CS5242 Neural Networks and Deep Learning

Lecture 02: Shallow Networks

Wei WANG

cs5242@comp.nus.edu.sg

NUS National University of Singapore | School of Computing

change log:

- slide 32, loss function of the vectorized multi-label multi-classification model
- removed the overfitting part

Agenda

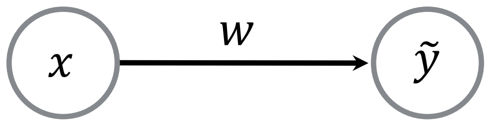
- Recap of last lecture
- Classification
  - Logistic regression
    - binary cross-entropy
  - Multinomial regression (Softmax regression)
- Overfitting and underfitting

Recap

- Linear regression
  - Univariate: single feature $x \in R$
  - Multivariate: multiple features $\boldsymbol{x} \in R^m$
  - Linear transformation: $\tilde{y} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}$
  - Loss: measure the difference between the prediction and ground truth
  - Training is to optimize (i.e., minimize) the loss w.r.t parameters (**w**)
- Gradient descent algorithm
  - Minimize the target loss iteratively; for each iteration,
  - Compute the gradient of the average loss (over all training examples) w.r.t **w**
  - Update **w** in the **opposite** of the gradient direction

$$\boldsymbol{w} = \boldsymbol{w} - \alpha \frac{\partial J}{\partial \boldsymbol{w}}$$

The simplest case



- Univariate linear regression without bias
  - $\tilde{y} = wx$

- Squared error as the loss function
  - $L(x, y|w) = \frac{1}{2}||\tilde{y} - y||^2 = \frac{1}{2}(\tilde{y} - y)^2 = \frac{1}{2}(wx - y)^2$

- Training objective
  - tune w to minimize the average loss over the training dataset
  - $J(w) = \frac{1}{m}\sum_{i=1}^{m} L\left(x^{(i)}, y^{(i)}\middle|w\right) = \frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} - y^{(i)}\right)^2$

Slide Credit: K. Kitani

Gradient descent algorithm

Randomly initialize w

Repeat

J = 0

For each sample $x^{(i)}, y^{(i)}$ (i=1, 2, … m)

a. Forward pass $\tilde{y}^{(i)} = wx^{(i)}$

b. Accumulate Loss $J += \frac{1}{m} L^{(i)} = \frac{1}{2m} \left|\left|\tilde{y}^{(i)} - y^{(i)}\right|\right|^2 = \frac{1}{2m}\left(wx^{(i)} - y^{(i)}\right)^2$

c. Accumulate gradient $\frac{\partial J}{\partial w} += \frac{1}{m}\frac{\partial L^{(i)}}{\partial w} = \frac{1}{2m}\left(wx^{(i)} - y^{(i)}\right)x^{(i)}$

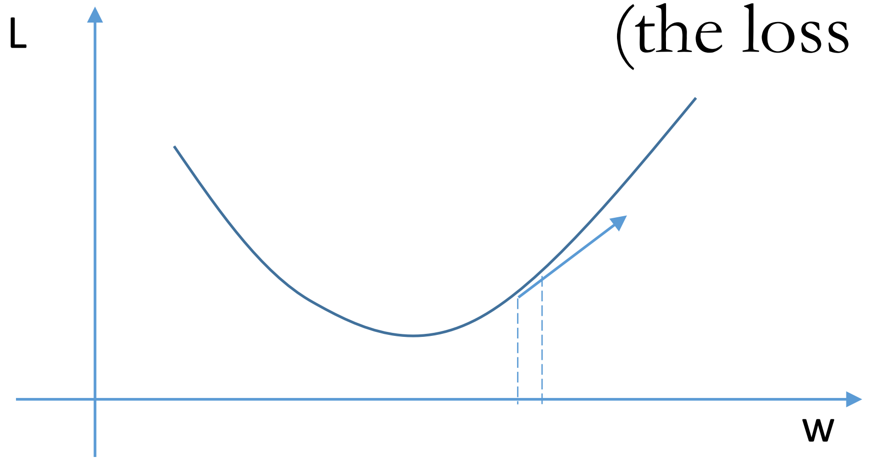
Update $w = w - \alpha \frac{\partial J}{\partial w}$ move in the opposite direction of gradient

Understanding Derivatives

$$\frac{d\mathcal{L}}{dw}$$ …is the rate at which **this** will change…

$$\mathcal{L} = \frac{1}{2}(y - \hat{y})^2$$

(the loss function)



… per unit change of **this**

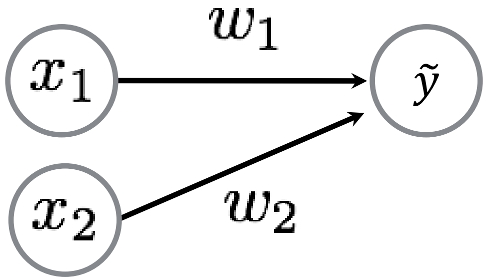
$$y = wx$$

(the weight parameter)

Slide Credit: K. Kitani

The second simplest case

- Multivariate linear regression with two features
  - the bias term is skipped
  - $\tilde{y} = \boldsymbol{w^T x} = \mathrm{w_1 x_1 + w_2 x_2}$



Gradient descent algorithm

```
Random initialize w1, w2
Repeat
      J = 0
      For each sample
```
$\boldsymbol{x}^{(i)}, y^{(i)}$

a. Forward pass

b. Accumulate Loss

c. Accumulate partial gradient (derivative) $\frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}$

what are ?

