Generalized linear models and logistic regression
Comments (Oct. 15, 2019)

- Assignment 1 almost marked
  - Any questions about the problems on this assignment?

- Assignment 2 due soon
  - Any questions?
Summary so far

• From chapters 1 and 2, obtained tools needed to talk about uncertainty/noise underlying machine learning
  • capture uncertainty about data/observations using probabilities
  • formalize estimation problem for distributions

• Identify variables $x_1, \ldots, x_d$
  • e.g. observed features, observed targets

• Pick the desired distribution
  • e.g. $p(x_1, \ldots, x_d)$ or $p(x_1 \mid x_2, \ldots, x_d)$ (conditional distribution)
  • e.g. $p(x_i)$ is Poisson or $p(y \mid x_1, \ldots, x_d)$ is Gaussian

• Perform parameter estimation for chosen distribution
  • e.g., estimate lambda for Poisson
  • e.g. estimate mu and sigma for Gaussian
Summary so far (2)

• For prediction problems, which is much of machine learning, first discuss
  • the types of data we get (i.e., features and types of targets)
  • goal to minimize expected cost of incorrect predictions

• Concluded optimal prediction functions use \( p(y \mid x) \) or \( E[Y \mid x] \)

• From there, our goal becomes to estimate \( p(y \mid x) \) or \( E[Y \mid x] \)

• Starting from this general problem specification, it is useful to use our parameter estimation techniques to solve this problem
  • e.g., specify \( Y = Xw + \text{noise} \), estimate \( \mu = xw \)
Summary so far (3)

- For linear regression setting, modeling $p(y | x)$ as a Gaussian with $\mu = <x, w>$ and a constant sigma.

- Performed maximum likelihood to get weights $w$.

- Possible question: why all this machinery to get to linear regression?
  - one answer: makes our assumptions about uncertainty more clear.
  - another answer: it will make it easier to generalize $p(y | x)$ to other distributions (which we will do with GLMs).
Estimation approaches for Linear regression

• Recall we estimated $w$ for $p(y \mid x)$ as a Gaussian

• We discussed the closed form solution

$$w = (X^\top X)^{-1}X^\top y$$

• and using batch or stochastic gradient descent

$$w_{t+1} = w_t - \eta X^\top (Xw_t - y)$$

$$w_{t+1} = w_t - \eta_t x_t^\top (x_t w_t - y_t)$$

• Exercise: Now imagine you have 10 new data points. How do we get a new $w$, that incorporates these data points?
Exercise: MAP for Poisson

- Recall we estimated lambda for Poisson p(x)
  - Had a dataset of scalars \{x_1, \ldots, x_n\}
  - For MLE, found the closed form solution \( \lambda = \text{average of } x_i \)

- Can we use gradient descent for this optimization? And if so, should we?
Exercise: Predicting the number of accidents

- In Assignment 1, learned $p(y)$ as Poisson, where $Y$ is the number of accidents in a factory

- How would the question from assignment 1 change if we also wanted to condition on features?
  - For example, want to model the number of accidents in the factory, given $x_1 =$ size of the factory and $x_2 =$ number of employees

- What is $p(y \mid x)$? What are the parameters?
Poisson regression

\[ p(y|x) = \text{Poisson}(y|\lambda = \exp(x^\top w)) \]

1. \( E[Y|x] = \exp(w^\top x) \)
2. \( p(y|x) = \text{Poisson}(\lambda) \)
Exponential Family Distributions

\[ p(y|\theta) = \exp(\theta y - a(\theta) + b(y)) \]

Useful property: \[ \frac{da(\theta)}{d\theta} = \mathbb{E}[Y] \]

Transfer \( f \) corresponds to the derivate of the log-normalizer function \( a \)

We will always linearly predict the natural parameter \( \theta = x^\top w \)
Examples

\[ \theta = x^\top w \]

\[ p(y|\theta) = \exp(\theta y - a(\theta) + b(y)) \]

- **Gaussian distribution**
  \[ a(\theta) = \frac{1}{2} \theta^2 \quad f(\theta) = \theta \]

- **Poisson distribution**
  \[ a(\theta) = \exp(\theta) \quad f(\theta) = \exp(\theta) \]

- **Bernoulli distribution**
  \[ a(\theta) = \ln(1 + \exp(\theta)) \quad f(\theta) = \frac{1}{1 + \exp(-\theta)} \]

sigmoid
Exercise: How do we extract the form for the Poisson distribution?

\[ p(y|\theta) = \exp(\theta y - a(\theta) + b(y)) \]

Example 17: The Poisson distribution can be expressed as

\[ p(x|\lambda) = \exp(x \log \lambda - \lambda - \log x!) , \]

where \( \lambda \in \mathbb{R}^+ \) and \( \mathcal{X} = \mathbb{N}_0 \). Thus, \( \theta = \log \lambda \), \( a(\theta) = e^\theta \), and \( b(x) = -\log x! \).

