Chapter 3

Linear and Logistic Regression

We start the chapter with a probabilistic view of linear regression. We first express the likelihood function for the regression and show that its maximization is equal to minimizing the residual sum squares, that is, to minimizing the square loss. We then compute the gradient of the square loss for linear regression, providing means to find the means of inferring parameters of the model from data through gradient descent. We also point to the alternative, closed-form solution. We use a similar approach for logistic regression, again starting with the likelihood function, and finding its gradient to be used in a gradient descent search for parameters that optimize the one-zero loss of the predictor on the training data.

3.1 Maximum Likelihood and Least Squares

Maximum likelihood estimation involves treating the problem as an optimization or search problem. We seek parameters that result in the best fit for the joint probability of the data sample. For a start, consider that we are given a regression (training) data set with pairs of values for dependent and vectors for independent variables, $\{(y_1, \boldsymbol{x}_1), \ldots, (y_N, \boldsymbol{x}_N)\}$. The target variable is continuous, and its estimate is given by

$$\hat{y}(\boldsymbol{x}) = f_{\boldsymbol{\beta}}(\boldsymbol{x}) = \beta_0 + \sum_{m=1}^{M} \beta_m \boldsymbol{x}_m. \tag{3.1}$$

We assume that the target variable y is given by a deterministic function $y(\boldsymbol{x}, \boldsymbol{\beta})$ with additive Gaussian noise, so that

$$y = y(\boldsymbol{x}, \boldsymbol{\beta}) + \epsilon. \tag{3.2}$$

We can then write

$$p(y|\boldsymbol{x}, \boldsymbol{\beta}, \sigma^2) = \mathcal{N}(y|y(\boldsymbol{x}, \boldsymbol{\beta}), \sigma^2), \tag{3.3}$$

which is to say that y is normally distributed around its (true) value provided by the deterministic function and with a variance of σ^2. That is, the distribution of noise in the data is $\epsilon = \mathcal{N}(0, \sigma^2)$. The probability of the approximation error, $e = y - \hat{y}$ is thus

$$p(e) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{e^2}{2\sigma^2}\right) \tag{3.4}$$

We are now ready to assign the probability for the estimated value of the target variable for the i-th data instances in the training data set:

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{\beta}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \hat{y}(\boldsymbol{x}))^2}{2\sigma^2}\right) \tag{3.5}$$

We can now compute the probability of observing the training data by our model, which we exclusively define through a set of parameters $\boldsymbol{\beta}$. We will assume that the data instances in the training data set are independent. We denote this probability with $L(\boldsymbol{beta})$, and write

$$\begin{aligned} L(\boldsymbol{\beta}) &= L(\boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{y}) && (3.6)\\ &= p(\boldsymbol{y}|\boldsymbol{X}; \boldsymbol{\beta}) && (3.7)\\ &= \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i; \boldsymbol{\beta}) && (3.8)\\ &= \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \hat{y}(\boldsymbol{x}_i))^2}{2\sigma^2}\right) && (3.9) \end{aligned}$$

We would like to maximize the probability with which we observe the target values y in the training data by our inferred model. In other words, we would like to find the parameters $\boldsymbol{\beta}$ to maximize the likelihood. For practical reasons, we compute the logarithm of the likelihood, and call it the log likelihood,

$$\begin{aligned} \ell(\boldsymbol{\beta}) &= \log L(\boldsymbol{\beta}) && (3.10)\\ &= \log \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \hat{y}(\boldsymbol{x}_i))^2}{2\sigma^2}\right) && (3.11)\\ &= \sum_{i=1}^{N} \log\left(\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \hat{y}(\boldsymbol{x}_i))^2}{2\sigma^2}\right)\right) && (3.12)\\ &= N \log \frac{1}{\sigma\sqrt{2\pi}} - \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \hat{y}(\boldsymbol{x}_i))^2 . && (3.13) \end{aligned}$$

Considering the result, everything else is constant, and the only term that depends on $\boldsymbol{\beta}$ is the sum of squared approximation errors. To maximize the log-likelihood, we need to minimize

the sum of squared errors! That is, to train the model that maximizes the probability of observing the training data, we need to minimize the following criteria function:

