

# Lecture 7: Simple Linear Regression

- <sup>2</sup> Text references: §12.1, 12.2, and 13.2 in Ruppert
- <sup>3</sup> This part provides an overview of the most basic regression model,
- <sup>4</sup> the **simple linear regression model**. In this case, there is a **single**
- <sup>5</sup> covariate  $x$  used to predict the response  $Y$ .  
*Predictor*
- <sup>6</sup> This model will serve as a building block for the more complex (and
- <sup>7</sup> more useful) models we will see later.
- <sup>8</sup> The model can be stated as follows:

<sup>9</sup> The response  $Y$  with single covariate  $x$  is assumed to normally distributed with mean  $\beta_0 + \beta_1 x$  and variance  $\sigma^2$ . Further, the response values  $Y$  are assumed to be independent from one observation to the next.

- <sup>13</sup> This is often written as follows:

$$Y = \beta_0 + \beta_1 x + \epsilon$$

- <sup>15</sup> where  $\epsilon$  is normal with mean zero and variance  $\sigma^2$ . The  $\epsilon$  are iid from one observation to the next.

<sup>1</sup> **Comment:** This model is sometimes written as

<sup>2</sup> 
$$Y = \alpha + \beta x + \epsilon.$$

<sup>3</sup> This notation is the origin for references to "the alpha" and "the beta"  
<sup>4</sup> for a stock.

<sup>5</sup> **Exercise:** Enumerate the assumptions that underlie the simple lin-  
<sup>6</sup> ear regression model.

<sup>7</sup> <sup>1</sup> A linear relationship between  $x$  and  $y$

<sup>8</sup> <sup>2</sup>  $E(\epsilon) = 0$

<sup>10</sup> <sup>3</sup>  $V(\epsilon) = \sigma^2$  (constant across observations)

<sup>11</sup> "homoskedasticity"

<sup>13</sup> <sup>4</sup>  $\epsilon$  are uncorrelated (or, sometimes, independent)

<sup>15</sup> <sup>5</sup>  $x$  is observed without error

<sup>17</sup> error in  $x \Rightarrow$  "attenuation bias" in slope estimate

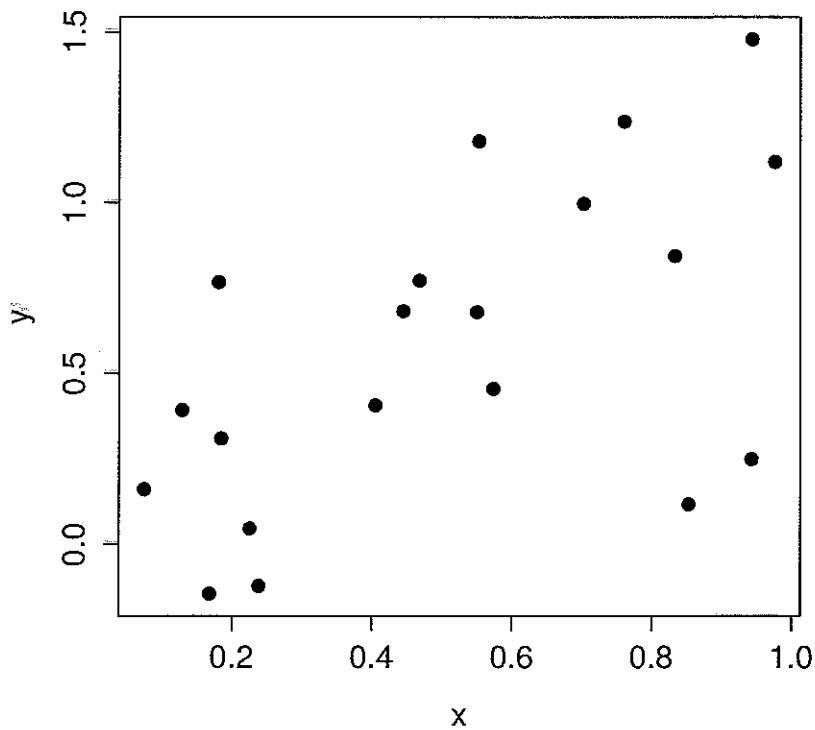
<sup>18</sup> <sup>6</sup>  $\epsilon$  are normal (but not really needed for  
many of the important results)

<sup>20</sup> <sup>7</sup> When more than one predictor, start to worry about  
collinearity

---

## 2 Estimation via a Sample

- 3 In practice, we rely on a sample of  $n$  pairs  $(x_i, y_i)$  for  $i = 1, 2, \dots, n$ ,
- 4 assumed to be drawn from a larger population. These will be used
- 5 to estimate  $\beta_0$ ,  $\beta_1$  and  $\sigma^2$ . As before, the standard notation for these
- 6 estimates will be  $\hat{\beta}_0$ ,  $\hat{\beta}_1$ , and  $\hat{\sigma}^2$ .
- 7 The sample is naturally depicted in a **scatter plot**, with example shown
- 8 in Figure 1.

Figure 1: A simple scatter plot, with  $n = 20$ .

- 1 Some additional notation we should establish immediately: For any  
 2 estimated regression line, the fitted values are

3  $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i, \quad i = 1, 2, \dots, n$

- 4 and the **residuals** are

5  $\hat{\epsilon}_i = y_i - \hat{y}_i, \quad i = 1, 2, \dots, n.$

approx. to  $\epsilon_i$ , but  $\hat{\epsilon}_i \neq \epsilon_i$

$$\epsilon_i = y_i - (\beta_0 + \beta_1 x_i)$$

$$\hat{\epsilon}_i = y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)$$

- 6 **Exercise:** Use the scatter plot below, with a fit regression line super-  
 7 imposed, to demonstrate the fitted values and the residuals.

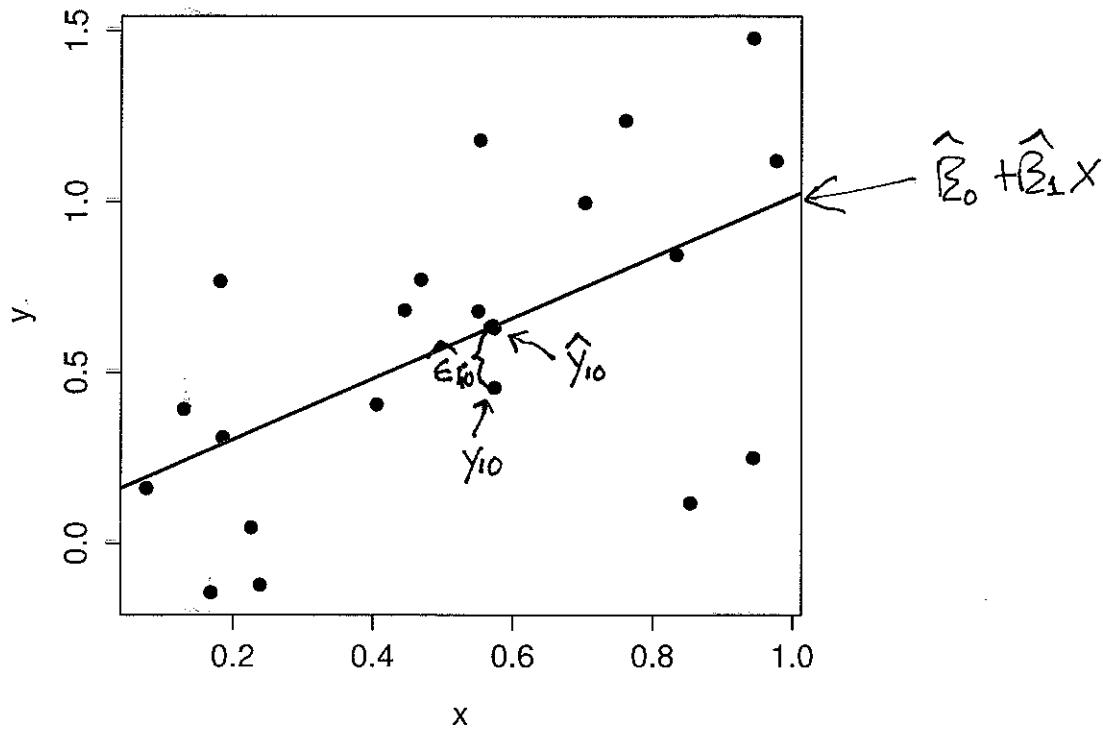


Figure 2: A simple scatter plot, with  $n = 20$ , with a regression line fit shown.

1

---

## 2 The Sample Covariance and Correlation

3 \* **Exercise:** Contrast covariance and correlation.

4 Recall that the covariance (and a related quantity, correlation) are  
5 measures of the **linear** association between two random variables  $X$   
6 and  $Y$ :

7

$$\text{Cov}(X, Y) = E[(X - E(X))(Y - E(Y))]$$

8 and

9

$$\text{Corr}(X, Y) = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y}$$

10 Not surprisingly, sample versions of these quantities are relevant to  
11 simple linear regression.

12 First, the **sample covariance** between observations of  $n$  pairs  $(x_i, y_i)$ :

13

$$s_{xy} = \left( \frac{1}{n-1} \right) \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

14 Note the similarity between this and the sample variance we defined  
15 earlier:

16

$$s_x^2 = \left( \frac{1}{n-1} \right) \sum_{i=1}^n (x_i - \bar{x})^2$$

- Then, the **sample correlation** is

$$r_{xy} = \frac{s_{xy}}{s_x s_y}.$$

- This quantity is typically just denoted with  $r$ . Figure 3 shows some examples of scatter plots, and the value of  $r$  for each. It is useful for building some intuition as to what different values of  $r$  mean.

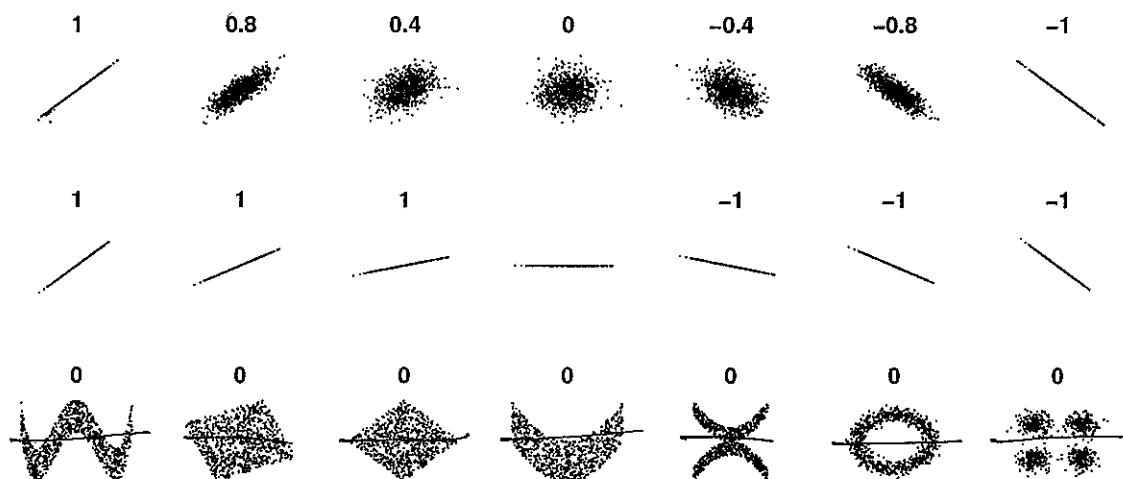
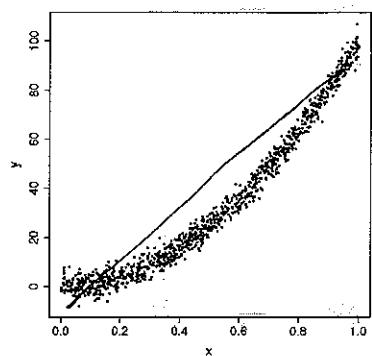


Figure 3: Examples of scatter plots, and corresponding values for  $r$ .

- Exercise:** Consider the scatter plot below. What, approximately, is the sample correlation?



$$r \approx 0.96$$

$|r|$  close to 1  $\Rightarrow$  linear model  
is a good fit

1

## 2 The Parameter Estimates

3 \* **Exercise:** What route would you use to derive  $\hat{\beta}_0$  and  $\hat{\beta}_1$ ?

4 Use maximum likelihood.

5  
6 See following page.

7  
8  
9  
10 If we were to follow through with this, we would find that the op-  
11 timal estimates would result from minimizing the **residual sum of**  
12 **squares (RSS):**

13

$$\text{RSS}(\beta_0, \beta_1) = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2$$

14 And a little bit of calculus would show that

15

$$\hat{\beta}_1 = \frac{\sum_i (y_i - \bar{y})(x_i - \bar{x})}{\sum_i (x_i - \bar{x})^2} = \frac{s_{xy}}{s_x^2} = r_{xy} \left( \frac{s_y}{s_x} \right)$$

16 and

17

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}.$$

$$Y_i \sim N(\beta_0 + \beta_1 x_i, \sigma^2) \quad i=1, 2, \dots, n \quad \Theta = [\beta_0 \ \beta_1 \ \sigma^2]$$

$$f_{Y_i}(y_i; \Theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - (\beta_0 + \beta_1 x_i))^2}{2\sigma^2}\right)$$

$$L(\Theta) = \prod_{i=1}^n f_{Y_i}(y_i; \Theta) \quad \text{RSS}(\beta_0, \beta_1)$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{\sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2}{2\sigma^2}\right)$$

Maximizing  $L(\Theta)$  over  $(\beta_0, \beta_1)$  is equivalent  
to minimizing  $\text{RSS}(\beta_0, \beta_1)$

- 1 \* **Exercise:** Why do we use **least squares** to estimate the parameters in regression?

3 (1) MLE in case of normal errors

4 (2) Analytically simple to derive  $\hat{\beta}_0$ , and  
5  $V(\hat{\beta}_i)$ , etc.

7 (3) Gauss-Markov Theorem: LS estimates are  
8 BLUE of  $\beta_i$ , regardless of dist. of  $\epsilon_i$

9 ↑  
10 "Best" = "smallest variance"

- 10 \* **Exercise:** Suppose that the roles of the predictor and response variables were interchanged in this regression model. What would be the effect on the slope of the regression line?

11 It's not simply the reciprocal.

12 original slope (y is resp) =  $r_{xy} \left( \frac{s_y}{s_x} \right)$   
13 new slope (x is resp) =  $r_{xy} \left( \frac{s_x}{s_y} \right)$

14 "There are two regression lines"

15

16

- 1 Figure 4 illustrates this idea. On the same scatter plot, two regression lines are shown: The “dashed-dotted” line is the regression line for predicting GOOG excess return from the S&P 500 excess return. The “dashed” line is the regression line for predicting S&P 500 excess return from GOOG excess return.
- 6 The solid line is the **SD Line**. This has a slope equal to  $s_y/s_x$ .

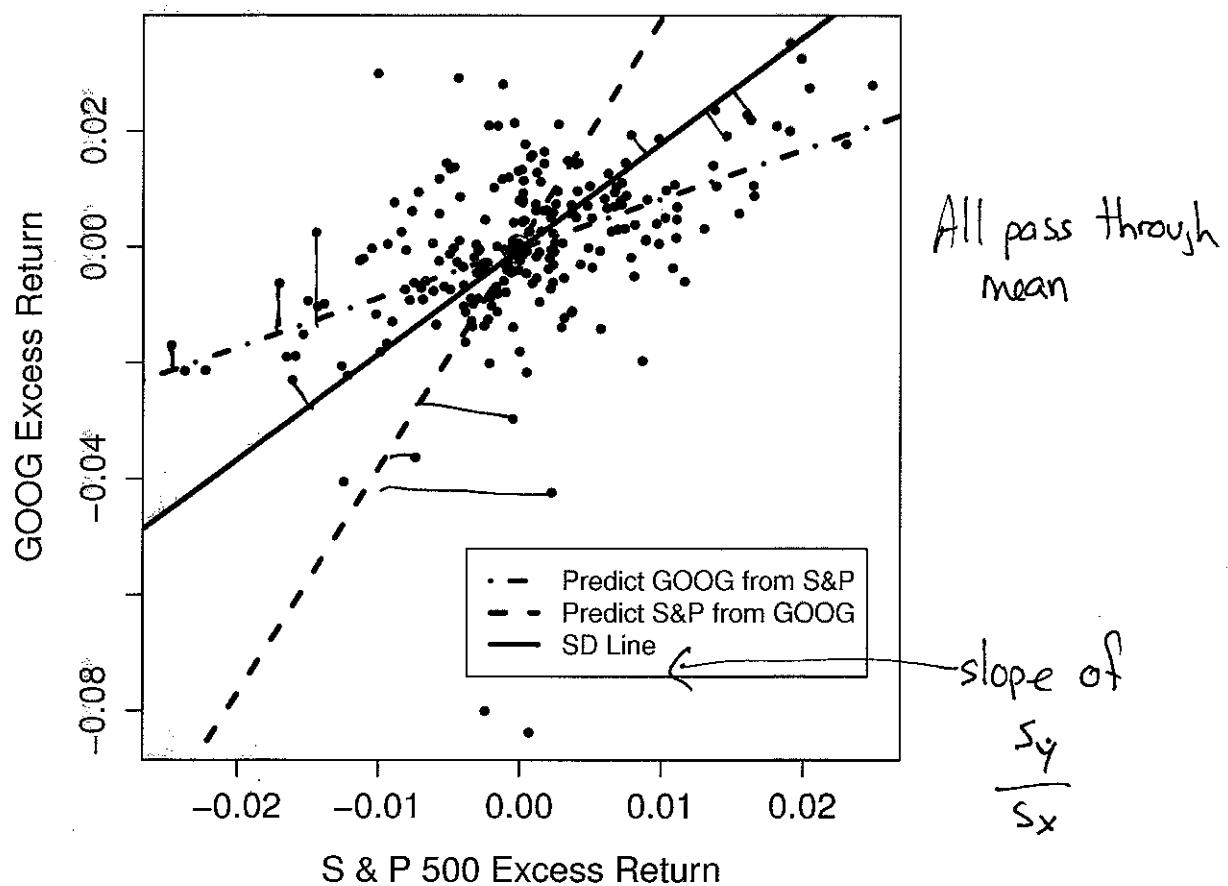


Figure 4: Daily excess returns for GOOG versus daily excess returns for S & P 500 for Jan. to Nov. 2012.

2 Estimating  $\sigma^2$ 3 The MLE for  $\sigma^2$  can be derived to be

4 
$$\hat{\sigma}_{\text{MLE}}^2 = \left( \frac{1}{n} \right) \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \text{RSS}/n.$$

5 This estimator is biased, but it can be adjusted to obtain an unbiased  
6 estimator

7 
$$\hat{\sigma}^2 = \left( \frac{n}{n-2} \right) \sigma_{\text{MLE}}^2 = \left( \frac{1}{n-2} \right) \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \text{RSS}/(n-2).$$

8 This is the estimator we will typically use, and what is reported by R  
9 and other software packages.10 **Exercise:** Is there intuition as to why we divide by  $(n-2)$ ?

$$11 \quad \begin{aligned} \mathbf{y} &= \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} & \hat{\mathbf{y}} &= \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} & \hat{\mathbf{e}} &= \mathbf{y} - \hat{\mathbf{y}} \end{aligned}$$

14  $\mathbf{y}$  lies in an  $n$ -dimensional space

15 
$$\hat{\mathbf{y}} = \hat{\beta}_0 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + \hat{\beta}_1 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \Rightarrow \hat{\mathbf{y}} \text{ is in a 2-dim. space}$$

19  $\hat{\mathbf{e}} = \mathbf{y} - \hat{\mathbf{y}}$  is in an  $n-2$  dim. space, i.e.  $\hat{\mathbf{e}}$  has  $(n-2)$  d.f.The are 2 linear constraints on  $\hat{\mathbf{e}}$

Note: If remove intercept from model, then

$$\sum_{i=1}^n \hat{\epsilon}_i \neq 0$$

$$Y = \beta_1 X + \epsilon \quad \text{"regression thru the origin"}$$

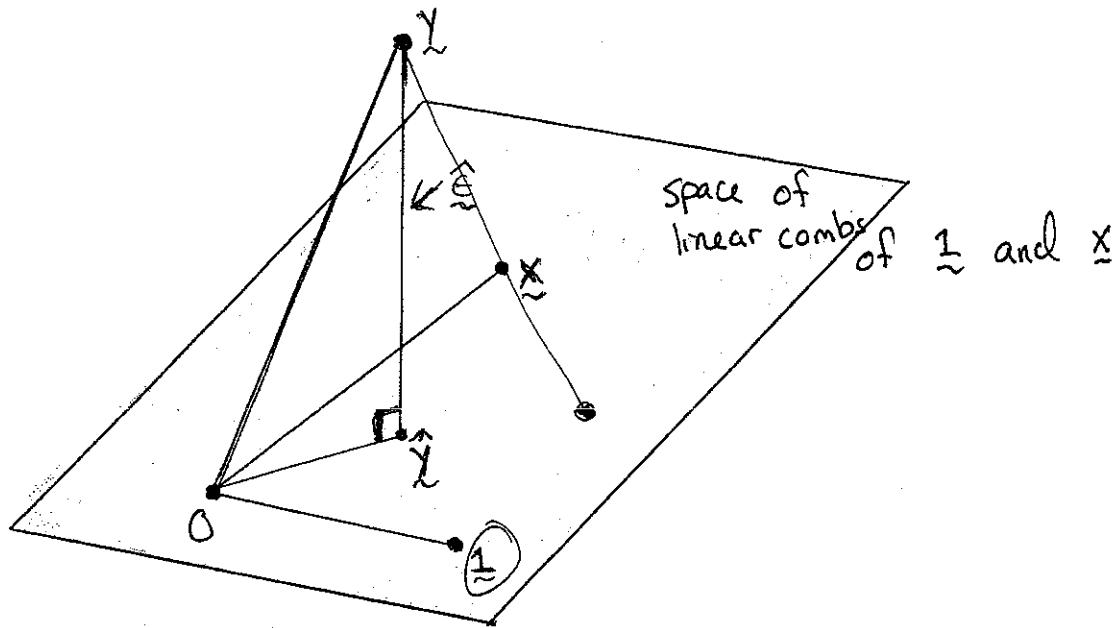


Figure 5: The geometric interpretation of least squares.

- 1. Figure 5 is a very useful geometric depiction of what happens when
- 2. we do least squares. The vector of fitted values  $\hat{y}$  is the orthogonal
- 3. projection of  $y$  onto the space spanned by the vectors  $1$  and  $x$ . (The
- 4. rectangle represents all of the possible vectors that can be formed
- 5. from linear combinations of  $1$  and  $x$ .)
- 6. **Exercise:** What is the sum of the residuals in OLS?

$$\sum_{i=1}^n \hat{e}_i = 0$$

8.  $\hat{e}$  is orthogonal to all vectors in the 2-d  
9. space spanned by  $1$  and  $x$

10.  $\hat{e}^T h = 0$  for any  $h$  in that space,  $1$  is in there  
11. so  $\hat{e}^T 1 = 0$

---

## 2 Inference Regarding $\beta_1$

- 3 Within the context of this simple model, the primary objective is to  
4 perform inference regarding  $\beta_1$ , since it captures information regard-  
5 ing the relationship (or lack thereof) between the response and the  
6 single predictor.
- 7 Here are some key results in this direction. Keep in mind that these  
8 are all true under the assumptions put forth for the simple linear  
9 regression model.

