7 Moving Beyond Linearity

Andrew Liang

11/20/2020

Notes

Polynomial Regression

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \ldots + \beta_d x_i^d + \epsilon_i$$

up to degrees of d

- is a linear model with predictors $x_i, x_i^2, \ldots, x_i^d$
- usually d is no greater than 3 or 4 otherwise model can become overly flexible

Step Functions

- polynomial functions imposes a *global* structure on the non-linear function of X
- use step functions to avoid imposing global strcuture
- break the range of X into *bins*, then fit different constant in each bin
  - converts continuous variable into an *ordered categorical variable*

Create cutpoints $c_1, c_2, \ldots, c_K$ in the range of X, then construct $K+1$ new variables:

$$\begin{aligned} C_0(X) &= I(X < c_1) \\ C_0(X) &= I(c_1 < X < c_2) \\ C_0(X) &= I(c_1 < X < c_2) \\ &\ldots \\ C_0(X) &= I(c_{K-1} < X < c_K) \\ C_0(X) &= I(c_K < X) \end{aligned}$$

where $I(\cdot)$ is an *indicator function* that returns a 1 if the condition is true, 0 otherwise.

For any value of X, $C_0(X) + C_1(X) + \ldots + C_K(X) = 1$, since X must be in exactly one of the $K+1$ intervals

We can then use least sqaure to fit a linear model using $C_1(X), C_2(X), \ldots, C_K(X)$ as predictors:

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \beta_K C_K(x_i) + \epsilon_i$$

- thus for a given X, at most one of $C_1, C_2, \ldots, C_K$ can be non-zero

Unless there are natural breakpoints in predictors, piecewise-constant functions can miss the action

Basis Functions

polynomial and piecewise-constant regression are special cases of basis function *approach* idea is to have a family of functions or transformations that can be applied to variable X: $b_1(X), b_2X, \ldots, b_K(X)$

Then fit the model:

$$y_i = \beta_o + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \ldots + \beta_K b_K(x_i) + \epsilon_i$$

* basis functions $b_1(\cdot), b_2(\cdot), \ldots, b_K(\cdot)$ must be fixed and known * since this is just a standard linear model with predictors $b_1(x_i), \beta_2 b_2(x_i), \ldots, b_K(x_i)$, all inference tools for linear models are still avaiable in this setting

Regression Splines

Piecewise Polynomials

- similar to polynomial regressions, but instead of fitting it over the entire range of X, we fit separate polynomial regressions over different regions of X
- a quadratic polynomial with a single *knot* at a point c takes the form:

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i^2 + \beta_{21} x_i^2 + \epsilon_i & \text{if } x_i < c \\ \beta_{02} + \beta_{12} x_i^2 + \beta_{22} x_i^2 + \epsilon_i & \text{if } x_i \geq c \end{cases}$$

- obviously more knots lead to a more flexible piecewise polynomial
  - K different knots through X results in $K + 1$ different polynomial regressions
- without constraints, the model will be discontinuous at each knot

Constraints and Splines

- in order to fix discontinuity problem, we apply a constraints that the model must be continuous, or that both *first* and *second* derivatives are continuous
  - setting both first and second derivatives to be continuous allows for piecewise polynomials to be smooth
    - continuous AT the knot
    - decreases degree of freedom by 3 (continuity, continuity of first derivative, continuity of second derivative)
  - setting only the model to be continuous allows for the model to be continuous but not as smooth (sudden changes in direction at the knots)
    - discontinuous AT the knot
  - each constraint lowers degree of freedom

Natural Cubic Splines

- splines can have high varaince at the outer range of predictors (Where X smaller than smallest knot, bigger than biggest knot)
  - to solve this, we add *boundary constraints*
    - enforce function to be linear at the boundary
      - will have lower CI at boundary regions

Choosing Number of Knots

- common practice to place knots in a uniform fashion
  - specify desired degrees of freedom
- can see which produces best looking curve, or use CV
- regression splines often perform superior to polynomial regression
  - especially at the boundary regions, where variance is highly volatile

Smoothing Splines

- to fit a smooth curve to data, want to find some function $g(x)$ so that:

$$RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2$$

is minimized

- however, no constraints on $g(x)$ would allow us to choose g such that it *interpolates* all of the y_{i}, in other words, we can simply just overfit the data to the extreme
  - to solve this, we can add a penalty term and minimize:

$$RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt$$

where λ is a nonnegative *tuning parameter* * want to minimize the integral of the second derivative of g because * it is the measure of the total change in the function $g'(t)$ in the range of t * if g is smooth, then $g'(t)$ will be close to constant and $\int g''(t)^2 dt$ will be small, vice versa * large λ values will penalize jumpy functions and as $\lambda \to \infty$, g will just be a straight line and thus perfectly smooth λ *controls bias-variance tradeoff of smoothing spline* a function $g(x)$ that minimizes the above equation actually places knots at *every* unique x values: $x_1, x_2, \ldots, x_n$! *NOT the same as a natural cubic spline, rather it is a *shrunken* version of it, where λ controls level of shrinkage

Choosing Smoothing Parameter

- seems like a smoothing spline might have far too many df, but λ effectively controls roughness of spline, and hence controls the *effective degrees of freedom*
- selecting λ is essentially equivalent to selecting how many df you want
- using LOOCV allows us to reduce RSS as small as possible:

$$RSS_{cv}(\lambda) = \sum_{i=1}^{n}(y_i - \hat{g}_{\lambda}^{(-i)}(x_i))^2$$

Local Regression

- idea is to fit a function at a target point x_0 using only the nearby training observations
- Local Regression Algorithm at $X = x_0$:

1) Gather fraction $s = k/n$ training points whose x_{i} are closest to x_{0}
2) Assign weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood, so that point furthest from x_0 has weight 0, while closest has highest weight. All but k nearest neighbors get weight 0
3) Fit *weighted least squares regression* of y_i on x_i using aforementioned weights, by finding $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize:
$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2$$
4) Fitted value at x_0 is given by $f(\hat{x_0}) = \hat{\beta}_0 + \hat{\beta}_1 x_0$

- the smaller value of s, the more local and wiggly our fit will be
- vice versa, the higher value of s leads to a more global fit

Generalized Additive Models (GAMs)

- extends standard linear model by allowing non-linear functions of each variable, while maintaining *additivity*
- can be applied to both quantitative/qualitative responses

GAMs for Regression

- replace each linear component $\beta_j x_{ij}$ with a smooth, non-linear function $f_j(x_{ij})$:
$$y_i = \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i$$
- it is an *additive* model because it calculates separate f_j for each X_j, then ads together all of their contributions
- can use all of the aforementioned methods as building blocks to fit an additive model

Pros of GAMs

- allow us to fit non-linear f_j to each X_j where standard linear regressions will fail to capture
- potentially allow more accurate predictions for response Y
- since model is additive, we can examine each effect X_j has on Y individually holding all other variables fixed - useful for inference
- smoothness of f_j for X_j can be summarized via degrees of freedom

Cons of GAMs

- model restricted to be additive, thus interactions between variables can be missed
  - however we can manually add interaction terms or low-dimensional interaction functions $f_{jk}(X_j, X_k)$ to the model

GAMs for Classification

$$log(\frac{p(X)}{1 - p(X)}) = \beta_0 + \beta_1 f_1(X_1) + \cdots + \beta_p f_p(X_p)$$