Update

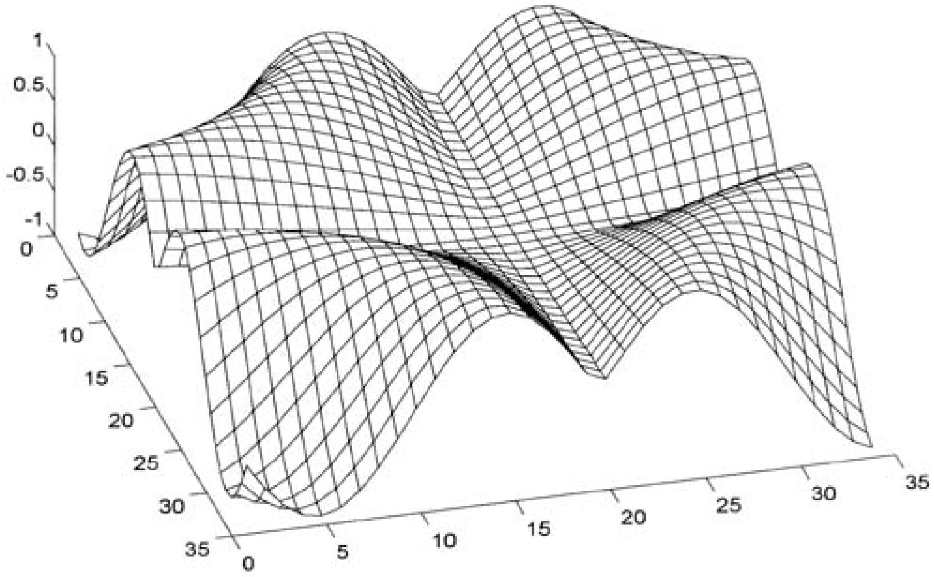
$$w_1 = w_1 - \alpha \frac{\partial J}{\partial w_1}$$

$$w_2 = w_2 - \alpha \frac{\partial J}{\partial w_2}$$

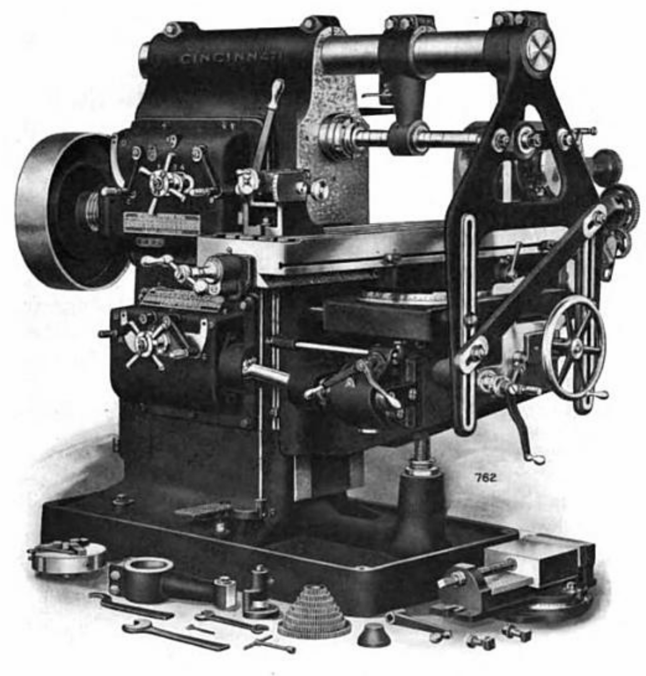
move in opposite direction of partial derivatives

(Partial) Derivatives

Two ways to think about them:



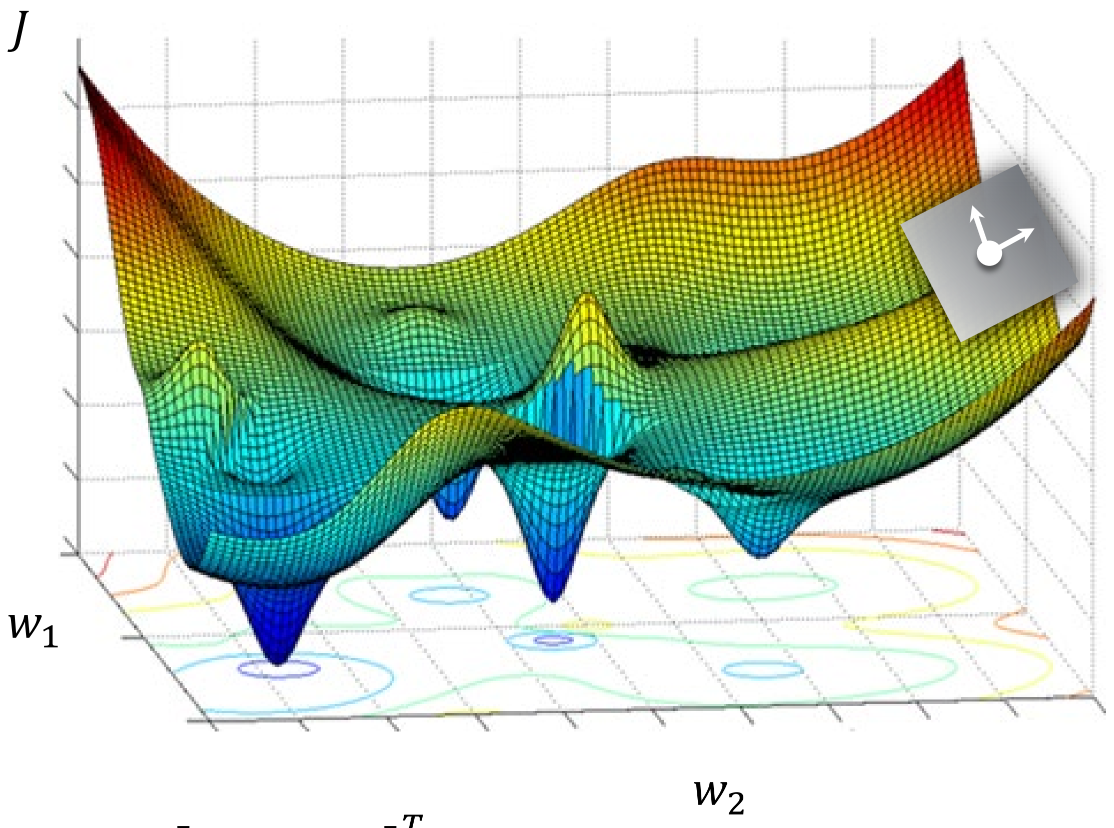
Slope of a function



Knobs on a machine

Slide Credit: K. Kitani

Slope of a function



$$\frac{\partial J}{\partial \boldsymbol{w}} = \left[\frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}\right]^T$$

