Data Mining and Machine Learning: Fundamental Concepts and Algorithms

dataminingbook.info

Mohammed J. Zaki[1] Wagner Meira Jr.[2]

[1]Department of Computer Science
Rensselaer Polytechnic Institute, Troy, NY, USA

[2]Department of Computer Science
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Chapter 24: Logistic Regression

Binary Logistic Regression

Given a set of d predictor or independent variables $X_1, X_2, \cdots, X_d$, and a *binary* or *Bernoulli* response variable Y that takes on only two values, namely, 0 and 1.

Since there are only two outcomes for Y, its PMF for $\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}$ is:

$$P(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}) = \pi(\tilde{\boldsymbol{x}}) \qquad P(Y = 0 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}) = 1 - \pi(\tilde{\boldsymbol{x}})$$

where $\pi(\tilde{\boldsymbol{x}})$ denotes the probability of $Y = 1$ given $\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}$.

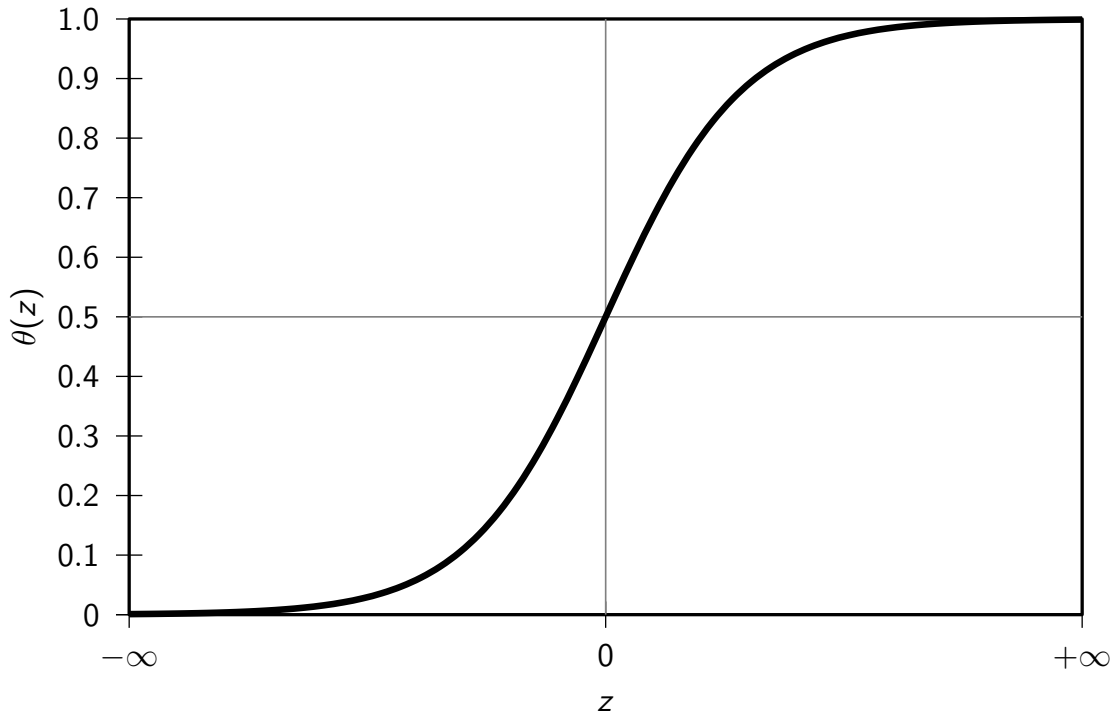
Instead of directly predicting the response value, the goal is to learn the probability, $P(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})$, which is also the expected value of Y given $\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}$.

Since $P(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})$ is a probability, it is not appropriate to directly use the linear regression model.

Logistic Function

Logistic regression comes from the *logistic* function (aka *sigmoid* function) that "squashes" the output to be between 0 and 1 for any scalar input z.

$$\theta(z) = \frac{1}{1+\exp\{-z\}} = \frac{\exp\{z\}}{1+\exp\{z\}}$$



Logistic Function

Example

Consider what happens when z is $-\infty$, $+\infty$ and 0:

$$\theta(-\infty) = \frac{1}{1+\exp\{\infty\}} = \frac{1}{\infty} = 0$$

$$\theta(+\infty) = \frac{1}{1+\exp\{-\infty\}} = \frac{1}{1} = 1$$

$$\theta(0) = \frac{1}{1+\exp\{0\}} = \frac{1}{2} = 0.5$$

$z = 0$ acts as a threshold, since, for $z > 0$, $\theta(z) > 0.5$, and, for $z < 0$, $\theta(z) < 0.5$.

Interpreting $\theta(z)$ as a probability, the larger the z value, the higher the probability.

Another interesting property of the logistic function is that

$$1 - \theta(z) = 1 - \frac{\exp\{z\}}{1+\exp\{z\}} = \frac{1+\exp\{z\}-\exp\{z\}}{1+\exp\{z\}} = \frac{1}{1+\exp\{z\}} = \theta(-z)$$

Binary Logistic Regression

We define the logistic regression model as follows:

$$P(Y=1|\,\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})=\pi(\tilde{\boldsymbol{x}})=\theta(f(\tilde{\boldsymbol{x}}))=\theta(\tilde{\omega}^T\tilde{\boldsymbol{x}})=\frac{\exp\{\tilde{\omega}^T\tilde{\boldsymbol{x}}\}}{1+\exp\{\tilde{\omega}^T\tilde{\boldsymbol{x}}\}}$$

The probability that $Y=1$ is the output of the logistic function for the input $\tilde{\omega}^T\tilde{\boldsymbol{x}}$. The probability for $Y=0$ is given as

$$P(Y=0|\,\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})=1-P(Y=1|\,\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})=\theta(-\tilde{\omega}^T\tilde{\boldsymbol{x}})=\frac{1}{1+\exp\{\tilde{\omega}^T\tilde{\boldsymbol{x}}\}}$$

that is, $1-\theta(z)=\theta(-z)$ for $z=\tilde{\omega}^T\tilde{\boldsymbol{x}}$.

Combining these two cases the full logistic regression model is given as

$$P(Y|\,\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})=\theta(\tilde{\omega}^T\tilde{\boldsymbol{x}})^Y\cdot\theta(-\tilde{\omega}^T\tilde{\boldsymbol{x}})^{1-Y}$$

Since Y is a Bernoulli binary random variable, $P(Y|\,\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})=\theta(\tilde{\omega}^T\tilde{\boldsymbol{x}})$ when $Y=1$ and $P(Y|\,\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})=\theta(-\tilde{\omega}^T\tilde{\boldsymbol{x}})$ when $Y=0$.

