SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

01.112 Machine Learning, Spring 2018
Lecture Notes for Week 5

10. Logistic Regression

Last update: Tuesday 20th February, 2018 13:37

1 Logistic Regression

We shall now look at another linear model called *logistic regression* that outputs a probability (or confidence score), i.e., a value between 0 and 1. This new model is similar to regression in that the output is real, and is similar to classification in that the output is bounded. Logistic regression has wide application in practice. For example, we would like to predict the occurrence of diabetes based on an individual's blood pressure, height, weight, age, and other features. Clearly, we cannot predict the occurrence of diabetes with any certainty, but we may be able to predict how likely it is to occur given these features. Thus, a probabilistic output would be appropriate than a binary decision. The closer the output is to 1, the more likely the person would have diabetes.

In logistic regression, the output is given by the formula

$$h(x) = \frac{\exp(\theta \cdot x + \theta_0)}{1 + \exp(\theta \cdot x + \theta_0)}$$

The *logistic function* $\sigma(s) = \frac{\exp(s)}{1+\exp(s)}$ varies continuously between 0 and 1, and its output can be interpreted as a probability for a binary event, e.g., diabetes (+1) or no diabetes (-1). The logistic function is also known as a *sigmoid* function because its shape looks like the letter s (Figure 1).

2 Objective Function

The target output that logistic regression is trying to learn is a probability (e.g., of a patient being at risk of having diabetes) that depends on the input x (e.g., the physiological features of the patient). More formally, the target function is $f(x) = P(y = +1|x)$. However, the data does not provide the value of f explicitly, and only gives samples generated by this probability, e.g., patients who had diabetes (+1) and patients who did not (-1). Thus, to learn from such data, we need to define a proper evaluation function (objective function) that gauges how close a given hypothesis h is to f in terms of these ± 1 examples.

The evaluation function used in logistic regression is based on the *likelihood* of getting output $y = f(x)$ if the target distribution $P(y|x)$ was captured by our hypothesis $h(x)$. This likelihood is given by

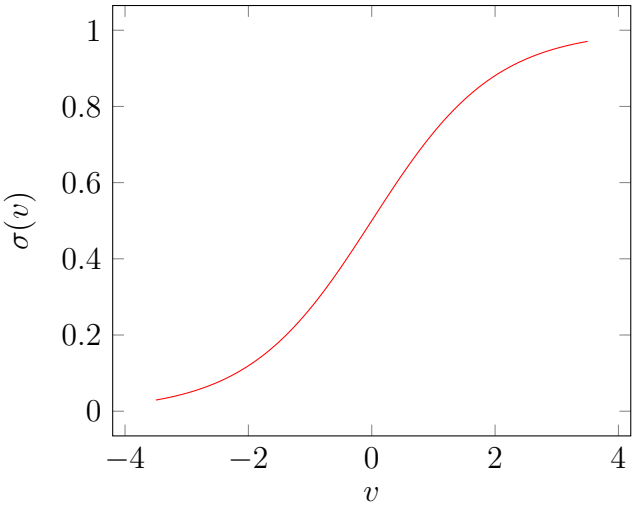


Figure 1: Sigmoid Function

$$P(y|x) = \begin{cases} h(x) & \text{for } y = +1 \\ 1 - h(x) & \text{for } y = -1 \end{cases}$$

Substituting $h(x) = \sigma(\theta \cdot x + \theta_0)$ in the above equation, and using $\sigma(-s) = 1 - \sigma(s)$, we get

$$P(y|x) = \sigma\big(y(\theta \cdot x + \theta_0\big).$$

Since the training examples $(x^1, y^1), \ldots, (x^n, y^n)$ are independently generated, the likelihood of predicting all y^i's from the corresponding x^i's is given by $\prod_{i=1}^{n} P(y^i|x^i)$. Hence, we wish to select the hypothesis h (i.e., find its θ and θ_0) that maximizes this probability (this is known as the method of *maximum likelihood*). Equivalently, we can minimize the following:

$$E(\theta, \theta_0) = -\frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{1}{P(y^i|x^i)}\right) = \frac{1}{n}\sum_{i=1}^{n} \log\Big(1 + \exp(-y^i(\theta \cdot x^i + \theta_0))\Big). \tag{1}$$

Why is this so? And why is this expression computationally more "convenient"? Think about it (and answer it in your homework)!

3 Learning

To train logistic regression (find θ, θ_0 that minimize Expression 1), we can try an approach similar to linear regression by setting $\nabla E(\theta, \theta_0) = 0$. Unfortunately, unlike linear regression, $\nabla E(\theta, \theta_0)$ for logistic regression is amenable to an analytic solution. Although we cannot analytically set the gradient to zero, we can *iteratively* optimize it using *stochastic gradient descent*.