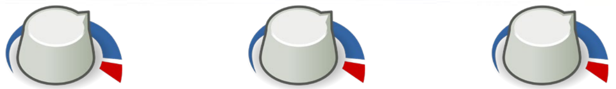
describes the slope around a point

Slide Credit: K. Kitani

Knobs on a machine

input
$\boldsymbol{x}$

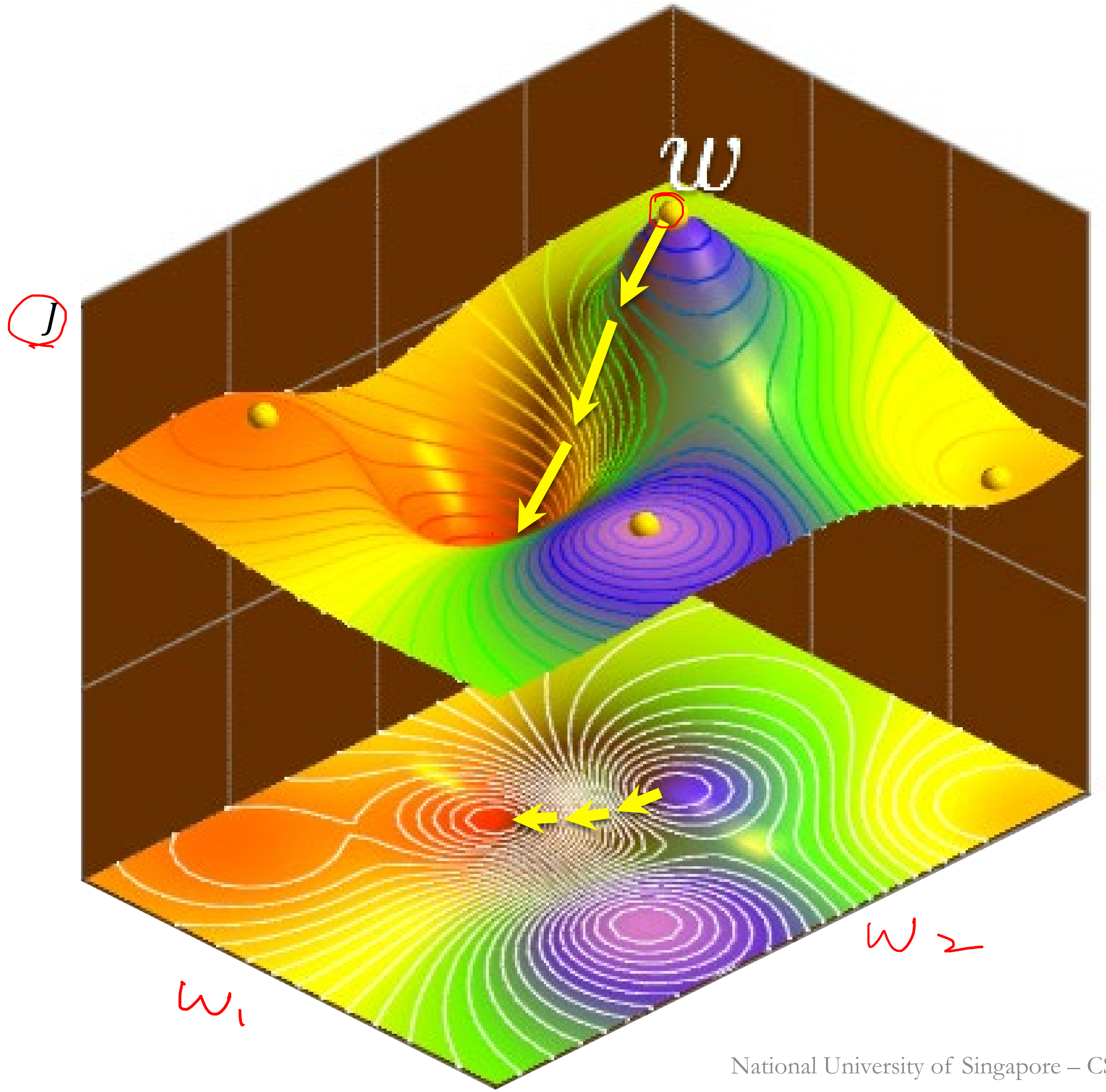
output
$f(\boldsymbol{x};\boldsymbol{w})$



describes how each 'knob' affects the output $\quad \dfrac{\partial f(x)}{\partial w_1} \quad \dfrac{\partial f(x)}{\partial w_2} \quad \dfrac{\partial f(x)}{\partial w_3}$

small change in parameter Δw_1 → output will change by $\dfrac{\partial f(x)}{\partial w_1}\Delta w_1$

Slide Credit: K. Kitani



Gradient Descent:

given a fixed-point on a function, move in the direction opposite of the gradient.

$$w_i = w_i - \alpha \frac{\partial J}{\partial w_i}$$

Slide Credit: K. Kitani

Recap

- Vectorization & denominator layout
- Let $\Delta/\blacksquare/\nabla$ be a scalar, vector or matrix.
- $\frac{\partial \Delta}{\partial \blacksquare}$
  - If Δ is scalar, then the result shape is the same as $\blacksquare$
  - If $\blacksquare$ is scalar, then the result shape is the same as Δ^T
  - If both are vectors, then the result is a matrix with rows decided by $\blacksquare$ and columns by Δ
- $\Delta = \mathrm{g}(\blacksquare), \blacksquare = u(\nabla)$
  - $\frac{\partial \Delta}{\partial \nabla}$ is the multiplication between $g'(\quad) and\ u'(\quad)$, but
  - Need to arrange the order and transpose to make sure the result's shape matches $\frac{\partial \Delta}{\partial \nabla}$

