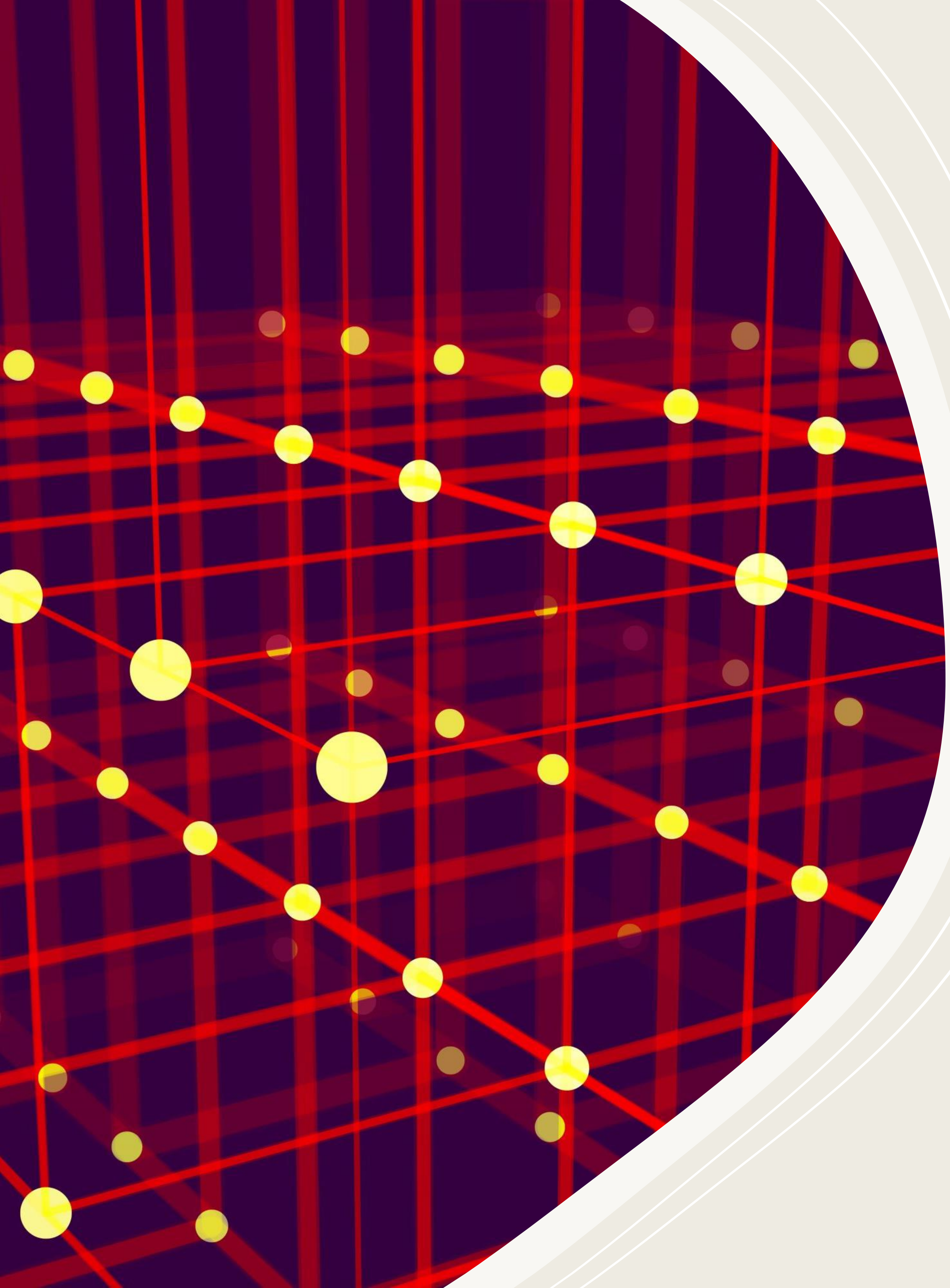


CPSC 340 – Tutorial 6

Lironne Kurzman
lironnek@cs.ubc.ca

Slides courtesy of Nam Hee Kim

University of British Columbia

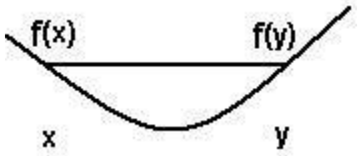
October 25th, 2021

Agenda

1. Convexity
2. Logistic Regression
3. Softmax Classifier

How do we show a function is convex?