- 10 •  $\hat{\beta}_1$  is an unbiased estimator of  $\beta_1$ .  
11 • The variance of  $\hat{\beta}_1$  is

$$12 \quad V(\hat{\beta}_1) = \frac{\sigma^2}{(n-1)s_x^2} \approx \frac{\hat{\sigma}^2}{(n-1)s_x^2}.$$

13 This means that the standard error is

$$14 \quad \text{SE}(\hat{\beta}_1) \approx \frac{\hat{\sigma}}{s_x \sqrt{n-1}}.$$

- 15 • Suppose that the true value of the slope is  $\beta_{10}$ . Then

$$16 \quad T = \frac{\hat{\beta}_1 - \beta_{10}}{\text{SE}(\hat{\beta}_1)}$$

17 has the  $t$ -distribution with  $(n-2)$  degrees of freedom. Of course,  
18 when  $n$  is sufficiently large,  $T$  will be approximately standard  
19 normal.

1. **Exercise:** Construct a  $100(1 - \alpha)\%$  confidence interval for  $\beta_1$ .

$$2. P(-t_{\alpha/2, n-2} \leq T \leq t_{\alpha/2, n-2}) = 1 - \alpha$$

$$3. P(-t_{\alpha/2, n-2} \leq \frac{\hat{\beta}_1 - \beta_{10}}{SE(\hat{\beta}_1)} \leq t_{\alpha/2, n-2}) = 1 - \alpha$$

$$4. P(\hat{\beta}_1 - t_{\alpha/2, n-2} SE(\hat{\beta}_1) \leq \beta_{10} \leq \hat{\beta}_1 + t_{\alpha/2, n-2} SE(\hat{\beta}_1)) = 1 - \alpha$$

5.  $\hat{\beta}_1 \pm t_{\alpha/2, n-2} SE(\hat{\beta}_1)$  is a  $100(1 - \alpha)\%$  CI  
 6. for  $\beta_1$

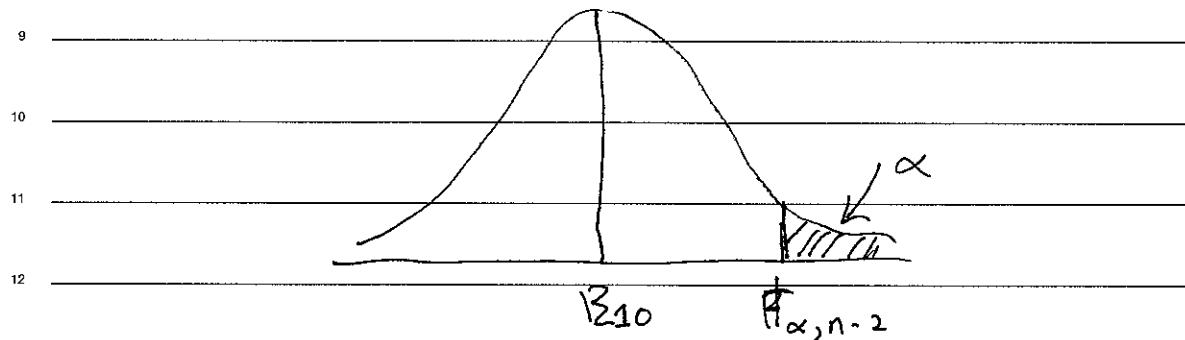
12.   
13.   
14.   
15.   
16.   
17.   
18.   
19.   
20.

- 1 \* **Exercise:** Construct a level  $\alpha$  hypothesis tests regarding  $\beta_1$ . Consider both the one-sided and two-sided alternatives.

3 Use  $T = \frac{\hat{\beta}_1 - \beta_{10}}{SE(\hat{\beta}_1)}$  as test statistic

4 to test  $H_0: \beta_1 = \beta_{10}$  - know  $T$  has  
5 t-dist. with  $n-2$  d.f. under  $H_0$

6 If testing  $H_0: \beta_1 = \beta_{10}$  vs.  $H_1: \beta_1 > \beta_{10}$ ,  
7 reject if  $T$  is larger than  $t_{\alpha, n-2}$



13  $P(\text{Type I error}) = P(T > t_{\alpha, n-2}) = \alpha$

15 If testing  $H_0: \beta_1 = \beta_{10}$  vs.  $H_1: \beta_1 \neq \beta_{10}$ ,  
16 Reject if  $T > t_{\alpha/2, n-2}$  or  $T < -t_{\alpha/2, n-2}$

18 Most common to test with  $\beta_{10} = 0$ , i.e.

20  $T = \frac{\hat{\beta}_1}{SE(\hat{\beta}_1)}$

- 1 A common (overused) objective is to test for evidence if one can
- 2 claim that  $\beta_1 \neq 0$ . Is there "significance?"
  
- 3 **Exercise:** Give two ways of using the inference procedures constructed
- 4 above to address this objective.

5 ① Construct CI for  $\beta_1$ , see if interval  
6 contains 0

7 0 in interval  $\Rightarrow \beta_1$  "might be = 0"

8 0 not in interval  $\Rightarrow \beta_1 \neq 0$  evidence for this

9 ② Test  $H_0: \beta_1 = 0$  vs.  $H_1: \beta_1 \neq 0$

10 If reject  $H_0$ , can "claim"  $\beta_1 \neq 0$

11

12

13

14

15

16

17

18

19

20

- 1 It is important to be mindful of the following fact: If you fail to reject
- 2  $H_0: \beta_1 = 0$ , or if the confidence interval for  $\beta_1$  includes zero, this is
- 3 often not because  $\beta_1 = 0$ , but instead because the standard error for
- 4  $\hat{\beta}_1$  is too large to detect the deviation from zero.
- 5 \* **Exercise:** How can the standard error of  $\hat{\beta}_1$  be decreased?

6 Get more data : increase  $n$

$$7 SE(\hat{\beta}_1) = \frac{\hat{\sigma}}{s_x \sqrt{n-1}}$$

- 9
- 10
- 11
- 12
- 13

- 14 One of the most common mistakes made when constructing models
- 15 is to blindly remove a predictor from the model solely because there
- 16 is not strong evidence that its coefficient does not equal zero. Model
- 17 building (the process of deciding which covariates to include) should
- 18 comprise many factors, including our knowledge of the situation,
- 19 the importance that we are placing on constructing a **parsimonious** (*simple*)
- 20 model, and the results of formal statistical inference procedures.

---

## 2 The Coefficient of Determination, $R^2$

- 3 When working with regression models it is very common to hear
- 4 references to "the  $R^2$  for the model."
- 5 This is also called the **coefficient of determination**.
- 6 The idea is simple:  $R^2$  equals the proportion of the sum of squared
- 7 response which is accounted for by the model that has been fit, rela-
- 8 tive to the model with no covariates.

- 9 In particular:

$$10 R^2 = 1 - \frac{\overbrace{\sum_{i=1}^n (y_i - \hat{y}_i)^2}^{\text{RSS for model with preds}}}{\underbrace{\sum_{i=1}^n (y_i - \bar{y})^2}_{\text{RSS for model with no preds}}}$$

- 11 Note that  $0 \leq R^2 \leq 1$ .

- 12 In the case of simple linear regression (a single predictor),  $R^2$  does
- 13 indeed equal the sample correlation ( $r$ ) squared.

- $R^2$  is overused. In particular, note that it is possible for  $R^2$  to be close to one, but the model be a terrible fit. And it is possible for a great-fitting model to have  $R^2$  close to zero. See Figure 6.

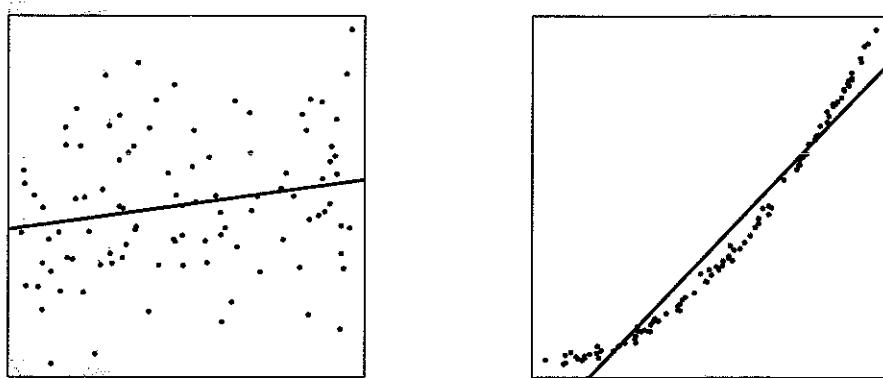
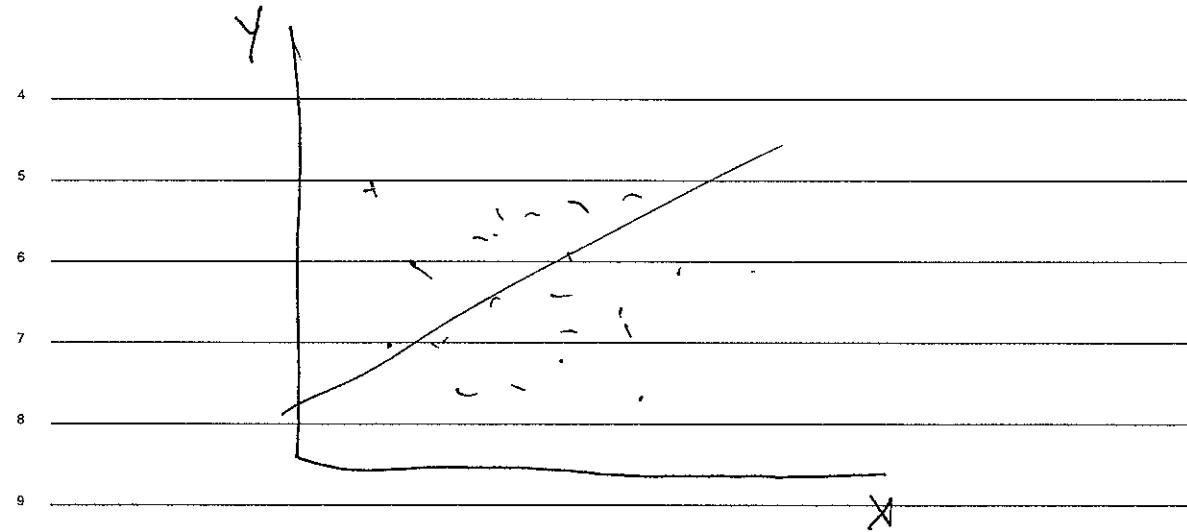


Figure 6: The scatter plot on the left has  $R^2 \approx 0.05$ , while that on the right has  $R^2 \approx 0.95$ . The data in the left plot were simulated from the normal linear model.

- In short,  $R^2$  should not be relied upon as a diagnostic to judge the quality of the model fit. If the model is a good fit, then  $R^2$  says something about the **predictive power** of the model.

- 1 \* **Exercise:** Suppose that, by mistake, each observed pair (response  
 2 and predictor) is included twice in the analysis. What is the effect on  
 3  $\hat{\beta}_i$ , the  $t$ -statistics, and  $R^2$ ?



10  $\hat{\beta}_0$  and  $\hat{\beta}_1$  are the same

11  $R^2$  won't change ( $r$  won't change)

13 t-stats will change  $\left( T = \frac{\hat{\beta}_i}{SE(\hat{\beta}_i)} \right)$

15  $SE(\hat{\beta}_1) = \frac{\hat{\sigma}}{S_x \sqrt{n-1}}$

17  $\hat{\sigma}^2 = \frac{RSS}{n-2}$  and  $S_x$  stay same (almost)

18 but  $\sqrt{n-1}$  changes to  $\sqrt{2n-1}$

19 Overall effect is to multiply t-stat  $\approx \sqrt{2}$

## 2 Diagnostics

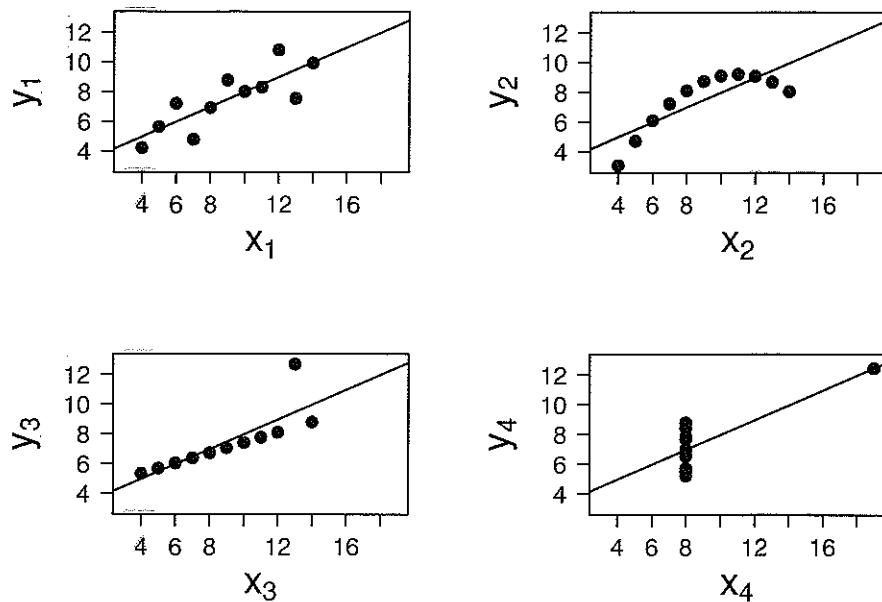


Figure 7: Anscombe's Quartet.

3 Figure 7 shows four scatter plots constructed by Francis Anscombe  
 4 for a 1973 research article. These are called **Anscombe's Quartet**. In  
 5 R, the data for these can be obtained by simply using the command  
 6 `> data(anscombe)`

7 **Exercise:** Use R to find the sample mean and variance for both  $x$  and  
 8  $y$  in all four cases, to find the correlation between  $x$  and  $y$  in each case,  
 9 and then use that to find the slope of the regression line for each.

10  $\bar{x}, \bar{y}, \text{cor}(x,y), s_x^2, s_y^2$  are same in all 4 cases

11  $\hat{B}_1$  same in all cases

- 1 **Exercise:** This can be taken a step further: We could find that the
- 2 residual sum of squares in all four cases is exactly the same. What
- 3 does this imply about the inference for  $\beta_1$  in all four cases?

4 T is the same in all 4 cases

5 All inference proc. give same result

- 6
- 7

- 8 **Exercise:** What is the message we take away from this example?

9 There is no replacement for looking at  
10 the data!

- 11

- 12

- 13

- 14

- 15

- 16

- 17

- 1 A very useful diagnostic plot is that of the **residuals versus the fitted values**. There are a wide range of models for which this is an invaluable tool.
- 4 Figure 8 shows these plots for the four scatter plots of Figure 7.

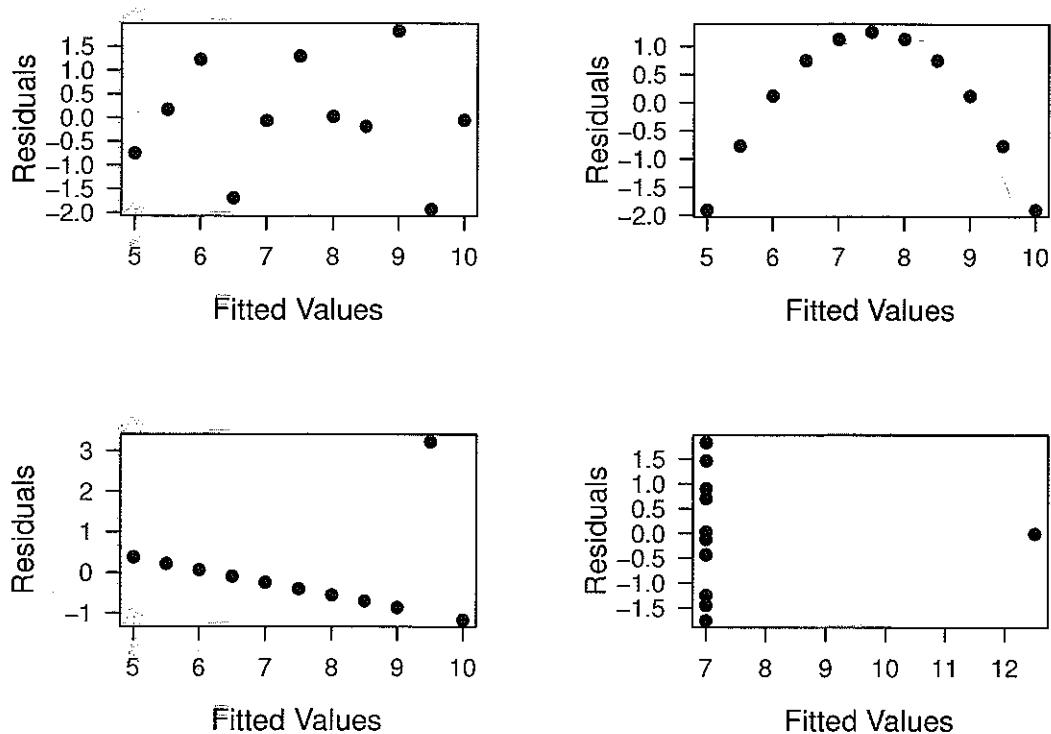


Figure 8: The plots of residuals versus fitted values for Anscombe's Quartet.

- 1 **Exercise:** If the assumptions of the model are correct, what should
- 2 we see in the plot of residuals versus fitted values?

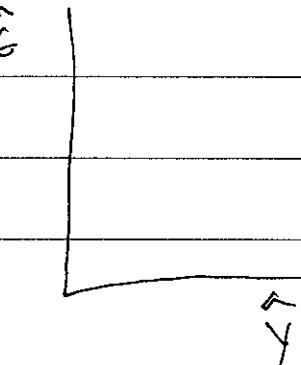
$\hat{\epsilon}$  approximate  $\epsilon$ , and  $\epsilon$  are assumed  
to be unrelated to anything else,  
including the fitted values

So, want to see lack of pattern in this  
plot

- 11 **Exercise:** Why is the plot of residuals versus fitted values any more
- 12 useful than simply looking at the scatter plot?

Imagine case where  $X$  is high-dimensional.

Fitted values are still a single vector.



- 1 \* **Exercise:** Your colleague says, "I found the correlation between the  
 2 residuals and fitted values to be equal to zero, so there is clearly no  
 3 relationship between the residuals and fitted values. So, the model  
 4 must be fine." Do you agree with this logic? No.

5  $\text{cor}(\hat{y}, \hat{\epsilon}) = 0$  by construction when  $B_0$  included  
 in model.

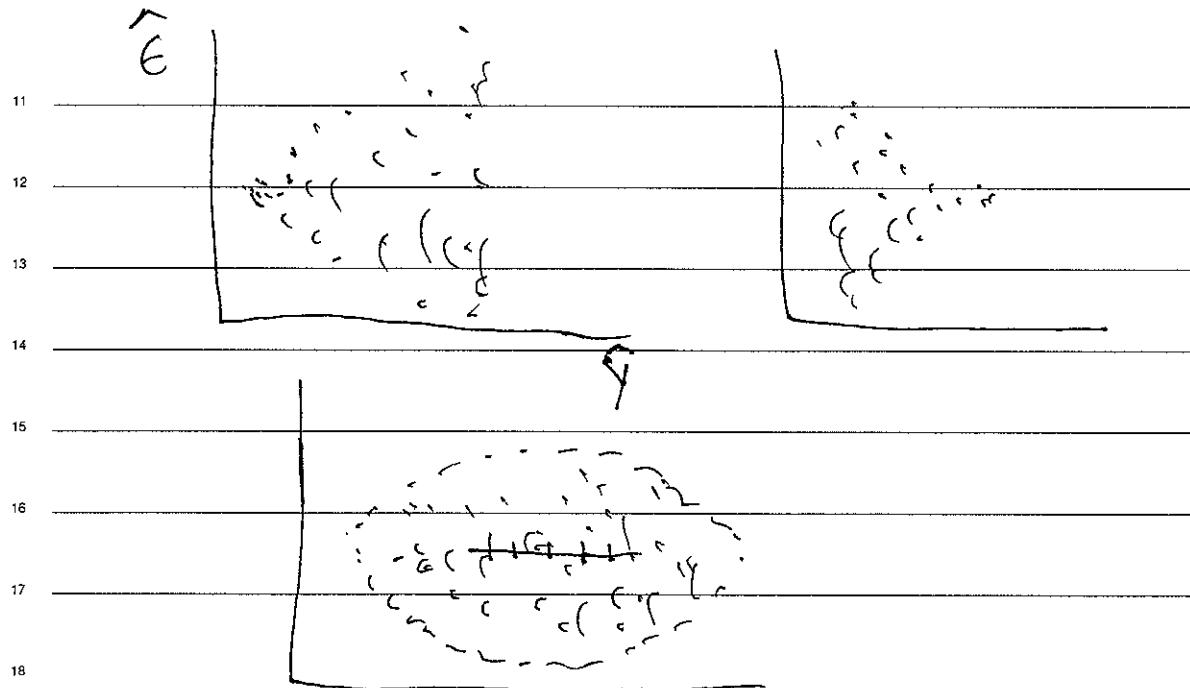
6  $\text{cor}(\hat{y}, \hat{\epsilon}) = \frac{1}{n-1} \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})(\hat{\epsilon}_i - \bar{\hat{\epsilon}})$   $= 0$   
 7  
 8  $= \frac{1}{n-1} \left[ \sum_{i=1}^n \hat{\epsilon}_i \hat{y}_i - \sum_{i=1}^n \hat{\epsilon}_i \bar{\hat{y}} \right]$   
 9  
 10

11  $= \frac{1}{n-1} [0 - 0]$   
 12

13  $= 0$   
 14

## 1 Heteroskedasticity

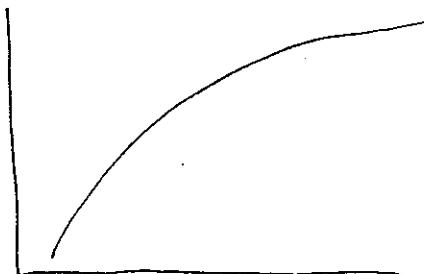
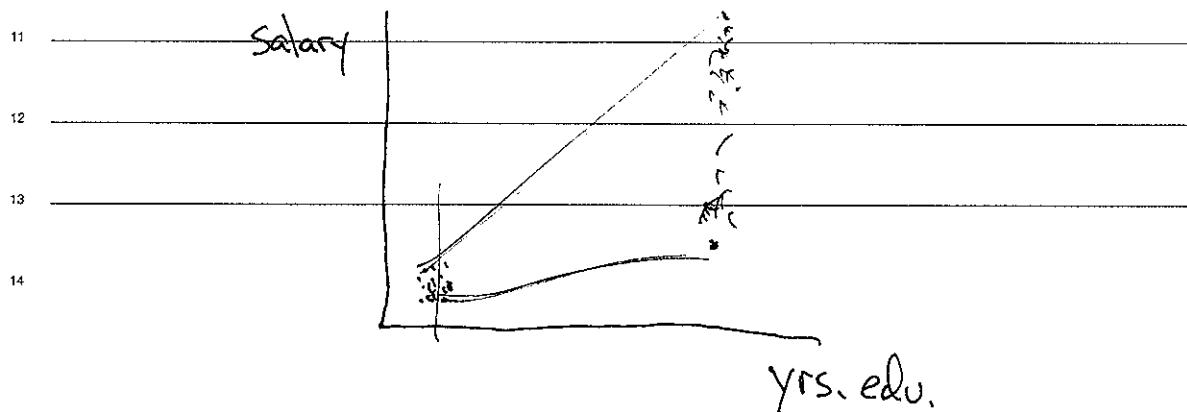
- 2 One assumption of our model is that all of the model errors have the
- 3 same variance; this condition is called **homoskedasticity**.
  
