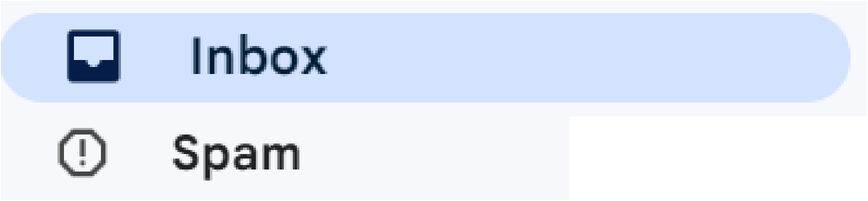
Logistic Regression & Regularization

07/21/2023

Logistic Regression

Binary Classification

1(cat) vs 0 (non cat)



1(spam) vs 0 (non spam)

Dataset: Annie's Youtube history

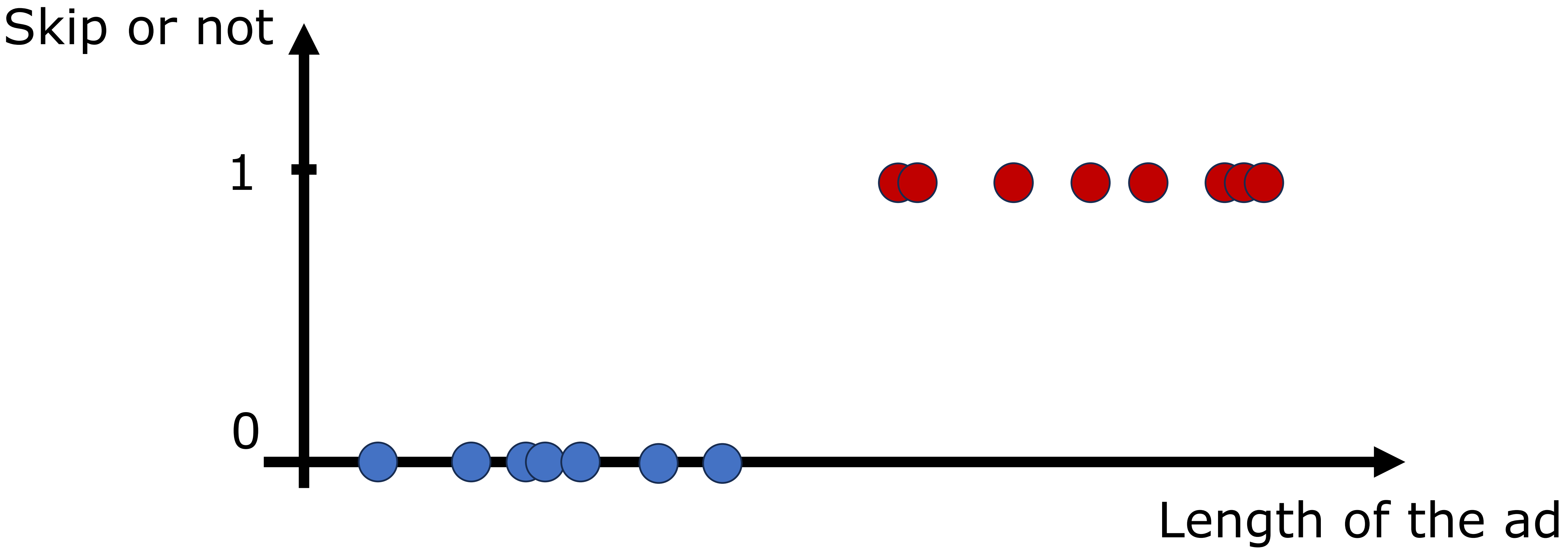
- Datapoints: $(x_i, y_i), \quad i = 1, 2, \dots, n$
- x: the length of the advertisement
- y : whether Annie skips the advertisement

  $y = 1$ if Annie skips the ad;

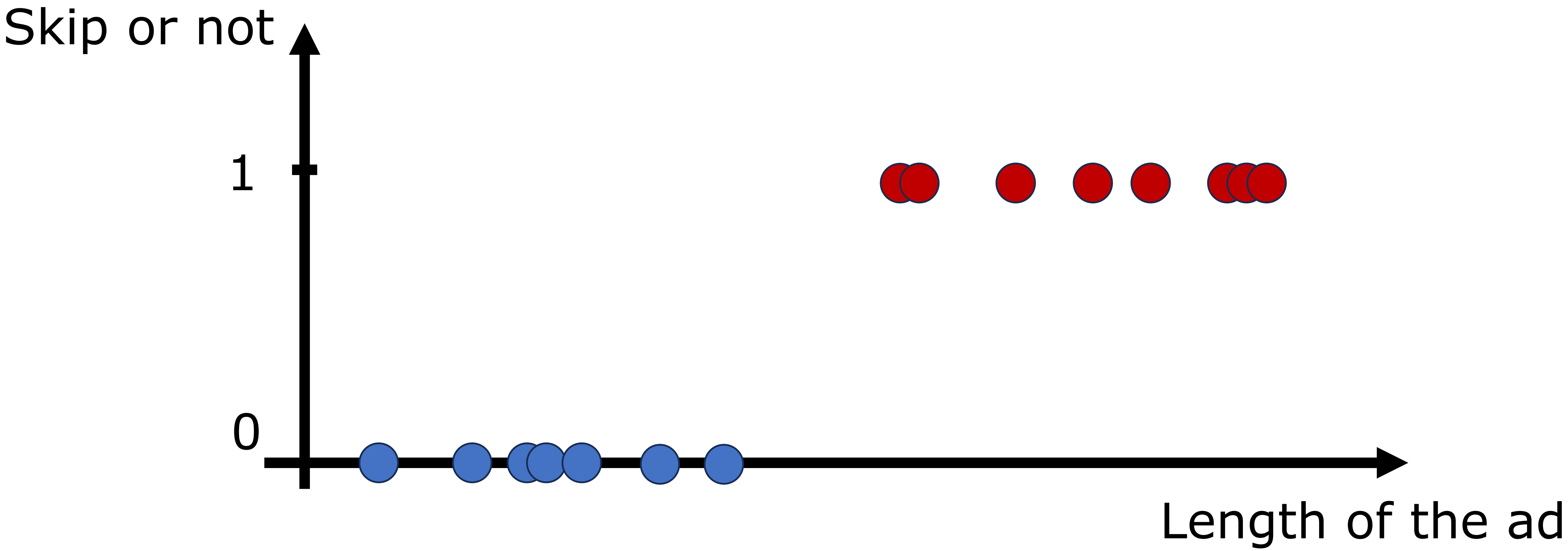
  $y = 0$ if Annie doesn't

- Goal: Given the length of an advertisement, predict whether Annie will skip it.

