Multiclass Classification

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-635

10/11/2023

Special Thanks: Sargur N. Srihari, Andrew Ng, Carlos Guestrin

Some Forms of Regularization

Regularization function	**Name**	**Solver**
$\|\theta\|_2^2 = \sum_{j=1}^{n} \theta_j^2$	Tikhonov regularization Ridge regression	Closed form
$\|\|\theta\|\|_1 = \sum_{j=1}^{n} \|\theta_j\|$	LASSO regression	Proximal gradient descent, least angle regression
$\alpha\|\|\theta\|\|_1 + (1-\alpha)\ \|\theta\|_2^2$	Elastic net regularization	Proximal gradient descent

How about MAP?

- **Maximum (conditional) likelihood estimate (MCLE)**

$$\boxed{\theta_{\mathrm{MCLE}} = \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^{m} P_\theta\left(y^{(i)} | x^{(i)}\right)}$$

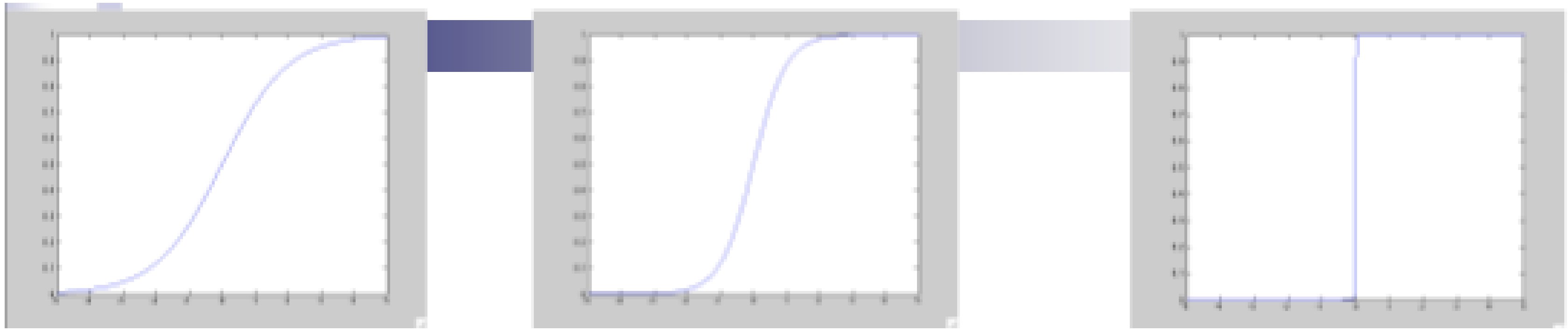
- **Maximum (conditional) a posterior estimate (MCAP)**

$$\boxed{\theta_{\mathrm{MCAP}} = \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^{m} P_\theta\left(y^{(i)} | x^{(i)}\right) P(\theta)}$$

Prior $P(\theta)$

- Common choice of $P(\theta)$:
  - Normal distribution, zero mean, identity covariance
  - “Pushes” parameters towards zeros
- Corresponds to **Regularization**
  - Helps avoid very large weights and overfitting

Notice the “saturation” effect on logistic link1



$$\frac{1}{1+e^{-x}} \qquad \frac{1}{1+e^{-2x}} \qquad \frac{1}{1+e^{-100x}}$$

MLE vs. MAP

- **Maximum (conditional) likelihood estimate (MCLE)**

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

- **Maximum (conditional) a posterior estimate (MCAP)**

$$\theta_j := \theta_j - \alpha\lambda\theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

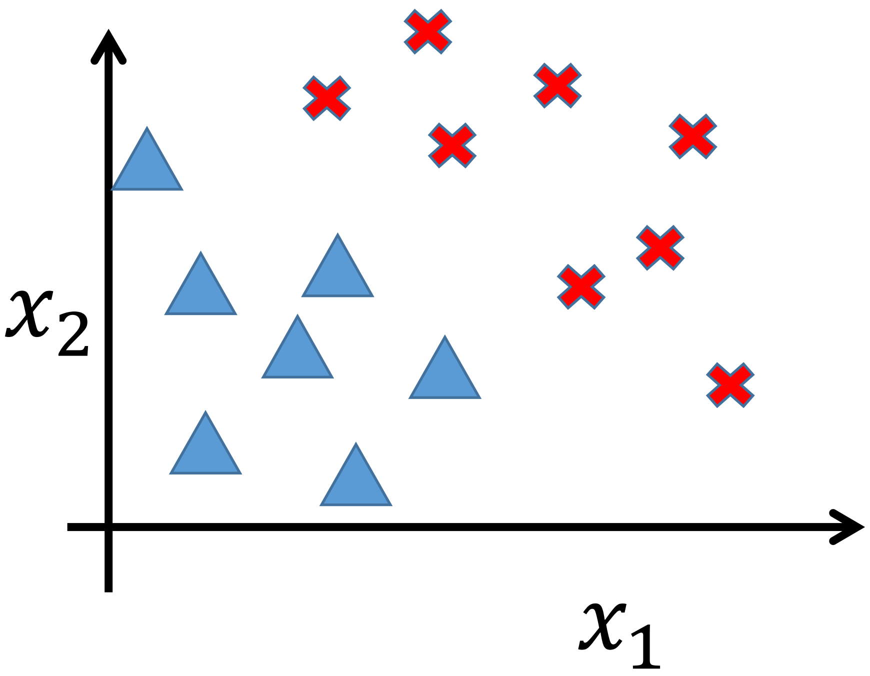
Logistic Regression

- Hypothesis representation
- Cost function
- Logistic regression with gradient descent
- Regularization
- **Multi-class classification**

