Notes

- Homework 3 due on Wednesday, March 9
- Friday's class for answering questions re homework
- No office hour this Friday
- HW 2, Question 2: don't need (can't) solve explicitly for $\hat{\beta}$
- HW 2, Question 3: don't need to divide into training and test error
- HW 2, Question 4: I thought smoothing was on the $\hat{\beta}$'s (256 of them), but it is perhaps on the x's directly

- "local" in R^1 is quite different that local in R^p
- Example: suppose each feature variable is uniformly distributed on $(0, 1)$. If we want 10% of the sample in R^1, we need a window of length 0.1. In R^p, to get 10% of the volume, we need a box with edge length $0.1^{1/10} = 0.80$, so on each axis we need a window of length 0.8.
- Example: N data points uniformly distributed on a unit ball in R^p. What is the distance to the nearest neighbour to the origin? Median is
- $(1 - 0.5^{1/N})^{1/p}$ ≈ 0.52 if $p = 10, N = 500$.
- Figures 2.6 and 2.7

- extend smoothing methods to p inputs, avoiding the curse of dimensionality
- retain interpretation, computationally feasible
- a 'linear' additive model has the form $E(Y \mid X_1, \ldots, X_p) = \mu(\underline{X}) = \alpha + \sum_{j=1}^{p} f_j(X_i)$
- f_j unspecified 'smooth' function, e.g. a smoothing spline, or a kernel regression function
- a 'generalized additive model' has the form $g\{\mu(\underline{X})\} = \alpha + \sum_{j=1}^{p} f_j(X_i)$
- examples of $g(\cdot)$ are $g(\mu) = \log\{\mu/(1-\mu)\}$ for binomial proportions, $g(\mu) = \log \mu$ for Poisson data, etc. (builds on generalized linear models as discussed in STA 410)

Fitting 'linear' additive models (§9.1.1.)

- first constrain f_j so that $\sum_{i=1}^{N} f_j(x_{ij}) = 0, \quad j = 1, \ldots, p$
- penalized residual sum of squares
  $\sum_{i=1}^{N}\{y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij})\}^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t)^2 dt$
- backfitting: (Algorithm 9.1)
  - $\hat{\alpha} = \frac{1}{N}\sum_{i=1}^{N} y_i, \quad \hat{f}_j \equiv 0, \quad j = 1, \ldots, p$
  - cycle through $j = 1, \ldots, p, 1, \ldots, p$ until convergence:

$$\begin{aligned}
\hat{y}_i &\longleftarrow y_i - \hat{\alpha} - \sum_{k \neq j}^{p} \hat{f}_k(x_{ik}) \quad i = 1, \ldots, N \\
\hat{f}_j &\longleftarrow \mathcal{S}_j(y_i, x_{ij})_{i=1}^{N} \\
\hat{f}_j &\longleftarrow \hat{f}_j - \frac{1}{N}\sum_{i=1}^{N} \hat{f}_j(x_{ij})
\end{aligned}$$

- the last step to enforce the constraint in the presence of roundoff error

Fitting ‘linear’ additive models (§9.1.1.)

- first constrain f_j so that $\sum_{i=1}^{N} f_j(x_{ij}) = 0, \quad j = 1, \ldots, p$
- penalized residual sum of squares $\sum_{i=1}^{N} \{y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij})\}^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t)^2 dt$
- backfitting: (Algorithm 9.1)
  - $\hat{\alpha} = \frac{1}{N} \sum_{i=1}^{N} y_i, \quad \hat{f}_j \equiv 0, \quad j = 1, \ldots, p$
  - cycle through $j = 1, \ldots, p, 1, \ldots, p$ until convergence:

$$\begin{aligned} \hat{y}_i &\longleftarrow y_i - \hat{\alpha} - \sum_{k \neq j}^{p} \hat{f}_k(x_{ik}) \quad i = 1, \ldots, N \\ \hat{f}_j &\longleftarrow \mathcal{S}_j(y_i, x_{ij})_{i=1}^{N} \\ \hat{f}_j &\longleftarrow \hat{f}_j - \frac{1}{N} \sum_{i=1}^{N} \hat{f}_j(x_{ij}) \end{aligned}$$

- the last step to enforce the constraint in the presence of roundoff error

– replacement y_i is the residual from the current fit

– fitting to the residuals is analogous to multiple regression by successive partial regressions (§3.3)

– iteratively reweighted least squares used for logistic regression in §4.4.1

Fitting 'linear' additive models (§9.1.1.)

