MLDL-I

Machine Learning and Deep Learning - I

Lecture 3

tmax	tmax_tomorrow
60.0	52.0

Initialize Parameters

Calculate Prediction

$$\hat{y} = wx + b$$

Estimate Error

$$J(w,b) = (\hat{y} - y_{actual})^2$$

Update Parameters

$$w = w - \frac{\partial J}{\partial w} \qquad b = b - \frac{\partial J}{\partial b_1}$$

Weight	New_Label
150	1

Initialize Parameters

Calculate Prediction

$$\hat{y} = \sigma(wx + b)$$

Estimate Error

$$J(w,b) = CrossEntropy(\hat{y}, y_{actual})$$

Update Parameters

$$w = w - \frac{\partial J}{\partial w} \qquad b = b - \frac{\partial J}{\partial b}$$

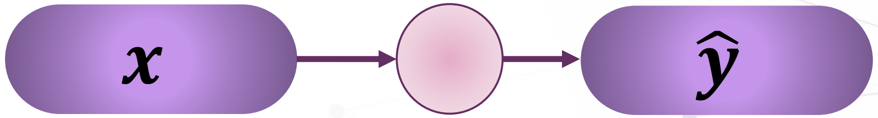
Weight	New_Label
150	1

Initialize Parameters

Calculate Prediction
$$\hat{y} = \sigma(wx + b)$$

Estimate Error
$$J(w, b) = CrossEntropy(\hat{y}, y_{actual})$$

Update Parameters
$$w = w - \frac{\partial J}{\partial w_0} \qquad b = b - \frac{\partial J}{\partial b_1}$$



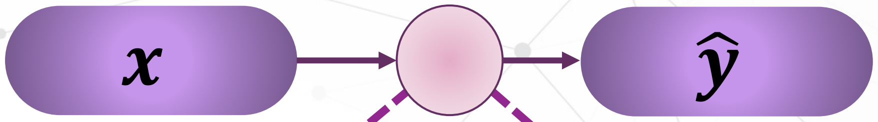
Weight	New_Label
150	1

Initialize Parameters

Calculate Prediction
$\hat{y} = \sigma(wx + b)$

Estimate Error
$J(w, b) = CrossEntropy(\hat{y}, y_{actual})$

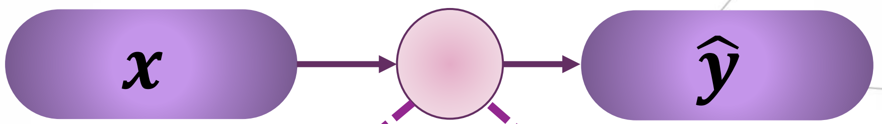
Update Parameters
$w = w - \frac{\partial J}{\partial w_0}$ $\quad b = b - \frac{\partial J}{\partial b_1}$



$\sigma(wx + b)$

1. There is a weight w
2. There is a bias b
3. There can be an activation, which is Sigmoid in this case

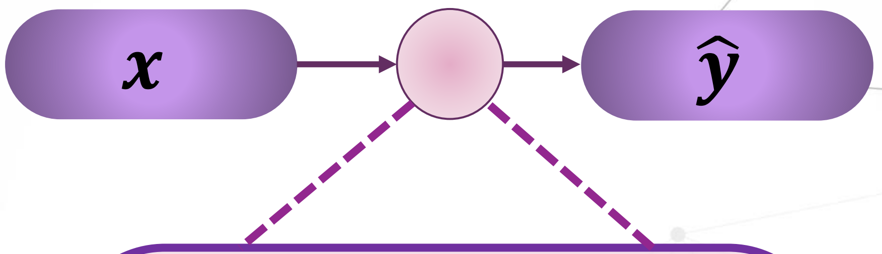
Single Layer Neural Network



$$\sigma(wx + b)$$

1. There is a weight w
2. There is a bias b
3. There can be an activation, which is Sigmoid in this case

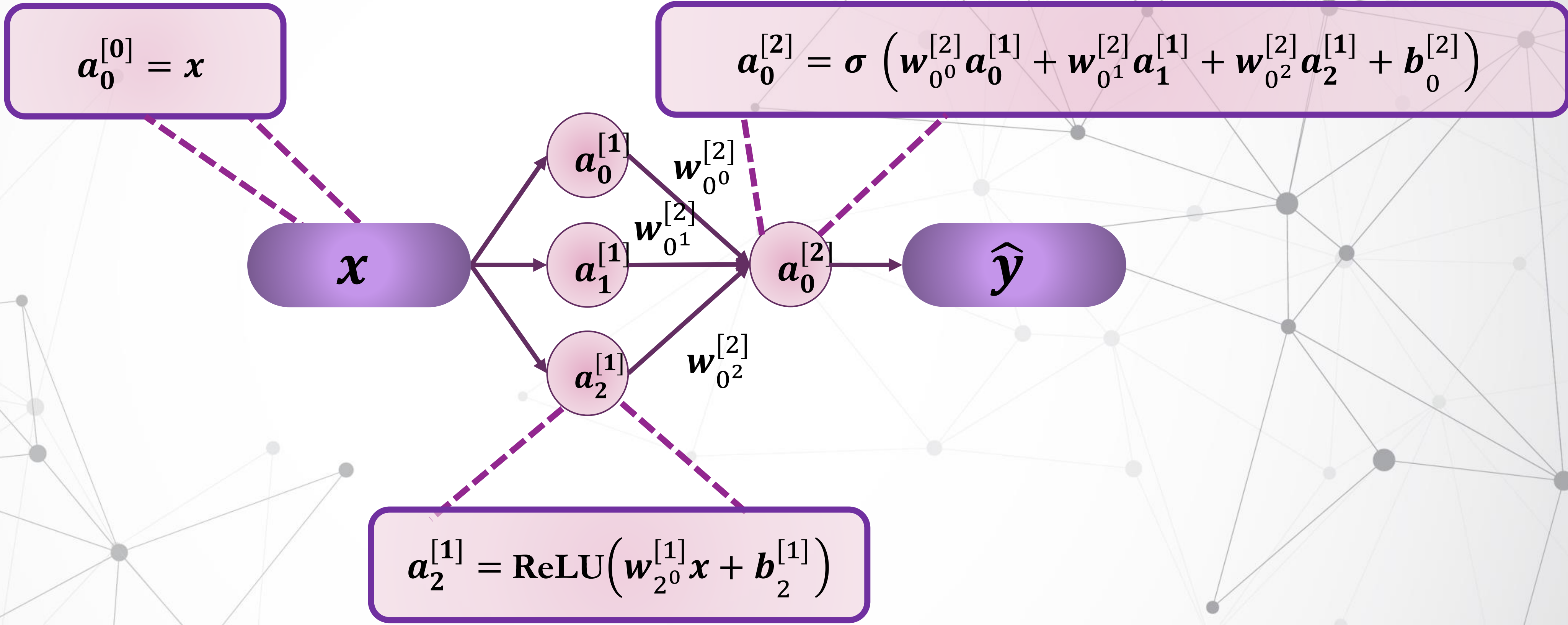
Single Layer Neural Network



$\boldsymbol{\sigma}(\boldsymbol{wx} + \boldsymbol{b})$

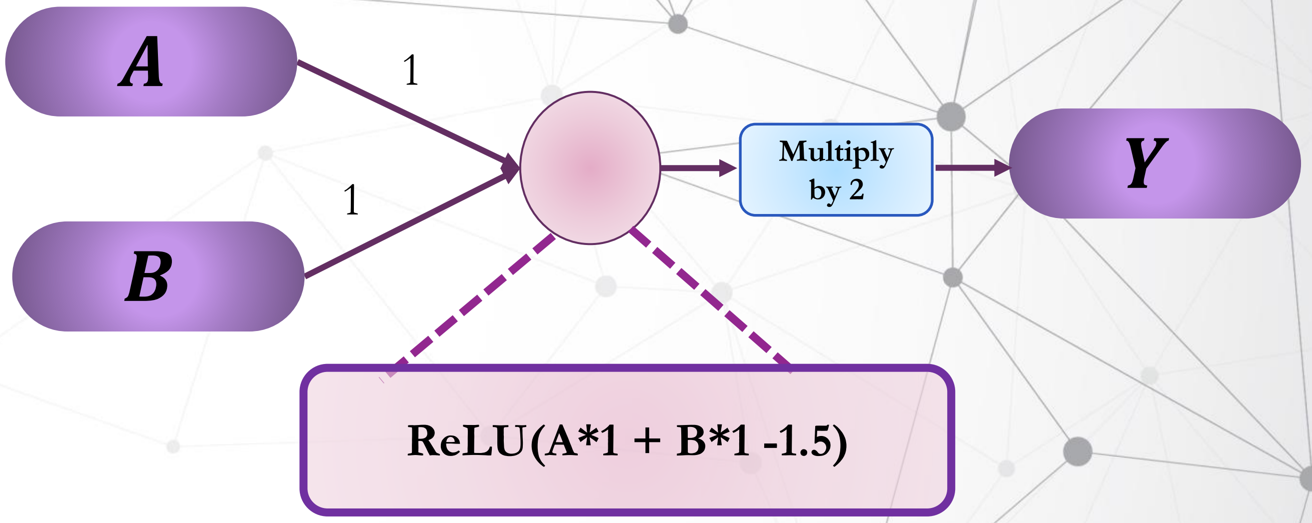
$$\mathrm{z} = \boldsymbol{wx} + \boldsymbol{b}$$
$$\mathrm{a} = \boldsymbol{\sigma}(\mathrm{z})$$