Log-Odds Ratio

Define the *odds ratio* for the occurrence of $Y = 1$ as follows:

$$\text{odds}(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}) = \frac{P(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})}{P(Y = 0 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})} = \frac{\theta(\tilde{\omega}^T \tilde{\boldsymbol{x}})}{\theta(-\tilde{\omega}^T \tilde{\boldsymbol{x}})}$$

$$= \frac{\exp\{\tilde{\omega}^T \tilde{\boldsymbol{x}}\}}{1 + \exp\{\tilde{\omega}^T \tilde{\boldsymbol{x}}\}} \cdot \left(1 + \exp\{\tilde{\omega}^T \tilde{\boldsymbol{x}}\}\right)$$

$$= \boxed{\exp\{\tilde{\omega}^T \tilde{\boldsymbol{x}}\}}$$

The logarithm of the odds ratio, called the *log-odds ratio*, is therefore given as:

$$\ln\left(\text{odds}(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})\right) = \ln\left(\frac{P(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})}{1 - P(Y = 1 | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})}\right) = \ln\left(\exp\{\tilde{\omega}^T \tilde{\boldsymbol{x}}\}\right) = \tilde{\omega}^T \tilde{\boldsymbol{x}}$$

$$= \omega_0 \cdot x_0 + \omega_1 \cdot x_1 + \cdots + \omega_d \cdot x_d$$

The log-odds ratio function is also called the *logit* function, defined as

$$\text{logit}(z) = \ln\left(\frac{z}{1 - z}\right)$$

It is the inverse of the logistic function.

Log-Odds Ratio

We can see that

$$\ln\big(\text{odds}(Y=1|\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})\big) = \text{logit}\big(P(Y=1|\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})\big)$$

The logistic regression model is therefore based on the assumption that the log-odds ratio for $Y=1$ given $\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}}$ is a linear function (or a weighted sum) of the independent attributes.

Consider the effect of attribute X_i by fixing the values for all other attributes:

$$\ln(\text{odds}(Y=1|\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}})) = \omega_i \cdot x_i + C$$

$$\implies \text{odds}(Y=1|\tilde{\boldsymbol{X}}=\tilde{\boldsymbol{x}}) = \exp\{\omega_i \cdot x_i + C\} = \exp\{\omega_i \cdot x_i\} \cdot \exp\{C\} \propto \exp\{\omega_i \cdot x_i\}$$

where C is a constant comprising the fixed attributes.

ω_i can be interpreted as the change in the log-odds ratio for $Y=1$ for a unit change in X_i, or, equivalently, the odds ratio for $Y=1$ increases exponentially per unit change in X_i.

Maximum Likelihood Estimation

We will use the maximum likelihood approach to learn the weight vector $\tilde{\boldsymbol{w}}$.

Likelihood is defined as the probability of the observed data given $\tilde{\boldsymbol{w}}$.

$$L(\tilde{\boldsymbol{w}}) = P(Y|\tilde{\boldsymbol{w}}) = \prod_{i=1}^{n} P(y_i|\,\tilde{\boldsymbol{x}}_i) = \prod_{i=1}^{n} \theta(\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}}_i)^{y_i} \cdot \theta(-\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}}_i)^{1-y_i}$$

Instead of maximizing the likelihood, we can maximize the *log-likelihood*, to convert the product into a summation:

$$\ln\left(L(\tilde{\boldsymbol{w}})\right) = \sum_{i=1}^{n} y_i \cdot \ln\left(\theta(\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}}_i)\right) + (1-y_i) \cdot \ln\left(\theta(-\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}}_i)\right)$$

Maximum Likelihood Estimation

The negative of the log-likelihood can also be considered as an error function, the *cross-entropy error function*:

$$E(\tilde{\boldsymbol{w}}) = -\ln(L(\tilde{\boldsymbol{w}})) = \sum_{i=1}^{n} y_i \cdot \ln\left(\frac{1}{\theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i)}\right) + (1 - y_i) \cdot \ln\left(\frac{1}{1 - \theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i)}\right)$$

The task of maximizing the log-likelihood is therefore equivalent to minimizing the cross-entropy error.

Maximum Likelihood Estimation

To obtain the optimal weight vector $\tilde{\boldsymbol{w}}$, we would differentiate the log-likelihood function with respect to $\tilde{\boldsymbol{w}}$, set the result to 0, and then solve for $\tilde{\boldsymbol{w}}$.

However, for the log-likelihood formulation presented,there is no closed form solution to compute the weight vector $\tilde{\boldsymbol{w}}$.

Instead, we use an iterative *gradient ascent* method to compute the optimal value.

The gradient ascent method relies on the gradient of the log-likelihood function, which can be obtained by taking its partial derivative with respect to $\tilde{\boldsymbol{w}}$, as follows:

$$\nabla(\tilde{\boldsymbol{w}}) = \frac{\partial}{\partial \tilde{\boldsymbol{w}}}\Big\{\ln\big(L(\tilde{\boldsymbol{w}})\big)\Big\} = \frac{\partial}{\partial \tilde{\boldsymbol{w}}}\Big\{\sum_{i=1}^{n} y_i \cdot \ln\big(\theta(z_i)\big) + (1-y_i)\cdot \ln\big(\theta(-z_i)\big)\Big\}$$

where $z_i = \tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i$.

After solving the derivative:

$$\nabla(\tilde{\boldsymbol{w}}) = \sum_{i=1}^{n} \big(y_i - \theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i)\big) \cdot \tilde{\boldsymbol{x}}_i$$

Maximum Likelihood Estimation

The gradient ascent method starts at $\tilde{\boldsymbol{w}}^0$.

At each step t, the method moves in the direction of steepest ascent, which is given by the gradient vector.

Given the current $\tilde{\boldsymbol{w}}^t$, the next estimate is:

$$\tilde{\boldsymbol{w}}^{t+1} = \tilde{\boldsymbol{w}}^t + \eta \cdot \nabla(\tilde{\boldsymbol{w}}^t)$$

$\eta > 0$ is the *learning rate*. It should not be too large, otherwise the estimates will vary wildly from one iteration to the next, and it should not be too small, otherwise it will take a long time to converge.

At the optimal value of $\tilde{\boldsymbol{w}}$, the gradient will be zero, i.e., $\nabla(\tilde{\boldsymbol{w}}) = 0$, as desired.

Stochastic Gradient Ascent

The gradient ascent method computes the gradient by considering all the data points, and it is therefore called *batch* gradient ascent.

