Department of Mathematics & Statistics
University of Guelph

A Brief Introduction to Deep Learning for Epidemiology

Carolyn Augusta
caugusta@uoguelph.ca

April 19, 2018

Content

What is machine learning?

- A combination of techniques from computer science, statistics, engineering, psychology, linguistics, and many other fields to teach a computer how to learn
- Basic idea: pattern recognition; focus is on feature extraction

What can machine learning do?

- Supervised learning (focus of today's tutorial)
    - Classification
    - Regression
- Unsupervised learning
    - Clustering
    - Density estimation
- Reinforcement learning (often about robotics)
    - Navigation
    - Gaming

What is deep learning?

- Traditional methods are not as good in the face of big data
- Use a neural network to perform feature extraction
- Discovering higher-order correlations among the input dimensions

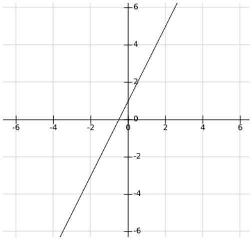
What can deep learning do?

- Image and speech recognition
- Recommender systems
- Translation (like Google Translate)
- All tasks that can be accomplished via machine learning can be accomplished via deep learning: it’s about choosing the right tool for the job.

Basic foundations

Review: what is a function?

For our purposes, a function is a mathematical relation that takes some input (x), transforms it somehow, and produces an output f(x).



For example, $f(x) = 2x + 1$ is a familiar function that takes an input x and transforms it (multiplies by 2 and adds 1) to give an output f(x). If we input $x = 2$ then we get output $f(x) = 5$.

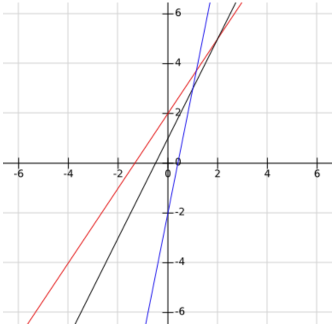
Basic foundations
Review: how do we manipulate functions?

Recall linear regression:
$f(x) = wx + b$

Find w and b to get the line of best fit

Different values of w and b change the slope and y-intercept of the function.

Black: $f(x) = 2x + 1$
Red: $f(x) = 1.5x + 2$
Blue: $f(x) = 5x - 2$

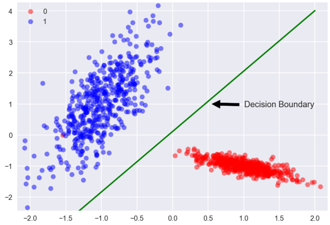


Basic foundations
What if the response is not numeric?

Say our output is binary: that is, we want to say whether an input x belongs to class 0 or class 1.

$$f(x) = \begin{cases} 0 & \text{if } wx + b > k \\ 1 & \text{otherwise} \end{cases}$$

We find values w and b that give us the best separating line between class 0 (fail) and class 1 (pass). [1]



Basic foundations
What if the response is not numeric?

Example of logistic regression

Suppose we have 10 students in an epidemiology class. Each spends a different number of hours studying. We record the number of hours spent studying, and whether the students passed or failed the exam (1 = pass, 0 = fail).

Student	1	2	3	4	5	6	7	8	9	10
Hours (x)	10	8	3	4	8	2	1	2.5	12	9
Pass (f(x))	1	1	0	0	1	0	0	0	1	1

We can’t use linear regression: our output is binary (a class label). We’ll use logistic regression.

Basic foundations
What if the response is not numeric?

Example of logistic regression

To decide whether an input number of study hours led to passing or failing the exam, we need to define the probability of passing the exam, given a number of study hours $P(Y = 1 \mid x)$.

We can't just use $P(Y = 1 \mid x) = wx + b$, because that would give us a numeric value anywhere from $-\infty$ to $+\infty$.

We'll use the logistic function:

$$P(Y = 1 \mid x) = 1/(1 + exp(-(wx + b))$$

What does this look like?

