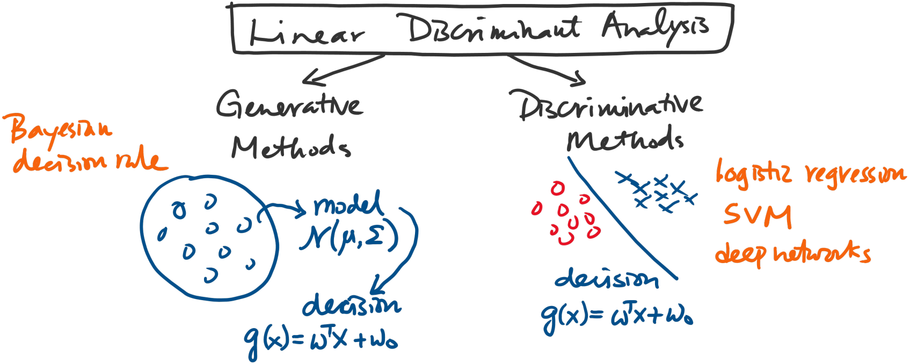
ECE595 / STAT598: Machine Learning I Lecture 15.1: Logistic Regression 2 - Gradient Descent

Spring 2020

Stanley Chan

School of Electrical and Computer Engineering
Purdue University

Overview



- In linear discriminant analysis (LDA), there are generally two types of approaches
- **Generative approach**: Estimate model, then define the classifier
- **Discriminative approach**: Directly define the classifier

Outline

Discriminative Approaches

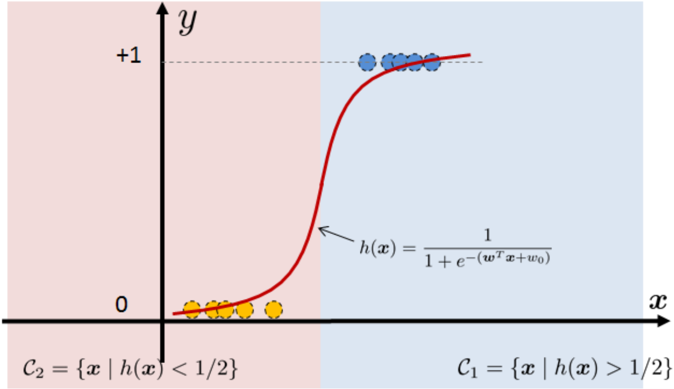
- Lecture 14 Logistic Regression 1
- Lecture 15 Logistic Regression 2

This lecture: Logistic Regression 2

- Gradient Descent
  - Convexity
  - Gradient
  - Regularization
- Connection with Bayes
  - Derivation
  - Interpretation
- Comparison with Linear Regression
  - Is logistic regression better than linear?
  - Case studies

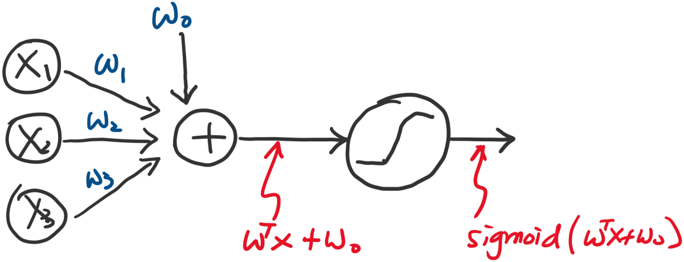
From Linear to Logistic Regression

- Can we replace $g(\boldsymbol{x})$ by $\text{sign}(g(\boldsymbol{x}))$?
- How about a soft-version of $\text{sign}(g(\boldsymbol{x}))$?
- This gives a logistic regression.



Logistic Regression and Deep Learning

- Logistic regression can be considered as the last layer of a deep network
- Inputs are $\boldsymbol{x}_n$, weights are $\boldsymbol{w}$
- The sigmoid function is the nonlinear activation
- To train the model, you compare the prediction error and minimize the loss by updating the weights



Training Loss Function

$$
\begin{aligned}
J(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \mathcal{L}(h_\theta(\boldsymbol{x}_n), y_n) \\
&= \sum_{n=1}^{N} -\Big\{ y_n \log h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) + (1-y_n)\log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \Big\}
\end{aligned}
$$

- This is called the cross-entropy loss
- Consider two cases

$$
y_n \log h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) = \begin{cases} 0, & \text{if} \quad y_n = 1, \quad \text{and} \quad h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) = 1, \\ -\infty, & \text{if} \quad y_n = 1, \quad \text{and} \quad h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) = 0, \end{cases}
$$

$$
(1-y_n)(1-\log h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) = \begin{cases} 0, & \text{if} \quad y_n = 0, \quad \text{and} \quad h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) = 0, \\ -\infty, & \text{if} \quad y_n = 0, \quad \text{and} \quad h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) = 1. \end{cases}
$$

