Stanford Statistics I

- **Author : Yunqiu Xu**

`Statistics` `MachineLearning` `R` `DataAnalysis` `DataMining`

Chapter 1 Introduction

- surpervised learning problem : need to get the response(Y) to make predictions
  - (Xn,Yn)-(examples,instances)
  - regression problem-Y is quantitive (e.g. price)
  - classification problem-Y is finite and unordered (e.g. c(1:9))
- unsupervised learning: no need to make a prediction
  - PCA/K-means clustering/Hierarchical clustering
  - No outcome variable, just a set of preditors(features) mesured on samples
  - Finding samples/features with similar behavior
  - Finding linear conbinations of features with the most variation
  - Difficult to know how well you are doing (no fixed standard)
  - Can be used as a pre-processing step for supervised learning
- Stastical Learning VS Machine Learning
  - SL: statistics,models,precision,uncertainty
  - ML: AI,large scale,prediction accuracy

Chapter 2 Overview of Statistical Learning

2.1 Introduction to Regression Models

2.1.1 some notations

- X : feature,input,preditor
- $X_1, X_2 \dots X_N$: input vector

- Y : a response or target we wish to predict
- $Y = f(X) + e$: e means errors and other diesrepancies
- $f(4) = E(Y|X = 4)$: the expected values of Y given X=4
- $f(x) = E(Y|X = x)$: the regression function of Y, in this ideal case, X contains all points of x

2.1.2 the regression function `f(x)=E(Y|X=x)`

- A 3-D regression function:

$$f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$$

- irreducible error

$$e = Y - f(x)$$

- reducible error + variance

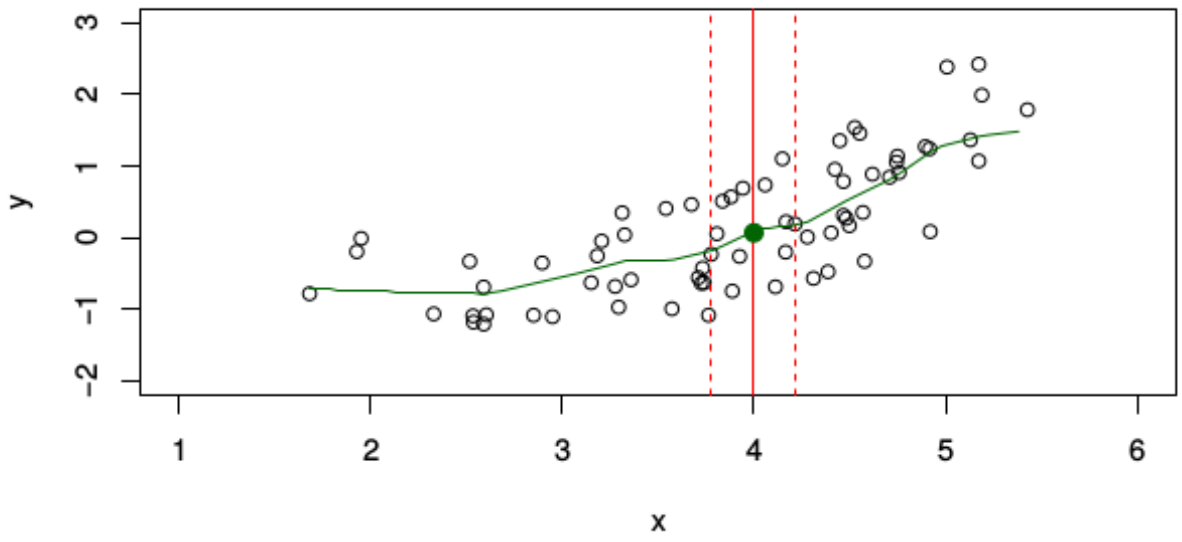
$$E[(Y - \hat{f}(X))^2|X = x] = \underbrace{[f(x) - \hat{f}(x)]^2}_{Reducible} + \underbrace{\mathrm{Var}(\epsilon)}_{Irreducible}$$

- 不可消除误差与模型无关，无论你怎么折腾都不会变

2.1.3 Estimate f--Nearest neighbor averaging

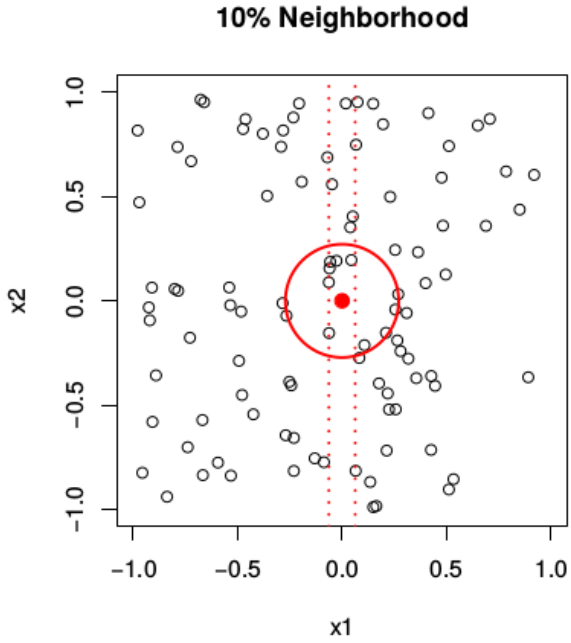
$$f'(x) = Ave(Y|X \in \mathbf{N}(x))$$

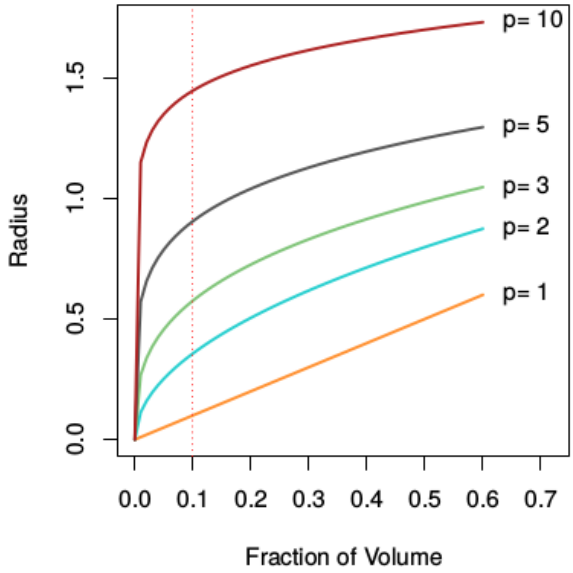
- N(x)为x点近邻区间
- y在x点没有准确值，只能尽可能取趋向这一点的近似值
- 估计值y': 求取X在x点邻近的Y，然后取这些y值得平均值



2.2 Dimensionality and Structured Models

2.2.1 维数灾难curse of dimensionality





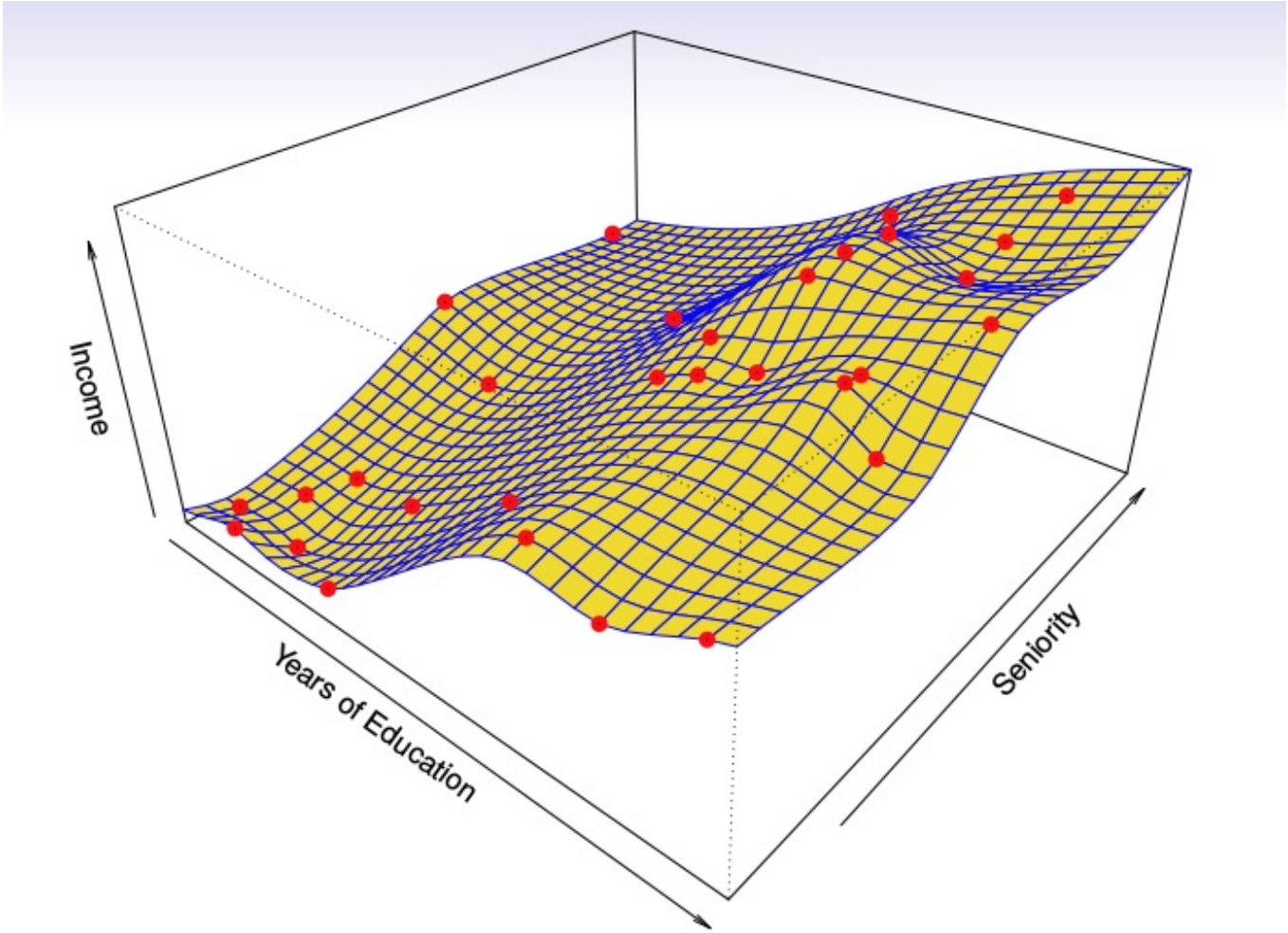
- 近邻平均值适合p<4(参数数/维数)的情况
- 维数灾难:每多一个参数，可能出现的问题和不确定性就会增加，如(Var变大)
- 维数灾难其实就是over-fitting,需要进行降维

2.2.2 Parametric and structured models

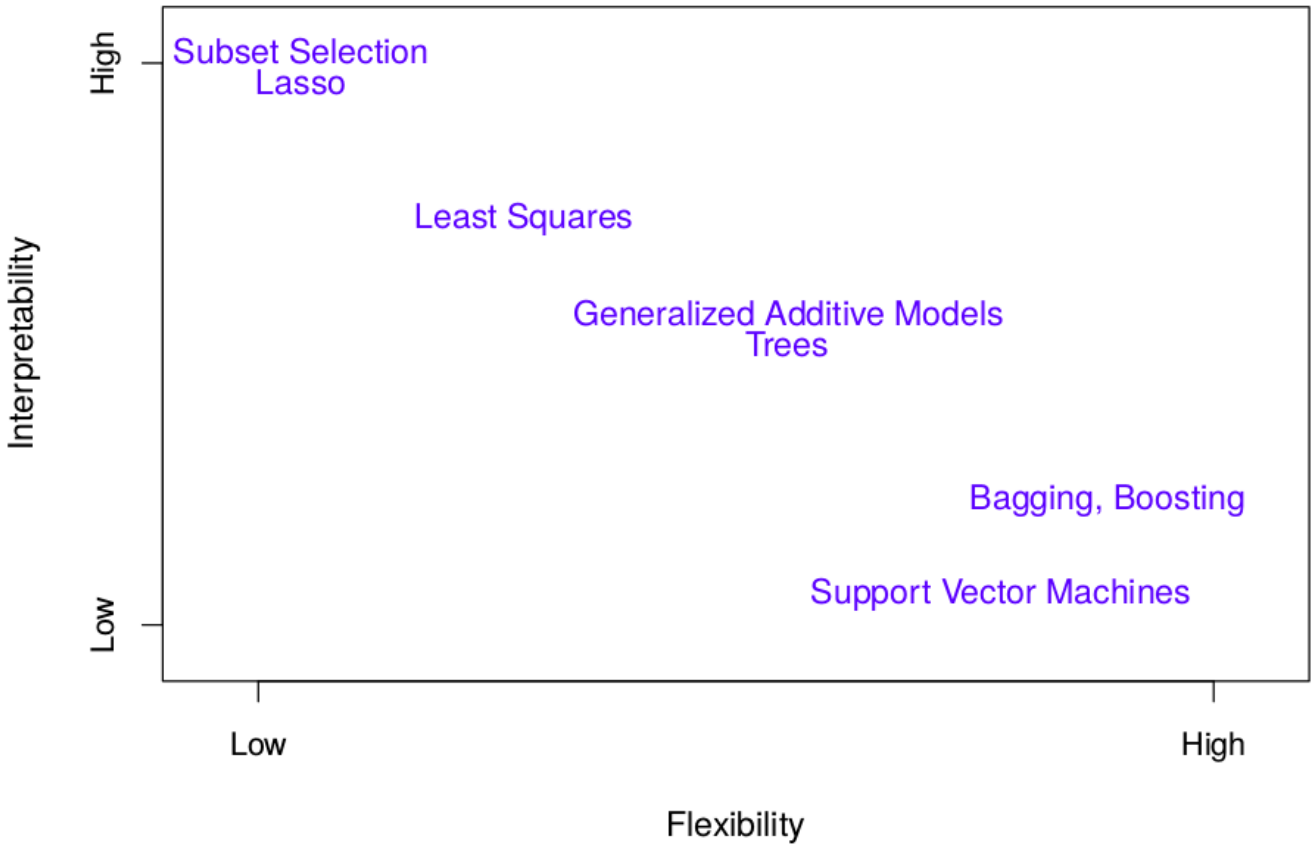
- p+1个参数的线性模型

$$f(x) = a_0 + a_1 X_1 + a_2 X_2 + \ldots + a_p X_p$$

- quadratic model 二次模型
- thin-plate spline 薄板样条，可用于控制roughness(chapter 7)
- overfitting 过度拟合: training error趋近于完美，而 error则表现糟糕



2.2.3 Trade-offs



- Prediction accuracy VS interpretability
  - linear models/decision tree-easier to interpret
  - thin-plate splines-more accurate
- Good fit VS over/under fit
- Parsimony(简化模型，尽可能少变量) VS black-box(尽可能多的变量)

2.3 Model Selection and Bias-Variance Trade-off

2.3.1 Assessing Model Accuracy

- AVE : average squared prediction error均方预测误差
- MSE : mean squared error均方误差
- Tr : training data

$$MSE_{Tr} = Ave_{i \in Tr}[y_i - f'(x_i)]^2$$

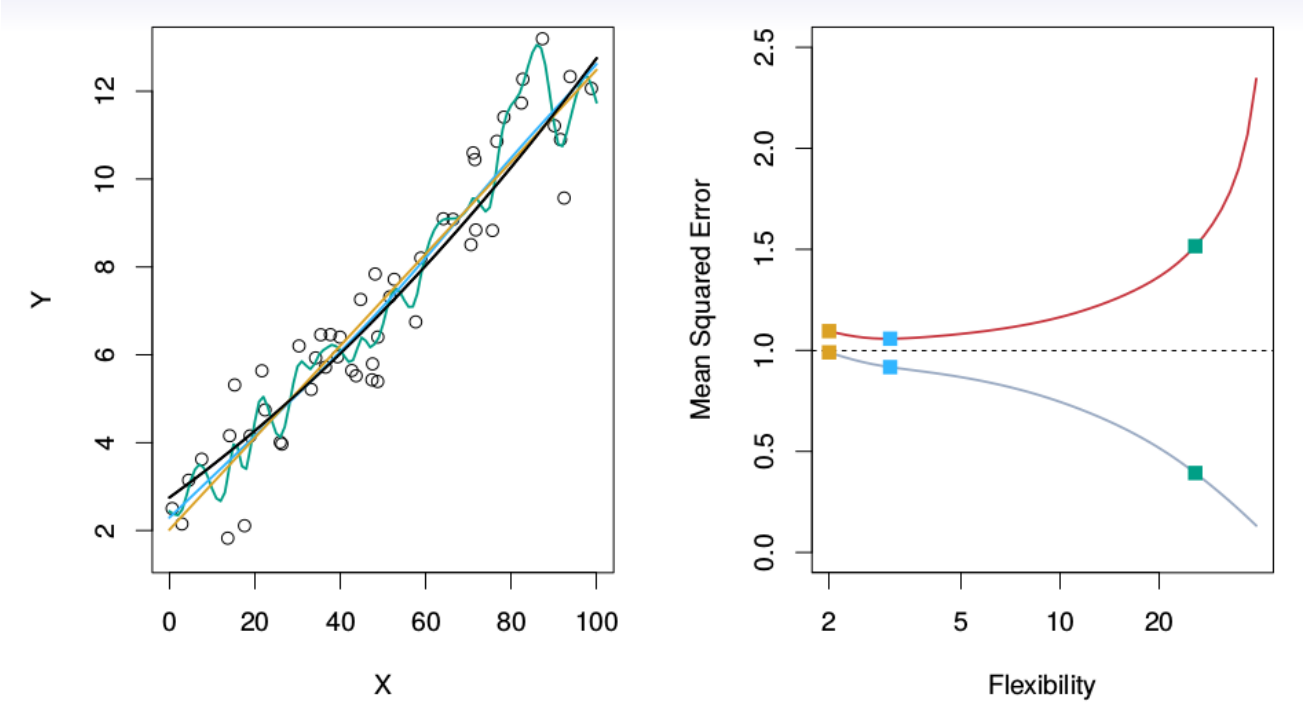
- Te : test data

$$MSE_{Te} = Ave_{i \in Te}[y_i - f'(x_i)]^2$$

- **Tr可能导致过度拟合，最终目标是尽可能优化Te**

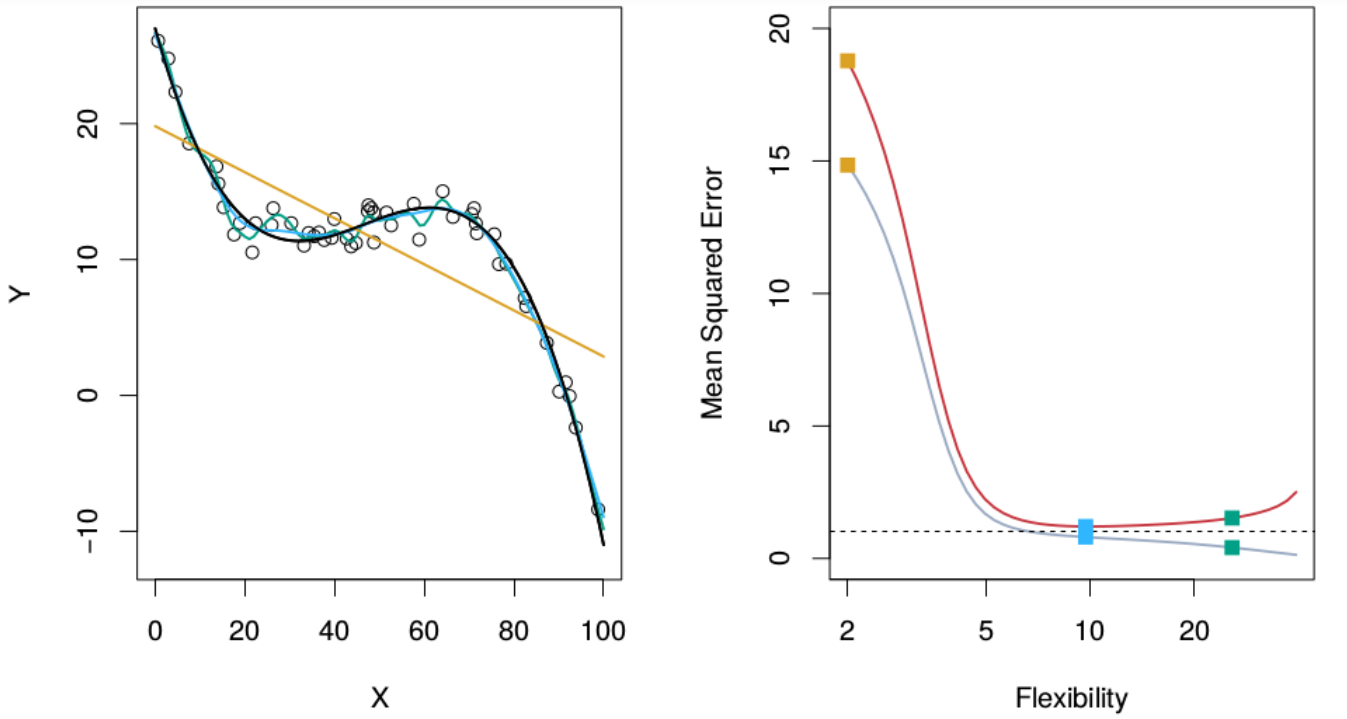
2.3.2 几个栗子

- 概念:
  - 黑线: Truth
  - 右图红线: MSE_{T_e}
  - 右图灰线: MSE_{T_r}
  - 左图不同颜色的线: 不同fit model
- 线性模型:



Here the truth is smoother, so the smoother fit and linear model do really well.

- 非线性模型:



Here the truth is wiggly and the noise is low, so the more flexible fits do the best.

2.3.3 Bias-Variance Trade-off

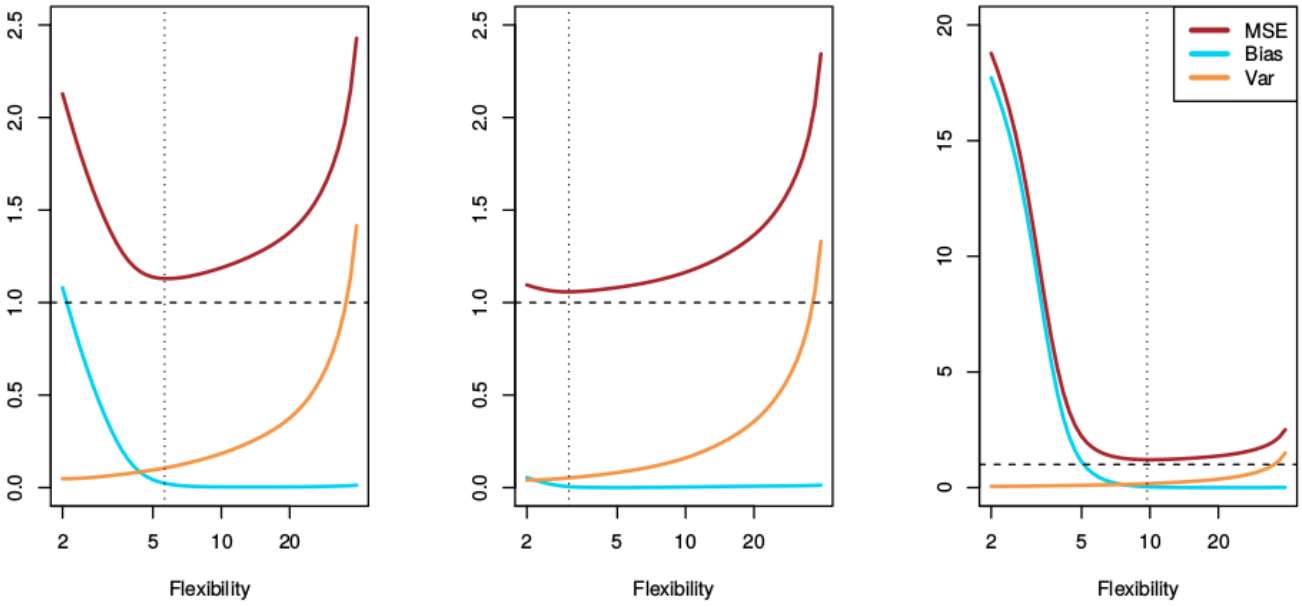
- True Model:

$$Y = f(X) + e, \quad f(x) = E(Y|X = x)$$

- Error:

$$E(y_0 - f'(x_0))^2 = Var(f'(x_0)) + [Bias(f'(x_0))]^2 + Var(e)$$

$$Bias(f'(x_0)) = E[f'(x_0)] - f(x_0)$$



- Bias：与真实值的偏差，用于量度精密度
  - Underfitting --> High Bias
- Variance：与期望的偏差，用于量度稳定性
  - Overfitting --> High Variance
- 这两个都是预测因子误差,距离truth越远,variance越小而bias越大
  - 例如预测因子越少，模型越简单，var越低，bias越高
  - 对于f'(x), flexibility↑(variable↑), variance↑, bias↓
- $Var(f'(x_0))$: 使用不同训练集带来的Var
- Bias相当于不同训练集中y_0预测值的平均数, 即真实的y_0
- Var(e) : irreducible error
- The type of error : P/N为预测结果，T/F为预测结果是否与实际相同

Error type	实际结果	预测结果
True positive	1	1
True negative	0	0
False positive	0	1
False negative	1	0

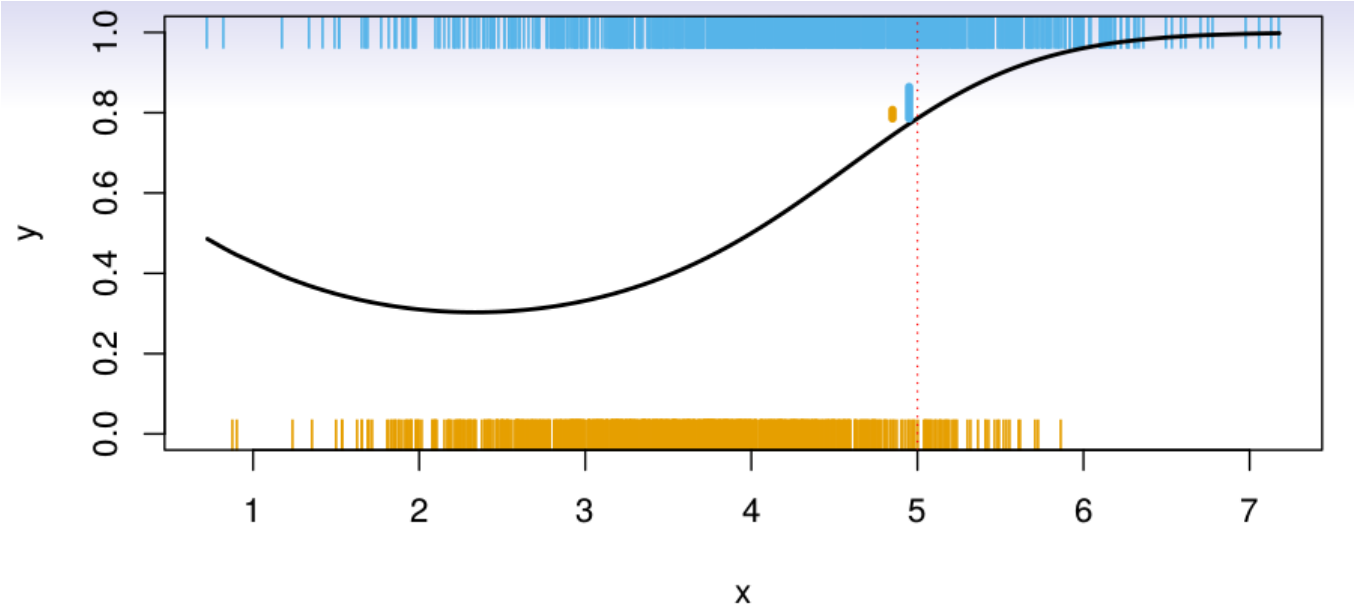
- 灵敏度Sensitivity/特异度specificity/准确度Precision/召回率Recall

$$Sensitivity(Recall) = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

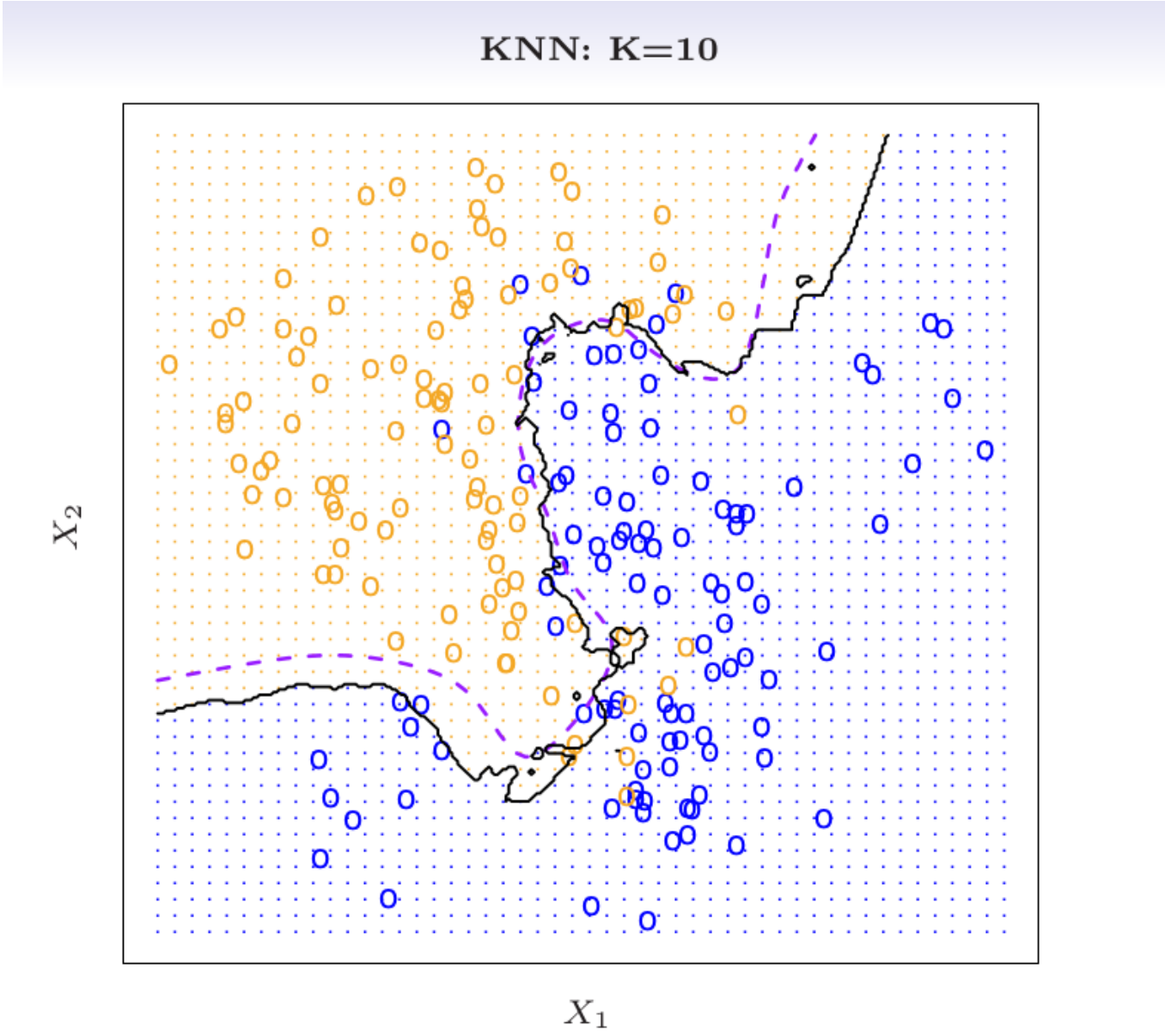
$$Precision = \frac{TP}{TP + FP}$$

2.4 分类问题classification

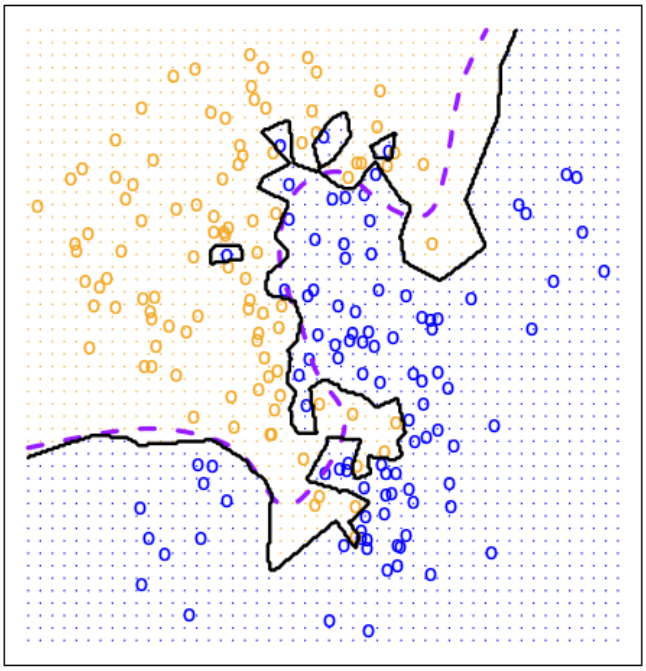


- 举个栗子: 通过构建一个分类器C(0:K), 对邮件进行分级，并评估不同分级方法的不确定性
  - X的条件概率
    $$p_k(x) = Pr(Y = k|X = x), k = 1, 2, \dots K$$
  - x的贝叶斯最优分类Bayes optimal classifier
    $$C(x) = j \text{ if } p_j(x) = \max\{p_1(x), p_2(x), \dots, p_K(x)\}$$
- Classification: some details
  - we measure the C'(x) by misclassfication error rate(Err):
    $$Err_{T_e} = Ave_{i \in T_e} I[y_i \neq C'(x_i)]$$
  - 使用了正确条件概率$p_k(x)$的贝叶斯分类器具有最小的error

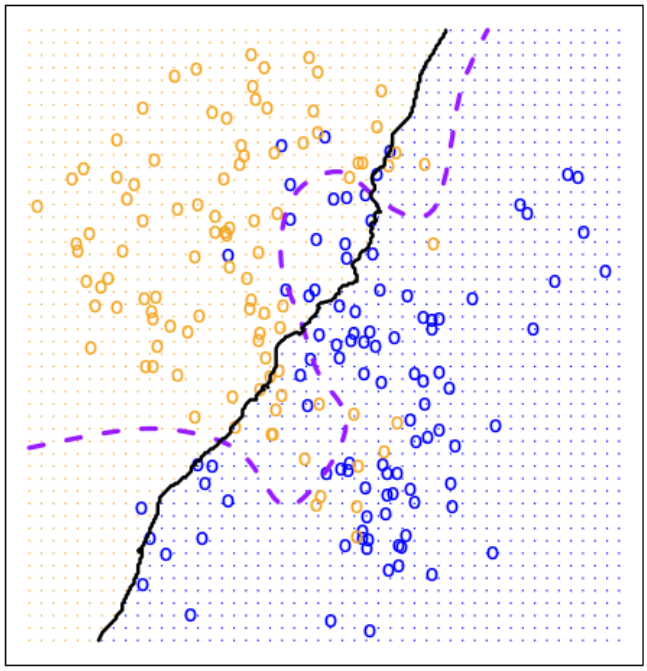
- 举个栗子: K最近邻算法(KNN)
  - 思路：如果一个样本在特征空间中的k个最临近(即最相似)的样本中的大多数属于某一个类别，则该样本也属于这个类别
  - KNN算法中所选择的邻居都已经被正确分类过了
  - 当K的范围不同时，边界也会有差异



