Logistic Regression

Aarti Singh

Machine Learning 10-315
Feb 2, 2022

Announcements

- Recitation Friday Feb 4
  - MLE, MAP examples, Convexity, Optimization

"Discriminative" Classifiers

"Generative" classifier

P(Y) P(X|Y)

P(Y|X)

Bayes Classifier:

$$f^*(x) = \arg\max_{Y=y} P(Y = y|X = x)$$

$$= \arg\max_{Y=y} P(X = x|Y = y)P(Y = y)$$

Why not learn P(Y|X) directly? Or better yet, why not learn the decision boundary directly?

- Assume some functional form for P(Y|X) or for the decision boundary
- Estimate parameters of functional form directly from training data

Today we will see one such classifier – **Logistic Regression**

Logistic Regression

Not really regression

binary Y

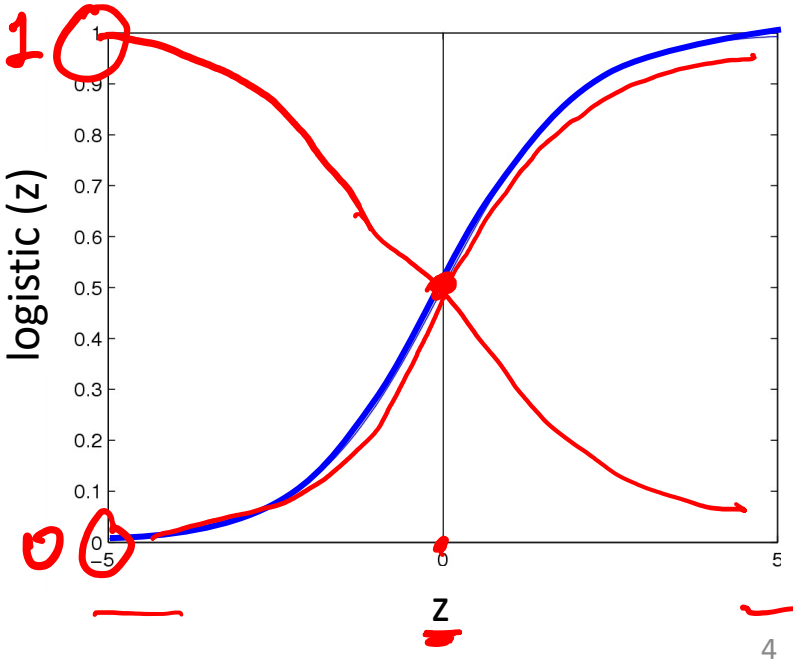
Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(-w_0 - \sum_i w_i X_i)}$$

$X = \begin{bmatrix} X_1 \\ \vdots \\ X_d \end{bmatrix}$

Logistic function applied to a linear function of the data

Logistic function (or Sigmoid), σ(z) = $\frac{1}{1+exp(-z)}$



Features can be discrete or continuous!

Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(-w_0 - \sum_i w_i X_i)} = \frac{1}{1+e^{-z}}$$

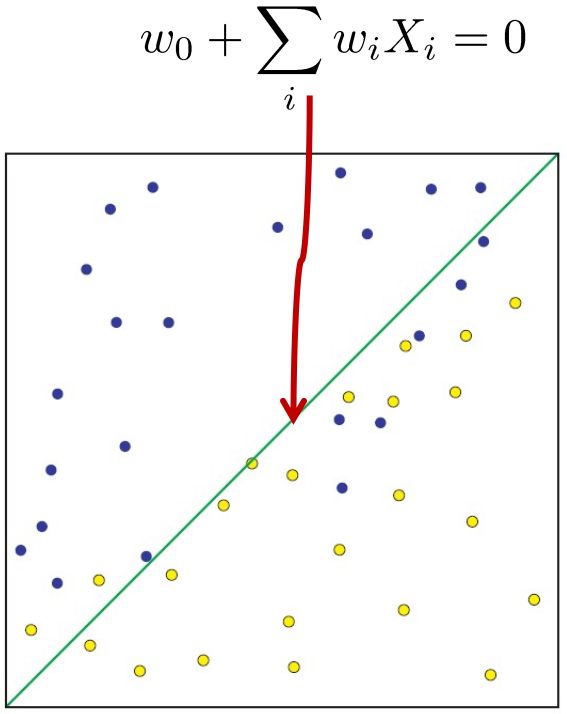
$$\Rightarrow P(Y=0|X) = 1 - \frac{1}{1+e^{-z}} = \frac{e^{-z}}{1+e^{-z}}$$

