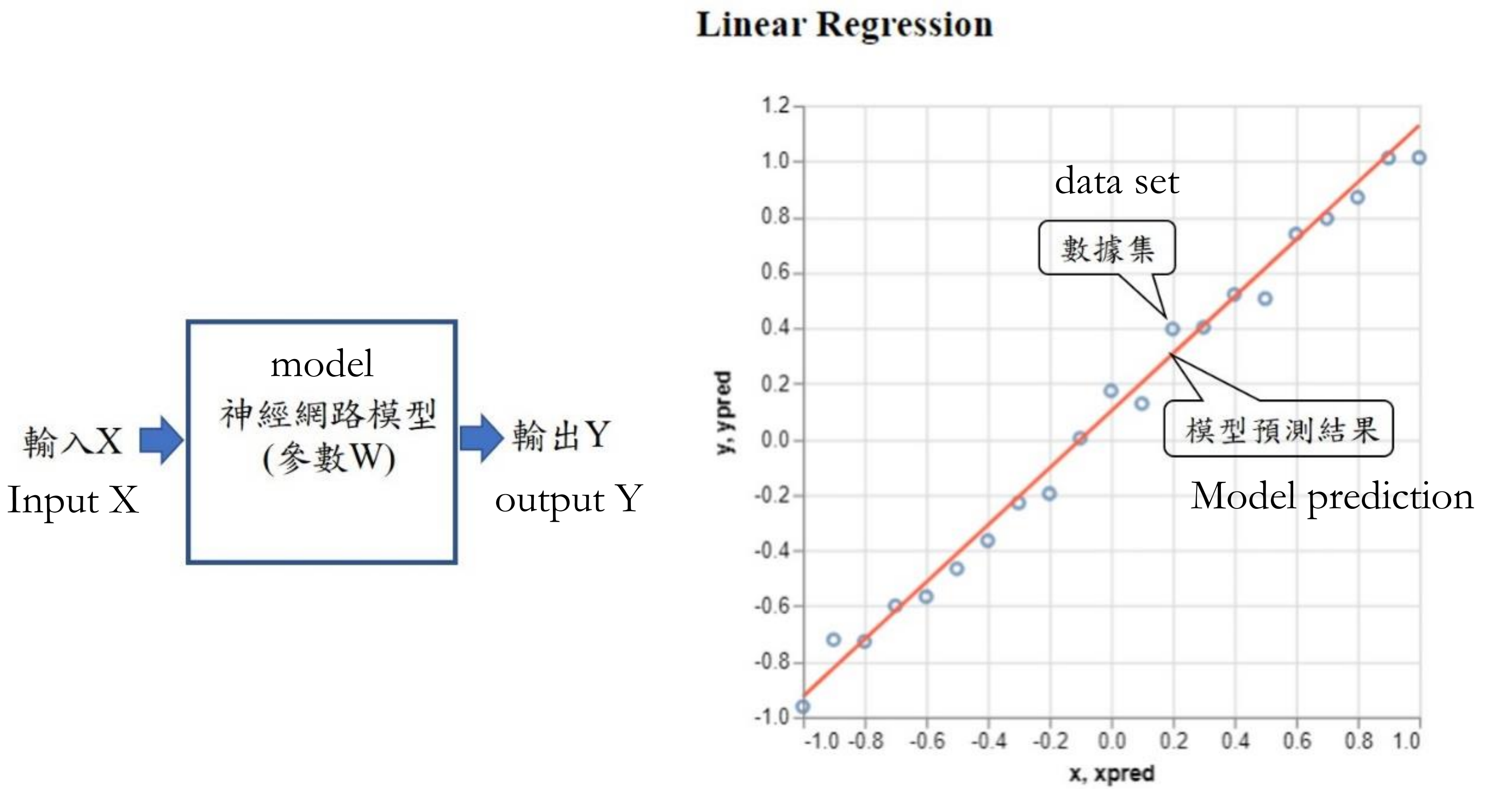




Save as SVGSave as PNGView SourceOpen in Vega Editor



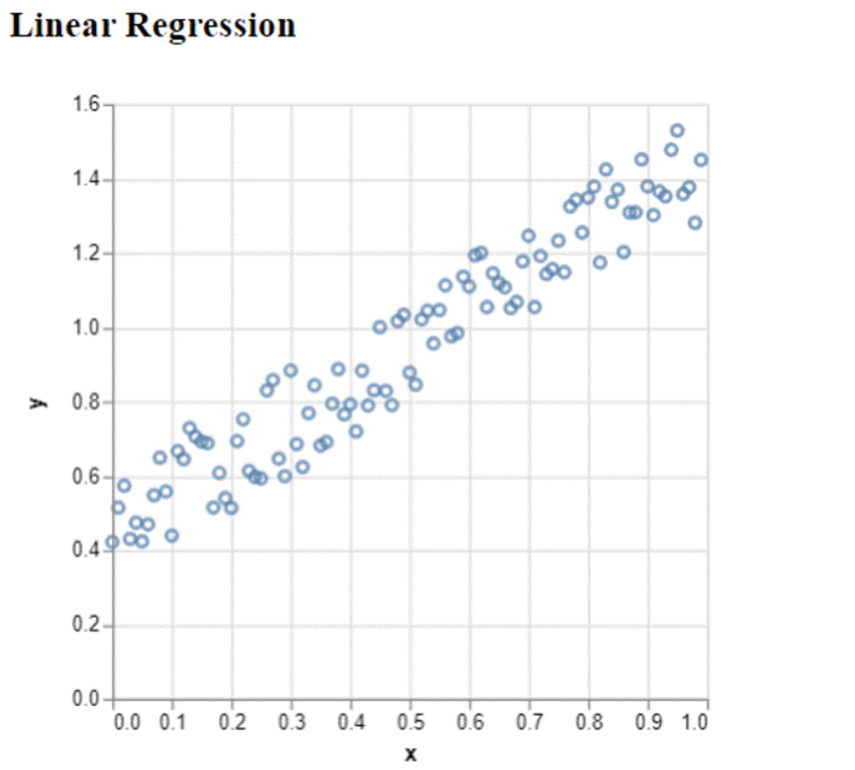
Linear Regression



coeffs = [1, 0.1];

Data set

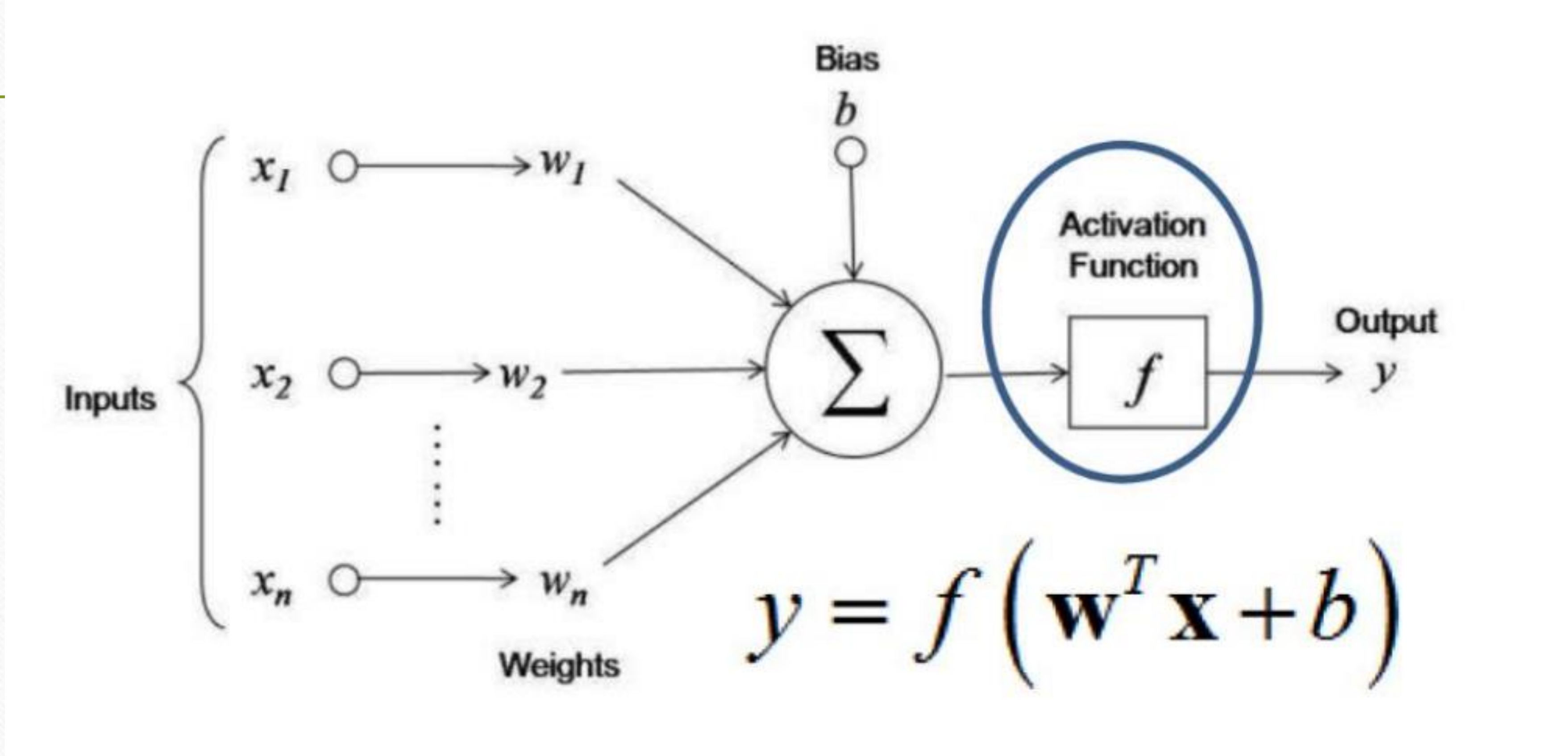
```
let y= coeffs[0] * x + coeffs[1]*(1+Math.random());
```



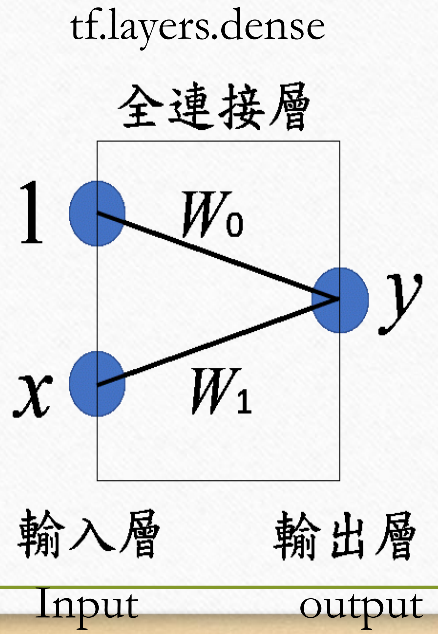
Save as SVGSave as PNGView SourceOpen in Vega Editor

Tensorflow.js

tf.layers.dense

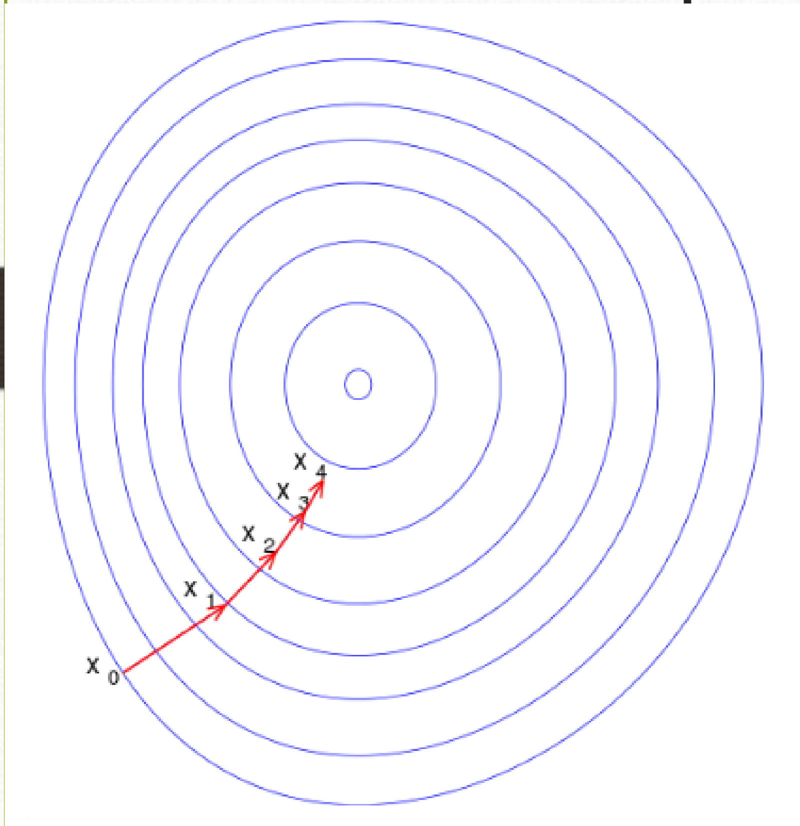


```
model = tf.sequential();//

model.add(tf.layers.dense({units: 1, inputShape: [2],
    useBias: false}));
```



$$y=1*W_0+x*W_1$$

Optimization algorithms



Gradient descent

- Gradient method

B

- Biconjugate gradient method
- Biconjugate gradient stabilized method

C

- Conjugate gradient method
- Contour currents
- Coordinate descent

D

- Derivation of the conjugate gradient method

F

- *Gradient flow*
- Frank–Wolfe algorithm

G

- Gradient descent

L

- Landweber iteration

M

- Mirror descent

N

- Nonlinear conjugate gradient method

P

- Proximal gradient method

R

- Random coordinate descent

S

- Stochastic gradient descent
- Stochastic gradient Langevin dynamics
- Stochastic variance reduction

https://en.wikipedia.org/wiki/Gradient_descent

https://en.wikipedia.org/wiki/Category:Gradient_methods

最佳化的演算法

- 梯度下降（gradient descent, GD）法，梯度下降法是一個一階找最佳解的一種方法，是希望用梯度下降法找到損失函數的最小值，如3-6圖的某模型參數座標點對應到曲面上梯度的方向是走向最大的方向，所以在梯度下降法中是往梯度的反方向走，變化模型參數往讓損失函數最小值方向移動，如式子(9)所示。
-
  - $W(t+1)= W(t)-\gamma\nabla(f)$ (9)

其中f為損失函數，$\nabla(f)$為函數f的梯度，γ為學習率(learning rate)，W(t)為在某時間點模型參數座標值，W(t+1)為調整後的模型參數座標。

Loss function

- 均方誤差(**mean-square error, MSE)函數，是各測量值誤差的平方和取平均值，以有n個量測值**y_i**與模型計算出的結果**$y_i{}^p$之均方誤差表示**如(8)所示:**
-
    - **MSE** $= \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i{}^p)^2$ **(8)**
- **其中**$y_i{}^p = 1 * W_0 + x_i * W_1$，$x_i$**為第*i*筆測試資料的*x*值，根據第i筆測試資料的x值帶入**W_0與W_1計算出的y值就是$y_i{}^p$

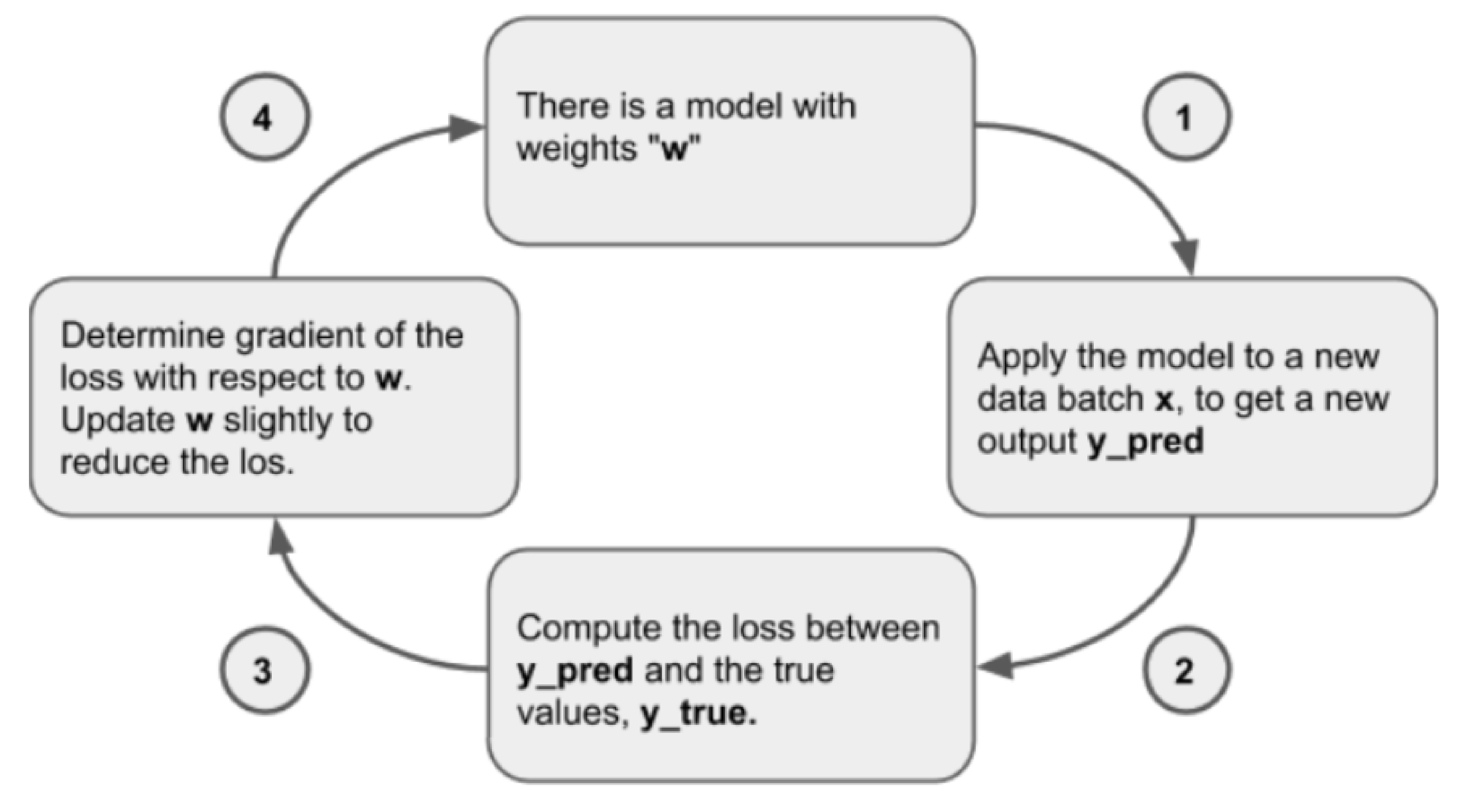
Setting Optimization algorithm & loss function

```
const learningRate = 0.01;
const sgd = tf.train.sgd(learningRate);
//'meanSquaredError
model.compile({optimizer: sgd, loss: 'meanSquaredError'});
```

