B455 HW2

Zac Monroe

February 2019

1. See `pcn_example_prob1.ipynb`.

2. See `lin_reg_prob2.ipynb`.

3. $$\ln\left(\frac{p(disease|x)}{1-p(disease|x)}\right) = \mathbf{w}^T \cdot \mathbf{x};\ \mathbf{w} = \begin{bmatrix}\beta_0\\ \beta_1\\ \beta_2\\ \beta_3\\ \beta_4\\ \beta_5\end{bmatrix},\ \mathbf{x} = \begin{bmatrix}1\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\end{bmatrix} = \begin{bmatrix}bias\\ AGE\\ OBS\\ E\\ AGEE := AGE * E \equiv x_1 * x_3\\ OBSE := OBS * E \equiv x_2 * x_3\end{bmatrix}$$

$$\iff \ln\left(\frac{p(disease|x)}{1-p(disease|x)}\right) = \begin{bmatrix}\beta_0 & \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5\end{bmatrix} \cdot \begin{bmatrix}1\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\end{bmatrix} = \beta_0 + \sum_{i=1}^{5} \beta_i x_i.$$

The model can be learned from the data with gradient descent. The function to minimize (the function whose gradient we are calculating and adding a small step in the negative of that gradient direction) is our error function, MLE error:

$$E(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} ln(1 + e^{-y_i \mathbf{w}^T \cdot \mathbf{x}}), \tag{1}$$

where y_i is the output of the model. In other words, we update the weight parameters at the k-th iteration ($\mathbf{w}^{(k)}$) to the weight parameters of the $k+1$-th iteration ($\mathbf{w}^{(k+1)}$) by this assignment:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \frac{\nabla E}{\|\nabla E\|} \tag{2}$$

After enough iterations, we will converge on a single weight vector that will be our model's final parameters that give us a minimum error value. In this way, the model can be properly learned from the input data.

4. (a) I could use an MLP to make the prediction by first deciding on a time window with which I would move over the whole dataset. Let's say that I choose a window size k (which is clearly one of the parameters of the network that I would have to choose). So the MLP would have $k+1$ input neurons, where the first input neuron represents the bias node whose activation is always -1.
   Prior to training the model, I would pre-process the input data by linearly normalizing the expected range (80-400) to a range between -1 and 1, i.e. $x_i \leftarrow \frac{x_i - (400+80)/2}{(400-80)/2}$

for each x_i in the given data.

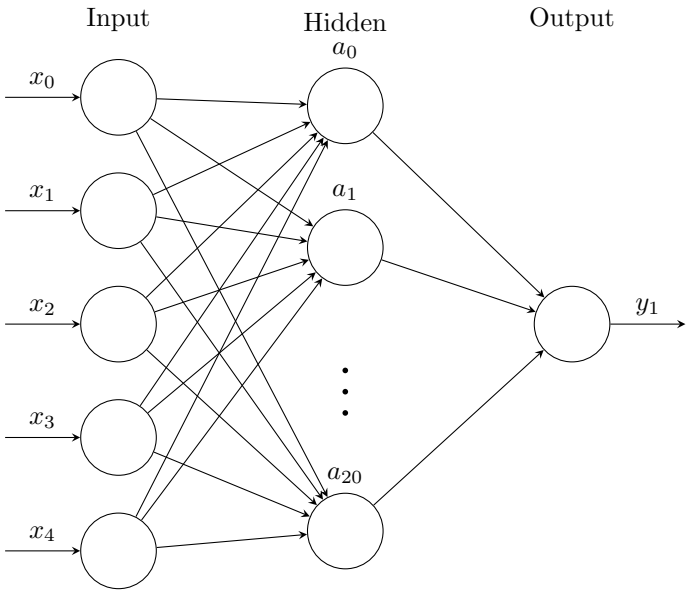
Suppose there are N data points ($N \approx 5 \cdot 365 = 1825$ because there are 5 years of data; ignoring leap days). There are then $N - (k-1)$ individual data vectors in $\mathbb{R}^k$ that we can use as inputs.