Gradient table

(denominator layout)

Shape of $\frac{\partial y}{\partial x}$	Scalar y	Vector **y** (m ,1)	Matrix **Y** (m, n)
Scalar x	1	(1, m)	(n, m)
Vector **x** (n ,1)	(n, 1)	(n, m)	
Matrix **X** (p, q)	(p, q)		

Shape check:
For denominator layout, the shape of the gradient of a scalar w.r.t a variable v is the same as the shape of v, where v could be a scalar, vector, or matrix

Gradient table: vector by vector

(denominator layout)

$\boldsymbol{y} =$	$\boldsymbol{a}$	$\boldsymbol{x}$	$\boldsymbol{A}\boldsymbol{x}$	$\boldsymbol{x}^T\boldsymbol{A}$
$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} =$	$\boldsymbol{0}$	I	$\boldsymbol{A}^T$	$\boldsymbol{A}$

Gradient table: vector by vector (denominator layout)

$\boldsymbol{y} =$	$a\boldsymbol{u}$ $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$	$v\boldsymbol{u}$ $v = v(\boldsymbol{x}), \boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$	$\boldsymbol{v} + \boldsymbol{u}$ $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$	$\boldsymbol{A}\boldsymbol{u}$ $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$	$g(\boldsymbol{u})$ $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$
$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} =$	$a\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}$	$v\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} + \frac{\partial v}{\partial \boldsymbol{x}}\boldsymbol{u}^T$	$\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}$	$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\boldsymbol{A}^T$	$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\frac{\partial g(\boldsymbol{u})}{\partial \boldsymbol{u}}$

Gradient table: scalar by vector

(denominator layout)

$y =$	a	$\boldsymbol{u}^{\boldsymbol{T}}\boldsymbol{v}$ $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$	$g(u)$ $u = u(\boldsymbol{x})$	$\boldsymbol{x}^{\boldsymbol{T}} A\boldsymbol{x}$
$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} =$	$\mathbf{0}$	$\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}}\boldsymbol{u} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\boldsymbol{v}$	$\frac{\partial g(u)}{\partial u}\frac{\partial u}{\partial \boldsymbol{x}}$	$(A + A^T)\boldsymbol{x}$

Vectorization

- Consider a single instance

$$J(\boldsymbol{w}) = L \quad (\boldsymbol{x}, y|\boldsymbol{w}) = \frac{1}{2}||\tilde{y} - y||^2 = \frac{1}{2}\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} - y\right)^2$$

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = ?$$

$$J(\boldsymbol{w}) = \frac{1}{2}z^2, \text{ where } z = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} - y$$

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial J}{\partial z}\frac{\partial z}{\partial \boldsymbol{w}} = z\boldsymbol{x} = \left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} - y\right)\boldsymbol{x}$$

Shape Check!

Vectorization

- Consider multiple examples $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}$, i=1, 2, …, m

$$X = \begin{pmatrix} \boldsymbol{x}^{(\mathbf{1})^T} \\ \boldsymbol{x}^{(\mathbf{2})^T} \\ \dots \\ \boldsymbol{x}^{(\boldsymbol{m})^T} \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \dots \\ y^{(m)} \end{pmatrix} \qquad \widetilde{\boldsymbol{y}} = X\boldsymbol{w}$$

$$J(\boldsymbol{w}) = \frac{1}{2m}||\widetilde{\boldsymbol{y}} - y||^2 = \frac{1}{2m}(\widetilde{\boldsymbol{y}} - \boldsymbol{y})^{\mathrm{T}}(\widetilde{\boldsymbol{y}} - \boldsymbol{y}) = \frac{1}{2\mathrm{m}}\boldsymbol{u}^{\boldsymbol{T}}\,\boldsymbol{u} \qquad (\boldsymbol{u} = \widetilde{\boldsymbol{y}} - \boldsymbol{y})$$

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{1}{2m}\left(\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{w}}\boldsymbol{u} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{w}}\boldsymbol{u}\right) = \frac{1}{m}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{u} = \frac{1}{m}\boldsymbol{X}^{\boldsymbol{T}}(\widetilde{\boldsymbol{y}} - \boldsymbol{y})$$

Shape Check!

Classification

Regression vs. classification, cross-entropy loss

Multi-class, multi-label classification

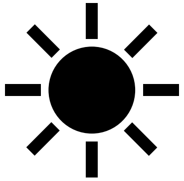
Multi-class, single-label classification

Regression VS Classification

Q: What is the difference?

Quantity vs. Label:
regression maps to a continuous domain, while classification maps to a finite set.

- Regression: What's the temperature of tomorrow?
- Classification (Binary): Is it sunny tomorrow?
- Classification (Multi-class): Is it sunny, cloudy, or rainy?



today	1
tomorrow	0

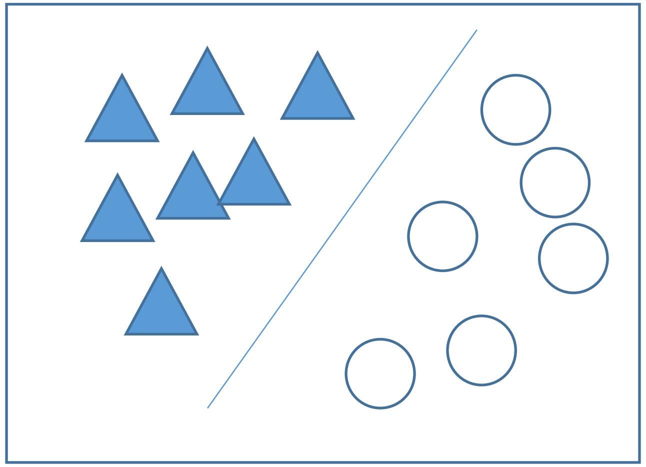
	Sunny	Rainy	Cloudy
Monday	1	0	0
Tuesday	0	1	0
Wednesday	0	0	1

Q: How many mm of rain makes it a "rainy" day? How many hours of sunshine makes it a "sunny" day?