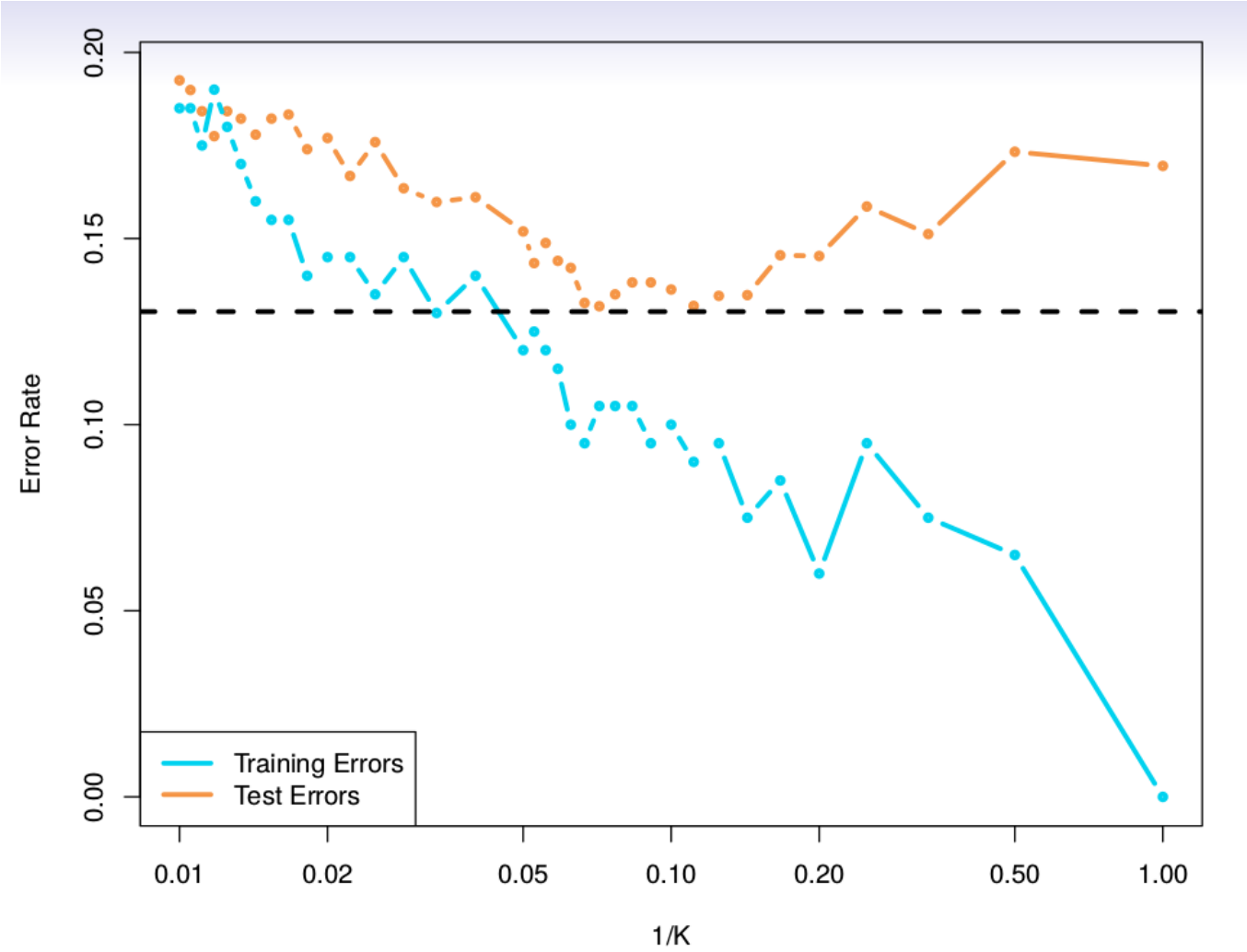
KNN: K=1



KNN: K=100



+ 误差评估:



2.5 Quiz

- A flexible model will allow us to take full advantage of our large sample size 样品量大，灵敏度高的模型更容易发现问题
- The flexible model will cause overfitting due to our small sample size 样品量小但变量太多会导致过度拟合
- A flexible model will be necessary to find the nonlinear effect 非线性需要更大的灵敏度以发现问题
- A flexible model will cause us to fit too much of the noise in the problem 如果方差本身就很大，灵敏度提高会增加误差与噪音

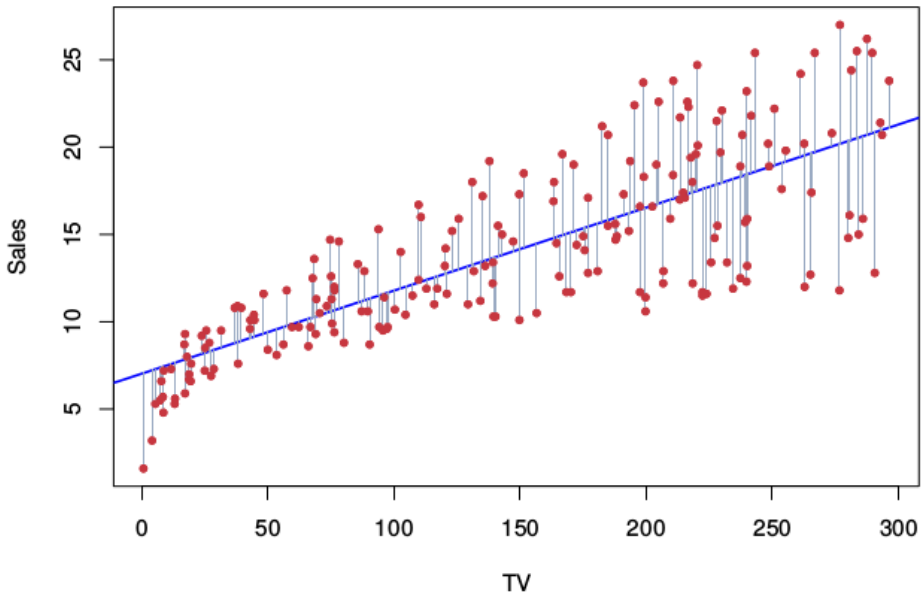
Chapter 3 Linear Regression

3.1 Simple Linear Regression#Supervised Learning

3.1.1 Linear model :

- $y = \beta_0 + \beta_1 x + e$
  - β_0 : intercept截距
  - β_1 : slope斜率
  - e : error
  - estimated $y' = \beta_0' + \beta_1' x$

3.1.2 通过最小二乘法least square来估测parameters



The least squares fit for the regression of `sales` onto `TV`.
In this case a linear fit captures the essence of the relationship, although it is somewhat deficient in the left of the plot.

- residual残差

$$e_i = y_i - \beta_0' - \beta_1' x_i = y_i - y_i'$$

- residual sum of squares(RSS)残差平方和:

$$RSS = e_1^2 + e_2^2 + \ldots + e_n^2$$

- 最小二乘法 : 通过选择合适的β_0', β_1',最小化RSS

$$\beta_1' = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$\beta_0' = \bar{y} - \beta_1' \bar{x}$$

3.1.3 使用standard error来量度[精密度+抽样误差]

- SE VS SD:
  - SE 用于量度精密度/抽样误差
  - SD 用于量度离散度
  - 离散度SD小,但是样本与均值差别大,则SE小
  - SE越小,抽样误差越小,表明抽取的样本越能较好代表总体

$$SE(\beta_1')^2 = \frac{Var(e)}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$SE(\beta_0')^2 = Var(e)[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}]$$

3.2 Hypothesis Testing and Confidence Intervals

3.2.1 confidence intervals

- 置信区间 : 指该范围内的数据有95%的可信概率,允许5%的偏差

$$\beta_1' \pm 2SE(\beta_1')$$

3.2.2 假设检验Hypothesis testing

- null hypothesis(零假设): H_0 VS H_A
  - H_0: X与Y无关 ($\beta_1 = 0 \rightarrow Y = \beta_0 + e$)
- H_A : alternative hypothesis (择一假设/备择假设) X与Y有一定关联
  - $\beta_1 \neq 0$

3.2.3 使用t统计量t-statistic来检验null hypothesis 样品与均值的差异

- T统计量 : 对回归模型个体系数的显著性进行假设检验

$$t = \frac{\beta_1' - 0}{SE(\beta_1')}$$

- |t|越大即SE越小：样品偏离均值越远,样品与均值的差别越显著(抽样误差越小)
- p-value假设检验中的p值
  - p>0.05 两组差别无意义
  - p<0.05 显著意义
  - p<0.01 两组差别有非常显著的意义

	Coefficient	Std. Error	t-statistic	p-value
Intercept	7.0325	0.4578	15.36	< 0.0001
TV	0.0475	0.0027	17.67	< 0.0001

3.2.4 Overall Accuracy of the Model

- RSE(Residual Standard Error)标准化残差
  - RSS越小，样本量越大，RSE越小

$$RSE = \sqrt{\frac{RSS}{n-2}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - y_i')^2}$$

- R-squared [拟合度/整体精密度/相关度]

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

  - R^2又称为可决系数
  - R^2 大, 模型与样本观察值拟合度好，总体精密性高
- TSS 总平方和 total sum of squares

$$TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

- r-X与Y的相关度
  - r变化范围[-1,1]，|r|越大相关度越强,正相关或负相关

$$r = \sqrt{(R^2)}$$

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Quantity	Value
Residual Standard Error	3.26
R^2	0.612
F-statistic	312.1

3.3 Multiple Linear Regression多元线性回归

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + e$$

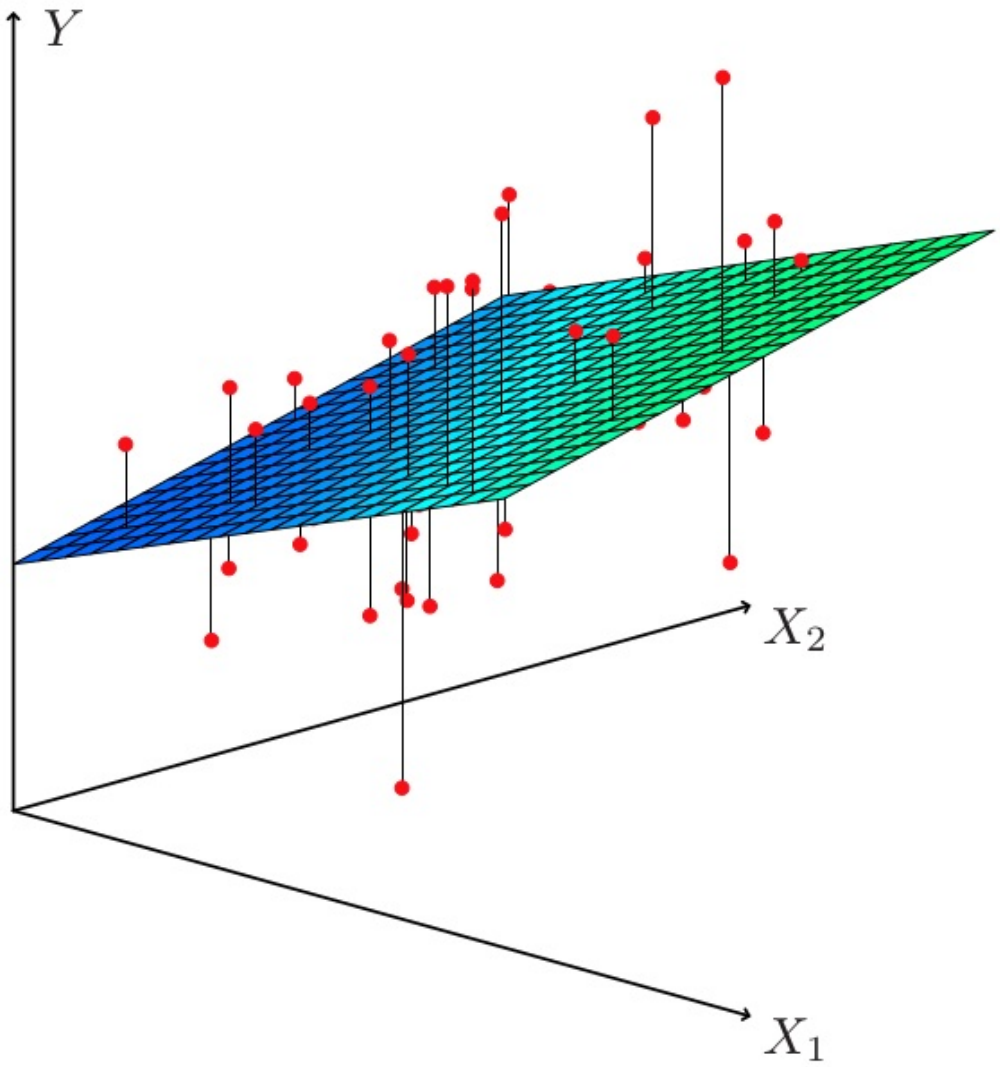
$$\texttt{sales} = \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{radio} + \beta_3 \times \texttt{newspaper} + \epsilon.$$

3.3.1 对回归系数的解释

- 理想的方案应该是所有预测因子互不相关的-a balanced design
- 相关的预测因子会增加方差
- 当某一个X变化时其他预测因子应该保持不变 : 控制变量
- 观测数据时应尽可避免因果性claims of causality

3.3.2 Estimation and Prediction for Multiple Regression

- 同样需要最小化RSS, 只是y的表达式变成了多元的



- 结果分析:

	Coefficient	Std. Error	t-statistic	p-value
Intercept	2.939	0.3119	9.42	< 0.0001
TV	0.046	0.0014	32.81	< 0.0001
radio	0.189	0.0086	21.89	< 0.0001
newspaper	-0.001	0.0059	-0.18	0.8599

Correlations:

	TV	radio	newspaper	sales
TV	1.0000	0.0548	0.0567	0.7822
radio		1.0000	0.3541	0.5762
newspaper			1.0000	0.2283
sales				1.0000

3.4 Some Important Questions

3.4.1 预测因子数对预测的影响 F-statistic

$$F = \frac{\frac{TSS-RSS}{p}}{\frac{RSS}{n-p-1}} \sim F(p, n-p-1)$$

- F可用于检验整体显著性
- F越大,残差越小,整体显著性越好

3.4.2 优化预测因子数 (可参考6.1)

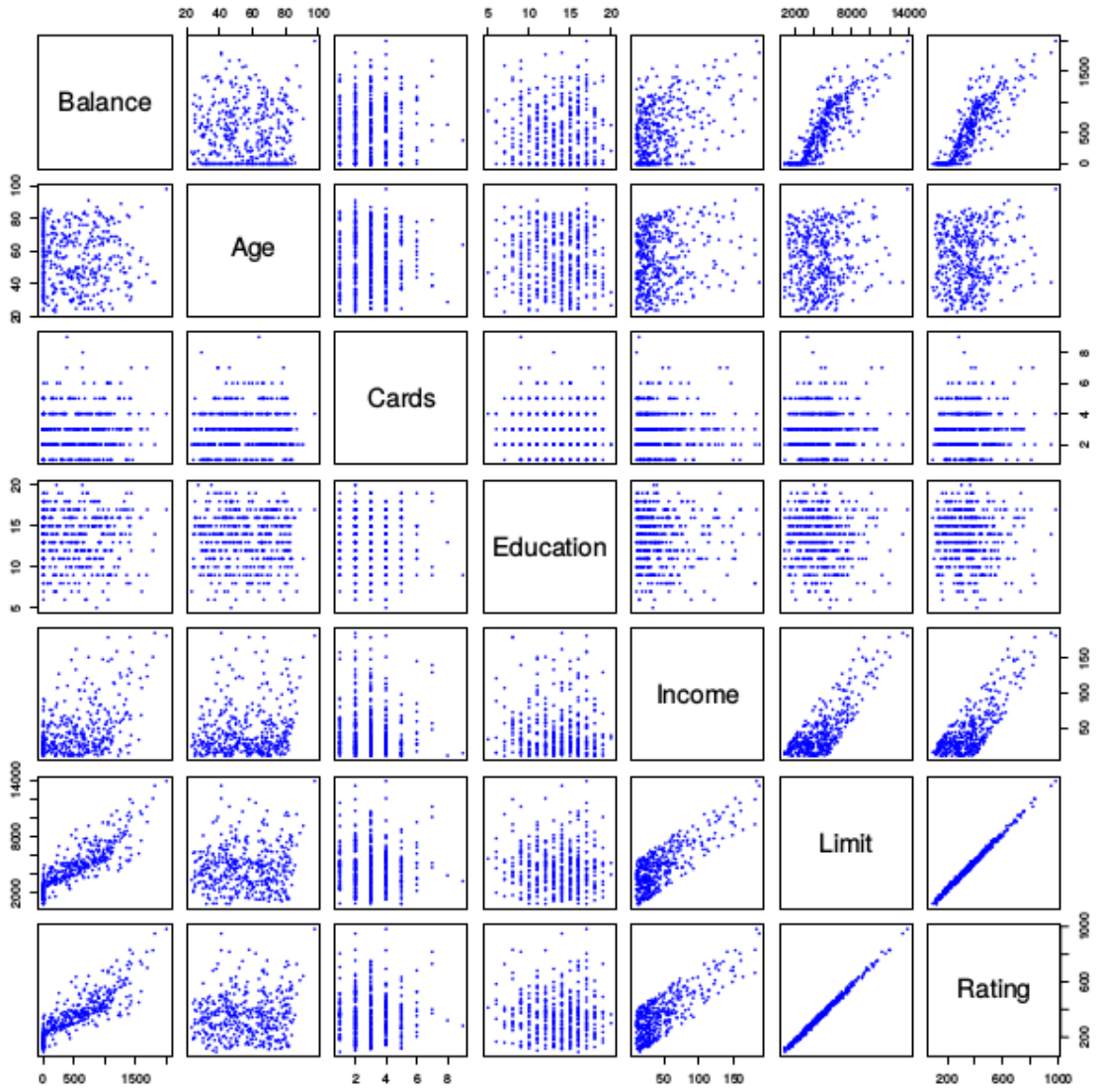
- 确定有效变量(预测因子)数，并移除无用的预测因子
- Subsets selection 不适合预测因子过多的情形
- Forward selection 变量越加越多
  - 始于零模型null model(1为截距intercept,0为预测因子predictor)
  - 拟合p simple 线性回归，并将可以最小化RSS的变量加入null model
  - 将适合二元模型的可以最小化RSS的变量加入null model
  - 继续这一多元化过程直到满足停止条件,如p值大于某个指标
- Backward selection 变量越减越少
  - 始于一个拥有所有变量的模型
  - 删除p value最大的变量(统计学意义最不显著的那个)

    - 继续按照p value删除变量,直到满足停止条件如p值小于某个指标
- 其他的一些可以优化F/B选择法的标准:
    - Mallow’ s Cp,
    - Akaike information criterion(AIC)
    - Bayesian information criterion (BIC)
    - Adjusted R square
    - Cross-validation (CV)

3.4.3 Qualitative Predictors 定性的预测因子

- 一些变量是定性而非定量的,其值属于分离的数据集 (性别,婚姻状况,民族等)
- 定性的预测因子又被称为categorical predictors/factor variables因子变量
- 对于这些变量一般要给予其权重赋值,如c(1,0)
- 栗子: 调查信用卡余额(Balance)和用户性别的关系

Credit Card Data



$$x_i = \begin{cases} 1 & \text{if } i\text{th person is female} \\ 0 & \text{if } i\text{th person is male} \end{cases}$$

Resulting model:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th person is female} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is male.} \end{cases}$$

- 结果:

	Coefficient	Std. Error	t-statistic	p-value
Intercept	509.80	33.13	15.389	< 0.0001
gender[Female]	19.73	46.05	0.429	0.6690

3.5 Extensions of the linear model

3.5.1 Interaction协同

- 举个栗子:销量与电视广告和收音机广告的关系中，电视广告和收音机广告是协同的
- 解决措施：引入一个合并项，来观测两个协同项的关系

$$\widehat{\texttt{sales}} = \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{radio} + \beta_3 \times \texttt{newspaper}$$

states that the average effect on sales of a one-unit increase in TV is always β_1, regardless of the amount spent on radio.

- 观测结果：
- 完整的栗子:

Model takes the form

$$\begin{aligned} \texttt{sales} &= \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{radio} + \beta_3 \times (\texttt{radio} \times \texttt{TV}) + \epsilon \\ &= \beta_0 + (\beta_1 + \beta_3 \times \texttt{radio}) \times \texttt{TV} + \beta_2 \times \texttt{radio} + \epsilon. \end{aligned}$$

Results:

	Coefficient	Std. Error	t-statistic	p-value
Intercept	6.7502	0.248	27.23	< 0.0001
TV	0.0191	0.002	12.70	< 0.0001
radio	0.0289	0.009	3.24	0.0014
TV×radio	0.0011	0.000	20.73	< 0.0001

  - $\textbf{p is low} \rightarrow b_3 \neq 0 \rightarrow H_A \rightarrow$ 有一定关联

- 协同模型的R^2(96.8%)大于无协同项的模型(89.7%)
  - This means that $(96.8 - 89.7)/(100 - 89.7) = 69\%$ of the variability in `sales` that remains after fitting the additive model has been explained by the interaction term.
  - The coefficient estimates in the table suggest that an increase in TV advertising of $\$1,000$ is associated with increased sales of $(\hat{\beta}_1 + \hat{\beta}_3 \times \texttt{radio}) \times 1000 = 19 + 1.1 \times \texttt{radio}$ units.
  - An increase in radio advertising of $\$1,000$ will be associated with an increase in sales of $(\hat{\beta}_2 + \hat{\beta}_3 \times \texttt{TV}) \times 1000 = 29 + 1.1 \times \texttt{TV}$ units.

3.5.2 继承hierarchy

- 如果要在模型中引入协同项,需要考虑main effects,除非其p值过高(相关度不显著)
- 原因: 协同项的意义不同于其原项目, 不加入main effects的话不容易解读

3.5.3 Interactions between qualitative and quantitative variables

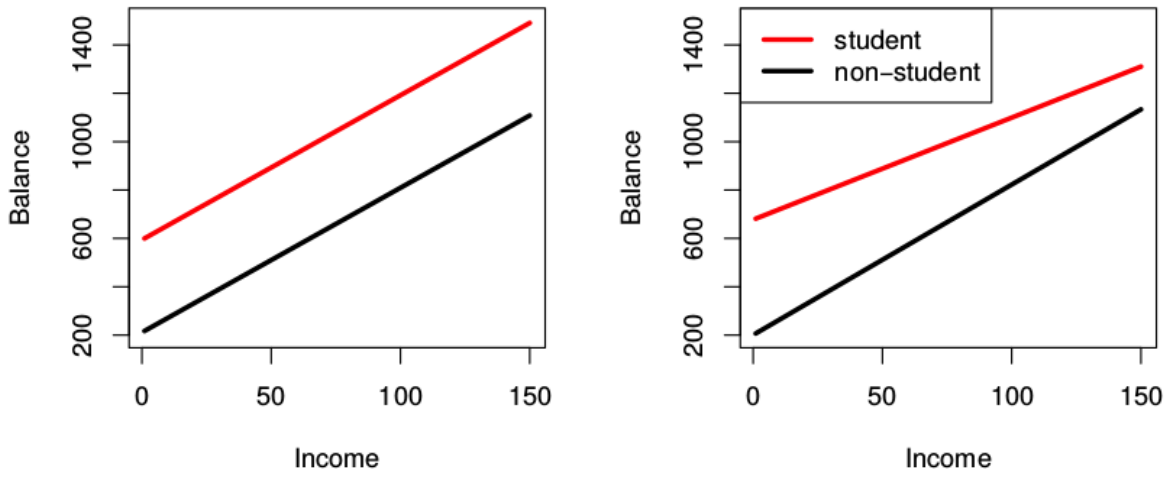
- 使用收入(quantitive)以及是否是学生(qualitive)预测信用卡余额
- 不加协同项:

$$\begin{aligned}\texttt{balance}_i &\approx \beta_0 + \beta_1 \times \texttt{income}_i + \begin{cases}\beta_2 & \text{if } i\text{th person is a student}\\ 0 & \text{if } i\text{th person is not a student}\end{cases}\\ &= \beta_1 \times \texttt{income}_i + \begin{cases}\beta_0 + \beta_2 & \text{if } i\text{th person is a student}\\ \beta_0 & \text{if } i\text{th person is not a student.}\end{cases}\end{aligned}$$

- 引入协同项:

$$\begin{aligned}\texttt{balance}_i &\approx \beta_0 + \beta_1 \times \texttt{income}_i + \begin{cases}\beta_2 + \beta_3 \times \texttt{income}_i & \text{if student}\\ 0 & \text{if not student}\end{cases}\\ &= \begin{cases}(\beta_0 + \beta_2) + (\beta_1 + \beta_3) \times \texttt{income}_i & \text{if student}\\ \beta_0 + \beta_1 \times \texttt{income}_i & \text{if not student}\end{cases}\end{aligned}$$

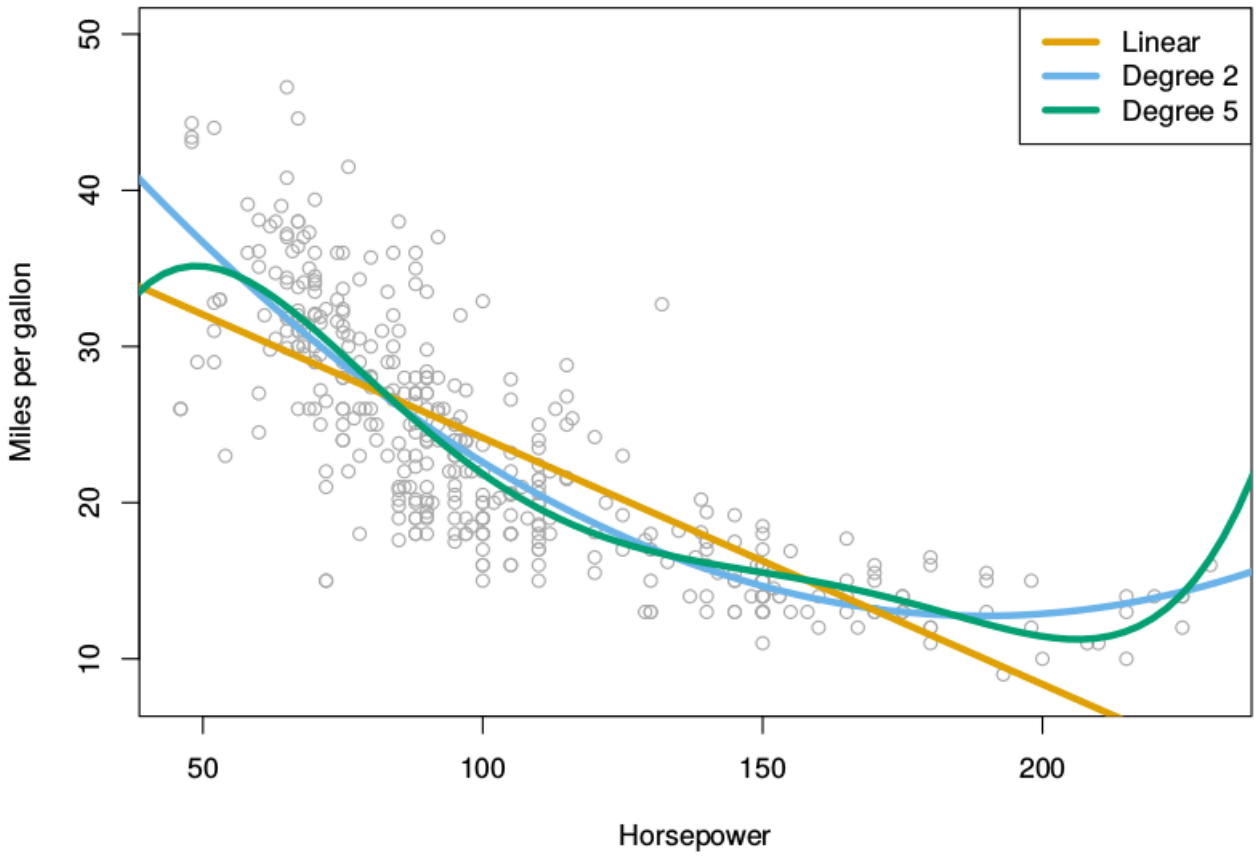
- 数据可视化:



Credit data; Left: no interaction between `income` and `student`. Right: with an interaction term between `income` and `student`.

3.5.4 Non-linear effects of predictors

polynomial regression on `Auto` data



- 引入高次变量, 进行多项式回归
- 可以参考广义可加模型GAM

The figure suggests that

$$\texttt{mpg} = \beta_0 + \beta_1 \times \texttt{horsepower} + \beta_2 \times \texttt{horsepower}^2 + \epsilon$$

may provide a better fit.

	Coefficient	Std. Error	t-statistic	p-value
Intercept	56.9001	1.8004	31.6	< 0.0001
horsepower	-0.4662	0.0311	-15.0	< 0.0001
horsepower^2	0.0012	0.0001	10.1	< 0.0001

3.6 Linear Regression in R

3.6.1 simple linear regression `lm() function`

```
plot(Boston$medv~lstat,data=Boston)
fit1=lm(Boston$medv~lstat,data=Boston) #线性模拟
abline(fit1,col="red") #增加趋势线
confint(fit1) #confidentence intervals
predict(fit1,data.frame(lstat=c(5,10,15)),interval="confidence") #预测函数
```

3.6.2 multiple linear regression

```
fit2=lm(medv~lstat+age,data=Boston)
fit3=lm(medv~.,Boston)
fit4=update(fit3,~.-age-indus)
```

3.6.3 non-linear

```
fit5=lm(medv~lstat*age,Boston)
fit6=lm(medv~lstat +I(lstat^2),Boston); summary(fit6) #GAM
fit7=lm(medv~poly(lstat,4)) #polynomial
plot(medv~lstat)
points(lstat,fitted(fit6),col="red",pch=20)
```

3.6.4 Qualitative predictors

```r
fit1=lm(Sales~.+Income:Advertising+Age:Price,Carseats)
contrasts(Carseats$ShelveLoc) #形成对立矩阵
```

3.7 Summary and Quiz

3.7.1 linear model, summary:

- 线性模拟得到参数的机理：最小二乘法(选择合适的β_0'、β_1'，最小化RSS)
  - residual残差e_i
  - RSS残差平方和
- SE标准误-量度精密度,SD标准差-量度离散度
  - SE越小,抽样误差越小,表明抽取的样本越能较好代表总体
  - SE小-离散度SD小(分子),但是样本与均值差别大(分母)
  - 置信区间$[\beta_1' \pm 2 * SE(\beta_1')]$
- H0零假设:X与Y无关;Ha择一假设:X与Y相关联(X系数不为0)
  - 一般原假设为H0，两组样本没有明显差别/要估计的参数为0
- t统计量t-statistic可用于假设检验:检验样品与均值差异
  - t是对回归模型个体系数的显著性进行假设检验,而f统计量则是整体
  - $t = \frac{\beta_1' - 0}{SE(\beta_1')}$,SE反比
  - |t|越大-样品偏离均值越远,样品与均值的差别越显著,p越小
- p-value:检验两组差别是否显著
  - p>0.05 两组差别无意义
  - p<0.05 显著意义
  - p<0.01 两组差别有非常显著的意义，不接受原假设
- RSE标准化残差=$\sqrt{\frac{RSS}{n-2}}$
- R^2[拟合度/整体精密度/相关度]
  - $R^2 = 1 - \frac{RSS}{TSS}$
  - R^2大,模型与样本观察值拟合度好,总体精密性高
  - TSS总平方和
- r某个X与Y的相关系数
  - 变化范围[-1,1]
  - |r|越大相关度越强,正相关或负相关

- 不同的预测因子应尽可能互不相关
- F统计量对多元回归整体显著性的检验,是否至少一个预测因子有助于预测

$$F = \frac{\frac{TSS-RSS}{p}}{\frac{RSS}{n-p-1}} \sim F(p, n-p-1)$$

  - F越大,残差越小,整体显著性越好
- 确定有效变量(预测因子)数:Backward/Forward Selection
  - 定性的预测因子:使用权重法将其向量化
  - 协同interaction:引入合并项
  - 非线性化non-linear:引入二次因子X^2

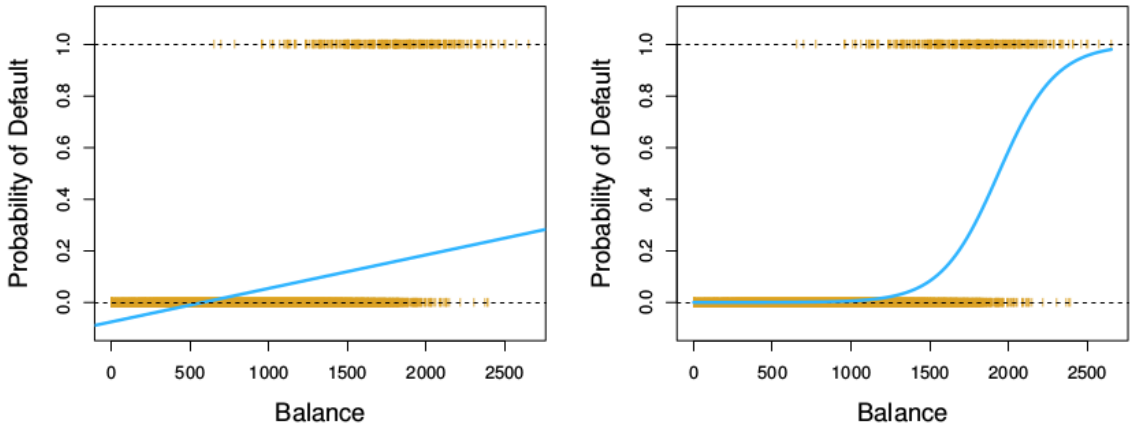
3.7.2 Quiz

- 正相关度仅仅意味着univariate regression为正相关.参考3.3

Chapter 4 Classification

4.1 Introduction to Classification Problems

- 分类的意义：枚举 / 布尔结果
- 举个栗子：一个学生是坏学生的概率
- Linear Regression VS Logical Regression
  - 逻辑回归更适合二元分类：线性回归可能造成大于1或小于0的情况



The orange marks indicate the response Y, either 0 or 1. Linear regression does not estimate $\Pr(Y = 1|X)$ well. Logistic regression seems well suited to the task.