Optimization algorithm

loss function

Training process



Training

```
const batchSize = 10;
 const epochs  = 200;
await model.fit( xtensor,ytensor, {
 batchSize: batchSize,
 epochs: epochs,
 callbacks: {
 onEpochEnd: async (epoch, log) => { console.log(epoch),
 console.log(log.loss); Prediction(x);
  plotloss("#vis2", log.loss, epoch);
  var W= Array.from(model.trainableWeights[0].read().dataSync());
  var style = log.loss;
  data3d.add({x:W[0],y:W[1],z:log.loss,style:style});
  drawVisualizationdot("vis3",data3d);
 }}
 });
```

Prediction

```
const xtensor = tf.tensor2d(xArrayData, [nVx.length, 2]);

xtensor.print();//xtensor

const predictOut = await model.predict(xtensor);

Ysfinal = predictOut.dataSync();
console.log('Ysfinal =', Ysfinal);

predictOut.dispose();//release GPU memory
xtensor.dispose();//release GPU memory
plotData2("#vis1", xyData[0], xyData[1], xyData[0],Ysfinal);
```

Plot functions

- function drawVisualizationdot(containerid,datadot) {

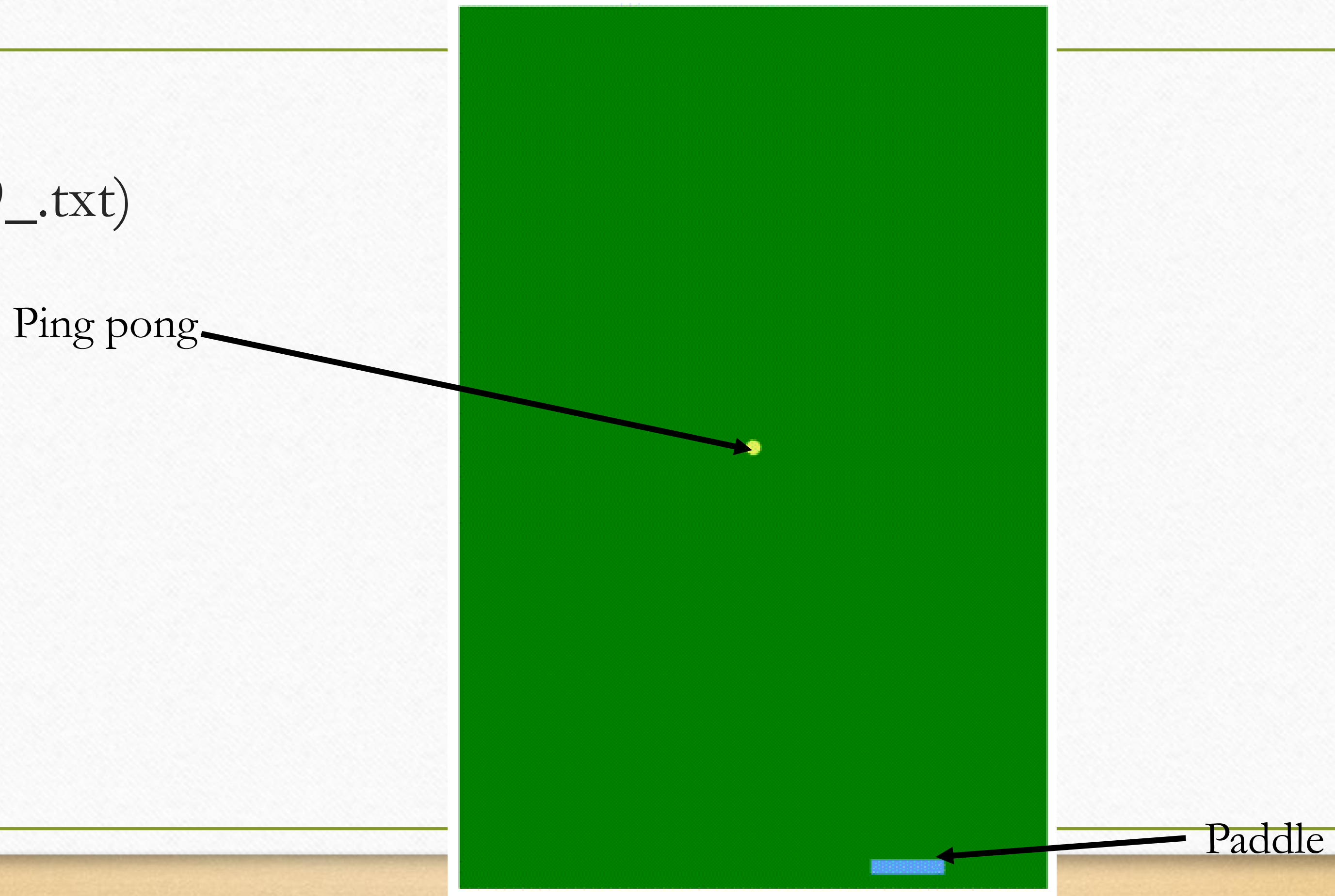
  }
- function plotloss(container, loss, epoch) {

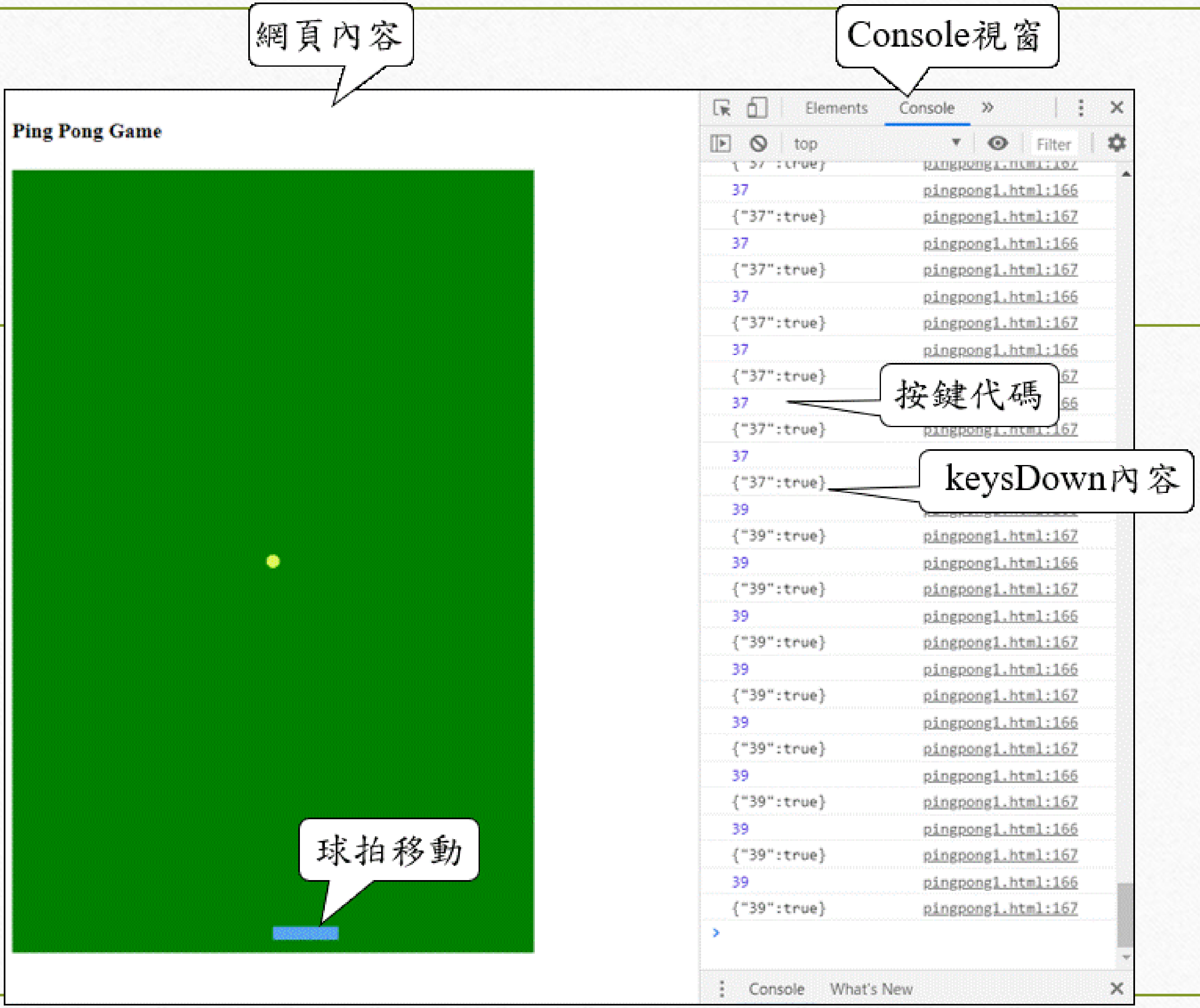
  }
- function plotData2(container, xs, ys, xspreds, yspreds) {

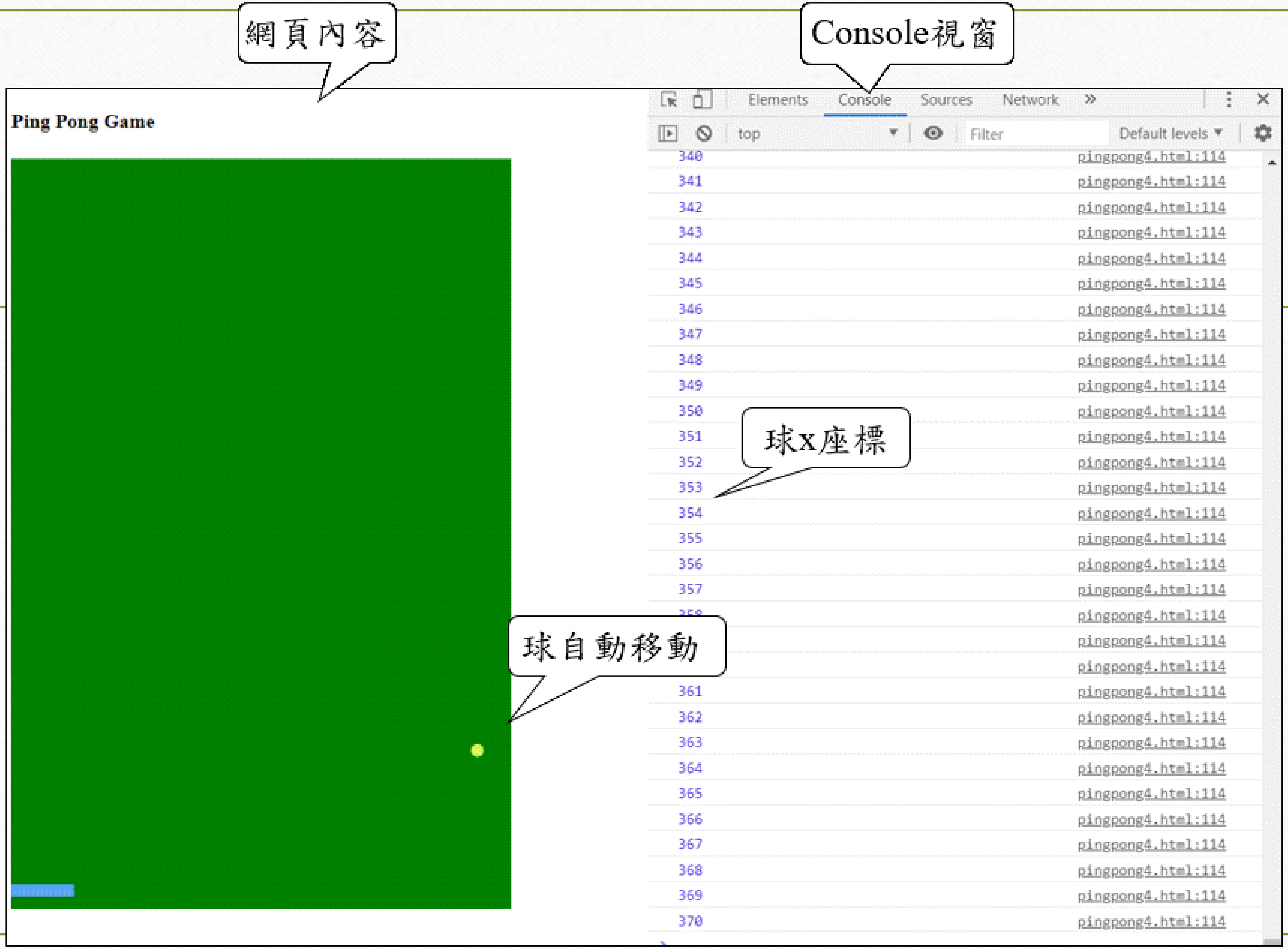
  }

Exercise 11-2

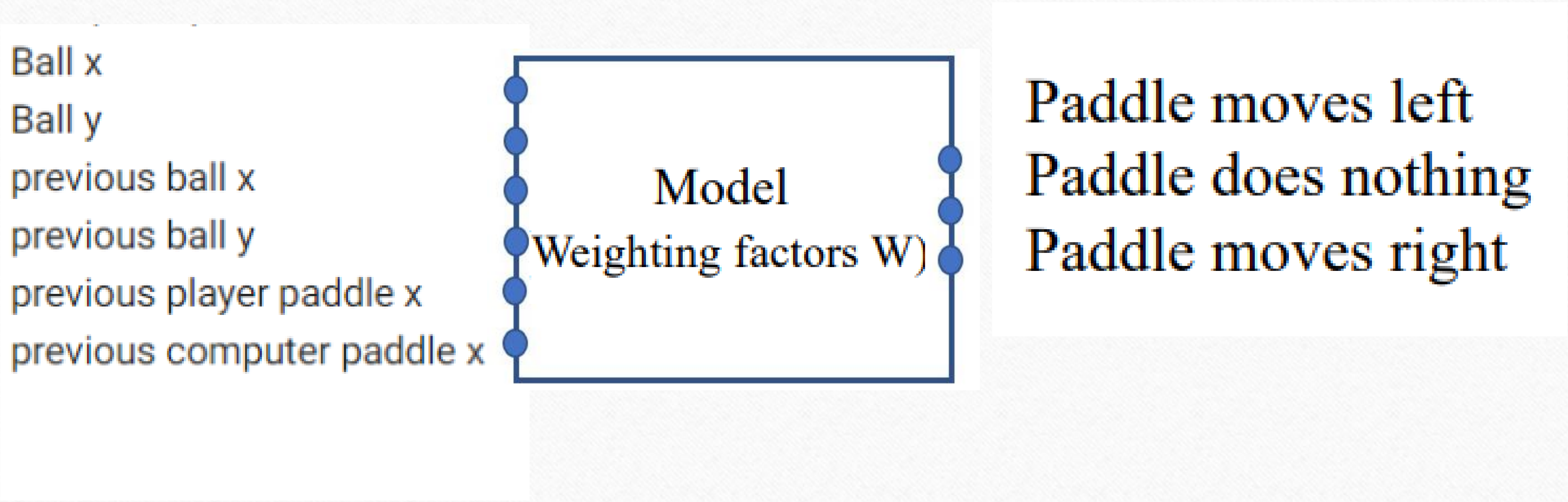
- ex11_29_1.html
- (copy from ex1129_.txt)
- Ping pong game




網頁內容
Console視窗
Ping Pong Game
Elements
Console
top
Filter
37
{"37":true}
pingpong1.html:166
pingpong1.html:167
按鍵代碼
keysDown內容
39
{"39":true}
球拍移動
What's New

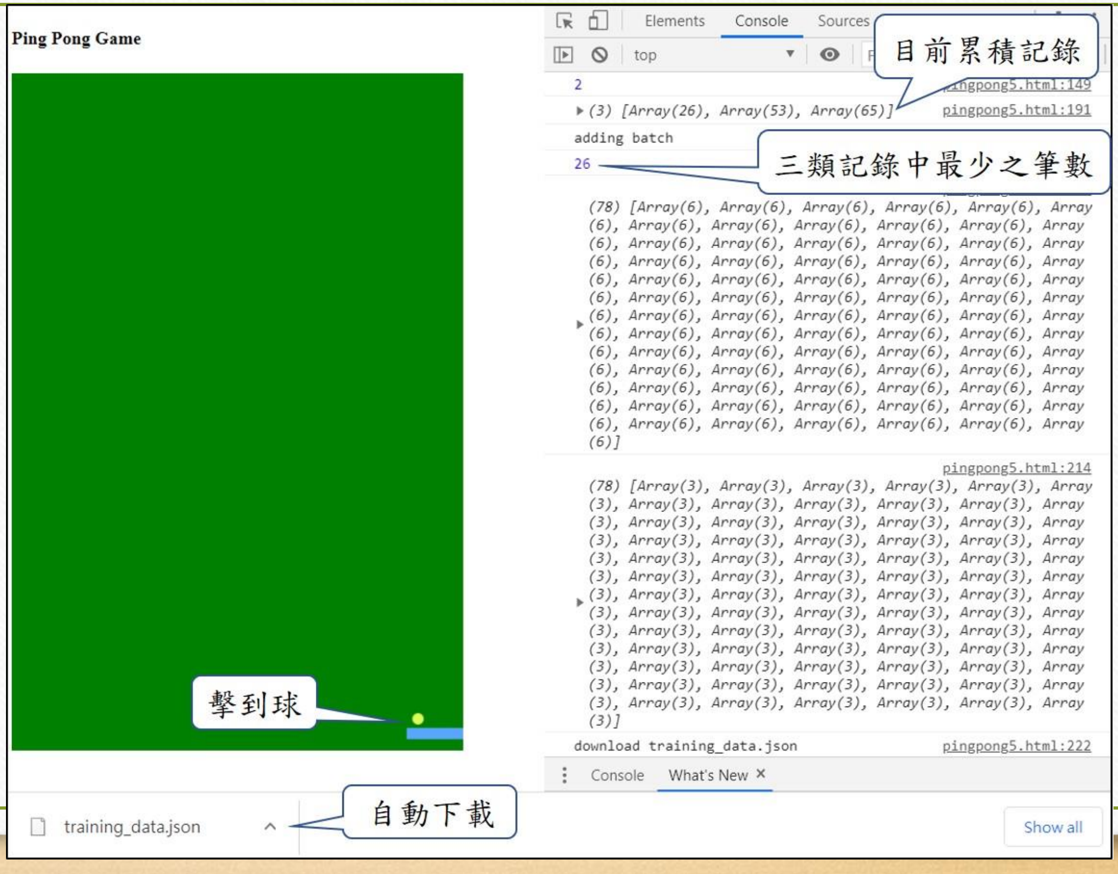

網頁內容
Console視窗
Ping Pong Game
Elements
Console
Sources
Network
top
Filter
Default levels
340 pingpong4.html:114
341 pingpong4.html:114
342 pingpong4.html:114
343 pingpong4.html:114
344 pingpong4.html:114
345 pingpong4.html:114
346 pingpong4.html:114
347 pingpong4.html:114
348 pingpong4.html:114
349 pingpong4.html:114
350 pingpong4.html:114
351 pingpong4.html:114
352 pingpong4.html:114
353 pingpong4.html:114
354 pingpong4.html:114
355 pingpong4.html:114
356 pingpong4.html:114
357 pingpong4.html:114
pingpong4.html:114
pingpong4.html:114
pingpong4.html:114
361 pingpong4.html:114
362 pingpong4.html:114
363 pingpong4.html:114
364 pingpong4.html:114
365 pingpong4.html:114
366 pingpong4.html:114
367 pingpong4.html:114
368 pingpong4.html:114
369 pingpong4.html:114
370 pingpong4.html:114
球x座標
球自動移動