- What is the transfer \( f \)?

\[ f(\theta) = \frac{da(\theta)}{d\theta} = \exp(\theta) \]
Exercise: How do we extract the form for the exponential distribution?

\[ \lambda > 0 \quad \lambda \exp(-\lambda y) \]

- Recall exponential family distribution

\[ p(y|\theta) = \exp(\theta y - a(\theta) + b(y)) \]

i.e., \[ p(y|\theta) = \exp(\theta y) \exp(-a(\theta)) \exp(b(y)) \]

- How do we write the exponential distribution this way?

\[ \theta = x^\top w \quad a(\theta) = -\ln(-\theta) \quad b(y) = 0 \]

- What is the transfer f?

\[ f(\theta) = \frac{d}{d\theta} a(\theta) = \frac{-1}{\theta} \]
Logistic regression

1. \( E[y|x] = \sigma(\omega^T x) \)

2. \( p(y|x) = \text{Bernoulli}(\alpha) \) with \( \alpha = E[y|x] \).

The Bernoulli distribution, with \( \alpha \) a function of \( x \), is

\[
p(y|x) = \begin{cases} 
\left( \frac{1}{1 + e^{-\omega^T x}} \right)^y & \text{for } y = 1 \\
\left( 1 - \frac{1}{1 + e^{-\omega^T x}} \right)^{1-y} & \text{for } y = 0 
\end{cases} 
\]

\[
= \sigma(x^T w)^y (1 - \sigma(x^T w))^{1-y}
\]

\[
E[y|x] = \frac{1}{1 + e^{-\omega^T x}}
\]

\[
p(y|x) = \left( \frac{1}{1 + e^{-\omega^T x}} \right)^y \left( 1 - \frac{1}{1 + e^{-\omega^T x}} \right)^{1-y}.
\]
What is $c(w)$ for GLMs?

- Still formulating an optimization problem to predict targets $y$ given features $x$

- The variables we learn is the weight vector $w$

- What is $c(w)$?

  \[ MLE : c(w) \propto - \ln p(D|w) \]

  \[ \propto - \sum_{i=1}^{n} \ln p(y_i|x_iw) \]

- \[
  \arg \min_w c(w) = \arg \max_w p(D|w)
\]
Cross-entropy loss for Logistic Regression

\[ c_i(w) = y_i \ln \sigma(w^T x_i) + (1 - y_i) \ln(1 - \sigma(w^T x_i)) \]
Extra exercises

- Go through the derivation of $c(w)$ for logistic regression
- Derive Maximum Likelihood objective in Section 8.1.2
Benefits of GLMs

- Gave a generic update rule, where you only needed to know the transfer for your chosen distribution
  - e.g., linear regression with transfer $f = \text{identity}$
  - e.g., Poisson regression with transfer $f = \exp$
  - e.g., logistic regression with transfer $f = \text{sigmoid}$

- We know the objective is convex in $w$!
Convexity

- Convexity of negative log likelihood of (many) exponential families
  - The negative log likelihood of many exponential families is convex, which is an important advantage of the maximum likelihood approach

- Why is convexity important?
  - e.g., \((\text{sigmoid}(xw) - y)^2\) is nonconvex, but who cares?
Cross-entropy loss versus Euclidean loss for classification

\[ c_i(w) = y_i \ln \sigma(w^\top x_i) + (1 - y_i) \ln(1 - \sigma(w^\top x_i)) \]

- Why not just use

\[
\min_{w \in \mathbb{R}^d} \sum_{i=1}^{n} \left( \sigma(x_i^\top w) - y_i \right)^2
\]

- The notes explain that this is a non-convex objective
  - from personal experience, it seems to do more poorly in practice

- If no obvious reason to prefer one or the other, we may as well pick the objective that is convex (no local minima)
How can we check convexity?

- Can check the definition of convexity

\[ f(tx_1 + (1 - t)x_2) \leq tf(x_1) + (1 - t)f(x_2) \]

- Can check second derivative for scalar parameters (e.g. \( \lambda \)) and Hessian for multidimensional parameters (e.g., \( \mathbf{w} \))
  - e.g., for linear regression (least-squares), the Hessian is \( \mathbf{H} = \mathbf{X}^\top \mathbf{X} \) and so positive semi-definite
  - e.g., for Poisson regression, the Hessian of the negative log-likelihood is \( \mathbf{H} = \mathbf{X}^\top \mathbf{C} \mathbf{X} \) and so positive semi-definite
Prediction with logistic regression

- So far, we have used the prediction $f(xw)$
  - e.g., $xw$ for linear regression, $\exp(xw)$ for Poisson regression

- For binary classification, want to output 0 or 1, rather than the probability value $p(y = 1 \mid x) = \text{sigmoid}(xw)$

- Sigmoid has few values $xw$ mapped close to 0.5; most values somewhat larger than 0 are mapped close to 0 (and vice versa for 1)