2 Layer Neural Network



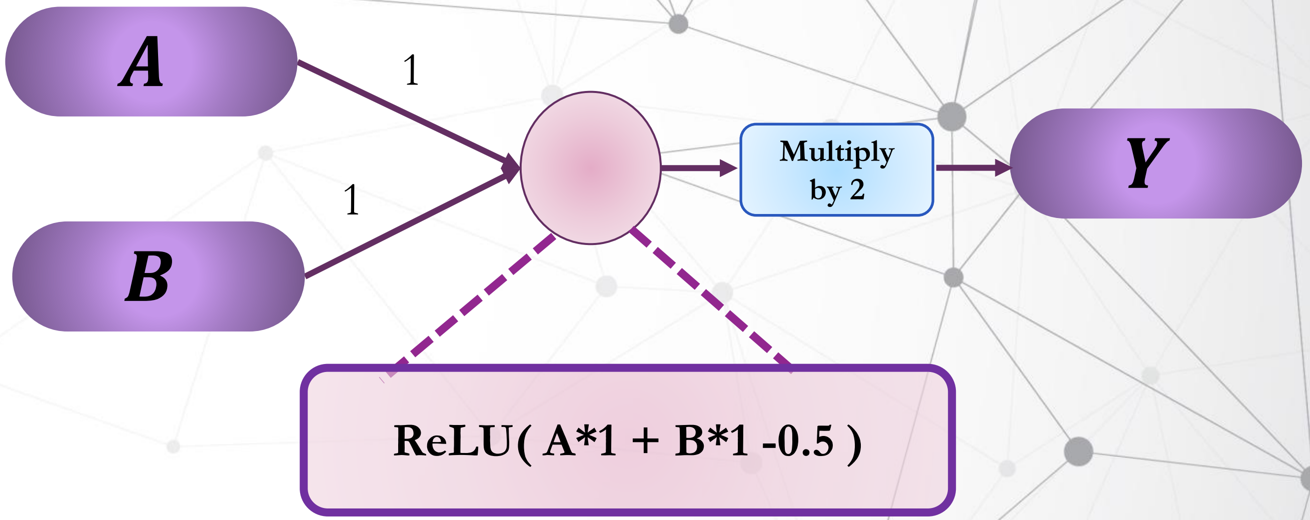
Logic Gates using ANN

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



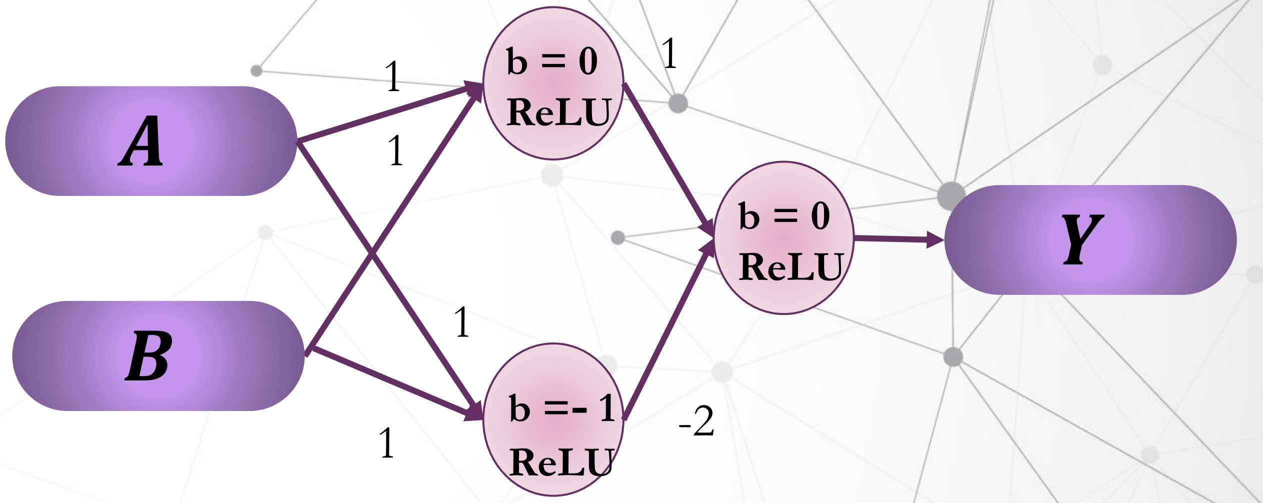
Logic Gates using ANN

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



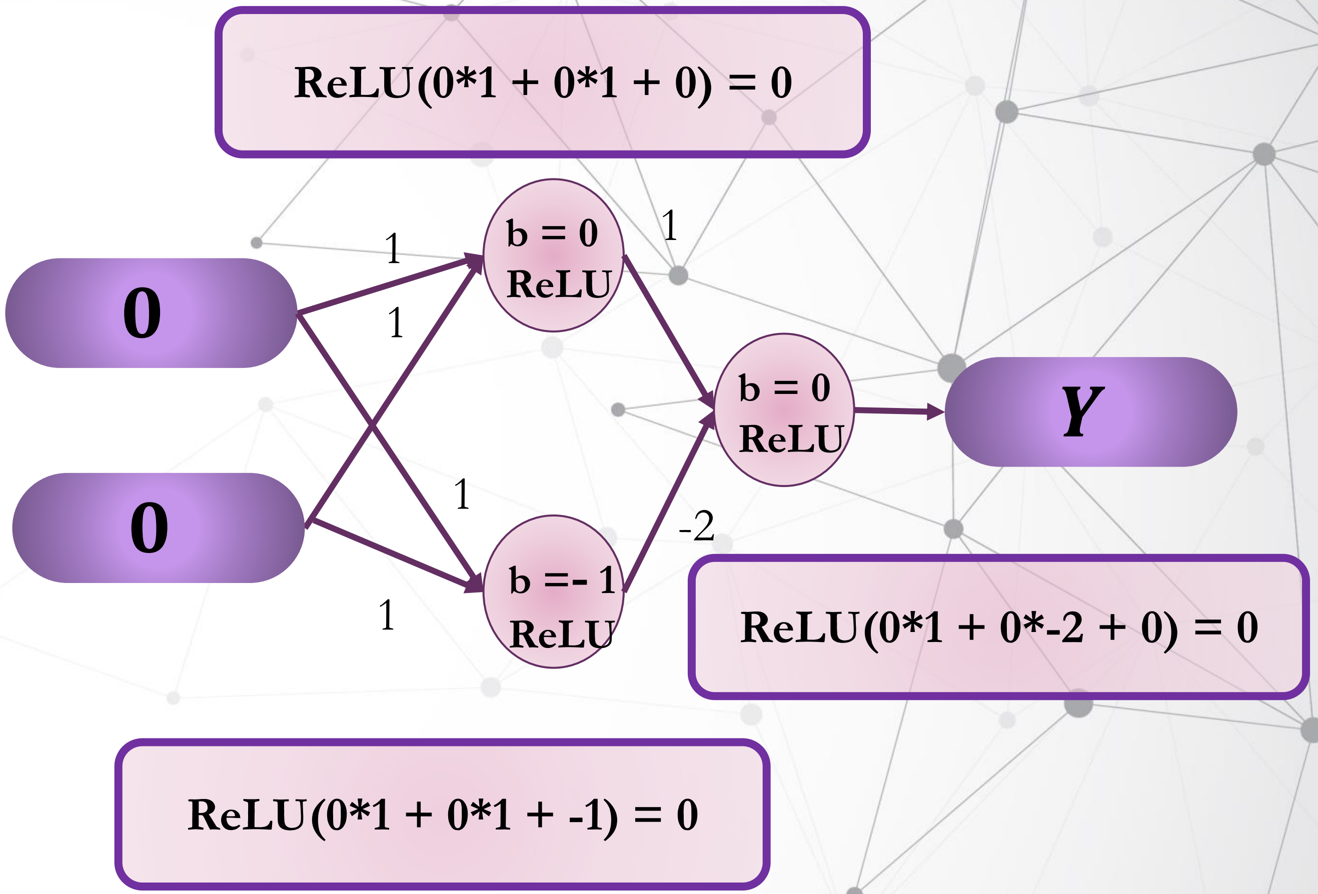
Logic Gates using ANN

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



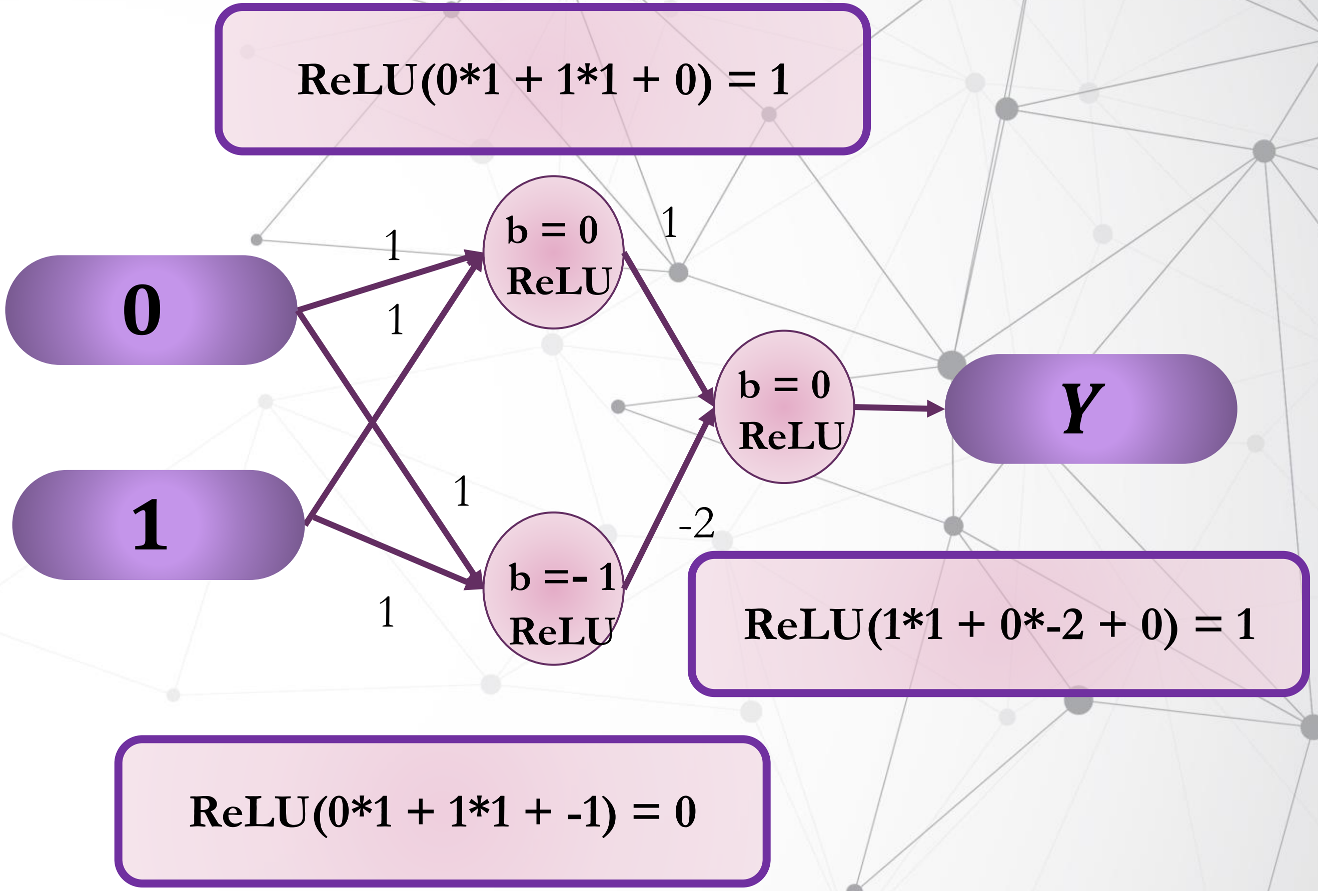
Logic Gates using ANN

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



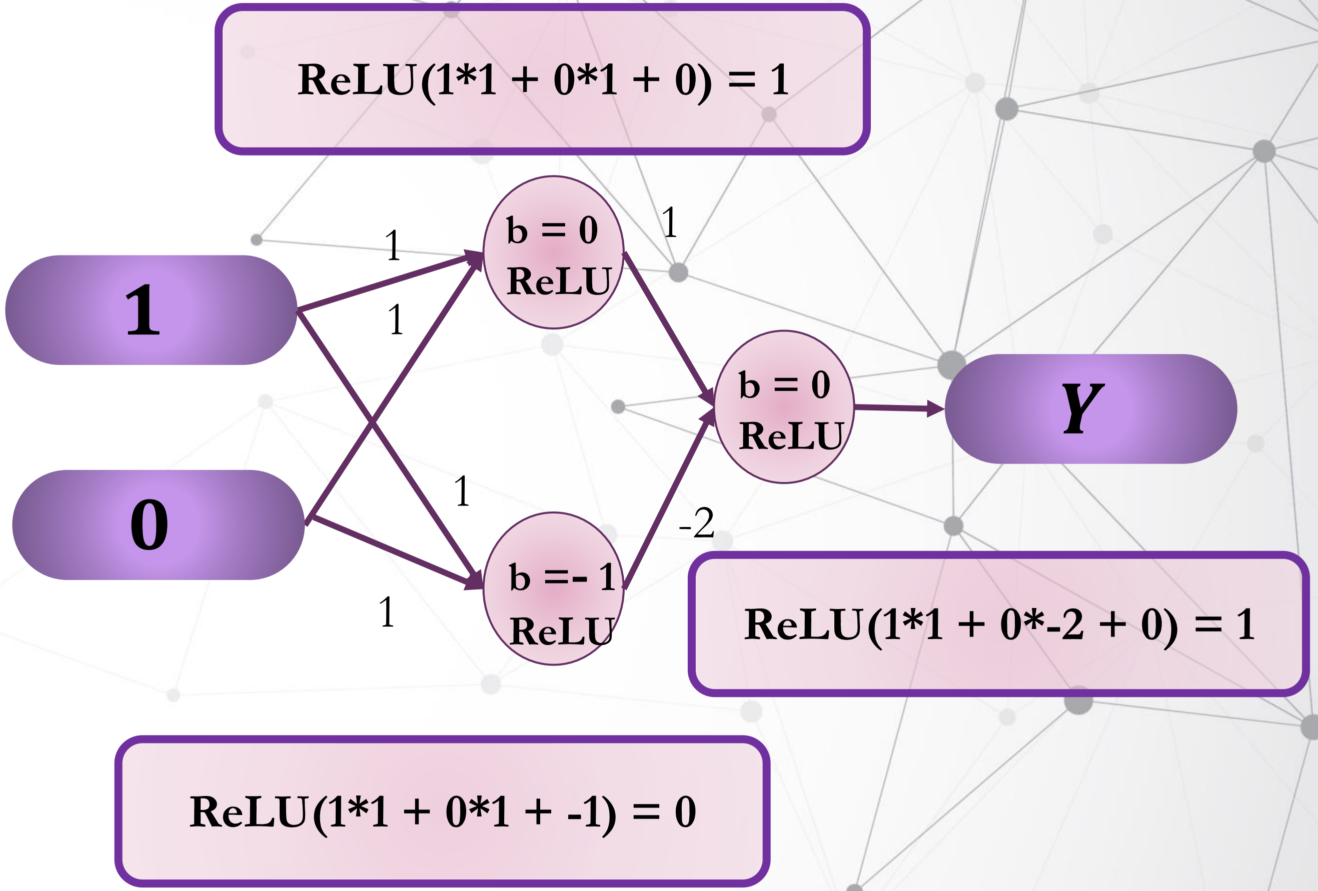
Logic Gates using ANN

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



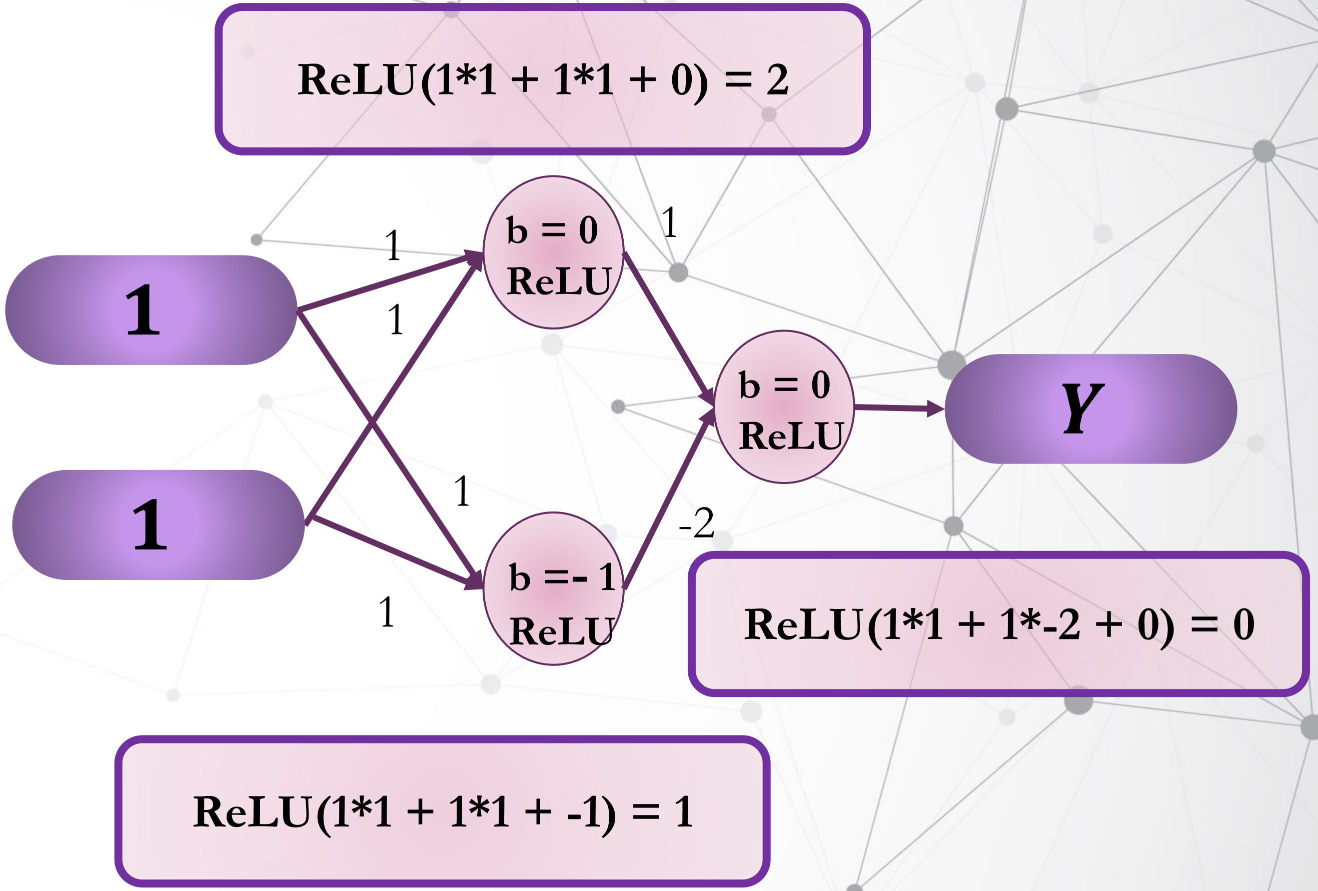
Logic Gates using ANN

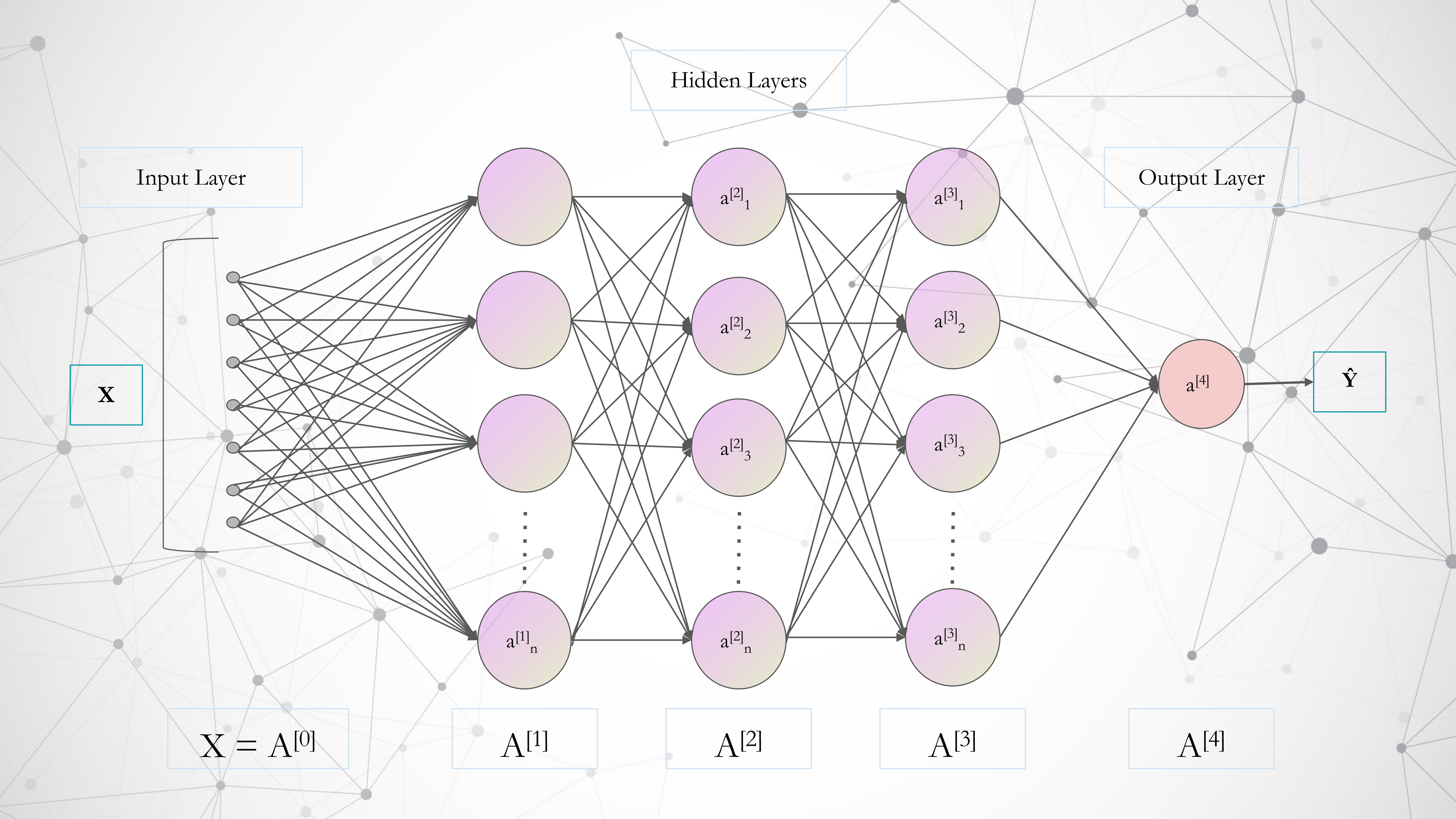
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



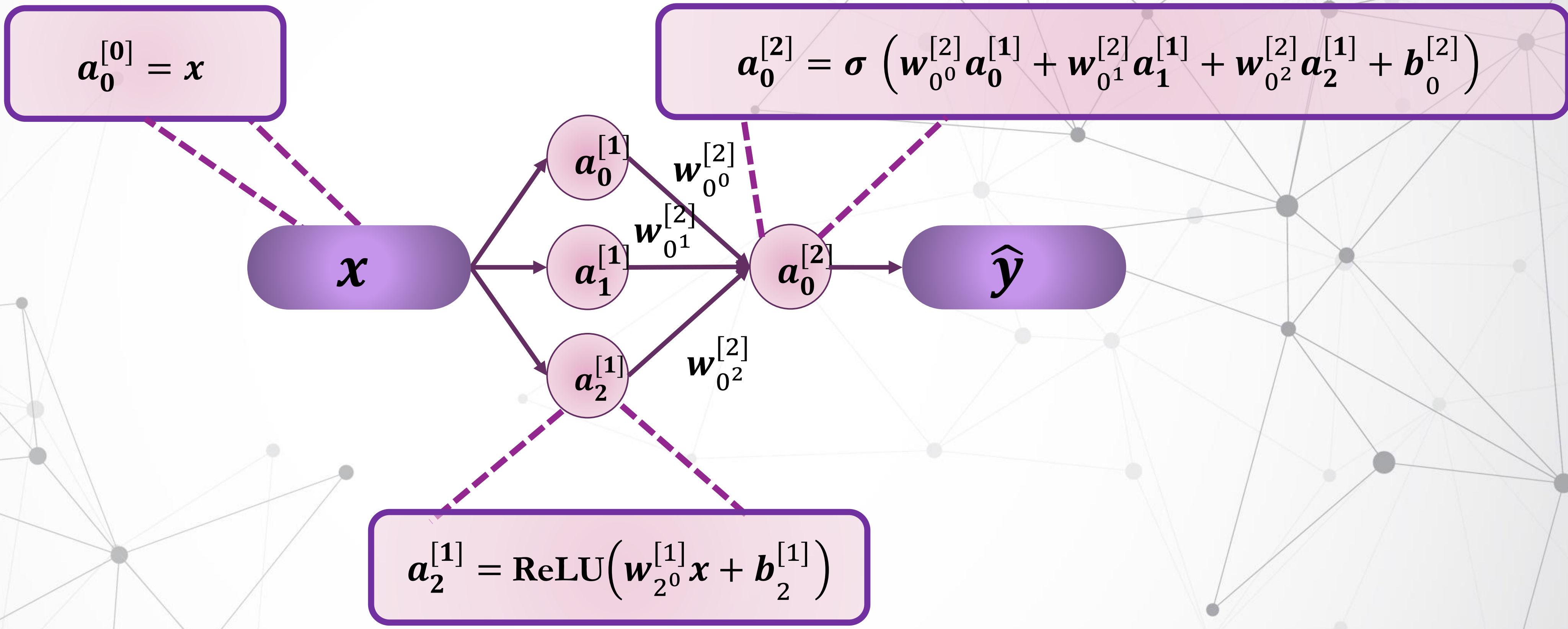
Logic Gates using ANN

