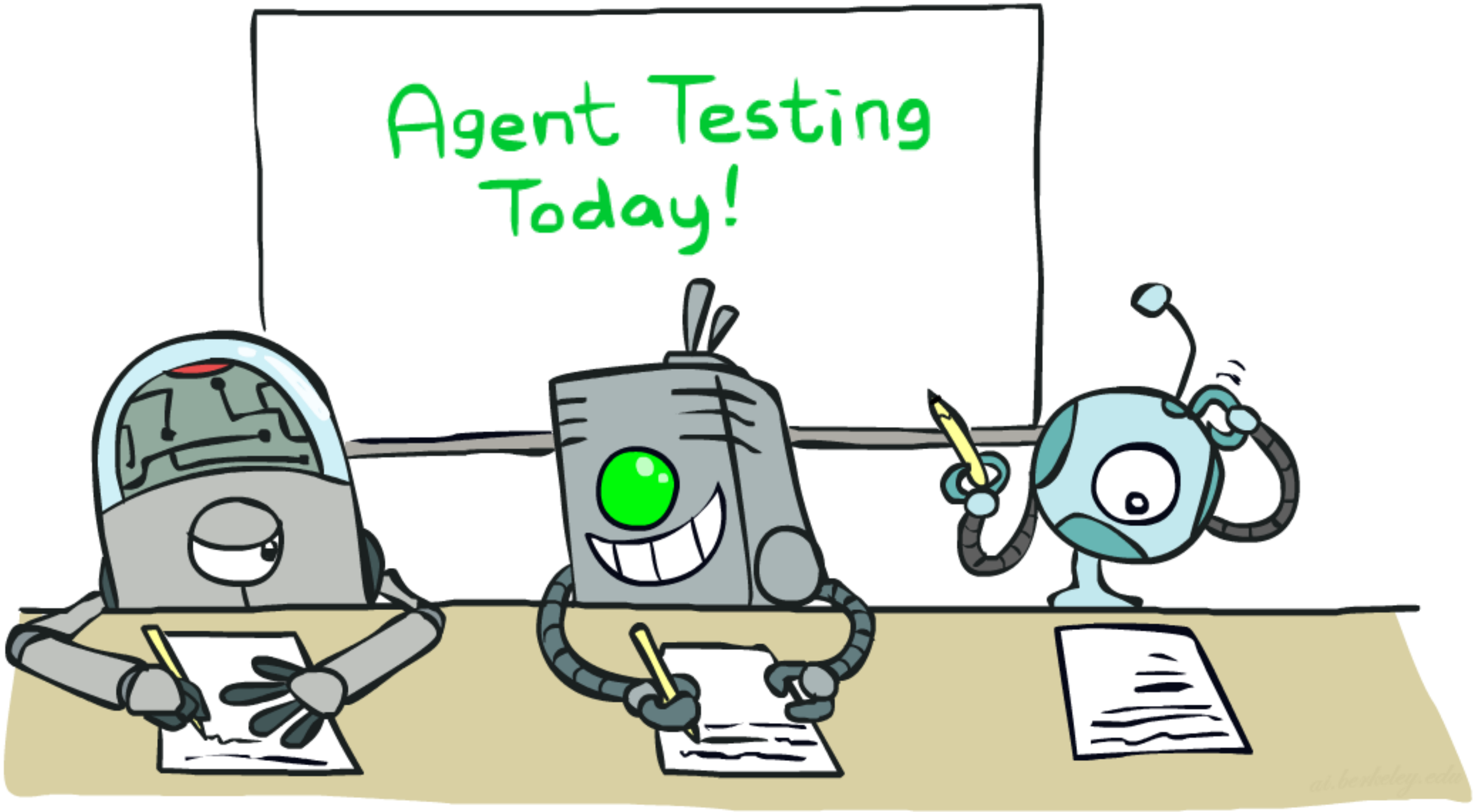
CS 188: Artificial Intelligence

Logistic Regression & Regularization



Summer 2024: Eve Fleisig & Evgeny Pobachienko

[Slides based on those by Nicholas Tomlin, Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. CS188 materials are available at http://ai.berkeley.edu.]

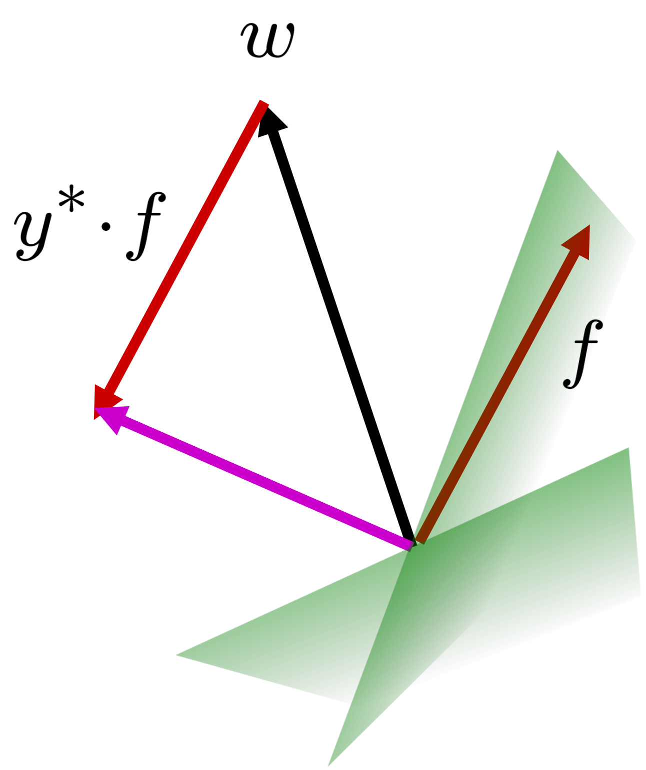
Learning: Binary Perceptron

- Start with weights = 0
- For each training instance:
  - Classify with current weights

$$y = \begin{cases} +1 & \text{if } \; w \cdot f(x) \geq 0 \\ -1 & \text{if } \; w \cdot f(x) < 0 \end{cases}$$

  - If correct (i.e., y=y*), no change!
  - If wrong: adjust the weight vector by adding or subtracting the feature vector. Subtract if y* is -1.

$$w = w + y^* \cdot f$$



Learning: Multiclass Perceptron

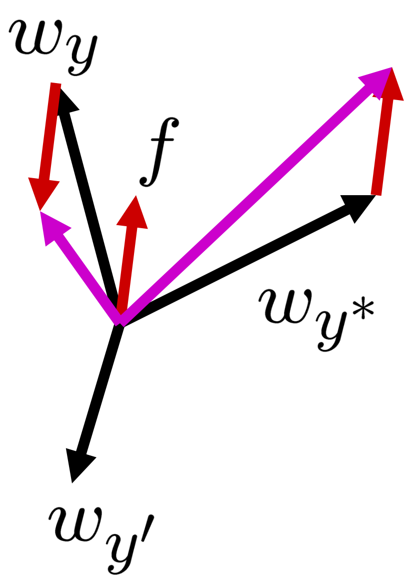
- Start with all weights = 0
- Pick up training examples one by one
- Predict with current weights

