Logistic Regression contd…

Aarti Singh

Machine Learning 10-315
Feb 7, 2022

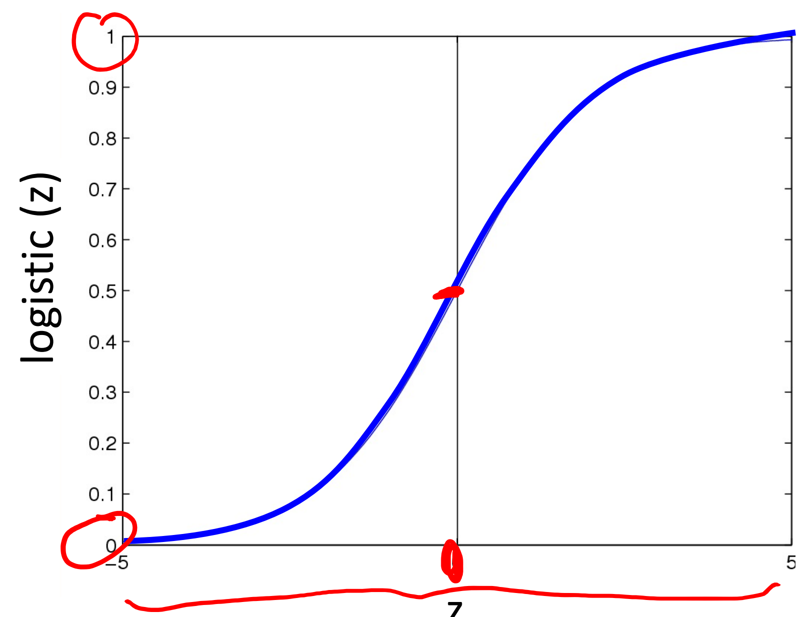
Logistic Regression

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

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

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!

Training Logistic Regression

How to learn the parameters w_0, w_1, … w_d? (d features)

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

Maximum (Conditional) Likelihood Estimates

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

$w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$

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

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

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

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

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

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)

Gradient Ascent for M(C)LE

1) Randomly initialize $w_0^{(0)}, w_1^{(0)} \dots w_d^{(0)}$

2) For each iteration $t = 1 \dots$

Gradient ascent rule for w_0:

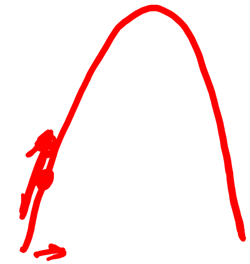
$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \left. \frac{\partial l(\mathbf{w})}{\partial w_0} \right|_t$$

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

$$\frac{\partial l(w)}{\partial w_0} = \sum_j \left[y^j - \frac{1}{1+exp(\cdots)} \cdot exp(\cdots) \cdot 1 \right]$$

Gradient Ascent for M(C)LE

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



Gradient ascent rule for w_0:

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \left.\frac{\partial l(\mathbf{w})}{\partial w_0}\right|_t$$

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

$$\frac{\partial l(\mathbf{w})}{\partial w_0} = \sum_j \left[y^j - \frac{1}{1 + exp(w_0 + \sum_i^d w_i x_i^j)} \cdot exp(w_0 + \sum_i^d w_i x_i^j) \right]$$

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]$$

$$\frac{\partial l(w)}{\partial w_0}$$

Gradient Ascent for M(C)LE Logistic Regression

Gradient ascent algorithm: iterate until change < ε

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]$$

For i=1,…,d,

$w_1 x_1 + w_2 x_2 + \ldots + w_d x_d$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]$$

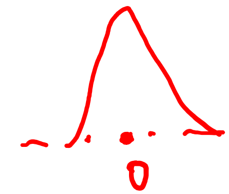
repeat

Predict what current weight thinks label Y should be

- Gradient ascent is simplest of optimization approaches
  - e.g. Stochastic gradient ascent, Momentum methods, Newton method, Conjugate gradient ascent, IRLS (see Bishop 4.3.3)

That's M(C)LE. How about M(C)AP?

