View on GitHub

stats-learning-notes

Notes from Introduction to Statistical Learning

Previous: Chapter 6 - Linear Model Selection and Regularization

---

Chapter 7 - Moving Beyond Linearity

Polynomial regression extends the linear model by adding additional predictors obtained by raising each of the original predictors to a power. For example, cubic regression uses three variables, X, X^2, and X^3 as predictors.

Step functions split the range of a variable into k distinct regions in order to produce a qualitative variable. This has the effect of fitting a piecewise constant function.

Regression splines are an extension of polynomials and step functions that provide more flexibility. Regression splines split the range of X into k distinct regions and within each region a polynomial function is used to fit the data. The polynomial functions selected are constrained to ensure they join smoothly at region boundaries called knots. With enough regions, regression splines can offer an extremely flexible fit.

Smoothing splines are similar to regression splines, but unlike regression splines, smoothing splines result from minimizing a residual sum of squares criterion subject to a smoothness penalty.

Local regression is similar to splines, however the regions are allowed to overlap in the local regression scenario. The overlapping regions allow for improved smoothness.

Generalized additive models extend splines, local regression, and polynomials to deal with multiple predictors.

Polynomial Regression

Extending linear regression to accommodate scenarios where the relationship between the predictors and the response is non-linear typically involves replacing the standard linear model

$$y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

with a polynomial function of the form

$$y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \beta_3 X_i^3 + . . . + \beta_d X_i^d + \epsilon_i$$

This approach is known as polynomial regression. For large values of d, polynomial regression can produce extremely non-linear curves, but a d greater than 3 or 4 is unusual as large values of d can be overly flexible and take on some strange shapes, especially near the boundaries of the X variable.

Coefficients in polynomial regression can be estimated easily using least squares linear regression since the model is a standard linear model with predictors X_i, X_i^2, …, X_i^d, which are derived by transforming the original predictor X.

Even though this yields a linear regression model, the individual coefficients are less important compared to the overall fit of the model and the perspective it provides on the relationship between the predictors and the response.

Once a model is fit, least squares can be used to estimate the variance of each coefficient as well as the covariance between coefficient pairs.

The obtained variance estimates can be used to compute the estimated variance of $\hat{f}(X_0)$. The estimated pointwise standard error of $\hat{f}(X_0)$ is the square root of this variance.

Step Functions

Polynomial functions of the predictors in a linear model impose a global structure on the estimated non-linear function of X. Step functions don't impose such a global structure.

Step functions split the range of X into bins and fit a different constant to each bin. This is equivalent to converting a continuous variable into an ordered categorical variable.

First, K cut points, $c_1, c_2, \ldots, c_k,$ are created in the range of X from which $K+1$ new variables are created.

$$C_0(X) = I(X < C_1),$$

$$C_1(X) = I(C_2 \leq X \leq C_3),$$

$$\ldots$$

$$C_{K-1}(X) = I(C_{K-1} \leq X \leq C_K),$$

$$C_K = I(C_K \leq X)$$

where I is an indicator function that returns 1 if the condition is true.

It is worth noting that each bin is unique and

$$C_0(X) + C_1(X) + \ldots + C_K(X) = 1$$

since each variable only ends up in one of $K+1$ intervals.

Once the slices have been selected, a linear model is fit using $C_0(X), C_1(X), \ldots, C_K(X)$ as predictors:

$$y_i = \beta_0 + \beta_1 C_1(X_i) + \beta_2 C_2(X_i) + \ldots + \beta_k C_k(X_i) + \epsilon_i$$

Only one of $C_1, C_2, \ldots, C_K$ can be non-zero. When $X < C,$ all the predictors will be zero. This means β_0 can be interpreted as the mean value of Y for $X < C_1$. Similarly, for $C_j \leq X < C_{j+1},$ the linear model reduces to $\beta_0 + \beta_j$, so β_j represents the average increase in the response for X in $C_j \leq X < C_{j+1}$ compared to $X < C_1$.

Unless there are natural breakpoints in the predictors, piecewise constant functions can miss the interesting data.

Basis Functions

Polynomial and piecewise constant functions are special cases of a basis function approach. The basis function approach utilizes a family of functions or transformations that can be applied to a variable $X: b_1(X), b_2(X), \ldots, b_K(X)$.

Instead of fitting a linear model in X, a similar model that applies the fixed and known basis functions to X is used:

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \ldots + \beta_k b_k(x_i) + \epsilon_i$$

For polynomial regression, the basis functions are $b_j(x) = x_i^j$. For piecewise constant functions the basis functions are $b_j(x_i) = I(c_j \leq x_i < c_{j+1})$.