- No solution if mismatch

Convexity of Logistic Training Loss

Recall that

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{n} -\left\{ y_n \log\left(\frac{h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)}{1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)}\right) + \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \right\}$$

- The first term is linear, so it is convex.
- The second term: Gradient:

$$\nabla_{\boldsymbol{\theta}}[-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))] = -\nabla_{\boldsymbol{\theta}}\left[\log\left(1 - \frac{1}{1 + e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}\right)\right]$$

$$= -\nabla_{\boldsymbol{\theta}}\left[\log \frac{e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}{1 + e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}\right] = -\nabla_{\boldsymbol{\theta}}\left[\log e^{-\boldsymbol{\theta}^T\boldsymbol{x}} - \log(1 + e^{-\boldsymbol{\theta}^T\boldsymbol{x}})\right]$$

$$= -\nabla_{\boldsymbol{\theta}}\left[-\boldsymbol{\theta}^T\boldsymbol{x} - \log(1 + e^{-\boldsymbol{\theta}^T\boldsymbol{x}})\right] = \boldsymbol{x} + \nabla_{\boldsymbol{\theta}}\left[\log\left(1 + e^{-\boldsymbol{\theta}^T\boldsymbol{x}}\right)\right]$$

$$= \boldsymbol{x} + \left(\frac{-e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}{1 + e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}\right)\boldsymbol{x} = h_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{x}.$$

Convexity of Logistic Training Loss

- Gradient of second term is

$$\nabla_{\boldsymbol{\theta}}[-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))] = h_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{x}.$$

- Hessian is:

$$\begin{aligned}
\nabla^2_{\boldsymbol{\theta}}[-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))] &= \nabla_{\boldsymbol{\theta}}\left[h_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{x}\right] \\
&= \nabla_{\boldsymbol{\theta}}\left[\left(\frac{1}{1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}\right)\boldsymbol{x}\right] \\
&= \left(\frac{1}{(1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}})^2}\right)\left(-e^{-\boldsymbol{\theta}^T\boldsymbol{x}}\right)\boldsymbol{x}\boldsymbol{x}^T \\
&= \left(\frac{1}{1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}\right)\left(1-\frac{1}{1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}\right)\boldsymbol{x}\boldsymbol{x}^T \\
&= h_{\boldsymbol{\theta}}(\boldsymbol{x})[1-h_{\boldsymbol{\theta}}(\boldsymbol{x})]\boldsymbol{x}\boldsymbol{x}^T.
\end{aligned}$$

Convexity of Logistic Training Loss

- For any $\boldsymbol{v} \in \mathbb{R}^d$, we have that

$$\begin{aligned}\boldsymbol{v}^T \nabla^2_{\boldsymbol{\theta}}[-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))]\boldsymbol{v} &= \boldsymbol{v}^T \left[h_{\boldsymbol{\theta}}(\boldsymbol{x})[1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})]\boldsymbol{x}\boldsymbol{x}^T \right] \boldsymbol{v} \\ &= (h_{\boldsymbol{\theta}}(\boldsymbol{x})[1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})]) \, \|\boldsymbol{v}^T\boldsymbol{x}\|^2 \geq 0.\end{aligned}$$

- Therefore the Hessian is positive semi-definite.
- So $-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})$ is convex in $\boldsymbol{\theta}$.
- Conclusion: The training loss function

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{n} -\left\{ y_n \log\left(\frac{h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)}{1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)} \right) + \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \right\}$$

  is **convex** in $\boldsymbol{\theta}$.
- So we can use convex optimization algorithms to find $\boldsymbol{\theta}$.

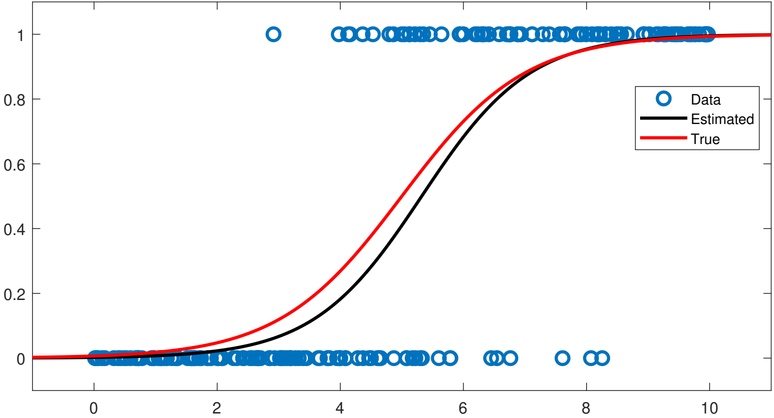
Convex Optimization for Logistic Regression

