CIS5200: Machine Learning Spring 2023

Lecture 1: Logistic and Linear Regression

Date: January 19, 2023 *Author: Surbhi Goel*

Acknowledgements. These notes are heavily inspired by material from CIS5200 Spring 2022 and Cornell University's CS 4/5780 — Spring 2022.

1 Logistic Regression

Last lecture, we discussed Perceptron. One of the drawbacks of Perceptron was the assumption of deterministic labels. In real-life, our decisions are often non-deterministic. For instance, suppose you need to decide whether to wear a sweater when you leave house. You would check the temperature outside and decide based on that. It is often not a single temperature flip that governs this, often it depends on other factors such as how sunny or windy it is, or how you are feeling today. Thus, it is more likely that within a range you would sometimes wear a sweater and sometimes not, but your decision is more certain when it is either the temperature is too low or too high. Figure 1 captures this example.

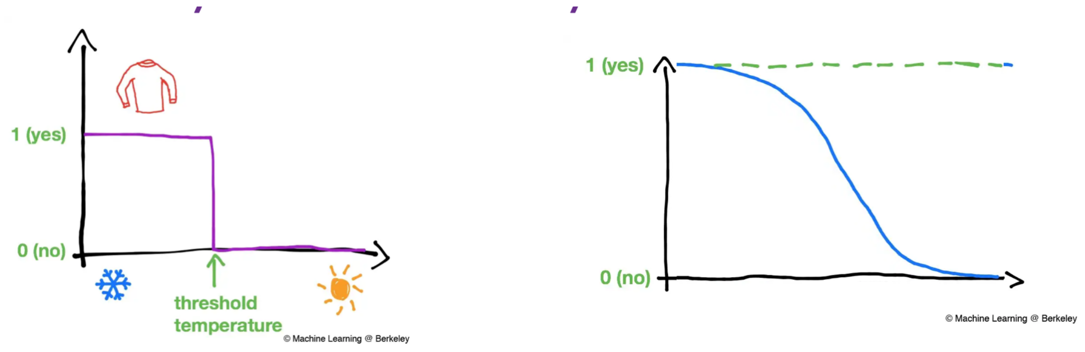

Figure 1: Decision boundaries for deciding on wearing a sweater based on outside temperature [source: `https://medium.com/@ml.at.berkeley/machine-learning-crash-course-part-2-3046b4a7f943`]

Let us formalize this by denoting $\eta(x)$ as the conditional probability of label 1 given input x under the distribution $\mathcal{D}$.

$$\eta(x) = \Pr[y = 1|x].$$

In the case of the Perceptron setup, the labels were deterministic, therefore $\eta(x) \in \{0, 1\}$. In general, $\eta(x) \in [0, 1]$.

In the setting of logistic regression, we model η using a sigmoid function applied to a linear function, in particular,

$$\Pr[y = 1|x] = \mathsf{sigmoid}(w^\top x) = \frac{1}{1 + \exp(-w^\top x)}.$$

The sigmoid function can be thought of as a smooth approximation of the sign/step function, which allows for uncertainty near the boundary.

Let us see what the decision boundary looks like here. We have,

$$\frac{\Pr[y = 1|x]}{\Pr[y = -1|x]} = \frac{\mathsf{sigmoid}(w^\top x)}{1 - \mathsf{sigmoid}(w^\top x)} = \exp(w^\top x).$$

When this ratio is 1, we are at the boundary, and when it is above 1, we predict 1 and -1 otherwise. Observe that this gives us a linear decision boundary as before. This is also known as the *Bayes Optimal* classifier. We will discuss this further in the homework or recitation (TBD).

