COMS 4721: Machine Learning for Data Science

Lecture 9, 2/19/2019

Prof. John Paisley

Department of Electrical Engineering
& Data Science Institute

Columbia University

LOGISTIC REGRESSION

Linear classifiers

Given: Data $(x_1, y_1), \ldots, (x_n, y_n)$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$

A **linear classifier** takes a vector $w \in \mathbb{R}^d$ and scalar $w_0 \in \mathbb{R}$ and predicts

$$y_i = f(x_i; w, w_0) = \text{sign}(x_i^T w + w_0).$$

We discussed two methods last time:

- Least squares: Sensitive to outliers
- Perceptron: Convergence issues, assumes linear separability

Can we combine the separating hyperplane idea with probability to fix this?

Linear discriminant analysis

We saw an example of a linear classification rule using a Bayes classifier.

For the model $y \sim \text{Bern}(\pi)$ and $x \,|\, y \sim N(\mu_y, \Sigma)$, declare $y = 1$ given x if

$$\ln \frac{p(x|y=1)p(y=1)}{p(x|y=0)p(y=0)} > 0\,.$$

In this case, the *log odds* is equal to

$$\ln \frac{p(x|y=1)p(y=1)}{p(x|y=0)p(y=0)} = \underbrace{\ln \frac{\pi_1}{\pi_0} - \frac{1}{2}(\mu_1 + \mu_0)^T \Sigma^{-1}(\mu_1 - \mu_0)}_{\text{a constant } w_0} + x^T \underbrace{\Sigma^{-1}(\mu_1 - \mu_0)}_{\text{a vector } w}$$

Original formulation

Recall that originally we wanted to declare $y = 1$ given x if

$$\ln \frac{p(y = 1|x)}{p(y = 0|x)} > 0$$

We didn't have a way to define $p(y|x)$, so we used Bayes rule:

- Use $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$ and let the $p(x)$ cancel each other in the fraction
- Define $p(y)$ to be a Bernoulli distribution (coin flip distribution)
- Define $p(x|y)$ however we want (e.g., a single Gaussian)

Now, we want to directly define $p(y|x)$. We'll use the log odds to do this.

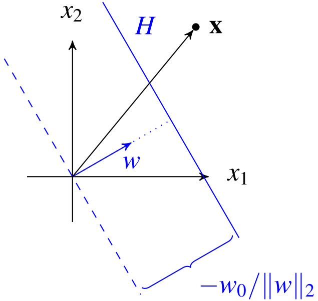
Log odds and hyperplanes

Classifying x based on the log odds

$$L = \ln \frac{p(y=+1|x)}{p(y=-1|x)},$$

we notice that

1. $L \gg 0$: more confident $y = +1$,
2. $L \ll 0$: more confident $y = -1$,
3. $L = 0$: can go either way



The linear function $x^T w + w_0$ captures these three objectives:

- The distance of x to a hyperplane H defined by (w, w_0) is $\left|\frac{x^T w}{\|w\|_2} + \frac{w_0}{\|w\|_2}\right|$.
- The sign of the function captures which side x is on.
- As x moves away/towards H, we become more/less confident.

LOG ODDS AND HYPERPLANES

Logistic link function

We can directly plug in the hyperplane representation for the log odds:

$$\ln \frac{p(y=+1|x)}{p(y=-1|x)} = x^T w + w_0$$

Question: What is different from the previous Bayes classifier?

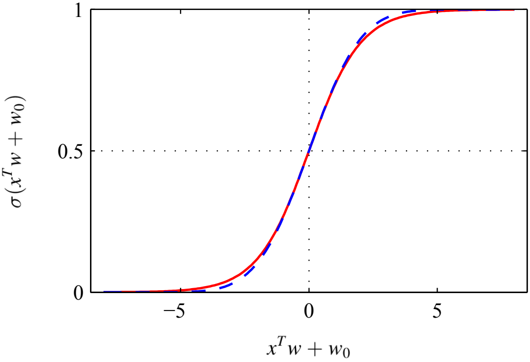
Answer: There was a formula for calculating w and w_0 based on the prior model and data x. Now, we put no restrictions on these values.

Setting $p(y=-1|x) = 1 - p(y=+1|x)$, solve for $p(y=+1|x)$ to find

$$p(y=+1|x) = \frac{\exp\{x^T w + w_0\}}{1+\exp\{x^T w + w_0\}} = \sigma(x^T w + w_0).$$

- This is called the *sigmoid function*.
- We have chosen $x^T w + w_0$ as the *link function* for the log odds.

Logistic sigmoid function



- Red line: Sigmoid function $\sigma(x^T w + w_0)$, which maps x to $p(y = +1|x)$.
- The function $\sigma(\cdot)$ captures our desire to be more confident as we move away from the separating hyperplane, defined by the $(x^T w + w_0)$-axis.
- (Blue dashed line: Not discussed.)

Logistic regression

As with regression, absorb the offset: $w \leftarrow \begin{bmatrix} w_0 \\ w \end{bmatrix}$ and $x \leftarrow \begin{bmatrix} 1 \\ x \end{bmatrix}$.