$$y = \arg\max_y \; w_y \cdot f(x)$$

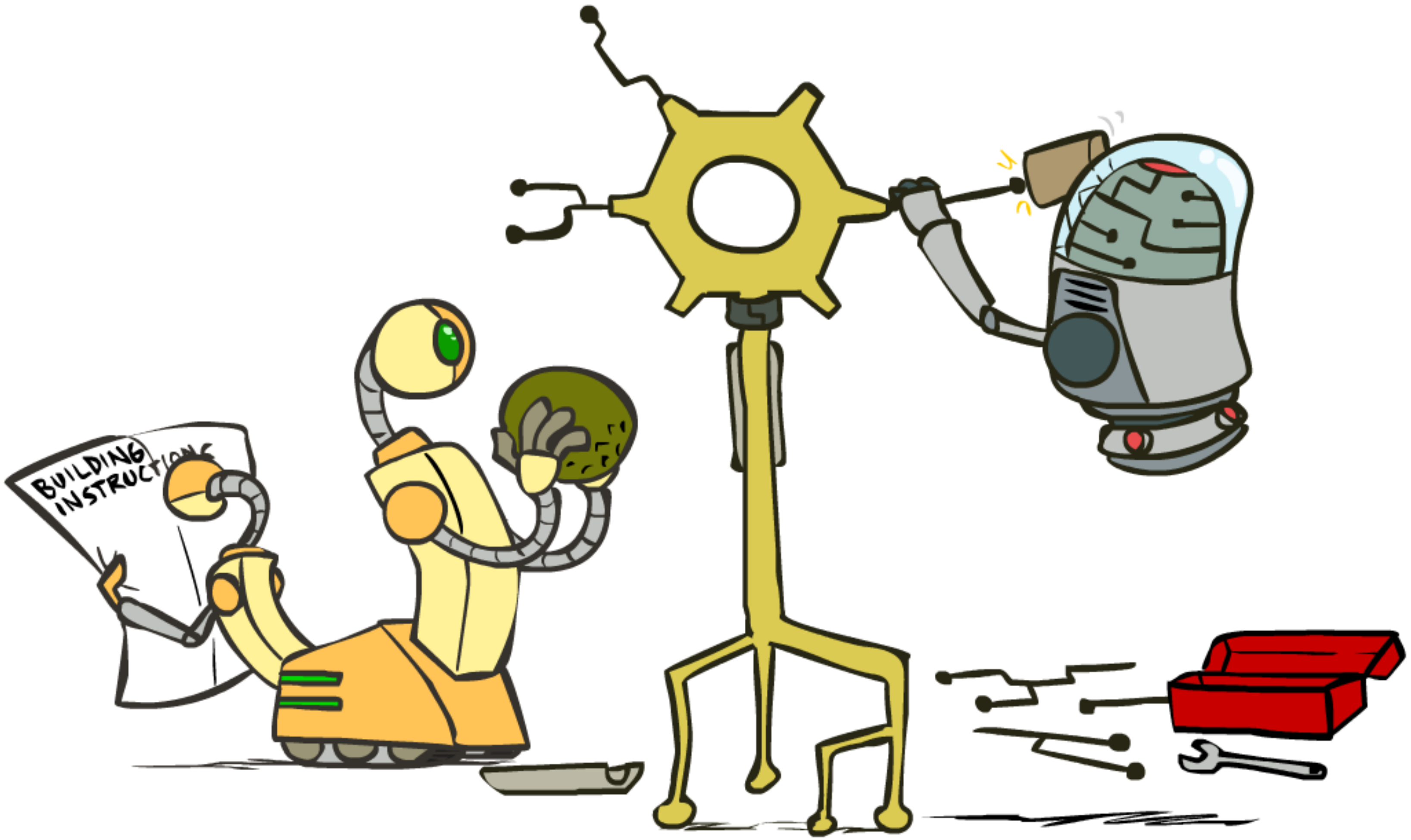
- If correct, no change!
- If wrong: lower score of wrong answer, raise score of right answer