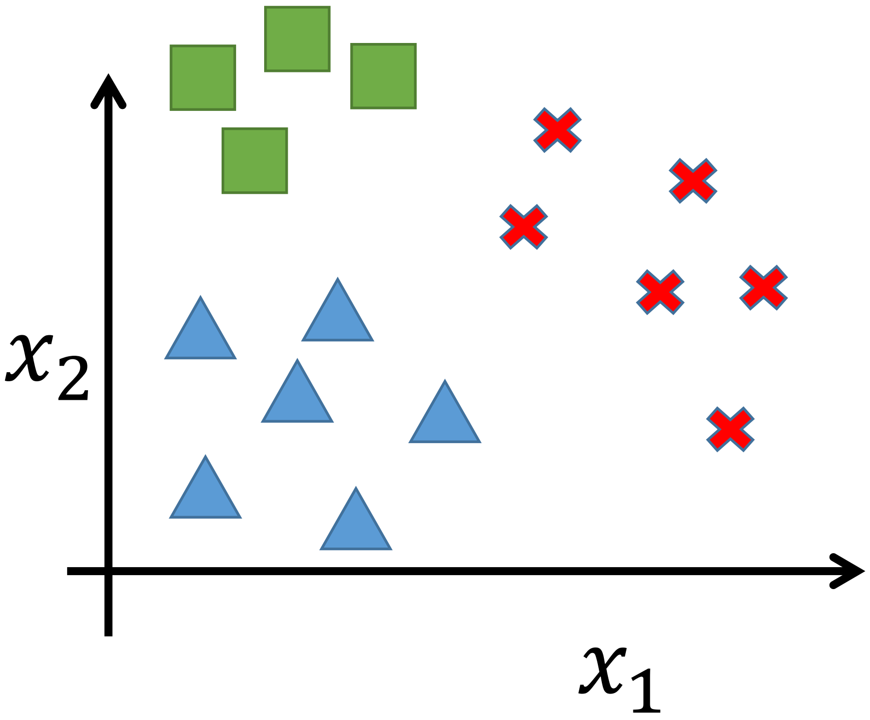
Multi-Class Classification

- ***What if we have more than two classes/categories/conditions?***
- Example scenarios:
  - Email foldering/tagging: Work, Friends, Family, Hobby
  - Medical diagrams: Not ill, Cold, Flu
  - Weather: Sunny, Cloudy, Rain, Snow

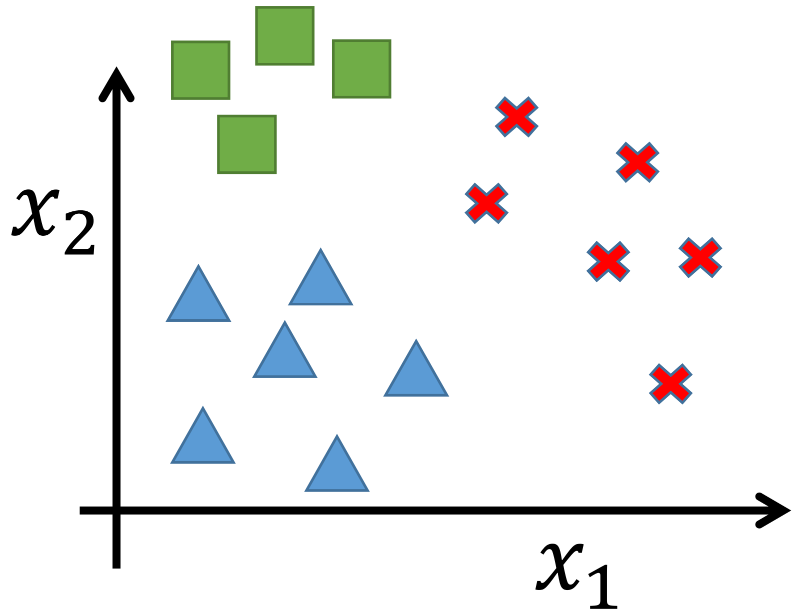
Binary classification



Multiclass classification



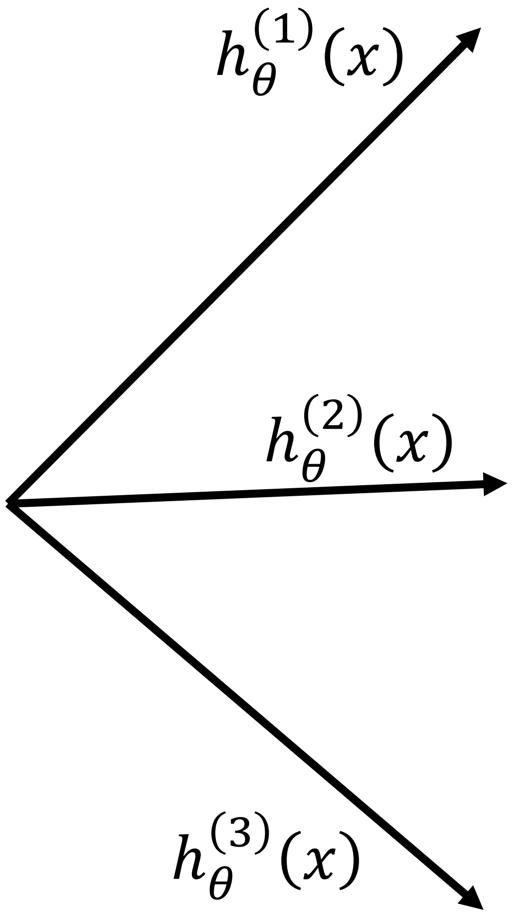
One-vs-All (One-vs-Rest)

