Logistic Regression

Kun He (何琨)

Data Mining and Machine Learning Lab
(John Hopcroft Lab)
Huazhong University of Science & Technology

brooklet60@hust.edu.cn

2022年5月

Table of contents

Table of Contents

Basic idea

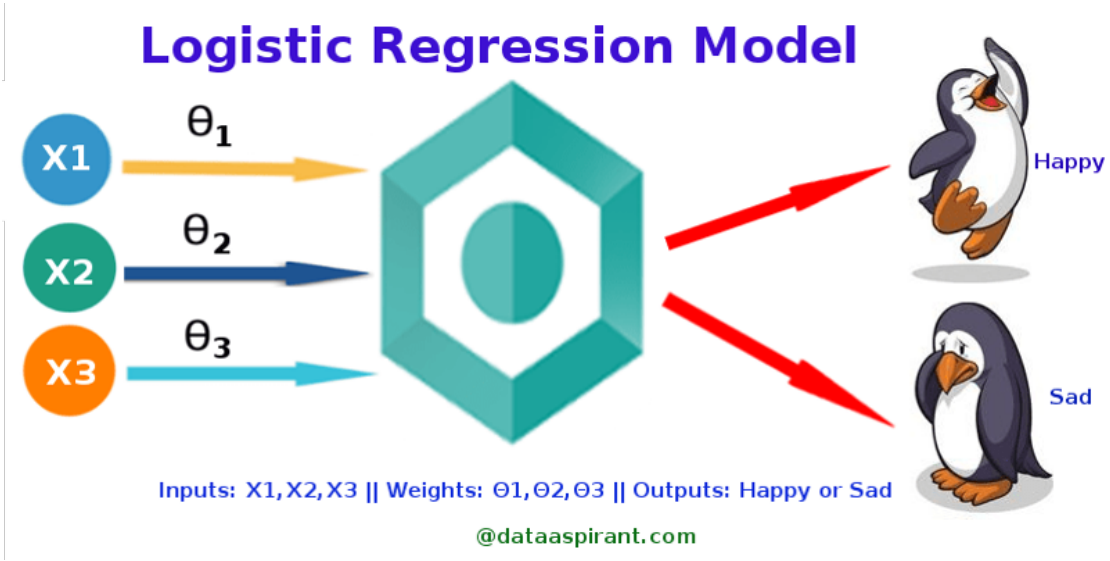


图: Logistic regression is a classic machine learning algorithm used for classification task. As shown in the picture, we first feed the data, the inputs to the model, and the model gives us its classification result.

Basic idea

- Logistic Regression is the discriminative counterpart to the Gaussian Naive Bayes (Naive Bayes for continuous features).
- Machine learning algorithms can be (roughly) categorized into two categories:
  a) **Generative** algorithms, that estimate $P(x_i, y)$ (often they model $P(x_i|y)$ and $P(y)$ separately).
  b) **Discriminative** algorithms, that model $P(y|x_i)$
- The Naive Bayes algorithm is generative. It models $P(x_i|y)$ and makes explicit assumptions on its distribution (e.g. categorical, multinomial, Gaussian, ...). The parameters of this distributions are estimated with MLE or MAP.

Recall:

1) Bayesian Rule:

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

2) Bayesian Classification:

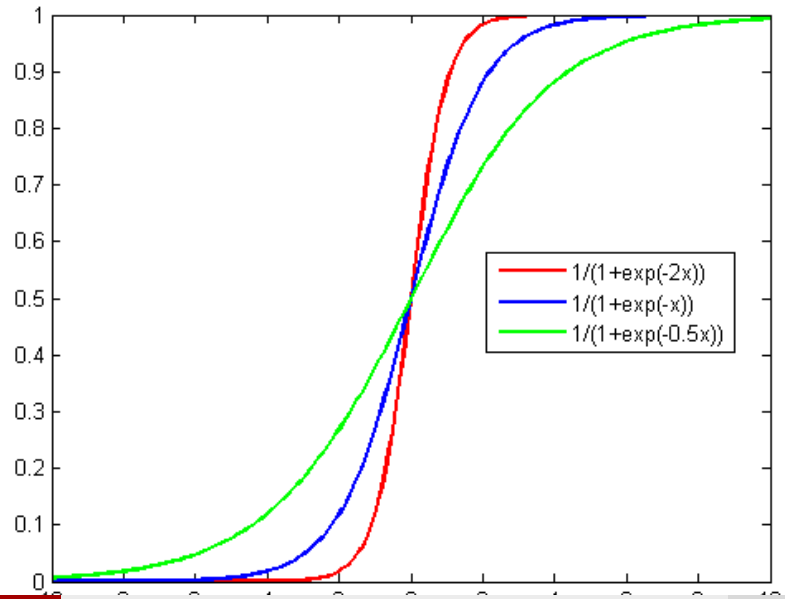
$$\underset{y}{\text{argmax}}\, p(y) \prod^{n} p(x_i|y, \theta) = \underset{y}{\text{argmax}}[log(p(y)) + \sum^{n} log(p(x_i|y, \theta))]$$

