Main challenges of machine learning

Insufficient quantity of training data

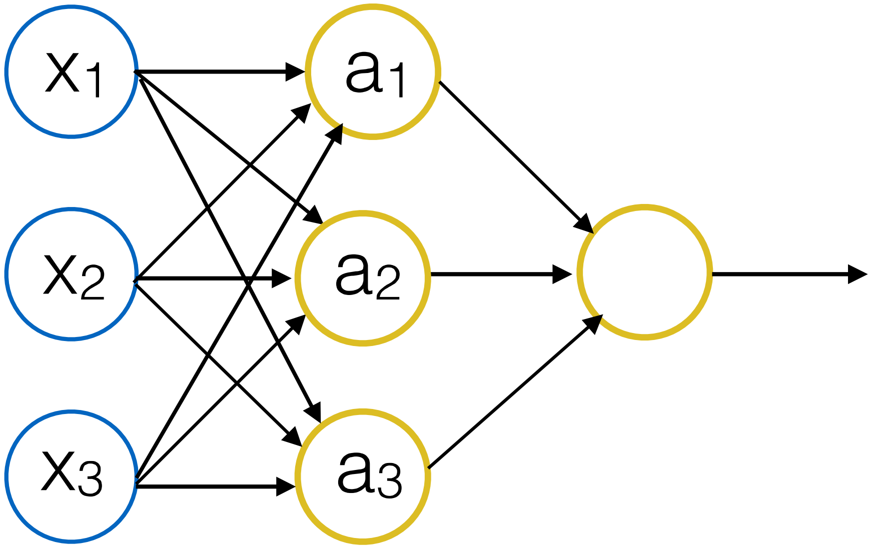
Non-representative training data

Poor quality data

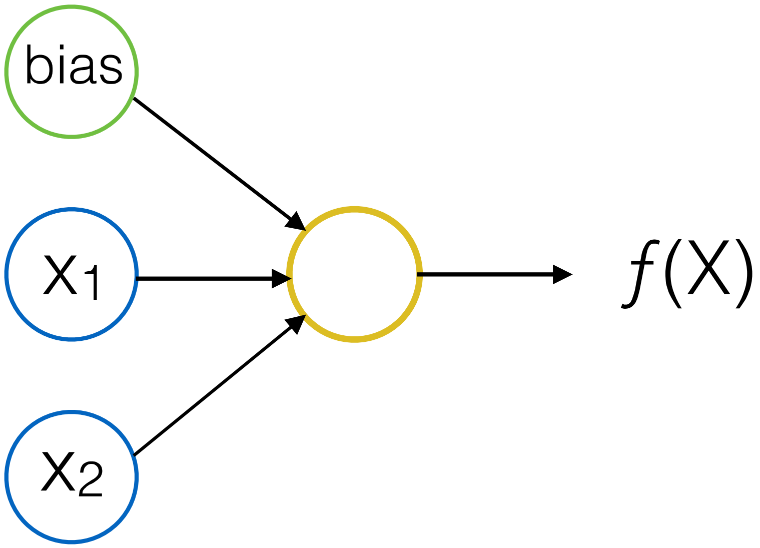
Irrelevant features

Neural Network

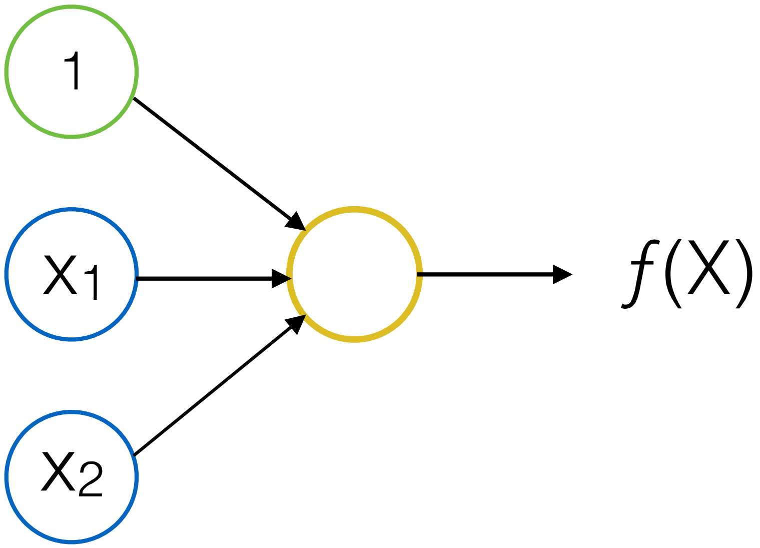
Neural Network



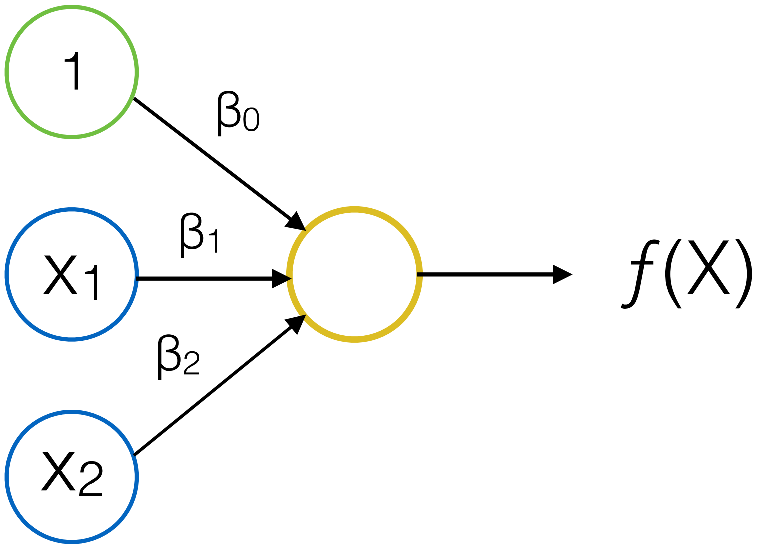
Perceptron



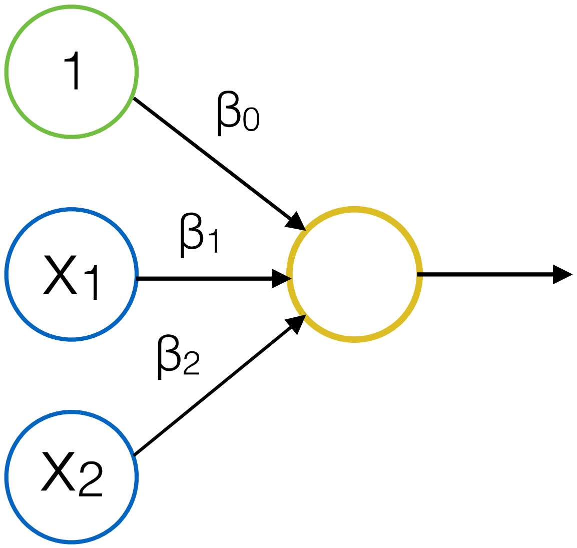
Perceptron



Perceptron



Perceptron

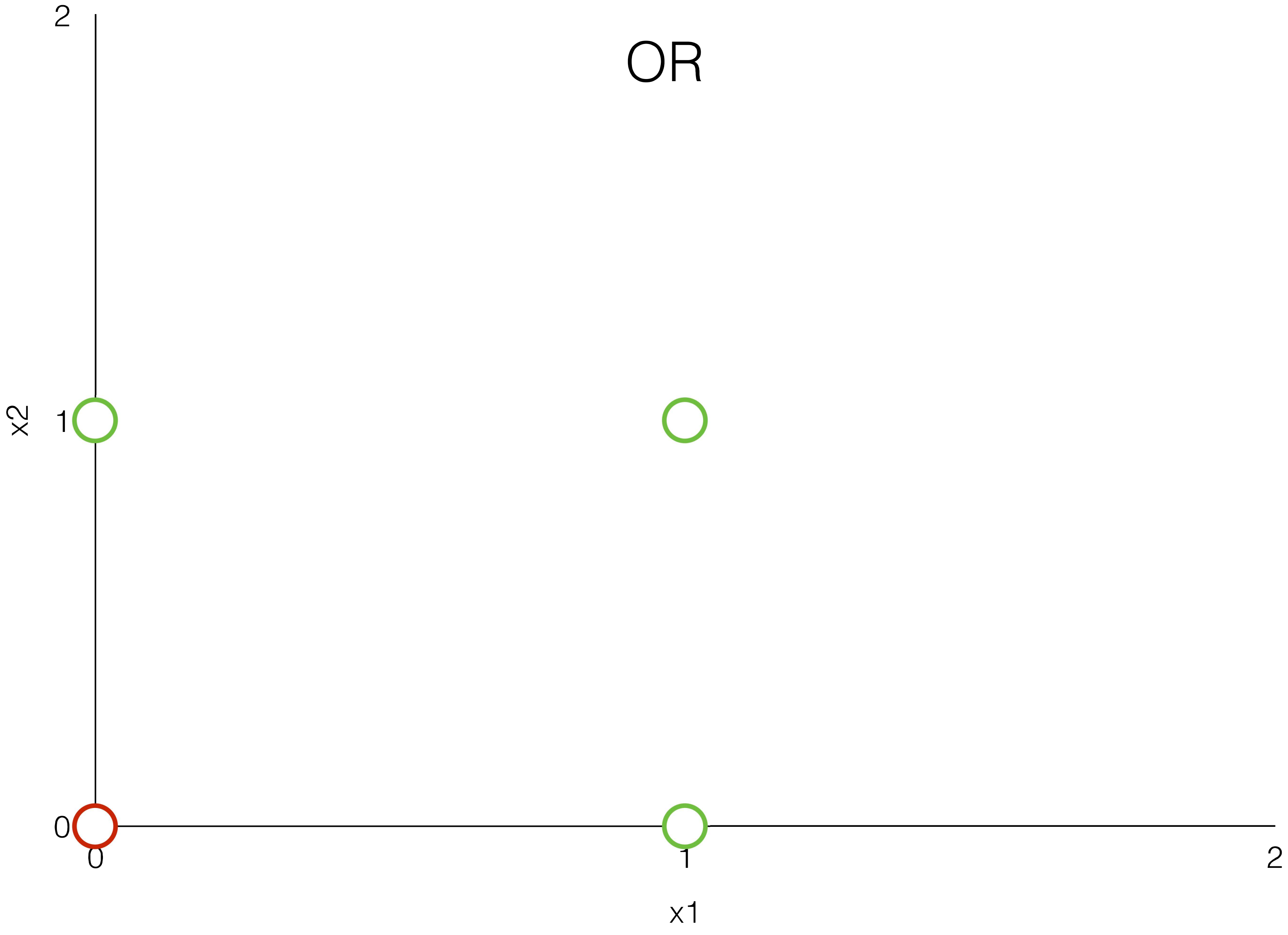


$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

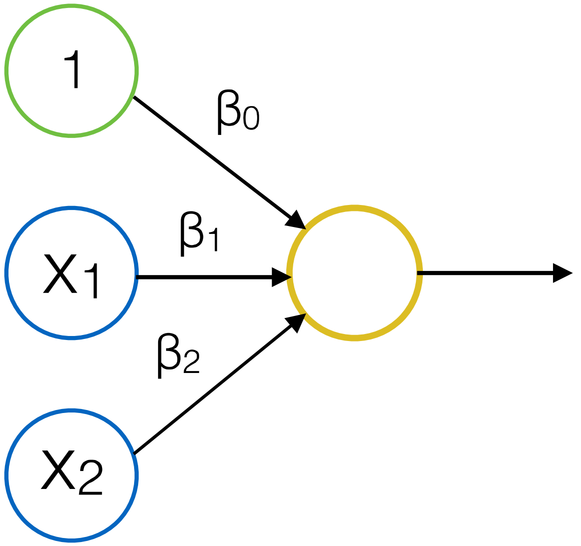
OR

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1

OR

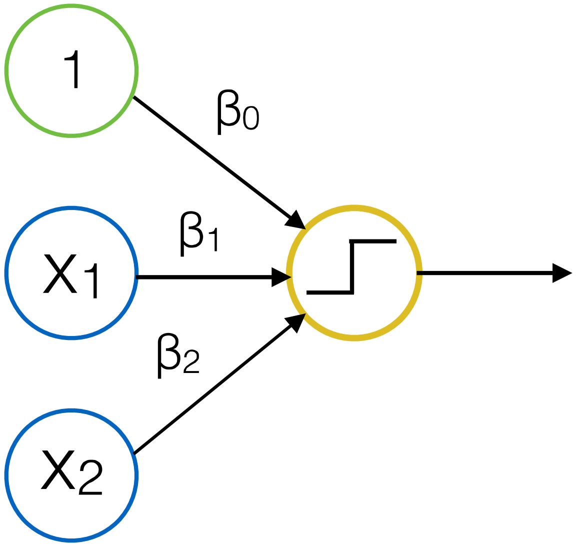


Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



$$f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

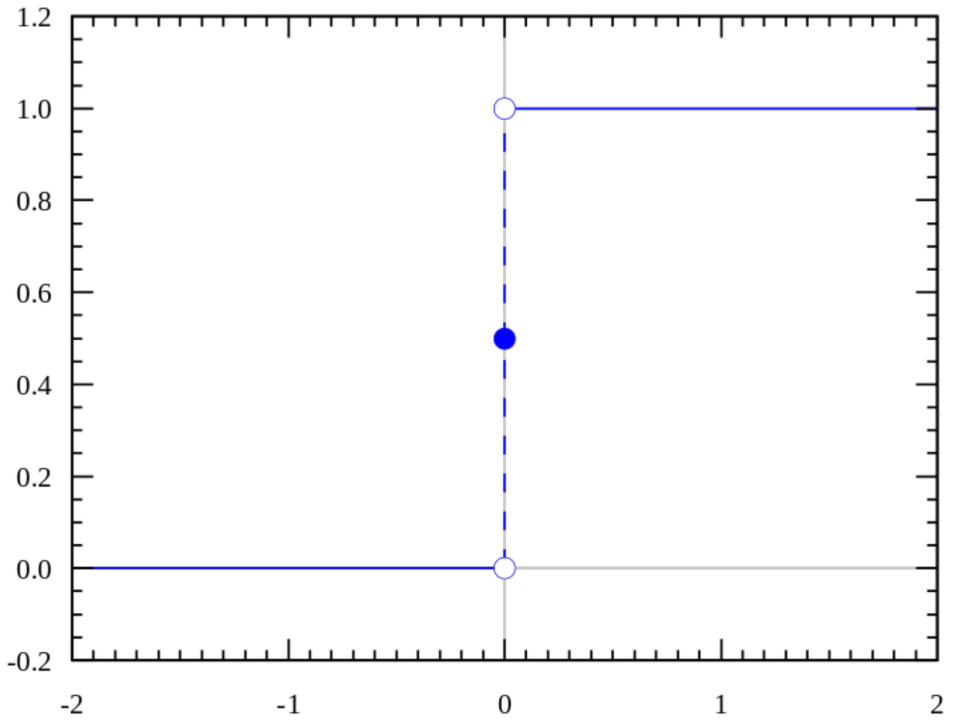
Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



$$f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Activation Function: Threshold

step function

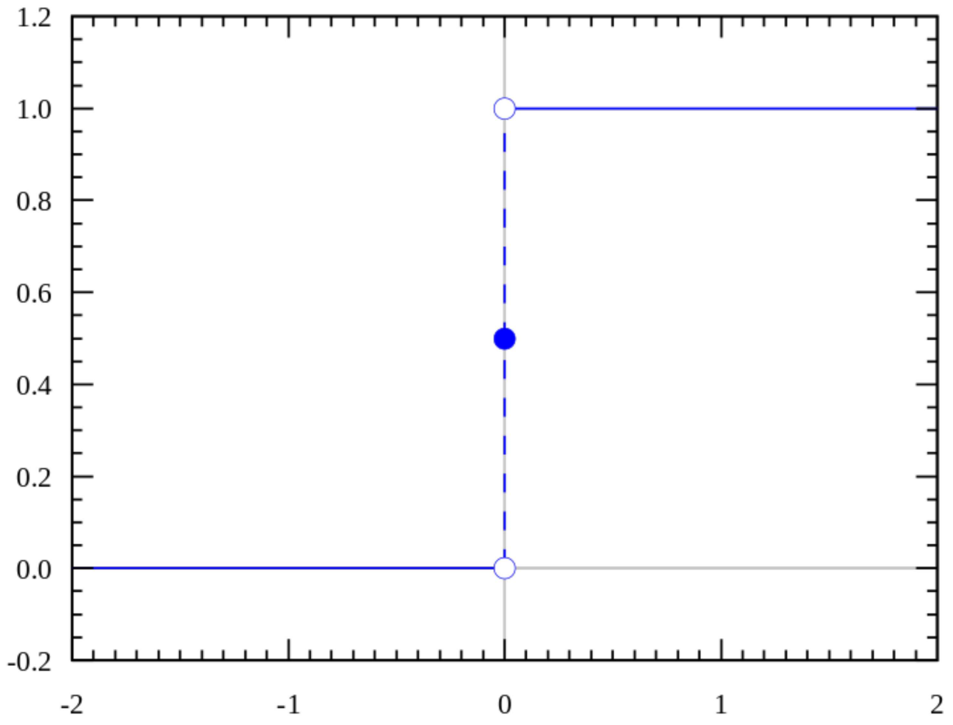


Activation Function: Threshold

if $\beta_0 + \beta_1 x_1 + \beta_2 x_2 > 0$:   1

Else: 0

step function



Activation Function: Threshold

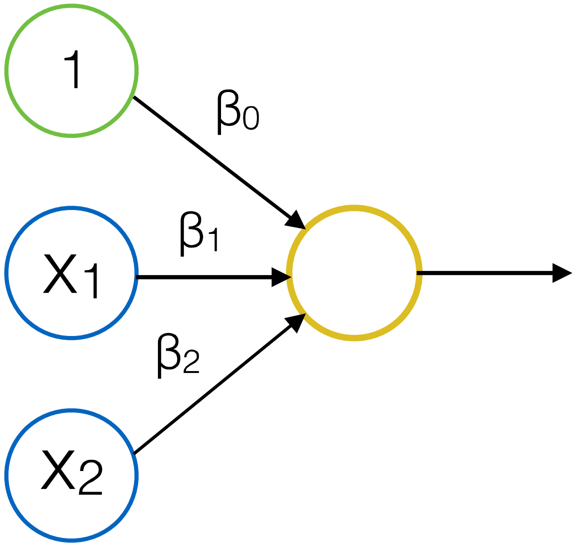
if $\beta_0 + \beta_1 x_1 + \beta_2 x_2 > 0$: 1