Since the basis function model is just linear regression with predictors $b_1(x_i), b_2(x_i), \ldots, b_K(x_i)$ least squares can be used to estimate the unknown regression coefficients. Additionally, all the inference tools for linear models like standard error for coefficient estimates and F-statistics for overall model significance can also be employed in this setting.

Many different types of basis functions exist.

Regression Splines

The simplest spline is a piecewise polynomial function. Piecewise polynomial regression involves fitting separate low-degree polynomials over different regions of X, instead of fitting a high-degree polynomial over the entire range of X.

For example, a piecewise cubic polynomial is generated by fitting a cubic regression in the form

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \epsilon_i$$

but where the coefficients, $\beta_0, \beta_1, \beta_2, \beta_3,$ differ in different regions of the range of X.

The points in the range where the coefficients change are called knots.

Assuming no functions are repeated, a range of X split at K knots would be fit to $K + 1$ different functions of the selected type (constant, linear, cubic, etc.), one for each region.

In many situations, the number of degrees of freedom in the piecewise context can be determined by multiplying the number of parameters ($\beta_0, \beta_1, \ldots, \beta_j$) by one more than the number of knots. For a piecewise polynomial regression of dimension d, the number of degrees of freedom would be

$$d \times (K + 1)$$

Piecewise functions often run into the problem that they aren't continuous at the knots. To remedy this, a constraint can be put in place that the fitted curve must be continuous. Even then the fitted curve can look unnatural.

To ensure the fitted curve is not just continuous, but also smooth, additional constraints can be placed on the derivatives of the piecewise polynomial.

A degree-d spline is a degree-d polynomial with continuity in derivatives up to degree $d - 1$ at each knot.

For example, a cubic spline, requires that each cubic piecewise polynomial is constrained at each knot such that the curve is continuous, the first derivative is continuous, and the second derivative is continuous. Each constraint imposed on the piecewise cubic polynomial effectively reclaims one degree of freedom by reducing complexity.

In general, a cubic spline with K knots uses a total of $4 + K$ degrees of freedom.

The Spline Basis Representation

The basis model can be used to represent a regression spline. For example, a cubic spline with K knots can be modeled as:

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \ \ldots \ + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i$$

with an appropriate choice of basis functions. Such a model could then be fit using least squares.

Though there are many ways to represent cubic splines using different choices of basis functions, the most direct way is to start off with a basis for a cubic polynomial (X, X^2, X^3) and then add one truncated power basis function per knot. A truncated power basis function is defined as

$$h(x, \xi) = (x - \xi)_+^3 = \begin{cases} (x - \xi)^3 & \text{if x } > \xi \\ 0 & \text{otherwise,} \end{cases}$$

where ξ is the knot. It can be shown that augmenting a cubic polynomial with a term of the form $\beta_4 h(x, \xi)$ will lead to discontinuity only in the third derivative of ξ. The function will remain continuous with continuous first and second derivatives at each of the knots.

The means that to fit a cubic spline to a data set with K knots, least squares regression can be employed with an intercept and $K + 3$ predictors of the form $X,\ X^2,\ X^3,\ h(X, \xi_1),\ h(X, \xi_2),\ \ldots,\ h(X, \xi_K)$ where $\xi_1,\ \xi_2,\ \ldots,\ \xi_K$ are the knots. This amounts to estimating a total of $K + 4$ regression coefficients and uses $K + 4$ degrees of freedom.

Cubic splines are popular because the discontinuity at the knots is not detectable by the human eye in most situations.

Splines can suffer from high variance at the outer range of the predictors. To combat this, a natural spline can be used. A natural spline is a regression spline with additional boundary constraints that force the function to be linear in the boundary region.

There are a variety of methods for choosing the number and location of the knots. Because the regression spline is most flexible in regions that contain a lot of knots, one option is to place more knots where the function might vary the most and fewer knots where the function might be more stable. Another common practice is to place the knots in a uniform fashion. One means of doing this is to choose the desired degrees of freedom and then use software or other heuristics to place the corresponding number of knots at uniform quantiles of the data.

Cross validation is a useful mechanism for determining the appropriate number of knots and/or degrees of freedom.

Regression splines often outperform polynomial regression. Unlike polynomials which must use a high dimension to produce a flexible fit, splines can keep the degree fixed and increase the number of knots instead. Splines can also distribute knots, and hence flexibility, to those parts of the function that most need it which tends to produce more stable estimates.