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0




Hidden Layers
Input Layer
Output Layer
X
a[2]1
a[2]2
a[2]3
a[2]n
a[3]1
a[3]2
a[3]3
a[3]n
a[1]n
a[4]
Ŷ
X = A[0]
A[1]
A[2]
A[3]
A[4]

2 Layer Neural Network



tmax	tmin	rain
60.0	35.0	0.0
52.0	39.0	0.0
52.0	35.0	0.0
53.0	36.0	0.0
52.0	35.0	0.0

tmax_tomorrow
52.0
52.0
53.0
52.0
50.0

$$\hat{y} = \left(\sum_{i=1}^{3} w_i x_i + b\right)$$

$$= w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

$$= \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b$$

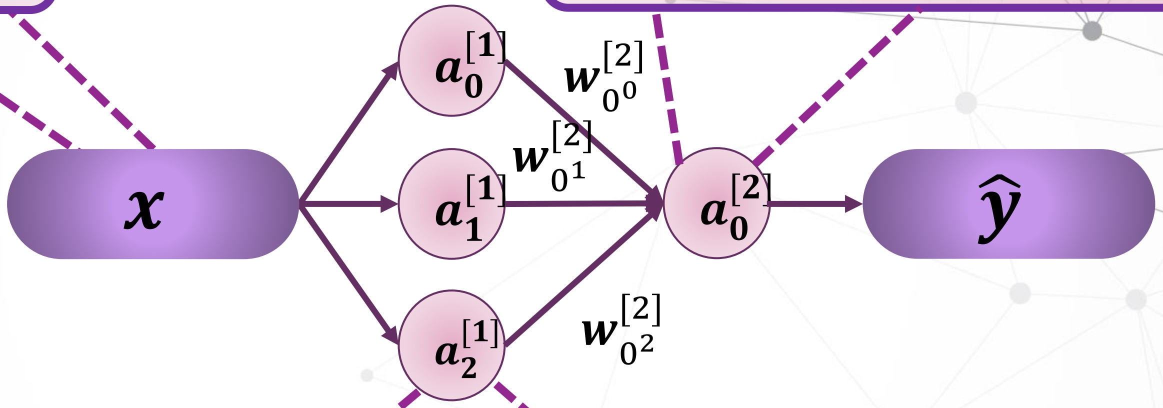
$$\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

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

2 Layer Neural Network

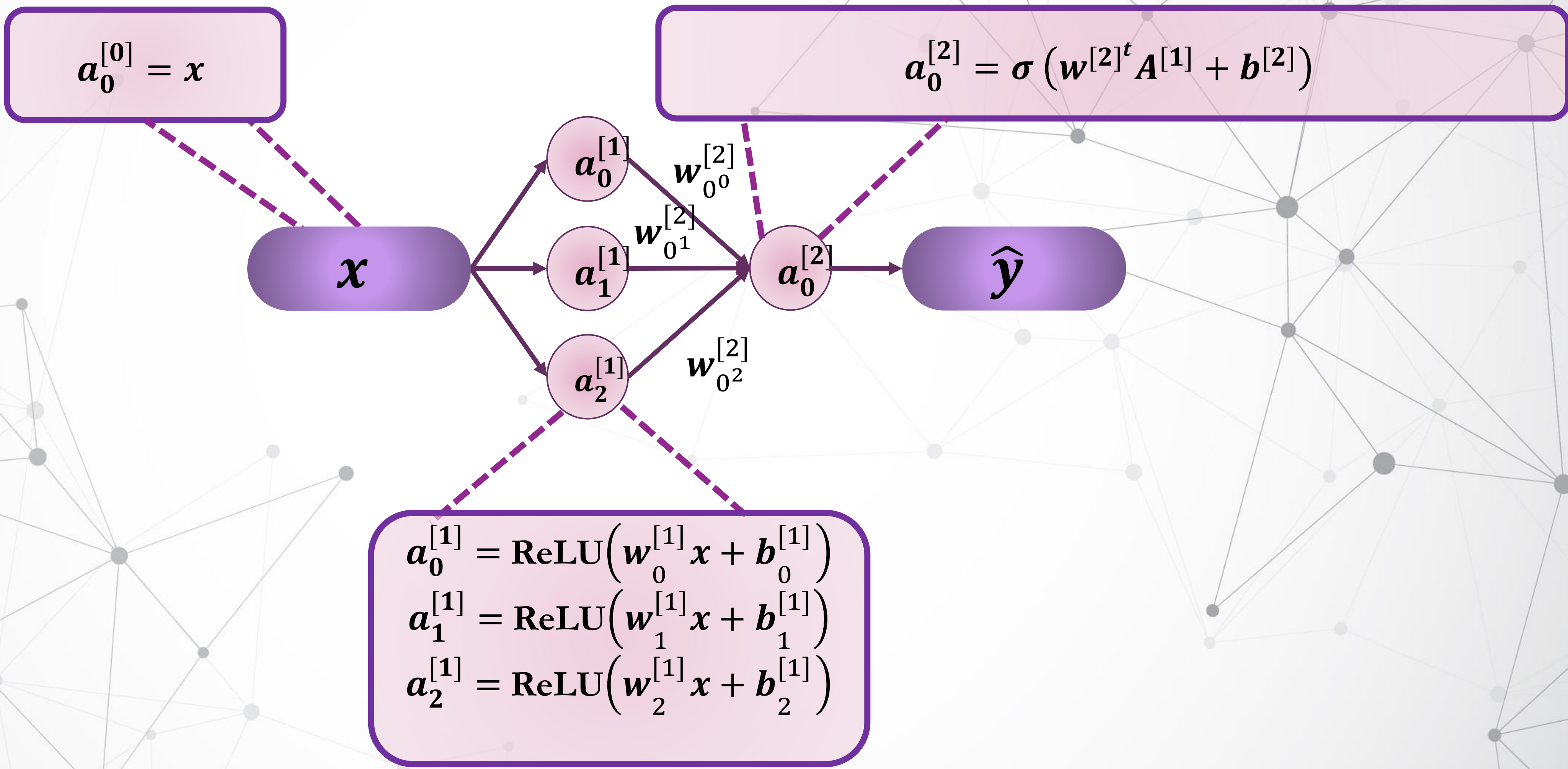
$$a_0^{[0]} = x$$

$$a_0^{[2]} = \sigma\left(w_{0^1}^{[2]} a_0^{[1]} + w_{0^1}^{[2]} a_1^{[1]} + w_{0^2}^{[2]} a_2^{[1]} + b_0^{[2]}\right)$$