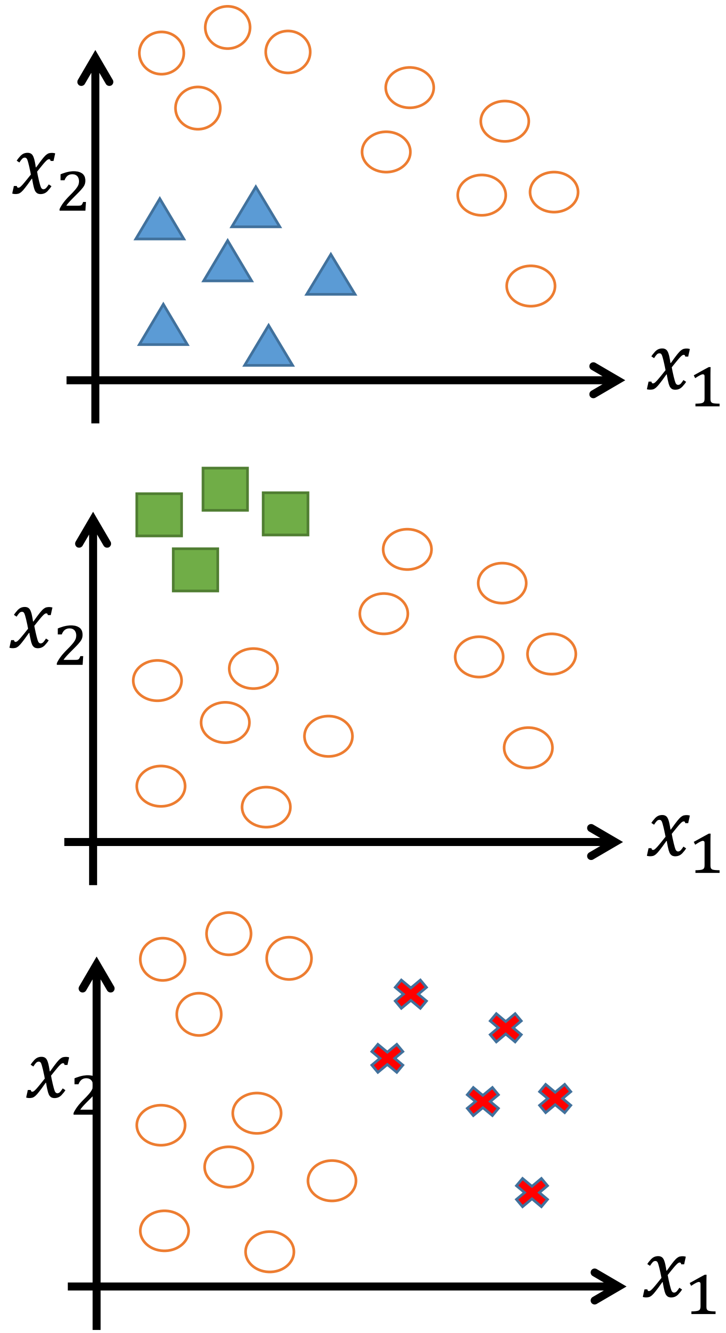


Class 1: △

Class 2: ■

Class 3: ✖





$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1, 2, 3)$$

One-vs-All

- Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$

- Given a new input x, pick the class i that maximizes

$$\max_{\mathrm{i}} h_\theta^{(i)}(x)$$

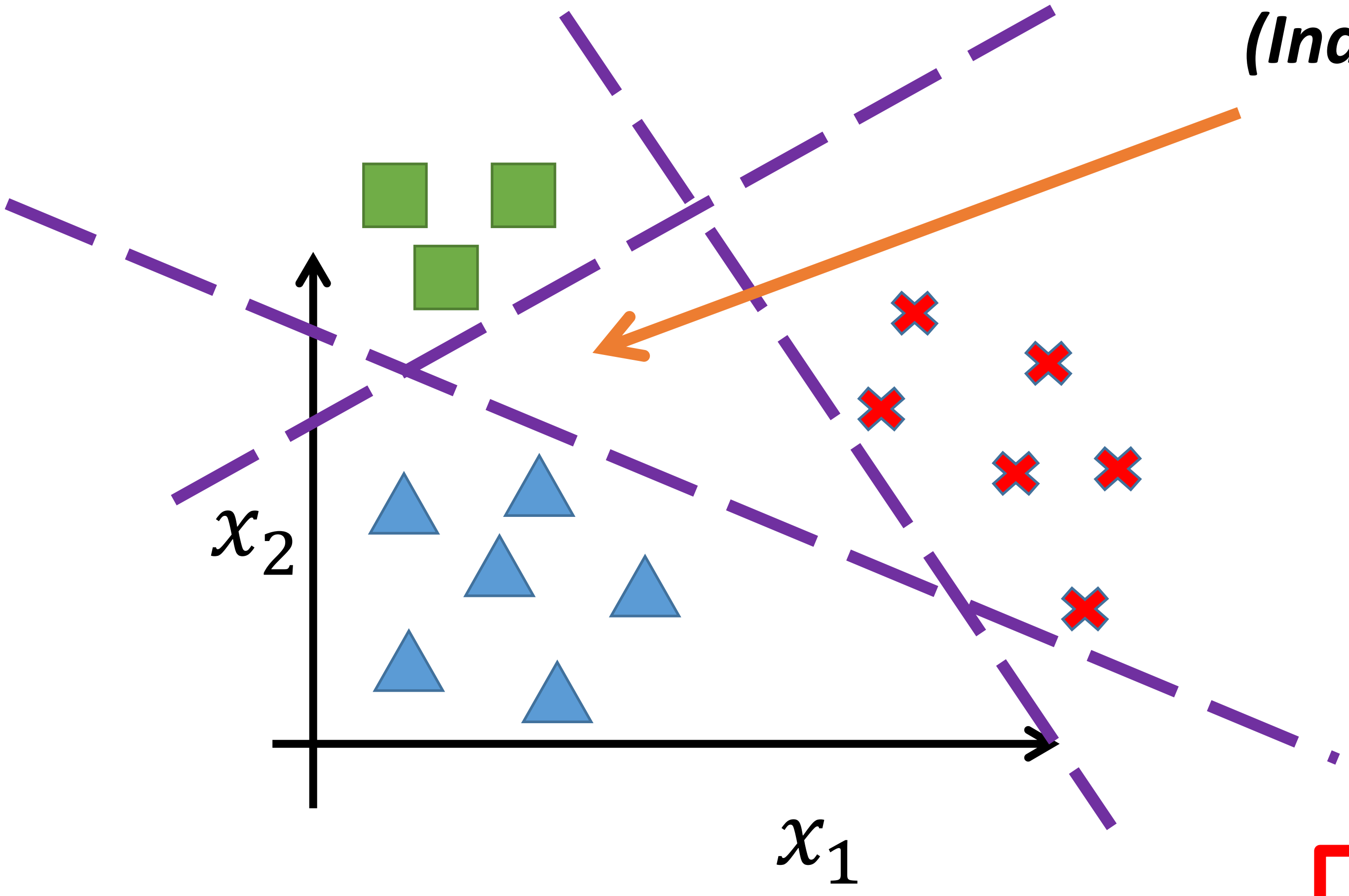
One-vs-All

- Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$

- Given a new input x, pick the class i that maximizes

$$\max_{\mathrm{i}} h_\theta^{(i)}(x)$$

Problem:
Indeterminacy



What class does this zone belong to? (Indeterminate region)

Problem: Indeterminacy

Discriminative Approach

Ex: Logistic regression, multinoulli regression

Estimate $P(Y|X)$ directly

(Or a discriminant function: e.g., a linear classifier or support vector machine)

Prediction (mode):
$\hat{y} = P(Y = y|X = x)$

Generative Approach

Ex: Naïve Bayes

Estimate $P(Y)$ and $P(X|Y)$

Prediction (mode):
$\hat{y} = \text{argmax}_y\, P(Y = y)P(X = x|Y = y)$

Discriminative Approach

Ex: Logistic regression, multinoulli regression

Estimate $P(Y|X)$ directly

(Or a discriminant function: e.g., a linear classifier or support vector machine)

Prediction (mode):
$\hat{y} = P(Y = y|X = x)$

Generative Approach

Ex: Naïve Bayes

Estimate $P(Y)$ and $P(X|Y)$

This is coming up soon…

Prediction (mode):
$\hat{y} = \text{argmax}_y \, P(Y = y)P(X = x|Y = y)$

Things to Remember

- Hypothesis representation $h_\theta(x) = \dfrac{1}{1 + e^{-\theta^\top x}}$
- Cost function $\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$