For simplicity, let us drop the θ_0 for the following derivations. Adding θ_0 is simple, which is left as an exercise for you.

Note that when doing stochastic gradient descent, one needs to consider the objective function associated with each instance. The objective associated with the t-th instance is:

$$e^{(t)}(\theta) = \log(1 + \exp(-y^{(t)}(\theta \cdot x^{(t)}))) \tag{2}$$

The gradient of this single example is used for the weight update in the same manner as the batch approach. The gradient is

$$\nabla e^{(t)}(\theta) = \frac{-y^{(t)}x^{(t)}}{1 + \exp(y^{(t)}(\theta \cdot x^{(t)}))} \tag{3}$$

The weight update is

$$\theta \leftarrow \theta - \eta \nabla e^{(t)}(\theta) \tag{4}$$

4 Prediction

Assume now we have already learned our model parameters in the training phase. We now would like to make predictions for the new input x. What shall we do?

Since we only have two possible class labels to consider in the case of binary classification, what we need to do is relatively simple – to check if there is a higher chance of predicting the label as positive or negative for the given input x, under the learned model parameters. To go a little more formally, we need to compute the following ratio:

$$\frac{P(y = +1|x)}{P(y = -1|x)} \tag{5}$$

and compare it against the value 1. If the value is larger than 1, then we should predict the label as +1, otherwise -1.

Since the above ratio is positive, we can instead take the logarithm of it and compare it against 0. Let us take a closer look at this:

$$\log \frac{P(y = +1|x)}{P(y = -1|x)} = \log \exp(\theta \cdot x + \theta_0) = \theta \cdot x + \theta_0 \tag{6}$$

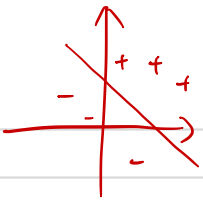
Now, we can see this is a linear function. What does this mean? It shows that the decision boundary for the logistic regression is again a linear function, although we actually had a very different way of learning θ and θ_0. As we will see in class, that the loss function (which will be plotted in class) associated with logistic regression is a surrogate of the hinge loss function used in SVM.

Learning Objectives

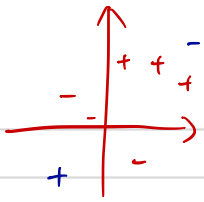
You need to know:

1. What is logistic regression and the model is designed for solving what class of problems.
2. What is the benefit of optimizing the log-likelihood rather than the likelihood of the data.
3. How to perform learning and prediction under logistic regression.

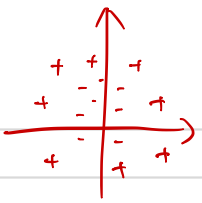
W501 Recap



$\alpha \to$ infinity



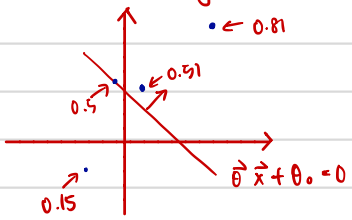
$\alpha = \frac{1}{2}$



kernel

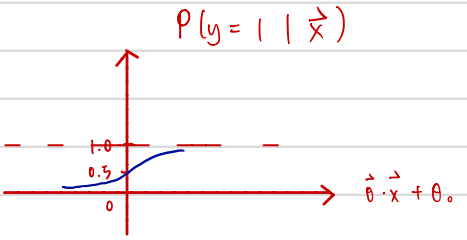
} Support Vector Machines derived from training set

W502 Tests



introduce confidence level into support vector machines

↳ Probability is a functn of distance b/w point & decision boundary



LOGISTIC REGRESSION

Sigma Functn

$\sigma(x) = \frac{1}{1+e^{-x}}$

$\to P(y=+1 \mid \vec{x}) = \frac{1}{1+e^{-(\vec{\theta}\cdot\vec{x}+\theta_0)}}$

$\to P(y=-1 \mid \vec{x}) = 1 - \frac{1}{1+e^{-(\vec{\theta}\cdot\vec{x}+\theta_0)}} = \frac{e^{-(\vec{\theta}\cdot\vec{x}+\theta_0)}}{1+e^{-(\vec{\theta}\cdot\vec{x}+\theta_0)}} = \frac{1}{e^{(\vec{\theta}\cdot\vec{x}+\theta_0)}+1}$

$\to P(y \mid \vec{x}) = \frac{1}{1+e^{-y(\vec{\theta}\cdot\vec{x}+\theta_0)}}$, $y \in \{-1, +1\}$

Another supervised learning problem.