For large datasets, it is typically much faster to compute the gradient by considering only one (randomly chosen) point at a time, which is called *stochastic gradient ascent* (SGA).

$$\nabla(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{x}}_i) = \left(y_i - \theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i)\right) \cdot \tilde{\boldsymbol{x}}_i$$

Once the model has been trained, we can predict the response for any new augmented test point $\tilde{\boldsymbol{z}}$ as follows:

$$\hat{y} = \begin{cases} 1 & \text{if } \theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{z}}) \geq 0.5 \\ 0 & \text{if } \theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{z}}) < 0.5 \end{cases}$$

Logistic Regression: Stochastic Gradient Ascent

LogisticRegression-SGA $(\boldsymbol{D}, \eta, \epsilon)$**:**

1 **foreach** $\boldsymbol{x}_i \in \boldsymbol{D}$ **do** $\tilde{\boldsymbol{x}}_i^T \leftarrow \begin{pmatrix} 1 & \boldsymbol{x}_i^T \end{pmatrix}$ // map to $\mathbb{R}^{d+1}$

2 $t \leftarrow 0$ // step/iteration counter

3 $\tilde{\boldsymbol{w}}^0 \leftarrow (0, \ldots, 0)^T \in \mathbb{R}^{d+1}$ // initial weight vector

4 **repeat**

5 $\quad \tilde{\boldsymbol{w}} \leftarrow \tilde{\boldsymbol{w}}^t$ // make a copy of $\tilde{\boldsymbol{w}}^t$

6 $\quad$ **foreach** $\tilde{\boldsymbol{x}}_i \in \tilde{\boldsymbol{D}}$ *in random order* **do**

7 $\quad\quad \nabla(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{x}}_i) \leftarrow \left(y_i - \theta(\tilde{\boldsymbol{w}}^T \tilde{\boldsymbol{x}}_i)\right) \cdot \tilde{\boldsymbol{x}}_i$ // gradient at $\tilde{\boldsymbol{x}}_i$

8 $\quad\quad \tilde{\boldsymbol{w}} \leftarrow \tilde{\boldsymbol{w}} + \eta \cdot \nabla(\tilde{\boldsymbol{w}}, \tilde{\boldsymbol{x}}_i)$ // update estimate for $\tilde{\boldsymbol{w}}$

9 $\quad \tilde{\boldsymbol{w}}^{t+1} \leftarrow \tilde{\boldsymbol{w}}$ // update $\tilde{\boldsymbol{w}}^{t+1}$

10 $\quad t \leftarrow t + 1$

11 **until** $\left\| \tilde{\boldsymbol{w}}^t - \tilde{\boldsymbol{w}}^{t-1} \right\| \leq \epsilon$

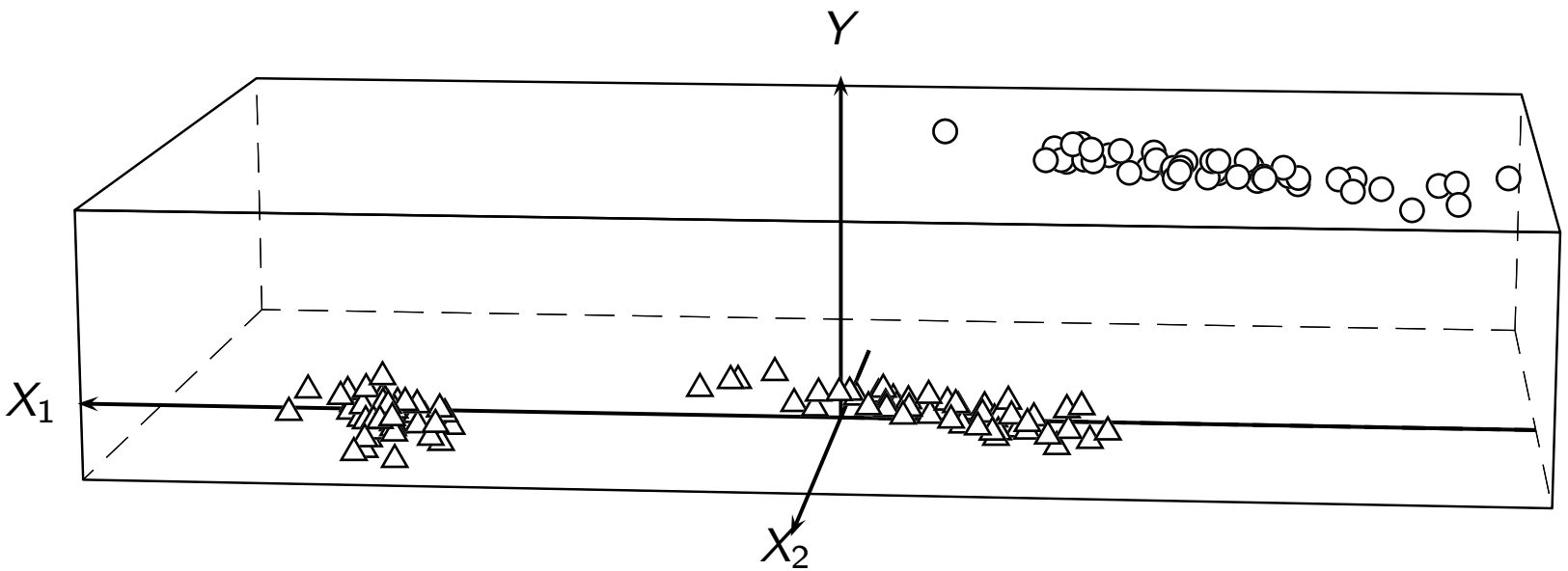
Logistic Regression

Example

Figure shows the Iris principal components data.

X_1 and X_2 are the independent attributes and represent the first two principal components.

Y is the binary response variable and represents the type of Iris flower; $Y = 1$ corresponds to `Iris-virginica`, whereas $Y = 0$ corresponds to the two other Iris types, namely `Iris-setosa` and `Iris-versicolor`.