Else: 0

Update Rule:

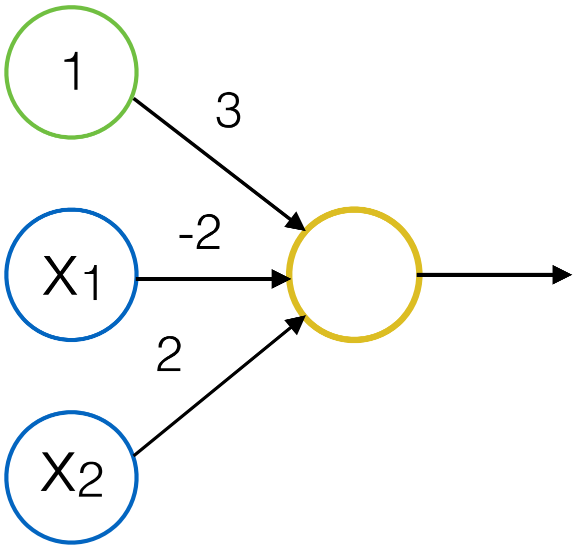
updated weight_i = weight_i - (output - target) * input_i

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



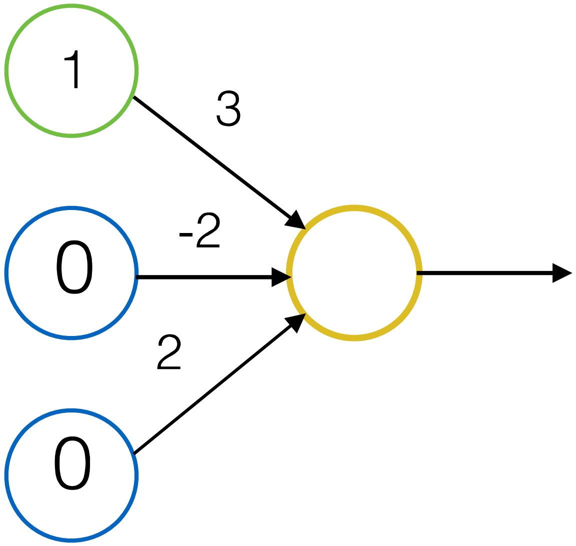
$$f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



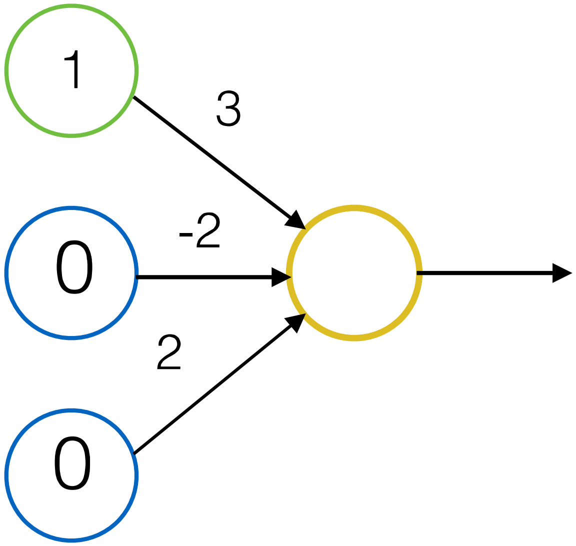
$$f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



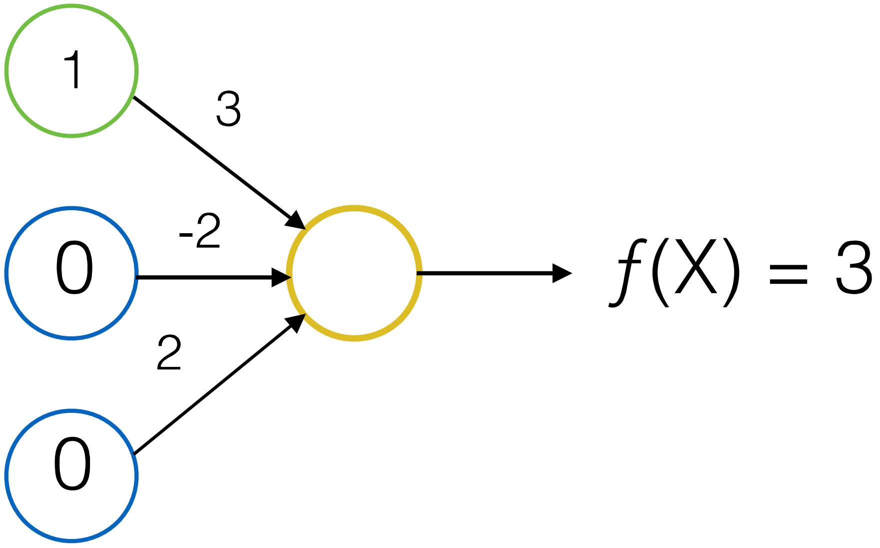
$$f(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1

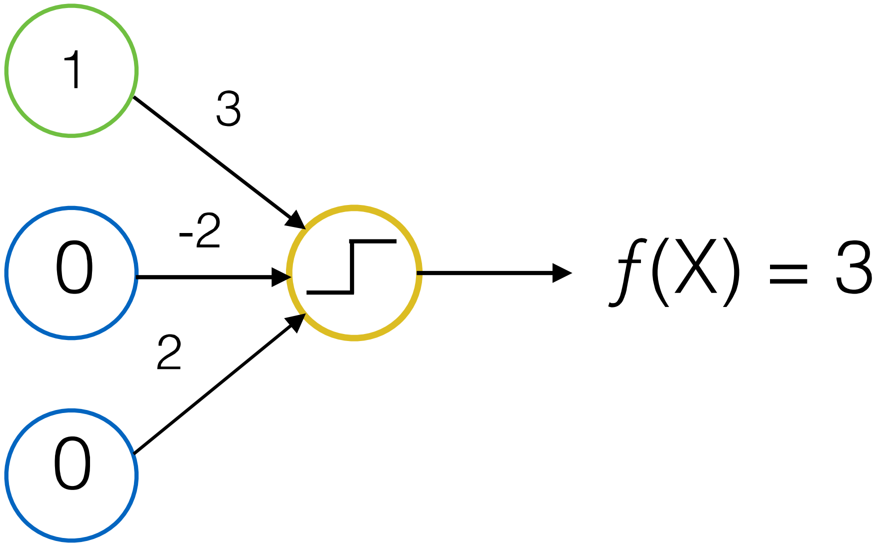


$$f(X) = 3 + 0 + 0$$

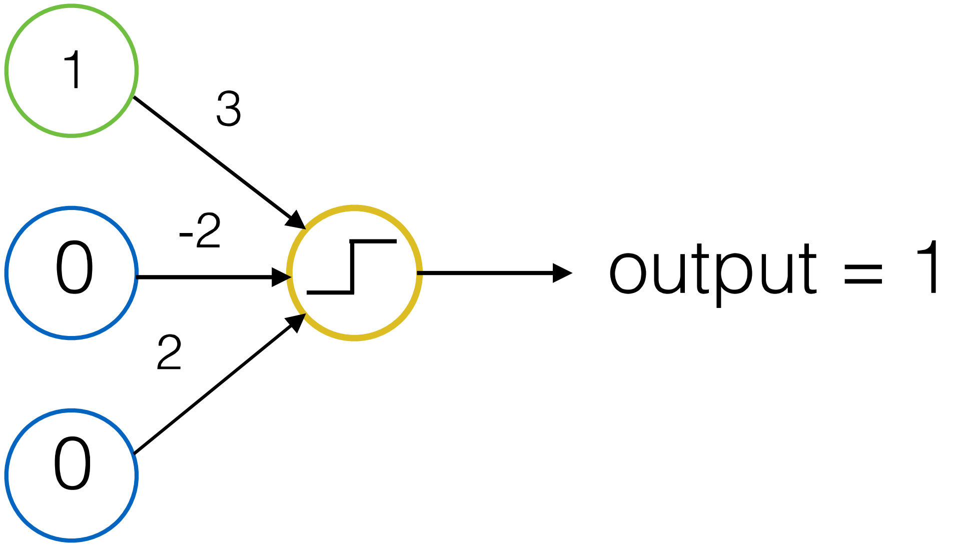
Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