Decision boundary:

$$\Rightarrow \frac{P(Y=1|X)}{P(Y=0|X)} = \exp(w_0 + \sum_i w_i X_i) \gtrless \begin{matrix}1\\ \\ 0\end{matrix} \quad \frac{1}{e^{z}} \le 1$$

$$\Rightarrow \boxed{w_0 + \sum_i w_i X_i \gtrless \begin{matrix}1\\ \\ 0\end{matrix} 0} \quad \text{by 1}$$

(Linear Decision Boundary)



Training Logistic Regression

How to learn the parameters $w_0, w_1, \ldots w_d$? (d features)

Training Data $\mathcal{D} = \{(X^{(j)}, Y^{(j)})\}_{j=1}^{n}$ $\quad X^{(j)} = (X_1^{(j)}, \ldots, X_d^{(j)})$

Maximum Likelihood Estimates

$$\hat{\mathbf{w}}_{MLE} = \arg\max_{\mathbf{w}} \prod_{j=1}^{n} P(X^{(j)}, Y^{(j)} \mid \mathbf{w})$$

But there is a problem …

Don't have a model for P(X) or P(X|Y) – only for P(Y|X)

Training Logistic Regression

How to learn the parameters $w_0, w_1, \ldots w_d$? (d features)

Training Data $\{(X^{(j)}, Y^{(j)})\}_{j=1}^{n}$ $X^{(j)} = (X_1^{(j)}, \ldots, X_d^{(j)})$

Maximum (Conditional) Likelihood Estimates

$$\hat{\mathbf{w}}_{MCLE} = \arg\max_{\mathbf{w}} \prod_{j=1}^{n} P(Y^{(j)} \mid X^{(j)}, \mathbf{w})$$

$$= \frac{1}{1+e^{-(w_0+w_i x_i^{(j)})}} \quad \text{if } Y^{(j)}=1$$

Discriminative philosophy – Don't waste effort learning P(X), focus on P(Y|X) – that's all that matters for classification!

Expressing Conditional log Likelihood

$$P(Y=1|X,w) = \frac{1}{1+\exp(-w_0 - \sum_i w_i X_i)}$$

$$P(Y=0|X,w) = \frac{1}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$l(\mathbf{w}) \equiv \ln \prod_j P(y^j|\mathbf{x}^j, \mathbf{w})$$

$$= \sum_j \ln P(y^j|x^j, w)$$

$$= \sum_j \ln \frac{e^{y^j(w_0 + \sum_i w_i x_i^j)}}{1+e^{w_0+\sum_i w_i x_i^j}}$$

$$= \sum_j y^j(w_0 + \sum_i w_i x_i^j) - \ln(1+e^{w_0+\sum_i w_i x_i^j})$$

Expressing Conditional log Likelihood

$$P(Y = 1|X, w) = \frac{1}{1 + \exp(-w_0 - \sum_i w_i X_i)}$$

$$P(Y = 0|X, w) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

$$l(\mathbf{w}) \equiv \ln \prod_j P(y^j|\mathbf{x}^j, \mathbf{w})$$

$$= \sum_j \left[y^j (w_0 + \sum_i^d w_i x_i^j) - \ln(1 + exp(w_0 + \sum_i^d w_i x_i^j)) \right]$$

$J(w_1 \ldots w_d)$

Good news: $l(\mathbf{w})$ is concave function of $\mathbf{w}$! QnA2

Expressing Conditional log Likelihood

$$P(Y=1|X,w) = \frac{1}{1+\exp(-w_0 - \sum_i w_i X_i)}$$

$$P(Y=0|X,w) = \frac{1}{1+\exp(w_0 + \sum_i w_i X_i)}$$

$$\begin{aligned} l(\mathbf{w}) &\equiv \ln \prod_j P(y^j|\mathbf{x}^j,\mathbf{w}) \\ &= \sum_j \left[y^j (w_0 + \sum_i^d w_i x_i^j) - \ln(1 + exp(w_0 + \sum_i^d w_i x_i^j)) \right] \end{aligned}$$

