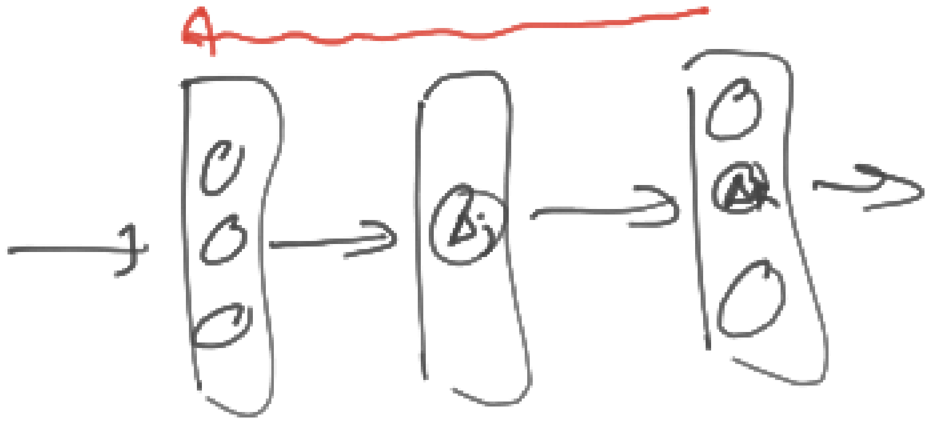
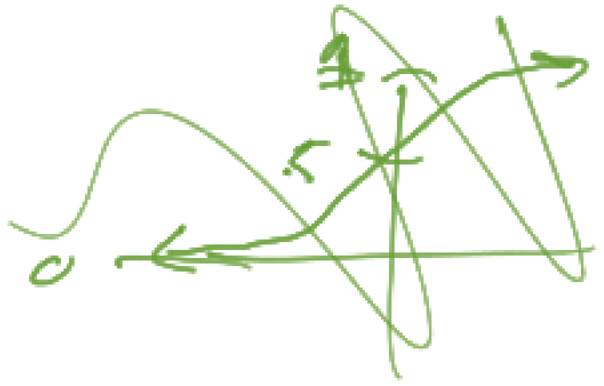
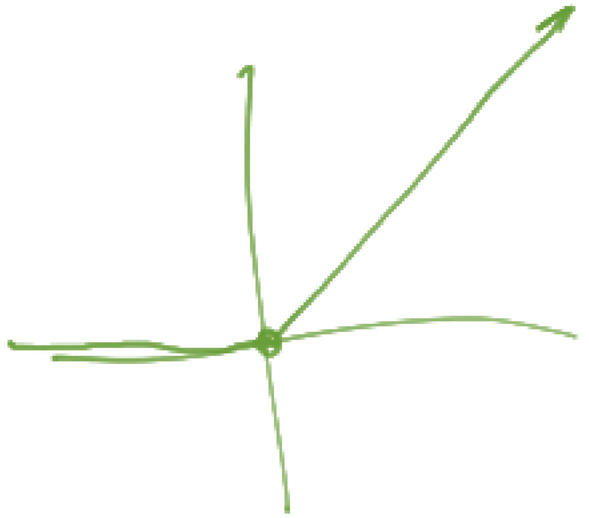
Vanishing Gradients

Recall:

$$\frac{\partial f_w(x)}{\partial w_{ij}} = \underbrace{\sum_k \Delta_k w_{jk} \overbrace{\sigma(in_j)(1-\sigma(in_j))}^{\frac{\partial \sigma(in_j)}{\partial in_j}}}_{\Delta_j} a_i$$







ReLU

- $\frac{\partial f_w(x)}{\partial w_{ij}} \to 0$ as you go farther and farther back in the network.

