

CS 687

Jana Kosecka

- Perceptron, Logistic Regression

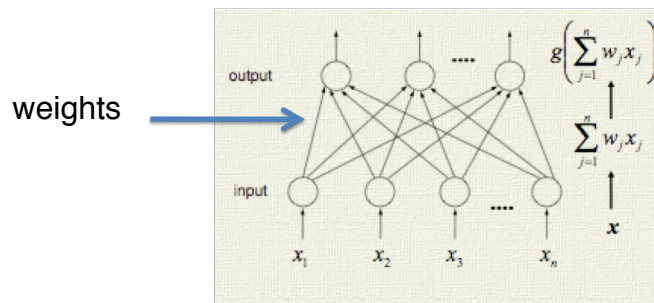
Some slides adopted from Peter Abbeel, Dan Klein AI course

Sample Applications

- Any place where we have features and categorical variables
- Medical diagnostic tasks (prediction of cancer, heart attack)
- Portfolio management (sell/no sell)
- Musical hit detection
- Using Logistic Regression to Predict the Probability of Debris Flows in Areas Burned by Wildfires
- Computer Vision – classification problems (face/no face)
- Art is often in selecting right features
- Can be deployed with alternative techniques for feature selection (by enforcing sparseness or regularization)

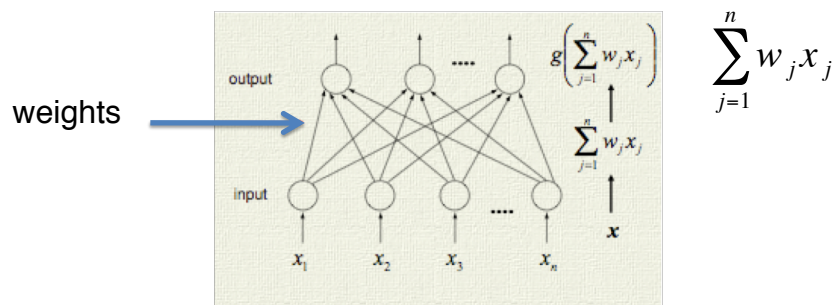
Perceptron (Frank Rosenblatt, 1957)

- First learning algorithm for neural networks
- Single layer feed-forward neural network
- Originally introduced for character classification, where each character is represented as an image;



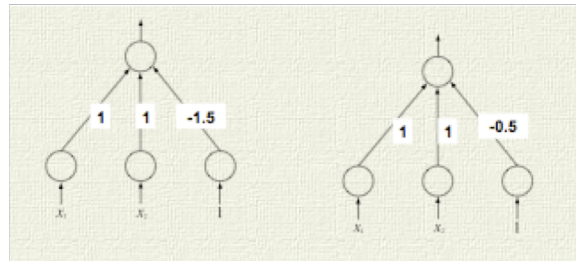
Perceptron (Frank Rosenblatt, 1957)

- Consider one output unit at the time – the others are independent
- As g use a simple threshold function if $w^T x > 0$



Perceptron (Frank Rosenblatt, 1957)

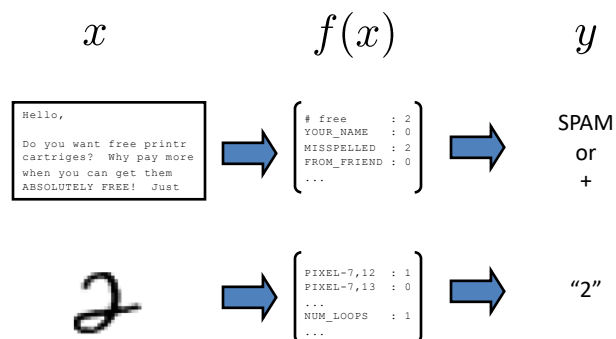
- Compare individual units with logic gates