$(\vec{x}^{(1)}, y^{(1)})$
$(\vec{x}^{(2)}, y^{(2)})$
$\vdots$
$(\vec{x}^{(n)}, y^{(n)})$

↓

Assumptn

Data points are ① independent ② identically distributed

$\max \prod_{t=1}^{n} P(y^{(t)} \mid \vec{x}^{(t)}) \to$ max the multiplicatn for t from 1 to n

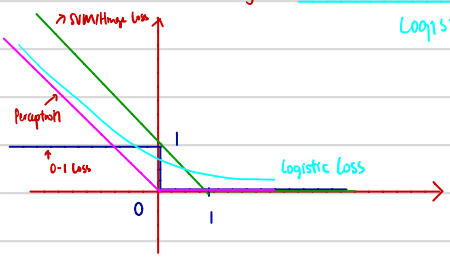
Take log: $-\log \prod_{t=1}^{n} P(y^{(t)} \mid \vec{x}^{(t)}) \to \min$

$= \sum_{t=1}^{n} (-) \log P(y^{(t)} \mid \vec{x}^{(t)})$

$= \sum_{t=1}^{n} \log \frac{1}{P(y^{(t)} \mid \vec{x}^{(t)})}$

$= \sum_{t=1}^{n} \log \left(1+e^{-y^{(t)}\cdot(\vec{\theta}\cdot\vec{x}^{(t)}+\theta_0)}\right) \to \min$

Logistic Loss → This functn is convex



Stochastic Gradient Descent , α: step size

① Initialize $\vec{\theta}$. Initialize $\vec{\theta}_0$ in HW.

② Pick a point k @ random from $\{1 \dots n\}$

$$\vec{\theta} \leftarrow \vec{\theta} - \alpha \left(\frac{-y^{(k)} \cdot \vec{x}^{(t)} \cdot e^{-y^{(t)}(\vec{\theta}\cdot\vec{x})}}{1+e^{-y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}} \right)$$

$$\vec{\theta} \leftarrow \vec{\theta} + \alpha \frac{y^{(t)} \cdot \vec{x}^{(t)}}{1+e^{y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}}$$

③ Repeat ② until you are satisfied.

$$R_n(\theta) = \sum_{t=1}^{n} \ln\left(1+e^{-y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}\right)$$

$$\nabla_\theta R_n(\theta) = \frac{-y^{(t)}(\vec{x}^{(t)}) e^{-y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}}{1+e^{-y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}}$$

$$\frac{\partial \left(-y^{(t)}\vec{x}^{(t)} \cdot \left(1+e^{y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}\right)^{-1}\right)}{\partial \vec{\theta}}$$

$$= -y^{(t)}\vec{x}^{(t)}(-1)\left(1+e^{y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}\right)^{-2} \cdot e^{y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})} \cdot \left(y^{(t)}\cdot\vec{x}^{(t)}\right)$$

$$= \left\| y^{(t)}\cdot\vec{x}^{(t)} \right\|^2 \left(1+e^{y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}\right)^{-2} \cdot e^{y^{(t)}(\vec{\theta}\cdot\vec{x}^{(t)})}.$$

During Prediction Phase

- Check whether $\log \frac{P(y=+1|\vec{x})}{P(y=-1|\vec{x})}$

$$= \log \frac{\frac{1}{1+e^{-(\vec{\theta}\cdot\vec{x}+\theta_0)}}\left(\frac{e^{(\vec{\theta}\cdot\vec{x}+\theta_0)}}{e^{(\vec{\theta}\cdot\vec{x}+\theta_0)}}\right)}{\frac{1}{1+e^{(\vec{\theta}\cdot\vec{x}+\theta_0)}}} = \log e^{(\vec{\theta}\cdot\vec{x}+\theta_0)} = \vec{\theta}\cdot\vec{x}+\theta_0 \geq 0$$

Why not interested in Joint Prob: $P(\vec{x}, y)$ → Generative models

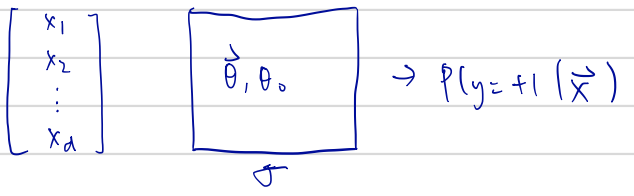
$P(y|\vec{x})$ → Discriminative models

supervised (for huge data set)

	Train	Test
Discriminative	$P(Y\|X)$	$P(y\|X)$
Generative	$P(x,y)$	$P(y\|X)$

↳ unsupervised (for small data set)

Summary



Neural Network (Logistic Regression)