$$w_y = w_y - f(x)$$

$$w_{y^*} = w_{y^*} + f(x)$$

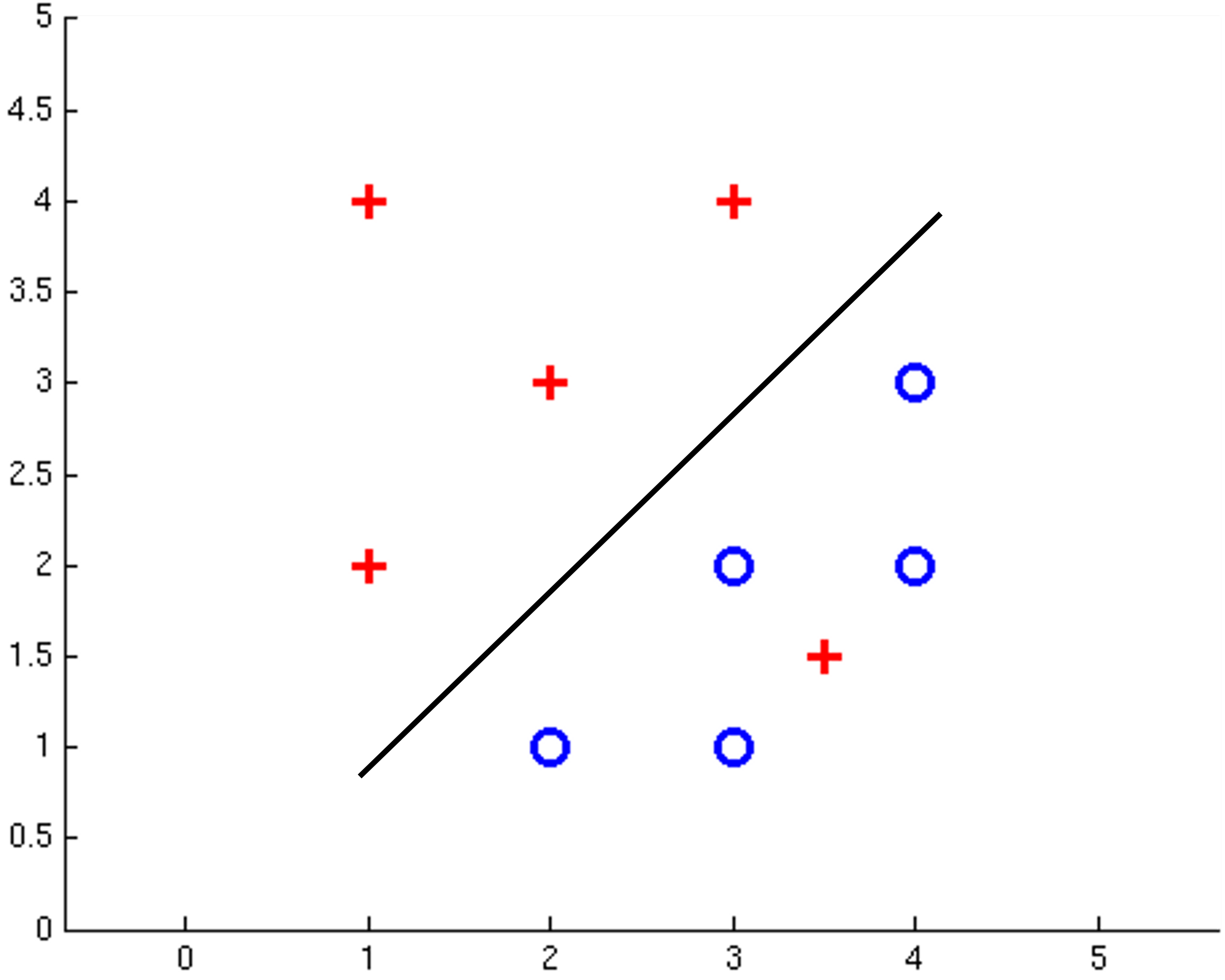


Improving the Perceptron: Logistic Regression

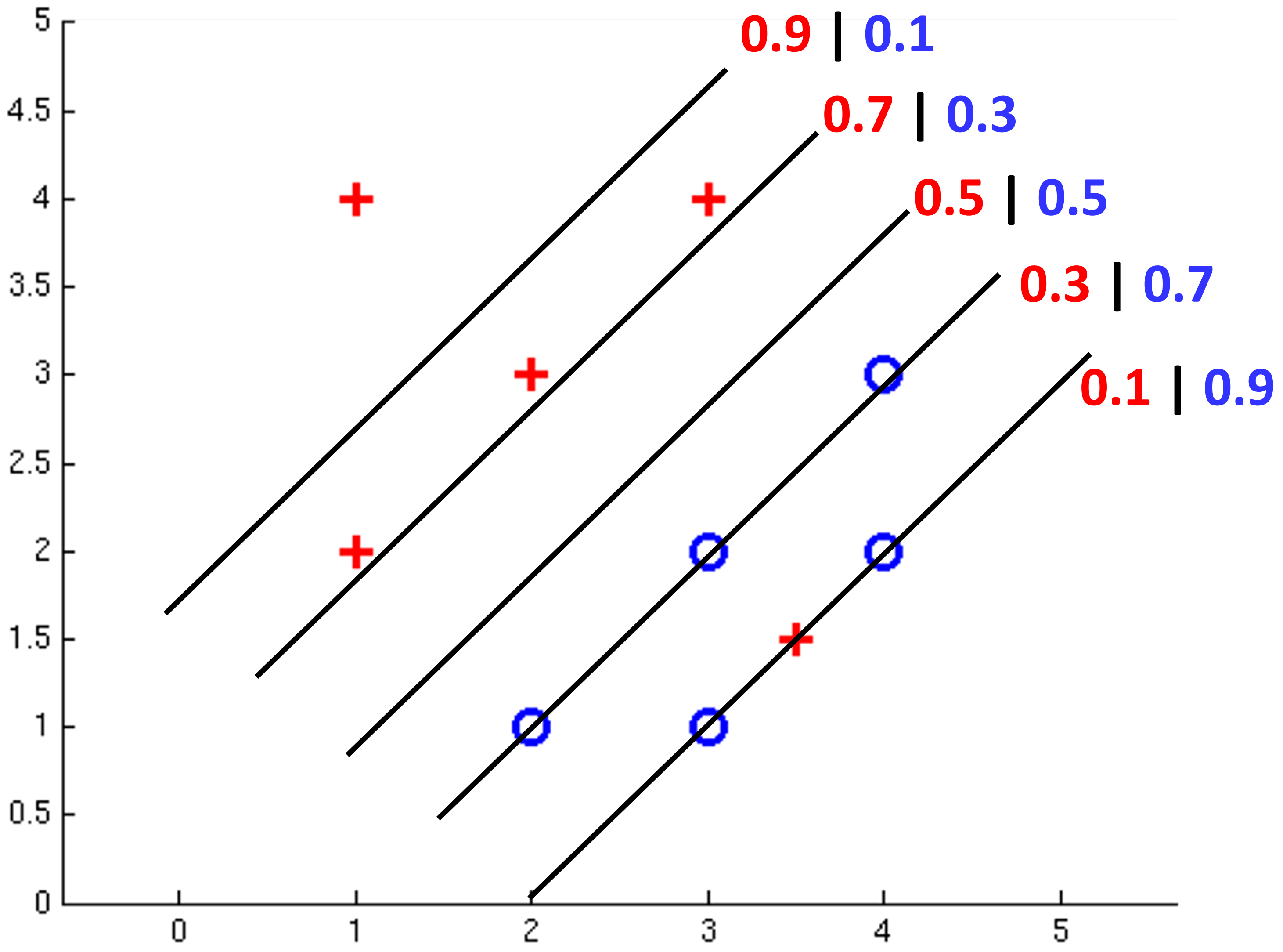


Non-Separable Case: Deterministic Decision

Even the best linear boundary makes at least one mistake



Non-Separable Case: Probabilistic Decision

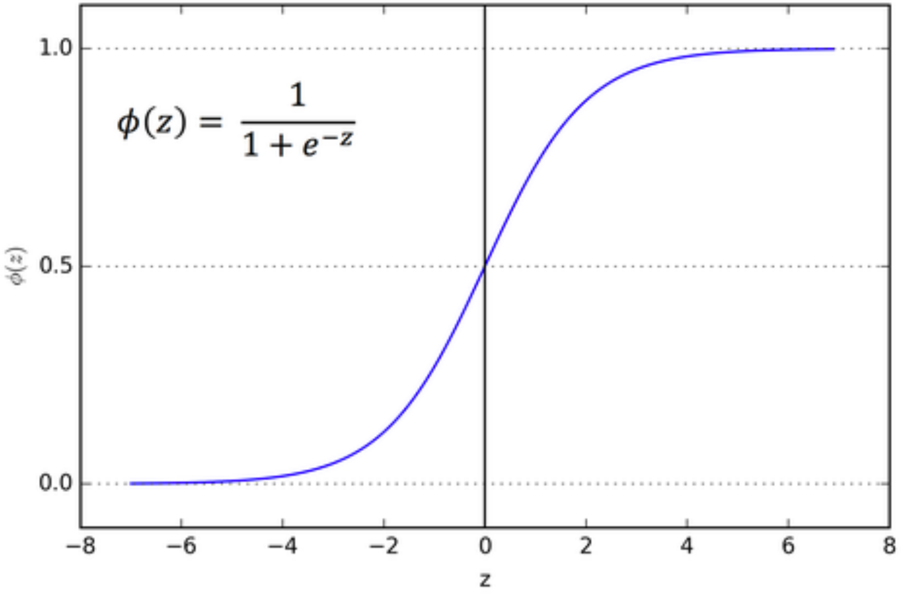


How to get probabilistic decisions?

- Perceptron scoring: $z = w \cdot f(x)$
- If $z = w \cdot f(x)$ very positive → want probability going to 1
- If $z = w \cdot f(x)$ very negative → want probability going to 0

- Sigmoid function

$$\phi(z) = \frac{1}{1 + e^{-z}}$$



Best w?