weights: 3, -2, 2

input: 1, 0, 0

output: 1

target: 0

updated weight_0 = weight_0 - (output - target) * input_0

updated weight_1 = weight_1 - (output - target) * input_1

updated weight_2 = weight_2 - (output - target) * input_2

weights: 3, -2, 2

input: 1, 0, 0

output: 1

target: 0

updated weight$_0$ = 3 - (output - target) * input$_0$

updated weight$_1$ = -2 - (output - target) * input$_1$

updated weight$_2$ = 2 - (output - target) * input$_2$

weights: 3, -2, 2

output: 1

input: 1, 0, 0

target: 0

updated weight_0 = 3 - (1 - target) * input_0

updated weight_1 = -2 - (1 - target) * input_1

updated weight_2 = 2 - (1 - target) * input_2

weights: 3, -2, 2

output: 1

input: 1, 0, 0

target: 0

updated weight_0 = 3 - (1 - 0) * input_0

updated weight_1 = -2 - (1 - 0) * input_1

updated weight_2 = 2 - (1 - 0) * input_2

weights: 3, -2, 2   **output**: 1

input: 1, 0, 0   **target**: 0

updated weight_0 = 3 - (1 - 0) * 1

updated weight_1 = -2 - (1 - 0) * 0

updated weight_2 = 2 - (1 - 0) * 0

weights: 3, -2, 2 **output**: 1

input: 1, 0, 0 **target**: 0

updated weight_0 = 3 -1

updated weight_1 = -2 - 0

updated weight_2 = 2 - 0

weights: 3, -2, 2

output: 1

input: 1, 0, 0

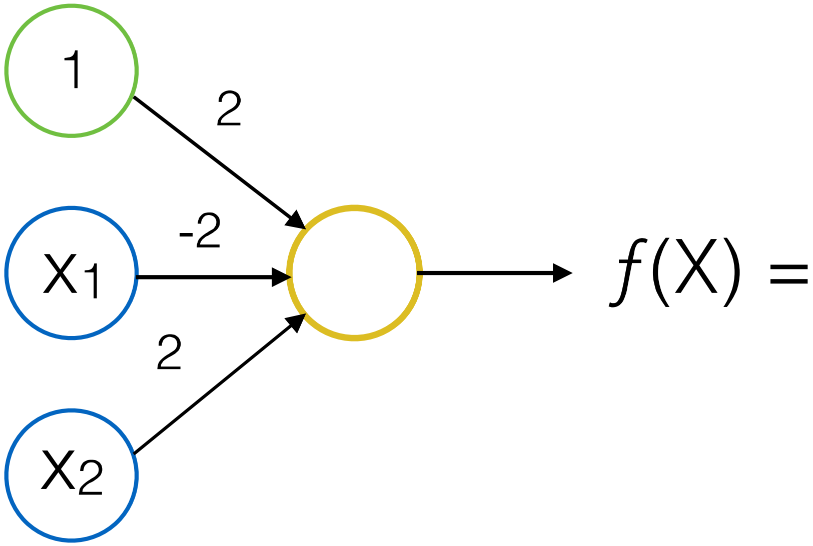
target: 0

updated weight$_0$ = 2

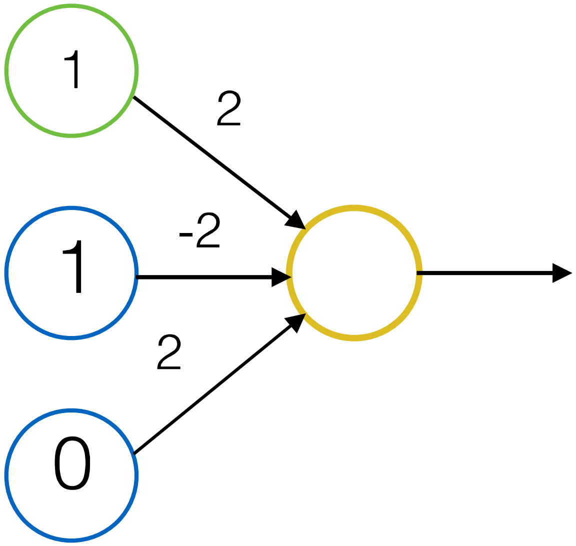
updated weight$_1$ = -2

updated weight$_2$ = 2

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1

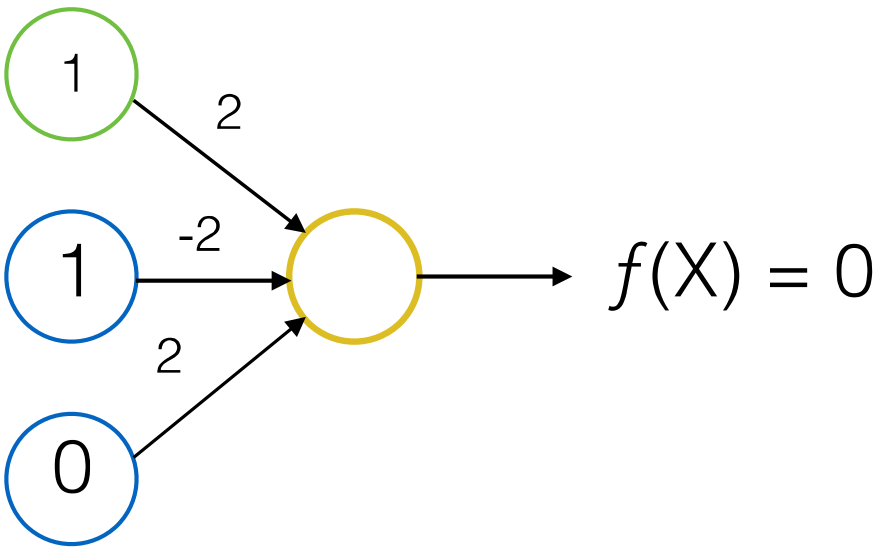


Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1

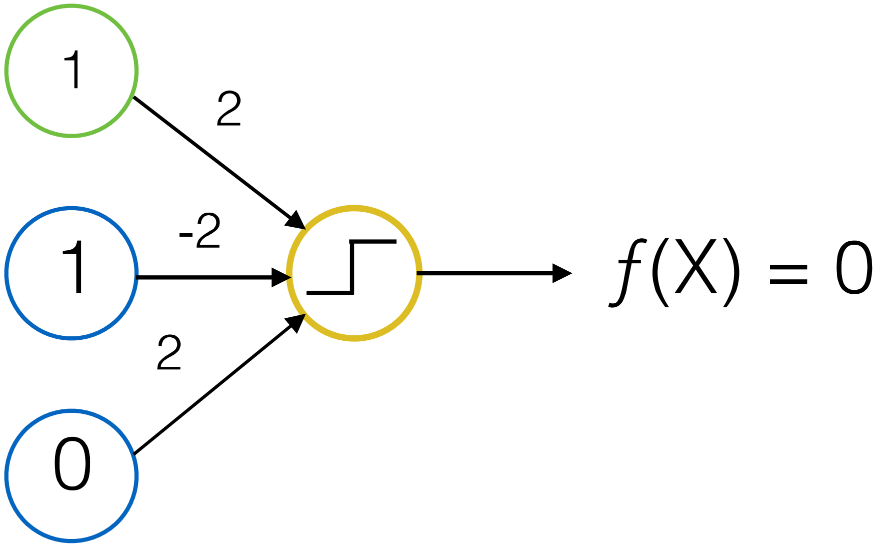


$$f(X) = 2 - 2 + 0$$

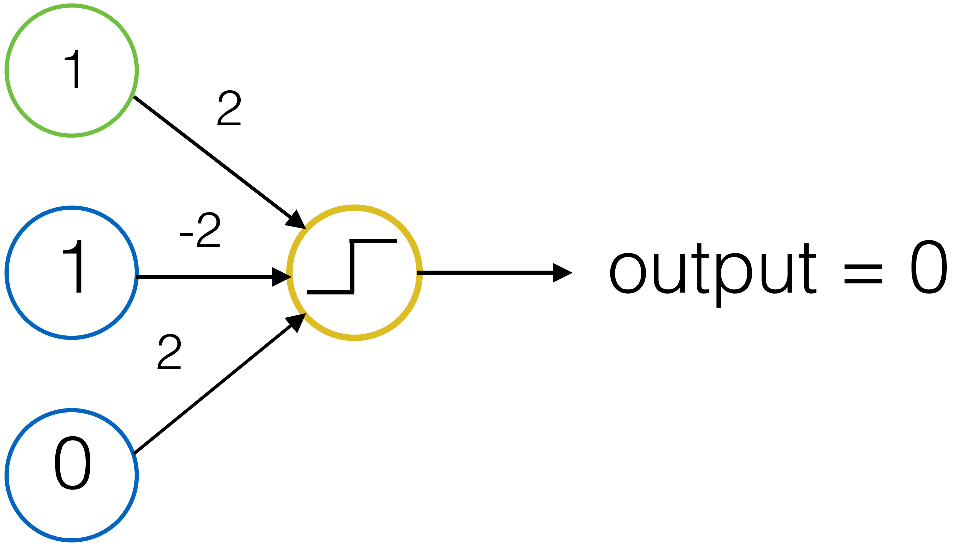
Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	1



weights: 2, -2, 2

input: 1, 1, 0

output: 0

target: 1

updated $\text{weight}_0 = \text{weight}_0 - (\text{output} - \text{target}) * \text{input}_0$

updated $\text{weight}_1 = \text{weight}_1 - (\text{output} - \text{target}) * \text{input}_1$

updated $\text{weight}_2 = \text{weight}_2 - (\text{output} - \text{target}) * \text{input}_2$

weights: 2, -2, 2 **output**: 0

input: 1, 1, 0 **target**: 1

updated weight_0 = 2 - (output - target) * input_0

updated weight_1 = -2 - (output - target) * input_1

updated weight_2 = 2 - (output - target) * input_2

weights: 2, -2, 2

input: 1, 1, 0

output: 0

target: 1

updated $\text{weight}_0 = 2 - (0 - \text{target}) * \text{input}_0$

updated $\text{weight}_1 = -2 - (0 - \text{target}) * \text{input}_1$

updated $\text{weight}_2 = 2 - (0 - \text{target}) * \text{input}_2$

weights: 2, -2, 2

input: 1, 1, 0

output: 0

target: 1

updated weight_0 = 2 - (0 - 1) * input_0

updated weight_1 = -2 - (0 - 1) * input_1

updated weight_2 = 2 - (0 - 1) * input_2

weights: 2, -2, 2 **output**: 0

input: 1, 1, 0 **target**: 1

updated $\text{weight}_0 = 2 - (0 - 1) * 1$

updated $\text{weight}_1 = -2 - (0 - 1) * 1$

updated $\text{weight}_2 = 2 - (0 - 1) * 0$

weights: 2, -2, 2 **output**: 0

input: 1, 1, 0 **target**: 1

updated weight$_0$ = 2 - (-1)

updated weight$_1$ = -2 - (-1)

updated weight$_2$ = 2 - 0

weights: 2, -2, 2

output: 0

input: 1, 1, 0

target: 1

updated weight$_0$ = 2 + 1

updated weight$_1$ = -2 + 1

updated weight$_2$ = 2 - 0

weights: 2, -2, 2 **output**: 0

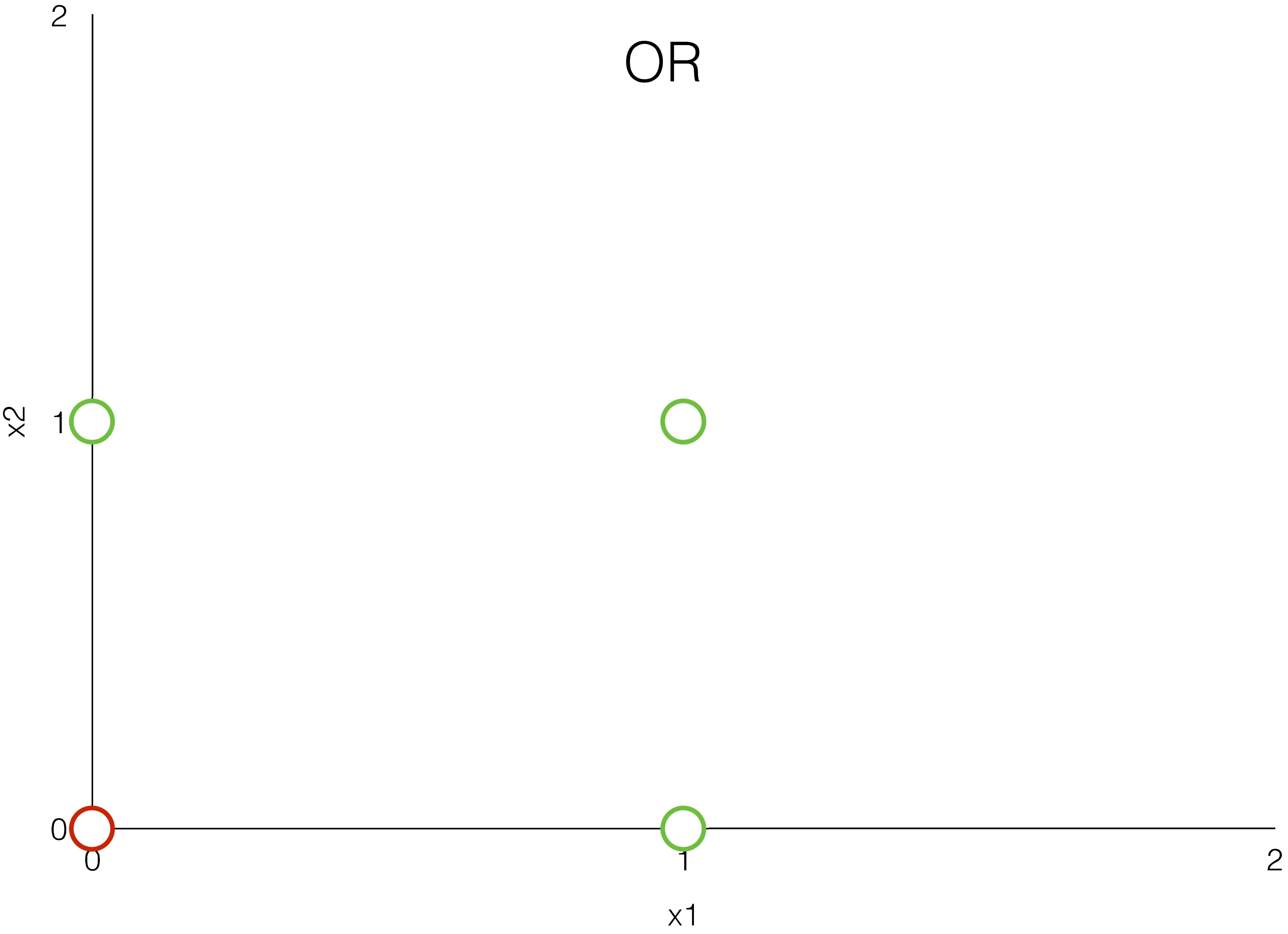
input: 1, 1, 0 **target**: 1

updated weight$_0$ = 3

updated weight$_1$ = -1

updated weight$_2$ = 2

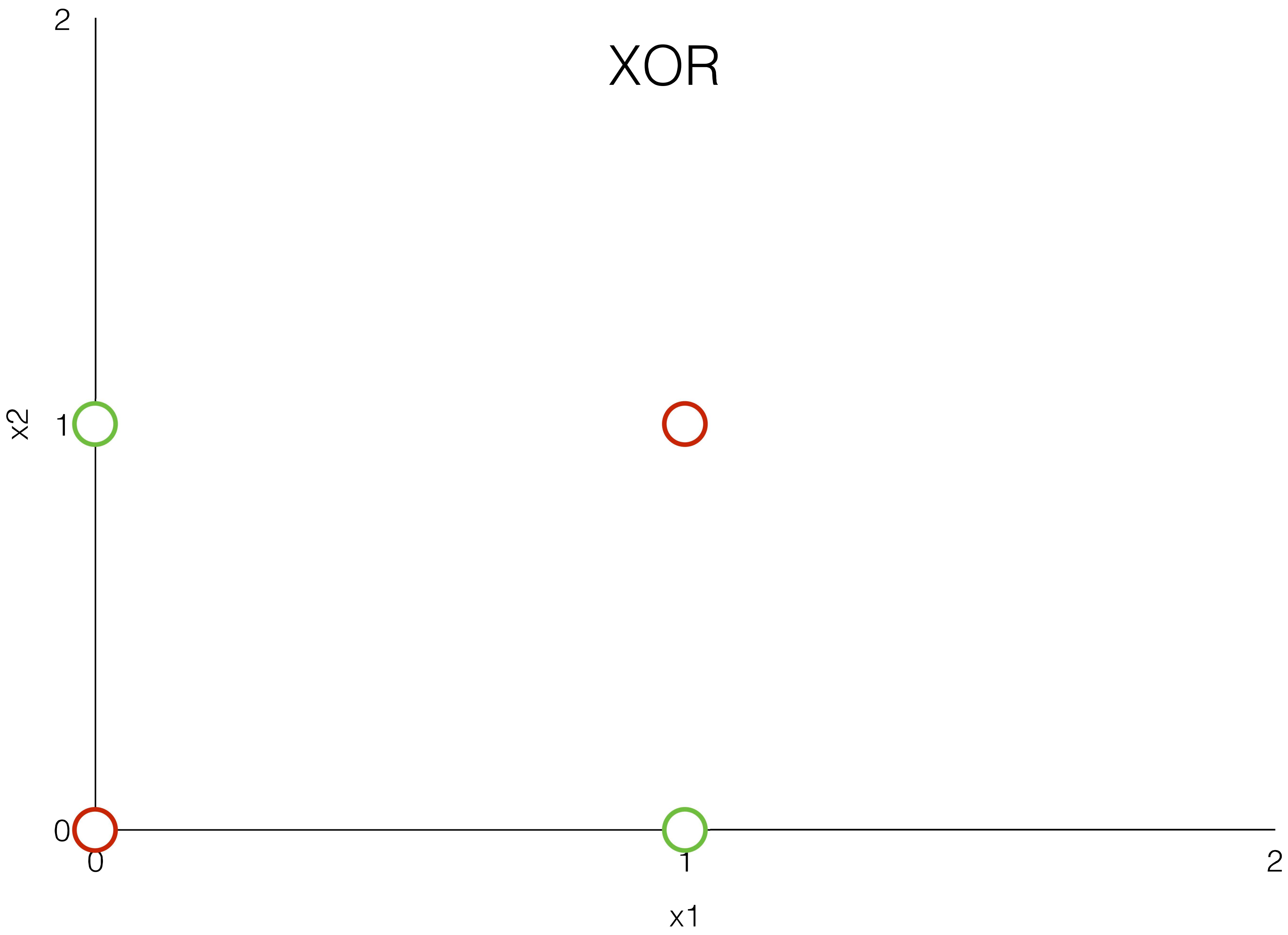
OR



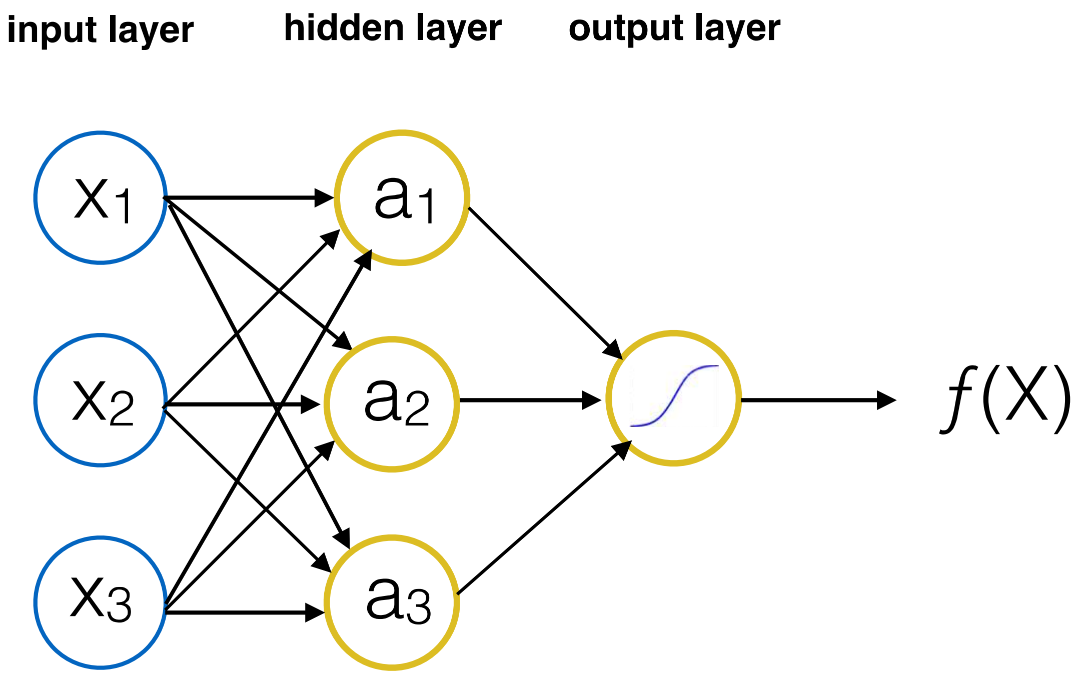
XOR

Feature 1	Feature 2	Target
0	0	0
1	0	1
0	1	1
1	1	0

XOR



Multi-Layer Perceptron (MLP)

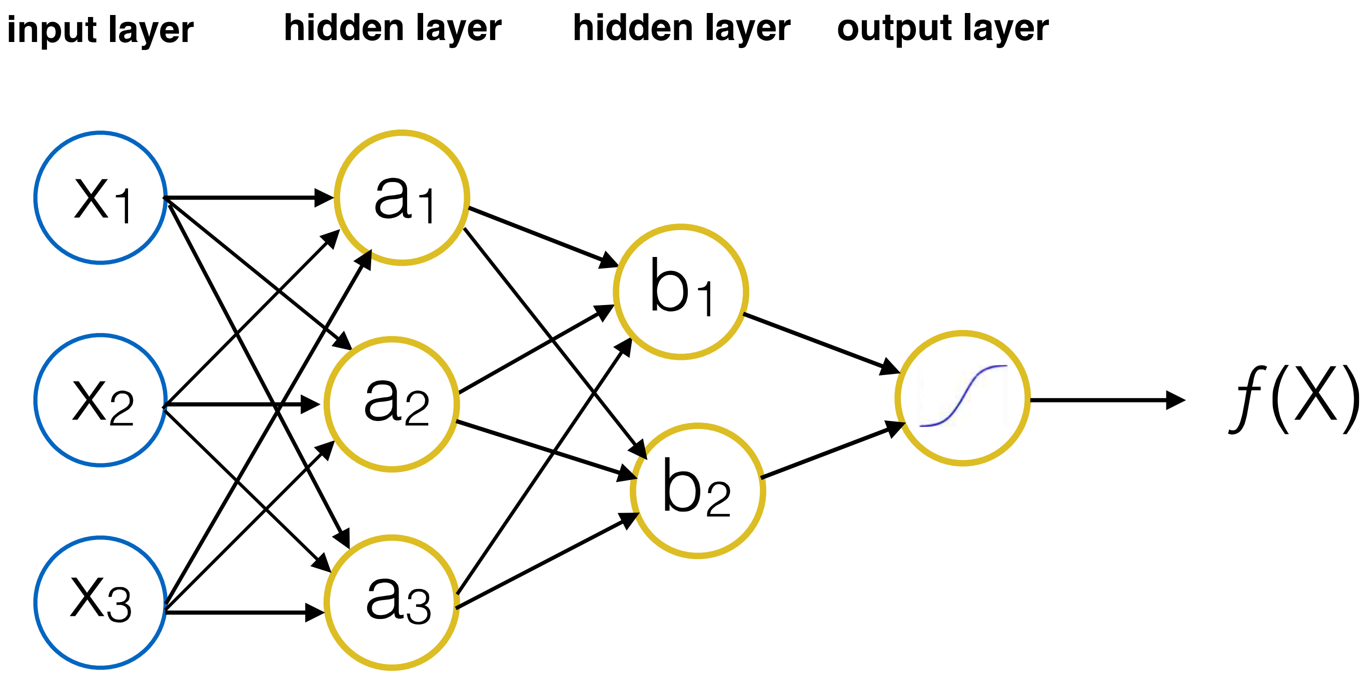


input layer

hidden layer

output layer

Multi-Layer Perceptron (MLP)