AND

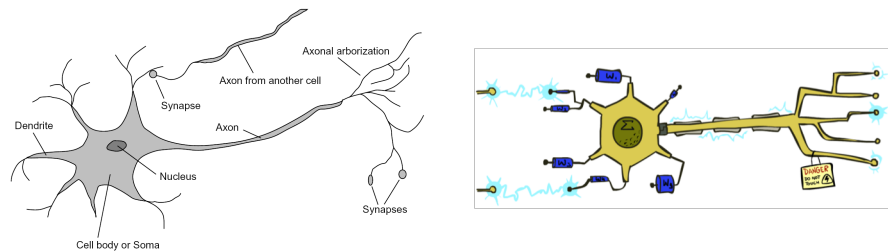
OR

Feature Vectors



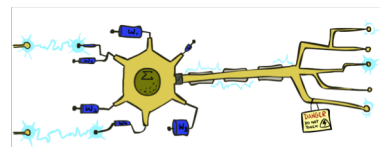
Some (Simplified) Biology

- Very loose inspiration: human neurons



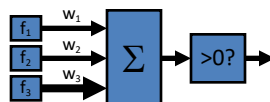
Linear Classifiers

- Inputs are **feature values**
- Each feature has a **weight**
- Sum is the **activation**



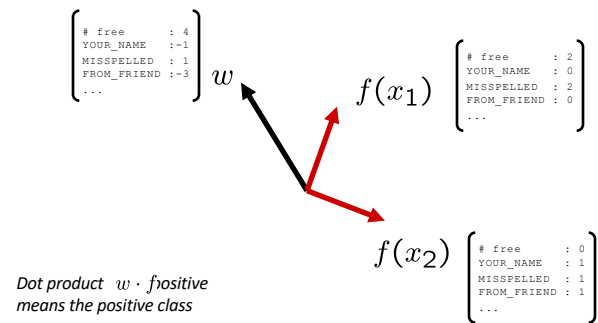
$$\text{activation}_w(x) = \sum_i w_i \cdot f_i(x) = w \cdot f(x)$$

- If the activation is:
 - Positive, output +1
 - Negative, output -1

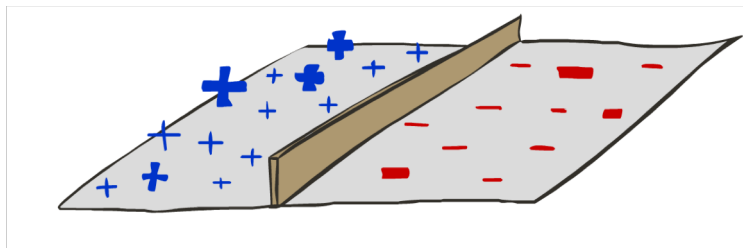


Weights

- Binary case: compare features to a weight vector
- Learning: figure out the weight vector from examples

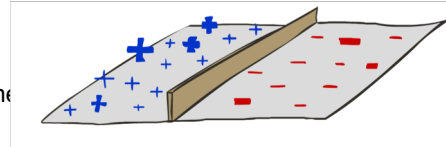


Decision Rules



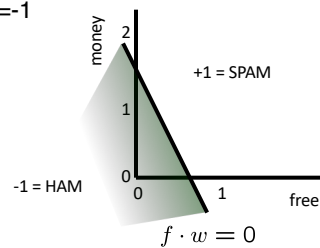
Binary Decision Rule

- In the space of feature vectors
 - Examples are points
 - Any weight vector is a hyperplane
 - One side corresponds to $Y=+1$
 - Other corresponds to $Y=-1$



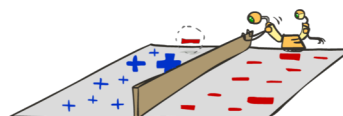
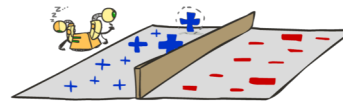
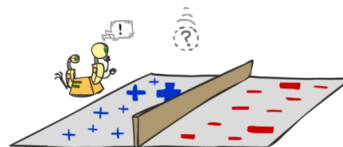
$$w$$

BIAS	: -3
free	: 4
money	: 2
...	



