Logistic Regression

Machine Learning Techniques

Dr. Ashish Tendulkar

IIT Madras

Logistic Regression

- Logistic regression is a classifier that can be applied in a single or multi-label classification set ups.
  - Logistic regression is a discriminative classifier.

- It obtains probability of sample belonging to a specific class by computing sigmoid (aka logistic function) of linear combination of features.

- The weight vector for linear combination is learnt via model training.

As usual, we will discuss five components of logistic regression just like any other ML model.

The first component is the training data.

Training Data

Binary classification

- Shape of feature matrix (n, m)
- Shape of label vector $(n,)$

$$D = \{(\mathbf{X}, \mathbf{y})\} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$$

Feature matrix

Label vector

Multiclass and multilabel classification

$$D = \{(\mathbf{X}, \mathbf{Y})\} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{n}$$

Label matrix

- Shape of feature matrix (n, m)
- Shape of label matrix (n, k)
- Shape of label vector $(k,)$

The second component is **model**.

Note that we will be focusing on binary setting in this topic.

The multi-class logistic regression will be covered in exponential family.

Model $h_{\mathbf{w}}(\mathbf{x})$

Weight vector

Non-linear activation function

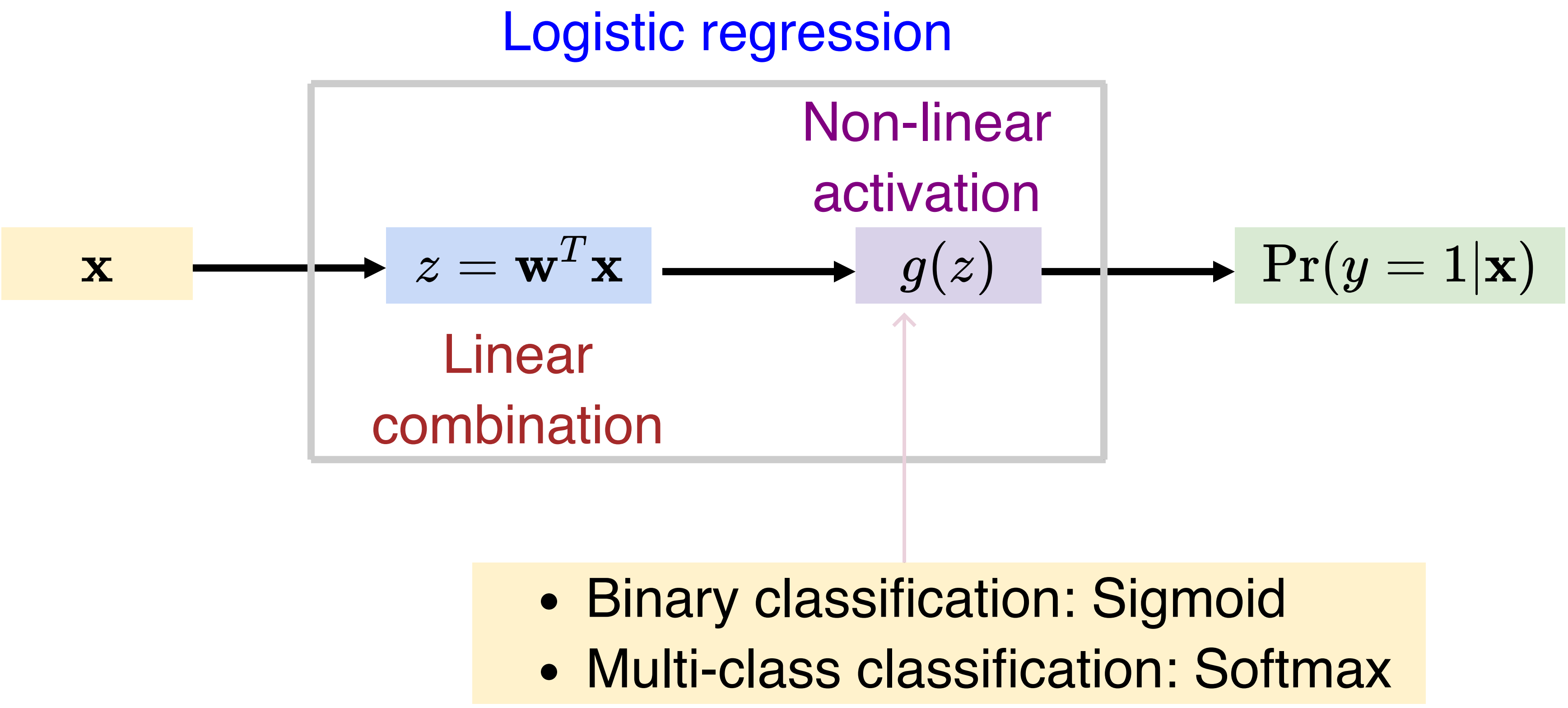
Feature vector

$$\Pr(y = 1|\mathbf{x}) = g(\underline{\mathbf{w}^T\mathbf{x}}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})}$$