- Logistic regression with gradient descent
  $$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$
- Regularization
  $$\theta_j := \theta_j - \alpha\lambda\theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$
- Multi-class classification
  $$\max_{\mathrm{i}} h_\theta^{(i)}(x)$$

So, even though we have a way of conducting multi-class classification, we still could ask

CAN WE GO FURTHER…BEYOND?

The Multinoulli Distribution

- Bernoulli distribution
  - k =2 outcomes, m = 1 trial
- Binomial distribution
  - k = 2 outcomes, m >= 1 trial
- ***Multinoulli (Categorical) distribution***
  - k >=2 outcomes, m = 1 trial
- Multinomial distribution
  - k >= 2 outcomes, m >= 1 trial

PDF/PMF:

$$\prod_{i=1}^{k} p_i^{x_i} \qquad (\text{where } 0 \leq p_i \text{ and } \sum_i p_i = 1)$$

over the support: $x_i \in \{0, 1\}$

where: $n \triangleq \sum_{i=1}^{k} x_i = 1$

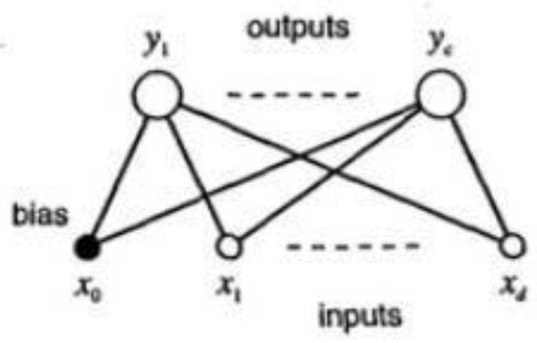
Equivalent by definition

Multinoulli (Softmax) Regression

$$z = w_0x_0 + w_1x_1 + \ldots + w_mx_m = \sum_{l=0}^{m} w_l x_l = \mathbf{w}^T\mathbf{x}.$$

θ_j^T

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$



Also known as:
Categorical regression
Maximum entropy (classifier)

```python
def softmax(X):
    exps = np.exp(X)
    return exps / np.sum(exps)
```

Note: Full generalization of logistic regression to handle Categorical distributions (over discrete categories/classes)

Multinoulli: Connection to Logistic Regression

1) Derivation of 2-Class multinoulli regressor in terms of logistic regression!