Basic idea

- We showed previously that for the Gaussian Naive Bayes $P(y|x_i) = \frac{1}{1+e^{-y(w^T x_i+b)}}$ for $y \in \{+1, -1\}$ for specific vectors w and b that are uniquely determined through the particular choice of $P(x_i|y)$.
- Logistic Regression is often referred to as the discriminative counterpart of Naive Bayes. Here, we model $P(y|x_i)$ and assume that it takes on exactly this form

$$P(y|x_i) = \frac{1}{1 + e^{-y(w^T x_i+b)}}$$

- The logistic (or sigmoid function):



Basic idea

- We make assumptions on $P(x_i|y)$, e.g. it could be Gaussian or Multinomial. Ultimately it doesn't matter, because we estimate the vector w and b directly with MLE or MAP to maximize the conditional likelihood of $\prod_i P(y_i|x_i; w, b)$.
- Throughout this lecture we absorbed the parameter b into w through an additional constant dimension (similar to the Perceptron).

$$P(y|X) = \frac{1}{1 + e^{-y(w^T X)}}$$

MLE for Logistic Regression

- In MLE we choose parameters that **maximize the conditional likelihood**. The conditional data likelihood $P(Y|X, w)$ is the probability of the observed values $Y \in R^n$ in the training data conditioned on the feature values x_i. Note that $X = [x_1, ,, x_i, ,, x_n] \in R^{d \times n}$. We choose the parameters that maximize this function and we assume that the y_i's are independent given the input features x_i and w.

$$P(\mathbf{y} \mid X, \mathbf{w}) = \prod_{i=1}^{n} P(y_i \mid \mathbf{x}_i, \mathbf{w}).$$

Now if we take the log, obtain

$$\log\left(\prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \mathbf{w})\right) = -\sum_{i=1}^{n} \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i})$$

$$\hat{\mathbf{w}}_{MLE} = \underset{\mathbf{w}}{\operatorname{argmax}} -\sum_{i=1}^{n} \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i})$$

$$= \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i})$$

MLE for Logistic Regression

$$\hat{\mathbf{w}}_{MLE} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i})$$

- We need to estimate the parameters w. To find the values of the parameters at minimum, we can try to find solutions for $\nabla_w \sum_i^n log(1 + e^{y_i w^T x_i}) = 0$.
- This equation has no closed form solution, so we will use Gradient Descent on the negative log likelihood

$$\ell(\mathbf{w}) = \sum_{i=1}^{n} \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i})$$

1-D Example

Consider a 1D problem, with positive class marked as pluses and negative as circles.