Definitions of Convex



1. Chord definition

$$f(\theta w + (1-\theta)v) \leq \theta f(w) + (1-\theta) f(v), 0 \leq \theta \leq 1$$

Convex combination "Chord"

2. **Non-negative eigenvalues of Hessian (non-negative second derivative)**

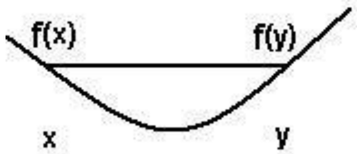
$f''(w) \geq 0 \text{ for all } w \text{ (1D case)}$ if f(w) is twice-differentiable everywhere

$\nabla^2 f(w) \succeq 0 \text{ for all } w$

3. **Operations that preserve convexity**

How do we show a function is convex?

- Any *p*-norm and squared *p*-norm function is convex (p >= 1)
- let f and g be convex functions, then
  - h(w) = max(f(w), g(w)) is convex
  - h(w) = f(Aw + b) is convex
  - h(w) = k * f(w) is convex (k >= 0)
  - h(w) = f(w) + g(w) is convex
- MANS: **M**aximum, **A**ffine map, **N**on-negative scaling, **S**um
- h(w) = f(g(w)) is not necessarily convex

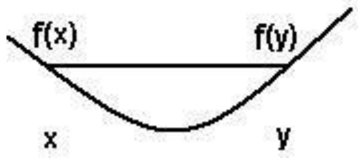


How do we show a function is convex?

Are these functions convex?

$$f(x) = -\log(x^2)$$

$$f(w) = \frac{1}{2}||Xw - y||^2 + \frac{\lambda}{2}||w||^2$$



What is Regularization?

- Attempt to reduce complexity (effective degrees of freedom) of the model
- Often causes higher training error and lower test error

$$f(w) = \frac{1}{2}||Xw - y||^2 + \frac{\lambda}{2}||w||^2$$

Objective function Loss function Penalty function

$$w = \text{argmin}_w \ f(w)$$

L2 Regularization

$$f(w) = \sum_{i=1}^{n} \left[\log(1 + \exp(-y_i w^T x_i))\right] \boxed{+ \frac{\lambda}{2}\|w\|^2.}$$

- Properties of L2 Regularization:

  - Insensitive to changes in data
  - Decreases the variance
  - **Closed form solution!**
  - **Solution is unique!**
  - **Weights are not sparse!**

L1 Regularization

$$f(w) = \sum_{i=1}^{n} \left[\log(1 + \exp(-y_i w^T x_i))\right] \boxed{+ \lambda \|w\|_1.}$$

- Properties of L1 Regularization:

  - Insensitive to changes in data
  - Decreases the variance
  - **Requires iterative solver!**
  - **Solution is not unique!**
  - **Weights are sparse!**

