STA561: Probabilistic machine learning

Linear Regression (9/11/13)

Lecturer: Barbara Engelhardt   Scribes: Zachary Abzug, Mike Gloudemans, Zhuosheng Gu, Zhao Song

1 Why use linear regression?

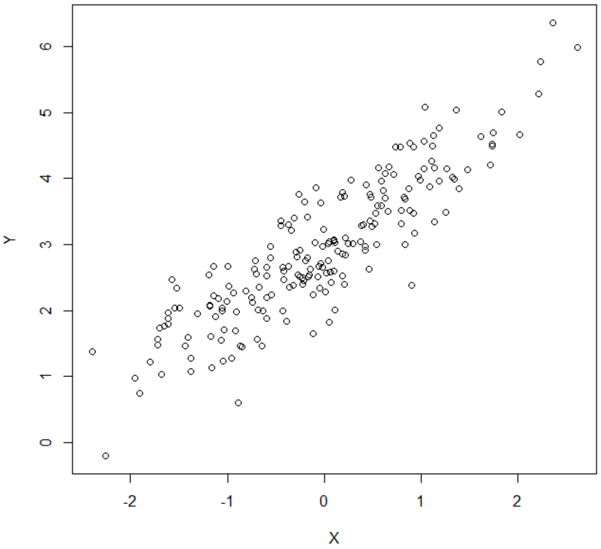

Figure 1: Scatter plot showing the relation between two variables, x and y

The general idea behind linear models is to capture how two observed variables are related in a linear way, assuming Gaussian noise. Given a fitted linear model (i.e., we have estimated the model parameters from training data), we may use this linear regression to:

- predict the value for y^* given a new observation x^*
- test if there is a linear relationship between two variables x and y
- when y is binary or multinomial, we can use this model to classify a new observation x^*.

Later in the course, we will learn non-linear and non-Gaussian versions of this basic model. Let's think about a bunch of possible relationships we want to model:

- x = age and y = height
- x = distance from the shore and y = weight of clams found

- x = gestational age and y = birthweight

Once we have fitted a linear model, then, given a new observation x^*, we can predict the corresponding value y^*. This model can also be used to test for associations between two variables. If there is no association between our variables, then the distribution of y should not depend conditionally on the value of x.

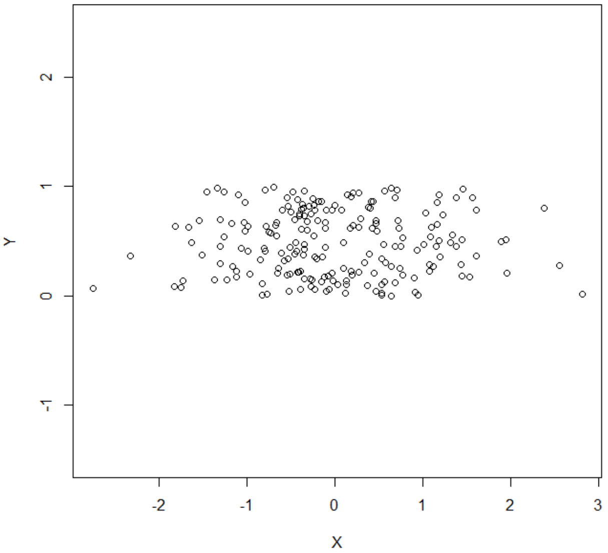

Figure 2: Scatter plot showing no relationship between two variables, x and y

2 Definitions & model specificiation

$$\begin{aligned} y \in \mathbb{R} &: \text{ response} \\ \boldsymbol{x} \in \mathbb{R}^p &: \text{ predictors (also called covariates or explanatory variables)} \\ \boldsymbol{\beta} \in \mathbb{R}^p &: \text{ regression coefficients (also called effects)} \end{aligned}$$

Our training set consists of observations for n samples $(\boldsymbol{x}, y)$. To fit the model, we will estimate the coefficients. Note that bold font for a random variable represents a vector.