$$a_2^{[1]} = \mathrm{ReLU}\left(w_{2^0}^{[1]} x + b_2^{[1]}\right)$$

2 Layer Neural Network



tmax	tmin	rain	tmax_tomorrow
60.0	35.0	0.00	52.0
52.0	39.0	0.00	52.0
52.0	35.0	0.00	53.0
53.0	36.0	0.00	52.0
52.0	35.0	0.00	50.0
50.0	38.0	0.00	52.0
52.0	43.0	0.00	56.0
56.0	49.0	0.24	54.0
54.0	50.0	0.40	57.0
57.0	50.0	0.00	57.0
57.0	50.0	0.31	58.0
58.0	52.0	0.05	59.0
59.0	54.0	0.25	58.0
58.0	53.0	1.94	56.0
56.0	51.0	0.63	61.0
61.0	56.0	0.62	59.0
59.0	54.0	0.00	58.0
58.0	53.0	0.00	60.0
60.0	53.0	0.14	60.0
60.0	53.0	1.21	61.0

```
print(len(df))
```

```
13509
```

Dataset length, m = 13509

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

tmax	tmin	rain	tmax_tomorrow
60.0	35.0	0.00	52.0
52.0	39.0	0.00	52.0
52.0	35.0	0.00	53.0
53.0	36.0	0.00	52.0
52.0	35.0	0.00	50.0
50.0	38.0	0.00	52.0
52.0	43.0	0.00	56.0
56.0	49.0	0.24	54.0
54.0	50.0	0.40	57.0
57.0	50.0	0.00	57.0
57.0	50.0	0.31	58.0
58.0	52.0	0.05	59.0
59.0	54.0	0.25	58.0
58.0	53.0	1.94	56.0
56.0	51.0	0.63	61.0
61.0	56.0	0.62	59.0
59.0	54.0	0.00	58.0
58.0	53.0	0.00	60.0
60.0	53.0	0.14	60.0
60.0	53.0	1.21	61.0

Batch_size = 5

Step 1:

60.0	35.0	0.00	52.0
52.0	39.0	0.00	52.0
52.0	35.0	0.00	53.0
53.0	36.0	0.00	52.0
52.0	35.0	0.00	50.0

```
print(len(df))
```

```
13509
```

Dataset length, m = 13509

Calculate Prediction

$$\hat{y} = w^T x + b$$

Estimate Error

$$J(w,b) = (\hat{y} - y_{actual})^2$$

Update Parameters

$$w = w - \frac{\partial J}{\partial w} \qquad b = b - \frac{\partial J}{\partial b}$$

tmax	tmin	rain	tmax_tomorrow
50.0	38.0	0.00	52.0
52.0	43.0	0.00	56.0
56.0	49.0	0.24	54.0
54.0	50.0	0.40	57.0
57.0	50.0	0.00	57.0
57.0	50.0	0.31	58.0
58.0	52.0	0.05	59.0
59.0	54.0	0.25	58.0
58.0	53.0	1.94	56.0
56.0	51.0	0.63	61.0
61.0	56.0	0.62	59.0
59.0	54.0	0.00	58.0
58.0	53.0	0.00	60.0
60.0	53.0	0.14	60.0
60.0	53.0	1.21	61.0

Batch_size = 5

Step 2:

50.0	38.0	0.00	52.0
52.0	43.0	0.00	56.0
56.0	49.0	0.24	54.0
54.0	50.0	0.40	57.0
57.0	50.0	0.00	57.0

```
print(len(df))
13509
```

Dataset length, m = 13509

Calculate Prediction

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

Estimate Error

$$J(w,b) = (\hat{y} - y_{actual})^2$$

Update Parameters

$$w = w - \frac{\partial J}{\partial w} \qquad b = b - \frac{\partial J}{\partial b}$$

tmax	tmin	rain	tmax_tomorrow

Step n (n=2702):

61.0	56.0	0.62	59.0
59.0	54.0	0.00	58.0
58.0	53.0	0.00	60.0
60.0	53.0	0.14	60.0
60.0	53.0	1.21	61.0

```
print(len(df))
```

```
13509
```

Dataset length, m = 13509

n = ceil(m / batch_size) = 13509 / 5 = 2702

Epoch 1 Completed!

Calculate Prediction

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

Estimate Error

$J(w,b) = (\hat{y} - y_{actual})^2$

Update Parameters

$w = w - \frac{\partial J}{\partial w}$ $\qquad b = b - \frac{\partial J}{\partial b}$

tmax	tmin	rain	tmax_tomorrow
60.0	35.0	0.00	52.0
52.0	39.0	0.00	52.0
52.0	35.0	0.00	53.0
53.0	36.0	0.00	52.0
52.0	35.0	0.00	50.0
50.0	38.0	0.00	52.0
52.0	43.0	0.00	56.0
56.0	49.0	0.24	54.0
54.0	50.0	0.40	57.0
57.0	50.0	0.00	57.0
57.0	50.0	0.31	58.0
58.0	52.0	0.05	59.0
59.0	54.0	0.25	58.0
58.0	53.0	1.94	56.0
56.0	51.0	0.63	61.0
61.0	56.0	0.62	59.0
59.0	54.0	0.00	58.0
58.0	53.0	0.00	60.0
60.0	53.0	0.14	60.0
60.0	53.0	1.21	61.0

Batch_size = 5

Epoch 2, Step 1:

60.0	35.0	0.00	52.0
52.0	39.0	0.00	52.0
52.0	35.0	0.00	53.0
53.0	36.0	0.00	52.0
52.0	35.0	0.00	50.0

```
print(len(df))
```

```
13509
```

Dataset length, m = 13509

Calculate Prediction

$\hat{y} = w^T x + b$

Estimate Error

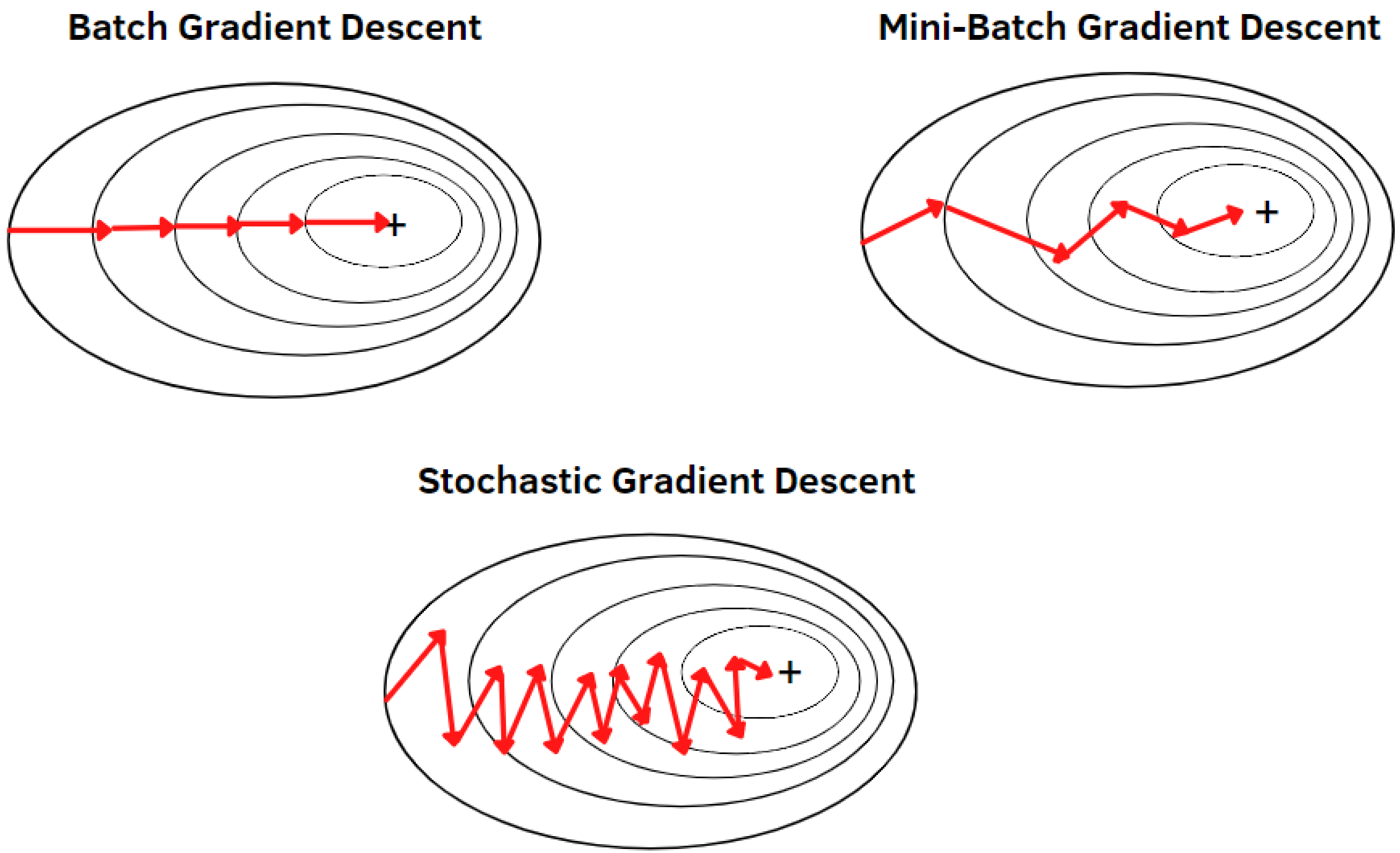
$J(w,b) = (\hat{y} - y_{actual})^2$

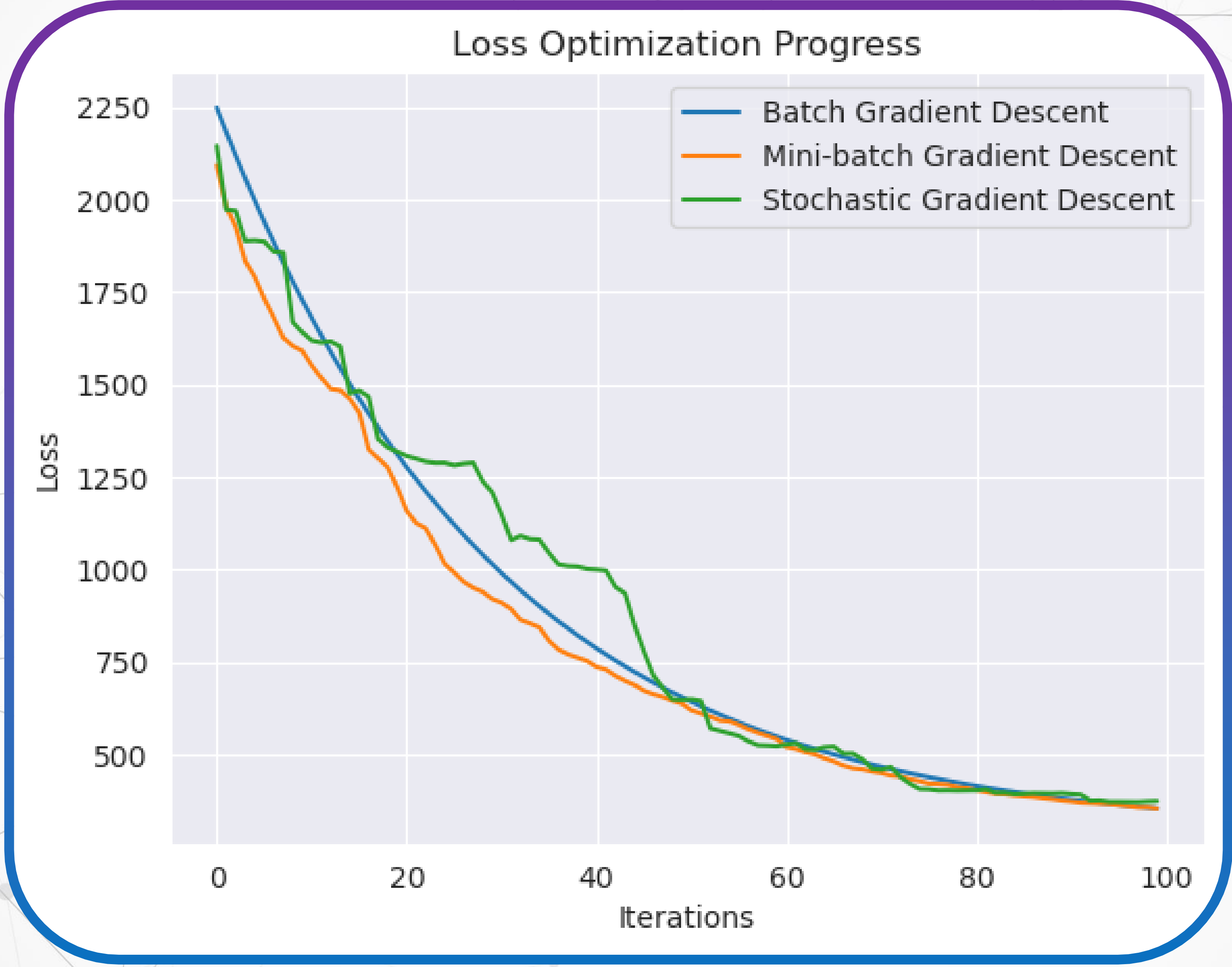
Update Parameters

$w = w - \frac{\partial J}{\partial w}$ $\quad b = b - \frac{\partial J}{\partial b}$

