Logistic Regression

Ata Kaban

What is logistic regression?

- It is a linear model for classification (contrary to its name!)

Recall the difference:

- In **regression**, the targets are real values
- In **classification**, the targets are categories, and they are called **labels**

Logistic regression - outline

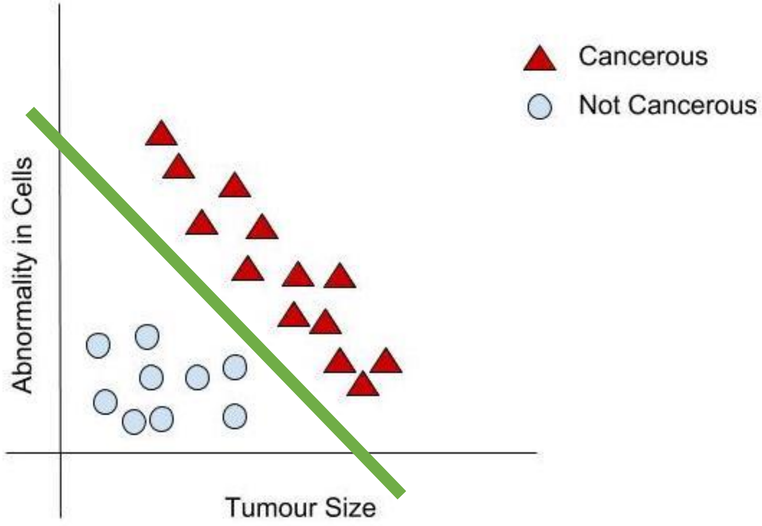
We will go through the same conceptual journey as before:

1) Model formulation
2) Cost function
3) Learning algorithm by gradient descent

1) Model

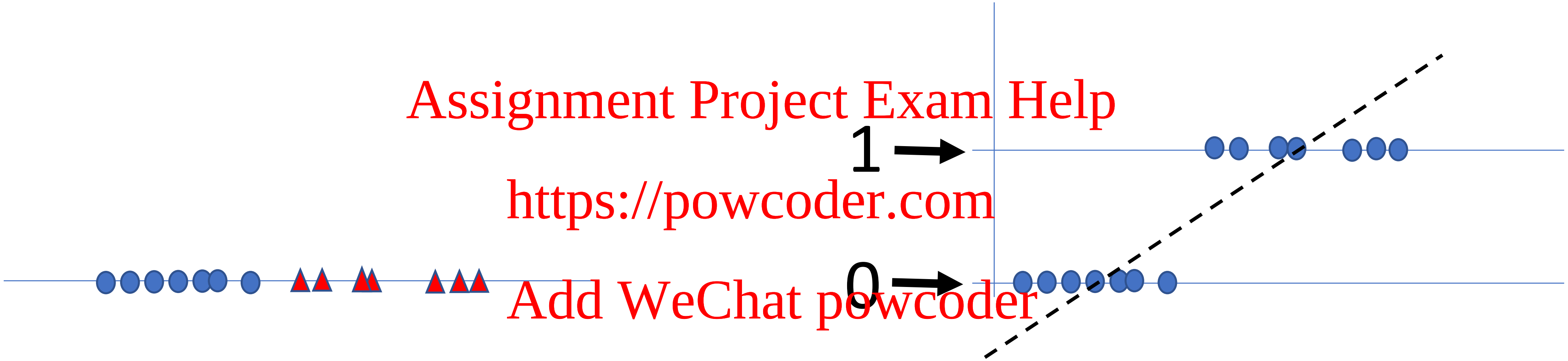
- We want to put a boundary between 2 classes
- If *x* has a single attribute, we can do it with a point

- If *x* has 2 attributes, we can do it with a line



- If *x* has 3 attributes, we can do it with a plane
- If *x* has more than 3 attributes, we can do it with a hyperplane (can't draw it anymore)
- If the classes are linearly separable, the training error will be 0.