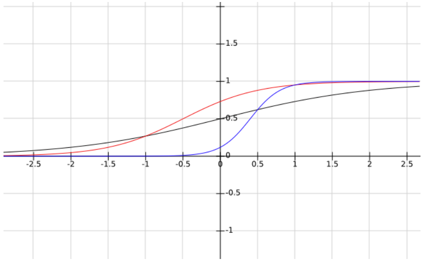
Basic foundations

What if the response is not numeric?

Sigmoid (logistic) function:

$f(x) = 1/(1 + \exp(-(wx + b))$

Different values of w and b change the curviness of the function.



Black: $f(x) = 1/(1 + \exp(-x)) = \sigma(x)$
Red: $f(x) = 1/(1 + \exp(-(1.5x + 2)) = \sigma(1.5x + 2)$
Blue: $f(x) = 1/(1 + \exp(-(5x - 2)) = \sigma(5x - 2)$

Note the sigmoid function $\sigma(x)$ has output that lies between 0 and 1, so the output can be interpreted as a probability.

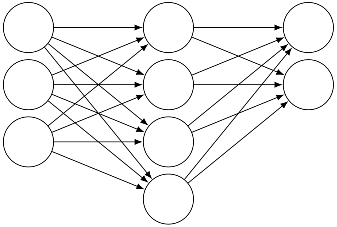
Basic foundations
Concept check

- Linear regression worked for real-valued output (numbers)
- Logistic regression works for output between 0 and 1 (probability of success/failure)
- In linear regression we relate a function of x to $y = f(x)$ via $f(x) = wx + b$ (remember $y = b_0 + b_1 x$? Same thing!)
- In logistic regression we relate a function of x to the probability of success $P(Y = 1 \mid x) = f(x)$ via: $f(x) = 1/(1 + exp(-(wx + b)))$

Hold that thought...wasn’t this a presentation on deep learning?
Let’s get into the good stuff - neural networks!

Basic foundations
What is a graph?

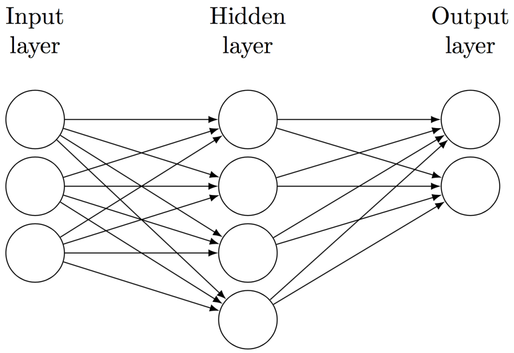
A graph $\mathcal{G}$ is a set of vertices (nodes) $\mathcal{V}$ and edges (links) $\mathcal{E}$.
We can group nodes into layers to form a network.



Basic foundations
What is a feed forward neural network?

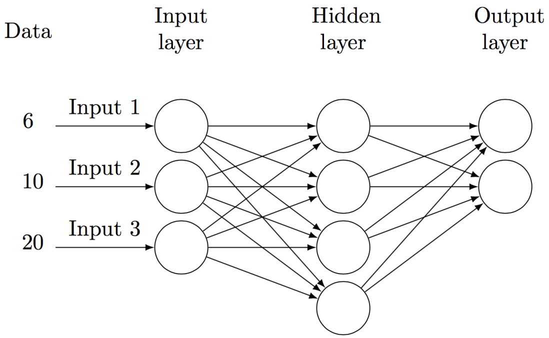
Also called a multilayer perceptron (MLP), a graph that has an input layer, at least one hidden layer, and an output layer.

Nodes are functions; arrows show the path the data follow through the network.



Basic foundations
The input layer

Toy example: an epidemic that lasts 3 days infects 6 individuals on day 1, 10 individuals on day 2, and 20 individuals on day 3. We want to know which of two classes this epidemic belongs to.



