Logistic Regression

Introduction to Artificial Intelligence with Mathematics
Lecture Notes

Ganguk Hwang

Department of Mathematical Sciences
KAIST

Odds vs Probabilities

$N =$ the total number of trials

$n =$ the number of something happening

$p =$ the probability of happening something

$o =$ the ratio of something happening to something not happening

From

$$p = \frac{n}{N}, \qquad o = \frac{n}{N-n},$$

we have

$$o = \frac{p}{1-p}, \qquad p = \frac{o}{1+o}.$$

Logit Function and Its Inverse

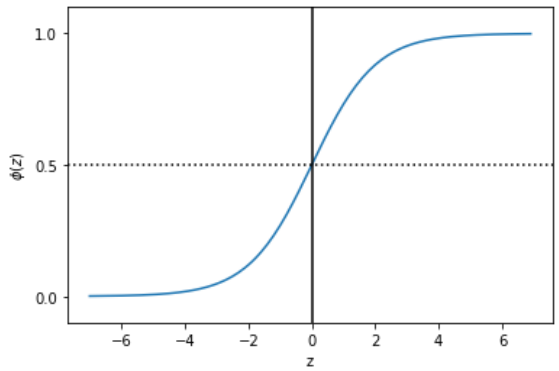
The definition of the logit function from $[0, 1]$ to $(-\infty, \infty)$

$$\text{logit}(p) := \log\left(\frac{p}{1-p}\right)$$

The inverse of the logit function (called logistic function) is

$$\sigma(z) := \text{logit}^{-1}(z) = \frac{1}{1+e^{-z}}, -\infty < z < \infty$$

which is a sigmoid function from $(-\infty, \infty)$ to $[0, 1]$ as shown below.



Logistic Regression Model in a 2-dimensional space

- Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable.
- In the logistic regression model, the log-odds (the logarithm of the odds) for the value labeled 1 is equal to

$$z := \beta_0 + \beta_1 x.$$

- This implies that the odds are expressed by the exponential function

$$\text{odds} = \frac{p}{1-p} = e^z = e^{\beta_0 + \beta_1 x}.$$

- We then have $p = \frac{e^z}{e^z + 1}$ where $z = \beta_0 + \beta_1 x$.

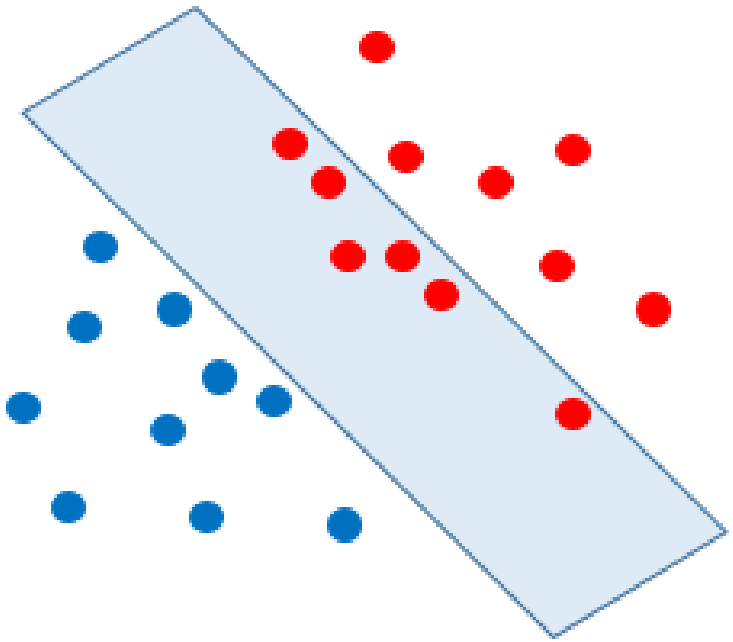
- Conversely, starting from p given above and taking log of the odds, i.e., $\frac{p}{1-p}$, we obtain

