1) Let’s say you have two problems — a linear regression and a logistic regression (classification). Which one of them is more likely to be benefited from a newly discovered super-fast large matrix multiplication algorithm? Why?

Linear regression because fast matrix multiply help computing inverse in Normal Eqs

2) You are asked to build a classification model about meteorites impact with Earth (important project for human civilization). After preliminary analysis, you get 99% accuracy. Should you be happy? Why not? What can you do about it?

No, because there are rare cases. If you predict no all the time you will get that. You can reframe the problem as predicting meteorites which pass below a given distance from Earth. Increasing that distance increases the number of positive cases in your dataset.

3) You are working on a classification problem. For validation purposes, you’ve randomly sampled the training data set into train and validation. You are confident that your model will work incredibly well on unseen data since your validation accuracy is high. However, you get shocked after getting poor test accuracy. What went wrong?

Hyperparameters overfit to the validation set (e.g. learning rate)

4) You decide that the vision tests given by eye doctors would be more precise if we used an approach inspired by logistic regression. In a vision test a user looks at a letter with a particular font size and either correctly guesses the letter or incorrectly guesses the letter. You assume that the probability that a particular patient is able to guess a letter correctly is: $p = \sigma(\theta + f)$ Where \( \theta \) is the user’s vision score and \( f \) is the font size of the letter. Explain how you could estimate a user’s vision score (\( \theta \)) based on their 20 responses \((f_1,y_1), \ldots, (f_{20},y_{20})\) where \( y_{(i)} \) is an indicator variable for whether the user correctly identified the ith letter and \( f_{(i)} \) is the font size of the ith letter. Solve for any and all partial derivatives required by the approach you describe in your answer.
We are going to solve this problem by finding the MLE estimate of $\theta$. To find the MLE estimate, we are going to find the argmax of the log likelihood function. To calculate argmax we are going to use gradient ascent, which requires that we know the partial derivative of the log likelihood function with respect to theta.

First we write the log likelihood:

$$L(\theta) = \prod_{i=1}^{20} p^{y_i^{(i)}}(1-p)^{1-y_i^{(i)}}$$

$$LL(\theta) = \sum_{i=1}^{20} (y_i^{(i)} \log(p) + (1-y_i^{(i)}) \log(1-p))$$

Then we find the derivative of log likelihood with respect to $\theta$. We first do this for one data point:

$$\frac{\partial LL}{\partial \theta} = \frac{\partial LL}{\partial p} \cdot \frac{\partial p}{\partial \theta}$$

We can calculate both the smaller partial derivatives independently:

$$\frac{\partial LL}{\partial p} = \frac{y_i^{(i)}}{p} - \frac{1-y_i^{(i)}}{1-p}$$

$$\frac{\partial p}{\partial \theta} = p(1-p)$$

Putting it all together for one letter:

$$\frac{\partial LL}{\partial \theta} = \frac{\partial LL}{\partial p} \cdot \frac{\partial p}{\partial \theta}$$

$$= \left[ \frac{y_i^{(i)}}{p} - \frac{1-y_i^{(i)}}{1-p} \right] p(1-p)$$

$$= y_i^{(i)}(1-p) - p(1-y_i^{(i)})$$

$$= y_i^{(i)} - p$$

$$= y_i^{(i)} - \sigma(\theta - f)$$

For all twenty examples:

$$\frac{\partial LL}{\partial \theta} = \sum_{i=1}^{20} y_i^{(i)} - \sigma(\theta + f^{(i)})$$

5) Let’s explore saliency, a measure of how important a feature is for classification. We define the saliency of the ith input feature for a given example (x, y) to be the absolute value of the partial derivative of the log likelihood of the sample prediction, with respect to that input feature $\frac{\partial LL}{\partial x_i}$. In the images below, we show both input images and the corresponding saliency of the input features (in this case, input features are pixels):
First consider a trained logistic regression classifier with weights $\theta$. Like the logistic regression classifier that you wrote in your homework it predicts binary class labels. In this question we allow the values of $x$ to be real numbers, which doesn’t change the algorithm (neither training nor testing).*

a) What is the Log Likelihood of a single training example $(x, y)$ for a logistic regression classifier?

b) Calculate the saliency of a single feature $(x_i)$ in a training example $(x,y)$.

c) Show that the ratio of saliency for features $i$ and $j$ is the ratio of the absolute value of their weights $\frac{|\theta_i|}{|\theta_j|}$.

