STATS 315A: Statistical Learning

Erich Trieschman

2022Q1 class notes

Contents

1 Supervised learning overview

Regression problems are of the form $f(X) = E(y \mid X)$

A large subset of the most popular techniques use variants of linear regression and nearest neighbors.

1.1 Nearest neighbors

The k-nearest neighbor fit for $\hat{y}$ is

$$\hat{y} = f(x) = \frac{1}{k} \sum_{x_i \in N_{k(x)}} y_i$$

1.1.1 Curse of dimensionality

Any method that attemps to produce locally varying functions in small neighborhoods will have issues in high dimensions because of the **curse of dimensionality**. All sample points are close to an edge of the sample in high dimensions

For point p sampled uniformly within a p-dimensional sphere, the probability that p is within ϵ of the boundary goes to 1

$$\begin{aligned} V_{p,r} =& \frac{\pi^{p/2}}{\Gamma(p/2+1)} r^p \\ P(p \text{ within } \epsilon \text{ of bound.}) =& 1 - \frac{V_{p,1-\epsilon}}{V_{p,1}} \\ & 1 - (1-\epsilon)^p \\ \lim_{p \to \infty} 1 - (1-\epsilon)^p =& 1 \end{aligned}$$

Extension: The angle between two points sampled uniformly on a unit ball tends to 90 degrees

1.2 Statistical decision theory

1.3 Bias-Variance tradeoff

1.4 Cross validation

Cross validation is used to select the tuning parameters of a particular model, not the variables themselves. For example with subset selection, we use cross validaiton to select s, the subset size, **not** the actual predictors to use in the model
It would be ideal to have both a test set and a cross validation set. Running the CV and tuning parameters can bias the results. A separate test set provides convincing independent assessment

1.4.1 K-fold cross validation

- For each k, fit the model with parameter λ to the other K-1 parts, getting $\hat{\beta}^{-k}(\lambda)$
- Compute error, $RSS_{-k} = \sum_{i \in k} (y_i - x_i \hat{\beta}^{-k}(\lambda))^2$
- Cross validation error, $CV(\lambda) = \frac{1}{K} \sum_{k=1}^{K} RSS_{-k}(\lambda)$

1.4.2 We expect test error to be greater than training error

Here we show that the test error is bounded from below by the training error, $E[R_{te}(\hat{\beta}))] \geq E[R_{tr}(\hat{\beta})]$

$$\begin{aligned} \text{Known: } E(R_{tr}(\beta)) =& \frac{1}{n} \sum_{i=1}^{n} E[(y_i - x_i^T \beta)^2] = E[(y_1 - x_1^T \beta)^2] \\ \text{Known: } E(R_{te}(\beta)) =& \frac{1}{m} \sum_{i=1}^{m} E[(\tilde{y}_i - \tilde{x}_i^T \beta)^2] = E[(\tilde{y}_i - \tilde{x}_i^T \beta)^2] \\ \text{Known: } & \left\| y - X\hat{\beta} \right\|_2 \leq \|y - X\beta\|_2 \, \forall \beta \neq \hat{\beta} \implies n R_{tr}(\hat{\beta}) \leq n R_{tr}(\beta) \Rightarrow E[R_{tr}(\hat{\beta})] \leq E[R_{tr}(\beta)] = E[R_{te}(\beta))] \\ & E[R_{te}(\hat{\beta}))] = E[E[R_{te}(\hat{\beta})) \mid \hat{\beta}]] = E[E[R_{te}(\hat{\beta})]] \geq E[E[R_{tr}(\hat{\beta})]] \geq E[R_{tr}(\hat{\beta})] \end{aligned}$$

1.5 Bootstrap

- Sample N times with replacement from teh training set to form a bootstrap data set
- Estimate model on bootstrap data, with predictions made from the original training data
- Repeat process many times and average results
- **Poor estimate of prediction error.** The reason is that the bootstrap datasets are acting as the training samples, while the original training set is acting as the test sample, and these two samples have observations in common. This overlap can make overfit predictions look unrealistically good, and is the reason that cross- validation explicitly uses non-overlapping data for the training and test samples
- Good estimate for standard errors of predictions and confidence intervals for parameters

2 Linear methods for regression

Functions in the real world are rarely linear, but linear approximations are a good heuristic for the biance-variance tradeoff.