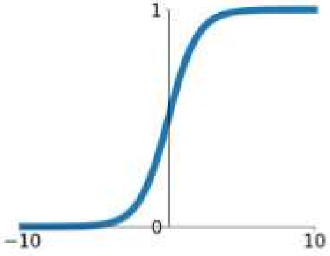


input layer hidden layer hidden layer output layer

Activation Functions

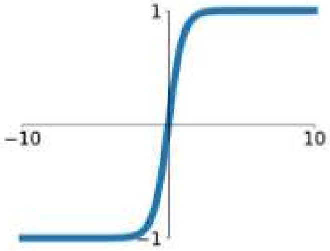
Sigmoid

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



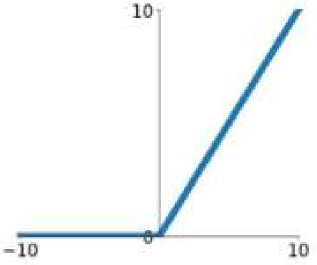
tanh

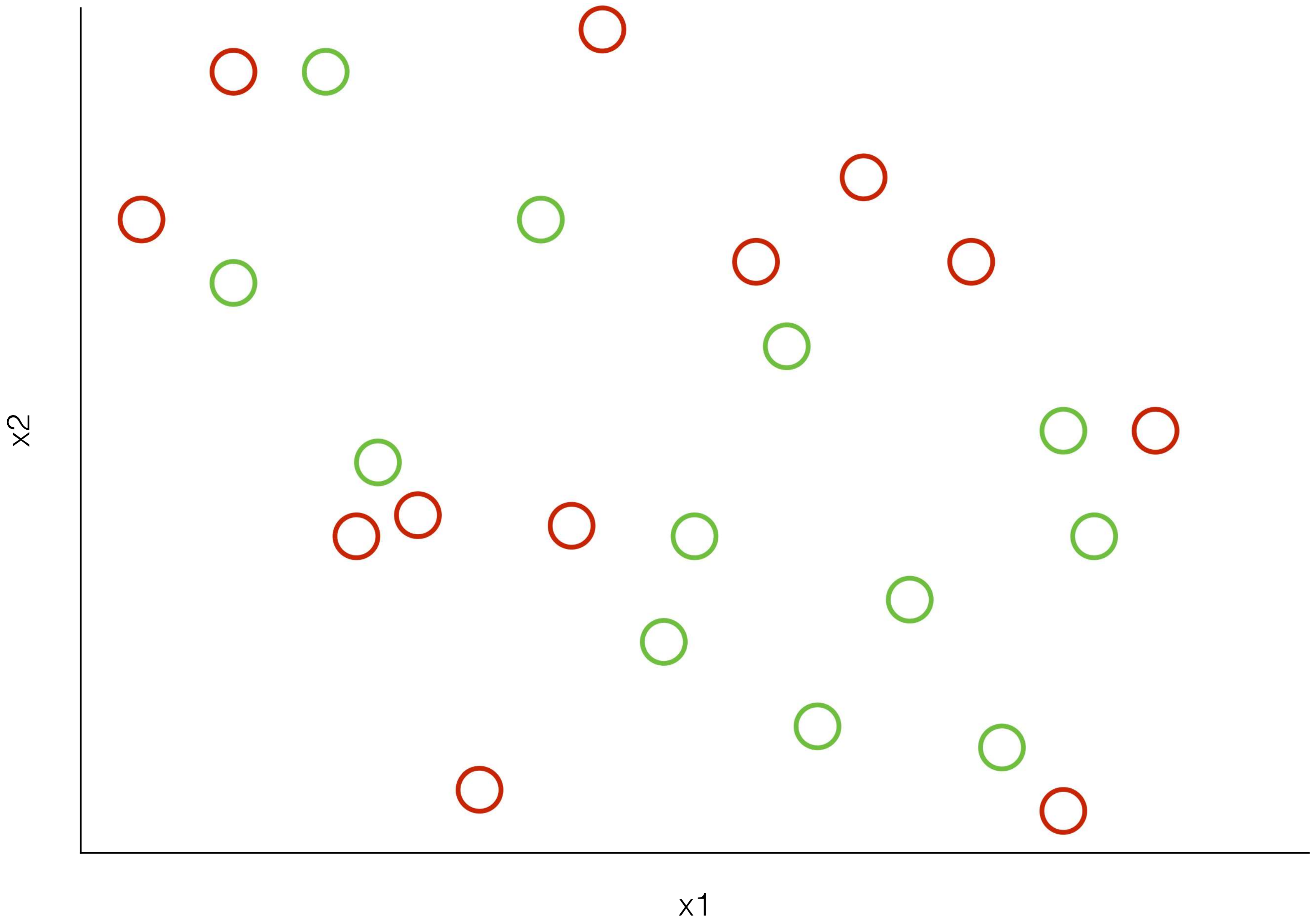
$\tanh(x)$

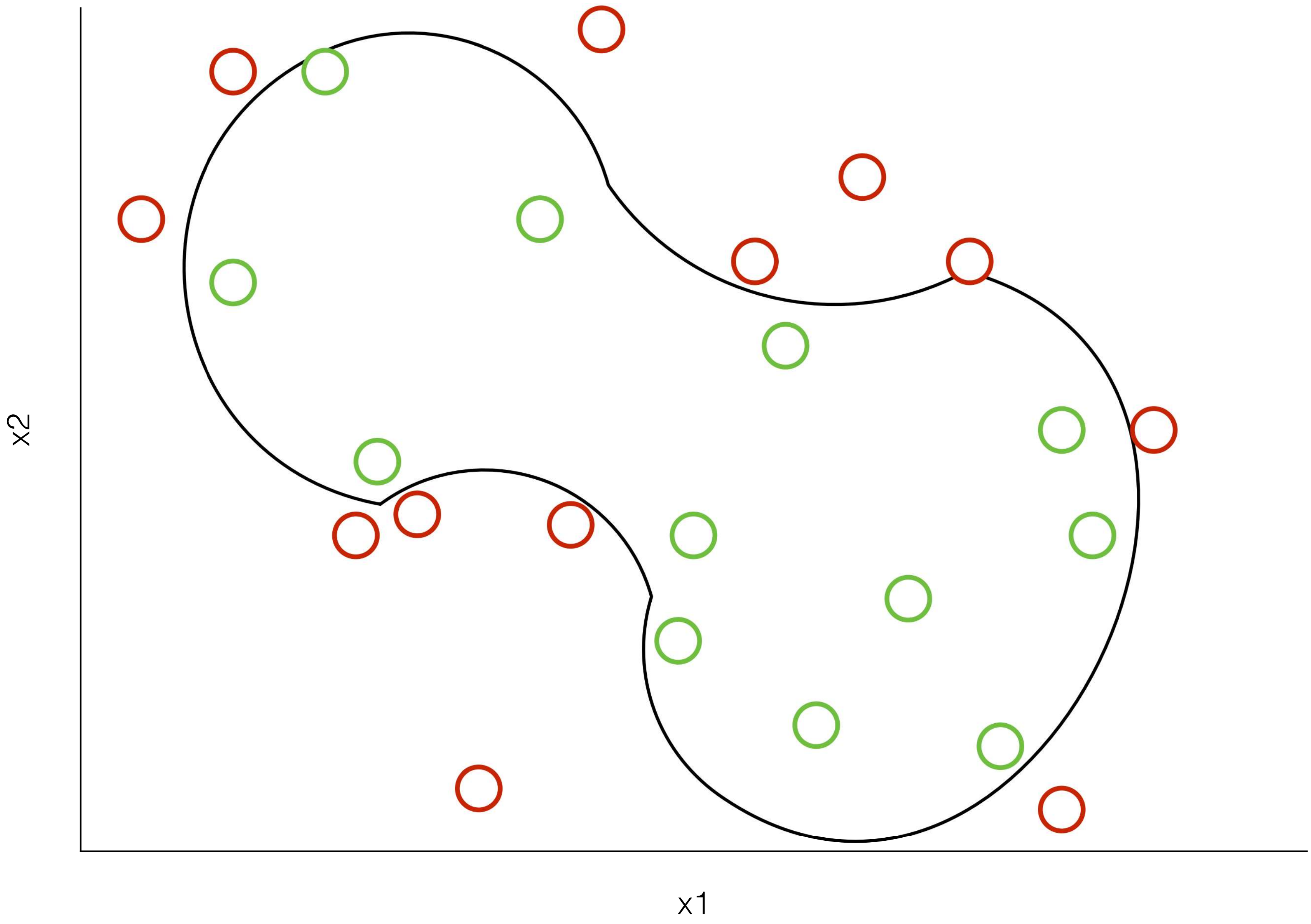


ReLU

$\max(0, x)$



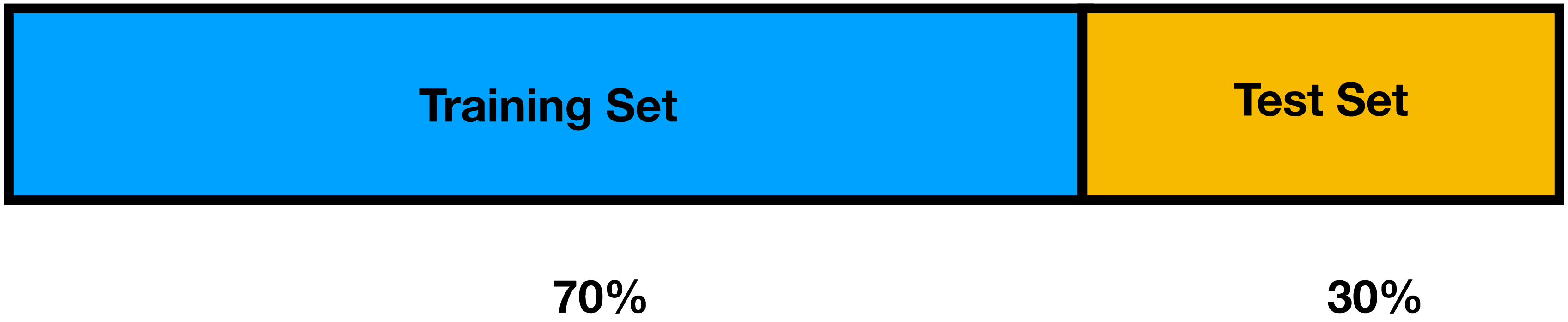

x2
x1


x2
x1

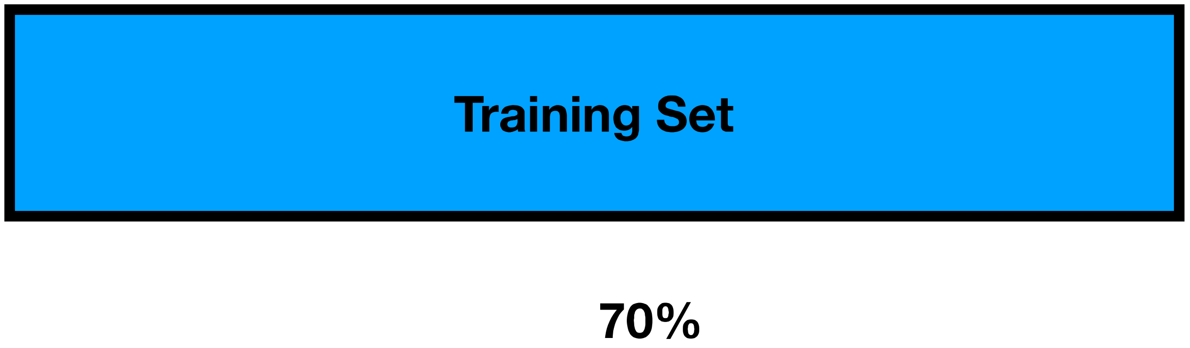
Model Selection

Test/Train Split

DATA SET



DATA SET



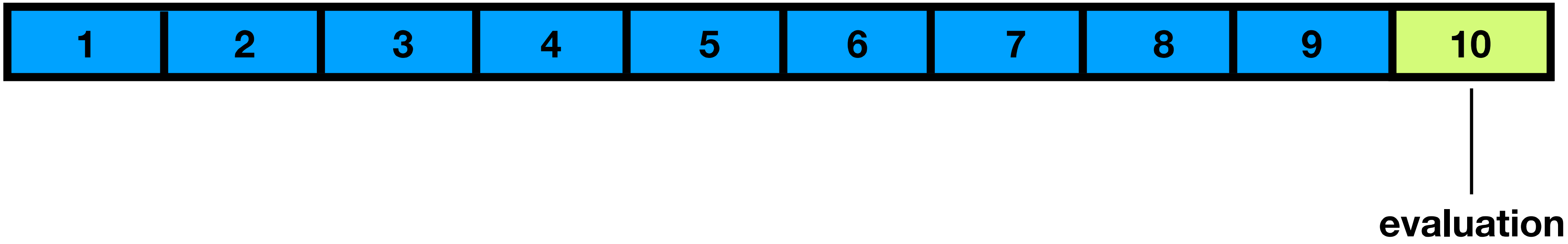
Training Set

70%

K-fold cross validation

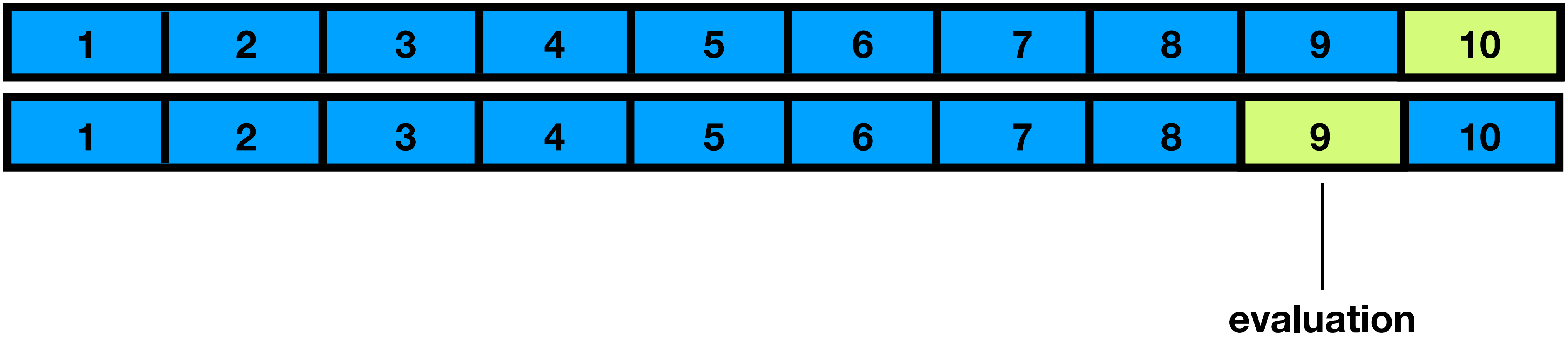
10-Fold Cross Validation

Training Set



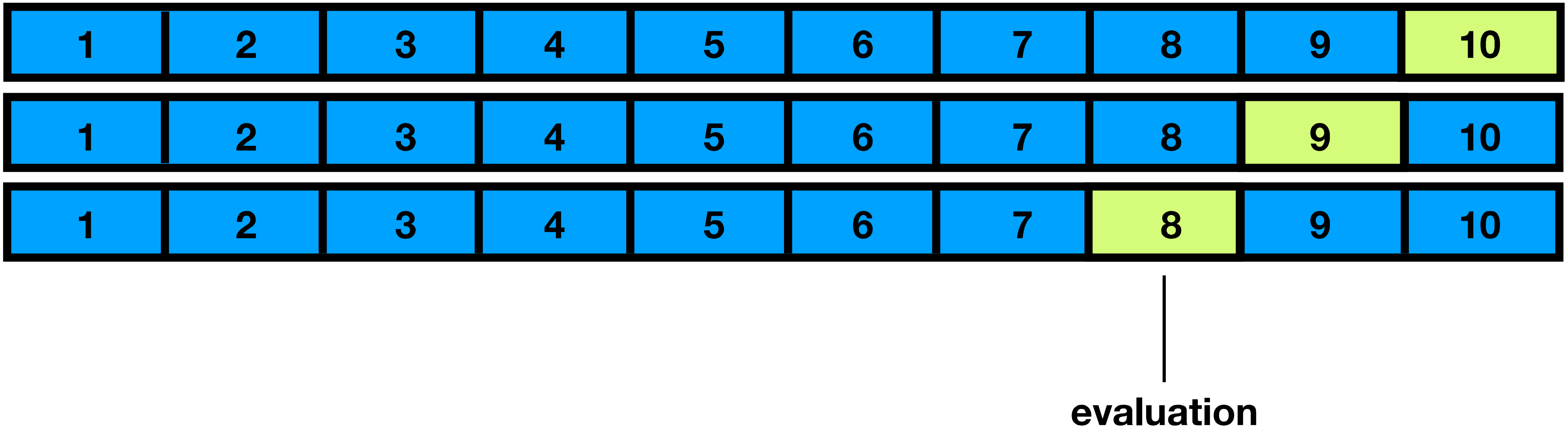
10-Fold Cross Validation

Training Set



10-Fold Cross Validation

Training Set

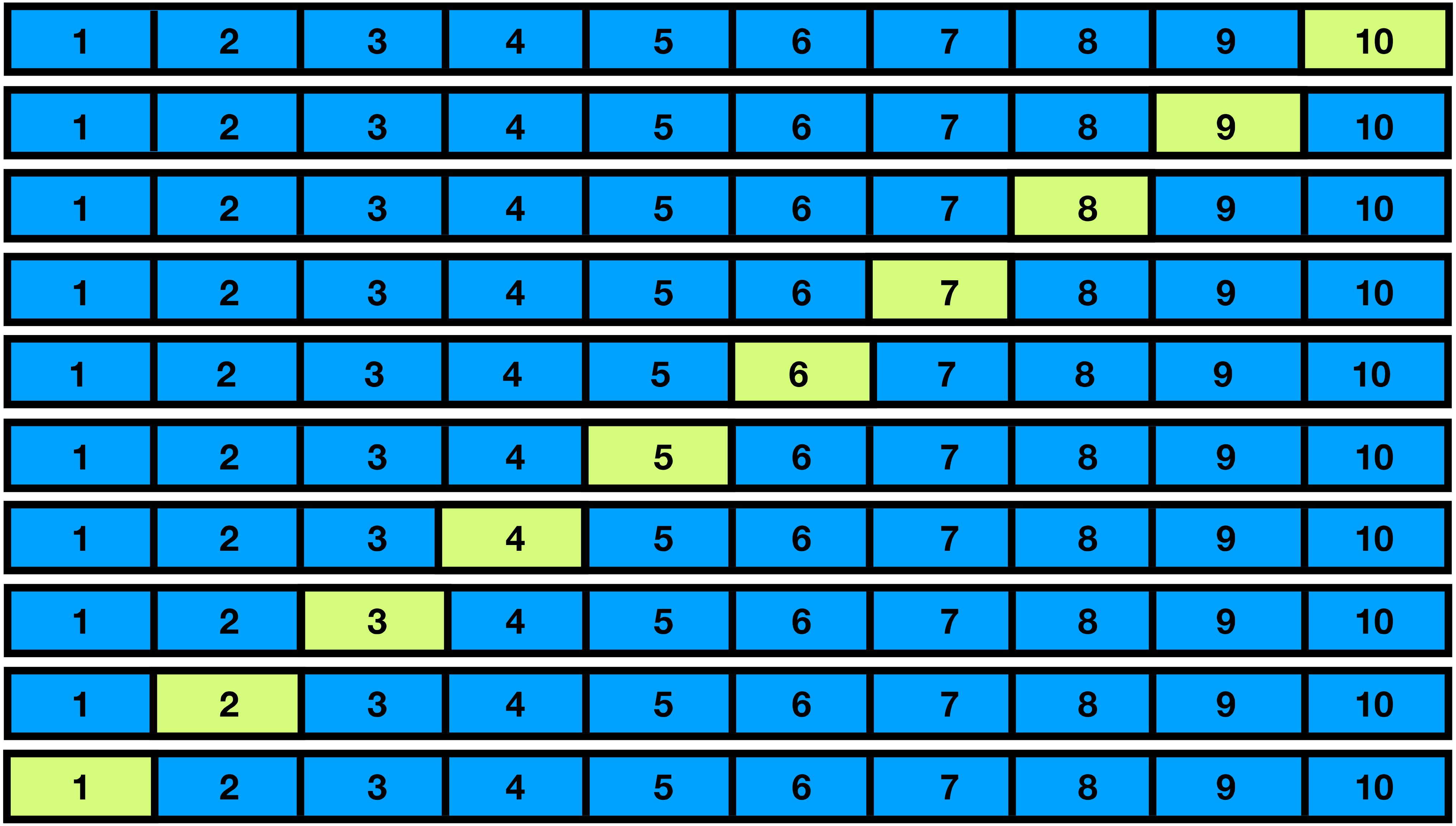


1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