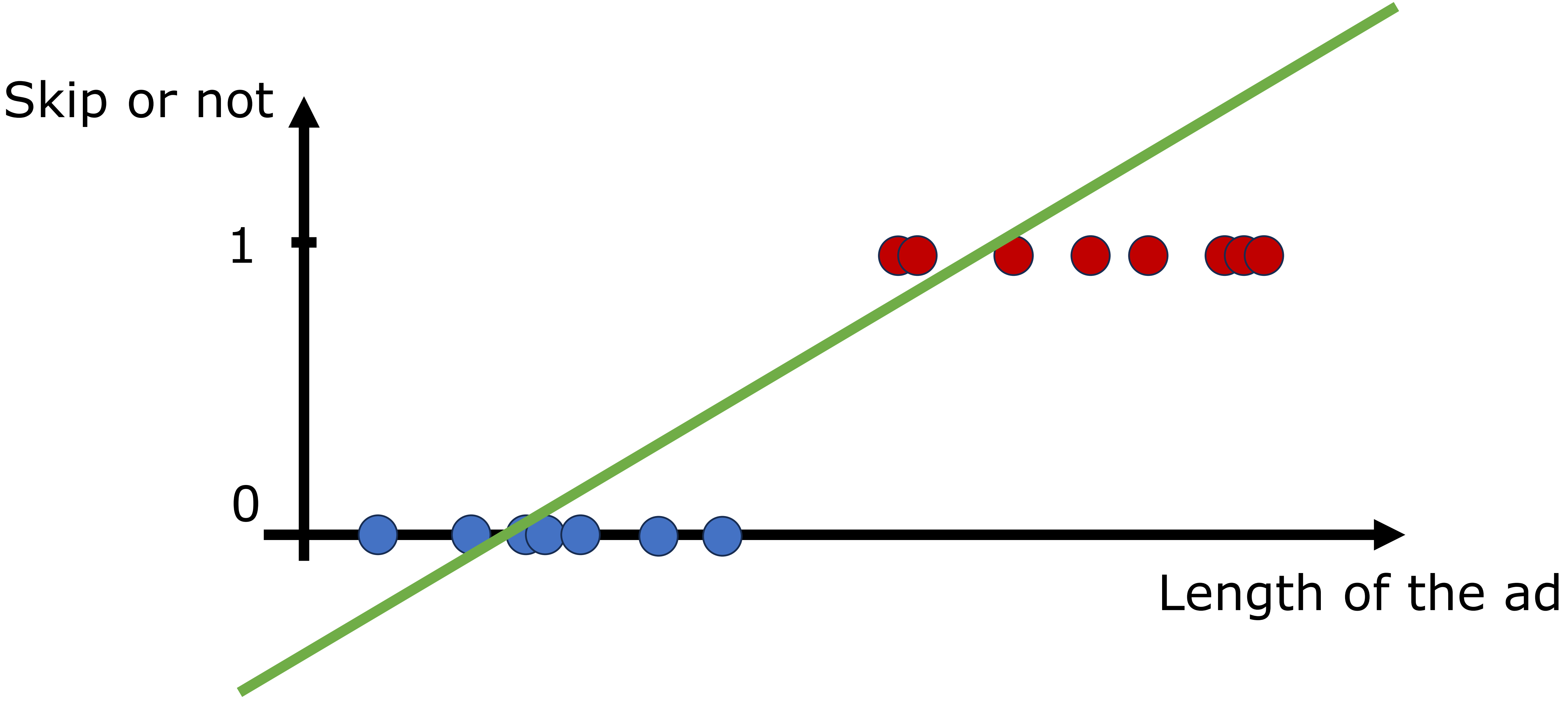
Visualize the datapoints



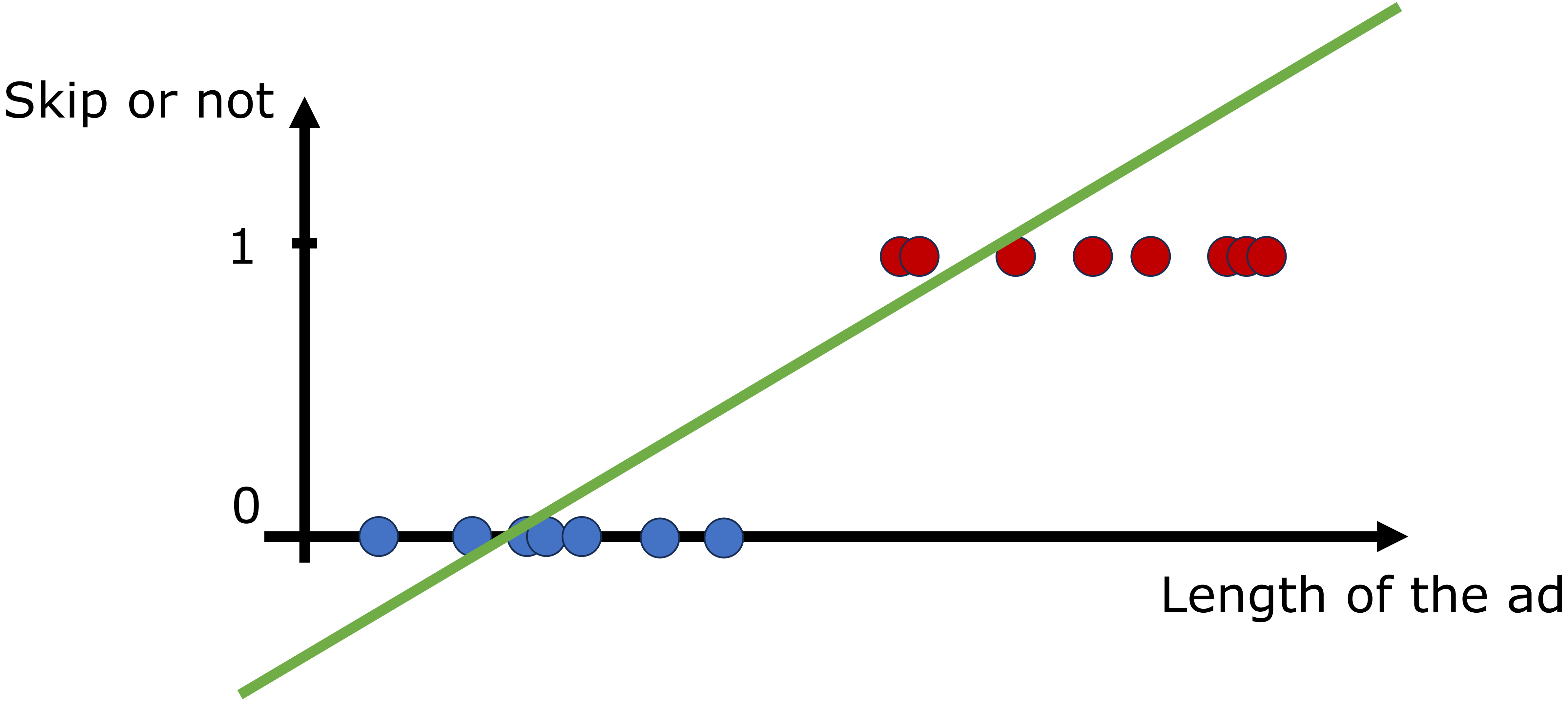
A function that fits the data?