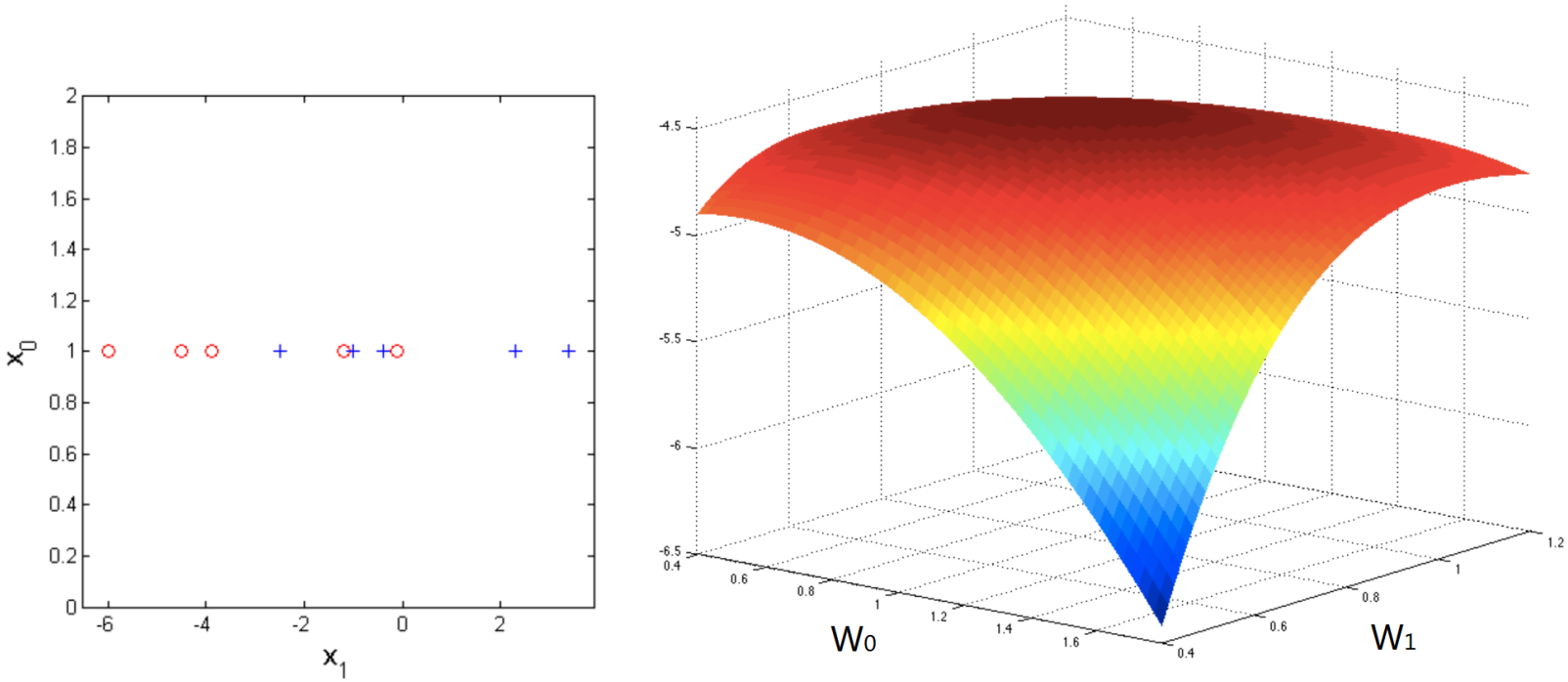


图: (a) Dataset. Vertical axis is the constant feature $X_0 = 1$. (b) Log likelihood.

- This log likelihood function is shown below in the space of the two parameters (w_0, w_1), with the maximum at ($w_0 = 1$ and $w_1 = 0.7$).

1-D Example

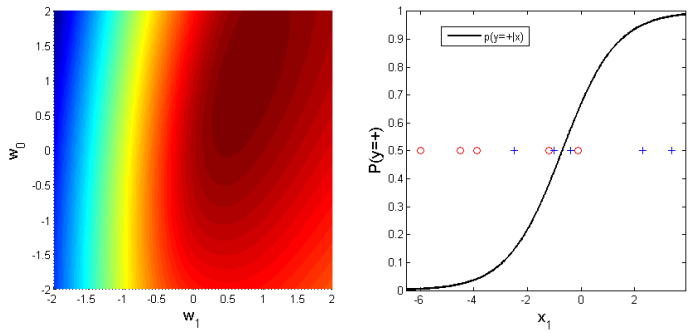
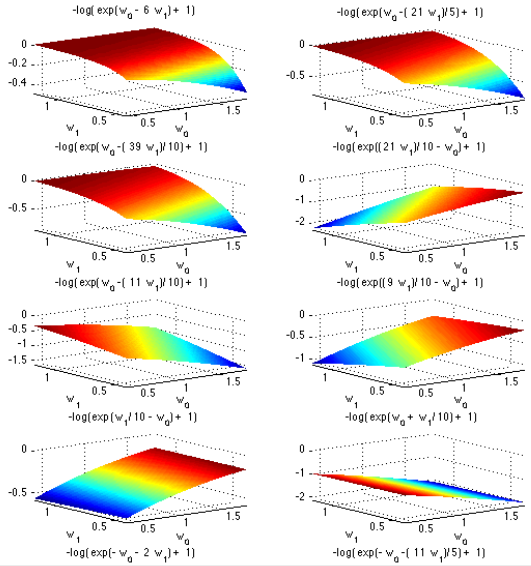


图: (a) Log likelihood heat map for dataset above. (b) The MLE solution.

- Here are the plots of the contributions of each example to the objective above.



2-D Example

Consider a 2D problem, with positive class marked as pluses and negative as circles.