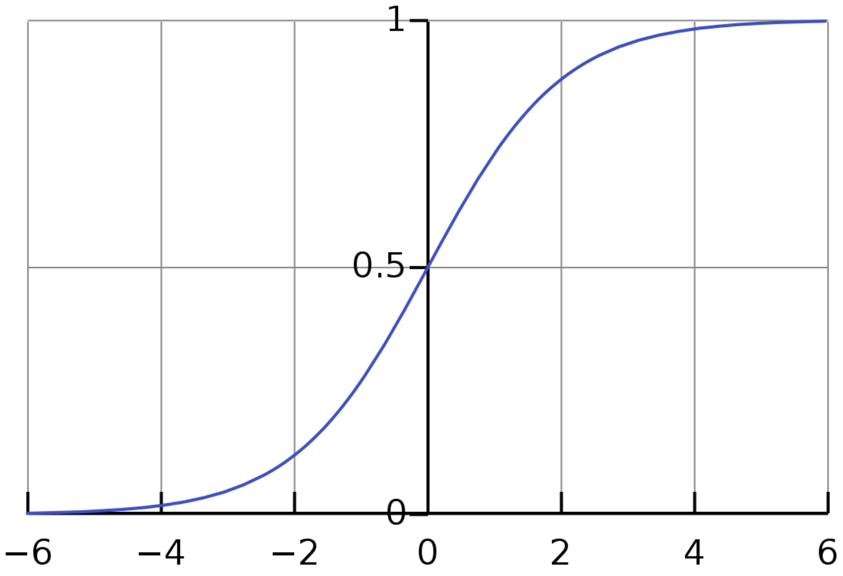
Label

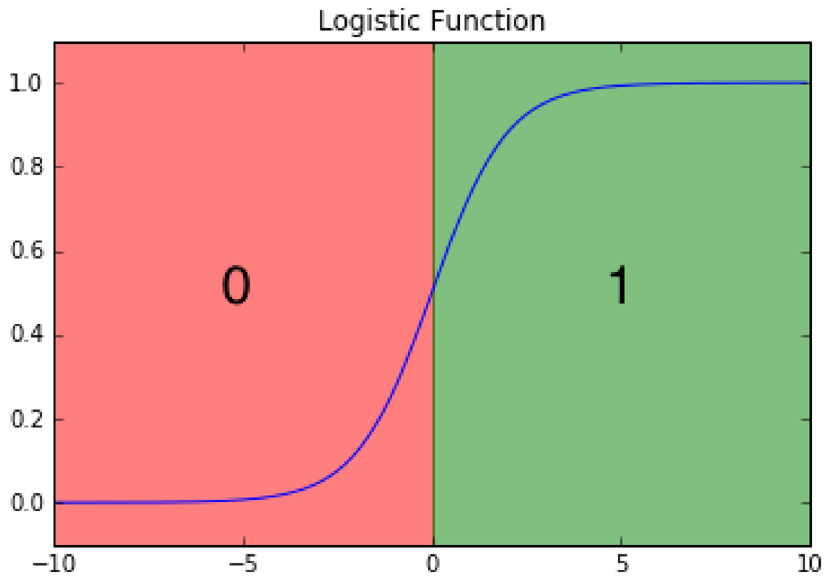
Linear combination of features

parameters = $m + 1$, where m is #features.


Logistic regression
Non-linear
activation
$\mathbf{x}$
$z = \mathbf{w}^T\mathbf{x}$
$g(z)$
$\Pr(y = 1|\mathbf{x})$
Linear
combination
• Binary classification: Sigmoid
• Multi-class classification: Softmax

Let's look at how the logistic (aka sigmoid) function looks like.





- x-axis is a linear combination of feature: $z = \mathbf{w}^T\mathbf{x}$
- y-axis is $g(z)$ - the output of logistic/sigmoid function.

