

Classification Methods

November 9th, 2022

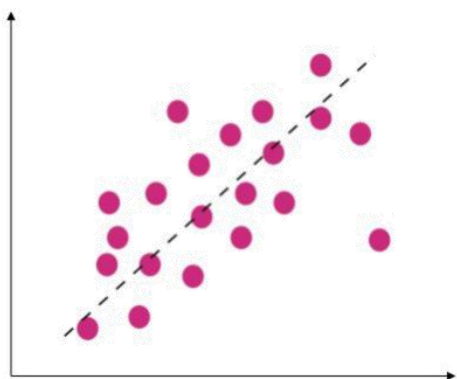


Logistics - Reminder

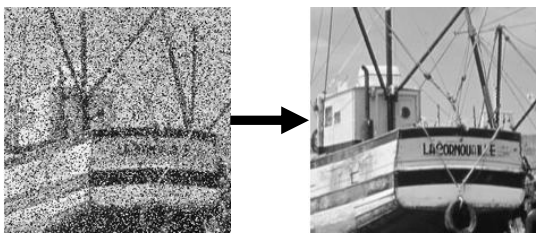
- Everything on course website / Moodle
- 4 credit points
- 2 hrs. lecture, 2 hr. tutorial (Tutorial covers new material)
- 4 homework assignments + Final Project
- All communication through Moodle

Supervised Learning

Regression



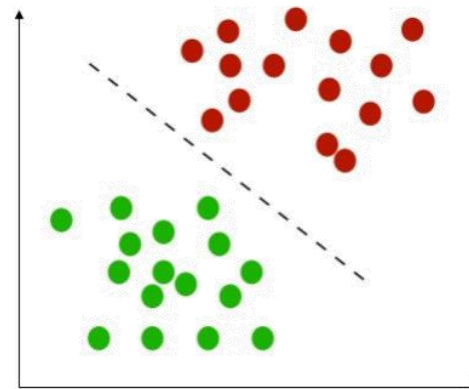
E.g. Image denoising



E.g. Object localization



Classification



E.g. Image classification



Binary Classification

בעקבות המצב:
עד ש50,000 באופן מיידני לכל מטרה
לשכירים/עצמאיים.
באישור משרד האוצר
למחזיקי כרטיס אשראי
לבדיקת זכאות ללא עלות לחצו:
<http://bit.ly/2Zrtplp>

Spam

Not Spam

Binary Classification

- Samples: $x_i \in \mathcal{R}^d, y_i \in \{0,1\}$
 - x_i - text message features, $y_i = \begin{cases} 1, & \text{spam} \\ 0, & \text{not spam} \end{cases}$
- Training Set: $\{x_i, y_i\}_{i=1}^m$
- Hypothesis: $h: \mathcal{R}^d \rightarrow \{0,1\}$, parameterized by θ

Linear Classifier

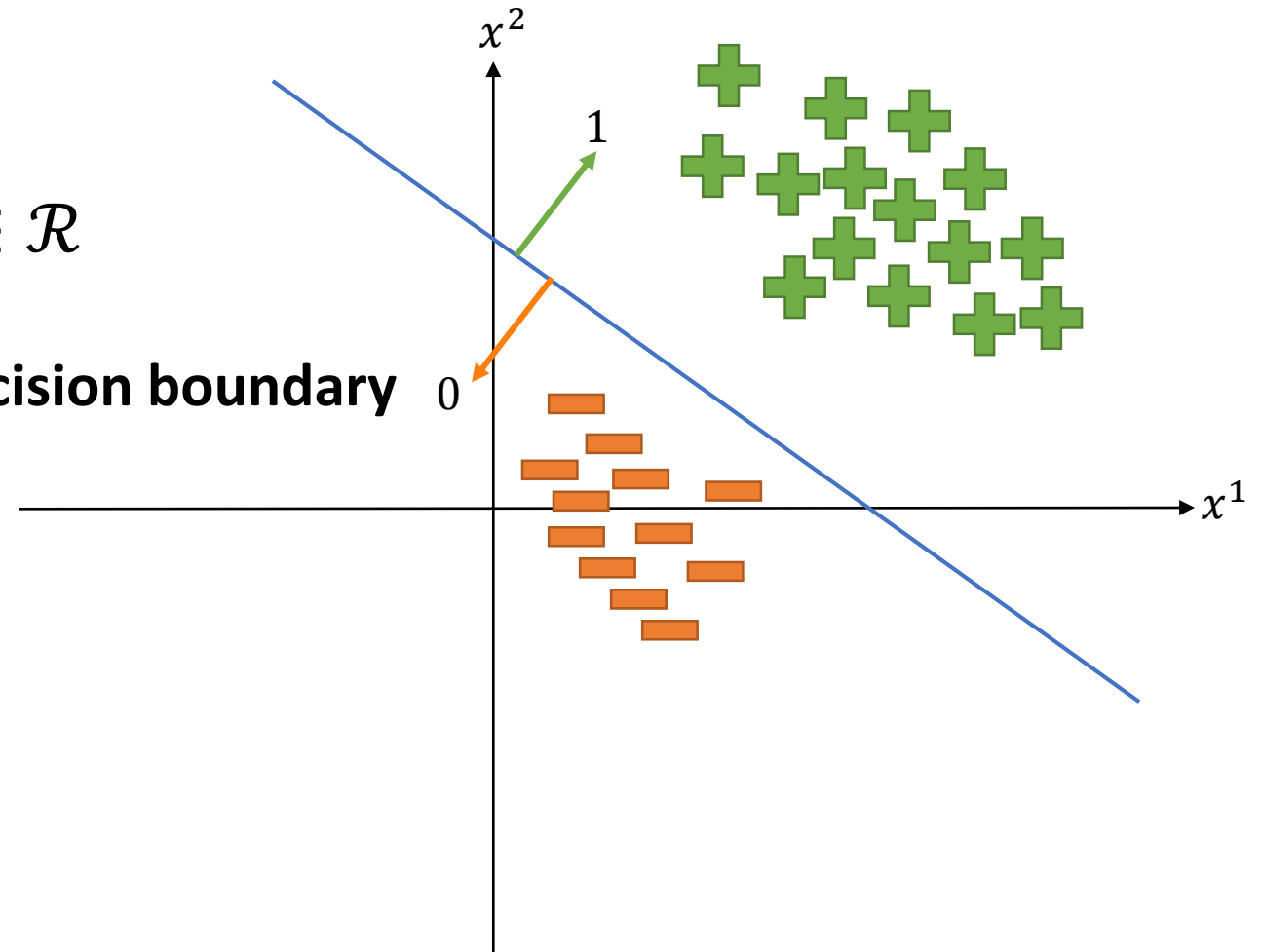
Indicator
function

- $\theta = [W, b], h_{W,b}(x_i) = \mathbb{1}(Wx_i + b > 0)$

- Assume:

$$x_i = \begin{pmatrix} x_i^1 \\ x_i^2 \end{pmatrix} \in \mathcal{R}^2, W \in \mathcal{R}^{1 \times 2}, b \in \mathcal{R}$$