Learning: Binary Perceptron

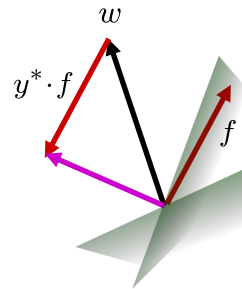
- Start with weights = 0
- For each training instance:
 - Classify with current weights
 - If correct (i.e., $y=y^*$), no change!
 - If wrong: adjust the weight vector



Learning: Binary Perceptron

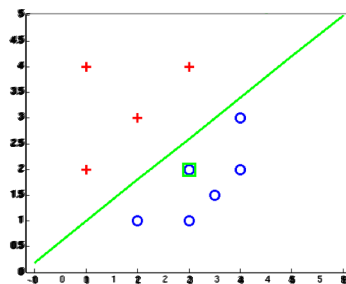
- Start with weights = 0
 - For each training instance:
 - Classify with current weights
- $$y = \begin{cases} +1 & \text{if } w \cdot f(x) \geq 0 \\ -1 & \text{if } w \cdot f(x) < 0 \end{cases}$$
- If correct (i.e., $y=y^*$), no change!
 - If wrong: adjust the weight vector by adding or subtracting the feature vector. Subtract if y^* is -1.

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



Examples: Perceptron

- Separable Case



Multiclass Decision Rule

- If we have multiple classes:

- A weight vector for each class:

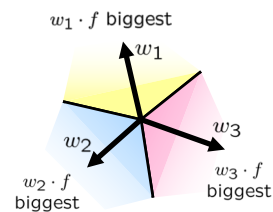
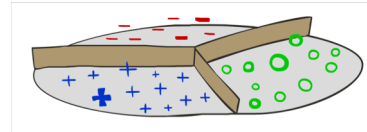
$$w_y$$

- Score (activation) of a class y :

$$w_y \cdot f(x)$$

- Prediction highest score wins

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



Binary = multiclass where the negative class has weight zero

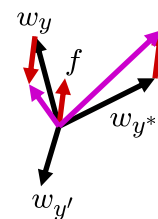
Learning: Multiclass Perceptron

- 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

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

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



Example: Multiclass Perceptron

“win the vote”

“win the election”

“win the game”

w_{SPORTS}

BIAS	: 1
win	: 0
game	: 0
vote	: 0
the	: 0
...	

$w_{POLITICS}$

BIAS	: 0
win	: 0
game	: 0
vote	: 0
the	: 0
...	

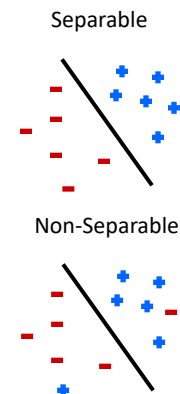
w_{TECH}

BIAS	: 0
win	: 0
game	: 0
vote	: 0
the	: 0
...	

Properties of Perceptrons

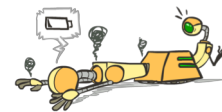
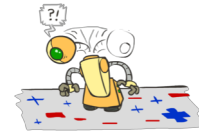
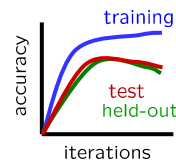
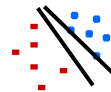
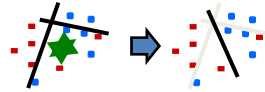
- Separability: true if some parameters get the training set perfectly correct
- Convergence: if the training is separable, perceptron will eventually converge (binary case)
- Mistake Bound: the maximum number of mistakes (binary case) related to the *margin* or degree of separability

