ECE276A: Sensing & Estimation in Robotics
Lecture 6: Logistic Regression

Instructor:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Qiaojun Feng: qjfeng@ucsd.edu

Arash Asgharivaskasi: aasghari@eng.ucsd.edu

Ehsan Zobeidi: ezobeidi@ucsd.edu

Rishabh Jangir: rjangir@ucsd.edu

UC San Diego

JACOBS SCHOOL OF ENGINEERING

Electrical and Computer Engineering

Supervised Learning

- **Data**: a set $D := \{\mathbf{x}_i, y_i\}_{i=1}^n$ of **iid** examples $\mathbf{x}_i \in \mathbb{R}^d$ with associated scalar labels y_i generated from an unknown joint pdf $p_*(y, \mathbf{x})$
- The training dataset is often also written in matrix notation, $D = (X, \mathbf{y})$, with $X \in \mathbb{R}^{n \times d}$ and $\mathbf{y} \in \mathbb{R}^n$
- **Generative model**: choose model $p(y, \mathbf{x}; \boldsymbol{\omega})$ with parameters $\boldsymbol{\omega}$ to approximate the unknown data-generating pdf
- **Discriminative model**: choose model $p(y|\mathbf{x}; \boldsymbol{\omega})$ with parameters $\boldsymbol{\omega}$ to approximate the unknown label-generating pdf
- Optimize $\boldsymbol{\omega}$ using $D = (X, \mathbf{y})$:
  - **Maximum Likelihood Estimation (MLE)**: maximize the likelihood of the data D given the parameters $\boldsymbol{\omega}$
  - **Maximum A Posteriori (MAP)**: maximize the likelihood of the parameters $\boldsymbol{\omega}$ given the data D
  - **Bayesian Inference**: estimate the whole distribution of the parameters $\boldsymbol{\omega}$ given the data D

Parametric Learning

- **Maximum Likelihood Estimation (MLE)**:

MLE	Discriminative Model	Generative Model
Training	$\boldsymbol{\omega}^* \in \arg\max_{\boldsymbol{\omega}} p(\mathbf{y} \mid X, \boldsymbol{\omega})$	$\boldsymbol{\omega}^* \in \arg\max_{\boldsymbol{\omega}} p(\mathbf{y}, X \mid \boldsymbol{\omega})$
Testing	$y_* \in \arg\max_{y} p(y \mid \mathbf{x}_*, \boldsymbol{\omega}^*)$	$y_* \in \arg\max_{y} p(y, \mathbf{x}_* \mid \boldsymbol{\omega}^*)$

- **Maximum A Posteriori (MAP)**:

MAP	Discriminative Model	Generative Model
Training	$\boldsymbol{\omega}^* \in \arg\max_{\boldsymbol{\omega}} p(\boldsymbol{\omega} \mid \mathbf{y}, X)$ $= \arg\max_{\boldsymbol{\omega}} p(\mathbf{y} \mid X, \boldsymbol{\omega}) p(\boldsymbol{\omega} \mid X)$	$\boldsymbol{\omega}^* \in \arg\max_{\boldsymbol{\omega}} p(\boldsymbol{\omega} \mid \mathbf{y}, X)$ $= \arg\max_{\boldsymbol{\omega}} p(\mathbf{y}, X \mid \boldsymbol{\omega}) p(\boldsymbol{\omega})$
Testing	$y_* \in \arg\max_{y} p(y \mid \mathbf{x}_*, \boldsymbol{\omega}^*)$	$y_* \in \arg\max_{y} p(y, \mathbf{x}_* \mid \boldsymbol{\omega}^*)$

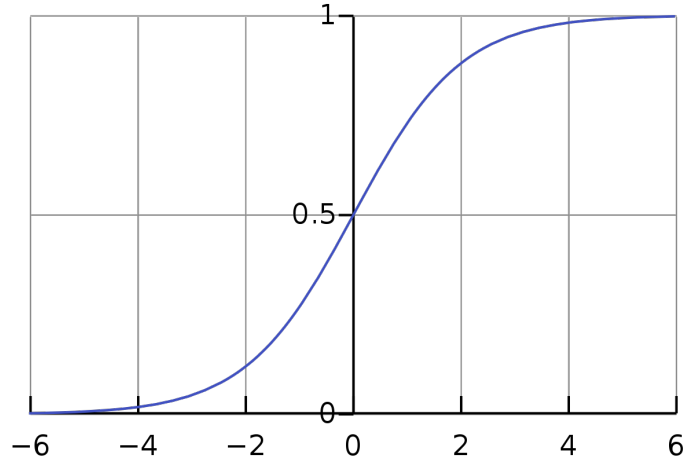
- **Bayesian Inference**:

BI	Discriminative Model	Generative Model
Training	$p(\boldsymbol{\omega} \mid \mathbf{y}, X) \propto p(\mathbf{y} \mid X, \boldsymbol{\omega}) p(\boldsymbol{\omega} \mid X)$	$p(\boldsymbol{\omega} \mid \mathbf{y}, X) \propto p(\mathbf{y}, X \mid \boldsymbol{\omega}) p(\boldsymbol{\omega})$
Testing	$p(y_* \mid \mathbf{x}_*, \mathbf{y}, X) = \int p(y_* \mid \mathbf{x}_*, \boldsymbol{\omega}) p(\boldsymbol{\omega} \mid \mathbf{y}, X) d\boldsymbol{\omega}$	$p(y_*, \mathbf{x}_* \mid \mathbf{y}, X) = \int p(y_*, \mathbf{x}_* \mid \boldsymbol{\omega}) p(\boldsymbol{\omega} \mid \mathbf{y}, X) d\boldsymbol{\omega}$

Logistic Sigmoid Function

- Useful for converting continuous (regression) preferences $z \in \mathbb{R}$ into a Bernoulli probability mass function, which can be used as a probabilistic model for **binary classification**, i.e., $y \in \{-1, 1\}$

$$\sigma(z) := \frac{1}{1 + \exp(-z)} = \frac{\exp(z)}{\exp(z) + \exp(0)}$$



- Properties:
  - $\sigma(z) = 1 - \sigma(-z) = \frac{1}{2} + \frac{1}{2} \tanh\left(\frac{z}{2}\right)$
  - $\sigma'(z) = \sigma(z)(1 - \sigma(z)) = \sigma(z)\sigma(-z)$
  - $\int \sigma(z) dz = \log(1 + \exp(z))$, which is known as the **softplus** function

Softmax Function

- Useful for converting continuous (regression) preferences $\mathbf{z} \in \mathbb{R}^K$ into a categorical probability mass function, which can be used as a probabilistic model for **classification**, i.e., $y \in \{1, \ldots, K\}$

$$\mathbf{s}(\mathbf{z}) := \left[\frac{\exp(z_1)}{\sum_j \exp(z_j)} \quad \cdots \quad \frac{\exp(z_K)}{\sum_j \exp(z_j)} \right] = \frac{e^{\mathbf{z}}}{\mathbf{1}^\top e^{\mathbf{z}}} \in \mathbb{R}^K$$

- Properties:
  - $\mathbf{s}(\mathbf{z}) = \mathbf{s}(\mathbf{z} - c\mathbf{1})$, where $c \in \mathbb{R}$ is any constant, e.g., $c = \max_i z_i$ is useful for numerical conditioning
  - $\log \frac{\mathbf{s}_i(\mathbf{z})}{\mathbf{s}_j(\mathbf{z})} = z_i - z_j$
  - $\frac{ds_i}{d\mathbf{z}_j} = \begin{cases} s_i(1 - s_i) & \text{if } i = j \\ -s_i s_j & \text{else} \end{cases}$
  - $\nabla_{\mathbf{z}} \log \left(\sum_{i=1}^K \exp(z_i) \right) = \mathbf{s}(\mathbf{z})$

Discriminative Classification via a Logistic Model

- Given an example $\mathbf{x} \in \mathbb{R}^d$, use a logistic sigmoid function to map the continuous value $\boldsymbol{\omega}^\top \mathbf{x}$ to a Bernoulli probability mass function for a binary label $y \in \{-1, 1\}$:

$$p(y = 1|\mathbf{x}, \boldsymbol{\omega}) = \sigma(\boldsymbol{\omega}^\top \mathbf{x})$$

- For $p(y|\mathbf{x}, \boldsymbol{\omega})$ to be a valid pmf, it needs to sum to 1 over $\{-1, 1\}$:

$$p(y = -1|\mathbf{x}, \boldsymbol{\omega}) = 1 - p(y = 1|\mathbf{x}, \boldsymbol{\omega}) = 1 - \sigma(\boldsymbol{\omega}^\top \mathbf{x}) \overset{\text{sigmoid}}{\underset{\text{properties}}{=}} \sigma(-\boldsymbol{\omega}^\top \mathbf{x})$$