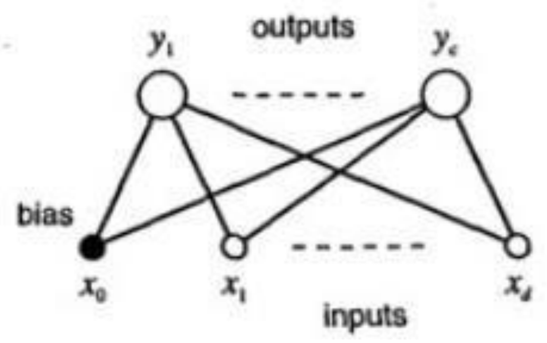
2) Derivation of the multinoulli (log) likelihood in terms of Bernoulli (log) likelihood!

Multinoulli (Softmax) Regression

$$z = w_0 x_0 + w_1 x_1 + \ldots + w_m x_m = \sum_{l=0}^{m} w_l x_l = \mathbf{w}^T \mathbf{x}.$$

θ_j^T

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$



$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}(1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) + y^{(i)} \log h_\theta(x^{(i)})\right]$$

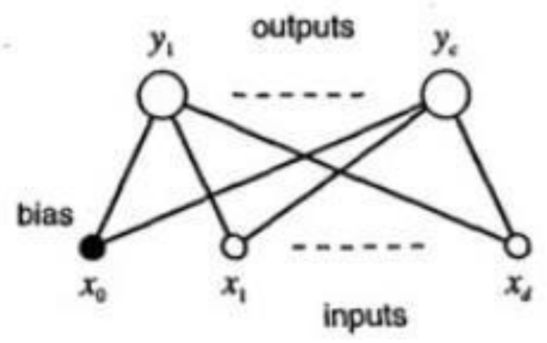
$$= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{1} 1\left\{y^{(i)} = j\right\} \log p(y^{(i)} = j | x^{(i)}; \theta)\right]$$

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = j\} - p(y^{(i)} = j | x^{(i)}; \theta)\right)\right]$$

Multinoulli (Softmax) Regression

$$z = w_0x_0 + w_1x_1 + \ldots + w_mx_m = \sum_{l=0}^{m} w_lx_l = \mathbf{w}^T\mathbf{x}.$$

θ_j^T



$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\}\log\frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}(1-y^{(i)})\log(1-h_\theta(x^{(i)})) + y^{(i)}\log \boxed{h_\theta(x^{(i)})}\right]$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{1} 1\left\{y^{(i)} = j\right\}\log p(y^{(i)} = j|x^{(i)};\theta)\right]$$

$$\boxed{p(y^{(i)} = j|x^{(i)};\theta)}$$

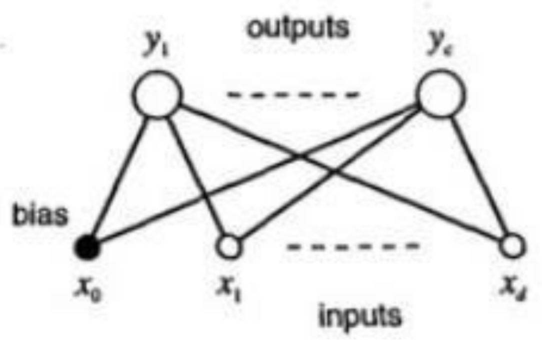
Again, the connection to logistic regression…

$$\nabla_{\theta_j}J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = j\} - p(y^{(i)} = j|x^{(i)};\theta)\right)\right]$$

Multinoulli (Softmax) Regression

$$z = w_0x_0 + w_1x_1 + \ldots + w_mx_m = \sum_{l=0}^{m} w_lx_l = \mathbf{w}^T\mathbf{x}.$$