Often, classification labels must be derived from continuous values (measured or regressed).

Modified from: Angela yao

From regression to classification



- Thresholding (Perceptron)
- $\tilde{y} = \begin{cases} 1, & if\, \boldsymbol{w}^T\boldsymbol{x} > c \\ 0, & else \end{cases}$
- How to set the threshold c?

Learn it as a part of learning weights.

$$\boldsymbol{w}^T\boldsymbol{x} > c$$

$$x_1w_1 + x_2w_2 \ldots + x_mw_m + b > c$$

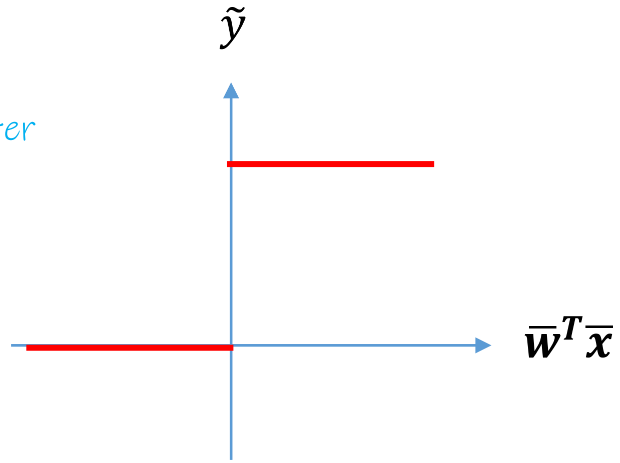
$$x_1w_1 + x_2w_2 \ldots + x_mw_m + \underbrace{(b - c)} > 0$$

Merge c as part of the offset / b parameter

What is $\overline{w}$, $\overline{x}$?

$$\overline{\boldsymbol{w}}^T\ \overline{\boldsymbol{x}} > 0$$

$$\tilde{y} = \begin{cases} 1, & if\, \overline{\boldsymbol{w}}^T\overline{\boldsymbol{x}} > 0 \\ 0, & else \end{cases}$$

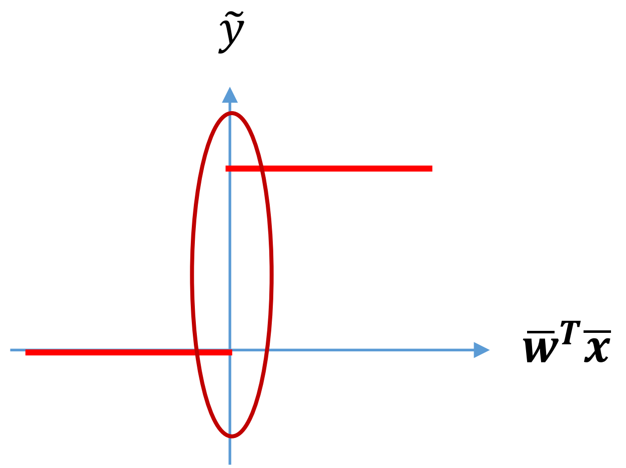


Modified from: Angela yao

Logistic regression

Only piecewise differentiable

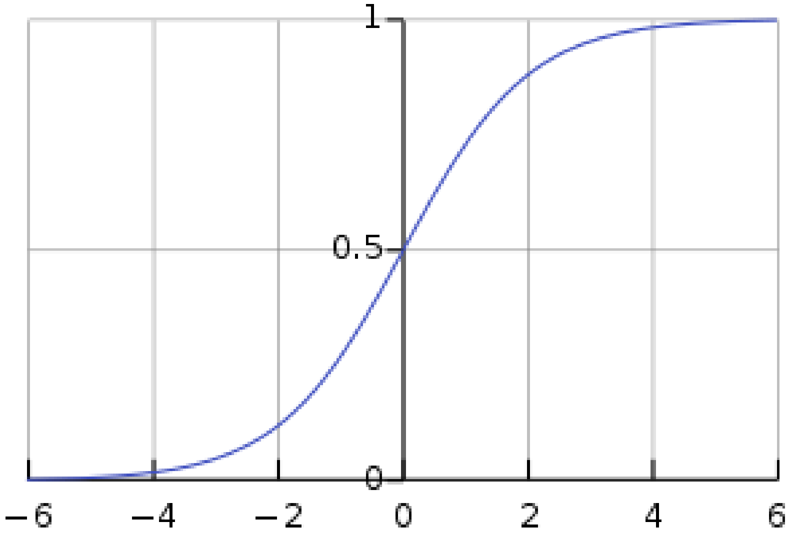
$$\tilde{y} = \begin{cases} 1, & if\, \boldsymbol{w}^T\boldsymbol{x} > 0 \\ 0, & else \end{cases}$$



Gradient is always 0, so cannot learn anything!

Logistic function: $p = \sigma(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}}}$

- Range is within [0, 1]
- Possible interpretation: probability of the label being 1
- Logistic function sometimes also referred to as a sigmoid function



Modified from: Angela yao

Gradient Vanishing

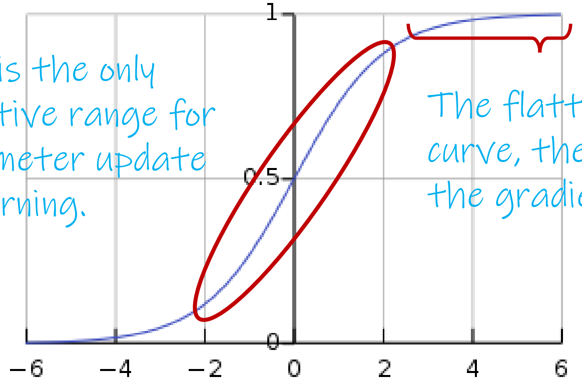
How should we learn the weights of the logistic function? Start with a simple L2 loss.