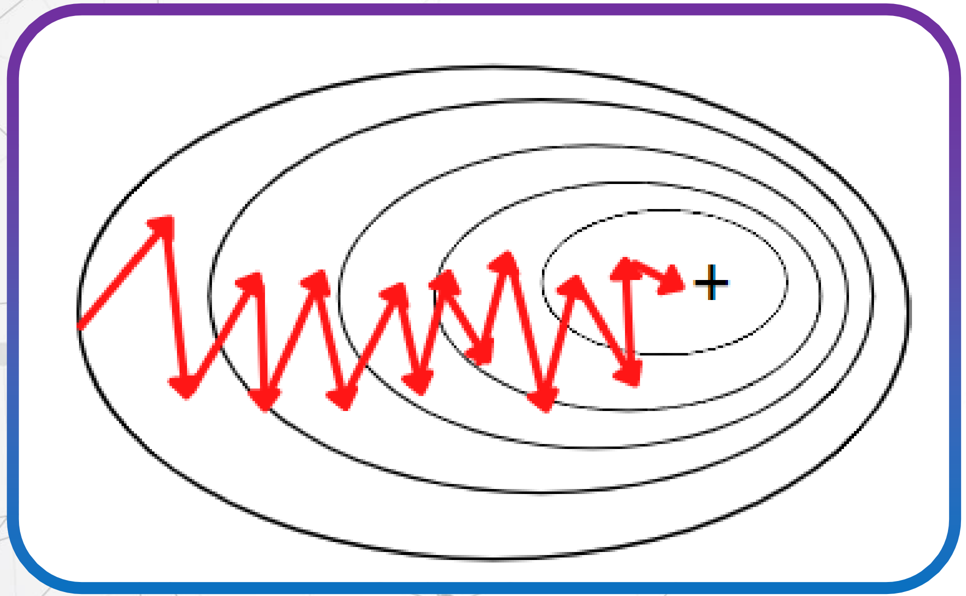
Types of Gradient Descent

- Batch Gradient Descent (batch size = dataset length)
- Mini-batch Gradient Descent (batch size < dataset length)
- Stochastic Gradient Descent (batch size = 1)


Batch Gradient Descent
Mini-Batch Gradient Descent
Stochastic Gradient Descent



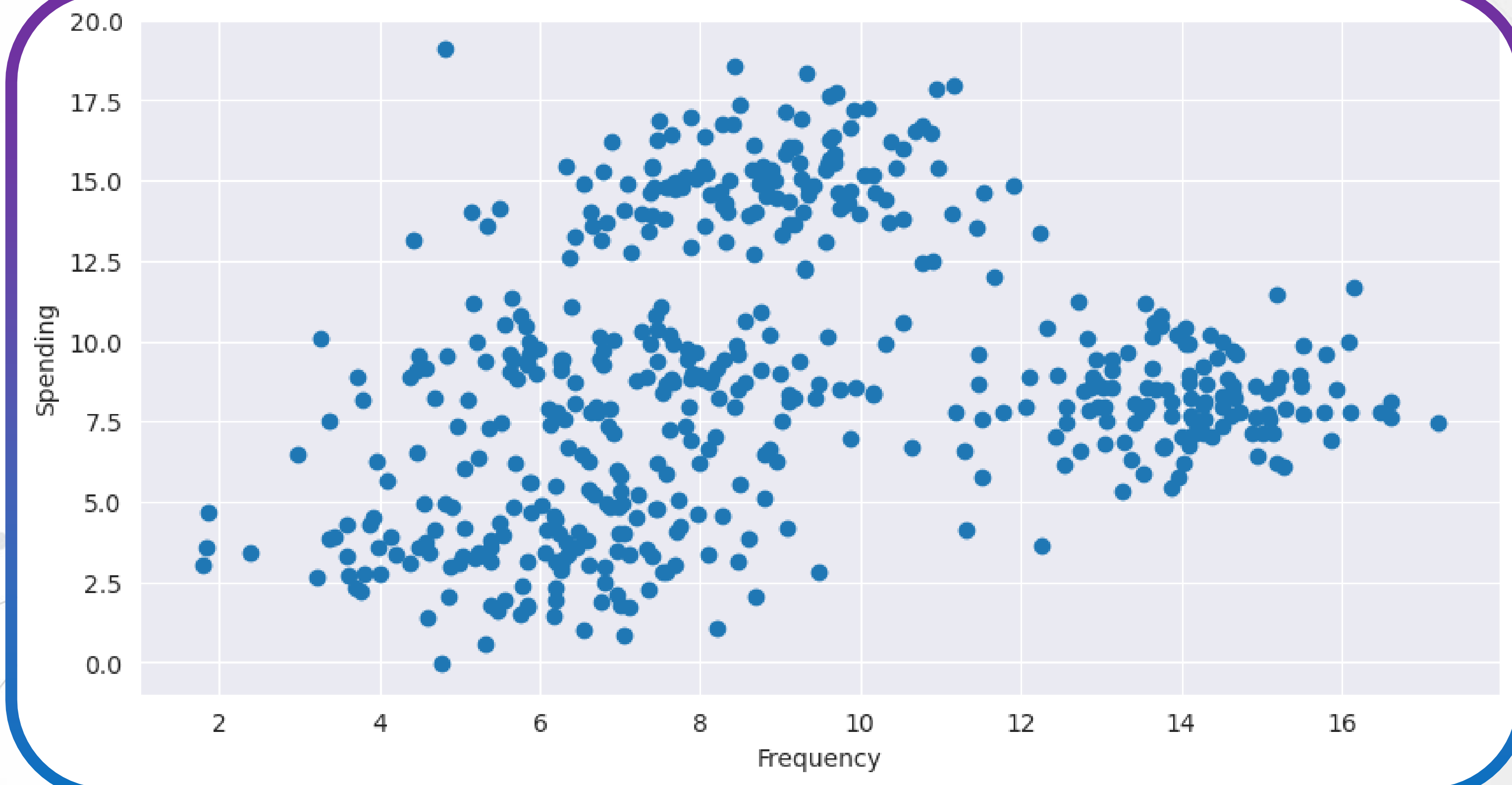
Loss Optimization Progress
Batch Gradient Descent
Mini-batch Gradient Descent
Stochastic Gradient Descent
2250
2000
1750
1500
1250
1000
750
500
Loss
0
20
40
60
80
100
Iterations