evaluation

10-Fold Cross Validation

Training Set

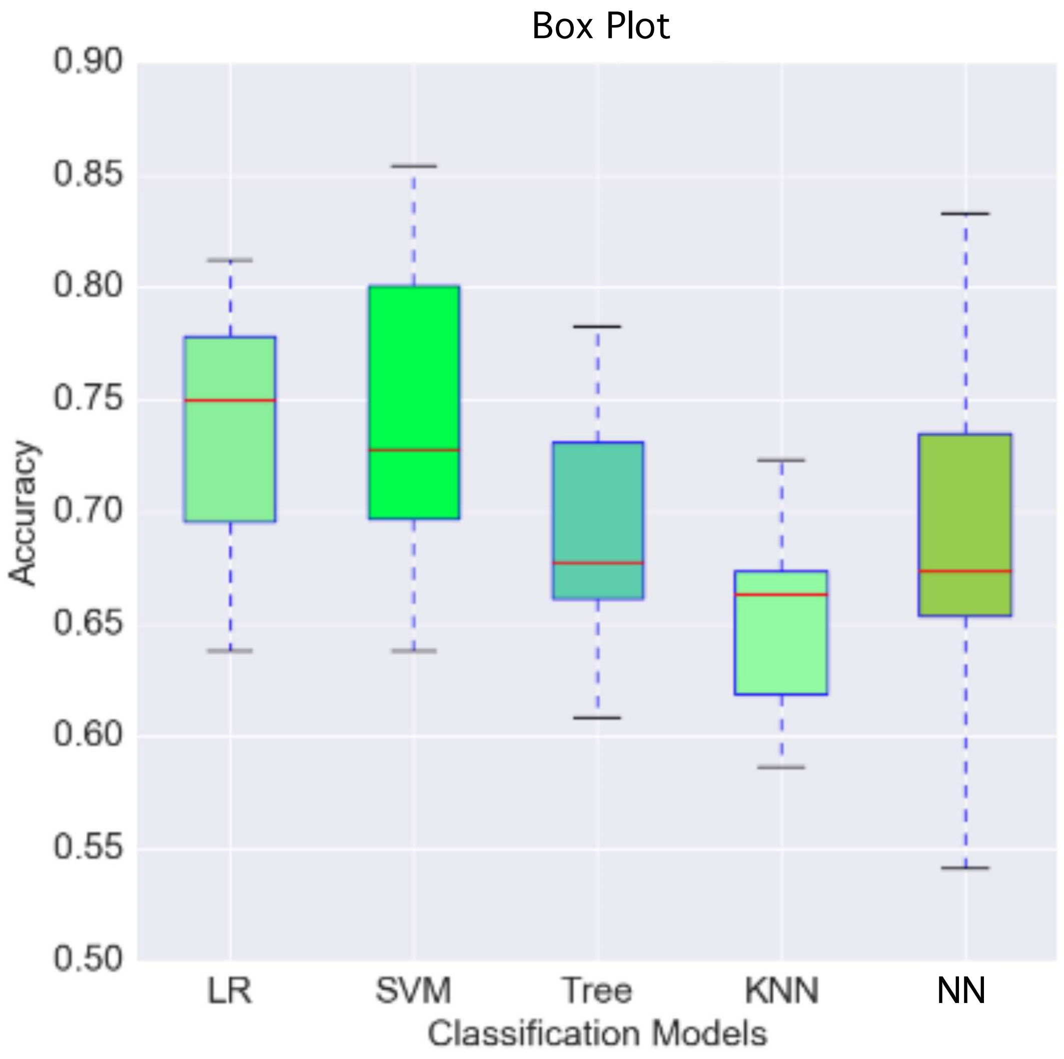


1	2	3	4	5	6	7	8	9	**10**
1	2	3	4	5	6	7	8	**9**	10
1	2	3	4	5	6	7	**8**	9	10
1	2	3	4	5	6	**7**	8	9	10
1	2	3	4	5	**6**	7	8	9	10
1	2	3	4	**5**	6	7	8	9	10
1	2	3	**4**	5	6	7	8	9	10
1	2	**3**	4	5	6	7	8	9	10
1	**2**	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

Logistic Regression

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	mean
0.69	0.64	0.73	0.82	0.64	0.70	0.68	0.71	0.70	0.69	**0.70**

Logistic Regression	Support Vector Machine	Decision Tree	K-Nearest Neighbor	Neural Network
0.705	0.722	0.635	0.675	0.607



Hyperparameter Tuning

Logistic Regression

$$\hat{f}(X) = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2)}}$$

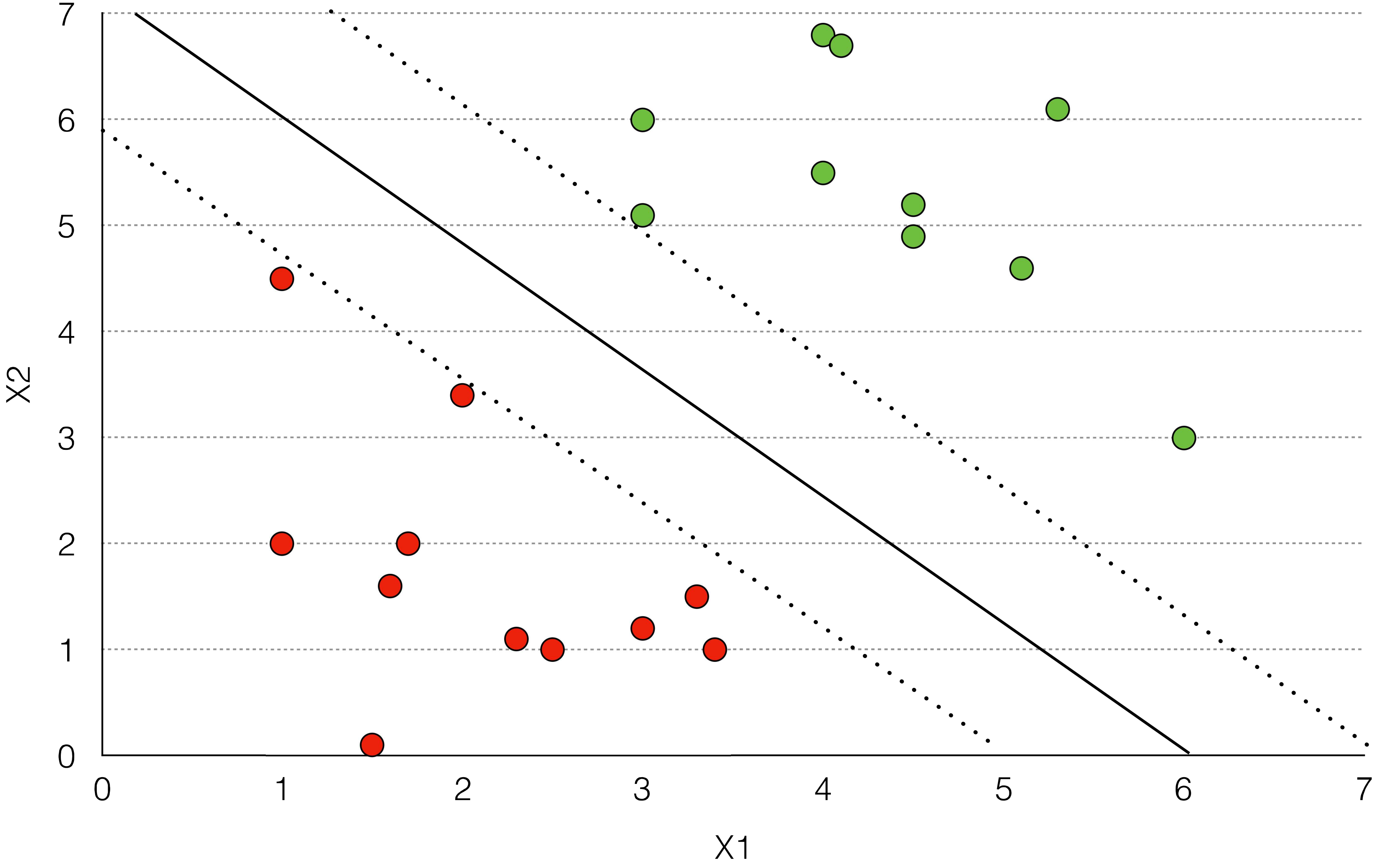
$$\text{LOC} = 227.63 + 9.51x_1 + 2.7x_2 - 7.08x_3$$

x_1 = hour pair programming

x_2 = gender (m = 0; f = 1)