- Decision threshold:
  - $\text{sigmoid}(xw) < 0.5$ is class 0
  - $\text{sigmoid}(xw) > 0.5$ is class 1

$$f(t) = \frac{1}{1 + e^{-t}}$$
Logistic regression is a linear classifier

- Hyperplane $\mathbf{w}^\top \mathbf{x} = 0$ separates the two classes
  - $P(y=1 \mid \mathbf{x}, \mathbf{w}) > 0.5$ only when $\mathbf{w}^\top \mathbf{x} \geq 0$.
  - $P(y=0 \mid \mathbf{x}, \mathbf{w}) > 0.5$ only when $P(y=1 \mid \mathbf{x}, \mathbf{w}) < 0.5$, which happens when $\mathbf{w}^\top \mathbf{x} < 0$

$$\mathbf{w} = [2.75 \ -1/3 \ -1]$$

$$\mathbf{x} = [1 \ 4 \ 3]$$

$$\mathbf{x}^\top \mathbf{w} = -1.8$$

$$\mathbf{x} = [1 \ 2 \ 1]$$

$$\mathbf{x}^\top \mathbf{w} = 0.27$$

For a previously unseen data point $\mathbf{x}$, we can estimate its class label $y$ based on the observed data. Based on the principles of maximum conditional likelihood estimation, we simply calculate the posterior probability as

$$P(y=1 \mid \mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^\top \mathbf{x})$$

where $\sigma$ is the sigmoid function, i.e., $\sigma(z) = \frac{1}{1 + e^{-z}}$. Thus, the predictor maps a zero or one. Note that

$$P(y=0 \mid \mathbf{x}, \mathbf{w}) > 0.5$$

only when $P(y=1 \mid \mathbf{x}, \mathbf{w}) < 0.5$, which happens when $\mathbf{w}^\top \mathbf{x} < 0$.

To frame the learning problem as parameter estimation, we will assume that

$$P(y=1 \mid \mathbf{x}, \mathbf{w}) > 0.5$$

only when

$$P(y=0 \mid \mathbf{x}, \mathbf{w}) > 0.5$$

Thus, the logistic regression model is a linear classifier.

Figure 6.1: A data set in a 2-dimensional vector space with a linear decision surface. The decision surface does not perfectly separate positives from negatives.

Figure 6.2: Sigmoid function in the interval $(-\infty, \infty)$.
Logistic regression versus Linear regression

• Why might one be better than the other? They both use a linear approach

• Linear regression could still learn $<x, w>$ to predict $E[Y | x]$

• Demo: logistic regression performs better under outliers, when the outlier is still on the correct side of the line

• Conclusion:
  • logistic regression better reflects the goals of predicting $p(y=1 | x)$, to finding separating hyperplane
  • Linear regression assumes $E[Y | x]$ a linear function of $x$!
Adding regularizers to GLMs

• How do we add regularization to logistic regression?

• We had an optimization for logistic regression to get $w$: minimize negative log-likelihood, i.e. minimize cross-entropy

• Now want to balance negative log-likelihood and regularizer (i.e., the prior for MAP)

• Simply add regularizer to the objective function
Adding a regularizer to logistic regression

- Original objective function for logistic regression
  \[
  \arg \max_{\mathbf{w}} \sum_{i=1}^{n} \left( (y_i - 1) \mathbf{w}^\top \mathbf{x}_i + \log \left( \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}} \right) \right)
  \]
  \[
  \arg \min_{\mathbf{w}} \sum_{i=1}^{n} \left( (y_i - 1) \mathbf{w}^\top \mathbf{x}_i + \log \left( \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}} \right) \right)
  \]

- Adding regularizer
  \[
  \arg \min_{\mathbf{w}} \sum_{i=1}^{n} \left( (y_i - 1) \mathbf{w}^\top \mathbf{x}_i + \log \left( \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}} \right) \right) + \lambda \| \mathbf{w} \|_2^2
  \]
Other regularizers

- Have discussed l2 and l1 regularizers

- Other examples:
  - elastic net regularization is a combination of l1 and l2 (i.e., l1 + l2): ensures a unique solution
  - capped regularizers: do not prevent large weights

Does this regularizer still protect against overfitting?

(a) Capped $\ell_1$-norm loss ($\varepsilon = 2.5$)  
(b) $\ell_1$-norm loss

* Figure from “Robust Dictionary Learning with Capped l1-Norm”, Jiang et al., IJCAI 2015
Practical considerations: outliers

• What happens if one sample is bad?

• Regularization helps a little bit

• Can also change losses

• Robust losses
  • use l1 instead of l2
  • even better: use capped l1

• What are the disadvantages to these losses?
Exercise: intercept unit

- In linear regression, we added an intercept unit (bias unit) to the features
  - i.e., added a feature that is always 1 to the feature vector

- Does it make sense to do this for GLMs?
  - e.g., sigmoid(<x,w> + w_0)
Adding a column of ones to GLMs

- This provides the same outcome as for linear regression

- $g(\mathbb{E}[y \mid x]) = x \mathbf{w} \rightarrow$ bias unit in $x$ with coefficient $w_0$ shifts the function left or right

*Figure from http://stackoverflow.com/questions/2480650/role-of-bias-in-neural-networks*