- We can use CVX to solve the logistic regression problem
- But it requires some re-organization of the equations

$$
\begin{aligned}
J(\boldsymbol{\theta}) &= \sum_{n=1}^{N} -\left\{ y_n \boldsymbol{\theta}^T \boldsymbol{x}_n + \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \right\} \\
&= \sum_{n=1}^{N} -\left\{ y_n \boldsymbol{\theta}^T \boldsymbol{x}_n + \log\left(1 - \frac{e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}}{1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}}\right) \right\} \\
&= \sum_{n=1}^{N} -\left\{ y_n \boldsymbol{\theta}^T \boldsymbol{x}_n - \log\left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}\right) \right\} \\
&= -\left\{ \left(\sum_{n=1}^{N} y_n \boldsymbol{x}_n \right)^T \boldsymbol{\theta} - \sum_{n=1}^{N} \log\left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}\right) \right\}.
\end{aligned}
$$

- The last term is a sum of `log-sum-exp`: $\log(e^0 + e^{\theta^T x})$.

Convex Optimization for Logistic Regression



- Black: The true model. You create it.
- Blue circles: Samples drawn from the true distribution.
- Red: Trained model from the samples.

Gradient Descent for Logistic Regression

- The training loss function is

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{n} -\Big\{ y_n \boldsymbol{\theta}^T \boldsymbol{x}_n + \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \Big\}.$$

- Recall that

$$\nabla_{\boldsymbol{\theta}}[-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))] = h_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{x}.$$

- You can run gradient descent

$$\begin{aligned}
\boldsymbol{\theta}^{(k+1)} &= \boldsymbol{\theta}^{(k)} - \alpha_k \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}^{(k)}) \\
&= \boldsymbol{\theta}^{(k)} - \alpha_k \left(\sum_{n=1}^{N} (h_{\boldsymbol{\theta}^{(k)}}(\boldsymbol{x}_n) - y_n)\boldsymbol{x}_n \right).
\end{aligned}$$

- Since the loss function is convex, guaranteed to find global minimum.

Regularization in Logistic Regression

- The loss function is

$$
\begin{aligned}
J(\boldsymbol{\theta}) &= \sum_{n=1}^{n} -\left\{ y_n \boldsymbol{\theta}^T \boldsymbol{x}_n + \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \right\} \\
&= \sum_{n=1}^{n} -\left\{ y_n \boldsymbol{\theta}^T \boldsymbol{x}_n + \log\left(1 - \frac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}_n}}\right) \right\}
\end{aligned}
$$

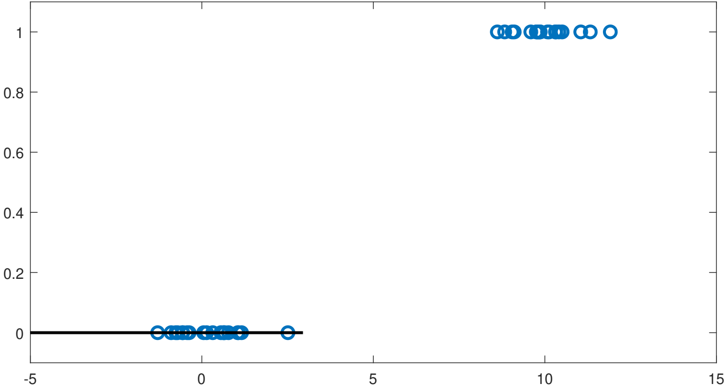
- What if $h_{\boldsymbol{\theta}}(\boldsymbol{x}_n) = 1$? (We need $\boldsymbol{\theta}^T \boldsymbol{x}_n = \infty$.)
- Then we have $\log(1-1) = \log 0$, which is $-\infty$.
- Same thing happens in the equivalent form

$$
J(\boldsymbol{\theta}) = -\left\{ \left(\sum_{n=1}^{N} y_n \boldsymbol{x}_n \right)^T \boldsymbol{\theta} - \sum_{n=1}^{N} \log\left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}\right) \right\}.
$$

- When $\boldsymbol{\theta}^T \boldsymbol{x}_n \to \infty$, we have $\log(\infty)$.