$$\log(\frac{p}{1-p}) = z = \beta_0 + \beta_1 x.$$

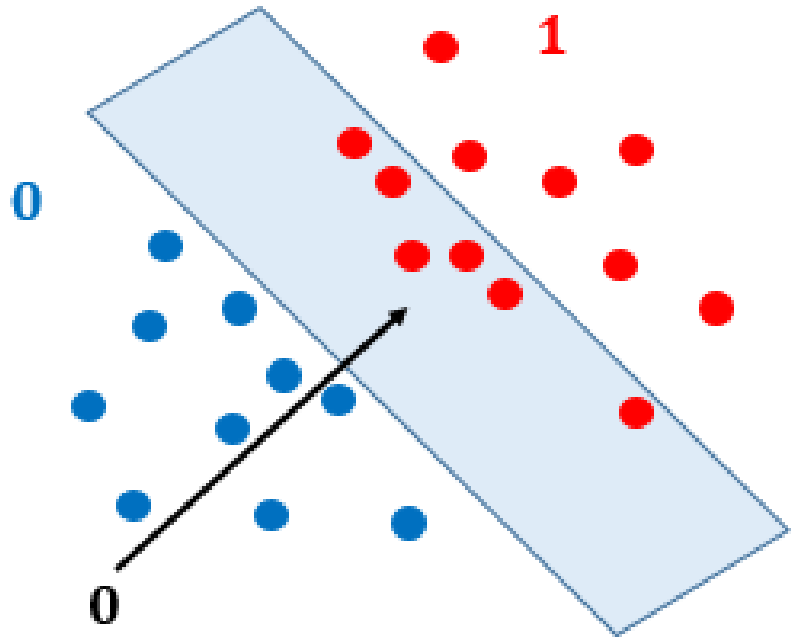
- So even though the probability is not linear, the above arragement leads to a linear regression.

Logistic Regression with Hyperplane

- Recall that logistic regression gives the probability that an input vector belongs to a certain class.
- This means that the input space should be separated into two regions, one for each class, by a hyperplane $\mathbf{w}^\top \mathbf{x} = \mathbf{0}$.
- Here, for $\mathbf{x} = (x_1, \cdots, x_n)$ and $\mathbf{w} = (w_0, w_1, \cdots, w_n)$, $\mathbf{w}^\top \mathbf{x} = w_0 + \sum_{i=1}^{n} w_i x_i$.



- Note that the hyperplane $\mathbf{w}^\top \mathbf{x} = \mathbf{0}$ separates input vectors with label 0 or 1.
- For those input vectors with $\mathbf{w}^\top \mathbf{x}_i > \mathbf{0}$ we have $y_i = 1$.
- For input vectors with $\mathbf{w}^\top \mathbf{x}_i < \mathbf{0}$, we have $y_i = 0$.



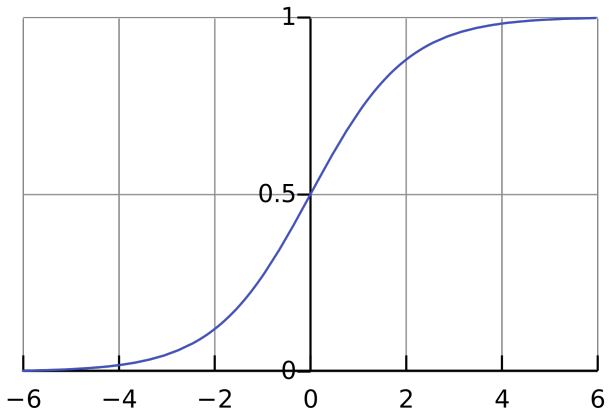
Geometric Observation: Positive class

- Let $\mathbf{x}_i$ lie in the positive region, i.e., $y_i = 1$.
- Then, $\mathbf{w}^\top \mathbf{x}_i$ is positive, lying somewhere in $(0, \infty)$.
- Mathematically, the larger the value $\mathbf{w}^\top \mathbf{x}_i$ is , the greater is the distance between $\mathbf{x}_i$ and the hyperplane.
- Intuitively speaking, the probability that $\mathbf{x}_i$ belongs to the positive class, lies in $(0.5, 1]$.

Geometric Observation: Negative class

- Let $\mathbf{x}_i$ lie in the negative region, i.e., $y_i = 0$.
- Then, $\mathbf{w}^\top \mathbf{x}_i$ is negative, lying somewhere in $(-\infty, 0)$.
- Similarly as before, the probability that $\mathbf{x}_i$ belongs to the negative class, lies in $[0, 0.5]$.

- We need some trick to convert the value $\mathbf{w}^\top \mathbf{x}_i$ into a probability in $[0, 1]$.
- For this purpose, we use $\sigma(z) = \frac{1}{1+\exp^{-z}}$.



The Trick in Logistic Regression

- We need to compute $p = P(y = 1|\mathbf{x}, \mathbf{w})$.
- First consider the odds $\frac{p}{1-p}$ and

$$\log\left(\frac{p}{1-p}\right) = \mathbf{w}^\top \mathbf{x}.$$

- We then have $p = \frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}}}$

Next, we see how the model is trained. We use the maximum likelihood estimation.

Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)\}$ be a given training set.