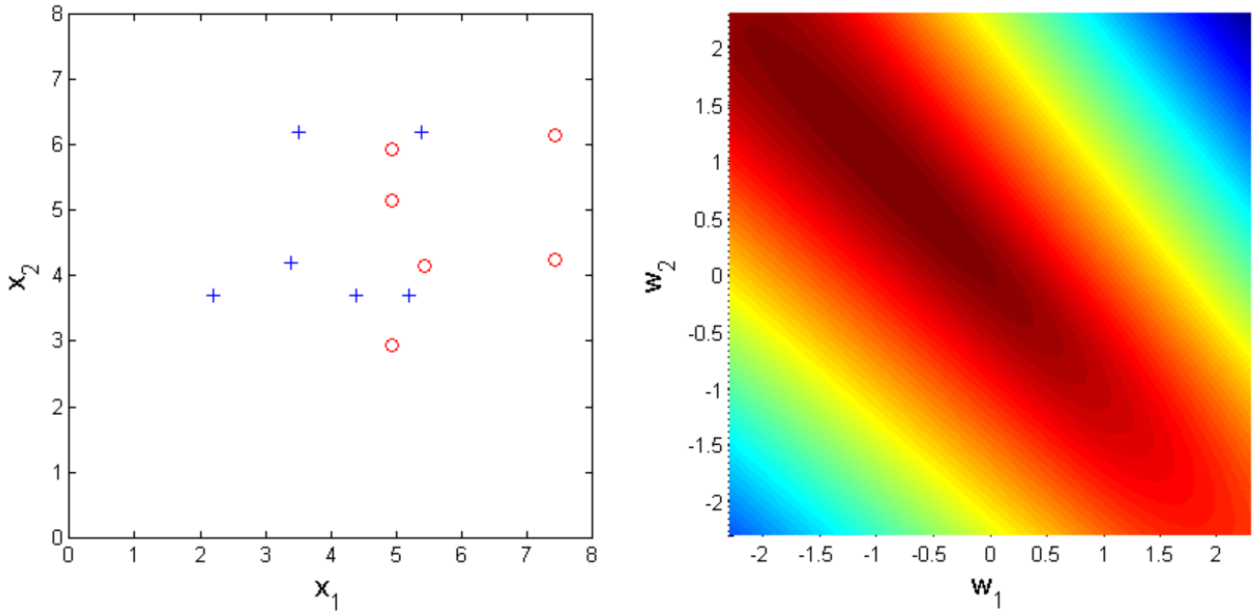


图: (a) Dataset. (b) Log likelihood function with the (w_1, w_2) be maximum at $(-0.81, 0.81)$.

1-D Example

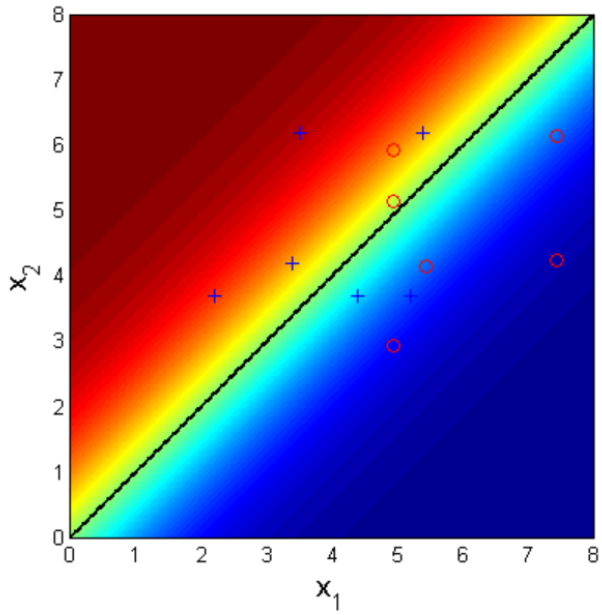


图: The MLE solution is displayed below, where the red color indicates high probability of positive class. The black line shows the decision boundary learned by MLE.