L0 Regularization

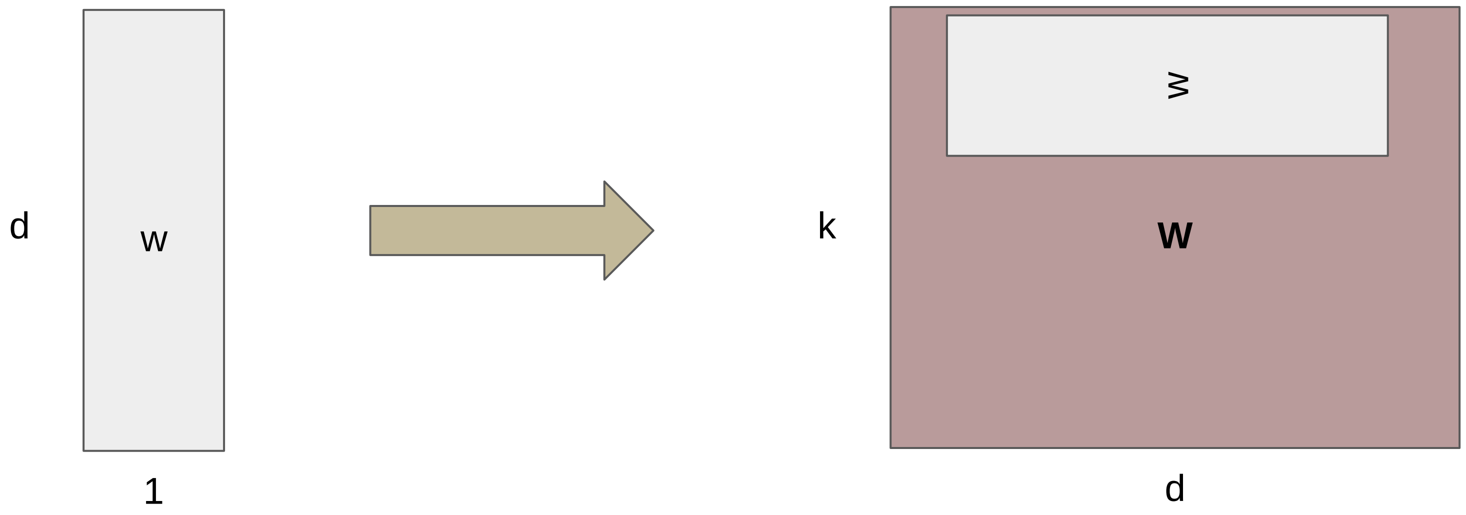
$$f(w) = \sum_{i=1}^{n} [\log(1 + \exp(-y_i w^T x_i))] \boxed{+ \lambda \|w\|_0.}$$

- Properties of L0 Regularization:

  - Constant penalty of λ for non-zero weights
  - Encourages w_j to be exactly zero!
  - Solution is not unique