Model


Ball x
Ball y
previous ball x
previous ball y
previous player paddle x
previous computer paddle x
Model
Weighting factors W)
Paddle moves left
Paddle does nothing
Paddle moves right

Exercise 11-3

- ex11_29_3.html
- (copy from ex11_29_3.txt)
- Record playing data



{"xs":[[272,328,188,264,320,180],[264,320,180,256,312,172],[256,312,172,248,30
4,164],[248,304,164,240,296,156],[240,296,156,232,288,148],[232,288,148,224,28
0,140],[224,280,140,216,272,132],[216,272,132,208,264,124],[208,264,124,200,25
6,116],[200,256,116,192,248,108],[192,248,108,184,240,100],[184,240,100,176,23
2,92],[176,232,92,168,224,84],[168,224,84,160,216,76],[160,216,76,152,208,68],[1
52,208,68,144,200,60],[144,200,60,136,192,52],[136,192,52,128,184,44],[128,184,
44,120,176,36],[120,176,36,112,168,28],[112,168,28,104,160,20],[104,160,20,96,1
52,12],[96,152,12,88,144,4],[88,144,4,80,136,12],[80,136,12,72,128,20],[72,128,20
,64,120,28],[40,88,572,40,96,564],[40,96,564,40,104,556],[40,104,556,40,112,548]
,[40,112,548,40,120,540],[40,120,540,40,128,532],[40,128,532,40,136,524],[40,13
6,524,40,144,516],[40,144,516,40,152,508],[40,152,508,40,160,500],[40,160,500,4
0,168,492],[40,168,492,40,176,484],[40,176,484,40,184,476],[272,384,244,272,37
6,236],[272,376,236,272,368,228],[272,368,228,272,360,220],[272,360,220,272,35
2,212],[272,352,212,272,344,204],[272,344,204,272,336,196],[272,336,196,272,32
8,188],[64,120,28,64,112,36],[64,112,36,64,104,44],[64,104,44,64,96,52],[64,96,52
,64,88,60],[64,88,60,64,80,68],[64,80,68,64,72,76],[64,72,76,64,64,84],[40,184,476
,48,192,468],[48,192,468,56,200,460],[56,200,460,64,208,452],[64,208,452,72,216
,444],[72,216,444,80,224,436],[80,224,436,88,232,428],[88,232,428,96,240,420],[9
6,240,420,104,248,412],[104,248,412,112,256,404],[112,256,404,120,264,396],[12
0,264,396,128,272,388],[128,272,388,136,280,380],[136,280,380,144,288,372],[14
4,288,372,152,296,364],[152,296,364,160,304,356],[160,304,356,168,312,348],[16
8,312,348,176,320,340],[176,320,340,184,328,332],[184,328,332,192,336,324],[19
2,336,324,200,344,316],[200,344,316,208,352,308],[208,352,308,216,360,300],[21
6,360,300,224,368,292],[224,368,292,232,376,284],[232,376,284,240,384,276],[24
0,384,276,248,392,268]],"ys":[[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[
1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,
0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[1,0,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[
0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,
0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,1,0],[0,0,1],[

```
console.log(len);
var data_xx=[];
var data_yy=[];

if (len > 10){
    for(i = 0; i < 3; i++){
     data_xx.push(...training_data[i].slice(0, len));
           // trims training data to 'len' length

     data_yy.push(...Array(len).fill([i==0?1:0, i==1?1:0
     , i==2?1:0]));    // creates 'len' number records
     of embedding data
```

Blob

```
 var a = document.createElement("a");
// var a = document.getElementById("a");
 var file = new Blob([JSON.stringify({xs: data_xx, ys
 : data_yy})], {type: 'application/json'});
 a.href = URL.createObjectURL(file);
 a.download = 'training_data.json';
 a.click();
 console.log('download training_data.json');
 //印出文字'download training_data.json'
```

Homework 11-1

- change code: when the length of the record is larger than 100, the json file will be downloaded automatically。

```
console.log(len);
var data_xx=[];
var data_yy=[];

if (len > 10){
   for(i = 0; i < 3; i++){
    data_xx.push(...training_data[i].slice(0, len));
         // trims training data to 'len' length

    data_yy.push(...Array(len).fill([i==0?1:0, i==1?1:0
    , i==2?1:0]));    // creates 'len' number records
```

Classification Problem

- y∈{dog,cat,chicken}   => y∈{1,2,3}   => y∈{(1,0,0),(0,1,0),(0,0,1)}
- In our case, a label yy would be a three-dimensional vector, with (1,0,0)corresponding to “cat”, (0,1,0) to “chicken”, and (0,0,1) to “dog”:

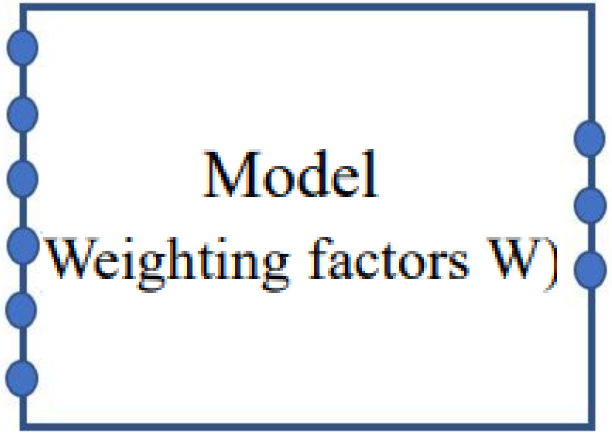
- y∈{baby,toddler,adolescent,young adult,geriatric} => y∈{1,2,….?}   => y∈{????}

Classification

Classification	Label
Paddle move left	[1,0,0]
Paddle does nothing	[0,1,0]
Paddle move right	[0,0,1]

Model

Ball x
Ball y
previous ball x
previous ball y
previous player paddle x
previous computer paddle x



Paddle moves left
Paddle does nothing
Paddle moves right

Network Architecture

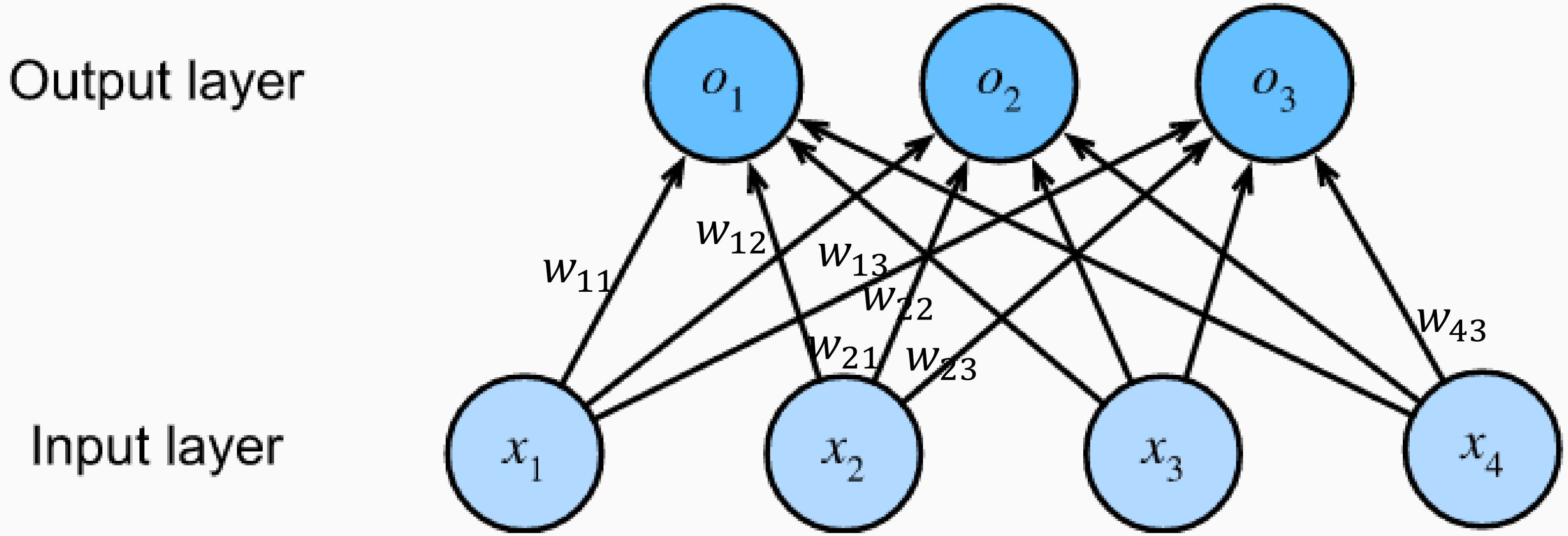


Fig. 3.4.1 Softmax regression is a single-layer neural network.

$$\mathbf{o} = \mathbf{W}\mathbf{x} + \mathbf{b},$$

Network Archite

$$\mathbf{o} = \mathbf{W}\mathbf{x} + \mathbf{b},$$

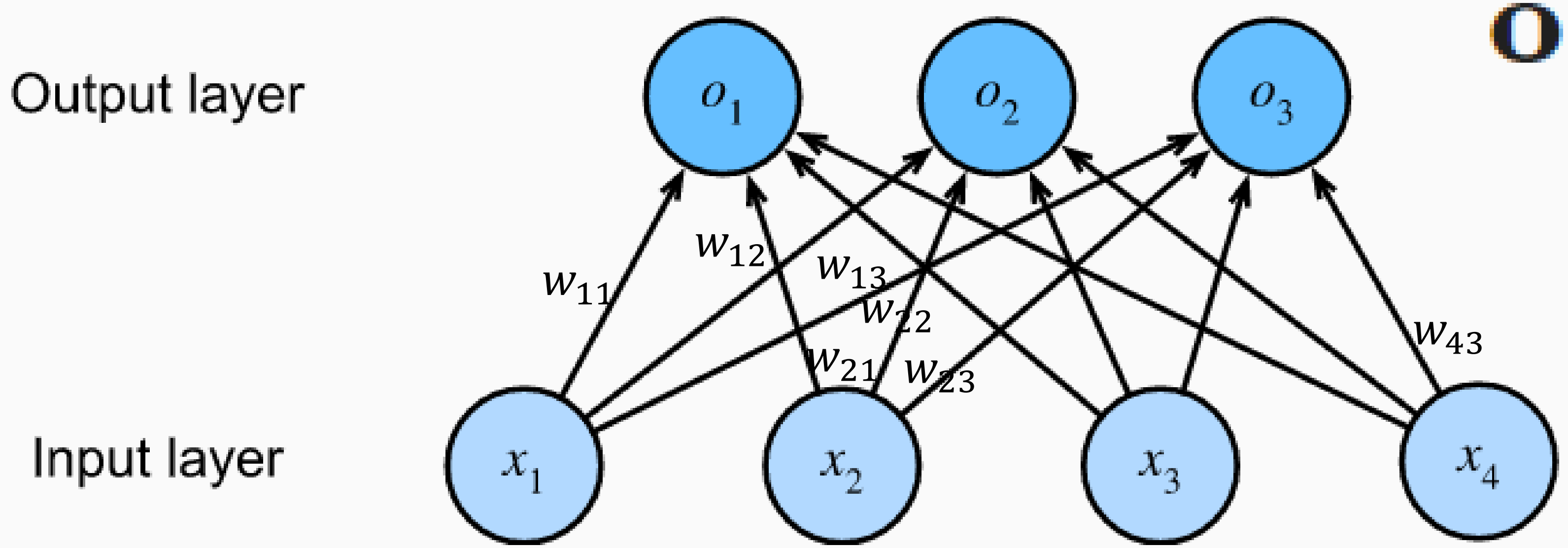


Fig. 3.4.1 Softmax regression is a single-layer neural network.

$$
\begin{aligned}
o_1 &= x_1 w_{11} + x_2 w_{12} + x_3 w_{13} + x_4 w_{14} + b_1,\\
o_2 &= x_1 w_{21} + x_2 w_{22} + x_3 w_{23} + x_4 w_{24} + b_2,\\
o_3 &= x_1 w_{31} + x_2 w_{32} + x_3 w_{33} + x_4 w_{34} + b_3.
\end{aligned}
$$

$o_1 = w_{11}x_1 + w_{12}x_2 + w_{13}\, x_3 + w_{14}x_4 + b_1$

$o_2 = ?$

$o_3 = ?$

Softmax Operation

$$\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{o}) \quad \text{where} \quad \hat{y}_j = \frac{\exp(o_j)}{\sum_k \exp(o_k)}. \tag{3.4.3}$$

It is easy to see $\hat{y}_1 + \hat{y}_2 + \hat{y}_3 = 1$ with $0 \leq \hat{y}_j \leq 1$ for all j. Thus, $\hat{\mathbf{y}}$ is a proper probability distribution whose element values can be interpreted accordingly.

we can still pick out the most likely class by

$$\operatorname*{argmax}_j \hat{y}_j = \operatorname*{argmax}_j o_j. \tag{3.4.4}$$

Although softmax is a nonlinear function, the outputs of softmax regression are still *determined* by an affine transformation of input features; thus, softmax regression is a linear model.

Vectorization for Minibatches

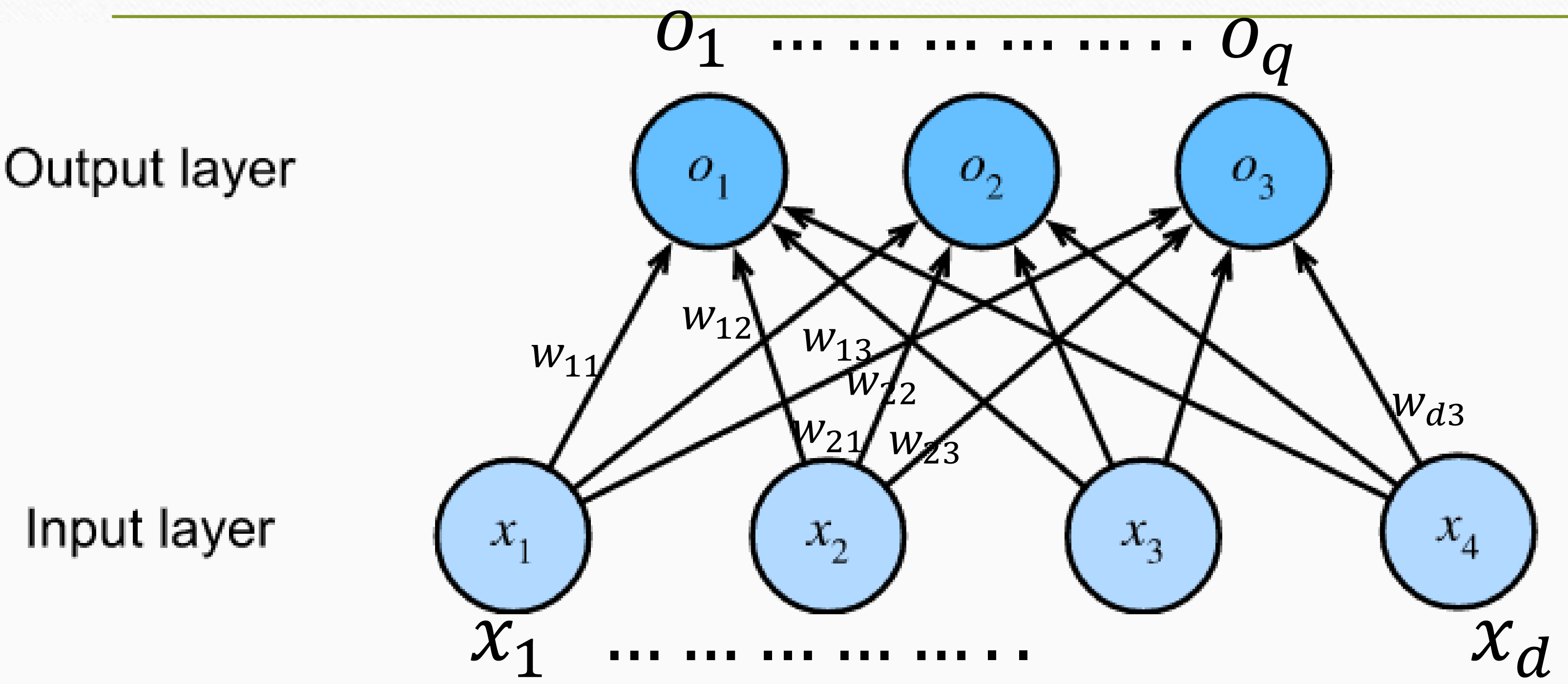


Fig. 3.4.1 Softmax regression is a single-layer neural network.

$\mathbf{X} \in \mathbb{R}^{n \times d}$.