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{N} (y_i - \hat{y}(\boldsymbol{x}_i))^2 \tag{3.14}$$

$$= \sum_{i=1}^{N} (y_i - \beta^\intercal \boldsymbol{x}_i)^2 \tag{3.15}$$

3.2 Gradient Descent for Linear Regression

We are given a set of training data instances for which we would like to infer a linear regression model. We define the model with a set of parameters $\boldsymbol{\beta}$, so that $\hat{y}_i = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 \boldsymbol{x}_{i1} + \ldots + \boldsymbol{\beta}_M \boldsymbol{x}_{iM}$, where $\boldsymbol{\beta}_j$ is the weight of the j-th independent variable and $\boldsymbol{\beta}_0$ is an intercept. Using gradient descent, we can start the search of the optimal set of parameters in some initial point, say, $\boldsymbol{\beta} = \vec{0}$. Our goal is to find $\boldsymbol{\beta}$ that minimize the criteria function $J(\boldsymbol{\beta})$. We do so by changing each $\boldsymbol{\beta}_j$ so that to make $J(\boldsymbol{\beta})$ smaller, that is, in the direction oposite to the partial derivative of the criteria function

$$\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_i - \alpha \frac{\partial}{\partial \boldsymbol{\beta}_i} J(\boldsymbol{\beta}), \tag{3.16}$$

where α is a learning rate whose value will determine the speed to which we proceed to the optimal value of the parameters. If α is too large, there is a chance the procedure will miss optimal point and get out of bounds. When α is too small, the convergence is slow.

Let us now compute the partial derivate of our criteria function for $\boldsymbol{\beta}_i$. Notice that these partial derivatives form a vector, or the gradient of the criteria function. For now, we will compute the gradient taking into account only one data instance from the training set, $(y, \boldsymbol{x})$.

$$\frac{\partial}{\partial \boldsymbol{\beta}_i} J(\boldsymbol{\beta}) = \frac{\partial}{\partial \boldsymbol{\beta}_i} \left(f_\beta(x) - y \right)^2 \tag{3.17}$$

$$= 2(f_\beta(x) - y) \frac{\partial}{\partial \boldsymbol{\beta}_i} \left(f_\beta(\boldsymbol{x}) - y \right) \tag{3.18}$$

$$= 2(f_\beta(x) - y) \frac{\partial}{\partial \boldsymbol{\beta}_i} (\boldsymbol{\beta}_0 x_0 + \ldots + \boldsymbol{\beta}_i \boldsymbol{x}_i + \ldots \boldsymbol{\beta}_n x_n - y) \tag{3.19}$$

$$= 2(f_\beta(x) - y)\boldsymbol{x}_i \tag{3.20}$$

Iterative correction of $\boldsymbol{\beta}_i$ will be therefore

$$\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_i - \frac{\alpha}{N} (f_\beta(\boldsymbol{x}) - y)\boldsymbol{x}_i \tag{3.21}$$

and considering all the data instances in the training set

$$\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_i - \frac{\alpha}{N}\sum_{j=1}^{N}(f_{\boldsymbol{\beta}}(\boldsymbol{x}_j) - y_j)x_{ji} \tag{3.22}$$

In the linear regression or least squares approach described above, the criteria function $J(\boldsymbol{\beta})$ is a quadratic function with a single minimum, so we do not need to fear that the optimization would stop at some local minimum. However, as we have already written above, at large values of α the gradient descent may overshoot and miss the minimum and start moving further and further away from it. It helps, of course, to reduce α to a value at which the optimization is stable and converges to optimal value of parameters $\boldsymbol{\beta}$.

3.3 Closed-Form Solution for Linear Regression

Let us rewrite the criteria function $J(\boldsymbol{\beta})$ for linear regression in a matrix-vector form,

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{N}(y_i - \hat{y}(\boldsymbol{x}_i))^2 \tag{3.23}$$

$$= (\boldsymbol{y} - \boldsymbol{X\beta})^\intercal(\boldsymbol{y} - X\boldsymbol{\beta}) \tag{3.24}$$

We are looking for the parameters $\boldsymbol{\beta}$ which minimize the value of the criteria function, that is, the value of the parameters where the gradient is $\mathbf{0}$. Let us first compute the gradient:

$$\frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\boldsymbol{X}^\intercal(\boldsymbol{y} - \boldsymbol{X\beta}). \tag{3.25}$$