Linear Decision Boundary. Why is the boundary linear? Consider the condition that holds at the boundary:

$$P(Y=1|\mathbf{x},\mathbf{w}) = P(Y=-1|\mathbf{x},\mathbf{w}) \;\rightarrow\; \frac{1}{1+\exp\{-\mathbf{w}^\top\mathbf{x}\}} = \frac{\exp\{-\mathbf{w}^\top\mathbf{x}\}}{1+\exp\{-\mathbf{w}^\top\mathbf{x}\}} \;\rightarrow\; \mathbf{w}^\top\mathbf{x} = 0$$

For the toy problem above, the optimal w is $(-0.81, 0.81)$, so solving

MAP for Logistic Regression

- In the MAP estimate we treat w as a random variable and can specify a prior belief distribution over it. We assume a prior: $w \sim N(0, \sigma^2 I)$. This is the Gaussian approximation for Logistic Regression.
- Our goal in MAP is to find the most likely model parameters given the data, i.e., the parameters that **maximize the posterior**.

$$P(w|D) = \frac{P(D|w)P(w)}{P(D)} \propto P(D|w)P(w)$$

$$\begin{aligned} P(w|D) = P(w|X, y) &\propto P(X, y|w)P(w) \\ &= P(y|X, w)P(X, w)P(w) \propto P(y|X, w)P(w) \\ \hat{w}_{MAP} &= \underset{w}{\text{argmax}}\, log(P(y|X, w)P(w)) \\ &= \underset{w}{\text{argmin}} \sum_{i=1}^{n} log(1 + e^{y_i w^T x_i}) + \lambda w^T w \end{aligned}$$

- Here $\lambda = \frac{1}{2\sigma^2}$. Once again, this function has no closed form solution, but we can use Gradient Descent on the negative log posterior to find the optimal parameters.

$$\ell(\mathbf{w}) = \sum^{n} \log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i}) + \lambda \mathbf{w}^\top \mathbf{w}$$

Summary

- Logistic Regression is the discriminative counterpart to Naive Bayes.
- In Naive Bayes, we first model $P(x|y)$ for each label y, and then obtain the decision boundary that best discriminates between these two distributions.
- In Logistic Regression we do not attempt to model the data distribution $P(x|y)$, instead, we model $P(y|x)$ directly.
- We assume the same probabilistic form $P(y|x_i) = \frac{1}{1+e^{y(w^T x_i+b)}}$, but we do not restrict ourselves in any way by making assumptions about $P(x|y)$ (in fact it can be any member of the Exponential Family).
- This allows logistic regression to be more flexible, but such flexibility also requires more data to avoid overfitting.
- Typically, in scenarios with little data and if the modeling assumption is appropriate, Naive Bayes tends to outperform Logistic Regression. However, as data sets become large logistic regression often outperforms Naive Bayes, which suffers from the fact that the assumptions made on $P(x|y)$ are probably not exactly correct.
- If the assumptions hold exactly, i.e. the data is truly drawn from the distribution that we assumed in Naive Bayes, then Logistic Regression and Naive Bayes converge

Naive Bayes vs. Logistic Regression

Let's consider the differences between the approaches:

Model	Naive Bayes	Logistic Regression
Assumption	$P(\mathbf{X}\|Y)$ is simple	$P(Y\|\mathbf{X})$ is simple
Likelihood	Joint	Conditional
Objective	$\sum_i \log P(y_i, \mathbf{x}_i)$	$\sum_i \log P(y_i \mid \mathbf{x}_i)$
Estimation	Closed Form	Iterative (gradient, etc)
Decision Boundary	Quadratic/Linear (see below)	Linear
When to use	Very little data vs parameters	Enough data vs parameters

Table of Contents

Introduction

Basic idea:

- $f(z) = h_\theta(x) = sigmoid(z) = \frac{1}{1+e^{-z}}$
- Output $= 0$ or 1
- *Hypothesis* $\longrightarrow z = w^T x + b$

We usually attach the sigmoid function to the end of the model, it gives us the probability of the label being 1