$\mathbf{W} \in \mathbb{R}^{d \times q}$

$$\mathbf{O} = \mathbf{X}\mathbf{W} + \mathbf{b},$$
$$\hat{\mathbf{Y}} = \mathrm{softmax}(\mathbf{O}).$$

Multilayer Perceptrons

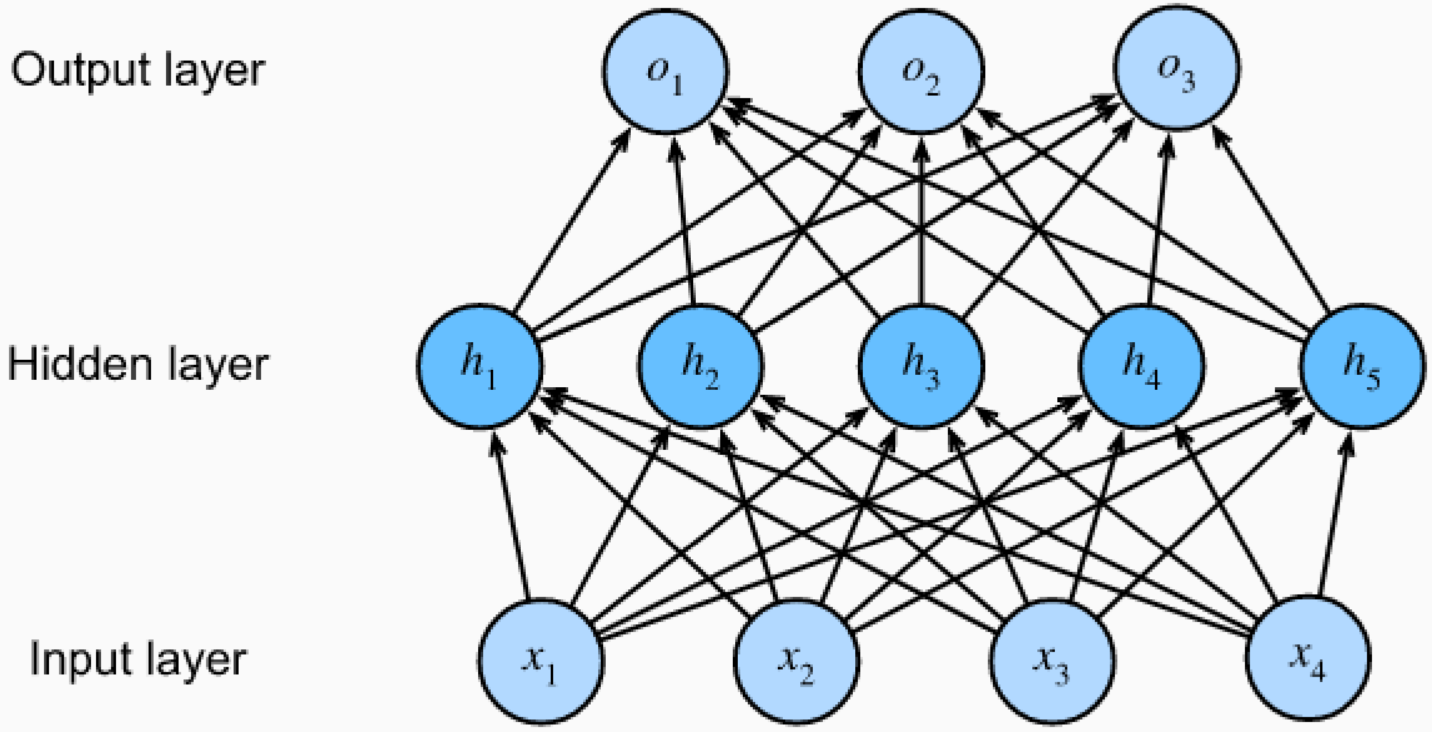


Fig. 4.1.1 An MLP with a hidden layer of 5 hidden units.

Multilayer Perceptrons

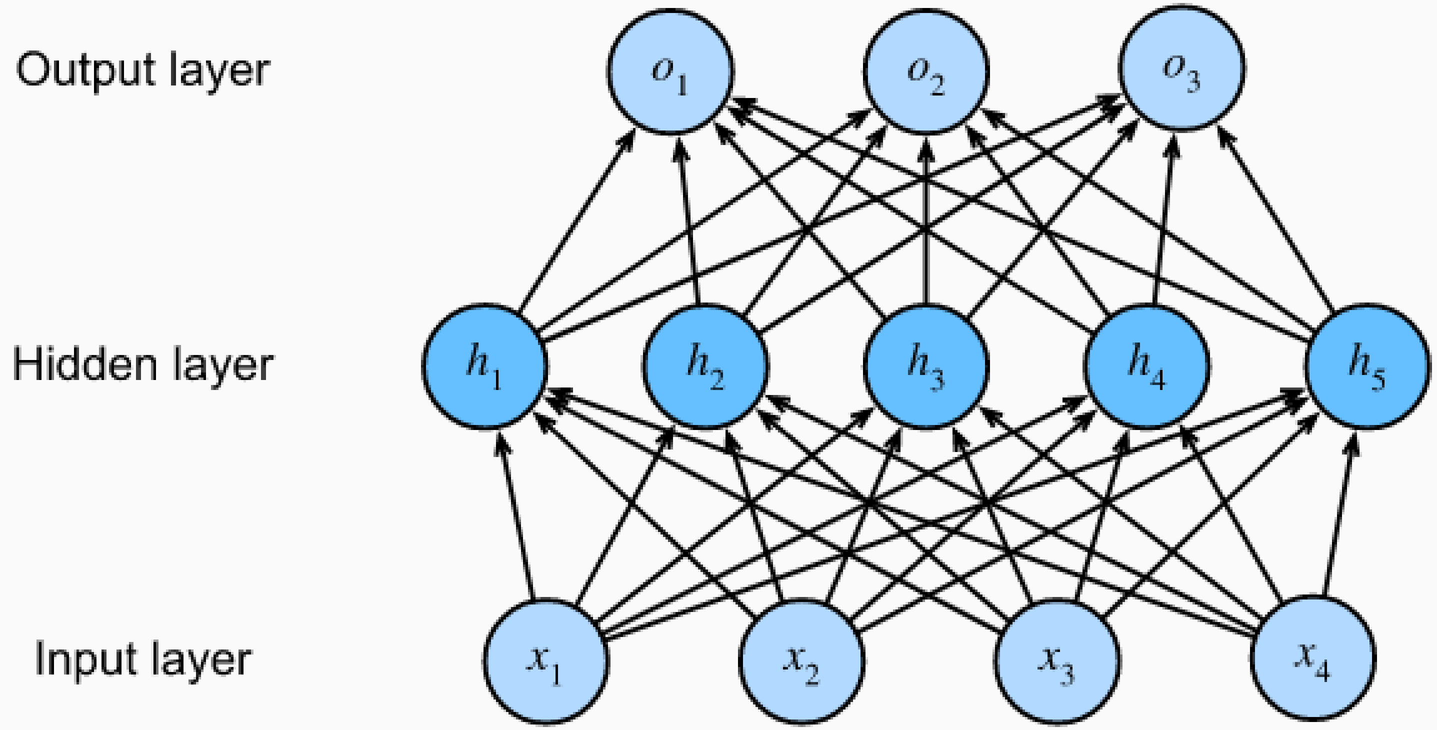


Fig. 4.1.1 An MLP with a hidden layer of 5 hidden units.

$$\mathbf{H} = \mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)},$$

Multilayer Perceptrons

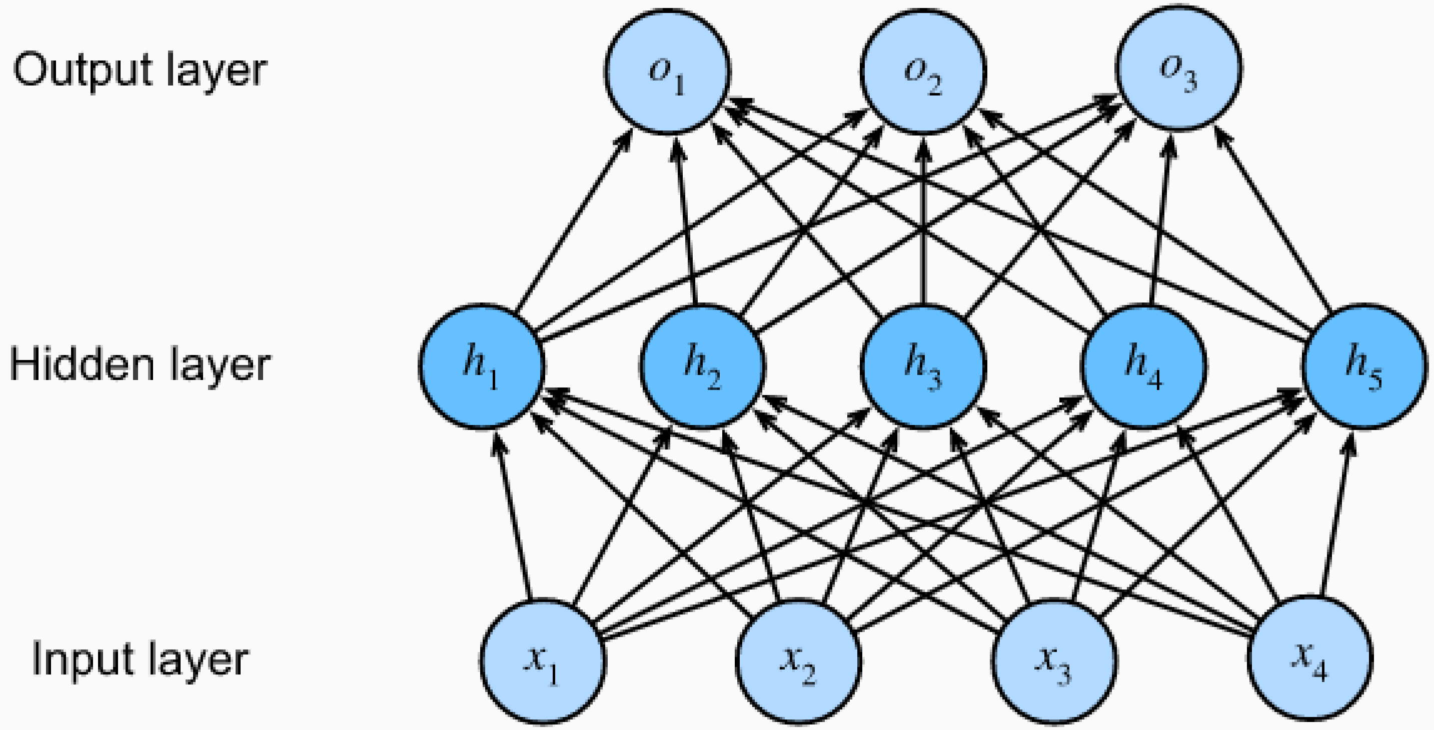


Fig. 4.1.1 An MLP with a hidden layer of 5 hidden units.

$$\mathbf{O} = \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}.$$

$$\mathbf{H} = \mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)},$$

From Linear to Nonlinear

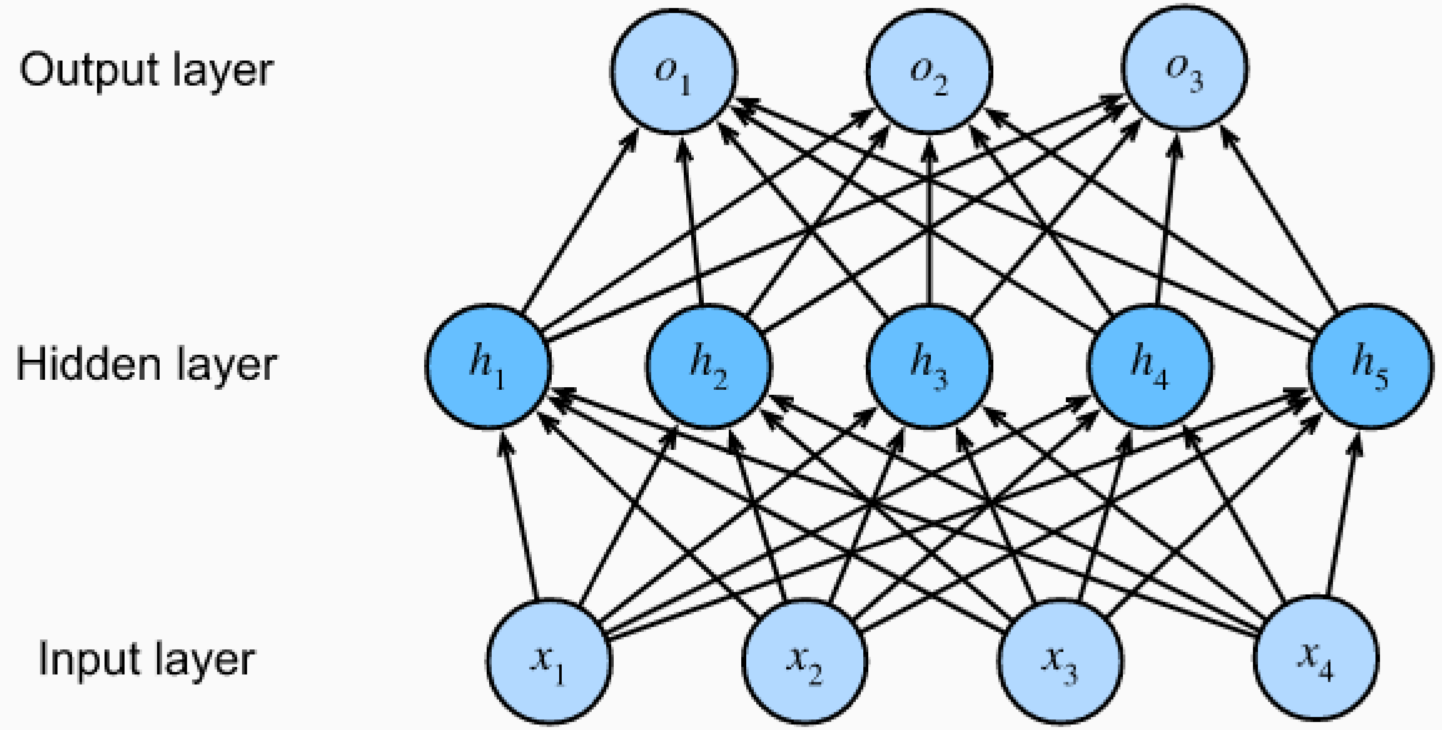


Fig. 4.1.1 An MLP with a hidden layer of 5 hidden units.

$$\mathbf{H} = \sigma(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}),$$

From Linear to Nonlinear

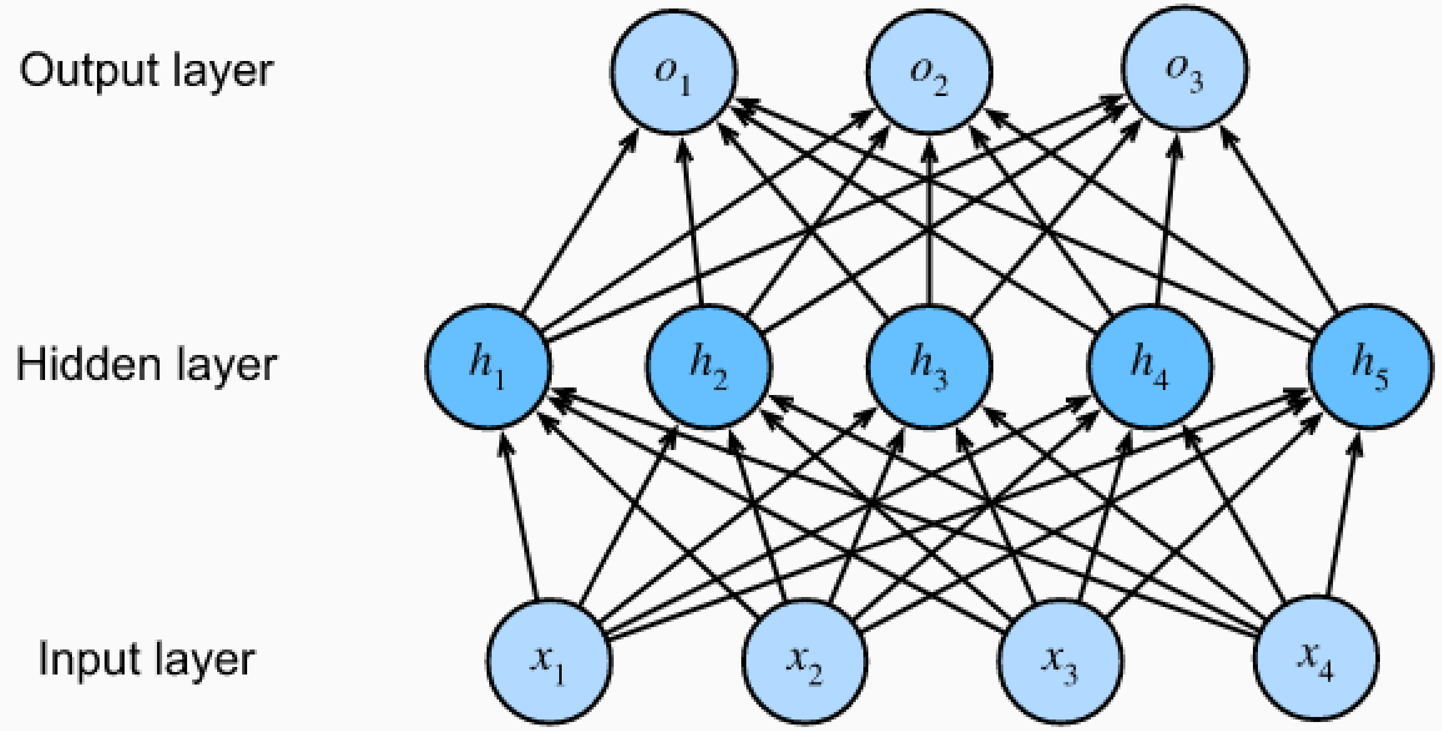


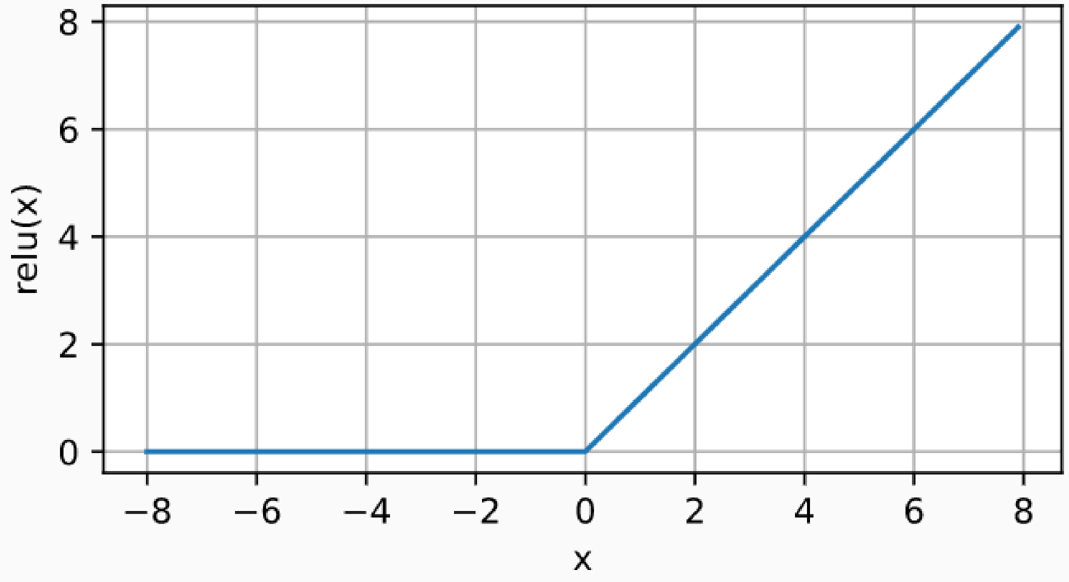
Fig. 4.1.1 An MLP with a hidden layer of 5 hidden units.