- 对于多个classes：线性回归不适合，应该使用多级逻辑回归或判别分析
- 多级逻辑回归：Multiple-class Logical Regression
- 判别分析：Discriminant Analysis

4.2 Logistic Regression

4.2.1 Basic form

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

$$log\frac{p(X)}{1-p(X)} = \beta_0 + \beta_1 X$$

- 无论参数及X值为如何，p(X)恒属于[0,1]

4.2.2 使用最大似然估测参数

- 最大似然：

$$l(\beta_0, \beta) = \prod_{i:y_i=1} p(x_i) \prod_{i:yi=0} (1-p(x_i))$$

- 似然与概率的差别：
  - 概率：已知参数，预测接下来的观测结果
  - 似然性：已知观测结果，估计有关事物的性质
  - 似然函数可认为条件概率的逆反

$$L(A=a|B) \Leftrightarrow P(A|B=b)$$

- 高斯似然函数

$$L(X|\mu,\sigma^2)=\prod N(x_n|\mu,\sigma^2)$$

$$\mu_{ML}=\frac{1}{N}\sum x_n$$

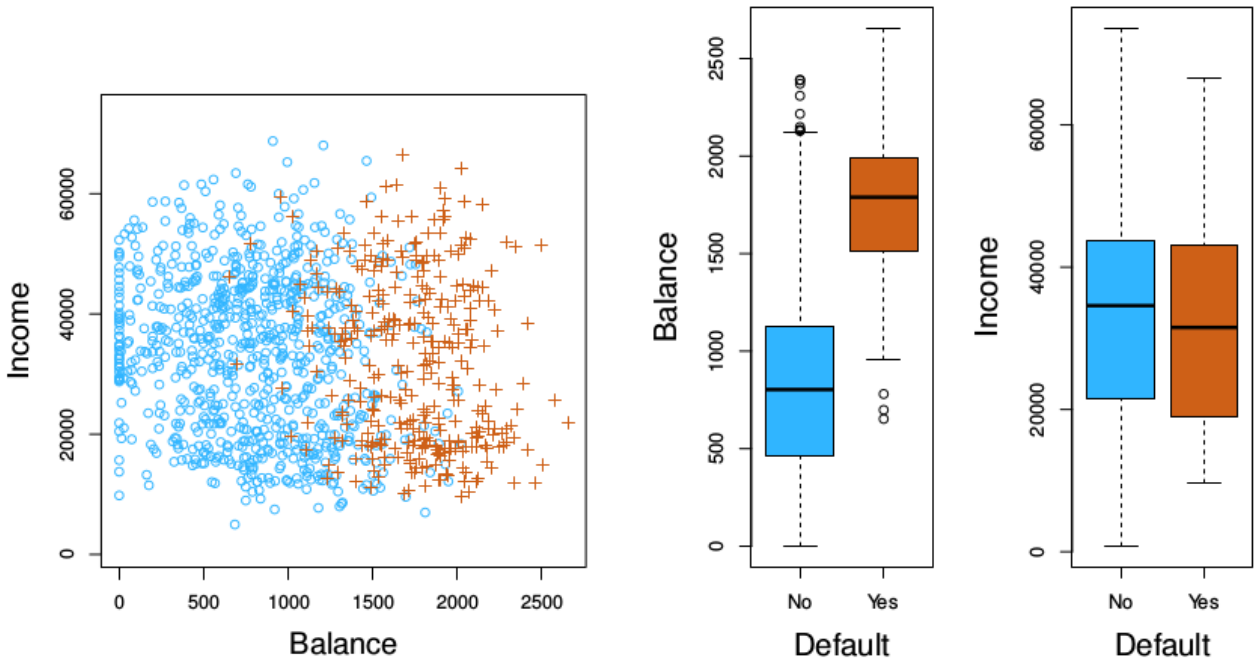
$$\sigma^2_{ML}=\frac{1}{N}\sum[(x_n-\mu_{ML})^2]$$

- 逻辑回归的目标：选取β_0,β_1来最大化观察值的likelihood

4.2.3 逻辑回归的适用性

- 逻辑回归可用于线性预测或分类;
- 仅能用于线性问题(feature与target线性相关);
- 非线性不适用逻辑回归
- 使用逻辑回归时利用与预测目标线性相关的feature
- 残差和因变量服从二项分布;
- 与朴素贝叶斯不同，逻辑回归不需要满足条件独立假设，但各个feature的贡献是独立计算，无法自动组合feature形成新feature;
- 重复计数现象指标不适用于逻辑回归;

4.3 栗子: 使用逻辑回归进行预测



- 使用学生作为预测因子:

Lets do it again, using `student` as the predictor.

	Coefficient	Std. Error	Z-statistic	P-value
`Intercept`	-3.5041	0.0707	-49.55	< 0.0001
`student[Yes]`	0.4049	0.1150	3.52	0.0004

$$\widehat{\Pr}(\texttt{default=Yes}|\texttt{student=Yes}) = \frac{e^{-3.5041+0.4049\times 1}}{1+e^{-3.5041+0.4049\times 1}} = 0.0431,$$

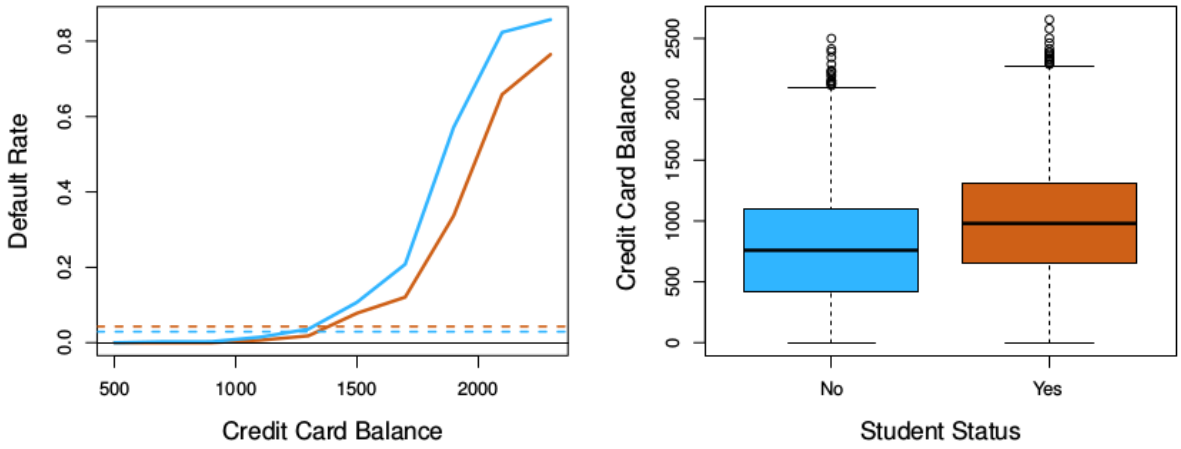
$$\widehat{\Pr}(\texttt{default=Yes}|\texttt{student=No}) = \frac{e^{-3.5041+0.4049\times 0}}{1+e^{-3.5041+0.4049\times 0}} = 0.0292.$$

- 结果展示:

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

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

	Coefficient	Std. Error	Z-statistic	P-value
Intercept	-10.8690	0.4923	-22.08	< 0.0001
balance	0.0057	0.0002	24.74	< 0.0001
income	0.0030	0.0082	0.37	0.7115
student[Yes]	-0.6468	0.2362	-2.74	0.0062



- Students tend to have higher balances than non-students, so their marginal default rate is higher than for non-students.
- But for each level of balance, students default less than non-students.
- Multiple logistic regression can tease this out.

4.4 Multivariate Logistic Regression

- 和前面的公式类似，只是变量变成了多个：

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_p X_p}}$$

4.5 Discriminant Analysis判别分析

4.5.1 基本原理

- 模拟X在每个class中的分布 (基于已知分类建立判别函数)
- 使用Bayes theorem来分布概率点 (使用判别函数对未知样本分类)
- 最终目标：得到$Pr(Y|X)$
- 对每个class使用正态（高斯）分布 : 线性或二次判别分析

4.5.2 LDA

- 原理：这个和PCA有类似之处
  - 将带上标签的数据（点），通过投影的方法，投影到维度更低的空间中;
  - 使得投影后的点，会形成按类别区分，一簇一簇的情况;
  - 相同类别的点，将会在投影后的空间中更接近。
- 分类的目标是: 使得类别内的点距离越近越好（集中），类别间的点越远越好

4.5.3 Bayes theorem for classification

$$Pr(Y=k|X=x)=\frac{Pr(X=x|Y=k)Pr(Y=k)}{Pr(X=x)}$$

$$P(X)P(Y|X)=P(Y)P(X|Y)$$

- 判别分析的一个变体 :

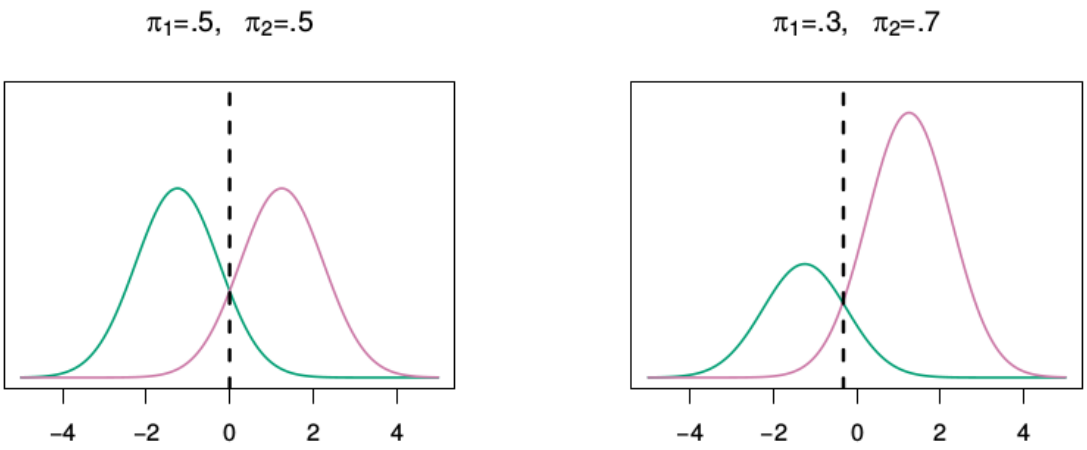
$$Pr(Y=k|X=x)=\frac{\pi_k f_k(x)}{\sum_{l=1}^{K}\pi_l * f_l(x)}$$

  - X在Class k的density为 :

$$f_k(x)=P_r(X=x|Y=k)$$

  - Class k的先验概率为 :

$$\pi_k=P_r(Y=k)$$



We classify a new point according to which density is highest.

When the priors are different, we take them into account as well, and compare $\pi_k f_k(x)$. On the right, we favor the pink class — the decision boundary has shifted to the left.

4.6 Gaussian Discriminant Analysis

4.6.1 Gaussian density

- 高斯密度公式：
  - μ_k : mean
  - σ_k^2 : Var(k) ,variance in class k,assuming all the $\sigma_k = \sigma$ are the same

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}(\frac{x-\mu_k}{\sigma_k})^2}$$

- 将高斯密度公式引入贝叶斯公式：

$$p_k(x) = P_r(Y=k|X=x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}(\frac{x-\mu_k}{\sigma_k})^2}}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}(\frac{x-\mu_l}{\sigma})^2}}$$

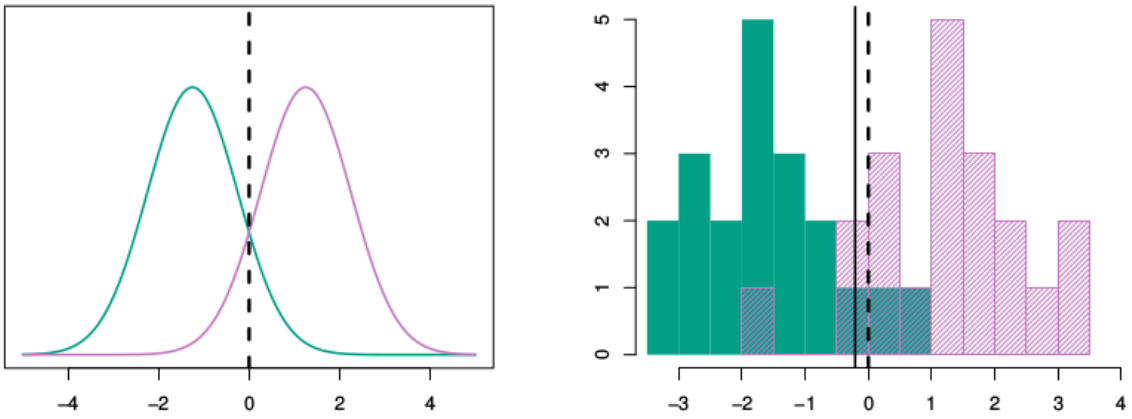
4.6.2 Discriminant functions

- 原理：通过使用logs,discarding terms(这些都是不依赖于k的) 将x分配到某个可以带有最

大判别值discriminant score的类别

$$\delta_k(x) = \frac{x\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + log(\pi_k)$$

- $\delta_k(x)$是关于x的线性函数



Example with $\mu_1 = -1.5$, $\mu_2 = 1.5$, $\pi_1 = \pi_2 = 0.5$, and $\sigma^2 = 1$.

- 参数未知 : 使用训练数据进行评估

$$\pi'_k = \frac{n_k}{n}$$

$$\mu'_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

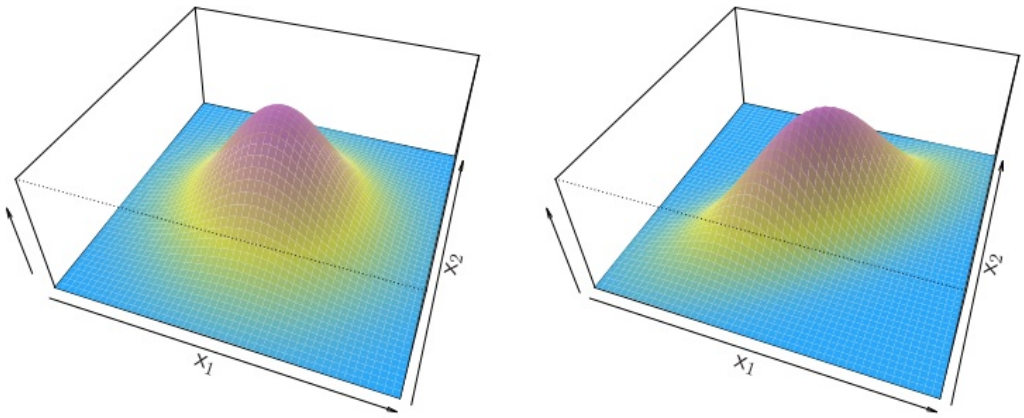
$$(\sigma^2)' = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \mu'_k)^2 = \sum_{k=1}^{K} \frac{n_k - 1}{n-K} (\sigma'_k)^2$$

  - 其中$(\sigma'_k)^2$为kth class的估测方差 $(\sigma'_k)^2 = \frac{1}{n_k - 1} \sum_{i:y_i=k} (x_i - \mu'_k)^2$

4.6.3 几个栗子

- p>1的情况

Linear Discriminant Analysis when $p > 1$

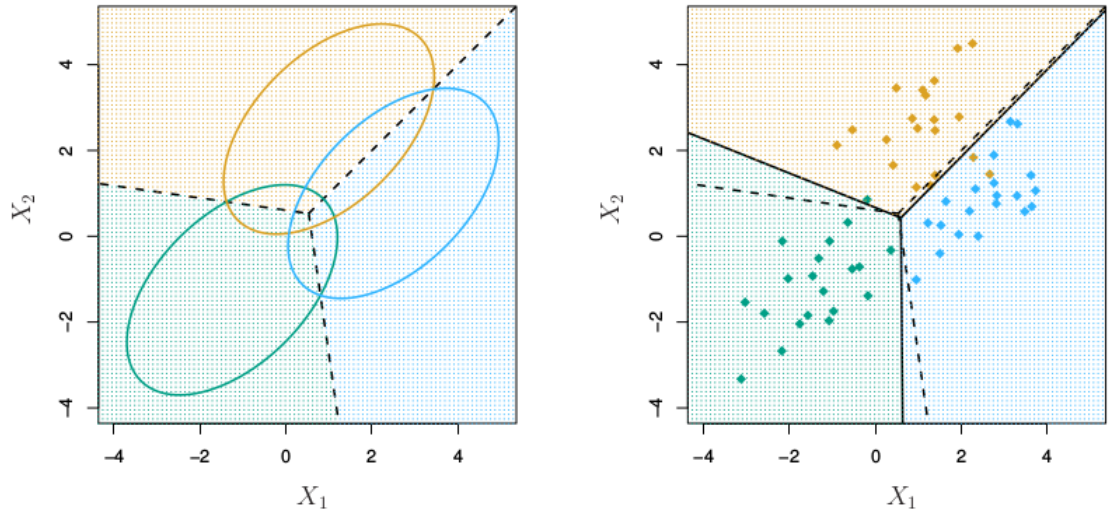


Density: $f(x) = \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \mathbf{\Sigma}^{-1}(x-\mu)}$

Discriminant function: $\delta_k(x) = x^T \mathbf{\Sigma}^{-1}\mu_k - \frac{1}{2}\mu_k^T \mathbf{\Sigma}^{-1}\mu_k + \log \pi_k$

Despite its complex form,
$\delta_k(x) = c_{k0} + c_{k1}x_1 + c_{k2}x_2 + \ldots + c_{kp}x_p$ — a linear function.

Illustration: $p = 2$ and $K = 3$ classes



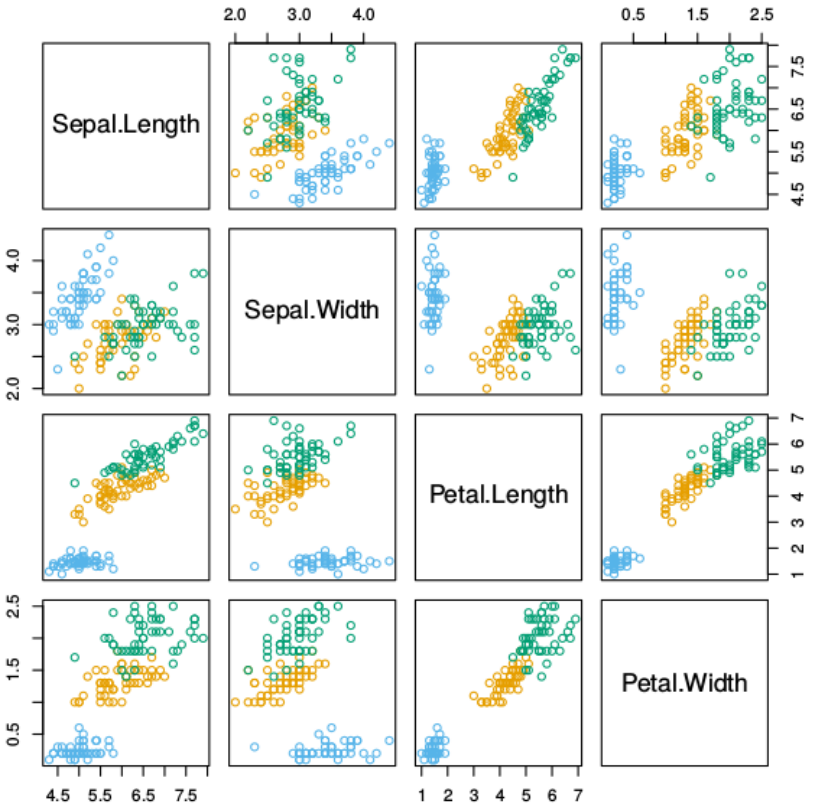
Here $\pi_1 = \pi_2 = \pi_3 = 1/3$.
The dashed lines are known as the *Bayes decision boundaries*.
Were they known, they would yield the fewest misclassification errors, among all possible classifiers.

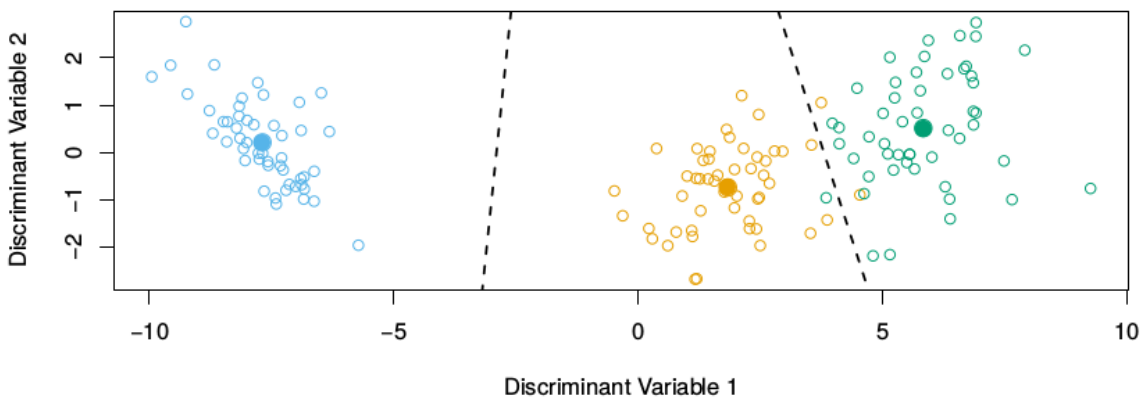
- 基于鸢尾花数据集的研究:

4 variables
3 species
50 samples/class

- Setosa
- Versicolor
- Virginica

LDA classifies all but 3 of the 150 training samples correctly.





When there are K classes, linear discriminant analysis can be viewed exactly in a $K-1$ dimensional plot.
Why? Because it essentially classifies to the closest centroid, and they span a $K-1$ dimensional plane.
Even when $K>3$, we can find the "best" 2-dimensional plane for vizualizing the discriminant rule.

4.6.4 $\delta_k(x)$可用于估测属于x某个类别k的概率

$$Pr'(Y=k|X=x)=\frac{e^{\delta'_k(x)}}{\sum_{l=1}^{K}e^{\delta'_l(x)}}$$

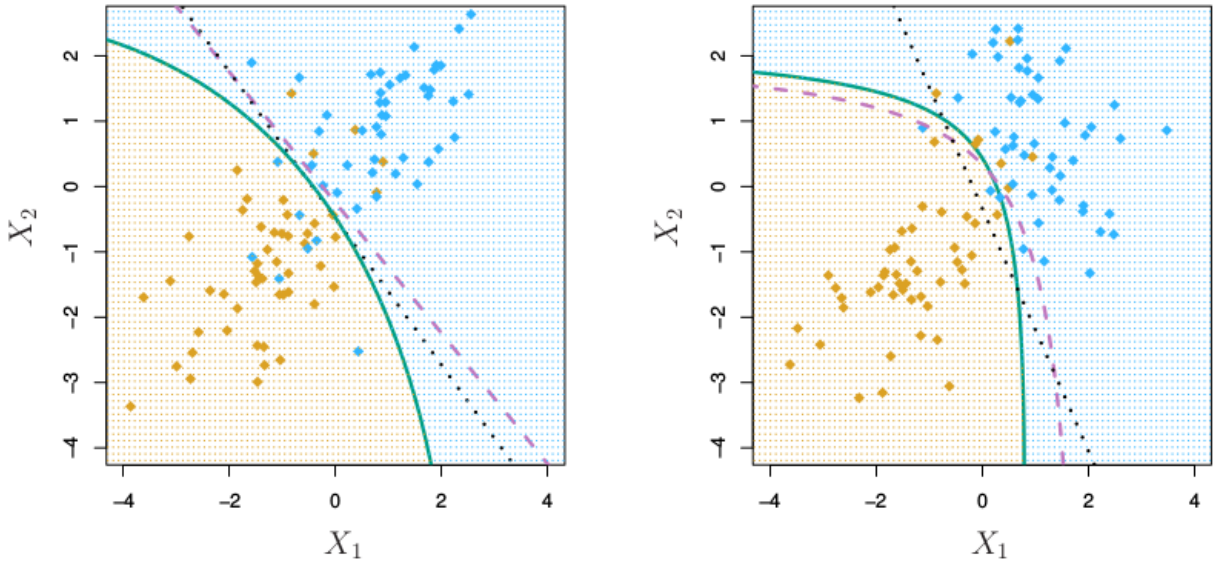
- 以下两个过程等效：
  - maximum $\delta'_k(x)$ via classifing;
  - get the class that can maximum Pr'(Y=k|X=x) via classifying

4.7 Quadratic Discriminant Analysis and Naive Bayes

$$Pr(Y=k|X=x)=\frac{\pi_k f_k(x)}{\sum_{l=1}^{K}\pi_l f_l(x)}$$

- 对于线性判别分析LDA，$f_k(x)$为Gaussian densities且每个class的协方差矩阵covariance matrix(就是∑)相同
- 二次判别分析：不同类别的协方差矩阵不同
- 朴素贝叶斯(条件独立模型)：$f_k(x)=\prod f_{jk}(x_j)$
- 对于高斯函数，条件独立模型意味着协方差矩阵是对角线矩阵

4.7.1 Quadratic Discriminant Analysis 二次判别分析



$$\delta_k(x) = -\frac{1}{2}(x-\mu_k)^T\mathbf{\Sigma}_k^{-1}(x-\mu_k) + \log\pi_k$$

Because the $\mathbf{\Sigma}_k$ are different, the quadratic terms matter.

4.7.2 Naive Bayes 朴素贝叶斯

- 各属性条件是彼此独立的

$$P(X|Y=y) = \prod P(X_i|Y=y)$$

- 条件独立

$$P(X|Y,Z) = P(X|Z)$$

$$P(X \cap Y) = 0$$

$$P(X,Y|Z) = P(X|Z)P(Y|Z)$$

- 朴素贝叶斯分类器的特征
  - 适合于孤立的噪声点或无关属性 e.g. text classification
  - 不适合彼此相关的属性(条件独立假设不成立)

4.7.3 LR VS LDA

- 对于二分类问题，LR与LDA形式相同

$$log\frac{p_1(x)}{1-p_1(x)}=log\frac{p_1(x)}{p_2(x)}=c_0+c_1x_1+...+c_px_p$$

- 不同之处:
  - 逻辑回归使用基于条件概率Pr(Y|X)的条件似然(conditional likelihood)，是一种判别学习(discriminative learning)
  - LDA使用基于 Pr(X,Y)的全概率, 是一种生成学习(generative learning)
- 除此之外，在实际应用中结果相似
- 使用LDA的情况：
  - 类别分离度良好时，逻辑回归参数不稳定，LDA不会遇到此类问题;
  - n很小或者X在每个类别里都是正态分布，LDA更加稳定;
  - 多分类问题适合使用LDA

4.8 Classification in R

4.8.1 logistic regression `glm()`

```
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial) #生成线性模型，得到b0,b1参数

glm.probs<-predict(glm.fit,type="response") #预测模型

glm.pred<-ifelse(glm.probs>0.5,"UP","DOWN") #条件选择

ifelse(test, yes,no) #f返回结果跟test形式相同，但是为逻辑格式
```

4.8.2 make training and test set

```
train<-Year<2005 #设置train
glm_fit<-glm(formula,data,
        family=binormial,  #family默认值为Gaussian，此处为二项式
        subset=train) #指定进行fit的子集，即<2005

glm_probs<-predict(glm_fit,
        newdata=Smarket[!train,], #Smarket[Y>2005,]
        type="response")

glm_pred<-ifelse(glm_probs>0.5,"UP","DOWN")
Direction_2005<-Smarket$Direction[!train]
table(glm_pred,Direction_2005) #得到结果，之后可以summary/mean
```

4.8.3 linear discriminant analysis `lda()`

```
lda.fit=lda(Direction~Lag1+Lag2,data=Smarket, subset=Year<2005)

Smarket.2005=subset(Smarket,Year==2005)
lda.pred=predict(lda.fit,Smarket.2005)
```

4.8.4 K-Nearest Neighbors `knn() in class package`

```
library(class)
attach(Smarket)
Xlag=cbind(Lag1,Lag2)
train=Year<2005

knn.pred=knn(Xlag[train,], #training data
        Xlag[!train,], #testing data
        Direction[train], #factor of true classifications of training set
        k=1) #number of neighbours considered

table(knn.pred,Direction[!train])
mean(knn.pred==Direction[!train])
```

4.9 Summary & Quiz

4.9.1 Summary

- Logistic regression

$$p(X) = \frac{e^{b_0+b_1X}}{1+e^{b_0+b_1X}}$$

- 在K=2时很常用
- n很小或classes are well separated时用LDA
- 贝叶斯函数，高斯密度函数
- p很大时朴素贝叶斯

4.9.2 Quiz

Chapter 5 Resampling Methods

- Training Error VS Test Error:
  - Test error: 预测过程中产生的误差
  - Training error: 训练过程中产生的误差
  - High variance ⇔ Overfitting
  - High bias ⇔ Underfitting
  - High variance ⇔ Low bias
- 参数数量与样品数量的影响
  - 参数数量d增加 $\Rightarrow$ J_{train}降低,J_{CV}和J_{test}先降低后升高;
  - 样本量m增加 $\Rightarrow$ J_{train}不断增大,J_{CV}
    不断降低, 但不交叉只是趋同 + 模型本身$underfit$\space\Rightarrow\space
    增加样品量意义不大 + 模型本身$overfit$\space\Rightarrow\space$有一定帮助,但需要极大的样本量
- 两种resampling methods: cross-validation, bootstrap
  - 目的：得到测试集的预测误差/标准差/参数偏差

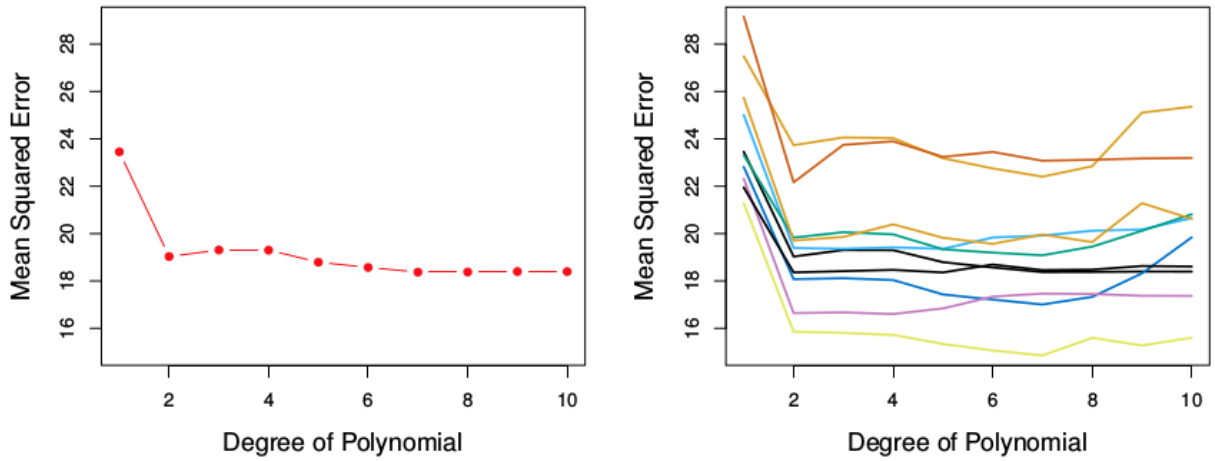
5.1 Cross-validation交叉检验

5.1.1 Validation-set approach

- 随机将训练样本分成两组：训练集和交叉验证集
- 使用训练集来拟合模型;
- 使用该模型对交叉验证集中的样本进行预测;
- 通过交叉验证集的误差可以估计出测试集的误差

Example: automobile data

- Want to compare linear vs higher-order polynomial terms in a linear regression
- We randomly split the 392 observations into two sets, a training set containing 196 of the data points, and a validation set containing the remaining 196 observations.



