CS5339 Lecture Notes #3:
Logistic Regression

Jonathan Scarlett

March 30, 2021

Useful references:

- MIT lecture notes,[1] lecture 4
- Section 4.3.2 of Bishop's "Pattern Recognition and Machine Learning" book (and also Section 1.5 for more on discriminative vs. generative models, Section 4.1.2 for a brief discussion on forming multi-class classifiers from binary ones)
- Section 9.3 of of "Understanding Machine Learning" book (and Section 17.1 for multi-class methods)
- For those wanting to learn about gradient-based optimization beyond the bare basics that we introduce, see `http://ruder.io/optimizing-gradient-descent/` for a good summary

1 Data Modeling

- So far, we have considered the data set $\mathcal{D} = \{(\mathbf{x}_t, y_t)\}_{t=1}^n$ as simply being fixed and given, and sometimes assumed it to satisfy certain assumptions (e.g., exact or approximate linear separability). We used $\mathcal{D}$ to learn a $\boldsymbol{\theta}$ corresponding to a binary linear classifier $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x}) \in \{-1, +1\}$, but we did not say anything about where the data set came from.
- We will now turn to the idea of placing *probabilistic models* on the data.
- Data models are often broadly categorized into the following two types:
  - Discriminative models focus on learning a conditional distribution $P(y|\mathbf{x})$, indicating the probability of each y value given the input $\mathbf{x}$.
  - Generative models also seek to learn $P(\mathbf{x}|y)$ and/or $P(\mathbf{x})$, which is often followed by an application of Bayes' rule to deduce $P(y|\mathbf{x})$.
- In this course, we will focus on discriminative models. These can already provide notable benefits over non-probabilistic methods:

[1] `http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/lecture-notes/`

- Instead of merely predicting $y = 1$ or $y = -1$, we can also give a *confidence* to our prediction, which can be very important – e.g., in sports betting, receiving the information "I predict Team A has a 55% chance of beating Team B" is much more useful (if accurate) than just "I predict Team A will beat Team B". (Even more so for applications in medicine, law, etc.)
- The simple methods that we introduce may be a sub-component in a larger learning system, and outputting "soft" (probabilistic) information may be more useful (e.g., if combining multiple classifiers' decisions, one could place less weight on those that are less confident).

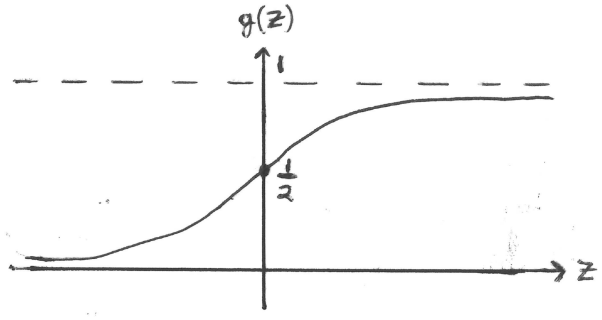
- Attaining an accurate generative model is a more demanding task, e.g., $P(\mathbf{x})$ or $P(\mathbf{x}|y)$ may be very complicated (and hence require lots of data to learn) even when $P(y|\mathbf{x})$ is simple. However, if it can be done, it can have further benefits:
  - It allows us to generate additional "synthetic" data (more data is always a good thing, at least when it is representative of the unseen inputs one ultimately wants to do prediction on)
  - When we are trying to classify new (unseen) $\mathbf{x}$, we can perform *outlier detection*, i.e., notice that this $\mathbf{x}$ is a "non-typical" one, and accordingly be wary of our predicted y.
  - Data generation can be of interest in its own right (e.g., imaging software, speech generation)
- The vast majority of theoretical studies assume (not necessarily realistically!) that different data samples $(\mathbf{x}_t, y_t)$ are statistically independent from each other (but with the same distribution), and we will do the same throughout the course. For instance, $P(y_1, y_2|\mathbf{x}_1, \mathbf{x}_2) = P(y_1|\mathbf{x}_1)P(y_2|\mathbf{x}_2)$.
- **A common phrase to keep in mind:** *All models are "wrong", but some models are useful*

2 The Logistic Model

- In this lecture, we will consider the *logistic likelihood* model:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-(\boldsymbol{\theta}^T\mathbf{x} + \theta_0))} \tag{1}$$

for some $\boldsymbol{\theta} \in \mathbb{R}^d$ and $\theta_0 \in \mathbb{R}$. When we want to make the dependence explicit, we will write $P(y = 1|\mathbf{x}; \boldsymbol{\theta}, \theta_0)$. We will also shorten notation by using $g(z) = \frac{1}{1+e^{-z}}$, which gives $P(y = 1|\mathbf{x}) = g(\boldsymbol{\theta}^T\mathbf{x}+\theta_0)$.



