

Splines and Linear and Polynomial Regression

Shrey Gupta

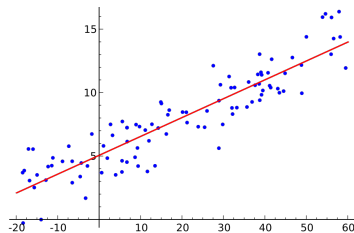
Applied Machine Learning (HOUSECS 59-01), Duke University

September 12, 2018

Regression

- ▶ Unsupervised learning: data (a subset from a larger distribution) is labeled, and we attempt to generalize to (predict) the larger distribution.
- ▶ Regression: predicts a continuous value output (i.e. estimates relationship among variables).

Regression: Examples



- ▶ Given data about square footage, age, zip code, and housing demand, predict the selling price of a house.
- ▶ Predict the percentage increase or decrease in the price of an equity.

Recall

$$X = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \dots & x_1^{(m)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \dots & x_2^{(m)} \\ \vdots & & & \ddots & \vdots \\ x_n^{(1)} & x_n^{(2)} & x_n^{(3)} & \dots & x_n^{(m)} \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- ▶ Data is stored in matrices and vectors.
- ▶ Given n (training) data points and m features (per data point).
- ▶ Given labeled data vector y .

Recall

$$X_{test} = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \dots & x_1^{(m)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \dots & x_2^{(m)} \\ \vdots & & & \ddots & \vdots \\ x_k^{(1)} & x_k^{(2)} & x_k^{(3)} & \dots & x_k^{(m)} \end{bmatrix}, \hat{y}_{test} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_k \end{bmatrix}$$

- ▶ Given k testing data points and m features (per data point).
- ▶ $\hat{y}_{test} = f(X_{test})$ contains *predictions* of the regression algorithm, where $f(\cdot)$ is learned by the algorithm.
- ▶ How do we define $f(\cdot)$, and how does the algorithm “learn” it?

Simple Linear Regression

$$y = \alpha + \beta x + \epsilon$$

$$\hat{y} = f(x) = \alpha + \beta x$$

- ▶ Goal: predict y from a single feature x .
- ▶ Allow α to be some *bias* not explained by x , and β the dependence of y on x .
- ▶ ϵ accounts for the “error” not explained by the model, and hence our estimate is \hat{y} .

Loss Function

$$\begin{aligned} & \min \ell(f(x), y) \\ &= \min \ell(\hat{y}, y) = \min \ell(\alpha + \beta x, y) \end{aligned}$$

- ▶ We want our estimates \hat{y} to be as accurate as possible for our choices of α and β .
- ▶ Allow ℓ to be some loss function, which gives a notion of the “distance” between \hat{y} and y .

Least-squares Error

$$\begin{aligned}\ell(f(x), y) &= \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2 \\ &= \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \frac{1}{n} \sum_{i=1}^n \epsilon_i^2\end{aligned}$$

- ▶ We will minimize the least-squares error, which is common in regression analysis for several reasons.
- ▶ Choices of α and β that minimize the loss, over the training data, will be used.

Multivariate Linear Regression

$$y = \beta_0 + \beta_1 x^{(1)} + \beta_2 x^{(2)} + \dots + \beta_m x^{(m)} + \epsilon$$

$$\hat{y} = f(x) = \beta_0 + \beta_1 x^{(1)} + \beta_2 x^{(2)} + \dots + \beta_m x^{(m)}$$

- ▶ Goal: predict y from multiple features $x^{(1)}, \dots, x^{(m)}$.
- ▶ Allow β_0 to be some *bias* not explained by the features, and β_i the dependence of y on feature $x^{(i)}$.
- ▶ ϵ accounts for the “error” not explained by the model, and hence our estimate is \hat{y} .

Compact Notation

$$y = \beta^T x + \epsilon$$

$$\hat{y} = f(x) = \beta^T x$$

- ▶ $\beta = (\beta_0, \beta_1, \dots, \beta_m)^T$ and $x = (1, x^{(1)}, x^{(2)}, \dots, x^{(m)})^T$.
- ▶ We can further extend this to allow for more data points.

Compact Notation

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \dots & x_1^{(m)} \\ 1 & x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \dots & x_2^{(m)} \\ \vdots & & & & \ddots & \vdots \\ 1 & x_n^{(1)} & x_n^{(2)} & x_n^{(3)} & \dots & x_n^{(m)} \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$y = X\beta + \epsilon$$

$$\hat{y} = f(X) = X\beta$$

- $\beta = (\beta_0, \beta_1, \dots, \beta_m)^T$ as before.