Smoothing Splines

Smoothing splines take a substantially different approach to producing a spline. To fit a smooth curve to a data set, it would be ideal to find a function $g(X)$ that fits the data well with a small residual sum of squares. However without any constraints on $g(X)$, it's always possible to produce a $g(X)$ that interpolates all of the data and yields an RSS of zero, but is over flexible and over fits the data. What is really wanted is a $g(X)$ that makes RSS small while also remaining smooth. One way to achieve this is to find a function $g(X)$ that minimizes

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

where λ is a non-negative tuning parameter. Such a function yields a smoothing spline.

Like ridge regression and the lasso, smoothing splines utilize a loss and penalty strategy.

The term

$$\lambda \int g''(t)^2 dt$$

is a loss function that encourages g to be smooth and less variable. $g''(t)$ refers to the second derivative of the function g. The first derivative $g'(t)$ measures the slope of a function at t and the second derivative measures the rate at which the slop is changing. Put another way, the second derivative measures the rate of change of the rate of change of $g(X)$. Roughly speaking, the second derivative is a measure of a function's roughness. $g''(t)$ is large in absolute value if $g(t)$ is very wiggly near t and is close to zero when $g(t)$ is smooth near t. As an example, the second derivative of a straight line is zero because it is perfectly smooth.

The symbol $\int$ indicates an integral which can be thought of as a summation over the range of t. All together this means that $\int g''(t)^2 dt$ is a measure of the change in $g'(t)$ over its full range.

If g is very smooth, then $g'(t)$ will be close to constant and $\int g''(t)^2 dt$ will have a small value. On the other extreme, if g is variable and wiggly then $g'(t)$ will vary significantly and $\int g''(t)^2 dt$ will have a large value.

The tuning constant, λ, controls how smooth the resulting function should be. When λ is large, g will be smoother. When λ is zero, the penalty term will have no effect, resulting in a function that is as variable and jumpy as the training observations dictate. As λ approaches infinity, g will grow smoother and smoother until it eventually is a perfectly smooth straight line that is also the linear least squares solution since the loss function aims to minimize the residual sum of squares.

At this point it should come as no surprise that the tuning constant, λ, controls the bias-variance trade-off of the smoothing spline.

The smoothing spline $g(X)$ has some noteworthy special properties. It is a piecewise cubic polynomial with knots at the unique values of $x_1,\ x_2,\ \ldots,\ x_n$, that is continuous in its first and second derivatives at each knot. Additionally, $g(X)$ is linear in the regions outside the outer most knots. Though the minimal $g(X)$ is a natural cubic spline with knots at $x_1,\ x_2,\ \ldots,\ x_n$, it is not the same natural cubic spline derived from the basis function approach. Instead, it's a shrunken version of such a function where λ controls the amount of shrinkage.

The choice of λ also controls the effective degrees of freedom of the smoothing spline. It can be shown that as λ increases from zero to infinity, the effective degrees of freedom (df_λ) decreases from n down to 2.

Smoothing splines are considered in terms of effective degrees of freedom because though it nominally has n parameters and thus n degrees of freedom, those n parameters are heavily constrained. Because of this, effective degrees of freedom are more useful as a measure of flexibility.

The effective degrees of freedom are not guaranteed to be an integer.

The higher df_λ, the more flexible the smoothing spline. The definition of effective degrees of freedom is somewhat technical, but at a high level, effective degrees of freedom is defined as

$$df_\lambda = \sum_{i=1}^{n} \{S_\lambda\}_{ii},$$

or the sum of the diagonal elements of the matrix S_λ which is an n-vector of the n fitted values of the smoothing spline at each of the training points, $x_1,\ x_2,\ \ldots,\ x_n$. Such an n-vector can be combined with the response vector y to determine the solution for a particular value of λ :

$$\hat{g}_\lambda = S_\lambda y.$$

Using these values, the leave-one-out cross validation error can be calculated efficiently via

$$RSS_{cv}(\lambda) = \sum_{i=1}^{n} (y_i - \hat{g}_\lambda^{(-i)}(x_i))^2 = \sum_{i=1}^{n} \left[\frac{y_i - \hat{g}_\lambda(x_i)}{1 - \{S_\lambda\}_{ii}} \right]^2$$

where $\hat{g}_{\lambda}^{(-i)}$ refers to the fitted value using all training observations except for the ith.

Local Regression

Local regression is an approach to fitting flexible non-linear functions which involves computing the fit at a target point x_0 using only the nearby training observations.

Each new point from which a local regression fit is calculated requires fitting a new weighted least squares regression model by minimizing the appropriate regression weighting function for a new set of weights.

A general algorithm for local regression is

1. Select the fraction $s = \frac{k}{n}$ of training points whose x_i are closest to x_0.
2. Assign a weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood such that the point furthest from x_0 has a weight of zero and the point closest to x_0 has the highest weight. All but the k nearest neighbors get a weight of zero.