2.1 Linear regression and least squares

Assuming X full rank. Geometrically, the point, $\hat{\beta}$ which solves $argmin_x \left\|X\hat{\beta} - y\right\|_2$ is one where $X\hat{\beta} - y$ is orthogonal to the range of X. To solve for this:

$$\text{Want: } (X\hat{\beta} - y) \perp \{z | z = X\hat{\beta}\} \longleftrightarrow (X\hat{\beta} - y) \perp range(A) \longleftrightarrow (X\hat{\beta} - y) \perp x_i, \forall i \in X$$
$$x_i^T(X\hat{\beta} - y) = 0,\ \forall i \in X \longleftrightarrow X^T(X\hat{\beta} - y) = 0 \longleftrightarrow \hat{\beta} = (X^TX)^{-1}X^TY$$

Properties:

- Regression coefficient $\hat{\beta}_i$ estimates the expected change in y per unit change in x_i *holding all other predictors fixed*
- For X_1, X_2, mutually orthogonal matrices or vectors, the joint regression coefficients for $X = (X_1, X_2)$ on y, can be found from separate regressions. (Proof: $X_1^T(y - X\hat{\beta}) = X_1^T(y - X_1\hat{\beta}_1) = 0$)
- The multiple regression coefficient of x_p, the last column of X, is the same as the univariate coefficient in the regression of $y \sim z_p$. Here, $z_p = x_p - X_p^T\alpha$ (the part of x_p orthogonal to X_p, all but column x_p of X). Variance also comes form the univariate regression.
  - $\hat{\beta}_p = (z_p^Tz_p)^{-1}z_p^Ty = z_p^Ty/z_p^Tz_p$
  - $Var(\hat{\beta}_p) = \sigma^2/z_p^Tz_p$

Assumptions:

- Errors, $\epsilon_i \sim N(0, \sigma^2)$ assumed to be *independent* of the x_i's
- X considered fixed, not random.
- X is full rank. When not (because multiple variables are perfectly correlated), X^TX is singular and the coefficients, $\hat{\beta}$, are not uniquely defined. In these cases, features can be reduced by filtering or with a regularization.
- Conditional expectation of y is linear in X, $y = E(y \mid X) + \epsilon$. With this assumption, we can show $\hat{\beta} \sim N(\beta, (X^TX)^{-1}\sigma^2)$

2.1.1 Standard error and confidence intervals

We often assume $y_i = \hat{\beta}x_i + \epsilon_i$ with $E(\epsilon_i) = 0$ and $Var(\epsilon_i) = \sigma^2$. Then

$$se(\hat{\beta}) = \left[\frac{\sigma^2}{\sum(x_i - \bar{x})^2}\right]^{\frac{1}{2}}, \text{ approximating with } \hat{se}(\hat{\beta}) = \left[\frac{\hat{\sigma}^2}{\sum(x_i - \bar{x})^2}\right]^{\frac{1}{2}} \text{ where } \hat{\sigma}^2 = \frac{\sum(y_- \hat{y}_i)^2}{N - 2}$$

2.1.2 Expectation of $\hat{\beta}$

$$
\begin{aligned}
Var(\hat{\beta} \mid X) =& (X^T X)^{-1}\sigma^2 \\
E(\hat{\beta}) =& E[(X^T X)^{-1}X^T y] = E[(X^T X)^{-1}X^T(X\beta + \epsilon)] \\
=& E[(X^T X)^{-1}X^T X\beta + (X^T X)^{-1}X^T\epsilon)] = E(\beta) + E[(X^T X)^{-1}X^T]E(\epsilon) \\
=& E(\beta) + E[(X^T X)^{-1}X^T] * 0 = E(\beta) \text{ for } y = X\beta + \epsilon \\
E(\hat{\beta} \mid X) =& E[(X^T X)^{-1}X^T y \mid X)] = (X^T X)^{-1}X^T E[f(x) + \epsilon \mid X] \\
=& (X^T X)^{-1}X^T[f(x) + E(\epsilon \mid X)] = (X^T X)^{-1}X^T f(x) \text{ for } y = f(x) + \epsilon \\
\hat{\sigma}^2 =& \frac{1}{N-p-1}\sum(y_i - \hat{y}_i)^2, \text{ the } N-p-1 \text{ denominator makes } \hat{\sigma} \text{ unbiased } (E(\hat{\sigma}^2) = \sigma^2)
\end{aligned}
$$