- As $z \to \infty$, $g(z) \to 1.$
- As $z \to -\infty$, $g(z) \to 0$
- For $z = 0, g(z) = 0.5$

Let's look at more general form of logistic regression - with feature transformation.

Model $h_{\mathbf{w}}(\mathbf{x})$ with feature transformation

Weight vector

Non-linear activation function

Feature transformation

$$\Pr(y = 1|\mathbf{x}) = g(\underline{\mathbf{w}^T \phi(\mathbf{x})}) = \frac{1}{1+\exp(-\mathbf{w}^T \phi(\mathbf{x}))}$$

Label

Feature vector

Feature vector

Linear combination of features

The Feature transformation (e.g. polynomial) enables us to fit non-linear decision boundaries.

$$\Pr(y=1|\mathbf{x}) = g(\mathbf{w}^T\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})}$$

$$\Pr(y=1|\mathbf{x}) = g(\mathbf{w}^T\phi(\mathbf{x})) = \frac{1}{1+\exp(-\mathbf{w}^T\phi(\mathbf{x}))}$$

The learning problem here is to estimate the weight vector $\mathbf{w}$ based on the training data by minimizing the loss function through appropriate optimization procedure.

Let's derive the loss function for logistic regression for the case of binary classification problem.

Let's assume that -

$$\Pr(y = 1|\mathbf{x}; \mathbf{w}) = h_{\mathbf{w}}(\mathbf{x})$$
$$\Pr(y = 0|\mathbf{x}; \mathbf{w}) = (1 - h_{\mathbf{w}}(\mathbf{x}))$$

We can rewrite this as

$$\Pr(y|\mathbf{x}; \mathbf{w}) = (h_{\mathbf{w}}(\mathbf{x}))^y (1 - h_{\mathbf{w}}(\mathbf{x}))^{(1-y)}$$

For $y = 0$

$$\begin{aligned} \Pr(y = 0|\mathbf{x}; \mathbf{w}) &= (h_{\mathbf{w}}(\mathbf{x}))^0 (1 - h_{\mathbf{w}}(\mathbf{x}))^{(1-0)} \\ &= (1 - h_{\mathbf{w}}(\mathbf{x})) \end{aligned}$$

For $y = 1$

$$\begin{aligned} \Pr(y = 1|\mathbf{x}; \mathbf{w}) &= (h_{\mathbf{w}}(\mathbf{x}))^1 (1 - h_{\mathbf{w}}(\mathbf{x}))^{(1-1)} \\ &= h_{\mathbf{w}}(\mathbf{x}) \end{aligned}$$

For n independently generated training examples, we can write the likelihood of parameter vector as

$$\begin{aligned} L(\mathbf{w}) &= p(\mathbf{y}|\mathbf{X};\mathbf{w}) \\ &= \prod_{i=1}^{n} p(y^{(i)}|\mathbf{x}^{(i)};\mathbf{w}) \\ &= \prod_{i=1}^{n} \left(h_{\mathbf{w}}(\mathbf{x}^{(i)})\right)^{y^{(i)}} \left(1 - h_{\mathbf{w}}(\mathbf{x}^{(i)})\right)^{1-y^{(i)}} \end{aligned}$$

Taking log on both side as maximizing the log likelihood is easier.