- $L(\boldsymbol{x}, y) = \frac{1}{2}||\sigma(\boldsymbol{w}^T\boldsymbol{x}) - y||^2, \frac{\partial L}{\partial \boldsymbol{w}}?$
  - denote $z = \boldsymbol{w}^T\boldsymbol{x}, L = \frac{1}{2}(\sigma(z) - y)^2$

HWQ: derive the following gradient result.

- $\frac{\partial L}{\partial \boldsymbol{w}} = (\sigma(z) - y) * \sigma(z)(1 - \sigma(z))\ \mathbf{x}$
- If $\sigma \approx 0\ or\ 1, \frac{\partial \sigma}{\partial z} \approx 0 \rightarrow \frac{\partial L}{\partial \boldsymbol{w}} \approx \mathbf{0}$ gradient vanishing
- Impact: training gets stuck since $\boldsymbol{w} = \boldsymbol{w} - \alpha * \frac{\partial L}{\partial \boldsymbol{w}}$

This is the only effective range for parameter update or learning.



The flatter the curve, the smaller the gradient.

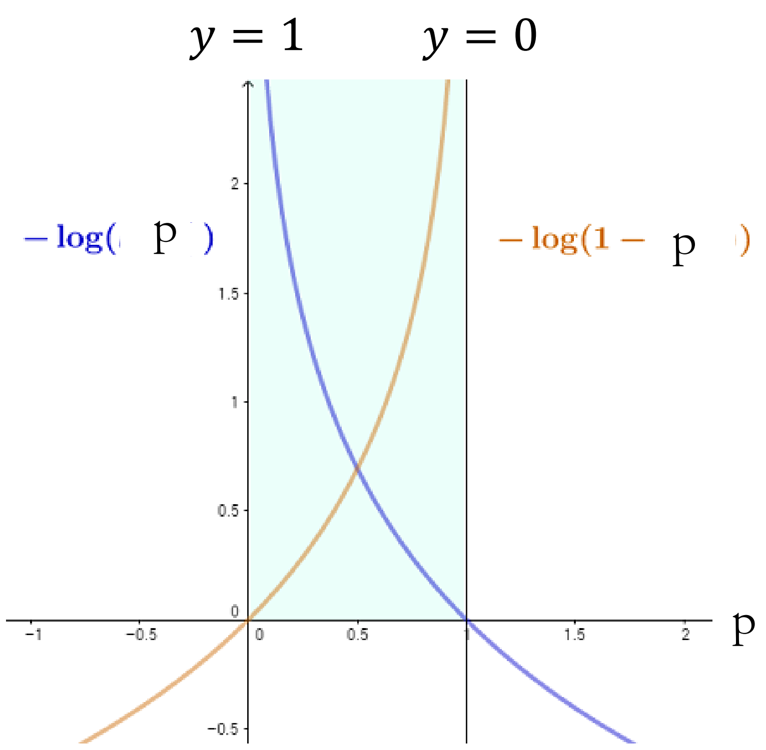
Modified from: Angela yao

Cross-entropy loss

- Denote $p = \sigma(z), z = \boldsymbol{w}^T\boldsymbol{x}$
- Instead of using an L2 loss, we use the following:
- $L_{ce}(x, y) = \underbrace{-y\log p}_{\text{This term goes to 0 if ground truth label is 0}} - \underbrace{(1-y)\log(1-p)}_{\text{This term goes to 0 if ground truth label is 1}}$

$$= \begin{cases} -y\log p\,, & if\ y = 1 \\ -(1-y)\log(1-p)\,, & if\ y = 0 \end{cases}$$

$$= \begin{cases} -\log p\,, & if\ y = 1 \\ -\log(1-p)\,, & if\ y = 0 \end{cases}$$



What is the intuition behind this loss? Does it actually help us learn the right weights?

Modified from: Angela yao

Cross-entropy loss explanation

Consider the probability of a classifier being correct.

$$P(correct|\boldsymbol{x}) = \begin{cases} P(\tilde{y} = 1|\boldsymbol{x}), & if\ y = 1 \\ P(\tilde{y} = 0|\boldsymbol{x}), & if\ y = 0 \end{cases}$$

(depends on the ground truth label y)

$$= P(\tilde{y} = 1|\boldsymbol{x})^y P(\tilde{y} = 0|\boldsymbol{x})^{1-y}$$

collapse cases into a single function

We want to maximize this, i.e. to maximize the probability of our classifier being correct!

Log-likelihood of our classifier being correct:

$$\log P(correct|\boldsymbol{x}) = y \log P(\tilde{y} = 1|\boldsymbol{x}) + (1-y) \log P(\tilde{y} = 0|\boldsymbol{x})$$

Objective equivalent to minimizing the negative log-likelihood

$$\min -\log P(correct|\boldsymbol{x}) = \min -y \log P(\tilde{y} = 1|\boldsymbol{x}) - (1-y) \log P(\tilde{y} = 0|\boldsymbol{x})$$

Note that so far, this is general and that we have not made any assumptions about the classifier itself, i.e. the specific form of $P(\tilde{y}|\boldsymbol{x})$

Modified from: Angela yao

Cross-entropy loss explanation

$P(\tilde{y} = 1|\boldsymbol{x}) = p = \sigma(z)$

Because range of logistic is between 0-1, we adopt p as the probability of the of x having a label.

$P(\tilde{y} = 0|\boldsymbol{x}) = 1 - p$

Problem is binary, equate probability of having label 0 as the complement.

Minimizing negative log likelihood:

$$\min -\log P(correct|\boldsymbol{x}) = \min -y \log P(\tilde{y} = 1|\boldsymbol{x}) - (1 - y) \log P(\tilde{y} = 0|\boldsymbol{x})$$

$$= \min \underbrace{-y \log p \; - (1 - y) \log (1 - p)}_{L_{ce}(\boldsymbol{x}, y)}$$

Substituting the logistic function for $P(\tilde{y}|\boldsymbol{x})$

The cross-entropy loss minimizes the classifier's log-likelihood.

