Explanatory Notes for 6.390

Shaunticlair Ruiz (Current TA)

Fall 2022

Learning LLCs: Loss Functions

Now that we have fully **built up** LLCs, we can start trying to **train** our own.

In order to do that, we need a way to **evaluate** our hypotheses: a **loss function**.

Earlier in the chapter, we tried **0-1 Loss**:

$$\mathcal{L}_{01}\Big(h(x;\Theta), \quad y\Big) = \begin{cases} 0 & \text{if } y = h(x;\Theta) \\ 1 & \text{otherwise} \end{cases}$$

But, this **loss** function the same problem our **sign** function did: it isn't **smooth**!

It's a **discrete** function based on our **discrete classes**: so, it won't have a smooth **gradient** we can do **descent** on.

For our **sign** function, we switched to the **sigmoid** function, which measures in terms of **probabilities**: this gave us some **smoothness** to our classification.

Could we do the same here?

Building our new loss function

So, the **output** of our sigmoid $\sigma(u)$ is a **probability**: how **likely** do we think a point is to be in class $+1$?

We want a loss function

$$\mathcal{L}(g, y) \tag{1}$$

That considers two facts: the **correct** answer y, and how likely we **expected** $+1$ to be, $g = \sigma(u)$.

Notation 1

For our **loss function**, rather than using $y \in \{-1, +1\}$, we'll use $y \in \{0, 1\}$.

That way, $\sigma(u)$ and y **match**:

$$y \in \{0, 1\} \qquad\qquad g \in (0, 1)$$

So, if the correct is 1, then we want $\sigma(u)$ to be **high**. If the correct answer is 0, we want $\sigma(u)$ to be **low**.

For **one** data point, then, we can consider, "how likely did we think the right answer was?"

$$G(g, y) = \begin{cases} g & \text{if } y = 1 \\ 1 - g & \text{else } (y = 0) \end{cases} \tag{2}$$

If we to choose 1 with probability g, this could also mean, "how likely were we to be **right**?"

This G is how "**good**" our function is, so the **loss** would need for us to take the **negative**: we'll do that later.

Loss Function for Multiple Data Points

Now, how do we consider **multiple** data points? Well, let's think in terms of **probability**: guessing each point is a separate **event**.

We *could* add or **average** our guesses. But, since we're working with **probabilities**, there's a natural way to **combine** them: multiple events **occurring** at the same time.

Before, we asked, "how likely were we to be **right**?" for **one** data point. We could **extend** this question to, "how likely are we to get **every** question right?"

Well, each question we get right is an **independent** event C_i. If we want two independent events to **both** happen, we have to **multiply** their probabilities.

Key Equation 2

The probability of two independent events A and B happening at the same time is

$$\mathbf{P}\{A \text{ and } B\} = \mathbf{P}\{A\} * \mathbf{P}\{B\}$$

So if we want **all** of them, we just multiply:

$$\mathbf{P}\{E_{all}\} = \mathbf{P}\{E_1\} * \mathbf{P}\{E_2\} * ... * \mathbf{P}\{E_n\} \tag{3}$$

Written using pi notation, and also $g^{(i)}$ for multiple data points:

This notation is described in the prerequisites chapter! The short version: instead of adding terms with $\sum$, you multiply with $\prod$.

$$\mathbf{P}\{E_{all}\} = \prod_{i=1}^{n} \mathbf{P}\{E_i\} = \prod_{i=1}^{n} \begin{cases} g^{(i)} & \text{if } y^{(i)} = 1 \\ 1 - g^{(i)} & \text{if } y^{(i)} = 0 \end{cases} \tag{4}$$

Simplifying our expression - Piecewise

Our piecewise function is a bit **annoying**, though: is there a way to **simplify** it so that it doesn't have to be **piecewise**?

Our goal is to **combine** our two piecewise cases into a **single** equation. That means one of them needs to **cancel out** whenever the other is true.

Well, let's see what we have to **work** with.

Our **two** cases happen when $y = 0$ or $y = 1$: these are **nice** numbers! Why? Because of the **exponent** rules for these two:

- $c^0 = 1$: an exponent of 0 outputs 1: a factor of 1 in a product might as well **not be there**. It has been effectively **cancelled** out.

- $c^1 = c$: an **exponent** of 1 leaves the factor **unaffected**.

So, let's consider the **first** case, g. we can use g^y: if $y = 1$, it's **unaffected**. If $y = 0$, the term is **removed**.