Q: Can you plug classification data into linear regression?



A: Yes. But it might not perform very well. No ordering between categories, like there is between real numbers. We need a better model

Model

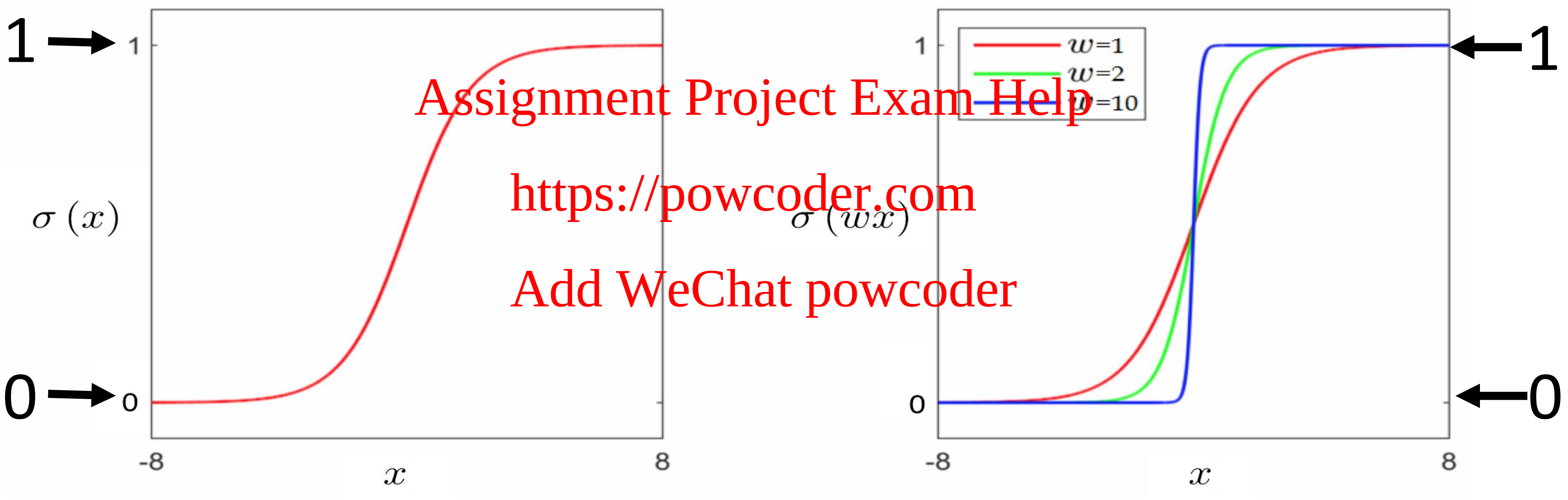
We change the linear model slightly by passing it through a nonlinearlity

- If x has 1 attribute, we will have

$$h(x; \boldsymbol{w}) = \sigma(w_0 + w_1 x) = \frac{1}{1+e^{-(w_0+w_1 x)}}$$

The function $\sigma(u) = \frac{1}{1+e^{-u}}$ is called the **sigmoid function** or **logistic function**

Sigmoid function

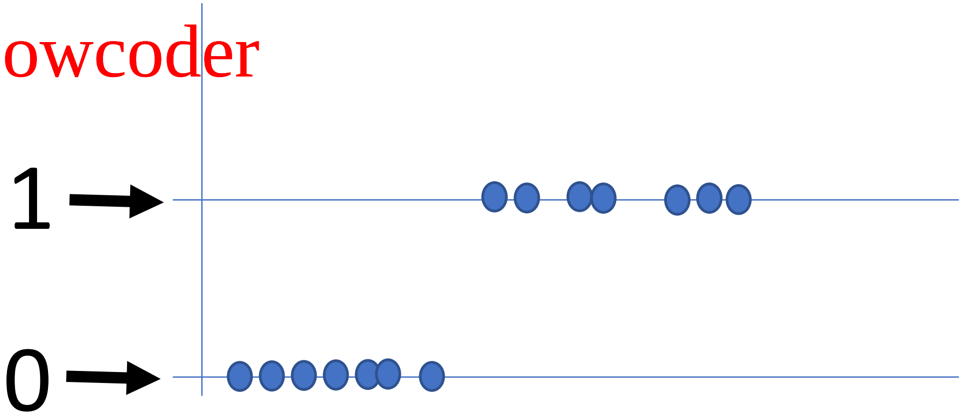


It is a smoothed version of a step function – note the step function would make optimisation difficult.