- Combine the two cases to obtain the final discriminative model for the pmf of y conditioned on the example $\mathbf{x}$ and the parameters $\boldsymbol{\omega}$:

$$p(y|\mathbf{x}, \boldsymbol{\omega}) = \sigma(y\mathbf{x}^\top \boldsymbol{\omega})$$

Discriminative Classification via a Logistic Model

- **Logistic regression**: approximates the unknown label-generating pdf with a logistic sigmoid function:

$$p(y|\mathbf{x}, \boldsymbol{\omega}) = \sigma(y\mathbf{x}^\top \boldsymbol{\omega})$$

- Since the data $D = (X, \mathbf{y})$ are **iid**, the joint data likelihood is:

$$p(\mathbf{y}|X, \boldsymbol{\omega}) = \prod_{i=1}^{n} \sigma(y_i \mathbf{x}_i^\top \boldsymbol{\omega}) = \prod_{i=1}^{n} \frac{1}{1 + \exp(-y_i \mathbf{x}_i^\top \boldsymbol{\omega})}$$

- Leads to these MLE and MAP (with $\boldsymbol{\omega} \sim \mathcal{N}(0, \Lambda)$) estimates for $\boldsymbol{\omega}$:

$$\boldsymbol{\omega}_{MLE} = \arg\max_{\boldsymbol{\omega}} \log p(\mathbf{y} \mid X, \boldsymbol{\omega}) = \arg\min_{\boldsymbol{\omega}} \sum_{i=1}^{n} \log \left(1 + \exp(-y_i \mathbf{x}_i^\top \boldsymbol{\omega})\right)$$

$$\begin{aligned} \boldsymbol{\omega}_{MAP} &= \arg\max_{\boldsymbol{\omega}} \log p(\mathbf{y} \mid X, \boldsymbol{\omega}) + \log p(\boldsymbol{\omega}) \\ &= \arg\min_{\boldsymbol{\omega}} \sum_{i=1}^{n} \log \left(1 + \exp(-y_i \mathbf{x}_i^\top \boldsymbol{\omega})\right) + \frac{1}{2} \boldsymbol{\omega}^\top \Lambda^{-1} \boldsymbol{\omega} \end{aligned}$$

Discriminative Classification via a Logistic Model

- $\nabla_{\boldsymbol{\omega}} \left(-\log p(\mathbf{y} \mid X, \boldsymbol{\omega})\right) = 0$ does not have a closed-form solution so we need to use an iterative optimization algorithm like gradient descent
- The negative log-likelihood $-\log p(\mathbf{y} \mid X, \boldsymbol{\omega})$ is **convex** in $\boldsymbol{\omega}$:
  - The composition of an affine function $f_i(\boldsymbol{\omega}) := -y_i \mathbf{x}_i^\top \boldsymbol{\omega}$ and a convex function $g(z) := \log(1 + \exp^z)$ is convex
  - The sum of convex functions $\sum_{i=1}^n g(f_i(\boldsymbol{\omega}))$ is convex
- The negative log-likelihood can be minimized iteratively to obtain a **global** minimum:

$$\begin{aligned}
\boldsymbol{\omega}_{MLE}^{(t+1)} &= \boldsymbol{\omega}_{MLE}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\omega}} \left(-\log p(\mathbf{y}|X, \boldsymbol{\omega})\right) \Big|_{\boldsymbol{\omega}=\boldsymbol{\omega}_{MLE}^{(t)}} \\
&= \boldsymbol{\omega}_{MLE}^{(t)} - \alpha^{(t)} \sum_{i=1}^{n} \frac{1}{1 + \exp(-y_i \mathbf{x}_i^\top \boldsymbol{\omega}_{MLE}^{(t)})} \exp(-y_i \mathbf{x}_i^\top \boldsymbol{\omega}_{MLE}^{(t)}) \left(-y_i \mathbf{x}_i\right) \\
&= \boldsymbol{\omega}_{MLE}^{(t)} + \alpha^{(t)} \sum_{i=1}^{n} (1 - \sigma(y_i \mathbf{x}_i^\top \boldsymbol{\omega}_{MLE}^{(t)})) y_i \mathbf{x}_i
\end{aligned}$$