Lecture 9

- vanishing gradients
- weight init
- Adaptive step size
- Classification
- Generalization Bounds

$in_j = \sum_i a_i w_{ij}$

Weight Initialization

- Heuristic
- Want:
  - mean of in_j to be zero.
  - variance of in_j should be the same across all layers.
- Xavier initialization: For layer l

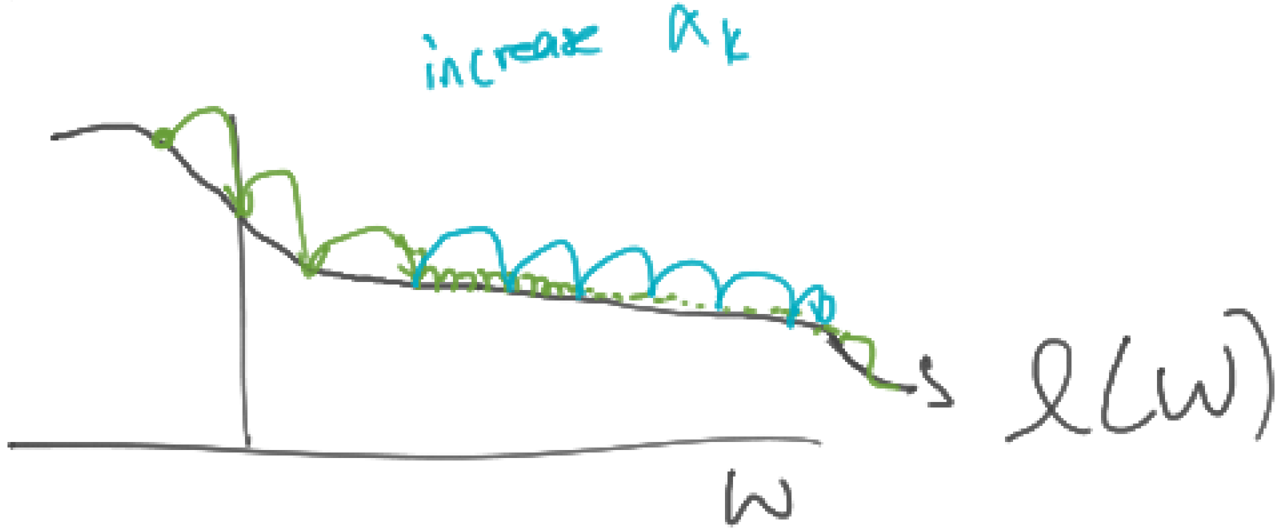
$$w_{ij} \sim N\left(0, \frac{1}{n_{l-1}}\right)$$

, where n_{l-1} is number of nodes in layer $l-1$.

- He initialization (effective for ReLU).

$$w_{ij} \sim N\left(0, \frac{2}{n_{l-1}}\right)$$

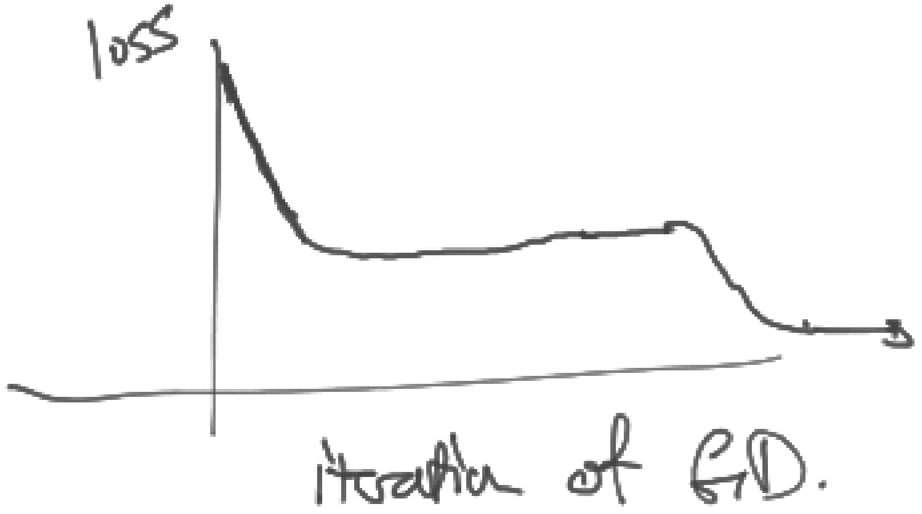
Adaptive Step Sizes. α_k (step size at k^{th} iteration of GD)