Play around with the logistic model

- Go to https://www.desmos.com/calculator
- Type: $y = \frac{1}{1+\exp(-(w_0+w_1 x))}$
- Change the values of the free parameters to see their effect

- Imagine how this function could fit this data better than a line did.
- What if your data happens to have class 1 on the left, and class 0 on the right?
  - w_1 can be negative, so the same model works.



Model

- If **x** has *d* attributes, that is $\boldsymbol{x} = (x_1, x_2, \ldots x_d)$, we will write

$$h(\boldsymbol{x}; \boldsymbol{w}) = \sigma(w_0 + w_1 x_1 + \cdots + w_d x_d) = \frac{1}{1+e^{-(\boldsymbol{w}^T \boldsymbol{x})}},$$ where:

all components of w are free parameters

$$\mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \cdots \\ w_d \end{pmatrix} \qquad \mathbf{x} = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \cdots \\ x_d \end{pmatrix} \in R^d$$

Meaning of the sigmoid function

- The sigmoid function takes a single argument (note, $\boldsymbol{w}^T\boldsymbol{x}$ is one number).
- It always returns a value between 0 and 1. The meaning of this value is the **probability that the label is 1**.

$$\sigma(\boldsymbol{w}^T\boldsymbol{x}) = P(y = 1|\boldsymbol{x}; \boldsymbol{w})$$

- If this is smaller than 0.5 then we predict label 0.

- if this is larger than 0.5 then we predict label 1.

- There is a slim chance that the sigmoid outputs exactly 0.5. The set of all possible inputs for which this happens is called the decision boundary.

Check your understanding

- Can you express the probability that the label is 0 using sigmoid?

$$\sigma(\boldsymbol{w}^T \boldsymbol{x}) = P(y = 1|\boldsymbol{x}; \boldsymbol{w})$$

$$\Rightarrow\ 1 - \sigma(\boldsymbol{w}^T \boldsymbol{x}) = 1 - P(y = 1|\boldsymbol{x}; \boldsymbol{w}) = P(y = 0|\boldsymbol{x}; \boldsymbol{w})$$

- In fact we can write both in 1 line as:

$P(y|\boldsymbol{x}; \boldsymbol{w}) = \sigma(\boldsymbol{w}^T \boldsymbol{x})^y (1 - \sigma(\boldsymbol{w}^T \boldsymbol{x}))^{1-y}$ //*y* given *x* has a Bernoulli distribution