Logistic Regression Summary

- **Discriminative model**: $p(\mathbf{y}|X,\boldsymbol{\omega})$ for binary labels $\mathbf{y} \in \{-1,1\}^n$:

$$p(\mathbf{y}|X,\boldsymbol{\omega}) = \prod_{i=1}^{n} \sigma(y_i \mathbf{x}_i^\top \boldsymbol{\omega}) = \prod_{i=1}^{n} \frac{1}{1+\exp(-y_i \mathbf{x}_i^\top \boldsymbol{\omega})}$$

- **Training**: given data $D = (X, \mathbf{y})$, optimize the model parameters:
  - MLE: $\boldsymbol{\omega}_{MLE}^{(t+1)} = \boldsymbol{\omega}_{MLE}^{(t)} + \alpha^{(t)} \sum_{i=1}^{n} (1 - \sigma(y_i \mathbf{x}_i^\top \boldsymbol{\omega}_{MLE}^{(t)})) y_i \mathbf{x}_i$
  - MAP: $\boldsymbol{\omega}_{MAP}^{(t+1)} = \boldsymbol{\omega}_{MAP}^{(t)} + \alpha^{(t)} \left(\sum_{i=1}^{n} (1 - \sigma(y_i \mathbf{x}_i^\top \boldsymbol{\omega}_{MAP}^{(t)})) y_i \mathbf{x}_i - \Lambda^{-1} \boldsymbol{\omega}_{MAP}^{(t)} \right)$

- **Testing**: given a test example $\mathbf{x}_* \in \mathbb{R}^d$, use the optimized parameters $\boldsymbol{\omega}^*$ to predict the label:

$$y_* = \begin{cases} 1 & \mathbf{x}_*^\top \boldsymbol{\omega}^* \geq 0 \\ -1 & \mathbf{x}_*^\top \boldsymbol{\omega}^* < 0 \end{cases}$$

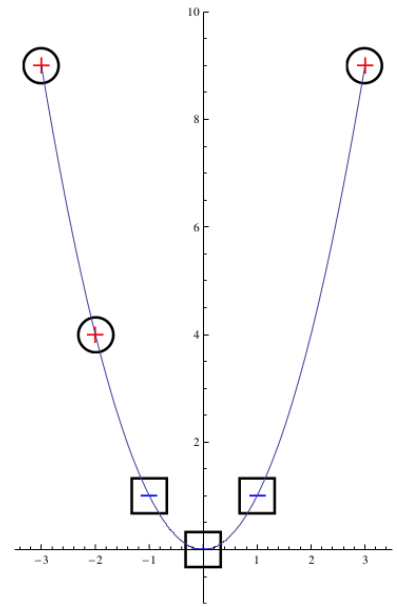
- Logistic regression generates a **linear decision boundary**, i.e., the values of $\mathbf{x}_*$ for which the predictions $y_* = 1$ and $y_* = -1$ are equally likely are:

$$1 = \frac{p(y=1 \mid \mathbf{x}_*, \boldsymbol{\omega}^*)}{p(y=-1 \mid \mathbf{x}_*, \boldsymbol{\omega}^*)} = \frac{1+\exp(\mathbf{x}_*^\top \boldsymbol{\omega}^*)}{1+\exp(-\mathbf{x}_*^\top \boldsymbol{\omega}^*)} \quad \Leftrightarrow \quad \mathbf{x}_*^\top \boldsymbol{\omega}^* = 0$$

Logistic Regression Example

- Consider the same data as before:

$$X = \begin{bmatrix} -3 & 9 & 1 \\ -2 & 4 & 1 \\ -1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ 3 & 9 & 1 \end{bmatrix} \in \mathbb{R}^{n \times d} \qquad \mathbf{y} = \begin{bmatrix} +1 \\ +1 \\ -1 \\ -1 \\ -1 \\ +1 \end{bmatrix} \in \mathbb{R}^{n}$$