θ_j^T



$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

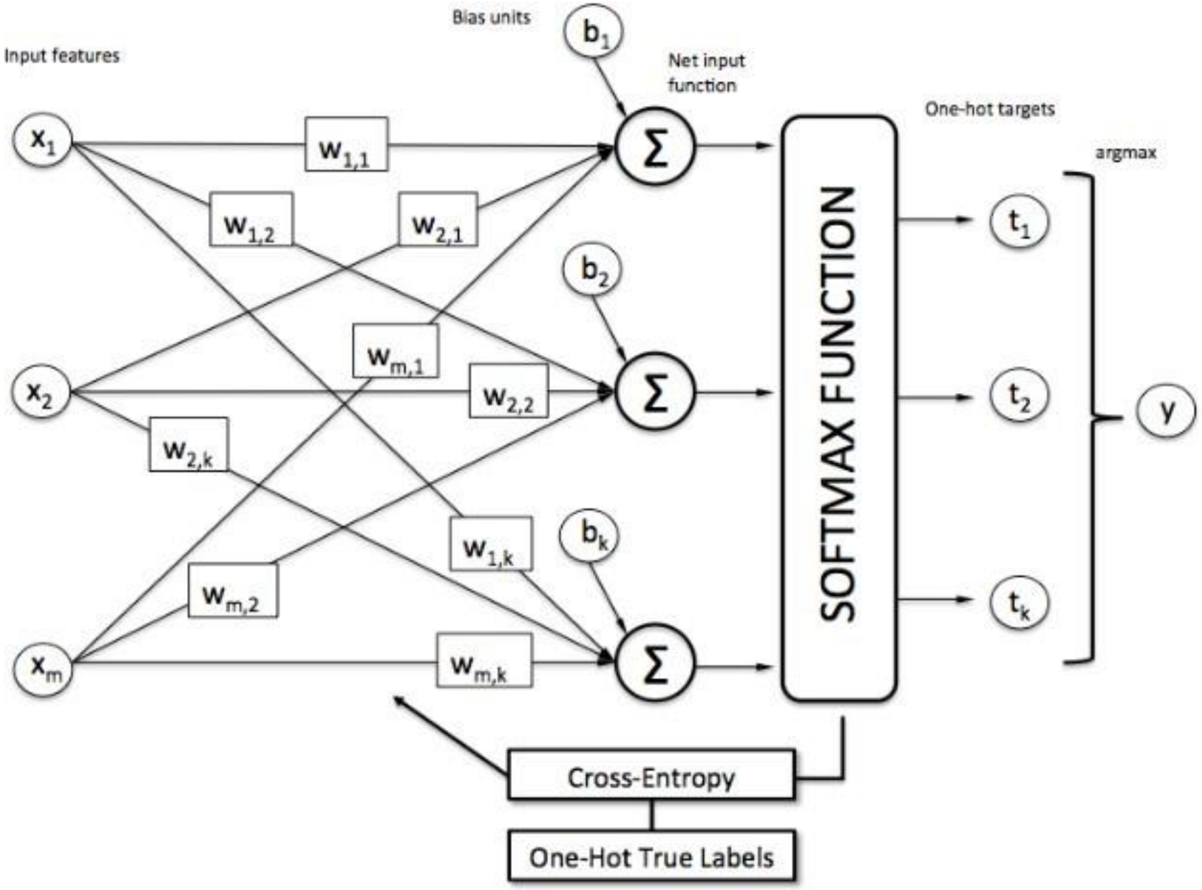
$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}(1 - y^{(i)})\log(1 - h_\theta(x^{(i)})) + y^{(i)}\log h_\theta(x^{(i)})\right]$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{1} 1\left\{y^{(i)} = j\right\}\log p(y^{(i)} = j|x^{(i)};\theta)\right]$$

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = j\} - p(y^{(i)} = j|x^{(i)};\theta)\right)\right]$$

Derivation is left as an exercise for you ;-)

Multinoulli Regressor: Architecture



Multinoulli Regressor: Architecture

We will learn parameters by minimizing the negative log likelihood of our model's predictive posterior. If we say that $\mathbf{p}$ is our model's vector of normalized output probabilities, we define the negative log loss (cost) as follows:

$$\mathbf{p}_k = \frac{e^{\mathbf{f}_k}}{\sum_j e^{\mathbf{f}_j}}, \qquad \mathcal{J} = -\frac{1}{N}\sum_i \Big(\log(\mathbf{p}_{y_i})\Big) \boxed{+\frac{\lambda}{2}\sum_d\sum_k (W_{d,k})^2}$$

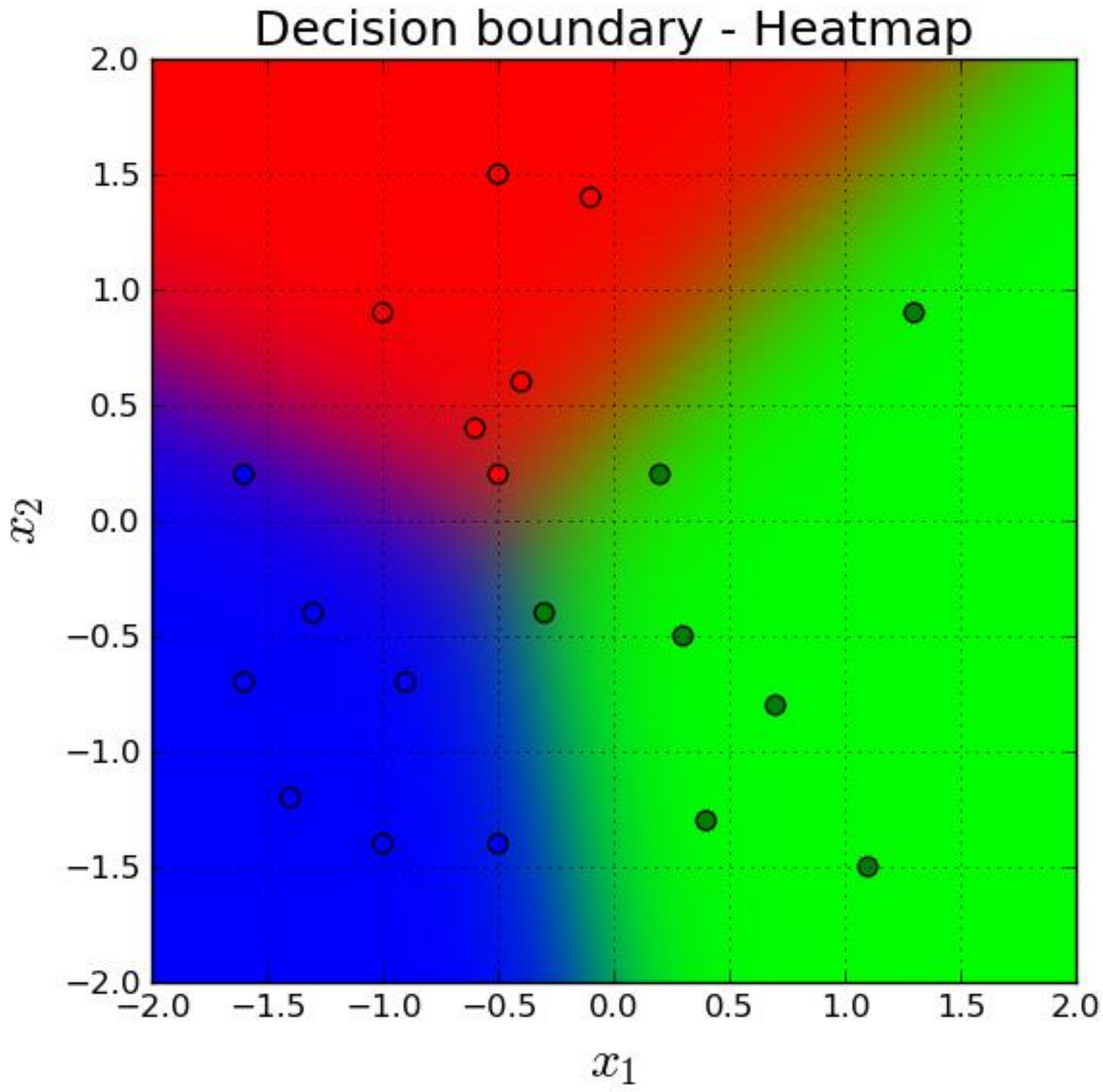
```
Inputs X:
 [[ 0.1  0.5]
 [ 1.1  2.3]
 [-1.1 -2.3]
 [-1.5 -2.5]]

[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 0.  0.  1.]
 [ 0.  0.  1.]]
```