2.1 Linear model

Our linear model has the following form:

$$y \quad = \boldsymbol{x}^T \boldsymbol{\beta} + \epsilon \tag{1}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Consequently, we have $y \mid \boldsymbol{x}^T, \boldsymbol{\beta}, \sigma^2 \sim \mathcal{N}(\boldsymbol{x}^T \beta, \sigma^2)$. Note here that the distribution of $\boldsymbol{x}$ doesn't *really* matter, as long as y is still distributed as a normal conditional on $\boldsymbol{x}$.

To build our linear model, we must first define our data. Let our training data $\mathcal{D}_{train} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$, where each $\boldsymbol{x}_i$ is a $p \times 1$ column vector of p different predictors, and we have n observations of each predictor and response. Then, using our knowledge of multivariate normals, we can expand (1) to

$$\begin{bmatrix} y_1 \\ . \\ . \\ . \\ y_n \end{bmatrix} = \mathcal{N}_n \left(\begin{bmatrix} \boldsymbol{x}_1^T \\ . \\ . \\ . \\ \boldsymbol{x}_n^T \end{bmatrix} \boldsymbol{\beta},\ \text{diag}_n(\sigma^2) \right)$$

where X is an $n \times p$ matrix. Our covariance matrix is a diagonal matrix with a single variance term for all samples (σ^2), and all covariance terms are zero, indicating independence across the n dimensions of this Gaussian (i.e., over the samples). When our model is fitted, if we are given a new $\boldsymbol{x}^*$, we can predict the associated y^* value; in particular, $E[y^*|\boldsymbol{x}^*] = \boldsymbol{x}^{*T}\beta$. Furthermore, if we take many samples with the same x-value, we should expect to see a normal distribution of y-values, centered around the mean that we've predicted.

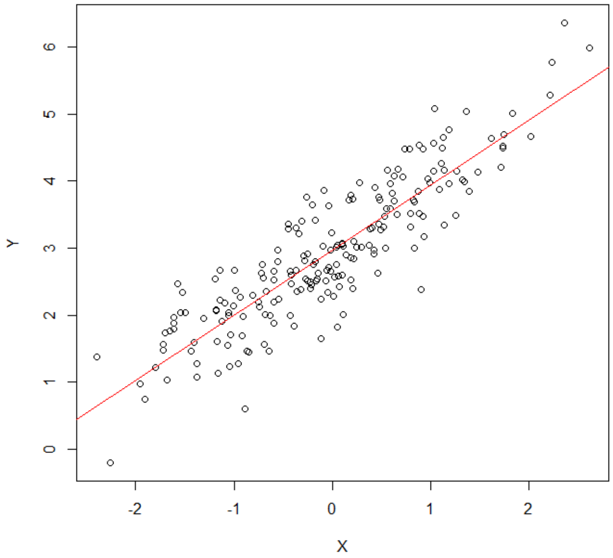

Figure 3: Our linear regression fit to the data. For any new x^*, we can use the model to predict the associated y^* by projecting the point $\boldsymbol{x}^*$ onto the red regression line and identifying the corresponding value of y^*.

2.2 What about offsets?

Our regression line rarely will go through the origin (a zero year old person is not zero inches tall), so we must also include an intercept term, β_0. We accomplish this by adding a 1 as the first element of our x vector, and a corresponding β_0 term as the first element of our β vector:

$$\boldsymbol{x} = [1,\, x_1,\, x_2, \cdots, x_p]^T$$

Our model equation will then take the form $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$, where we can read off the value of the intercept from β_0.

3 Evaluating our model

3.1 Residuals

How well do our response values we predict using the fitted model differ from the actual response values? We provide the following notation first (univariate case):

$$\begin{aligned} r_i &= y_i - \hat{\beta} x_i = y_i - \hat{y_i} \\ \hat{\beta} &= \text{our trained } \beta \\ \hat{y_i} &= \text{predicted value} \end{aligned}$$

For each data point, the error is the same as the residual: $\epsilon_i = y_i - \hat{\beta} x_i = r_i$. Additionally, by our previous definition of the error term, $\mathbb{E}[r_i] = \mathbb{E}[\epsilon_i] = \mathbb{E}[y_i - \hat{\beta} x_i] = 0$. In other words, the expected residual is zero according to the Gaussian model.