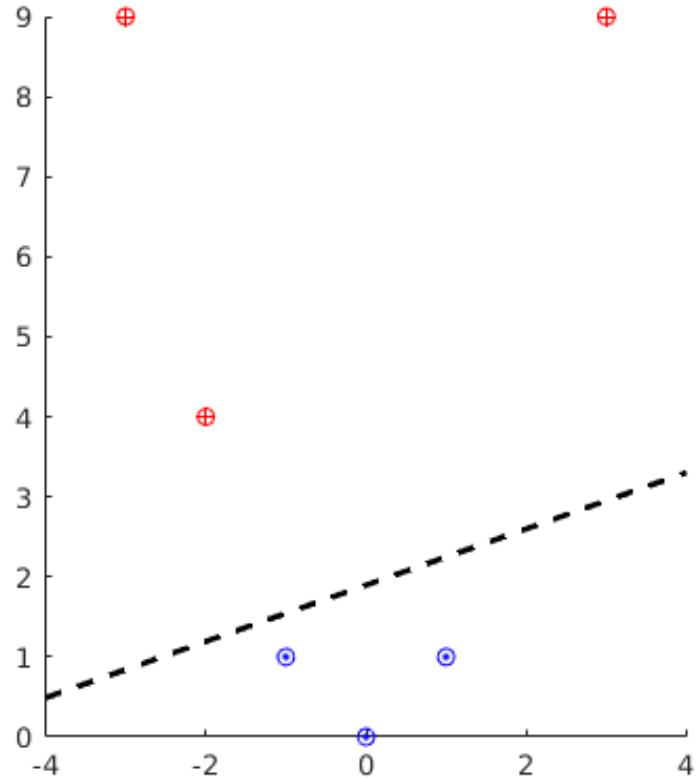


Logistic Regression Example

- **Training**: start with $\boldsymbol{\omega}_{MLE}^{(0)} = \mathbf{0} \in \mathbb{R}^3$ and iterate:

$$\boldsymbol{\omega}_{MLE}^{(t+1)} = \boldsymbol{\omega}_{MLE}^{(t)} + \alpha \sum_{i=1}^{n} (1 - \sigma(y_i \mathbf{x}_i^\top \boldsymbol{\omega}_{MLE}^{(t)})) y_i \mathbf{x}_i$$

- After 10 iterations with $\alpha = 0.1$, we have: $\boldsymbol{\omega}_{MLE}^{(10)} = \begin{bmatrix} -0.2115 \\ 0.6015 \\ -1.1408 \end{bmatrix}$
- **Testing**: the decision boundary is a line with equation $0 = \mathbf{x}^\top \boldsymbol{\omega}$:



Logistic Regression vs Gaussian Naïve Bayes

- Logistic regression generates a **linear decision boundary**: $\boldsymbol{\omega}^\top \mathbf{x} = 0$.
- Gaussian Naïve Bayes generates a **quadratic decision boundary**. It looks like an ellipse, parabola, or hyperbola in 2-D:

$$\log \frac{\theta_-^2}{\theta_+^2} + \sum_{l=1}^{d} \log \frac{\sigma_{+,l}^2}{\sigma_{-,l}^2} + \frac{(x_l - \mu_{+,l})^2}{\sigma_{+,l}^2} - \frac{(x_l - \mu_{-,l})^2}{\sigma_{-,l}^2} = 0$$

- When the variance is shared among the classes, i.e., $\sigma_{-,l} = \sigma_{+,l} = \sigma_l$, Gaussian Naïve Bayes generates a **linear decision boundary**:

$$\log \frac{\theta_-^2}{\theta_+^2} + \sum_{l=1}^{d} \frac{2(\mu_{-,l} - \mu_{+,l})}{\sigma_l^2} x_l + \frac{(\mu_{+,l}^2 - \mu_{-,l}^2)}{\sigma_l^2} = 0$$

- Logistic regression has **lower bias** but **higher variance** than Gaussian Naïve Bayes.

K-ary Logistic Regression

- **Logistic regression with K-classes**: approximates the unknown label-generating pdf for $\mathbf{y} \in \{1, \ldots, K\}^n$ with a softmax function with parameters $W \in \mathbb{R}^{K \times d}$:

$$p(\mathbf{y}|X, W) = \prod_{i=1}^{n} \mathbf{e}_{y_i}^\top \mathbf{s}\left(W\mathbf{x}_i\right) := \prod_{i=1}^{n} \mathbf{e}_{y_i}^\top \frac{\exp\left(W\mathbf{x}_i\right)}{\mathbf{1}^\top \exp\left(W\mathbf{x}_i\right)}$$

where $\mathbf{e}_j$ is the j-th standard basis vector, e.g., $\mathbf{e}_1 := \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^\top$

- To optimize the parameters W via MLE, we need to compute the gradient of the data log-likelihood:

$$\begin{aligned} W_{MLE}^{(t+1)} &= W_{MLE}^{(t)} + \alpha \left(\nabla_W \left[\log p(\mathbf{y} \mid X, W) \right] \Big|_{W = W_{MLE}^{(t)}} \right) \\ &= W_{MLE}^{(t)} + \alpha \left(\sum_{i=1}^{n} \left(\mathbf{e}_{y_i} - \mathbf{s}(W_{MLE}^{(t)} \mathbf{x}_i) \right) \mathbf{x}_i^\top \right) \end{aligned}$$