- Maximum likelihood estimation:

$$\max_w \quad ll(w) = \max_w \quad \sum_i \log P(y^{(i)}|x^{(i)}; w)$$

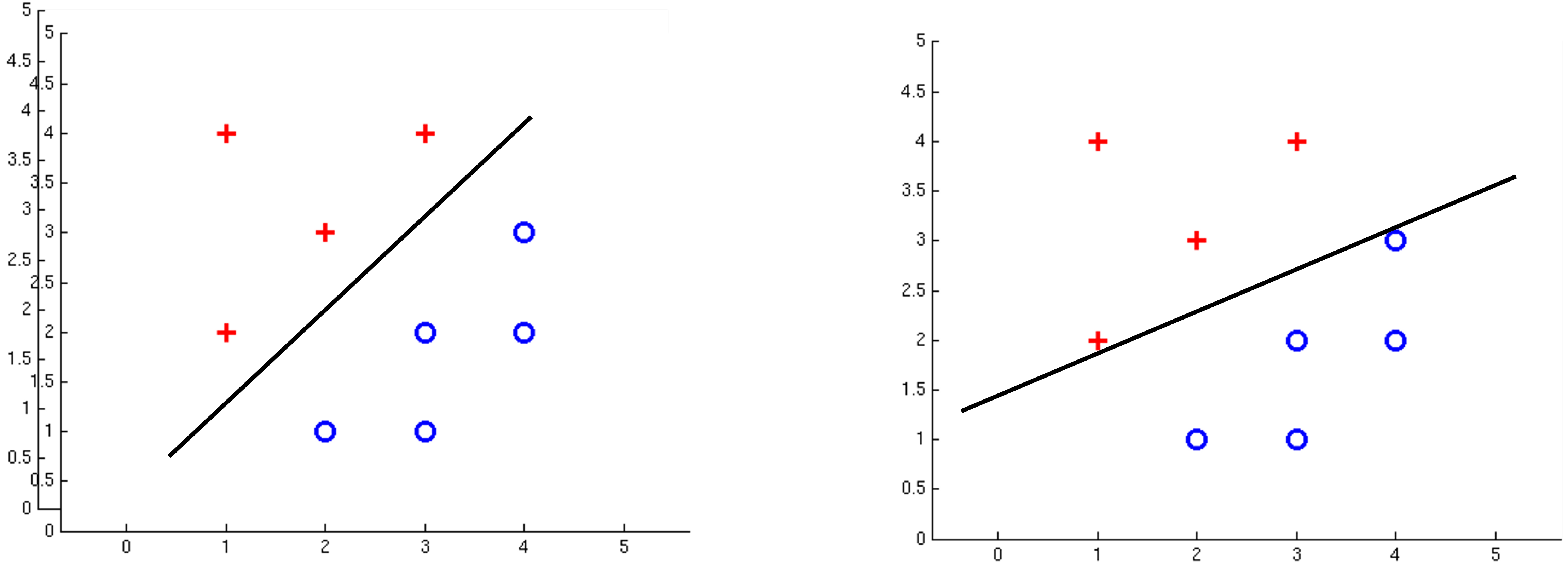
with:

$$P(y^{(i)} = +1|x^{(i)}; w) = \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

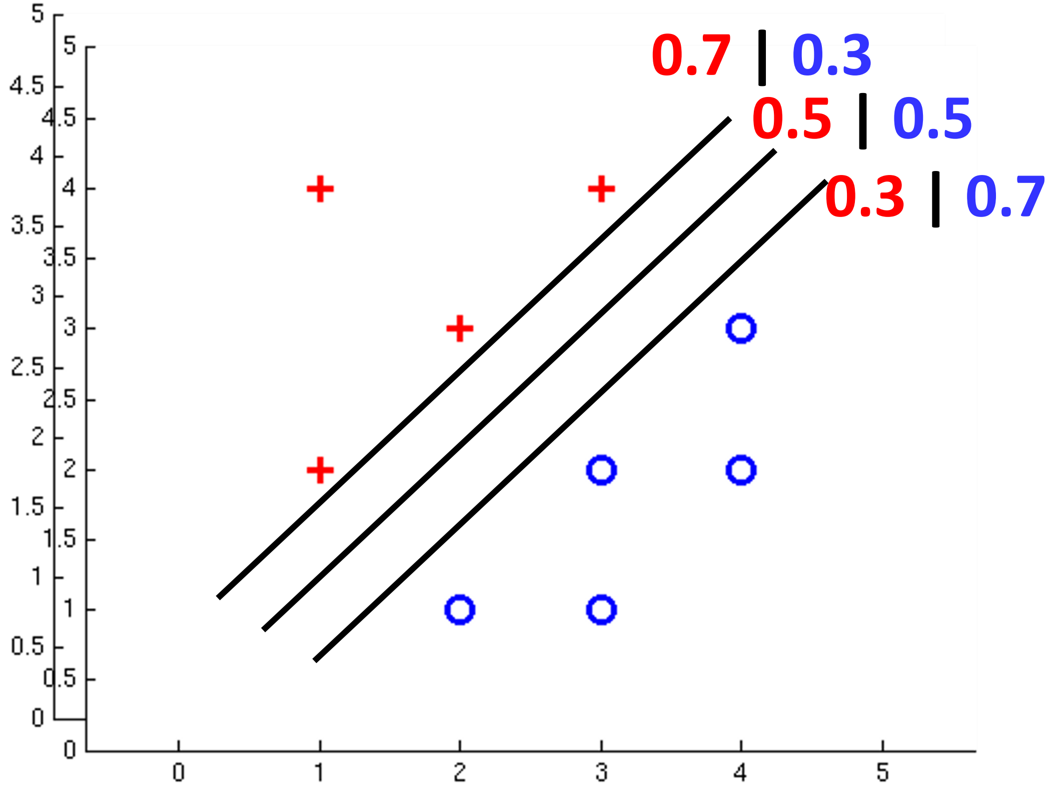
$$P(y^{(i)} = -1|x^{(i)}; w) = 1 - \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