2.2 Subset selection

Subset methods help us tradeoff an increase in bias with lower variance. Here we retain a subset of predictor variables for the final regression. **Approaches:**

- All subsets regression: finds the best subset of size $s \in \{1, \ldots, p\}$ that minimizes the residual sum of squares. Limited use in high dimensions (¿30) because of the computational complexity.
- Forward stepwise selection: beginning with a model of the intercept only, sequentially add to the model the predictor that most reduces the residual sum of squares
- Backward stepwise selection: beginning with the full OLS model, sequentially remove from the model the predictor to most reduce residual sum of squares

Note: The tuning parameter, s of each subset selection approach should be determined through cross validation

2.3 Shrinkage methods

- Shrinkage methods often help us tradeoff an increase in bias with lower variance
- It is important to standardize (mean=0, variance=1) the predictors before running shrinkage methods to make the pentalty meaningful; centering also eliminates the need for an intercept

2.3.1 Ridge regression

Ridge regression is a linear regression with a square penalty on the size of the model parameters:

$$
\begin{aligned}
\hat{\beta}^{ridge} &= argmin(y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta \\
\hat{\beta}^{ridge} &= (X^T X + \lambda I)^{-1}X^T y \\
\hat{\beta}^{ridge} &= X^T(XX^T + \lambda I)^{-1}y
\end{aligned}
$$

- See the "kernel trick" for derivation of the last equation
- Ridge regression is a biased estimator for y that may reduce MSE. Note when $\lambda = 0$, this is the same as OLS
- The ridge regression solution can be derived as the posterior distribution of the bayes equation as well, assuming $f(y \mid \beta) \sim N(X\beta, \sigma^2 I)$ and $\beta \sim N(0, \tau I)$. Here, $\lambda = \frac{\sigma^2}{\tau}$

Ridge regression shrinks the coefficients of the principal components (Xv_j), with relatively more shrinkage on the smaller components. Proof:

$$
\begin{aligned}
X\hat{\beta} &= X(X^T X + \lambda I)^{-1}X^T y \\
X\hat{\beta} &= UDV^T(VD^2V^T + \lambda I)^{-1}VDU^T y \\
&= UD(D^2 + \lambda I)^{-1}DU^T y \\
&= \sum_{j=1}^{p} u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^T y
\end{aligned}
$$

2.3.2 The Lasso

The lasso is a shrinkage method for linear regressions like ridge, but uses the 1-norm as a penalty instead of the 2-norm.

$$\hat{\beta}^{lasso} = argmin(y - X\beta)^T(y - X\beta) + \lambda \|\beta\|_1$$

There is no analytical solution to this objective, but the lasso is a convex problem when stated below (meaning it can be minimized)

$$\begin{aligned} &\text{min. } argmin(y - X\beta)^T(y - X\beta) \\ &\text{subject to } \lambda \|\beta\|_1 \leq t \end{aligned}$$

We find with the lasso that the parameter vector often inclues zeros for specific parameters. Inuitively this makes sense since the pointed 1-norm ball is likely to be maximized at one of its corners.

Elastic net combines the ridge and lasso penalties through tuning parameter α. It can be effective for sparse models with correlated predictors.

$$\hat{\beta}^{enet} = argmin(y - X\beta)^T(y - X\beta) + (1 - \alpha) \|\beta\|_2 + \alpha \|\beta\|_1$$

2.4 Methods using derived input directions

Here we choose a set of linear combinations of $x_i \in X$, and run a regression on these combinations

2.4.1 Principal component regression

Linear combinations are selected to maximize variance. These maximal-variance combinations are called the **principal components**. For standardized X, the principal components, z_i, are

$$\begin{aligned} &z_1 = Xv \text{ such that } v \text{ maximizes } Var(Xv) = \frac{1}{N} v^T X^T X v \text{ subject to } \|v\|_2 = 1 \\ &z_i = Xv_i \text{ where } v_i \text{ is the ith column of the SVD; singular values determine the ordering} \end{aligned}$$