Equating the gradient with $\mathbf{0}$, we obtain

$$\boldsymbol{X}^\intercal(\boldsymbol{y} - \boldsymbol{X\beta}) = \mathbf{0} \tag{3.26}$$

$$\boldsymbol{X}^\intercal - \boldsymbol{X}^\intercal\boldsymbol{X\beta} = \mathbf{0} \tag{3.27}$$

$$\boldsymbol{X}^\intercal X\boldsymbol{\beta} = \boldsymbol{X}^\intercal\boldsymbol{y} \tag{3.28}$$

$$\boldsymbol{\beta} = (\boldsymbol{X}^\intercal\boldsymbol{X})^{-1}\boldsymbol{X}^\intercal\boldsymbol{y} \tag{3.29}$$

Above is a closed form solution to the computation of parameters $\boldsymbol{\beta}$ of a linear regression model. Note that from here we can also compute the approximations of the values for indepent variable:

$$\hat{\boldsymbol{y}} = X\boldsymbol{\beta} \tag{3.30}$$

$$= \boldsymbol{X}(\boldsymbol{X}^\intercal\boldsymbol{X})^{-1}\boldsymbol{X}^\intercal\boldsymbol{y} \tag{3.31}$$

3.4 Linear Regression as a Classifier

In theory, linear regression could also be used for classification. Consider the following example and the data from Table 3.1. The data includes the mesaurements of body temperature and the record of the state of the visitor of doctor's office. We would like to infer the model that predicts the state (sick or healthy) from the body temperature. We encode the class variable with a number, 0 for healthy and 1 for sick, and present the data in a graph (Fig. 3.1).

Table 3.1: Body temperature and state of the visitor at doctor's office, where we state the class with a categorical variable and encode it with a number, a class variable y.

body temperature	state	y
36,5	healthy	0
36,6	healthy	0
36,8	healthy	0
36,9	sick	1
37,0	healthy	0
37,2	sick	1
37,5	sick	1
37,6	sick	1
39,5	sick	1

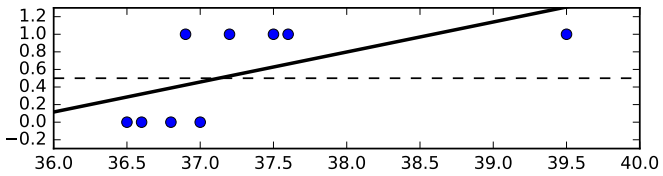

Figure 3.1: An attempt to develop a classifier with a linear regression.

In the graphically presented data (Fig. 3.1) we dare to fit the linear function $\hat{y} = f(x)$. We first notice that the range of the function $f(x)$ is inappropriate, since it ranges from $-\infty$ to ∞. In the interval of body temperatures around the point 37°, the function has a value between

0 and 1. The question arises how to interpret the estimated value, or how to convert it into probability. Recall that each probability function returns values between 0 and 1, and our linear function is limited to this interval only in a certain range of values of the input attribute. Additional problem is with outliers, or visitors with extreme value of the temperature. If we were to add another case of a patient with a very high temperature to the data, our function would change considerably and shift to the right. We need to instead develop a function that would output the probabilities, soften the output at left and right edges of the data, and then pose the problem in probabilistic and formal terms.

3.5 Logistic Function

Before we continue with a formal introduction of logistic regression, let us introduce a logistic function, a function that converts a real value into an interval from 0 to 1:

$$g(z) = \frac{1}{1 + e^{-z}} \tag{3.32}$$

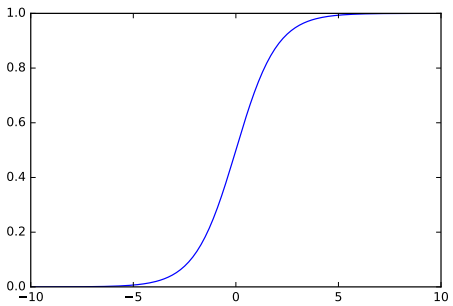

Figure 3.2: Logistic function.