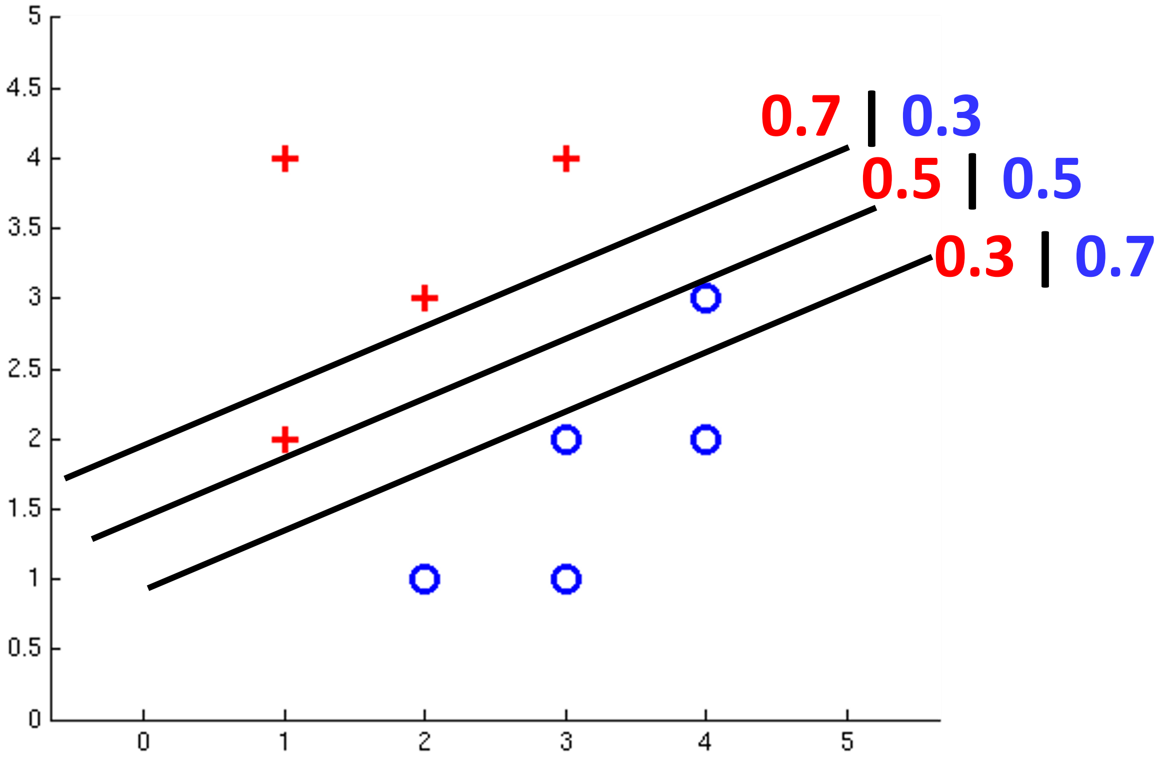
= Logistic Regression

Separable Case: Deterministic Decision – Many Options



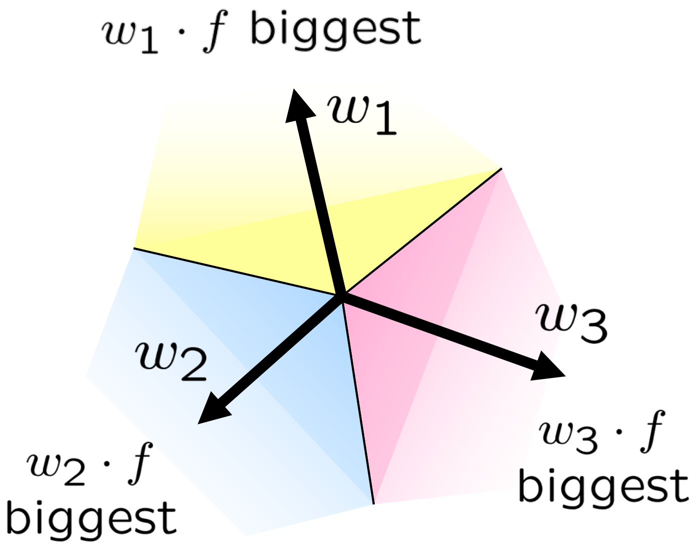
Separable Case: Probabilistic Decision – Clear Preference





Multiclass Logistic Regression

- Recall Perceptron:
  - A weight vector for each class: w_y
  - Score (activation) of a class y: $w_y \cdot f(x)$
  - Prediction highest score wins $y = \arg\max_y \; w_y \cdot f(x)$



- How to make the scores into probabilities? Softmax function

$$\underbrace{z_1, z_2, z_3}_{\text{original activations}} \rightarrow \underbrace{\frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_2}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_3}}{e^{z_1}+e^{z_2}+e^{z_3}}}_{\text{softmax activations}}$$

Best w?

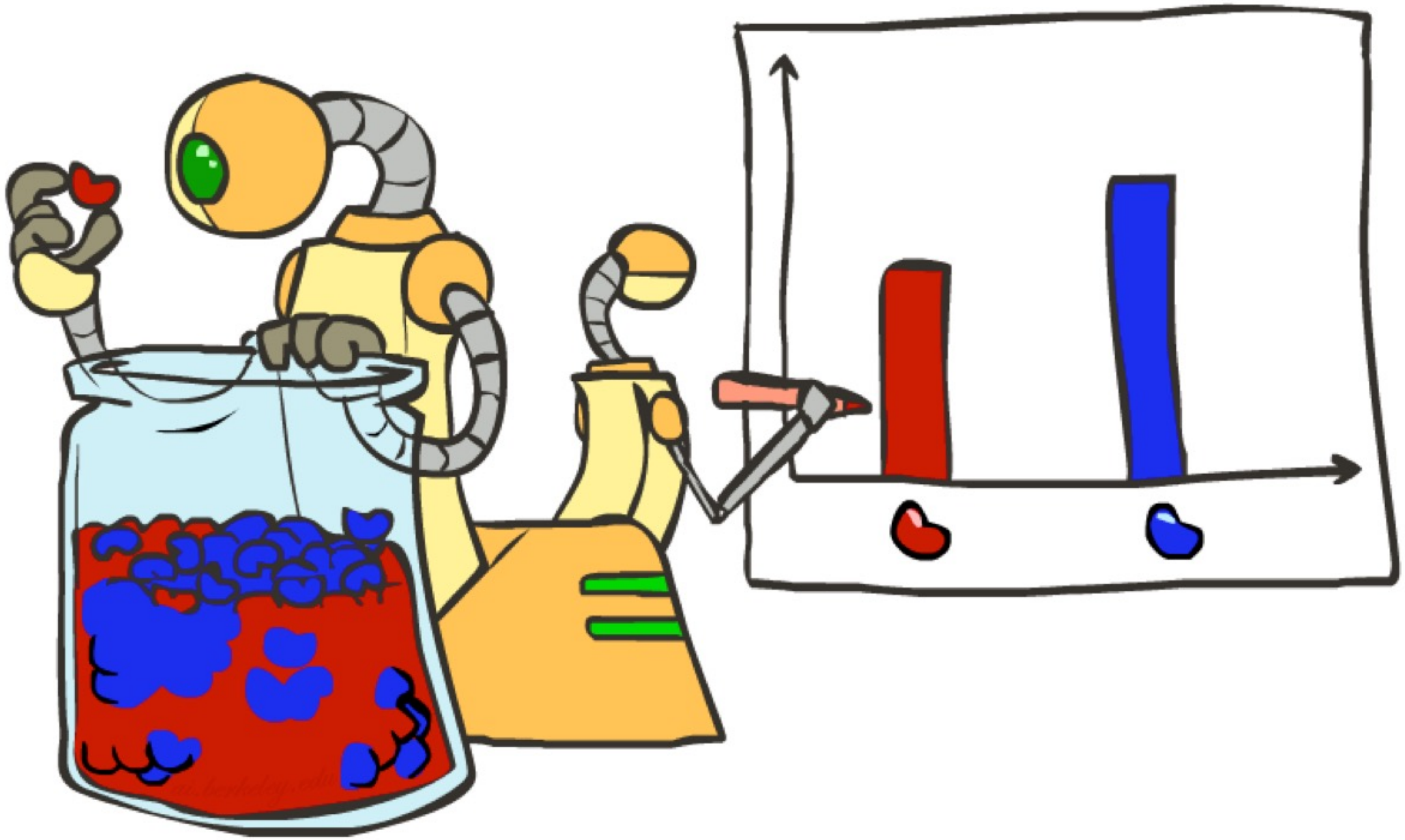
- Maximum likelihood estimation:

$$\max_{w} \quad ll(w) = \max_{w} \quad \sum_{i} \log P(y^{(i)}|x^{(i)}; w)$$

with: $$P(y^{(i)}|x^{(i)}; w) = \frac{e^{w_{y^{(i)}} \cdot f(x^{(i)})}}{\sum_y e^{w_y \cdot f(x^{(i)})}}$$

= Multi-Class Logistic Regression

Maximum Likelihood Estimation

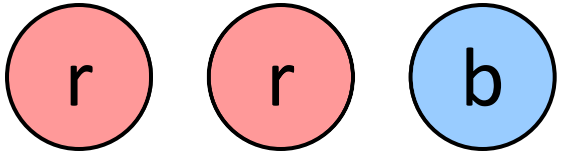


Parameter Estimation with Maximum Likelihood

X	red	blue
$P_\theta(x)$	θ	$1-\theta$

- Estimating the distribution of a random variable
- Use training data (learning!)
  - For each outcome x, look at the **empirical rate** of that value:

$$\mathrm{P}_{ML} = \frac{\mathrm{count}(x)}{\text{total samples}}$$



- Example: probability of x=red given the training data:

$$\mathrm{P}_{ML}(r) = \frac{2}{3}$$

- This estimate maximizes the **likelihood of the data** for the parametric model:

$$L(\theta) = \mathrm{P}(\,\mathrm{r}, \mathrm{r}, \mathrm{b} \mid \theta\,) = \mathrm{P}_\theta(r) \cdot \mathrm{P}_\theta(r) \cdot \mathrm{P}_\theta\,(b)$$
$$= \theta^2 \cdot (1-\theta)$$