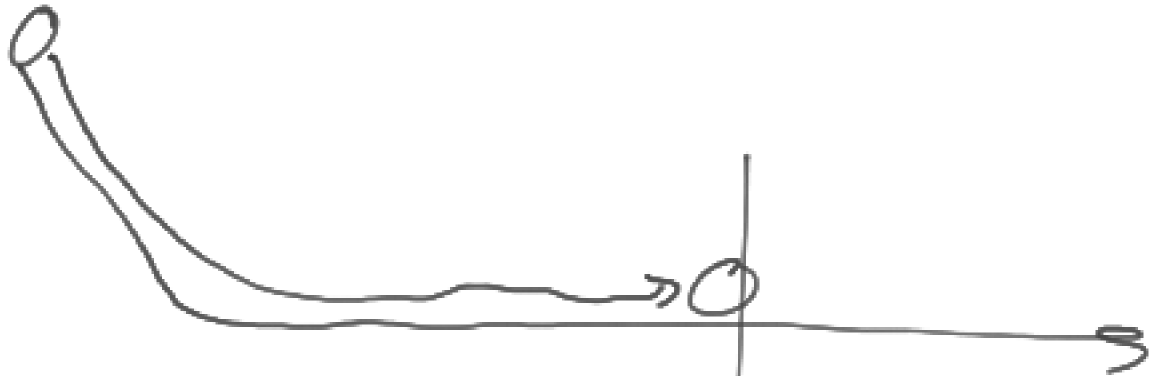
Momentum:

eta $\in [0,1]$ (often decayed to zero)

$$\Delta\theta_k = \eta\,\Delta\theta_{k-1} + \alpha \nabla \ell(w_k)$$



$$\theta_{k+1} = \theta_k - \Delta\theta_k$$



Many more! (Keep a different step size for each model parameter!)

- Adaptive Gradient (AdaGrad)
- Root mean square propagation (RMS Prop)
- Adaptive moment estimation (Adam)

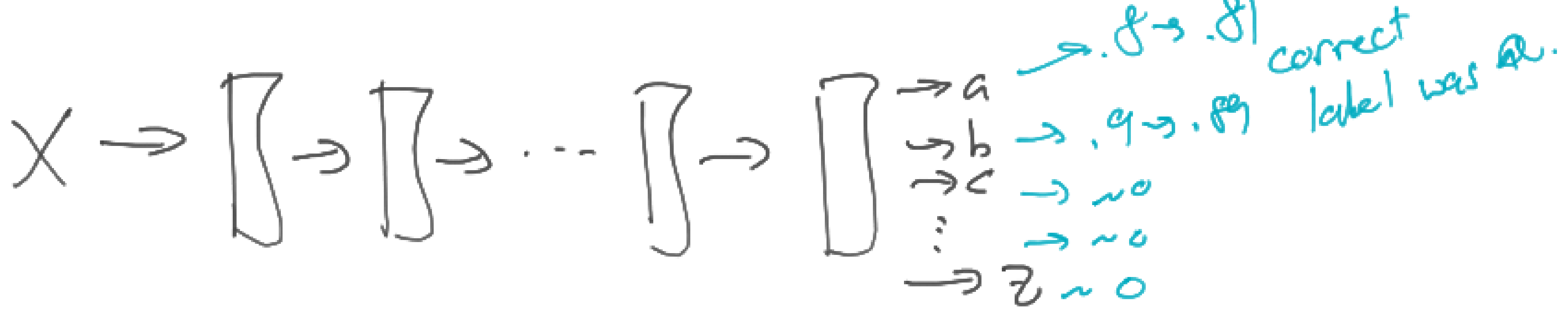
Classification

Like regression, but $Y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is a set (typically finite) of labels.

Examples: $\mathcal{Y} = \{\text{is a cat in image, there is no cat}\}$

$\mathcal{Y} = \{a, b, c, \ldots, z\}$

Idea: One model output per label.
Larger values mean that label is more reasonable



- Predicted label is the largest output.
  - Not differentiable. (due to max operator)

<u>Idea:</u> Use "softmax" for last layer.

$$\Pr(\text{label} = \ell) = \frac{e^{a_\ell}}{\sum_{\ell'} e^{a_{\ell'}}}$$

make positive

10	e^{10}	e^{10}/β
-20	e^{-20}	e^{-20}/β
700	e^{700}	e^{700}/β
-3	e^{-3}	e^{-3}/β

sum to one.

