# Model Selection, Perceptron

## CSC 461: Machine Learning

Fall 2022

Prof. Marco Alvarez
University of Rhode Island

---

# Evaluation (model selection)

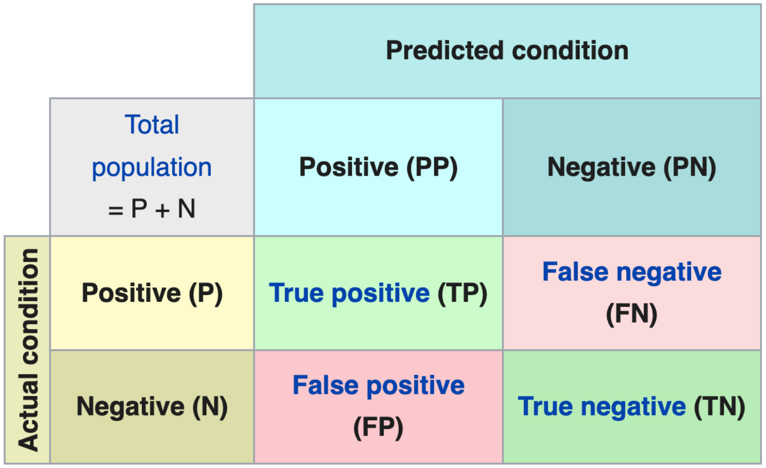
---

# Actuals/Predictions (example)

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| Actual | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Predicted | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| | FN | FN | TP | TP | TP | TP | TP | TP | FP | TN | TN | TN |

---

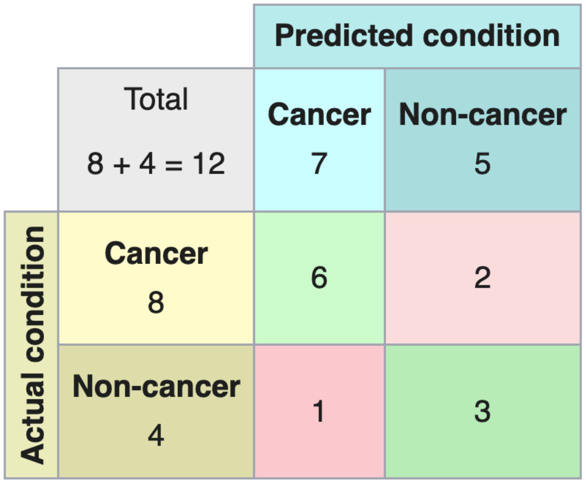
# Confusion matrix (2 classes)



| | | | |
|---|---|---|---|
| | | **Predicted condition** | |
| | Total population = P + N | **Positive (PP)** | **Negative (PN)** |
| **Actual condition** | **Positive (P)** | **True positive (TP)** | **False negative (FN)** |
| | **Negative (N)** | **False positive (FP)** | **True negative (TN)** |

https://en.wikipedia.org/wiki/Confusion_matrix

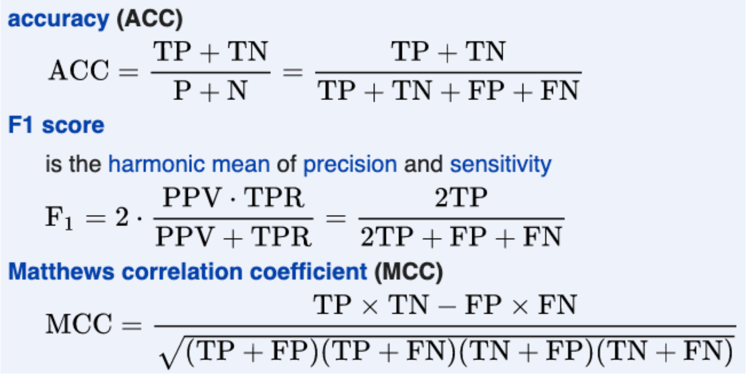
# Confusion matrix (example)



| | | | |
|---|---|---|---|
| | | **Predicted condition** | |
| | Total<br>8 + 4 = 12 | **Cancer**<br>7 | **Non-cancer**<br>5 |
| **Actual condition** | **Cancer**<br>8 | 6 | 2 |
| | **Non-cancer**<br>4 | 1 | 3 |

https://en.wikipedia.org/wiki/Confusion_matrix

# Evaluation metrics (2 classes)



**accuracy (ACC)**

$$\mathrm{ACC} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{P}+\mathrm{N}} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$$