Modified from: Angela yao

Cross-entropy loss gradient

$$\frac{\partial L_{ce}}{\partial \boldsymbol{w}} = \frac{\partial L_{ce}}{\partial z}\frac{\partial z}{\partial \boldsymbol{w}} = \frac{\partial L_{ce}}{\partial p}\frac{\partial p}{\partial z}\frac{\partial z}{\partial \boldsymbol{w}}$$

$$= \left(-\frac{y}{p} + \frac{1-y}{1-p}\right) * p * (1-p)\boldsymbol{x}$$

$$= (p-y)\boldsymbol{x}$$

Modified from: Angela yao

Multi-class classification

- Single-label

	Sunny	Rainy	Cloudy
Monday	1	0	0
Tuesday	0	1	0
Wednesday	0	0	1

- Multi-label: can belong to more than one class

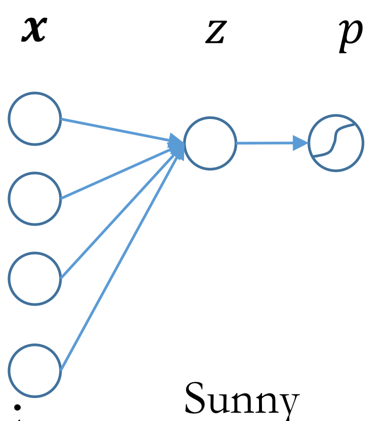
Monday	1	0	1	Sunny & Cloudy
Tuesday	0	1	1	Rainy & Cloudy

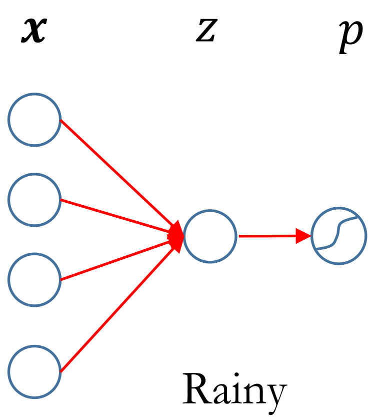
Applications

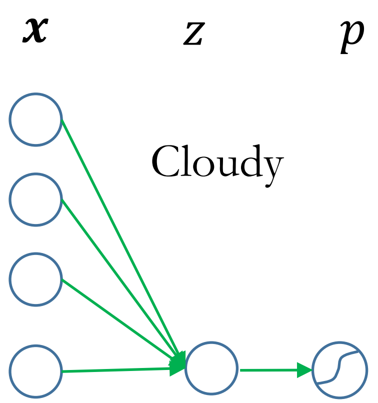
- Multi-class image classification
  - Classify each image into one of the class
  - MNIST dataset
    - {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
    - What is **x?** i.e., how to represent an image
  - Cifar10 dataset
    - {Dog, Cat, Horse, Ship, Truck, Frog, Deer, Bird, Automobile, Airplane}
- Multi-class document classification
  - 20 Newsgroups, {hardware, autos, space, etc}

Multi-class multi-label classification

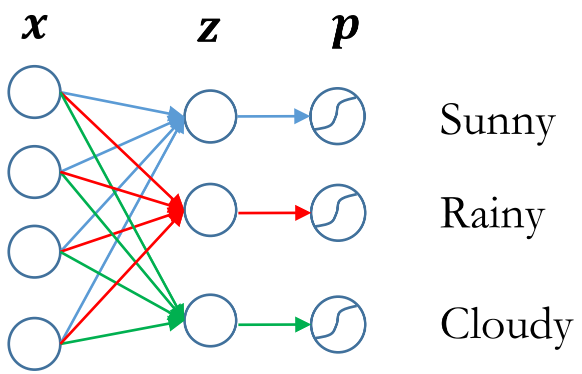
- Binary classification for each label



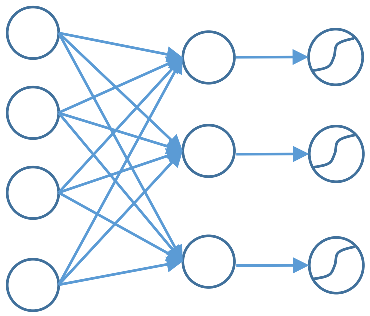




- All in one network



Multi-class multi-label classification



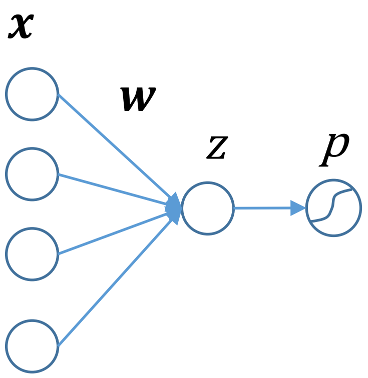
- For binary classification
  - $z = \boldsymbol{w}^T \boldsymbol{x} + b, \qquad p = \sigma(z), \qquad L_{ce} = -y \log p - (1-y)\log(1-p)$
  - $\boldsymbol{x} \in R^n, \boldsymbol{w} \in R^n, b \in R, \ p \in R$
- For multi-class, the i-th class
  - We assume that the classes are independently conditioned on the input
  - $z_i = \boldsymbol{W}_i \boldsymbol{x} + b_i, \qquad p_i = \sigma(z_i), \qquad L_{ce} = -y_i \log p_i - (1-y_i)\log(1-p_i)$
  - $x \in R^n, \ \boldsymbol{W}_i \in R^{1\times n}, \ b_i \in R^1, \quad p_i \in R^1$

Probability of belonging to class i is independent of belonging to class j, once conditioned on the input evidence

- For multi-class, multi-label (vectorized form)
  - $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \qquad \boldsymbol{p} = \sigma(\boldsymbol{z}), \qquad L_{ce} = -\boldsymbol{y}^T \log \boldsymbol{p} - (\boldsymbol{1}-\boldsymbol{y})^T \log(\boldsymbol{1}-\boldsymbol{p})$
  - $\boldsymbol{x} \in R^n, \ \boldsymbol{W} \in R^{k\times n}, \ \boldsymbol{b} \in R^k, \ \boldsymbol{p} \in R^k, \ \boldsymbol{y} \in \{0,1\}^k$