3. Fit a weighted least squares regression of the y_i on to the x_i using the weights calculated earlier by finding coefficients that minimize a modified version of the appropriate least squares model. For linear regression that modified model is

$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2$$

4. The fitted value at x_0 is given by

$$\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0.$$

Local regression is sometimes referred to as a memory-based procedure because the whole training data set is required to make each prediction.

In order to perform local regression, a number of important choices must be made.

- How should the weighting function K be defined?
- What type of regression model should be used to calculate the weighted least squares? Constant, linear, quadratic?
- What size should be given to the span S?

The most important decision is the size of the span S. The span plays a role like λ did for smoothing splines, offering some choice with regard to the bias-variance trade-off. The smaller the span S, the more local, flexible, and wiggly the resulting non-linear fit will be. Conversely, a larger value of S will lead to a more global fit. Again, cross validation is useful for choosing an appropriate value for S.

In the multiple linear regression setting, local regression can be generalized to yield a multiple linear regression model in which some variable coefficients are globally static while other variable coefficients are localized. These types of varying coefficient models are a useful way of adapting a model to the most recently gathered data.

Local regression can also be useful in the multi-dimensional space though the curse of dimensionality limits its effectiveness to just a few variables.

Generalized Additive Models

Generalized additive models (GAM) offer a general framework for extending a standard linear model by allowing non-linear functions of each of the predictors while maintaining additivity. GAMs can be applied with both quantitative and qualitative models.

One way to extend the multiple linear regression model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip} + \epsilon_i$$

to allow for non-linear relationships between each feature and the response is to replace each linear component, $\beta_j x_{ij}$, with a smooth non-linear function $f_j(x_{ij})$, which would yield the model

$$y_i = \beta_0 + \beta_1 f_1(x_{i1}) + \beta_2 f_2(x_{i2}) + \ \ldots \ + \beta_p f_p(x_{ip}) + \epsilon_i = \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i$$

This model is additive because a separate f_j is calculated for each x_i and then added together.

The additive nature of GAMs makes them more interpretable than some other types of models.

GAMs allow for using the many methods of fitting functions to single variables as building blocks for fitting an additive model.

Backfitting can be used to fit GAMs in situations where least squares cannot be used. Backfitting fits a model involving multiple parameters by repeatedly updating the fit for each predictor in turn, hold the others fixed. This approach has the benefit that each time a function is updated the fitting method for a variable can be applied to a partial residual.

A partial residual is the remainder left over after subtracting the products of the fixed variables and their respective coefficients from the response. This residual can be used as a response in a non-linear regression of the variables being updated.

For example, given a model of

$$y_i = f_1(x_{i1}) + f_2(x_{i2}) + f_3(x_{i3}),$$

a residual for x_{i3} could be computed as

$$r_i = y_i - f_1(x_{i1}) - f_2(x_{i2}).$$

The yielded residual can then be used as a response in order to fit f_3 in a non linear regression on x_3.

Pros and Cons of GAMs

- GAMs allow fitting non-linear functions for each variable simultaneously, allowing for non-linearity while also avoiding the cost of trying many different transformations on each variable individually.
- The additive model makes it possible to consider each x_j on y individually while holding other variables fixed. This makes inference more possible. Each function f_j for the variable x_{ij} can be summarized in terms of degrees of freedom.
- The additivity of GAMs also turns out to be their biggest limitation, since with many variables important interactions can be obscured. However, like linear regression, it is possible to manually add interaction terms to the GAM model by adding predictors of the form $x_j \times x_k$. In addition, it is possible to add low-dimensional interaction terms of the form $f_{jk}(x_j, x_k)$ that can be fit using two dimensional smoothers like local regression or using two-dimensional splines.

Overall, GAMs provide a useful compromise between linear and fully non-parametric models.

GAMs for Classification Problems

GAMs can also be used in scenarios where Y is qualitative. For simplicity, what follows assumes Y takes on values 0 or 1 and $p(X) = Pr(Y = 1|X)$ to be the conditional probability that the response is equal to one.

Similar to using GAMs for linear regression, using GAMs for classification begins by modifying the logistic regression model

$$log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ \ldots \ + \beta_p X_p + \epsilon_i$$

to pair each predictor with a specialized function instead of with a constant coefficient:

$$log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + f_1(X_1) + f_2(X_2) + \ \ldots \ + f_p(X_p) + \epsilon_i$$

to yield a logistic regression GAM. From this point, logistic regression GAMs share all the same pros and cons as their linear regression counterparts.

Next: Chapter 8 - Tree-Based Methods

stats-learning-notes maintained by tdg5

Published with GitHub Pages