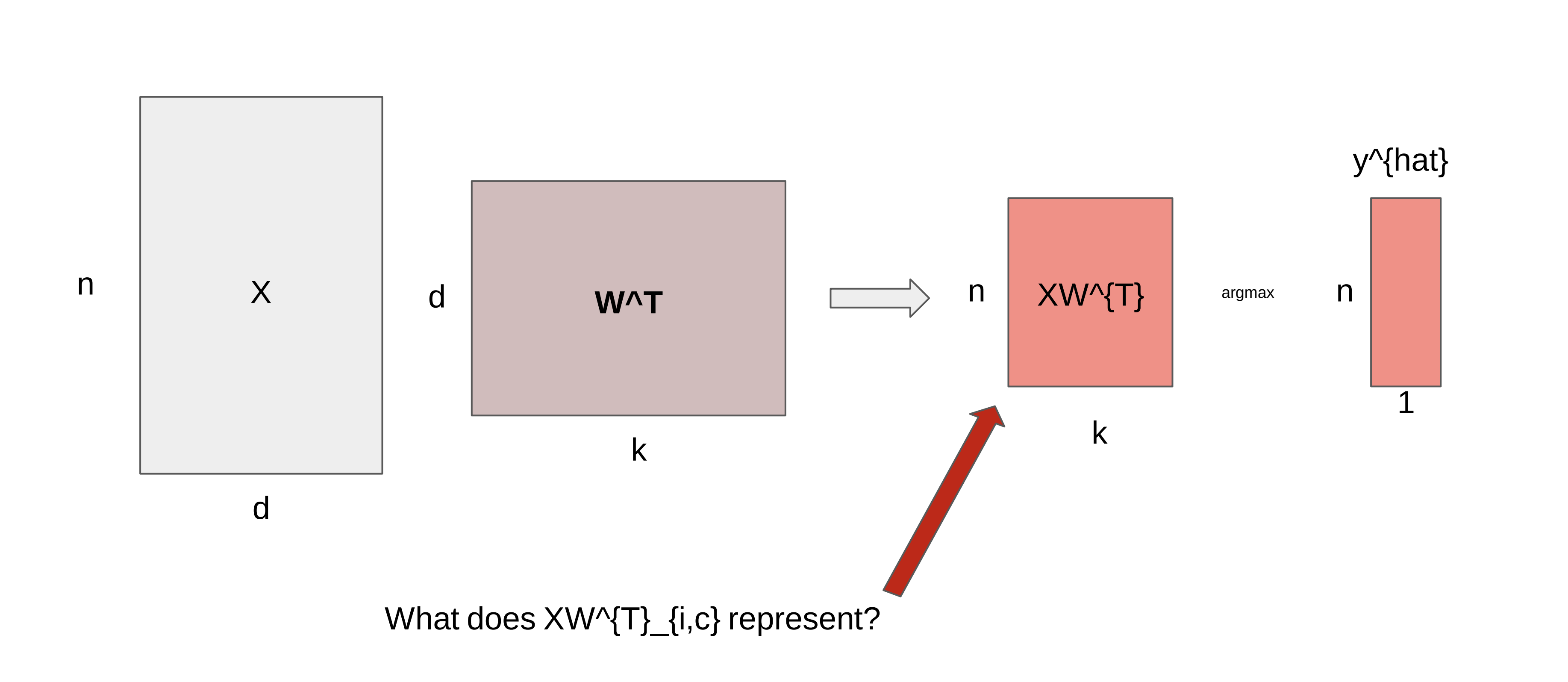
Multi Class Classification

Single class classification



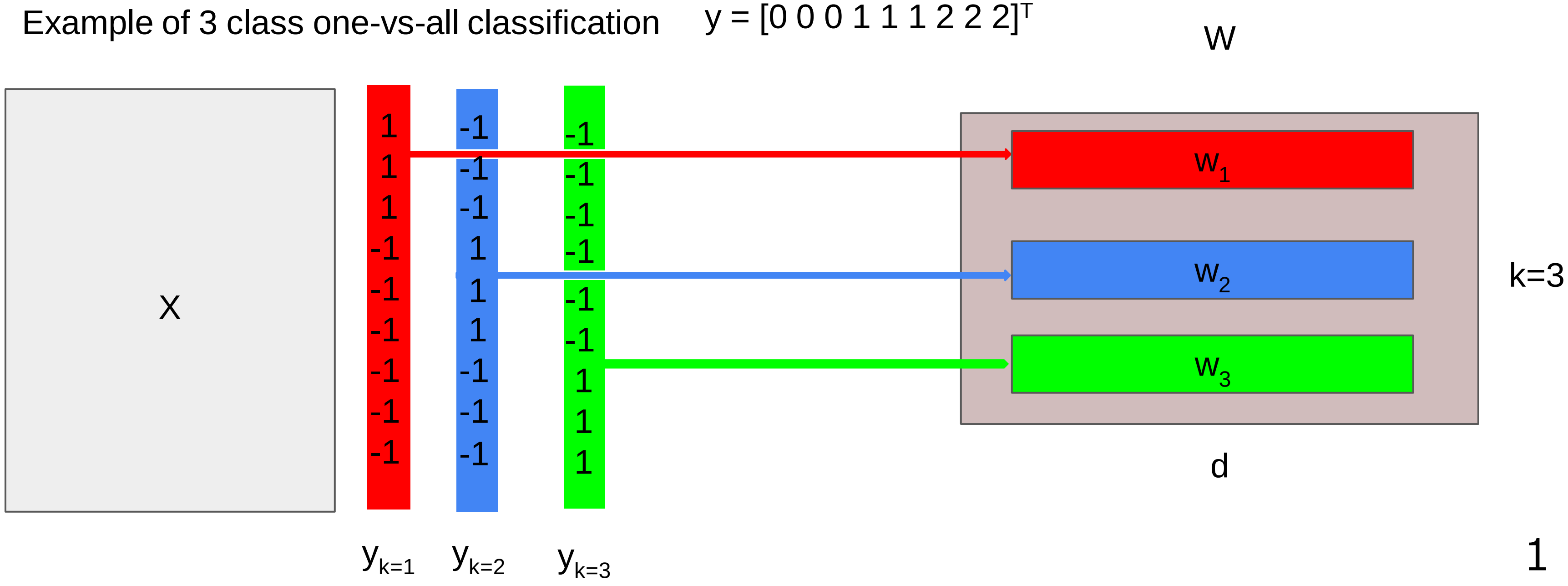
What is k in the multi-class classification weight matrix?

