Introduction to Machine Learning

Logistic Regression

Mingchen Gao

Computer Science & Engineering
State University of New York at Buffalo
Buffalo, NY, USA
Slides adapted from Varun Chandola mgao8@buffalo.edu

University at Buffalo
Department of Computer Science and Engineering
School of Engeering and Applied Sciences

Outline

Generative vs. Discriminative Models

Logistic Regression

Logistic Regression - Training

- Using Gradient Descent for Learning Weights
- Using Newton's Method
- Regularization with Logistic Regression
- Handling Multiple Classes

Generative vs. Discriminative Classifiers

- Probabilistic classification task:

$$p(Y = benign|\mathbf{X} = \mathbf{x}), p(Y = malicious|\mathbf{X} = \mathbf{x})$$

- How do you estimate $p(y|\mathbf{x})$?

$$p(y|\mathbf{x}) = \frac{p(y, \mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

- Two step approach - Estimate generative model and then posterior for y (Naïve Bayes)
- Solving a more general problem [2, 1]
- Why not directly model $p(y|\mathbf{x})$? - **Discriminative approach**

Examples of Generative vs. Discriminative Models

Generative models

1. Naive Bayes
2. Gaussian Discriminate Analysis
3. Gaussian Mixture Model
4. Hidden Markov Model
5. Generative Adversarial Network (GAN)

Discriminative Models

1. Linear Regression
2. Logistic Regression
3. Support Vector Machine (SVM)
4. Neural Networks
5. Random Forests

Logistic Regression

- $y|\mathbf{x}$ is a *Bernoulli* distribution with parameter $\theta = sigmoid(\mathbf{w}^\top \mathbf{x})$
- When a new input $\mathbf{x}^*$ arrives, we toss a coin which has $sigmoid(\mathbf{w}^\top \mathbf{x}^*)$ as the probability of heads
- If outcome is heads, the predicted class is 1 else 0
- Learns a linear boundary

Learning Task for Logistic Regression

Given training examples $\langle \mathbf{x}_i, y_i \rangle_{i=1}^D$, learn $\mathbf{w}$

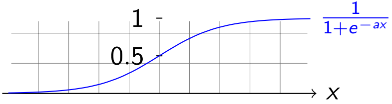
Logistic Regression - Recap

Bayesian Interpretation

- Directly model $p(y|\mathbf{x})$ ($y \in \{0, 1\}$)
- $p(y|\mathbf{x}) \sim Bernoulli(\theta = sigmoid(\mathbf{w}^\top \mathbf{x}))$

Geometric Interpretation

- Use regression to predict discrete values
- *Squash* output to $[0, 1]$ using sigmoid function
- Output less than 0.5 is one class and greater than 0.5 is the other



Learning Parameters

- MLE Approach
- Assume that $y \in \{0, 1\}$
- What is the likelihood for a bernoulli sample?
  - If $y_i = 1$, $p(y_i) = \theta_i = \frac{1}{1+exp(-\mathbf{w}^\top \mathbf{x}_i)}$
  - If $y_i = 0$, $p(y_i) = 1 - \theta_i = \frac{1}{1+exp(\mathbf{w}^\top \mathbf{x}_i)}$
  - In general, $p(y_i) = \theta_i^{y_i}(1-\theta_i)^{1-y_i}$

Negative Log-likelihood (NLL)

$$NLL(\mathbf{w}) = \sum_{i=1}^{N} -y_i \log \theta_i - (1 - y_i) \log (1 - \theta_i)$$

- No closed form solution for maximizing log-likelihood/or minimizing negative log-likelihood

Using Gradient Descent for Learning Weights

- Compute gradient of LL with respect to **w**
- A convex function of **w** with a unique global maximum

$$\frac{d}{d\mathbf{w}} NLL(\mathbf{w}) = \sum_{i=1}^{N} (\theta_i - y_i)\mathbf{x}_i$$

- Update rule:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \frac{d}{d\mathbf{w}_k} LL(\mathbf{w}_k)$$

Using Newton's Method

- Setting η is sometimes *tricky*
- Too large – incorrect results
- Too small – slow convergence
- Another way to speed up convergence:

Newton's Method

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbf{H}_k^{-1} \frac{d}{d\mathbf{w}_k} NLL(\mathbf{w}_k)$$

What is the Hessian?

- Hessian or **H** is the second order derivative of the objective function
- Newton's method belong to the family of **second order optimization algorithms**
- For logistic regression, the Hessian is:

$$H = -\sum_i \theta_i(1-\theta_i)\mathbf{x}_i\mathbf{x}_i^\top$$

Regularization with Logistic Regression

- **Overfitting** is an issue, especially with large number of features
- Add a *Gaussian prior* $\sim \mathcal{N}(\mathbf{0}, \tau^2)$ (Or a regularization penalty)
- Easy to incorporate in the gradient descent based approach

$$NLL'(\mathbf{w}) = NLL(\mathbf{w}) + \frac{1}{2}\lambda \mathbf{w}^\top \mathbf{w}$$

$$\frac{d}{d\mathbf{w}} NLL'(\mathbf{w}) = \frac{d}{d\mathbf{w}} NLL(\mathbf{w}) + \lambda \mathbf{w}$$

$$H' = H + \lambda I$$

where I is the identity matrix.

Handling Multiple Classes

- One vs. Rest and One vs. Other
- $p(y|\mathbf{x}) \sim Multinoulli(\boldsymbol{\theta})$
- Multinoulli parameter vector $\boldsymbol{\theta}$ is defined as:

$$\theta_j = \frac{exp(\mathbf{w}_j^\top \mathbf{x})}{\sum_{k=1}^{C} exp(\mathbf{w}_k^\top \mathbf{x})}$$

- Multiclass logistic regression has C weight vectors to learn

References

Murphy Book Chapter 10

A. Y. Ng and M. I. Jordan.
On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes.
In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *NIPS*, pages 841–848. MIT Press, 2001.

V. Vapnik.
Statistical learning theory.
Wiley, 1998.