$$\mathbf{O} = \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}.$$

$$\mathbf{H} = \sigma(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}),$$

Activation Functions

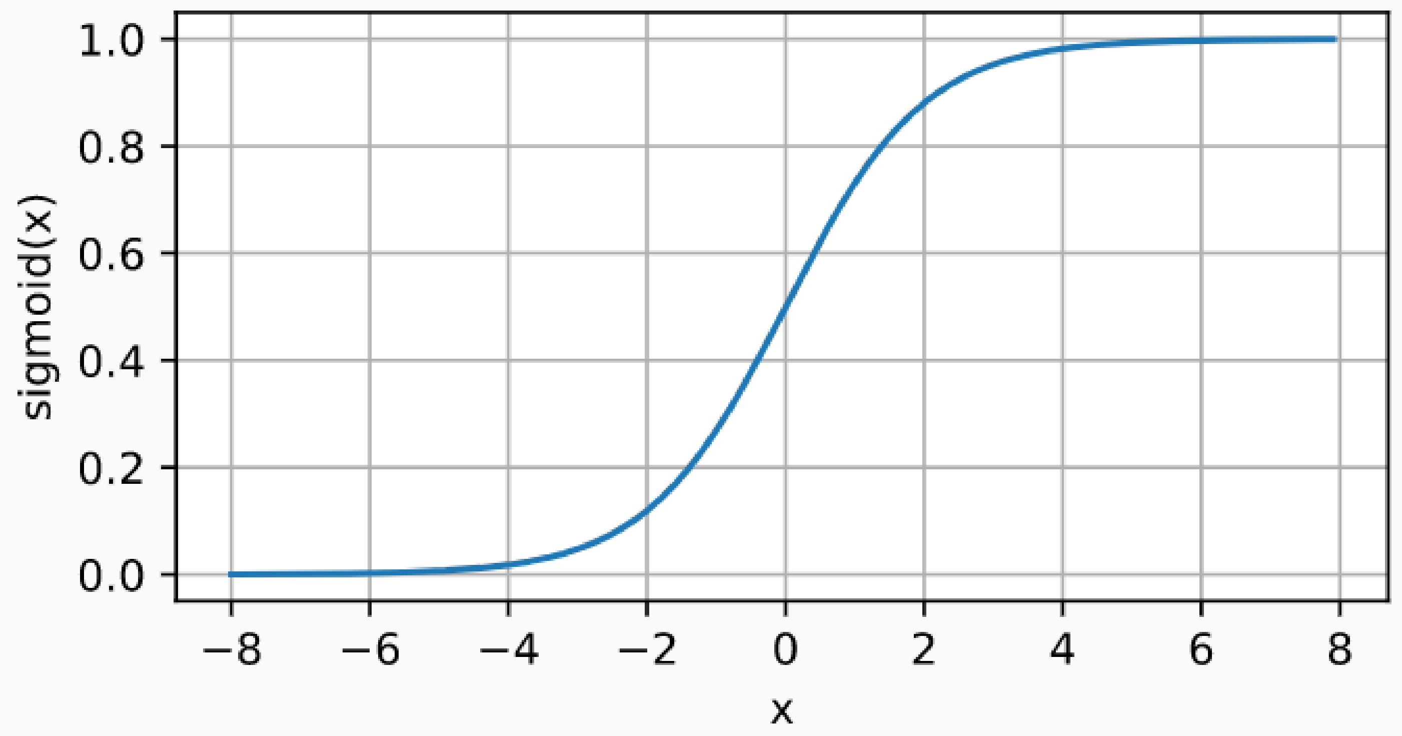
Activation Functions



$$\mathrm{ReLU}(x) = \max(x, 0).$$

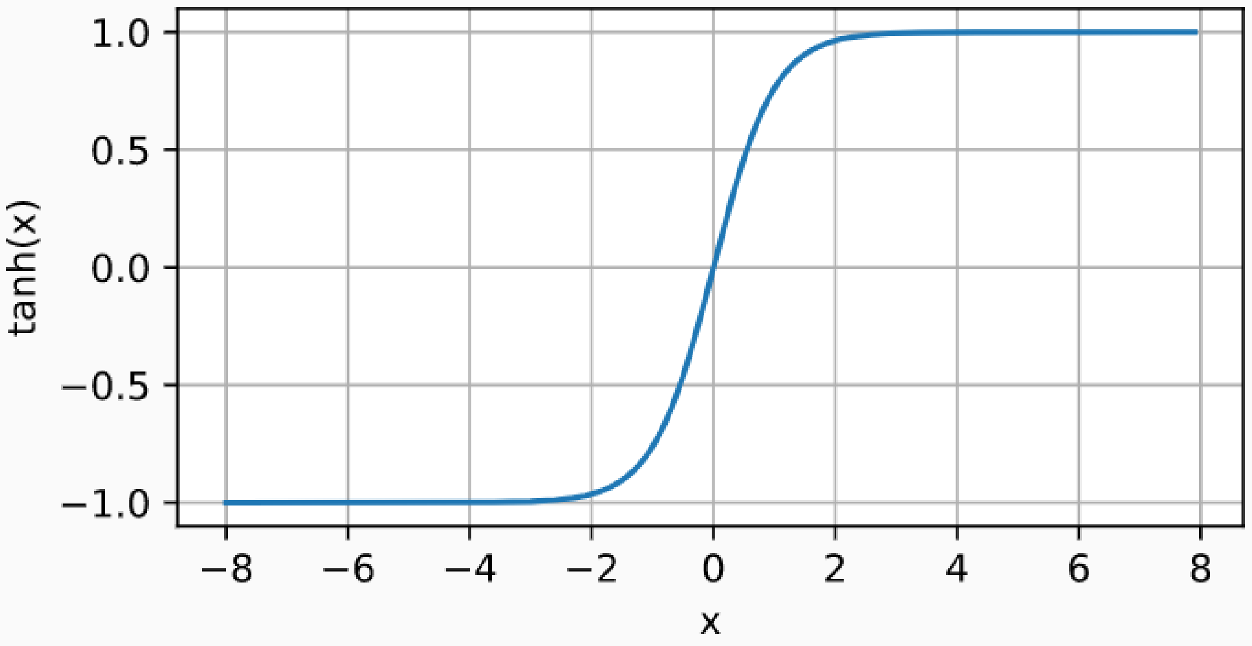
Sigmoid Function

$$\mathrm{sigmoid}(x) = \frac{1}{1+\exp(-x)}.$$

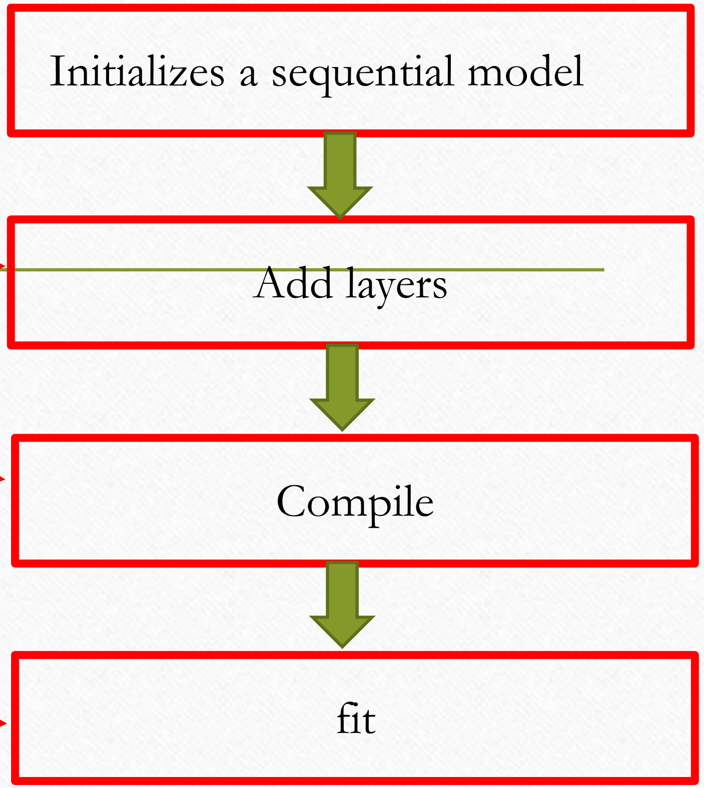


Tanh Function

$$\tanh(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}.$$

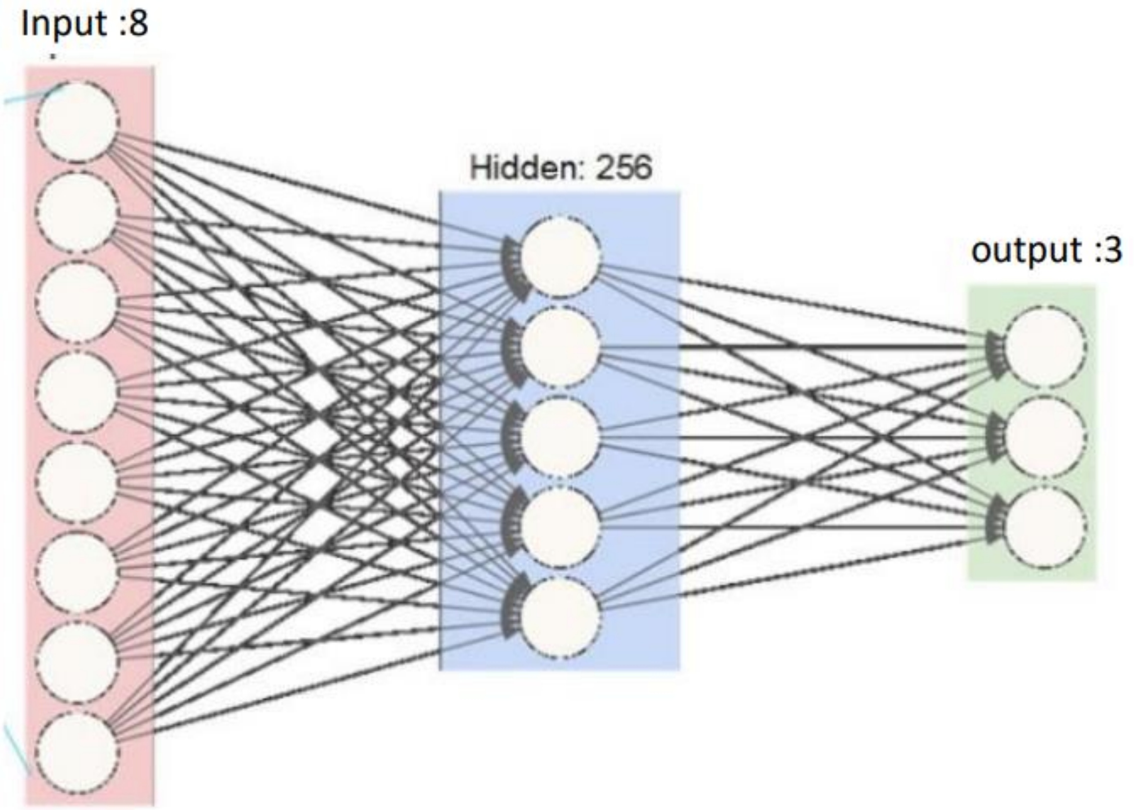


```
// Initializes a sequential model
const model = tf.sequential();
// The line below needs curly braced inputs (note the syntax similarity with TF K
// We are using 10 units and ReLU activation. The input shape is a 28 by 28 monoc
model.add(tf.layers.dense({units:64,inputShape:[28,28,1],activation:'relu'}));
// This line compiles the model quite similar to model.compile in TensorFlow, wit
model.compile({
  optimizer:'adam',
  loss:'categoricalCrossentropy'
});
// This line fits the model on the dataset, which is currently stored in a tensor
await model.fit(xs,ys,{
  epochs: 100,
  callbacks: { // The line below may look scary, but all it does it prints the lo
    onEpochEnd: async(epoch,logs) =>{
          console.log("Epoch :" + epoch + " Loss:" + logs.loss);
    }
  }
});
// Done!
```

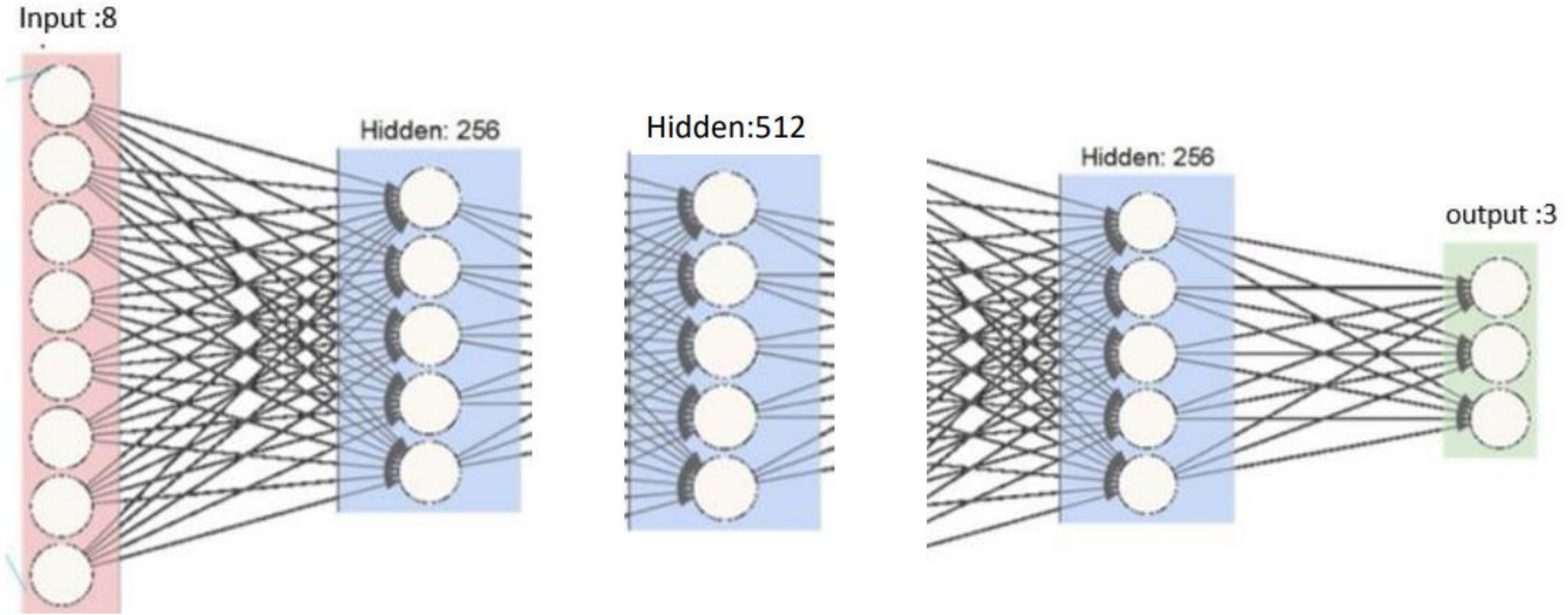


model

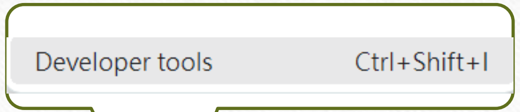
```
const model = tf.sequential();
model.add(tf.layers.dense({units: 256, inputShape: [8]}));
model.add(tf.layers.dense({units: 3, inputShape: [256]}));
//returns a 1x3
```



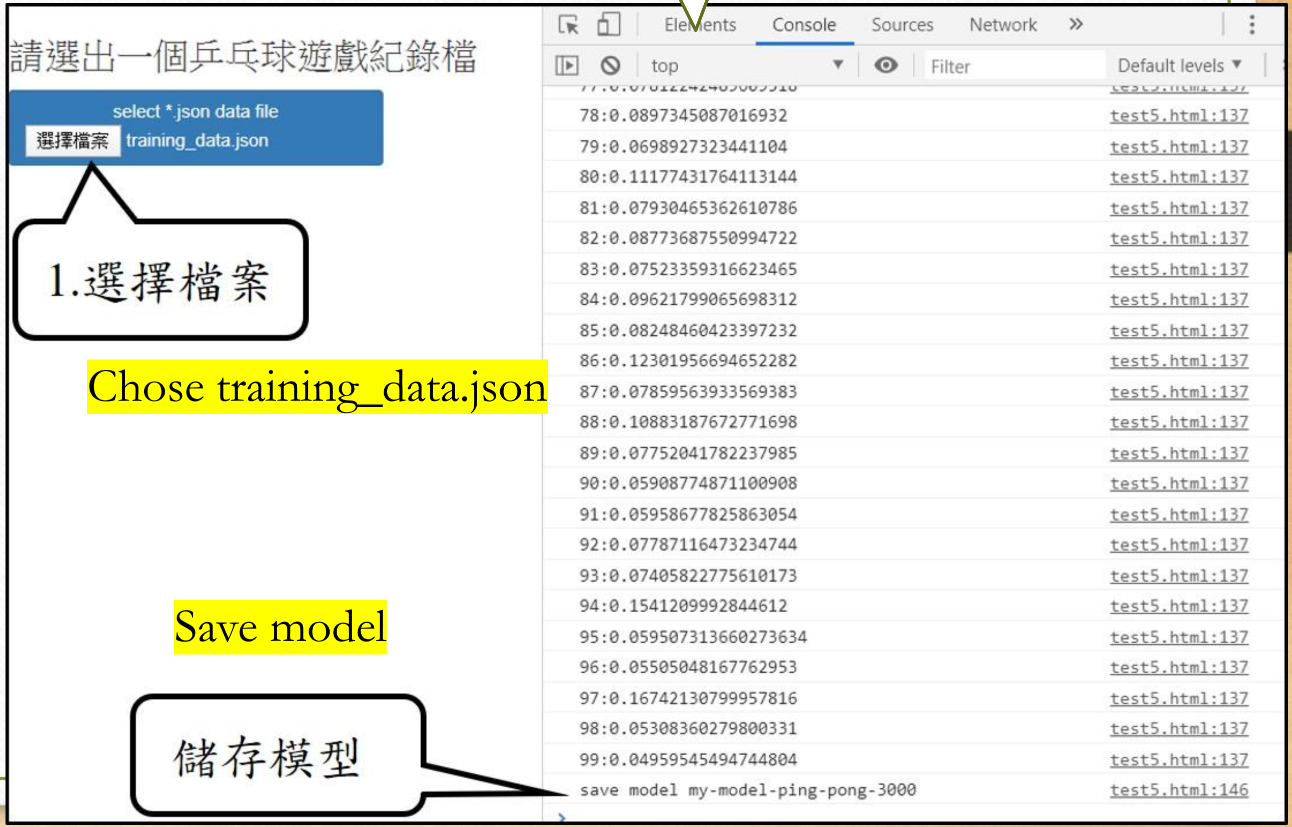
```
// initial model definition
const model = tf.sequential();
model.add(tf.layers.dense({units: 256, inputShape: [8]})); //input is a 1x8
model.add(tf.layers.dense({units: 512, inputShape: [256]}));
model.add(tf.layers.dense({units: 256, inputShape: [512]}));
model.add(tf.layers.dense({units: 3, inputShape: [256]})); //returns a 1x3
```