- 4 The opposite case, the presence of **heteroskedasticity** is a serious
- 5 concern. The validity of the confidence intervals, hypothesis tests,
- 6 and so forth rests on this assumption.
  
- 7 The plot of residuals versus fitted values is a useful tool for diagnos-
- 8 ing heteroskedasticity.
  
- 9 **Exercise:** What shapes would one expect to see in the plot of residu-
- 10 als versus fitted values when heteroskedasticity is present?



- 1 There are two major strategies for dealing with heteroskedasticity.
- 2 First, the response variable can be **transformed**, meaning you would
- 3 use  $g(y)$  instead of  $y$  as the response variable. The two most popular
- 4 choices for  $g()$  are the logarithm, and the square root.
- 5 **Exercise:** Why do you think that the log transform is so often used
- 6 with financial variables (prices, volume, etc.)?

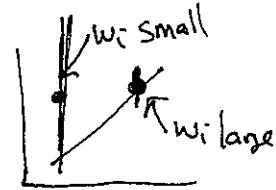
7 they are often positive variables

8 tends to be more variability on the  
9 high end of the dist.



- 1 The other major strategy for dealing with heteroskedasticity is to use
- 2 **weighted least squares.** Briefly stated, instead of minimizing the
- 3 residual sum of squares, one minimizes

$$4 \quad \text{WRSS}(\beta) = \sum_{i=1}^n w_i (y_i - \hat{y}_i)^2$$



- 5 where the weights are chosen to deemphasize those observations for
- 6 which the variance is larger.

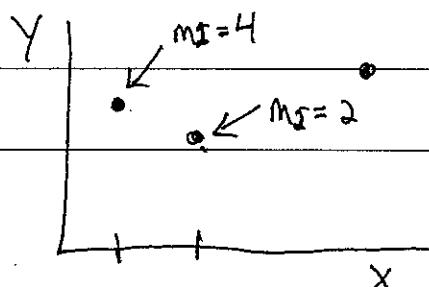
- 7 In fact, the optimal choice is  $w_i = 1/\sigma_i^2$ , where  $\sigma_i^2$  is the variance for
- 8 the  $i^{th}$  model error.

- 9 One would not typically know the  $\sigma_i^2$ , but it is sometimes the case
- 10 that you are willing to assume that  $\sigma_i^2 = \sigma^2/m_i$ . Then, the weight
- 11 can be chosen as  $w_i = m_i$ . Or, the  $\sigma_i^2$  could be estimated from the
- 12 residuals:  $w_i = \frac{1}{\sigma_i^2} = \frac{m_i}{\sigma^2}$

- 13 **Exercise:** Explain why it may be the case that you would be willing
- 14 to assume that  $\sigma_i^2 = \sigma^2/m_i$ .

15 If  $y_i$  is the mean of  $m_i$  observations,

16 each of which has variance  $\sigma^2$



1 Example

- 2 In this example we will use 82 NYSE stocks in an effort to predict the  
3 **realized volatility**, here defined as

4

$$\frac{1}{m} \sum_{i=1}^m \left( \log\left(\frac{P_i}{P_{i-1}}\right) \right)^2$$

- 5 where  $P_i$  is the closing price on day  $i$ .  $m = \# \text{ days obs.}$

- 6 For this simple model, we will use the logarithm of the total trading  
7 volume over the prior month as the predictor of the realized volatil-  
8 ity over the upcoming month.

- 9 Of course, to **train** or **fit** the model we need data on both the predictor  
10 and the **response**. So, we use two recent consecutive months to form  
11 a **training set**. We will use the aforementioned set of 82 stocks for this  
12 purpose.

- 13 A simple linear model is fit in attempt to model the relationship be-  
14 tween these two variables. Figure 9 shows the raw data and the re-  
15 sult of the fit.

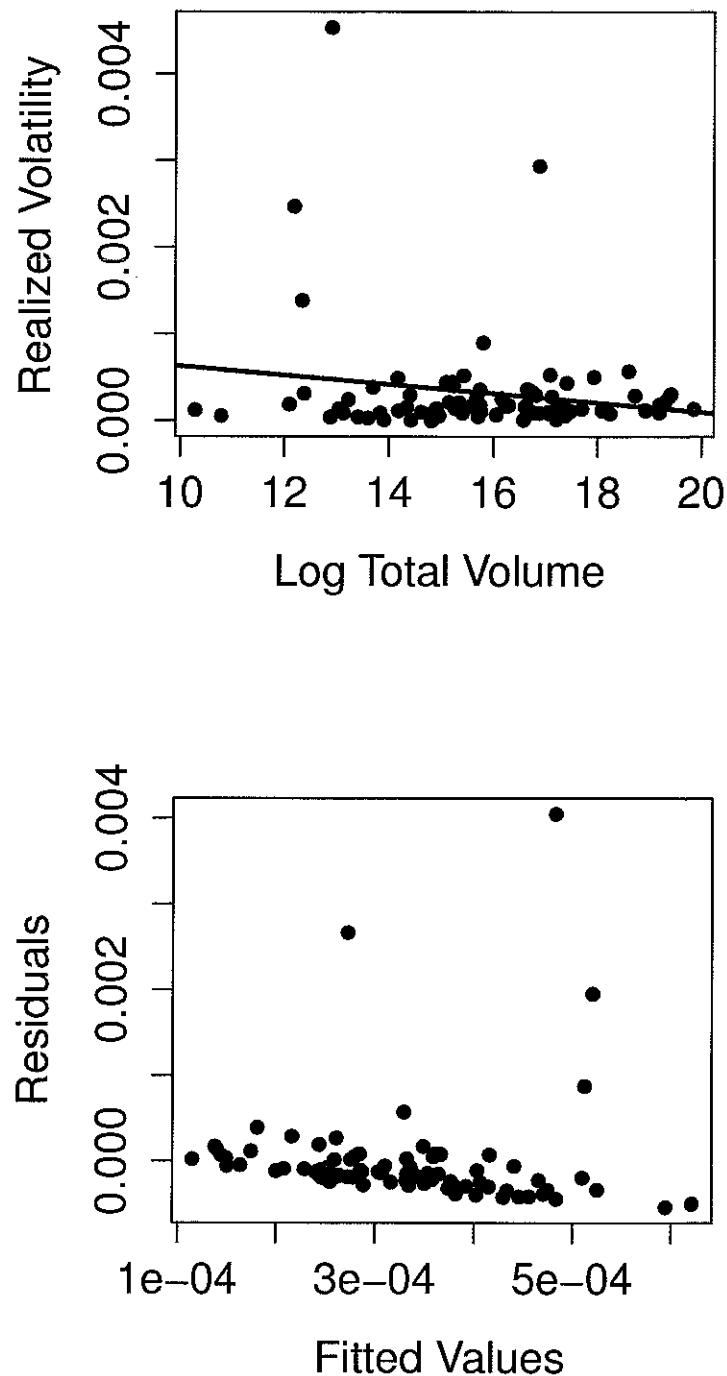


Figure 9: The initial model fit to realized volatility. The upper plot shows the scatter plot with the regression line superimposed, the lower plot shows the plot of residuals versus fitted values.

- 1 **Exercise:** Comment on the quality of the fit of this first model.

2 Terrible. Clear pattern in plot of resids. vs. fitted  
3 values.

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18 The model was refit with the four extreme observations removed.

19 Figure 10 shows the result.

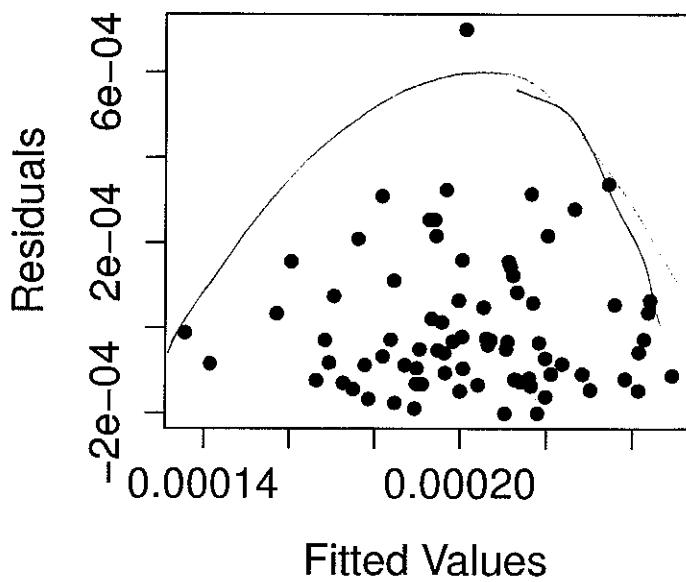
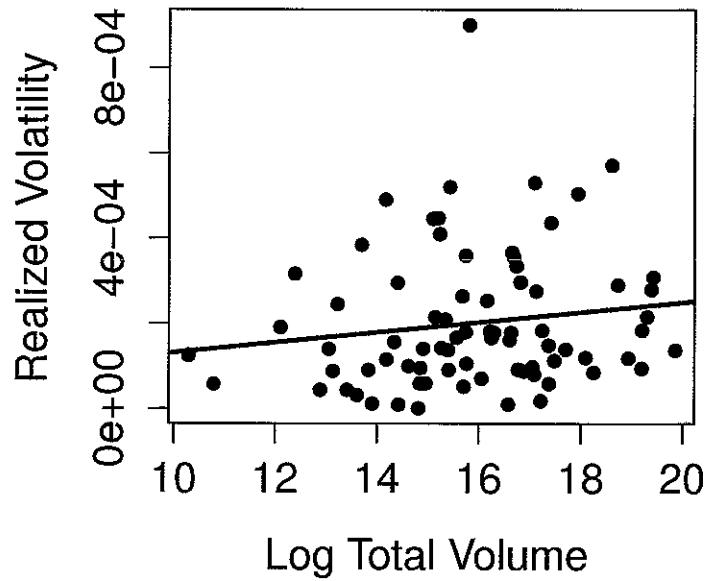


Figure 10: The model fit to realized volatility after the four extreme observations have been removed. The upper plot shows the scatter plot with the regression line superimposed, the lower plot shows the plot of residuals versus fitted values.

- 1 **Exercise:** Comment on the quality of the fit of this second model.

2 See improvement, but evidence of problems,  
3 maybe heteroskedasticity

4

5

6

7

8

9

10

11

12

13

14

15

16

17

- 18 The model was then refit with the four extreme observations returned,  
19 but using the log transform of the response. Figure 11 shows the re-  
20 sult.

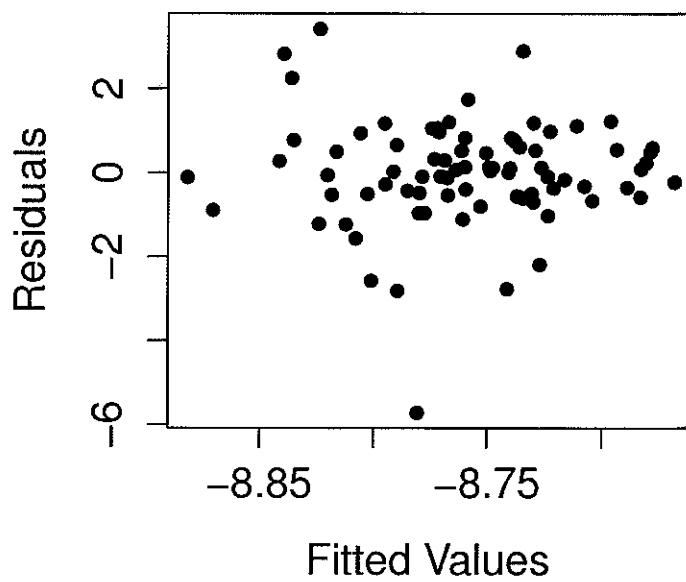
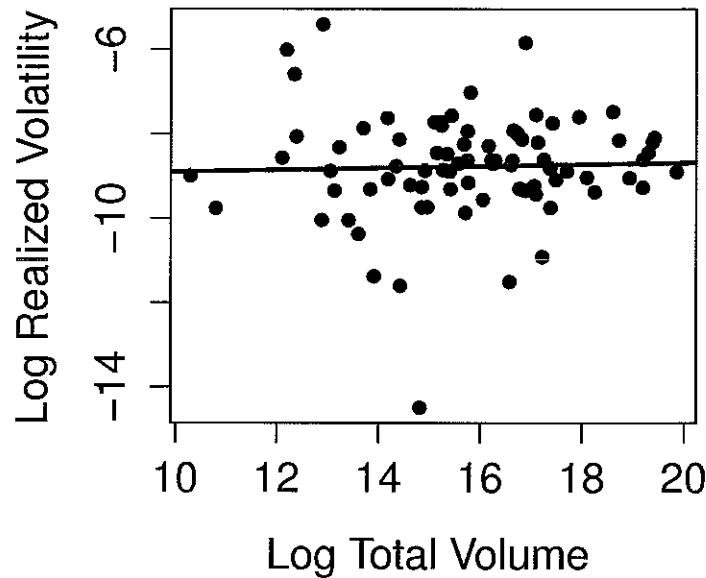


Figure 11: The model fit to realized volatility after taking the log transform. The upper plot shows the scatter plot with the regression line superimposed, the lower plot shows the plot of residuals versus fitted values.

- 1 **Exercise:** Comment on the quality of the fit of this third model.

2 Much better. Used all the data, and little  
3 pattern in plot of resids. vs. fits.

4

---

5

---

6

---

7

---

8

---

9

---

10

---

11

---

12

---

13

---

14

---

15

---

16

---

17

---

18

---

19

---

20

## 1 The Normality Assumption

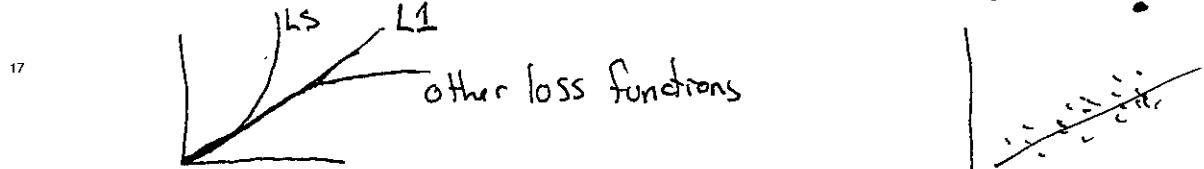
- 2 We should also assess the validity of the assumption that the model
- 3 errors are normally distributed. This is typically done by looking at
- 4 a normal probability plot of the residuals.
  
- 5 The distribution of the  $\hat{\beta}_i$  is only exactly normal when the model er-
- 6 rors are normal. **But**, there is a version of the central limit theorem
- 7 that states that  $\hat{\beta}_i$  will be approximately normal as long as the sample
- 8 size is sufficiently large.
  
- 9 Hence, the standard errors, confidence intervals, and hypothesis tests
- 10 we construct will still be approximately valid if the model errors are
- 11 not normal, as long as the sample size is sufficiently large.

- 1 Still, the normal probability plot is a valuable tool for identifying
- 2 outliers, which are objects with errors which are much larger than
- 3 would be expected under the normal distribution.
  
- 4 Also, if we find that the error distribution has heavier tails than would
- 5 be expected under the normal distribution, we may want to recon-
- 6 sider our use of least squares as the criterion for estimating  $\beta$ .
  
- 7 **\*Exercise:** Explain why least squares may no longer be a good choice
- 8 if the distribution of the errors was, for instance, the  $t$ -distribution
- 9 with 5 degrees of freedom.

10 LS no longer the MLE

11  
 12 least squares puts a large penalty on points  
 13 far from the line, much larger than using  
 14 abs. error, for instance

$$15 \sum_{i=1}^n \left| y_i - (B_0 + B_1 x_i) \right|^p \quad \begin{array}{l} LS \Rightarrow p=2 \\ L1 \text{ regr} \Rightarrow p=1 \end{array}$$



- 18 We will discuss the implementation of **robust regression** approaches
- 19 which use criterion other than least squares.

*t-dist. with 5 d.f. will generate large errors*

- 1 Figure 12 shows the normal probability plot for the example pre-  
2 sented above. This is for the case of the third model (Figure 11).

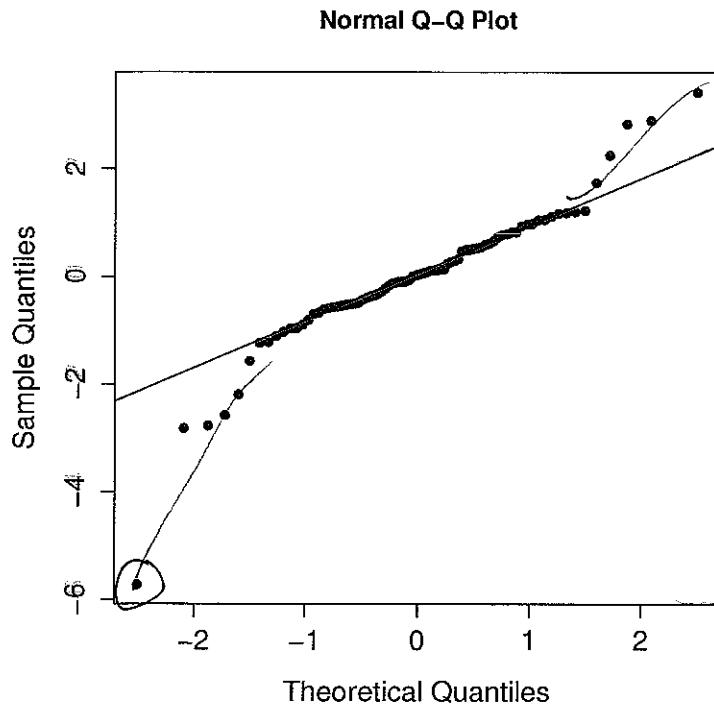


Figure 12: Normal probability plot corresponding to the model shown in Figure 11.

- 3 **Exercise:** Comment on the normal probability plot shown in Figure  
4 12. Is there an issue with outliers? Is the normal assumption valid?

5 Evidence of an outlier, clear that dist. of  
6 errors has heavier tail than normal.

7  
8 Since  $n$  only 82, would be concerned about  
9 using inference procedures

10

## 1 Other Diagnostic Plots

- 2 If there is a time dimension to the data that is being used (usually the
- 3 case with financial data), then one should plot the residuals versus
- 4 time, and see if there is time-dependence that is not being adequately
- 5 modelled.
  
- 6 In a subsequent course you will learn about ARMA and GARCH
- 7 models; these can be built into the assumptions regarding the model
- 8 error. See §18.12 in Ruppert. *ACF of residuals*
  
- 9 It is also a good idea to individually plot the residuals versus each of
- 10 the predictors in the model. If our model is adequate, there should
- 11 be no relationship between these quantities.

1 

## 2 Adding a Predictor

- 3 Practical regression models typically contain many predictors, i.e.,
- 4 we fit models of the form

5 
$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$$

- 6 As more predictors are added to a model, there can be surprising
- 7 effects on the previously-included predictors:

- 8 1. predictors can switch from being “significant” to being “not significant”
- 9
- 10 2. predictors can switch from being “not significant” to being “significant”
- 11
- 12 3. the signs on the parameter estimates can flip from negative to
- 13 positive, and vice-versa

*slopes*

- 12 3. the signs on the parameter estimates can flip from negative to
- 13 positive, and vice-versa

- \* **Exercise:** Explain why each of the above three things could occur,
- in the context of going from a model with a single predictor, to a
- model with two predictors.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

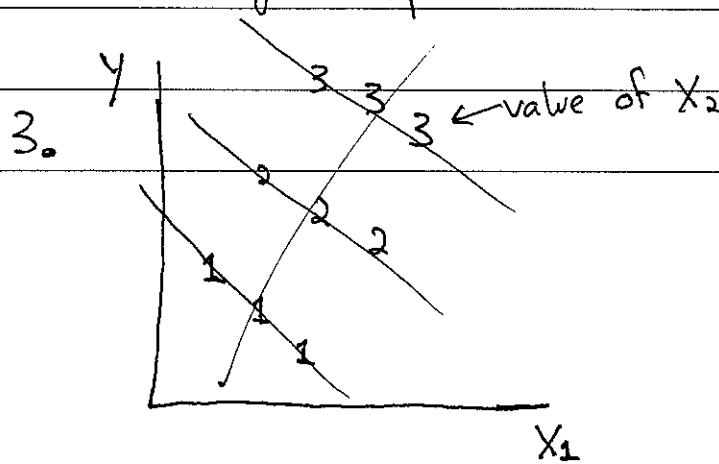
Suppose  $X_1$  is original predictor

1. If  $X_2$  is strongly correlated with  $X_1$  and  $X_2$  has a stronger correlation with  $Y$  than does  $X_1$ .  $X_2$  does the work of explaining variability in  $Y$  that  $X_1$  was doing previously.

2. Suppose  $X_1$  and  $X_2$  are not correlated and  $X_2$  has strong correlation with  $Y$ .

RSS in new model will be much smaller.

$SE(\hat{\beta}_1)$  is much smaller, T stat for  $\beta_1$  goes up



For fixed  $X_2$ , see negative relationship between  $y$  and  $X_1$

Ignoring  $X_2$ , see a positive relationship between  $y$  and  $X_1$

- 1 When there are two predictors (and the intercept) in a model, the
- 2 least squares slope estimates can be written as

$$\hat{\beta}_1 = \left( \frac{r_{y1} - r_{y2}r_{12}}{1 - r_{12}^2} \right) \left( \frac{s_y}{s_1} \right) \quad \hat{\beta}_1 = r_{y1} \left( \frac{s_y}{s_1} \right)$$

3 and

$$\hat{\beta}_2 = \left( \frac{r_{y2} - r_{y1}r_{12}}{1 - r_{12}^2} \right) \left( \frac{s_y}{s_2} \right)$$

4 where  $s_i$  is the sample standard deviation for predictor  $i$ .