Having set up the model, now we are ready to talk about the loss function. The loss used in logistic regression is known as the logistic loss, and it has the following form,

$$\ell(f(x), y) = \begin{cases} -\log(f(x)) & \text{if } y = 1 \\ -\log(1 - f(x)) & \text{otherwise} \end{cases}$$

For the particular function class we have chosen $\mathcal{F} := \{x \mapsto \mathsf{sigmoid}(w^\top x + b) | w \in \mathbb{R}^d, b \in \mathbb{R}\}$, this evaluates to

$$\log\left(1 + \exp(-y\ w^\top x)\right).$$

This loss is an upper bound to the 0/1 loss we used for Perceptron. It penalizes incorrect prediction heavily as the points distance from the boundary increases. See Figure 2 for a plot of the losses. The figure also has hinge loss, $\max(0, 1 - y\ w^\top x)$ which is also an upper bound on the 0/1 loss. These losses are known as *surrogate* losses which serve as an approximation to the 0/1 loss while being more amenable to optimization. Both logistic and hinge loss are convex.

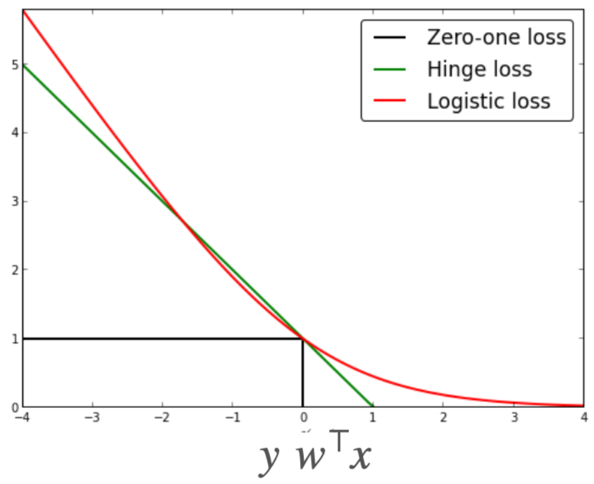

Figure 2: Comparison of 0/1 loss, logistic loss, and hinge loss.

The setup then looks like:

- Input space $\mathcal{X} = \mathbb{R}^d$
- Label space $\mathcal{Y} = [0, 1]$
- Hypothesis class $\mathcal{F} := \{x \mapsto \mathsf{sigmoid}(w^\top x + b) \mid w \in \mathbb{R}^d, b \in \mathbb{R}\}$ where $\mathsf{sigmoid}(a) = \frac{1}{1+\exp(-a)}$.
- Loss function $\ell(f(x), y) = \begin{cases} -\log(f(x)) & \text{if } y = 1 \\ -\log(1 - f(x)) & \text{otherwise} \end{cases}$.

1.1 Why logistic loss? A probabilistic perspective

There is a very good justification for using this loss, if we take a probabilistic view. An alternate view of solving a supervised learning task is to

- First pick an explicit model on the data distribution,
- Then find parameters such that they maximize the likelihood of seeing the training data S.

This approach is known as the Maximum Likelihood Approach (MLE). Suppose the parameters of the underlying model are θ, then the likelihood is

$$\mathcal{L}(\theta) = \mathcal{P}(S|\theta) = \prod_{i=1}^{m} \mathcal{P}(x_i, y_i|\theta).$$

Here the last equality follows from the i.i.d. assumption on the training dataset.

We use $\mathcal{P}$ to denote probability in the case of discrete variables, and the density of the distribution at the input. This distinction is important, since the probability of any x in a continuous distribution will be 0.

Example. Consider the data distribution being coin tosses of a coin with probability of heads being θ. Here θ parametrizes the different distributions of heads and tails we may observe when tossing the coin. Suppose we observe $S = \{t_1, \dots, t_m\}$, a sequence of outcomes of tosses, where $t_i = 1$ implies heads and -1 implies tails. Then we have,

$$\begin{aligned} \mathcal{L}(\theta) &= \mathcal{P}(S|\theta) \\ &= \prod_{i=1}^{m} \mathcal{P}(t_i|\theta) \\ &= \prod_{i=1}^{m} (\mathbb{1}[t_i = 1]\theta + \mathbb{1}[t_i = -1](1 - \theta) \\ &= \prod_{i=1}^{m} \theta^{\mathbb{1}[t_i=1]}(1 - \theta)^{\mathbb{1}[t_i=-1]} \\ &= \theta^{\sum_{i=1}^{m} \mathbb{1}[t_i=1]}(1 - \theta)^{\sum_{i=1}^{m} \mathbb{1}[t_i=-1]} \end{aligned}$$