$$\boxed{\frac{\partial \mathcal{J}_i}{\partial \mathbf{f}_k} = \mathbf{p}_k - \mathbf{1}_{y_i=k}}$$

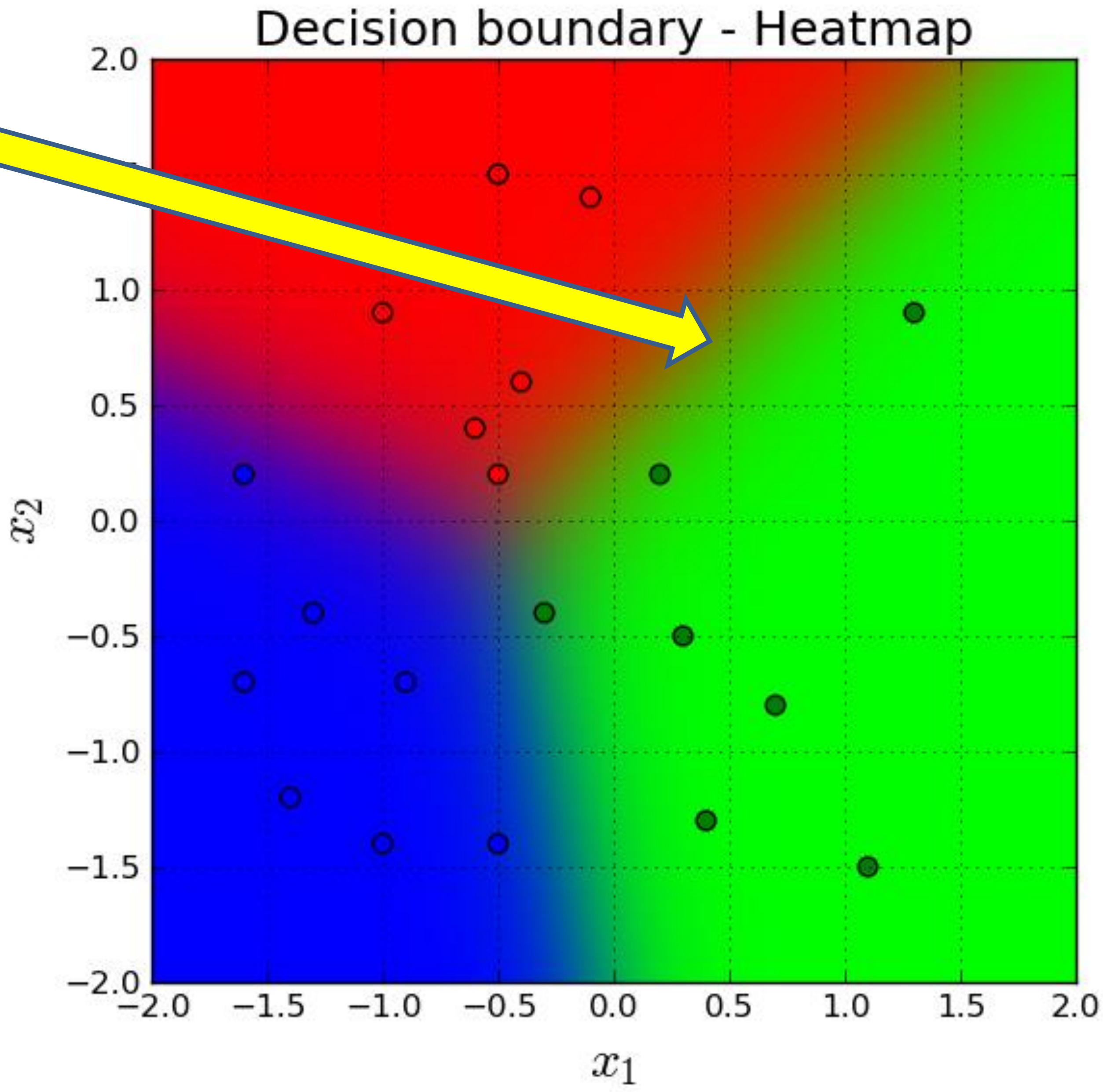
Recall: what is this?