Gradient Descent Optimizer is tweaked to get Adam

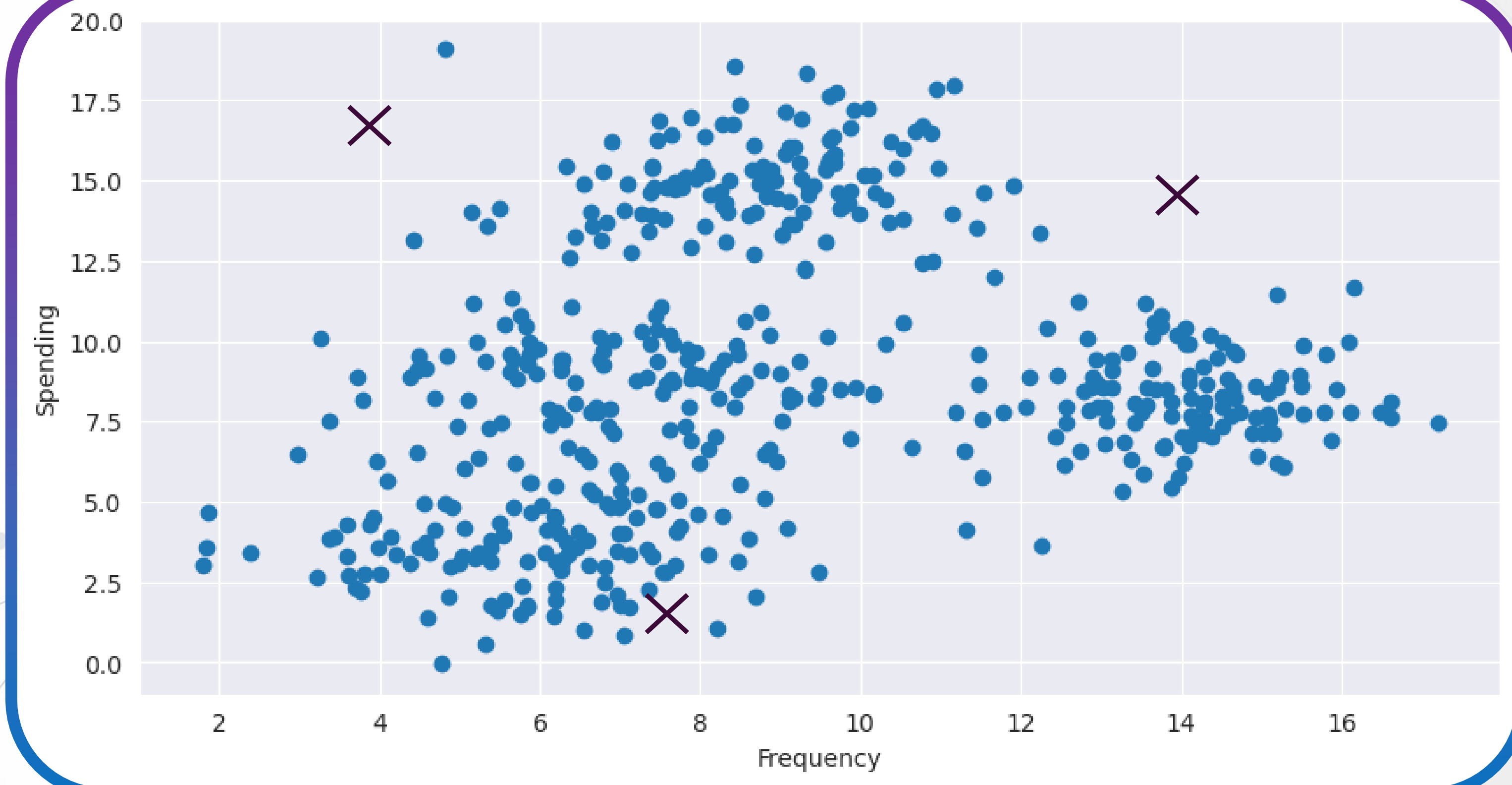
Up Next: Unsupervised Learning

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



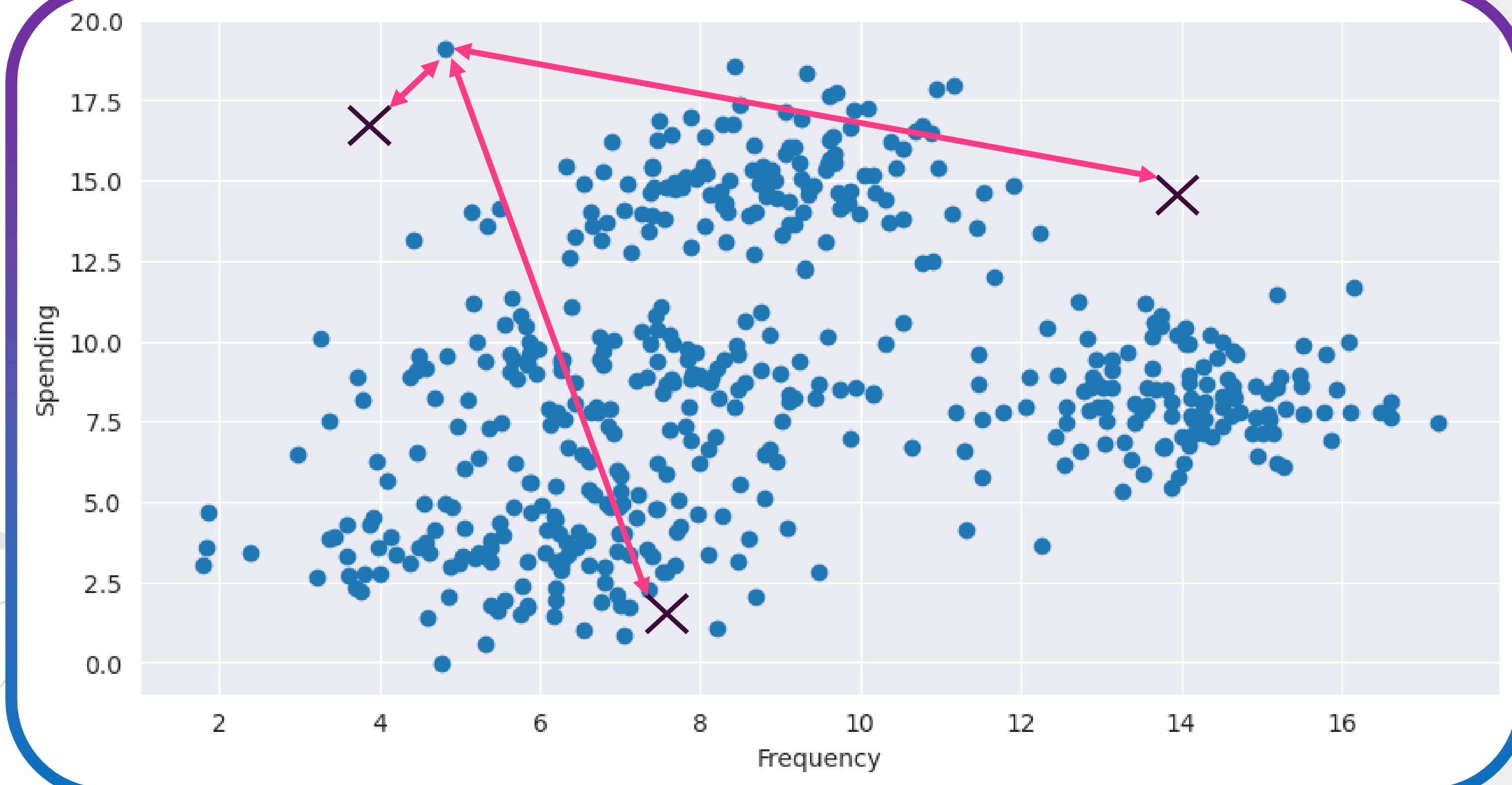
K = 3

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



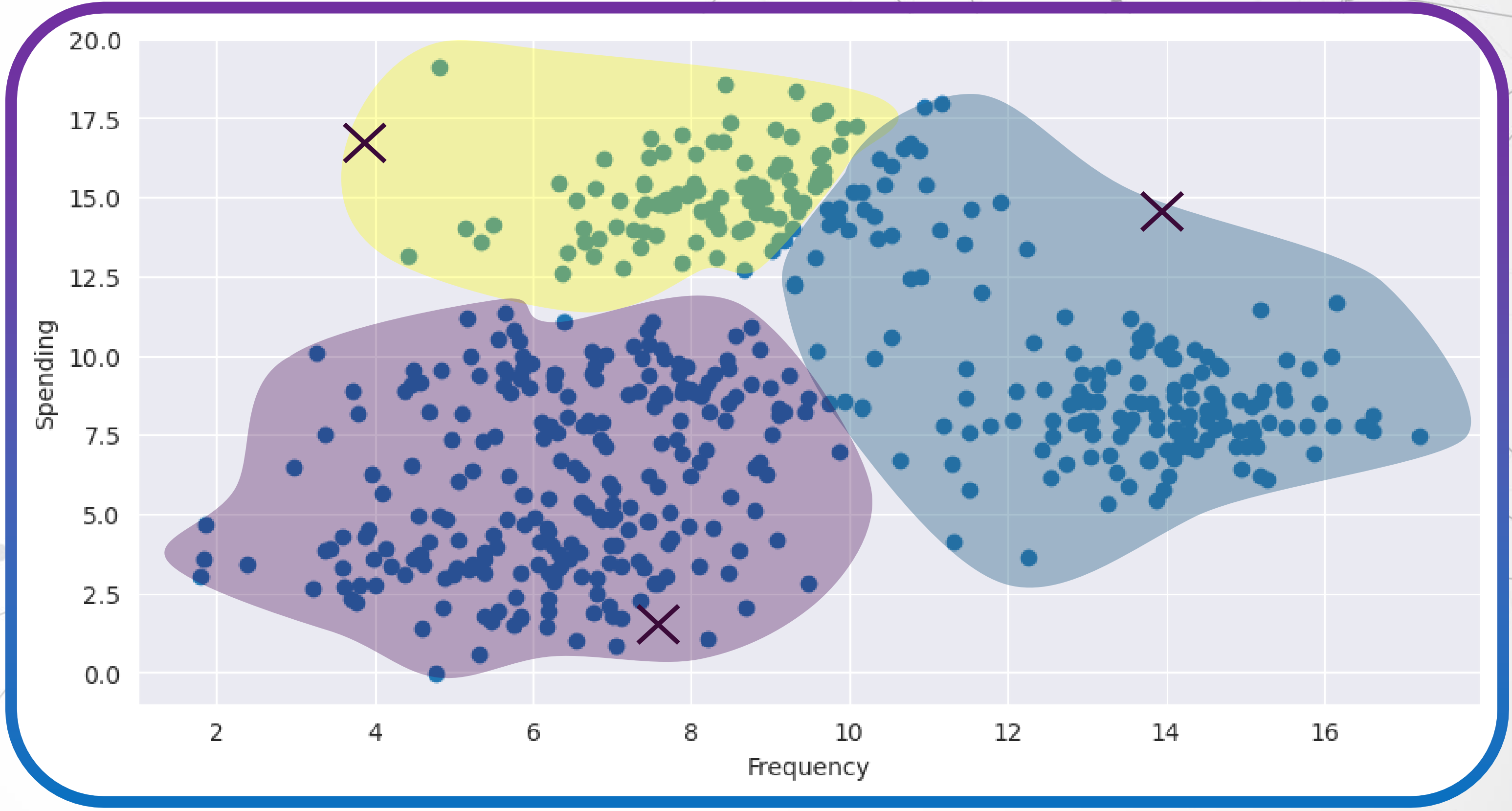
$K = 3$

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



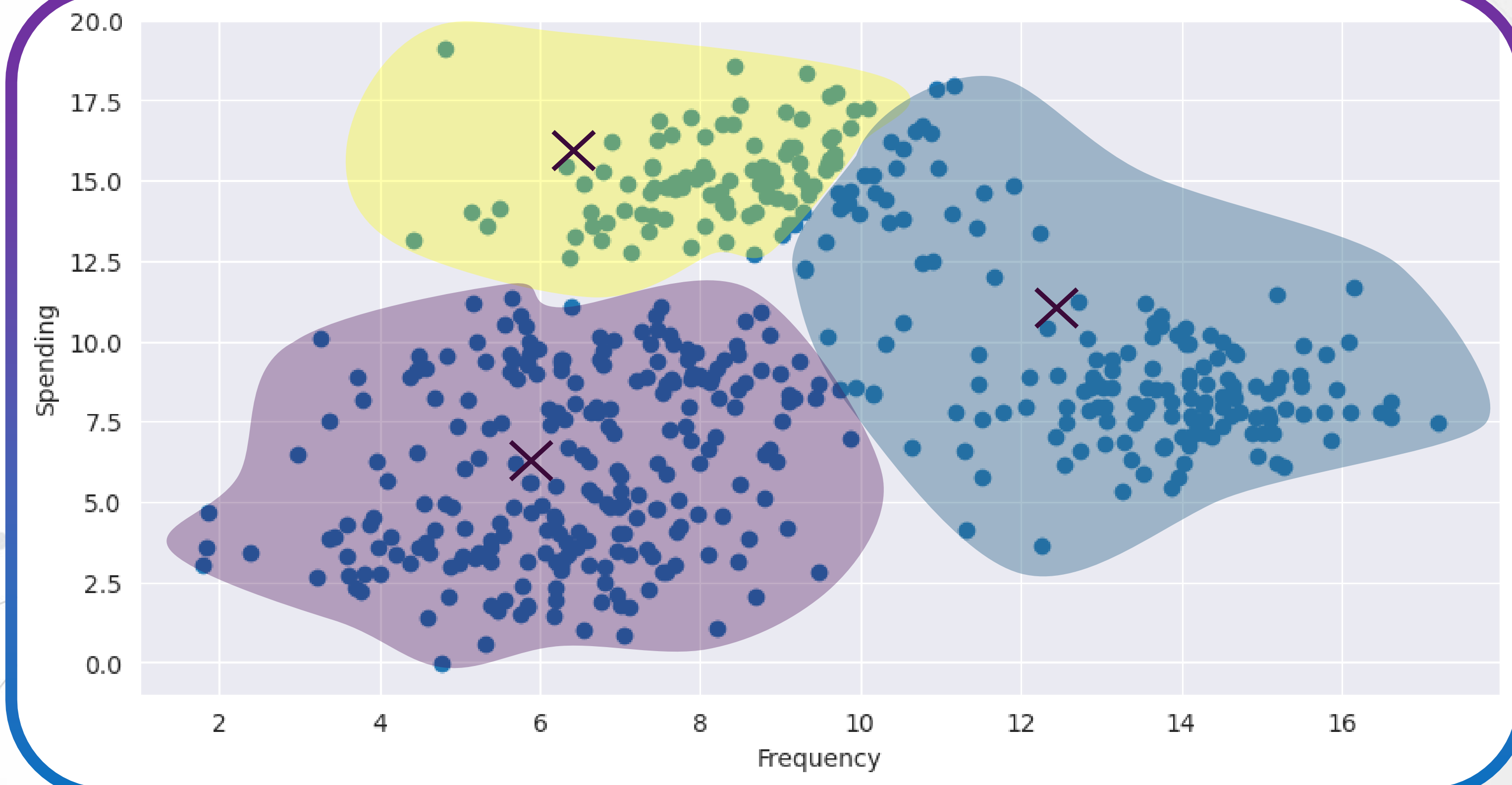
$K = 3$

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



K = 3

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



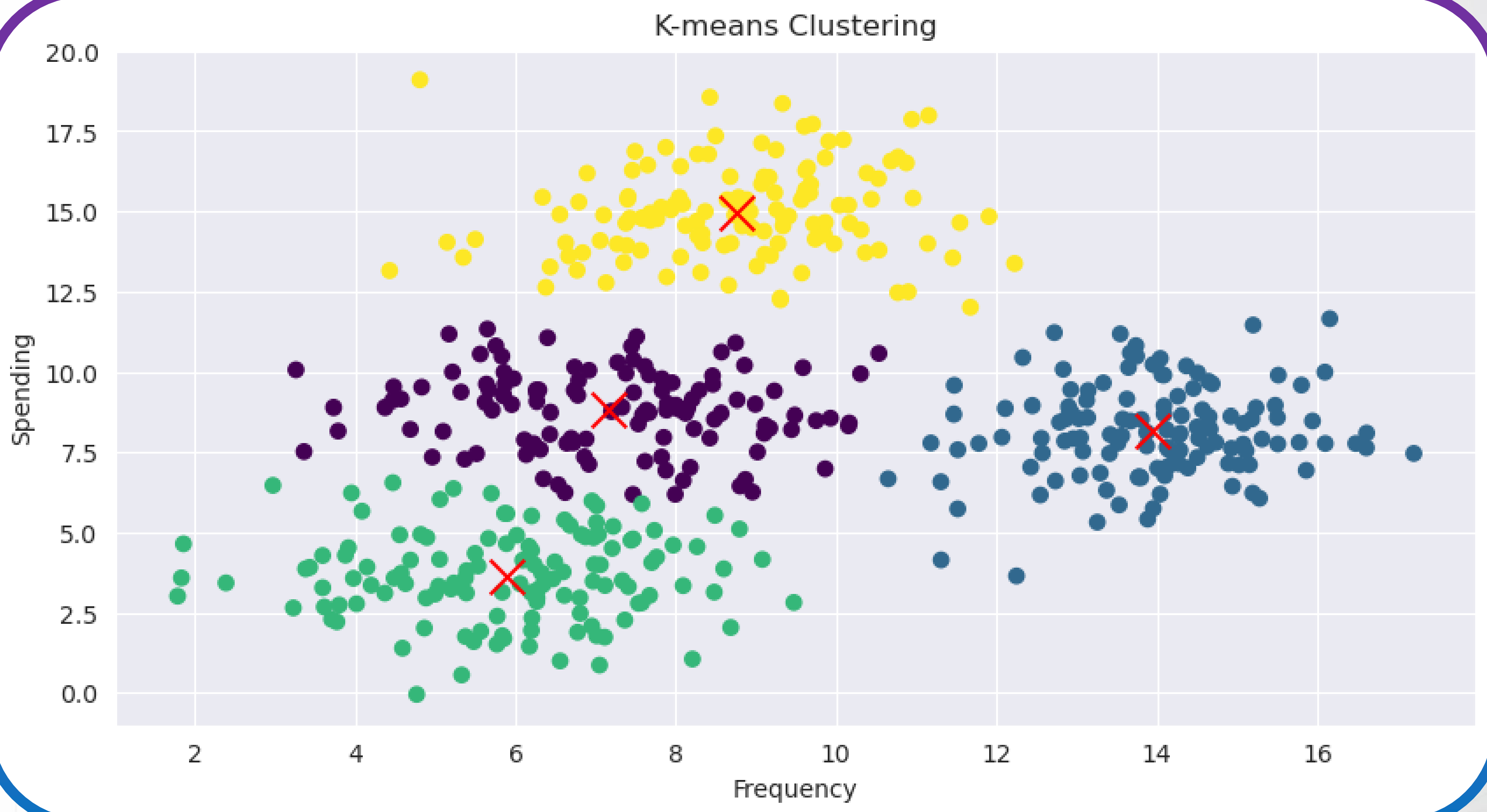
K = 3

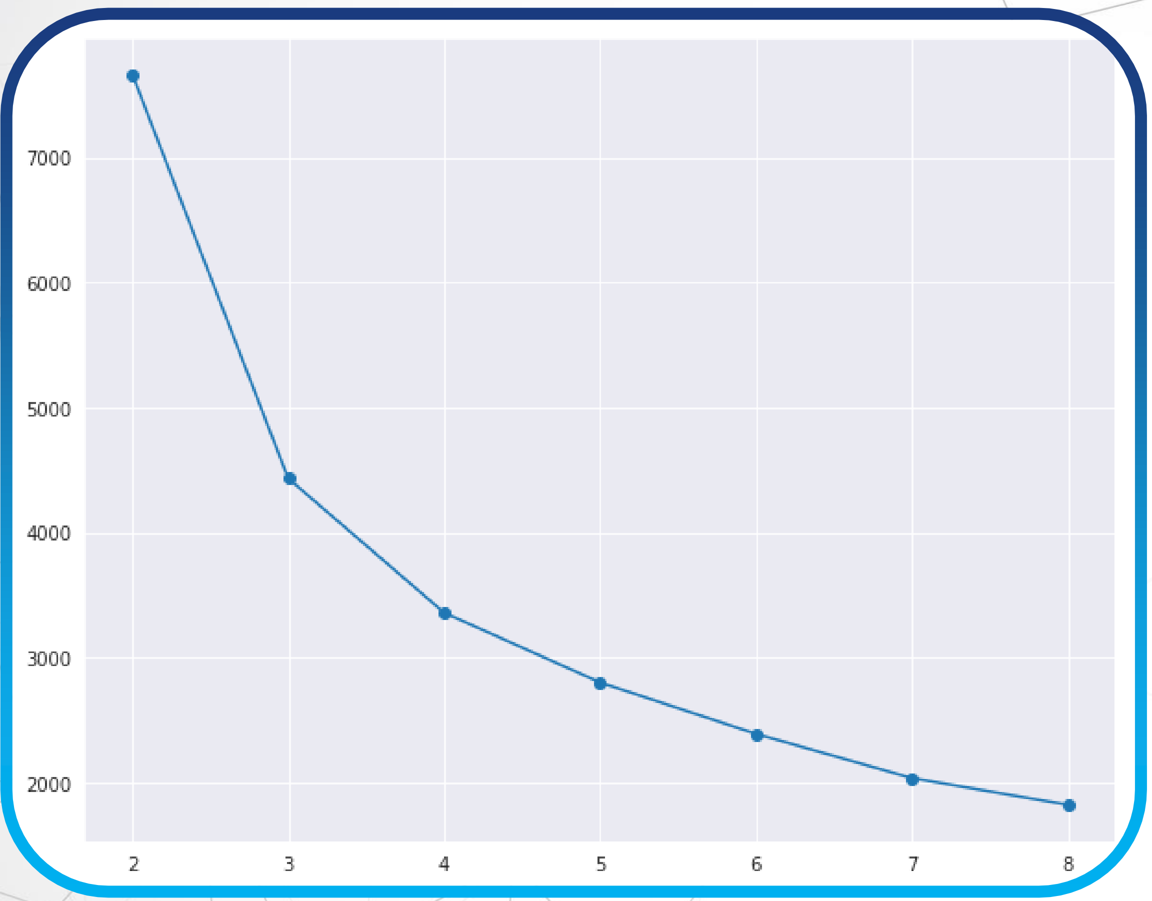
Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74

1. Form initial K centroids
2. For each points:
    1. Calculate distance from each centroids