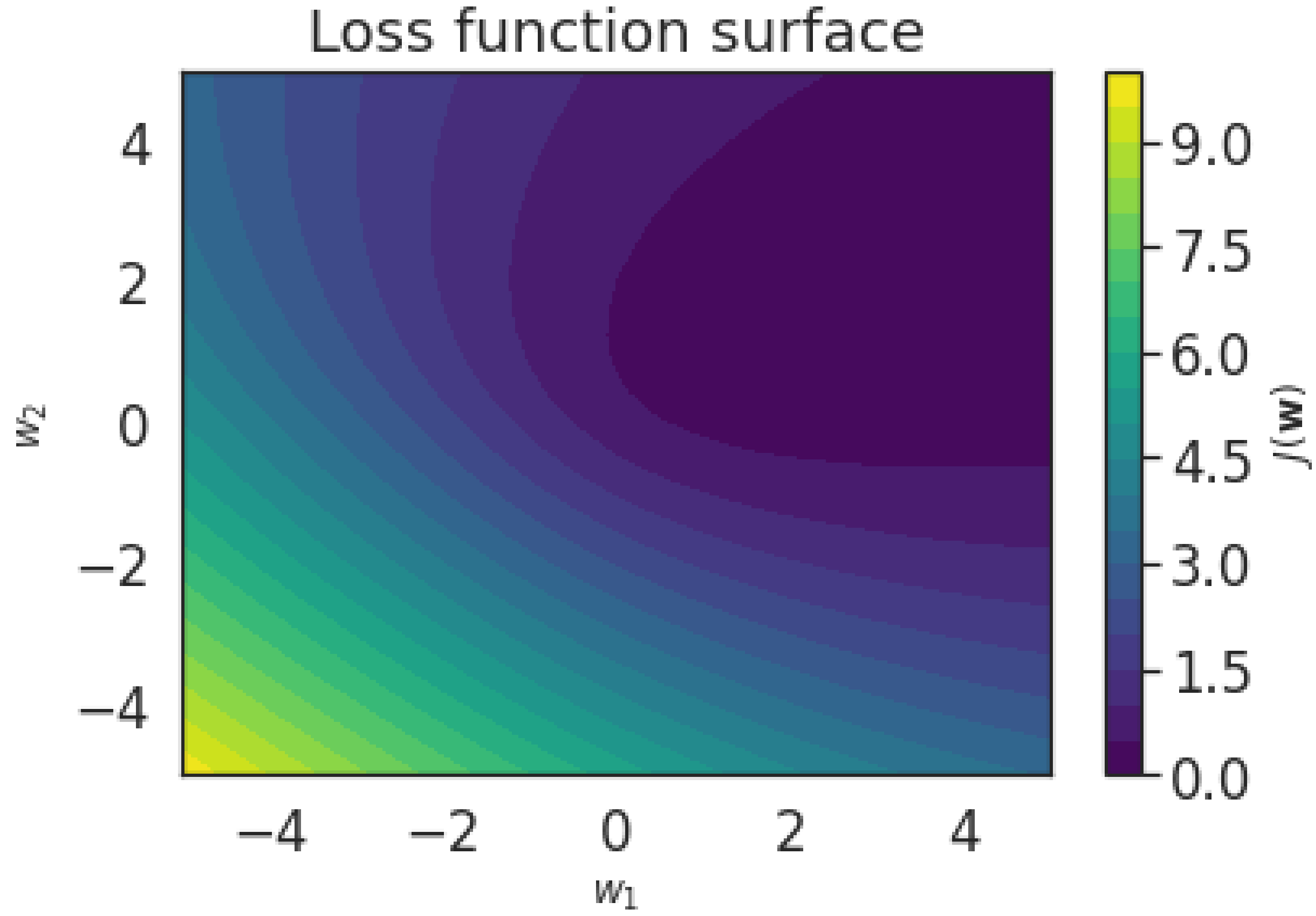
$$\log(L(\mathbf{w})) = \log\left(\prod_{i=1}^{n}\left(h_{\mathbf{w}}(\mathbf{x}^{(i)})\right)^{y^{(i)}}\left(1-h_{\mathbf{w}}(\mathbf{x}^{(i)})\right)^{1-y^{(i)}}\right)$$

$$l(\mathbf{w}) = \sum_{i=1}^{n} y^{(i)}\log\left(h(\mathbf{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-h(\mathbf{x}^{(i)})\right)$$

Our job is to find the parameter vector $\mathbf{w}$ such that the $l(\mathbf{w})$ is maximized.

Equivalently we can minimize the negative log likelihood (NLL) to maintain uniformity with other algorithms:

$$\begin{aligned} J(\mathbf{w}) &= -l(\mathbf{w}) \\ &= -\sum_{i=1}^{n} y^{(i)}\log\left(h(\mathbf{x}^{(i)})\right) + (1-y^{(i)})\log\left(1-h(\mathbf{x}^{(i)})\right) \end{aligned}$$



Loss function is convex.

Binary cross entropy loss

$$J(\mathbf{w}) = -\sum_{i=1}^{n} y^{(i)} \log\left(h(\mathbf{x}^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h(\mathbf{x}^{(i)})\right)$$

Binary cross entropy loss with L2 regularization

$$J(\mathbf{w}) = -\sum_{i=1}^{n} y^{(i)} \log\left(h(\mathbf{x}^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h(\mathbf{x}^{(i)})\right) + \frac{\lambda}{2}||\mathbf{w}||^2$$

Binary cross entropy loss with L1 regularization

$$J(\mathbf{w}) = -\sum_{i=1}^{n} y^{(i)} \log\left(h(\mathbf{x}^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h(\mathbf{x}^{(i)})\right) + \frac{\lambda}{2}||\mathbf{w}||$$

Now that we have derived our loss function, let's focus on optimizing loss function to obtain the weight vector.

$$\mathbf{w} = \arg\min_{\mathbf{w}} J(\mathbf{w})$$

We can use gradient descent for minimizing the loss that is negative log likelihood.