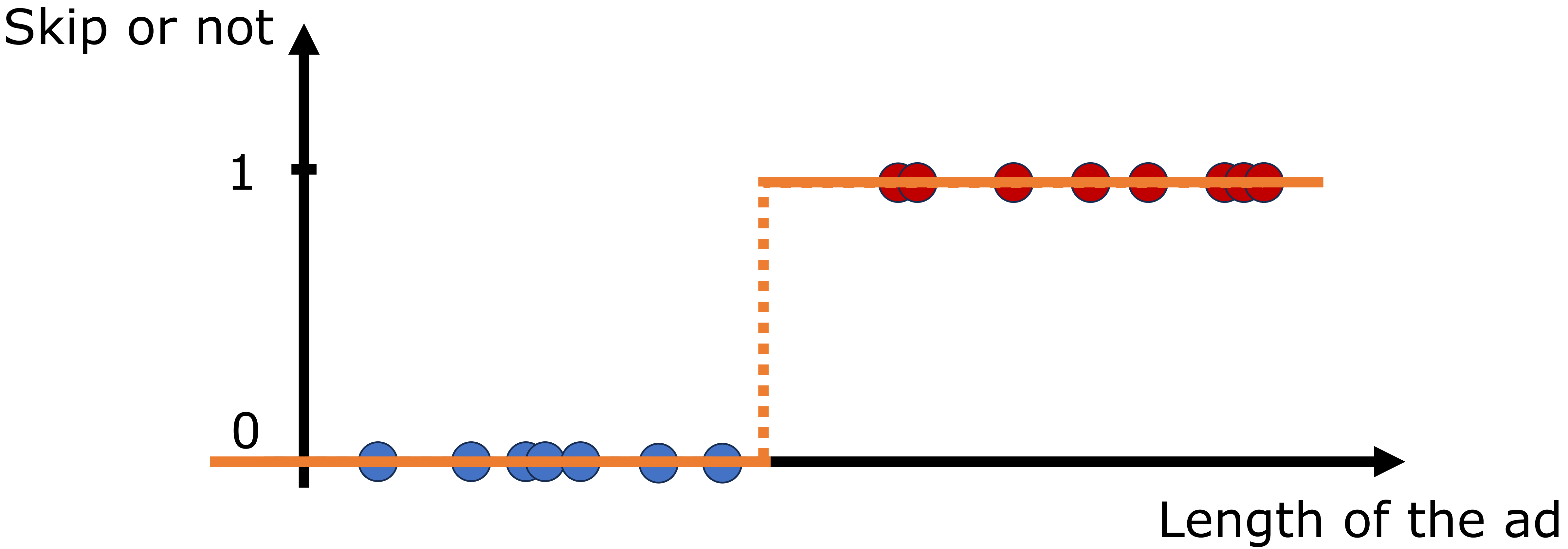
Linear Functions



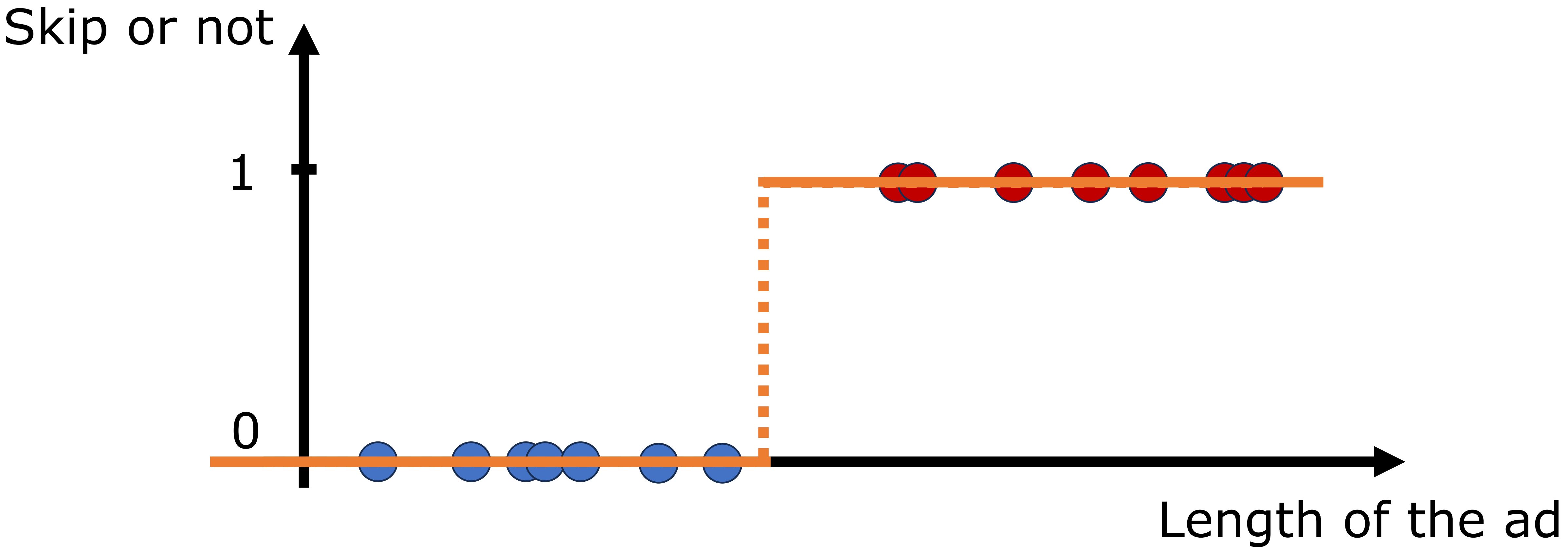
…too far away from the data



Step Functions

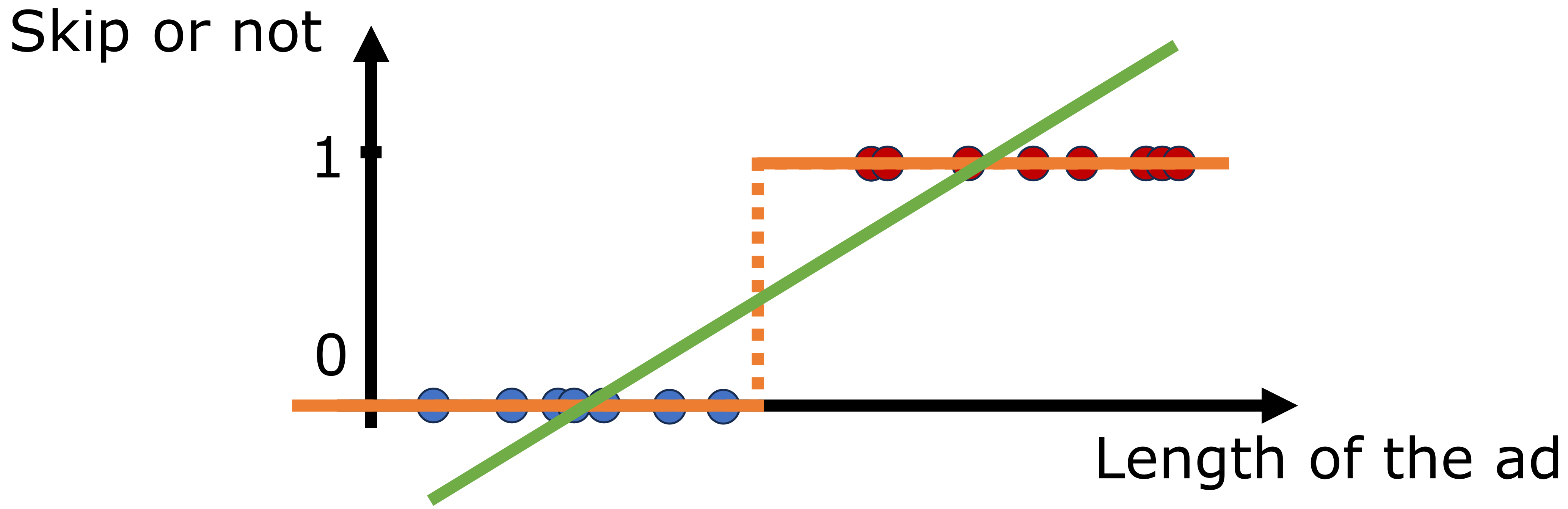


…not continuous



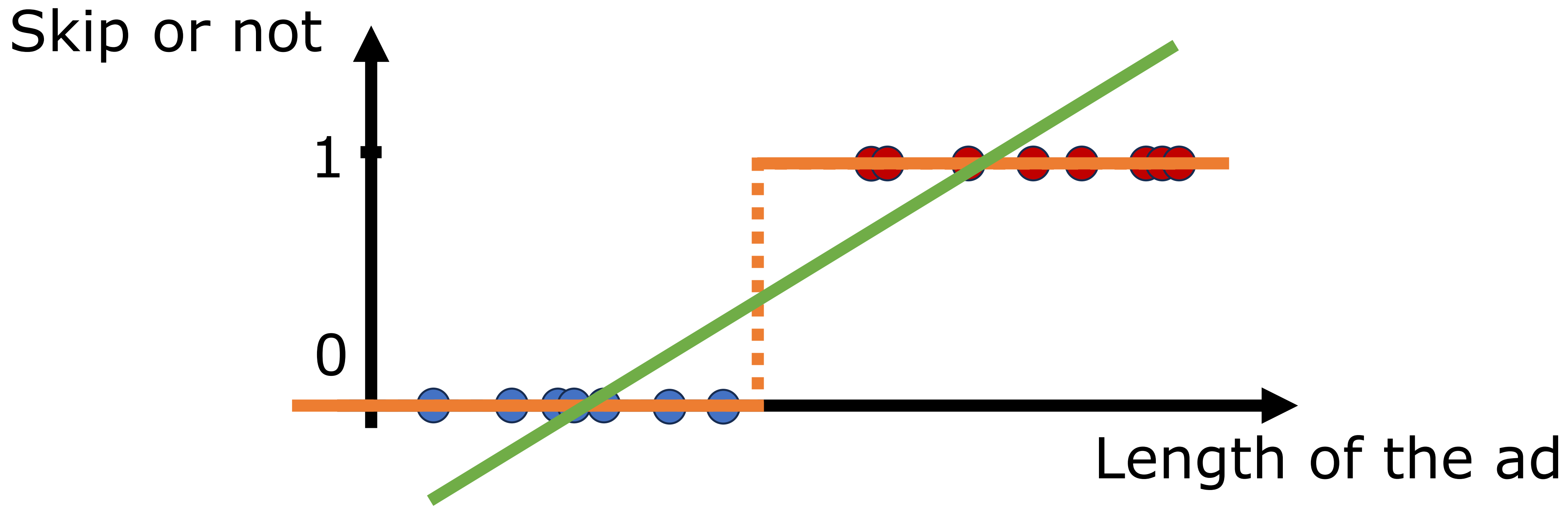
A good function class?

- 1. Close to the data
- 2. Continuous, tractable



A good function class?

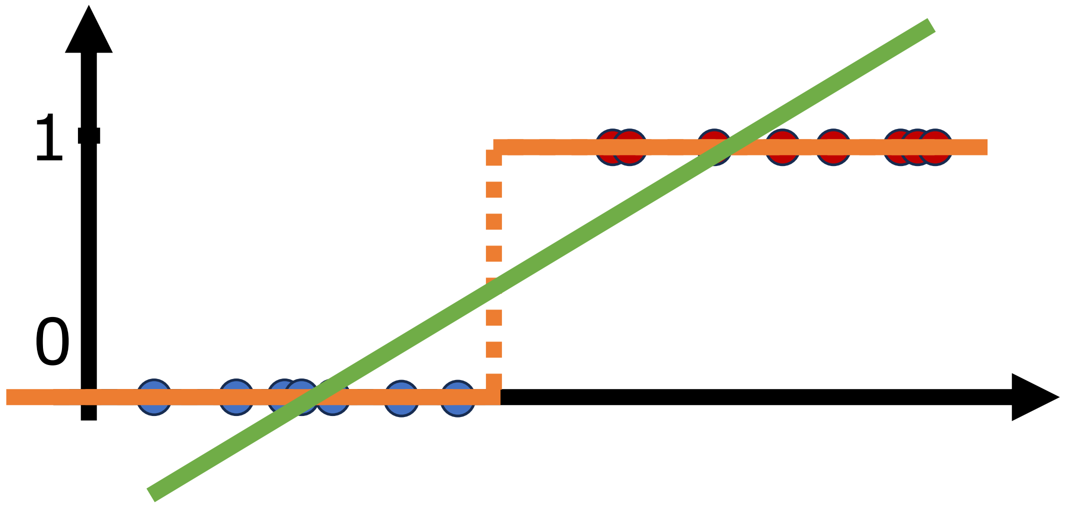
- 1. Close to the data – Step functions
- 2. Continuous, tractable – Linear functions



Can we combine them?

- “Compress”(or transform) the linear function into another continuous function that shapes like the step function.
- The range of linear functions is $[-\infty, \infty]$, but $y_i \in \{0, 1\}$.
- We want to find a function that maps