Parameter Estimation with Maximum Likelihood

- Likelihood function:

$$L(\theta) = \mathrm{P}(\, r, r, b \mid \theta\,) = \mathrm{P}_\theta(r)\cdot \mathrm{P}_\theta(r)\cdot \mathrm{P}_\theta\,(b)$$
$$= \theta^2 \cdot (1-\theta)$$
$$= \theta^2 - \theta^3$$

- MLE: find the θ that maximizes data likelihood

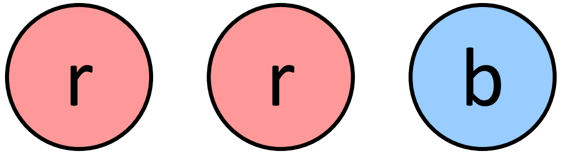
$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, L(\theta)$$

- Approach: take derivatives and set to 0

$$\frac{\partial L(\theta)}{\partial \theta} = 2\theta - 3\theta^2$$
$$= \theta(2-3\theta)$$

- Find the maximum at $\theta = \frac{2}{3}$

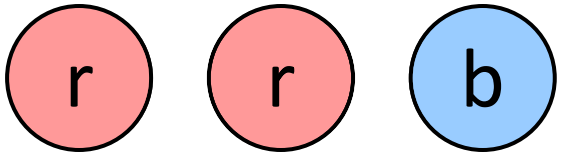
X	red	blue
$P_\theta(x)$	θ	$1-\theta$



Parameter Estimation (General Case)

- **Model**:

X	red	blue
$P_\theta(x)$	θ	$1-\theta$



- **Data**: draw N balls. N_r come up red, N_b come up blue
  - Dataset: $D = \{x_1, \dots, x_n\}$
  - Ball draws are independent and identically distributed (i.i.d.):

$$P(\, D \mid \theta \,) = \prod_i P(\, x_i \mid \theta \,) = \prod_i P_\theta(x_i) = \theta^{N_r} \cdot (1-\theta)^{N_b}$$

- **Maximum likelihood estimation**: find θ that maximizes $P(\, D \mid \theta \,)$

$$\theta = \operatorname*{argmax}_\theta P(\, D \mid \theta \,) = \operatorname*{argmax}_\theta \log P(\, D \mid \theta \,)$$

  - Approach: take derivative and set to 0

Parameter Estimation (General Case)

- **Maximum likelihood estimation**: find θ that maximizes $P(\,D \mid \theta\,)$

$$\theta = \underset{\theta}{\operatorname{argmax}}\, P(\,D \mid \theta\,) = \underset{\theta}{\operatorname{argmax}} \log P(\,D \mid \theta\,)$$

$$\boxed{P(\,D \mid \theta\,) = \theta^{N_r} \cdot (1-\theta)^{N_b}}$$

$$\begin{aligned}
\frac{\partial}{\partial\theta} \log P(\,D \mid \theta\,) &= \frac{\partial}{\partial\theta}[N_r \log(\theta) + N_b \log(1-\theta)] \\
&= N_r \frac{\partial}{\partial\theta} \log(\theta) + N_b \frac{\partial}{\partial\theta} \log(1-\theta) \\
&= N_r \frac{1}{\theta} - N_b \frac{1}{1-\theta} \\
&= 0
\end{aligned}$$

Multiply by $\theta(1-\theta)$:

$$N_r(1-\theta) - N_b\theta = 0$$

$$N_r - \theta(N_r + N_b) = 0$$

$$\boxed{\hat{\theta} = \frac{N_r}{N_r + N_b}}$$

Example

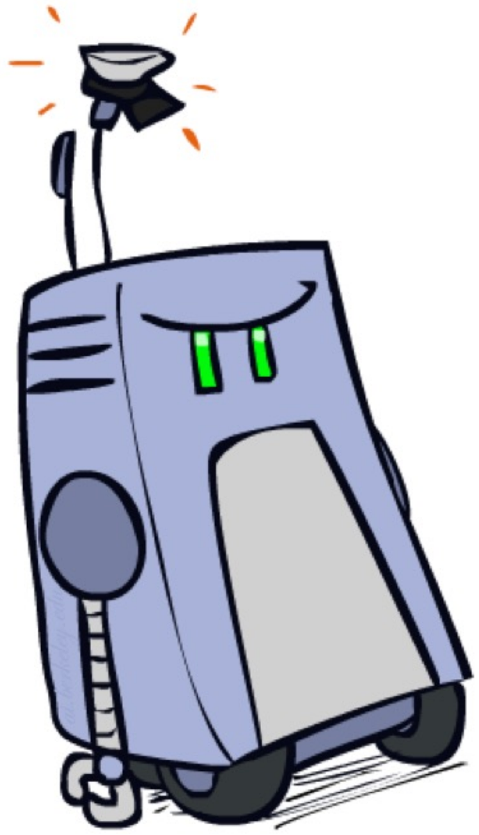
Recall that a Geometric distribution is a defined as the number of Bernoulli trials needed to get one success. $P(X=k)=p(1-p)^{k-1}$.
We observe the following samples from a Geometric distribution:
$x_1=5,\ x_2=8,\ x_3=3,\ x_4=5,\ x_5=7$
What is the maximum likelihood estimate for p?

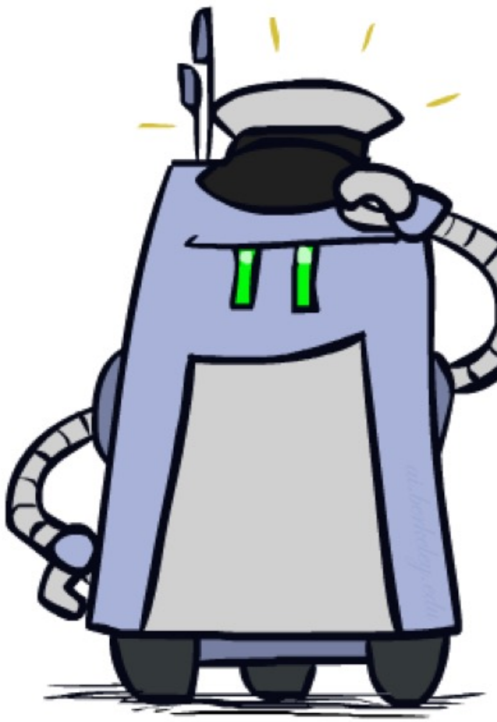
$$\begin{aligned}
L(p) &= P(X=x_1)P(X=x_2)P(X=x_3)P(X=x_4)P(X=x_5) && (1)\\
&= P(X=5)P(X=8)P(X=3)P(X=5)P(X=7) && (2)\\
&= p^5(1-p)^{23} && (3)\\
\log(L(p)) &= 5\log(p) + 23\log(1-p) && (4)\\
& && (5)
\end{aligned}$$

We must maximize the log-likelihood of p, so we will take the derivative, and set it to 0.