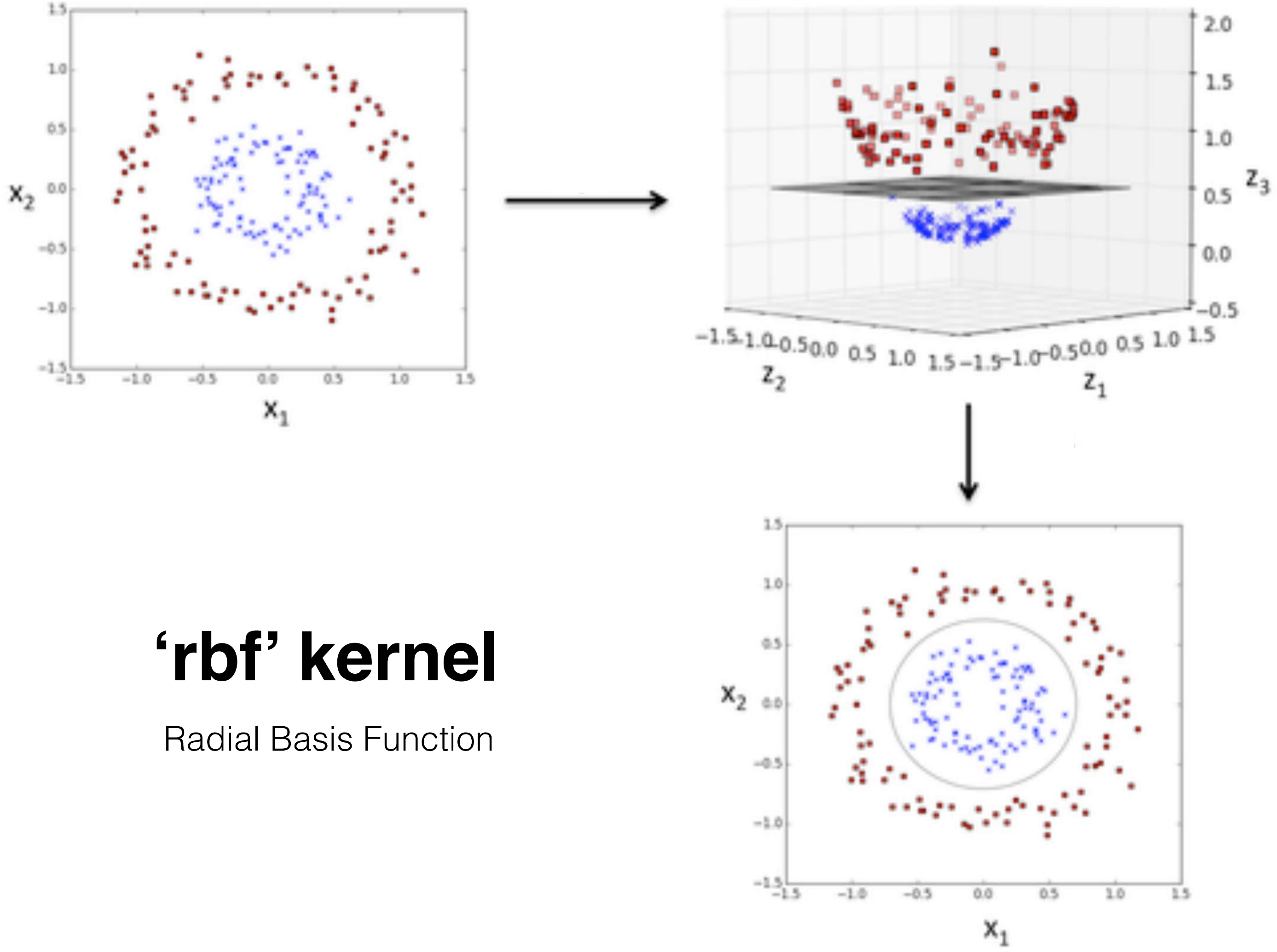
x_3 = number of social accounts

Support Vector Machine

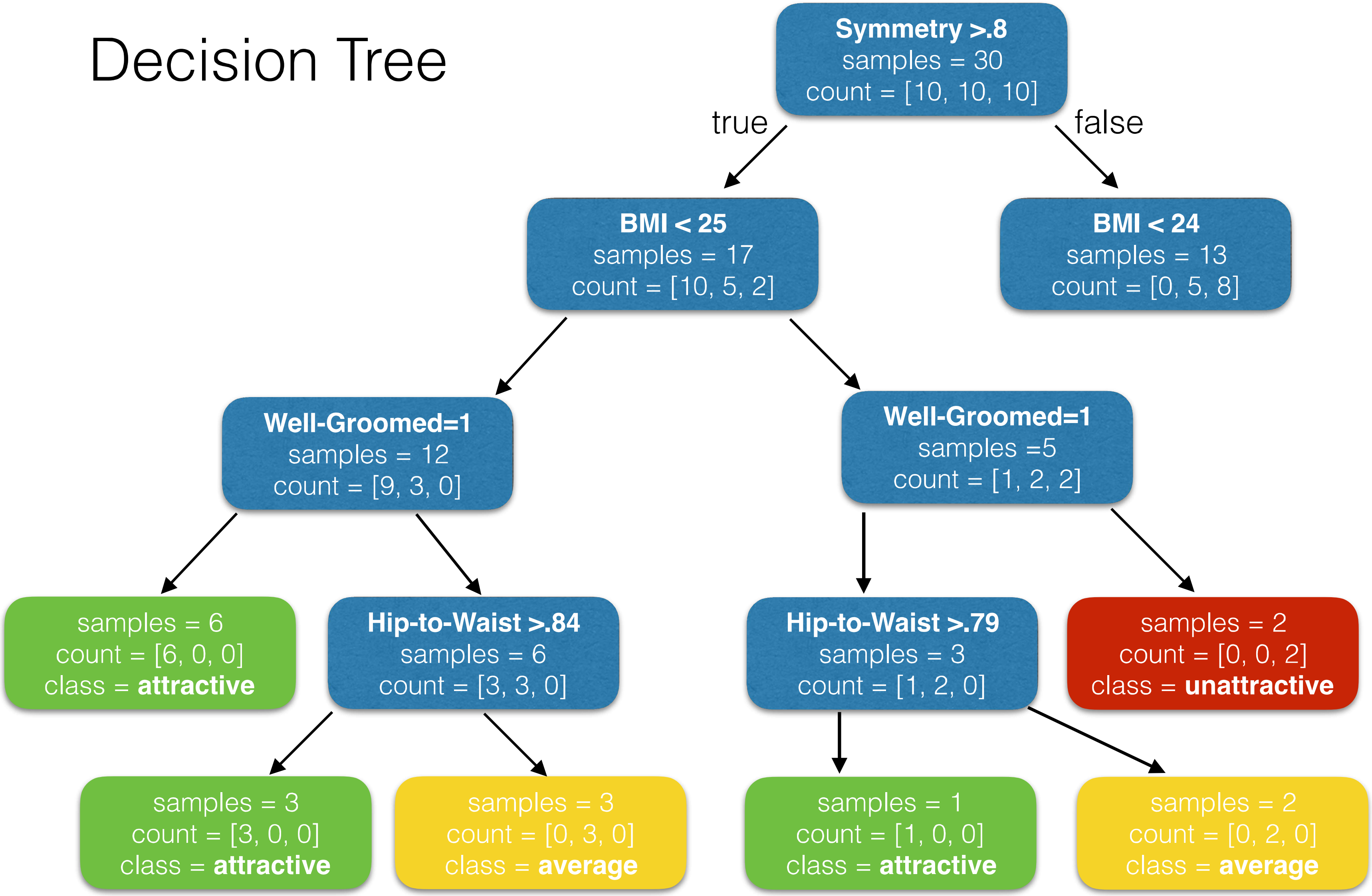


‘rbf’ kernel

Radial Basis Function



Decision Tree



Symmetry >.8
samples = 30
count = [10, 10, 10]

true

false

BMI < 25
samples = 17
count = [10, 5, 2]

BMI < 24
samples = 13
count = [0, 5, 8]

Well-Groomed=1
samples = 12
count = [9, 3, 0]

Well-Groomed=1
samples =5
count = [1, 2, 2]

samples = 6
count = [6, 0, 0]
class = **attractive**

Hip-to-Waist >.84
samples = 6
count = [3, 3, 0]

Hip-to-Waist >.79
samples = 3
count = [1, 2, 0]

samples = 2
count = [0, 0, 2]
class = **unattractive**

samples = 3
count = [3, 0, 0]
class = **attractive**

samples = 3
count = [0, 3, 0]
class = **average**

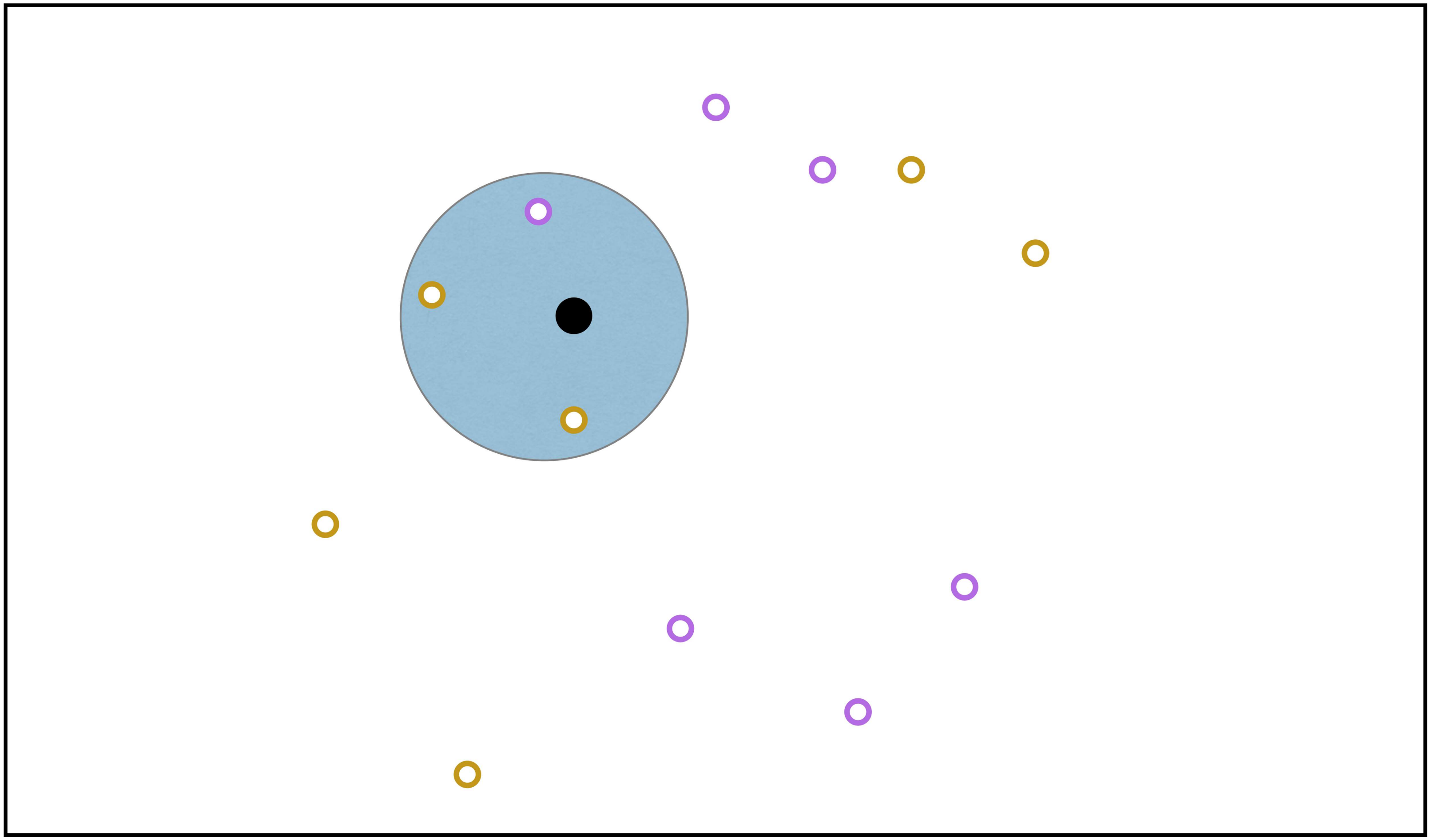
samples = 1
count = [1, 0, 0]
class = **attractive**

samples = 2
count = [0, 2, 0]
class = **average**

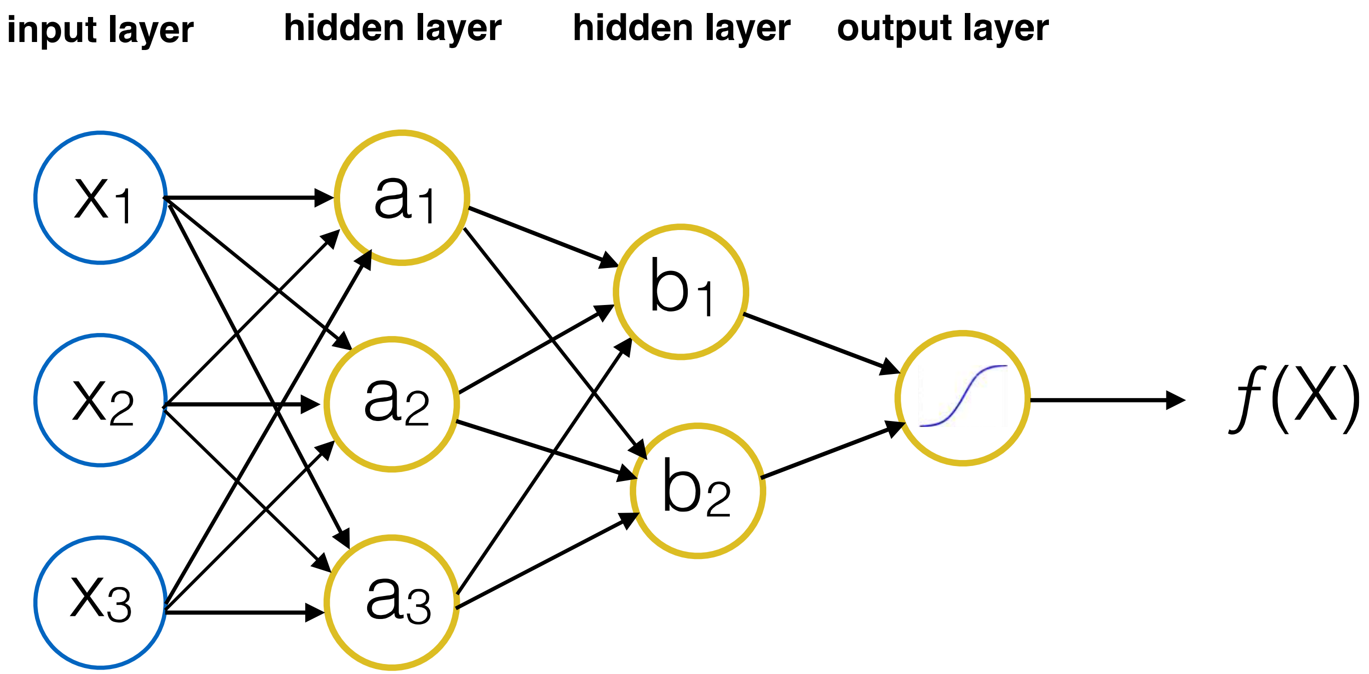
[att, ave, un]

k-Nearest Neighbor

k = 3



Multi-Layer Perceptron (MLP)



input layer hidden layer hidden layer output layer

Grid Search

Grid Search

Support Vector Machine

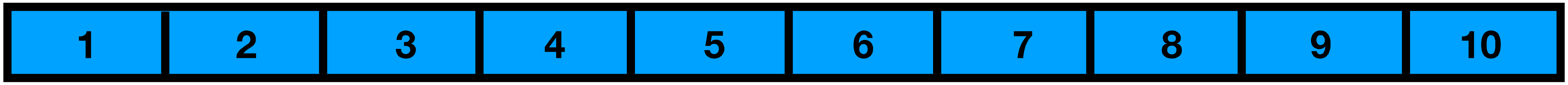
Grid Search

Support Vector Machine

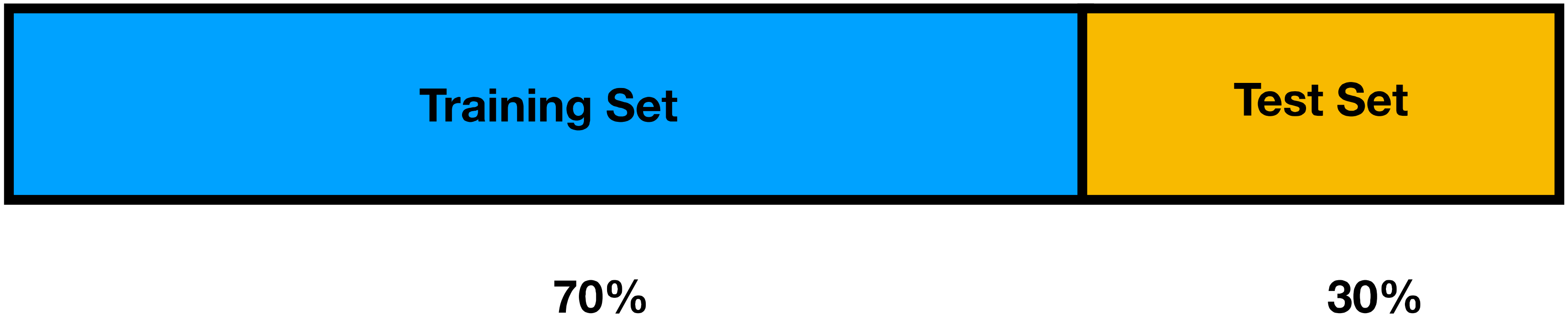
param_grid=[{'C': [.1, 1, 10]}], ‘kernel’: [‘linear’, ‘rbf’], cv=3)}]

C	kernel
0.1	linear’
1	linear’
10	linear’
0.1	‘rbf’
1	'rbf’
10	‘rbf’

