Deep Learning Lecture 2: The Perceptron (cont)

Abdelrahman Khaled

Machine Learning Research Cluster
German University in Cairo

April 8, 2019

Outline

Recall: Perceptron Update Rule

Previously we used this equation to calculate the error and update the weights:

$$\Delta w_i = (t_i - y_i)x_i$$

We also said that for that specific purpose, proportionality matters a lot more than magnitude, so in that sense the below equation is equivalent (sufficient is a better term here).

$$\Delta w_i = t_i x_i$$

Perceptron Convergence Theorem

Theorem

Given an untrained perceptron with an initial set of weights w and dataset X with n datapoints.
Assuming:

- *There exist an optimal set of weights w^* such that $\|w^*\| = 1$*
- *There exist constants $\gamma > 0$ and $R > 0$.*
- *That for all $i = 1 \dots n$, $\|x_i\| \leq R (x_i \in X)$*

Such that $t_i(x_i \cdot w^*) \geq \gamma$

Then the perceptron algorithm makes at most $\frac{R^2}{\gamma^2}$ errors before being fully trained.

Perceptron Convergence Proof

Proof.

Proof Idea: If we can bound $\|w^k\|$ we can use those bounds to bind k
Let w^k denote the weight vector after the algorithm makes its kth error.
Let $w^0 = 0$

Perceptron Convergence Proof

Proof.

Proof Idea: If we can bound $\|w^k\|$ we can use those bounds to bind k

Let w^k denote the weight vector after the algorithm makes its kth error.

Let $w^0 = 0$

- Take the dot product between w^k and w^*

Perceptron Convergence Proof

Proof.

Proof Idea: If we can bound $\|w^k\|$ we can use those bounds to bind k

Let w^k denote the weight vector after the algorithm makes its kth error.

Let $w^0 = 0$

- Take the dot product between w^k and w^*
- Since $\|w^*\| = 1$, therfore

$$w^k \cdot w^* \leq \|w^k\| \tag{1}$$

Perceptron Convergence Proof

Proof.

- Assuming the kth error occurs on example i, we have:

$$\begin{aligned} w^k \cdot w^* &= (w^{k-1} + t_i x_i) \cdot w^* \\ &= w^{k-1} \cdot w^* + t_i x_i \cdot w^* \\ &\geq w^{k-1} \cdot w^* + \gamma \end{aligned}$$

Perceptron Convergence Proof

Proof.

- Assuming the kth error occurs on example i, we have:

$$\begin{aligned} w^k \cdot w^* &= (w^{k-1} + t_i x_i) \cdot w^* \\ &= w^{k-1} \cdot w^* + t_i x_i \cdot w^* \\ &\geq w^{k-1} \cdot w^* + \gamma \end{aligned}$$

- It follow by induction that:

$$w^k \cdot w^* \geq k\gamma \tag{2}$$

Perceptron Convergence Proof

Proof.

- Using equations 1 and 2 we can conclude the lower bound for $\|w^k\|$ is:

$$\|w^k\| \geq k\gamma \tag{3}$$

This is the lower bound.

Perceptron Convergence Proof

Proof.

- Consider the square of $\|w^k\|$

$$
\begin{aligned}
\|w^k\|^2 &= \|w^{k-1} + t_i x_i\|^2 \\
&= \|w^{k-1}\|^2 + \|t_i x_i\|^2 + 2 t_i x_i \cdot w^{k-1} \\
&\leq \|w^{k-1}\|^2 + R^2
\end{aligned}
$$

Perceptron Convergence Proof

Proof.

- Consider the square of $\|w^k\|$

$$\begin{aligned}\|w^k\|^2 &= \|w^{k-1} + t_i x_i\|^2 \\ &= \|w^{k-1}\|^2 + \|t_i x_i\|^2 + 2 t_i x_i \cdot w^{k-1} \\ &\leq \|w^{k-1}\|^2 + R^2\end{aligned}$$

- It follows by induction that:

$$\|w^k\|^2 \leq kR^2 \tag{4}$$

This is the upper bound.

Perceptron Convergence Proof

Proof.

- Combining both inequalities 3 and 4 we can derive the following inequality:

$$k^2R^2 \leq \|w^k\|^2 \leq kR^2$$

■

Perceptron Convergence Proof

Proof.

- Combining both inequalities 3 and 4 we can derive the following inequality:

$$k^2 R^2 \leq \|w^k\|^2 \leq kR^2$$

- Finally, we can simplify it to get:

$$k \leq \frac{R^2}{\gamma^2}$$

∎

Regression Problems VS. Classification Problems

In a classification problem datapoints in the given data set should fall into different **discrete** groups.
We used a perceptron to find a line separates the data.

In a regression problem the data points in the data set have no obvious group, but have a **continuous** real number value.
We will use a perceptron to find a line that **best-fits** the data.