The weight update rule looks as follows:

$$\mathbf{w} := \mathbf{w} - \alpha \frac{\partial}{\partial \mathbf{w}} \left(J(\mathbf{w}) \right)$$

Let's derive partial derivative of sigmoid function: $\frac{\partial}{\partial z} g(z)$

$$\begin{aligned}\frac{\partial}{\partial z}\frac{1}{1+\exp(-z)} &= \frac{1}{(1+\exp(-z))^2}(\exp(-z)) \\ &= \frac{1}{(1+\exp(-z))}\left(1-\frac{1}{(1+\exp(-z))}\right) \\ &= g(z)(1-g(z))\end{aligned}$$

Remember: $g(z) = 1/(1+\exp(-z))$

We will use this in the next derivation.

We need to derive partial derivative of loss function w.r.t. the weight vector.

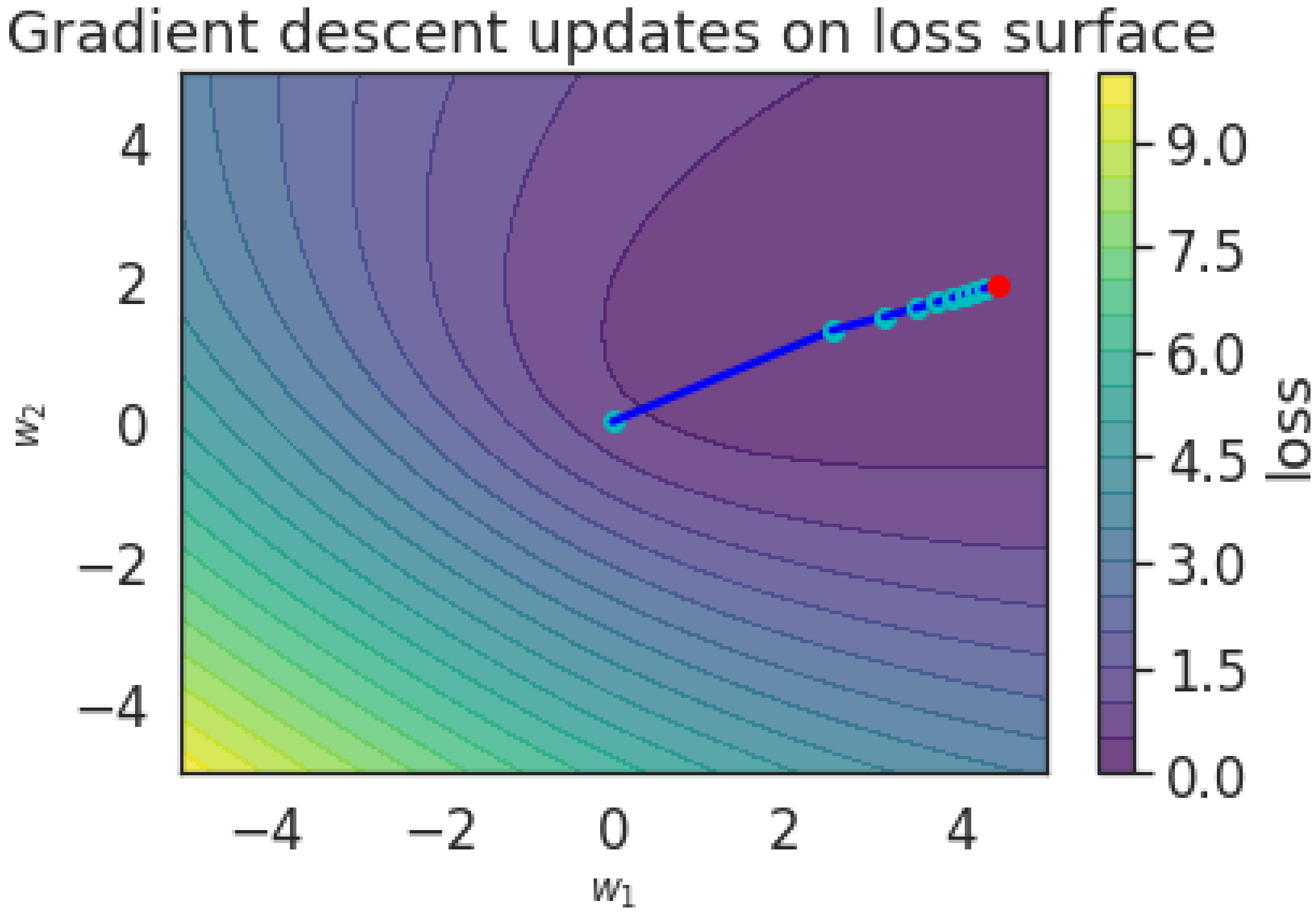
$$
\begin{aligned}
\frac{\partial J(\mathbf{w})}{\partial w_j} &= -\frac{\partial}{\partial w_j}\sum_{i=1}^{n} y^{(i)}\log(h_{\mathbf{w}}(\mathbf{x}^{(i)})) + (1-y^{(i)})\log(1-h_{\mathbf{w}}(\mathbf{x}^{(i)})) \\
&= -\frac{\partial}{\partial w_j}\sum_{i=1}^{n} y^{(i)}\log g(\mathbf{w}^T\mathbf{x}^{(i)}) + (1-y^{(i)})\log\left(1-g(\mathbf{w}^T\mathbf{x}^{(i)})\right) \\
&= -\sum_{i=1}^{n}\left(y^{(i)}\frac{1}{g(\mathbf{w}^T\mathbf{x}^{(i)})} - (1-y)\frac{1}{1-g(\mathbf{w}^T\mathbf{x}^{(i)})}\right)\frac{\partial}{\partial w_j}g(\mathbf{w}^T\mathbf{x}^{(i)}) \\
&= -\sum_{i=1}^{n}\left(y^{(i)}\frac{1}{g(\mathbf{w}^T\mathbf{x}^{(i)})} - (1-y)\frac{1}{1-g(\mathbf{w}^T\mathbf{x}^{(i)})}\right)g(\mathbf{w}^T\mathbf{x}^{(i)})(1-g(\mathbf{w}^T\mathbf{x}^{(i)}))\frac{\partial}{\partial w_j}\mathbf{w}^T\mathbf{x}^{(i)} \\
&= -\sum_{i=1}^{n}\left(y^{(i)}(1-g(\mathbf{w}^T\mathbf{x}^{(i)})) - (1-y)g(\mathbf{w}^T\mathbf{x}^{(i)})\right)x_j^{(i)} \\
&= -\sum_{i=1}^{n}\left(y^{(i)} - g(\mathbf{w}^T\mathbf{x}^{(i)})\right)x_j^{(i)} \\
&= -\sum_{i=1}^{n}\left(y^{(i)} - h_{\mathbf{w}}(\mathbf{x}^{(i)})\right)x_j^{(i)} \\
&= \sum_{i=1}^{n}\left(h_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}
\end{aligned}
$$