3.2 Residual Sum of Squares (and friends!)

There are a number of metrics to quantify the fit of the model to the test set or training set data based on the residual values:

- Residual Sum of Squares (RSS): $\text{RSS}(\hat{\beta}, \mathcal{D}) = \sum_{i=1}^{n} (y_i - \hat{\beta}^T x_i)^2$
- Mean Squared Error (MSE): $\text{MSE} = \frac{1}{n} \text{RSS}(\hat{\beta}, \mathcal{D})$
- Root-Mean-Square Error (RMSE): $\text{RMSE} = \sqrt{\text{MSE}}$

4 Parameter Estimation

For a univariate model ($p = 1$), the maximum likelihood estimate (MLE) for β, $\hat{\beta}$, can be found by taking the log likelihood, differentiating with respect to β, and setting equal to 0 and solving for β.

$$\begin{aligned} \ell(\beta; \mathcal{D}) &= \sum_{i=1}^{n} \log \left[\left(\frac{1}{2\pi\sigma^2} \right)^{1/2} \exp \left\{ \frac{-1}{2\sigma^2} (y_i - \beta x_i)^2 \right\} \right] \\ &= -\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \beta x_i)^2 - \frac{n}{2} \log (2\pi\sigma^2) \\ &= -\frac{1}{2\sigma^2} \text{RSS}(\beta, \mathcal{D}) + c \end{aligned}$$

where c is not a function of β. The first term inside the summed logarithm may be ignored because it is constant with respect to β. By looking at this equation, we see that we can maximize the log likelihood by equivalently minimizing the residual sum of squares (RSS).

Let's return to the general multivariate case, i.e. $p > 1$. We derive the MLE estimate for β as follows:

$$\begin{aligned} \frac{\partial \left[-\ell(\boldsymbol{\beta}; \mathcal{D}) \right]}{\partial \boldsymbol{\beta}} &= \frac{\partial [\frac{1}{2} (\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta})]}{\partial \boldsymbol{\beta}} \\ &= X^T X \boldsymbol{\beta} - X^T \boldsymbol{y} \qquad (2) \end{aligned}$$

Setting (2) to zero, and solving for β, we have

$$\hat{\boldsymbol{\beta}}_{MLE} = (X^T X)^{-1} X^T \boldsymbol{y} \tag{3}$$

(3) is called the *normal equation.* This is a wonderful result: If $X^T X$ is invertible, then we have an efficient way of computing the MLE for the regression coefficients (the cost of a matrix inversion). However, if $X^T X$ is non-invertible (singular), then we cannot compute an estimate of β using the normal equation. One example of a singular matrix $X^T X$ happens when the number of samples n is smaller than the number of predictors p.

4.1 Ridge regression

To regularize this problem, and guard ourselves against singular matrices, we can include a prior distribution for β. One advantage of selecting a prior distribution for the regression coefficients is that it helps prevent distortions of the fit due to outliers. If we select a normal prior, the variance τ^2 of the coefficients reflects our confidence in the regression: a high value of τ^2 shrinks the coefficients more heavily toward zero, whereas a low value of τ^2 lets the data guide the selection of β more heavily. With

$$p(\boldsymbol{\beta}|\tau^2) = \prod_{j=1}^{p} \mathcal{N}(\beta_j|0, \tau^2)$$

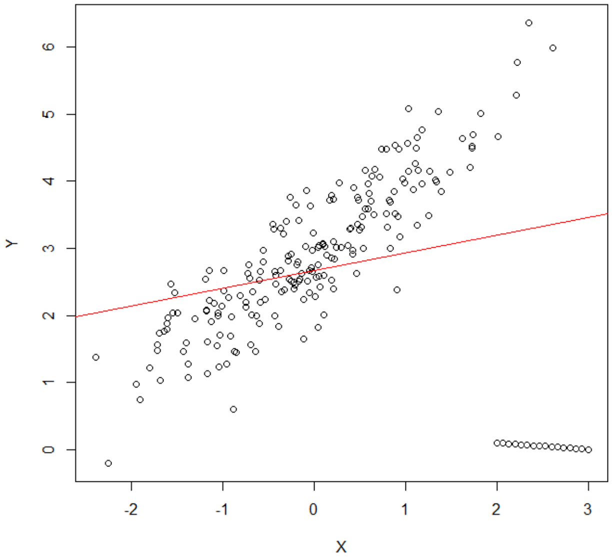

Figure 4: Distortion of the linear regression fit due to outliers.