The logistic function (Fig. 3.2) is continuous, monotonic; with $z \to -\infty$ it converges to 0

and with $z \to \infty$ it converges to 1. It derivative exists for all values of its parameter:

$$\begin{aligned}
\frac{dg(z)}{dz} &= \frac{d}{dz}\frac{1}{1+e^{-z}} \\
&= \frac{1}{(1+e^{-z})^2}e^{-z} \\
&= \frac{1}{1+e^{-z}}\frac{e^{-z}}{1+e^{-z}} \\
&= \frac{1}{1+e^{-z}}\frac{1+e^{-z}-1}{1+e^{-z}} \\
&= \frac{1}{1+e^{-z}}\left(1-\frac{1}{1+e^{-z}}\right) \\
&= g(z)[1-g(z)]
\end{aligned} \tag{3.33}$$

3.6 Likelihood for Logistic Regression

Following is an equation for a logistic regression model. In the core, this is a linear model, that is, a weighted sum of the values of the input variables.

$$\begin{aligned}
f_\beta(x) &= g(\boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 x_1 + \boldsymbol{\beta}_2 x_2 + \ldots \boldsymbol{\beta}_M x_M) \\
&= g(\boldsymbol{\beta}^\intercal \boldsymbol{x}) \\
&= \frac{1}{1+e^{-\beta^\intercal x}}
\end{aligned} \tag{3.34}$$

Notice that the linear combination of $\boldsymbol{\beta}$ and values of input variables, when equated to 0, defines the hyperplane in the feature space. Notice also the all the points that lie on the plane have a value of this function of 0. The value 0 is transformed by logistic function to 0.5. For any other data point that does not lie on the hyperpane the logistic function transforms this value to the value in the interval of $(0.5, 1.0]$ for the points on one side of the hyperplane, or the the value between $(0, 0.5]$ for the points on the other, negative side of the hyperplane. Conveniently, the farther away the point is from the hyperplane, the more to the extremes (0 or 1) will be its value of logistic regression. The hyperplane defined within the logistic regression model can thus serve as a decision boundary between two classes.

In what follows, we will use the logistic function so that our model $f_\beta(\boldsymbol{x})$ returns values between 0 and 1. We will continue to assume that our class variable is two-valued, but assume that its target value is a class marked with $y = 1$ and that for it, the model $f_\beta(\boldsymbol{x})$ returns the probability of this class. Notice that the parameters $\boldsymbol{\beta}$ fully define our logistic

regression model. Thus, we can write:

$$p(y = 1|\boldsymbol{x};\boldsymbol{\beta}) = f_{\boldsymbol{\beta}}(x) \tag{3.35}$$

$$p(y = 0|\boldsymbol{x};\boldsymbol{\beta}) = 1 - f_{\boldsymbol{\beta}}(x) \tag{3.36}$$

The expression for $p(y = 1|\boldsymbol{x};\boldsymbol{\beta})$ therefore gives us the probability that the target class of the data instance described by the vector of attribute values $\boldsymbol{x}$ is equal to 1. That is, it gives the probability that the independent variable takes the value of 1 for the data instance described with $\boldsymbol{x}$, where the model is parametrized with parameters $\boldsymbol{\beta}$. We can combine the two equations into a single one as

$$p(y|x;\boldsymbol{\beta}) = (f_{\boldsymbol{\beta}}(x))^y(1 - f_{\boldsymbol{\beta}}(x))^{1-y} \tag{3.37}$$

The expression for $p(y|\boldsymbol{x};\boldsymbol{\beta})$ in the above equation therefore gives the probability for a certain value of the independent variable and a certain vector of attribute values in the model given by $\boldsymbol{\beta}$. Now imagine that the values of the elements of the vector $\boldsymbol{\beta}$ change. Certainly, in this way, the probability for a given class in a selected case will also change; once this probability will be higher, another time lower. Change in the parameters of the model changes the probabilities with which we observe the data from the training set.

Let us now freeze the parameters of the model and compute the joint probability $L(\boldsymbol{\beta})$, the likelihood, for all the instances in the training set. We assume that the examples from the training set are independent and therefore the probability, with which we observe the values of the classes of particular data instanes described with the vectors $\boldsymbol{x}$ can be written as the product of the probabilities of individual data instances:

$$\begin{aligned} L(\boldsymbol{\beta}) &= p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\beta}) \\ &= \prod_{i=1}^{N} p(y_i|x_i;\boldsymbol{\beta}) \\ &= \prod_{i=1}^{N} f_{\boldsymbol{\beta}}(x_i)^{y_i}(1 - f_{\boldsymbol{\beta}}(x_i))^{1-y_i} \end{aligned} \tag{3.38}$$