- $\mathcal{S}$ is a cubic smoothing spline if we use penalized residual sum of squares
- could in principle be any smoothing operation: e.g. natural cubic spline or other regression spline, kernel regression function, `loess`, etc.
- claim (p.260) solution is unique if (x_{ij}) matrix of full column rank (theory project)
- can allow some terms to be ordinary linear regression terms; no smoothing needed
- can (in principle) allow some terms to be smooth functions of, e.g., pairs of inputs
- degrees of freedom for smoother $\mathcal{S}_j$ is $df_j = \mathrm{trace}(S_j) - 1$, where S_j is the $N \times N$ operator matrix (correctness of this still open)
- for generalized additive models, goal is to maximize the penalized log-likelihood, not minimize the penalized residual sum of squares

- combine the iteratively reweighted least squares algorithm for generalized linear models with Algorithm 9.1 (backfitting)
- model

$$\log \frac{\Pr(Y=1 \mid X)}{\Pr(Y=0 \mid X)} = \alpha + f_1(X_1) + \cdots + f_p(X_p)$$

  - initialize: $\hat{\alpha} = \log(\bar{y})/(1-\bar{y})$, where $\bar{y}$ is sample mean (proportion of 1's); $\hat{f}_j \equiv 0$
    - $\hat{\eta}_i = \hat{\alpha} + \sum_j \hat{f}_j(x_{ij}), \quad \hat{\mu}_i = 1/\{1+\exp(-\hat{\mu}_i)\}$
    - working variable $z_i = \hat{\eta}_i + \frac{y_i - \hat{\mu}_i}{\hat{\mu}_i(1-\hat{\mu}_i)}$
    - working weight $\hat{w}_i = \hat{\mu}_i(1-\hat{\mu}_i)$
    - use Alg. 9.1 to fit an additive model to z_i with weights w_i
  - update $\hat{\eta}_i$ and continue until convergence

```
> hr.gam <- gam(chd~s(sbp)+s(tobacco)+s(ldl)+famhist+s(obesity

> hr.gam$coef
   (Intercept) famhistPresent        s(sbp).1        s(sbp).2
 -1.314442e+00   9.479920e-01   3.882599e-01   2.927006e-02
       s(sbp).3        s(sbp).4        s(sbp).5        s(sbp).6
 -2.103390e-02   9.864767e-03  -6.265013e-03  -2.081766e-03
       s(sbp).7        s(sbp).8        s(sbp).9       s(sbp).10
  3.421593e-03   2.544205e-04  -3.386699e-01   1.413812e-01
  s(tobacco).1    s(tobacco).2    s(tobacco).3    s(tobacco).4
  4.730698e-10   2.510070e-11   1.280031e-11  -2.759787e-11
  s(tobacco).5    s(tobacco).6    s(tobacco).7    s(tobacco).8
 -5.695848e-11  -8.963317e-12  -2.058901e-12   1.842881e-12
  s(tobacco).9   s(tobacco).10        s(ldl).1        s(ldl).2
 -4.983894e-10   3.713993e-01   5.901141e-08  -1.337292e-10
       s(ldl).3        s(ldl).4        s(ldl).5        s(ldl).6
 -7.556864e-09   8.533137e-10  -2.491124e-10   4.035828e-10
       s(ldl).7        s(ldl).8        s(ldl).9       s(ldl).10
  7.538201e-11   1.255975e-10   5.284967e-08   4.021833e-01
```

```
> hr.gam$sp
[1] 1.386301e+04 7.306222e+10 1.076798e+08 6.958563e+01 1.6240
[6] 5.315837e+01

> plot.gam(hr.gam)

> hr.gam

Family: binomial
Link function: logit

Formula:
chd ~ s(sbp) + s(tobacco) + s(ldl) + famhist + s(obesity) + s(
    s(age)

Estimated degrees of freedom:
 1.713315 1 1.000000 2.097546 1 3.984891   total =  12.79575

UBRE score:  0.02242598
```

