Outline

- Binary Logistic Regression
- Convexity of Binary LR
- Multiclass LR
- Regularized LR
LR for Binary Classification

Using a sigmoid function to model the probability of the class label (yes or no) for each input instance as:

\[ P_w(y=1|x) = \sigma(z) = \frac{1}{1 + \exp(-z)} \]
\[ P_w(y=0|x) = 1 - \sigma(z) = \frac{\exp(-z)}{1 + \exp(-z)} \]

where

\[ x = (1, x_1, \ldots, x_m)^T, \quad y \in \{0, 1\} \]
\[ w = (w_0, w_1, \ldots, w_m)^T \quad \text{model parameters} \]
\[ z_w(x) = w^T x = w_0 + w_1 x_1 + \cdots + w_m x_m \quad \text{Linear in } x_i \text{'s} \]

---

Logistic Sigmoid Function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
\[ z \in (-\infty, \infty), \quad \sigma(z) \in (0, 1), \quad \sigma(0) = 0.5, \quad \sigma(z) = (1 - \sigma(-z)) \]
Derivative of Sigmoid Function

\[ \frac{d\sigma(z)}{dz} = \frac{d}{dz} \left( \frac{1}{1 + \exp(-z)} \right) \]

\[ = -1 \cdot \exp(-z) \]

\[ = \frac{1}{(1 + \exp(-z))^2} \cdot \exp(-z) \]

\[ = \frac{1}{(1 + \exp(-z))^2} \]

\[ = \sigma(z)(1 - \sigma(z)) \]

Likelihood of data given model \( w \)

\[ P_w(y_i = 1 \mid x_i) = \sigma(w^T x_i) = \sigma_i \]

\[ P_w(y_i = 0 \mid x_i) = 1 - \sigma(w^T x_i) = 1 - \sigma_i \]

\[ \rightarrow P_w(y_i \mid x_i) = \sigma_i^{y_i} (1 - \sigma_i)^{1-y_i} \quad \text{where} \quad y_i \in \{0,1\} \]

\[ \rightarrow P_w(D) = \prod_{i=1}^{n} \sigma_i^{y_i} (1 - \sigma_i)^{1-y_i} \quad \text{where} \quad \sigma_i = \frac{1}{1 + e^{-w^T x_i}} \]

\[ \rightarrow \log P_w(D) = \sum_{i=1}^{n} \left[ y_i \log \sigma_i + (1 - y_i) \log(1 - \sigma_i) \right] \]
Optimizing the model w.r.t. $w$

Likelihood function:
\[
L(w) = P(D \mid w) = \prod_{i=1}^{n} \sigma(w^T x_i)^{y_i} \left(1 - \sigma(w^T x_i)\right)^{1-y_i}
\]

Log-likelihood:
\[
l(w) = \ln L(w) = \sum_{i=1}^{n} \left\{ y_i \ln \sigma(w^T x_i) + (1 - y_i) \ln(1 - \sigma(w^T x_i)) \right\}
\]

Optimize the model:
\[
\hat{w} = \arg \max_w \{ P(D \mid w) \} = \arg \max_w \{ l(w) \}
\]

---

Two Popular Approaches

- **Gradient Ascent (or Descent)**
  - Use the first-order derivative of $l(w)$
  - Need to pre-specify the "learning rate" (step size)
  - Fast to compute in each step but may take many steps

- **Newton-Raphson**
  - Use the first-order and second-order derivatives of $l(w)$
  - No need to pre-specify the step size
  - Converge faster but costly in each step
The gradient on a single training instance

\[ l_i(w) = y_i \ln \sigma(z_i) + (1 - y_i) \ln(1 - \sigma(z_i)) \]

\[ z_i = w^T x_i = w_0 + \sum_{j=1}^m w_j x_{ij} \]

\[ \frac{\partial}{\partial w_j} l_i(w) = \frac{d}{d \sigma(z_i)} \left( \frac{d \sigma(z_i)}{d z_i} \right) \frac{d z_i}{d w_j} \]