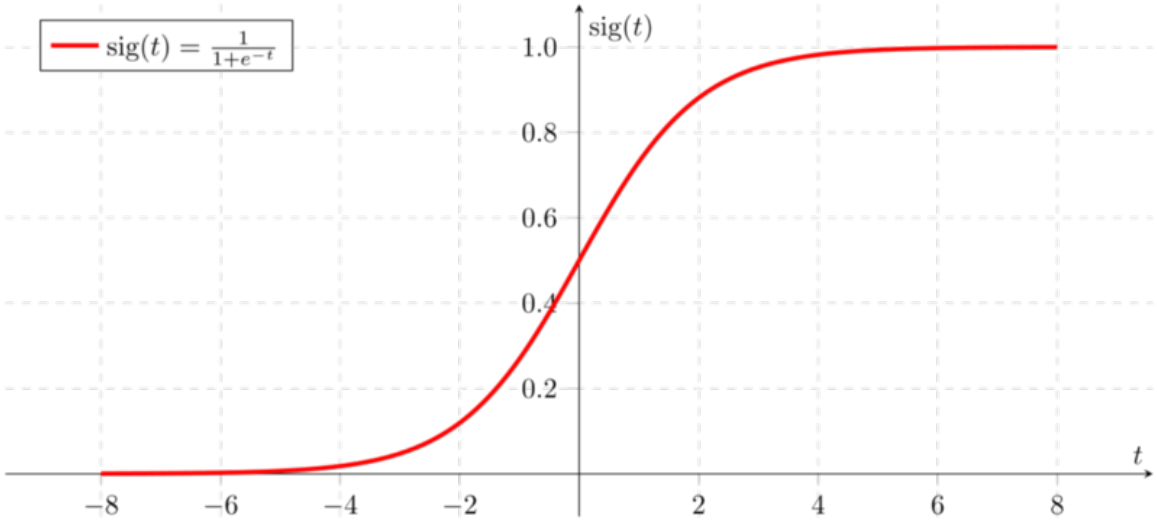


图: If 't' goes to infinity, y(predicted) will become 1 and if 't' goes to negative infinity, y(predicted) will become 0

Loss Function

Predicted Probability:

$$P(y=1|x) = h_\theta(x) = \frac{1}{1+e^{(-\theta^T x)}} = \sigma(\theta^T x) \tag{1}$$

$$P(y=0|x) = 1 - P(y=1|x) = 1 - h_\theta(x) \tag{2}$$

Combined (1).(2):

$$P(y|x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y} \tag{3}$$

y stands for the labels, being 0 or 1

Using maximum likelihood estimation(MLE) according to the m given samples:

$$L(\theta) = \prod_{i=1}^{m} P(y^{(i)}|x^{(i)};\theta) = \prod_{i=1}^{m} (h_\theta(x^{(i)}))y^{(i)} (1-h_\theta(x^{(i)}))^{1-y^{(i)}} \tag{6}$$

$$\longrightarrow \ell(\theta) = \log L(\theta) = \sum_{i=1}^{m} (y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(y^{(i)}))) \tag{7}$$

$$J(\theta) = -\frac{1}{m}\ell(\theta)$$

$J(\theta)$ is the desired loss function

Regularization

Basic idea

- L2

$$J(\theta) = \frac{1}{2m}[\sum_{i=1}^{m}(h_\theta(x^{(i)}))^2 + \lambda \sum_{j=1}^{n} \theta_j^2] \tag{8}$$

- Regularization : prevent the weights from getting too large
- Regularization can avoid overfitting to some extent through the restraint of weights