$$[-\infty, \infty] \rightarrow [0,1]$$

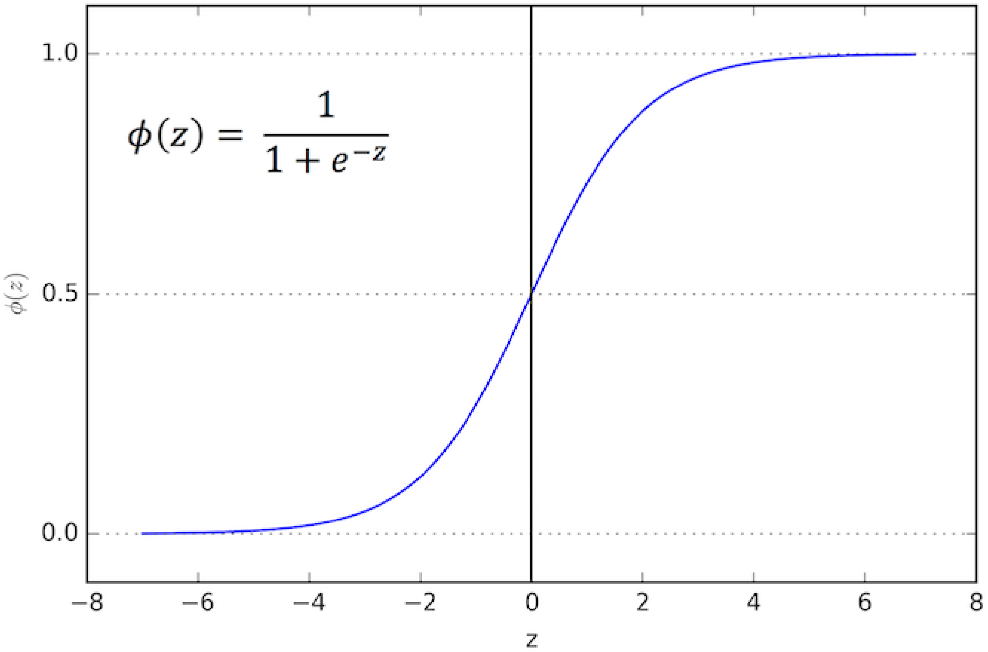


Logistic Function

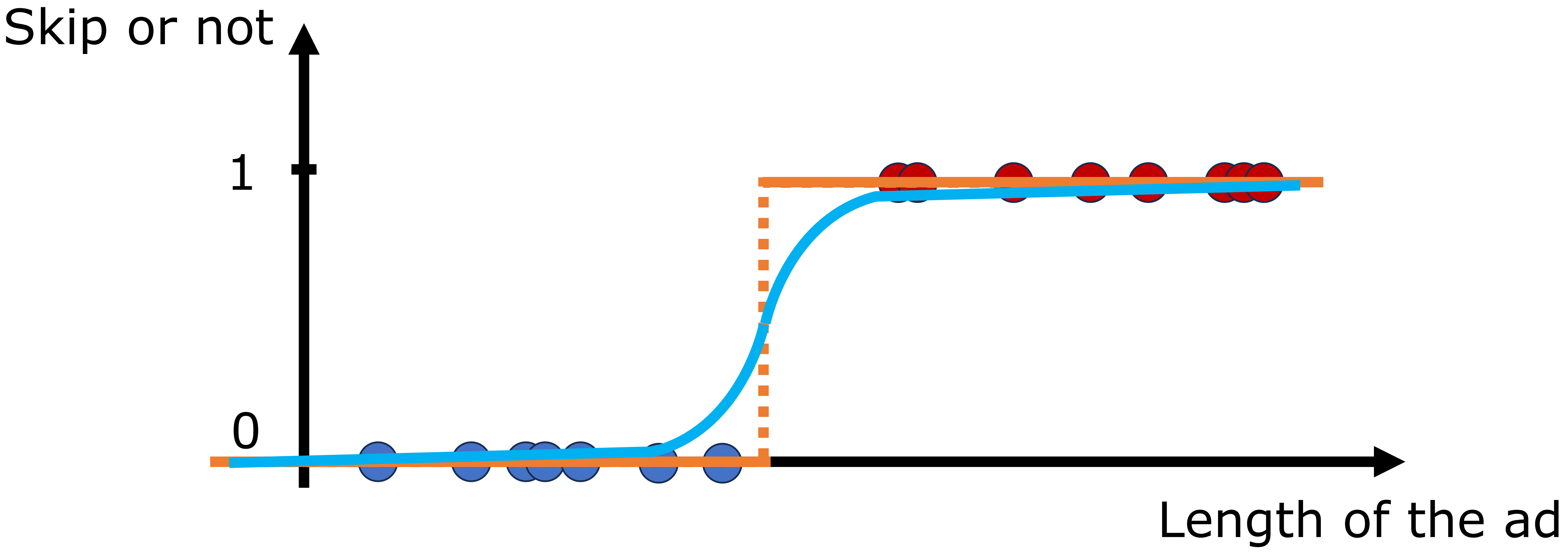
- A standard logistic function is an S-shaped curved

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

- $z \to \infty,\ \sigma \to \frac{1}{1+0} = 1$
- $z \to -\infty,\ \ \sigma \to \frac{1}{1+\infty} = 0$
- $z = 0,\ \sigma = \frac{1}{2}$



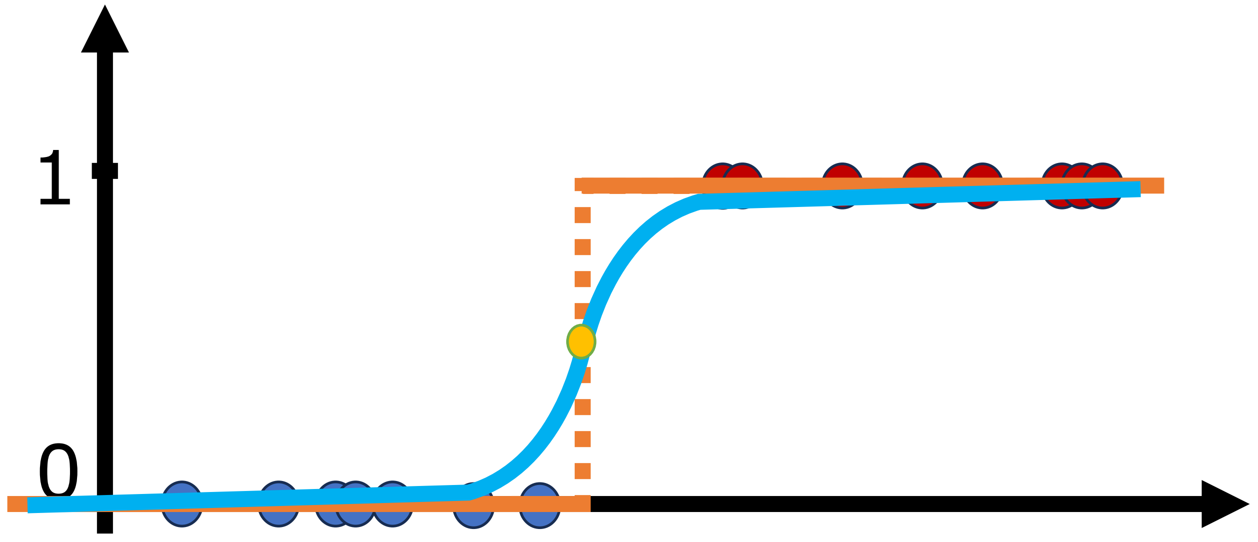
Smooth the step function



Logistic Functions

- Linear function: $z = wx + b$.
- $H(x) = \sigma(wx + b) = \frac{1}{1+e^{-(wx+b)}}$
- H is the standard logistic function composed with the linear function.

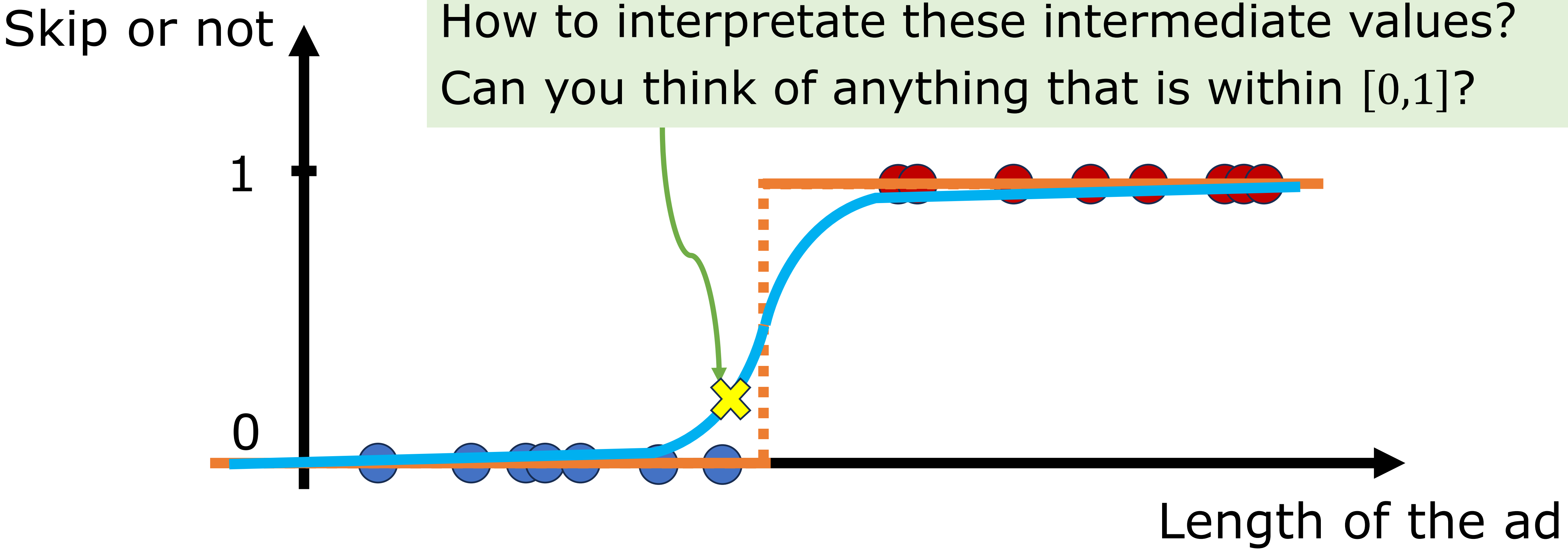
- w: the growth rate
- b: affect the middle point



Interpretation: Logistic Functions

How to interpretate these intermediate values?

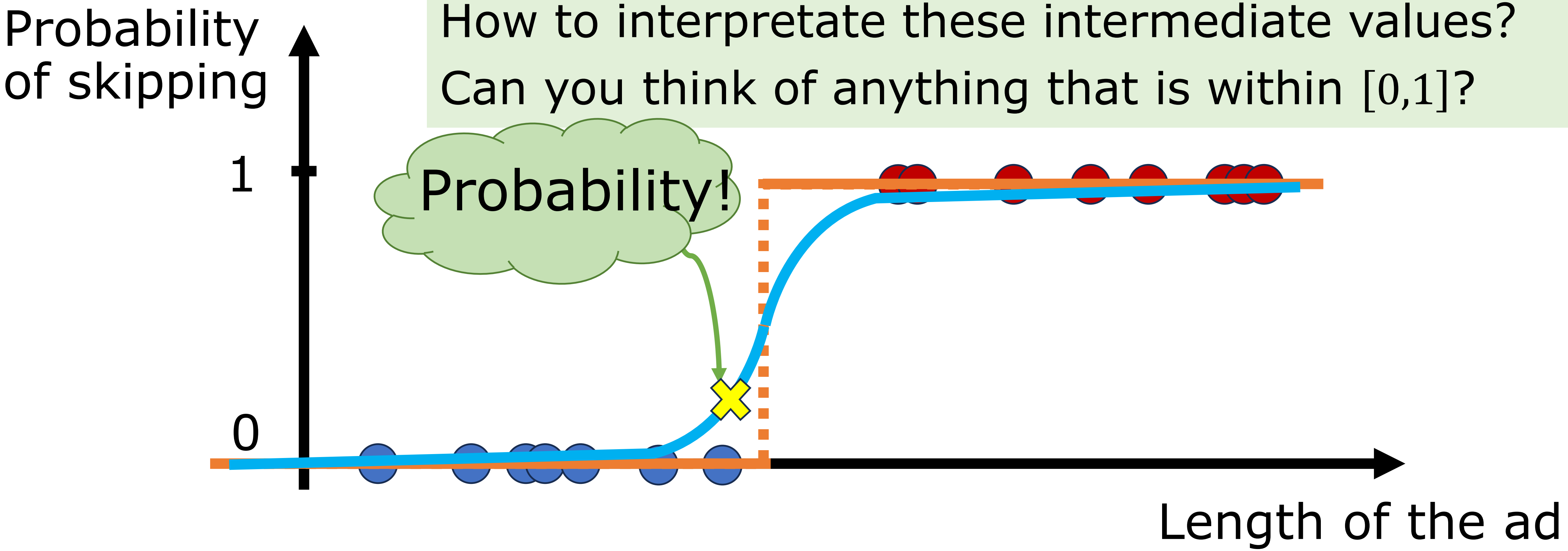
Can you think of anything that is within $[0,1]$?