$$\frac{\partial \mathcal{J}}{\partial W} = \mathbf{X}^T * \left(\frac{1}{N}\frac{\partial \mathcal{J}}{\partial \mathbf{f}_k}\right) + \lambda(W) = \mathbf{X}^T * \left(\frac{1}{N}(\mathbf{p}_k - \mathbf{1}_{y_i=k})\right) + \lambda(W)$$

$$\frac{\partial \mathcal{J}}{\partial b} = \sum_{n=0}^{N}\frac{1}{N}\frac{\partial \mathcal{J}}{\partial \mathbf{f}_k} = \sum_{n=0}^{N}\frac{1}{N}(\mathbf{p}_k - \mathbf{1}_{y_i=k})$$

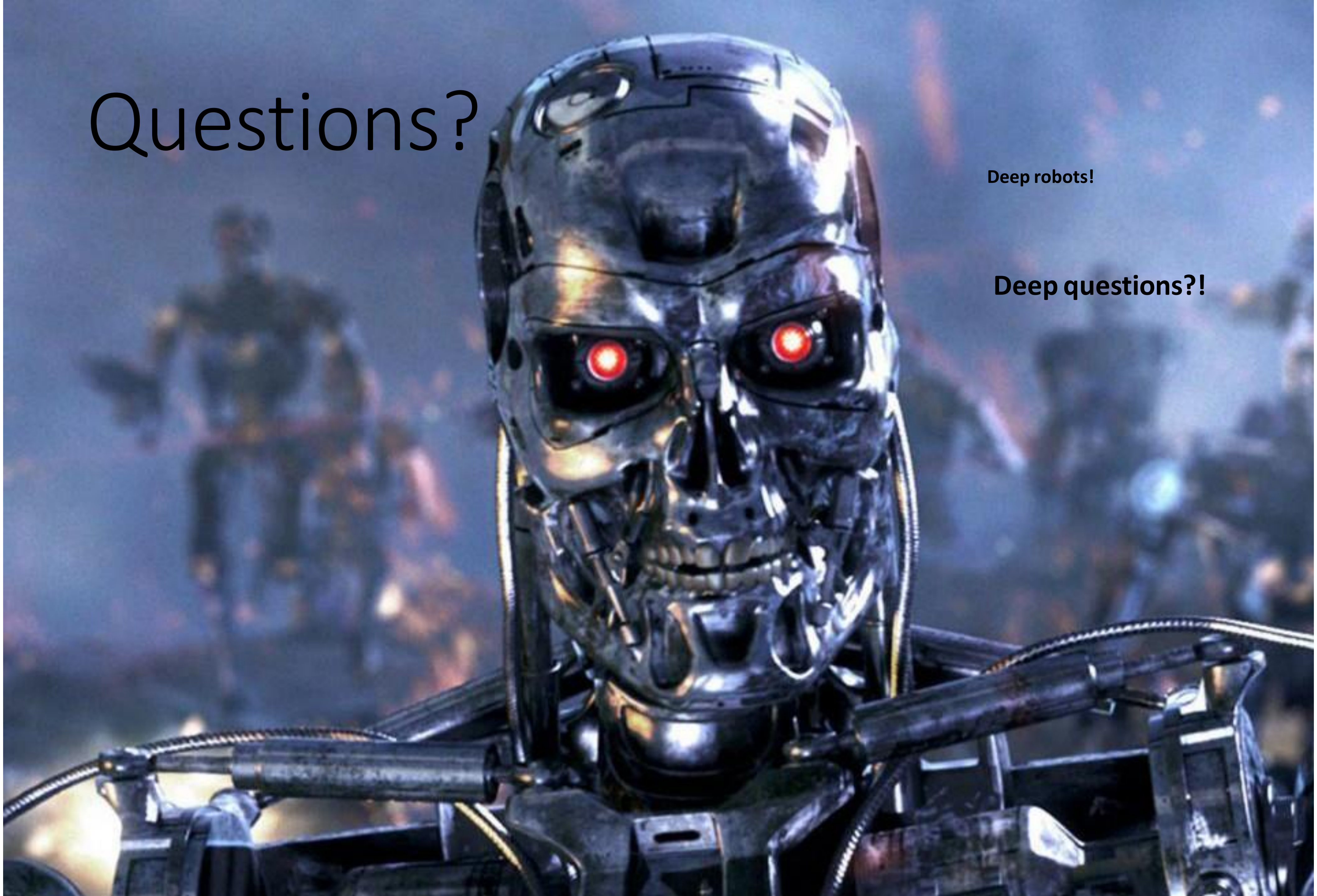

Decision boundary - Heatmap
x_1
x_2
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

The boundaries



Notice that the data is linearly separable!

Questions?



Deep robots!

Deep questions?!