We want the **opposite** for $1-g$. We can **swap** 1 and 0 by doing $1-y$. This gives us $(1-g)^{1-y}$.

For one data point:

$$\mathbf{P}\{E\} = \overbrace{g^y}^{y=1} \overbrace{(1-g)^{1-y}}^{y=0} \tag{5}$$

We've gotten rid of the piecewise function! Let's add back in the product:

$$\mathbf{P}\{E_{all}\} = \prod_{i=1}^{n} \mathbf{P}\{E_i\} = \prod_{i=1}^{n} g^{(i)\,y^{(i)}} \left(1 - g^{(i)}\right)^{1-y^{(i)}} \tag{6}$$

Looks pretty ugly, but we'll work on that.

Getting rid of the product

Our exponents look pretty **ugly**. Can we do something about that?

More importantly, **products** are also pretty unpleasant: we can't use **linearity**!

Linearity makes lots of problems easy to work with, so we try to keep it.

Linearity uses **addition** between variables. What sort of **function** could change a **product** into a **sum**?

Well, we could **list** out different basic functions, to see which ones connect sums and products. It turns out, one **interesting** function is

$$\overbrace{\log ab}^{\text{product}} = \overbrace{\log a + \log b}^{\text{sum}} \tag{7}$$

Aha! If we take the **log** of our function, we can turn a **product** into the **sum**!

The below equation looks complicated, but all we've done is swap the product for a sum!

$$\overbrace{\log\left(\prod_{i=1}^{n} \mathbf{P}\{E_i\}\right)}^{\text{product}} = \overbrace{\sum_{i=1}^{n} \log\left(\mathbf{P}\{E_i\}\right)}^{\text{sum}} \tag{8}$$

We can also separate our two **factors**:

$$\sum_{i=1}^{n} \left(\log\left(g^{(i)\,y^{(i)}}\right) + \log\left(\left(1 - g^{(i)}\right)^{1-y^{(i)}}\right) \right) \tag{9}$$

And **finally**, we can remove the **exponents**:

$$\sum_{i=1}^{n} \left(y^{(i)} \log g^{(i)} + \left(1 - y^{(i)}\right) \log \left(1 - g^{(i)}\right) \right) \tag{10}$$

Concept 3

Our **negative log likelihood** (NLL) comes from a couple steps:

- **Use** $y \in \{0, 1\}$ instead of $y \in \{-1, +1\}$ so that y and g have **matching** outcomes.
- Get the **chance** the model is right on every **guess**: a **product**.
- Use **exponents** to convert the **piecewise** expression into a single **equation**.
- Take the **log** of our expression to switch from a **product** to a **sum**.
- Take the **negative** to get the **loss** rather than the "**goodness**" of our function.

Negative Log Likelihood

Remember, at the **beginning**, we said that we need to take the **negative**: our function represents how **good** our function is, but we want the **loss**.

With this, our function is in its final form:

Key Equation 4

We can get the loss of our **linear logistic classifier (LLC)** using the **negative log likelihood (NLL)** loss function

$$\mathcal{L}_{nll}(g^{(i)}, \quad y^{(i)}) = -\left(y^{(i)} \log g^{(i)} + \left(1 - y^{(i)}\right) \log \left(1 - g^{(i)}\right) \right)$$

Or,

$$-\Big((answer) \log(guess) + (1 - answer) \log(1 - guess) \Big)$$

Our total loss is

$$\sum_{i=1}^{n} \mathcal{L}_{nll}(g^{(i)}, \quad y^{(i)}) \tag{11}$$

Finally, we add our **regularizer**:

$$J_{lr}(\theta, \theta_0; \mathcal{D}) = \frac{1}{n}\sum_{i=1}^{n}\left(\mathcal{L}_{nll}(g^{(i)}, \quad y^{(i)})\right) + \lambda\|\theta\|^2 \tag{12}$$

Key Equation 5

The full **objective function** for **LLC** is given as

$$J_{lr}(\theta, \theta_0; \mathcal{D}) = \frac{1}{n}\sum_{i=1}^{n}\left(\mathcal{L}_{nll}\left(\sigma(\theta^T x + \theta_0), \quad y^{(i)}\right)\right) + \lambda\|\theta\|^2$$

Using our **loss** function $\mathcal{L}_{nll}$, and our **logistic** function $\sigma(u)$.