Exercise 11-4



- ex11_29_4.html
- (copy from ex11_29_4.txt)
- Train model



model

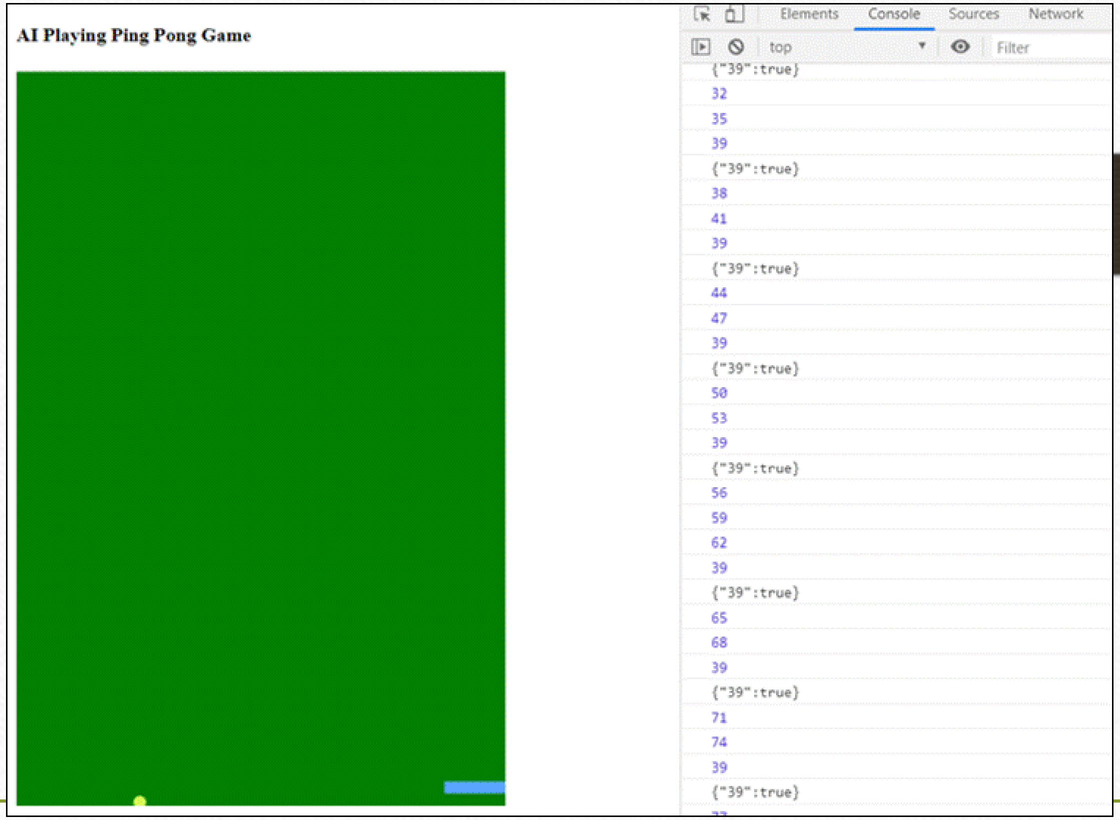
```
model = tf.sequential();
model.add(tf.layers.dense({units: 64,activation:'relu',
inputShape: [6]})); //input is a 1x8
model.add(tf.layers.dropout(0.5));
model.add(tf.layers.dense({units: 64,activation:'relu'}));
model.add(tf.layers.dropout(0.5));
model.add(tf.layers.dense({units: 3,activation:'softmax'}));
```

Setting Optimization algorithm & loss function

```
// set optimiser and compile model
 const learningRate = 0.001;
 const optimizer = tf.train.adam(learningRate);
 model.compile({loss: 'categoricalCrossentropy', optimizer:
 optimizer, metrics: ['accuracy']});
 console.log( 'compile finished');
```

Exercise 11-5

- ex11_29_5.html
- (copy from ex11_29_5.txt)
- AI play ping pong game



Model summary

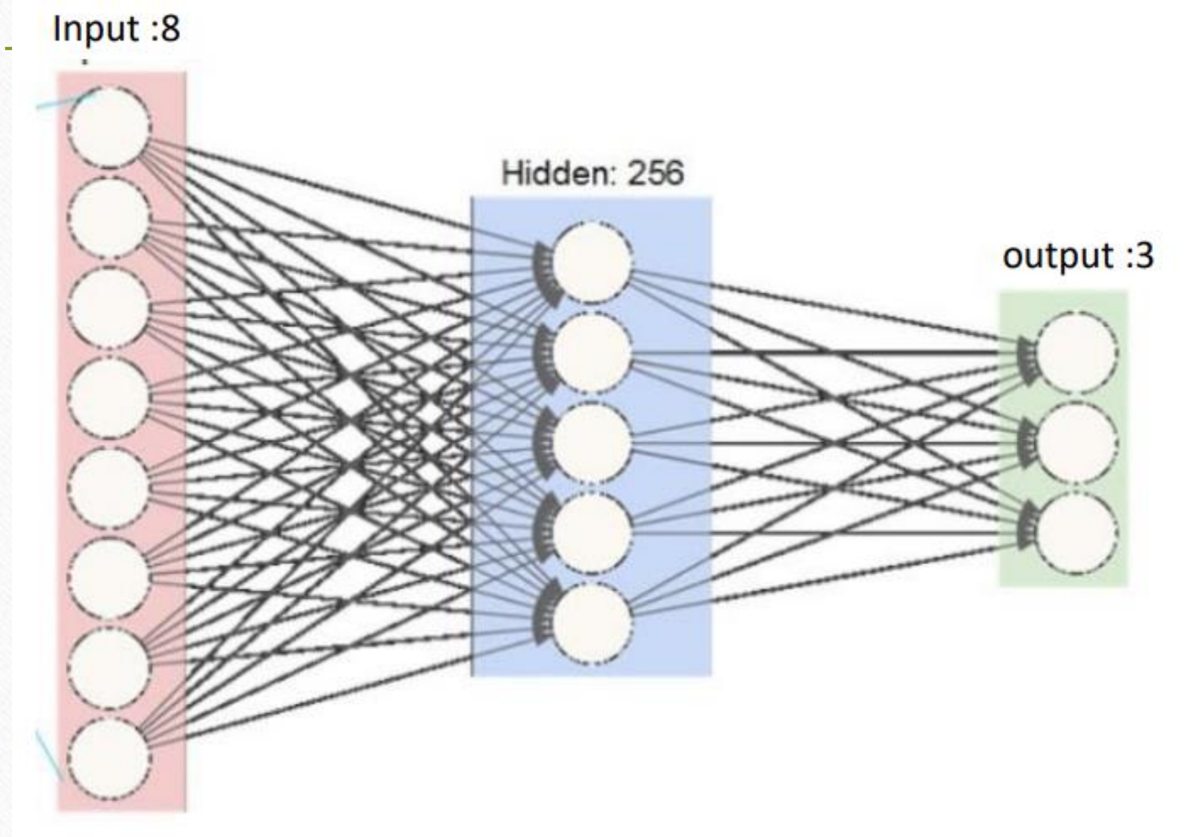
Layer (type)	Output shape	Param #
dense_Dense1 (Dense)	[null,64]	448
dropout_Dropout1 (Dropout)	[null,64]	0
dense_Dense2 (Dense)	[null,64]	4160
dropout_Dropout2 (Dropout)	[null,64]	0
dense_Dense3 (Dense)	[null,3]	195

Total params: 4803

Trainable params: 4803

Non-trainable params: 0

Neural Networks



Initializes a sequential model

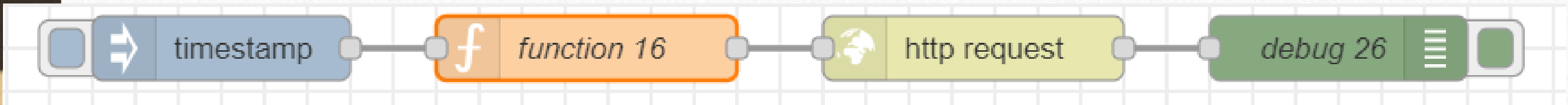
Add layers

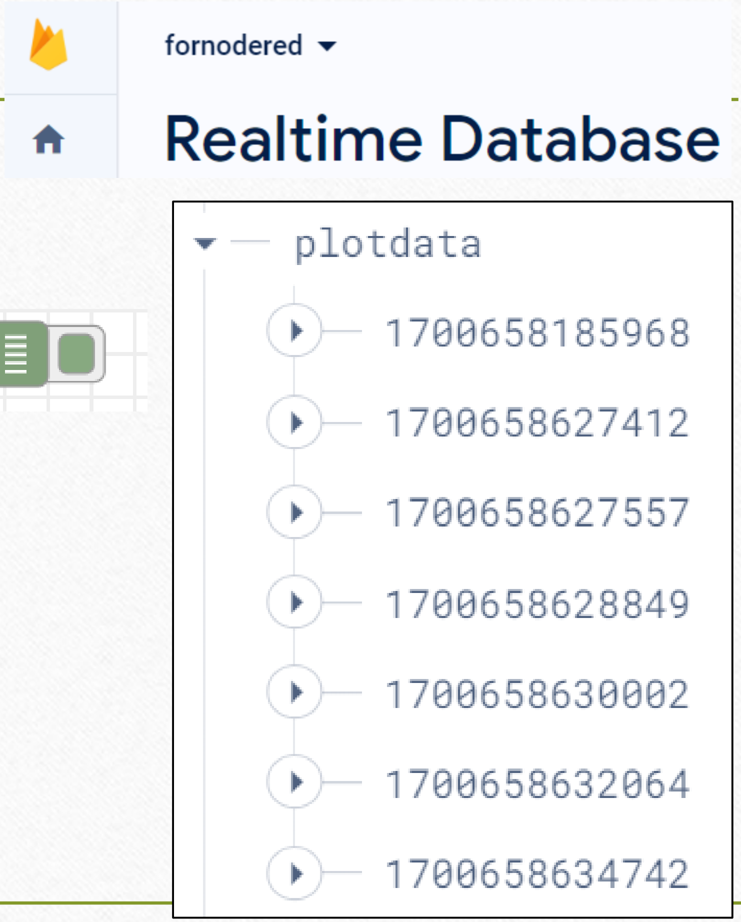
Compile

fit

Exercise 11-6

- Generate data to firebase:





JavaScript Get Date Methods

The new Date() Constructor

In JavaScript, date objects are created with `new Date()`.

`new Date()` returns a date object with the current date and time.

Get the Current Time

```
const date = new Date();
```

Try it Yourself »

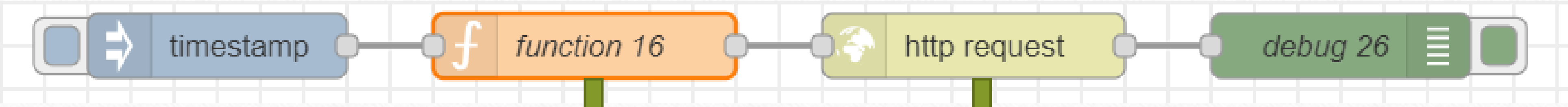
https://www.w3schools.com/js/js_date_methods.asp

Date Get Methods

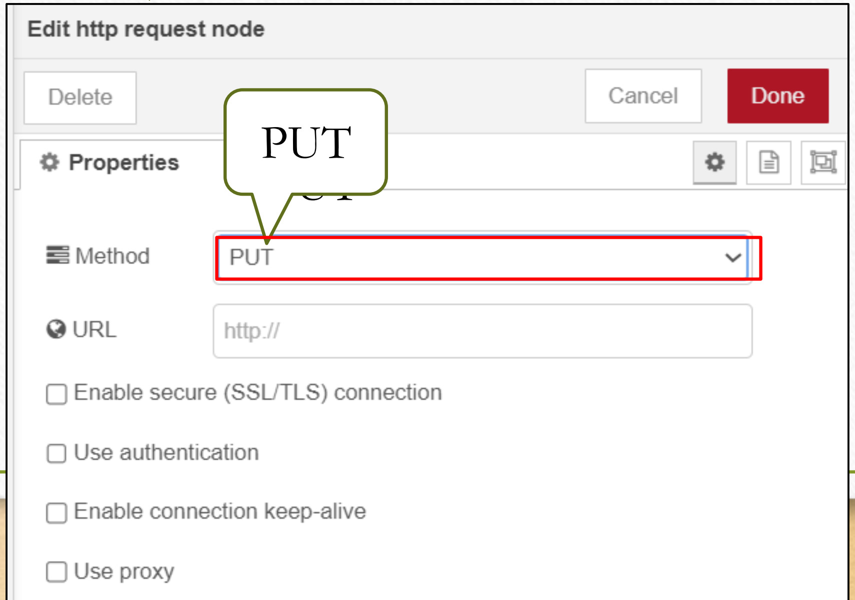
Method	Description
getFullYear()	Get **year** as a four digit number (yyyy)
getMonth()	Get **month** as a number (0-11)
getDate()	Get **day** as a number (1-31)
getDay()	Get **weekday** as a number (0-6)
getHours()	Get **hour** (0-23)
getMinutes()	Get **minute** (0-59)
getSeconds()	Get **second** (0-59)
getMilliseconds()	Get **millisecond** (0-999)
getTime()	Get **time** (milliseconds since January 1, 1970)

Edit nodes

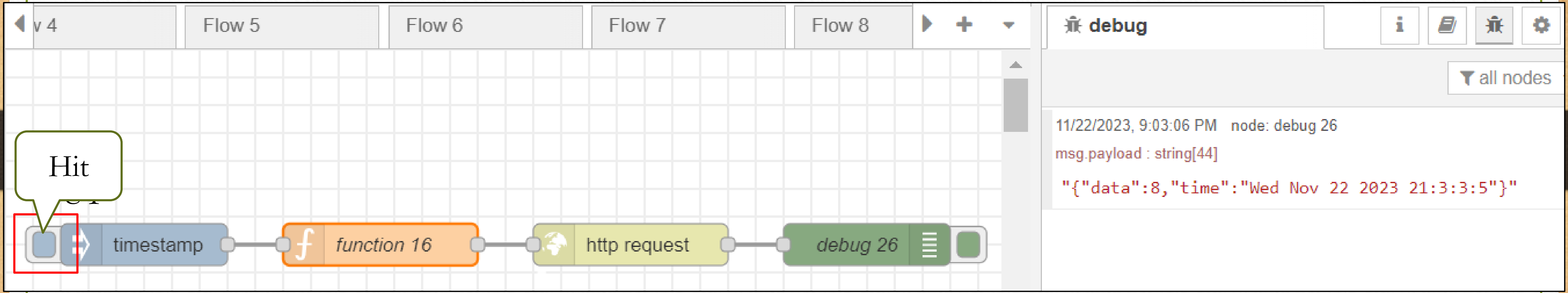
- Generate data to firebase:

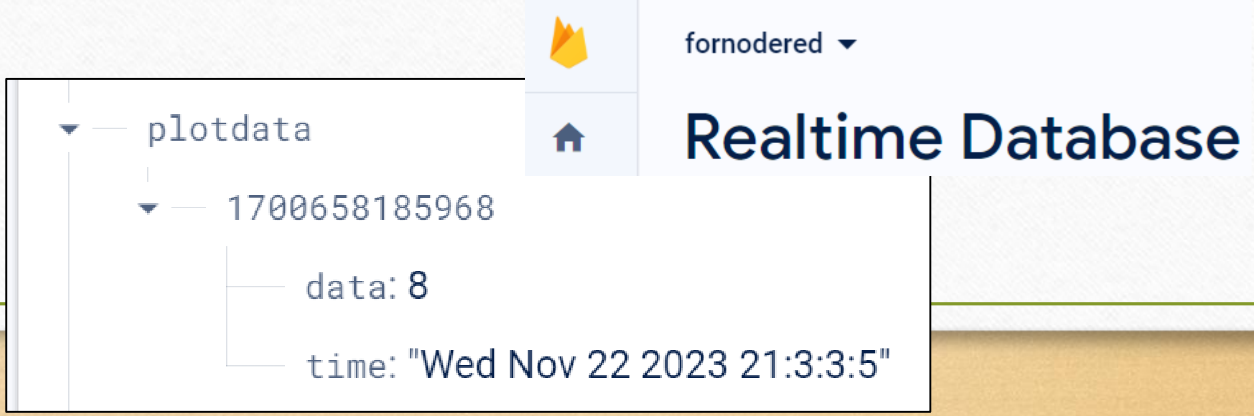



```
let data = Math.round((Math.random()*100));
const d = new Date();
var timestamp = d.getTime();
var time = d.toDateString() + " " + d.getHours()
+ ":" + d.getMinutes() + ":" +
d.getMinutes() + ":" + d.getSeconds();
msg.url="https://fornodered-312ce-default-
rtdb.firebaseio.com/plotdata/"+timestamp+".json";
msg.payload = {   "data": data, "time": time };
return msg;
```