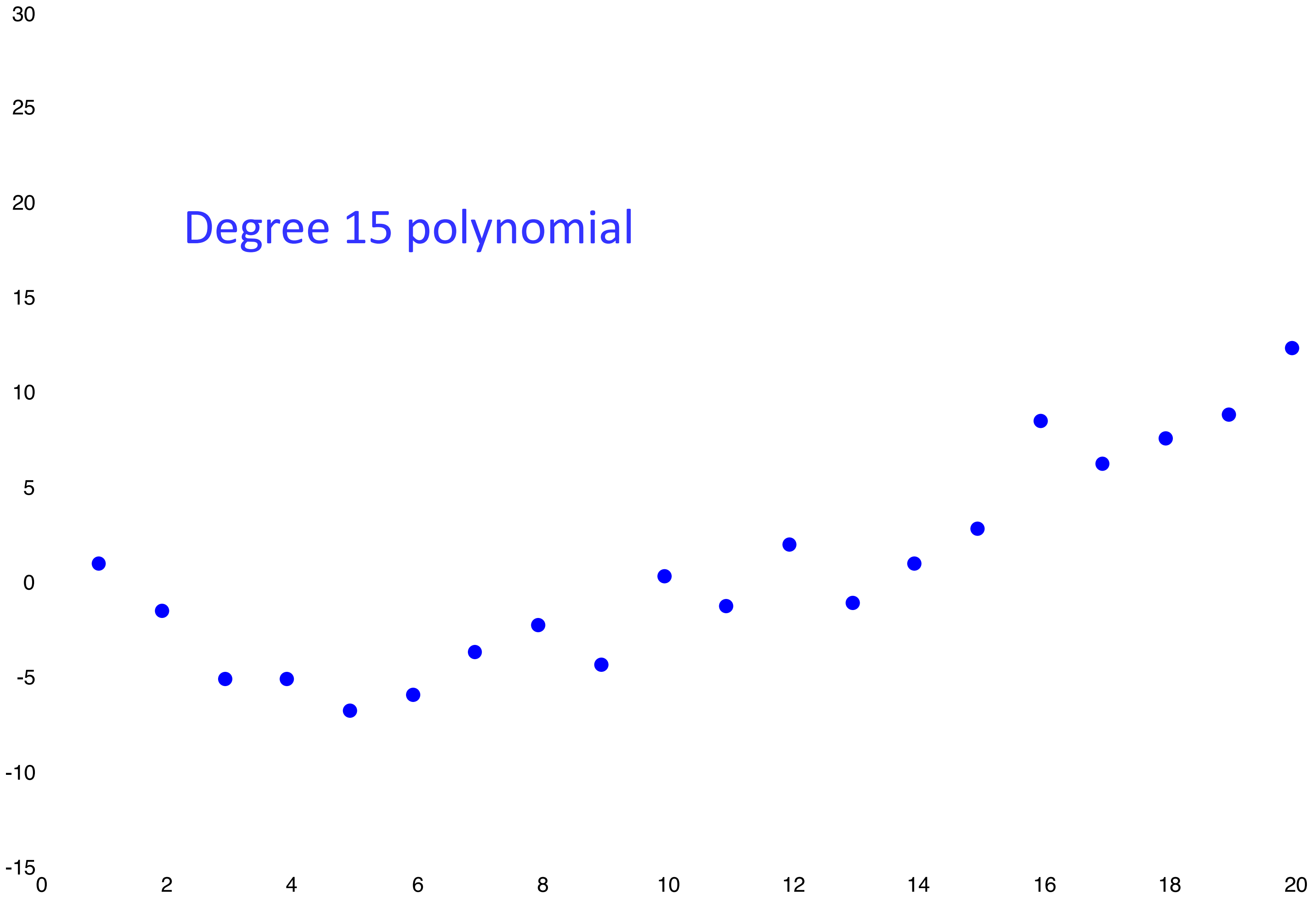
$$\begin{aligned}
0 &= \frac{5}{p} - \frac{23}{1-p} && (6)\\
p &= 5/28 && (7)
\end{aligned}$$