Consider the likelihood function of $(y_1, y_2, \cdots, y_N)$:

$$\mathcal{L} = p(y_1, y_2, \cdots, y_N | \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N, \mathbf{w}) = \prod_{i=1}^{N} p(y_i | \mathbf{x}_i, \mathbf{w}).$$

Note that

$$p(y_i | \mathbf{x}_i, \mathbf{w}) = p(y_i = 1 | \mathbf{x}_i, \mathbf{w})^{y_i} (1 - p(y_i = 1 | \mathbf{x}_i, \mathbf{w}))^{1-y_i}$$

So

$$\mathcal{L} = \prod_{i=1}^{N} p(y_i = 1 | \mathbf{x}_i, \mathbf{w})^{y_i} (1 - p(y_i = 1 | \mathbf{x}_i, \mathbf{w}))^{1-y_i}$$

We need to find

$$\begin{aligned}
\mathbf{w}^* &= \arg\max_{\mathbf{w}} \log(\mathcal{L}) \\
&= \arg\max_{\mathbf{w}} \sum_{i=1}^{N} \{y_i \log(p(y_i = 1|\mathbf{x}_i, \mathbf{w})) \\
&\qquad\qquad +(1 - y_i) \log(1 - p(y_i = 1|\mathbf{x}_i, \mathbf{w}))\} \\
&= \arg\min_{\mathbf{w}} \left\{ - \sum_{i=1}^{N} y_i \log(p(y_i = 1|\mathbf{x}_i, \mathbf{w})) \right. \\
&\qquad\qquad \left. - \sum_{i=1}^{N} (1 - y_i) \log(1 - p(y_i = 1|\mathbf{x}_i, \mathbf{w})) \right\}.
\end{aligned}$$

Observe that

$$p(y_i = 1|\mathbf{x}_i, \mathbf{w}) = \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}}, p(y_i = 0|\mathbf{x}_i, \mathbf{w}) = \frac{e^{-\mathbf{w}^\top \mathbf{x}_i}}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}}$$

It then follows that

$$\begin{aligned}
-\log(\mathcal{L}) &= -\sum_{i=1}^{N} y_i \log(p(y_i = 1|\mathbf{x}_i, \mathbf{w})) - \sum_{i=1}^{N}(1 - y_i) \log(1 - p(y_i = 1|\mathbf{x}_i, \mathbf{w} \\
&= \sum_{i=1}^{N} y_i \log(1 + e^{-\mathbf{w}^\top \mathbf{x}_i}) \\
&\quad + \sum_{i=1}^{N}(1 - y_i)\mathbf{w}^\top \mathbf{x}_i + \sum_{i=1}^{N}(1 - y_i) \log(1 + e^{-\mathbf{w}^\top \mathbf{x}_i}) \\
&= \sum_{i=1}^{N} \log(1 + e^{-\mathbf{w}^\top \mathbf{x}_i}) + \sum_{i=1}^{N}(1 - y_i)\mathbf{w}^\top \mathbf{x}_i.
\end{aligned}$$

By differentiating $-\log(\mathcal{L})$ w.r.t. $\mathbf{w}$, we get

$$\begin{aligned}\nabla_{\mathbf{w}}(-\log(\mathcal{L})) &= -\sum_{i=1}^{N} \frac{e^{-\mathbf{w}^\top \mathbf{x}_i}\mathbf{x}_i}{1+e^{-\mathbf{w}^\top \mathbf{x}_i}} + \sum_{i=1}^{N}(1-y_i)\mathbf{x}_i \\ &= \sum_{i=1}^{N}\left(\frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}_i}} - y_i\right)\mathbf{x}_i\end{aligned}$$

We now have the gradient descent algorithm:

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \sum_{i=1}^{N}\left(\frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}_i}} - y_i\right)\mathbf{x}_i$$

where α is the learning rate.

Example 1.

As the logistic function is a non-decreasing, smooth function with range $[0, 1]$, logistic regression is usually used to predict an **assign probability** for a classification problem.

We consider a simple binary classification problem in a 2-dimensional space. The objective is to find **a decision boundary**.