\[ = \left( y_i \frac{1}{\sigma(z_i)} - (1 - y_i) \frac{1}{1 - \sigma(z_i)} \right) \sigma(z_i)(1 - \sigma(z_i)) x_{ij} \]

\[ = (y_i - \sigma(z_i)) x_{ij} \]

\[ \nabla l_i(w) = \left( \frac{\partial}{\partial w_0} l_i(w), \frac{\partial}{\partial w_1} l_i(w), \ldots, \frac{\partial}{\partial w_m} l_i(w) \right)^T \]

Gradient ascent on a training set

The single-instance version:

\[ D = \{ (x^{(i)}, y^{(i)}) \} \]

Loop until convergence {

for \( i = 1 \) to \(|D|\) {

\( w := w + \eta \nabla l_i(w) \) \hspace{1cm} (\( \eta > 0 \), the prespecified learning rate)

}

The batch version:

Loop until convergence {

\( w := w + \eta \sum_{i=1}^{\text{\#}} \nabla l_i(w) \) \hspace{1cm} (\( \eta > 0 \))

}

Guaranteed: \( l(w^{(0)}) \leq l(w^{(1)}) \leq l(w^{(2)}) \ldots \)
Gradient Ascent/Descent

\[ x_{t+1} = x_t \pm \eta \nabla f(x_t) \]
\[ \eta > 0, \quad t = 0, 1, 2, \ldots \]

In our case: \( x \rightarrow w \)
and \( f = l(w) \)

(from Wikipedia)

Newton-Raphson Method
(in the case of one-dimensional \( w \))

- Given current \( w \), we want move it with the optimal step size (\( \varepsilon \)) in the right direction.
- Taylor series:
  \[ l(w + \varepsilon) = l(w) + \frac{l'(w)}{1!} \varepsilon + \frac{l''(w)}{2!} \varepsilon^2 + \cdots \]
- At the mode (with respect to \( \varepsilon \))
  \[ 0 = \frac{d}{d\varepsilon} l(w + \varepsilon) \approx l'(w) + l''(w) \varepsilon \quad \Rightarrow \quad \varepsilon = -\frac{l'(w)}{l''(w)} \]
- Update rule:
  \[ w := w - \frac{l'(w)}{l''(w)} \]
Newton-Raphson Method
(in the case of high-dimensional \( w \))

- Taylor series:
  \[
  l(w + \varepsilon) = l(w) + \nabla l(w)\varepsilon + \varepsilon^T \frac{\nabla \nabla l(w)}{2!} \varepsilon + \ldots
  \]

- Update rule:
  \[
  w := w - \frac{(\nabla \nabla l(w))^{-1} \nabla l(w)}{H(w)}
  \]
  where the gradient

\[
\nabla l(w) = \left( \frac{\partial}{\partial w_0} l(w), \frac{\partial}{\partial w_1} l(w), \ldots, \frac{\partial}{\partial w_m} l(w) \right)^T
\]

\[
\nabla \nabla l(w) = H(w) = \left( H_{ij} \right), \quad H_{ij} = \frac{\partial^2}{\partial w_j \partial w_i} l(w)
\]

@Yiming Yang, Lecture on Logistic Regression

Newton-Raphson Method (cont’d)

First order derivative (as shown before in slide #14):

\[
\frac{\partial}{\partial w_j} l_i(w) = \sum_{i=1}^{n} (y_i - \sigma_i) x_{ij}
\]

where \( \sigma_i = \sigma(w^T x^{(i)}) \)

Second order derivative:

\[
\frac{\partial^2}{\partial w_j \partial w_i} l_i(w) = \frac{\partial}{\partial w_j} \left( \frac{\partial}{\partial w_i} l_i(w) \right) = \sum_{i=1}^{n} \frac{\partial}{\partial w_i} \left( (y_i - \sigma(w^T x^{(i)})) x_{ij} \right)
\]

\[
= -\sum_{i=1}^{n} \frac{d\sigma_i}{dz_i} \left( \frac{\partial z_i}{\partial w_j} \right) x_{ij} = -\sum_{i=1}^{n} \sigma_i (1 - \sigma_i) x_{ij} x_{ij}
\]