Linear Regression

Example

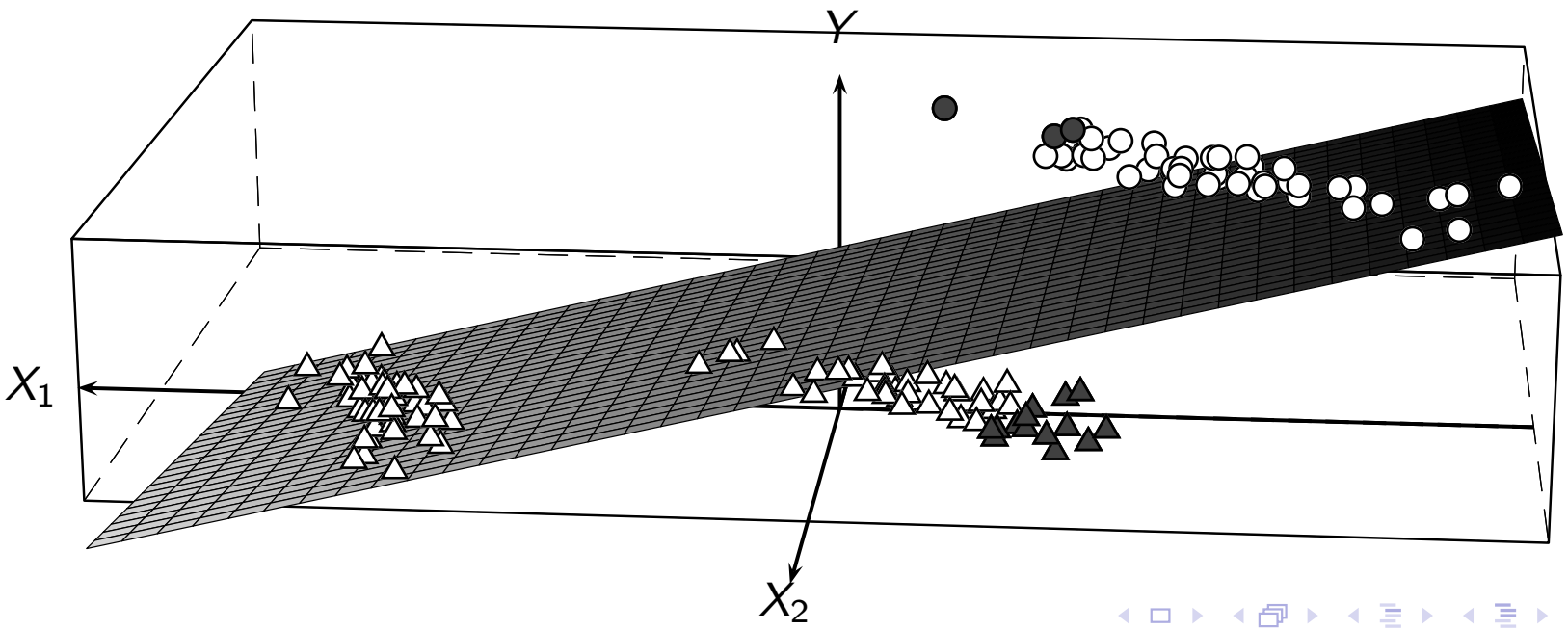
The plane of best fit in linear regression has the weight vector:

$$\tilde{\boldsymbol{w}} = (0.333, -0.167, 0.074)^T$$

$$\hat{y} = f(\tilde{\boldsymbol{x}}) = 0.333 - 0.167 \cdot x_1 + 0.074 \cdot x_2$$

Since Y is binary, we predict $y = 1$ if $f(\tilde{\boldsymbol{x}}) \geq 0.5$, and $y = 0$ otherwise.

Linear regression misclassifies 17 points, achieving 88.7% accuracy.



Logistic Regression

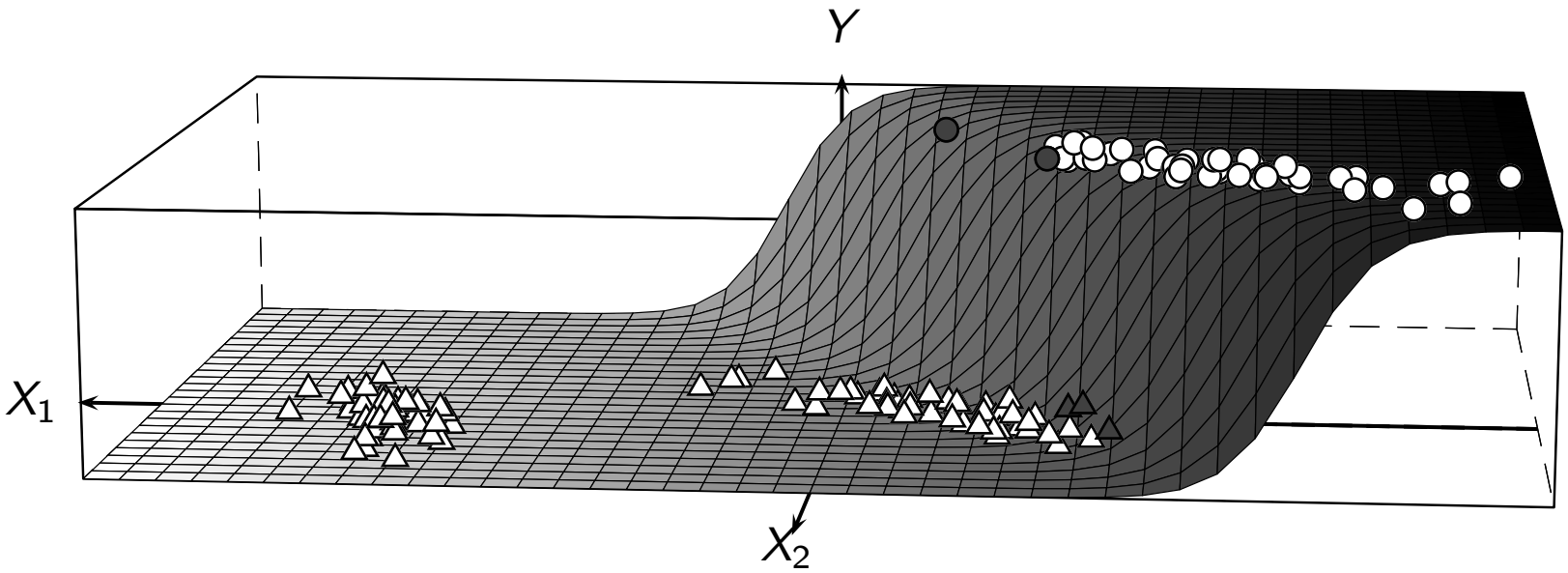
The fitted logistic model is given as

$$\tilde{\boldsymbol{w}} = (w_0, w_1, w_2)^T = (-6.79, -5.07, -3.29)^T$$

$$P(Y = 1|\tilde{\boldsymbol{x}}) = \theta(\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}}) = \frac{1}{1 + \exp\{6.79 + 5.07 \cdot x_1 + 3.29 \cdot x_2\}}$$

Given $\tilde{\boldsymbol{x}}$, if $P(Y = 1|\tilde{\boldsymbol{x}}) \geq 0.5$, then we predict $\hat{y} = 1$, otherwise we predict $\hat{y} = 0$.

Logistic regression misclassifies only 5 points, achieving 96.7% accuracy.



Multiclass Logistic Regression

We now generalize logistic regression to the case when Y can take on K distinct nominal categorical values called *classes*, i.e., $Y \in \{c_1, c_2, \cdots, c_K\}$.

We model Y as a K-dimensional multivariate Bernoulli random variable. Since Y can assume only one of the K values, we use the *one-hot encoding* approach to map each categorical value c_i to the K-dimensional binary vector

$$\boldsymbol{e}_i = (\overbrace{0, \ldots, 0}^{i-1}, 1, \overbrace{0, \ldots, 0}^{K-i})^T$$

whose ith element $e_{ii} = 1$, and all other elements $e_{ij} = 0$, so that $\sum_{j=1}^{K} e_{ij} = 1$.