5 **Exercise:** Consider three regression models:

$$Y = \beta_0 + \beta_1 x_1 + \epsilon$$

$$Y = \beta'_0 + \beta_2 x_2 + \epsilon$$

$$Y = \beta''_0 + \beta_1^* x_1 + \beta_2^* x_2 + \epsilon$$

- 6 Assume parameters are estimated using least squares. What is the
- 7 relationship between  $\hat{\beta}_1$  and  $\hat{\beta}_1^*$  and between  $\hat{\beta}_2$  and  $\hat{\beta}_2^*$ ? (It's cleaner
- 8 to restrict to the case where the response and the predictors have
- 9 been rescaled to each have standard deviation of one.)

$$\hat{\beta}_1^* = \left( \frac{\hat{\beta}_1 - \hat{\beta}_2 r_{12}}{1 - r_{12}^2} \right) \quad \hat{\beta}_1 = r_{y1}$$

$$\hat{\beta}_2^* = \left( \frac{\hat{\beta}_2 - \hat{\beta}_1 r_{12}}{1 - r_{12}^2} \right)$$

19

20

- 1 \* **Exercise:** Suppose that random variables  $X$ ,  $Y$ , and  $Z$  are such that  
 2  $\text{Corr}(X, Y) = 0.9$  and  $\text{Corr}(Y, Z) = 0.8$ . What is the smallest that  
 3  $\text{Corr}(X, Z)$  could be?

$$\begin{vmatrix} 1 & 0.9 & 0.8 \\ 0.9 & 1 & X \\ 0.8 & X & 1 \end{vmatrix}$$

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - (cef + bdi + afh)$$

$$\begin{vmatrix} 1 & a & b \\ a & 1 & c \\ b & c & 1 \end{vmatrix} = 1 + 2abc - (a^2 + b^2 + c^2)$$

$$1 + 2abc - (a^2 + b^2 + c^2) = 0$$

$$a = 0.9 \quad b = 0.8$$

$$c = 0.4585$$

# 1 Part 1: Overview of Regression and

## 2 Supervised Learning

- 3 Text references: Chapters 1 and 2 in ISL, §14.4 in Ruppert.

4 **The basic problem statement:**

- 5
- 
- 6 For each of  $n$  cases, there is a measured **response variable** or **output**  
7 or **dependent variable**. Denote this  $Y_i$  for the  $i^{th}$  object
- 8 Along with this, there are  $p$  additional measurements for each ob-  
9 ject, called the **predictors** or **independent variables** or **covariates** or  
10 **features** or .... Denote these  $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip})$
- 11 The goal is to construct a **model** which predicts  $Y_i$  given  $\mathbf{x}_i$ .

12

---

- 1 **Example:** "Modelling corporate bond rating with the use of market-based indicators" (Novotna, 2012)
- 3 "The paper presents an estimation of corporate bond rating models
- 4 based on both financial and market company indicators."
- 5 **The response (y):** Bond rating (Aaa, Baa, etc.) *Categorical response*
- 6 **The predictors (x):** Total assets, Equity to total assets ratio, Return on assets, Return on equity, etc.
- 8 "The models being derived allow classifying bond rating of companies with relatively high accuracy, even when a limited set of input variables is considered. The practical use of models lies in the area of management decision process and managing credit risk."

- 1 **Example:** "Data Mining Techniques for the detection of fraudulent
- 2 financial statements" (Kirkos, et al. 2007)
  
- 3 **The response (y):** Whether or not the firm submitted fraudulent fi-  
4 nancial statements *response is binary*
  
- 5 **The predictors (x):** Many, including: Total debt, Debt to equity ratio,
- 6 Gross Profit to Total Assets ratio, etc.
  
- 7 "The results obtained from the experiments agree with prior research
- 8 results indicating that published financial statement data contains
- 9 falsification indicators."

- Example:** "Parametric option pricing: A divide-and-conquer approach"
  - (Gradojevic, 2011)
- "The observed biases [in the Black-Scholes model] have motivated the development of more complex parametric as well as non-parametric option pricing models."
- The response ( $y$ ):** Option price
- The predictors ( $x$ ):** Risk-free interest rate, Implied volatility, Implied dividend yield, Time to maturity, Strike price

- 1 **Example:** “Nonparametric option pricing under shape restrictions”
- 2 (Ait-Sahalia and Duarte, 2003)
  
- 3 “The motivation for our empirical work is the theory-imposed re-
- 4 striction that the price of a call option must be a decreasing and con-
- 5 vex function of the options strike price.”
  
- 6 **The response (y):** Option price
  
- 7 **The predictors (x):** Risk-free interest rate, Implied volatility, Implied
- 8 dividend yield, Time to maturity, Strike price

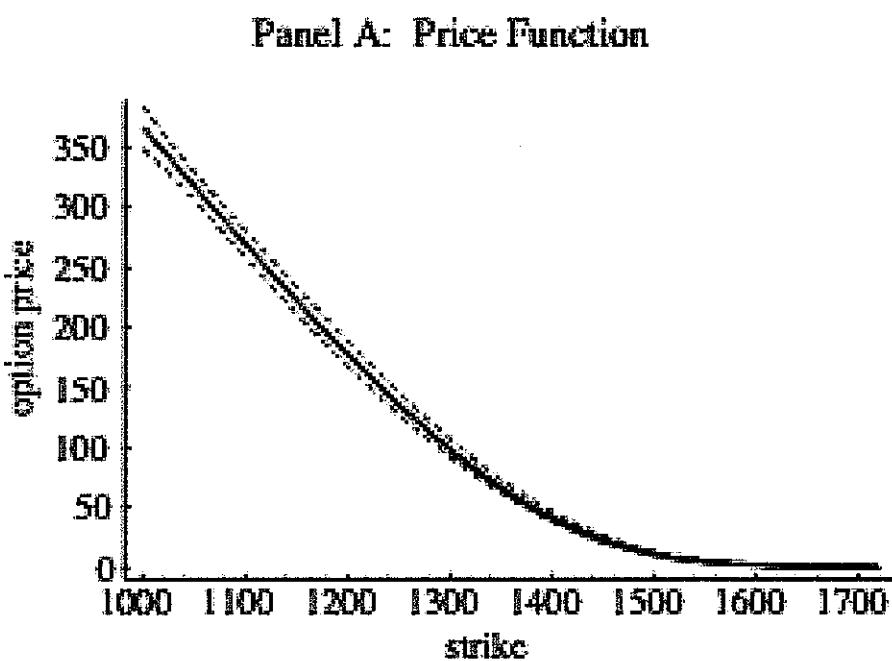


Figure 1: Part of Figure 2 in Ait-Sahalia and Duarte (2003). The solid line is the true relationship, while the 95% confidence band is shown as the dashed lines.

- 1 **Example:** “A Comprehensive Look at Financial Volatility Prediction
- 2 by Economic Variables” (Christiansen, et al. 2012)

3 “Financial volatility is a crucial input for risk management,  
4 asset pricing, and portfolio management and it may exert im-  
5 portant repercussions on the economy as a whole as evinced  
6 forcefully by the recent financial crisis. It is therefore of pri-  
7 mary interest to learn more about the economic drivers of  
8 volatility in financial markets. In this paper, we empirically  
9 investigate whether information in a broad set of economic  
10 variables measuring financial and macro conditions is help-  
11 ful in predicting future volatility.”

- 12 **The response ( $y$ ):** Realized volatility

- 13 **The predictors ( $x$ ):** Many, including:

- 14 1. Equity Market Variables and Risk Factors
- 15 2. Interest Rates, Spreads, and Bond Market Factors
- 16 3. Foreign Exchange Variables and Risk Factors
- 17 4. Liquidity and Credit Risk Variables
- 18 5. Macroeconomic Variables

- 1 Example:** “Predict short term movements in stock prices using news and sentiment data provided by RavenPack”
- 2**

**3** [www.kaggle.com/c/battlefin-s-big-data-combine-forecasting-challenge/](http://www.kaggle.com/c/battlefin-s-big-data-combine-forecasting-challenge/)

**4** “RavenPack analyzes novel and relevant stories published  
**5** by major news sources to look for key scheduled and unexpected  
**6** geopolitical, macro-economic and corporate events,  
**7** topics and opinions that indicate changes in market sentiment.  
**8** Sampled news sources represent the most reliable and  
**9** authoritative publishers of business and financial news.  
**10** RavenPack continuously analyzes relevant information . . . to  
**11** produce real time news sentiment scores and events from en-  
**12** tities across multiple asset classes, including currencies, com-  
**13** modities, organizations, companies, sectors, and industries.”

**14** **The response ( $y$ ):** Percent change in equity value

**15** **The predictors ( $x$ ):** “Sentiment scores” derived from text analysis.

**16** See [www.ravenpack.com](http://www.ravenpack.com) for more information

## **Types of Models**

- 1
- 2 Here we will go through a number of different model types, some are
- 3 standard in “Statistics,” some are standard in “Machine Learning,”
- 4 but they all share a common goal: Trying to predict the response,
- 5 given the predictors.
- 6
- 7 And they share a common framework of underlying parameters,
- 8 model (distribution) assumptions, concern with errors, concern with
- 9 overfitting, and so forth.
- 10 The models that we will consider will range from simple to complex.

- Examples include **linear regression models**:

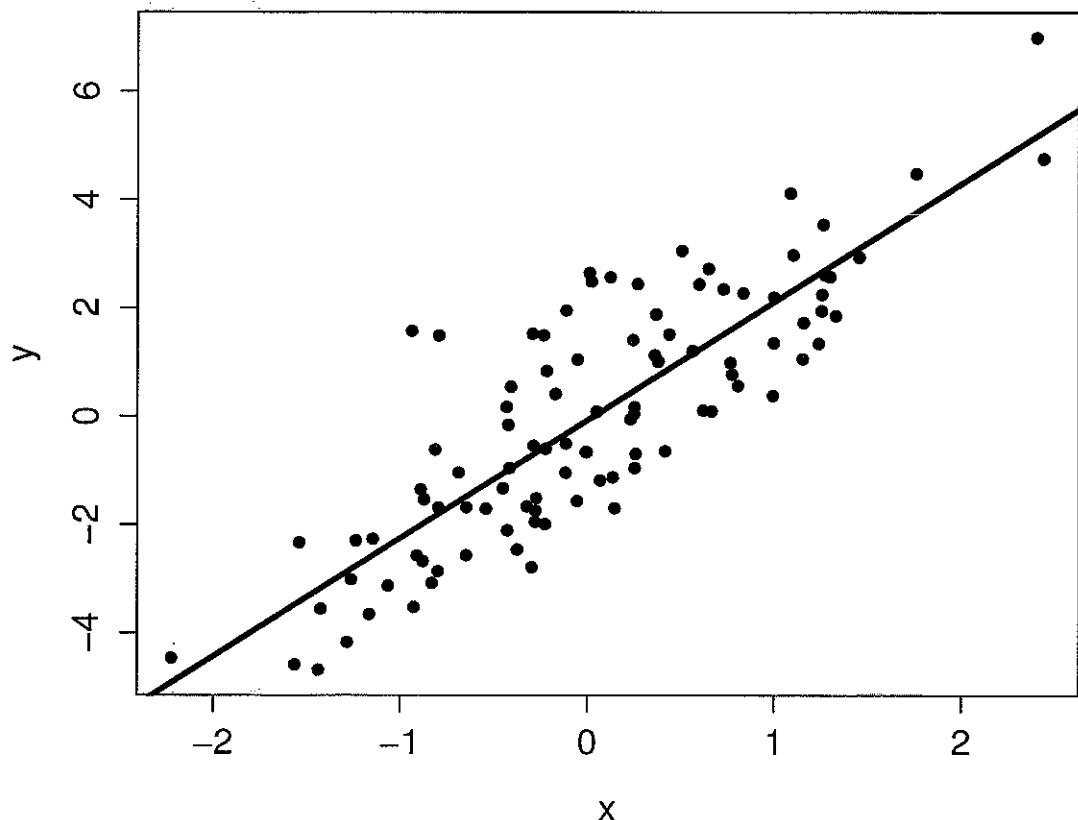


Figure 2: A simple linear regression model.

Nonparametric regression models:

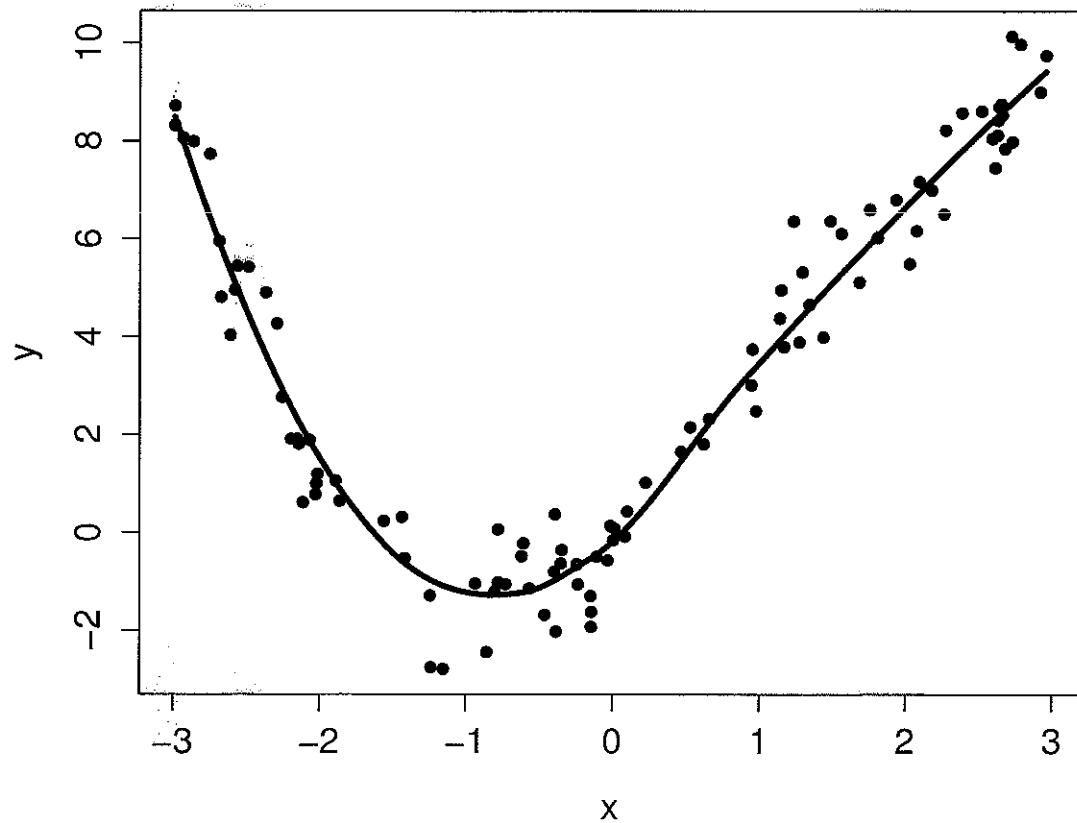


Figure 3: A nonparametric regression model.

## Tree-based models:

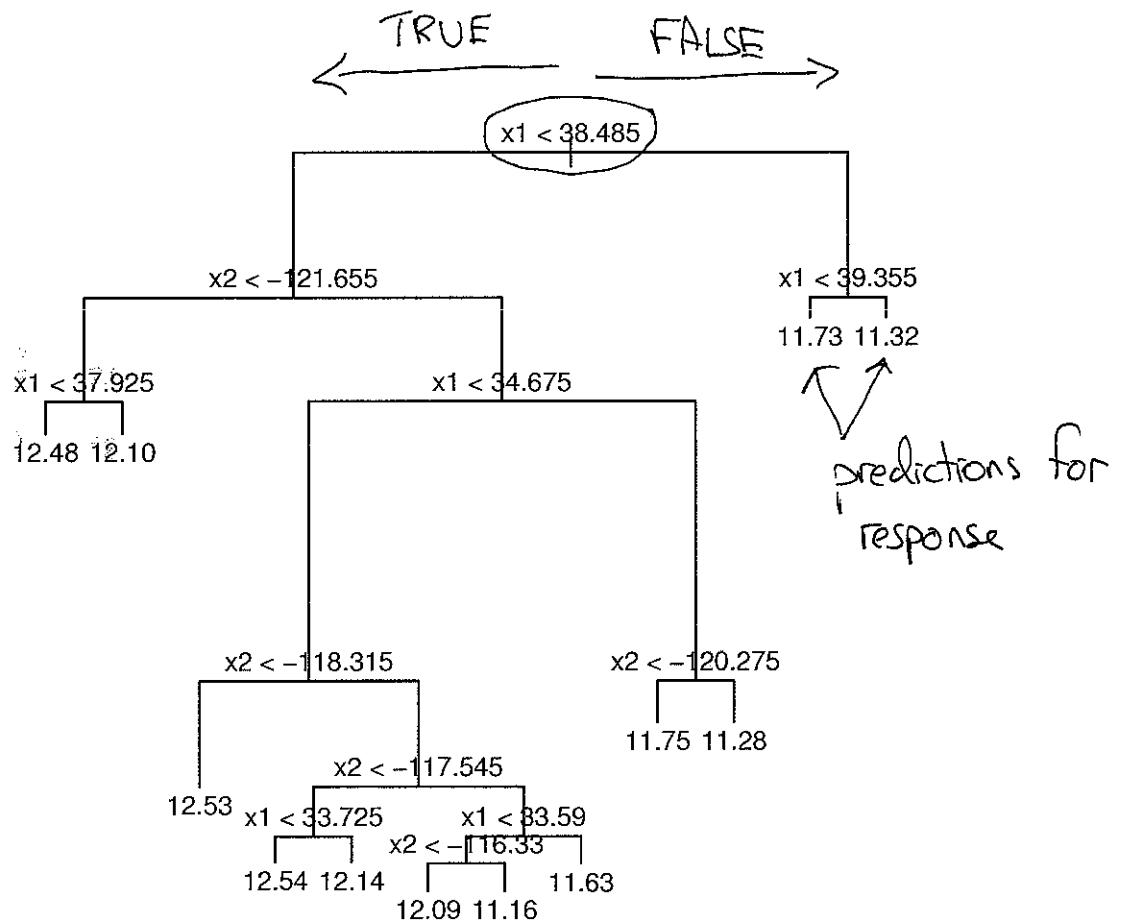


Figure 4: A regression tree model with two predictors,  $x_1$  and  $x_2$ .

## Classification via SVM:

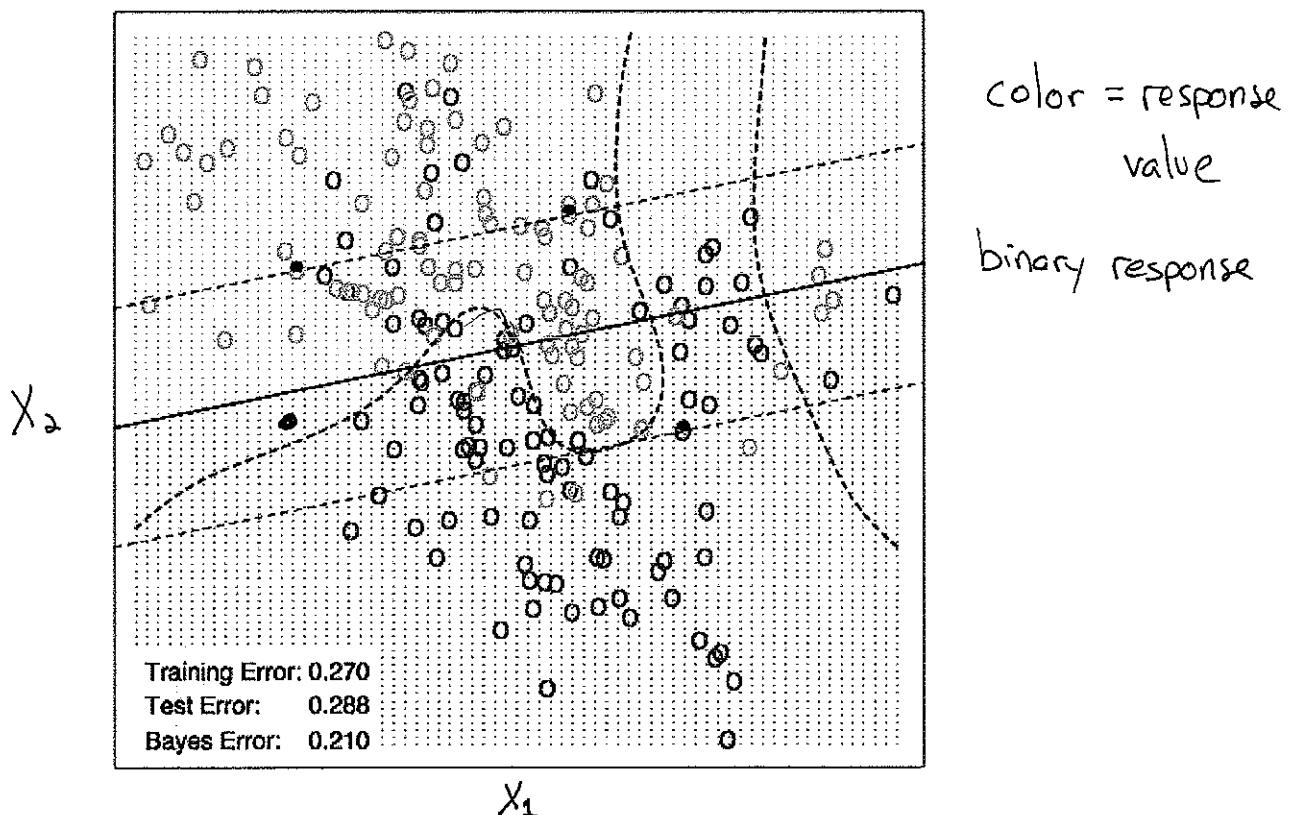


Figure 5: An illustration of the application of support vector machines (SVM). Circles of different colors have different values for the response. The axes represent the values for the two predictors. The black solid line and the dashed purple line are two classification rules “learned” from the data. Taken from Hastie, Tibshirani, and Friedman.

- 1 The nature and form of the model we choose for a particular situation
- 2 depends on many things:
  - 3 1. The **type** of the response: Is it continuous, binary, categorical?
  - 4 2. The types of the predictors
  - 5 3. The complexity of the relationship between the predictors and
  - 6 response: Is a linear model sufficient?
  - 7 4. The amount of data we have: The more complex a model be-
  - 8 comes, the more data we need to actually estimate the parame-
  - 9 ters
  - 10 5. How important are the predictions? Or are we just trying to get
  - 11 a rough understanding of the nature of the relationship between
  - 12 the variables?

1

---

## 2 The Key Questions

3 There are four **key questions** that we should consider for any pro-  
4 posed method for the regression/supervised learning problem:

- 5 1. What assumptions is the model making regarding the relation-  
6 ship between the response and the covariates?
- 7 2. How will we assess the validity of those assumptions? *residual analysis*
- 8 3. How can we be confident that the model that has been fit ex-  
9 tends outside of the training sample? (I.e., it neither overfits nor  
10 underfits to the sample.)
- 11 4. How do we make predictions with the model, and quantify the  
12 uncertainty in those predictions?

13 Note that this list does not include “How will the parameters of the  
14 model be estimated?” The answer is simple: maximum likelihood.  
15 We will not go into detail with that here. We will also not concern  
16 ourselves with the actual computational algorithms utilized to max-  
17 imize the likelihood. We will let R handle that.

- 1 Example: (From Section 14.4 in Ruppert, starting on page 381.)

- 2 Here, we seek to estimate the **forward-rate function**  $r(t)$  from empirical estimates based on the current prices of zero-coupon bonds.

- 4 From Equation (14.16) in Ruppert: The response in this case is

$$5 Y_i = -\frac{\log(P(T_i)) - \log(P(T_{i-1}))}{T_i - T_{i-1}} \quad (3.22) \text{ in Ruppert}$$

- 6 and our simple model assumes that

$$7 Y_i = r(T_i) + \epsilon_i$$

- 8 where the  $\epsilon_i$  are iid, normally distributed, with mean zero. Hence, in 9 this example we have a single covariate, time to maturity ( $T$ ).

- 10 The data are shown in Figure 6.