Decision boundary 0



Linear Classifier

- How to find good W, b ?
- How to interpret the results?

$$Wx_1 + b = \mathbf{1000}$$
$$\Rightarrow h_{W,b}(x_1) = 1$$

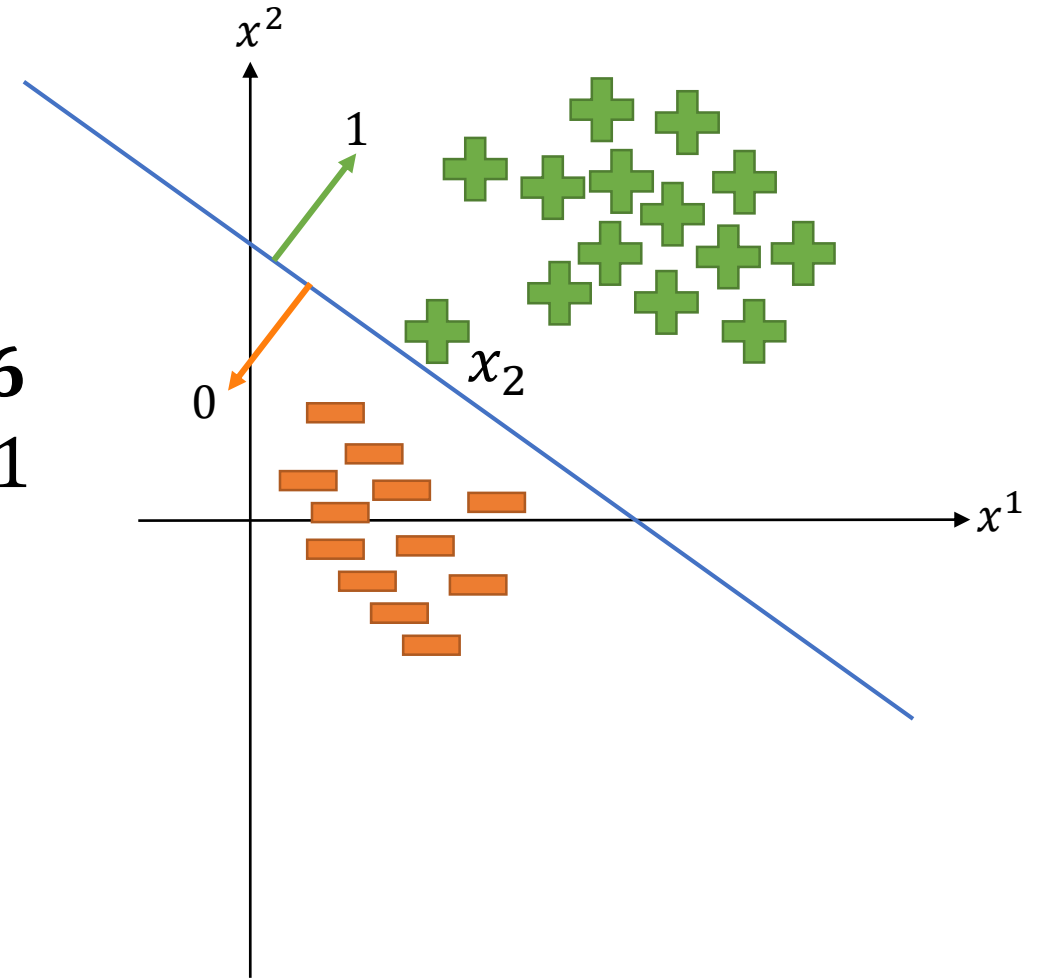
For sure!

$$Wx_2 + b = \mathbf{0.6}$$
$$\Rightarrow h_{W,b}(x_2) = 1$$

Possibly...

- Only intuitive, not measurable

$$h_{\theta}(x_i) = 1(Wx_i + b > 0) \quad \text{+ } x_1$$



Logistic Regression (Classification)

- Interpretable results
- Linear decision boundary
- Easy to find W, b



Sigmoid (Logistic Function)

$Wx + b \rightarrow$ probability

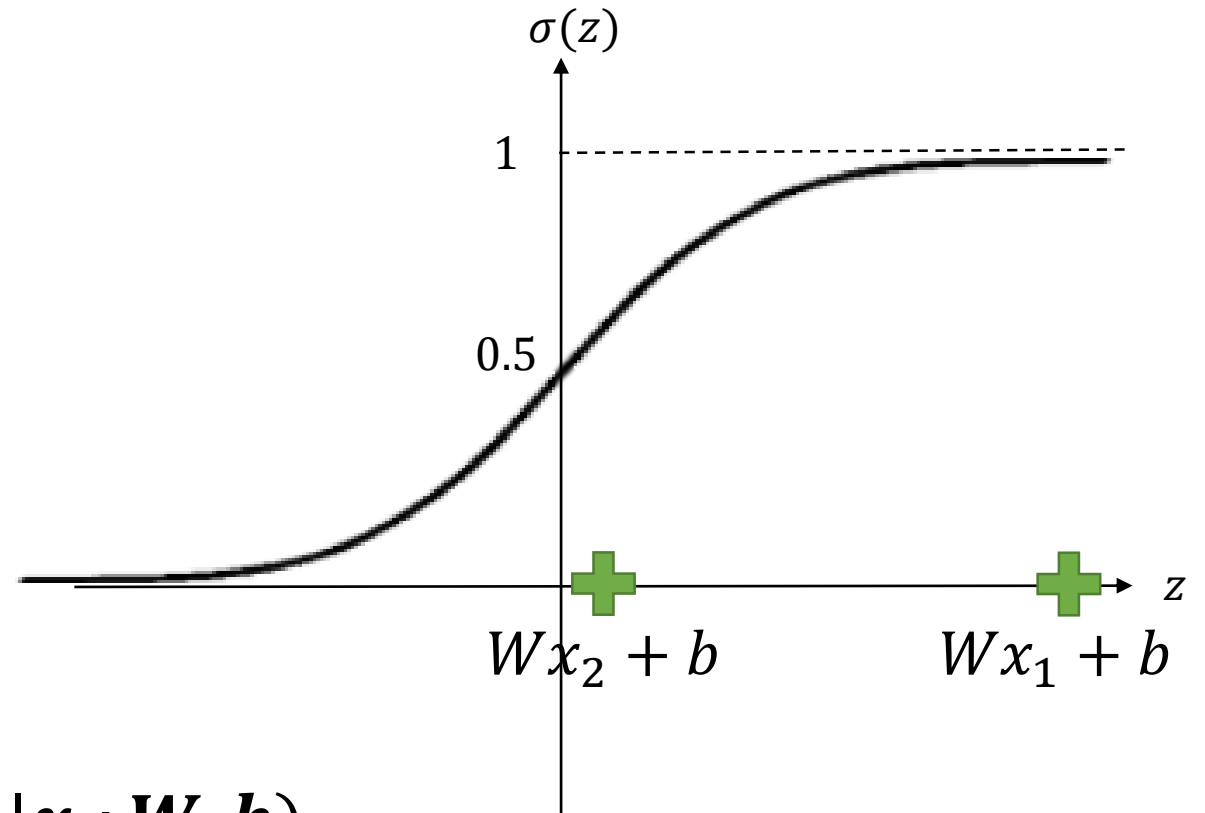
$$z = Wx + b \Rightarrow \sigma(z) = \frac{1}{1 + e^{-z}}$$