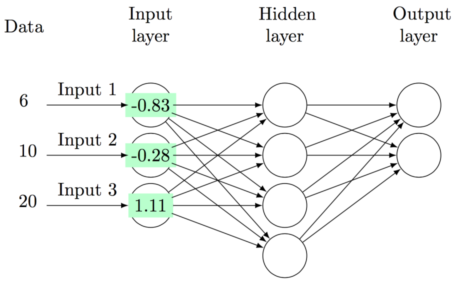
Basic foundations
The input layer

We give the input to the neural network by normalizing the data (subtract the mean $\bar{x} = 12$ and divide by the standard deviation $s = 7.211103$). This happens 'inside' each input node. Nodes are functions.

$$i_1 = \frac{6 - 12}{7.211103} \approx -0.83$$

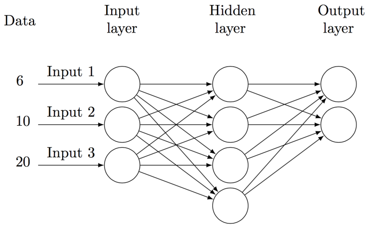
$$i_2 = \frac{10 - 12}{7.211103} \approx -0.28$$

$$i_3 = \frac{20 - 12}{7.211103} \approx 1.11$$

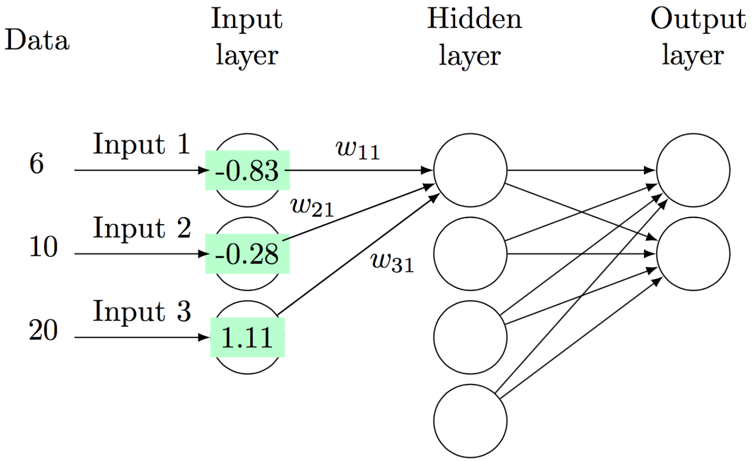


Basic foundations
The hidden layer

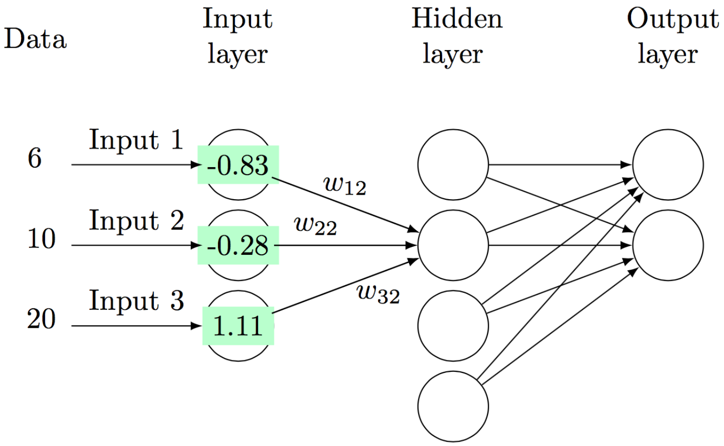
- Goal: find the best curvy line that separates the two classes.
- Need something nonlinear - sigmoid functions!
- Weights from the input layer to hidden layer
- Recall $\sigma(wx + b) = 1/(1 + \exp(-(wx + b))$
- We have one value of *w* for every link from the input layer to the hidden layer
- We have one value of *b* for every hidden unit in the hidden layer
- The hidden layer is the feature extraction (discovery) engine