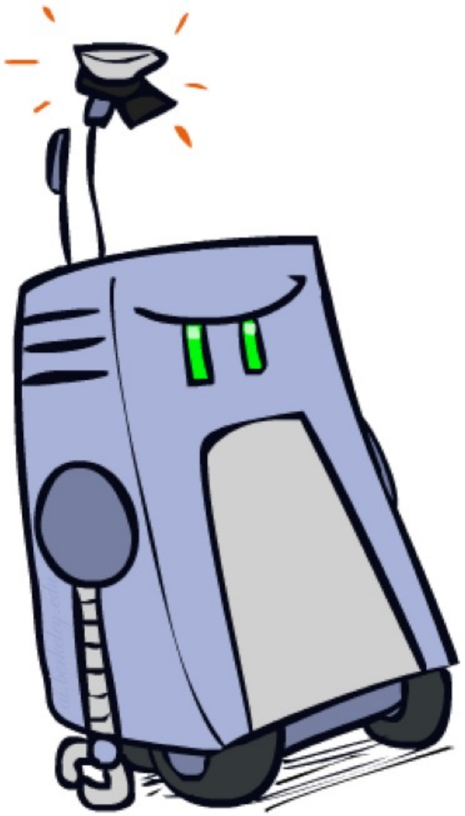
Regularization





Recall: Overfitting





Example: Overfitting

$P(\text{features}, C = 2)$

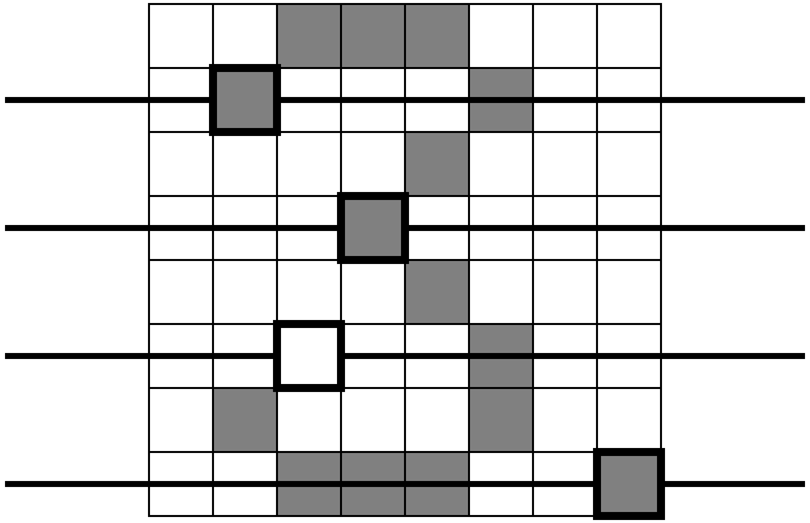
$P(C = 2) = 0.1$

$P(\text{on}|C = 2) = 0.8$

$P(\text{on}|C = 2) = 0.1$

$P(\text{off}|C = 2) = 0.1$

$P(\text{on}|C = 2) = 0.01$



$P(\text{features}, C = 3)$

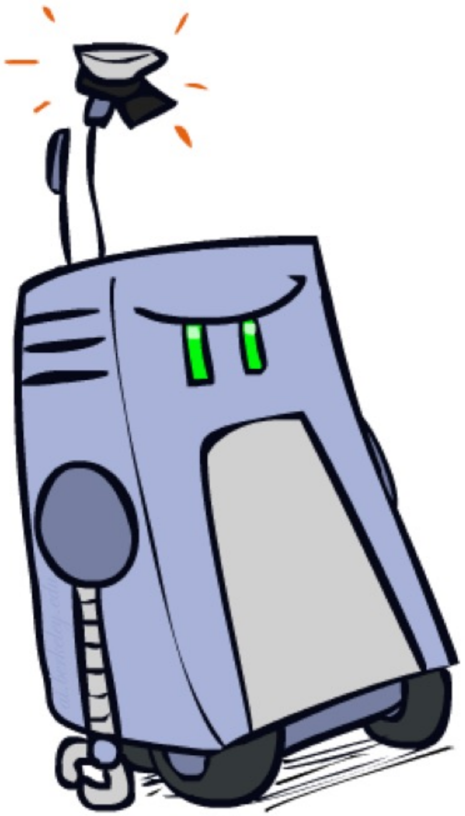
$P(C = 3) = 0.1$

$P(\text{on}|C = 3) = 0.8$

$P(\text{on}|C = 3) = 0.9$

$P(\text{off}|C = 3) = 0.7$

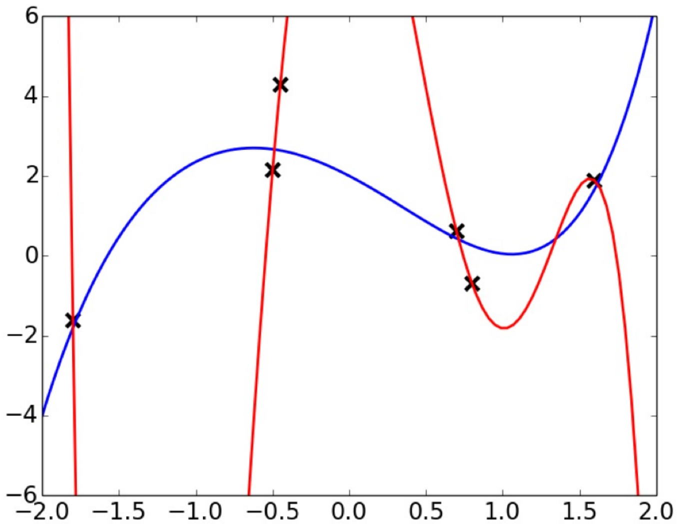
$P(\text{on}|C = 3) = 0.0$

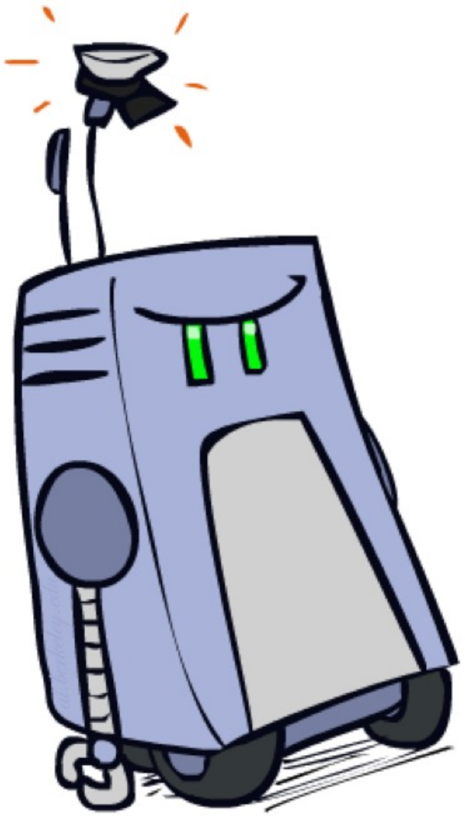


2 wins!!

Recall: Overfitting

- Observation: polynomials that overfit tend to have large coefficients





$$y = 0.1x^5 + 0.2x^4 + 0.75x^3 - x^2 - 2x + 2$$
$$y = -7.2x^5 + 10.4x^4 + 24.5x^3 - 37.9x^2 - 3.6x + 12$$

- Let's try to keep coefficients small!

L1 and L2 Regularization

- Previously:

$$\hat{w} = \arg\max_{w} \sum_{i=1}^{n} \log P\left(y^{(i)} \mid x^{(i)}; w\right)$$

- Now: add a penalty term to keep the weight vector small

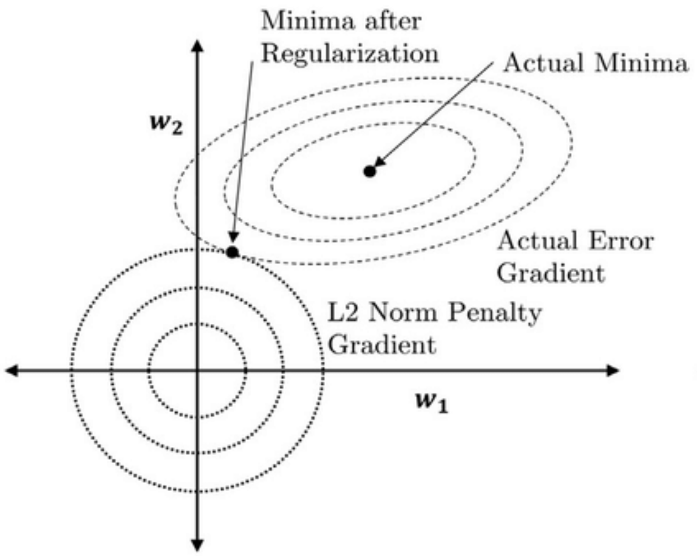
L1
(aka lasso regression)

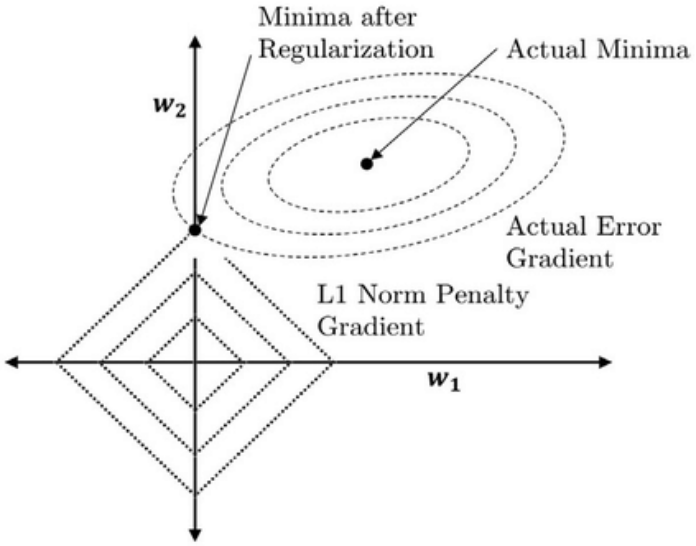
$$\hat{w} = \arg\max_{w} \sum_{i=1}^{n} \log P\left(y^{(i)} \mid x^{(i)}; w\right) - \alpha \sum_{i=1}^{n} |w_i|$$

L2
(aka ridge regression)

$$\hat{w} = \arg\max_{w} \sum_{i=1}^{n} \log P\left(y^{(i)} \mid x^{(i)}; w\right) - \alpha \sum_{i=1}^{n} w_i^2$$

L1 and L2 Regularization





L1
(aka lasso regression)

$$\hat{w} = \arg\max_{w} \sum_{i=1}^{n} \log P\left(y^{(i)} \mid x^{(i)}; w\right) - \alpha \sum_{i=1}^{n} |w_i|$$

L2
(aka ridge regression)

$$\hat{w} = \arg\max_{w} \sum_{i=1}^{n} \log P\left(y^{(i)} \mid x^{(i)}; w\right) - \alpha \sum_{i=1}^{n} w_i^2$$