Interpretation: Logistic Functions

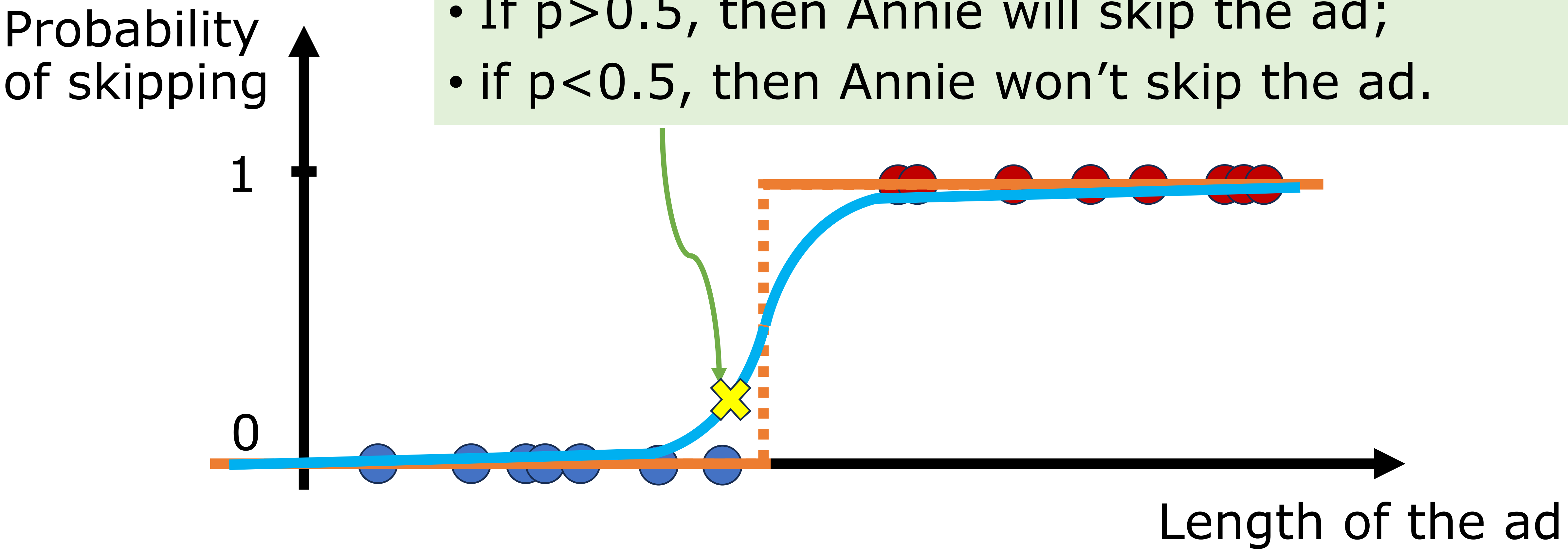
How to interpretate these intermediate values?

Can you think of anything that is within $[0,1]$?



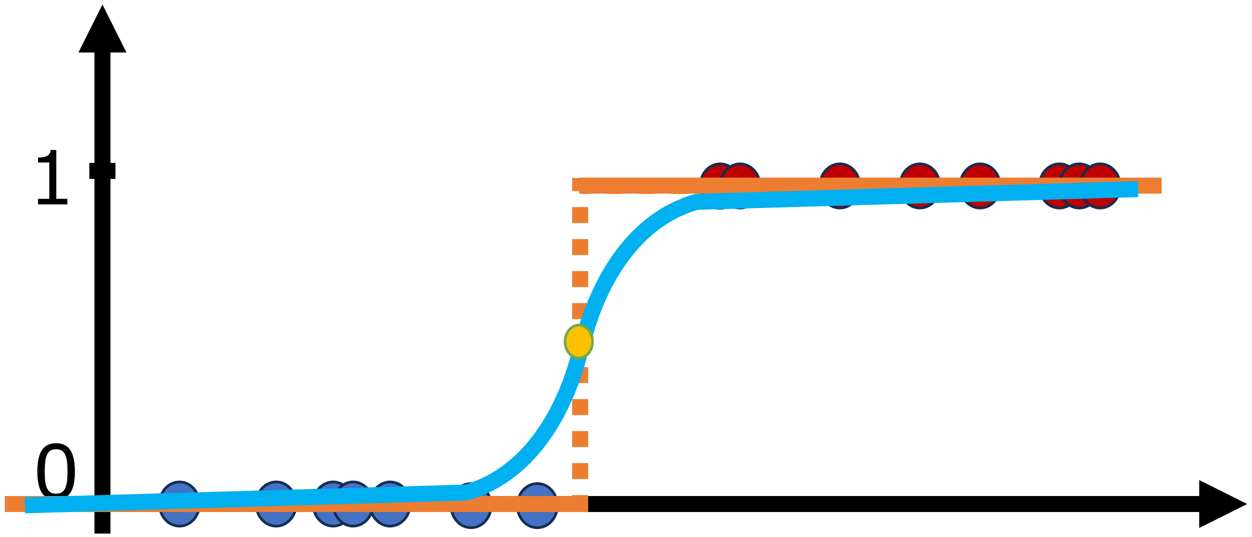
Interpretation: Logistic Functions

- Predict whether Annie will skip the ad.
- Predict the chance that Annie will skip the ad.
- If p>0.5, then Annie will skip the ad;
- if p<0.5, then Annie won’t skip the ad.



Prediction rules

- $H(x) = \sigma(wx + b) = \frac{1}{1+e^{-(wx+b)}}$
- w: the growth rate; b: affect the middle point
- With different values of w and b, we get different prediction rules.
- Which one is the best?

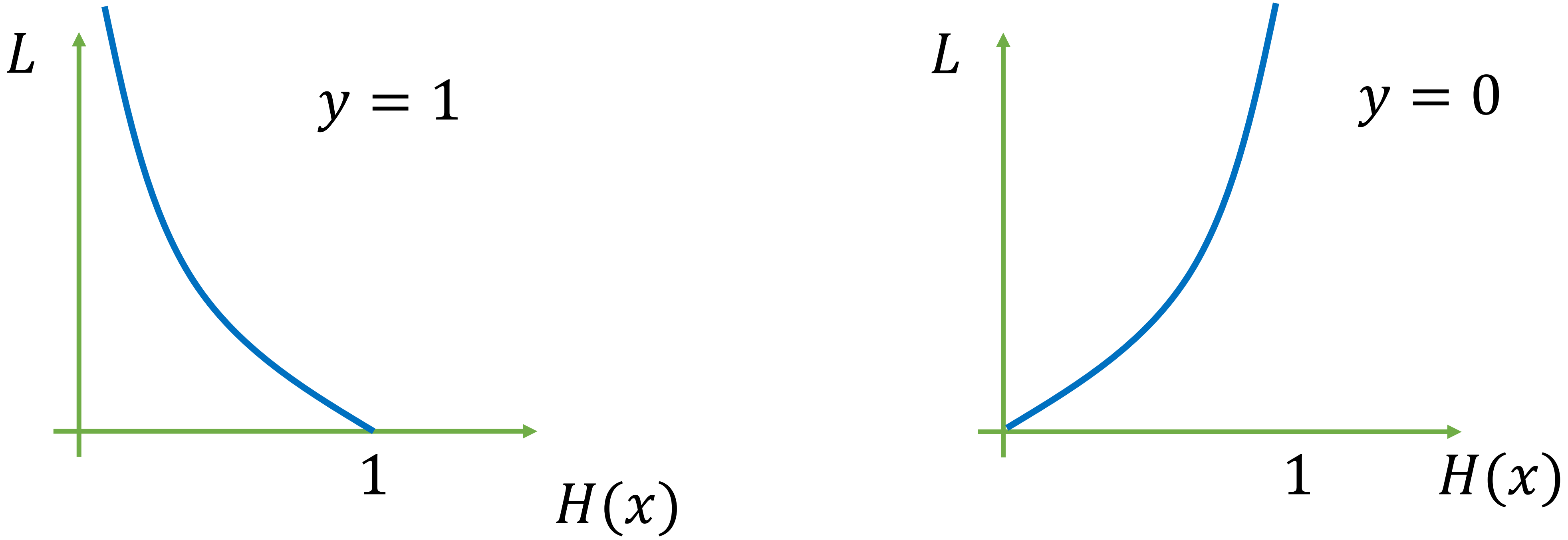


Find the optimal parameters w^*, b^*

- In linear regression, how do we find the best prediction rule?
- We find the prediction line that minimizes the mean square error (MSE)!
- We also need to define a loss function for logistic regression. And the best logistic model is the one that minimizes this loss function.