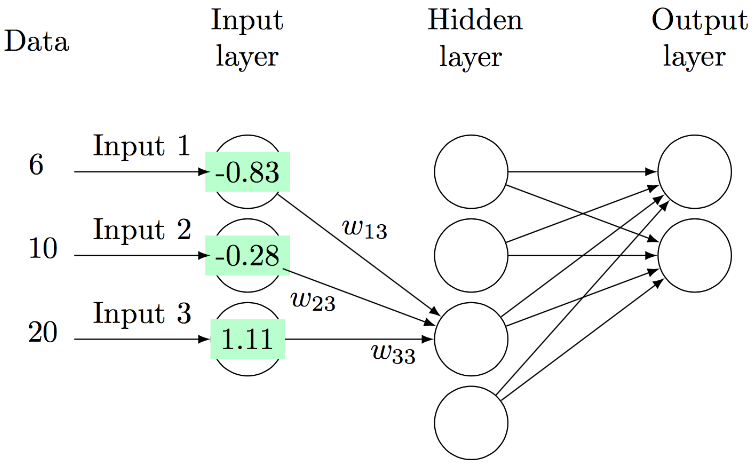
Basic foundations
The hidden layer



Basic foundations
The hidden layer

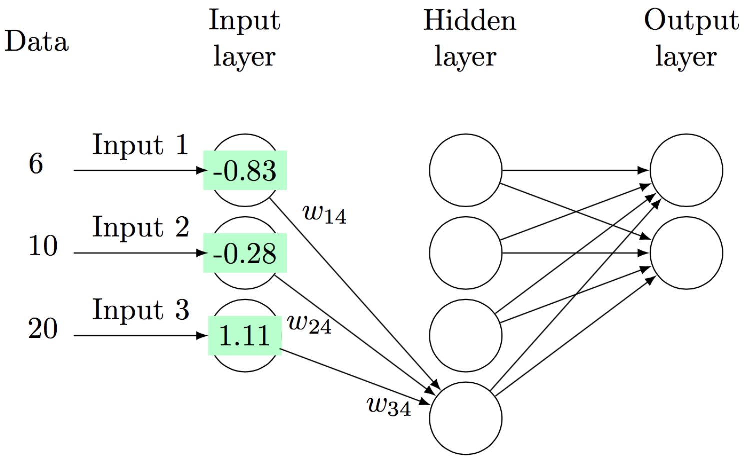


Basic foundations
The hidden layer



Basic foundations

The hidden layer

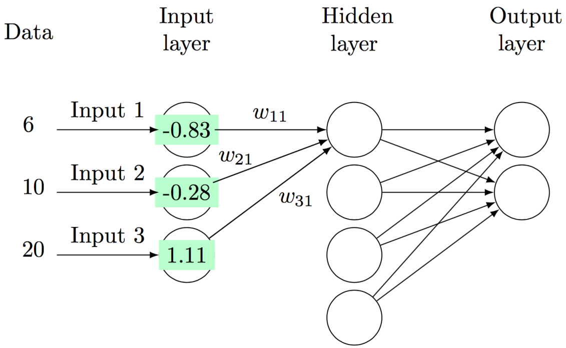


Basic foundations

The hidden layer

Each hidden node computes $\sigma\left[\sum_{i=1}^{3}(w_{ih}x_i) + b_h\right]$, where i indexes the input layer and h indexes the hidden layer.

Let's initialize our w_{ih} values to 0.1 and our b_h values to 0.2, and work out only the first hidden node for simplicity.