**F1 score**

is the harmonic mean of precision and sensitivity

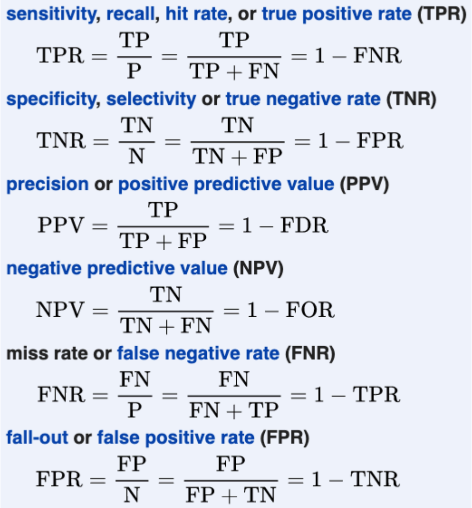
$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV}+\mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}$$

**Matthews correlation coefficient (MCC)**

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}$$

https://en.wikipedia.org/wiki/Confusion_matrix

# Evaluation metrics (2 classes)



**sensitivity, recall, hit rate, or true positive rate (TPR)**

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} = 1 - \mathrm{FNR}$$

**specificity, selectivity or true negative rate (TNR)**

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}} = 1 - \mathrm{FPR}$$

**precision or positive predictive value (PPV)**

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} = 1 - \mathrm{FDR}$$

**negative predictive value (NPV)**

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FN}} = 1 - \mathrm{FOR}$$

**miss rate or false negative rate (FNR)**

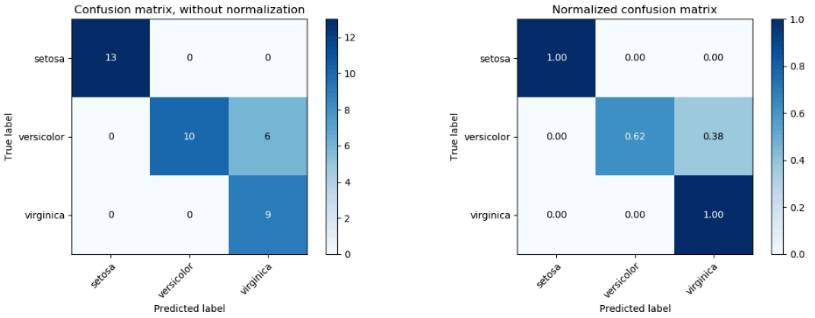
$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN}+\mathrm{TP}} = 1 - \mathrm{TPR}$$

**fall-out or false positive rate (FPR)**

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} = 1 - \mathrm{TNR}$$

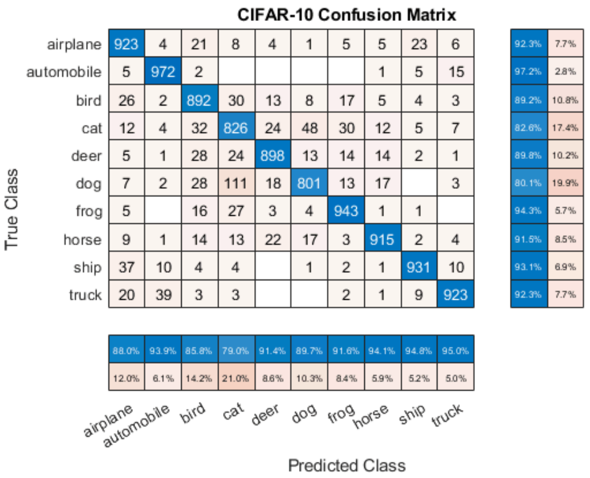
https://en.wikipedia.org/wiki/Confusion_matrix