Definition

Let $(x_1, y_1), \ldots, (x_n, y_n)$ be a set of binary labeled data with $y \in \{-1, +1\}$. *Logistic regression* models each y_i as independently generated, with

$$P(y_i = +1 | x_i, w) = \sigma(x_i^T w), \quad \sigma(x_i; w) = \frac{e^{x_i^T w}}{1 + e^{x_i^T w}}.$$

Discriminative vs Generative classifiers

- This is a *discriminative* classifier because x is not directly modeled.
- Bayes classifiers are known as *generative* because x is modeled.

Discriminative: $p(y|x)$ Generative: $p(x|y)p(y)$.

Logistic regression likelihood

Data likelihood

Define $\sigma_i(w) = \sigma(x_i^T w)$. The joint likelihood of $y_1, \dots, y_n$ is

$$\begin{aligned} p(y_1, \dots, y_n | x_1, \dots, x_n, w) &= \prod_{i=1}^{n} p(y_i | x_i, w) \\ &= \prod_{i=1}^{n} \sigma_i(w)^{\mathbb{1}(y_i=+1)} \left(1 - \sigma_i(w)\right)^{\mathbb{1}(y_i=-1)} \end{aligned}$$

- Notice that each x_i modifies the probability of a '$+1$' for its respective y_i.
- Predicting new data is the same:
  - If $x^T w > 0$, then $\sigma(x^T w) > 1/2$ and predict $y = +1$, and vice versa.
  - We now get a confidence in our prediction via the probability $\sigma(x^T w)$.

More notation changes

Use the following fact to condense the notation:

$$\underbrace{\frac{e^{y_i x_i^T w}}{1 + e^{y_i x_i^T w}}}_{\sigma_i(y_i \cdot w)} = \Bigg(\underbrace{\frac{e^{x_i^T w}}{1 + e^{x_i^T w}}}_{\sigma_i(w)}\Bigg)^{\mathbb{1}(y_i = +1)} \Bigg(\underbrace{1 - \frac{e^{x_i^T w}}{1 + e^{x_i^T w}}}_{1 - \sigma_i(w)}\Bigg)^{\mathbb{1}(y_i = -1)}$$

therefore, the data likelihood can be written compactly as

$$p(y_1, \ldots, y_n | x_1, \ldots, x_n, w) = \prod_{i=1}^{n} \sigma_i(y_i \cdot w)$$

We want to maximize this over w.

Maximum likelihood

The maximum likelihood solution for w can be written

$$\begin{aligned} w_{\text{ML}} &= \arg\max_{w} \sum_{i=1}^{n} \ln \sigma_i(y_i \cdot w) \\ &= \arg\max_{w} \mathcal{L} \end{aligned}$$

As with the Perceptron, we can't directly set $\nabla_w \mathcal{L} = 0$, and so we need an iterative algorithm. Since we want to *maximize* $\mathcal{L}$, at step t we can update

$$w^{(t+1)} = w^{(t)} + \eta \nabla_w \mathcal{L}, \qquad \nabla_w \mathcal{L} = \sum_{i=1}^{n} (1 - \sigma_i(y_i \cdot w))\, y_i x_i.$$

We will see that this results in an algorithm similar to the Perceptron.

LOGISTIC REGRESSION ALGORITHM (STEEPEST ASCENT)

Input: Training data $(x_1, y_1), \ldots, (x_n, y_n)$ and step size $\eta > 0$

1. **Set** $w^{(1)} = \vec{0}$
2. **For iteration** $t = 1, 2, \ldots$ **do**
   - Update $w^{(t+1)} = w^{(t)} + \eta \sum_{i=1}^{n} \Big(1 - \sigma_i(y_i \cdot w^{(t)})\Big)\, y_i x_i$

Perceptron: Search for misclassified (x_i, y_i), update $w^{(t+1)} = w^{(t)} + \eta y_i x_i$.

Logistic regression: Something similar except we sum over all data.

- Recall that $\sigma_i(y_i \cdot w)$ is probability of observed y_i.
- Therefore $1 - \sigma_i(y_i \cdot w)$ is the probability assigned to the *wrong* value.
- Perceptron is "all-or-nothing." Either it's correctly or incorrectly classified.
- Logistic regression has a probabilistic "fudge-factor."

Bayesian logistic regression

Problem: If a hyperplane can separate all training data, then $\|w_{\text{ML}}\|_2 \to \infty$. This drives $\sigma_i(y_i \cdot w) \to 1$ for each (x_i, y_i).

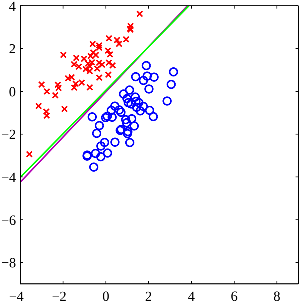
Even for nearly separable data it might get a few very wrong in order to be more confident about the rest. This is a case of "over-fitting."

A solution: Regularize w with $\lambda w^T w$:

$$w_{\text{MAP}} = \arg\max_w \sum_{i=1}^n \ln \sigma_i(y_i \cdot w) - \lambda w^T w$$

We've seen how this corresponds to a Gaussian prior distribution on w.