Loss function

- $L(H(x), y) = \begin{cases} -\ln H(x), & if\ y = 1 \\ -\ln\big(1 - H(x)\big), & if\ y = 0 \end{cases}$



Loss function

- $L(H(x), y) \ = \begin{cases} -\ln H(x), & if\ y = 1 \\ -\ln\big(1 - H(x)\big), & if\ y = 0 \end{cases}$
- These can be combined to a single expression:

$$L(H(x), y) = -y \ln H(x) - (1 - y) \ln\big(1 - H(x)\big)$$

Find the optimal parameters w^*, b^*

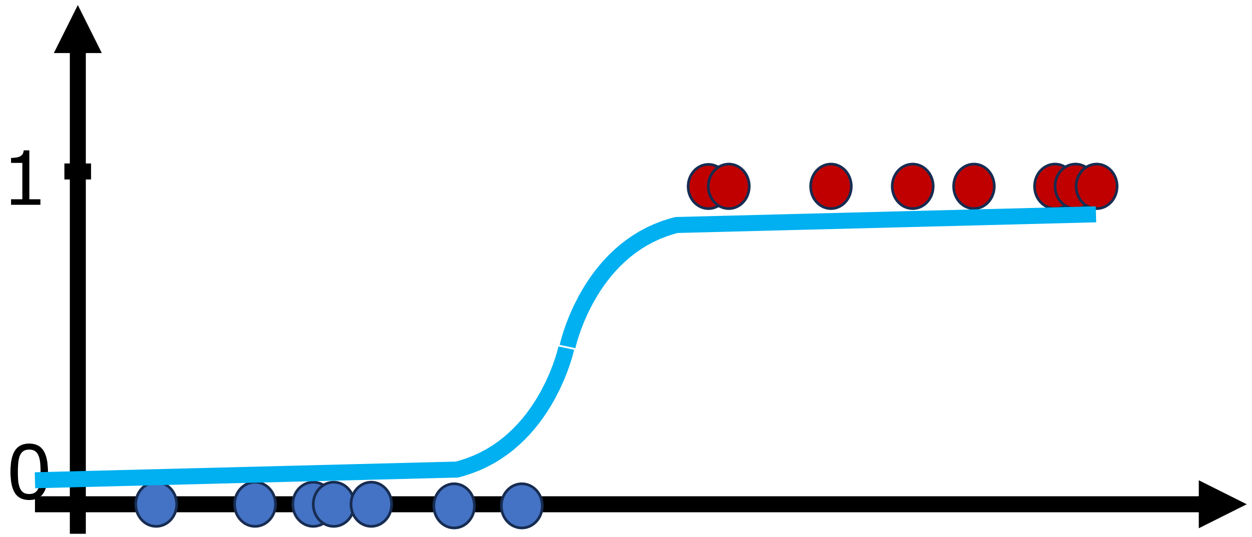
- We find w^*, b^* that minimizes the total loss function (log-likelihood).
- Gradient Descent

Discussion

- Why not just use the least squares in logistic regression?
- What is the difference between “classification” and “regression”?
- Since logistic regression is an algorithm for classification, why is it called logistic “regression” instead of logistic “classification”?

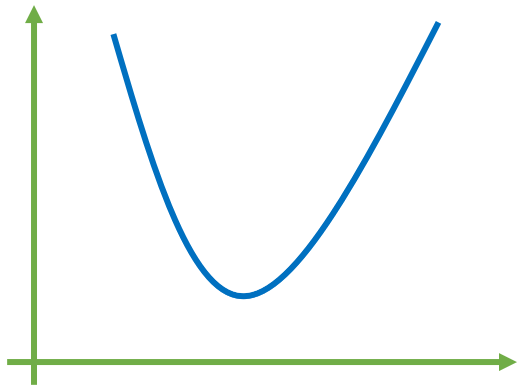
Why not just use the least squares in logistic regression?

1. When $y = 0$ but $H(x) \approx 1$, we make the wrong prediction. However, in this case, the square loss is $(H(x) - y)^2 = 1$, which is not large enough. We want to assign more punishment in this case. That is what we do in log loss. In log loss, if $y = 0$ but $H(x) \approx 1$, $L(H(x), y) \approx \infty$.

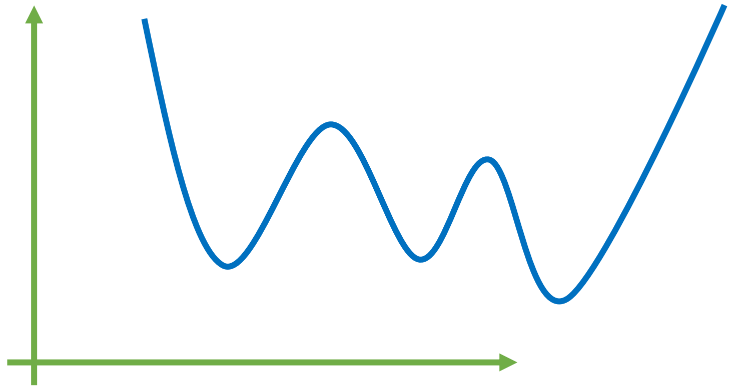


Why not just use the least squares in logistic regression?

2. If we use least squares, the objective function in the optimization step would be non-convex. In this case, gradient descent is not guaranteed to converge to a global minimum.



Convex

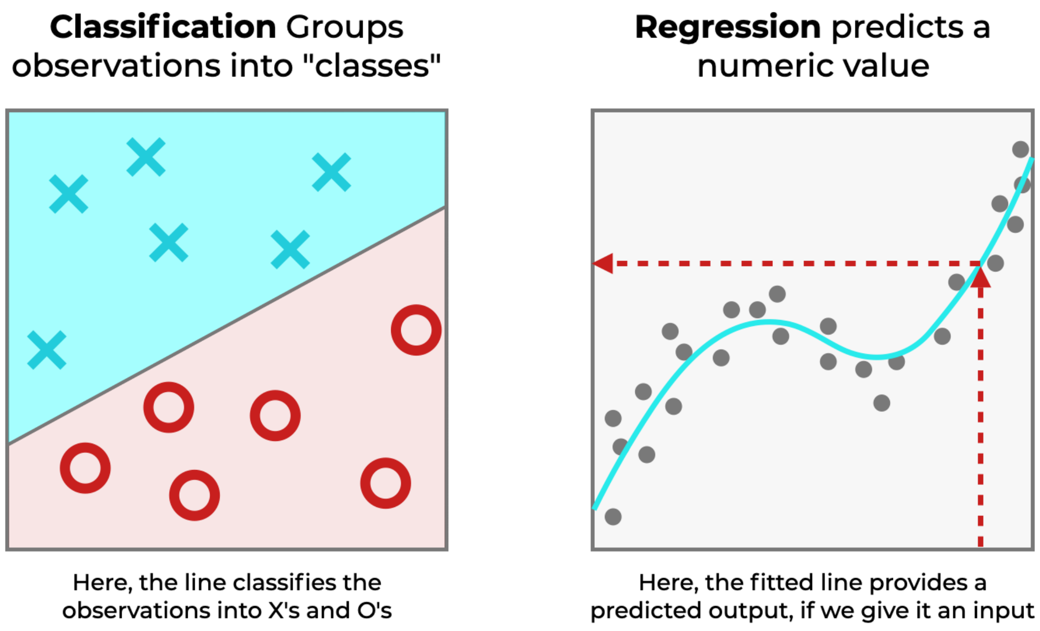


Non-convex

3. Other reasons…

What is the difference between “classification” and “regression”?

- Classification is about predicting a label (discrete): yes or no
- Regression is about predicting a quantity (often continuous): price, salary…



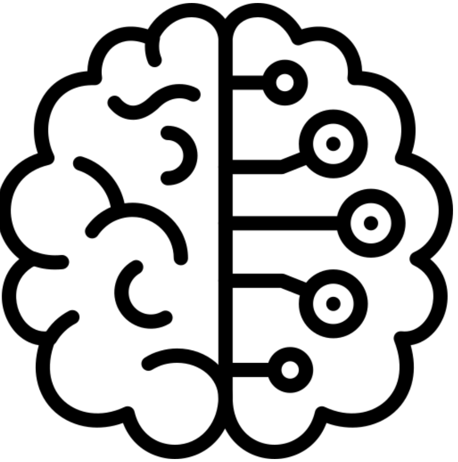
https://r-craft.org/r-news/regression-vs-classification-explained/

Since logistic regression is an algorithm for classification, why is it called logistic “regression” instead of logistic “classification”?