Now, our final prediction:

$$h_{W,b}(x_i) = \mathbb{1}(\sigma(z) > 0.5)$$

Interpretation: $\sigma(z) = \Pr(\mathbf{y}_i = \mathbf{1} | \mathbf{x}_i; W, b)$

Note: $1 - \sigma(z) = \Pr(\mathbf{y}_i = \mathbf{0} | \mathbf{x}_i; W, b)$



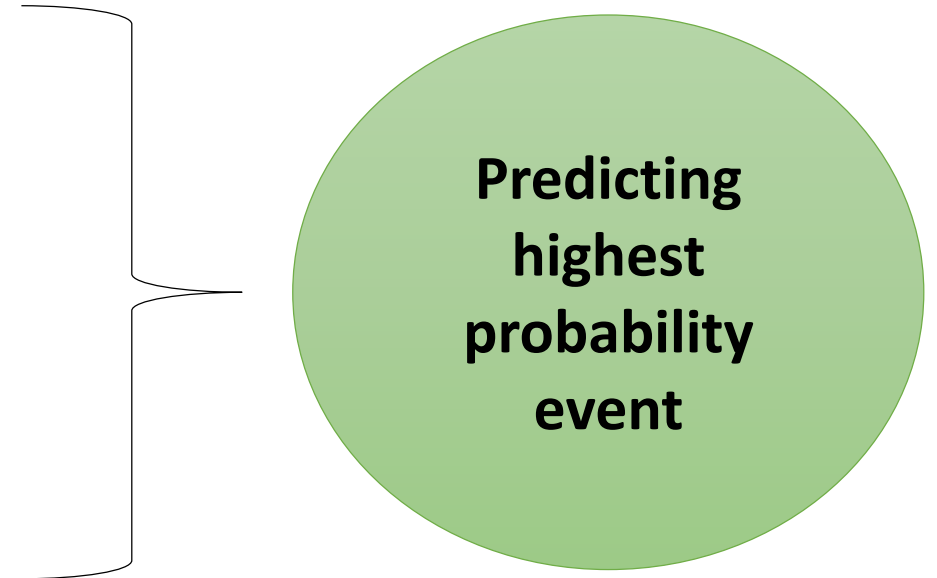
Sigmoid (Logistic Function)

Final prediction:

$$h_{W,b}(x_i) = \mathbb{1}(\sigma(z) > 0.5)$$

$$\sigma(z) = \Pr(y_i = \mathbf{1} | x_i; W, b)$$

$$1 - \sigma(z) = \Pr(y_i = \mathbf{0} | x_i; W, b)$$



Logistic Regression

- Interpretable results
- Linear decision boundary
- Easy to find W, b

Sigmoid (Logistic Function)

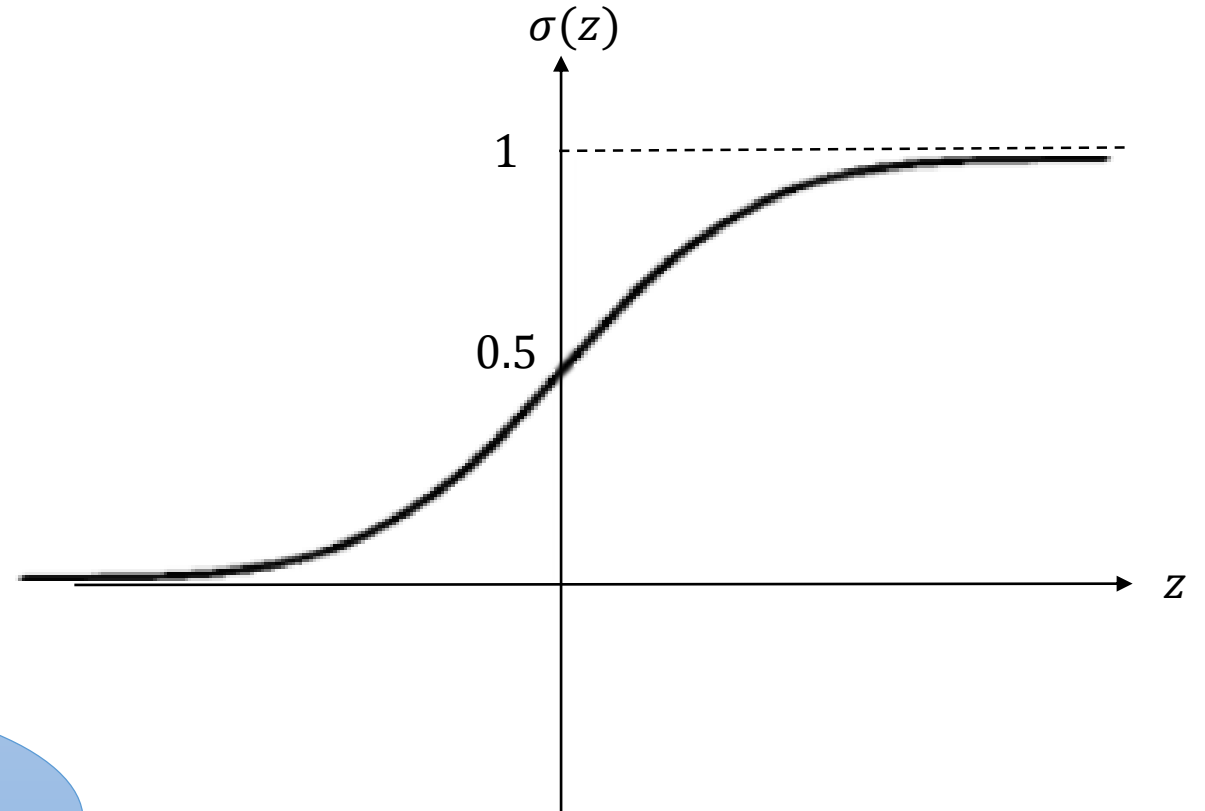
$$h_{W,b}(x_i) = \mathbb{1}(\sigma(z) > 0.5)$$