- Note: The distribution $P(\mathbf{x})$ will not play any significant role in this lecture, and in fact, the $\{\mathbf{x}_t\}_{t=1}^n$ can be viewed as remaining non-probabilistic.

- **Simple example.**
  - Suppose that y_t indicates whether user t will like a product.
  - Possible inputs: $x_1 = \mathbb{1}\{\text{user is male}\}$, $x_2 = (\text{age})$, $x_3 = $ (#similar products bought).
  - From (1), the higher $\boldsymbol{\theta}^T\mathbf{x} + \theta_0$ is, the higher the probability that the user likes the product.
  - Therefore, e.g., θ_3 should be positive, θ_2 should be negative if the product is a toy, etc.
- We have $P(y = -1|\mathbf{x}) = 1 - P(y = 1|\mathbf{x}) = \frac{\exp(-(\boldsymbol{\theta}^T\mathbf{x}+\theta_0))}{1+\exp(-(\boldsymbol{\theta}^T\mathbf{x}+\theta_0))}$, and therefore

$$\log \frac{P(y = 1|\mathbf{x})}{P(y = -1|\mathbf{x})} = \boldsymbol{\theta}^T\mathbf{x} + \theta_0,$$

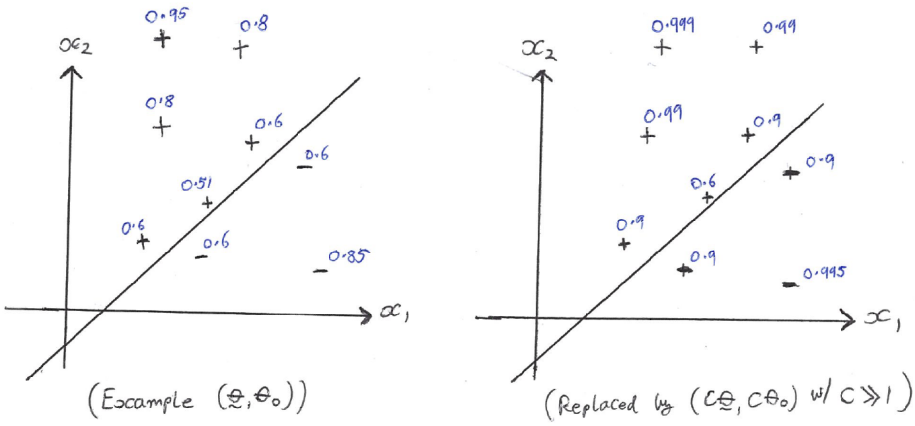
  which tells us that

$$P(y = 1|\mathbf{x}) > P(y = -1|\mathbf{x}) \iff \boldsymbol{\theta}^T\mathbf{x} + \theta_0 > 0.$$

  In other words, the classifier $f_{\boldsymbol{\theta}}(\mathbf{x}) = \text{sign}(\mathbf{x}^T\boldsymbol{\theta} + \theta_0)$ can be interpreted as choosing the label that is more likely under the logistic model.
- By further simplifying $\frac{\exp(-(\boldsymbol{\theta}^T\mathbf{x}+\theta_0))}{1+\exp(-(\boldsymbol{\theta}^T\mathbf{x}+\theta_0))} = g(-(\boldsymbol{\theta}^T\mathbf{x} + \theta_0))$, we find a unified way to write down the likelihood function for $y = 1$ and $y = -1$:

$$P(y|\mathbf{x}) = g(y(\boldsymbol{\theta}^T\mathbf{x} + \theta_0)).$$

  This is verified by just checking the cases $y = +1$ and $y = -1$ separately.
- For linear classifiers, the decision boundaries of $(\boldsymbol{\theta}, \theta_0)$ remains unchanged when the pair is scaled by a positive constant $c > 0$. However, despite the same decision boundary, such scaling can still affect the *likelihoods* assigned to points in the logistic model:



  Notice that scaling by $c > 1$ pushes the predictions closer to 0 (if it was originally $< \frac{1}{2}$) or 1 (if it was originally $> \frac{1}{2}$).
- Next, we discuss basic methods for learning "good" choices of $(\boldsymbol{\theta}, \theta_0)$ from data.

3 Maximum Likelihood Estimation

Formulation.