we now calculate the log posterior distribution using Bayes Rule:

$$\begin{aligned} p(\boldsymbol{\beta}|\boldsymbol{x}, \boldsymbol{y}, \tau^2, \sigma^2) &\propto \sum_{i=1}^{n} \log \left[\left(\frac{1}{2\pi\sigma^2} \right)^{1/2} \exp \left\{ \frac{-1}{2\sigma^2} (y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i)^2 \right\} \right] + \sum_{j=1}^{p} \log \mathcal{N}\left(\beta_j|0, \tau^2\right) \\ &= \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i \right)^2 + \frac{\sigma^2}{\tau^2} \sum_{p=1}^{j} \beta_j^2 \end{aligned} \tag{4}$$

The solution for $\hat{\boldsymbol{\beta}}_{MAP}$ in Eqn. (4) is referred to as *ridge regression*

$$\hat{\boldsymbol{\beta}}_{MAP} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

where the first term of the argmax is our MSE equation (see section 4.2) and the second term is referred to as the *penalty*. Here, λ can be regarded as a regularization parameter corresponding to $\frac{\sigma^2}{\tau^2}$ in Eq (4). Consequently, we could compute the MAP estimate for $\boldsymbol{\beta}$ using the same strategy as in the normal equation and obtain

$$\hat{\boldsymbol{\beta}}_{\text{MAP}} = (X^T X + \lambda I_p)^{-1} X^T \boldsymbol{y}$$

Notice here that $(X^T X + \lambda I_p)^{-1}$ exists as the matrix $(X^T X + \lambda I_p)$ is generally invertible. In general, a simple way to make a singular matrix invertible is to add a very small value to the diagonal, in order to (artificially) create a full rank (and hence invertible) matrix.

5 Least mean squares (LMS) algorithm

Another way to solve for β is directly minimize the least square cost function (the residual sum of squares):

$$C(\beta) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\frac{\partial C(\beta)}{\partial \beta} = -\sum_{i=1}^{n}(y_i - \hat{y}_i)\, x_i$$

Then we can use the steepest descent algorithm to find the estimate for β

$$\beta^{(t+1)} = \beta^{(t)} + \rho\sum_{i=1}^{n}\left(y_i - {\beta^{(t)}}^T x_i\right) x_i$$

where ρ is the step size. Note here the summation is from 1 to n, which means we have to wait to see all the n data instances before we update $\beta^{(t+1)}$. This is called a *batch method*. However, it is often tempting to update our estimate as we collect each sample: In some cases n is too large for machines to process at once, or the matrix inversion in the normal equation may be difficult if there are too many features p). So we would like a way to process the data one sample at a time, which is called an *online method*. The LMS algorithm can be rewritten as:

$$\beta^{(t+1)} = \beta^{(t)} + \rho\left(y_i - {\beta^{(t)}}^T x_i\right) x_i,$$

which is known as *stochastic gradient decent*. This means at each iteration t, we randomly pick a data sample x_i, y_i from the data set and update our estimate of β until we convergence. This algorithm considers a single data point at a time, and is stochastic in selecting a sample from the full data.

5.1 Accounting for non-linearities

Imagine a model:

$$y_i = \phi(x_i)\boldsymbol{\beta} + \epsilon$$

$$\phi(x) = [1, x, x^2, \ldots, x^p]$$

This would allow us to find non-linear relationships between x and y. We can also try combinations of x-values, for $x_i = [x_1, x_2]_i$:

$$\phi(x) = [x_1, x_2, x_1x_2, x_1^2x_2, \ldots]$$

This function $\phi(x)$ is called the **basis function**. It turns out that the regression model above works with any basis function. Furthermore, the log-likelihood is convex with respect to $\boldsymbol{\beta}$, and this is true for all $\phi(x)$. Regardless of the $\phi(\cdot)$ term, this is a *linear model*, because the latent parameter β enters the model in a linear way.