The principal component analysis is highly connected to the singular value decomposition of standardized X. Since the SVD can help us construct the eigendecomposition of $X^T X$

$$\frac{1}{N} X^T X = \frac{1}{N} V D^2 V^T \iff \frac{1}{N} V^T X^T X V = D^2$$

Principal Component Analysis regression then generates a linear regression using a subset $s \leq p$ of the principal compoents. Since these principal components are orthogonal, the regression is a sum of univariate regressions

2.4.2 Partial least squares

Linear combinations are constructed using both y and X, both standardized.

- Compute univariate regression coefficients, $\hat{\gamma}_l$ of y on eacy x_l
- Construct $z_1 = \sum_l \hat{\gamma}_l x_l$
- Get $\hat{\beta}_1$ from $y \sim z_1$
- Orthogonalize $y, x_1, \ldots, x_p$ with respect to z_1
  - $y^* = y - \hat{\beta}_1 z_1$
  - $x_l^* = x_l - \frac{z_1^T x_l}{z_1^T z_1} z_1$
- Repeat until $s \leq p$ directions have been obtained (we get back OLS if $s = p$)

2.5 Degrees of freedom

Degrees of freedom for linear regressions is the number of free parameters that determines the model.

$$\begin{aligned} &\text{For } \hat{y} = Hy, \, df = tr(H) \\ &\text{Note } \hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y \text{ so } H = X(X^T X)^{-1} X^T \text{ in OLS} \\ &\qquad \hat{y} = X\hat{\beta} = X(X^T X + \lambda I)^{-1} X^T y \text{ so } H = X(X^T X + \lambda I)^{-1} X^T \text{ in Ridge regression} \end{aligned}$$

The lasso is not a linear regression. Its degrees of freedom are defined as

$$df = \sum_i cov(y_i, \hat{y}_i)/\sigma^2$$

3 Linear methods for classification

For classification, the input space can be divided into regions of constant classification, with decision boundaries

$$\{x \mid \hat{\beta}_k x = \hat{\beta}_l x\}$$

In general, linear methods for classification model *discriminant functions*, $\delta_k(x)$, or *posterior probabilities*, $P(G = k \mid X = x)$, for each class, classifying x to the class with the largest discriminant or probability.
For this theory, we require that these functions have some monotone transformation to a linear function; it turns out that both linear discriminant analysis and linear logistic regression result in linear log-odds (logits).

3.1 Linear regression of indicator matrix

Categorical $y \in \mathbb{R}^{n\times 1}$ is one-hot-encoded into a series of boolean vectors in $Y \in \mathbb{R}^{n\times k}$, and model parameters, $\hat{\beta} \in \mathbb{R}^{p\times k}$ are estimated in the same way

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$
$$\hat{Y} = X\hat{\beta} = X(X^T X)^{-1} X^T Y$$
$$\hat{G} = argmax_{k\in\mathcal{G}} x^T \hat{\beta} \text{ for new observation } x$$

Notes

- Rigid nature of regression model, classes can be masked by other classes for $K > 2$
- A loose, but general rule is that if $K \geq 3$ classes are lined up in one direction, polynomial terms of degree $K - 1$ might be needed to resolve the classes.
- Bias has less of an effect in classification than in regression

3.2 Linear discriminant analysis (LDA)

Category of backward selection models, meaning we use y to generate boundaries of X. Suppose $f(x \mid k)$ is the density of X conditional on class $G = k$, and π_k is the prior probability for class $G = k$. Bayes rule says

$$P(G = k \mid X = x) = \frac{f(x \mid k)\pi_k}{\sum_{l=1}^{K} f(x \mid l)\pi_l}$$