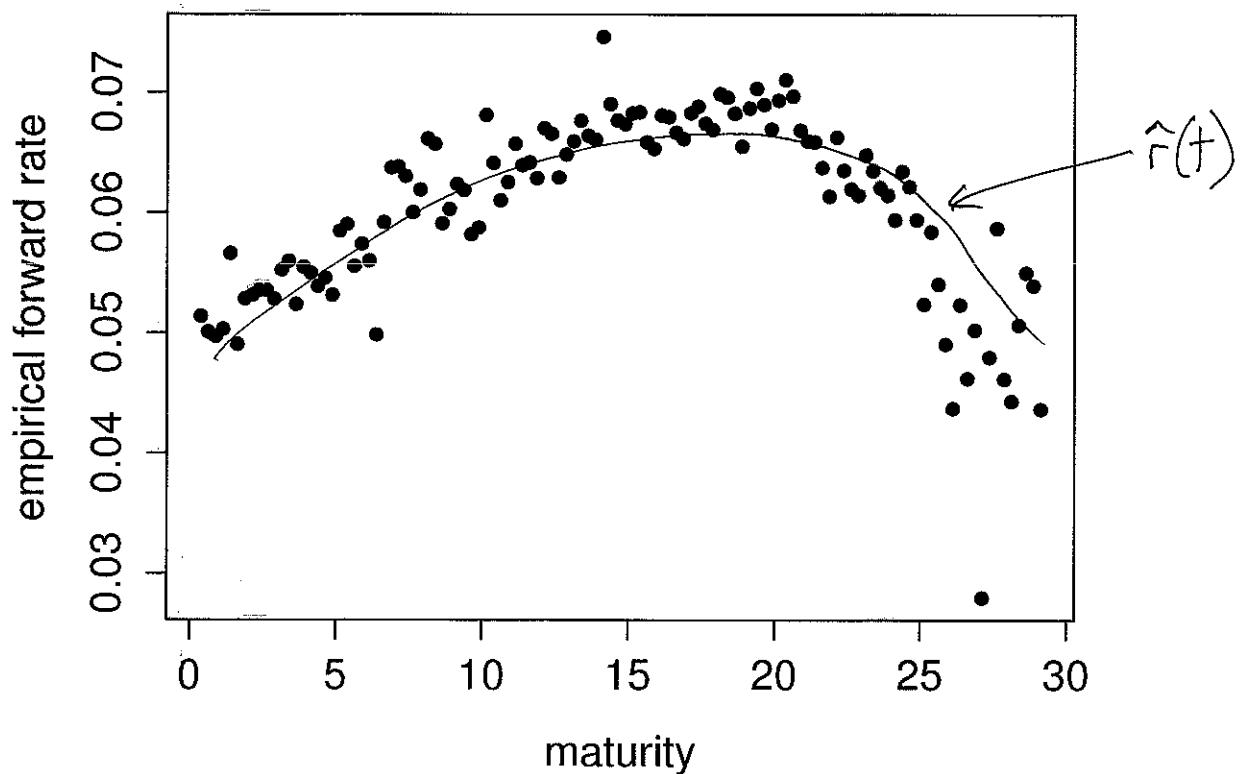


Figure 6: The response versus time for the estimation of the forward rate function.

- We seek to fit a model to this relationship.
- In particular, we are going to fit a parametric model of the form

$$r(t) = \sum_{j=0}^p \beta_j t^j. \quad \text{parameters to be estimated}$$

- In other words, we're fitting a  $p^{th}$  order polynomial.

- Figure 7 shows the **regression line** fit when  $p = 2$ . The regression line gives the model's estimate of  $E(Y|x)$  for all  $x$ .

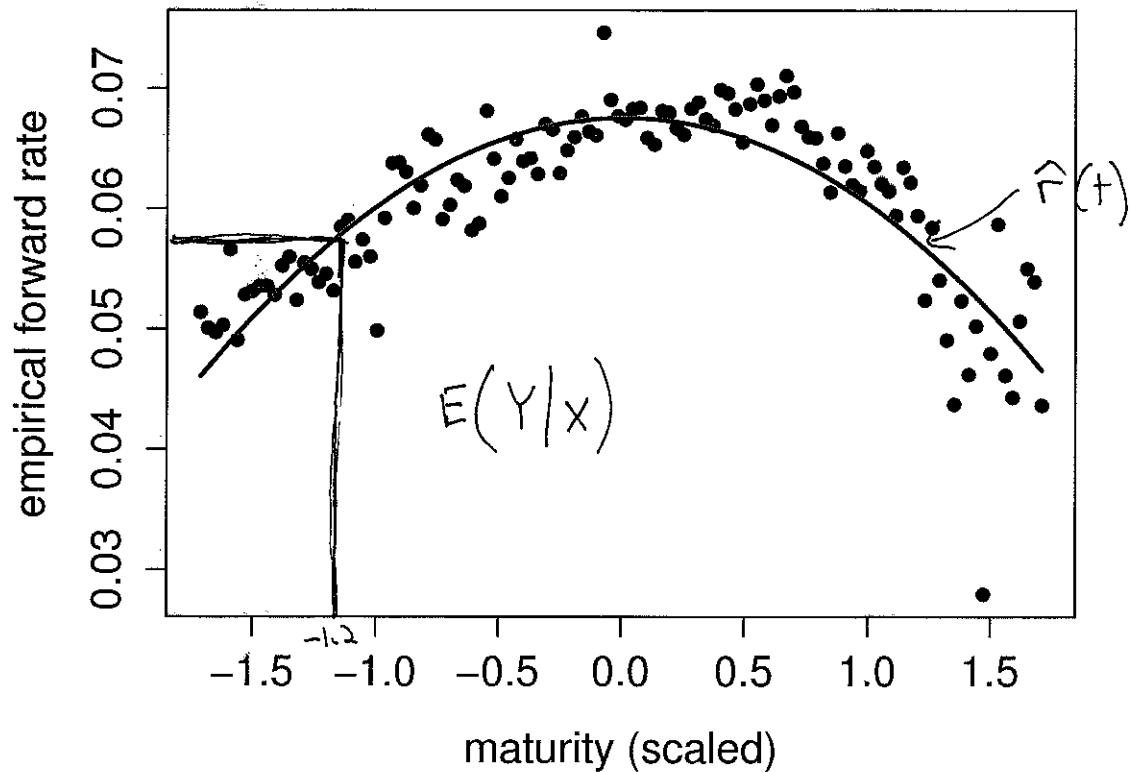


Figure 7: Fitting a quadratic model to the data.

## 1 Aside: Scaling Predictors

- 2 In the regression models that are fit in this example, we work with
- 3 the **scaled** predictor instead of the raw predictor. This is often a good
- 4 (even necessary) approach when the predictor could be raised to a
- 5 large power.
  
- 6 Unless otherwise clarified, “scaling” of a variable implies that it has
- 7 been transformed so that it has sample mean zero and sample vari-
- 8 ance one. In R this can be easily accomplished using `scale()`.
  
- 9 In our example, the largest maturity is close to 30 years. If this pre-
- 10 dictor value is raised to the fifth (or higher) powers, the result is a
- 11 number that is sufficiently large to possibly create numerical prob-
- 12 lems. Hence, we used scaled maturity in our models.
  
- 13 Since the scaling is a linear transformation, it does not change the
- 14 shape of the relationship between the response and predictor.

- 1 We will consider each of the four key questions briefly, in turn.

2

- 3 **Question One:** What assumptions is the model making regarding  
4 the relationship between the response and the covariates?

- 5 Regardless of its type, any regression model returns an estimate of  
6 **the distribution of the** response  $Y$  given a vector of covariates  $\mathbf{x}$ .

- 7 In our example, we could express the model as

8

$$Y_i = \sum_{j=0}^p \beta_j T_i^j + \epsilon_i$$

- 9 where the  $\epsilon_i$  are assumed to be iid, normally distributed, with mean  
10 zero and variance  $\sigma^2$ .

- 11 This is equivalent to stating the response  $Y_i$  is assumed be normally  
12 distributed with mean

13

$$\sum_{j=0}^p \beta_j T_i^j$$

- 14 and variance  $\sigma^2$ .

- 15 Note how the statements specify completely the distribution of the  
16 response, given the vector of covariates  $\mathbf{x}$  and the parameters.

2 **Question Two:** How will we assess the validity of the assumptions?

3 Although we know that the model is only an approximation to the  
4 true relationship between the covariates and response, we should  
5 test the model assumptions to the extent that is possible.

6 Often, these judgements are based on **residual analysis**.

7 The model's prediction of the expected value of the response, given  
8 the covariates  $x$ , is called the **fitted value** for that case. The standard  
9 notation is  $\hat{y}$ :

10 
$$\hat{y} = \widehat{E}(Y | x)$$



11 The **residual** is the difference between the fitted value and the actual  
12 response. These are denoted  $\hat{\epsilon}$ :

13 
$$\hat{\epsilon} = y - \hat{y}$$

Exercise: Where are the fitted values and residuals on Figure 8?

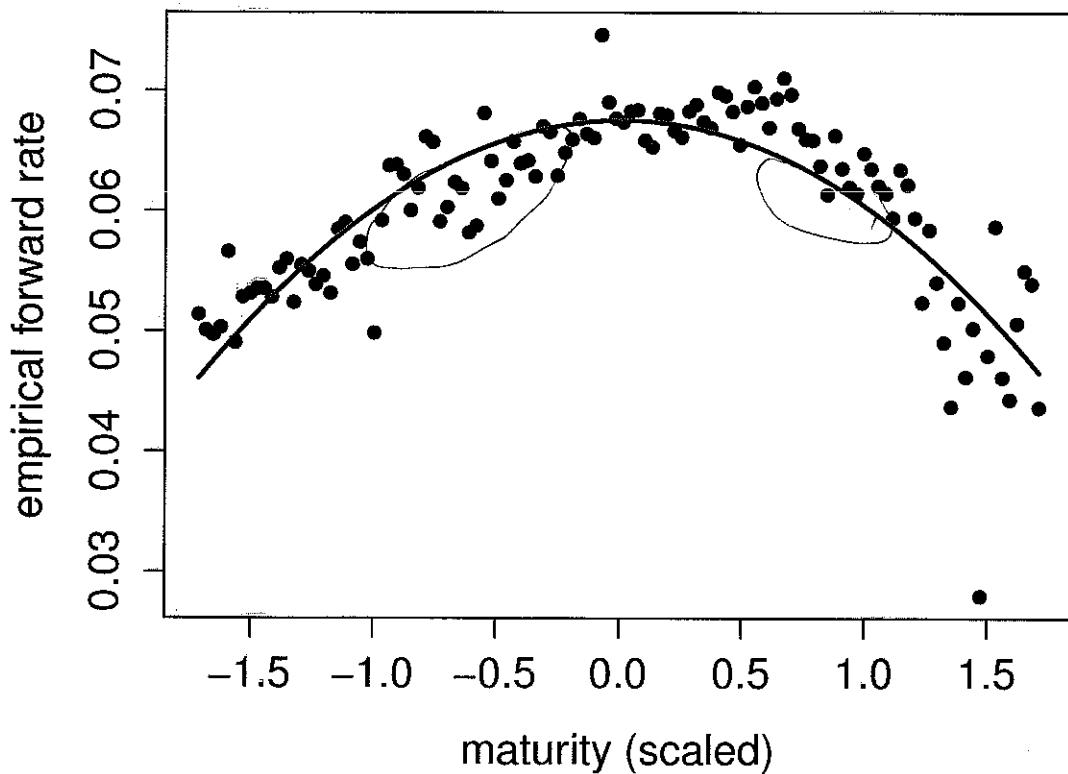


Figure 8: Fitting a quadratic model to the data.

- ${}_1$  The value of the residuals is that they can be thought of as approximations to the unobservable model errors  $\epsilon_i$ .
- ${}_2$
- ${}_3$  For example, a standard assumption is that the model error is “noise”
- ${}_4$  unrelated to the response and the covariates. Hence, a plot of  $\epsilon_i$  versus the covariates should reveal no pattern.
- ${}_5$
- ${}_6$  We can instead plot the residuals versus the covariates.

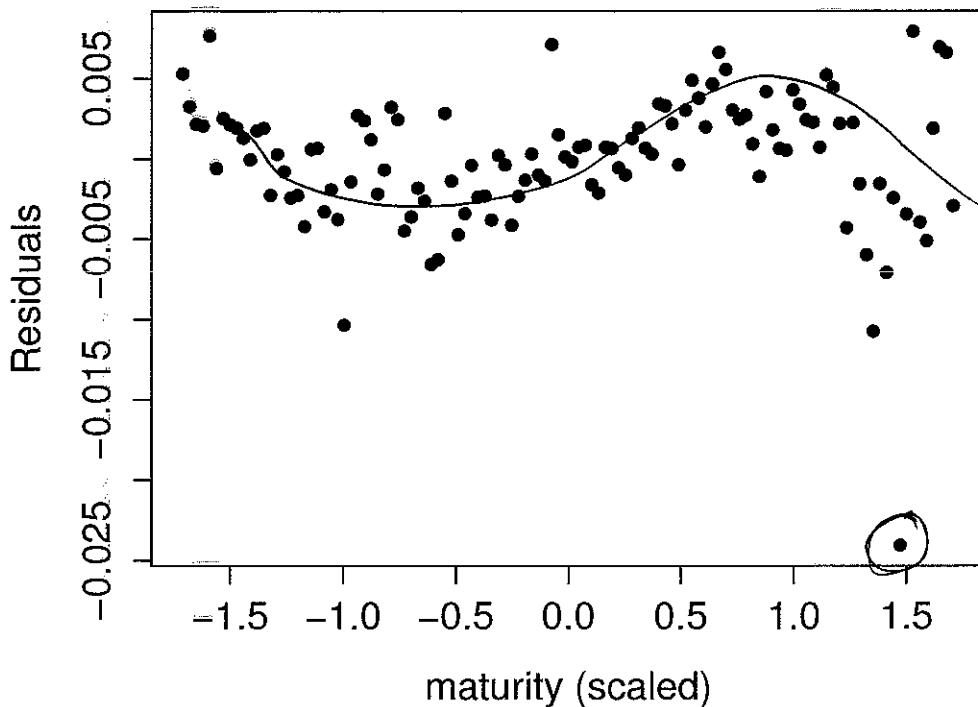


Figure 9: Plot of residuals versus maturity for the quadratic model.

- 1 Figure 9 plots the residuals versus maturity for our example. There
- 2 is a noticeable pattern, which creates concern regarding this model. If
- 3 the model error was unrelated to maturity, then there should be no
- 4 clear patterns in this plot. We will look at many plots of this type
- 5 when we assess models.
  
- 6 Also, there is one point for which the residual is quite extreme rela-
- 7 tive to the others. This should also create concern; such observations
- 8 can be **influential** on the fit of the model.

- 1
- 2 **Question Three:** How can we be confident that the model that has
- 3 been fit extends outside of the training sample?
- 4 When fitting a distribution to a sample of data, there is always a con-
- 5 cern of **overfitting** to the sample, and hence fitting to **artifacts** in the
- 6 sample which are not real features of the population. Fitting regres-
- 7 sion models is no different.
- 8 Every model we propose will include an adjustable **tuning parameter**,
- 9 which controls the complexity of the model.
- 10 In this context, higher complexity implies more parameters to esti-
- 11 mate, but more flexibility in the form of the relationship between the
- 12 response and the covariates.
- 13 In our example, with response  $Y$  and a single covariate  $t$ ,

14

$$Y_i = \sum_{j=0}^p \beta_j t_i^j + \epsilon_i,$$

- 15 the choice of  $p$  controls the complexity.

- 1 In **nonparametric regression models**, a single parametric form for
- 2 the model is not assumed, but there is still a tuning parameter that
- 3 controls the smoothness of the regression function.
  
- 4 The one fact we have to keep in mind: **Increasing complexity will**
- 5 **necessarily increase the likelihood.**
  
- 6 If we make model choices just by seeking to maximize the likelihood,
- 7 we will overfit to the available data.

- 1 Again in our example, we can adjust the complexity of the model by
- 2 changing  $p$ . From our residual analysis, it seemed that the quadratic
- 3 model was not a good fit; will increasing the complexity help?
  
- 4 Figure 7 showed the case where  $p = 2$ , in that case it appeared that
- 5 we were underfitting to the data.
  
- 6 Figure 10 shows the fit when  $p = 4$  and  $p = 9$ . It is evident that as  $p$
- 7 increases, the regression line starts to fit more closely to the data.
  
- 8 In fact, if we chose  $p$  equal to  $n - 1$ , we could get the line to pass
- 9 through every one of the points.

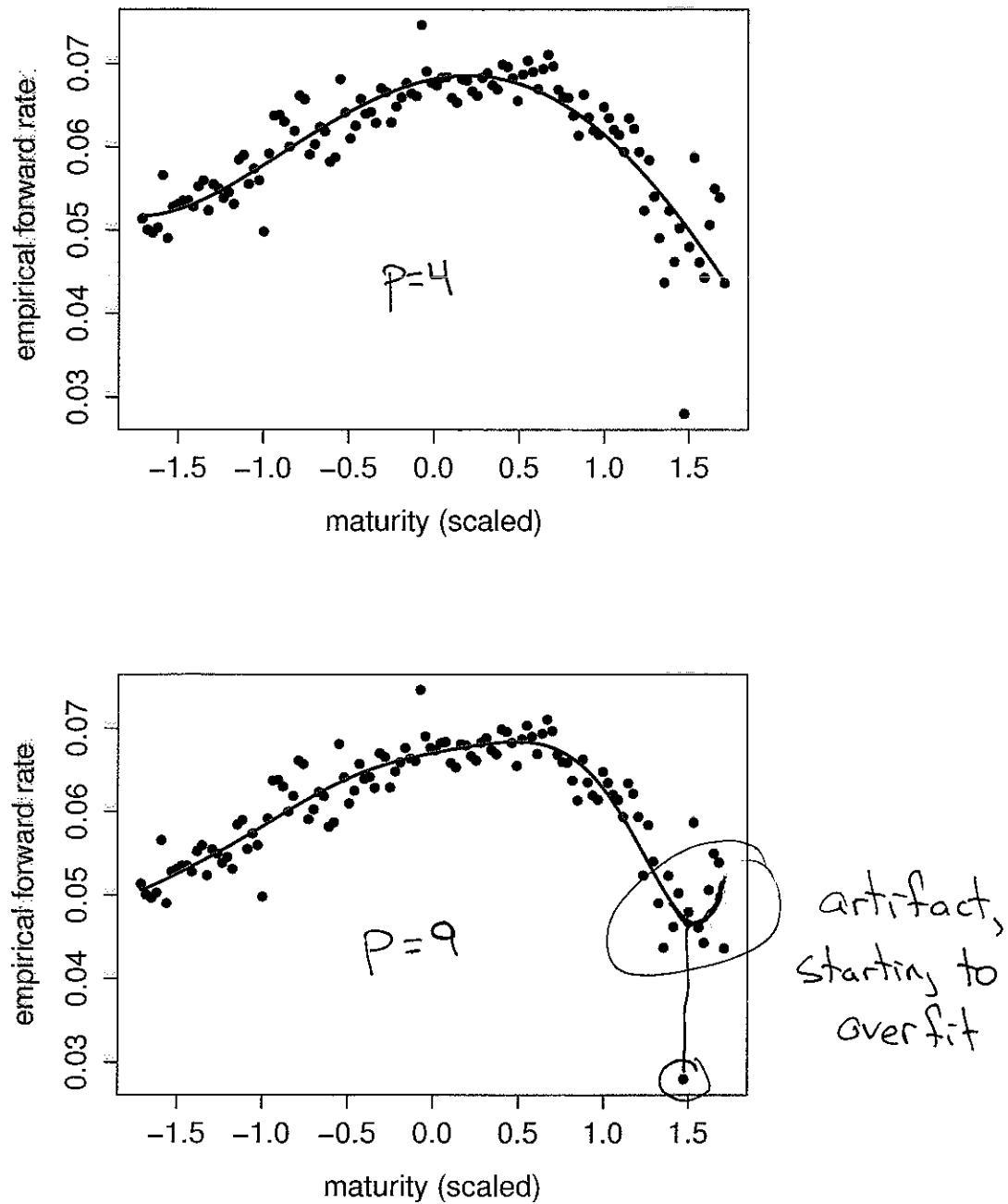


Figure 10: The cases where  $p = 4$  (top) and  $p = 9$  (bottom).

- 1 We can use AIC to compare models with different numbers of pa-
- 2 rameters. Recall that

3 
$$\text{AIC} = -2 \times \log \text{likelihood} + 2k$$

- 4 where  $k$  equals the number of parameters being estimated. We prefer  
5 models with small AIC. Here,  $k = p + 2$ . Figure 11 shows how AIC  
6 changes as a function of  $p$ . Based on this, it appears that  $p = 7$  is the  
7 best choice.

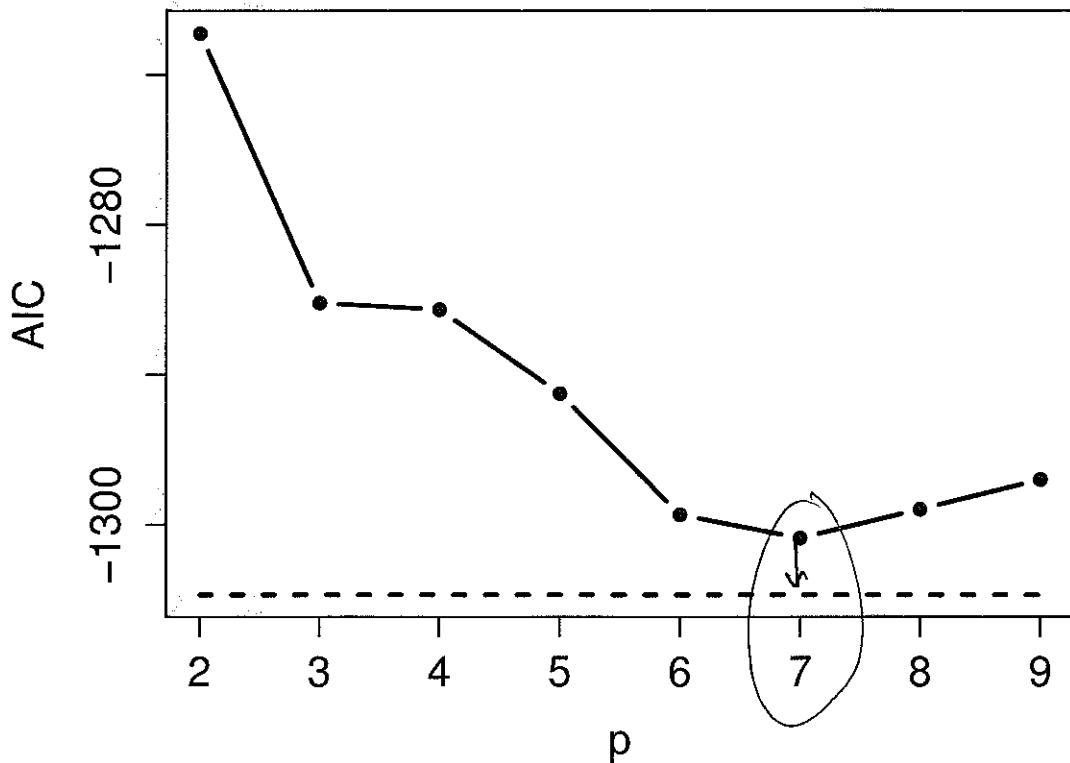


Figure 11: The AIC plot.