$$\sigma(z) > 0.5$$

$$\Leftrightarrow z > 0$$

\Rightarrow Linear decision boundary


$$z = Wx + b$$

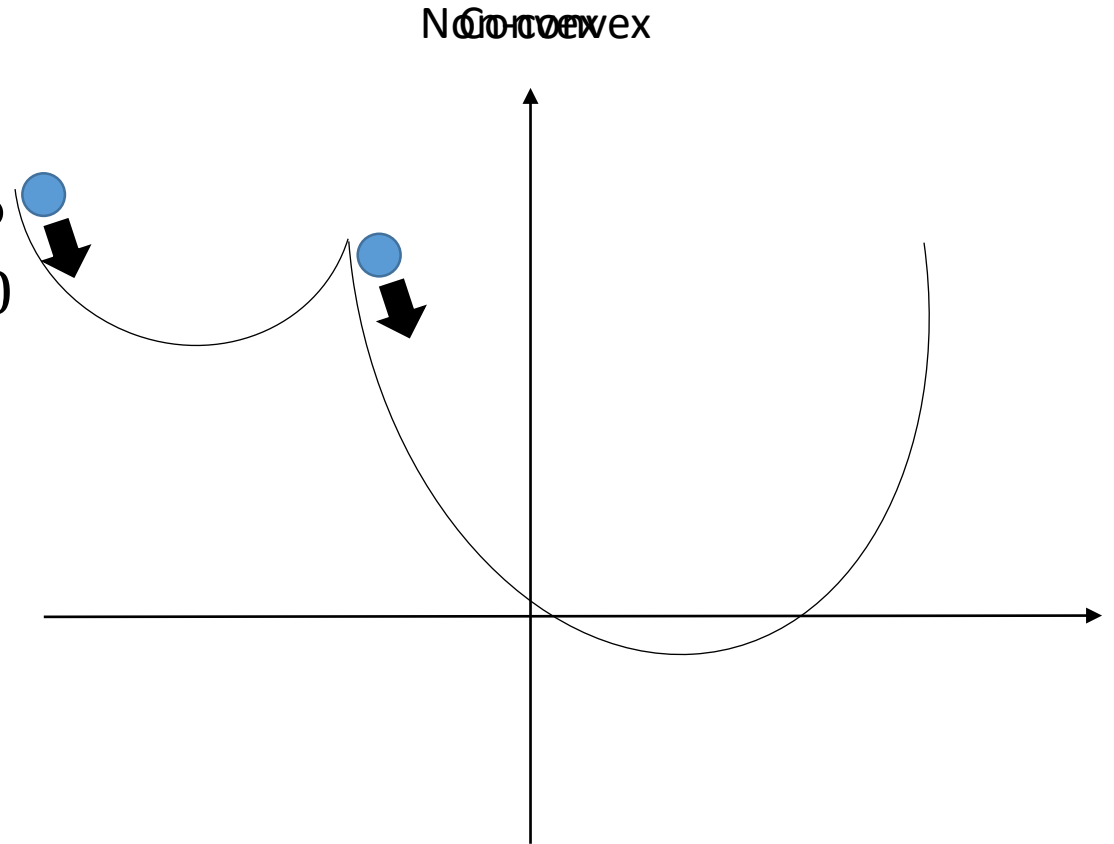


Logistic Regression

- Interpretable results
- Linear decision boundary
- Easy to find W, b

How to find good W, b ?

- Define a **convex** cost function \mathcal{L}
- \mathcal{L} penalizes errors
 - Wrong prediction, large penalty: $\mathcal{L} \rightarrow \infty$
 - Correct prediction, small penalty: $\mathcal{L} \rightarrow 0$
- Use gradient descent to update W, b



Cross-Entropy Loss

$$\Pr[y_i = 1|x_i]$$

- **Prediction:** $\hat{y}_i = \sigma(z) = \sigma(Wx + b)$
- **Label (GT):** y_i

$$\mathcal{L}_i = y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i))$$

- Note that $y_i \in \{0,1\}$, $\hat{y}_i \in (0,1)$
- Is this cost function good?
 - Convex w.r.t W, b ✓
 - Penalizes wrong predictions ✓

Cross-Entropy Loss

$$\mathcal{L}_i = y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i))$$

	$y_i = 0$	$y_i = 1$
$\hat{y}_i \rightarrow 0$	$\mathcal{L}_i \rightarrow 0$	$\mathcal{L}_i \rightarrow \infty$
$\hat{y}_i \rightarrow 1$	$\mathcal{L}_i \rightarrow \infty$	$\mathcal{L}_i \rightarrow 0$

Cross-Entropy Loss - Intuition

- Entropy (information theory) – measures “uncertainty”:
 - Uncertain – all events have the same probability
 - Certain – only one event possible
- Formally: $H(X) = -\sum_{x \in X} p(x) \log p(x)$
 - Expected number of bits to represent an event
- Cross-entropy: have only $q(x)$ - an approximation of $p(x)$
- How many bits are needed now?
- Formally: $H(p, q) = -\sum_{x \in X} p(x) \log q(x)$
 - Minimal when $q = p$

Cross-Entropy Loss - Intuition

$$H(p, q) = - \sum_{x \in X} p(x) \log q(x)$$

- How is it related to us?
- Want to learn W, b , such that:
$$\sigma(Wx_i + b) = \hat{y}_i = q(y_i = 1 | x_i; W, b) \approx p(y_i = 1 | x_i)$$
- Note: $q(y_i = 0 | x_i; W, b) = 1 - q(y_i = 1 | x_i; W, b) = 1 - \hat{y}_i$

Cross-Entropy Loss - Intuition

$$H(p, q) = - \sum_{x \in X} p(x) \log q(x)$$

- During training - **known label**:

$$p(y_i = 1 | x_i) = \begin{cases} 1, & y_i = 1 \\ 0, & y_i = 0 \end{cases}$$

$$\Rightarrow p(y_i = 0 | x_i) = 1 - y_i$$

- Finally:

$$\begin{aligned} H(p, q) &= -p(y_i = 1 | x_i) \cdot \log(\hat{y}_i) - p(y_i = 0 | x_i) \cdot \log(1 - \hat{y}_i) \\ &= y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i)) \\ &= \mathcal{L}_i \end{aligned}$$

How to find good W, b ?

- Define a **convex** cost function \mathcal{L}
- \mathcal{L} penalizes errors
 - Wrong prediction, large penalty: $\mathcal{L} \rightarrow \infty$
 - Correct prediction, small penalty: $\mathcal{L} \rightarrow 0$
- Use gradient descent to update W, b

Gradient Descent – Single Sample

$$x_i \in \mathcal{R}^{d \times 1}, y_i \in \{0,1\}$$

$$\hat{y}_i = \sigma(Wx_i + b) \quad (W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R} \Rightarrow \hat{y}_i \in (0,1))$$

$$\mathcal{L}_i = y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i))$$

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot x_i^T (\hat{y}_i - y_i)$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial b} = b - \alpha \cdot (\hat{y}_i - y_i)$$

Find prediction

Calculate Loss

Update weights

Update bias

Stochastic Mini-Batch Gradient Descent

Repeat until convergence:

Sample a mini batch $\{x_i, y_i\}_{i=1}^m$:

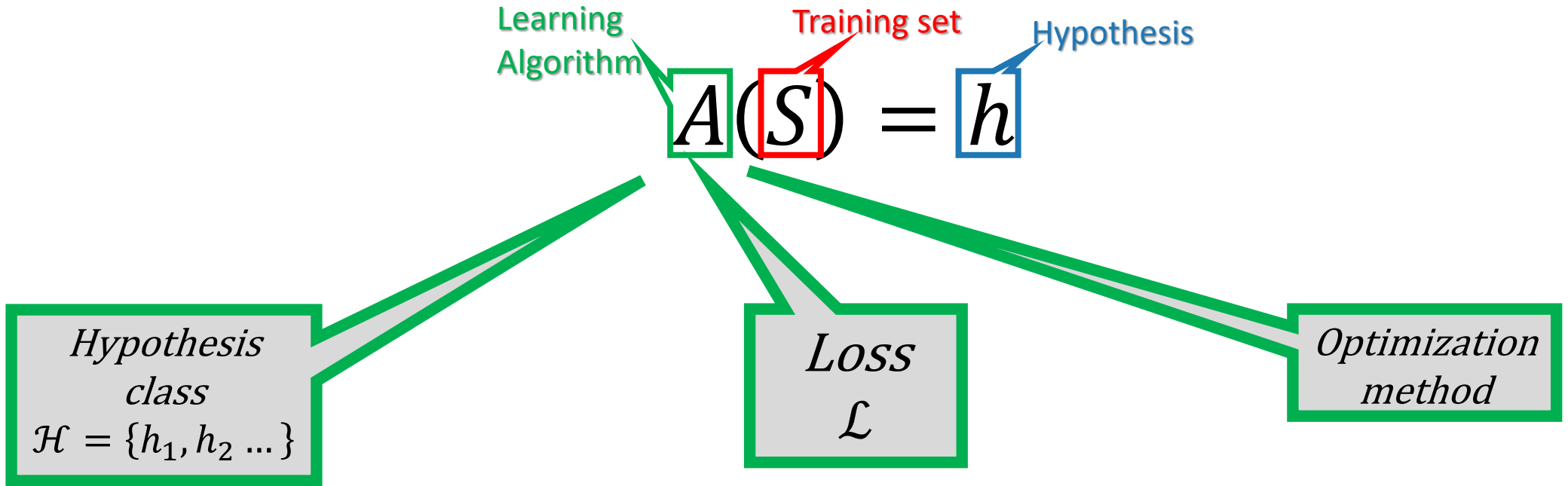
Calculate \mathcal{L}_i for each pair

$$\mathcal{L} = \frac{1}{m} \sum_i \mathcal{L}_i$$

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}}{\partial W}$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}}{\partial b}$$

Supervised Learning – Logistic Regression

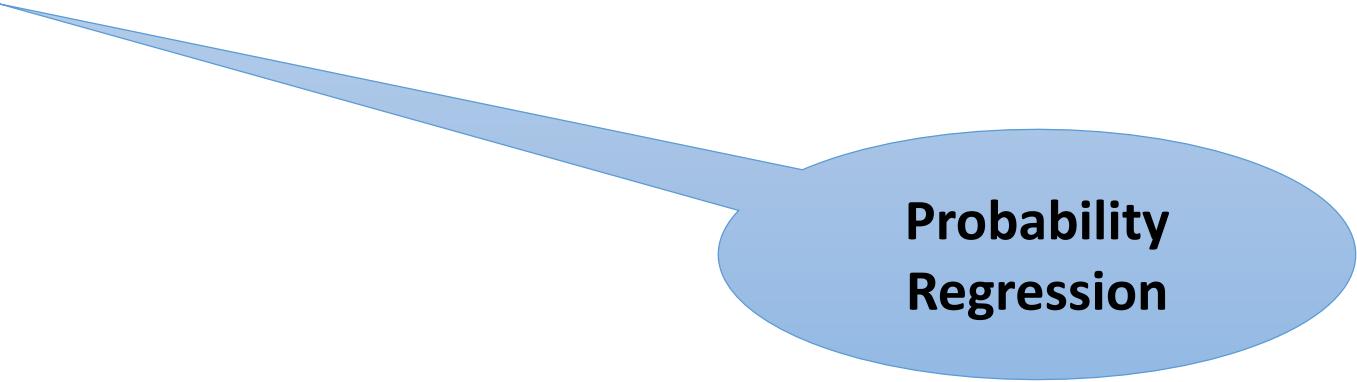


$$\mathcal{H} = \{h_{W,b}(x_i) = 1(\sigma(Wx_i + b) > 0.5)\}$$

Cross-entropy loss

SGD

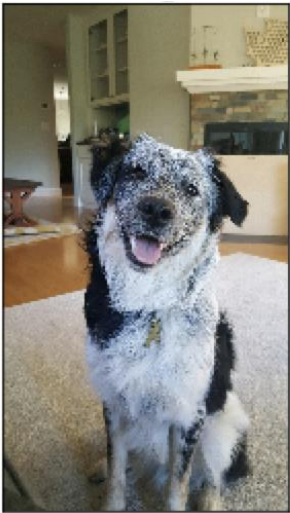
Logistic Regression - Summary



Probability
Regression

- **Binary** classification
- **Linear** ($Wx + b$)
- **Sigmoid** for interpretable (probability) output
- **Predict** highest probability event
- **Learning:**
 - **Cross-entropy** loss (convex, penalizes mistakes)
 - **Gradient descent** to update weights and bias

Multi Class Classification



Multi Class Classification

בעקבות המצב:
עד ש50,000 באופן מיידני לכל מטרה
לשכירים/עצמאיים.
באישור משרד האוצר
למחזיקי כרטיס אשראי
לבדיקת זכאות ללא עלות לחצו:
<http://bit.ly/2Zrtplp>

Spam

Maybe

Not Spam

Multi Class Classification

- C classes
- Samples: $x_i \in \mathcal{R}^d, y_i \in [C]$
 - x_i - text message features, $y_i = \begin{cases} 2, & \text{maybe} \\ 1, & \text{spam} \\ 0, & \text{not spam} \end{cases}$
- Training Set: $\{x_i, y_i\}_{i=1}^m$
- Hypothesis: $h: \mathcal{R}^d \rightarrow [C]$, parameterized by θ

Softmax Classifier

- Generalizing Logistic Regression, **binary** → **multi-class**
- Begin with **Linear Transformation** ($z = Wx + b$)

Similar Learning:
1. Cross-entropy loss
2. SGD

Logistic Regression	Softmax classifier
$W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R}$	$W \in \mathcal{R}^{C \times d}, b \in \mathcal{R}^C$

- Apply **non-linearity** to transform z to \hat{y} (probability)

Sigmoid (Logistic function)	Softmax
-----------------------------	---------

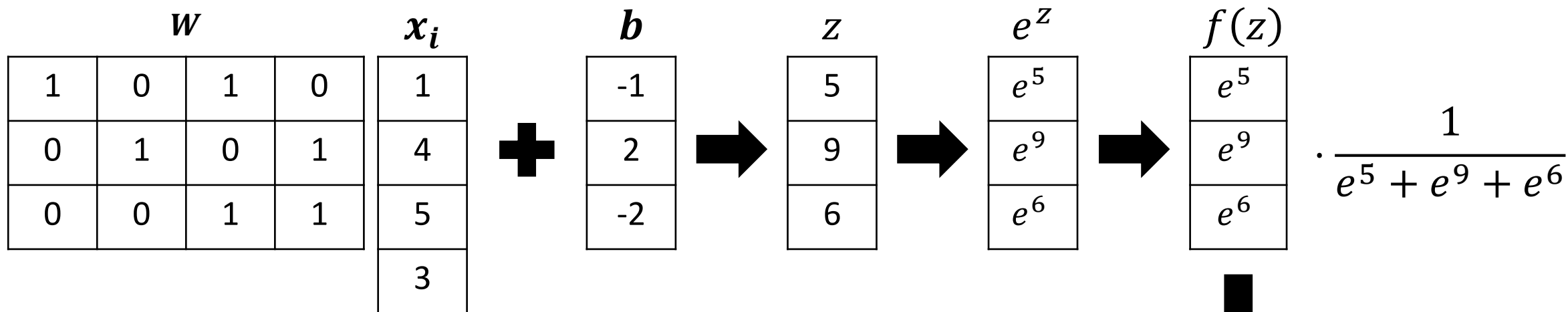
C dimensional z
 $z^j = j$ 'th class score