---

a. $LL(\theta) = y \cdot \log \sigma(\theta^T \cdot x) + (1 - y) \log [1 - \sigma(\theta^T \cdot x)]$

b. We can calculate the saliency for a single feature as follows.

$$\frac{\partial LL}{\partial x_i} = \frac{\partial LL}{\partial z} \cdot \frac{\partial z}{\partial x_i}$$

chain rule

$$\frac{\partial LL}{\partial z} = \left( \frac{y}{z} - \frac{1 - y}{1 - z} \right) \cdot \left( z(1 - z)\theta_i \right)$$

partial derivatives

saliency = $\left| \frac{y}{z} - \frac{1 - y}{1 - z} \right| z(1 - z)\theta_i$

---

c. We can take the ratio as follows using our expression above.

$saliency$ for feature $i$, $S_i = \left| \left( \frac{y}{z} - \frac{1 - y}{1 - z} \right) z(1 - z)\theta_i \right|$, and same for $S_j$

$$\frac{S_i}{S_j} = \left| \left( \frac{y}{z} - \frac{1 - y}{1 - z} \right) z(1 - z)\theta_i \right| \cdot \frac{S_i}{S_j} = \left| \theta_i \right| \text{ by elimination}$

---

6) Consider the dataset: $D = \{x_{(1)}, ..., x_{(100)}\}$ with $x$ in $R^3$. The problem is a 3-class classification problem. Consider a neural network architecture with 2 hidden layers of dimension 4 and 5 - using a sigmoid and a softmax respectively.
a) How would you represent graphically this neural network?

b) What are the feedforward equations for this neural network?

For a single training example $x$ in $\mathbb{R}^3$ (here, we do not consider the $x_0$ set as 1 previously used to account for the bias), so $x$ is $(x_1, x_2, x_3)$

$$z_1 = W_1 x + b_1$$

$$a_1 = g(z_1)$$

$$z_2 = W_2 a_1 + b_2$$

$$a_2 = g(z_2)$$

$$h = \text{softmax}(a_2)$$

Similar notations can be adopted with a matrix $\Theta$ that incorporates the bias, and $x$ incorporating a $x_0$ component equal to 1

c) Describe the relationship between the graphical representation and the feedforward equations.

What do the nodes represent? What do the edges represent?

- 1 node = 1 component of a vector ($x$ or activation)
- 1 edge = 1 vector component * 1 matrix weight
  - Edges that point to same node mean that the scalars they represent are summed together to obtain the value of the node

d) What is the total number of parameters in this neural network?

Number of weights per equation:

1. $4 \times 3 + 4 = 16$
2. 0
(3) 5 x 4 + 5 = 25
(4) 0
(5) 3 x 5 + 3 = 18
Total = 59 parameters

7) Why should features be normalized when applying regularization to logistic regression?
Consider a simple example: you try to predict height in meters from leg length x1 in meters and arm length x2 in centimeters

Model: \( y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \)

Where the true hypothesis is
\( y = 1 \ast x_1 + (1/100) \ast x_2 \)

If we regularize those weights, \( \theta_1 \) will be heavily regularized compared to \( \theta_2 \) because its value is 100x higher. However this value difference is just due to a difference in units.

Note that even if variables are measured in the same units, their orders of magnitude can be different, so normalization still must be applied.

Remark: in the height example, you should notice that the two variables considered are correlated (leg length and arm length), so in all rigor those two variables should not be both used as input of a linear regression model.