The probability mass function for $\boldsymbol{Y}$ given $\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}$ is

$$P(\boldsymbol{Y} = \boldsymbol{e}_i | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}) = \pi_i(\tilde{\boldsymbol{x}}), \text{ for } i = 1, 2, \ldots, K$$

There are K unknown parameters, which must satisfy the following constraint:

$$\sum_{i=1}^{K} \pi_i(\tilde{\boldsymbol{x}}) = \sum_{i=1}^{K} P(\boldsymbol{Y} = \boldsymbol{e}_i | \tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}) = 1$$

Multiclass Logistic Regression

Given that only one element of $\boldsymbol{Y}$ is 1, the PMF of $\boldsymbol{Y}$ is:

$$P(\boldsymbol{Y}|\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}) = \prod_{j=1}^{K} (\pi_j(\tilde{\boldsymbol{x}}))^{Y_j}$$

We select c_K as a reference or base class, and consider the log-odds ratio of the other classes w.r.t. c_K.

We assume these log-odd ratios are linear in $\tilde{\boldsymbol{X}}$, but $\tilde{\omega}_i$ is specific to for class c_i:

$$\ln(\text{odds}(\boldsymbol{Y} = \boldsymbol{e}_i|\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})) = \ln\left(\frac{P(\boldsymbol{Y} = \boldsymbol{e}_i|\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})}{P(\boldsymbol{Y} = \boldsymbol{e}_K|\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}})}\right) = \ln\left(\frac{\pi_i(\tilde{\boldsymbol{x}})}{\pi_K(\tilde{\boldsymbol{x}})}\right) = \tilde{\omega}_i^T \tilde{\boldsymbol{x}}$$

$$= \omega_{i0} \cdot x_0 + \omega_{i1} \cdot x_1 + \cdots + \omega_{id} \cdot x_d$$

where $\omega_{i0} = \beta_i$ is the true bias value for class c_i.

Multiclass Logistic Regression

Setting $\tilde{\omega}_K = 0$, we have $\exp\{\tilde{\omega}_K^T \tilde{x}\} = 1$, and thus we can write the full model for multiclass logistic regression as follows:

$$\pi_i(\tilde{x}) = \frac{\exp\{\tilde{\omega}_i^T \tilde{x}\}}{\sum_{j=1}^{K} \exp\{\tilde{\omega}_j^T \tilde{x}\}}, \qquad \text{for all } i = 1, 2, \cdots, K$$

This function is also called the *softmax* function.

When $K = 2$, this formulation yields exactly the same model as in binary logistic regression.

Note that the choice of the reference class is not important, since we can derive the log-odds ratio for any two classes c_i and c_j.

Maximum Likelihood Estimation

To find the K sets of regression weight vectors $\tilde{\boldsymbol{w}}_i$, for $i = 1, 2, \cdots, K$, we use the gradient ascent approach to maximize the log-likelihood function. The likelihood of the data is given as

$$L(\tilde{\boldsymbol{W}}) = P(\boldsymbol{Y}|\tilde{\boldsymbol{W}}) = \prod_{i=1}^{n} P(\boldsymbol{y}_i|\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{x}}_i) = \prod_{i=1}^{n}\prod_{j=1}^{K} (\pi_j(\tilde{\boldsymbol{x}}_i))^{y_{ij}}$$

where $\tilde{\boldsymbol{W}} = \{\tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2, \cdots, \tilde{\boldsymbol{w}}_K\}$ is the set of K weight vectors.

The log-likelihood is then given as:

$$\ln\Big(L(\tilde{\boldsymbol{W}})\Big) = \sum_{i=1}^{n}\sum_{j=1}^{K} y_{ij} \cdot \ln(\pi_j(\tilde{\boldsymbol{x}}_i)) = \sum_{i=1}^{n}\sum_{j=1}^{K} y_{ij} \cdot \ln\left(\frac{\exp\{\tilde{\boldsymbol{w}}_j^T \tilde{\boldsymbol{x}}_i\}}{\sum_{a=1}^{K}\exp\{\tilde{\boldsymbol{w}}_a^T \tilde{\boldsymbol{x}}_i\}}\right)$$

Note that the negative of the log-likelihood function can be regarded as an error function, commonly known as *cross-entropy error*.

For stochastic gradient ascent, we update the weight vectors by considering only one point at a time.

Maximum Likelihood Estimation

The gradient of the log-likelihood function w.r.t. $\tilde{\boldsymbol{w}}_j$ at a given point $\tilde{\boldsymbol{x}}_i$ is:

$$\nabla(\tilde{\boldsymbol{w}}_j, \tilde{\boldsymbol{x}}_i) = \big(y_{ij} - \pi_j(\tilde{\boldsymbol{x}}_i)\big) \cdot \tilde{\boldsymbol{x}}_i$$

which results in the following update rule for the jth weight vector:

$$\tilde{\boldsymbol{w}}_j^{\,t+1} = \tilde{\boldsymbol{w}}_j^{\,t} + \eta \cdot \nabla(\tilde{\boldsymbol{w}}_j^{\,t}, \tilde{\boldsymbol{x}}_i)$$

where $\tilde{\boldsymbol{w}}_j^{\,t}$ denotes the estimate of $\tilde{\boldsymbol{w}}_j$ at step t, and η is the learning rate.

Once the model has been trained, we can predict $\hat{y}$ for any new $\tilde{\boldsymbol{z}}$ as:

$$\hat{y} = \arg\max_{c_i}\big\{\pi_i(\tilde{\boldsymbol{z}})\big\} = \arg\max_{c_i}\left\{\frac{\exp\{\tilde{\boldsymbol{w}}_i^T \tilde{\boldsymbol{z}}\}}{\sum_{j=1}^{K}\exp\{\tilde{\boldsymbol{w}}_j^T \tilde{\boldsymbol{z}}\}}\right\}$$

We evalute the softmax function and the predicted $\hat{y}$ has the highest probability.