If the number of heads is denoted by m_H, then we have,

$$\mathcal{L}(\theta) = \theta^{m_H}(1-\theta)^{m-m_H}$$

Let's take log to make this easier.

$$\log \mathcal{L}(\theta) = m_H \log \theta + (m - m_H)\log(1-\theta)$$

Now minimizing with respect to θ gives us

$$\frac{m_H}{\theta} - \frac{m - m_H}{1-\theta} = 0 \implies \theta = \frac{m_H}{m}.$$

So for an observed sequence $\{H, T, H, H\}$, the MLE estimate is 3/4.

Conditional Likelihood. Often if we do not have an underlying model on x, but only have one on $y|x$ as in logistic regression, we can use the *conditional* likelihood.

$$\mathcal{L}(\theta) = \prod_{i=1}^{m} \mathcal{P}(y_i|x_i, \theta).$$

Let us compute this for our logistic model,

$$\begin{aligned}\mathcal{L}(w) &= \prod_{i=1}^{m} \mathcal{P}(y_i|x_i, w)\\ &= \prod_{i=1}^{m} \frac{1}{1+\exp(-y_i\ w^\top x_i)}.\end{aligned}$$

The log-likelihood then is,

$$\begin{aligned}\log \mathcal{L}(w) &= \log\left(\prod_{i=1}^{m} \frac{1}{1+\exp(-y_i\ w^\top x_i)}\right)\\ &= -\sum_{i=1}^{m} \log(1+\exp(-y_i\ w^\top x_i)).\end{aligned}$$

Observe that this is just the negative of the logistic loss up to the scaling by m.

1.2 Optimizing the loss.

The logistic loss is convex in w and hence the empirical risk minimization problem turns to be convex. There is no closed form for a solution, but we can use standard convex optimization methods to find a solution. We will discuss these in the convex optimization lecture.

2 Linear Regression

We will now shift gears and move away from the classification setup. We will now look at the regression setting, where we want to predict a continuous real-valued label for our input.[1] Some examples of regression problems we saw in lectures includes stock price prediction, predicting the size of arctic sea ice over time, etc.

The setup for linear regression is:

- Input space $\mathcal{X} = \mathbb{R}^d$
- Label space $\mathcal{Y} = \mathbb{R}$
- Hypothesis class $\mathcal{F} := \{x \mapsto w^\top x + b \mid w \in \mathbb{R}^d, b \in \mathbb{R}\}$
- Loss function $\ell(f(x), y) = (f(x) - y)^2$ which is called the squared loss or ℓ_2-loss.

Squared loss is the most popular loss, though there are other losses such as the absolute loss $|f(x) - y|$. It is good to stop and think about the differences between these losses, and how they behave on outliers (points that deviate by large amounts).

Question: Suppose the function class is just a scalar ($d = 0$), find the minimizers of the squared loss and absolute loss for a data set $(y_1, \ldots, y_m)$.

2.1 Optimizing the loss

. Similar to the logistic regression setting, this loss is convex. In fact, we can find a closed form solution. The emprical risk/loss is:

$$\widehat{R}(w) = \frac{1}{m} \sum_{i=1}^{m} (y_i - w^\top x_i)^2.$$

In order to find the minimizer, we can compute the gradient and equate it to 0. After differentiating, we get

$$\begin{aligned} \nabla_w \hat{R}(w) &= \frac{2}{m} \sum_{i=1}^{m} (w^\top x_i - y_i) x_i = 0 \\ \implies \left(\sum_{i=1}^{m} x_i x_i^\top \right) w &= \sum_{i=1}^{m} y_i x_i \end{aligned}$$

For convenience, it would be useful to view this in matrix form. Let

$$X = \begin{bmatrix} - \; x_1^\top \; - \\ - \; x_2^\top \; - \\ \vdots \\ - \; x_m^\top \; - \end{bmatrix} \in \mathbb{R}^{m \times d}, Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^{m \times 1}.$$

[1]Logistic regression solves a classification problem but inherently learns a probability value. So it is similar to regression in some ways.