The update rule becomes:

$$w_j := w_j - \alpha \left(\sum_{i=1}^{n} \left(h_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} \right)$$

It can be written in vectorized form as follows:

$$\begin{aligned} \mathbf{w} &:= \mathbf{w} - \alpha \left(\mathbf{X}^T \left(h_{\mathbf{w}}(\mathbf{X}) - \mathbf{y} \right) \right) \\ &:= \mathbf{w} - \alpha \left(\mathbf{X}^T \left(g(\mathbf{X}\mathbf{w}) - \mathbf{y} \right) \right) \end{aligned}$$


Gradient descent updates on loss surface
4
2
0
−2
−4
w_2
−4
−2
0
2
4
w_1
9.0
7.5
6.0
4.5
3.0
1.5
0.0
loss

Inference

$$y = \begin{cases} 1, & \text{if } \Pr(y = 1|\mathbf{x}) > 0.5 \\ 0, & \text{otherwise.} \end{cases}$$

Evaluation metrics

- Confusion matrix
- Precision, recall, F1 scores, accuracy
- AUC of ROC and PR curves

Logistic regression: Recap

(1) Data	Features and label (discrete)
(2) Model	Linear combination of features + non-linear activation function
(3) Loss function	Cross entropy loss
(4) Optimization procedure	GD/MBGD/SGD
(5) Evaluation	Precision, recall, F1-score