$\beta = e^{10} + e^{-20} + e^{700} + e^{-3}$

drop σ from last layer.

$$\Pr(\text{label prediction} = \text{apple}) = \frac{e^{.9}}{e^{.9} + e^{.7} + e^{.1}}$$

$$\approx 0.44.$$

$\Pr(\text{orange}) \approx .36$

$\Pr(\text{peach}) \approx .19$

Cross Entropy Loss (Binary)

- Binary Classification, $|\mathcal{Y}| = 2$, $\mathcal{Y} = \{0, 1\}$

$\mathcal{Y} = \{-1, 1\}$

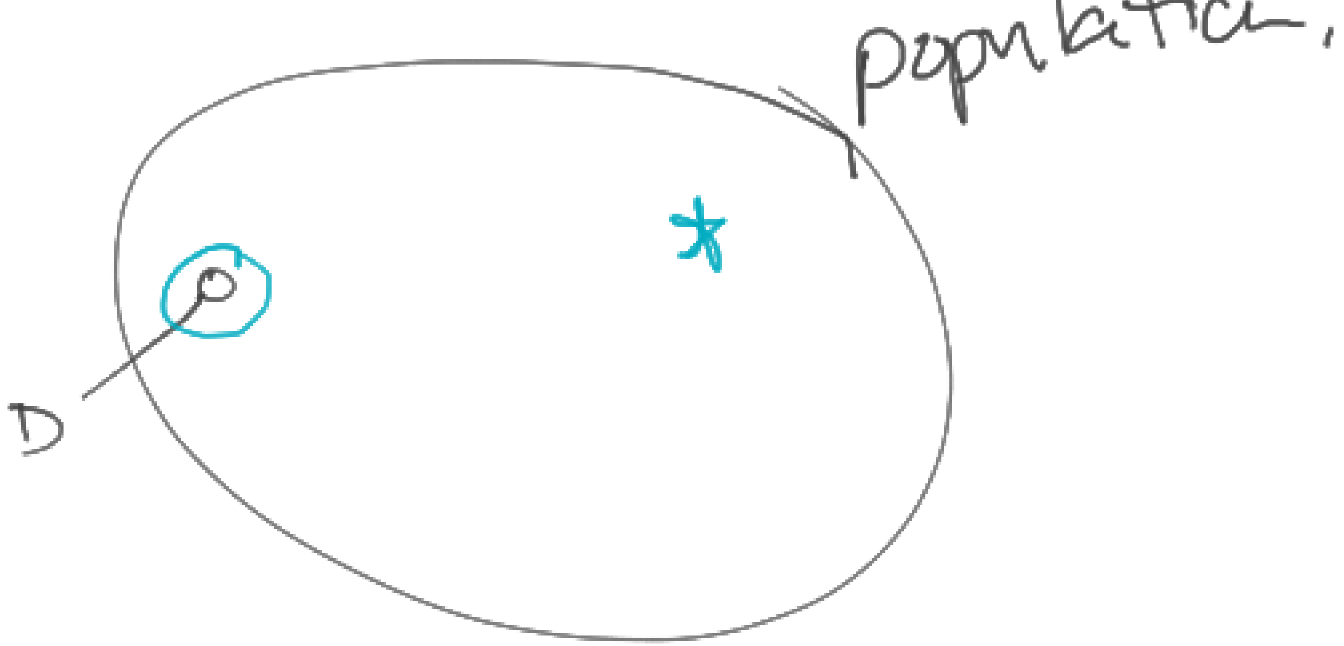
$$\ell(w) = -\frac{1}{n}\sum_{i=1}^{n} \ln\left(\Pr\left(f_w(x_i) = y_i\right)\right)$$

monotonic.

random due to softmax

Generalization bounds

D = n samples for training.



Data is provided with no explanation.
Could be chosen any (not random) way.

→ Assume data points are sampled i.i.d. from population.

independent
identically distributed.

$$\ell(w) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

(x_i, y_i) are i.i.d.

X_i (not input x_i)

$$\frac{X_1 + X_2}{2} \qquad \frac{X_1 + X_2 + X_3}{3} \qquad \bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Hoeffding's inequality:

If $X_1, \ldots, X_n$ are i.i.d. then

$$\Pr(|\bar{X}_n - \mu| \geq t) \leq 2e^{-2nt^2}$$

$X_1, \ldots, X_{1000}$

random

$$\Pr(|\boxed{1.7\text{m}} - \mu| \geq 0.1) \leq 2e^{-2(1000)(0.1)^2}$$

$\bar{X}_n = 1.7\text{m}$

.00000004