@Yiming Yang, Lecture on Logistic Regression
Newton-Raphson Method (cont’d)

The gradient (batch version):
\[ \nabla l(w) = \sum_{i=1}^{n} (y_i - \sigma_i) x_i = X^T (y - \sigma) \]
\[ \nabla l(w) \] is the weighted sum of the training documents;
\( X \) is \( m \times n \), whose columns \( (x_i)'s \) are the training documents;
\( y = (y_1, y_2, \ldots, y_n)' \) is the vector of true labels of \( n \) training doc's;
\( \sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)' \) is the vector of predicted probabilities \( \sigma_i = \sigma(w^T x_i) \).

The Hessian:
\[ H(w) = -\sum_{i=1}^{n} \sigma_i (1 - \sigma_i) x_i (x_i)^T = -X^T \Lambda X \]
\[ \Lambda = \text{diag}(\sigma_1 (1 - \sigma_1), \sigma_2 (1 - \sigma_2), \ldots, \sigma_n (1 - \sigma_n)) \]

Update rule in 1-dimentional LR
\[ w' := w^{\text{old}} - \frac{l'(w^{\text{old}})}{l''(w^{\text{old}})} \]

Update rule in high-dimensional LR
\[ w := w^{\text{old}} - H(w^{\text{old}})^{-1} \nabla l(w^{\text{old}}) \]
\[ := w^{\text{old}} + (X^T \Lambda X)^{-1} X^T (y - \sigma) \]

Solving a weighted linear regression problem in each iteration, and hence called "iteratively reweighted least squares" (IRLS)
Newton-Raphson Method (cont’d)

Update rule

\[
\mathbf{w} := \mathbf{w}^{\text{old}} + (\mathbf{X}^T \mathbf{A} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{\sigma})
\]

\[
= (\mathbf{X}^T \mathbf{A} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{A} \mathbf{w}^{\text{old}} + (\mathbf{X}^T \mathbf{A} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{A} \mathbf{w}^{\text{old}} + \mathbf{A}^{-1} (\mathbf{y} - \mathbf{\sigma})
\]

\[
= (\mathbf{X}^T \mathbf{A} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{A} \mathbf{z}
\]

Standard linear regression

\[
\min_w (z - \mathbf{X} \mathbf{w})^T (z - \mathbf{X} \mathbf{w}) \quad \rightarrow \quad \mathbf{\hat{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{z}
\]

@Yiming Yang, Lecture on Logistic Regression

Newton-Raphson Method (cont’d)

- Standard LLSF

\[
\mathbf{\hat{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{z}
\]

- Weighted LLSF

Let \( \mathbf{z}' := \mathbf{A}^{1/2} \mathbf{z}, \mathbf{X}' := \mathbf{A}^{1/2} \mathbf{X} \)

\[
\mathbf{\hat{w}} = \min_w (\|z' - \mathbf{X}' \mathbf{w}\|^2) = (\mathbf{X}^T \mathbf{A}^{1/2} \mathbf{A}^{1/2} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{A}^{1/2} \mathbf{A}^{1/2} \mathbf{z} = (\mathbf{X}^T \mathbf{A} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{A} \mathbf{z}
\]

the updating rule in Newton-Raphson

@Yiming Yang, Lecture on Logistic Regression
Outline

- Binary Logistic Regression
- Convexity of Binary LR
- Decision Boundaries
- Multiclass LR
- Regularized LR

Globally optimal solution guaranteed?