- Final prediction – pick the event with **highest probability**

$1(\hat{y} > 0.5)$	$\operatorname{argmax}(\hat{y})$
--------------------	----------------------------------

Softmax Function - Example

Given a text x_i : **Not Spam (0)**, **Spam(1)** or **Maybe (2)**



\hat{y}_i - Probability distribution

$\hat{y}_i [j]$ - Predicted probability that the class is j

↓

\hat{y}_i	
0.02	2% Not Spam
0.93	93% Spam!
0.05	5% Maybe

Softmax Function - Formally

Converting z to probability distribution over the classes \mathcal{C} :

$$z = Wx + b, z \in \mathcal{R}^{\mathcal{C}}$$

$$\Rightarrow f(z)^i = \frac{e^{z^i}}{\sum_{j \in \mathcal{C}} e^{z^j}}$$

$$\Rightarrow \hat{y} = [f(z)^0, \dots, f(z)^{\mathcal{C}-1}], \sum_{j \in \mathcal{C}} \hat{y}^j = 1$$

Softmax Classifier

- Generalizing Logistic Regression, **binary** → **multi-class**
- Begin with **Linear Transformation** ($z = Wx + b$):

Logistic Regression	Softmax classifier
$W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R}$	$W \in \mathcal{R}^{C \times d}, b \in \mathcal{R}^C$

- Apply **non-linearity** to transform z to \hat{y} (probability)

Sigmoid (Logistic function)	Softmax
-----------------------------	---------

- Final prediction – the event with **highest probability**

$1(\hat{y} > 0.5)$	$\operatorname{argmax}(\hat{y})$
--------------------	----------------------------------

Similar Learning:
1. Cross-entropy loss
2. SGD

Softmax Classifier

- Generalizing Logistic Regression, **binary** → **multi-class**
- Begin with **Linear Transformation** ($z = Wx + b$):

Logistic Regression	Softmax classifier
$W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R}$	$W \in \mathcal{R}^{C \times d}, b \in \mathcal{R}^C$

- Apply **non-linearity** to transform z to \hat{y} (probability)

Sigmoid (Logistic function)	Softmax
-----------------------------	---------

- Final prediction – the event with **highest probability**

$1(\hat{y} > 0.5)$	$\operatorname{argmax}(\hat{y})$
--------------------	----------------------------------

Similar Learning:
1. Cross-entropy loss
2. SGD

Cross-Entropy Loss – Softmax Classifier

$$H(p, q) = - \sum_{x \in X} p(x) \log q(x)$$

$p(y_i = c|x)$ – Ground Truth (known), $\hat{y}_i^c = q(y_i = c|x)$ – Our Prediction

$p(y_i = c x_i)$		\hat{y}_i	
0	Spam	0.02	2% Spam
1	Not spam!	0.93	93% Not spam!
0	Maybe	0.05	5% Maybe

$$H(p, q) = - \sum_{c \in \{0,1,2\}} p(y_i = c|x_i) \cdot \log(\hat{y}_i^c) = - \log(\hat{y}_i^1)$$

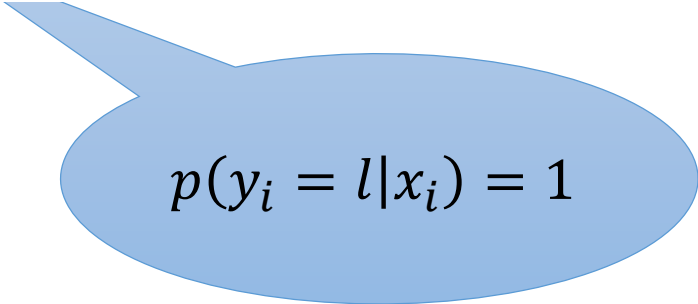
Goal: Prediction should be closer to GT.

Cross-Entropy Loss – Softmax Classifier

- Generally, with C classes, training example $\{x_i, l\}$

$$\mathcal{L}_i = H(p, q) = - \sum_{c \in C} p(y_i = c | x_i) \cdot \log(\hat{y}_i^c)$$

$$= -\log \frac{e^{z^l}}{\sum_{j \in [C]} e^{z[j]}}$$


$$p(y_i = l | x_i) = 1$$

- Wish to make the loss **smaller** (log argument larger):
 - **Increase numerator** → higher confidence of class l
 - **Decrease denominator** → higher confidence it's not other classes!

Softmax Classifier

- Generalizing Logistic Regression, **binary** → **multi-class**
- Begin with **Linear Transformation** ($z = Wx + b$):

Logistic Regression	Softmax classifier
$W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R}$	$W \in \mathcal{R}^{C \times d}, b \in \mathcal{R}^C$

- Apply **non-linearity** to transform z to \hat{y} (probability)

Sigmoid (Logistic function)	Softmax
-----------------------------	---------

- Final prediction – the event with **highest probability**

$1(\hat{y} > 0.5)$	$\operatorname{argmax}(\hat{y})$
--------------------	----------------------------------

Similar Learning:
1. Cross-entropy loss
2. SGD

Gradient Descent – Logistic Regression

$$x_i \in \mathcal{R}^{d \times 1}, y_i \in \{0,1\}$$

$$\hat{y}_i = \sigma(Wx_i + b) \quad (W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R} \Rightarrow \hat{y}_i \in [0,1])$$

$$\mathcal{L}_i = y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i))$$

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot x_i^T (\hat{y}_i - y_i)$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial b} = b - \alpha \cdot (\hat{y}_i - y_i)$$

Find prediction

Calculate Loss

Update weight

Update bias

Gradient Descent – Softmax Classifier

$$x_i \in \mathcal{R}^{d \times 1}, y_i \in \{\mathbf{0}, \mathbf{1}, \dots, \mathbf{C} - \mathbf{1}\}$$

$$\hat{y}_i = \sigma(Wx_i + b) \quad (W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R} \Rightarrow \hat{y}_i \in [0,1])$$

$$\mathcal{L}_i = y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i))$$

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot x_i^T (\hat{y}_i - y_i)$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial b} = b - \alpha \cdot (\hat{y}_i - y_i)$$

Find prediction

Calculate Loss

Update weight

Update bias

Gradient Descent – Softmax Classifier

$$x_i \in \mathcal{R}^{d \times 1}, y_i \in \{0, 1, \dots, C - 1\}$$

$$\hat{y}_i = f(Wx_i + b) \quad (W \in \mathcal{R}^{C \times d} \quad b \in \mathcal{R}^C, \sum_{j \in C} \hat{y}_i[j] = 1)$$