Left panel shows single split; right panel shows multiple splits

5.1.2 Drawbacks of validation set approach

- 测定结果可变性强,与这两个集合的观察值有很大关系;
- 只使用一部分观察值来拟合模型 ⇒ 样品量偏少;
- 因此交叉验证集有可能过度估计全局误差;
- 根据样本量m与两种误差的关系可得： 样本量越少获得信息越少，越可能产生误差 ⇒ 导致测试误差偏高

5.2 K-fold Cross-Validation

5.2.1 基本原理

- 将数据等量分为K个子集，选择一个作为交叉验证集，剩下的K-1个合在一起作为训练集,其余同上

- 交叉检验的过程实际上是选择不同交叉验证集把实验重复做K次，然后取MSE

5.2.2 details

- 将训练集分为K部分$C_1, C_2, \ldots C_K$,C_k为子集k，共有n_k个观察值;

$$n_k = \frac{n}{K}$$

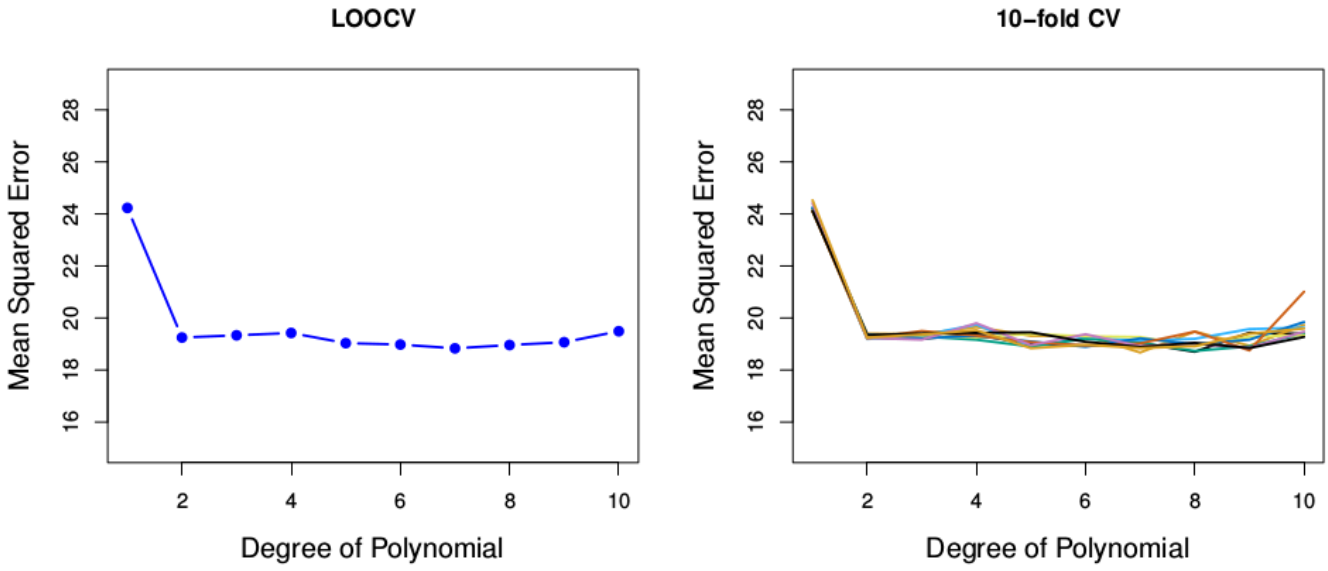
- 计算CV: 先求出每个交叉验证集的MSE,取均值
  - 拟合模型: $N - n_k$个训练数据拟合而成
  - y_i' : C_k中观察值i的拟合结果

$$CV_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} MSE_k = \frac{n_1 MSE_1 + n_2 MSE_2 + \ldots + n_K MSE_K}{n}$$

$$MSE_k = \sum_{i \in Ck} \frac{(y_i - y_i')^2}{n_k}$$

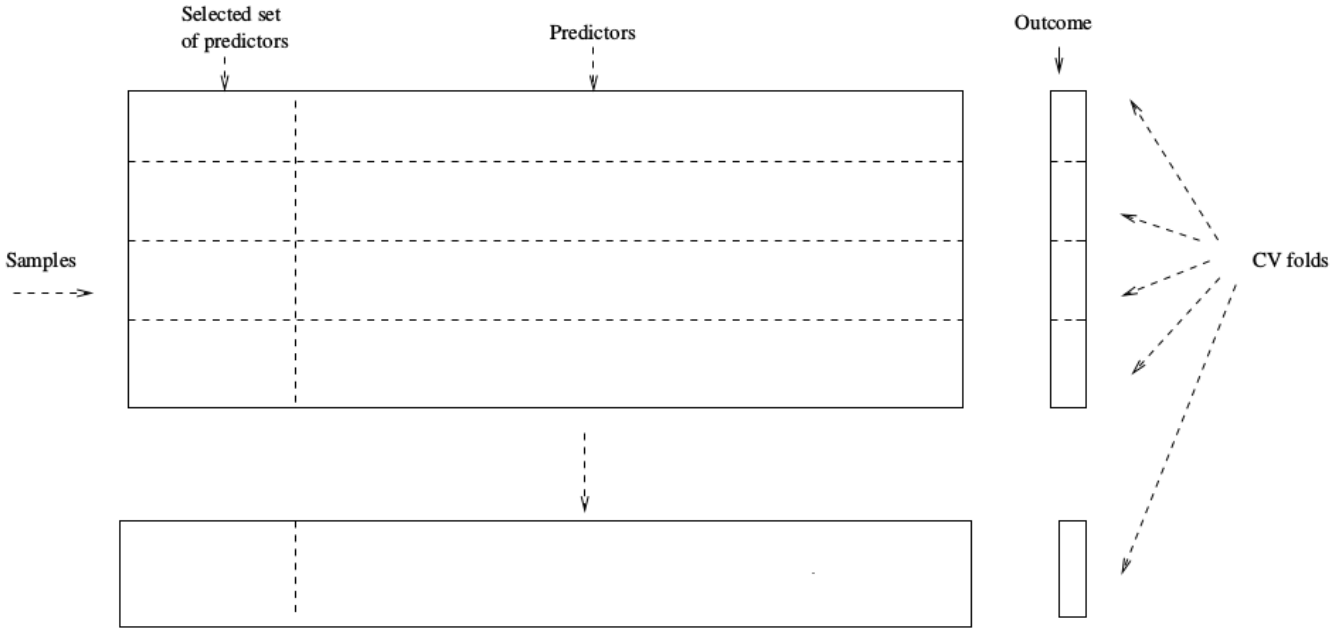
5.2.3 LOOCV 留一交叉检验

- 一种特殊的K-fold CV : K=n
- 相当于每一层只有一个样本，其余n-1个样本作为训练数据集
- LOOCV的cost可以等于单个拟合模型
- LOOCV很有用，但可能并不均匀：`higher variance`
- 每个fold的估计值高度相关，导致均值`high variance`
- **比较理想的层数是[K=5 OR 10] ⇒ trade off**



5.2.4 CV的其他一些问题

- 每次训练集的样本量为全局数据量的$\frac{K-1}{K}$,K过小会导致`higher bias`
- 当$K=n(LOOCV)$时bias最小 $\Rightarrow$ K过大导致 `higher variance`
- K=5 or 10 时可以较好平衡bias-variance trade-off



5.2.5 CV for Classification Problems

- 跟前面那个公式相比就是MSE_k变成了Err_k

$$CV_K = \sum_{k=1}^{K} \frac{n_k}{n} Err_k$$

$$Err_k = \sum_{i \in C_k} I(y_i \neq y_i')/n_k$$

- CV_K的估测标准差为：

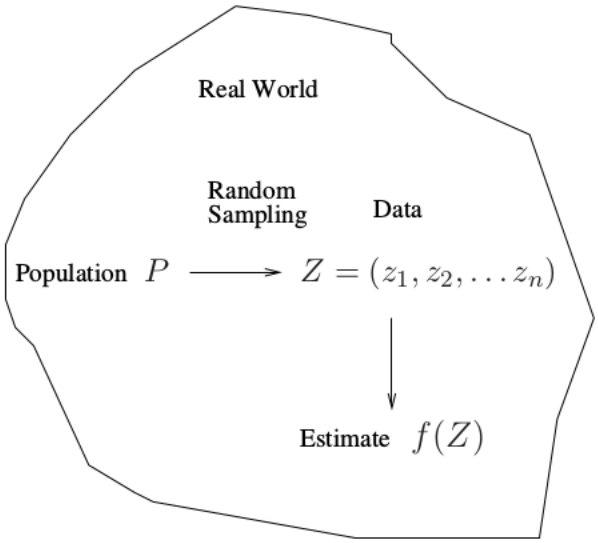
$$SE'_{CV_K} = \sqrt{\sum_{k=1}^{K} \frac{(Err_k - Err'_k)^2}{K-1}}$$

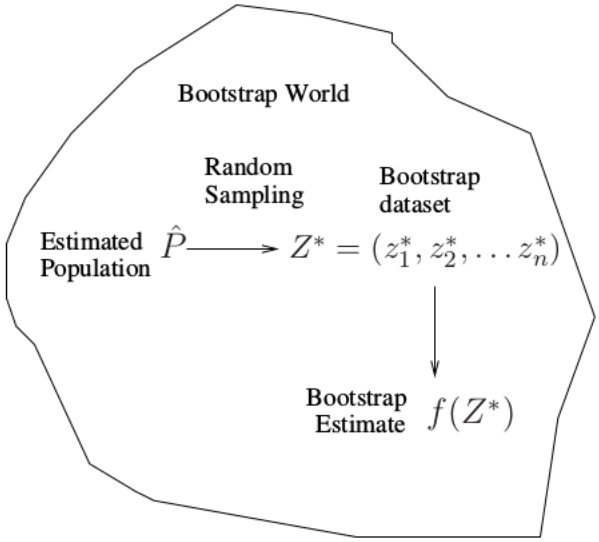
- CV可以用于分类问题但是不严谨
- 原因：不同训练集并不彼此独立，而是共享部分训练样本

5.3 Bootstrap

- 一种统计方法，属于非参数统计
- 作用：估量系数的SE / 置信区间(Bootstrap Percentile CI) / 最小化方差

- 基本做法：
  - 从原始样本中重复抽取一定数量的样本;
  - 根据抽出的样本计算给定的统计量T;
  - 重复上述N次（一般大于1000），得到N个统计量T;
  - 计算上述N个统计量T的样本方差，得到统计量的方差





5.3.1 栗子

- 将钱分配到两个集合:X,Y;
- 选定一定比例α的钱给X,1-α的钱给Y;
- 目标在于最小化风险函数$Var(\alpha X + (1-\alpha)Y)$;

$$\alpha = \frac{Var(Y) - Covar(X,Y)}{Var(X) + Var(Y) - 2Covar(X,Y)}$$

- 以上几个Var都是未知的 $\Rightarrow$ 通过估计Var，进而得到α的估计值
- 通过重复抽取，得到n的α估计值，取均值得到α

5.3.2 Bootstrap是否适合预测误差

- 在CV中，每个K validation都与剩下的K-1个训练folds不同，不存在重叠;
- 若使用bootstrap,存在很强的重叠 $\Rightarrow$ `test error显著偏低`
- 也可以通过一些方法尽可能避免这个问题，但这样一来CV明显要更方便

5.3.3 Bootstrap VS Permutaion tests(置换检验)

- bootstrap
  - 样本来源 : estimated population

    - 用途：用于估计SE以及CI
- permutation
    - 样本来源：数据集的零分布
    - 用途 ： 估计p值以及False Discovery Rates,从而进行假设检验
- bootstrap可以用于进行null hypothesis
- bootstrap也可从零分布中获取样本，但和permutation相比没有优势

5.4 Resampling/validation in R

5.4.1 LOOCV

```
cv.glm(data, glmfit, cost, K) #需要调用boot package
```

- 举个栗子：

```
#前面就是前面学的glm,得到系数的估计值
library(boot)
plot(mpg~hp,data=mtcars)
glm.fit<-glm(mpg~hp,data=mtcars)

#这里默认的LOOCV：
cv.glm(mtcars,glm.fit)$delta #delta就是prediction error

#结果得到两个数字：
LOOCV的prediction error
bias correction #考虑到training set比总样本稍小进行的修正，因此比LOOCV稍小
```

- 再举个栗子：自定义LOOCV函数

```
loocv=function(fit){
        h=lm.influence(fit)$h
        mean((residuals(fit)/(1-h))^2)
}
loocv(glm.fit) #试运行得到的和刚刚的LOOCV一样

cv.error=rep(0,5)
degree=1:5
for(d in degree){
        glm.fit=glm(mpg~poly(horsepower,d), data=Auto)
        cv.error[d]=loocv(glm.fit)
}
```

```
13.
14.  plot(degree,cv.error,type="b")
```

5.4.2 10-fold CV

```
1.  cv.error10=rep(0,5) #fit 10 times
2.  for(d in degree){
3.          glm.fit=glm(mpg~poly(horsepower,d), data=Auto)
4.          cv.error10[d]=cv.glm(Auto,glm.fit,K=10)$delta[1]
5.  }
6.
7.  lines(degree,cv.error10,type="b",col="red")
```

5.5 Summary and Quiz

5.5.1 Summary

- Cross-validation：数据集比较小时适用，估计的test error可能偏大
- K-fold CV,重复K次取MSE
  - LOOCV,K=n, low bias, high variance
  - 综合考虑 K=5/10 比较合适
- CV不适合classification,因为training sets 并不彼此独立
- Bootstrap：估量系数的SE/置信区间/方差
  - 重复取样，存在重叠-prediction error 偏低

Chapter 6 Linear Model Selection and Regularization

6.1 Introduction and Best-Subset Selection

- 替换最小二乘法, 选择合适的模型

6.1.1 优化最小二乘法的原因：

- prediction accuracy : p>n时需要控制variance
- model interpretability: 通过参数选择, 移除无关参数, 令模型可解释性更强

6.1.2 三种减少参数的方法

- subset selection: 只选取和response相关的p个预测因子,组成子集进行模型拟合

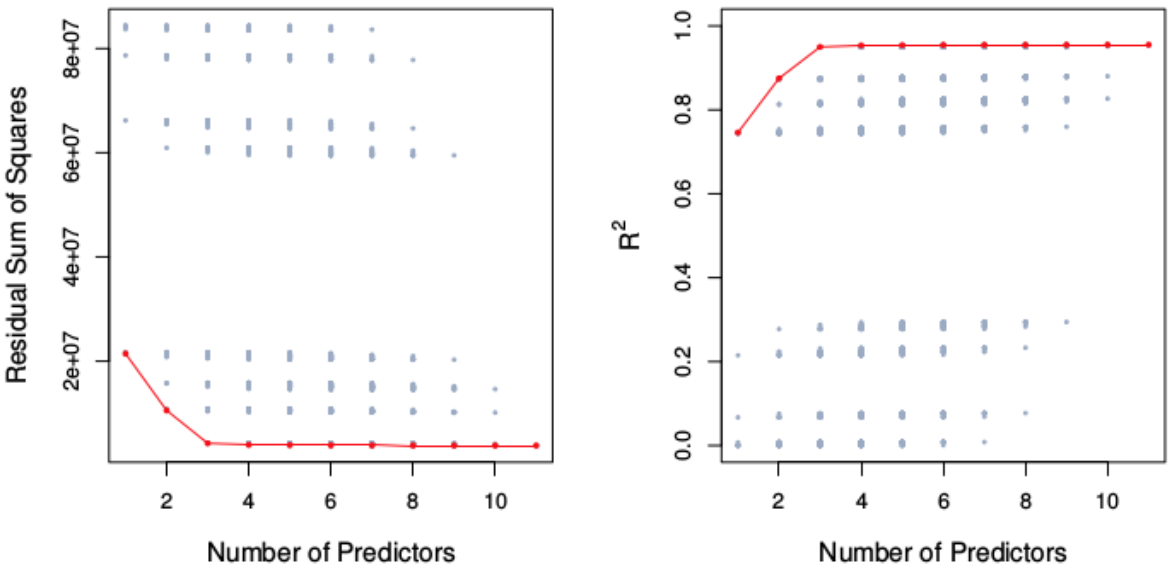
- shrinkage (regularization)
  - 不改变预测因子数量，但是部分预测因子系数为0;
  - 包括Ridge Regression+Lasso
- dimension reduction
  - 通过对原有预测因子进行线性组合，得到一组简化的新预测因子
  - 使用新预测因子进行线性拟合，优化bias-variance trade-off

6.1.3 Subset selection

- 选取M_0作为null model, 不含预测因子, 使用该模型预测每个观察值的样本均值
- for k in 1,2,...p :
  - 得到最佳M_1 ：
    - 拟合所有带1个预测因子的模型;
    - 模型数量为$C_p^1 = p$，即从p个预测因子里挑一个
    - 从这些模型里择优称作M_1;
    - 选择标准：最小的RSS，最大的R^2
  - 同理得到最佳M_2，$M_3 \dots M_p$
  - 上述过程总共拟合的模型数量

$$C_p^0 + C_p^1 + \dots + C_p^p = 2^p$$

$$C_a^b = \frac{a!}{b!(a-b)!}$$



- 从上一步得到的$M_0 - M_p$中选择一个最佳模型, 选择标准：
  - CV预测以最小化test error;
  - 小Mallow’ s Cp;
  - 小AIC;小BIC;
  - 大adjusted R^2

6.2 Stepwise Selection

6.2.1 p很大时, best subset selection不适用

- 搜索空间过大会导致overfitting以及high variance;
- 一共2^p个模型，效率不高，计算量大 $\Rightarrow O(2^n)$
- stepwise selection挑选模型更严格，在这种情况下更适合;
- stepwise selection仅仅需要$1+\frac{p(p+1)}{2}$个模型 $\Rightarrow O(n^2)$

6.2.2 forward stepwise selection

- 始于零参数模型，逐个添加预测因子
- 在每一步，可以给予拟合最大的额外提升的变量会被添加进model
- details：
  - M_0为null model;
  - for k in 0,1,2,... p-1:
    - 向M_k中随机添加一个预测因子，得到p-k个新模型 (e.g. M_0中加一个，得到p个)
    - 在这p-k个模型中选择最佳，称作M_{k+1}
    - M_0中添加1个, 择优得到M_1, M_1中添加一个, 择优得到M_2...
  - 从$M_0 - M_p$这p+1个模型中选择最佳模型，标准同前
- Analysis:
  - FSS选择法计算量相较于subset selection有明显的优势

$$1+p+(p-1)+(p-2)+\ldots+1=1+\frac{p(p+1)}{2}$$

  - 但是并不能保证能在这堆模型中找到全局最优
  - 局部最优: M_1的参数永远会存在于M_n，但 含x个参数的最佳模型Mx可能并不包含M1的参数
  - 总体而言FSS法的RSS是要高于best subset selection的

# Variables	Best subset	Forward stepwise
One	rating	rating
Two	rating, income	rating, income
Three	rating, income, student	rating, income, student
Four	cards, income	rating, income,
	student, limit	student, limit

The first four selected models for best subset selection and forward stepwise selection on the Credit data set. The first three models are identical but the fourth models differ.

6.2.3 Backward stepwise selection

- 始于p个参数的最小二乘法模型，逐个移除无关预测因子
- details:
  - M_p为full model，p个预测因子;
  - for k in p,p-1,...1:
    - 向M_k中移除一个因子，得到k-1个新模型(e.g. 从Mp中移除得到p-1个模型)
    - 在这k-1个模型中选择最佳，称作M_{k-1}
    - 以此类推直到M_0
  - 从M_0 – M_p这p+1个模型中选择最佳模型，标准同前
- 总体而言和FSS类似，比subset selection简单

6.2.4 前后逐步选择的异同

- 都只搜索$1+p(p+1)/2$个模型，比BSS效率高很多
- 都只能得到局部最佳而非全局最佳 ： RSS略大于BSS
- 后逐步选择要求样本量n>变量量p，以便拟合整个模型，但前逐步选择允许n

6.3 Estimating test error

6.3.1 模型选择标准

- 包含所有预测因子的模型具备最小化的RSS及最大的R^2
- 这些标准都是基于training error的
- 最终目标是要最小化test error而非training error
- overfitting : training error小但test error很大

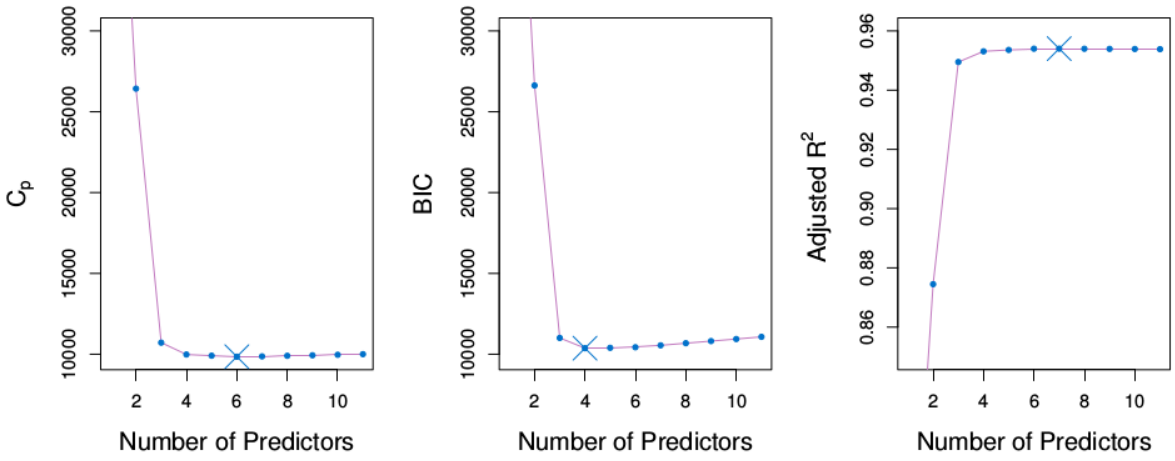
- $min(RSS)$与$max(R^2)$并不适用于在预测因子数量不同的一系列模型中选取最优模型

6.3.2 估测test error的两种方法

- 间接法：通过调整适应training error来抵消overfitting导致的bias
- 直接法：使用validation set/cross-validation直接估测test error

6.3.3 间接法：`Cp/AIC/BIC/Adjusted R Square`

- 调节training error, 从一堆变量数不相同的模型中择优



- Mallow’ s Cp评估模型：**越小越好**

$$C_p = \frac{RSS + 2dVar'(error)}{n}$$

  - n:样本量
  - d:参数数量(e.g. M_3有4个参数，d=4)
  - $Var'(error)$: 估计方差
- AIC(Akaike Information Criteria): **越小越好**
  - 先基于最大似然拟合一批模型

$$AIC = -2logL + 2d$$

    L: 估计模型的最大似然函数
  - 特例: 带有Gaussian误差的线性模型, 这种情况下最大似然和最小二乘是同一种东西 $C_p = AIC$
- BIC(Bayesian Information Criterion：**越小越好**

$$BIC = (RSS + log(n)dVar'(error))$$

    - 与Cp类似，BIC越小越好(small test error)
    - 当n>7时系数log(n)>2,对于大样本量模型权重增加
    - 导致的结果，BIC选择的模型预测因子数跟Cp/AIC比偏小！
- Adjusted R Square:**越大越好**

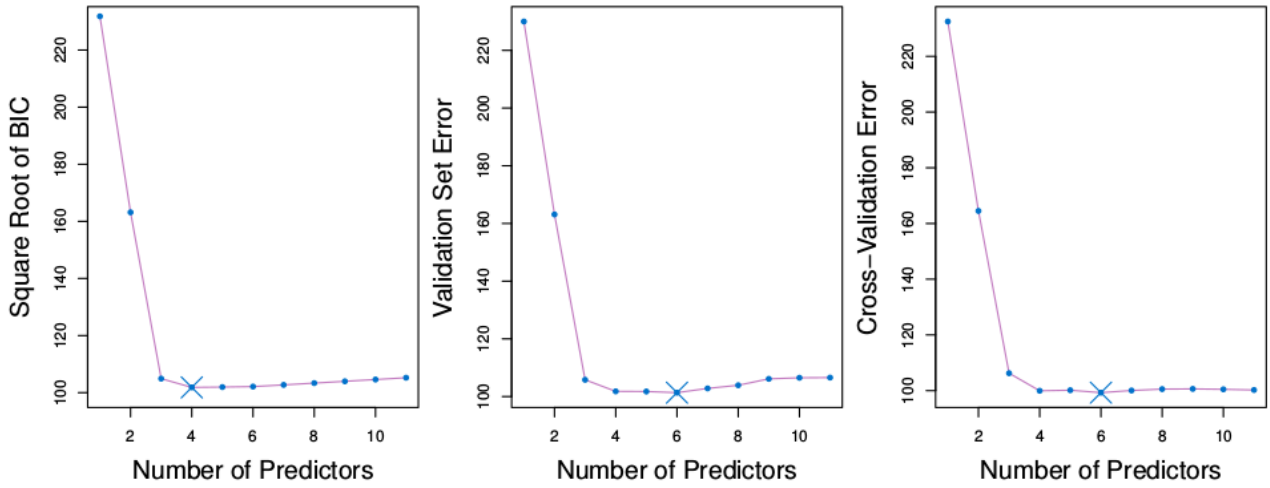
$$R^2 = 1 - \frac{RSS}{TSS}$$

$$Adjusted R^2 = 1 - \frac{\frac{RSS}{n-p-1}}{\frac{TSS}{n-1}}$$

    - 最大化Adjusted R Square即是最小化$RSS/(n-p-1)$，而变量数增加RSS减少
    - $RSS/(n-p-1)$取决于变量数p
    - 与R平方不同在于，Adjusted R Square考虑到了加入的无关变量

6.4 直接法: `Validation set/ cross-validation`

- 模型 $M_1, M_2, \ldots M_k$
- 将每个模型的交叉验证集/交叉验证误差纳入考虑，并选择能最小化test error估计值的k
- 相对于之前AIC/BIC等的优点
    - 可直接估计test error，且不需要估计Var(error)
    - 适用范围更广，包含难以估测自由度(预测因子数)/Var(error)的情形



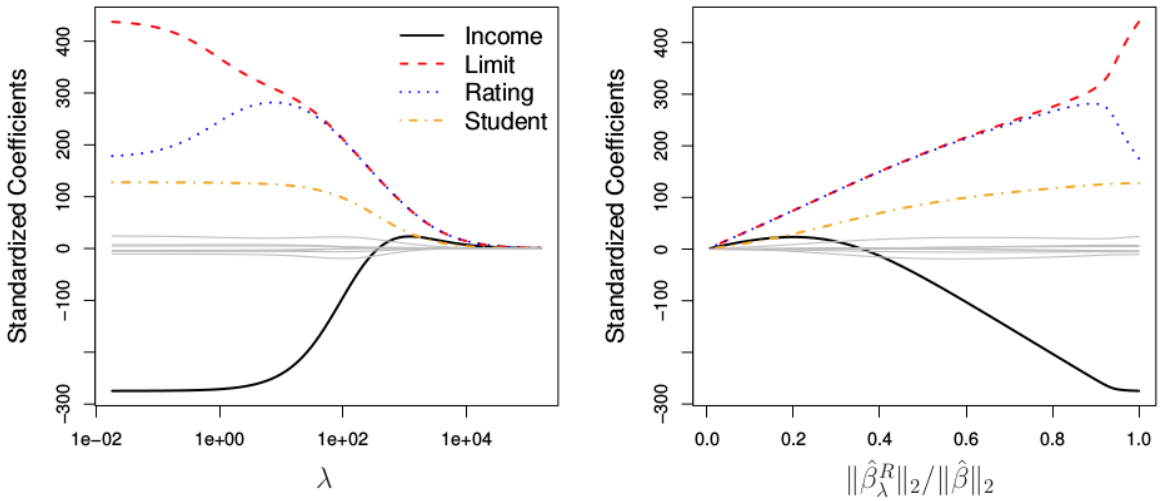
6.5 Shrinkage

6.5.1 Ridge regression

- 岭回归: 限制系数模的平方
- 之前通过最小化RSS拟合系数：$RSS = e_1^2 + e_2^2 + \ldots + e_n^2$
- 岭回归最小化RSS+shrinkage penalty

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}) + \lambda \sum_{j=1}^{p} \beta^2$$

- $\lambda \geq 0$决定RSS和惩罚项的权重
- λ过大会导致underfitting (模型过于简单)
- 栗子:



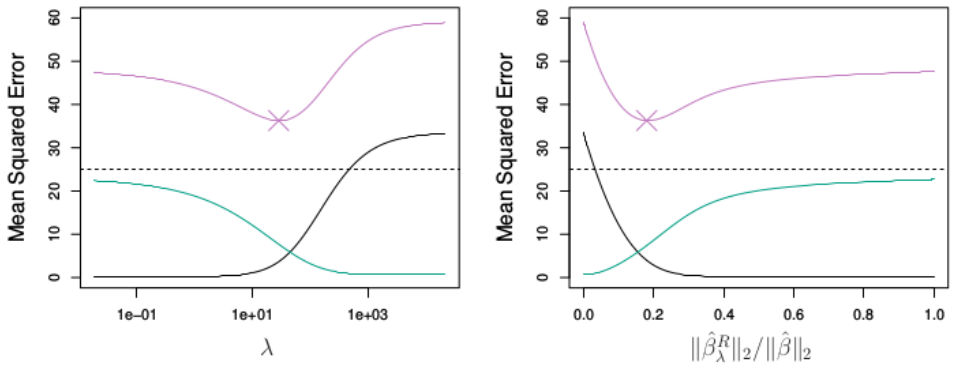
- In the left-hand panel, each curve corresponds to the ridge regression coefficient estimate for one of the ten variables, plotted as a function of λ.
- The right-hand panel displays the same ridge coefficient estimates as the left-hand panel, but instead of displaying λ on the x-axis, we now display $\|\hat{\beta}_\lambda^R\|_2/\|\hat{\beta}\|_2$, where $\hat{\beta}$ denotes the vector of least squares coefficient estimates.
- The notation $\|\beta\|_2$ denotes the ℓ_2 norm (pronounced "ell 2") of a vector, and is defined as $\|\beta\|_2 = \sqrt{\sum_{j=1}^{p} {\beta_j}^2}$.

6.5.2 预测因子归一化

- 标准最小二乘法估计系数是尺度相同的,无论X_j尺度如何，$X_j\beta_j$始终不变

- 有了惩罚项，岭回归可以改变尺度
- 可以在先选择合适的预测因子，再引入岭回归对系数进行优化

The Bias-Variance tradeoff



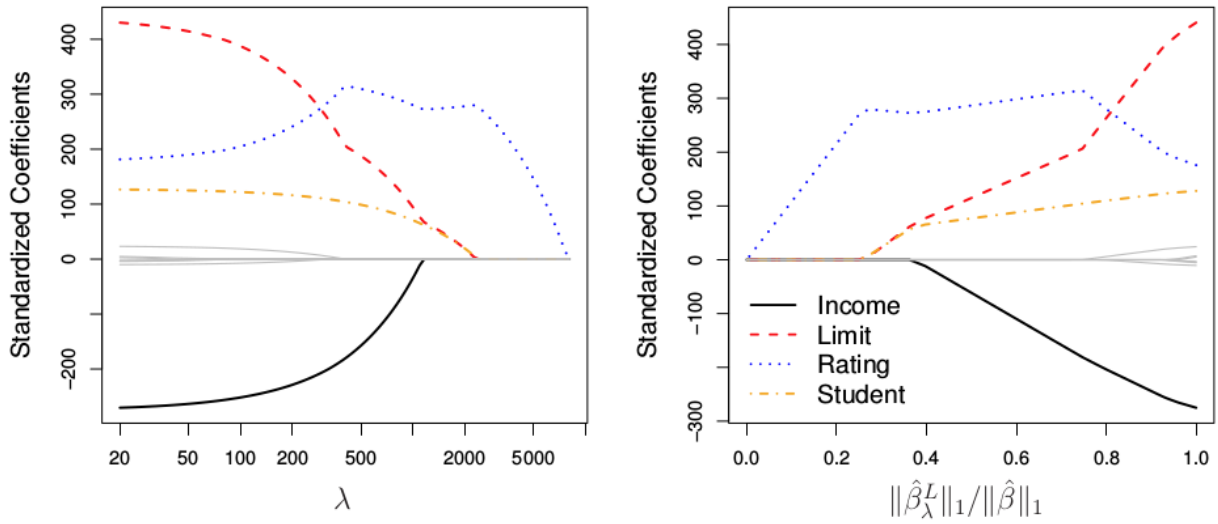
Simulated data with $n = 50$ observations, $p = 45$ predictors, all having nonzero coefficients. Squared bias (black), variance (green), and test mean squared error (purple) for the ridge regression predictions on a simulated data set, as a function of λ and $\|\hat{\beta}^R_\lambda\|_2/\|\hat{\beta}\|_2$. The horizontal dashed lines indicate the minimum possible MSE. The purple crosses indicate the ridge regression models for which the MSE is smallest.

6.5.3 Lasso (Least Absolute Shrinkage and Selectionator operator)

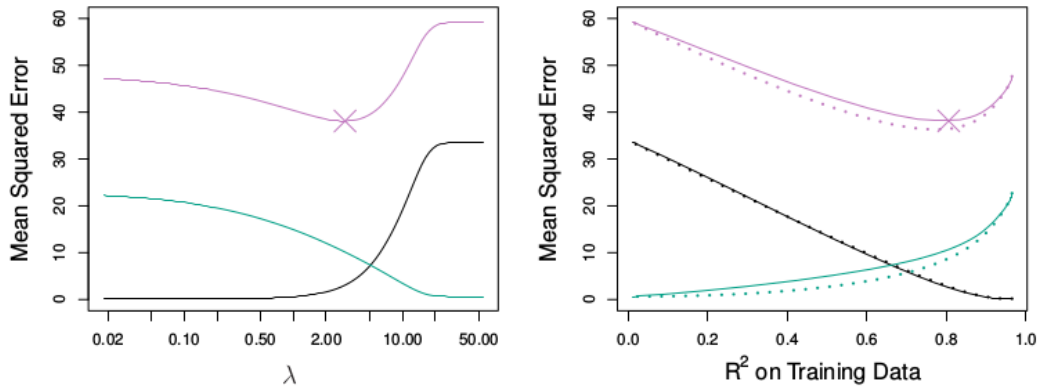
- 岭回归的不足：和子集选择不同，岭回归的结果包含了所有的预测因子
- Lasso系数可以通过最小化

  $RSS + \lambda * (\sum_{j=1}^{p} |\beta|)$

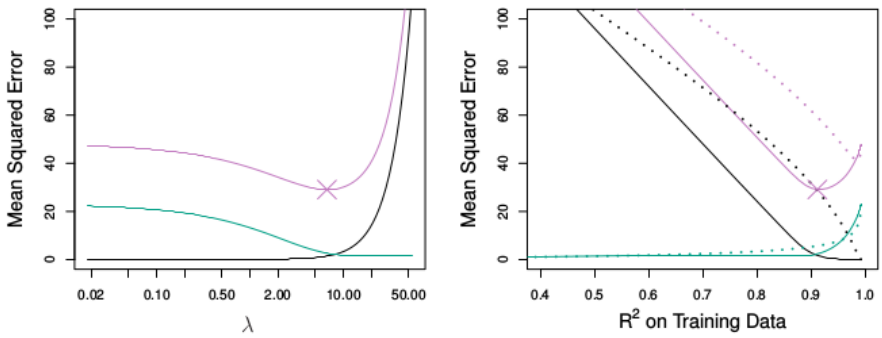
  对变量数量进行优化，减少无关变量 (类似subset selection)



6.5.4 Lasso和Ridge Regression的比较



Left: Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso on simulated data set of Slide 32. Right: Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed). Both are plotted against their R^2 on the training data, as a common form of indexing. The crosses in both plots indicate the lasso model for which the MSE is smallest.

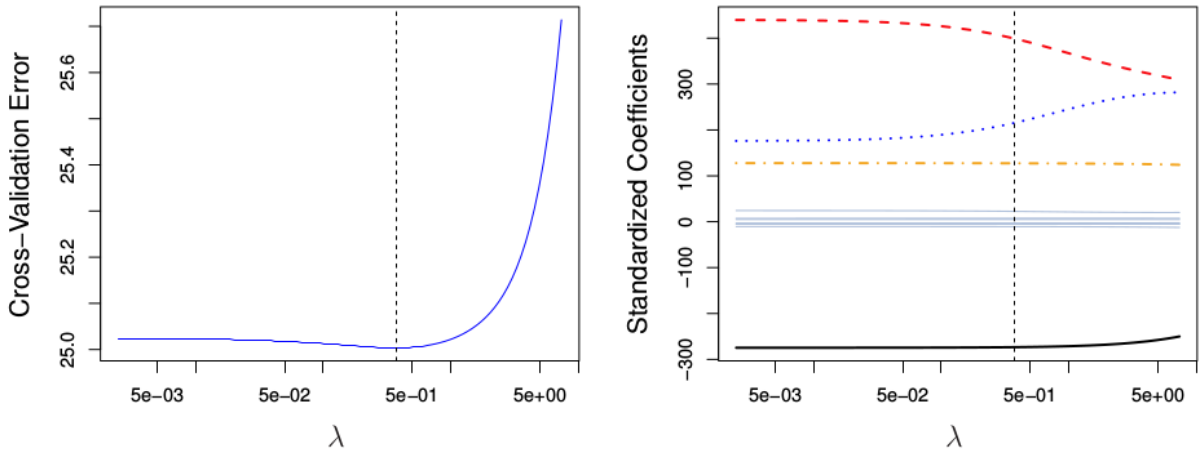


Left: Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso. The simulated data is similar to that in Slide 38, except that now only two predictors are related to the response. Right: Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed). Both are plotted against their R^2 on the training data, as a common form of indexing. The crosses in both plots indicate the lasso model for which the MSE is smallest.