Linear discriminant analysis and its relatives are based on the assumption that $f(x \mid k)$ is multivariate Gaussian

$$f(x \mid k) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k))$$

3.2.1 Linear discriminant analysis

LDA specifically arises out of the special case when we assume all classes have a common variance, i.e., $\Sigma_k = \Sigma \forall k$. This results in sufficient cancelations when comparing classes, leading to an equation linear in x

$$\begin{aligned}\log \frac{P(G = k \mid X = x)}{P(G = k \mid X = x)} &= \log \frac{f(x \mid k)}{f(x \mid l)} + \log \frac{\pi_k}{\pi_l} \\ &= \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k + \mu_l)^T \Sigma^{-1}(\mu_k - \mu_l) + x^T \Sigma^{-1}(\mu_k - \mu_l)\end{aligned}$$

Notes

- This implies the decision boundary between classes is linear in x
- These decision boundaries are *not* perpendicular bisectors to the line segments joining centroids (this would be the case if $\Sigma = \sigma^2\mathbf{I}$) and the priors, π_i were equal)
- In practice we don't know the parameters of the Gaussian distributions and we estimate with
  - $\hat{\pi}_k = n_k/N$
  - $\hat{\mu}_k = \sum_{g_i=k} x_i/n_k$
  - $\hat{\Sigma}_k = \sum_{k=1}^{K} \sum_{g_i=k} (x_i - \hat{\mu}_k)^T (x_i - \hat{\mu}_k)/(N - K)$
- In the case of a binary y, LDA and linear regression have a direct correspondence

3.2.2 Quadratic discriminant analysis (QDA)

If we tighten our assumption about Σ_k so that the covariance matrices are not even across groups, we get a **Quadratic discriminant analysis** that is quadratic in x:

$$\delta_k(x) = \frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + log\pi_k$$

Notes

- Decision boundaries between classes are described by a quadratic equation: $\{x \mid \delta_k(x) = \delta_l(x)\}$
- We can produce quadratic boundaries with LDA as well by enlarging the parameter space to include quadratics ($(x_1, x_2) \Longrightarrow (x_1, x_2, x_1x_2, x_1^2, x_2^2)$). In general, QDA is the preferred approach

3.2.3 Regularized discriminant analysis (RDA)

Suposing we want to compromise between LDA and QDA, **regularized discriminant analysis** allows us to tune in between our assumptions of common across-class covariance (LDA) and unique within-class covariance (QDA) with a tuning parameter, α. Regularized covariance matrices have the form

$$\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}$$

We can also consider a regularization approach to tune a caommon across-class covariance to an assumption about independence with tuning parameter, γ

$$\hat{\Sigma}(\gamma) = \gamma\hat{\Sigma} + (1-\gamma)\hat{\sigma}^2\mathbf{I}$$

3.2.4 Additional LDA notes

We can reduce the computational complexity of LDA by taking advantage of the eigendecomposition of the covariance matrix: $\hat{\Sigma} = UDU^T$, noticing

$$(x-\hat{\mu}_k)^T\Sigma^{-1}(x-\hat{\mu}_k) = [U^T(x-\hat{\mu}_k)]^TD^{-1}[U^T(x-\hat{\mu}_k)]$$
$$\log|\hat{\Sigma}| = \sum_l \log d_{ll}$$

With these relations, X can be classified through two steps

- Sphere data: $X^* = D^{-1/2}U^TX$ where $\hat{\Sigma} = UDU^T$ (note now that the common covariance matrix is $\mathbf{I}$)
- Classify the closest class centroid in the transformed space, modulo the effect of class probabilities, π_k

3.3 Logistic Regression

Logistic regression models the posterior probabilities of K classes through a linear function in x. Logistic regressions also ensure that the probabilities sum to one and remain in $[0, 1]$. The model form is (with intercept)

$$\log\frac{P(G=1 \mid X=x)}{P(G=K \mid X=x)} = \beta_1^Tx$$
$$\log\frac{P(G=2 \mid X=x)}{P(G=K \mid X=x)} = \beta_2^Tx$$
$$\vdots$$
$$\log\frac{P(G=K-1 \mid X=x)}{P(G=K \mid X=x)} = \beta_{K-1}^Tx$$