# Confusion matrix (>2 classes)



https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

# Confusion matrix (>2 classes)



CIFAR-10 Confusion Matrix

| True Class | airplane | automobile | bird | cat | deer | dog | frog | horse | ship | truck | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| airplane | 923 | 4 | 21 | 8 | 4 | 1 | 5 | 5 | 23 | 6 | 92.3% | 7.7% |
| automobile | 5 | 972 | 2 | | | | | 1 | 5 | 15 | 97.2% | 2.8% |
| bird | 26 | 2 | 892 | 30 | 13 | 8 | 17 | 5 | 4 | 3 | 89.2% | 10.8% |
| cat | 12 | 4 | 32 | 826 | 24 | 48 | 30 | 12 | 5 | 7 | 82.6% | 17.4% |
| deer | 5 | 1 | 28 | 24 | 898 | 13 | 14 | 14 | 2 | 1 | 89.8% | 10.2% |
| dog | 7 | 2 | 28 | 111 | 18 | 801 | 13 | 17 | | 3 | 80.1% | 19.9% |
| frog | 5 | | 16 | 27 | 3 | 4 | 943 | 1 | 1 | | 94.3% | 5.7% |
| horse | 9 | 1 | 14 | 13 | 22 | 17 | 3 | 915 | 2 | 4 | 91.5% | 8.5% |
| ship | 37 | 10 | 4 | 4 | | 1 | 2 | 1 | 931 | 10 | 93.1% | 6.9% |
| truck | 20 | 39 | 3 | 3 | | | 2 | 1 | 9 | 923 | 92.3% | 7.7% |
| | 88.0% | 93.9% | 85.8% | 79.0% | 91.4% | 89.7% | 91.6% | 94.1% | 94.8% | 95.0% | | |
| | 12.0% | 6.1% | 14.2% | 21.0% | 8.6% | 10.3% | 8.4% | 5.9% | 5.2% | 5.0% | | |

Predicted Class

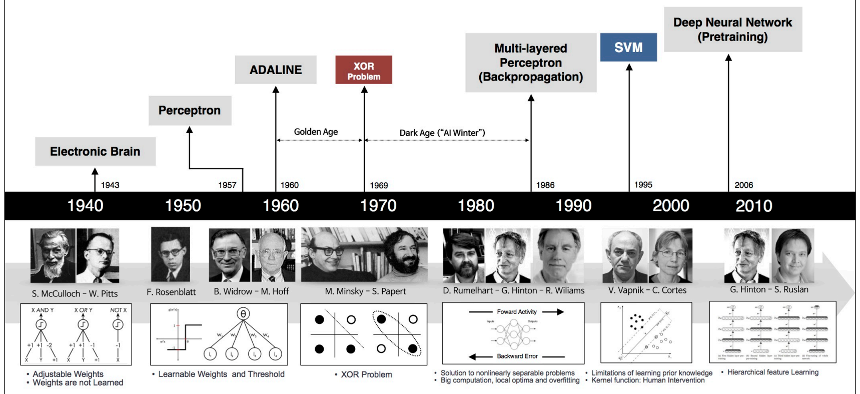
https://www.mathworks.com/help/deeplearning/ref/confusionchart.html