K-ary Logistic Regression MLE

- Taking the gradient of $\log p(\mathbf{y} \mid X, W)$ with respect to W gives:

$$\nabla_W \left[\sum_{i=1}^{n} \log \mathbf{e}_{y_i}^\top \mathbf{s}(W\mathbf{x}_i) \right] = \sum_{i=1}^{n} \frac{1}{\mathbf{e}_{y_i}^\top \mathbf{s}(W\mathbf{x}_i)} \nabla_W \left[\mathbf{e}_{y_i}^\top \mathbf{s}(W\mathbf{x}_i) \right]$$

- Let $\mathbf{z}_i := W\mathbf{x}_i$. The derivative of the last term via the chain rule is:

$$\frac{\partial}{\partial \mathbf{z}_i} \mathbf{e}_{y_i}^\top \mathbf{s}(\mathbf{z}_i) \frac{\partial}{\partial W} W\mathbf{x}_i = \frac{\partial}{\partial W} \mathbf{e}_{y_i}^\top \frac{\partial \mathbf{s}(\mathbf{z}_i)}{\partial \mathbf{z}_i} W\mathbf{x}_i = \frac{\partial}{\partial W} \operatorname{tr}\left(\mathbf{x}_i \mathbf{e}_{y_i}^\top \frac{\partial \mathbf{s}(\mathbf{z}_i)}{\partial \mathbf{z}_i} W \right)$$
$$= \mathbf{x}_i \mathbf{e}_{y_i}^\top \frac{\partial \mathbf{s}(\mathbf{z}_i)}{\partial \mathbf{z}_i} = \mathbf{x}_i \mathbf{e}_{y_i}^\top \frac{\partial \mathbf{s}}{\partial \mathbf{z}_i}(W\mathbf{x}_i)$$

- Arrange the derivatives of the softmax in matrix form:

$$\frac{d\mathbf{s}(\mathbf{z})}{d\mathbf{z}} = \begin{bmatrix} s_1 - s_1^2 & -s_1 s_2 & \cdots & -s_1 s_d \\ -s_2 s_1 & s_2 - s_2^2 & \cdots & -s_2 s_d \\ \vdots & & \ddots & \vdots \\ -s_d s_1 & -s_d s_2 & \cdots & s_d - s_d^2 \end{bmatrix} = \mathbf{diag}(\mathbf{s}(\mathbf{z})) - \mathbf{s}(\mathbf{z})\mathbf{s}(\mathbf{z})^\top$$

K-ary Logistic Regression MLE

- The gradient is the transpose of the derivative:

$$\begin{aligned}\nabla_W \left[\log p(\mathbf{y} \mid X, W)\right] &= \sum_{i=1}^{n} \frac{1}{\mathbf{e}_{y_i}^\top \mathbf{s}(W\mathbf{x}_i)} (\mathbf{diag}(\mathbf{s}(W\mathbf{x}_i)) - \mathbf{s}(W\mathbf{x}_i)\mathbf{s}(W\mathbf{x}_i)^\top)\mathbf{e}_{y_i}\mathbf{x}_i^\top \\ &= \sum_{i=1}^{n} \left(\mathbf{e}_{y_i} - \mathbf{s}(W\mathbf{x}_i)\right)\mathbf{x}_i^\top \in \mathbb{R}^{K\times d}\end{aligned}$$

- MLE leads to gradient ascent of the form:

$$W_{MLE}^{(t+1)} = W_{MLE}^{(t)} + \alpha \left(\sum_{i=1}^{n} \left(\mathbf{e}_{y_i} - \mathbf{s}(W_{MLE}^{(t)}\mathbf{x}_i) \right) \mathbf{x}_i^\top \right)$$

- MAP with $W \sim \mathcal{N}(0, \lambda I_{Kd\times Kd})$ leads to gradient ascent of the form:

$$W_{MAP}^{(t+1)} = W_{MAP}^{(t)} + \alpha \left(\sum_{i=1}^{n} \left(\mathbf{e}_{y_i} - \mathbf{s}(W_{MAP}^{(t)}\mathbf{x}_i) \right) \mathbf{x}_i^\top \right) - \alpha\lambda^{-1} W_{MAP}^{(t)}$$