Basic foundations
The hidden layer

Hidden node h_1 computes $\sigma\left[\left(\sum_{i=1}^{3} w_{i1}x_i\right)+b_1\right]$

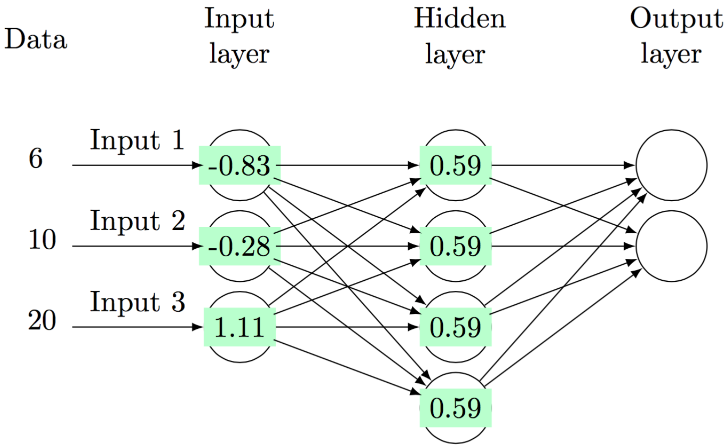
$$h_1=\frac{1}{1+\exp\left\{-\left[\left(\sum_{i=1}^{3} w_{i1}x_i\right)+b_1\right]\right\}}$$

$$\begin{aligned}-\left[\left(\sum_{i=1}^{3} w_{i1}x_i\right)+b_1\right] &= -\left[(w_{11}\cdot x_1)+(w_{21}\cdot x_2)+(w_{31}\cdot x_3)+b_1\right]\\ &= -\left[(0.1\cdot -0.83)+(0.1\cdot -0.28)+(0.1\cdot 1.11)+0.2\right]\\ &\approx -0.366\end{aligned}$$

$$h_1=\frac{1}{1+\exp\{-0.366\}}=0.5904921\approx 0.59$$

Basic foundations

The hidden layer



Basic foundations
The output layer

- The hidden layer finds useful, abstract features of the data, which are often not human-interpretable.
- The output layer will decide which class the epidemic belongs to. THIS we can interpret.
- Now we need a notion of probability: another sigmoid function!

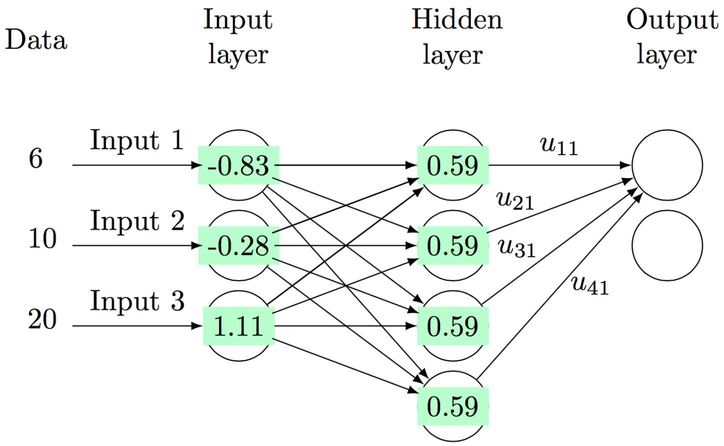
Basic foundations
The output layer

Each hidden node computes $\sigma\left[\sum_{h=1}^{4}(u_{ho}h_h) + d_o\right]$, where h indexes the hidden layer and o indexes the output layer.

Let's initialize $u_{11} = 0.1, u_{21} = 0.15, u_{31} = 0.2, u_{41} = 0.25$ and our d_o values to 0, and work out only the first output node for simplicity.

Basic foundations

The output layer



Basic foundations

The output layer

Output node o_1 computes $\sigma\left[\left(\sum_{h=1}^{4} u_{h1} h_h\right) + d_1\right]$

$$o_1 = \frac{1}{1 + \exp\left\{-\left[\left(\sum_{h=1}^{4} u_{h1} h_h\right) + d_1\right]\right\}}$$

$$\begin{aligned} -\left[\left(\sum_{i=1}^{4} u_{h1} h_h\right) + d_1\right] &= -\left[(u_{11} \cdot h_1) + (u_{21} \cdot h_2) + (u_{31} \cdot h_3) + (u_{41} \cdot h_4) + d_1\right] \\ &= -\left[(0.1 \cdot 0.59) + (0.15 \cdot 0.59) + (0.2 \cdot 0.59) + (0.25 \cdot 0.59)\right] \\ &= -0.413 \end{aligned}$$