---

# The Perceptron

---

# Neural Networks



http://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

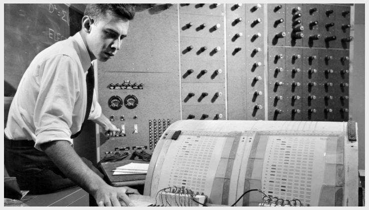
---

# Rosenblatt (1958)

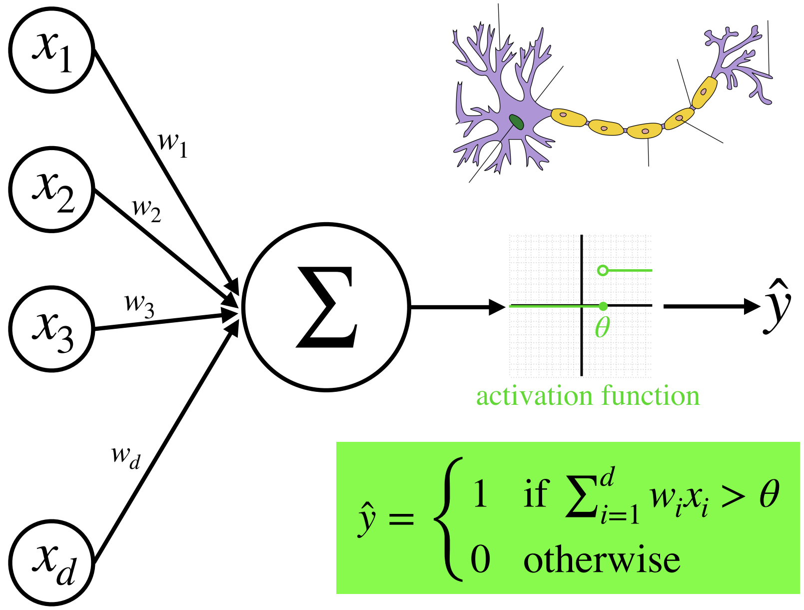
- Perceptron introduced by Frank Rosenblatt (psychologist, logician)
  - ✓ based on work from McCulloch-Pitts and Hebb
  - ✓ very powerful **learning** algorithm with high expectations

***NEW NAVY DEVICE LEARNS BY DOING; Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser***

WASHINGTON, July 7, 1958 (UPI) -- The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.



https://news.cornell.edu/stories/2019/09/professors-perceptron-paved-way-ai-60-years-too-soon



$$\hat{y} = \begin{cases} 1 & \text{if } \sum_{i=1}^{d} w_i x_i > \theta \\ 0 & \text{otherwise} \end{cases}$$

# Absorbing the threshold/bias

$$\hat{y} = \begin{cases} 1 & \text{if } \sum_{i=1}^{d} w_i x_i > \theta \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{y} = \begin{cases} 1 & \text{if } \sum_{i=1}^{d} w_i x_i - \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{y} = \begin{cases} 1 & \text{if } \sum_{i=0}^{d} w_i x_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$x_0 = +1, \quad w_0 = -\theta$$

# Another look

For convenience we will use +1 and -1 instead of 1 and 0

$$h_{\mathbf{w}}(\mathbf{x}) = \sigma\left(\sum_{i=0}^{d} w_i x_i\right) = \sigma(\mathbf{w}^T\mathbf{x})$$

$$\sigma(z) = \begin{cases} 1 & \text{if } z > 0 \\ -1 & \text{if } z \leq 0 \end{cases}$$

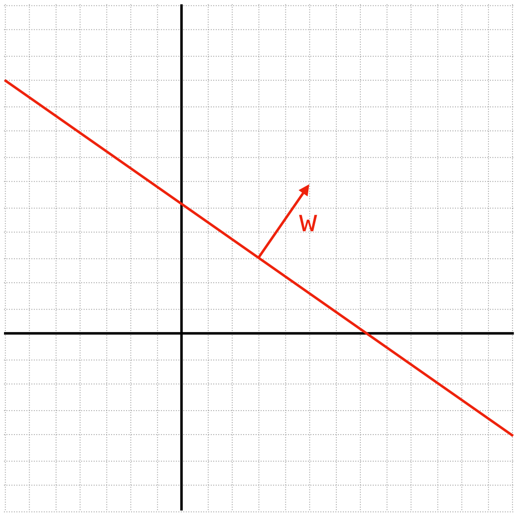
$$\mathscr{H} = \{h_w : \mathbf{w} \in \mathbb{R}^{d+1}\}$$

# Perceptron Algorithm

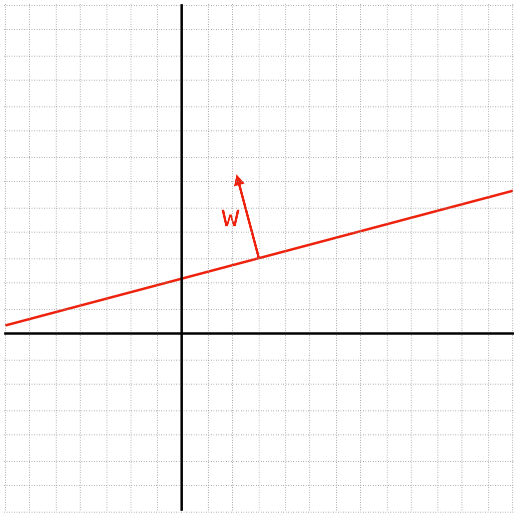
- Start with a null vector **w**
- Repeat for T epochs
  - ✓ shuffle the data instances
  - ✓ for all examples in training data
    - ✓ if misclassified
      - update the weight vector by adding **x** to **w** if the actual label is positive and subtracting **x** from **w** otherwise
- Return **w**