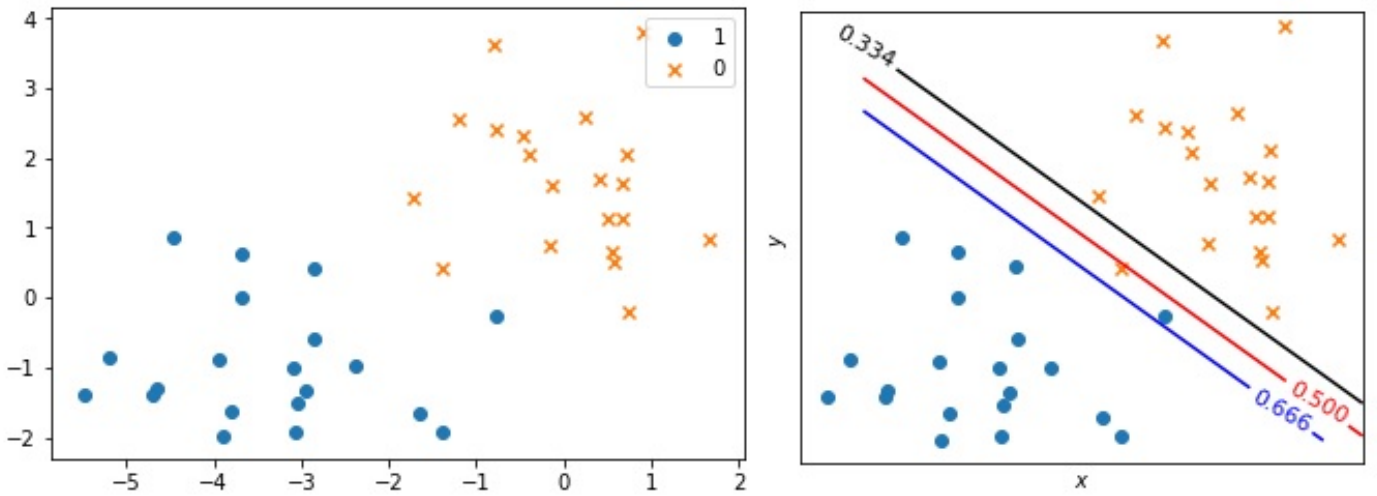


Figure: Result

Here, the points below the blue contour have assign probabilities greater than 0.666 and hence are classified as 1.

Example 2.

We next deal with the well known the iris data classification problem, which is a multiclass classification problem.

If a number of classes is $C > 2$, our logistic regression model can be extended by using a **softmax function**.

$$p_k = P(y = k|\mathbf{x}, \mathbf{W}) = \frac{e^{\mathbf{w}_k^\top \mathbf{x}}}{\sum_{j=1}^{C} e^{\mathbf{w}_j^\top \mathbf{x}}}, \ \mathbf{W} = \begin{bmatrix} \mathbf{w}_1^\top \\ \vdots \\ \mathbf{w}_C^\top \end{bmatrix}$$

Similarly, a given training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)\}$, the objective function is extended as follow.

$$\begin{aligned}
\mathcal{L} &= p(y_1, y_2, \cdots, y_N | \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N, \mathbf{W}) \\
&= \prod_{i=1}^{N} p(y_i | \mathbf{x}_i, \mathbf{W}) \\
&= \prod_{i=1}^{N} \prod_{k=1}^{C} p(y_i = k | \mathbf{x}_i, \mathbf{W})^{y_{i_k}}
\end{aligned}$$

where

$$y_{i_k} = \begin{cases} 1 & \text{if } y_i = k, \\ 0 & \text{if } y_i \neq k. \end{cases}$$

Maximizing $\mathcal{L}$ is equivalent to minimizing

$$J(\mathbf{W}) = -\log \mathcal{L} = -\sum_{i=1}^{N}\sum_{k=1}^{C} y_{i_k} \log(p(y_i = k|\mathbf{x}_i, \mathbf{W}))$$

which is called the **cross entropy loss**.

We use the Gradient Descent Algorithm to get the optimal weight matrix $\mathbf{W}$.

For the Gradient Descent Algorithm we need the following:

for w_{mn}, the n-th element in $\mathbf{w}_m$, and x_{in}, the n-th element in $\mathbf{x}_i$,

$$p_k(\mathbf{x}, \mathbf{W}) = P(y = k|\mathbf{x}, \mathbf{W}) = \frac{e^{\mathbf{w}_k^\top \mathbf{x}}}{\sum_{j=1}^{C} e^{\mathbf{w}_j^\top \mathbf{x}}},$$

$$\frac{\partial}{\partial w_{mn}} p_k(\mathbf{x}_i, \mathbf{W}) = (\delta_{mk} p_k(\mathbf{x}_i, \mathbf{W}) - p_k(\mathbf{x}_i, \mathbf{W}) p_m(\mathbf{x}_i, \mathbf{W})) x_{in},$$

$$\begin{aligned} \frac{\partial}{\partial w_{mn}} J(\mathbf{W}) &= -\sum_{i=1}^{N} \sum_{k=1}^{C} y_{i_k} \frac{1}{p_k(\mathbf{x}_i, \mathbf{W})} \frac{\partial}{\partial w_{mn}} p_k(\mathbf{x}_i, \mathbf{W}) \\ &= -\sum_{i=1}^{N} (y_{i_m} - p_m(\mathbf{x}_i, \mathbf{W})) x_{in}. \end{aligned}$$

Note that $\frac{\partial}{\partial w_{mn}} J(\mathbf{W})$ is the error times the input value.

The following figure is an example of multiclass logistic regression. The iris data has 3 classes with 4 features. To visualize the results, we only use two features, petal length and width.

The figure on the right hand side shows the result of multiclass logistic regression. Similarly as in the previous example, each contour means a boundary of an assign probability 0.5 for each class.