Multi Class Classification



One-vs-All Logistic Regression

Example of 3 class one-vs-all classification $y = [0\ 0\ 0\ 1\ 1\ 1\ 2\ 2\ 2]^T$



What are the labels in each y?

Softmax Gradient

The softmax function for k classes:

$$f(W) = \sum_{i=1}^{n} \left[-w_{y_i}^T x_i + \log \left(\sum_{c'=1}^{k} \exp(w_{c'}^T x_i) \right) \right] ,$$

How are these two weight vectors different?

Softmax Gradient

The softmax function for k classes:

$$\frac{\partial f}{\partial W_{cj}} = \sum_{i=1}^{n} x_{ij}[p(y_i = c \mid W, x_i) - I(y_i = c)]$$

What is this taking the partial derivative of?
What are c and j?
What is the “size” of this (vector or scalar)?

- $I(y_i = c)$ is the indicator function (it is 1 when $y_i = c$ and 0 otherwise)
- $p(y_i = c \mid W, x_i)$ is the predicted probability of example i being class c, defined as

$$p(y_i = c \mid W, x_i) = \frac{\exp(w_c^T x_i)}{\sum_{c'=1}^{k} \exp(w_{c'}^T x_i)}$$

Expanding the Product

The softmax function for k classes:

$$f(W) = \sum_{i=1}^{n} \left[\boxed{-w_{y_i}^T x_i} + \log \left(\sum_{c'=1}^{k} \exp(w_{c'}^T x_i) \right) \right] ,$$

$$\Downarrow$$

$$\boldsymbol{w_{y_i}^T x_i = w_{y_i,0} x_{i,0} + w_{y_i,1} x_{i,1} + \ldots + w_{y_i,d} x_{i,d}}$$

Softmax Gradient

The softmax function for k classes:

$$f(W) = \sum_{i=1}^{n} \left[\boxed{-w_{y_i}^T x_i} + \log \left(\sum_{c'=1}^{k} \exp(w_{c'}^T x_i) \right) \right],$$

What is the partial derivative of this w.r.t class c?
Is it always non-zero?

$$\frac{\partial f}{\partial W_{cj}} = \sum_{i=1}^{n} x_{ij} [p(y_i = c \mid W, x_i) - I(y_i = c)]$$

- $I(y_i = c)$ is the indicator function (it is 1 when $y_i = c$ and 0 otherwise)

Speeding up Softmax

- Look for things to pre-compute:
  - Are there any matrix computations used repeatedly?
  - Are certain matrices able to be computed independently and reused?

- Use NumPy broadcasting/vectorization for quick matrix multiplication
- Use NumPy array operations (np.sum, np.exp, np.log) where applicable

Tips for Coding Softmax

- Implement it with as many for loops as you need and make sure it works!
- Then if you want you can try and speed up the computation through precomputing and vectorization
- Make sure that the dimensions of your gradients is correct
- Make sure your indexing for your matrices are correct (if applicable)