Where probabilities can be calculated as

$$P(G=k \mid X=x) = \frac{\exp(\beta_k^Tx)}{1+\sum_{l=1}^{K-1}\exp(\beta_l^Tx)},\ k \in \{1,\ldots,K-1\}$$
$$P(G=K \mid X=x) = \frac{1}{1+\sum_{l=1}^{K-1}\exp(\beta_l^Tx)}$$
$$P(G=k \mid X=x) = p_k(x;\theta)$$

Logistic regressions are fit by maximizing the likelihood function

$$max_\theta\left\{\sum_{i=1}^n \log p_i(x_i;\theta)\right\}$$

3.3.1 Additional logistic regression notes

We can add regularization terms to increase the robustness of our estimations

$$\max_{\theta} \left\{ \sum_{i=1}^{n} \log p_i(x_i; \theta) - \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

Notes

- If two classes in a logistic regression are linearly separable, then the solution is undefined (MLE tries to achieve probabilities of 0 and 1, leading to $\hat{\beta} \to \pm\infty$)
- It turns out logistic regression and LDA both model log-posterior odds between classes as a linear function of x. The difference lies in the way the linear coefficients are estimated.
- Logistic regression makes fewer assumptions: it leaves the marginal density of X, $f(x \mid k)$ as an arbitrary density function, $f(x)$ and maximizes the conditional likelihood with $f(x)$ implicitly
- LDA relies on additional model assumptions, which gives us more information about parameters (lower variance). If normality of the underlying data holds, LDA is more effective

3.3.2 Pivot classes

The loinear logistic regression model without a regularization term requires a pivot class. This is because the parameters, β are not uniquely defined without one:

$$p_i(x) = P(G = i \mid x) = \frac{e^{\beta_i x}}{\sum_{j=1}^{k} e^{\beta_j x}} = \frac{e^{\beta_i x} e^{Cx}}{\sum_{j=1}^{k} e^{\beta_j x} e^{Cx}} = \frac{e^{(\beta_i + C)x}}{\sum_{j=1}^{k} e^{(\beta_j + C)x}}$$

However, fixing $\beta_K = 0$ (or any one class) as the pivot allows us to uniquely define our β. Note that when we include a ridge penalty in the linear logistic regression model, this additional requirement on β allows us to uniquely define β without a pivot class

3.4 Naive Bayes models

Naive Bayes gives us a very simple model that assumes within-class independence. Even when this assumption isn't accurate, it can give us a good estimator with a simple implementation

$$\text{Assume: } f(X \mid j) \approx \prod_{m=1}^{p} f_m(X_m \mid j)$$

$$P(G = k \mid X) = \frac{f(X \mid k)\pi_k}{\sum_{l=1}^{K} f(X \mid l)\pi_l}$$

Notes

- Each component density f_m is estimated separately within each class
- The nearest shrunken centroid model has this structure
- More general models have less bias, but are typically hard to estimate in high dimensions, so independence assumption may not hurt too much

3.5 Receiver operating characteristics (ROC)

Consider a binary classifier. To classify a new observation, x_0 we can threshold $P(Y = 1 \mid X = x_0)$ at 0.5. Other thresholds can change the sensitivity and specificity.

- *Sensitivity*: $P(\hat{G} = 1 \mid Y = 1)$, the true positive rate
- *Specificity*: $P(\hat{G} = 1 \mid Y = 0)$, the true negative rate (or 1 - false positive rate)

We can construct a **Receiver operating characteristics (ROC)** curve by plotting the false positive rate against the true positive rate for a variety of thresholds

4 Basis expansions and regularizations

Core concept of basis expansions is to expand beyond linear models to linear basis expansions in X:

$$f(X) = \sum_{m=1}^{M} \beta_m h_m(X) \text{ where } h_m(X) : \mathbb{R}^p \to \mathbb{R}$$

Examples include

- $h_m(X) = X_m$, the original linear model
- $h_m(X) = X_j^2$ or $X_j X_k$, polynomial terms to achieve higher-order Taylor expansions
- $h_m(X) = log(X_m)$, $\sqrt{X_m}$, or $\|X\|_2$, nonlinear transformations
- $h_m(X) = I(L_m \leq X_m < U_m)$, indicator functions