    2. Assign the closest centroid to the point
3. Calculate the mean of the points assigned to each centroid and reassign each centroids
4. Go to step 2

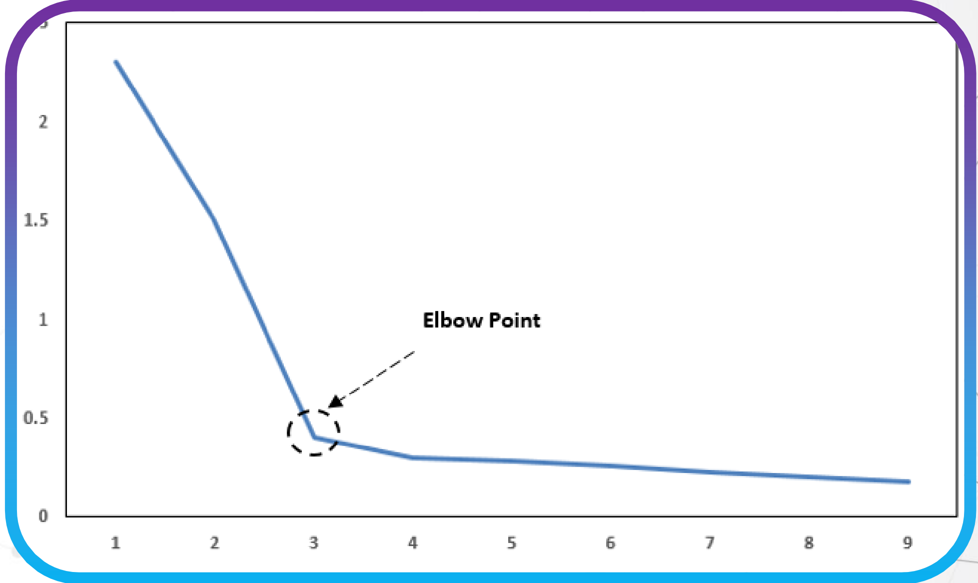
K = 4

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



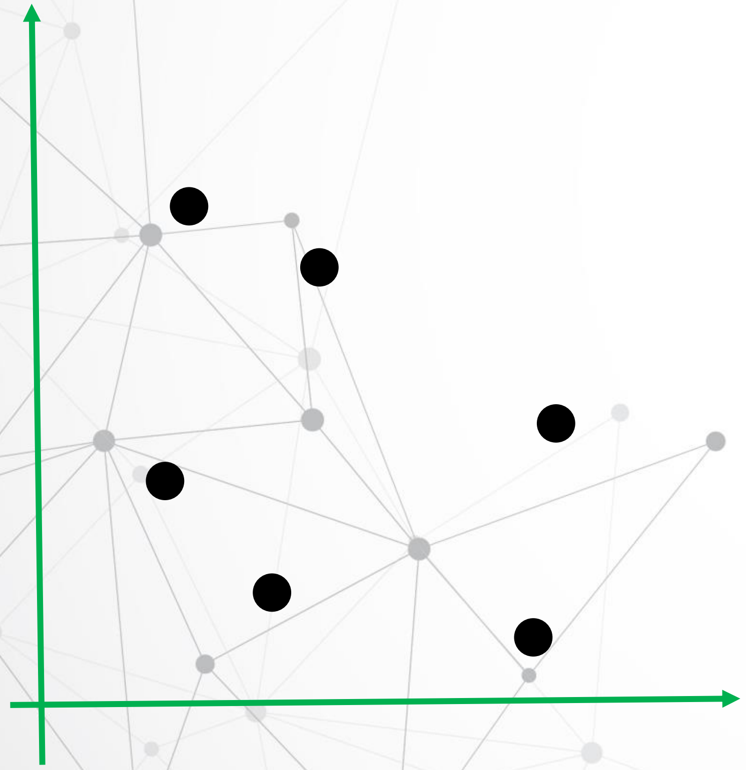


7000
6000
5000
4000
3000
2000
2
3
4
5
6
7
8

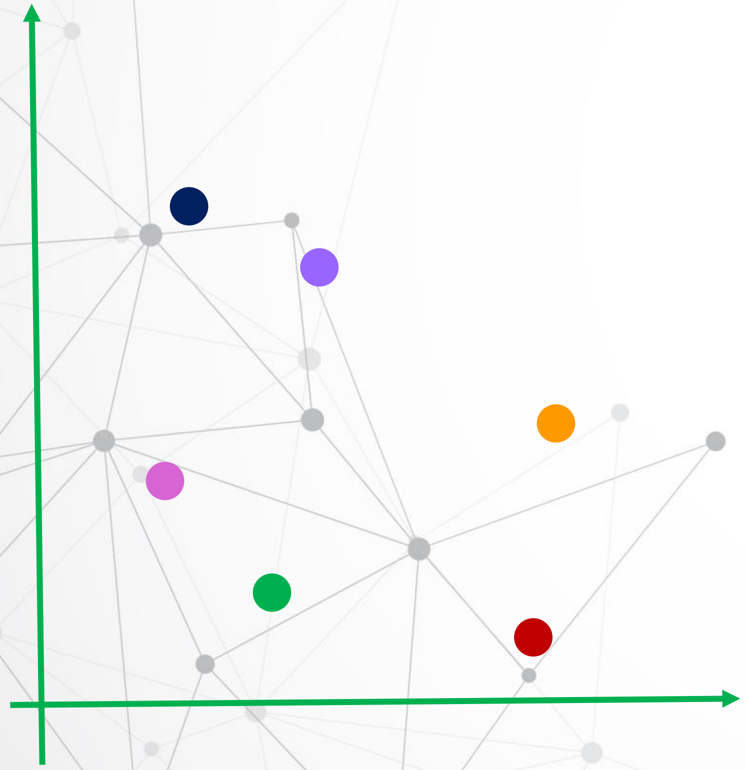


2
1.5
1
0.5
0
Elbow Point
1
2
3
4
5
6
7
8
9

Hierarchical Clustering

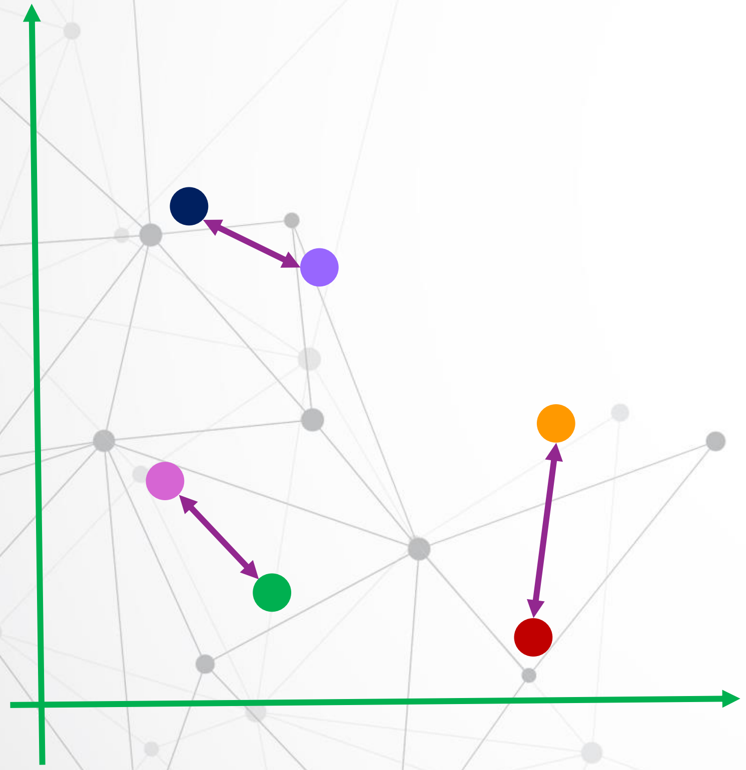


Hierarchical Clustering



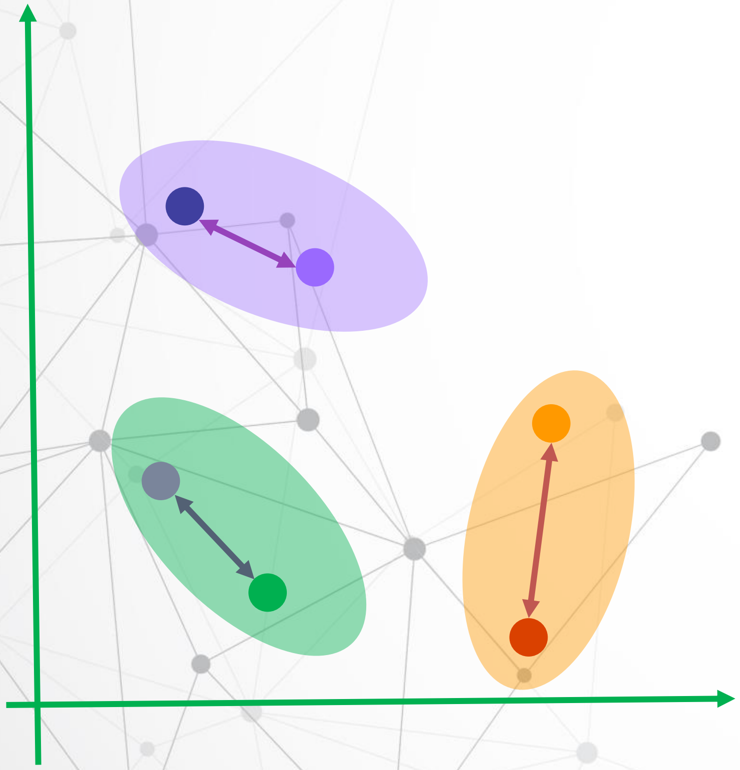
1. Assign a cluster to each datapoints

Hierarchical Clustering



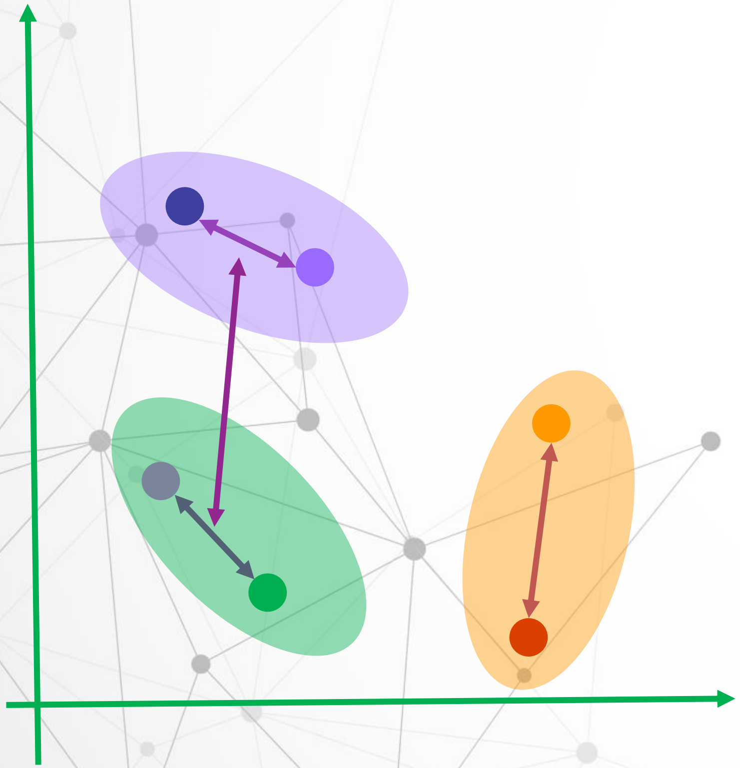
1. Assign a cluster to each datapoints
2. For each data point:
    1. Calculate the closest data point
    2. Merge their clusters