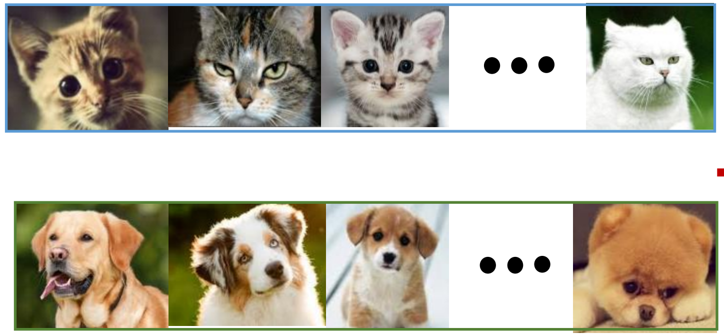
- "Logistic regression" is named based on its historical development from linear regression. Though logistic regression is indeed an algorithm primarily used for classification tasks, it predicts a continuous value of the probability $P(Y = 1)$. Logistic regression is a *generalized linear model*. And it uses the same basic technique of linear regression but it is regressing for the probability of a categorical outcome. If you are interested in the origin of logistic regression, please read: https://papers.tinbergen.nl/02119.pdf

Regularization and evaluation metrics

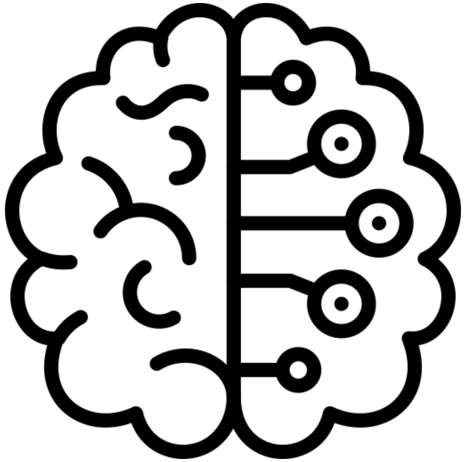
sources used:
1) UCSDX, Prof. Dasgupta fundamentals of machine learning
2) SUT, Prof. Sharifi Zarchi, Introduction to machine learning



Classification problem



Training data

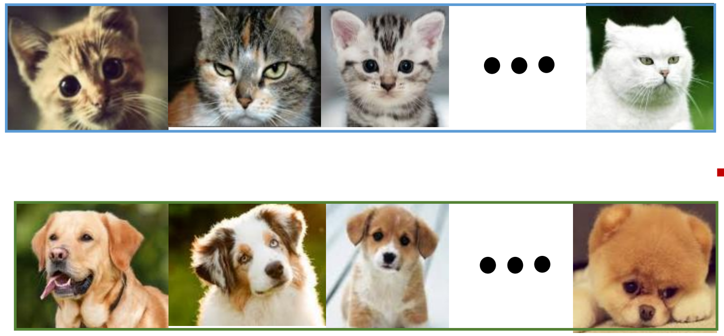


Trained model

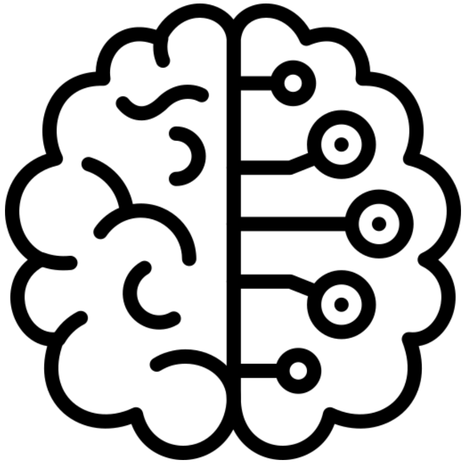
Inference data

Is accuracy the most important evaluation metric?

Accuracy and Confusion Matrix



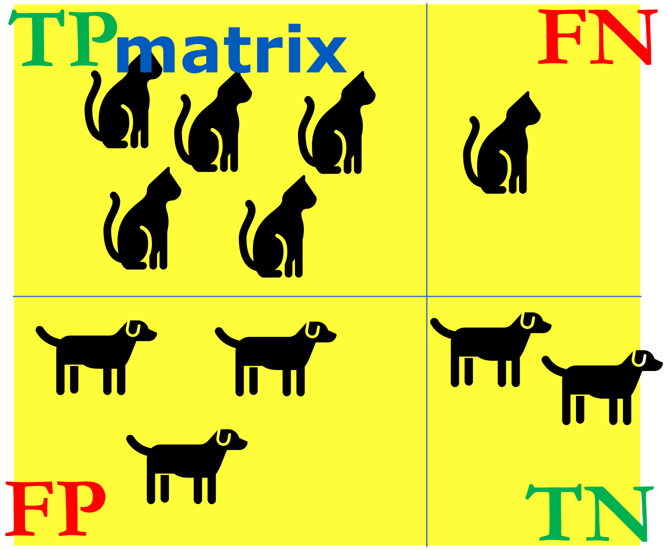
Training data



Trained model

Inference data

Confusion matrix



$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Other metrics

Accuracy is not always the best metric

Let's say we have two tests A and B, their confusion matrix are as following.
Which one is a better test? Whose accuracy is higher?

TP FN

7	3
40	950

FP TN

TP FN

0	10
0	990

FP TN

Other metrics

		Predicted		
		Negative (0)	Positive (1)	
Actual	Negative (0)	**True Negative** **TN**	**False Positive** **FP** (Type I error)	**Specificity** $= \frac{TN}{TN + FP}$
	Positive (1)	**False Negative** **FN** (Type II error)	**True Positive** **TP**	**Recall**, **Sensitivity**, True positive rate (**TPR**) $= \frac{TP}{TP + FN}$
		Accuracy $= \frac{TP + TN}{TP + TN + FP + FN}$	**Precision**, Positive predictive value (**PPV**) $= \frac{TP}{TP + FP}$	**F1-score** $= 2 \times \frac{Recall \times Precision}{Recall + Precision}$

Image source

Precision – recall trade-off

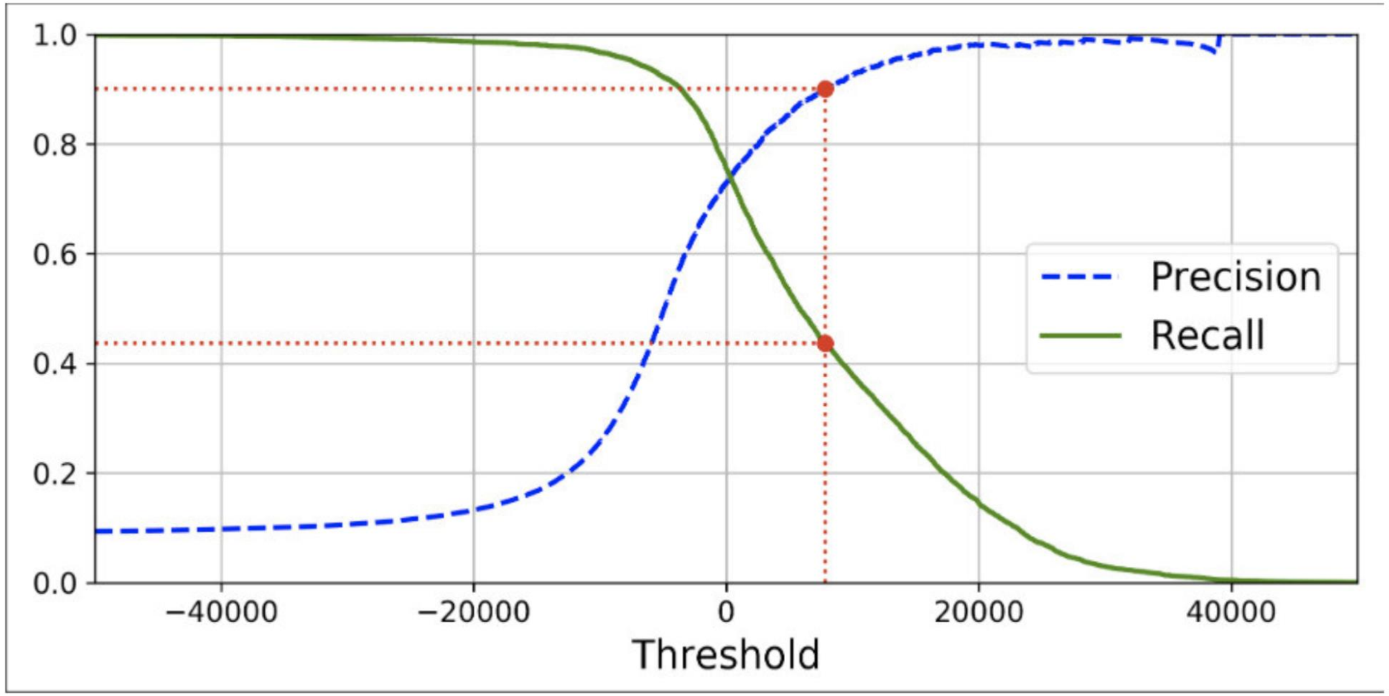


Image source

Other metrics: Area under ROC curve (AUC)