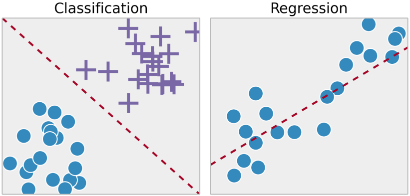


Figure: Image Source

Perceptron Regression

- We want to find the line that best fits the data.
- We can try to do that by minimizing the distance between each data point and the current line until convergence.
- A popular and effective way of doing this is Least Square Optimization.
  1. Calculate the squared distance between all data points and the line (perpendicular distance).
  2. Sum all the distnaces
  3. Minimize!

Batch Training

- The data set is split into many smaller (often disjoint) subsets.
- Datapoints in one subset are used to calculate a cumulative error which is then averaged.
- That average error is used to update the weights normally.

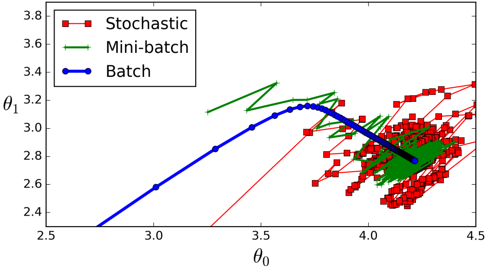


Figure: Image Source

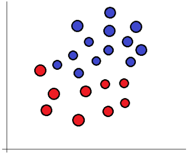
Programming Task

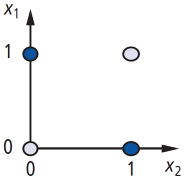
Use what you already know and implement the following:

- A Perceptron Regressor
  1. Find (or create) a data set with an approximately linear behavior.
  2. Implement a perceptron that can solve a regression problem
  3. Train your algorithm multiple times on that dataset and record your results.
- Batch Training VS. Online Training
  1. Tweak the Perceptron that we already made so that it can be trained using batch training.
  2. Change the batch size, log the results and compare them.

Non-Linearly Separable Problem

A non-linearly separable problem is one where an assortment of lines or a curve would better classify the data than just a single line. A simple example of which is the XOR problem shown on the right.





The Multi-Layer Perceptron

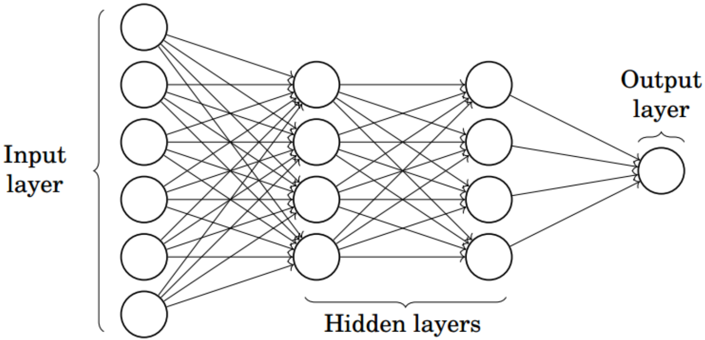
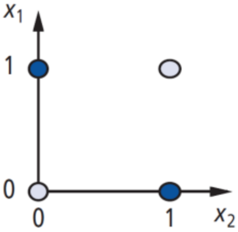


Figure: Image Source

We couldn't solve the XOR problem before, maybe we can now?

Solving the XOR Problem

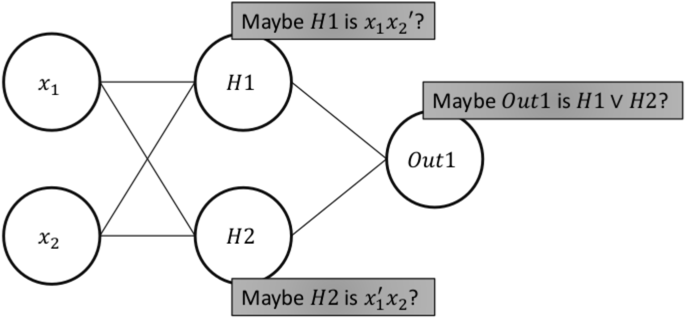
$$x_1 \otimes x_2$$



$$(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$$

$$x_1 x_2{}' + x_1' x_2$$

Solving the XOR Problem



Next Time

1. Multi-Layer Perceptron Continued.
2. Overfitting.
3. Testing and Validating our model.

References

- Goodfellow, Bengio, Courville. *Deep Learning*. Massachusetts: MIT Press, 2017. **Chapter 6**
- Chapman, Hall. *Machine Learning: An Algorithmic Perspective*. CRC Press, 2014. **Chapter 4**
- Non-academic resources (YouTube)
  - 3 blue 1 brown. **Neural Networks**. link
  - The Coding Train. **10: Neural Networks - The Nature of Code**. link