- 都是对系数的模进行限制, 其中岭是模平方，而lasso是模长度;
- 岭回归有助于提升共线性，lasso要生硬一些;
- Lasso可以在压缩系数的同时减少变量;
- 岭回归和Lasso各有所长，不可互相替代;
- Response是一个预测因子少的函数时，lasso效果更好;
- 然而在实际情况下，与response相关的预测因子数量是无法先验priori的;
- 可以使用CV等技术进行具体问题具体分析,选择合适的方法

6.6 Tuning parameter selection

- 先使用通过使用不同的λ得到不同的J(θ)及对应的θ向量;
- 通过CV对θ向量进行评估，从而选取合适的λ



Left: *Cross-validation errors that result from applying ridge regression to the* `Credit` *data set with various values of* λ. Right: *The coefficient estimates as a function of* λ. *The vertical dashed lines indicates the value of* λ *selected by cross-validation.*

6.7 Dimension Reduction Methods

- `PCR+PLS`
- 转换参数，并用转换过的参数fit最小二乘模型
- $X_1 - X_p$代表初始的p个预测因子
- $Z_1 - Z_M (M < p)$代表将初始preditor进行线性组合后得到的新预测因子

$$Z_M = \sum_{j=1}^{p} \varphi_{mj} X_j, \varphi_{mj}\text{为常数}$$

- 降维后使用$Z_1 - Z_M$进行最小二乘线性拟合了：

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \varepsilon_i, i = 1, \dots n$$

- 经转换可得:

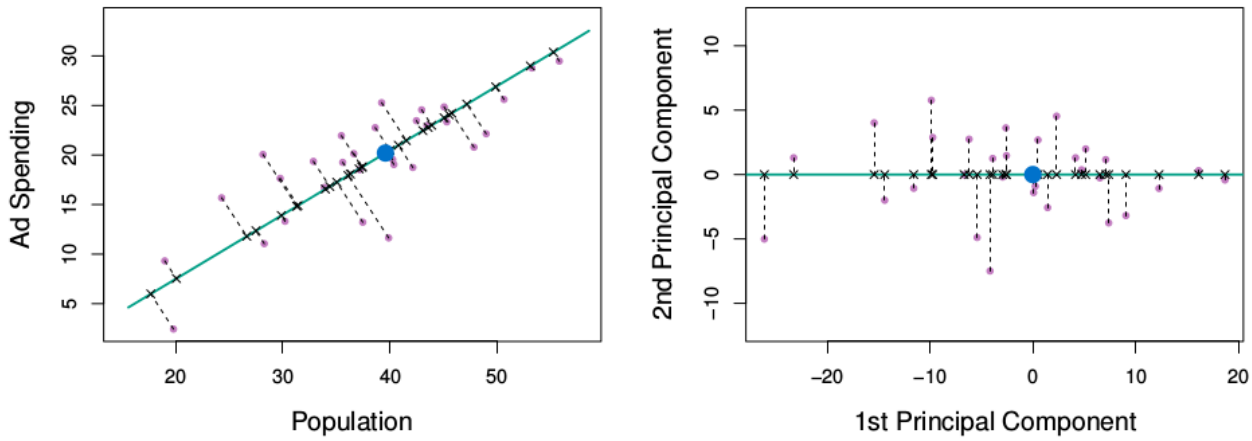
$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \varphi_{mj} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} (\theta_m \varphi_{mj}) x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij}$$

- 其中:

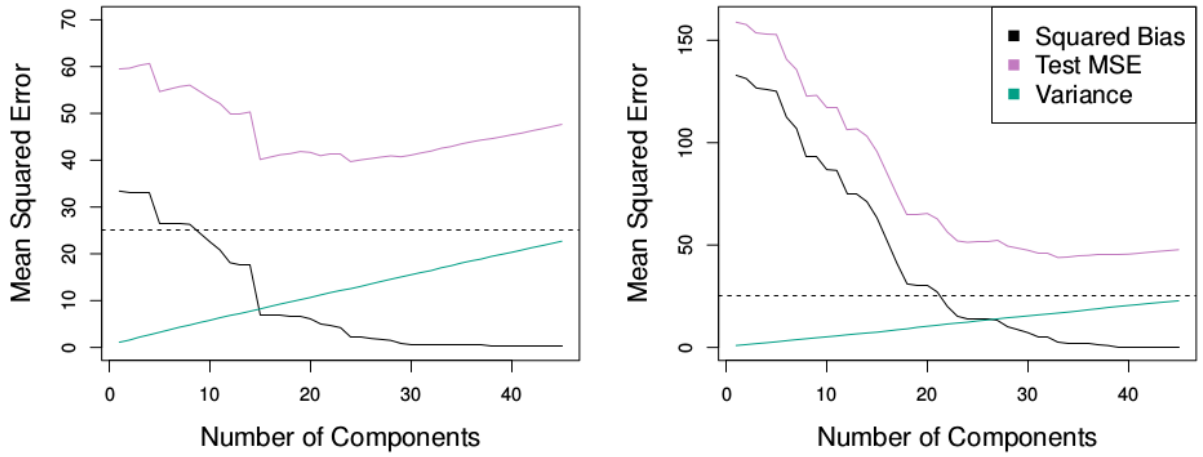
$$\beta_j = \sum_{m=1}^{M} \theta_m \varphi_{mj}$$

6.8 Principal Components Regression and Partial Least Squares

6.8.1 Principal Components Regression(PCR)



- 这个在后面非监督学习时也有提及(PCA)，可参考
- 引入principal components analysis(PCA)来定义预测因子的线性组合
  - 第一个principal component: 具有最大方差的变量的线性组合
  - 第二个principal component:与第一个不相关的方差最大的线性组合
  - 以此类推，每个线性组合尽可能方差大，且与之前的所有组合相互正交
- 栗子:



PCR was applied to two simulated data sets. The black, green, and purple lines correspond to squared bias, variance, and test mean squared error, respectively. Left: *Simulated data from slide 32.* Right: *Simulated data from slide 39.*

6.8.2 Partial Least Squares(PLS)

- PCR通过线性组合预测因子来优化，是一种非监督方法
- PCR的一大缺点：无法保证新预测因子是预测response的最佳选择
- PLS通过引入Y的作用，可以监督的形式定义M：
  - 使预测因子既能符合原始变量，又可以与response相关
  - 既可以解释预测因子，又可以解释response
- details：
  - 标准化原始预测因子后，$Z_M = \sum_{j=1}^{p} \varphi_{mj} X_j$;
  - φ_{mj}与与Y和Xj的线性回归式的系数成比例;
  - 与结果相关度越高的变量，其权重系数越大;

6.9 Model Selection in R

6.9.1 最佳子集模型 `regsubsets in leaps package`

```
regfit_full<-regsubsets(salary~,data=hitters,nvmax=19) #默认method

reg_summary<-summary(regfit_full)
plot(reg_summary$cp,xlab="Number of Variables",ylab="Cp")
which.min(reg_summary$cp) #给出最小值的位置,如10
points(10,reg_summary$cp[10],col="red") #将最小值变为红色

coef(regfit_full,10) #选择第十个模型，提取系数
```

6.9.2 Forward Stepwise Selection `regsubsets in leaps package`

```
regfit.fwd<-regsubsets(Salary~.,data=Hitters,
        nvmax=19, #19 models
        method="forward") #将method调为forward

plot(regfit.fwd,scale="Cp")
```

6.9.3 Model Selection Using a Validation Set

```
val.errors=rep(NA,19) #set a vector to record errors
x.test=model.matrix(Salary~.,data=Hitters[-train,])

for(i in 1:19){
    coefi=coef(regfit.fwd,id=i)
```

```
    pred=x.test[,names(coefi)]%*%coefi
    val.errors[i]=mean((Hitters$Salary[-train]-pred)^2)
}

plot(sqrt(val.errors),ylab="RootMSE",ylim=c(300,400),pch=19,type="b")

points(sqrt(regfit.fwd$rss[-1]/180),col="blue",pch=19,type="b")

legend("topright",legend=c("Training","Validation"),col=c("blue","black
"),pch=19)
```

6.9.4 Model Selection by Cross-Validation

```
folds=sample(rep(1:10,length=nrow(Hitters)))
cv.errors=matrix(NA,10,19) #和前面一样，这个空向量用于存放errors

for(k in 1:10){
    best.fit=regsubsets(Salary~.,data=Hitters[folds!=k,],nvmax=19,metho
d="forward")
    for(i in 1:19){
        pred=predict(best.fit,Hitters[folds==k,],id=i)
        cv.errors[k,i]=mean( (Hitters$Salary[folds==k]-pred)^2)
    }
}

rmse.cv=sqrt(apply(cv.errors,2,mean))
plot(rmse.cv,pch=19,type="b")
```

6.9.5 Ridge Regression and the Lasso `glmnet package`

```
library(glmnet)
x=model.matrix(Salary~.-1,data=Hitters)
y=Hitters$Salary

#岭回归（这个岭回归同时也进行了CV）
fit.ridge=glmnet(x,y,alpha=0)
plot(fit.ridge,xvar="lambda",label=TRUE)
cv.ridge=cv.glmnet(x,y,alpha=0) #cv.glmnet()
plot(cv.ridge)

#fit.lasso=glmnet(x,y)
plot(fit.lasso,xvar="lambda",label=TRUE)
cv.lasso=cv.glmnet(x,y)#default alpha=0
coef(cv.lasso)
```

6.10 Summary and Quiz

6.10.1 Summary

- 优化线性模型的三种方法：
  - Subset selection: 选取一部分预测因子进行拟合，减少无用变量，共2^p个选择
  - Forward/backward stepwise可以进一步优化
    - 仅仅需要$1+\frac{p(p+1)}{2}$个选择
    - 但RSS略大于subset selection
- 选择参数标准:
  - 在参数相同的项目中选取最佳：最小化RSS，最大化R^2
  - 在参数不同的项目中选取最佳：最小化test error
  - 间接法：Cp,AIC,BIC,Adjusted R^2
  - 直接法：使用validation set/cross-validation直接估测test error
- Shrinkage:让一些系数为0
  - 最小化RSS+λ*模参量
  - Ridge Regression VS Lesso
    - 都是对系数的模进行限制;其中岭是模平方，而lasso是模长度
    - 岭回归有助于提升共线性，lasso要生硬一些;
    - lasso可以在压缩系数的同时减少变量
  - Tuning parameter λ 使用CV获取
- Dimension reduction：将预测因子进行线性组合，减少无用变量
- PCR VS PLS：
  - PCR为尽可能最大化principle component之间的方差，为非监督方法
  - PLS则考虑到了Y，对结果有显著作用的预测因子会被给予更多的权重，为supervised

6.10.2 Quiz (主要是关于shrink里面λ的)

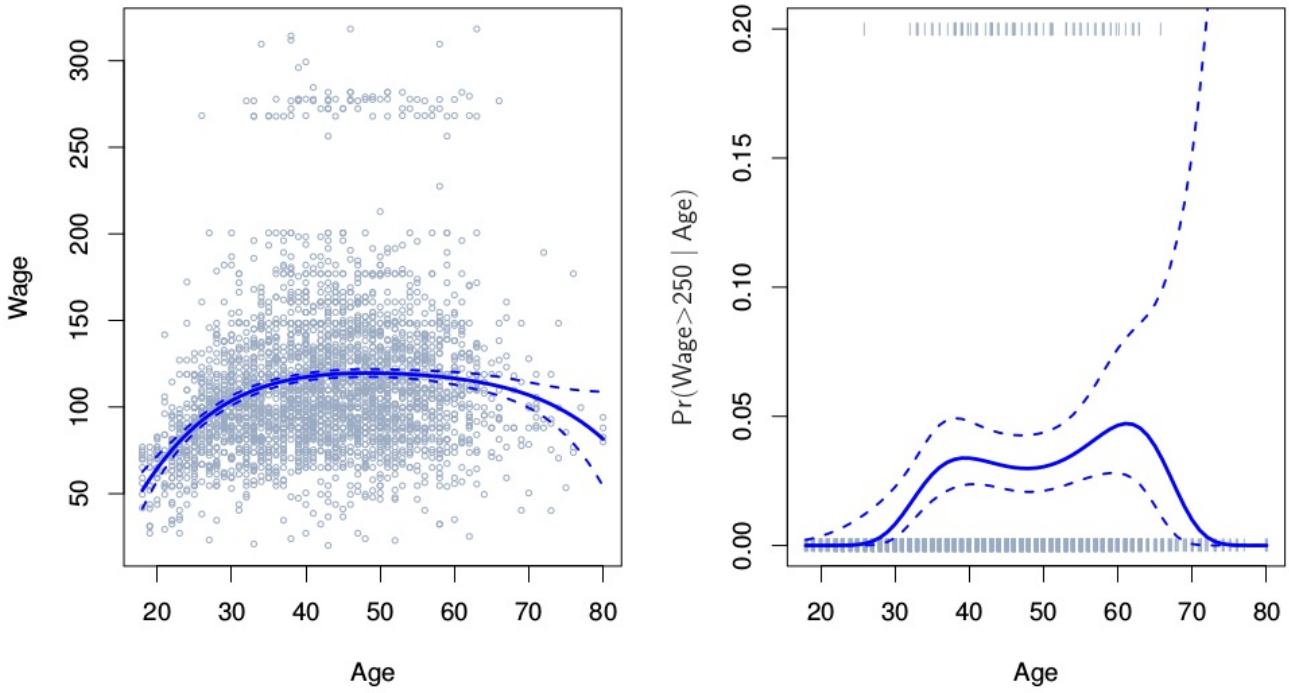
- λ增加，系数模权重增加，相当于更多的θj被设置为0，模型被简化
- 对于更简化的模型，Var逐渐降低，Bias逐渐升高(underfitting)
  - Training error逐渐升高
  - Test error先降低后升高
  - Inreducible error 不变(固有误差，与model无关)

Chapter 7 Non-linear Model

7.1 Polynomial and Step Functions

7.1.1 polynomial regression 多项式回归

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



- 相对于系数更看重任意样本x_0输入拟合函数后得到的结果

$$f'(x_0) = \beta'_0 + \beta'_1 x_0 + \beta'_2 x_0^2 + \ldots + \beta i' x_0^i$$

- 得到逐点方差 $Var[f'(x0)]$以及逐点标准误$f'(x_0) \pm 2 * se[f'(x_0)]$
- 通过拟合或CV等方法来选择幂数d
- 多项式逻辑回归:

$$Pr(y_i > 250 | x_i) = \frac{exp(\beta_0 + \ldots \beta_d x_i^d)}{1 + exp(\beta_0 + \ldots \beta_d x_i^d)}$$

- 得到置信区间:先归一化，再转化成probability scale
- 注意：多项式回归tail behavior性能很差,很难进行逆推
- Can fit using `y ~ poly(x,degree = 3)` in formula

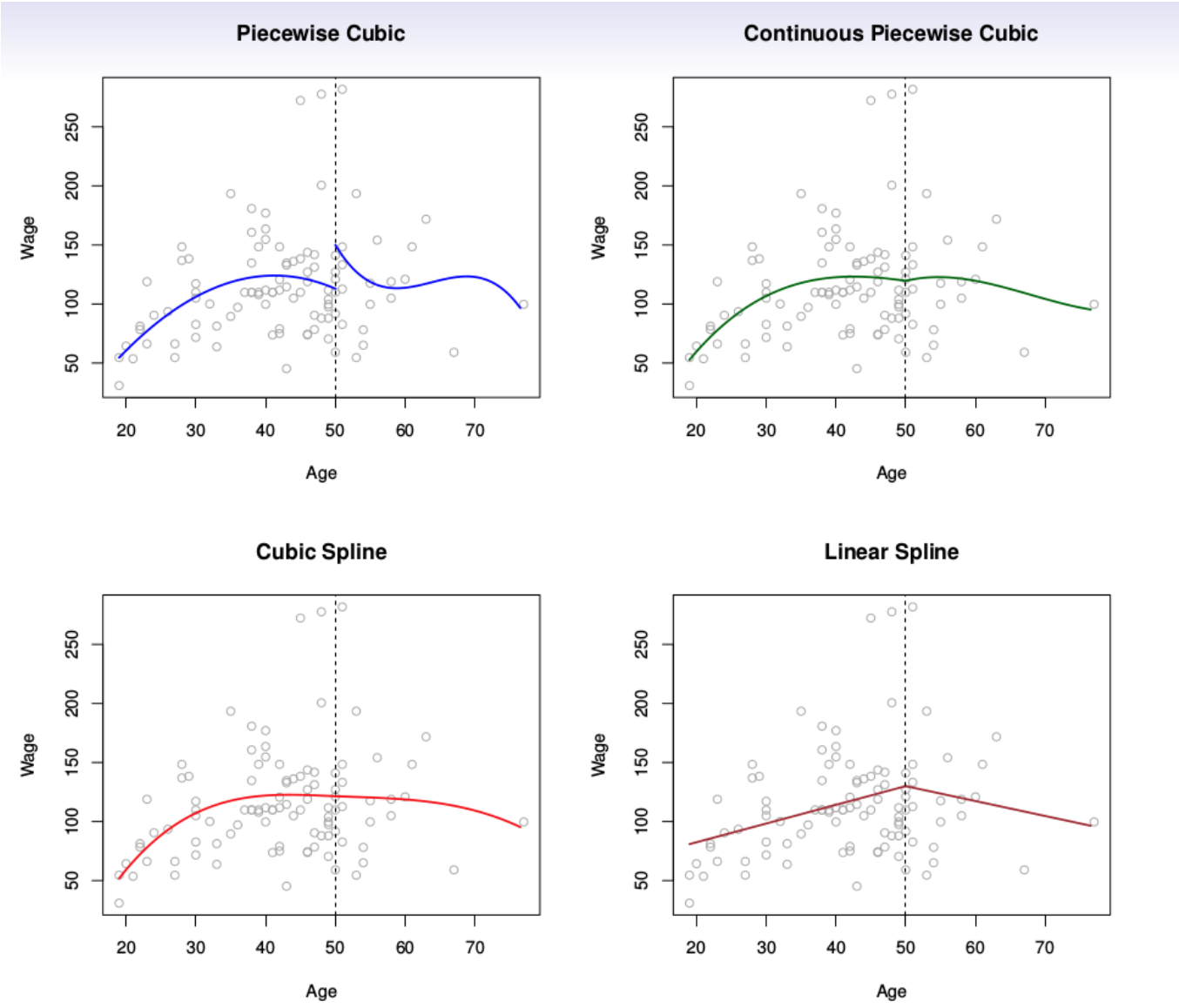
7.1.2 Step Functions

- Creates a series of dummy variables representing each group;
- Useful way of creating interactions that are easy to interpret;

```
cut(age,c(18,25,40,65,90))
```

- Choice of cutpoints or knots can be problematic
  - alternative : splines

7.2 Piecewise-Polynomials and Splines

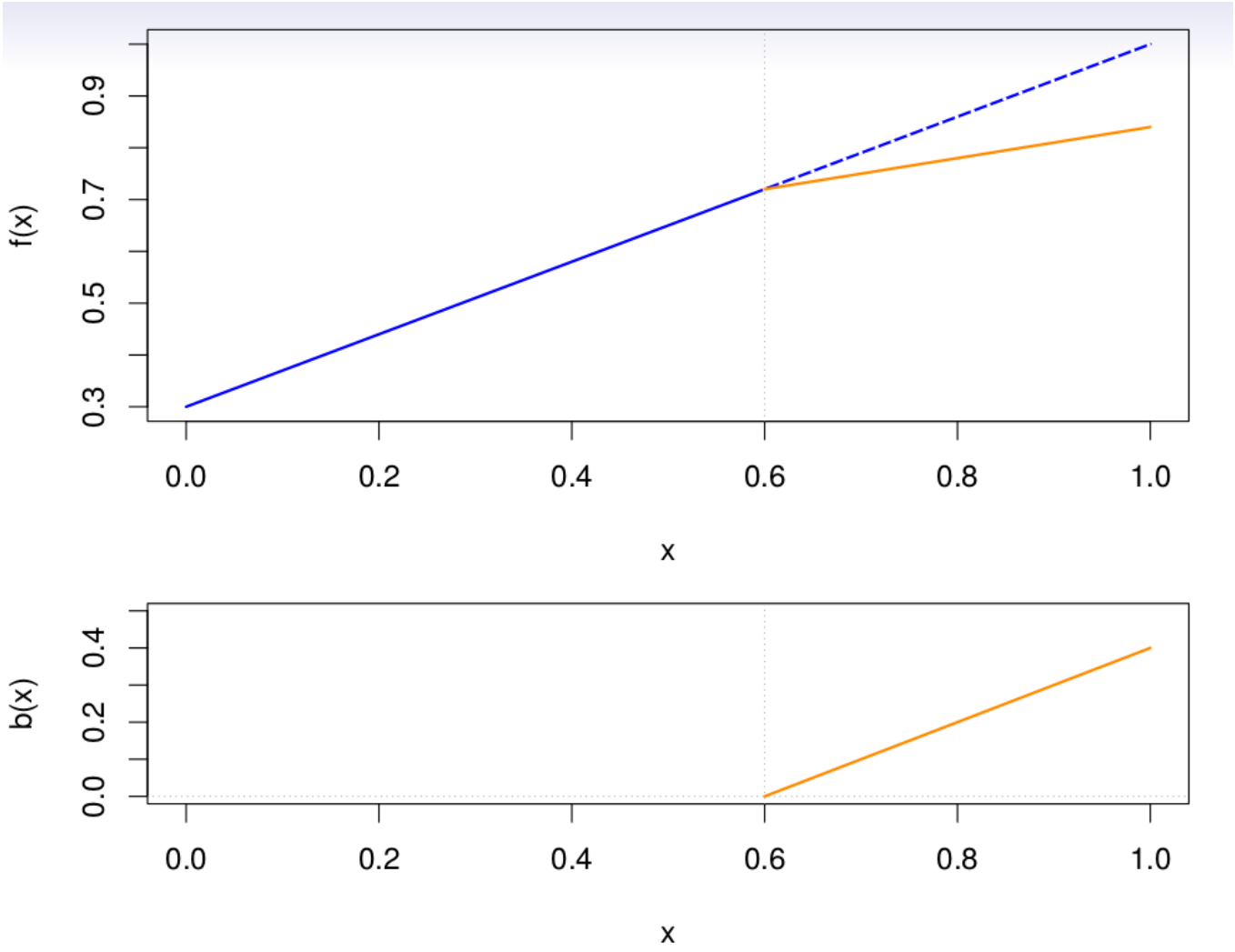


7.2.1 Piecewise-Polynomials分段多项式

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + e_i, & \text{if } x_i < \text{c} \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + e_i, & \text{otherwise} \end{cases}$$

- Better to add constraints to the polynomials, e.g. continuity.
- splines的连续性是最佳的

7.2.2 Linear Splines



- Splines:给定一组控制点1,2,...K而得到一条曲线

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

  - K个预测因子
  - b_k:basis functions

$$b_1(x_i) = x_i$$

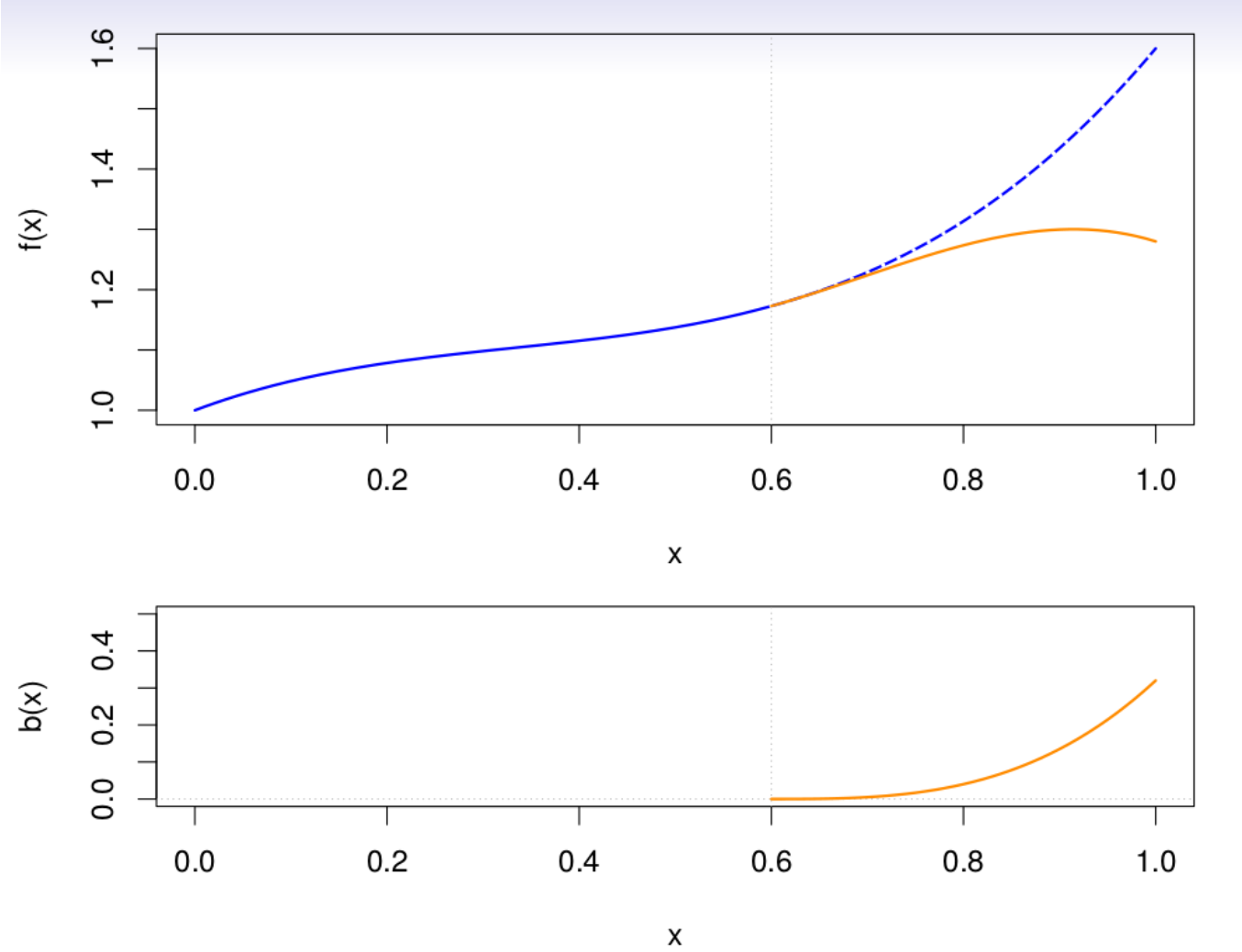
$$b_2(x_i) = x_i^2$$

$$b_{k+3}(x-i) = (x_i - \xi_k)_+, k = 1, \ldots, K$$

- 带有$()_+$的意为positive part，例如下面的分段不等式：

$$(x_i - \xi_k)_+ = \begin{cases} (x_i - \xi_k)_+ = x_i - \xi_k, & \text{if } x_i > \xi_k \\ 3n + 1, & \text{otherwise} \end{cases}$$

7.2.3 Cubic Splines



- 立方样条,格式跟前面类似，只是最后一项变成了立方
- 举个栗子:自由度(参数数量)为K+4的立方样条

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

$$b_1(x_i) = x_i$$

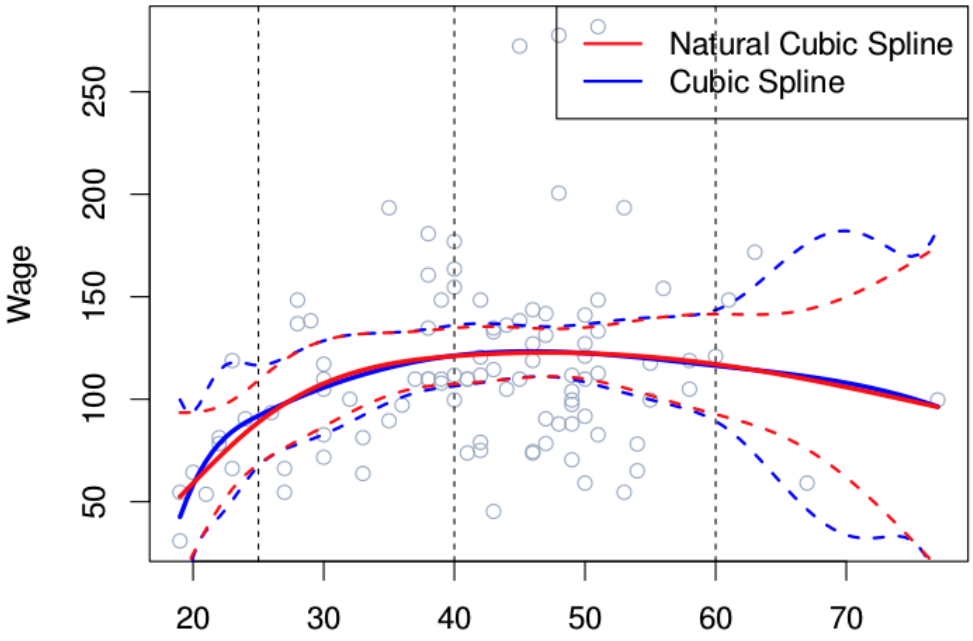
$$b_2(x_i) = x_i^2$$

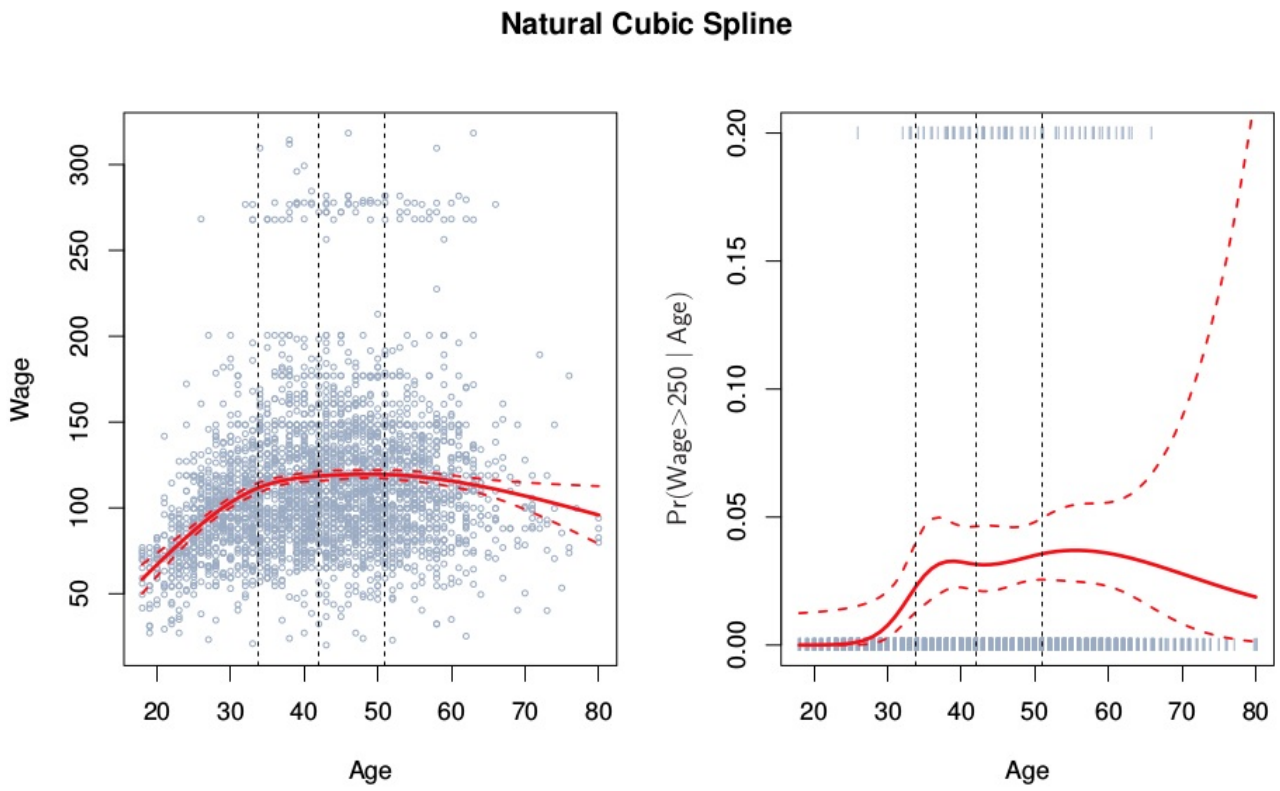
$$b_3(x_i) = x_i^3$$

$$b_{k+3}(x_i) = (x_i - \xi_k)_+^3, k = 1, \ldots, K$$

7.2.4 Natural Cubic Splines

A natural cubic spline extrapolates linearly beyond the boundary knots. This adds $4 = 2 \times 2$ extra constraints, and allows us to put more internal knots for the same degrees of freedom as a regular cubic spline.





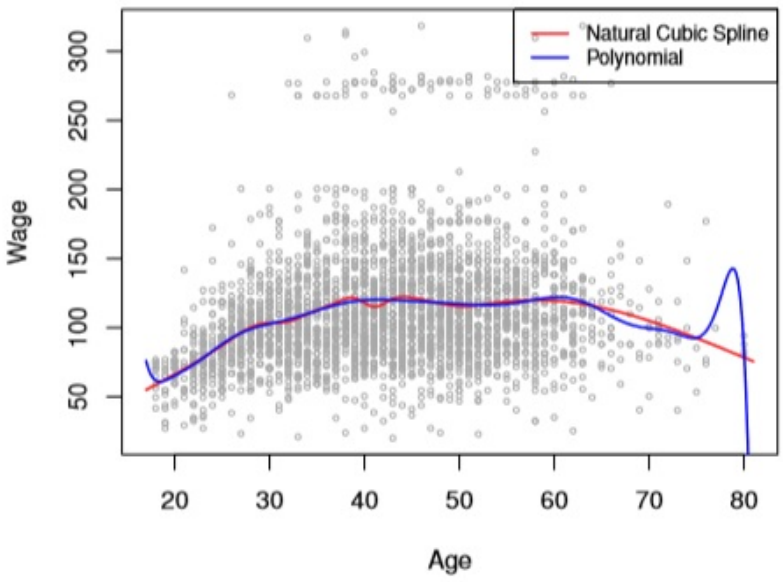
- 下面的几个函数在·splines package·里

```
bs() #any degree splines
ns() in splines package #natural cubic splines
```

7.2.5 Knot placement

- 确定K个knots，将其置于合适的观察值X的分位,进行拟合
- cubic spline with K knots：K+4个参数(即自由度为K+4)
- natural spline with K knots ：K级自由度

```
ns(age,df=14)
poly(age,deg=14)
```



Comparison of a degree-14 polynomial and a natural cubic spline, each with 15df.