$$p(\mathbf{w} \mid Y, \mathbf{X}) \;\propto\; P(Y \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})$$



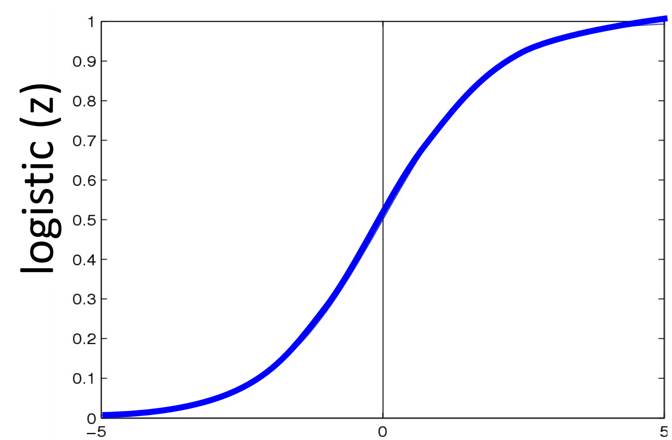
- Define priors on **w**
  - Common assumption: Normal distribution, zero mean, identity covariance
  - "Pushes" parameters towards zero

$$p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \; e^{\frac{-w_i^2}{2\kappa^2}}$$

Zero-mean Gaussian prior

$z = \sum_i w_i X_i + w_0$

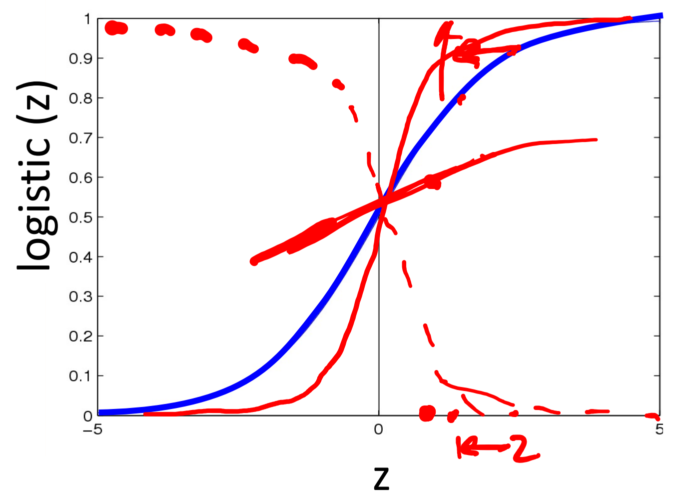
Logistic function (or Sigmoid), σ(z) = : $\dfrac{1}{1 + exp(-z)}$



➢ What happens if we scale z by a large constant?

$w_0 + \sum_i w_i x_i = 0$

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



- Poll: What happens if we scale z (equivalently weights w) by a large constant?

A) The logistic decision boundary shifts towards class 1
B) The logistic decision boundary remains same
C) The logistic classifier tries to separate the data perfectly
D) The logistic classifier allows more mixing of labels on each side of decision boundary

$p(w) \sim N(0, \kappa)$

That's M(C)LE. How about M(C)AP?

$$p(\mathbf{w} \mid Y, \mathbf{X}) \propto P(Y \mid \mathbf{X}, \mathbf{w}) p(\mathbf{w})$$

$$p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \, e^{\frac{-w_i^2}{2\kappa^2}}$$

- M(C)AP estimate

Zero-mean Gaussian prior

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[p(\mathbf{w}) \prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{j=1}^{n} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w}) - \sum_{i=1}^{d} \frac{w_i^2}{2\kappa^2}$$

Still concave objective!