- 1 But, in fact, one could search over a wider class of models: Maybe,  
2 for instance, the term  $x^7$  is important to include, but  $x^3$  is unimpor-  
3 tant. R makes such a search for the best model easy to perform, and  
4 the result is that the regression line

5 
$$r(t) = \beta_0 + \beta_1 t + \beta_4 t^4 + \beta_5 t^5 + \beta_6 t^6 + \beta_7 t^7$$

- 6 has lowest AIC. The value of AIC for this model is  $-1304.65$ , and is  
7 shown as the dashed line in Figure 11. The regression line for this  
8 model is shown in Figure 12.

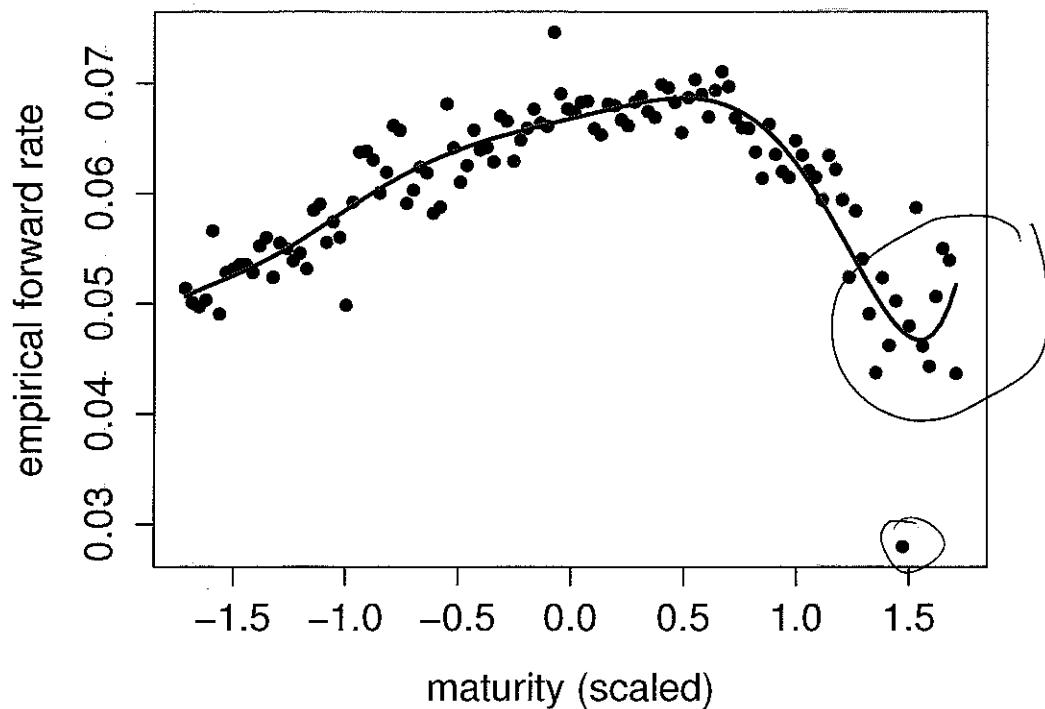


Figure 12: The “optimal” case.

- 1 **Exercise:** Are you concerned about the fit of the model shown in
- 2 Figure 12? Explain.

3 Yes. There seems to be an artifact being  
4 fit to on the upper end.

5  
6 This behavior is common when using higher  
7 order polynomials

8  
9 Could do better using a nonparametric  
10 approach

11

12

13

14

15

16

17

18

19

20

21

- 1
- 2 **Question Four:** How do we make predictions with the model, and  
3 quantify the uncertainty in those predictions?

- 4 Many (but not all) of the times that we construct a regression model,  
5 the objective is to make predictions for new observations.

- 6 In other words, if presented with a new vector of covariates  $x$ , what  
7 is the prediction for the response  $Y$ ?

- 8 In the context of the simple linear regression model, if our value for  
9 the new predictor is  $x^*$ , then we would start by finding

10

$$\hat{y}^* = \hat{\beta}_0 + \hat{\beta}_1 x^*$$

- 11 We have plugged in this new value for  $x$  into the line that has been  
12 fit. This is our best estimator for  $y^*$ , the true response for this object.

- 13 One should not be satisfied with only this, however: We should  
14 quantify the amount of error in this prediction.

The variance in the error in the prediction is

$$V(\underbrace{\hat{y}^* - y^*}_{2}) = \sigma^2 + \sigma^2 \left[ \frac{1}{n} + \frac{(x^* - \bar{x})^2}{(n-1)s_x^2} \right]$$

<sup>3</sup> **Exercise:** Explain how the above decomposes the error in the prediction into two key sources.

$$\begin{aligned}
 5 \quad V(\hat{y}^* - y^*) &= V(\hat{y}^* - E(y^*) + E(y^*) - y^*) \\
 6 \quad &= V(\hat{y}^* - E(y^*)) + V(E(y^*) - y^*) \\
 7 \quad &\quad + 2 \operatorname{Cov}(\hat{y}^* - E(y^*), E(y^*) - y^*) \\
 8 \quad &\quad \underbrace{E_1, E_2, \dots, E_n}_{-E^*} \\
 9 \quad &= V(\hat{y}^* - E(y^*)) + V(E(y^*) - y^*) \\
 10 \quad &\quad \text{due to error in } \underbrace{E_0, E_1}_{\dots} \\
 11 \quad &= \sigma^2 \left[ \frac{1}{n} + \frac{(x^* - \bar{x})^2}{(n-1)S_x^2} \right] + \sigma^2 \\
 12 \quad &\quad \underbrace{\left[ \frac{1}{n} + \frac{(x^* - \bar{x})^2}{(n-1)S_x^2} \right]}_{\text{var. due to error in the } \cancel{\text{fitted model}}} \quad \underbrace{\sigma^2}_{\text{model error } (\epsilon \text{ error})}
 \end{aligned}$$

15 Hence the standard error in the prediction is

$$\text{SE}(\hat{y}^*) = \sqrt{\sigma^2 + \sigma^2 \left[ \frac{1}{n} + \frac{(x^* - \bar{x})^2}{(n-1)s_x^2} \right]}$$

<sup>17</sup> Of course, in practice  $\sigma^2$  will be replaced with  $\hat{\sigma}^2$ .

- <sub>1</sub> **Exercise:** What happens to the standard error in the prediction as  $n$   
<sub>2</sub> increases?

<sub>3</sub> It going to converge to 0.

<sub>4</sub>  
<sub>5</sub> No matter how large  $n$  is, there is  
<sub>6</sub> still is still error in the predictions.

<sub>7</sub> Must remember to take that into account.

<sub>8</sub>

<sub>9</sub>

<sub>10</sub>

<sub>11</sub>

<sub>12</sub>

<sub>13</sub>

<sub>14</sub>

<sub>15</sub>

- <sub>16</sub> A  $100(1 - \alpha)\%$  **prediction interval for the future response** can be  
<sub>17</sub> written as

$$\hat{y}^* \pm t_{\alpha/2, n-2} \text{SE}(\hat{y}^*)$$

<sub>18</sub> predict( )

- <sub>19</sub> In practice, we will rely on R to construct these intervals.

- Figure 13 shows a simple example of how the prediction interval will appear, as  $x^*$  ranges over the extent of the predictor space.

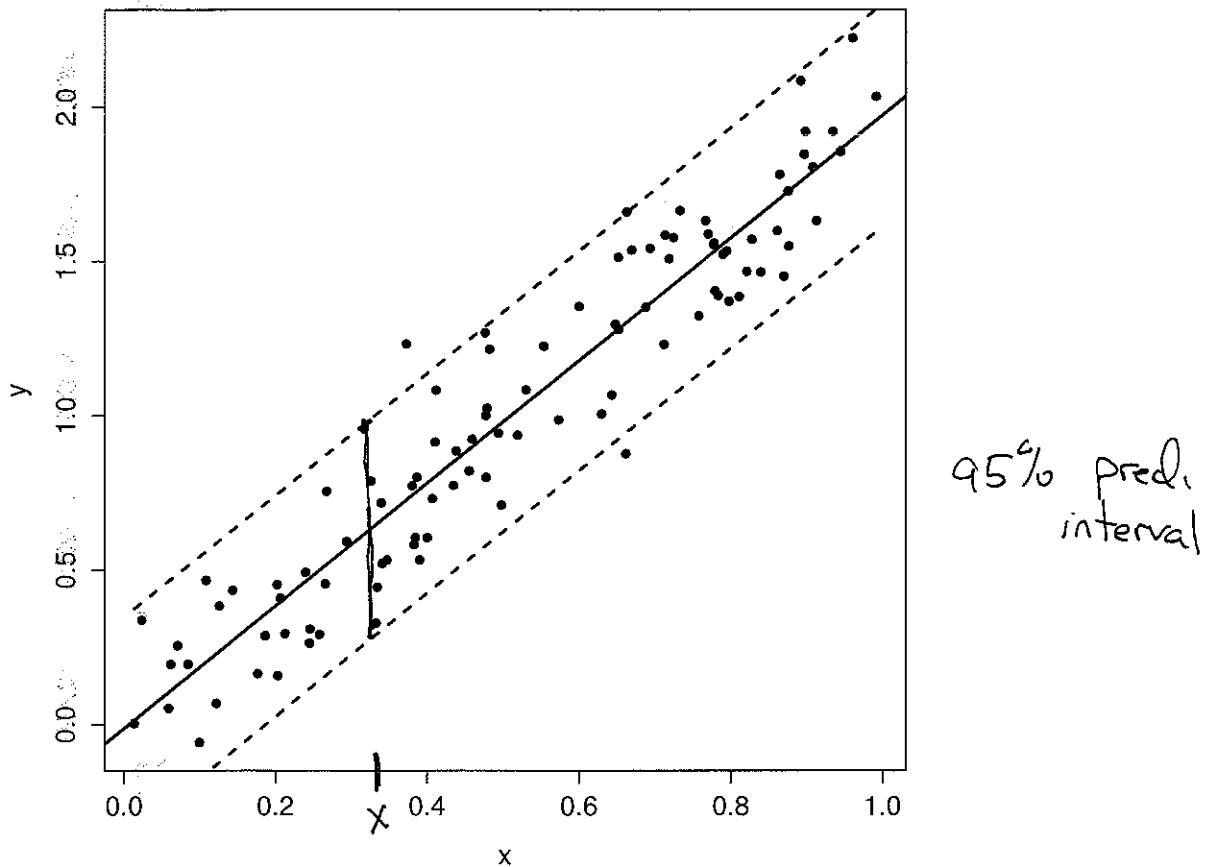


Figure 13: An example showing a regression line (solid line), along with the prediction bands (dashed lines).

## Part 2: Multiple Linear Regression

- <sup>2</sup> Text references: §3.2 in ISL, Chapters 12 and 13 in Ruppert.
- <sup>3</sup> In this part, we will extend from the simple linear regression model
- <sup>4</sup> to **multiple regression**.
- <sup>5</sup> These models comprise the familiar linear models that take the form

$$Y = \beta_0 + \sum_{i=1}^p \beta_i x_i + \epsilon$$

- <sup>6</sup>
- <sup>7</sup> where  $\epsilon$  are assumed to be i.i.d. normal errors with mean zero and
- <sup>8</sup> variance  $\sigma^2$ .
- <sup>9</sup> We will go through the fundamental steps of fitting such a model via
- <sup>10</sup> an example.

## 2 Example: Factor Models

- 3 In our introduction to simple linear regression, we considered a sim-  
4 ple linear regression model relating the excess return for GOOG to  
5 the excess market return; models of this form are an important com-  
6 ponent of Capital Asset Pricing Model (CAPM).
- 7 The term **Factor Model** is used to indicate any linear model for excess ←  
8 return on an equity. The predictors in the model are the **factors**. response
- 9 Quoting Ruppert (page 453), examples of factors include
- 10 1. returns on the market portfolio;
- 11 2. growth rate of the GDP;
- 12 3. interest rate on short term Treasury bills or changes in this rate;
- 13 4. inflation rate or changes in this rate;
- 14 5. interest rate spreads;
- 15 6. return on some portfolio of stocks;
- 16 7. the difference between the returns on two portfolios.

- 1 The CAPM only includes the factor for excess market return.
- 2 The **Fama-French Three Factor Model** adds two factors.
- 3 The first is **small minus big (SMB)**, the difference in returns between portfolios of small and large stocks.
- 4 The second is **high minus low (HML)**, the difference in returns between portfolios of high and low book-to-market stocks.
- 5 See page 456 in Ruppert for more details on these factors, and motivation on their inclusion in the model.

- 1 Kenneth French maintains a web site with data on the three factors
- 2 [http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data\\_library.html](http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html)
- 3 I wrote a simple R function that will allow for easy access to the daily factor data, going back to 2000.
- 4
- 5 This function, named `getFamaFrench()`, is located at
- 6 <http://www.stat.cmu.edu/~cschafer/MSCF/getFamaFrench.txt>
- 7 The syntax is much like the functions in `quantmod`:
- 8 

```
> ffhold = getFamaFrench(from="2012-1-1",
  to="2012-6-30")
```
- 9
- 10 (Note that data is typically unavailable for the most recent months.)

- 11 The first few rows of the data returned are as follows:

12

	Date	Mkt.RF	SMB	HML	RF	
13	3020	2012-01-03	1.61	-0.09	0.49	0
14	3021	2012-01-04	-0.04	-0.60	0.01	0
15	3022	2012-01-05	0.31	0.22	-0.02	0
16	3023	2012-01-06	-0.28	0.00	-0.27	0
17	3024	2012-01-09	0.28	0.20	-0.13	0
18	3025	2012-01-10	1.04	0.43	0.29	0
19	...					

13 3020 2012-01-03 1.61 -0.09 0.49 0  
14 3021 2012-01-04 -0.04 -0.60 0.01 0  
15 3022 2012-01-05 0.31 0.22 -0.02 0  
16 3023 2012-01-06 -0.28 0.00 -0.27 0  
17 3024 2012-01-09 0.28 0.20 -0.13 0  
18 3025 2012-01-10 1.04 0.43 0.29 0  
19 ...

*excess returns*

- 1 Let's get the data for PNC for the same time period
- 2 > PNC = getSymbols("PNC", from="2012-1-1",  
3 to="2012-6-30", auto.assign=F)
  
- 4 Now find the excess returns for PNC. Note that the Fama-French  
5 data file includes the risk free rate, with name RF:
- 6 > ffhold\$PNCexret = 100\*dailyReturn(PNC) - ffhold\$RF
  
- 7 Note that this command added a new variable (column) named PNCexret  
8 to the **data frame** ffhold.

1 The three factor model we fit is

2  $PNC.Ex.Ret. = \beta_0 + \beta_1 Mkt.Ex.Ret. + \beta_2 SMB + \beta_3 HML + \epsilon$

3 The R command for fitting a linear regression is `lm()`:

4 > `ff3modPNC = lm(PNCexret ~ Mkt.RF + SMB + HML,`  
5 `data=ffhold)`

6 Note the use of the `data` argument to specify the data frame from  
7 which the variables are taken.

8 The key information from the fit of the model is stored in the object  
9 `ff3modPNC`. In particular, note that

10 > `attributes(ff3modPNC)`  
11 `$names`  
12 `[1] "coefficients" "residuals" "effects" "rank"`  
13 `[5] "fitted.values" "assign" "qr" "df.residual"`  
14 `[9] "xlevels" "call" "terms" "model"`

15 For example, you can obtain the residuals from

16 `as.numeric(ff3modPNC$residuals)`.<sup>1</sup>

---

<sup>1</sup>It is necessary to transform `ff3modPNC$residuals` using `as.numeric()` because it inherits the time series format from the original data.

## Diagnostic Plots

- 2 Figure 1 shows three key diagnostic plots:
  - 3 1. Plot of residuals versus fitted values.
  - 4 2. Normal probability plot.
  - 5 3. Plot of residuals versus time.
- 6 Below are the R commands to create this figure.

```
7 par(mfrow=c(2, 2))

8

9 # Plot of residuals versus fitted values
10 plot(as.numeric(ff3modPNC$fit), as.numeric(ff3modPNC$resid),
11       pch=16, xlab="Fitted Values", ylab="Residuals",
12       cex.axis=1.3, cex.lab=1.3)
13
14 # Normal probability plot
15 qqnorm(as.numeric(ff3modPNC$resid), cex.axis=1.3, cex.lab=1.3,
16        pch=16, main="")
17 qqline(as.numeric(ff3modPNC$resid))

18

19 # Plot of residuals versus time
20 plot(ff3modPNC$resid, xlab="Time", ylab="Residuals", cex.axis=1.3,
21       cex.lab=1.3, pch=16, main="")
```

*Plotting Symbol* *changes font sizes*

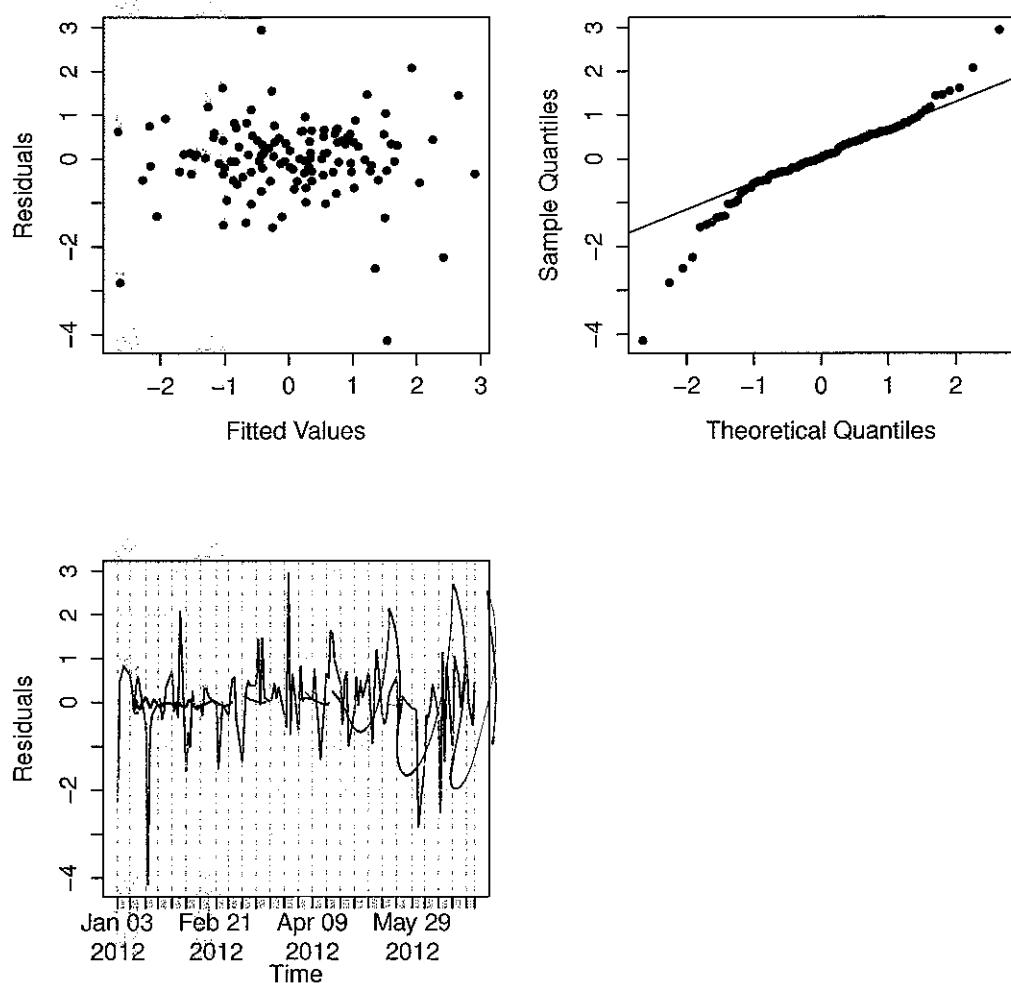


Figure 1: Diagnostic plots for the three factor model for PNC.

Exercise: Comment on the quality of the fit.

---

---

---

---

---

---

- 1 Use the `summary()` function to see the parameter estimates and
- 2 their standard errors:

```

3 > summary(ffc3modPNC)
4
5 Call:
6 lm(formula = PNCexret ~ Mkt.RF + SMB + HML, data = ffhold)
7
8 Residuals:
9   Min     1Q  Median     3Q    Max
10 -4.1476 -0.3383  0.0206  0.4908  2.9468
11
12 Coefficients:
13             Estimate Std. Error t value Pr(>|t|)    
14 (Intercept) -0.02886   0.08101  -0.356   0.722    
15 Mkt.RF       1.19368   0.10286  11.605 < 2e-16 ***
16 SMB          0.14936   0.21362   0.699   0.486    
17 HML          1.09569   0.23056   4.752 5.59e-06 ***
18 ---
19 Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
20
21 Residual standard error: 0.9023 on 121 degrees of freedom
22 Multiple R-squared: 0.6124, Adjusted R-squared: 0.6028
23 F-statistic: 63.72 on 3 and 121 DF,  p-value: < 2.2e-16
24

```

$$t\text{-value} = \frac{\hat{\beta}_i}{\text{SE}(\hat{\beta}_i)}$$

$$\Pr(|t|) = p\text{-value}$$

for  $H_0: \beta_i = 0$

vs.  $H_1: \beta_i \neq 0$

- 1 **Exercise:** There is strong evidence that  $\beta_3$ , the coefficient for HML,  
 2 is larger than zero. How did I reach this conclusion, and what is the  
 3 interpretation?

4 If I construct a 95% CI for  $\beta_3$

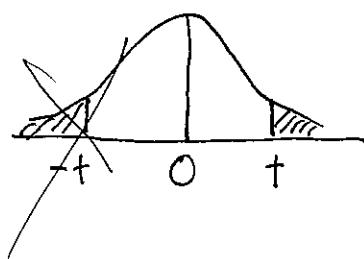
$$5 \hat{\beta}_3 \pm (2) SE(\hat{\beta}_3)$$

$$6 \hat{\beta}_3 \approx 1.09569 \pm (2)(0.23056)$$

7 Lower point of interval is clearly  
 8 above 0, ie. the entire interval  
 9 is above 0

10 Also note that p-value for  $H_0: \beta_3 = 0$  vs.  
 11  $H_1: \beta_3 \neq 0$  is very small  $5.59 \times 10^{-6}$

12 So, p-value for  $H_0: \beta_3 = 0$  vs.  $H_1: \beta_3 > 0$   
 13 is  $(5.59 \times 10^{-6})/2$



14 ~~2.5%~~

$$15 \ln(Y \sim X_1 + X_2 + X_3 - 1)$$

- 1 Another model was fit, but with only HML as a predictor, and the
- 2 result is shown below.

```
3 Call:  
4 lm(formula = PNCexret ~ HML, data = ffhold)  
5  
6 Residuals:  
7   Min     1Q  Median     3Q    Max  
8 -5.7908 -0.7914 -0.0228  0.8704  3.9148  
9  
10 Coefficients:  
11             Estimate Std. Error t value Pr(>|t|)  
12 (Intercept)  0.04842   0.12696   0.381   0.704  
13 HML         0.65484   0.35827   1.828   0.070 .  
14 ---  
15 Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1  
16  
17 Residual standard error: 1.418 on 123 degrees of freedom  
18 Multiple R-squared: 0.02644,    Adjusted R-squared: 0.01853  
19 F-statistic: 3.341 on 1 and 123 DF,  p-value: 0.07
```

- 1 **Exercise:** The interesting result here is that the "significance" of HML
- 2 has dropped considerably. What is the cause of this?