Worked example

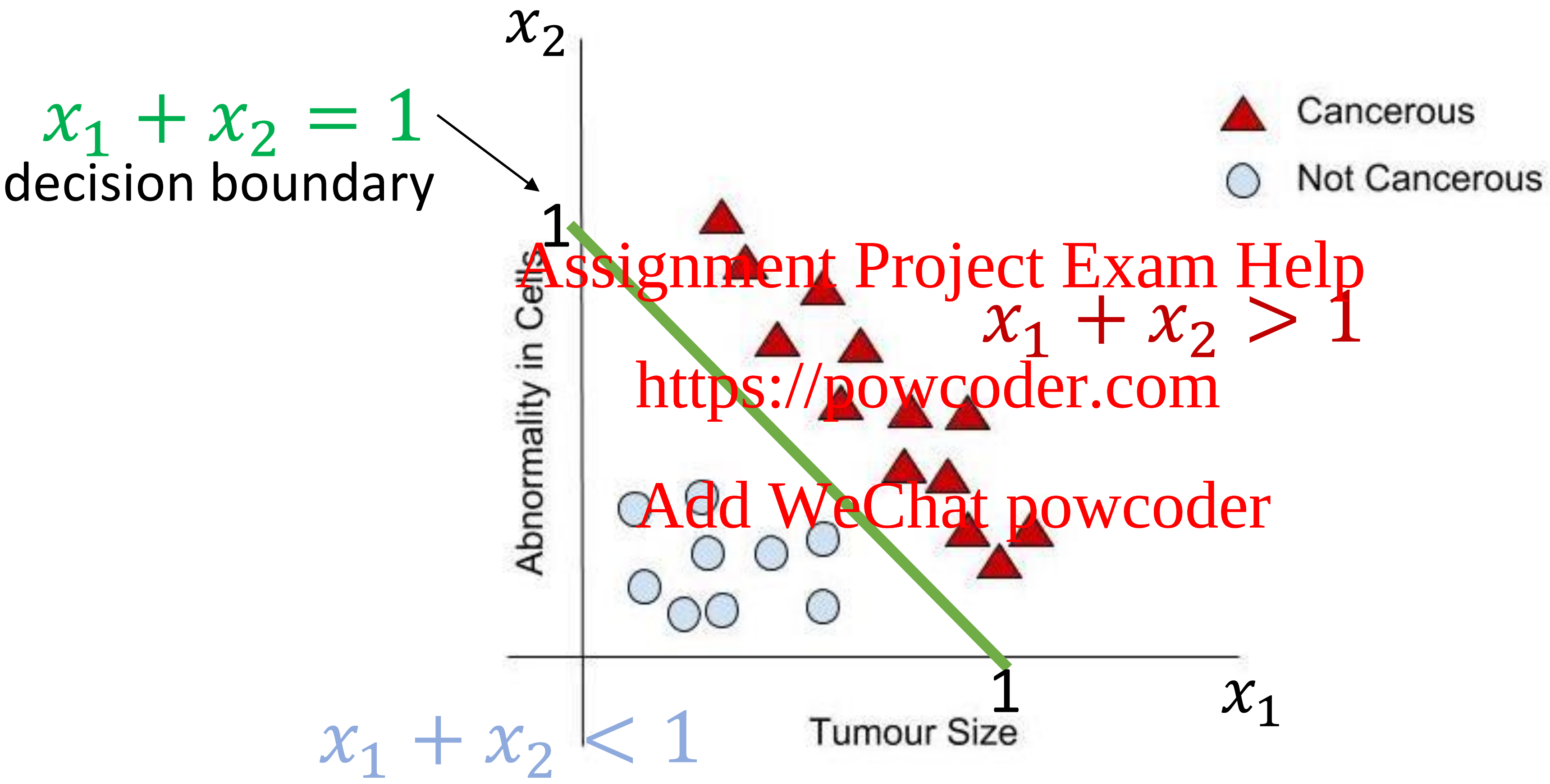
- Suppose we have 2 input attributes, so our model is
$$h(\boldsymbol{x};\boldsymbol{w}) = \sigma(w_0 + w_1 x_1 + w_2 x_2).$$
- Suppose we know that $w_0 = -1, w_1 = 1, w_2 = 1$.

- When do we predict 1? What is the decision boundary?

  - We predict 1 precisely when $P(y = 1|\boldsymbol{x};\boldsymbol{w})$>0.5. That is, when $h(\boldsymbol{x};\boldsymbol{w})$>0.5.

  - This happens precisely when the argument of the sigmoid is positive!
  - Decision boundary: $-1 + x_1 + x_2 = 0$ This is a line
- Q: Is the decision boundary of logistic regression always linear?

  A: Yes.