4.1 Piecewise polynomials and regression splines

This section assumes X is one-dimensional

Polynomials are limited by their global nature (changes in a far end of the data can impact the entire model). **Piecewise polynomials** divide the domain of X into contiguous intervals, representing f separately in each interval. In order of complexity we have

- Piecewise constant, defined with three basis functions
  - $h_1(X) = I(X < \xi_1)$
  - $h_2(X) = I(\xi_1 \leq X < \xi_2)$
  - $h_3(X) = I(X \geq \xi_3)$
- Piecewise linear, defined with 4 basis functions include the above, plus three additional basis functions, $h_{m+1} = h_m(X)X$
- Piecewise cubic, adding basis functions for the quadratic and cubic terms
- Piecewise linear continuous, defined with 4 basis functions (e.g., incorporating constraints that $f(\xi_1^-) = f(\xi_1^+)$)
  - $h_1(X) = 1$
  - $h_2(X) = X$
  - $h_3(X) = (X - \xi_1)_+$ where t_+ denotes the positive part
  - $h_4(X) = (X - \xi_2)_+$
- Piecewise cubic continuous
- Piecewise cubic first-derivative continuous
- Piecewise cubic second-derivative continuous (cubic spline). There is seldom a reason to go beyond cubic splines unless one is interested in smooth derivatives
  - $h_1(X) = 1$
  - $h_2(X) = X$
  - $h_3(X) = X^2$
  - $h_4(X) = X^3$
  - $h_5(X) = (X - \xi_1)_+^3$
  - $h_6(X) = (X - \xi_2)_+^3$
- Piecewise order-M spline with knots $\xi_j, j = 1, \ldots, K$
  - $h_j(X) = X^{j-1}, j = 1, \ldots M$
  - $h_{m+l}(X) = (X - \xi_l)_+^{M-1}, l = 1, \ldots K$

4.1.1 Additional notes

- Fixed-knot splines are known as **regression splines**. One needs to select the order of the spline, number of knots, and location of knots
- Behavior of polynomal fits tends to be erratic near boundaries. This is exacerbated with regression splines. A **Natural cubic spline** adds additional constraints that the function is linear beyond the boundary knots, which can alleviate this issue, but may increase boundaries at the edge of the data

4.2 Smoothing splines

Smoothing splines are a spline basis method that avoids the knot-selection problem by using a maximal set of knots (i.e., every point is a knot). The complexity of fit is controlled by regularization. It seeks a function $f(x)$ to minimize

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

- The first term measures closeness to the data, while the second term penalizes curvature in the function.
- Note for $\lambda = \infty$ we get back least squares line. For $\lambda = 0$ the data interpolates every point
- Even though this objective is for an infinite class of functions, it can be shown to have an explicit, finite dimensional minimizer: a natural cubic spline with knots at the values of x_i

The solution can be rewritten as $f(x) = \sum_{j=1}^{N} N_j(x)\theta_j$ where $N_j(X)$ are an N-dimensional set of basis functions representing this family of natural splines. The criterion reduces to

$$\begin{aligned} RSS(\theta, \lambda) =& (y - N\theta)^T (y - N\theta) + \lambda \theta^T \Omega_N \theta \\ & \text{where } \{N\}_{ij} = N_j(x_i) \text{ and } \{\Omega_N\}_{jk} = \int N_j''(t) N_k''(t) dt \\ \hat{\theta} =& (N^T N + \lambda \Omega_N)^{-1} N^T y \\ \hat{f}(x) =& \sum_{j=1}^{N} N_j(x) \hat{\theta}_j \end{aligned}$$

4.2.1 Smoothing matrices

A smoothing spline is a linear smoother (i.e., linear operator) because the estimated parameters are a linear combination of the y_i

$$\hat{y} = N(N^T N + \lambda \Omega)^{-1} N^T y = S_\lambda y$$

$S_\lambda y$ is known as the *smoother matrix* that takes a weighted average of y near the target window

- S_λ only depends on X and λ
- Analogous to least squares hat matrix, $H = X(X^T X)^{-1} X^T$ (OLS), $H = X(X^T X + \lambda I)^{-1} X^T$ (Ridge regression)
- S_λ is a symmetric positive semidefinite matrix
- While $H^T H = HH = H$ (idempotent), $S_\lambda^T S_\lambda = S_\lambda S_\lambda \preceq S_\lambda$
- While H has rank p, S_λ has rank N
- $df_\lambda = trace(S_\lambda)$