$$o_1 = \frac{1}{1 + \exp\{-0.413\}} = 0.60180700394 \approx 0.60$$

Basic foundations
The output layer

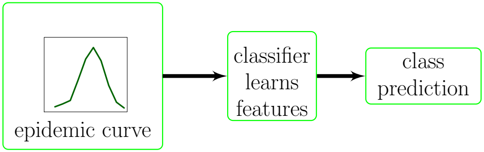
- Because the output layer gives the probability that the epidemic belongs to class 1, we know the output node for class 2 has to be $1 - 0.60 = 0.4$
- We can say that it's more likely the epidemic belongs to class 1 than to class 2
- We implement a cost function that allows the network to learn the best weights for classifying epidemics (like mean squared error, or cross-entropy error)
- The weights and biases are then updated via stochastic gradient descent-like algorithms

What does a real example look like?

- In reality, there are usually multiple hidden layers, and not all neural networks are MLPs.
    - Time series models: recurrent neural networks
    - Autoencoders
- We can apply this idea to more a more general problem

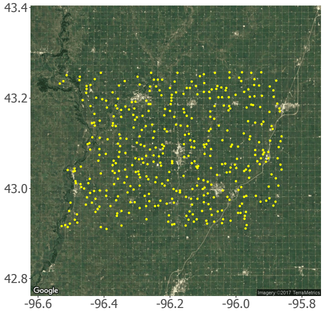
Example problem: classify epidemic curves

- General idea: we have a huge library of epidemic curves. Given a new epidemic curve, match it to the model that generated the curve.
- Advantages: fast, accurate prediction of epidemic properties without computational intensive likelihood calculations
- Disadvantage: initial training takes a long time
- Cool note: this is very similar to what Dr. Greer referenced last week regarding images being classified regarding TB without a radiologist.



Example problem: classify epidemic curves

- Swine farms in Sioux County, Iowa, USA
- Locations simulated from Farm Location and Agricultural Production Simulator (FLAPS) [2]
- 413 farms in an area of just under 2000 km^2



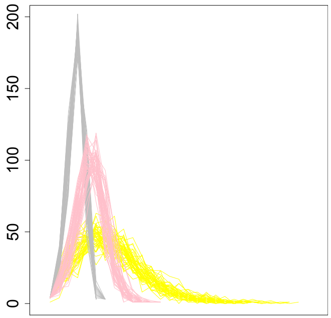
Example problem: classify epidemic curves

Individual-level models generating SIR epidemics [3]

$$P(i,t) = 1 - \exp\left\{ -\alpha \sum_{j \in I(t)} \kappa(i,j) + \epsilon \right\}$$

- α: level of susceptibility
- β: level of spatial infectivity
- ϵ: random sparks
- $I(t)$: set of infectious individuals at time t
- $\kappa(i,j)$: distance between infectious j and susceptible i
- γ: infectious period (days)

Colour	yellow	grey	pink
α	0.4	0.4	0.4
β	3	1.5	2
γ	2	1	1
ϵ	0.1	0.1	0.05



Example problem: classify epidemic curves

Tutorial (in Python) of how to train an MLP to predict the class of epidemic curves

▸ as a Jupyter notebook

References

[1] A. Dertat, Applied Deep Learning - Part 1: Artificial Neural Networks. Blog post. https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6

[2] C. Burdett, B. Kraus, S. Garza, R. Miller, K. Bjork, Simulating the distribution of individual livestock farms and their populations in the United States: An example using domestic swine (sus scrofa domesticus) farms, PloS one 10 (11) (2015) e0140338.

[3] C. Augusta, R. Deardon, and G. W. Taylor. Deep learning for classifying epidemic curves. [Under review] Spatial and Spatio-Temporal Epidemiology.

Thank you!