3 The other predictors (excess market return  
4 and SMB) were "explaining" some  
5 of the variability in the response.

6  
7 By removing them, that variability is  
8 put into the RSS, hence  $\hat{\sigma}^2$  is also  
9 increasing, and  $SE(\hat{\beta}_{HML})$  is also  
10 increasing. Finally, the "t-value" ~~is~~  
11 decreases.

12  
13 Also,  $\hat{\beta}_{HML}$  is changing itself

14

15

16

17

18

1

---

## 2 The Coefficient of Determination, $R^2$

- 3 When working with multiple regression models of this type it is very
- 4 common to hear references to “the  $R^2$  for the model.”
- 5 This is also called the **coefficient of determination**.
- 6 The idea is simple:  $R^2$  equals the proportion of the sum of squared
- 7 response which is accounted for by the model that has been fit, rela-
- 8 tive to the model with no covariates.
- 9 In particular:

10

$$R^2 = 1 - \left[ \sum_{i=1}^n (y_i - \hat{y}_i)^2 \Big/ \sum_{i=1}^n (y_i - \bar{y})^2 \right]$$

- 11 Note that  $0 \leq R^2 \leq 1$ .
- 12 In the case of simple linear regression (a single predictor),  $R^2$  does
- 13 indeed equal the sample correlation ( $r$ ) squared.

- $R^2$  is overused. In particular, note that it is possible for  $R^2$  to be close to one, but the model be a terrible fit. And it is possible for a great-fitting model to have  $R^2$  close to zero. See Figure 2.

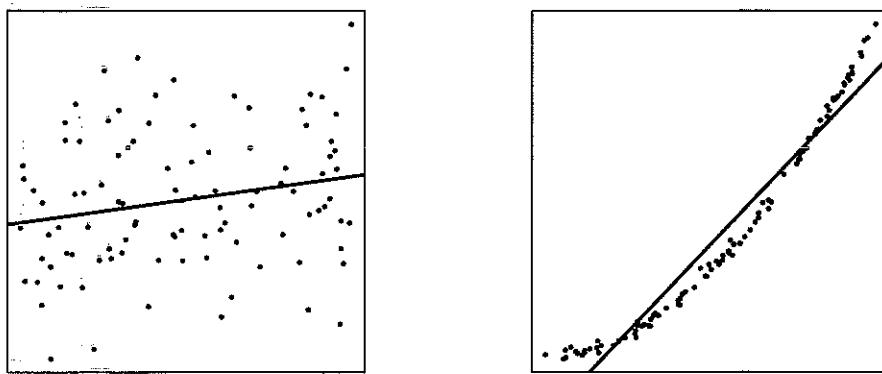


Figure 2: The scatter plot on the left has  $R^2 \approx 0.05$ , while that on the right has  $R^2 \approx 0.95$ . The data in the left plot were simulated from the normal linear model.

- In short,  $R^2$  should not be relied upon as a diagnostic to judge the quality of the model fit. If the model is a good fit, then  $R^2$  says something about the **predictive power** of the model.
- $R^2$  can be found in the R output as `Multiple R-squared`.
- In our example,  $R^2 = 0.61$ .
- Note that, by definition,  $R^2$  cannot decrease if  $p$  increases. **Adjusted  $R^2$**  is a modification of  $R^2$  that introduces a penalty for increasing  $p$ .

1

---

## 2 Basic Variable Selection

- 3 The basic, general strategy for avoiding overfitting with models is
- 4 to compare the values of AIC across different number/choices of co-
- 5 variates
- 6 Recall that we prefer the model which has the smallest value of AIC
- 7 Unfortunately, when the number of covariates is large, the number of
- 8 possible models is enormous. (There are  $2^p$  possible models if there
- 9 are  $p$  covariates.)
- 10 Later we will discuss another more modern strategy for dealing with
- 11 this problem, the **lasso**.

- 1 We'll utilize the regression problem introduced in Part 1 in which we
- 2 sought to estimate the forward-rate function based on the current
- 3 prices of zero-coupon bonds. The data are shown again as Figure 3.

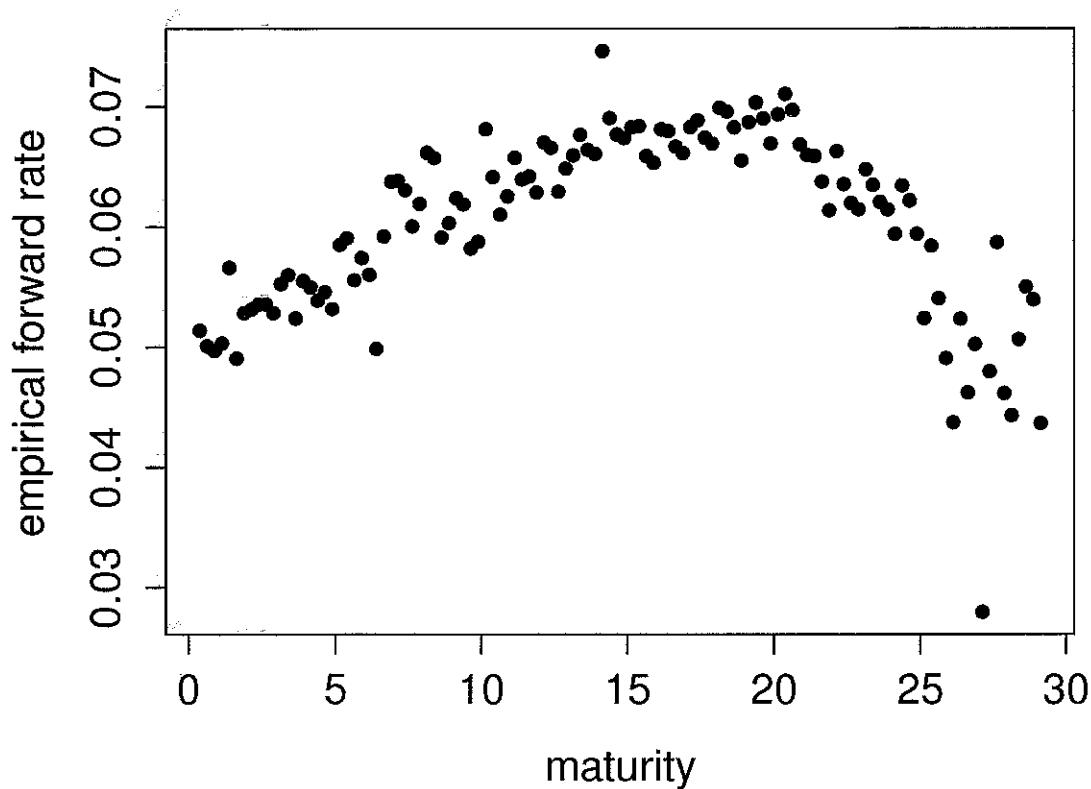


Figure 3: The response versus time for the estimation of the forward rate function.

- 4 You can read in this data set using the command

```

5 > forratedat =
6   read.table("http://www.stat.cmu.edu/~cschafer/MSCF/forratedat.txt",
7   header=T)
8 > attach(forratedat) ← can refer to the variable names
   without forratedat$
```

- 1 We are considering polynomial models of the form

2

$$Y_i = \sum_{j=0}^p \beta_j T_i^j + \epsilon_i$$

- 3 but our goal is to find the best subset of the  $p$  polynomial terms to  
4 include in the model. (As is usually the case, we do not consider  
5 removing the intercept  $\beta_0$ .)

- 6 The “full” model (largest possible model) we’ll consider includes up  
7 to the ninth power.

- 8 We will scale the predictor first, so that it has mean zero and standard  
9 deviation one:

10 > maturity = scale(maturity)

- 11 Now, we can fit the full model:

12 > fullmod = lm(emp ~ maturity + I(maturity^2) +  
13 I(maturity^3) + I(maturity^4) + I(maturity^5) +  
14 I(maturity^6) + I(maturity^7) + I(maturity^8) +  
15 I(maturity^9))

- 16 Note the use of `I()` function to wrap the polynomial terms is re-  
17 quired by R.

*Q.B. (sk) "as-is function"*

## 1. Exhaustive Search

- 2 In cases where the number of predictors is small, an exhaustive search
- 3 over all possible models is possible.
- 4 One implementation of this in R is the function `bestglm()` which is
- 5 part of the package `bestglm`.

`install.packages("bestglm")`

- 6 Unfortunately, the syntax for this function requires that the predictors and response be in a single data frame, with the response as the
- 7 last column.

- 8 Consider the following commands

```

10 > allpreds = cbind(maturity, maturity^2,
11      maturity^3, maturity^4, maturity^5,
12      maturity^6, maturity^7, maturity^8,
13      maturity^9)
14 > Xyframe = data.frame(cbind(allpreds, emp))

```

↗ response

- 15 Note that now `Xyframe` holds the predictors and the response.

```
> bestmod = bestglm(Xyframe, IC="AIC")
```

- 16 By specifying `IC = "AIC"` we are telling the function to use AIC to
- 17 choose the model. `glm` = "Generalized Linear Model"

1 The R command

2 > print(bestmod)

3 will show the output below

```
4 AIC
5 BICq equivalent for q in (0.0443666836953486, 0.908095644854053)
6 Best Model:
7
8   Estimate Std. Error t value Pr(>|t|)
9 (Intercept) 0.066820498 0.0004770183 140.079524 8.381181e-126
10 V1 0.004950250 0.0008628244 5.737262 8.591912e-08
11 V4 -0.008479843 0.0007636180 -11.104824 1.134645e-19
12 V5 -0.004005042 0.0008106249 -4.940684 2.801999e-06
13 V6 0.002275548 0.0002881148 7.898061 2.313625e-12
14 V7 0.001183985 0.0002729549 4.337659 3.206729e-05
```

14 We see that the optimal model takes the form

$$15 Y_i = \beta_0 + \beta_1 T_i + \beta_4 T_i^4 + \beta_5 T_i^5 + \beta_6 T_i^6 + \beta_7 T_i^7 + \epsilon_i$$

16 This is the same model that was found in Part 1.

## <sup>1</sup> Stepwise Regression

- <sup>2</sup> An exhaustive search is not feasible when there is a large number of
- <sup>3</sup> predictors.
- <sup>4</sup> A classic strategy for dealing with the large number of possible mod-
- <sup>5</sup> els is to use a **stepwise variable selection** procedure.
- <sup>6</sup> With this method, one (typically) starts with the largest model under
- <sup>7</sup> consideration, and then takes a sequence of **steps**: At each step one
- <sup>8</sup> covariate is either **added** or **dropped**.
- <sup>9</sup> The decision for which step to take (i.e., which covariate to add or
- <sup>10</sup> drop) is based on AIC: You take the step which reduces AIC the most.
- <sup>11</sup> Once AIC no longer changes, the process stops, and you have your
- <sup>12</sup> final model.
- <sup>13</sup> This algorithm is far from perfect, but is useful in the appropriate
- <sup>14</sup> situations.
- <sup>15</sup> This is built into R using the function `step()`

- Now we will use the `step()` function on the example above.

- The syntax is simple:

> `finalmod = step(fullmod, direction="both")` all predictors

- Recall that `fullmod` holds the output of the full model fit.

- The use of `direction = "both"` tells R to consider not only removing predictors, but also adding predictors to the model. (Once a predictor is removed, it could be returned to the model in a later step.)

- 1 The output of the first step of algorithm appears below.

```

2 Start: AIC=-1296.97
3 emp ~ maturity + I(maturity^2) + I(maturity^3) + I(maturity^4) +
4     I(maturity^5) + I(maturity^6) + I(maturity^7) + I(maturity^8) +
5     I(maturity^9)
6
7             Df  Sum of Sq      RSS      AIC
8 - I(maturity^6)  1  3.0370e-07  0.0013612 -1298.9
9 - I(maturity^9)  1  3.3910e-07  0.0013613 -1298.9
10 - I(maturity^8) 1  9.2100e-07  0.0013619 -1298.9
11 - I(maturity^3) 1  9.7950e-07  0.0013619 -1298.9
12 - I(maturity^7) 1  1.7992e-06  0.0013627 -1298.8
13 - I(maturity^2) 1  2.1071e-06  0.0013630 -1298.8
14 - I(maturity^5) 1  3.4433e-06  0.0013644 -1298.7
15 - I(maturity^4) 1  4.7036e-06  0.0013656 -1298.6
16 - maturity      1  2.2945e-05  0.0013839 -1297.0
17 <none>                  0.0013609 -1297.0

```

If remove  $X_6$ ,  
what is AIC?

- 18 Note how the table lists each of the covariates currently in the model,  
19 and gives what AIC would be if that covariate were removed.

- 20 We also see the line labelled <none>, indicating that the AIC would  
21 stay at  $-1297.0$  if none were removed.

- 22 The covariates are ordered by increasing values of AIC. (Unfortu-  
23 natley, due to rounding it's not possible to see some distinctions in  
24 this output.)

1 So, the term  $I(\text{maturity}^6)$  is removed at the first step.

2 The second step is shown as:

3 Step:  $AIC = -1298.94$   
4  $\text{emp} \sim \text{maturity} + I(\text{maturity}^2) + I(\text{maturity}^3) + I(\text{maturity}^4) +$   
5  $I(\text{maturity}^5) + I(\text{maturity}^7) + I(\text{maturity}^8) + I(\text{maturity}^9)$

6

	Df	Sum of Sq	RSS	AIC
8 - $I(\text{maturity}^9)$	1	$3.3900e-07$	0.0013616	-1300.9
9 - $I(\text{maturity}^3)$	1	$9.8000e-07$	0.0013622	-1300.9
10 - $I(\text{maturity}^7)$	1	$1.7990e-06$	0.0013630	-1300.8
11 - $I(\text{maturity}^5)$	1	$3.4430e-06$	0.0013647	-1300.7
12 - $I(\text{maturity}^2)$	1	$1.9184e-05$	0.0013804	-1299.3
13 - $\text{maturity}$	1	$2.2945e-05$	0.0013842	-1299.0
14 <none>			0.0013612	-1298.9
15 + $I(\text{maturity}^6)$	1	$3.0400e-07$	0.0013609	-1297.0
16 - $I(\text{maturity}^4)$	1	$8.0262e-05$	0.0014415	-1294.3
17 - $I(\text{maturity}^8)$	1	$1.2814e-04$	0.0014894	-1290.5

← no changes to model

18 The decision of this step is to remove  $I(\text{maturity}^9)$ . Note that  
19 also under consideration was to add back in the term that was re-  
20 moved.

1 This process continues until a final model is reached:

```

2 Step: AIC=-1303.2
3 emp ~ maturity + I(maturity^4) + I(maturity^5) + I(maturity^7) +
4 I(maturity^8) ] ]
5
6             Df  Sum of Sq      RSS      AIC
7 <none>                 0.0013818 -1303.2
8 + I(maturity^2)  1  0.00001918  0.0013626 -1302.8
9 + I(maturity^6)  1  0.00001738  0.0013644 -1302.7
10 + I(maturity^3) 1  0.00000105  0.0013808 -1301.3
11 + I(maturity^9) 1  0.00000041  0.0013814 -1301.2
12 - I(maturity^7) 1  0.00023342  0.0016152 -1287.1
13 - I(maturity^5) 1  0.00030284  0.0016847 -1282.2
14 - maturity      1  0.00040836  0.0017902 -1275.2
15 - I(maturity^8) 1  0.00075672  0.0021385 -1254.5
16 - I(maturity^4) 1  0.00212230  0.0035041 -1197.3

```

17 So, we have arrived (after only four steps) at a model that includes  
18 powers 1, 4, 5, 7, and 8.

19 **Exercise:** How does this model compare to the one we stated as being  
20 optimal using the exhaustive search? What happened?

21 The exhaustive search found 1, 4, 5, 6, 7 as  
22 best.

23  $(maturity)^6$  and  $(maturity)^8$  are strongly  
24 correlated, including one or the other  
25 Doesn't make a significant difference

## <sup>1</sup> <sup>2</sup> Cross Validation

- <sup>3</sup> We've discussed the use of AIC for the purpose of **model selection**,
- <sup>4</sup> i.e. the process of determining which covariates should be included
- <sup>5</sup> in the model.
  
- <sup>6</sup> An important alternative is **leave-one-out cross validation**.
  
- <sup>7</sup> In this approach, one imagines refitting the model  $\underline{n}$  times, each time
- <sup>8</sup> excluding one of the  $n$  observations.
  
- <sup>9</sup> At iteration  $i$ , observation  $i$  is excluded. The fitted model is then used
- <sup>10</sup> to predict the response for this observation. Call this  $\hat{y}_{(-i)}$ .
  
- <sup>11</sup> Finally, we calculate the quantity

$$\text{PRESS} = \sum_{i=1}^n (y_i - \hat{y}_{(-i)})^2$$

- <sup>12</sup> where **PRESS** stands for **Prediction Error Sum of Squares**.

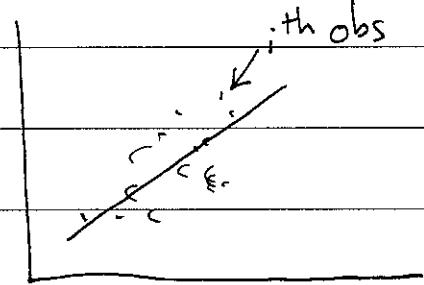
- 1 The motivation for this is as follows: Once observation  $i$  is removed
  - 2 from the fit, we have removed the **influence** that this observation has
  - 3 on the parameter estimates.
  
  - 4 Now, we can think of observation  $i$  as being a "new" observation,
  - 5 and we can ask: "How well does the model, containing only these
  - 6 covariates, predict the response for this case?"
- $\nwarrow$
- 
- 7 The quantity  $(y_i - \hat{y}_{(-i)})^2$  quantifies the amount of error in this prediction.
  
  - 9 **Exercise:** Which is larger: PRESS or RSS?

$$10 \quad RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$11 \quad PRESS = \sum_{i=1}^n (y_i - \hat{y}_{(-i)})^2$$

$$13 \quad PRESS \geq RSS$$

15      Least squares tries to  
 16      Minimizes RSS



$$18 \quad For \ each \ i, \ (y_i - \hat{y}_i)^2 \leq (y_i - \hat{y}_{(-i)})^2$$

- 1 Fortunately, it is not actually necessary to fit  $n$  models in order calculate PRESS.
- 2
- 3 The remarkable result is as follows:

4

- 5 Let  $\mathbf{Y}$  be a vector consisting of the  $n$  responses, and let  $\widehat{\mathbf{Y}}$  be a
- 6 vector consisting of the  $n$  fitted values (from the full model).

- 7 If it is the case that  $\widehat{\mathbf{Y}} = \mathbf{H}\mathbf{Y}$ , then

$$y_i - \widehat{y}_{(-i)} = \frac{\widehat{\epsilon}_i}{1 - h_{ii}}$$

- 8 where  $h_{ii}$  is the  $i^{th}$  diagonal element of  $\mathbf{H}$ .

10

- 11 In the case of linear regression, it is the case that

$$\mathbf{H} = \mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$$

- 12 where  $\mathbf{X}$  whose columns hold the values of the predictors. See page

- 13 373 in Ruppert for the details.

$\mathbf{H}$  is  $n \times n$  matrix

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \dots \\ x_{12} & x_{22} & \dots \\ \vdots & \vdots & \ddots \\ x_{1n} & x_{2n} & \dots \end{bmatrix}$$

$\mathbf{X}$  = "Design matrix"

- 14 This matrix  $\mathbf{H}$  is called the **hat matrix**. projection onto the column space of  $\mathbf{X}$
- 15
- 16 The diagonal elements  $h_{ii}$  are called the **leverages**.

1 The leverages can be found in R using

2 > levs = hatvalues(fullmod)

↙ output of  $lm()$

3 Then we can find PRESS using

4 > PRESS = sum((fullmod\$resid/(1-levs))^2)

$$PRESS = \sum_{i=1}^n \left( \frac{\hat{\epsilon}_i}{1-h_{ii}} \right)^2$$

5 We find PRESS to be 0.001974 for this model.

6 Of course, our goal is to find the model with lowest PRESS. In R we

7 can do this using the function `bestglm()` with `IC="LOOCV"`:

8 > bestmod2 = `bestglm(Xyframe, IC="LOOCV")`

9 LOOCV

10 BICq equivalent for q in (0.0443666836953498, 0.908095644854043)

11 Best Model:

		Estimate	Std. Error	t value	Pr(> t )
13	(Intercept)	0.066820498	0.0004770183	140.079524	8.381181e-126
14	V1	0.004950250	0.0008628244	5.737262	8.591912e-08
15	V4	-0.008479843	0.0007636180	-11.104824	1.134645e-19
16	V5	-0.004005042	0.0008106249	-4.940684	2.801999e-06
17	V6	0.002275548	0.0002881148	7.898061	2.313625e-12
18	V7	0.001183985	0.0002729549	4.337659	3.206729e-05

19 We arrive at the same model as when we used AIC.

## 1 AIC versus PRESS

- 2 AIC and PRESS will often give similar, if not the same, result for the
- 3 model choice.
  
- 4 There are a couple main reasons to prefer PRESS to AIC: First, the
- 5 quantity being estimated is by PRESS is a very natural measure of
- 6 prediction performance. Second, the theory behind AIC is based on
- 7 the assumption that the distributional assumption for the response is
- 8 correct. Look at normal prob. plot. keep in mind  
~~code~~ ~~assuming~~ assuming normality.
- 9 A main reason to choose AIC over PRESS is that AIC is more stable.
- 10 The estimate that PRESS provides of the expected prediction error is
- 11 subject to large variance. Hence, PRESS may not perform well for
- 12 small to moderate sample sizes.
  
- 13 Although PRESS is simple to calculate in the case of linear regression,
- 14 this will not always be the case. In general, AIC will be easier to
- 15 calculate than PRESS.
  
- 16 As is often the case, our past experience will often guide our choice
- 17 of criterion.

## <sup>1</sup> <sup>2</sup> Detecting Influential Observations

<sup>3</sup> There is a difference between an observation having a *large absolute*  
<sup>4</sup> *residual* and the observation having a *large influence on  $\hat{\beta}$* . Indeed,  
<sup>5</sup> some of the most influential observations are ones whose residuals  
<sup>6</sup> are small: Consider the fourth case in “Anscombe’s Quartet” for an  
<sup>7</sup> extreme example.

<sup>8</sup> To account for this, it is useful to inspect **Cook’s Distance** for the  
<sup>9</sup> model fit:

<sup>10</sup> Heuristically, Cook’s Distance for observation  $i$  equals the amount by  
<sup>11</sup> which the vector of fitted values would move if the  $i^{th}$  observation  
<sup>12</sup> were removed from the data set. Normalization by the number of  
<sup>13</sup> parameters and  $\hat{\sigma}^2$  helps to put this quantity on a standard scale.

<sup>14</sup> In the case of the linear regression model,

<sup>15</sup> Cook’s distance for observation  $i$  = 
$$\left( \frac{1}{(p+1)\hat{\sigma}^2} \right) \sum_{j=1}^n (\hat{y}_{j(-i)} - \hat{y}_j)^2$$
<sup>16</sup> where  $\hat{y}_{j(-i)}$  equals the prediction for observation  $j$  from the model  
<sup>17</sup> that excluded observation  $i$  in fitting the model.