$$\text{mistakes} < \frac{k}{\delta^2}$$

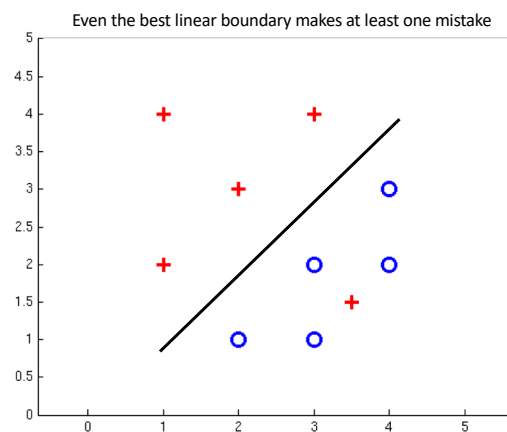


Problems with the Perceptron

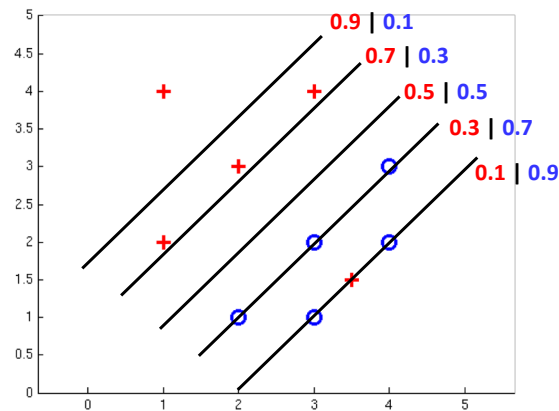
- Noise: if the data isn't separable, weights might thrash
 - Averaging weight vectors over time can help (averaged perceptron)
- Mediocre generalization: finds a “barely” separating solution
- Overtraining: test / held-out accuracy usually rises, then falls
 - Overtraining is a kind of overfitting



Non-Separable Case: Deterministic Decision



Non-Separable Case: Probabilistic Decision

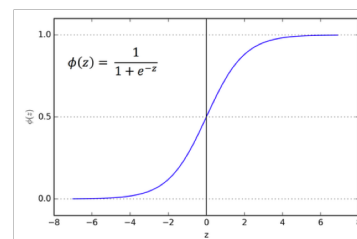


How to get probabilistic decisions?

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

- Sigmoid function

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



Best w ?

- Maximum likelihood estimation:

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

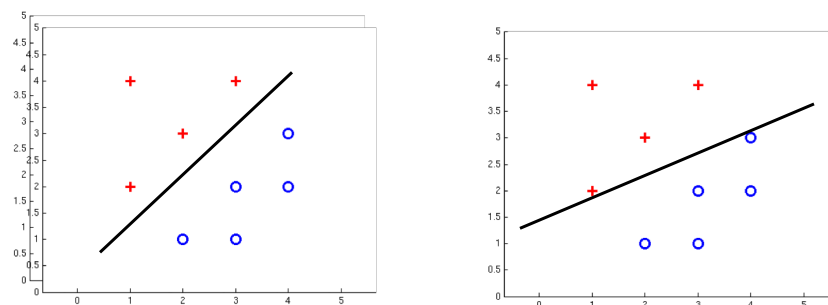
with:

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

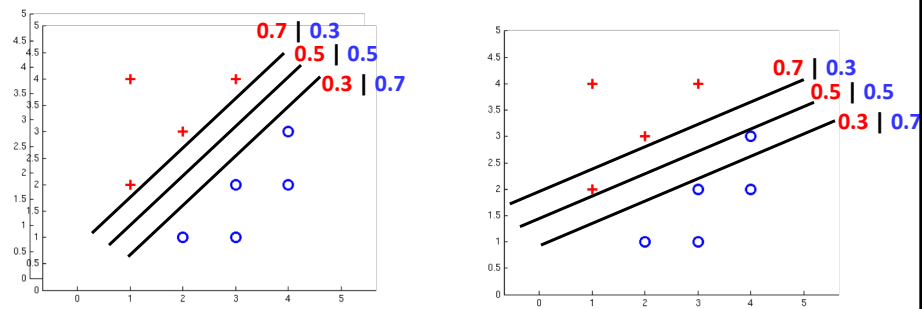
$$P(y^{(i)} = -1|x^{(i)}; w) = 1 - \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

= **Logistic Regression**

Separable Case: Deterministic Decision – Many Options

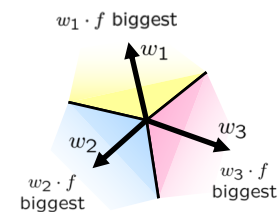


Separable Case: Probabilistic Decision – Clear Preference



Multiclass Logistic Regression

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



- How to make the scores into probabilities?

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

Best w ?

- Maximum likelihood estimation:

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

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

= Multi-Class Logistic Regression

Next Lecture

- Optimization

– i.e., how do we solve:

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