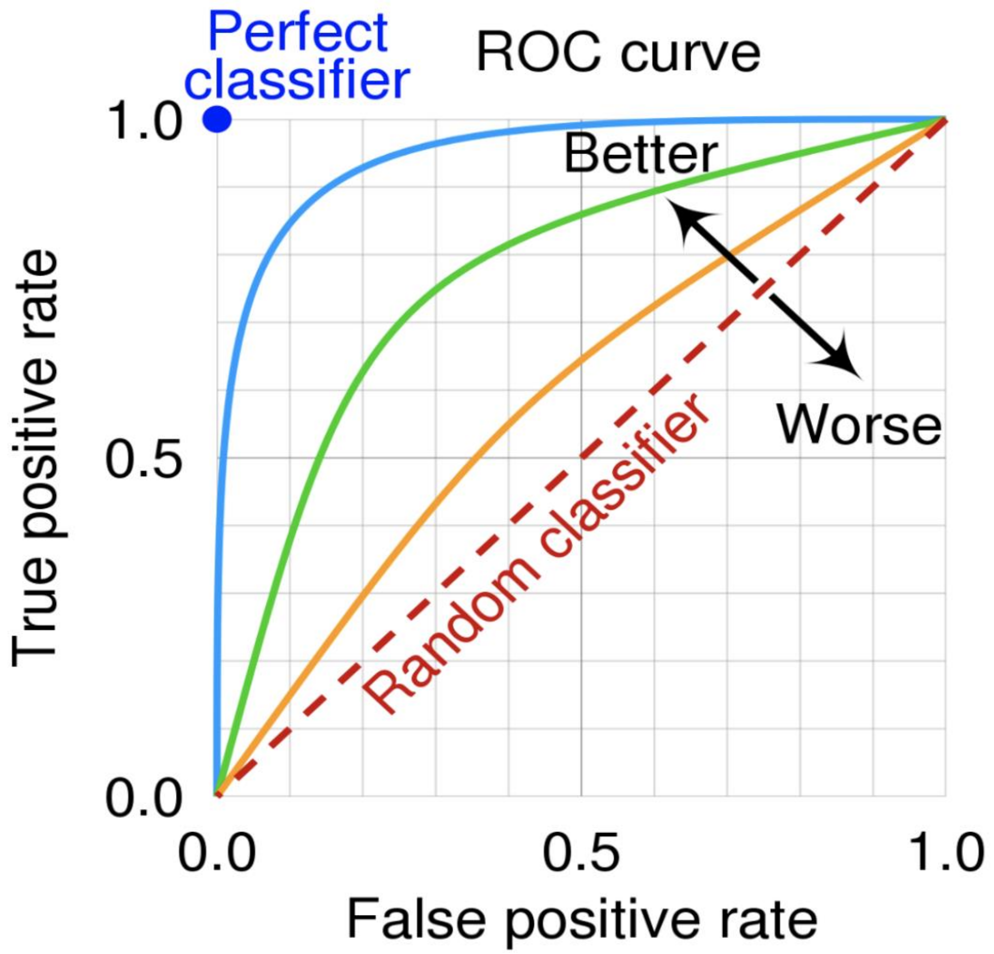
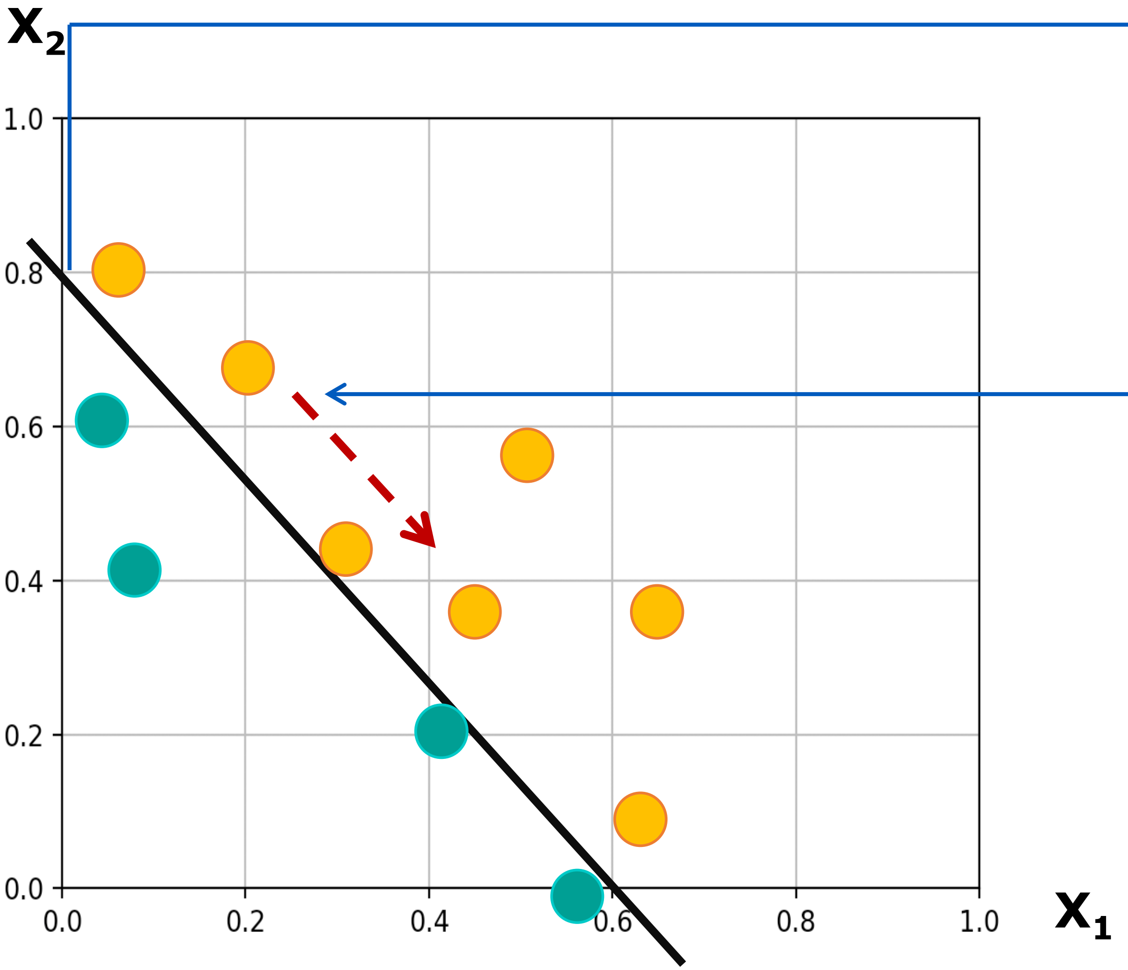


Image source

Simple linear regression



$$y = mx + b$$

$$b = 0.8$$

$$m = \frac{0 - 0.8}{0.6 - 0} = \frac{-4}{3}$$

$$y = \frac{-4}{3}\mathrm{x} + \frac{4}{5}$$

Least Square Regression

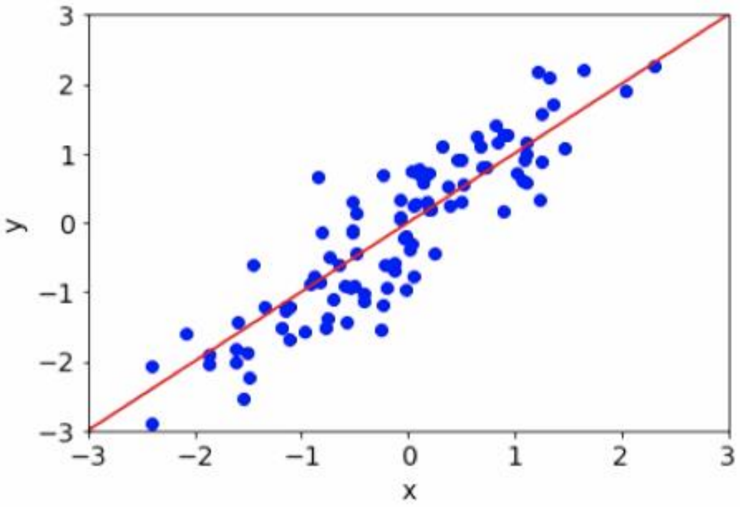
Given a training set $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$, find a linear function, given by $w \in R^d$ and $b \in R$, that minimizes the squared loss:

$$L(w, b) = \sum_{i=1}^{n} (y^i - (w.x^i + b))^2$$

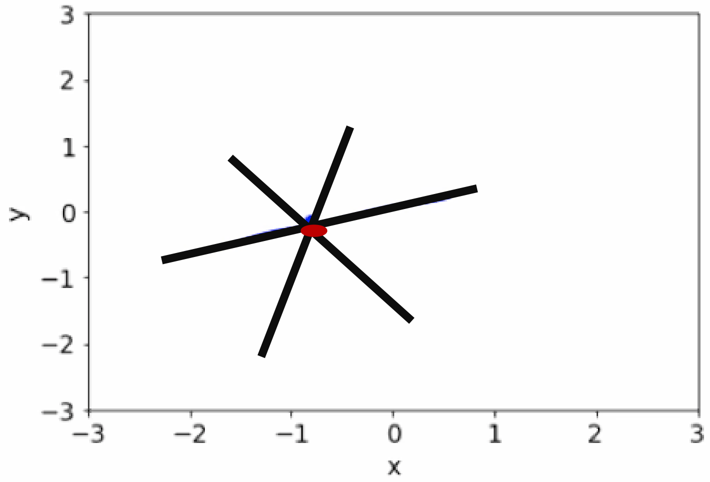
Is training loss a good estimator of future performance?

If n is large enough, maybe
Otherwise, probably an underestimate of future error

Example



n is large given the number of parameters so probably training loss is fine.



We can have multiple different lines and they all will have zero training error, so training loss is not necessarily a good estimator of the performance for future data.

How do we get a good estimate of future error?

k-fold cross validation

Divide the data set into k equal-sized groups $S_1, S_2, \ldots, S_n$

For i=1 to k:

  Train a regressor on all data except S_i

  Let E_i be its error on S_i

Error estimate: average of $E_1, E_2, \ldots, E_k$

But what if training error is not a good measure of future prediction?

Regularization

Minimize squared loss plus a term that penalizes "complex" w:

$$\boldsymbol{L(w,b)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} (\boldsymbol{y^i} - (\boldsymbol{w.x^i} + \boldsymbol{b}))^2 + \boxed{\lambda||w||^2} \rightarrow \text{regulizer}$$

Adding a penalty term like the above is called regularization

$\texttt{constant}\ \lambda = 0 \longrightarrow$ Least square solution

$\texttt{constant}\ \lambda \rightarrow\rightarrow \infty \longrightarrow$ Only the regulizer matters, we set w=0 (no data)

Example

Given a training set $\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(100)}, y^{(100)}\right)$ and the following train/test MSE

λ	training MSE	test MSE	
0.00001	0.00	585.81	→ **least squares**
0.0001	0.00	564.28	
0.001	0.00	404.08	
0.01	0.01	83.48	
0.1	0.03	19.26	
1.0	0.07	7.02	
10.0	0.35	2.84	
100.0	2.40	5.79	
1000.0	8.19	10.97	
10000.0	10.83	12.63	→ **w is almost 0**

Lasso and Ridge Regression

Lasso

$$\boldsymbol{L(w,b) = \sum_{i=1}^{n}(y^i - (w.x^i + b))^2} + \boxed{\lambda||w||^2} \rightarrow \text{regulizer}$$

Ridge

$$\boldsymbol{L(w,b) = \sum_{i=1}^{n}(y^i - (w.x^i + b))^2} + \boxed{\lambda||w||} \rightarrow \text{regulizer}$$

produces a sparse w
More interpretable and simple