- NS比polynomial曲线更平滑

7.3 Smoothing Spines

- Consider this criterion for fitting a smooth function g(x) to some data

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

  - 第一项RSS: 努力让g(x)符合任意一个xi
  - 第二项roughness penalty:控制g(x)的弯曲度,可以用tuning para λ进行调节
  - λ>0,越小函数越弯曲，λ->无穷大即线性函数
- details:
  - 其实就是搞了个natural cubic spline with a knot at every unique value of xi
  - Smoothing splines避免了knot选择问题,leaving a single λ to be chosen
  - `smooth.spline()` in R
- effective degrees of freedom 有效自由度

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

- 选择λ：
  - 可以指定有效自由度df而非λ`smooth.spline(age,wage,df = 10)`

- LOOCV error：`smooth.spline(age,wage)`

7.4 Generalized Additive Models and Local Regression

7.4.1 Generalized Additive Models

- 允许 flexible nonlinearities in several variables，但保持线性模型的可加结构

$$y_i = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \ldots + f_p(x_{ip}) + e_i$$

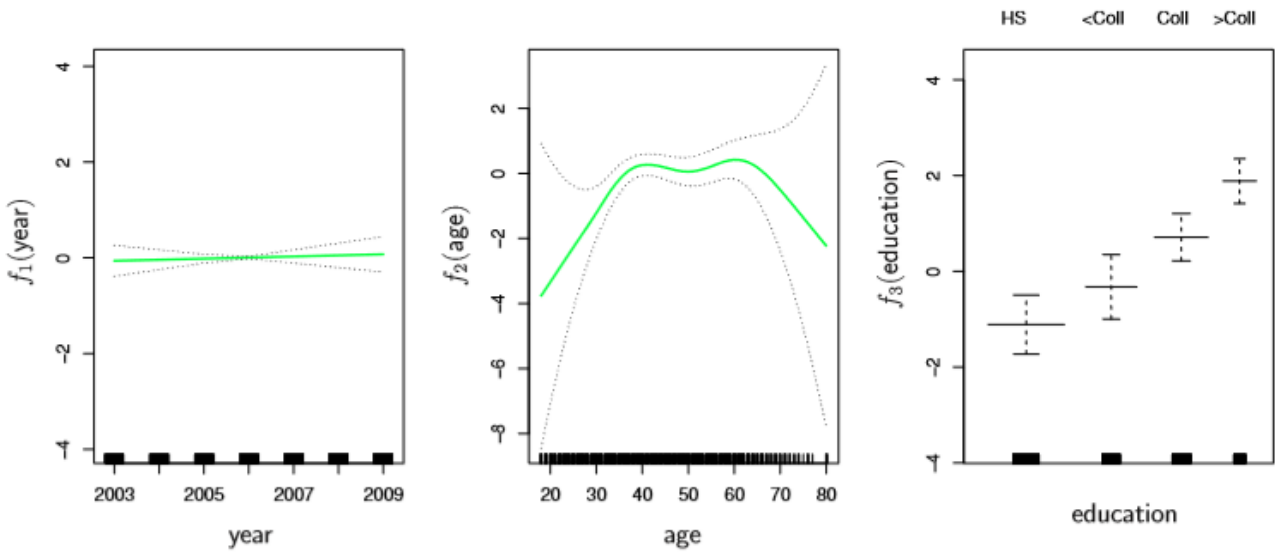
- details:
  - 可以将几个函数放到一起拟合
  - 相比系数更看重fitted function，使用`plot.gam`制图
  - 混合线性-非线性单元，并使用`anova()`进行比较
  - Can use [smoothing splines] or [local regression]

```
lm(wage~ns(year,df = 5) + ns(age,df = 5) + education)

gam(wage~s(year,df = 5)+lo(age,span = .5)+education)
```

- GAMs for classification

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

```
gam(I(wage > 250)~year + s(age,df = 5) + education,family = binomial)
```



7.4.2 Local Regression

- sliding weight function
- fit separate fits over the range of X by weighted least squares
- `loess()`

7.5 Non-linear Functions in R

7.5.1 polynomials

```
fit<-lm(mpg~poly(hp,4),data=mtcars) #polynomial,df=4
horsep<-range(hp) #这个返回的是min&max
hpg<-seq(from=horsep[1],to=horsep[2]) #从hp的min到max

predhp<-predict(fit,newdata = list(hp=hpg),se=TRUE)
se_bands<-cbind(predhp$fit+2*predhp$se.fit,predhp$fit-2*predhp$se.fit)
#standard error bands标准误差带

plot(hp,mpg,col="red")
lines(hpg,predhp$fit,lwd=2,col="blue") #添加趋势线
matlines(hpg,se_bands,col="green",lty=2) #添加标准误差带
#matlines:可以直接添加matrix
```

- 另一种方法，直接搞出四阶polynomial：

```
fita<-lm(mpg~hp+I(hp^2)+I(hp^3)+I(hp^4),data=mtcars)
#跟前面比p不同-使用了different bases of polynomial
```

7.5.2 使用anova来评估最适合的model

```
fita<-lm(mpg~wt,data=mtcars)
fitb<-lm(mpg~wt+hp,data=mtcars)
fitc<-lm(mpg~wt+poly(hp,2),data=mtcars)
fitd<-lm(mpg~wt+poly(hp,3),data=mtcars)

anova(fita,fitb,fitc,fitd) #比较哪个模型更适合

>>>p最小的是fitb，最适合
```

7.5.3 logistic regression

```
fit<-glm(I(mpg>50)~poly(hp,3),data=mtcars,family=binomial)
```

```
preds<-predict(fit,list(hp=hpg),se=T)
se_bands<-preds$fit+cbind(fit=0,lower=-2*preds$se,upper=2*predhp$se)
#这里跟前面不大一样，给出了三列，fit/lower/upper

prob_bands<-exp(se_bands)/(1+exp(se_bands)) #得到prob_bands,公式参见7.1.1

matplot(hpg,prob_bands,col="blue",lwd=c(2,1,1),lty=c(1,2,2),type="l",ylim=c(0,.1)) #给出预测的prob_bands及其confidence intervals
```

7.5.4 Splines `library(splines)`

- Cubic Splines：基本方法跟前面类似

```
fit<-lm(mpg~bs(hp,knots=c(25,40,60)),data=mtcars)
plot(hp,mpg,col="darkgrey")
lines(hpg,predict(fit,list(hp=hpg)),col="darkgreen",lwd=2)
abline(v=c(25,40,60),lty=2,col="darkgreen")
```

- Smoothing Splines: 不需要指定knots，但需要平滑参数(自由度)

```
fit=smooth.spline(hp,mpg,df=16)
lines(fit,col="red",lwd=2)

#使用LOOCV来自动选择smoothing paramete
fit=smooth.spline(hp,mpg,cv=TRUE)
```

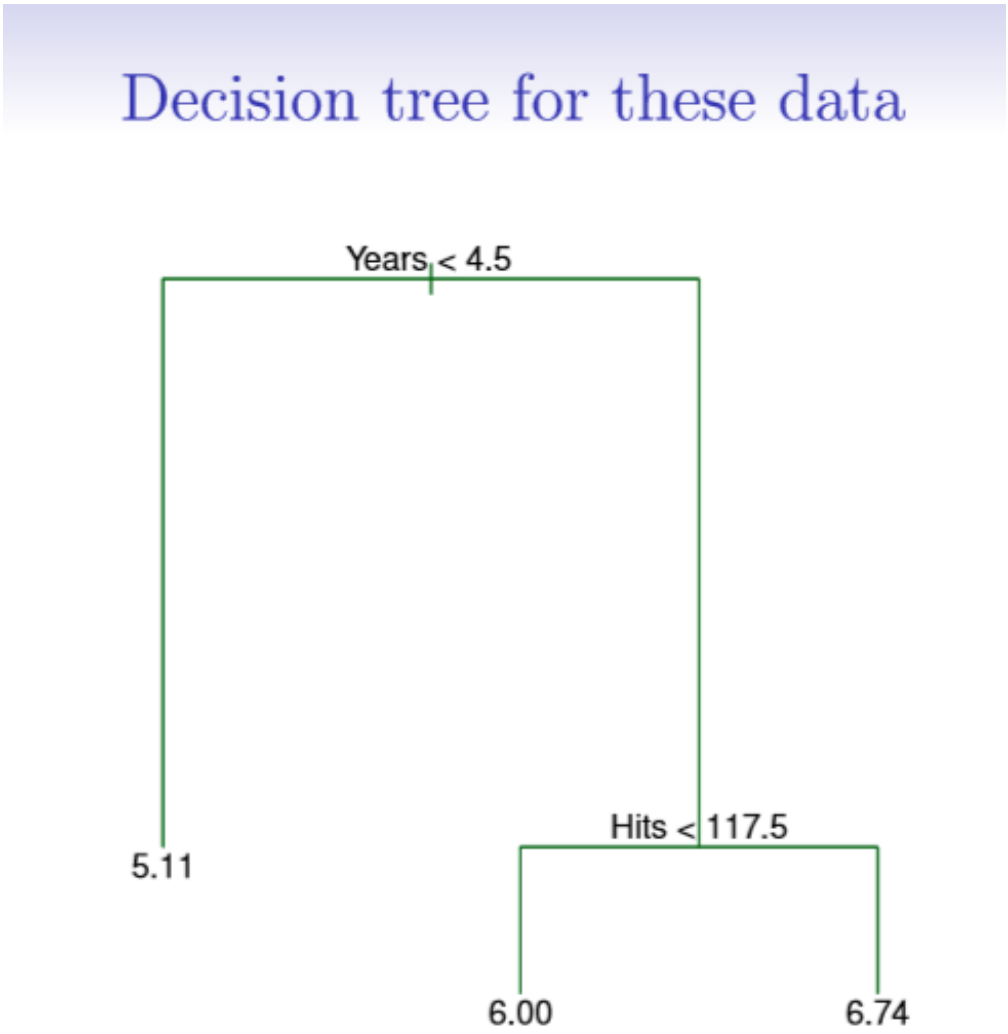
7.5.5 GAM `library(gam)`

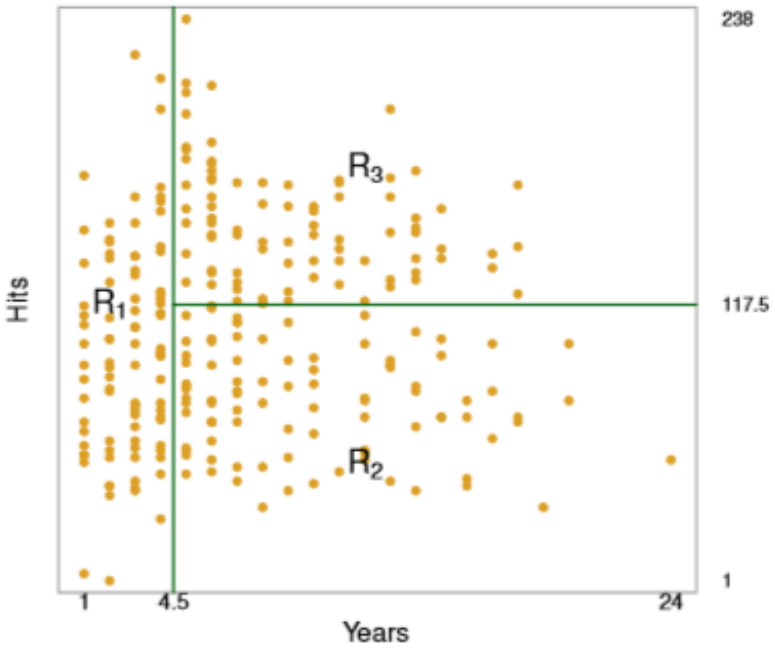
```
gam1<-gam(mpg~s(hp,df=4)+s(wt,df=4)+cyl,data=mtcars)
gam2<-gam(I(mpg>25)~s(hp,df=4)+s(wt,df=4)+cyl,data=mtcars,family=binomial)

par(mfrow=c(1,3))
plot(gam1,se=T)
plot(gam2)

#检验是否需要非线性项wt
gam2a<-gam(I(mpg>25)~s(hp,df=4)+wt+cyl,data=mtcars,family=binomial)

anova(gam2a,gam2,test="Chisq")
```

7.6 Summary and Quiz

Chapter 8 Tree-Based Methods

- 将预测因子空间stratifying分层or segmenting分割成几个simple regions
- 举个栗子





8.1 Tree-Based Methods

- decision-tree methods
- application: regression/classification

8.1.1 Terminology for Trees

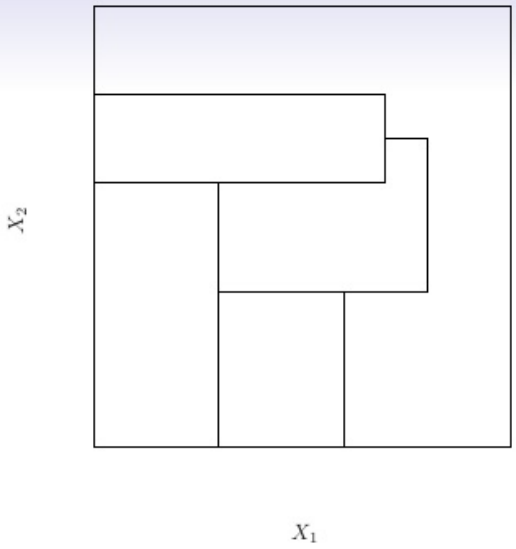
- terminal nodes:将观察值分为数个regions，称作终端节点
- upside down：决策树一般都是上枝干下树叶的
- internal nodes：将预测因子空间分为数个regions，称作内部节点

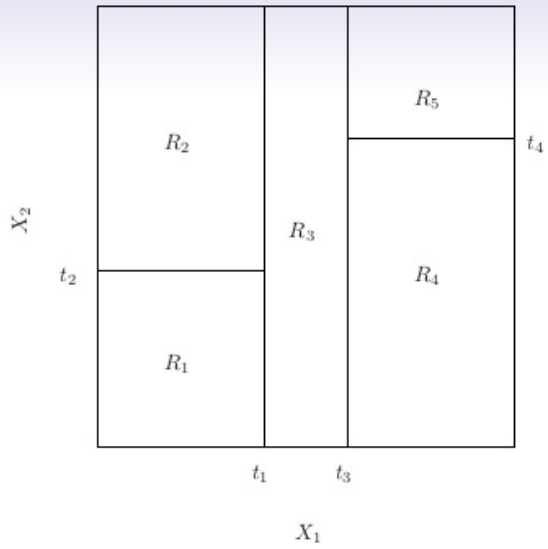
8.1.2 Some details

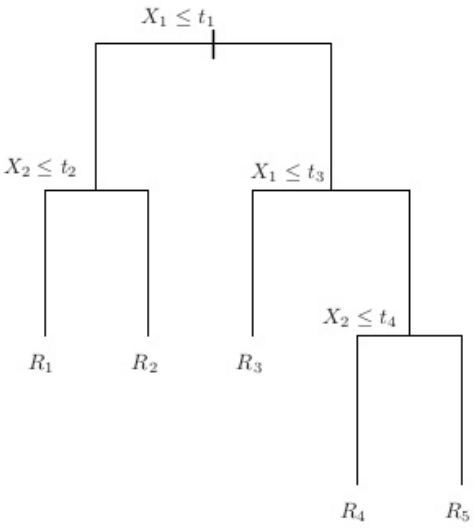
- devide 预测因子空间$X_1, X_2, \dots X_p$ into J个不相互重叠的部分$R_1, R_2, \dots R_j$
- 对区域R_j中的所有观察值取相同prediction
- 理论上regions形状多样，而我们将预测因子空间按high-dimensional boxes划分以提升interpretation
- goal:find boxes that minimize RSS(y'_{R_j}该区域观察值返回结果的均值)

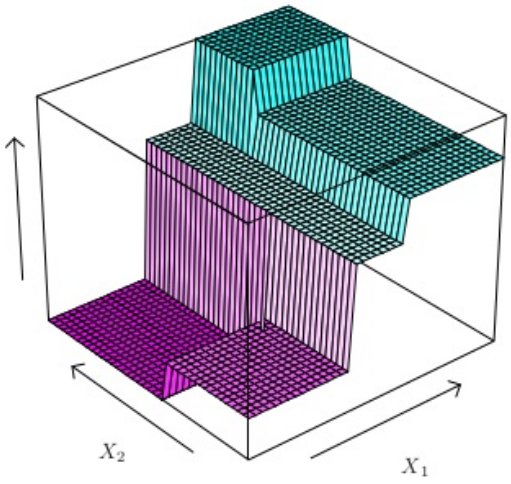
$$\sum_{i \in R_j} (y_i - y'_{R_j})^2$$

8.1.3 栗子


X_2
X_1


R_1
R_2
R_3
R_4
R_5
t_1
t_2
t_3
t_4
X_2
X_1


$X_1 \leq t_1$
$X_2 \leq t_2$
$X_1 \leq t_3$
$X_2 \leq t_4$
R_1
R_2
R_3
R_4
R_5


X_2
X_1

Top Left: A partition of two-dimensional feature space that could not result from recursive binary splitting.

Top Right: The output of recursive binary splitting on a two-dimensional example.

Bottom Left: A tree corresponding to the partition in the top right panel.

Bottom Right: A perspective plot of the prediction surface corresponding to that tree.

8.2 Recursive binary splitting

- Reasons: It is computationally infeasible to consider every possible partition of the feature space into J boxes
- Features:
  - top-down:从上到下，每个split都分成两个新regions
  - greedy: 每一步只考虑当前条件下的最佳分割方案，即只考虑局部最优解，而不考虑整体
  - Stepwise Selection也是贪心算法，每次只考虑当前最优解
- Details:
  - 选择预测因子X_j及分割点s，分割为$\{X|X_j < s\} + \{X|X_j \geq s\}$
  - 分割思想：最小化RSS
  - 重复上述过程,但不同的是,仅仅分割 [two previously identified regions]
  - Termination: reaching stop criterion(e.g. no region contains > 5 observations)

8.3 Pruning a tree

- 如果每个观察值都分到一片叶子:决策树过大 $\Rightarrow$ overfitting
- 更少的分割数 $\Rightarrow$ lower variance,better interpretation $\Rightarrow$ underfitting

- 另一种方法: 先得到大树，再通过不断修剪得到合适的子树
- 代价复杂度修剪(Cost complexity pruning)，即薄弱环节修剪(weakest link pruning)

8.3.1 Cost complexity pruning

- 从原始决策树T_0开始生成一个子树序列$T_0, T_1, T_2, \ldots, T_n$,其中$T_{i+1}$是从$T_i$中产生，$T_n$为根节点;
- 目标 : 最小化RSS

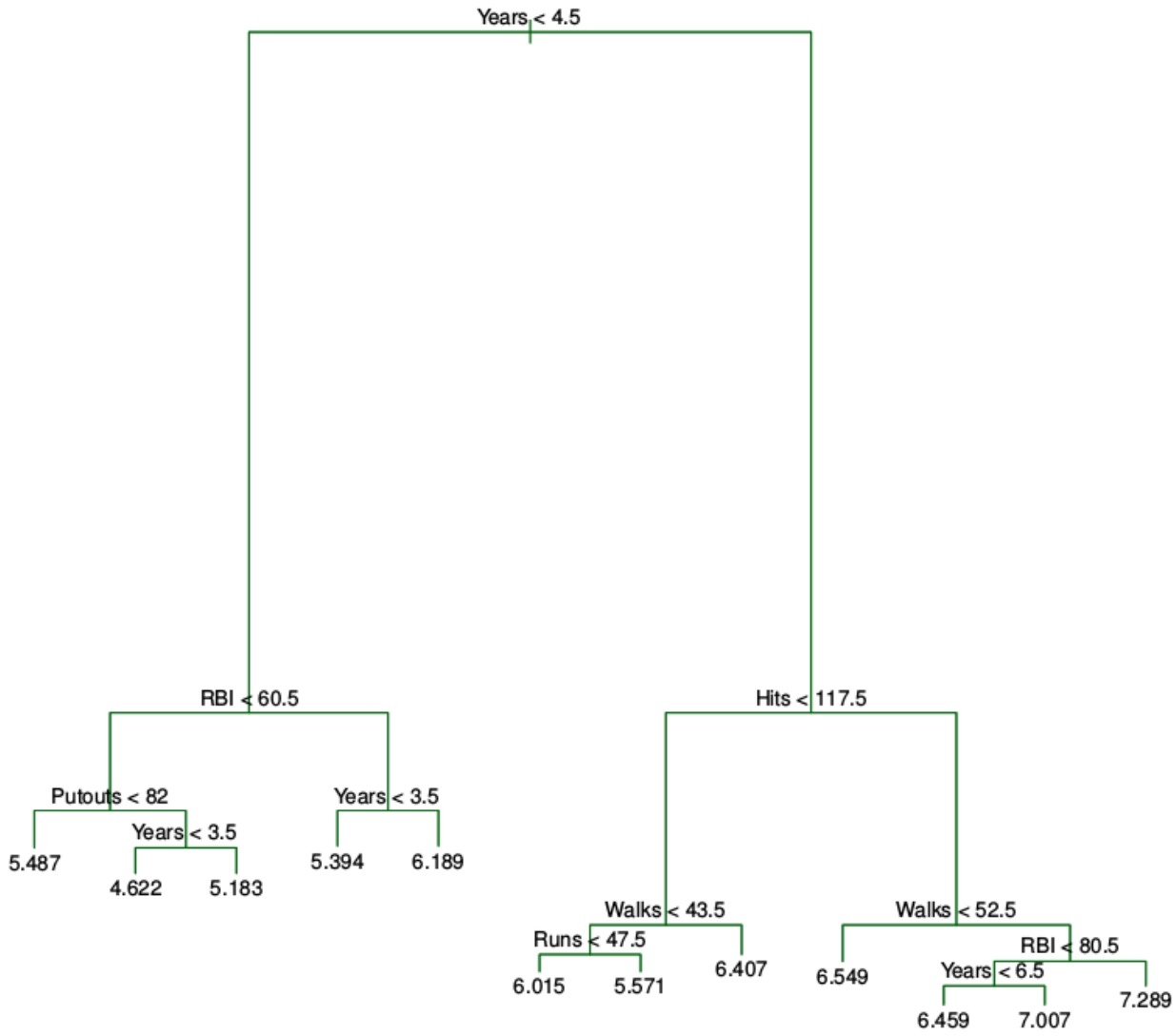
$$RSS = \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - y'_{R_m})^2 + \alpha |T|$$

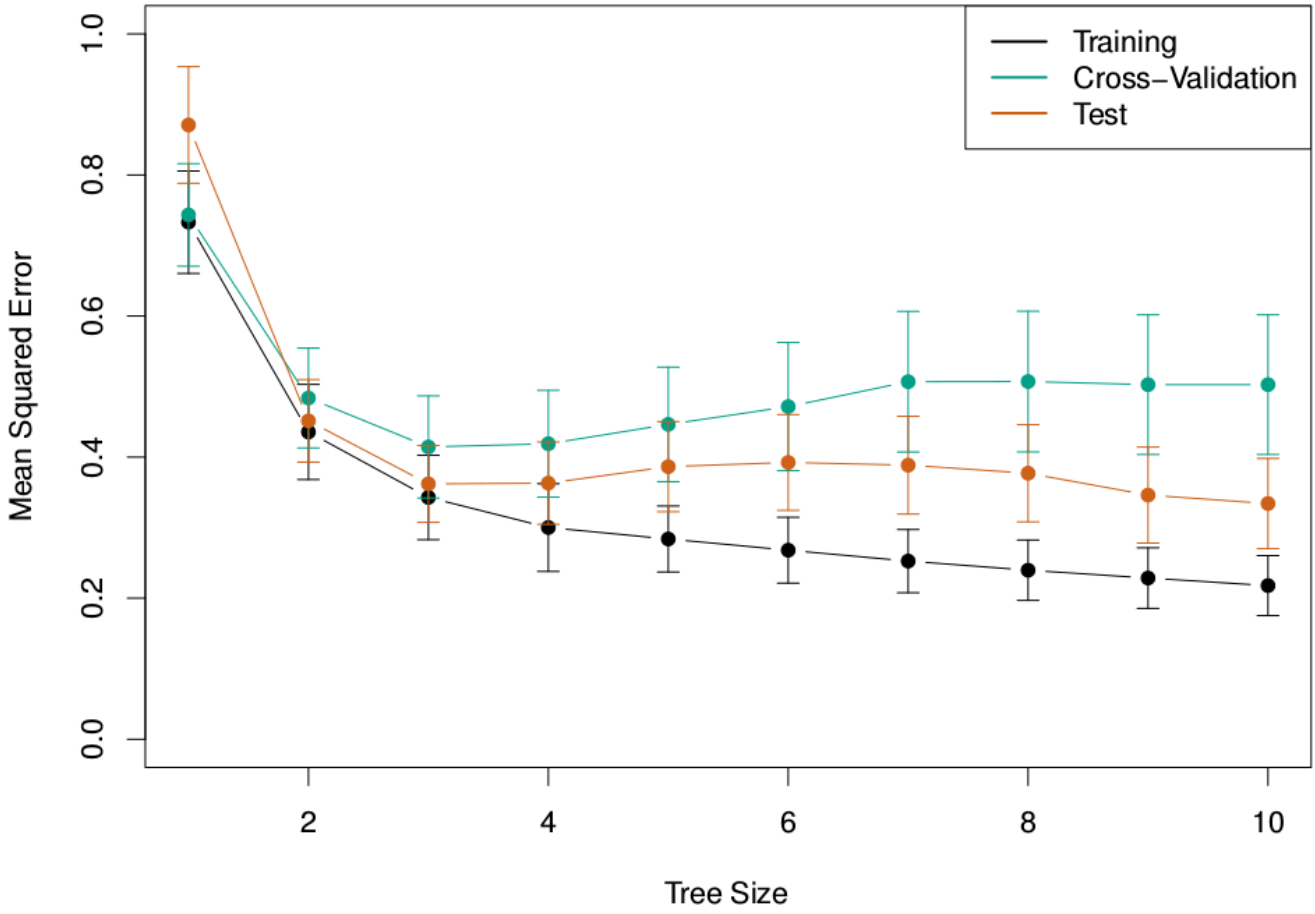
  - α>=0:tuning parameter
  - |T|: the number of terminal nodes
  - R_m:对应mth 叶子结点的预测因子空间，y'_{R_m}为训练结果的均值

8.3.2 Choosing the best subtree

- 调节参数α：控制复杂度与拟合度的trade-off
- 选择最优的α：K-folds CV
- 随着树枝的增加，CV-MSE先减小后增大，test error也是这个趋势，training error则不断减小

8.3.3 栗子


Years < 4.5
RBI < 60.5
Hits < 117.5
Putouts < 82
Years < 3.5
Years < 3.5
Walks < 43.5
Walks < 52.5
5.487
Years < 3.5
5.394
6.189
Runs < 47.5
6.407
6.549
RBI < 80.5
6.015
5.571
Years < 6.5
7.289
6.459
7.007
4.622
5.183



8.4 Classification Trees

- 与逻辑树类似，只不过用于预测定性回复值(分类)而非定量
- 预测值分类法：某区域最常出现的class便代表这一区域
- 同样使用递归二叉树分割来构建，但不以RSS为分类标准
- 分类标准:Classification error rate/Gini index/Cross-entropy

8.4.1 Classification error rate

$$E = 1 - \max_k(p_{i,t})$$

- E:某个区域内不属于指定类别i的比例
- $p_{i,t}$:来自区域t且被分类为类别i的观察值比例
- 注意：分类误差法并不够灵敏,实际用后两个

8.4.2 Gini index

$$G = \sum_{i=1}^{K} p_{i,t}(1 - p_{i,t})$$

- 可量度K个类别的total Var
- 如果所有$p_{i,t}接近0或1时，G很小
- 用途：量度node purity

8.4.3 cross-entropy交叉熵

$$CE = -\sum_{i=1}^{K} p_{i,t} \log p_{i,t}$$

- 结果和Gini Index类似

8.4.4 栗子

$Class0 : 1$
$Class1 : 5$

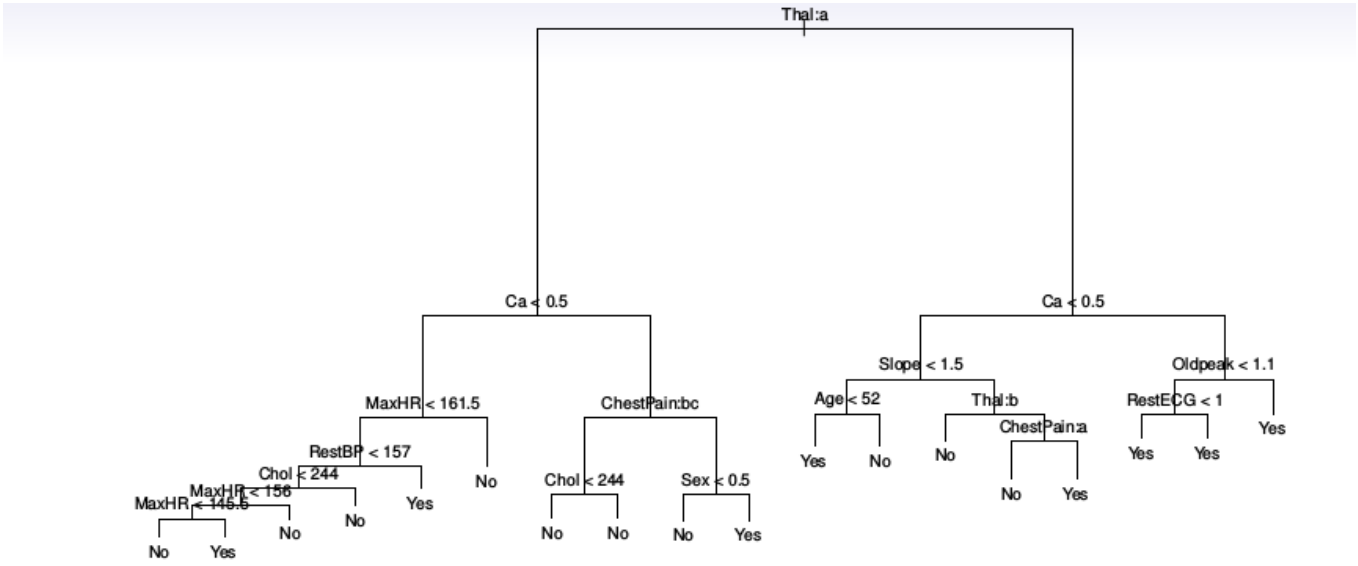
$$E = 1 - \max(\frac{1}{6}, \frac{5}{6})$$

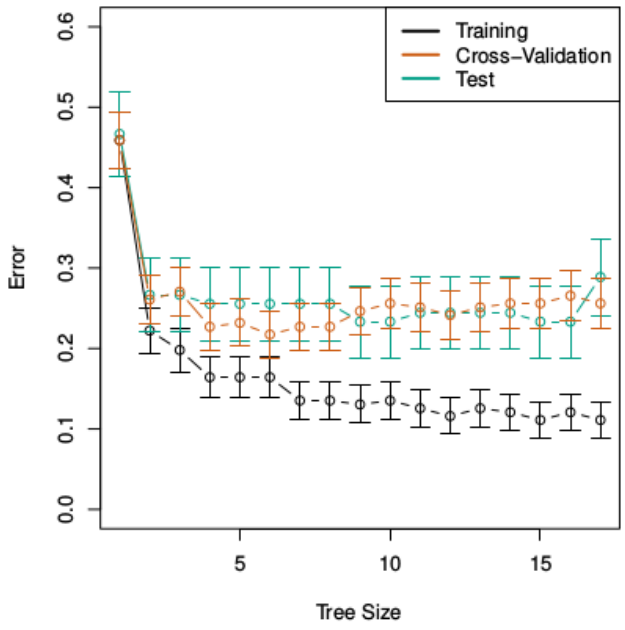
$$G = 1 - \frac{1}{6} * \frac{5}{6} - \frac{5}{6} * \frac{1}{6}$$

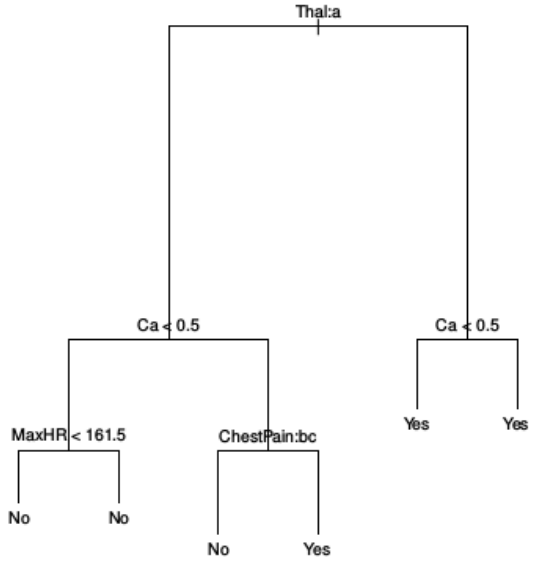
$$CE = -\frac{1}{6} \log \frac{1}{6} + -\frac{5}{6} \log \frac{5}{6}$$

8.4.5 另一个栗子: 心脏病数据

- These data contain a binary outcome `HD` for 303 patients who presented with chest pain.
- An outcome value of `Yes` indicates the presence of heart disease based on an angiographic test, while `No` means no heart disease.
- There are 13 predictors including `Age`, `Sex`, `Chol` (a cholesterol measurement), and other heart and lung function measurements.
- Cross-validation yields a tree with six terminal nodes. See next figure.







8.5 Bagging

8.5.1 Bagging(bootstrap aggregation)

- 复习bootstrap的原理：重复选取n个子集，分别求取Var，然后取平均值

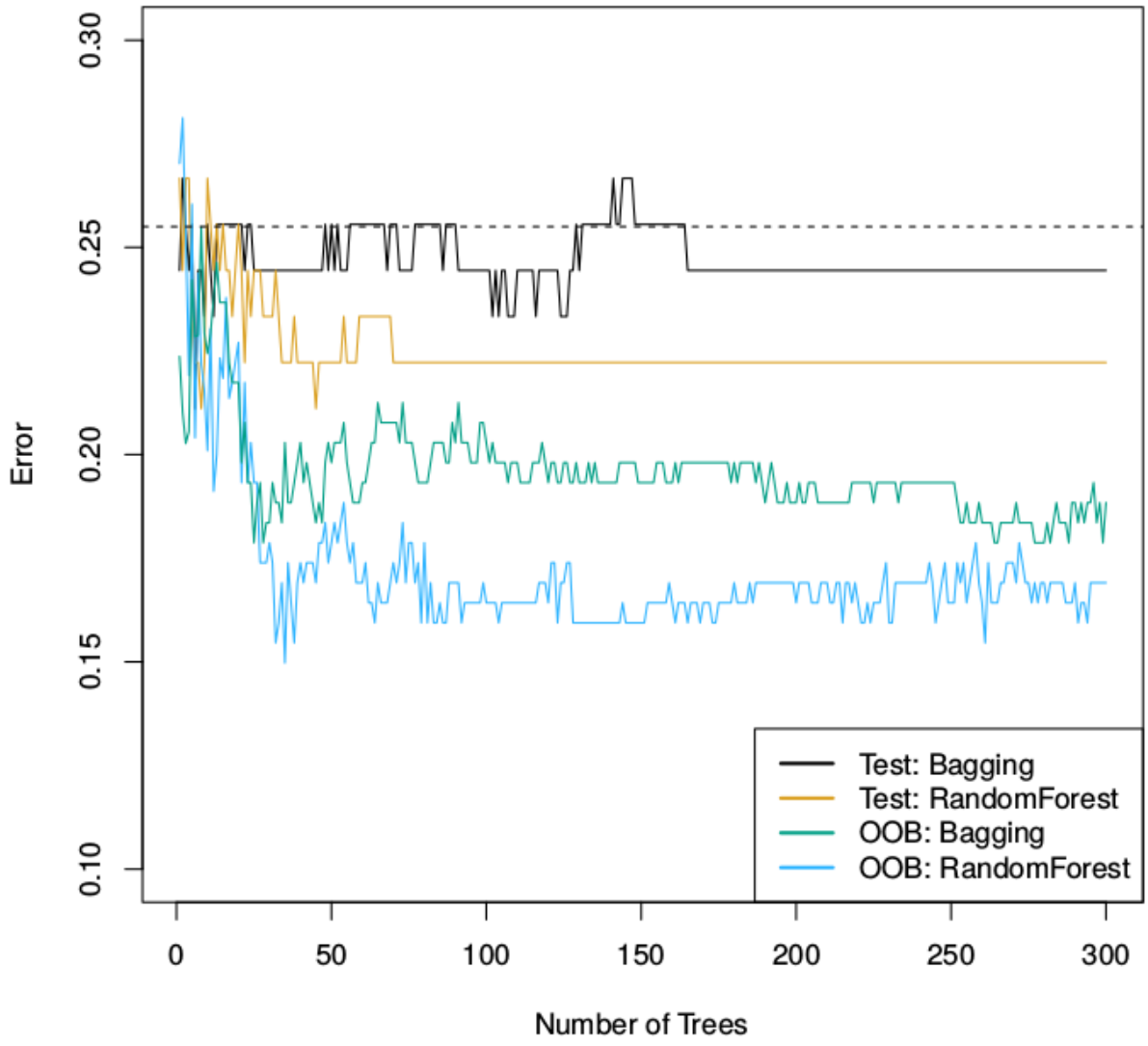
$$f'_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} f'^{*b}(x)$$

- bagging回归树:
  - 生成B个bootstraped训练集;
  - 每个训练集都生成一个独立的决策树;
  - 组合所有决策树生成预测模型;
  - 最后取平均;
- bagging分类树:
  - 对于每个test观察值,记录每个B trees预测的class;
  - 对分类结果进行投票;
  - 总体预测是B个预测值中最可能存在的那个class(票数最多);

8.5.2 Out-of-Bag Error Estimation

- Out-of-Bag(OOB)观察值：每个bagged树平均使用2/3B的观察值，剩下的不用于fit的那部分作为OOB
- 可以通过使用OOB观察值的tree对ith 结果误差进行预测
- ith 观察结果总共有1/3B个，取平均值即可
- 如果B很大的话，其实就是LOOCV 误差评估

8.5.3 栗子



Bagging and random forest results for the Heart data.