- if estimated degrees of freedom near 1, then linear fit is satisfactory (e.g. tobacco)
- if the plotted confidence band includes zero, then term is not needed (e.g. alcohol)
- generalized cross-validation is used to estimate each smoothing parameter (Wood, 2001)
- this is not exactly the same as the back-fitting algorithm described in the text, but seems to be more reliable
- UBRE is a version of generalized cross-validation recommended for binomial data
- from Wood (2001): if the GCV score drops when a term is omitted, and the confidence band includes zero, then the term is not needed in the model
- the default is 10 knots for a smooth term if unspecified

- each smooth can be specified as
  - `s(x)`: default 10 knots + penalty for smoothness
  - `s(x,k=5,fx=TRUE)` or `s(x,5|f)`: force 5 knots (4 df) for *x*, no shrinkage permitted
  - `s(x,20)`: start with 20 knots (maximum 19 df), and choose fewer by GCV
- the default basis is the *thin plate* basis; to get cubic regression splines use argument `bs="cr"`

- example in text uses “spam” data from UC Irvine: 4601 instances, training set of size 3065 (indicators available on web site)
- binary response (1=spam, 0= not spam), cubic splines, 4df per predictor
- Table 9.1:

	predicted class	
true class	email	spam
email	58.5%	2.5%
spam	2.7 %	36.2%

- linear logistic regression has test error of 7.6%

- Table 9.2:

Name	num.	df	coef	se	Z	nonlinear *P*-value
			Positive effects			
`our`	5	3.9	0.566	0.114	4.970	0.052
`over`	7	3.9	0.244	0.195	1.249	0.004
`CAPMAX`	56	3.8	0.247	0.228	1.080	0.000
`CAPTOT`	47	4.0	0.755	0.165	4.566	0.063

- see Figure 9.1

- §9.2.1, 9.2.2 regression trees (y is continuous)
- formalism: $\hat{f}(x) = \sum_{m=1}^{M} c_m 1\{x \in \mathcal{R}_m\}$
- $\mathcal{R}_m$ is a subspace of R^p obtained by partitioning the feature space using binary splits
- if $\mathcal{R}_m$ is fixed, then the optimal choice of c_m to minimize $\sum\{y_i - f(x_i)\}^2$ is just $\text{ave}(y_i \mid x_i \in \mathcal{R}_m)$
- trees are 'grown' in a greedy fashion, starting with any node and finding the variable to split on $X_j, \quad j = 1, \ldots, p$ and the split point s
- to minimize squared error after splitting

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right]$$

- $R_1(j,s) = \{X \mid X_j \leq s\}, \qquad R_2(j,s) = \{X \mid X_j > s\}$
- $\hat{c}_1 = \text{ave}\{y_i \mid x_i \in R_1(j,s)\}, \quad \hat{c}_2 = \text{ave}\{y_i \mid x_i \in R_2(j,s)\}$

- trees are grown to be quite large and then pruned, using a cos-complexity criterion
- $C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T|$ (9.16)
- $Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2, \quad \hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$
- $|T|$ is number of terminal nodes
- $C_\alpha(T)$ trades off fit to data Q_m and tree size T
- For each α there is a pruning strategy
- Choose α by 5 or 10 fold CV
- see Figure 9.5 for a classification tree
- more on trees and MARS on March 16
- next week neural networks (Ch 11)