## Write the pseudocode

## Mistake on a positive (update)



## Mistake on a negative (update)



## Intuition

- Suppose a mistake on the positive side:

$$y = +1 \qquad \mathbf{w}^T x \leq 0$$

- After 1 update the new weight vector will be:

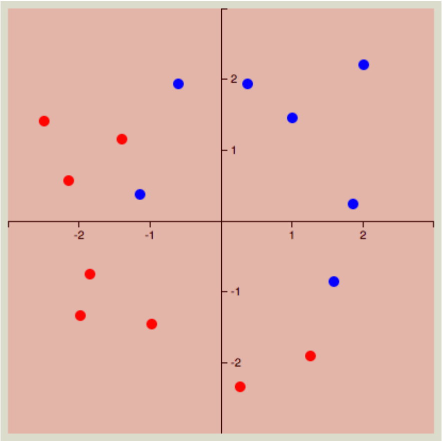
$$\mathbf{w}_{t+1} = \mathbf{w}_t + \mathbf{x}$$

- Classifying the datapoint with the new weight vector:

$$\mathbf{w}_{t+1}{}^T\mathbf{x} = (\mathbf{w}_t + \mathbf{x})^T\mathbf{x} = \mathbf{w}_t^T\mathbf{x} + \mathbf{x}^T\mathbf{x} \geq \mathbf{w}_t^T\mathbf{x}$$

use same idea for mistakes on the negative side

# Demo



https://planspace.org/20150907-interactive_perceptron_training_toy/

---

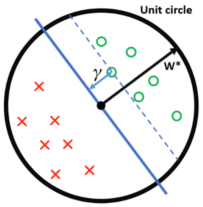
# Perceptron convergence theorem

The argument goes as follows: Suppose $\exists \mathbf{w}^*$ such that $y_i(\mathbf{x}^\top \mathbf{w}^*) > 0 \; \forall (\mathbf{x}_i, y_i) \in D$.

Now, suppose that we rescale each data point and the $\mathbf{w}^*$ such that

$$||\mathbf{w}^*|| = 1 \qquad \text{and} \qquad ||\mathbf{x}_i|| \leq 1 \;\; \forall \mathbf{x}_i \in D$$

Let us define the Margin $\gamma$ of the hyperplane $\mathbf{w}^*$ as $\gamma = \min_{(\mathbf{x}_i, y_i) \in D} |\mathbf{x}_i^\top \mathbf{w}^*|$.



To summarize our setup:

- All inputs $\mathbf{x}_i$ live within the unit sphere
- There exists a separating hyperplane defined by $\mathbf{w}^*$, with $\|\mathbf{w}\|^* = 1$ (i.e. $\mathbf{w}^*$ lies exactly on the unit sphere).
- $\gamma$ is the distance from this hyperplane (blue) to the closest data point.