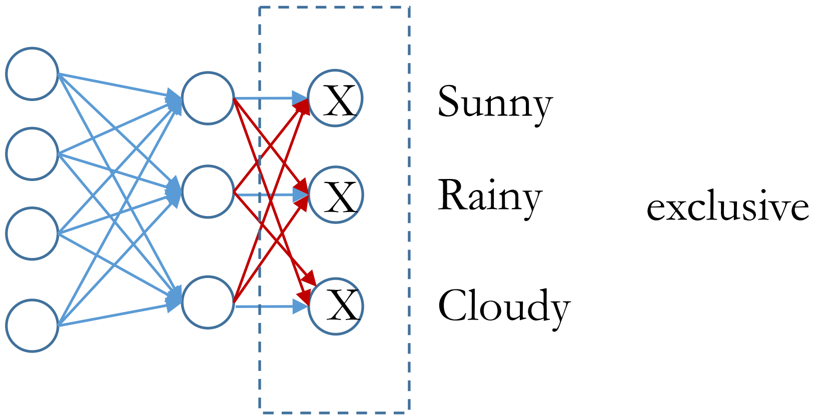
Multi-class single-label classification

- Choose one label from multiple classes
  - Exclusive
    - If $p(sunny)$ is large, then $p(rainy) + p(cloudy)$ small



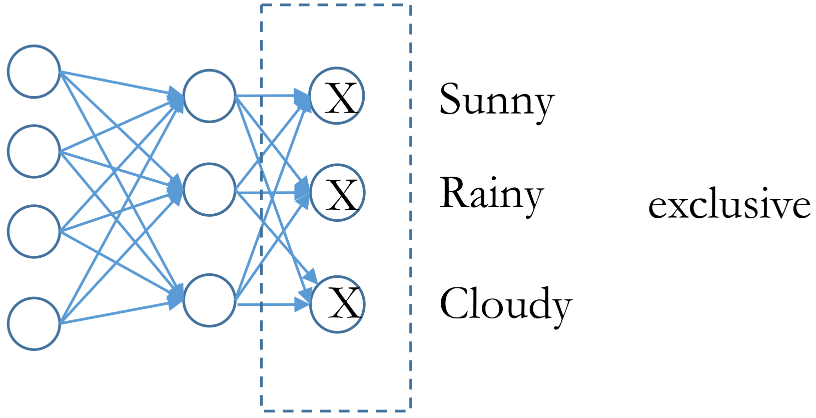
- How to enforce this constraint?
  - $p(sunny) + p(rainy) + p(cloudy) = 1$
  - $\sum_{i=1} p_i = 1$

Our previous model had connections only from each z_i to p_i; now we add the red arrows to connect all z_i to p_i



Multi-class single-label classification

- Softmax regression or multinomial logistic regression
  - $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}, \boldsymbol{W} \in R^{k \times n}$,
  - $p_i = \frac{e^{z_i}}{\sum_j e^{z_j}} = \text{softmax}(z_i)$
  - Then we have $\sum_i p_i = 1$
  - z_i is called a logit
  - $t = \underset{i}{\text{argmax}}\, p_i\,;\ \tilde{y}_i = 1\ if\ i = t;\ else\ 0;$



Multi-class single-label classification

- Loss function: $L_{ce}(\boldsymbol{x},\boldsymbol{y}) = \sum_i -y_i \log p_i \; = -\boldsymbol{y}^T \log \boldsymbol{p}$
  - Each row of X is the feature vector $\boldsymbol{x}$
  - Each row of Y is the target one-hot vector $\boldsymbol{y}$

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}^{(1)^T} \\ \boldsymbol{x}^{(2)^T} \\ \dots \\ \boldsymbol{x}^{(N)^T} \end{pmatrix} \qquad \boldsymbol{Y} = \begin{pmatrix} \boldsymbol{y}^{(1)^T} \\ \boldsymbol{y}^{(2)^T} \\ \dots \\ \boldsymbol{y}^{(N)^T} \end{pmatrix}$$

- $\dfrac{\partial L_{ce}}{\partial \boldsymbol{W}} = \;?$ $\quad \dfrac{\partial L_{ce}}{\partial \boldsymbol{z}} \dfrac{\partial \boldsymbol{z}}{\partial \boldsymbol{w}}$

HWQ: derive the gradient of Lce w.r.t to W.

Cross-entropy

- *“Average number of bits needed to identify an event drawn from the set, if a coding scheme of the set is optimized for an estimated probability distribution p, rather than the "true" distribution q”--- Wikipedia*
- $H(q,p) = H(q) + D_{kl}(q||p) = -\sum_k q_k \log p_k$
  - $H(q)$ is a constant, i.e., the entropy of the true distribution
  - $D_{kl}(q||p)$ measures the difference between two distributions
  - Therefore, $H(q,p)$ measures the difference between two distributions
- For multi-class classification
  - $\boldsymbol{p}$ is the artificial distribution to be optimized
  - $\boldsymbol{y}$ is the true distribution (q), i.e., the label distribution

Summary

- Regression vs. classification as basic machine learning tasks
  - Using logistic regression to approximate the decision for classification
  - Logistic functions should be learned with cross-entropy and not L2 loss due to gradient vanishing proble
  - cross-entropy loss tries to minimize the difference between the output distribution and the (ground truth) label distribution

Homework Questions

1. Derive the gradient of the L2 loss wrt $\boldsymbol{w}$ using a logistic regression function (slide 23)
2. Derive the gradient of the cross entropy loss (slide 27)
3. Derive the gradient of Lce w.r.t to $\mathbf{W}$ (slide 34)

Next Lecture

Overfitting & Multi-layer perceptron & Back-propagation