How about the posterior $p(w|x, y)$?



Laplace approximation

Posterior calculation

Define the prior distribution on w to be $w \sim N(0, \lambda^{-1} I)$. The posterior is

$$p(w|x,y) = \frac{p(w) \prod_{i=1}^{n} \sigma_i(y_i \cdot w)}{\int p(w) \prod_{i=1}^{n} \sigma_i(y_i \cdot w)\, dw}$$

This is not a "standard" distribution and we can't calculate the denominator.

Therefore we can't actually say what $p(w|x,y)$ is.

Can we approximate $p(w|x,y)$?

Laplace approximation

One strategy

Pick a distribution to approximate $p(w|x,y)$. We will say

$$p(w|x,y) \approx \text{Normal}(\mu, \Sigma).$$

Now we need a method for setting μ and Σ.

Laplace approximations

Using a condensed notation, notice from Bayes rule that

$$p(w|x,y) = \frac{\mathrm{e}^{\ln p(y,w|x)}}{\int \mathrm{e}^{\ln p(y,w|x)} dw}.$$

We will approximate $\ln p(y,w|x)$ in the numerator and denominator.

Let's define $f(w) = \ln p(y, w|x)$.

Taylor expansions

We can approximate $f(w)$ with a **second order Taylor expansion**.

Recall that $w \in \mathbb{R}^{d+1}$. For any point $z \in \mathbb{R}^{d+1}$,

$$f(w) \approx f(z) + (w-z)^T \nabla f(z) + \frac{1}{2}(w-z)^T \left(\nabla^2 f(z)\right)(w-z)$$

The notation $\nabla f(z)$ is short for $\nabla_w f(w)|_z$, and similarly for the matrix of second derivatives. We just need to pick z.

The Laplace approximation defines $z = w_{\text{MAP}}$.

LAPLACE APPROXIMATION (SOLVING)

Recall $f(w) = \ln p(y, w|x)$ and $z = w_{\text{MAP}}$. From Bayes rule and the Laplace approximation we now have

$$
\begin{aligned}
p(w|x, y) &= \frac{e^{f(w)}}{\int e^{f(w)} dw} \\
&\approx \frac{e^{f(z)+(w-z)^T \nabla f(z) + \frac{1}{2}(w-z)^T \left(\nabla^2 f(z)\right)(w-z)}}{\int e^{f(z)+(w-z)^T \nabla f(z) + \frac{1}{2}(w-z)^T (\nabla^2 f(z))(w-z)} dw}
\end{aligned}
$$

This can be simplified in two ways,

1. The term $e^{f(w_{\text{MAP}})}$ in the numerator and denominator can be viewed as a multiplicative constant since it doesn't vary in w. They therefore cancel.
2. By definition of how we find w_{MAP}, the vector $\nabla_w \ln p(y, w|x)|w_{\text{MAP}} = 0$.

Laplace approximation (solving)

We're therefore left with the approximation

$$p(w|x,y) \quad \approx \quad \frac{\mathrm{e}^{-\frac{1}{2}(w-w_{\text{MAP}})^T\left(-\nabla^2 \ln p(y,w_{\text{MAP}}|x)\right)(w-w_{\text{MAP}})}}{\int \mathrm{e}^{-\frac{1}{2}(w-w_{\text{MAP}})^T(-\nabla^2 \ln p(y,w_{\text{MAP}}|x))(w-w_{\text{MAP}})}dw}$$

The solution comes by observing that this is a multivariate normal,

$$p(w|x,y) \approx \text{Normal}(\mu, \Sigma),$$

where

$$\mu = w_{\text{MAP}}, \quad \Sigma = \left(-\nabla^2 \ln p(y, w_{\text{MAP}}|x)\right)^{-1}$$

We can take the second derivative (Hessian) of the log joint likelihood to find

$$\nabla^2 \ln p(y, w_{\text{MAP}}|x) = -\lambda I - \sum_{i=1}^{n} \sigma_i(y_i \cdot w_{\text{MAP}})\left(1 - \sigma_i(y_i \cdot w_{\text{MAP}})\right) x_i x_i^T$$

Laplace approximation for logistic regression

Given labeled data $(x_1, y_1), \ldots, (x_n, y_n)$ and the model

$$p(y_i|x_i, w) = \sigma_i(y_i \cdot w), \quad w \sim N(0, \lambda^{-1}I), \qquad \sigma_i(y_i \cdot w) = \frac{e^{y_i x_i^T w}}{1 + e^{y_i x_i^T w}}$$

1. Find: $w_{\text{MAP}} \; = \; \arg\max_w \; \sum_{i=1}^{n} \ln \sigma_i(y_i \cdot w) - \frac{\lambda}{2} w^T w$
2. Set: $-\Sigma^{-1} = -\lambda I - \sum_{i=1}^{n} \sigma_i(y_i \cdot w_{\text{MAP}}) \, (1 - \sigma_i(y_i \cdot w_{\text{MAP}})) \, x_i x_i^T$
3. Approximate: $p(w|x, y) \approx N\left(w_{\text{MAP}}, \Sigma\right)$.