4.2.2 LOO cross validation for smoothing splines

The leave-one-out CV curve for smoothing splines is approximately unbiased as an estimate of the EPE curve

$$\begin{aligned} CV(\hat{f}_\lambda) &= \sum_{i=1}^{N} (y_i - \hat{f}_\lambda^{(-i)}(x_i))^2 \\ &= \sum_{i=1}^{N} \frac{(y_i - \hat{f}_\lambda(x_i))^2}{(1 - S_\lambda(i,i))^2} \end{aligned}$$

4.2.3 Smoothing spline logistic regression

For a binary class variable, y

$$p(x) = P(y = 1 \mid x) = \frac{e^f(x)}{1 + e^f(x)}$$
$$l(f; \lambda) = \sum_{i=1}^{n} [y_i \log(p(x_i)) + (1 - y_i) \log(1 - p(x_i))] - \frac{1}{2}\lambda \int f''(t)^2 dt$$

Similar argument as before, the optimal function, f can be shown to be a natural cubic spline with knots at the unique values of x_i. This leads to a **nonparametric** logistic regression

4.3 Multidimentional splines

Extension of smoothing splines into more than a single dimension. Suppose $X \in \mathbb{R}^2$ with basis functions $h_{1k}(X_1)$ representing functions of parameter X_1 and $h_{2l}(X_2)$ representing functions of parameter X_2. The $M_1 \times M_2$ dimensional tensor product basis (inner product of functions for each parameter matrices) can be ued for representing a two-dimensional function

$$g(X) = \sum_{j=1}^{M_1} \sum_{k=1}^{M_2} \theta_{jk} h_{1j}(X_1) h_{2k}(X_2)$$

4.4 Regularization and reproducing kernel Hilbert spaces

4.4.1 The kernel trick

The *kernel trick* is useful in scenarios when $M \gg N$. It allows us to write a ridge solution for $\hat{y}$ that only requires inversion of an $N \times N$ matrix (as opposed to an $M \times M$ matrix normally)

$$\text{Objective: } R(\beta) = (y - H\beta)^T (y - H\beta) + \lambda \beta^T \beta \text{ for } f(x) = h(x)^T \beta, H = \{h_j(x_i)\}_{N \times M}$$
$$\text{Solution: } \hat{y} = H(H^T H + \lambda I)^{-1} H^T y \text{ requiring inverse of MxM matrix}$$
$$\text{Rewritten solution: } \hat{y} = HH^T (HH^T + \lambda I)^{-1} Hy \text{ requiring inverse of NxN matrix}$$

Proof: Starting with the derivative condition

$$\frac{\partial R}{\partial \beta} = 0 \iff -2H^T(y - H\beta) + 2\lambda\beta = 0$$
$$\hat{\beta} = H^T(y - H\hat{\beta}) = H^T\alpha$$
$$HH^T(y - HH^T\alpha) = \lambda HH^T\alpha, \text{ plugging in and left multiplying by } H$$
$$\alpha = (HH^T + \lambda I)^{-1} y$$
$$\hat{\beta} = H^T\alpha = H^T(HH^T + \lambda I)^{-1} y$$

Note that HH^T is $N \times N$ and the inner product of all $h(x_i)^T h(x_j)$. For a function $K(x, x') = h(x)^T h(x')$, we can let $K = HH^T$ and get

$$\hat{y} = h(x)^T \beta = \sum_{i=1}^{N} \hat{\alpha}_i K(x, x_i)$$
$$\hat{\alpha} = (K + \lambda I)^{-1} y$$
$$\hat{\beta}^T \hat{\beta} = \hat{\alpha}^T K \hat{\alpha}$$

4.4.2 Kernel examples

- Polynomial kernel: $K(x, x') = (1 + x^T x')^d$
- Radial kernel: $K(x, x') = \exp(-\gamma \|x - x'\|_p^2)$; γ controls the width of the kernel

$f(x)$ in turn is a weighting of all these kernels at each point x.