Penalizes large weights

M(C)AP – Gradient

- Gradient

$$p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \; e^{\frac{-w_i^2}{2\kappa^2}}$$

Zero-mean Gaussian prior

$$\frac{\partial}{\partial w_i} \ln \left[p(\mathbf{w}) \prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$\ln p(w) \propto -\sum_i \frac{w_i^2}{2\kappa^2}$$

$$\frac{\partial}{\partial w_i} \ln p(\mathbf{w}) + \frac{\partial}{\partial w_i} \ln \left[\prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$\frac{\partial \ln p(w)}{\partial w_i} \propto -\frac{w_i}{\kappa^2}$$

Same as before

$$\propto \frac{-w_i}{\kappa^2}$$

Extra term Penalizes large weights

Penalization = Regularization

M(C)LE vs. M(C)AP

- Maximum conditional likelihood estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[\prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - P(Y = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]$$

- Maximum conditional a posteriori estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[p(\mathbf{w}) \prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

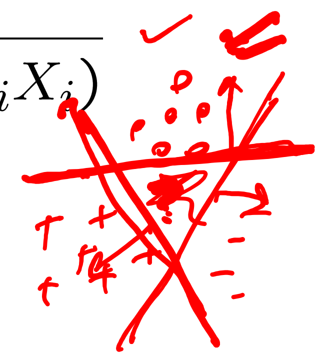
$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\frac{1}{\kappa^2} w_i^{(t)} + \sum_j x_i^j [y^j - P(Y = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})] \right\}$$

Logistic Regression for more than 2 classes

- Logistic regression in more general case, where $Y \in \{y_1,\ldots,y_K\}$

for $k<K$

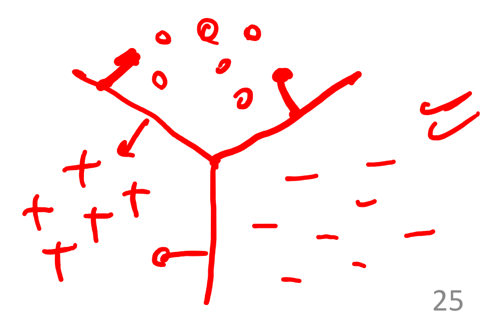
$$P(Y=y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{d} w_{ki}X_i)}{1 + \sum_{j=1}^{K-1}\exp(w_{j0} + \sum_{i=1}^{d} w_{ji}X_i)}$$



for $k=K$ (normalization, so no weights for this class)

$$P(Y=y_K|X) = \frac{1}{1 + \sum_{j=1}^{K-1}\exp(w_{j0} + \sum_{i=1}^{d} w_{ji}X_i)}$$

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



Is the decision boundary still linear?

LR: (d+1) = O(d)

Comparison with Gaussian Naïve Bayes

Gaussian Bayes : d^2

Discrete - : e^d

Naive Bayes : O(d)