I would set $k = 5$, so that each input vector has the data for the last five days. Then, using 10-fold cross-validation, there would be about $0.9 * (1825 - 5) = 1638$ training data points for each sub-model to be validated. If I choose the number of hidden neurons in the single hidden layer to be 15 ($\approx 3 * k$), there would be $(1 + 5) \cdot 15 + 5 \cdot 15 = 165$ weights to be trained, which is roughly 10% the quantity of training data that we have, so the empirical guidelines for ratio of hidden neurons to input neurons and total weights to training data set size are met.

(b) I would add that into the system by changing the time window to $k = 4$ and having the inputs be the bias, weather, and the previous 4 days of power consumption. I would normalize the weather data similarly to the consumption data but with a different sensible ideal range (i.e. map temperatures of -10^{o}C through 30^{o}C to a range of -1 to 1). I would expect this to help the system by a considerable amount because the weather determines how much people use heating/air conditioning systems to alter their inside temperature; more extreme temperatures tends to lead to higher power consumption.

(c) I do think that this system would work well for predicting power consumption. There are plenty of demands that it would not be able to predict, however. Things like power outages (after a storm, sometimes the power is out for lots of people at a time, and often only for a couple of days; these are often unpredictable) and large-scale events (like the superbowl or any other large sports game which requires lots of power in a power-inefficient stadium) or a widespread celebration that uses power (i.e. Christmas lights on peoples' houses) are also things that the system cannot necessarily predict just because the window is only 5 days, and not a whole year.

5. My MLP for this problem as a directed graph appears below.



$x_0 = 1$,
$x_1 =$ number of people in ward each week,
$x_2 =$ weather,
$x_3 =$ season $\in \{1, 2, 3, 4\}$,
$x_4 =$ epidemic $\in \{0, 1\}$.

I chose the number of hidden neurons to be 4 times the number of inputs (including the bias input) because it is in the middle range of 3-5 times, which is an empirical range that seems to work properly for MLPs. We would have $\tilde{5} * 365 = 1825$ data points, split into 5 for 5-fold cross-validation, which would leave us with $\tilde{1}460$ data points for training, which is more than 14x the number of weights that our model would have (21*5 + 21 = 126), so the other empirical piece of advice (ratio of parameter count to training examples ¡ $\frac{1}{10}$) holds true in our example as well.

The only other input that I considered was a bias node, whose activation is always 1.

I would pre-process the data by normalizing each continuous field between -1 and 1, i.e.

$$x_i \leftarrow \frac{x_i - (max(x_i) + min(x_i))/2}{range(x_i)/2} \text{ for } i \in \{1, 2\}.$$

I would expect the system to work mostly well. There would need to be a decent amount of tweaking done to the hyper-parameters in order for it to work as well as it can, and there would definitely be a statistically significant margin of error, but it would work as well enough at approximating the number of beds needed in the geriatric ward as such a network could. I expect at least 20% margin of error, but it's hard to know if that's correct without computing the network concretely.

6. 1) If all $k = 3$ nearest neighbors are given equal weight, then U will be classified as Red. Two out of the three closest neighbors are Red, while one out of the closest three is Orange, and 2 ¿ 1 so Red prevails as the final tally of the vote between the neighbors of U.

   2) If vote weights are determined by $w = \frac{1}{d^2}$ with $k = 3$, here are the votes:

Label	Width	Height	$\frac{1}{d^2}$ to U
Red	1	2	$\frac{1}{2^2} = 0.25$
Orange	2	5	$\frac{1}{1^2 + 1^2} = 0.5$
Red	2	6	$\frac{1}{1^2 + 2^2} = 0.2$

   $0.5 > 0.25 + 0.2$, so the weights given to Orange's vote outweigh Red's vote weights, so U would be classified as Orange.