- We can check the convexity of the objective function.
  - If it is convex, then there is a single global minimum.
  - If it is concave, then there is a single global maximum.
  - If it is neither convex or concave, then the global optimal is not guaranteed.
Examples of 1-dimensional functions
Convex, Concave or Neither

- A function $f$ is called **convex** if:
  $$\forall x_1, x_2 \in X, \forall t \in [0, 1]: \quad f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2).$$

- A function $f$ is called **strictly convex** if:
  $$\forall x_1 \neq x_2 \in X, \forall t \in (0, 1): \quad f(tx_1 + (1-t)x_2) < tf(x_1) + (1-t)f(x_2).$$
How can we tell the convexity?

- Convex function in one-dimensional case
  \[ f'' \geq 0 \quad \rightarrow \quad \text{convex} \]
  e.g. \( f(x) = x^2, \quad f' = 2x, \quad f'' = 2 \).

- If \( f(x) \) is convex, then \(-f(x)\) is concave
  
  e.g. \( f(x) = 3 - 2x^2, \quad f' = -4x, \quad f'' = -4 \).

- A multi-dimensional function is convex (concave) if its hessian matrix is positive (negative) semi-definite:
  \[ H \succeq 0, \quad u^T Hu \succeq 0, \forall u \]

Example: A two-dims functions
Convex, Concave or Neither (cont’d)

\[
\begin{align*}
\forall u, u^T Hu &\geq 0 & H \text{ is positive semidefinite} &\rightarrow \text{convex} \\
\forall u, u^T Hu &< 0 & H \text{ is negative semidefinite} &\rightarrow \text{concave} \\
\text{neither of the above} &\rightarrow \text{local opt.}
\end{align*}
\]
The Hessian is negative semi-definite

- We have shown \( H^{(LR)} = -X^TAX \)
- We need to show it being positive/negative semi-definite

\[
H^{(LR)} = -X^TAX = -\frac{X^T}{X} \Lambda^{1/2} \frac{X^T}{X} \\
\text{where } \Lambda = \text{diag} \left( \sigma_i (1 - \sigma_i) \right)_{i=1,...,n} \\
\forall u \in \mathbb{R}^m, u^T H^{(LR)} u = \frac{-u^T}{\sqrt{y}} X^T X u = -v^T v = -\|v\| \leq 0
\]

Thus, \( H \) is negative semi-definite, which means that LR has a concave objective function.

Outline

- Binary Logistic Regression
- Convexity of Binary LR
- Multiclass LR
- Regularized LR
Outline

- Binary Logistic Regression
- Convexity of Binary LR
- Decision Boundaries
- Multiclass LR
- Regularized LR

Difference in training set construction

Multi-class Labeled Training Set

Binary-labeled Training Set

Negative Examples

Positive Examples
Multiclass LR (Softmax LR)

Denoting by \( Y \in \{1, 2, \cdots, K\} \) a multivariate target variable, and the multivariate distribution as:

\[
Y \mid x \sim Mul(p_1(x), p_2(x), p_K(x))
\]

where

\[
p_k(x) = \Pr(Y = k \mid x) = \begin{cases} 
\frac{\exp(w_k^T x)}{1 + \sum_{k'=1}^{K} \exp(w_{k'}^T x)} & \text{for } k < K; \\
\frac{1}{1 + \sum_{k'=1}^{K} \exp(w_{k'}^T x)} & \text{for } k = K.
\end{cases}
\]

Decision Rule: \( \hat{Y} \mid x = \arg \max_k \{ p_k(x) \} \)

Softmax Objective Function

Softmax discriminant function of each category

\[
p_k(x) = \frac{\exp(w_k^T x)}{\sum_{k=1}^{K} \exp(w_k^T x)}
\]

- Log-likelihood

\[
l(D | W) = \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log p_k(x_i) \\
= \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i - \sum_{i=1}^{N} \log \sum_{k'=1}^{K} \exp(w_{k'}^T x_i)
\]

where \( D = \{(x_i, y_{ik})\}, W = \{w_k\} \) and \( y_{ik} = I(y(x_i) = k) \).