- The test error (black and orange) is shown as a function of B, the number of bootstrapped training sets used.
- Random forests were applied with $m = \sqrt{p}$.
- The dashed line indicates the test error resulting from a single classification tree.
- The green and blue traces show the OOB error, which in this case is considerably lower

- 两次随机选取

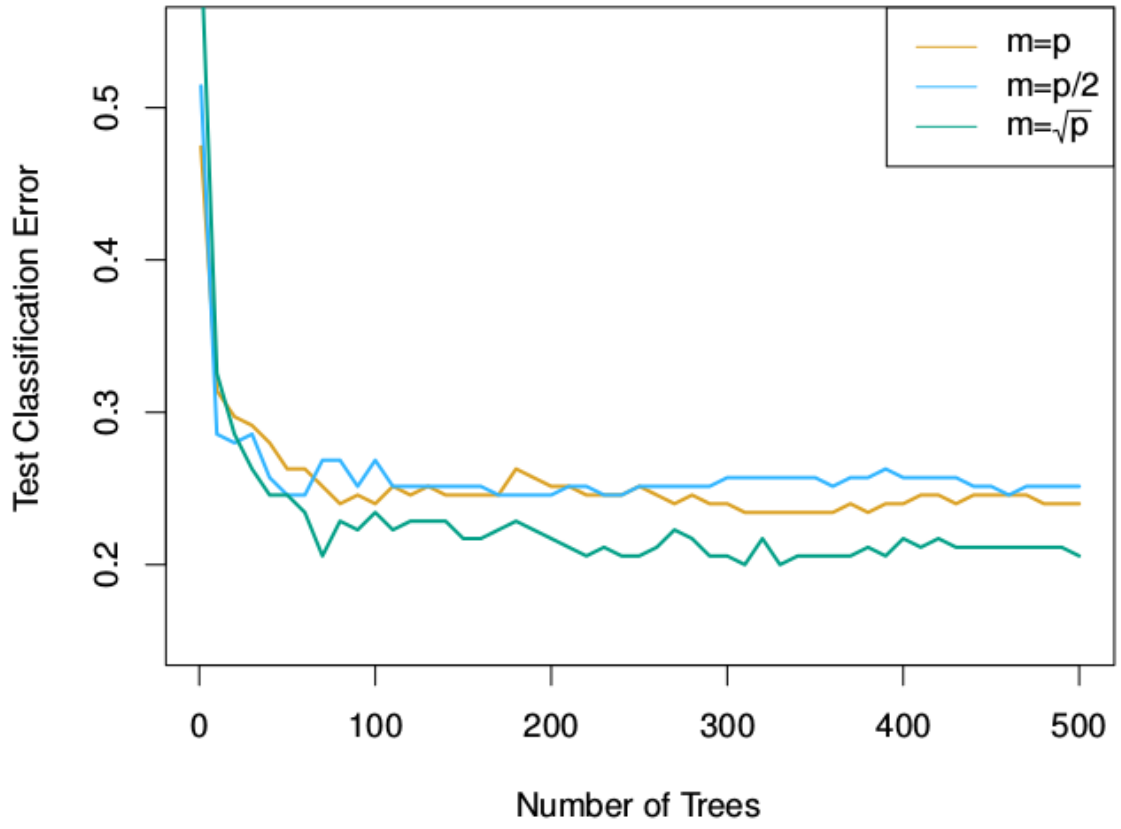
8.6.1 基本原理

- bootstrap取样N次，得到N个训练集，生成N个分类树，组成随机森林;
- 森林中的每个决策树都是没有关联的;
- 每当加入一个新样本时，用每一个决策树进行判断进行分类;
- 进行split的变量并不是全部p个预测因子，而是随机选取m个，一般情况$m \approx \sqrt{p}$
- 对每一个分类树的结果进行vote

8.6.2 采样和完全分割

- 采样：两次随机，可以有效避免overfitting
    - 行采样(样本)：建立树时取样N次，每棵树的输入样本都不是全部的样本
    - 列采样(变量)：split分类时在p个预测因子中取m个
    - 与bagging对比：RF $m < p$,相比bagging($m = p$),错误率降低
- 完全分割：不需要进行剪枝
    - 某个叶节点无法继续分裂/所有样本指向一个分类
    - 每个人的决策力都是很弱的, 但是大家集合起来对某一个样本进行决策，最后投票得到结果

8.6.3 栗子



- Results from random forests for the fifteen-class gene expression data set with $p = 500$ predictors.
- The test error is displayed as a function of the number of trees. Each colored line corresponds to a different value of m, the number of predictors available for splitting at each interior tree node.
- Random forests ($m < p$) lead to a slight improvement over bagging ($m = p$). A single classification tree has an error rate of 45.7%.

8.7 Boosting(GBM)

8.7.1 Boosting机理和Bagging类似

- Bagging：建立bootstrap数据集，并生成彼此独立的决策树，组合决策树形成模型
- 区别：Boosting的trees按顺序生成，每棵树基于前面树的信息

- 每棵树相当于是迭代并改进前一棵树

8.7.2 Boosting for regression trees

- set $f'(x) = 0$ and $r_i = y_i$ for all i in the training set;
- for b=1,2,...B,repeat:
  - fit a tree f'^b with d splits(d+1 terminal nodes)to the training data(X,r);
  - 通过加入新tree的压缩版本，对f'进行update

$$f'(x) = f'(x) + \lambda f'_b(x)$$

  - 以此类推更新残差

$$r_i = r_i - \lambda f'_b(x_i)$$

  - 输出boosted model：

$$f'(x) = \sum_{b=1}^{B} \lambda f'^b(x_i)$$

8.7.3 原理

- 与单独成树再剪枝不同，boosting将样本分为多个分类器(小决策树)
- 每次更新在残差减少的梯度上建立新的决策树模型，通过不断迭代最小化残差
- 每个小决策树的终端节点数取决于算法里的参数d(depth，即split数目)

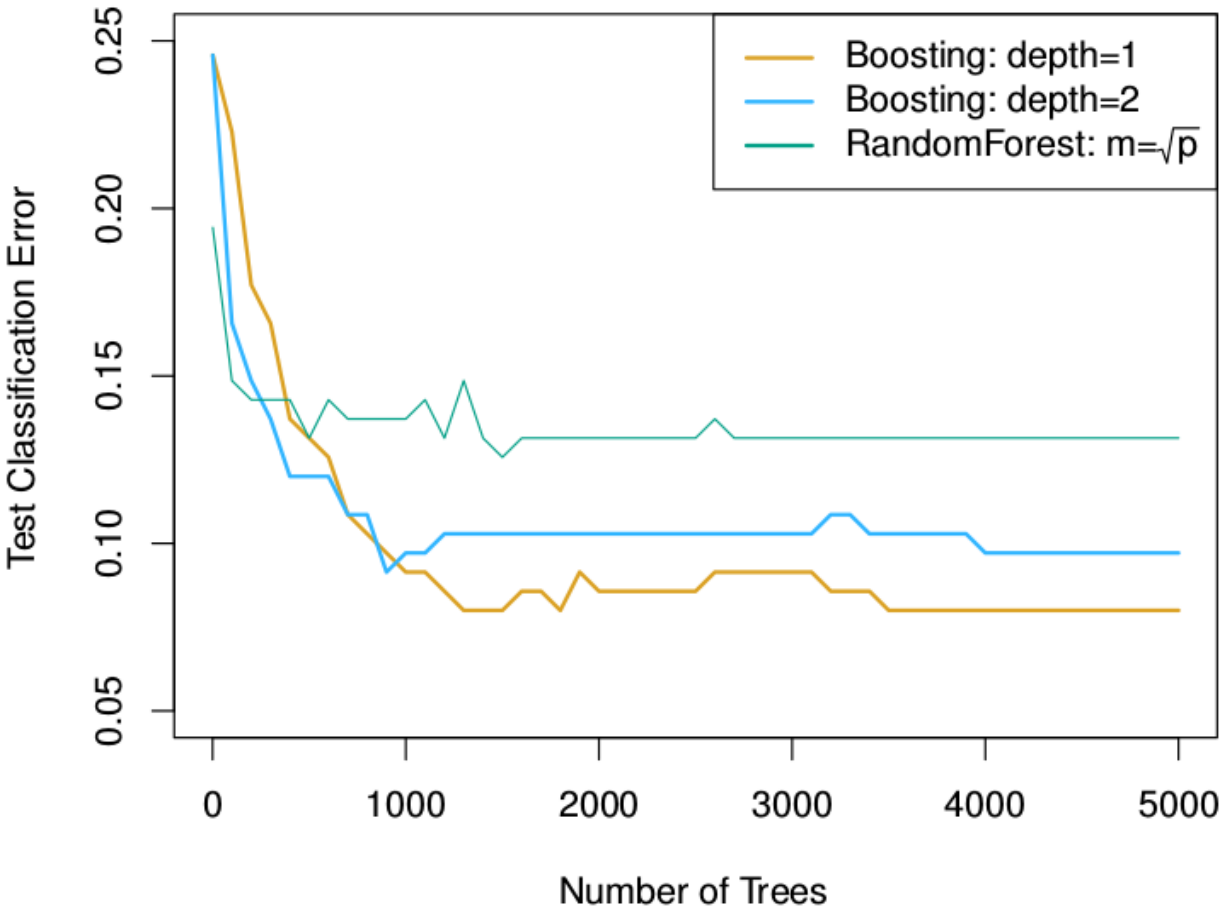
8.7.4 Tuning parameters for boosting

- number of trees B:不同于bagging/RF，如果B过大，会overfitting
  - 通过CV来对B进行选择
- shrinkage parameter λ：控制boosting学习的速度
  - λ一般为0.01/0.001，取决于实际情况
  - λ很小时更新巨慢，为了提高性能，需要让B很大
- number of splits d：控制boosted整体的复杂度
  - d=1时works well,每棵树都是"stump"树桩，每次更新增加一个split
  - 一般情况下d也被称作interaction depth，等于变量数
  - 随着d增加，test分类误差降低，同样变量的GBM要比RF误差更低

8.7.6 变量权重的量度

- 对于bagged/RF回归树，记录RSS下降的过程，下降幅度越大该变量权重越大
- 同理，对于分类树是Gini Index下降的过程

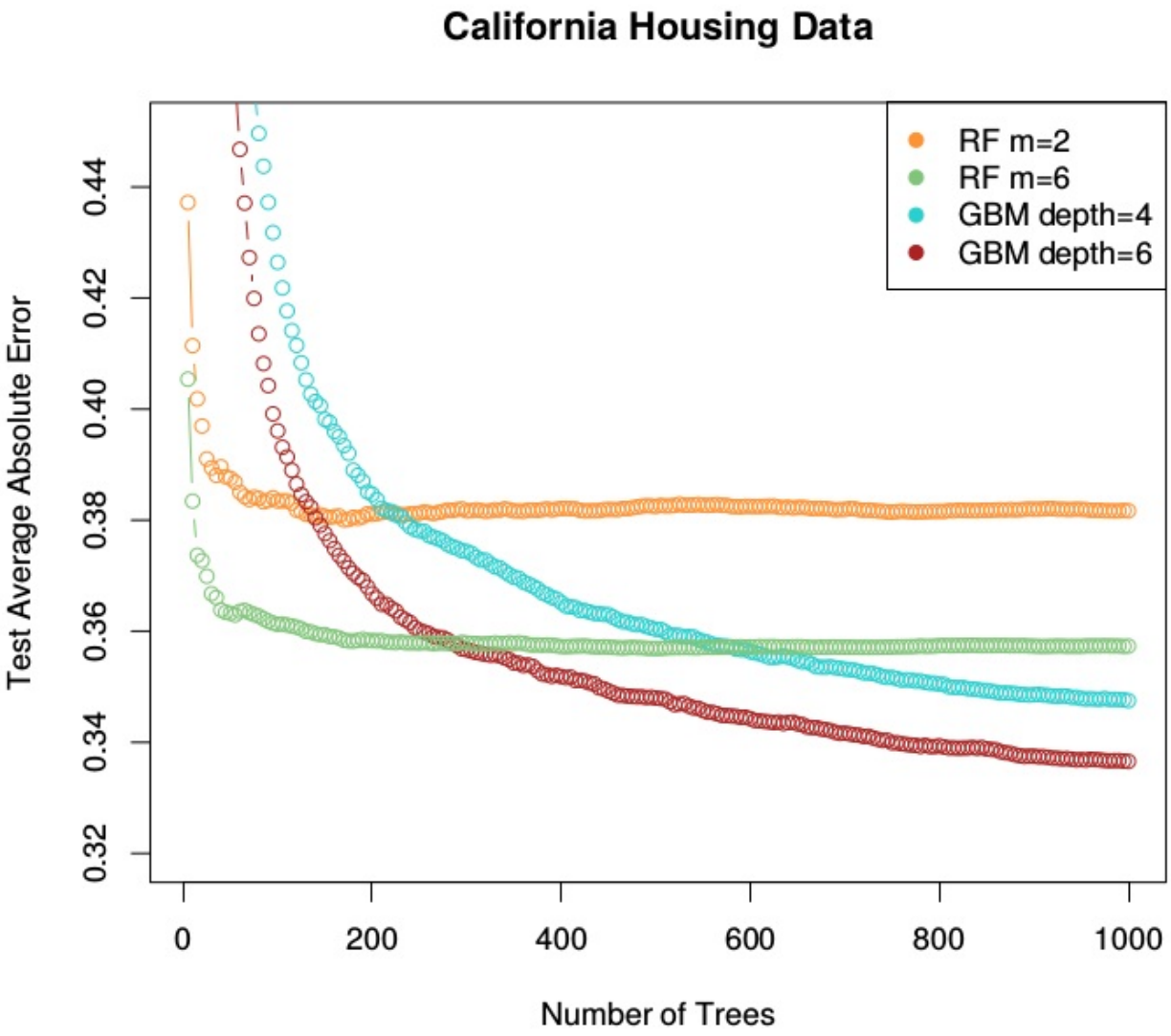
8.7.7 栗子



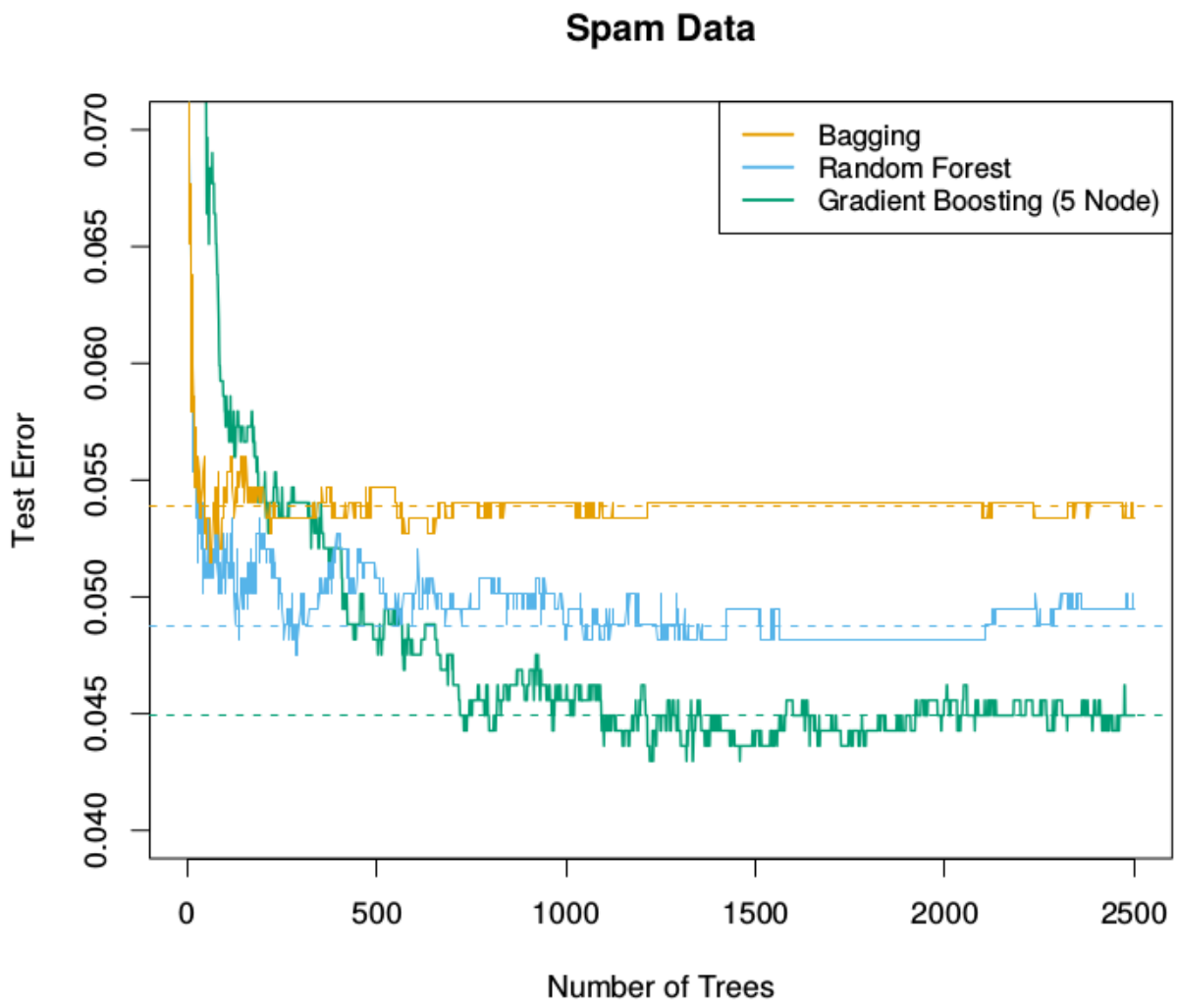
- Results from performing boosting and random forests on the fifteen-class gene expression data set in order to predict *cancer* versus *normal*.
- The test error is displayed as a function of the number of trees. For the two boosted models, $\lambda = 0.01$. Depth-1 trees slightly outperform depth-2 trees, and both outperform the random forest, although the standard errors are around 0.02, making none of these differences significant.
- The test error rate for a single tree is 24%.

8.7.8 再举个栗子: Boosting和随机森林的对比

- Regression



- Classification



8.8 Tree-Based Methods in R

8.8.1 举个栗子：

- (250,150)样本，前者用于训练集建树，并用测试集来评估表现

```
library(tree)
tree.carseats=tree(High~.-Sales,data=Carseats)

set.seed(1011)
train=sample(1:nrow(Carseats),250)
tree.carseats=tree(High~.-Sales,Carseats,subset=train)
plot(tree.carseats)
text(tree.carseats,pretty=0)

tree.pred=predict(tree.carseats,
        Carseats[-train,], #data, remove了训练集
        type="class") #这里的结果会显示完整的树形图
```

```
13.
14.  with(Carseats[-train,],table(tree.pred,High))  #with交叉图，给出True Pos
     itive等的数量
15.  #准确度(NN+YY)/TOTAL=0.7
```

- 为防止深度过大，使用CV进行prune

```
1.  cv.carseats=cv.tree(tree.carseats,FUN=prune.misclass)
2.  plot(cv.carseats) #随着变量变化，deviance也在变化
3.  prune.carseats=prune.misclass(tree.carseats,best=13)
4.  #13个变量是最优选择
5.  plot(prune.carseats)
6.  text(prune.carseats,pretty=0)
7.
8.  #使用修剪过的树来评估test data
9.  tree.pred=predict(prune.carseats,Carseats[-train,],type="class")
10. with(Carseats[-train,],table(tree.pred,High))
11.
12. #跟前面比准确度差别不大，而树更简洁
```

8.6.2 Random Forests

- 建立一堆小树取均值，从而减少Variance

```
1.  require(randomForest)
2.  require(MASS)
3.  set.seed(101)
4.  dim(Boston) #506,14
5.  train=sample(1:nrow(Boston),300)
6.
7.  rf.boston=randomForest(medv~., #房价中位数
8.          data=Boston,
9.          subset=train)
```

- 结果分析：
  - MSR与%Var都是基于OOB的;
  - mtry=4(每个split随机选择的变量数);
  - p=13-->总共有C(13,4)个mtries
- 完整算法+可视化：

```
1.  oob.err=double(13) #double意指双精度向量，这里等于13个0...
2.  test.err=double(13)
```

```
3.
4.  for(mtry in 1:13){
5.
    fit=randomForest(medv~.,data=Boston,subset=train,mtry=mtry,ntree=400);
    #随机森林拟合
6.      oob.err[mtry]=fit$mse[400];
7.      pred=predict(fit,Boston[-train,]);
8.      test.err[mtry]=with(Boston[-train,],mean((medv-pred)^2));
9.      cat(mtry," ");
10. }
11.
12. matplot(1:mtry,cbind(test.err,oob.err),pch=19,
13. col=c("red","blue"),type="b",ylab="Mean Squared Error") #作图
14. legend("topright",legend=c("OOB","Test"),pch=19,
15. col=c("red","blue")) #标注
```

- 结果分析：test-error曲线略低于OOB曲线，因为需要考虑数据本身的标准误，还是比较合理的

8.6.3 Boosting

- 后一棵树对前一棵树的缺陷不断修正，逐步完善

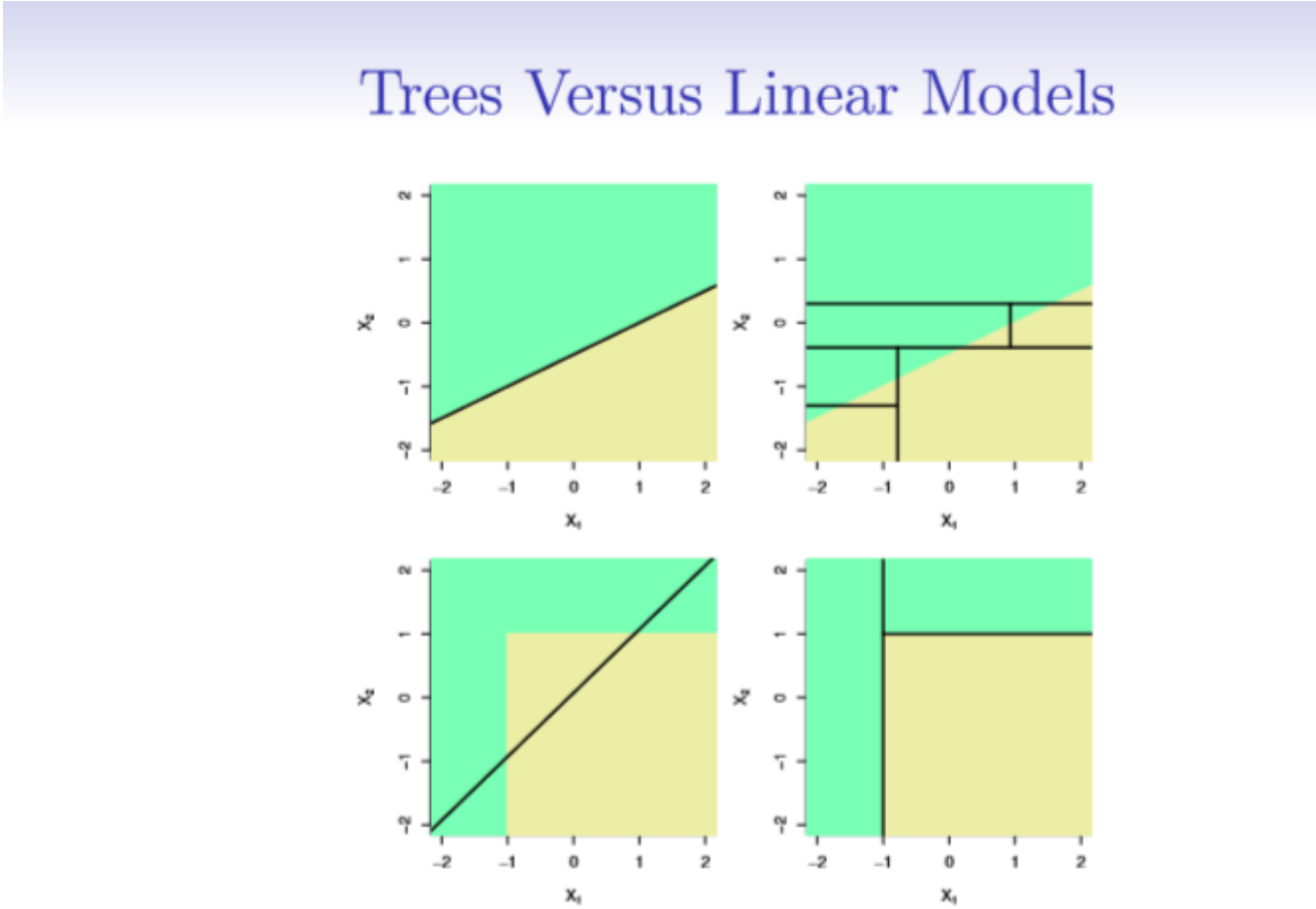
```
1.  require(gbm)
2.  boost.boston=gbm(medv~.,data=Boston[train,],
3.          distribution="gaussian",
4.          n.trees=10000,shrinkage=0.01,
5.          interaction.depth=4)#深度(变量数)
6.
7.  summary(boost.boston)
8.  plot(boost.boston,i="lstat")
9.  plot(boost.boston,i="rm")
```

- 树的数量为调整参数，过大会overfit，使用CV来确定合适的值

```
1.  n.trees=seq(from=100,to=10000,by=100)
2.
3.  predmat=predict(boost.boston,newdata=Boston[-train,],n.trees=n.trees)
4.
5.  dim(predmat)
6.  berr=with(Boston[-train,],
7.  apply((predmat-medv)^2,2,mean))
8.
9.  plot(n.trees,berr,pch=19,ylab="Mean Squared Error", xlab="#
```

```
    Trees",main="Boosting Test Error")
10.
11.  abline(h=min(test.err),col="red")
```

8.7 Summary and Quiz



Top Row: True linear boundary; Bottom row: true non-linear boundary.
Left column: linear model; Right column: tree-based model

8.7.1 summary in tree algorithm(regression tree)

- 使用 recursive binary splitting建立大树，只有当观察值过少时才停止split
- 使用cost complexity pruning建立关于α的函数以得到最佳subtrees
- 使用K-fold CV对α进行选择

8.7.2 Classification Trees

- 对定性结果进行分类，某区域最常出现的class便代表这一区域

- Gini Index VS Cross Entropy

8.7.3 Advantages and Disadvantages of Trees

- 相比于其他显性/非线性手段可得到的类别更多;
- 可解释性强,易于可视化 $\Rightarrow$ tree is the model
- 比regression/classification更类似人类决策;
- 处理定性数据更容易, 不需要建立冗杂的变量体系;
- 预测精度不足

8.7.4 三种算法有助于提升精密度：bagging/RF/boosting

- 基于训练数据生成多个trees，并组合所有树的预测结果
- 牺牲可读性
- 误差：GBM < RF < Bagging

8.7.5 Quiz

- 贪心算法
- What's the one-standard-error-rule in CV?
- Tree可生成的class要多于GAM

Chapter 9 Support Vector Machines

- 上这个课的时候对SVM还是一脸懵逼的状态, 因此没怎么总结, 可以参考我机器学习的笔记

9.1 SVM in R

9.5.1 Linear SVM classifier

```
set.seed(10111)
x=matrix(rnorm(40),20,2)
y=rep(c(-1,1),c(10,10))
x[y==1,]=x[y==1,]+1 #创建二维样本
plot(x,col=y+3,pch=19)

#引入e1071 package中的svm()
library(e1071)
dat=data.frame(x,y=as.factor(y)) #权重分类，1是一类-1是一类

svmfit=svm(y~.,data=dat,
```

```
          kernel="linear", #polynomial/radial basis/tanh
          cost=10, #tuning parameter
          scale=FALSE
          ) #type默认为c-classification
print(svmfit)
plot(svmfit,dat)

#结果γ=0.5, support vectors=6
```

9.5.2 Nonlinear SVM

```
plot(x,col=y+1)
dat=data.frame(y=factor(y),x)

fit=svm(factor(y)~.,
    data=dat,scale=FALSE,
    kernel="radial",cost=5)
```

Chapter 10 Unsupervised Learning

10.1 Principal Components

10.1.1 Unsupervised VS Supervised

- Supervised: regression and classification
- 监督：观察变量X以及得到的Y，用以建立模型对未知的Y进行预测
- 非监督：对预测结果不感兴趣，不考虑结果变量Y

10.1.2 Features of unsupervised learning

- Goal:
  - Visualize the data;
  - Discover subgroups among the variables/observations.
- Methods: PCA/Clustering
- Challenges: UL比SL更主观，分析的目标不仅仅是预测结果
- Advantages：无标签数据更易获得

10.1.3 PCA

- PCA produces a low-dimensional representation of a dataset

- 寻找一系列可以最大化var的变量的线性组合
- 这些组合是互相线性无关的(正交)
- 除了推导出向量用以解决SL问题外，PCA还能用于数据可视化

10.1.4 PCA details

- 第一个主成分：归一化变量的线性组合

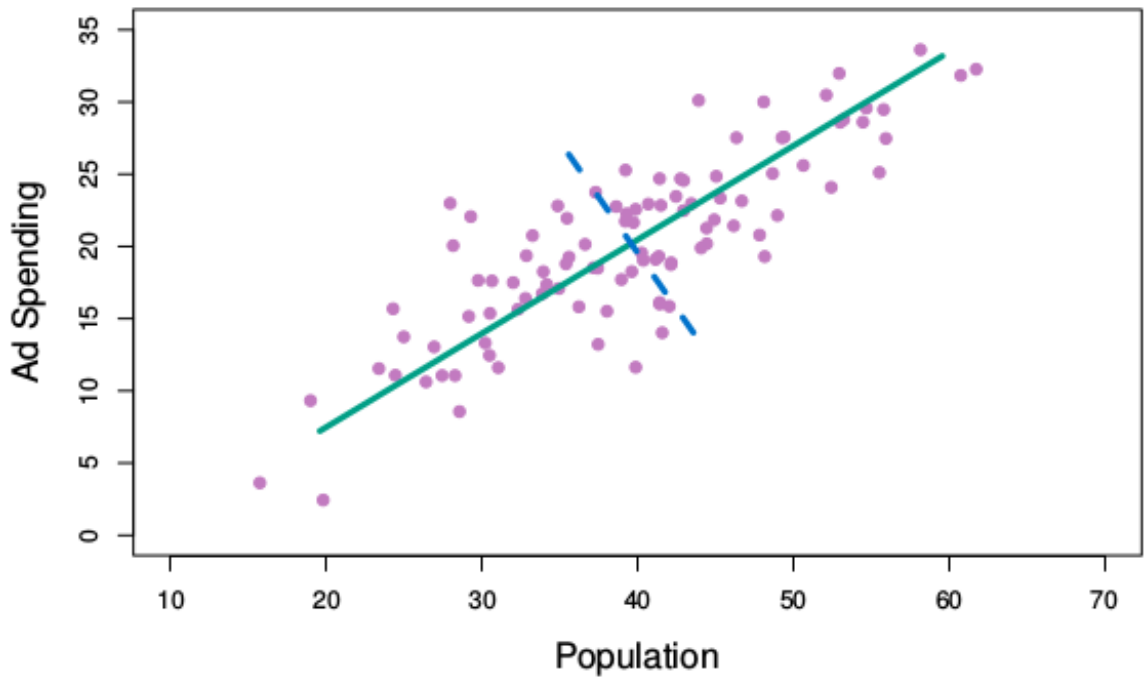
$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p$$

  - 给予变量不同的权重 $\Rightarrow$ 最大化variance
- Normalized的含义：

$$\sum_{j=1}^{p} \phi_{j1}^2 = 1$$

  - 如果不归一化的话，各元素绝对值过大会导致High Variance
- 最终得到第一个主成分向量:

$$\phi_1 = (\phi_{11}, \phi_{21}, \ldots, \phi_{p1})^T$$



10.1.5 Computation of PCs

- 假定一个$n \times p$数据集 X，只考虑方差，将X中每个变量的均值(列平均值)归一化为0;
- 在满足归一化条件的前提下,寻找最大化样本方差的特征值线性组合

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \ldots + \phi_{p1} x_{ip},\ \mathrm{i} = 1,2,\ldots,\mathrm{n}$$

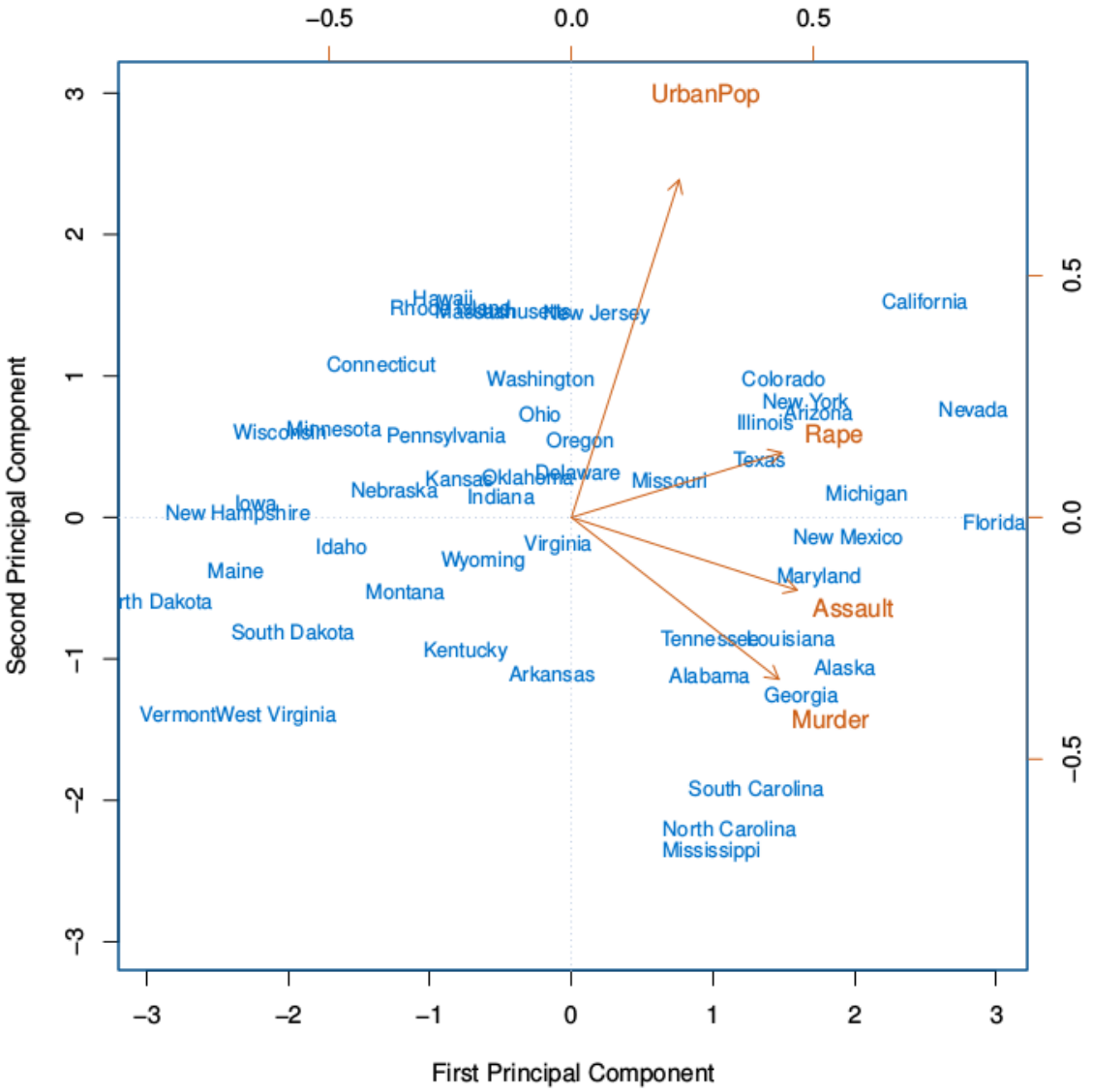
- 由于所有的xij均值为0 $\Rightarrow$ $Avg(z_{i1}) = 0$ $\Rightarrow$ $Var(z_{i1}) = \frac{1}{n}\sum_{i=1}^{n} z_{i1}^2$
- 得到第一主成分后, 可用其解决优化问题:

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{j1}x_{ij}\right)^2 \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1.$$

  - This problem can be solved via a singular-value decomposition of the matrix $\mathbf{X}$, a standard technique in linear algebra.
  - We refer to Z_1 as the first principal component, with realized values $z_{11}, \ldots, z_{n1}$

10.1.6 Geometry of PCA

- 主成分向量指向数据变化最大的方向
- 栗子: 美国犯罪分析
  - `USAarrests` data: For each of the fifty states in the United States, the data set contains the number of arrests per 100,000 residents for each of three crimes: `Assault`, `Murder`, and `Rape`. We also record `UrbanPop` (the percent of the population in each state living in urban areas).
  - The principal component score vectors have length $n = 50$, and the principal component loading vectors have length $p = 4$.
  - PCA was performed after standardizing each variable to have mean zero and standard deviation one.