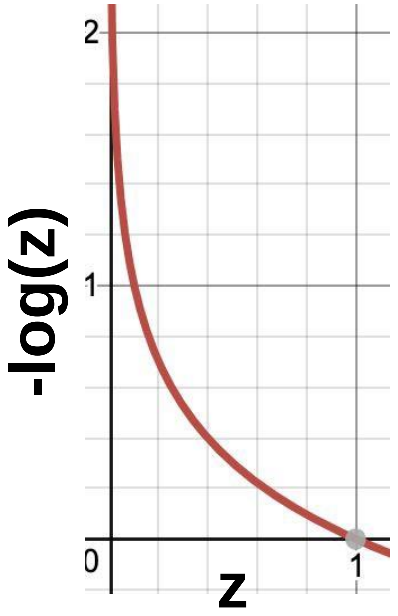

$x_1 + x_2 = 1$
decision boundary
x_2
1
Cancerous
Not Cancerous
Abnormality in Cells
$x_1 + x_2 > 1$
1
x_1
Tumour Size
$x_1 + x_2 < 1$

2) Cost function

- We need a new cost function, because the Mean Square Error used in linear regression produces a very wiggly function with the new hypothesis function, which would be difficult to optimise.
- But as before we will still have that:
  - each data point contributes a cost, and the overall cost function is the average of these
  - the cost is a function of the free parameters of the model

Logistic cost function

For each $(\boldsymbol{x},y)$ pair, $Cost(h(\boldsymbol{x};\boldsymbol{w}),y) = \begin{cases} -\log\left(\overbrace{h(\boldsymbol{x};\boldsymbol{w})}^{\mathbf{z}}\right), & if\ y=1 \\ -\log\left(1-\underbrace{h(\boldsymbol{x};\boldsymbol{w})}_{\mathbf{z}}\right), & if\ y=0 \end{cases}$



Overall cost: $g(\boldsymbol{w}) = \frac{1}{N}\sum_{n=0}^{N} Cost\left(h\left(\boldsymbol{x}^{(n)};\boldsymbol{w}\right),y^{(n)}\right)$ convex (easy to minimise)

Writing the cost function in a single line

$$g(\boldsymbol{w}) = \frac{1}{N}\sum_{n=0}^{N} Cost(h(\boldsymbol{x}^{(n)};\boldsymbol{w}), y^{(n)})$$

$$Cost(h(\boldsymbol{x};\boldsymbol{w}), y) = \begin{cases} -\log(h(\boldsymbol{x};\boldsymbol{w})), & if\ y = 1 \\ -\log(1 - h(\boldsymbol{x};\boldsymbol{w})), & if\ y = 0 \end{cases}$$

$$\mathrm{g}(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N}(y^{(n)}\log h(\boldsymbol{x}^{(n)};\boldsymbol{w}) + (1 - y^{(n)})\log(1 - h(\boldsymbol{x}^{(n)};\boldsymbol{w})))$$

This is also called the cross-entropy.

Logistic regression – what we want to do

- Given training data

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(N)}, y^{(N)})$$

- Fit the model

$$y = h(x; \boldsymbol{w}) = \boldsymbol{\sigma}(\boldsymbol{w}^T \boldsymbol{x})$$

- By minimising the cross-entropy cost function

$$\mathrm{g}(\mathbf{w}) = -\frac{1}{N} \sum_{n=1}^{N} (y^{(n)} \log h(\boldsymbol{x}^{(n)}; \boldsymbol{w}) + (1 - y^{(n)}) \log(1 - h(\boldsymbol{x}^{(n)}; \boldsymbol{w})))$$

3) Learning algorithm by gradient descent

- We use gradient descent (again!) to minimise the cost function, i.e. to find the best weight values.

- The gradient vector is*:

$$\nabla g(\boldsymbol{w}) = -(\mathrm{y}^{(\mathrm{n})} - \mathrm{h}(\mathbf{x}^{(\mathrm{n})}; \mathbf{w})) \cdot \mathbf{x}^{(\mathrm{n})}$$

$$\mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \dots \\ w_d \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \dots \\ x_d \end{pmatrix} \in R^d$$

We plug this into the general gradient descent algorithm given last week.