Gaussian Naïve Bayes vs. Logistic Regression

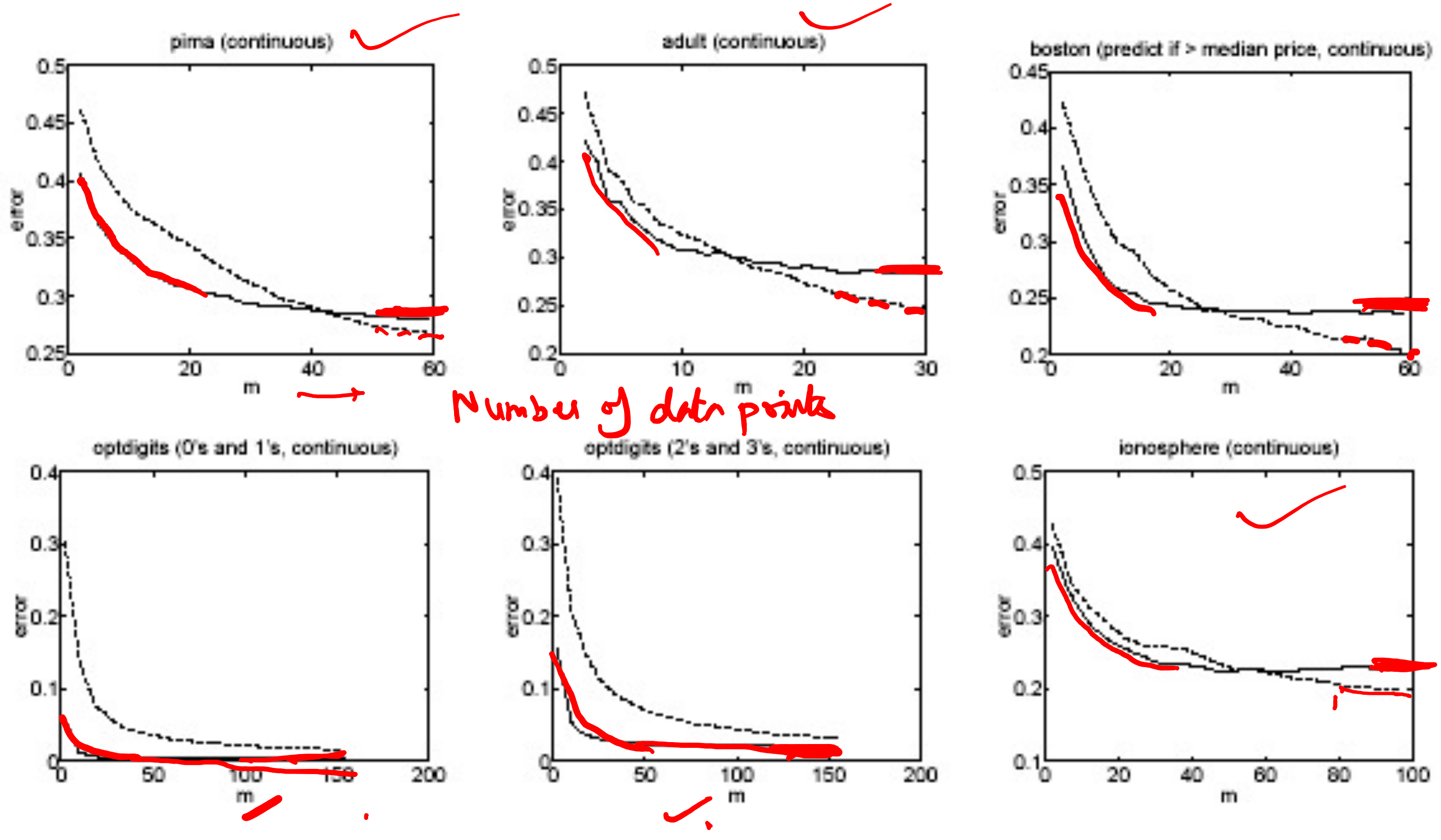
$w_0 + \sum_i w_i x_i = 0$

Set of Gaussian Naïve Bayes parameters (feature variance independent of class label) ⟺ **Set of Logistic Regression parameters**

- Representation equivalence (both yield linear decision boundaries)
  - **But only in a special case!!!** (GNB with class-independent variances)
  - **LR makes no assumptions about** P(**X**|Y) **in learning**!!!
  - **Optimize different functions (MLE/MCLE) or (MAP/MCAP)! Obtain different solutions**

Experimental Comparison (Ng-Jordan'01)

UCI Machine Learning Repository 15 datasets, 8 continuous features, 7 discrete features



More in Paper…

—— Naïve Bayes ----- Logistic Regression

Gaussian Naïve Bayes vs. Logistic Regression

Both GNB and LR have similar number $O(d)$ of parameters.

- GNB error converges faster with increasing number of samples as its parameter estimates are not coupled,

however,

- GNB has higher large sample error if conditional independence assumption DOES NOT hold.

GNB outperforms LR if conditional independence assumption holds.

What you should know

- LR is a linear classifier
- LR optimized by maximizing conditional likelihood or conditional posterior
  - no closed-form solution
  - concave ! global optimum with gradient ascent
- Gaussian Naïve Bayes with class-independent variances representationally equivalent to LR
  - Solution differs because of objective (loss) function
- In general, NB and LR make different assumptions
  - NB: Features independent given class ! assumption on P(**X**|Y)
  - LR: Functional form of P(Y|**X**), no assumption on P(**X**|Y)
- Convergence rates
  - GNB (usually) needs less data
  - LR (usually) gets to better solutions in the limit