Presentation Summary

Giorgio Morales and Kyle Webster

November 2020

1 Linear Regression Model

Given a set of variables $X = X_1, X_2, ..., X_n$, or independent variables, and given a real-values attribute Y, or the dependent variable, regression is a process that aims to predict the value of Y based on the independent variables. One such process is Linear Regression. Linear Regression is the process of finding a regression function f such that:

$$Y = f(X_1, X_2, ..., X_n) + \epsilon = f(X) = \epsilon \tag{1}$$

where ϵ is the error term, which is independent of X, and accounts for the uncertainty inherent in Y.

The regression function f can be expressed based on the multivariate random variable X and it's parameters such that:

$$f(X) = \beta + \omega_1 X_1 + \omega_2 X_2 + ... + \omega_n X_n = \beta + \sum_{i=1}^{n} \omega_i X_i = \beta + \omega^T X \tag{2}$$

where β is the bias, ω_i is the weight for X_i, and $\omega = (\omega_1, \omega_2, ..., \omega_n)^T$.

In general, the function f is a representation of a hyperplane with ω as the vector normal and β is the offset. When working with a given dataset, β and ω are unknown, so we have to estimate them utilizing a training set D that has points $x_i \in \mathbb{R}^n$. The estimated dependent can be determined using the function:

$$\hat{y} = b + w_1 x_1 + w_2 x_2 + ... + w_n x_n = \beta + \omega^T X \tag{3}$$

where b and w_i are trained from input test point x.

1.1 Bivariate Regression

Bivariate regression is an example of a linear regression problem where we are trying to estimate a single attribute. Consider the equation:

$$\hat{y_i} = f(x_i) = b + w \cdot x_i \tag{4}$$

where the goal is to minimize the error between the estimated value $\hat{(y_i)}$ and y_i. Since we are trying to minimize the error over all of the points in the dataset, then the goal is to minimize the sum of the squared error for the equation:

$$min_{b,w} SSE = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \hat{y_i})^2 = \sum_{i=1}^{n} (y_i - b - w \cdot x_i)^2 \tag{5}$$

To accomplish this, we solve by $b = \mu_Y - w \cdot \mu_X$ where w is the covariance between X and Y divided by the variance of X.

2 Multiple Regression

Multiple Regression is the case where there are n attributes in the training set. We start off by augmenting the training dataset D by setting $b = w_0 * x_{i0} = w_0$ to let us solve the equation $\hat{Y} = \tilde{D}\tilde{w}$. To find the best fitting hyperplane, we are minimizing the sum of the squared error as before. To solve for the weight vector $\tilde{w}$, we take the derivative of the sum of the squared error to get us the formula $\tilde{w} = (\tilde{D}^T\tilde{D})^{-1}\tilde{D}^T Y$. This lets us create the prediction formula $\hat{Y} = \tilde{D}(\tilde{D}^T\tilde{D})^{-1}\tilde{D}^T Y$.

2.1 Geometry of Multiple Regression

The linear regression model is interesting because the predicted vector $\tilde{Y}$ lies in the column space of the augmented dataset $\tilde{D} = \begin{bmatrix} | & | & ... & | \\ X_0 & X_1 & ... & X_n \\ | & | & ... & | \end{bmatrix}$ We want to minimize the residual vector error $\epsilon = Y - \hat{Y}$, which is orthogonal to the column space of $\tilde{D}$ and ϵ is orthogonal to each attribute X_i, then we can generate the normal equation $w_0 x_i^T X_0 + w_1 X_i^T X_1 + ... + w_n X_i^T X_n = X_i^T Y = \tilde{D}^T Y$.

2.2 QR-Factorization

Since the attribute vectors are not necessarily orthogonal, then we need to construct an orthogonal basis for $col(\tilde{D})$ to obtain the projected vector $\hat{Y}$. Using the *Gram-Schmidt orthogonization* method, we can constuct a set of orthoginal basis vectors $U_1, U_2, ..., U_n$ for $col(\tilde{D})$. This allows us to let $\tilde{D}$ which are a combination of the sum of the scalar projections of the previous X_j vectors onto the previous basis vectors U_j added to the X_i vector. This can be arranged to such that $\tilde{D} = QR$ where Q is the matrix of the basis vectors and R is the projections on the basis vectors.

2.3 Multiple Regression Algorithm

To solve the regression problem, we can simply use the expression found in the QR-Factorization process and replace it using our own processes such that $R\tilde{w} = (\Delta)^{-1}Q^T Y$ where Δ is the squared norms of the Q matrix. Given that $\hat{Y} = \tilde{D}\tilde{w}$, then we have $\hat{Y} = Q((\Delta)^{-1}Q^T Y)$. Therefore, we can make a prediction matrix by comparing the normalization of the orthogonal basis vectors with the training data.