Multiclass Logistic Regression Algorithm

LogisticRegression-MultiClass $(\boldsymbol{D}, \eta, \epsilon)$**:**

foreach $(\boldsymbol{x}_i^T, y_i) \in \boldsymbol{D}$ **do**
  $\tilde{\boldsymbol{x}}_i^T \leftarrow \begin{pmatrix} 1 & \boldsymbol{x}_i^T \end{pmatrix}$ // map to $\mathbb{R}^{d+1}$
  $\boldsymbol{y}_i \leftarrow \boldsymbol{e}_j$ if $y_i = c_j$ // map y_i to K-dim Bernoulli vector
$t \leftarrow 0$ // step/iteration counter
foreach $j = 1, 2, \cdots, K$ **do** $\tilde{\boldsymbol{w}}_j^t \leftarrow (0, \ldots, 0)^T \in \mathbb{R}^{d+1}$
repeat
  foreach $j = 1, 2, \cdots, K-1$ **do** $\tilde{\boldsymbol{w}}_j \leftarrow \tilde{\boldsymbol{w}}_j^t$ // copy $\tilde{\boldsymbol{w}}_j^t$
  foreach $\tilde{\boldsymbol{x}}_i \in \tilde{\boldsymbol{D}}$ *in random order* **do**
    foreach $j = 1, 2, \cdots, K-1$ **do**
      $\pi_j(\tilde{\boldsymbol{x}}_i) \leftarrow \exp\left\{\tilde{\boldsymbol{w}}_j^T \tilde{\boldsymbol{x}}_i\right\} / \sum_{a=1}^{K} \exp\left\{\tilde{\boldsymbol{w}}_a^T \tilde{\boldsymbol{x}}_i\right\}$
      $\nabla(\tilde{\boldsymbol{w}}_j, \tilde{\boldsymbol{x}}_i) \leftarrow \left(y_{ij} - \pi_j(\tilde{\boldsymbol{x}}_i)\right) \cdot \tilde{\boldsymbol{x}}_i$ // gradient at $\tilde{\boldsymbol{w}}_j$
      $\tilde{\boldsymbol{w}}_j \leftarrow \tilde{\boldsymbol{w}}_j + \eta \cdot \nabla(\tilde{\boldsymbol{w}}_j, \tilde{\boldsymbol{x}}_i)$ // update estimate for $\tilde{\boldsymbol{w}}_j$
  foreach $j = 1, 2, \cdots, K-1$ **do** $\tilde{\boldsymbol{w}}_j^{t+1} \leftarrow \tilde{\boldsymbol{w}}_j$ // update $\tilde{\boldsymbol{w}}_j^{t+1}$
  $t \leftarrow t + 1$
until $\sum_{j=1}^{K-1} \left\| \tilde{\boldsymbol{w}}_j^t - \tilde{\boldsymbol{w}}_j^{t-1} \right\| \leq \epsilon$

Multiclass Logistic Regression

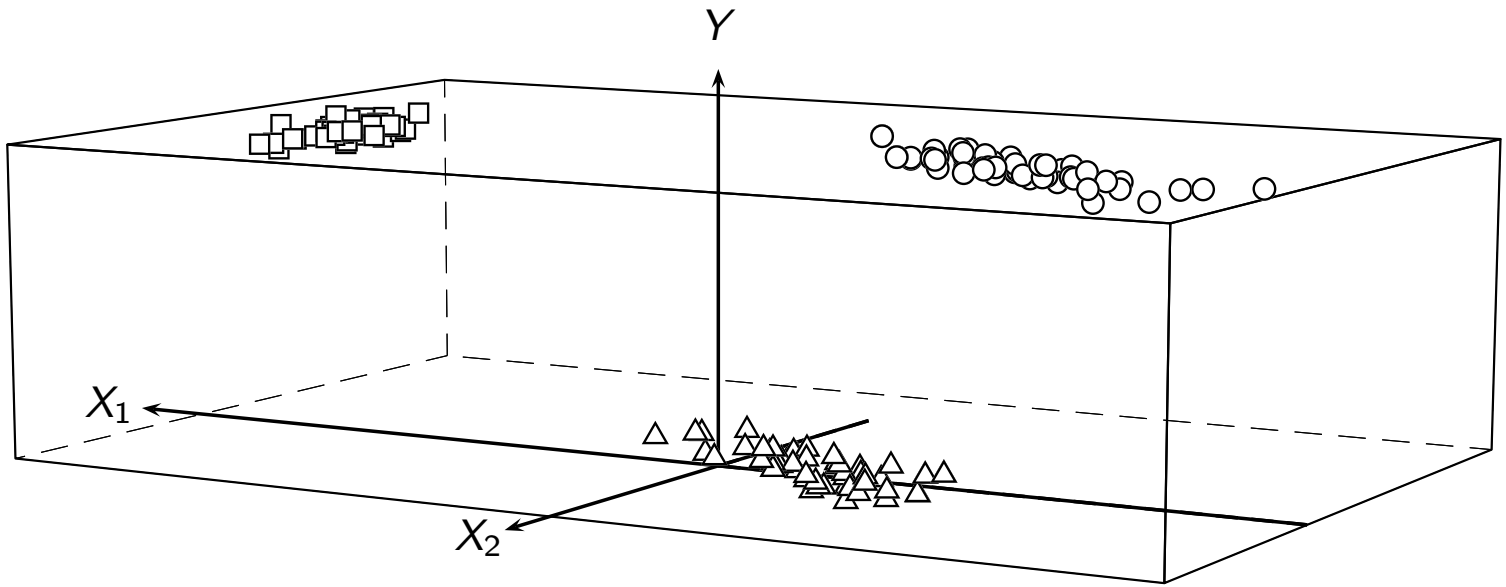
Example

Consider the 2D Iris PCA dataset, $n = 150$.

Y takes on three values: $Y = c_1$: `Iris-setosa` ($\square$), $Y = c_2$: `Iris-versicolor` ($\bigcirc$) and $Y = c_3$: `Iris-virginica` ($\triangle$).

$Y = c_1$ to $\boldsymbol{e}_1 = (1,0,0)^T$, $Y = c_2$ to $\boldsymbol{e}_2 = (0,1,0)^T$ and $Y = c_3$ to $\boldsymbol{e}_3 = (0,0,1)^T$.

All the points actually lie in the (X_1, X_2) plane, but c_1 and c_2 are shown displaced along Y with respect to the base class c_3 purely for illustration purposes.