Regularization

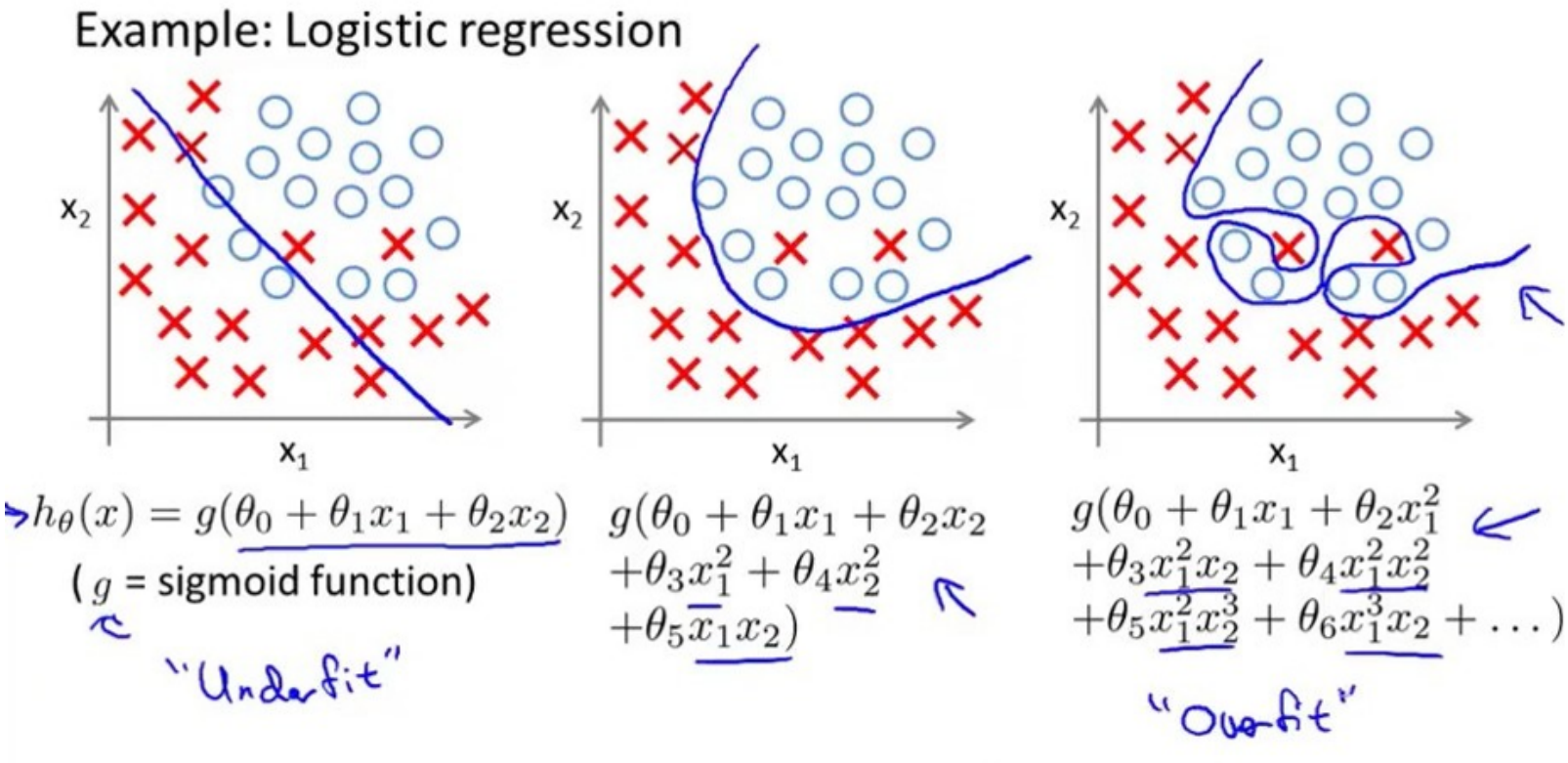


图: Underfitting/overfitting

Regularization

Classification task through Logistic Regression

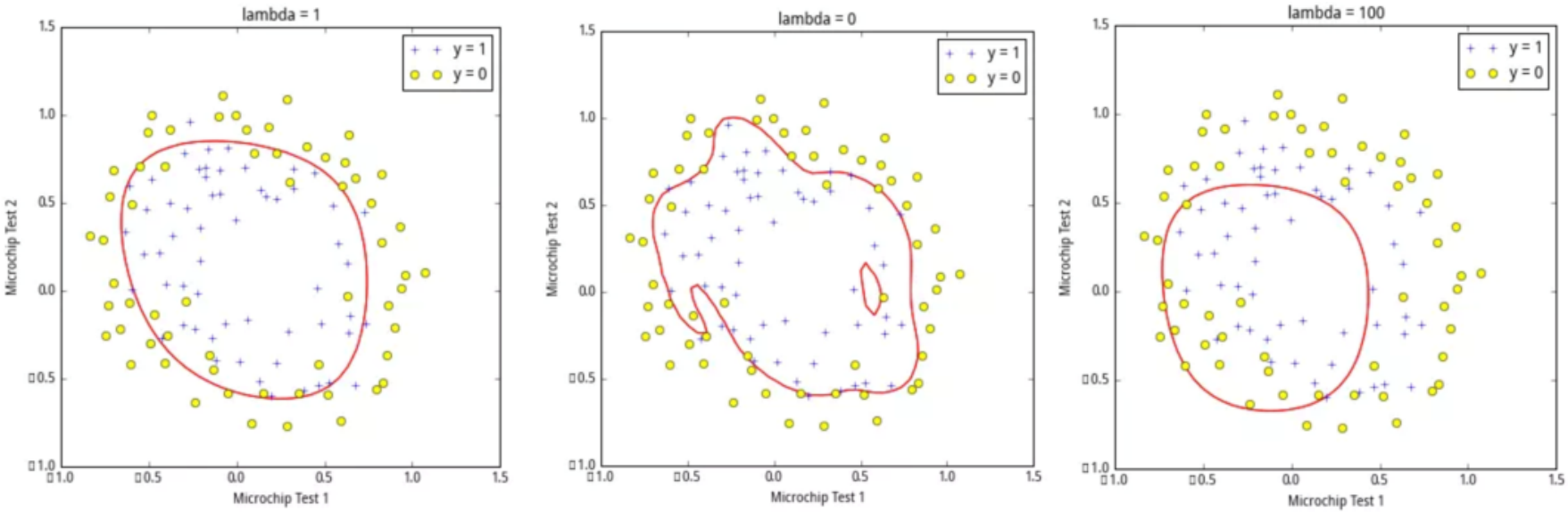
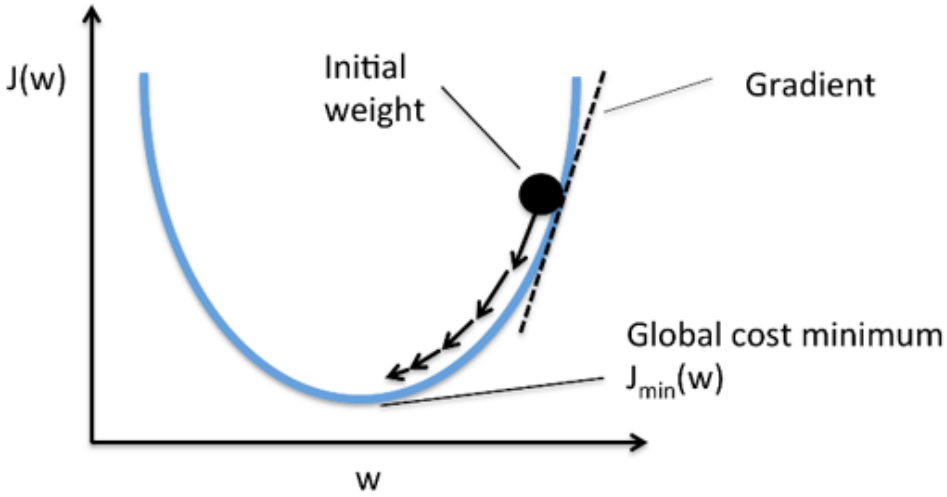
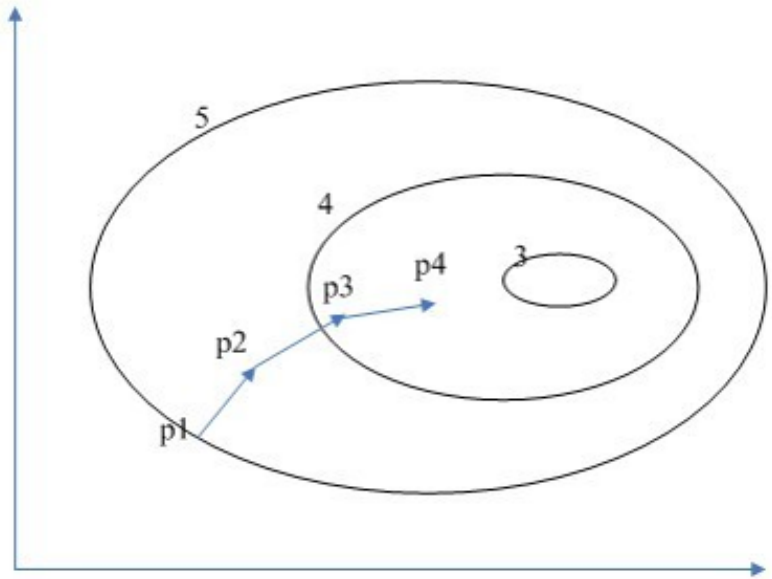


图: Two-class classification when λ has different values

Gradient Descent





Gradient Descent

$$J(\theta) = -\sum_{i=1}^{m}(y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(y^{(i)}))) \tag{9}$$

$$\downarrow$$

Partial derivative:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \sum_i x_j^{(i)}(h_\theta(x^{(i)}) - y^{(i)})$$

$$\downarrow$$

Weights update:

$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial J(\theta_j)}$$

Table of Contents

Comparison

Generative vs. Discriminative

- Machine learning algorithms can be (roughly) categorized into two categories:
- Generative algorithms, that estimate $P(x_i, y)$ (often they model $P(x_i|y)$ and $P(y)$ separately).
- Discriminative algorithms, that model $P(y|x_i)$

1. The Naive Bayes algorithm is generative. $(p(y), p(x|y) \rightarrow p(y|x))$

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

2. Logistic Regression is discriminative. (Gradient descent$\rightarrow w$)

$$P(y|x_i) = \frac{1}{1 + e^{-y(w^T x_i + b)}}$$

The End