Multivariate Linear Regression

$$\min \ell(f(x), y) = \min \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

- ▶ Again, we want our estimates \hat{y} to be as accurate as possible for our choice of β .
- ▶ Utilize the least-squares error as the loss function.
- ▶ Closed form solution: $\beta = (X^T X)^{-1} X y$ (over the training data).

Multivariate Polynomial Regression

- ▶ y may not necessarily have a linear dependence on x .
- ▶ Solution: simply create new “features” $(x^{(i)})^2, (x^{(i)})^3, \dots$ for each feature $x^{(i)}$ (polynomial regression).
 - ▶ Closed form solution: $\beta = (X^T X)^{-1} X y$ (remains same).
 - ▶ High degree polynomials may lead to overfitting: choose degree via cross-validation (discussed later).

Practicalities

- ▶ Closed form solution $\beta = (X^T X)^{-1} X y$ may not be possible, as $(X^T X)^{-1}$ may not exist: use *pseudoinverse* instead.
- ▶ Still, computing the pseudoinverse (which uses the *singular value decomposition*) takes $O(\min(mn^2, m^2n))$ running time.
- ▶ May need to use *gradient descent* instead, with some learning rate α .
 - ▶ Choose some initial value of β .
 - ▶ Compute gradient of loss function, and move in direction of steepest (negative) change with step size α (chosen carefully).
 - ▶ Update β , and repeat until convergence.

Gradient Descent

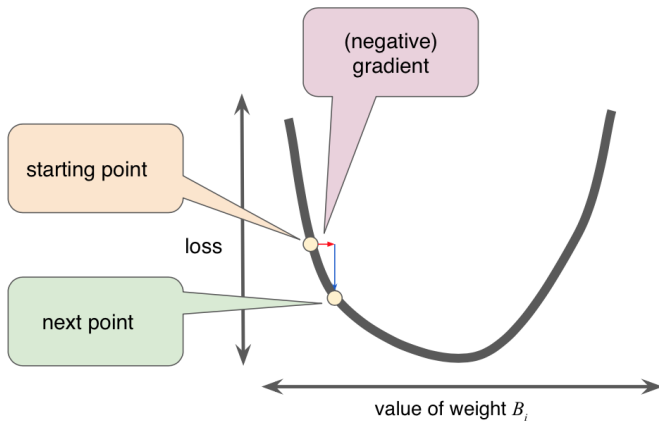


Image source: Google Developers

Feature Scaling

- ▶ In general, it is important to scale features such that $\mu^{(j)} = 0$ and $\sigma^{(j)} = 1$.
- ▶ Allows for proper convergence (in gradient descent) and assigns equal weight to features (in other applications and algorithms).

Regularization

- ▶ Goal is to prevent overfitting.
- ▶ General form: $\min \ell(f(x), y) + \lambda R(f)$, where ℓ is the loss function, R is the regularization function, and λ is the regularization coefficient.
 - ▶ Ordinary least squares regression: $\lambda = 0$.
 - ▶ Choose λ via cross-validation (discussed later).
- ▶ Has applications beyond linear and polynomial regression.

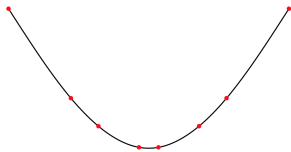
LASSO vs. Ridge Regression

- ▶ LASSO regression: $\min \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2 + \lambda |\beta|_1$
 - ▶ Used for variable selection: certain coefficients β_j can be 0.
 - ▶ Does not have a closed form solution.
- ▶ Ridge regression: $\min \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2 + \lambda |\beta|_2^2$
 - ▶ Closed form solution: $\beta = (X^T X + \lambda I)^{-1} X y$.

Interpolation

- ▶ Keeps “training” set of data points fixed and constructs a function around these data points.
- ▶ Note that, by construction, we are “overfitting” on the training data.
- ▶ Not very useful in machine learning, but applicable to other disciplines, and important nevertheless.
 - ▶ Used to estimate values *within* the range of data we have.
 - ▶ Regression is used to *extrapolate* to outside data points.

Spline Interpolation



- ▶ Create piecewise polynomials between each pair of consecutive points.
- ▶ Usually cubic polynomials (“splines”) to make the first and second derivatives continuous.

Image source: Wikipedia

Notebook

- ▶ Today's notebook will work through an example of regression, using the closed form solution and gradient descent.
- ▶ We'll also look at a brief example of spline interpolation for comparison.