Multiclass Logistic Regression

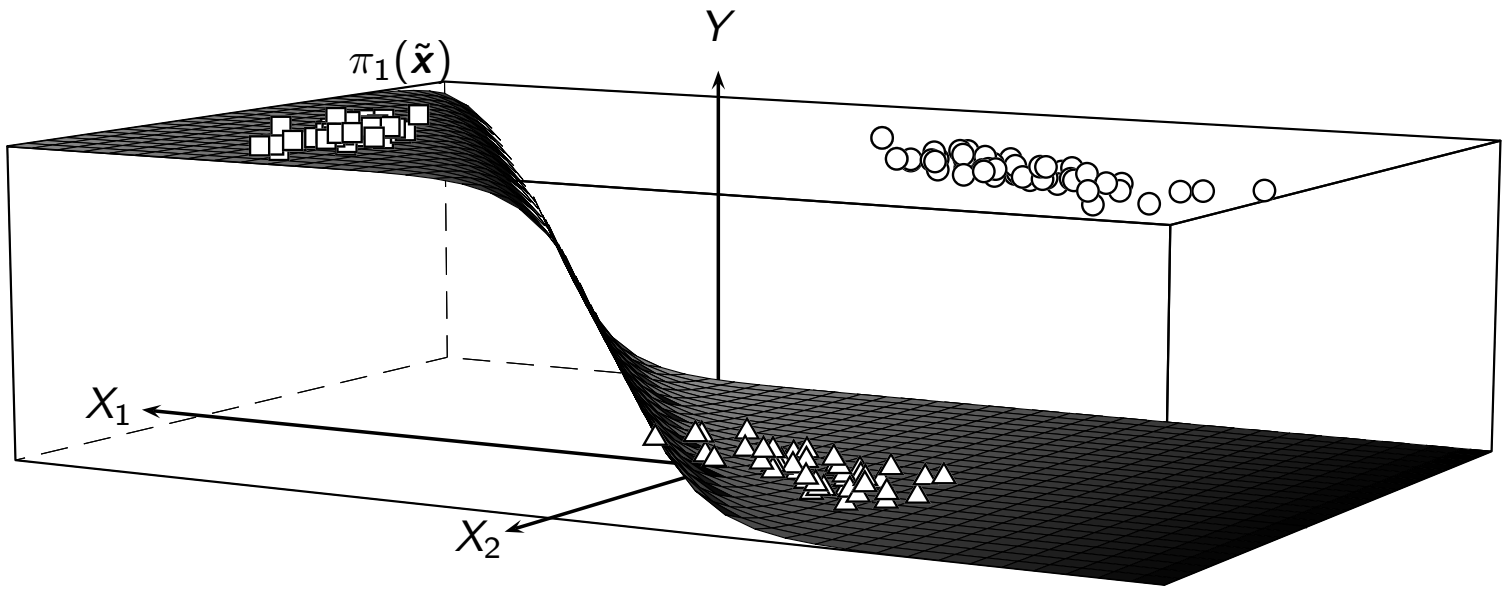
Example

We use $Y = c_3$ as the reference or base class. The fitted model is:

$$\tilde{\boldsymbol{w}}_1 = (-3.52, 3.62, 2.61)^T \quad \tilde{\boldsymbol{w}}_2 = (-6.95, -5.18, -3.40)^T \quad \tilde{\boldsymbol{w}}_3 = (0, 0, 0)^T$$

The decision surface corresponding to c_1 is:

$$\pi_1(\tilde{\boldsymbol{x}}) = \frac{\exp\{\tilde{\boldsymbol{w}}_1^T \tilde{\boldsymbol{x}}\}}{1 + \exp\{\tilde{\boldsymbol{w}}_1^T \tilde{\boldsymbol{x}}\} + \exp\{\tilde{\boldsymbol{w}}_2^T \tilde{\boldsymbol{x}}\}}$$



Multiclass Logistic Regression

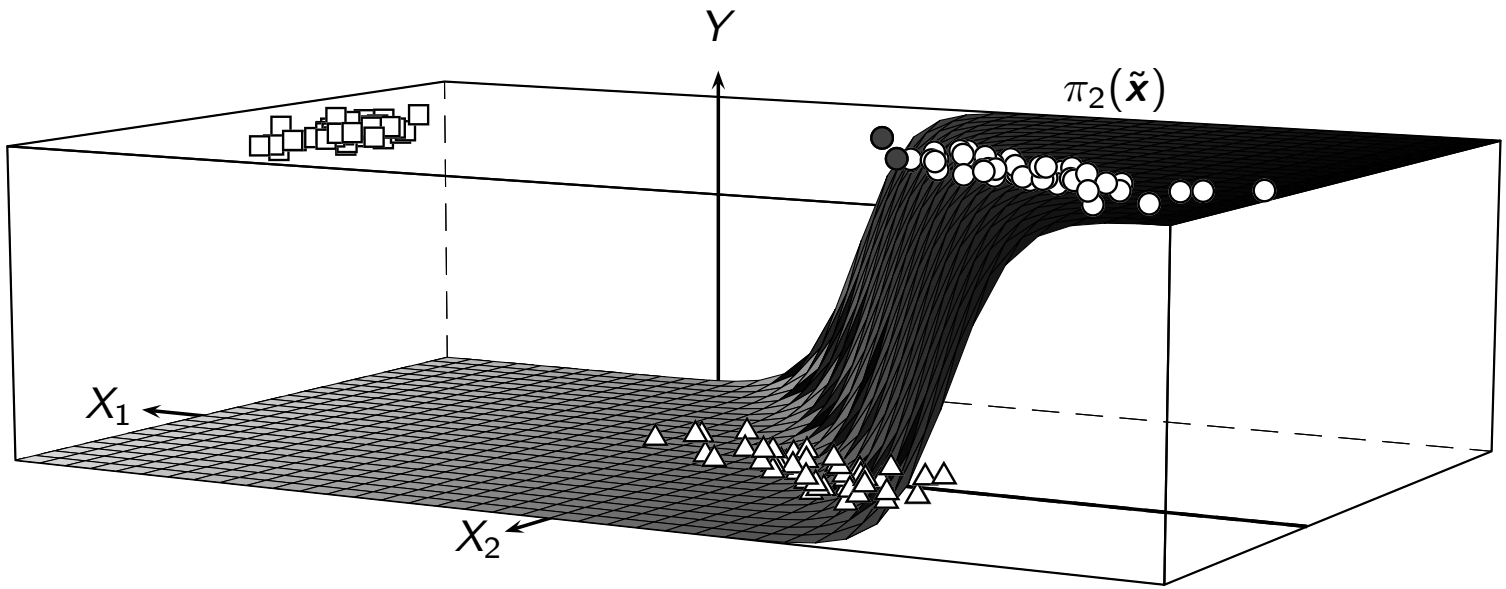
Example

We use $Y = c_3$ as the reference or base class. The fitted model is:

$$\tilde{\boldsymbol{w}}_1 = (-3.52, 3.62, 2.61)^T \quad \tilde{\boldsymbol{w}}_2 = (-6.95, -5.18, -3.40)^T \quad \tilde{\boldsymbol{w}}_3 = (0, 0, 0)^T$$

The decision surface corresponding to c_2 is:

$$\pi_2(\tilde{\boldsymbol{x}}) = \frac{\exp\{\tilde{\boldsymbol{w}}_2^T \tilde{\boldsymbol{x}}\}}{1 + \exp\{\tilde{\boldsymbol{w}}_1^T \tilde{\boldsymbol{x}}\} + \exp\{\tilde{\boldsymbol{w}}_2^T \tilde{\boldsymbol{x}}\}}$$