Regularization in Logistic Regression

- Example: Two classes: $\mathcal{N}(0, 1)$ and $\mathcal{N}(10, 1)$.
- Run CVX



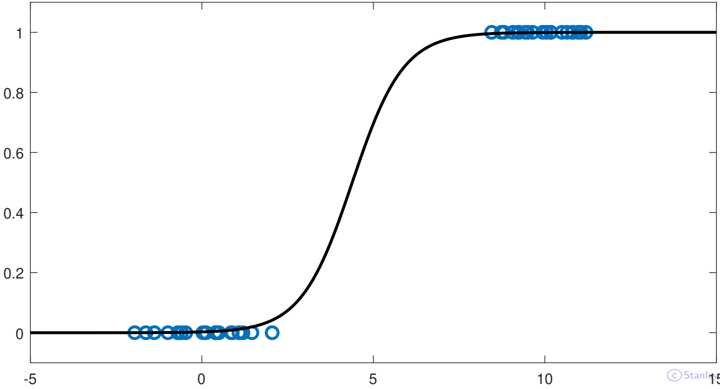
-
- NaN for $y_n = 1$

Regularization in Logistic Regression

- Add a small regularization

$$J(\boldsymbol{\theta}) = -\left\{ \left(\sum_{n=1}^{N} y_n \boldsymbol{x}_n \right)^T \boldsymbol{\theta} - \sum_{n=1}^{N} \log \left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n} \right) \right\} + \lambda \|\boldsymbol{\theta}\|^2.$$

- Re-run the same CVX program



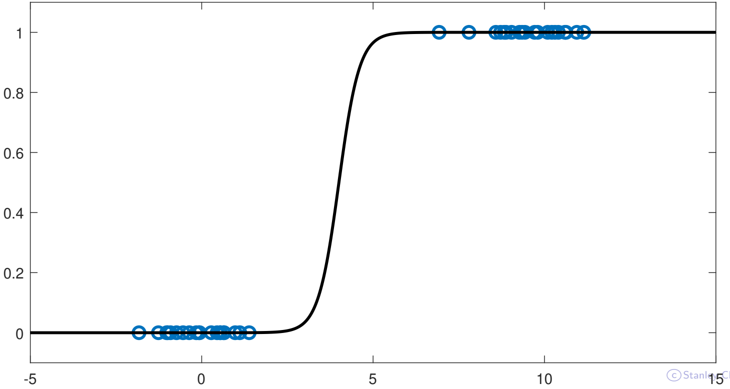
-

Regularization in Logistic Regression

- If you make λ really really small ...

$$J(\boldsymbol{\theta}) = -\left\{ \left(\sum_{n=1}^{N} y_n \boldsymbol{x}_n \right)^T \boldsymbol{\theta} - \sum_{n=1}^{N} \log\left(1 + e^{\boldsymbol{\theta}^T \boldsymbol{x}_n}\right) \right\} + \lambda \|\boldsymbol{\theta}\|^2.$$

- Re-run the same CVX program



-

Try This Online Exercise

- Classify two digits in the MNIST dataset
- `http://ufldl.stanford.edu/tutorial/supervised/LogisticRegression/`

Exercise 1B

Starter code for this exercise is included in the Starter Code GitHub Repo in the ex1/ directory.

In this exercise you will implement the objective function and gradient computations for logistic regression and use your code to learn to classify images of digits from the MNIST dataset as either "0" or "1". Some examples of these digits are shown below:

Each of the digits is is represented by a 28x28 grid of pixel intensities, which we will reformat as a vector $x^{(i)}$ with 28*28 = 784 elements. The label is binary, so $y^{(i)} \in \{0, 1\}$.

You will find starter code for this exercise in the `ex1/ex1b_logreg.m` file. The starter code file performs the following tasks for you:

1. Calls `ex1_load_mnist.m` to load the MNIST training and testing data. In addition to loading the pixel values into a matrix X (so that that j'th pixel of the i'th example is $X_{ji} = x_j^{(i)}$) and the labels into a row-vector y, it will also perform some simple normalizations of the pixel intensities so that they tend to have zero mean and unit variance. Even though the MNIST dataset contains 10 different digits (0-9), in this exercise we will only load the 0 and 1 digits — the ex1_load_mnist function will do this for you.
2. The code will append a row of 1's so that θ_0 will act as an intercept term.
3. The code calls `minFunc` with the `logistic_regression.m` file as objective function. Your job will be to fill in `logistic_regression.m` to return the objective function value and its gradient.
4. After `minFunc` completes, the classification accuracy on the training set and test set will be printed out.

As for the linear regression exercise, you will need to implement `logistic_regression.m` to loop over all

- Exercise: PCA Whitening
- Sparse Coding
- ICA
- RICA
- Exercise: RICA

Self-Taught Learning

- Self-Taught Learning
- Exercise: Self-Taught Learning