Training Set



DATA SET



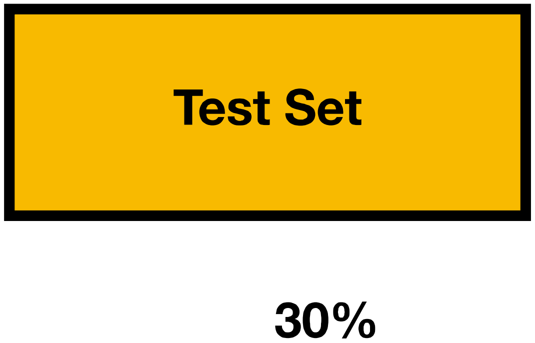
Training Set

Test Set

70%

30%

DATA SET



Test Set

30%

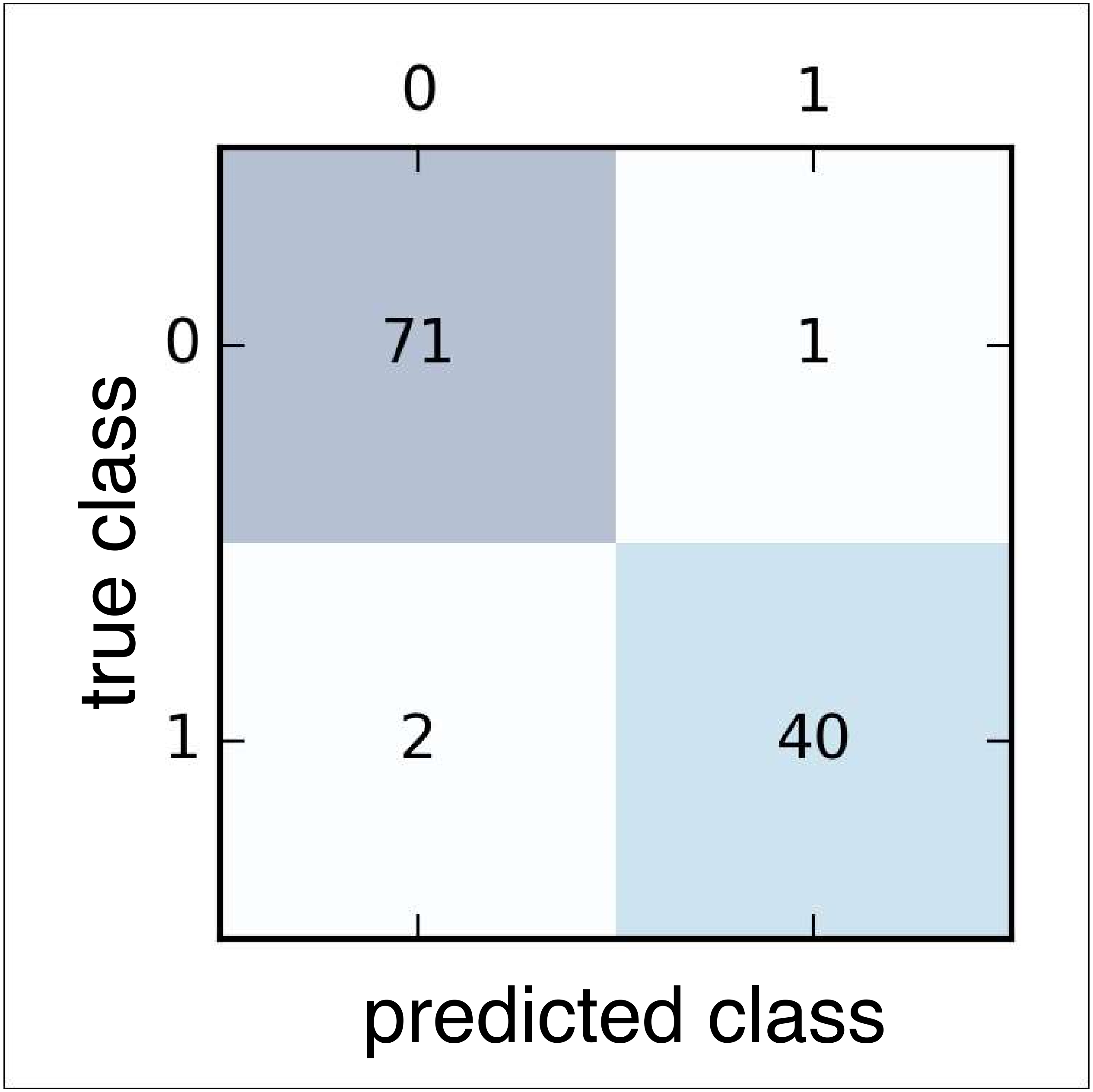
Model Evaluation Metrics

Confusion Matrix

Confusion Matrix

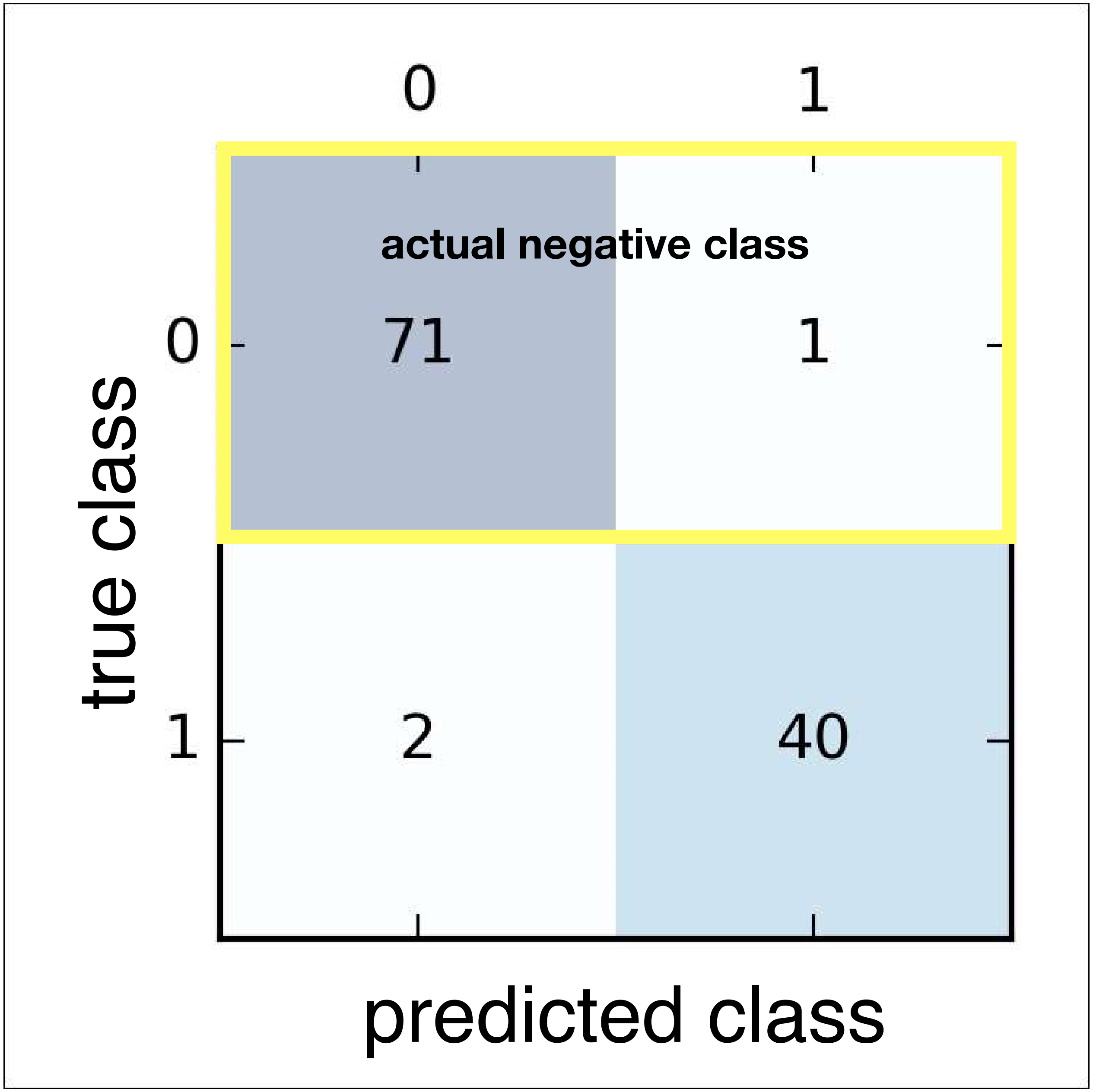
	Predicted Negative	Predicted Positive
Negative Class	TN True Negative	FP False Positive
Positive Class	FN False Negative	TP True Positive