Find prediction

$$\mathcal{L}_i = y_i \cdot (-\log \hat{y}_i) + (1 - y_i) \cdot (-\log(1 - \hat{y}_i))$$

Calculate Loss

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot x_i^T (\hat{y}_i - y_i)$$

Update weight

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial b} = b - \alpha \cdot (\hat{y}_i - y_i)$$

Update bias

Gradient Descent – Softmax Classifier

$$x_i \in \mathcal{R}^{d \times 1}, y_i \in \{0, 1, \dots, C - 1\}$$

$$\hat{y}_i = f(Wx_i + b) \quad (W \in \mathcal{R}^{C \times d} \quad b \in \mathcal{R}^C, \sum_{j \in C} \hat{y}_i[j] = 1)$$

$$\mathcal{L}_i = - \sum_{c \in C} p(y_i = c | x_i) \cdot \log(f(z)[c])$$

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot x_i^T (\hat{y}_i - y_i)$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial b} = b - \alpha \cdot (\hat{y}_i - y_i)$$

Find prediction

Calculate Loss

Update weight

Update bias

Gradient Descent – Softmax Classifier

$$x_i \in \mathcal{R}^{d \times 1}, y_i \in \{0, 1, \dots, C - 1\}$$

$$\hat{y}_i = f(Wx_i + b) \quad (W \in \mathcal{R}^{C \times d}, b \in \mathcal{R}^C)$$

$$\delta_{y_i}[c] = \begin{cases} 1, & c = y_i \\ 0, & \text{o.w.} \end{cases}$$

$$\mathcal{L}_i = -\sum_{c \in C} p(y_i = c | x_i) \cdot \log(f(z)_c)$$

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot (\hat{y}_i - \delta_{y_i}) x_i^T$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial b} = b - \alpha \cdot (\hat{y}_i - \delta_{y_i})$$

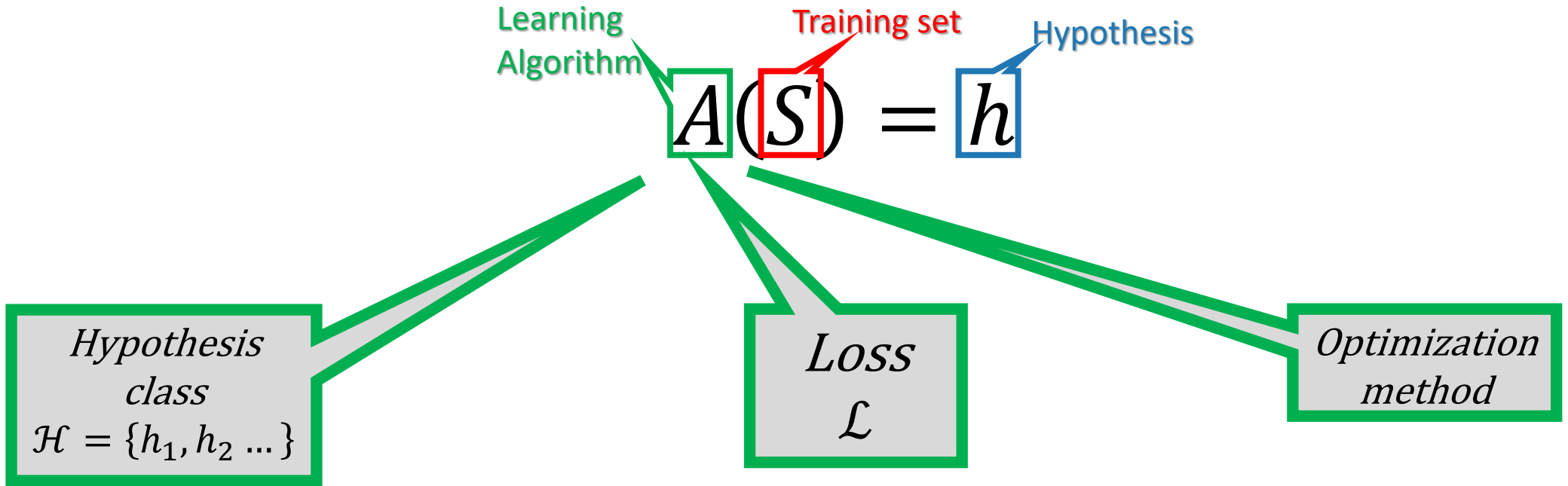
Find prediction

Calculate Loss

Update weight

Update bias

Supervised Learning – Softmax Classifier



$$\mathcal{H} = \{h_{W,b}(x_i) = \operatorname{argmax}(f(Wx_i + b))\}$$

Cross-entropy loss

SGD

Conclusion

- Begin with **Linear Transformation** ($z = Wx + b$):

Logistic Regression	Softmax classifier
$W \in \mathcal{R}^{1 \times d}, b \in \mathcal{R}$	$W \in \mathcal{R}^{c \times d}, b \in \mathcal{R}^c$

- Apply **non-linearity** to transform z to \hat{y} (probability)

Sigmoid (Logistic function)	Softmax
-----------------------------	---------

- Final prediction – the event with **highest probability**

$1(\hat{y} > 0.5)$	$\operatorname{argmax}(\hat{y})$
--------------------	----------------------------------

Similar Learning:
1. Cross-entropy loss
2. SGD

Pracitcal Considerations

- **Mini Batches**
- **Numerical Stability**

Stochastic Mini-Batch Gradient Descent

Repeat until convergence:

Sample a mini batch $\{x_i, y_i\}_{i=1}^m$:

Calculate \mathcal{L}_i for each pair

$$\mathcal{L} = \frac{1}{m} \sum_i \mathcal{L}_i$$

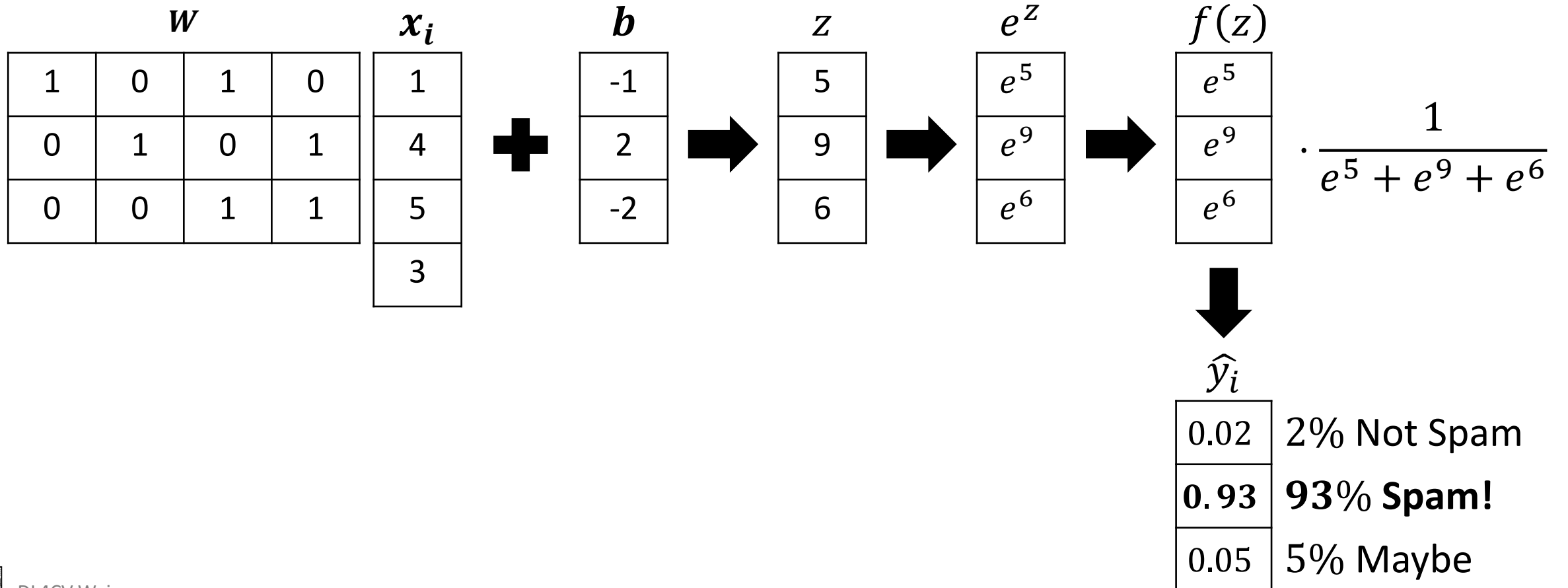
$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}}{\partial W}$$

$$b \leftarrow b - \alpha \cdot \frac{\partial \mathcal{L}}{\partial b}$$

**Not efficient
sequentially**

Softmax Function - Reminder

Given a text x_i : **Not Spam (0)**, **Spam(1)** or **Maybe (2)**

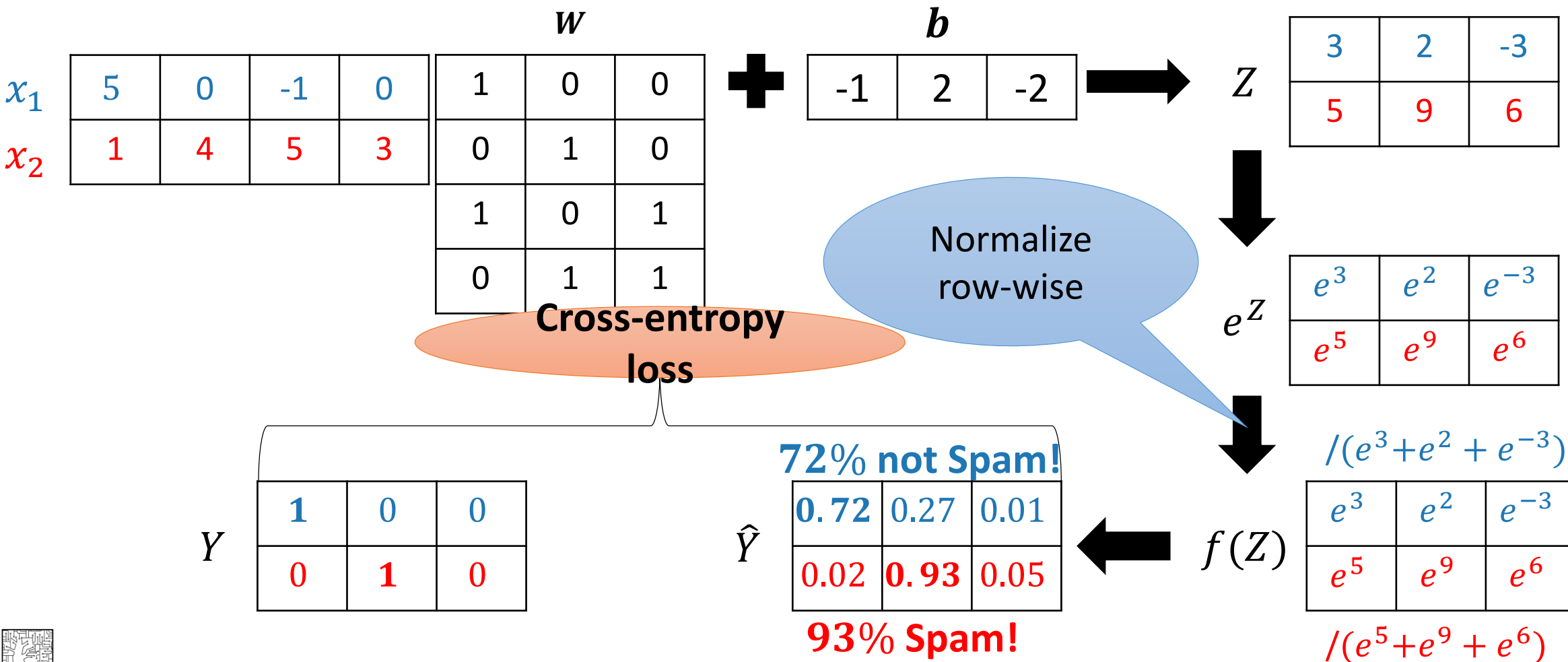


Softmax Classifier – Batched Example

Given two pairs: $\{x_1, 0\}$, $\{x_2, 1\}$

Not spam

spam



Mini Batches - Formally

Define $X = [x_1, \dots, x_m]^T \in \mathcal{R}^{m \times d}$ - samples matrix

$Y \in \mathcal{R}^{m \times C}$ - GT labels matrix (Each row in Y is one-hot vector)

Reminder: $W \in \mathcal{R}^{C \times d}, b \in \mathcal{R}^C$

$$Z = \underbrace{XW^T}_{\in \mathcal{R}^{m \times c}} + b$$

(b is added to each row)

Mini Batches - Formally

$$Z = \underbrace{XW^T}_{\in \mathcal{R}^{m \times c}} + b, Z \in \mathcal{R}^{m \times c}$$

Softmax & Cross-entropy applied row-wise

$$\hat{Y} = f(Z)$$

$$\mathcal{L} = \frac{1}{m} \sum_i \mathcal{L}_i$$

Reminder: Weights update (single-sample):

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}_i}{\partial W} = W - \alpha \cdot (\hat{y}_i - \delta_{y_i}) x_i^T$$

Weights update (mini-batch):

$$W \leftarrow W - \alpha \cdot \frac{\partial \mathcal{L}}{\partial W} = W - \frac{\alpha}{m} \cdot (\hat{Y} - Y)^T X$$

Similarly
derive cross-
entropy rule

Numerical Stability

- Softmax: e^z can be extremely large

$$f(z)^i = \frac{e^{z^i}}{\sum_{j \in [C]} e^{z^j}}$$

- Cross-entropy: \hat{y} possibly close to zero $\Rightarrow \log(\hat{y}_i) \rightarrow -\infty$

- Many possible solutions

$$H(p, q) = - \sum_{c \in [C]} p(y_i = c | x_i) \cdot \log(\hat{y}_i^c)$$



imgflip.com