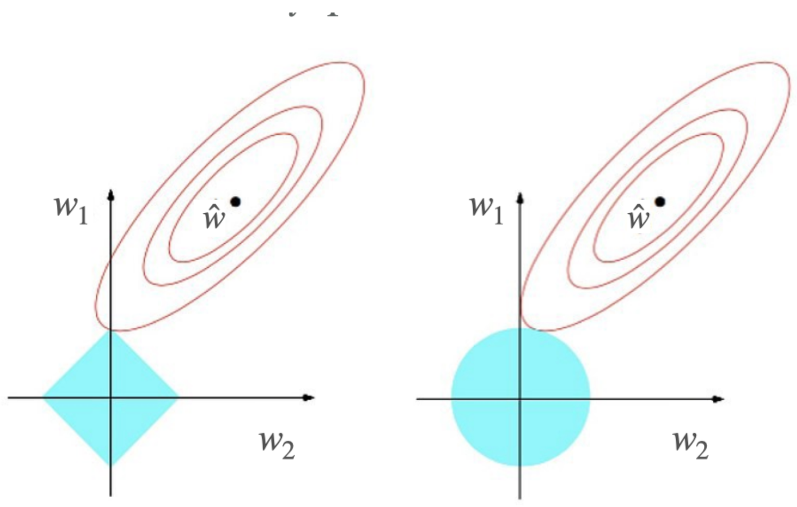

Figure 3: (*left*) Lasso and (*right* Ridge regression. Here the concentric ellipses are contour lines of the empirical risk (every point on each ellipse has the same empirical risk). The blue shapes denote the ℓ_1 and ℓ_2 balls respectively of some fixed radius. The point where the blue shapes touch the contour lines is the solution under constrained norms. Note that for lasso, the solution lies at one of the vertices of the ℓ_1-ball end points of the hypercube, which is sparse.

Then,

$$\left(\sum_{i=1}^{m} x_i x_i^\top\right) w = \sum_{i=1}^{m} y_i x_i \iff X^\top X w = X^\top y.$$

These are called the normal equations for least squares regression. Now if $X^\top X$ is full rank (invertible) then we have,

$$\hat{w} = (X^\top X)^{-1} X^\top Y.$$

This gives us the prediction $\hat{Y} = X\hat{w} = X(X^\top X)^{-1}X^\top Y$ which is the projection of Y onto the subspace spanned by the columns of X.

2.2 Ridge Regression and Lasso

When $X^\top X$ is very close to being non-singular[2] then $\hat{w}$ can contain large values that may lead to over-fitting. To prevent such over-fitting, often a regularization term is added.

So we get the optimization problem of the following form:

$$\min_w \hat{R}(w) + \lambda\psi(w)$$

where $\psi : \mathbb{R}^d \to \mathbb{R}$ is the regularizer and $\lambda \geq 0$ controls the amount of regularization. In practice, λ is tuned based on the problem.

[2]A matrix is singular when at least one of the columns can be expressed as a linear combination of the other. "very close to singular" implies that there is a column that is very close to being a linear combination of the others. High condition number is also a measure of this.

One of the most common regularization functions is $\psi(w) = \|w\|_2^2$. This gives us,

$$\min_w \frac{1}{m} \sum_{i=1}^{m} (y_i - w^\top x_i)^2 + \lambda \psi(w).$$

This is known as *ridge regression*. Similar to the way we computed the least squares solution, we can optimize this to get

$$\hat{w}_\lambda = (X^\top X + \lambda m I)^{-1} X^\top Y.$$

Observe that $X^\top X + \lambda m I$ is always invertible as long as $\lambda > 0$ since the minimum eigenvalue of this matrix is λm.

Another commonly used regularizer is $\psi(w) = \|w\|_1$. We can provably show that for large enough λ, the solution found with this regularization is **sparse** (see Figure 3). This is known as the lasso and is widely used in high-dimensional settings where we desire sparse solutions (dependence on a few features).

There are probabilistic interpretations of these regularizers as well, in the same spirit as our MLE derivation of the loss function.