- The overall likelihood (i.e., the conditional probability of $(y_1, \ldots, y_n)$ given $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ as a function of $(\boldsymbol{\theta}, \theta_0)$) is

$$L(\boldsymbol{\theta}, \theta_0 | \mathcal{D}) = \prod_{t=1}^{n} P(y_t | \mathbf{x}_t; \boldsymbol{\theta}, \theta_0),$$

  where the product $\prod_{t=1}^{n}$ is due to the assumption of independent data samples.

- Given the data set $\mathcal{D}$ and knowledge that it comes from a logistic model, it is natural to construct a linear classifier by selecting $(\boldsymbol{\theta}, \theta_0)$ to maximize the likelihood:

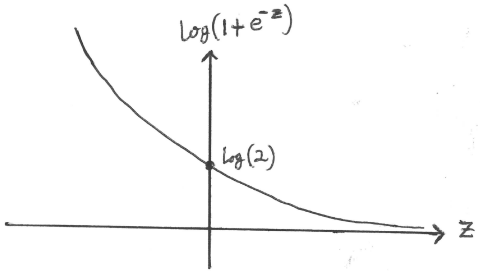
$$(\hat{\boldsymbol{\theta}}, \hat{\theta}_0) = \arg\max_{\boldsymbol{\theta}, \theta_0} L(\boldsymbol{\theta}, \theta_0 | \mathcal{D}).$$

  - Intuitively, choose $(\boldsymbol{\theta}, \theta_0)$ that "best explains" the data.
  - For now, we take for granted that this is a reasonable thing to do; there is theory showing it to succeed in a certain sense when the number of data points n is large enough. We will later discuss limitations when n is small.

- Maximizing L is equivalent to maximizing $\log L$, so we get

$$\begin{aligned}
(\hat{\boldsymbol{\theta}}, \hat{\theta}_0) &= \arg\max_{\boldsymbol{\theta}, \theta_0} \sum_{t=1}^{n} \log P(y_t | \mathbf{x}_t; \boldsymbol{\theta}, \theta_0) \\
&= \arg\min_{\boldsymbol{\theta}, \theta_0} \sum_{t=1}^{n} -\log P(y_t | \mathbf{x}_t; \boldsymbol{\theta}, \theta_0) \\
&= \arg\min_{\boldsymbol{\theta}, \theta_0} \sum_{t=1}^{n} -\log g(y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0)) \\
&= \arg\min_{\boldsymbol{\theta}, \theta_0} \sum_{t=1}^{n} \log\big(1 + \exp(-y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0))\big). \qquad (2)
\end{aligned}$$

- The function $z \to \log(1 + e^{-z})$ is often called the *logistic loss*:



Optimization.

- Unfortunately, we cannot solve (2) in closed form. Instead, we can resort to numerical optimization.

- We will cover simple (but extremely useful and widespread) gradient-based optimization methods in Lecture 6b. For now, it suffices to say that there exist efficient methods for finding the minimizer to a high degree of accuracy.

Making a prediction.

- As suggested above, once we have chosen $(\boldsymbol{\theta}, \theta_0)$, upon observing a new input $\mathbf{x}'$ we can predict that $y' = 1$ *with probability* $g(\boldsymbol{\theta}^T \mathbf{x} + \theta_0)$.
  - Hence, we are giving a confidence level in the prediction.
  - We report complete confidence when $\boldsymbol{\theta}^T \mathbf{x} + \theta_0 \to \pm\infty$, but only 50% confidence when $\boldsymbol{\theta}^T \mathbf{x} + \theta_0 = 0$.
  - Word of caution: These "confidence levels" may be highly misleading if the logistic modeling assumption was incorrect/inaccurate.

4 Regularization

- If the data set $\mathcal{D}$ is linearly separable, then there exists $\boldsymbol{\theta}, \theta_0$ such that $y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0) > 0$ for all t.
  - By scaling $\boldsymbol{\theta}, \theta_0$ up by a common constant factor, we can make $y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0)$ arbitrarily large, leading to a lower value of $\sum_{t=1}^n \log\left(1 + \exp(-y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0))\right)$ in (2).
  - Therefore, the optimal parameters are unbounded, and it is easy to show that this corresponds to always reporting *100% confidence* in the prediction. In this case, the likelihood is 1 (meaning the logistic loss is zero) on the training data.
  - However, maybe the data set was only linearly separable because we didn't collect enough data points! (i.e., n is too small) If we were to then collect more data and make a wrong prediction with 100% confidence, this would correspond to a likelihood of zero, or a logistic loss of ∞ – the worst possible.
  - More generally, even if the confidence is not always 100%, similar "over-confidence" problems can occur when there are too few data points.
- To avoid this type of behavior, one can penalize large parameters in the optimization problem. To do this, we regularize just like in the SVM objective function:

$$\text{minimize}_{\boldsymbol{\theta}, \theta_0} \quad \sum_{t=1}^n \log\left(1 + \exp(-y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0))\right) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2$$

for some $\lambda > 0$, or equivalently,

$$\text{minimize}_{\boldsymbol{\theta}, \theta_0} \quad \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^n \log\left(1 + \exp(-y_t(\boldsymbol{\theta}^T \mathbf{x}_t + \theta_0))\right)$$

for some $C > 0$.

  - We will study regularization in more detail (but in a different context) next lecture, and see that it acts as a *stabilizer* (i.e., avoiding wildly different solutions due to small changes in the data) and *mitigates overfitting* (i.e., learning spurious patterns that only occur due to fluctuations/noise)

- In fact, this simply corresponds to taking (one form of) the SVM optimization problem and replacing the *hinge loss* by the *logistic loss*.
- Once again, C is a parameter that needs to be tuned (e.g., via cross-validation covered later)

5 Multi-Class Classification

- We have looked at binary classification: $y_t \in \{-1, 1\}$ (e.g., spam or not spam). What if we have more than two classes (e.g., action, comedy, drama, etc.)? Generically, let's call these class labels $\{1, \ldots, M\}$.
- One possibility is to try to solve the multi-class problem using binary methods.
- <u>One vs. rest.</u>
  - For each $c \in \{1, \ldots, M\}$, apply binary classification with labels $y_t = 1$ if the t-th sample has class c, and $y_t = -1$ otherwise. Hence, the label simply says "Is this in class c?"
  - Let $\boldsymbol{\theta}^{(c)}$, $\theta_0^{(c)}$ be the c-th learned classifier parameters.
  - To predict a new sample, plug the input $\mathbf{x}$ into all of the M classifiers. Let the estimate $\hat{c}$ be the class with the highest value of $(\boldsymbol{\theta}^{(c)})^T\mathbf{x} + \theta_0^{(c)}$.
- <u>One vs. one.</u>
  - Take all $\binom{M}{2}$ pairs c, c' from $\{1, \ldots, M\}$, and train a binary classifier for each pair to get $\boldsymbol{\theta}^{(c,c')}$, $\theta_0^{(c,c')}$. That is, the (c, c')-th classifier tries to distinguish the class c (corresponding to $y = 1$) from the class c' (corresponding to $y = -1$).
  - When training for c, c', all samples with labels differing from these two values are omitted.
  - To predict a new sample, plug the input $\mathbf{x}$ into all of the $\binom{M}{2}$ classifiers, and let $\hat{c}$ be the one that was preferred over its competitor the highest number of times.
- Both of these approaches are heuristic, perform well in some cases but not others, and have known potentially major issues.
  - As an example, try applying the one vs. rest rule to 2D data with 3 classes, with the data points being spread evenly among 3 circles (one per class) of unit radius centered at $(-2, 0)$, $(0, 0)$, and $(2, 0)$. Assume that the fraction of points in each class is 0.4, 0.2, and 0.4, respectively.
  - See Example 17.1 of "Understanding Machine Learning" for the solution.
- *What about a more direct approach?*
  - Different classification algorithms have different difficulties in deriving multi-class counterparts (e.g., impossible/difficult/do-able/easy/trivial).
  - Examples: SVM (see Section 17.2.5 of "Understanding Machine Learning"), boosting (to be covered later; multi-class version will be an advanced tutorial question).
  - Logistic regression has a very natural multi-class counterpart: Replace (1) by the *soft-max function*

$$P(y = c|\mathbf{x}) = \frac{\exp(\boldsymbol{\theta}_c^T\mathbf{x} + \theta_{0,c})}{\sum_{c'=1}^{M} \exp(\boldsymbol{\theta}_{c'}^T\mathbf{x} + \theta_{0,c'})}, \quad c = 1, \ldots, M, \tag{3}$$

where we now have a different pair $(\boldsymbol{\theta}_c, \theta_{0,c})$ for each class. Without loss of generality we can assume that one of the classes has $(\boldsymbol{\theta}_c, \theta_{0,c}) = (\mathbf{0}, 0)$ (why?), which is useful for showing that (1) is a special case of (3).