Multiclass Logistic Regression

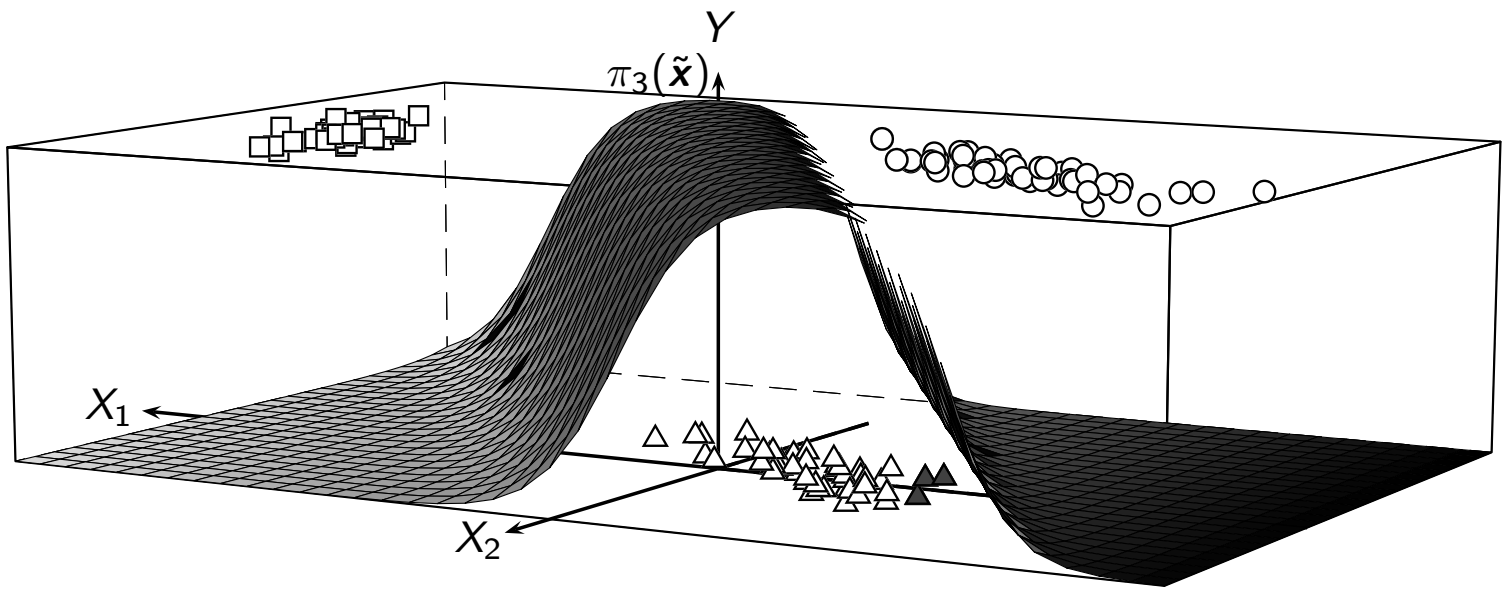
Example

We use $Y = c_3$ as the reference or base class. The fitted model is:

$$\tilde{\boldsymbol{w}}_1 = (-3.52, 3.62, 2.61)^T \quad \tilde{\boldsymbol{w}}_2 = (-6.95, -5.18, -3.40)^T \quad \tilde{\boldsymbol{w}}_3 = (0, 0, 0)^T$$

The decision surface corresponding to c_3 is:

$$\pi_3(\tilde{\boldsymbol{x}}) = \frac{1}{1 + \exp\{\tilde{\boldsymbol{w}}_1^T \tilde{\boldsymbol{x}}\} + \exp\{\tilde{\boldsymbol{w}}_2^T \tilde{\boldsymbol{x}}\}}$$



Multiclass Logistic Regression

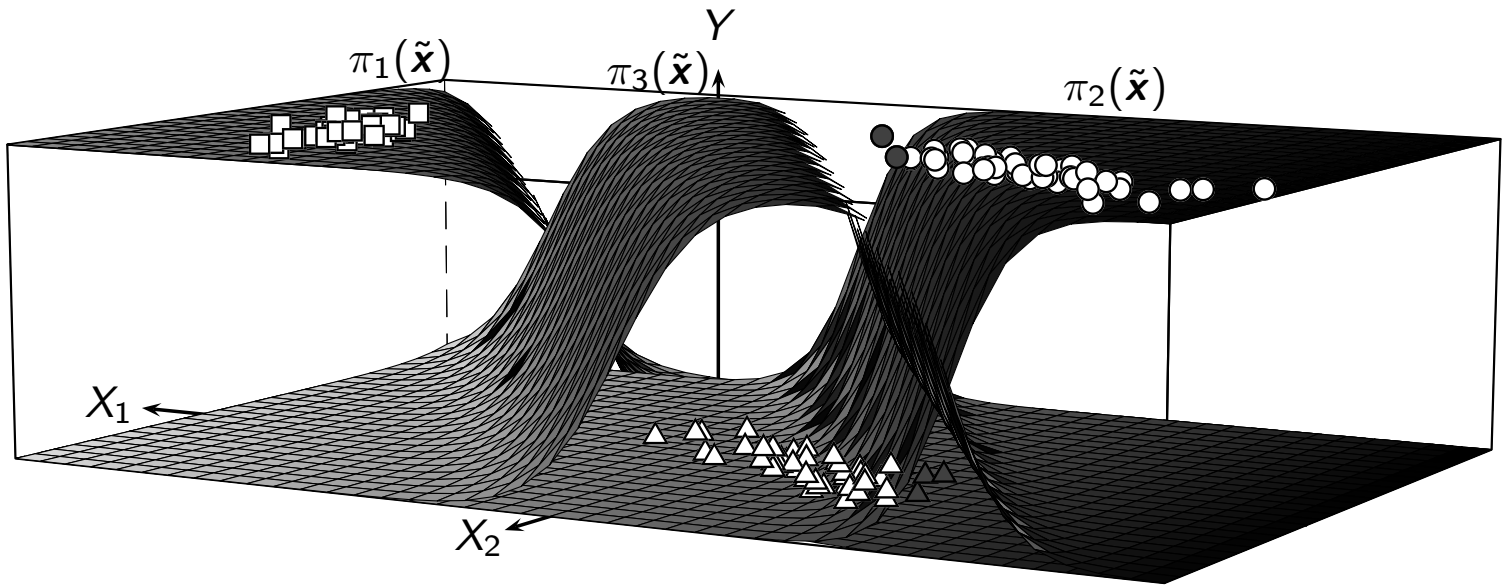
Example

The training set accuracy is 96.7%, since it misclassifies only five points (shown in dark gray).

For example, for the point $\tilde{\boldsymbol{x}} = (1, -0.52, -1.19)^T$, we have:

$$\pi_1(\tilde{\boldsymbol{x}}) = 0 \qquad \pi_2(\tilde{\boldsymbol{x}}) = 0.448 \qquad \pi_3(\tilde{\boldsymbol{x}}) = 0.552$$

$\hat{y} = \arg\max_{c_i}\{\pi_i(\tilde{\boldsymbol{x}})\} = c_3$, whereas the true class is $y = c_2$.



Data Mining and Machine Learning: Fundamental Concepts and Algorithms

dataminingbook.info

Mohammed J. Zaki[1] Wagner Meira Jr.[2]

[1]Department of Computer Science
Rensselaer Polytechnic Institute, Troy, NY, USA

[2]Department of Computer Science
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Chapter 24: Logistic Regression