Good news: *l*(**w**) is concave function of **w !** QnA2

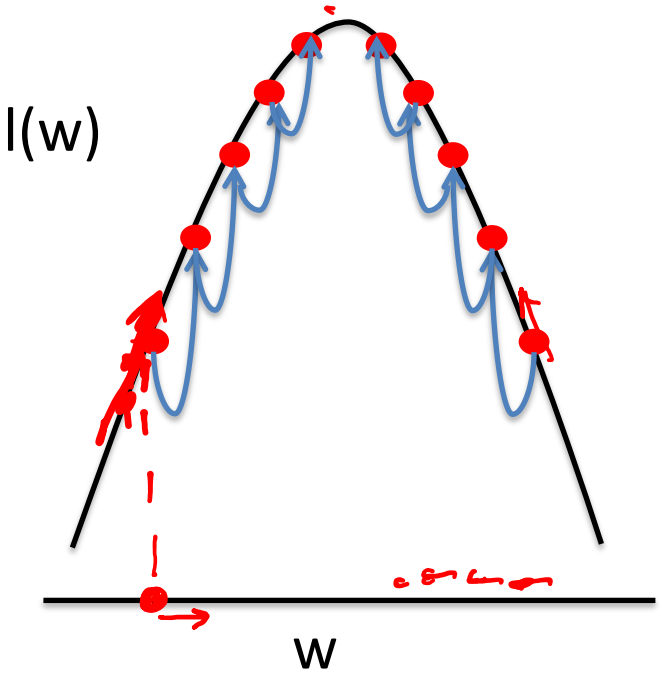
Bad news: no closed-form solution to maximize *l*(**w**)

Good news: can use iterative optimization methods (gradient ascent)

Iteratively optimizing concave function

- Conditional likelihood for Logistic Regression is concave
- Maximum of a concave function can be reached by

Gradient Ascent Algorithm



Initialize: Pick **w** at random

Gradient:

$$\nabla_{\mathbf{w}} l(\mathbf{w}) = [\frac{\partial l(\mathbf{w})}{\partial w_0}, \ldots, \frac{\partial l(\mathbf{w})}{\partial w_d}]'$$

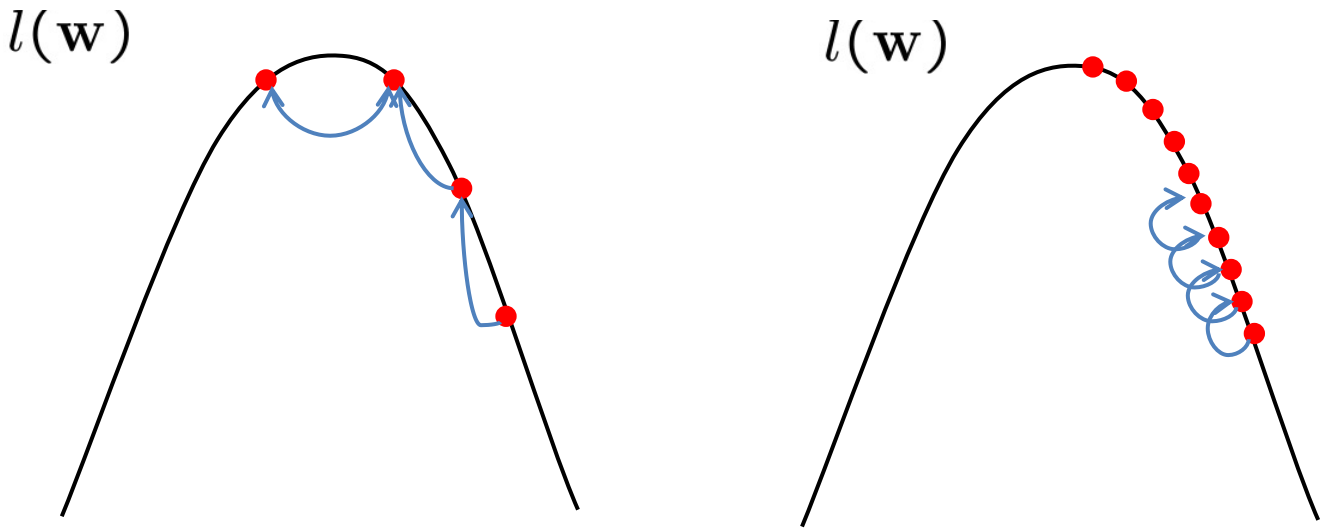
Update rule: Learning rate, $\eta > 0$

$$\Delta \mathbf{w} = \eta \nabla_{\mathbf{w}} l(\mathbf{w})$$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left. \frac{\partial l(\mathbf{w})}{\partial w_i} \right|_t$$

➢ Poll: Effect of step-size η?

Effect of step-size η



Large η => Fast convergence but larger residual error
Also possible oscillations

Small η => Slow convergence but small residual error