UrbanPop
Rape
Assault
Murder
California
Nevada
Florida
Connecticut
Washington
Colorado
New York
Arizona
Illinois
Ohio
Oregon
Wisconsin
Minnesota
Pennsylvania
Texas
Delaware
Missouri
Kansas
Oklahoma
Nebraska
Indiana
Michigan
New Hampshire
Iowa
Idaho
Virginia
New Mexico
Wyoming
Maine
Maryland
Montana
South Dakota
Kentucky
Tennessee
Louisiana
Arkansas
Alabama
Alaska
Georgia
Vermont
West Virginia
South Carolina
North Carolina
Mississippi
First Principal Component
Second Principal Component

The first two principal components for the USArrests data.

- The blue state names represent the scores for the first two principal components.
- The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for `Rape` on the first component is 0.54, and its loading on the second principal component 0.17 [the word `Rape` is centered at the point $(0.54, 0.17)$].
- This figure is known as a *biplot*, because it displays both the principal component scores and the principal component loadings.

	PC1	PC2
Murder	0.5358995	-0.4181809
Assault	0.5831836	-0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

10.2 Higher Order Principal Components

10.2.1 Further principal components

- 第二主成分的做法和刚刚类似:

$$z_{i2} = \phi_{12}x_{i1} + \phi_{22}x_{i2} + \ldots + \phi_{p2}x_{ip},\ \mathrm{i} = 1,2,\ldots,\mathrm{n}$$

- 进而得到:

$$\phi_2 = (\phi_{12}, \phi_{22}, \ldots, \phi_{p2})^T$$

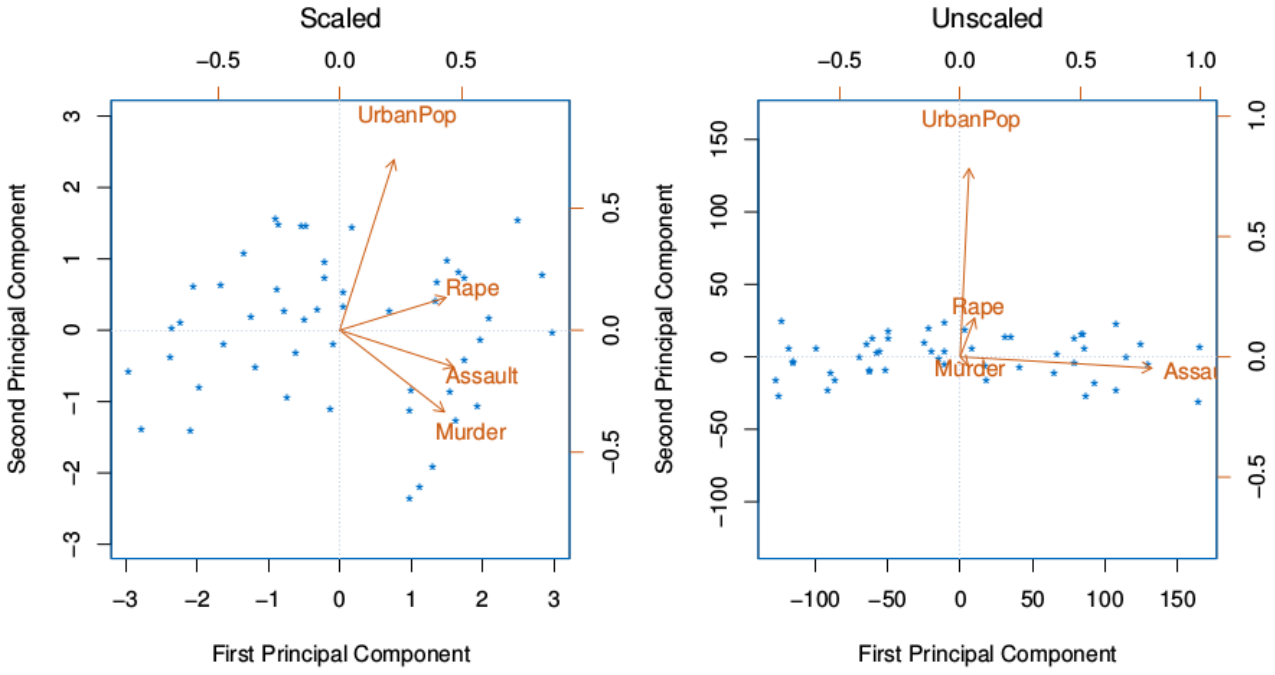
- 注意需要限制Z2与Z1不相关： 通过调节权重，让ϕ_1与ϕ_2正交

10.2.2 PCA find the hyperplane closest to the observations

- The first principal component loading vector has a very special property: it defines the line in p-dimensional space that is *closest* to the n observations (using average squared Euclidean distance as a measure of closeness)
- The notion of principal components as the dimensions that are closest to the n observations extends beyond just the first principal component.
- For instance, the first two principal components of a data set span the plane that is closest to the n observations, in terms of average squared Euclidean distance.

10.2.3 Scaling of the variables matters

- 若变量单位不同: 进行归一化使得标准差相等
- 相同单位的变量不需要归一化



10.2.4 Proportion Variance Explained

- The *total variance* present in a data set (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2,$$

  and the variance explained by the mth principal component is

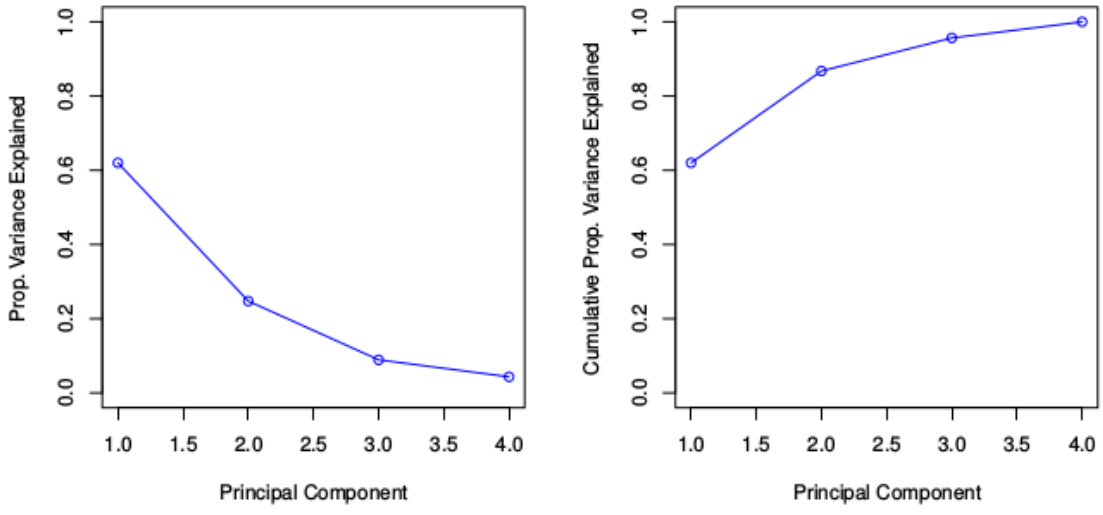
$$\mathrm{Var}(Z_m) = \frac{1}{n} \sum_{i=1}^{n} z_{im}^2.$$

- It can be shown that $\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{m=1}^{M} \mathrm{Var}(Z_m)$, with $M = \min(n-1, p)$.

- Therefore, the PVE of the mth principal component is given by the positive quantity between 0 and 1

$$\frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}.$$

- The PVEs sum to one. We sometimes display the cumulative PVEs.



10.2.5 How many PC should we use

- Only using CV is not sufficient： no response, 而CV需要对response进行predict
- 可以使用CV的情况： 决定要在回归中使用多少变量时，可以使用CV对变量数进行优化

10.2.6 LDA VS PCA

- LDA的点投影到linear上，而PCA的点投影到超平面上;
- LDA的输入数据是带标签的，而PCA的输入数据是不带标签的;
- LDA作为独立的算法存在，PCA更多的用于数据的预处理的工作, 不用于拟合模型

10.3 k-means Clustering

10.3.1 Basic knowledge

- 物以类聚，人以群分
- Clustering: deviding data into different groups by their classes
- 与监督学习中的分类不同的是：聚类要划分的类是未知的
- PCA versus Clustering:
  - PCA:PCA looks for a low-dimensional representation of the observations that explains a good fraction of the variance.
  - Clustering looks for homogeneous subgroups among the observations.
- 2 clustering methods:
  - K-means clustering:预先指定clusters的数目K，再将观察值分配进去
  - Hierarchical clustering：
    - 提前不知clusters数量;
    - 结果为树状图形式;
    - 可以看到各种clusters数量[1:n]的聚类

10.3.2 K-means clustering

- K is the number of clusters;
- 每个观察值至少属于一个cluster;
- cluster互不重叠：每个观察值最多属于一个cluster;
- 分类指标：minimize within-cluster variation, WCV(Ck);
- 最终目标: 最小化所有聚类WCV的和

10.3.3 Details of K-means clustering

- 使用欧氏距离来定义WCV：

$$\mathrm{WCV}(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2, \qquad (3)$$

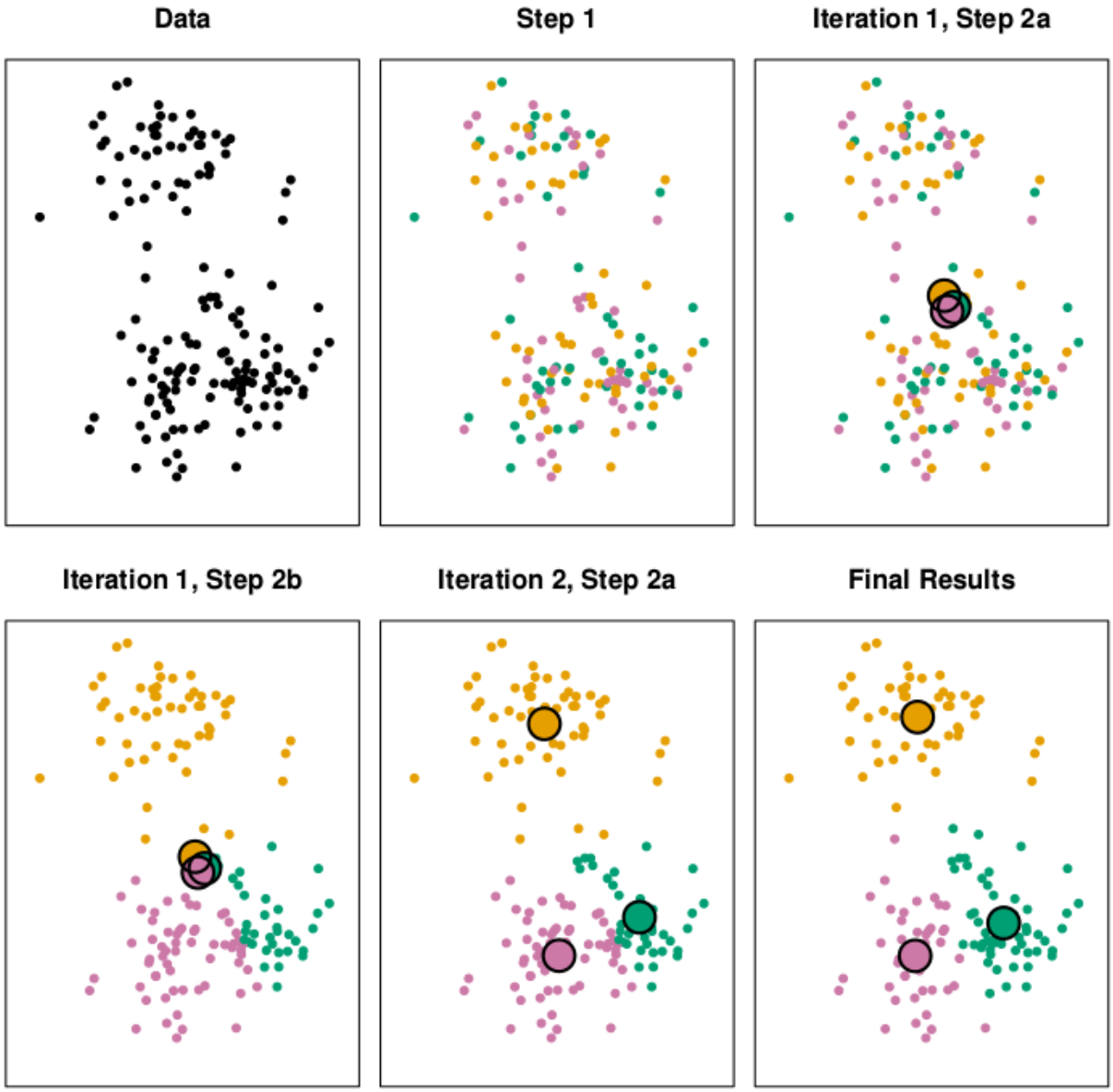
where $|C_k|$ denotes the number of observations in the kth cluster.

  - 其实就是用Ck内每个观察值x减去质心的平方和在除以观察值数量
- 通过推导，得到优化问题：

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}. \qquad (4)$$

10.3.4 Algorithm of K-means Clustering

- 从N个样本随机选取K个样本作为初始质心;
- 测量剩余样本到质心的距离，归类到最近的质心;
- 重新计算已经得到的聚类的质心;
- 不断迭代直到质心无法再更新或更新范围很小



10.3.5 Properties of the Algorithm

- 该算法保证每一步都能最小化WCV和，进行优化
- 该算法无法保证全局最小：只能保证极值，有多个峰谷时可能得不到最小值
- 不同初始质心得到不同聚类结果

10.3.6 另一种K均值算法: 二分K-Means

- 将所有的初始数据分为2类;
- 选取其中一个类(e.g. SSE较大的)进行二分,另一个类不变;
- 不断二分直到得到K个聚类

10.3.7 K-means的优缺点

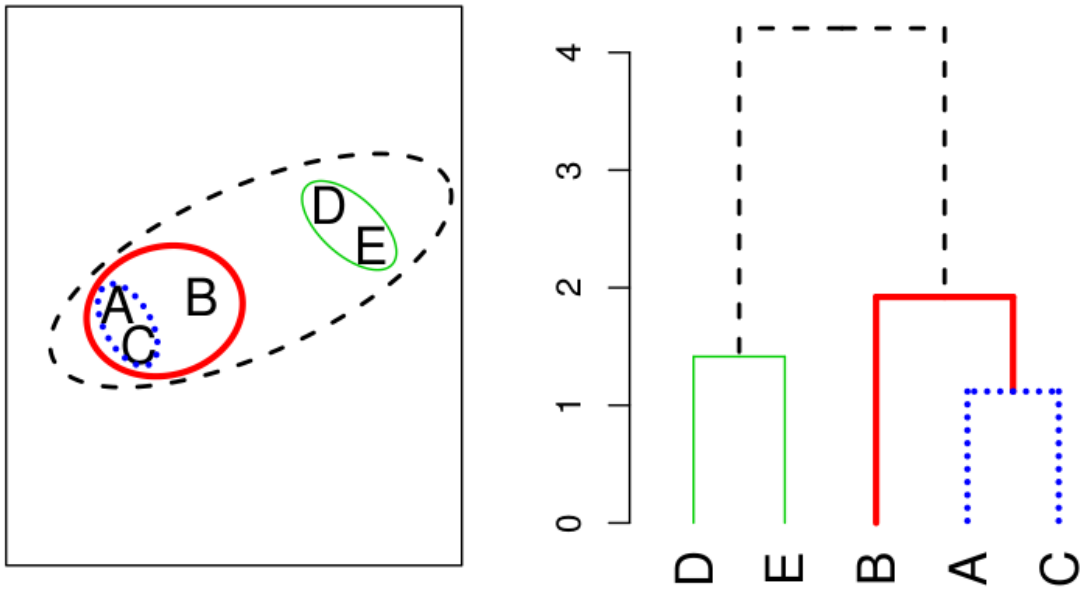
- Pros: 简单有效
- Cons:
  - 不能处理非球形/不同尺寸/密度的聚类;
  - 需要提前指定聚类数量K;
  - 可能无法得到全局最优

10.4 Hierarchical Clustering

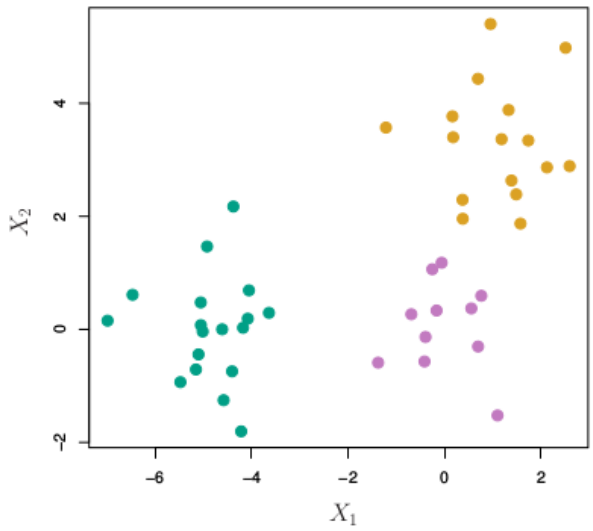
- 不需要提前得到cluster的数量K
- 两种层次聚类：bottom-up & agglomerative
- 层次聚类有点类似决策树的逆过程，先生成复杂树，不断合并距离最近的枝叶
- 需要确定linkage以及dissimilarity measure

10.4.1 Hierarchical Clustering Algorithm

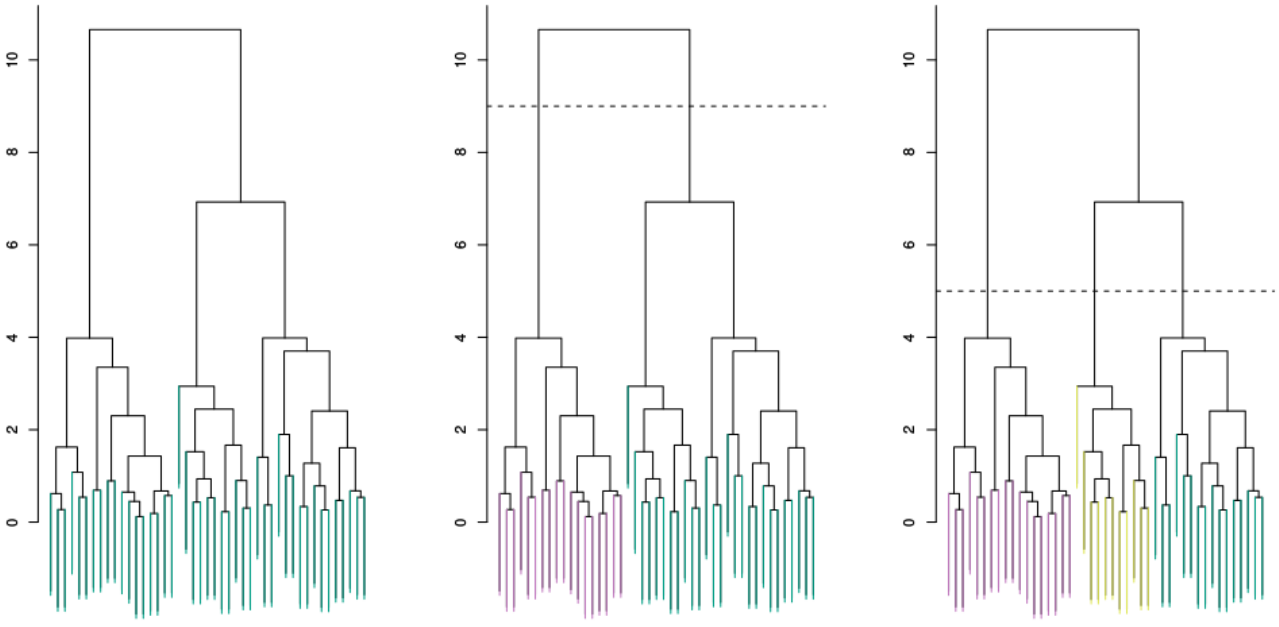
- Start with each point in its own cluster.
- Identify the closest two clusters and merge them.
- Repeat.
- Ends when all points are in a single cluster.



- 栗子:

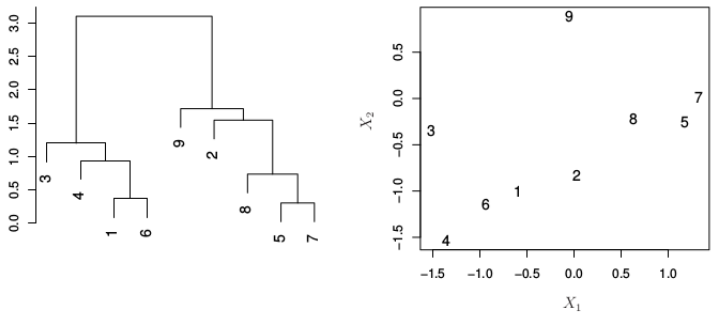


45 observations generated in 2-dimensional space. In reality there are three distinct classes, shown in separate colors. However, we will treat these class labels as unknown and will seek to cluster the observations in order to discover the classes from the data.

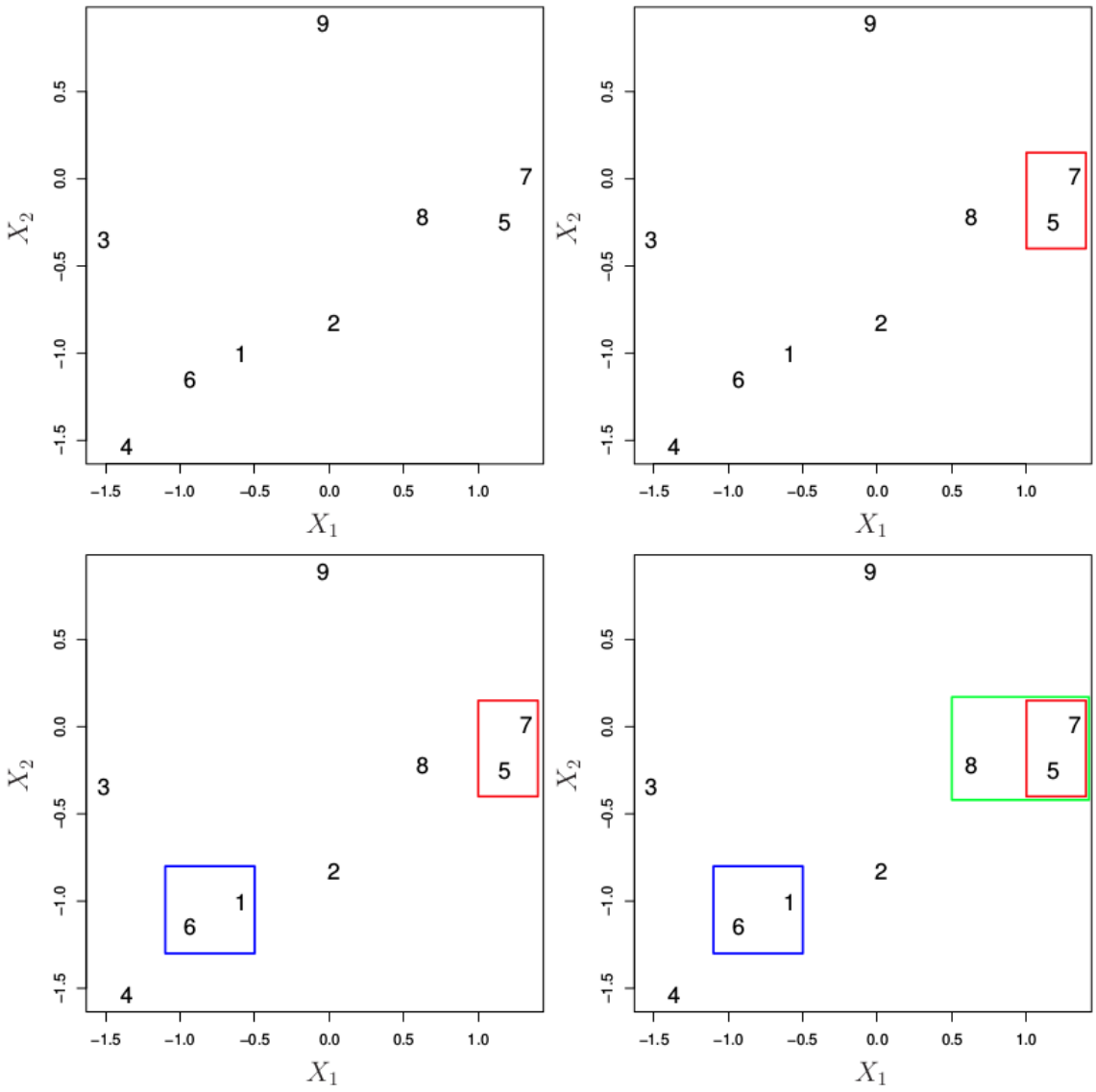


- *Left:* Dendrogram obtained from hierarchically clustering the data from previous slide, with complete linkage and Euclidean distance.
- *Center:* The dendrogram from the left-hand panel, cut at a height of 9 (indicated by the dashed line). This cut results in two distinct clusters, shown in different colors.
- *Right:* The dendrogram from the left-hand panel, now cut at a height of 5. This cut results in three distinct clusters, shown in different colors. Note that the colors were not used in clustering, but are simply used for display purposes in this figure

- 再来个栗子:



- An illustration of how to properly interpret a dendrogram with nine observations in two-dimensional space. The raw data on the right was used to generate the dendrogram on the left.
- Observations 5 and 7 are quite similar to each other, as are observations 1 and 6.
- However, observation 9 is *no more similar to* observation 2 than it is to observations 8, 5, and 7, even though observations 9 and 2 are close together in terms of horizontal distance.
- This is because observations 2, 8, 5, and 7 all fuse with observation 9 at the same height, approximately 1.8.



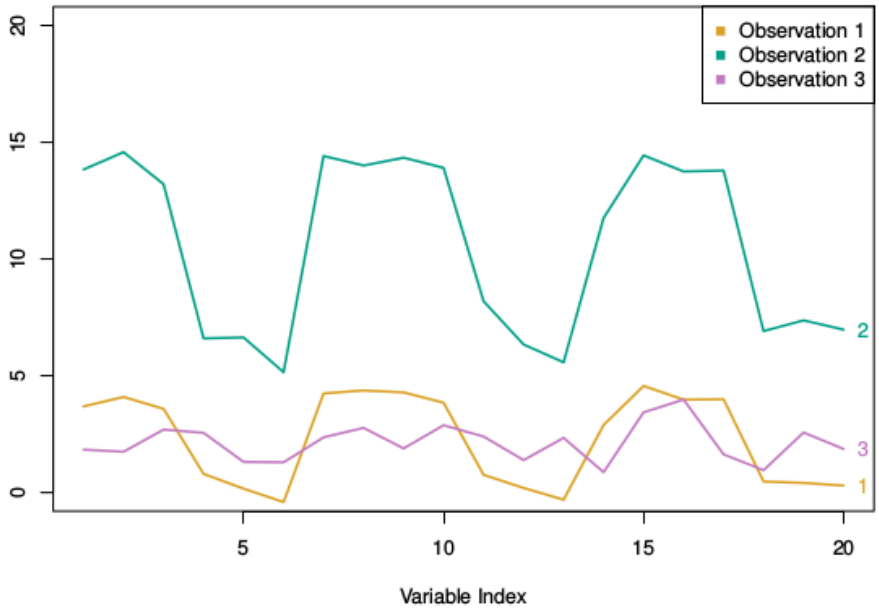
10.4.2 Types of Linkage

- Complete:
  - Maximal inter-cluster dissimilarity;
  - Compute all pairwise dissimilarities between cluster A&B;
  - Record the largest of these dissimilarities
  - 即记录两个cluster彼此距离最远的部分(最差情况的聚类对)
- Single:
  - Minimal inter-cluster dissimilarity;
  - Record the smallest of these dissimilarities
  - 记录两个cluster中彼此距离最近的部分(最佳情况的聚类对)
- Average:

    - Mean inter-cluster dissimilarity;
    - Record the average of these dissimilarities.
    - 两个cluster中的元素彼此配对，然后记录平均距离
- Centroid:
    - Dissimilarity between A与B的质心(p向量的平均长度)
    - Can result in undesirable inversions
    - 记录两个cluster的质心的距离

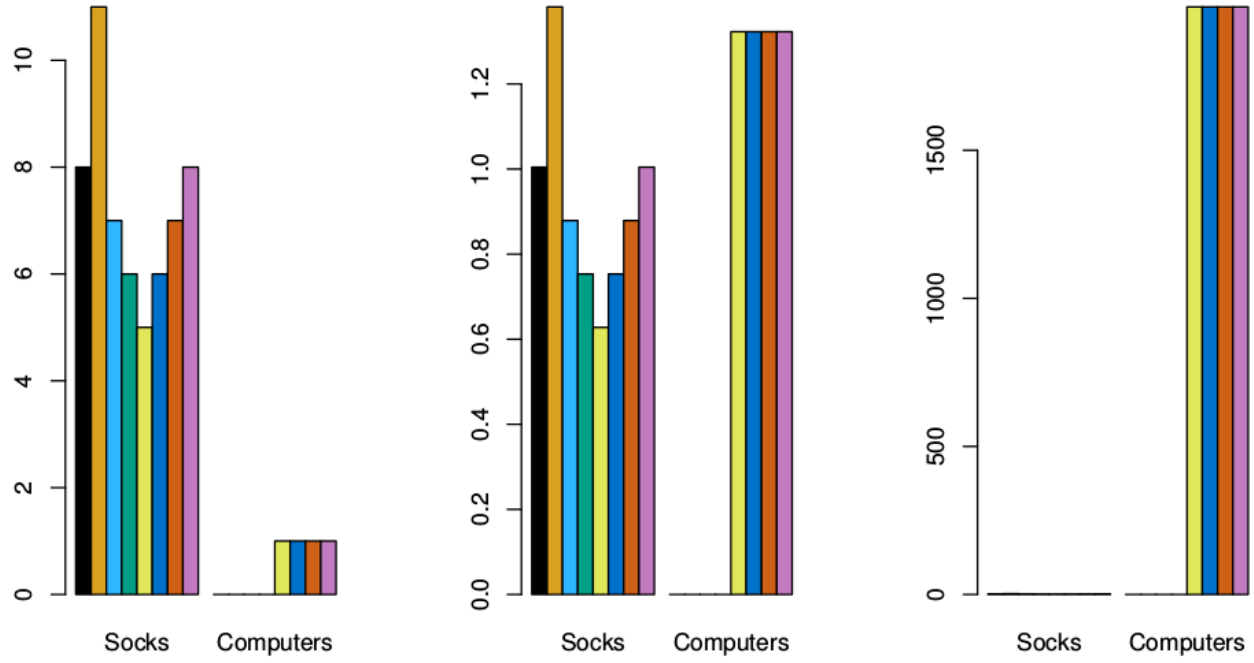
10.4.3 Dissimilarity Measure

- Euclidean distance
- Alternative: correlation-based distance
    - if the features of 2 observations are highly correlated,consider them to be similar
    - 这个方法一般用于变量而较少用于相关性



10.4.4 Practical issues in K-means and hierarchical clustering

- 是否需要归一化? #e.g. mean=0,sd=1



- 对于hierarchical clustering:
  - 确定dissimilarity量度方式;
  - 确定linkage类型;
  - 缺乏全局目标函数
  - 合并决策是最终的，代价昂贵
- 确定Clusters的数量
- 确定用于聚类的特征值

10.5 Unsupervised in R

10.5.1 Principal Components

```
dimnames(USArrests) #该data set给出各州不同犯罪类型的数量
apply(USArrests,2,mean) #查看全国各种犯罪平均值
apply(USArrests,2, var)

#使用 prcomp() 进行PCA
pca.out=prcomp(USArrests, scale=TRUE)
pca.out

#结果分析：PC1-4 SD递减
names(pca.out)
biplot(pca.out, scale=0)
```

10.5.2 K-means Clustering

```
#构造一个样本集
set.seed(101)
x=matrix(rnorm(100*2),100,2)
xmean=matrix(rnorm(8,sd=4),4,2)
which=sample(1:4,100,replace=TRUE) #K=4
x=x+xmean[which,]
plot(x,col=which,pch=19)

#使用 kmeans() 函数
km.out=kmeans(x,
        4, #K=4
        nstart=15) #选择15个随机数据集

plot(x,col=km.out$cluster,cex=2,pch=1,lwd=2)
points(x,col=which,pch=19)
points(x,col=c(4,3,2,1)[which],pch=19)
```

10.5.3 Hierarchical Clustering

```
#和刚才一样的数据集，演示三种模式

dist(x) #Distance Matrix Computation计算距离矩阵

hc.complete=hclust(dist(x),method="complete")
plot(hc.complete)

hc.single=hclust(dist(x),method="single")
plot(hc.single)

hc.average=hclust(dist(x),method="average")
plot(hc.average)

#使用 cutree() 将树分为4个level
hc.cut=cutree(hc.complete,4)
table(hc.cut,which)
table(hc.cut,km.out$cluster)

#或者可以使用组名which作为树形图叶片的标签
plot(hc.complete,labels=which)
```

10.6 Summary and Quiz

10.6.1 Summary

- 非监督学习可用于了解无标签数据的变量和组结构
- 主成分分析可以用于监督学习预处理
- 非监督学习比监督学习更难操作的原因：
  - no gold standard (e.g. outcome variable)
  - no single objective (e.g. test set accuracy)

10.6.2 Quiz

- K-means 为贪婪算法
- 如果非贪婪，需要考虑每种可能
- 对于每个观察值，都有K种可能: N1有K种可能，N2有K种可能...
- 由于彼此独立，总共需要K^n次运算，指数爆炸