**Theorem:** If all of the above holds, then the Perceptron algorithm makes at most $1/\gamma^2$ mistakes.

http://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html

---

# Perceptron (remarks)

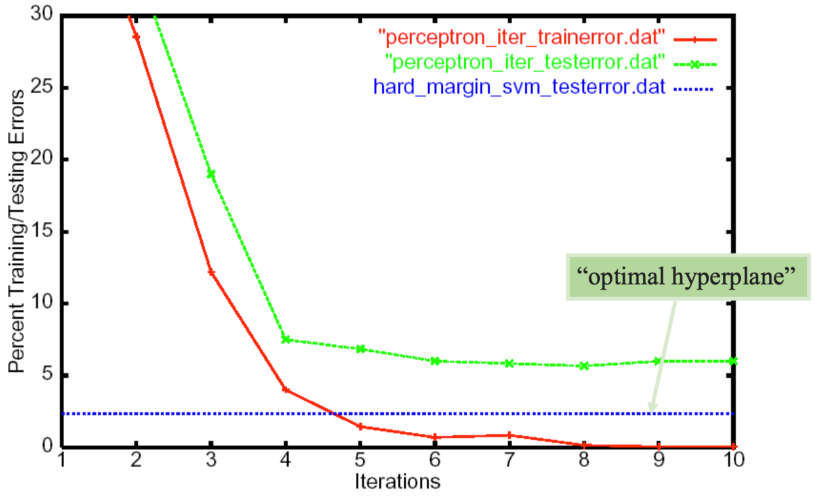
- Assumes data is linearly separable
  - ✓ **does not converge** if classes are not linearly separable
- Different correct solutions can be found
  - ✓ most are not optimal in terms of generalization
- Averaged Perceptron
  - ✓ returns a weighted average of earlier hypotheses

---

# Parameters vs Hyperparameters

- Parameters
  - ✓ weights and bias
- **Hyperparameters**
  - ✓ number of epochs (one epoch is one pass over the training data)

# Example: Reuters Text Classification



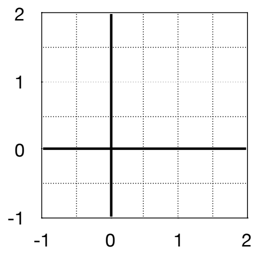
“optimal hyperplane”

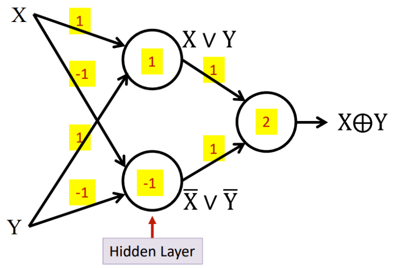
Credit: http://www.cs.cornell.edu/courses/cs4780/2019fa/lectures/06-perceptron.pdf

---

# Minsky & Papert (1969)

- Perceptrons
  - ✓ influential book
  - ✓ analyzed the algorithm and showed limitations (e.g. XOR)





http://deeplearning.cs.cmu.edu/document/lecture/lecture-1.pdf

---

# Example

| X0 | X1 | X2 | Y |
|---|---|---|---|
| 1 | 0 | 0 | -1 |
| 1 | 1 | 0 | +1 |
| 1 | 1 | 1 | +1 |
| 1 | 0 | 1 | +1 |

$$\sigma(z) = \begin{cases} +1 & \text{if } z > 0 \\ -1 & \text{if } z \leq 0 \end{cases}$$

$$h_w(\mathbf{x}) = \sigma\left(\sum_{i=0}^{d} w_i x_i\right) = \sigma(\mathbf{w}^T\mathbf{x})$$

$$L_{0/1}(h, \mathcal{D}) = \frac{1}{n} \sum_{(x_i,y_i)\in\mathcal{D}} I(h(x_i) \neq y_i)$$

$$\mathbf{w}_a = [0,0,0]^T$$
$$\mathbf{w}_b = [0,1,0]^T$$
$$\mathbf{w}_c = [-1,2,2]^T$$

$$L_{0/1}(h_{w_a}, \mathcal{D}) = ?$$
$$L_{0/1}(h_{w_b}, \mathcal{D}) = ?$$
$$L_{0/1}(h_{w_c}, \mathcal{D}) = ?$$

---

# Applying SGD to the Perceptron

# Back to the perceptron …

$$L(\mathbf{w}) = \sum_{i=1}^{n} - y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)}$$

$$\mathbf{w}^* = \underset{\mathbf{w}}{\arg\min}\ L(\mathbf{w})$$

Note this loss function has a problem, it is not bounded below

---

# Gradient

With respect to a single $w_j$

$$\frac{\partial}{\partial w_j} L(\mathbf{w}) = \frac{\partial}{\partial w_j}\left[- \sum_{i=1}^{m} y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)}\right]$$

$$= - \sum_{i=1}^{m} y^{(i)} x_j^{(i)}$$

The training algorithm can focus on batches of misclassified instances, then the batch loss cannot be negative

---

# Pseudocode