Again, just like with the linear regression, we would like to infer the model that maximizes the likelihood. That is, the parameters where we maximize the probability with which the model observes the training set. For convenience of deriving the $\boldsymbol{\beta}$ which maximize the likelihood, we compute its logarithm, and then maximize the log-likelihood:

$$\begin{aligned} \ell(\boldsymbol{\beta}) &= \log L(\boldsymbol{\beta}) \\ &= \sum_{i=1}^{N} \Big[y_i \log f_{\boldsymbol{\beta}}(x_i) + (1 - y_i) \log(1 - f_{\boldsymbol{\beta}}(x_i)) \Big] \end{aligned} \tag{3.39}$$

3.7 Gradient Descent for Logistic Regression

We are therefore looking for such $\boldsymbol{\beta}$ that maximizes the log-likelihood $\ell(\boldsymbol{\beta})$. Since we will use the gradient method, and since $\boldsymbol{\beta}$ is a vector $[\beta_0\ \beta_1\ \ldots\ \beta_M]$, we need to calculate the partial derivatives of our criterion function:

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\beta}_j}\ell(\boldsymbol{\beta}) &= \sum_{i=1}^{M}\frac{\partial}{\partial \boldsymbol{\beta}_j}\Big[y_i \log f_{\boldsymbol{\beta}}(x_i) + (1-y_i)\log(1-f_{\boldsymbol{\beta}}(x_i))\Big] \\
&= \sum_{i=1}^{M}\Big[y_i\frac{1}{g(\boldsymbol{\beta}^\intercal x_i)} - (1-y_i)\frac{1}{1-g(\boldsymbol{\beta}^\intercal x_i)}\Big]\frac{\partial}{\partial \boldsymbol{\beta}_j}g(\boldsymbol{\beta}^\intercal x_i) \\
&= \sum_{i=1}^{M}\Big[\frac{y_i}{g(\boldsymbol{\beta}^\intercal x_i)} - \frac{(1-y_i)}{1-g(\boldsymbol{\beta}^\intercal x_i)}\Big]g(\boldsymbol{\beta}^\intercal x_i)(1-g(\boldsymbol{\beta}^\intercal x_i))\frac{\partial}{\partial \boldsymbol{\beta}_j}\boldsymbol{\beta}^\intercal x_i \\
&= \sum_{i=1}^{M}\Big[\frac{y_i - g(\boldsymbol{\beta}^\intercal x_i)}{g(\boldsymbol{\beta}^\intercal x_i)(1-g(\boldsymbol{\beta}^\intercal x_i))}\Big]g(\boldsymbol{\beta}^\intercal x_i)(1-g(\boldsymbol{\beta}^\intercal x_i))x_{ji} \\
&= \sum_{i=1}^{M}(y_i - g(\boldsymbol{\beta}^\intercal x_i))x_{ji} \\
&= \sum_{i=1}^{M}(y_i - f_{\boldsymbol{\beta}}(x_i))x_{ji} \qquad (3.40)
\end{aligned}$$

Very easy! Have we seen such a result or such an equation before? Of course! In linear regression. Partial derivatives are identical to those for linear regression. Of course with a small difference. This time our function $f_{\boldsymbol{\beta}}$ uses the logistic function, while in linear regression $f_{\boldsymbol{\beta}}$ was only the weighted sum of attribute values.

The iterative process using gradient descent to find the parameters of the logistic regression model is identical to that of linear regression. Of course, with the small difference that this time the $f_{\boldsymbol{\beta}}(x)$ is a logistic regression model. However, we write down the step for refreshing the value of the parameter $theta_j$:

$$\boldsymbol{\beta}_j \leftarrow \boldsymbol{\beta}_j + \alpha\sum_{i=1}^{N}\Big(y_i - f_{\boldsymbol{\beta}}(x_i)\Big)x_{ji} \qquad (3.41)$$

Again, the α learning rate is typically small for normalized data (e.g., 0.001).

A warning applies here. The gradient descent approach is slow, and even for medium-sized data, many iterations of correcting the values of the $\boldsymbol{\beta}$ parameters are required. Instead, we typically use optimization techniques that have faster convergence. One of these is the L-BFGS, which is typically used from an accessible Python library.