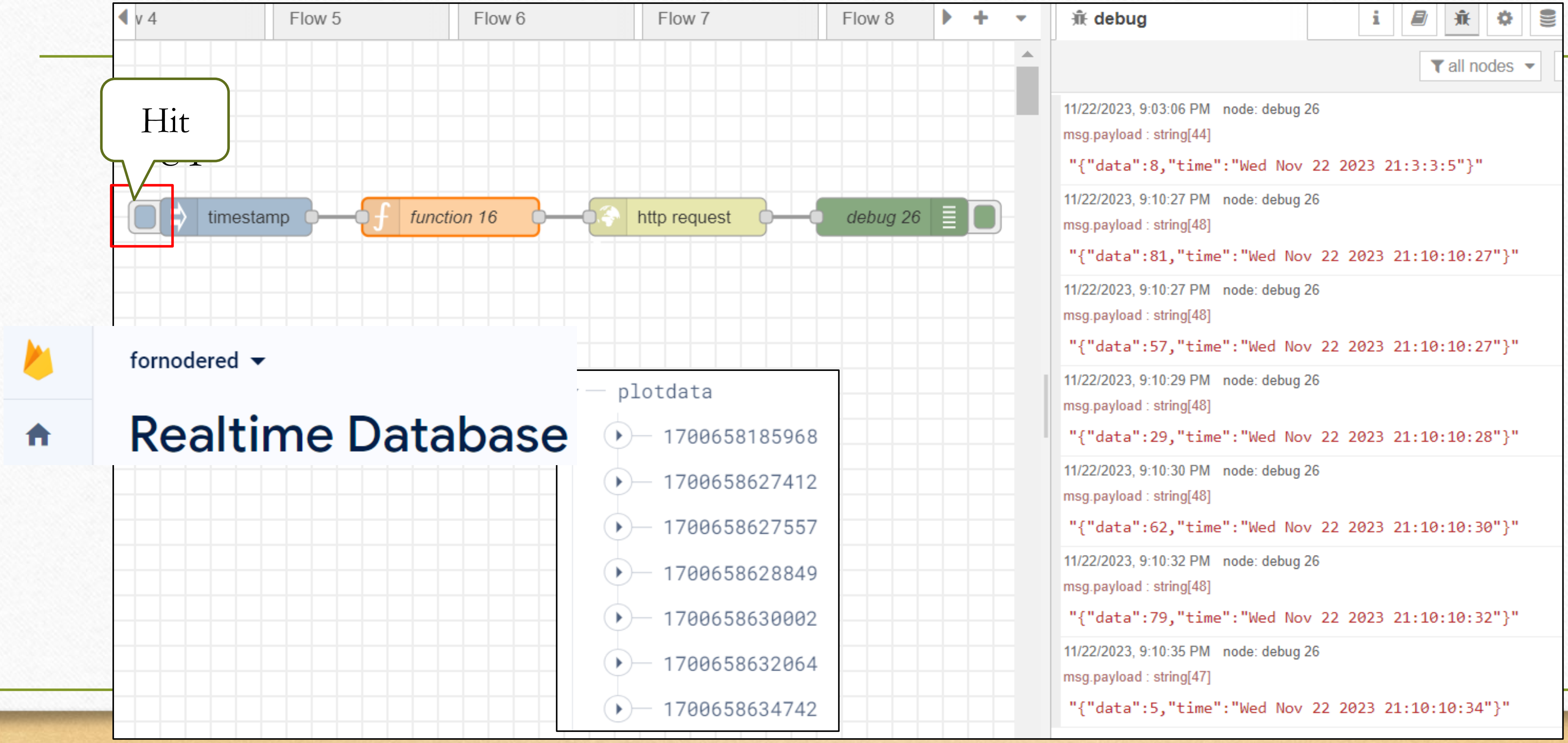


Trigger


v 4
Flow 5
Flow 6
Flow 7
Flow 8
debug
all nodes
11/22/2023, 9:03:06 PM node: debug 26
msg.payload : string[44]
"{"data":8,"time":"Wed Nov 22 2023 21:3:3:5"}"
Hit
timestamp
function 16
http request
debug 26

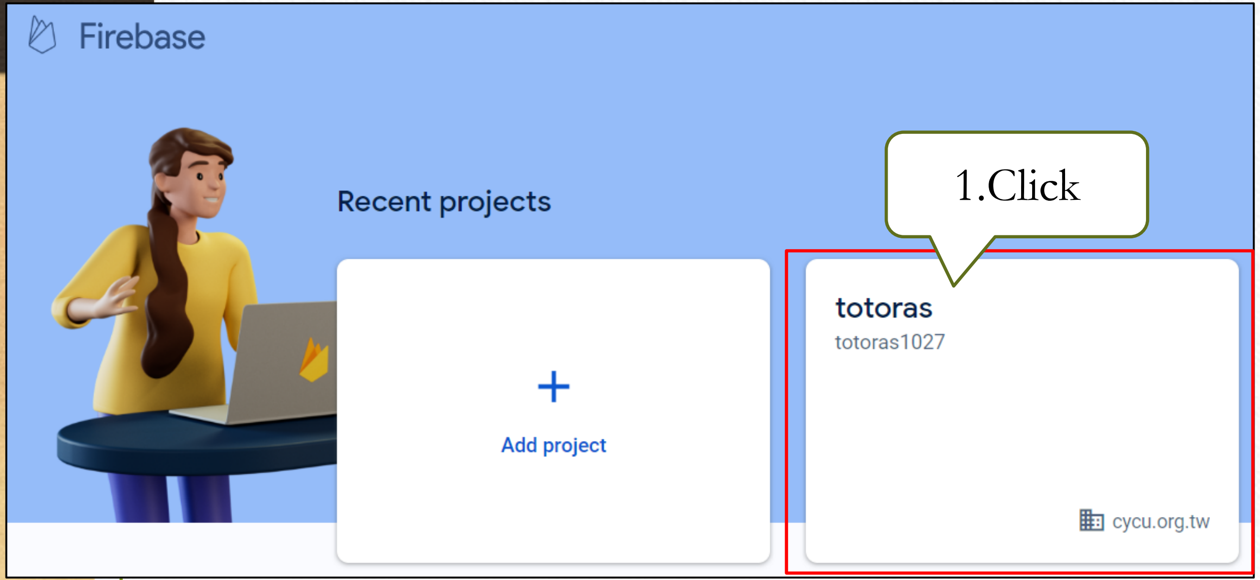

fornodered
Realtime Database
plotdata
1700658185968
data: 8
time: "Wed Nov 22 2023 21:3:3:5"

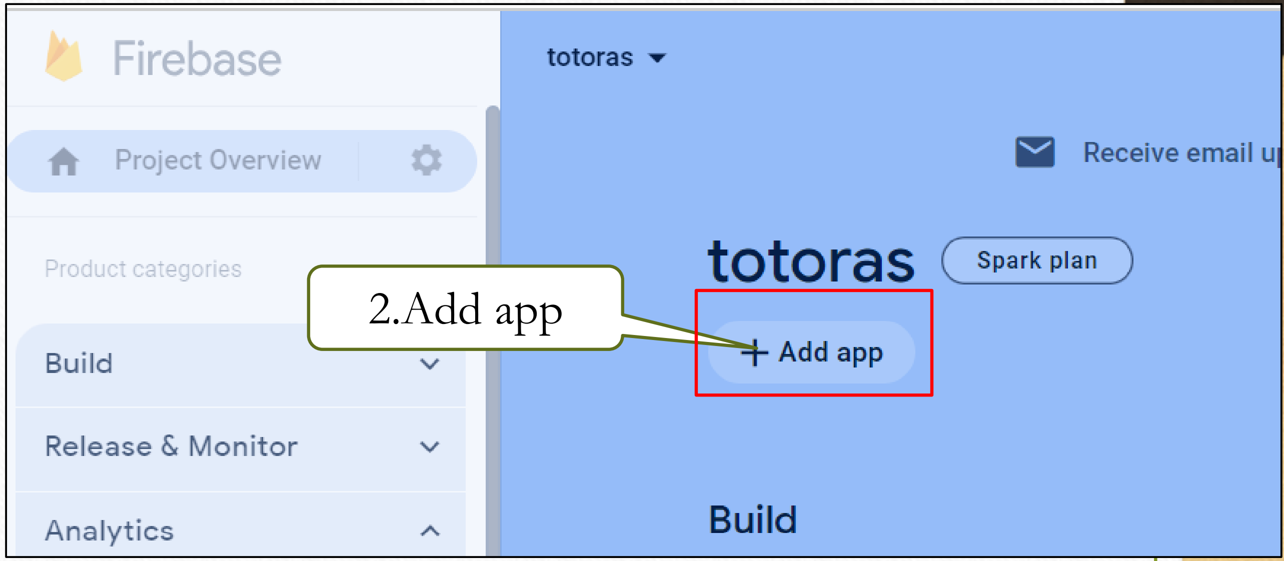
Trigger 6 times



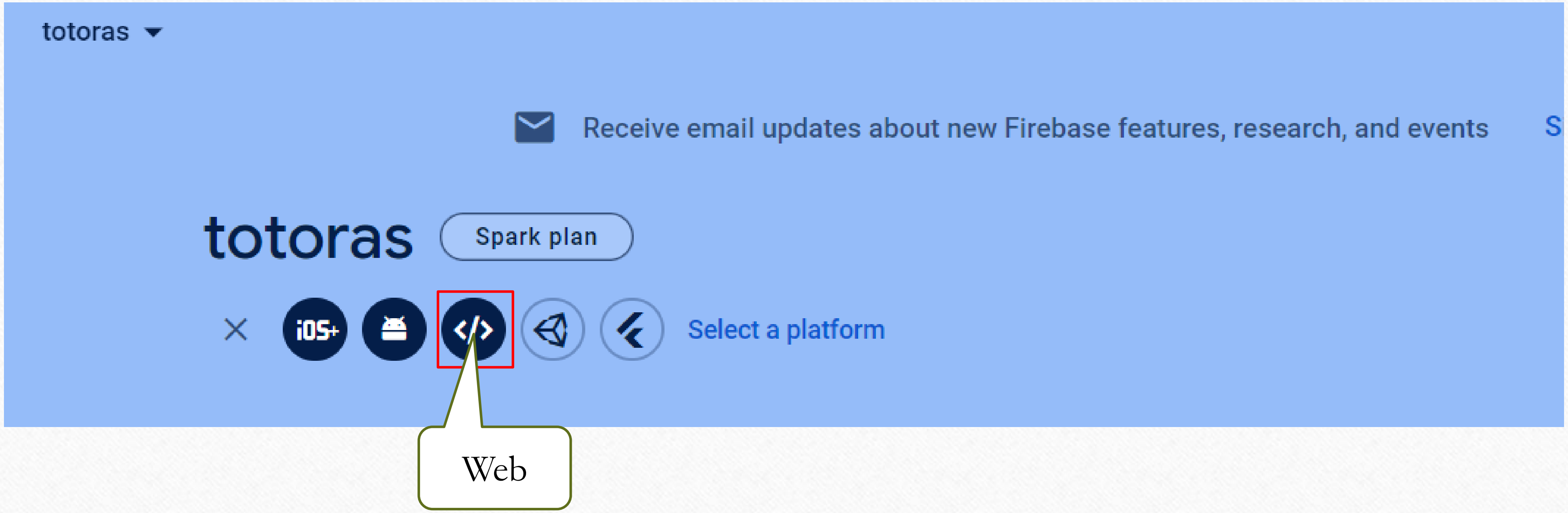
Exercise 11-7

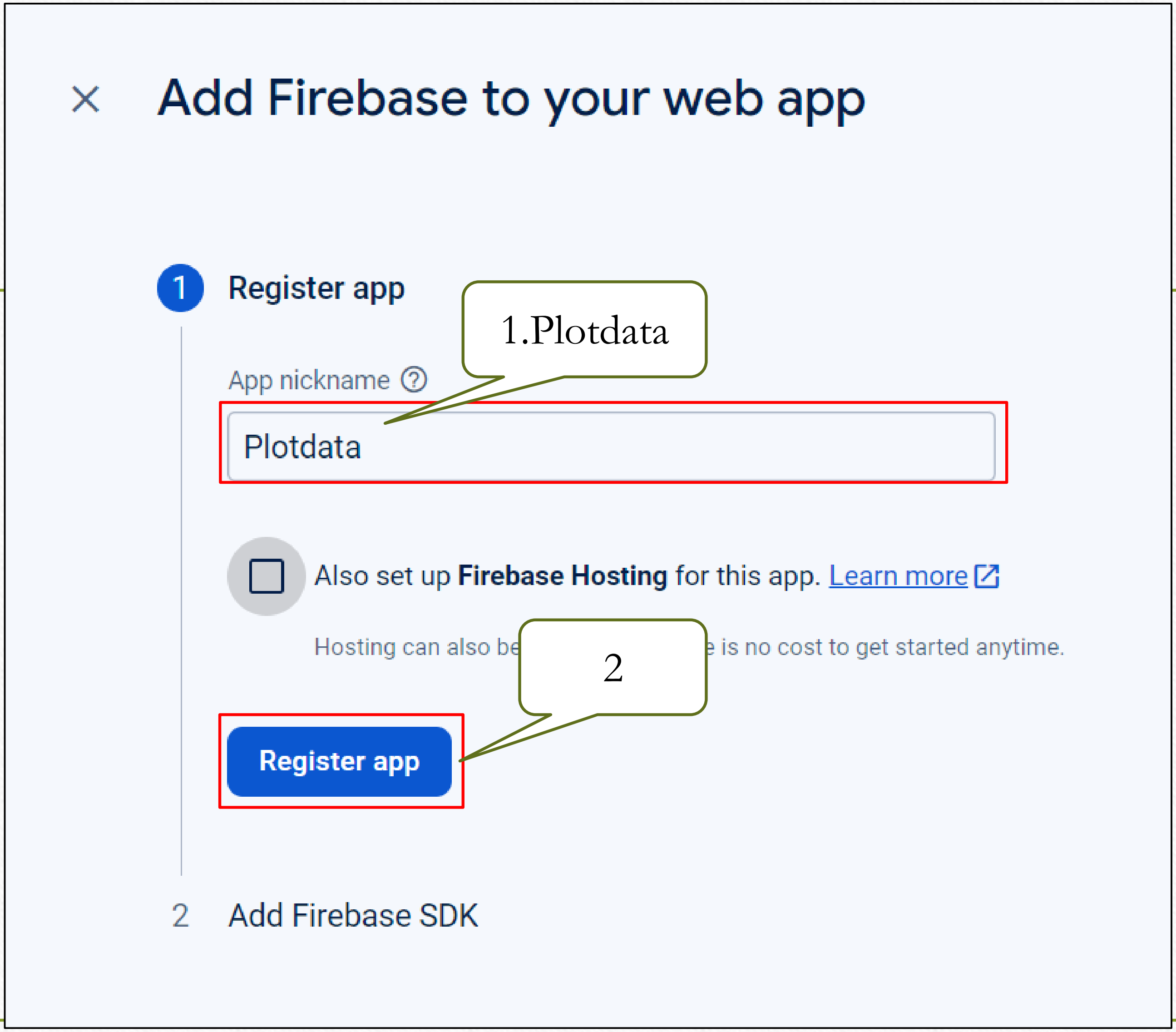
- Add App





Add Web


totoras
Receive email updates about new Firebase features, research, and events
totoras
Spark plan
Select a platform
Web


Add Firebase to your web app
1
Register app
1.Plotdata
App nickname
Plotdata
Also set up Firebase Hosting for this app. Learn more
Hosting can also be
is no cost to get started anytime.
2
Register app
2
Add Firebase SDK

2 Add Firebase SDK

Use a <script> tag

Use npm

Use a <script> tag

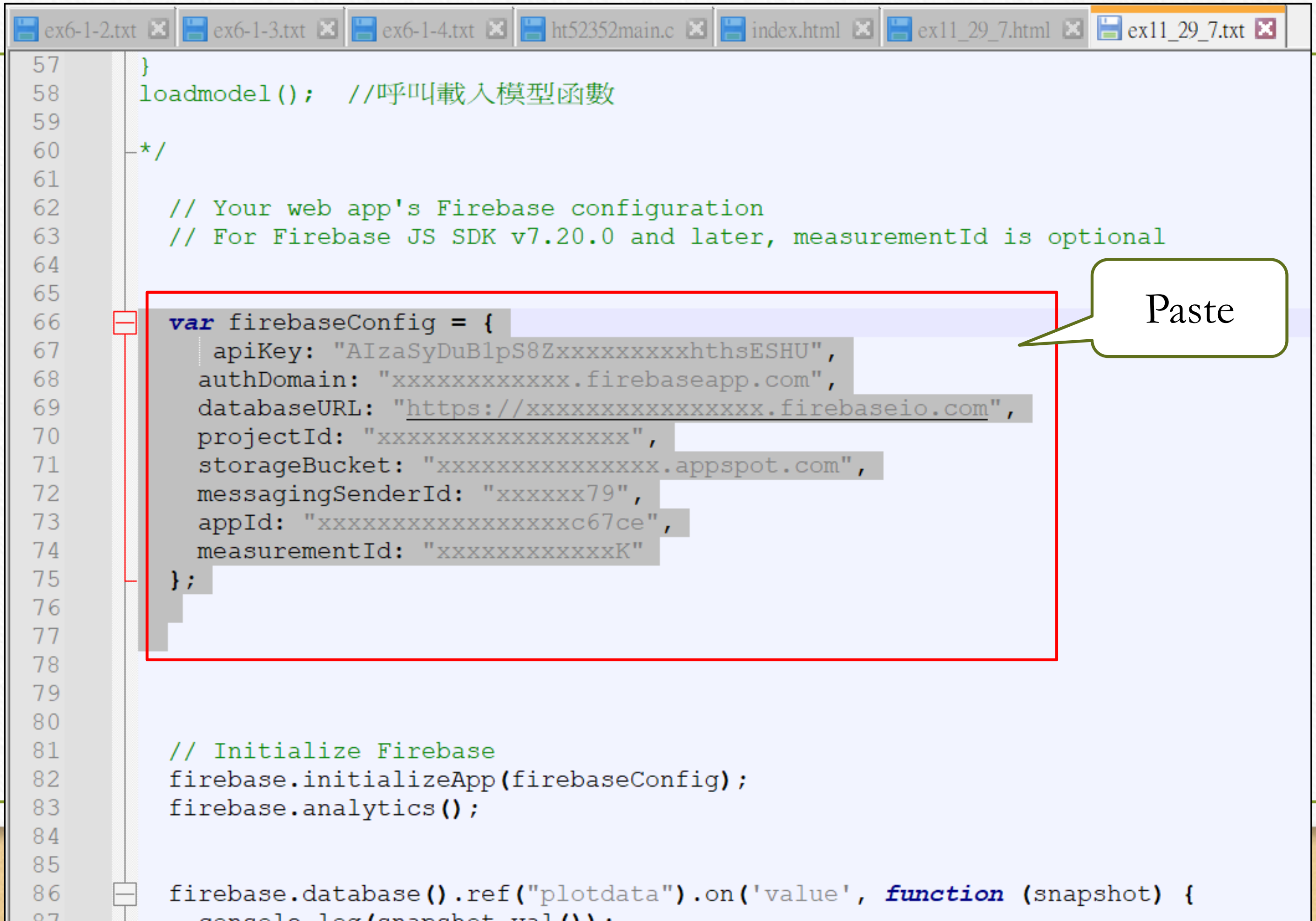
If you don't use build tools, use this option to add and use the Firebase JS SDK. Use this option to get started, but it's not recommended for production apps. Learn more.

Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

```
<script type="module">
  // Import the functions you need from the SDKs you need
  import { initializeApp } from "https://www.gstatic.com/firebasejs/10.6.0/fire
  // TODO: Add SDKs for Firebase products that you want to use
  // https://firebase.google.com/docs/web/setup#available-libraries

  // Your web app's Firebase configuration
  const firebaseConfig = {
    apiKey: "AIzaSyDqJ[redacted]M0Vw",
    authDomain: "totor[redacted]
    databaseURL: "http[redacted].com",
    projectId: "totora[redacted]
    storageBucket: "to[redacted]
    messagingSenderId:[redacted]
    appId: "1:71973988[redacted]34ae8"
  };

  // Initialize Firebase
  const app = initializeApp(firebaseConfig);
</script>
```

Copy

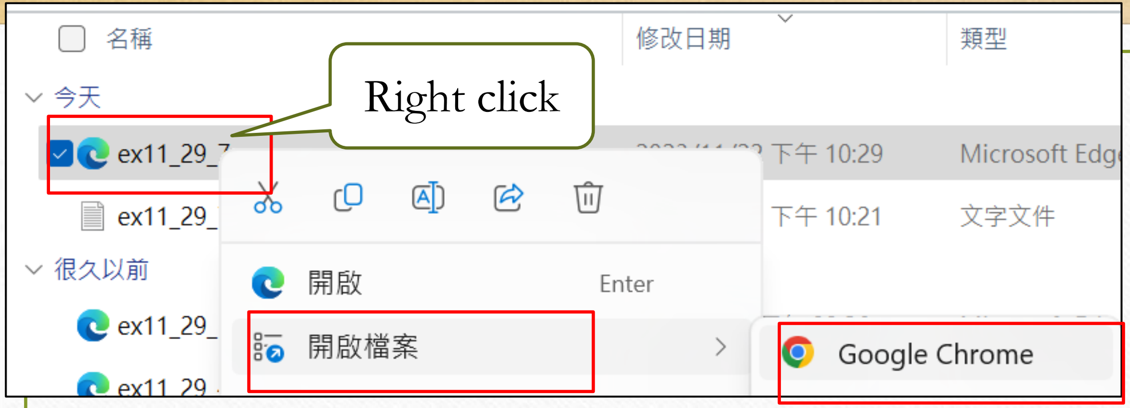
Paste to ex_11_29_7.txt

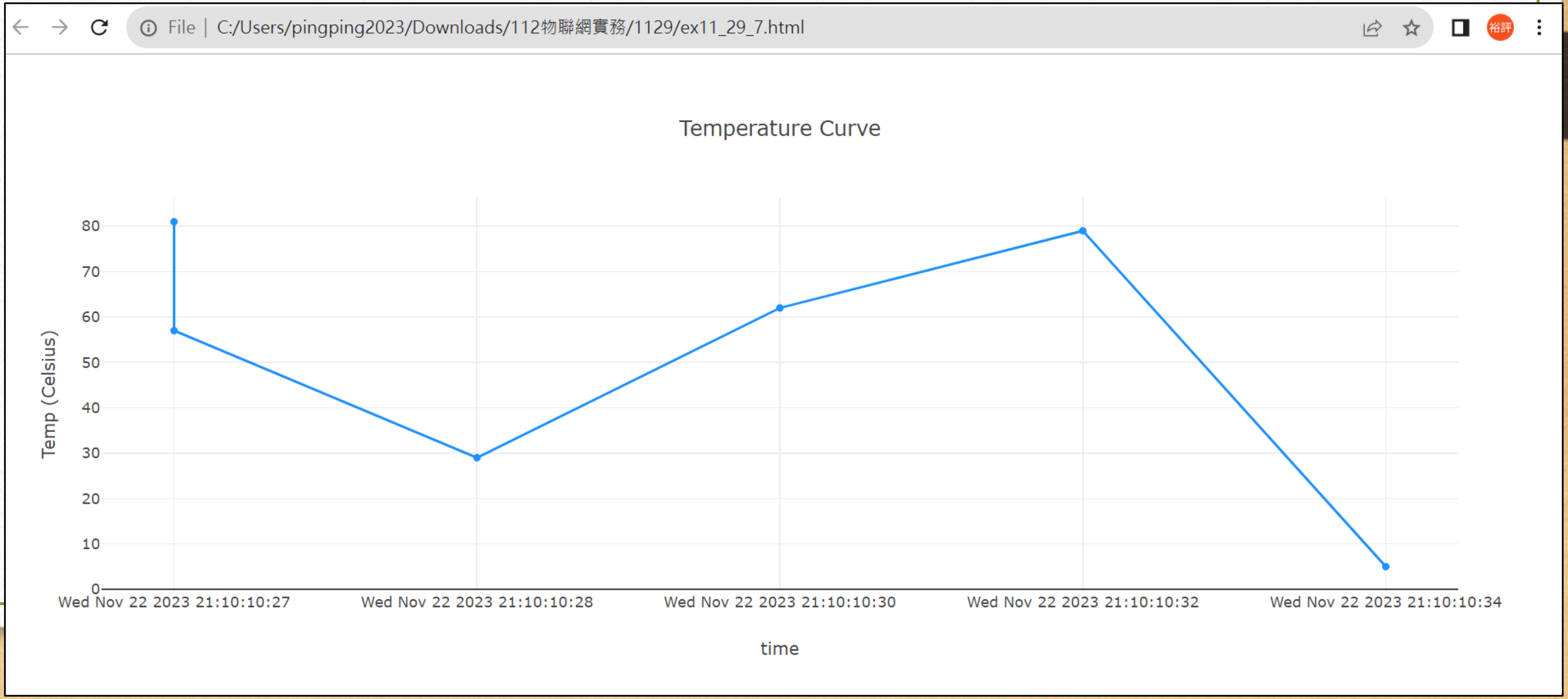


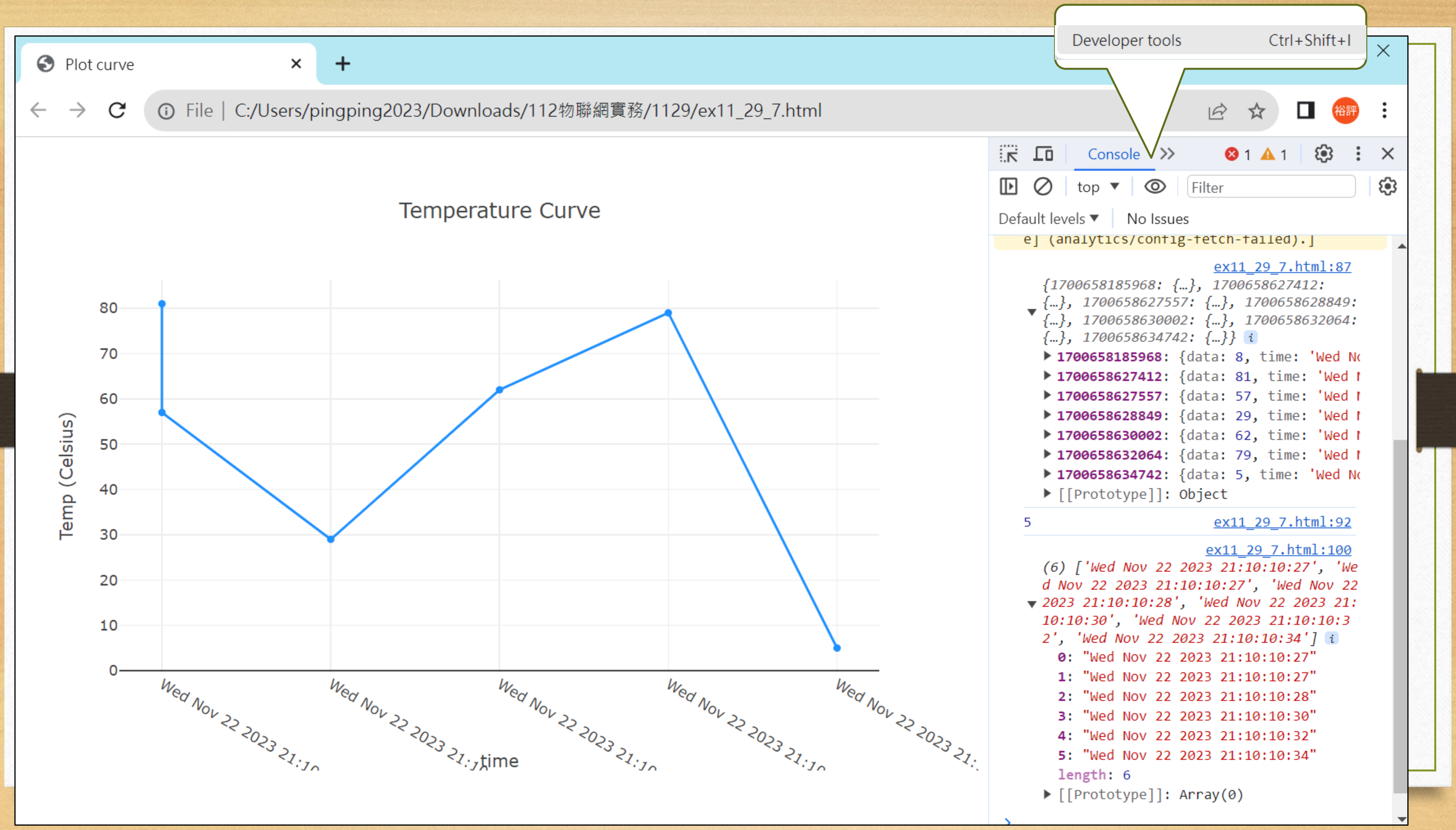

```
}
loadmodel();  //呼叫載入模型函數

*/

  // Your web app's Firebase configuration
  // For Firebase JS SDK v7.20.0 and later, measurementId is optional

  var firebaseConfig = {
     apiKey: "AIzaSyDuB1pS8ZxxxxxxxxxhthsESHU",
    authDomain: "xxxxxxxxxxxx.firebaseapp.com",
    databaseURL: "https://xxxxxxxxxxxxxxxx.firebaseio.com",
    projectId: "xxxxxxxxxxxxxxxxx",
    storageBucket: "xxxxxxxxxxxxxxx.appspot.com",
    messagingSenderId: "xxxxxx79",
    appId: "xxxxxxxxxxxxxxxxxxc67ce",
    measurementId: "xxxxxxxxxxxxK"
  };



  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  firebase.analytics();


  firebase.database().ref("plotdata").on('value', function (snapshot) {
```

Save As ex_11_29_7.html

```
<!DOCTYPE html>
<html>
<head>

<title>Ping Pong</title>
<script src="https://unpkg.com/@tensorflow/tfjs"></script>

<script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.21.0/firebase-app.js"></script>

<!-- TODO: Add SDKs for Firebase products that you want to use
     https://firebase.google.com/docs/web/setup#available-libraries -->
<script src="https://www.gstatic.com/firebasejs/7.21.0/firebase-analytics.js"></script>


<script src="https://www.gstatic.com/firebasejs/7.21.0/firebase-database.js"></script>
<style>
h1 {text-align: center;}


</style>
</head>

<body>
<p>
</p>
<div id="epf" style="width:100%;height:500px;"></div>
<h1><div id="status" style="center" ></div></h1>
```

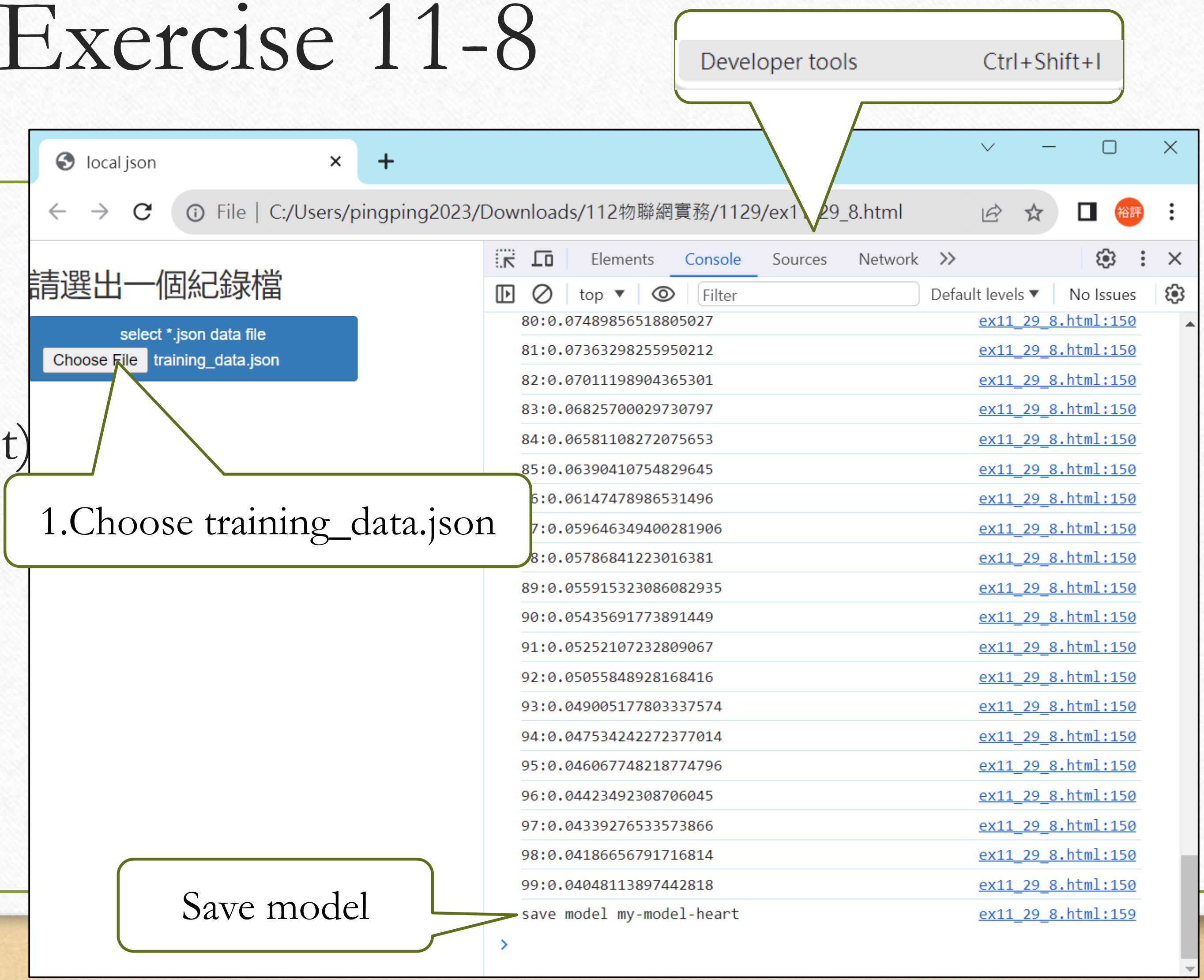

名稱
修改日期
類型
今天
Right click
ex11_29_7
下午 10:29
Microsoft Edg
ex11_29_
下午 10:21
文字文件
很久以前
ex11_29_
開啟
Enter
開啟檔案
Google Chrome


File | C:/Users/pingping2023/Downloads/112物聯網實務/1129/ex11_29_7.html
Temperature Curve
Temp (Celsius)
80
70
60
50
40
30
20
10
0
Wed Nov 22 2023 21:10:10:27
Wed Nov 22 2023 21:10:10:28
Wed Nov 22 2023 21:10:10:30
Wed Nov 22 2023 21:10:10:32
Wed Nov 22 2023 21:10:10:34
time


Developer tools Ctrl+Shift+I
Plot curve
C:/Users/pingping2023/Downloads/112物聯網實務/1129/ex11_29_7.html
Temperature Curve
Temp (Celsius)
80
70
60
50
40
30
20
10
0
Wed Nov 22 2023 21:10
time
Console
top
Filter
Default levels
No Issues
e] (analytics/config-fetch-failed).]
ex11_29_7.html:87
{1700658185968: {…}, 1700658627412: {…}, 1700658627557: {…}, 1700658628849: {…}, 1700658630002: {…}, 1700658632064: {…}, 1700658634742: {…}}
1700658185968: {data: 8, time: 'Wed No
1700658627412: {data: 81, time: 'Wed N
1700658627557: {data: 57, time: 'Wed N
1700658628849: {data: 29, time: 'Wed N
1700658630002: {data: 62, time: 'Wed N
1700658632064: {data: 79, time: 'Wed N
1700658634742: {data: 5, time: 'Wed No
[[Prototype]]: Object
5
ex11_29_7.html:92
ex11_29_7.html:100
(6) ['Wed Nov 22 2023 21:10:10:27', 'Wed Nov 22 2023 21:10:10:27', 'Wed Nov 22 2023 21:10:10:28', 'Wed Nov 22 2023 21:10:10:30', 'Wed Nov 22 2023 21:10:10:32', 'Wed Nov 22 2023 21:10:10:34']
0: "Wed Nov 22 2023 21:10:10:27"
1: "Wed Nov 22 2023 21:10:10:27"
2: "Wed Nov 22 2023 21:10:10:28"
3: "Wed Nov 22 2023 21:10:10:30"
4: "Wed Nov 22 2023 21:10:10:32"
5: "Wed Nov 22 2023 21:10:10:34"
length: 6
[[Prototype]]: Array(0)

Exercise 11-8

- Training Model
- ex11_29_8.html
- (copy from ex11_29_8.txt)
- AI play ping pong game



Homework 11-2

- Open “ex11_29_7.html” and save as “ex11_29_9.html”
- Delete /* */
- /* */

Open "ex11_29_9.html" with google chrome