Confusion Matrix



	0	1
0	71	1
1	2	40

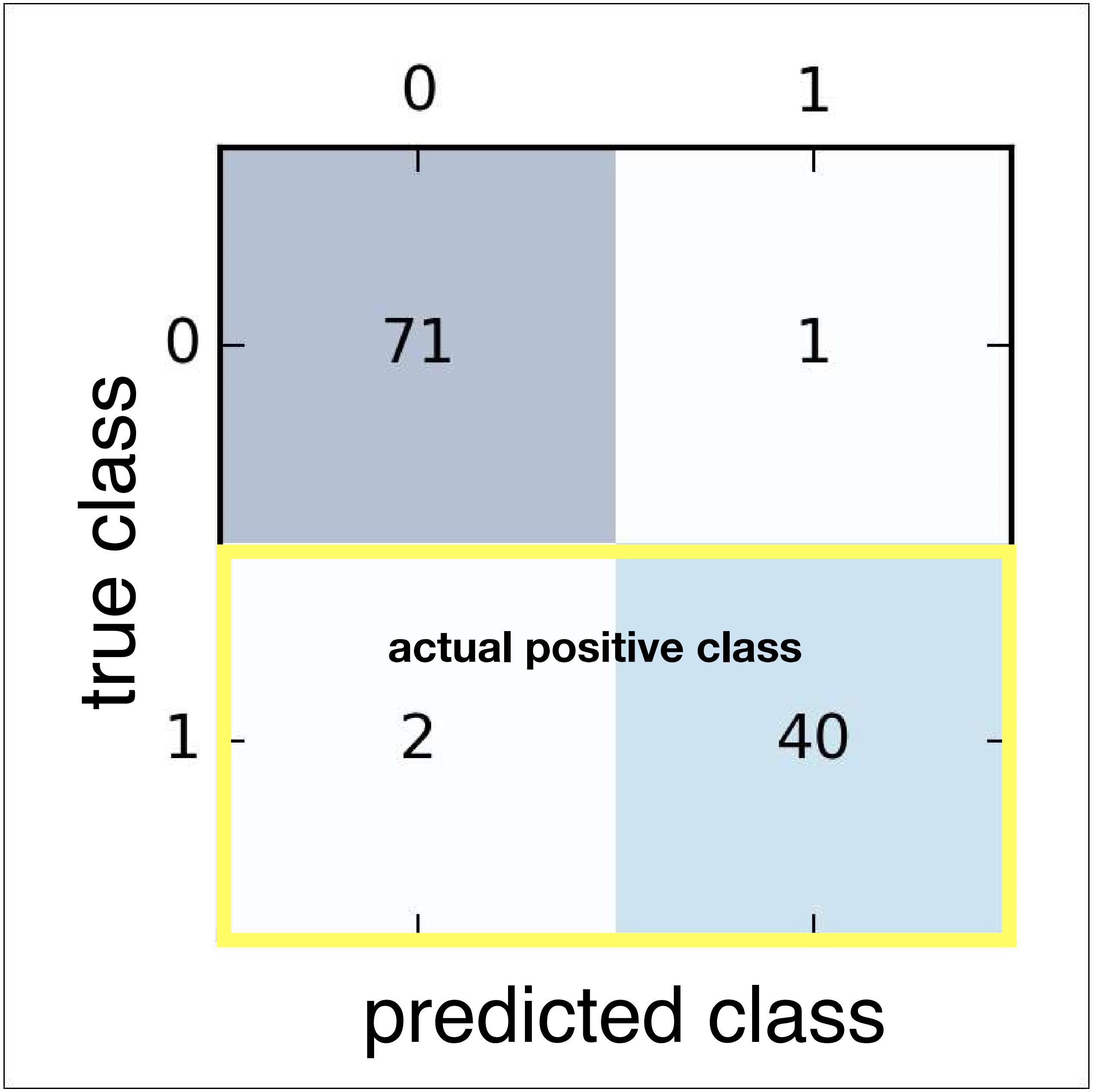
Confusion Matrix



actual negative class

true class \ predicted class	0	1
0	71	1
1	2	40

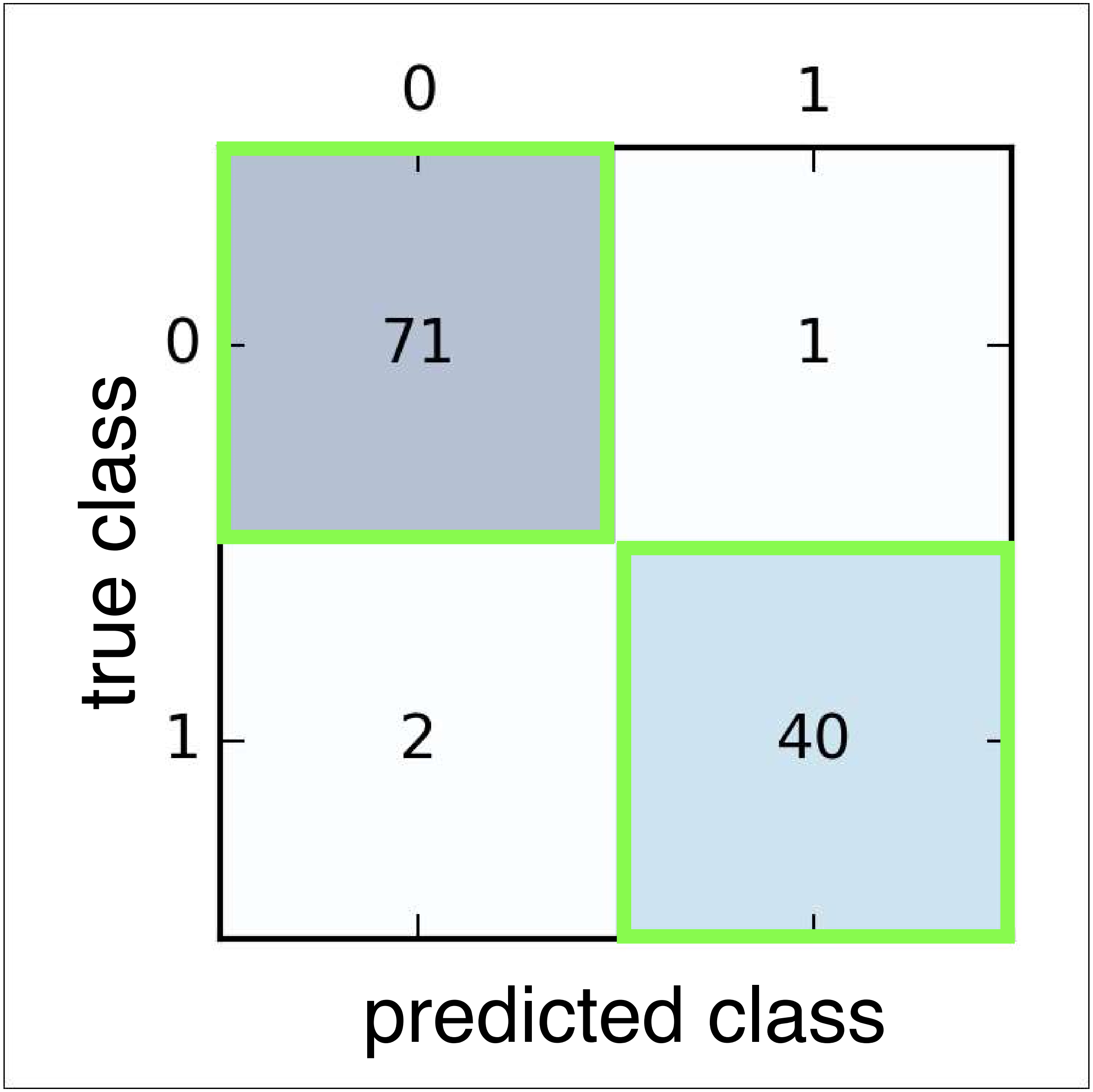
Confusion Matrix



true class \ predicted class	0	1
0	71	1
1	2	40

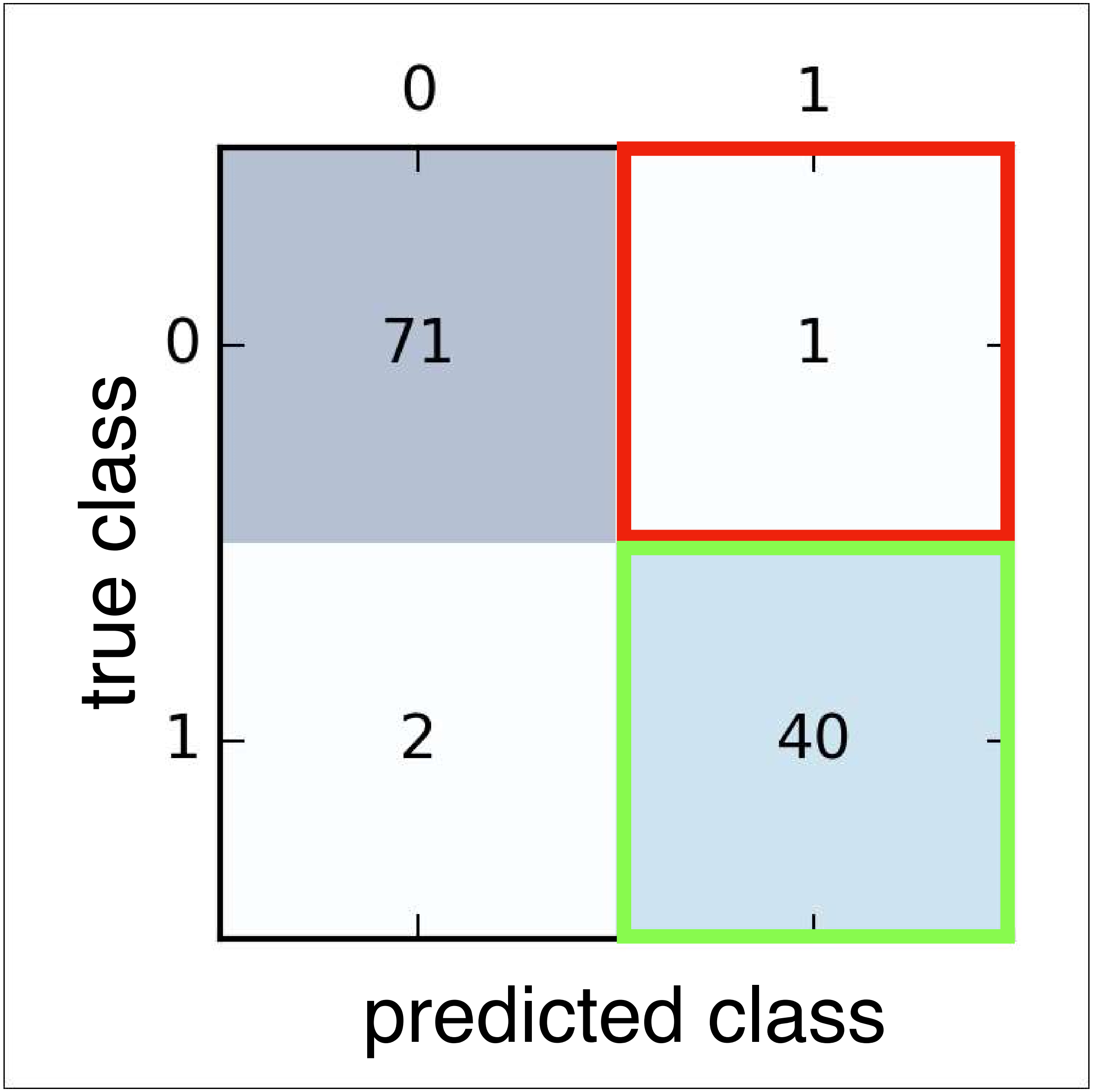
actual positive class

Accuracy



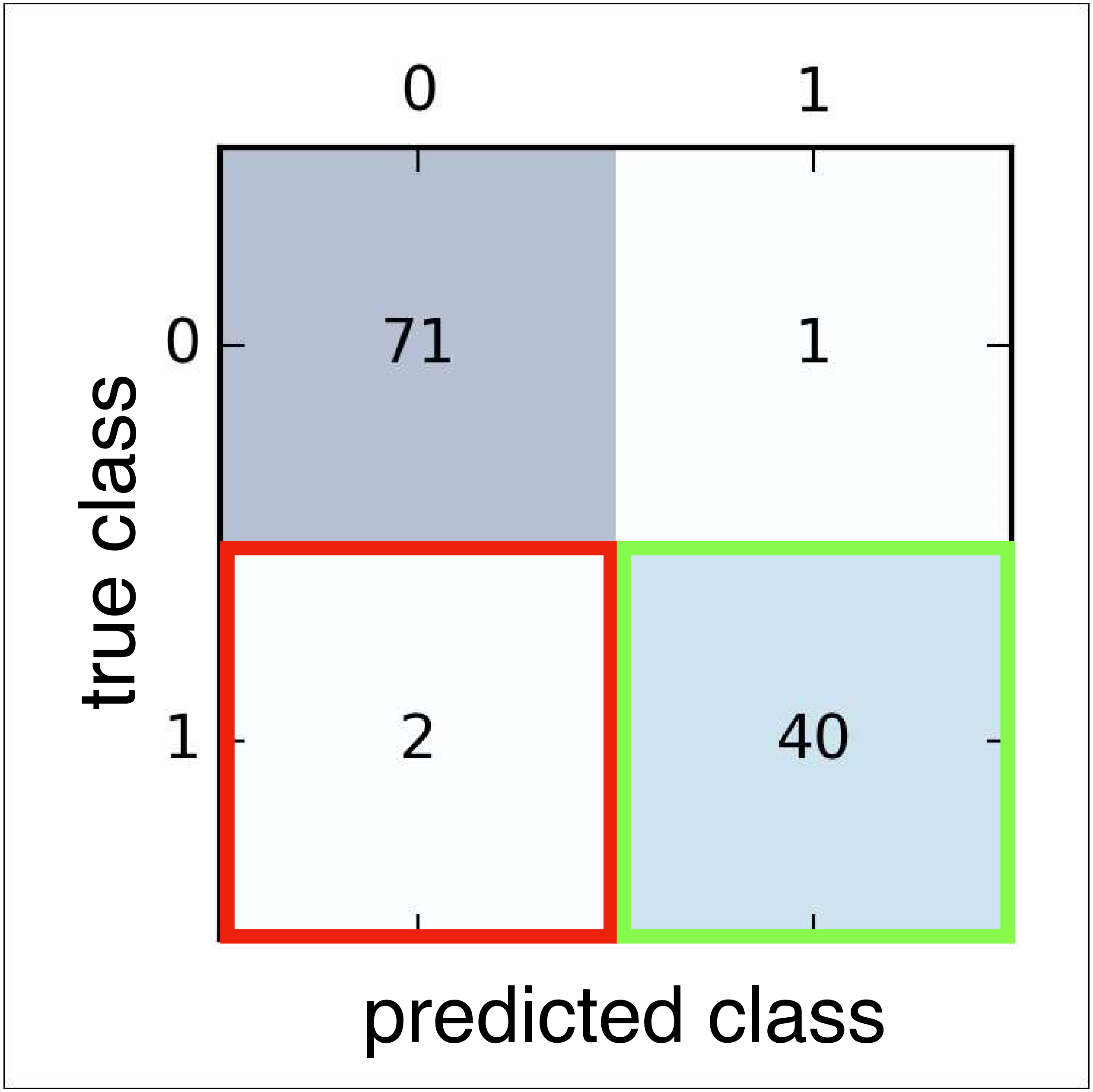
true class \ predicted class	0	1
0	71	1
1	2	40

Precision



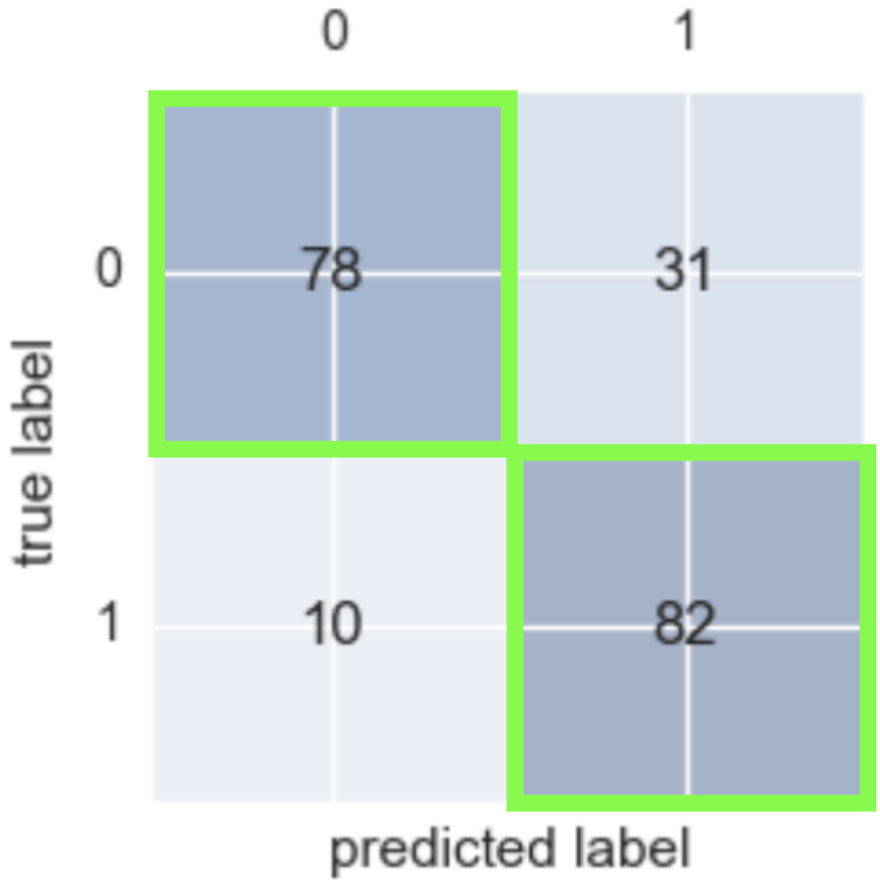
		predicted class	
		0	1
true class	0	71	1
	1	2	40

Recall



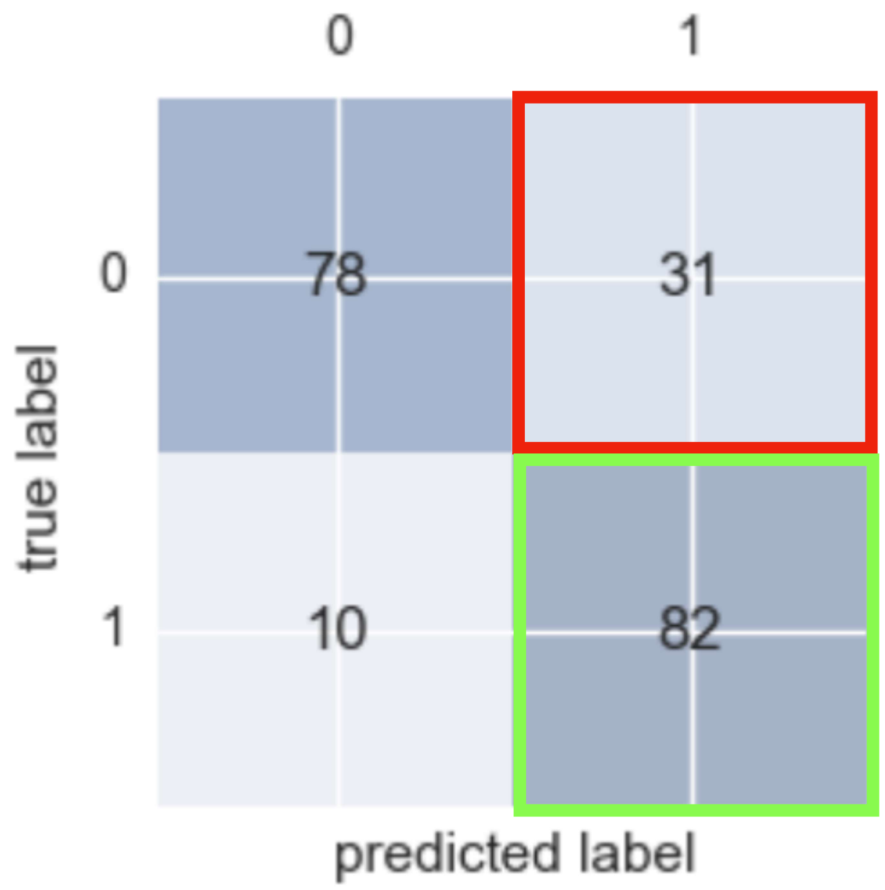
	0	1
0	71	1
1	2	40

Confusion Matrix



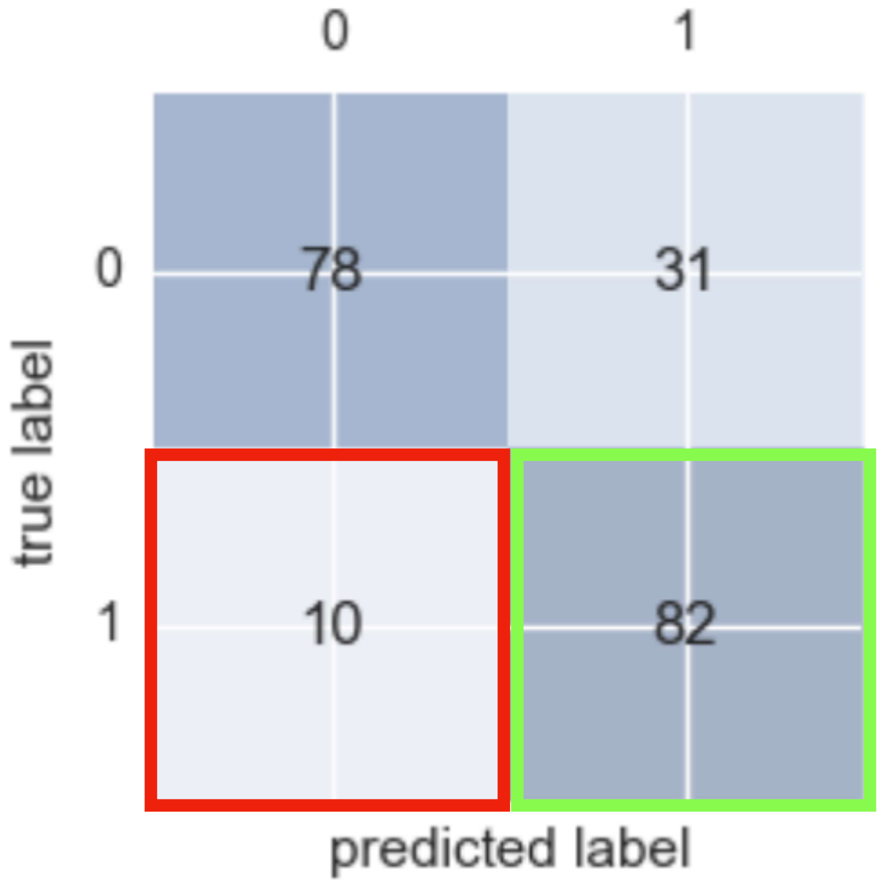
Accuracy
0.796

Confusion Matrix



Precision
0.725

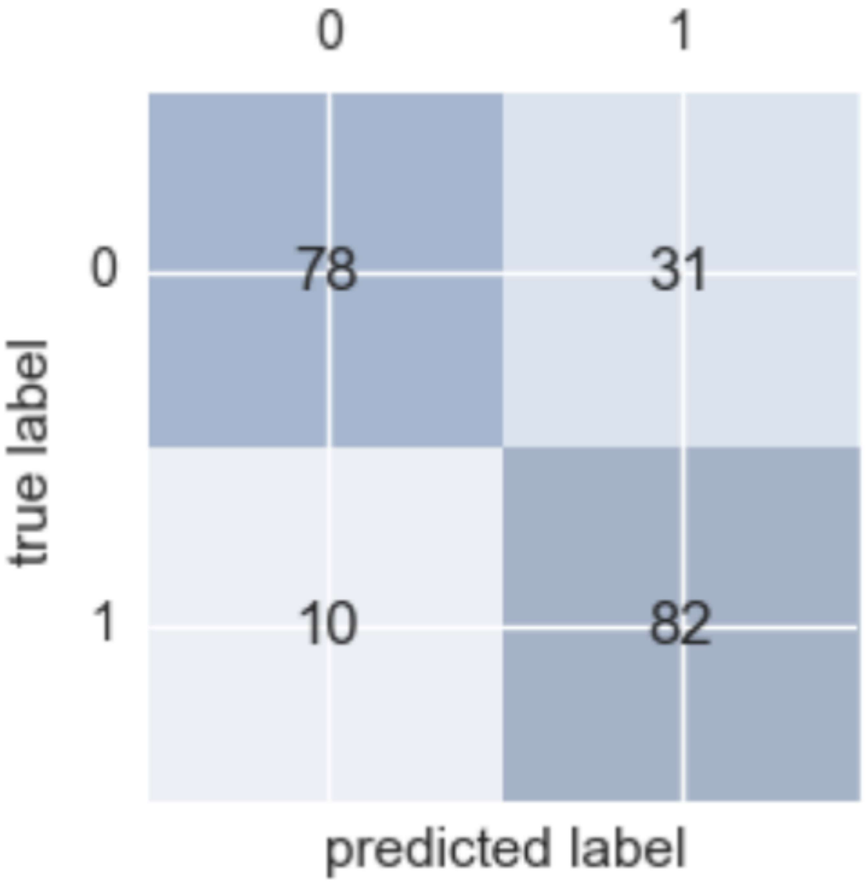
Confusion Matrix



Recall
0.891

Model Evaluation Metrics

	Precision	Recall	F1-score
Rejected	0.89	0.72	0.79
Liked	0.73	0.89	0.81



Some things to keep in mind:

Get more and/or better data

Feature Engineering

Hyperparameter tuning

Generalization