--

* This follows after differentiating the cost function w.r.t. weights – we omit the lengthy math!

Learning algorithm for logistic regression

While not converged

For n = 1,…,N // each example in the training set

$$\mathbf{w} = \mathbf{w} + \alpha\left(y^{(n)} - h\left(\mathbf{x}^{(n)}; \mathbf{w}\right)\right) \cdot \mathbf{x}^{(n)}$$

Learning algorithm for logistic regression

The same, written component-wise:

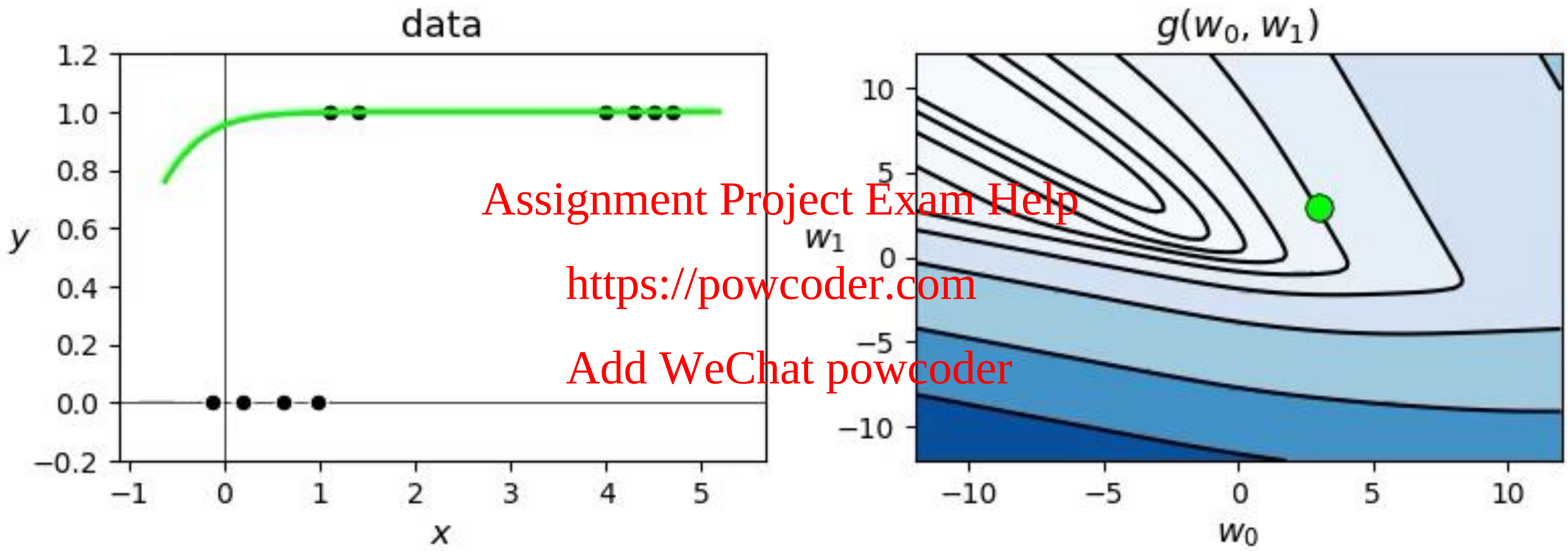
While not converged

For n = 1,…,N // each example in the training set

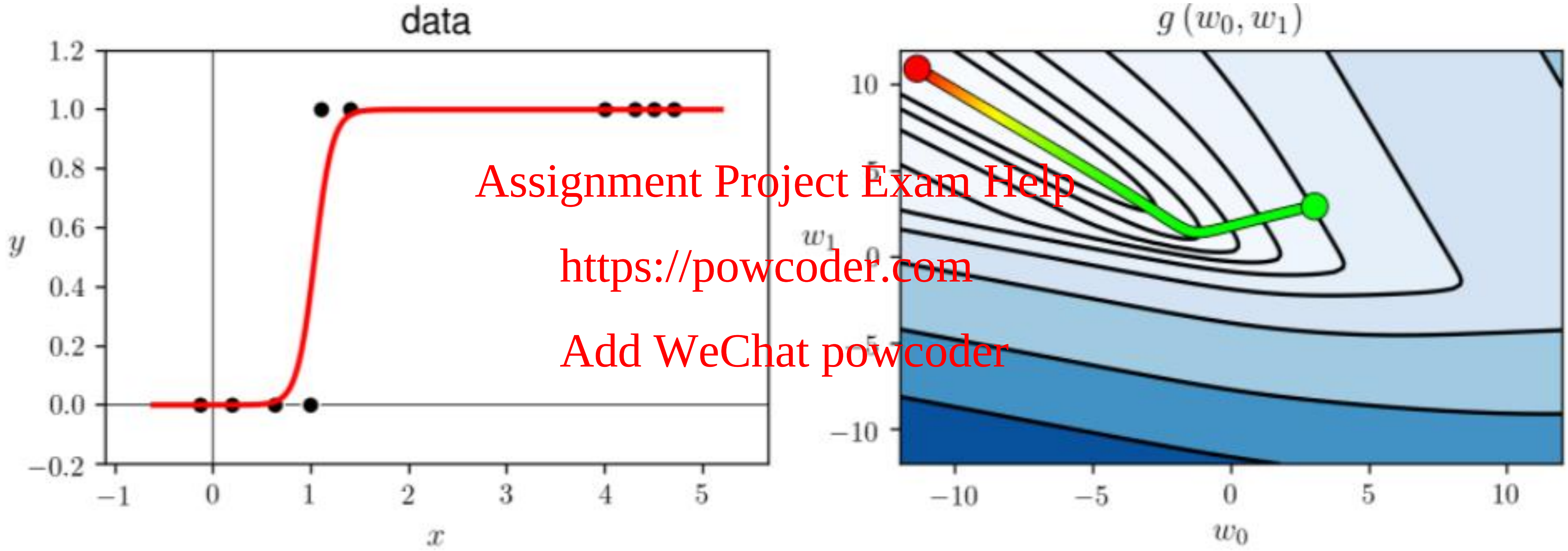
$$\mathrm{w}_0 = \mathrm{w}_0 + \alpha\left(\mathrm{y}^{(\mathrm{n})} - \mathrm{h}\left(\mathbf{x}^{(\mathrm{n})}; \mathbf{w}\right)\right)$$

For *i=1,…,d*

$$\mathrm{w}_\mathrm{i} = \mathrm{w}_\mathrm{i} + \alpha\left(\mathrm{y}^{(\mathrm{n})} - \mathrm{h}\left(\mathbf{x}^{(\mathrm{n})}; \mathbf{w}\right)\right)\mathrm{x}_i^{(\mathrm{n})}$$



https://github.com/jermwatt/machine_learning_refined/blob/gh-pages/notes/6_Linear_twoclass_classification/videos/animation_1.mp4



https://github.com/jermwatt/machine_learning_refined/blob/gh-pages/notes/6_Linear_twoclass_classification/videos/animation_1.mp4

Extensions

- We studied logistic regression for linear binary classification
- There are extensions, such as:
  - Nonlinear logistic regression: instead of linear function inside the exp in the sigmoid, we can use polynomial functions of the input attributes
  - Multi-class logistic regression: uses a multi-valued version of sigmoid
- Details of these extensions are beyond of our scope in this module

Examples of application of logistic regression

- Face detection: classes consist of images that contain a face and images without a face

- Sentiment analysis: classes consist of written product-reviews expressing a positive or a negative opinion

- Automatic diagnosis of medical conditions: classes consist of medical data of patients who either do or do not have a specific disease