Hierarchical Clustering



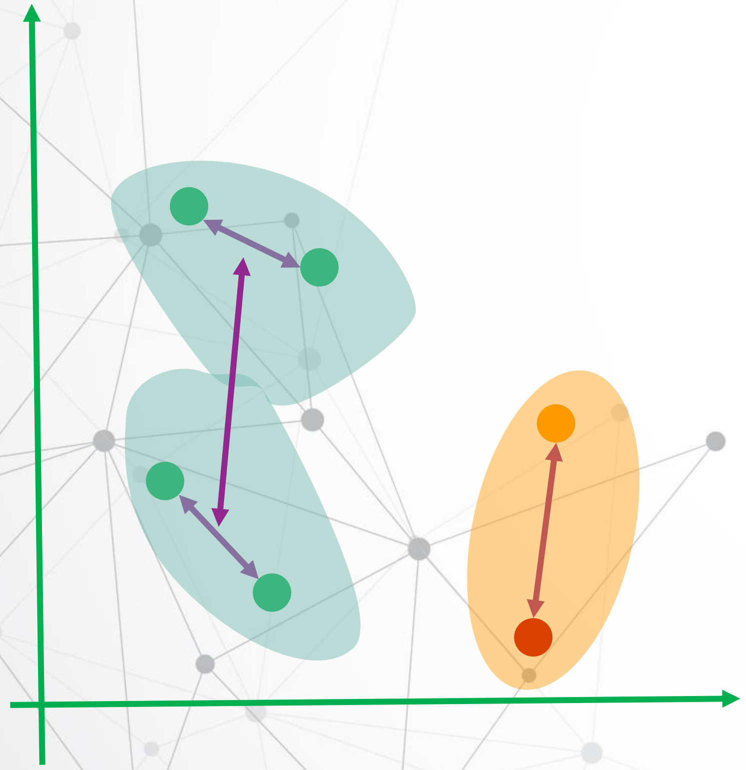
1. Assign a cluster to each datapoints
2. For each data point:
   1. Calculate the closest data point
   2. Merge their clusters
3. Repeat 2

Hierarchical Clustering



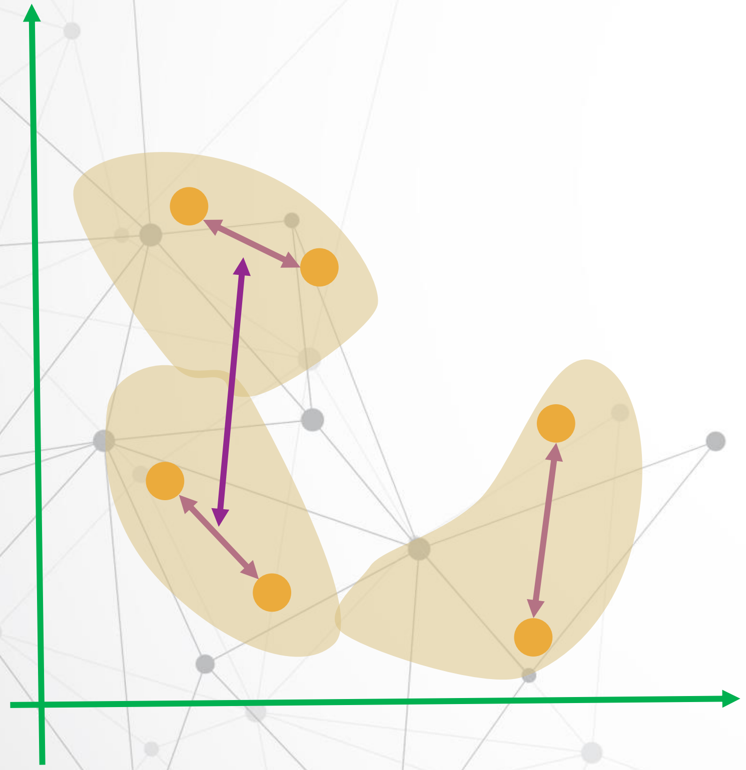
1. Assign a cluster to each datapoints
2. For each data point:
    1. Calculate the closest data point
    2. Merge their clusters
3. Repeat 2

Hierarchical Clustering



1. Assign a cluster to each datapoints
2. For each data point:
    1. Calculate the closest data point
    2. Merge their clusters
3. Repeat 2

Hierarchical Clustering

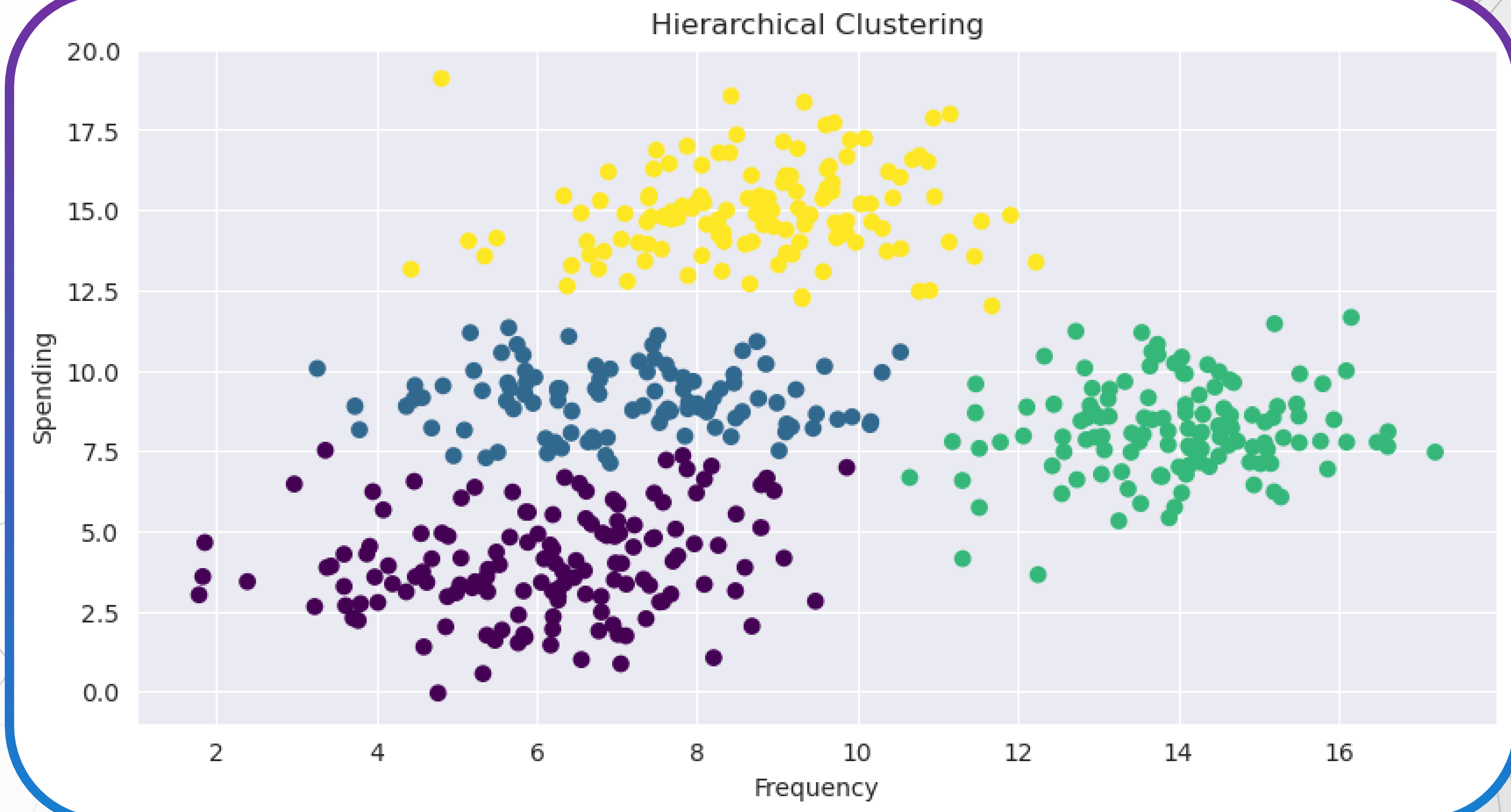


1. Assign a cluster to each datapoints
2. For each data point:
   1. Calculate the closest data point
   2. Merge their clusters

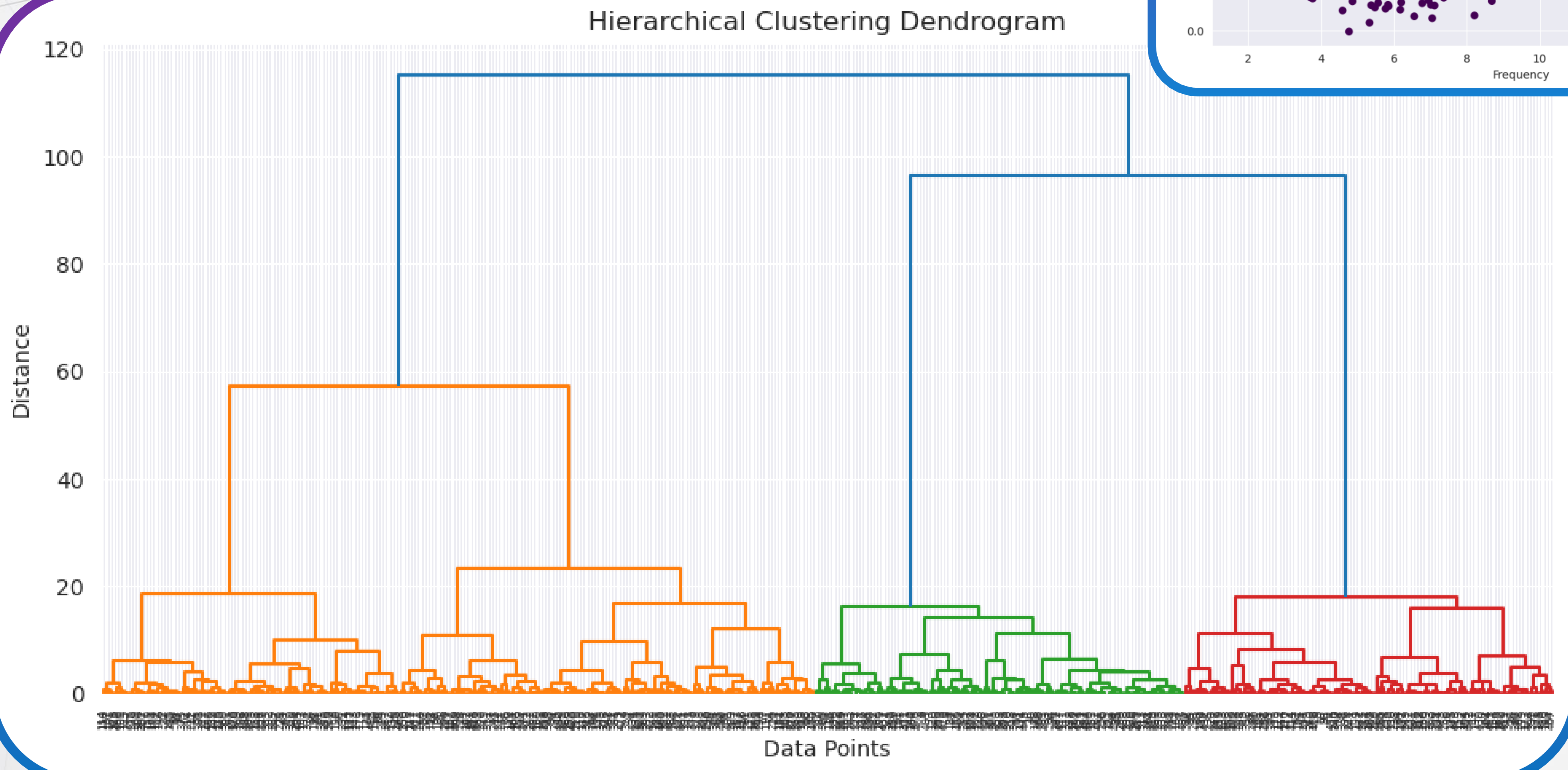
3. Repeat 2

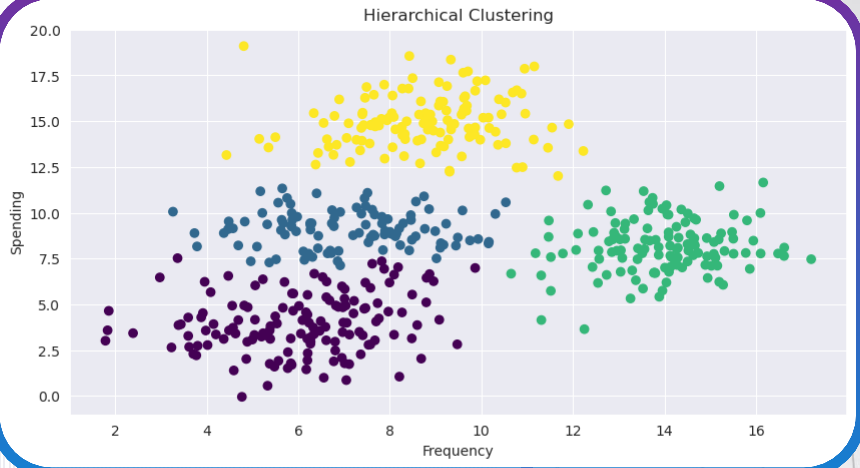
Hierarchical Clustering

Frequency	Spending
1.86	464.60
9.10	1437.92
8.21	914.85
13.13	910.83
8.69	203.74



Hierarchical Clustering





Any Question?