- Objective function of regularized softmax

\[
L(W) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \sum_{i=1}^{N} \log \sum_{k'=1}^{K} \exp(w_{k'}^T x_i)
\]
Bottleneck in parallel computing

- **Optimization Problem**

\[
\min_W L(W) = \min_w \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \sum_{i=1}^{N} \log \sum_{k'=1}^{K} \exp(w_{k'}^T x_i)
\]

- How to decouple the third term for parallel computing?

Distributed training of regularized LR classifiers

- **Bottleneck**

\[
L(W) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} w_k^T x_i + \sum_{i=1}^{N} \log \sum_{k'=1}^{K} \exp(w_{k'}^T x_i)
\]

- **Log-concavity bound** (the 1st order concavity property) of log function

\[
\log(v) \leq \alpha v - \log(\alpha) - 1 \quad \forall v, \alpha > 0
\]

- **Log-partition term** for each instance \(i\) in LR is bounded as:

\[
\log \left( \sum_{k=1}^{K} \exp(w_k^T x_i) \right) \leq \alpha \sum_{k=1}^{K} \exp(w_k^T x_i) - \log(\alpha) - 1 \quad \alpha > 0.
\]
The Modified Objective Function

\[ \min_{\alpha > 0, \mathbf{w}} F(\mathbf{W}, \alpha) \]

\[ F(\mathbf{W}, \alpha) = \frac{\lambda}{2} \sum_{k=1}^{K} \|w_k\|^2 - \sum_{i=1}^{N} \sum_{k=1}^{K} (y_{ik} w_k^T x_i - \alpha_i \exp(w_k^T x_i) - \log \alpha_i - 1) \]

Convergence-related Properties (proof in our ICML 2013 paper)

1) It does not preserve the convexity of the original objective function (because we have \(\alpha_i\)'s as the additional variables).

2) However, it has exactly one stationary point that is the same stationary point of the original convex function of softmax.

3) A block co-ordinate descent procedure guarantees to converge to the stationary point which is the optimal solution of softmax.

07/10/2017 @Yiming Yang, lecture on Large-scale Text Classification

Outline

- Binary Logistic Regression
- Convexity of Binary LR
- Multiclass LR
- Regularized LR
Regularized Logistic Regression (RLR)

- So far we have focused on the MLE objective as:
  \[ \hat{w}^{LR} = \arg\max_w \{ I_D(w) \} \]
  where \( I_D(w) = \sum_{i=1}^{n} \left\{ y_i \ln \sigma(w^T x_i) + (1 - y_i) \ln(1 - \sigma(w^T x_i)) \right\} \)

- Now we add a regularization term as:
  \[ \hat{w}^{RLR} = \arg\max_w \left\{ I_D(w) - \frac{1}{2} C \| w \|^2 \right\} \]

- Equivalent to adding a Bayesian prior for \( w \) (next slide)

Maximum A Posterior (MAP) Solution

- Bayesian Prior (Assumption)
  \( w \sim N(0, \sigma^2 I) \)
  \[ P(w) = \frac{1}{Z_0} \exp \left( -\frac{w^T w}{2\sigma^2} \right) \]
  where \( Z_0 \) is some constant (normalization factor).

- Posterior Probability
  \[ P(w | D) = \frac{P(D | w) P(w)}{P(D)} = \arg\max_w P(D | w) P(w) \]

- Objective
  \[ \hat{w}^{RLR} = \arg\max_w \left\{ \log P(D | w) + \log P(w) \right\} \]
  \[ = \arg\max_w \left\{ \log P(D | w) + \log P(w) \right\} \]
  \[ = \arg\max_w \left\{ (w) - \lambda w^T w + \text{some constant} \right\} \text{ where } \lambda = \frac{1}{2\sigma^2} \]

- MAP solution for RLR assumes a "non-informative" Gaussian prior of \( w \).
Summary on LR

- Explicit probabilistic reasoning
- As effective as SVM when regularized
- Relatively efficient to compute
- Easy to extend with regularization (e.g., L1 or L2 norm of the parameter vector)