- 1 A standard “rule of thumb” is that one should consider an observation influential if Cook’s Distance is larger than  $4/n$ , but this should be taken as a rough guide. One should be particularly aware of values of Cook’s Distance which are large relative to the others.

- 5 In R, these are found using

```
6 > cookd = as.numeric(cooks.distance(ff3modPNC))2
```

↑  
output of lm()

- 7 We can create Figure 4 as follows:

```
8 > plot(cookd, xlab="Observation", ylab="Cook's Distance")
9 > lines(c(1, length(cookd)),
10 c(4/length(cookd), 4/length(cookd)), lwd=2, col=2, lty=2)
```

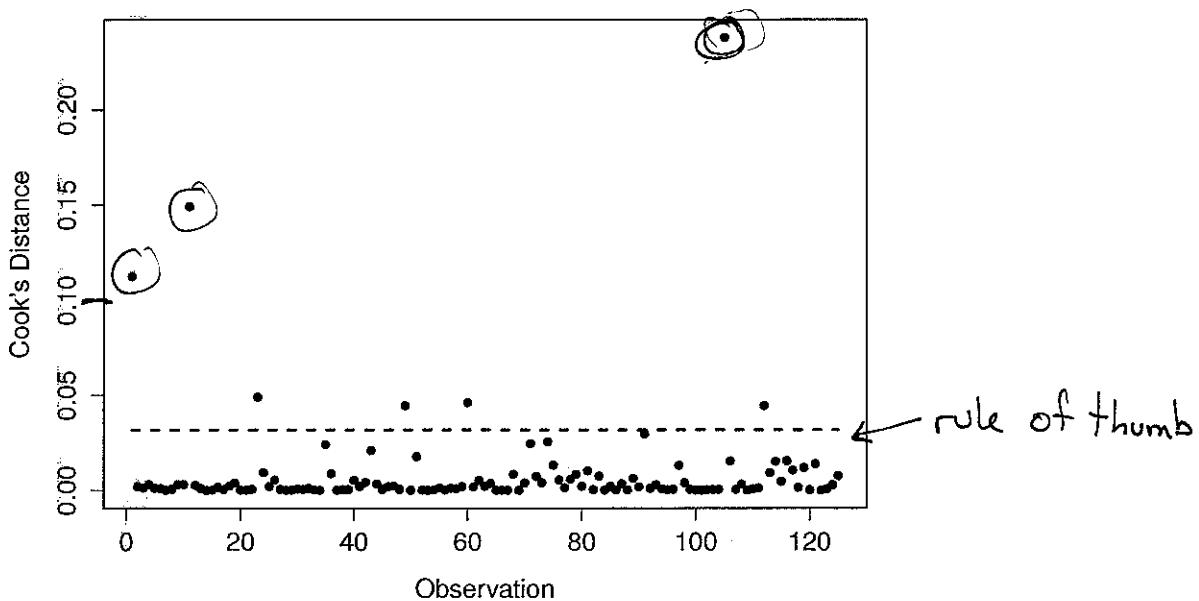


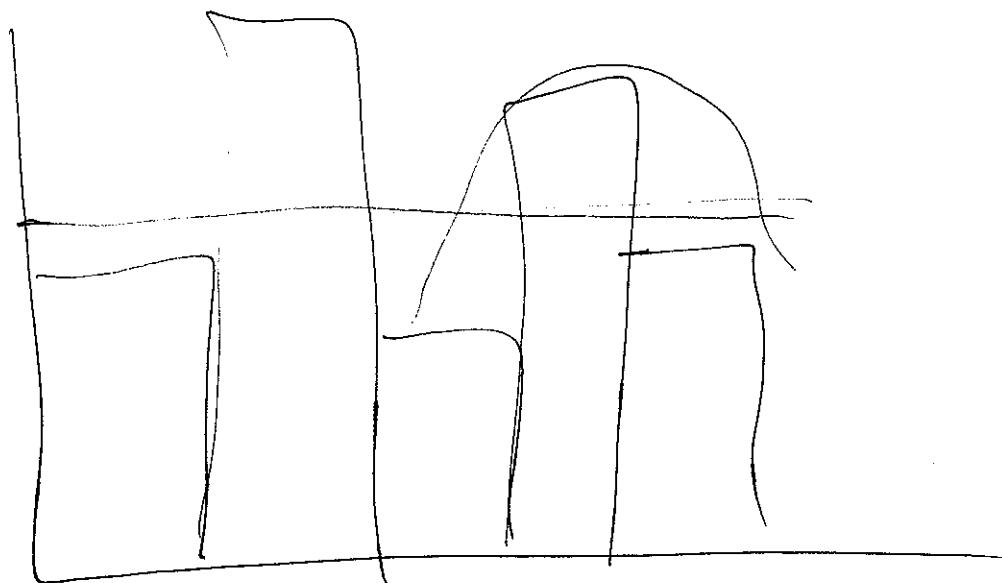
Figure 4: Plot of Cook’s Distance for our Fama-French Example.

`which(cookd > 0.10)` returns indices where exceeds 0.10

<sup>2</sup>Again, the `as.numeric()` is needed to convert the output from the time series format.

- 1 Based on the "rule of thumb" there appear to be many possibly-influential observations. It would make sense to inspect the most extreme cases.
- 4 **Warning:** It is important to keep in mind that these calculations are done based on the removal of **one** observation. Once a single observation is removed, the influence of the remaining observations may change entirely, as the model itself will change.

Consider Using robust regression



1

## 2 Box-Cox Transformation

3 We earlier discussed the value of **transforming the response** for im-  
4 proving the fit of the model. This was particularly helpful in certain  
5 cases of heteroskedasticity.

6 The **Box-Cox Transformation** makes the choice of transformation  
7 into a more formal process.

8 This is **only** useful in the case where the response is **strictly positive**.

9 First, we define **Box-Cox power transformation** function:

$$10 y^{(\lambda)} = \begin{cases} \frac{y^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(y) & \text{if } \lambda = 0 \end{cases}$$
$$\lim_{\lambda \rightarrow 0} \frac{y^\lambda - 1}{\lambda} = \log(y)$$

11 This defines a wide range of possible transformations for the response,  
12 including  $\sqrt{y}$ ,  $y^2$ ,  $\log(y)$ ,  $1/\sqrt{y}$ , and so forth. Also, note that this is a  
13 continuous function of  $\lambda$ .

- Now we put forth the model

$$Y^{(\lambda)} = \beta_0 + \sum_{j=1}^p \beta_j x_j + \epsilon$$

- and then ask: What choice of  $\lambda$  yields the best fitting model?

- Through a maximum likelihood procedure, the best  $\lambda$  can be found,
- and then this choice is used to transform the response prior to fitting
- the full regression model.

- Typically, a confidence interval is formed for  $\lambda$ , and then a "nice" value is chosen from this interval to serve as the transformation. Examples of "nice" values include -1, -0.5, 0, 0.5, 1, and so forth.

$$\lambda = 0.457$$

- <sup>1</sup> This is implemented in R via the function `boxCox()`, which is part of the package `car`.
  
- <sup>3</sup> Recall the example that we considered when discussing simple linear regression, the prediction of the realized volatility from the log total volume. Ultimately, we settled on a model where the log transform was applied to the response.
  
- <sup>7</sup> See Figure 5 for the fit in that case, along with the plot of residuals versus fitted values.

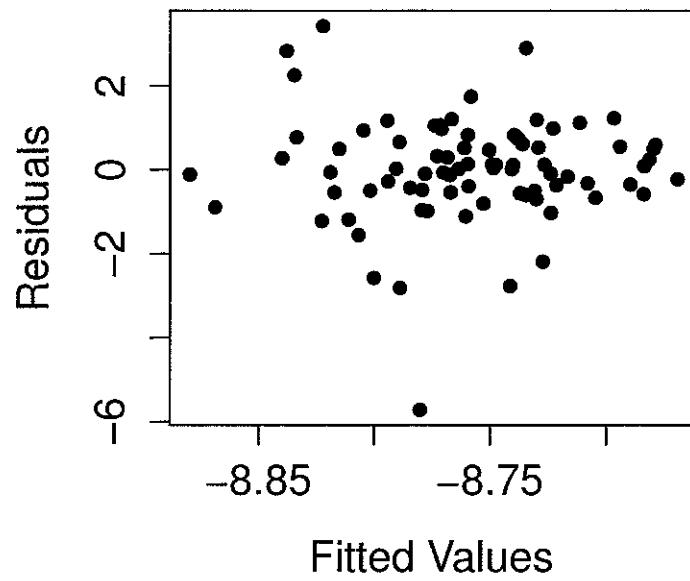
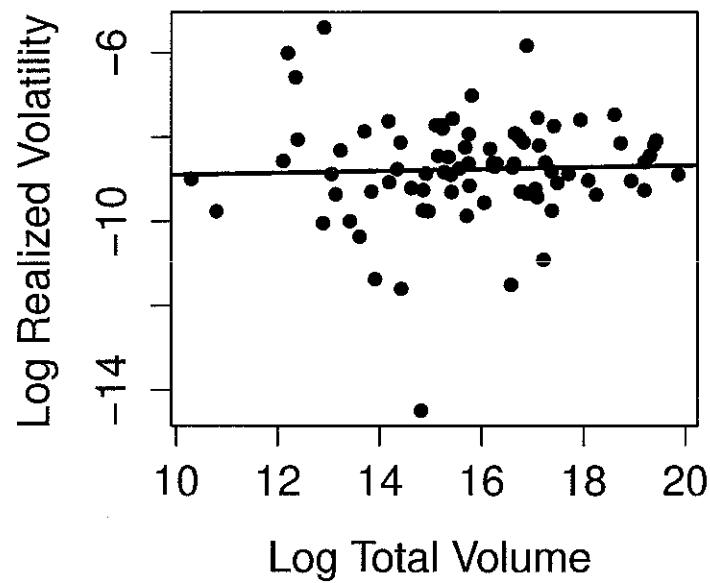


Figure 5: The model fit to realized volatility after taking the log transform. The upper plot shows the scatter plot with the regression line superimposed, the lower plot shows the plot of residuals versus fitted values.

- 1 Let's now try the Box-Cox procedure on this example.
- 2 Assume that the response is stored `realizedvol` and the predictor is `logvolume`. Then, the command is simply
- 3 

```
> boxCox(realizedvol ~ logvolume)
```

*Same syntax as lm()*
- 4 This creates the plot shown as Figure 6.

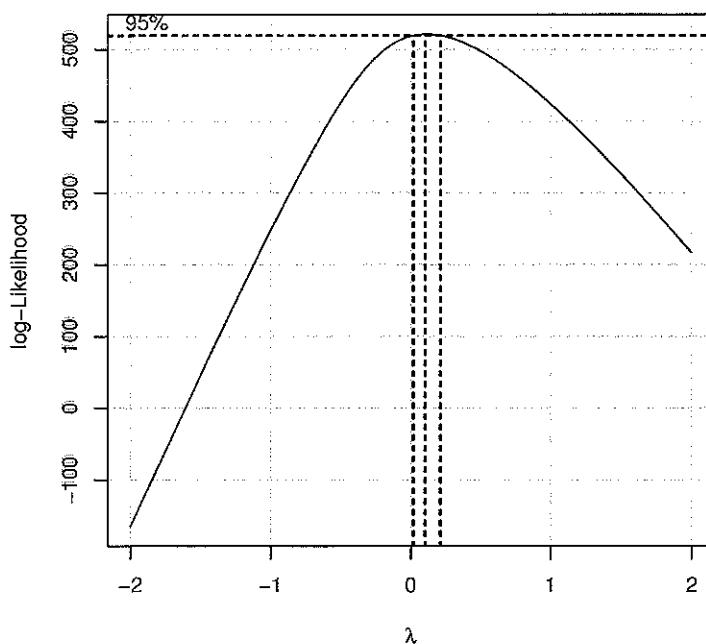


Figure 6: The output of `boxCox()`, when applied to the simple realized volatility example.

1 **Exercise:** Was the log transform a good choice?

$$(\lambda=0)$$

2 Yes,  $\lambda=0$  is very close to the (tight) CI  
3 for  $\beta$ .

4

---

5

6 We see that the optimal choice is  $\lambda \approx 0.1$ .

7 This can then be applied to the data using the function `bcPower()`:

8 > `transrealizedvol = bcPower(realizedvol, 0.1)`

$\lambda$

## 1 The Yeo-Johnson Transformation

- 2 The Box-Cox Transformation suffers from the limitation that it can
- 3 only be applied to strictly positive variables. A generalization that
- 4 works for any variable is the **Yeo-Johnson Transformation**.

- 5 The Yeo-Johnson Power Transformation is

$$y^{(\lambda, \underline{y})} = \begin{cases} \left( (y + 1)^\lambda - 1 \right) / \lambda, & \text{if } \lambda \neq 0, y \geq 0 \\ \log(y + 1), & \text{if } \lambda = 0, y \geq 0 \\ - \left[ (-y + 1)^{2-\lambda} - 1 \right] / (2 - \lambda), & \text{if } \lambda \neq 2, y < 0 \\ - \log(-y + 1), & \text{if } \lambda = 2, y < 0 \end{cases}$$

- 7 It appears somewhat messy, but the intention is much like for the
- 8 Box-Cox Transformation: As  $\lambda$  varies, we get a wide range of power
- 9 and log transformations of the data; the optimal transformation can
- 10 be sought via maximum likelihood.

- 11 In R, this can be implemented using `boxCox()`, but with the additional argument `family = "yjPower"`. To apply the transformation, use the function `yjPower()`.

1

## 2 Robust Regression

- 3 In the least squares approach to regression, we seek the  $\beta$  that mini-  
4 mizes the residual sum of squares

5

$$RSS = \sum_{i=1}^n (y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2.$$

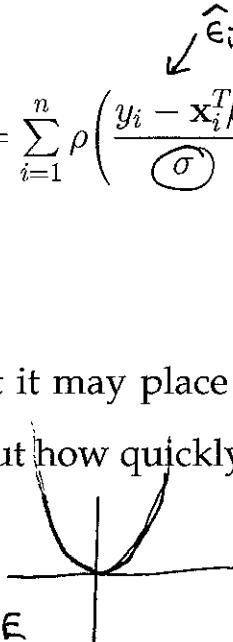
- 6 Of course, this is equivalent to minimizing

7

$$\frac{1}{\sigma^2} \sum_{i=1}^n (y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2 = \sum_{i=1}^n \left( \frac{y_i - \mathbf{x}_i^T \boldsymbol{\beta}}{\sigma} \right)^2 = \sum_{i=1}^n \rho \left( \frac{y_i - \mathbf{x}_i^T \boldsymbol{\beta}}{\sigma} \right)$$

- 8 where  $\rho(x) = x^2$ .

- 9 The general concern with  $\rho(x) = x^2$  is that it may place too much  
10 weight on extreme observations: Think about how quickly the func-  
11 tion  $x^2$  increases as  $|x|$  increases.



- 12 The choice  $\rho(x) = x^2$  may be optimal when the errors are normal, but  
13 it is not very robust to deviations from this assumption.

- 14 If we generated errors from the  $t$  distribution with four degrees of  
15 freedom, for instance, we are likely to see a few extreme errors. These  
16 observations would have a large influence on the fit of the model, if  
17 we proceeded with  $\rho(x) = x^2$ .

- There are many possible choices for  $\rho$ , but here we will focus on a
- popular one, the **Huber loss function**, which is defined as

$$\rho_{\text{Hub}}(x) = \begin{cases} x^2 & \text{if } |x| < k \\ k(2|x| - k) & \text{otherwise} \end{cases}$$

- where  $k > 0$  is set by the user. For technical reasons, the default choice is  $k = 1.345$ .
- Figure 7 compares squared error loss with the Huber loss function.

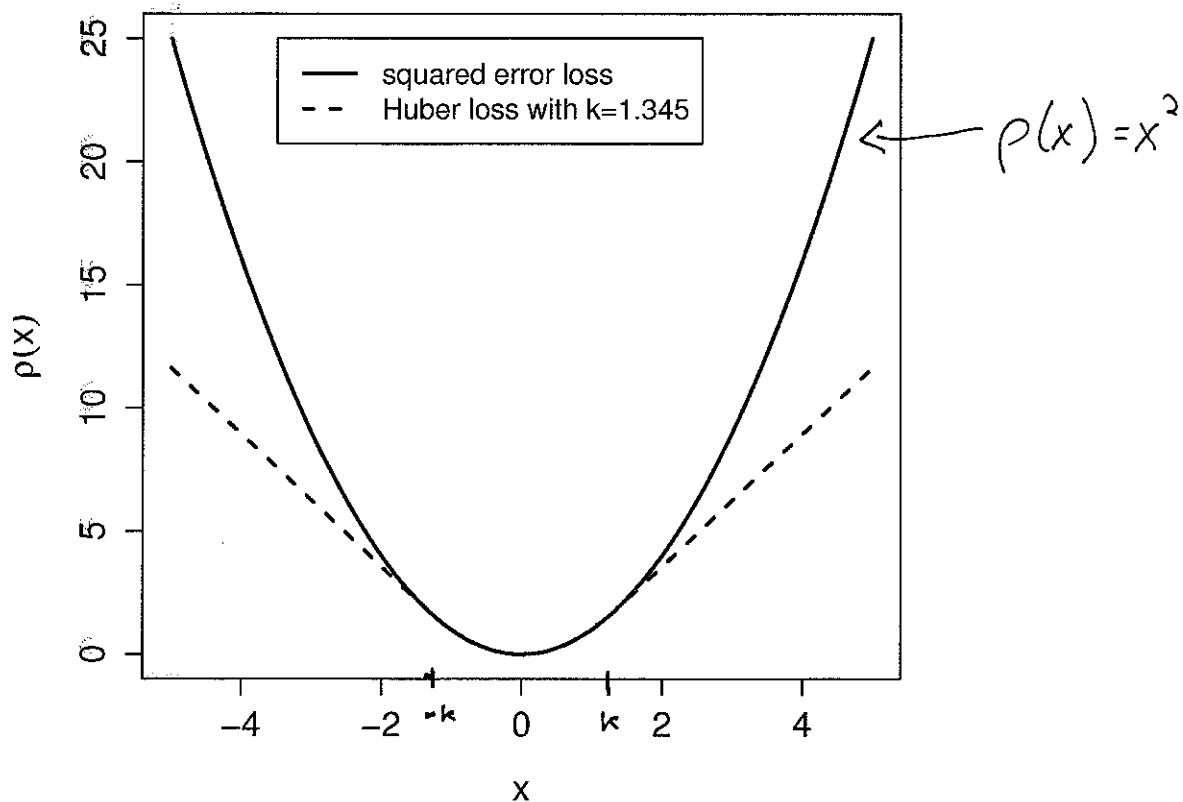


Figure 7: A comparison of the squared error loss function and the Huber loss function.

- Exercise: What is the effect of varying  $k$  in the Huber loss function?

As  $k \nearrow \infty$ ,  $\rho_{\text{Hub}}$  behaves like  $x^2$

As  $k \searrow 0$ ,  $\rho_{\text{Hub}}$  behaves like  $|x|$

This allows for compromise between those

7

8

9

10

11

12

13

14

15

16

17

18

19

20

## 1 Simulated Example

- 2 In this example, the truth is that  $\beta_0 = 1$  and  $\beta_1 = 10$ . The sample size  
 3 is  $n = 100$ . The error has the  $t$  distribution with a single degree of  
 4 freedom, so there is great potential for extreme observations.
- 5 Figure 8 shows two simulated data sets under these conditions, each  
 6 with the true line (solid), the regression line estimated using least  
 7 squares (dashed), and the regression line estimated using the Huber  
 8 loss (dashed-dot).

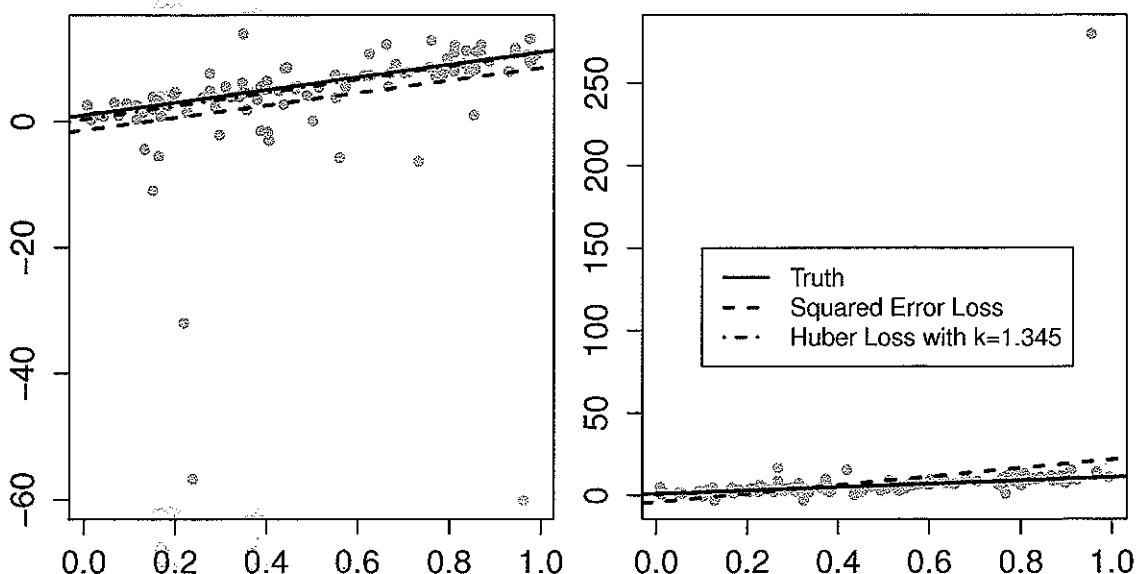


Figure 8: Comparing the performance of the squared error loss and the Huber loss on a simulated regression problem where the error has the  $t$ -distribution with a single degree of freedom.

- <sup>1</sup> It is clear that in both instances that the Huber loss function effectively downweights the extreme observations, and, as a result, results in a closer approximation to the truth.
  
- <sup>4</sup> These examples are, admittedly, chosen to be extreme to illustrate the point. Nevertheless, robust regression is useful in situations where <sup>5</sup> it is suspected that the error distribution is not normal, <sup>6</sup> and there is <sup>7</sup> concern over extreme observations.  
*and/or*

- 1 Robust regression is implemented in R using the function `rlm()`,
- 2 which is part of the MASS package.

- 3 The default setting in `rlm()` is to use the Huber loss function with
- 4  $k = 1.345$ . Otherwise, the command is used much like `lm()`:

```
5 > holdrlm = rlm(PNCexret ~ Mkt.RF + SMB + HML,  
6   data=ffhold)
```

- 7 The result of this fit is shown below. As is the case in many situations,
- 8 the difference relative to the fit with squared error loss is negligible.

9 Coefficients:

```
10          Value  Std. Error t value  
11 (Intercept) 0.0225  0.0615    0.3654  
12 Mkt.RF      1.2139  0.0781   15.5431  
13 SMB         0.1302  0.1622    0.8026  
14 HML         1.1159  0.1751    6.3745  
15  
16 Residual standard error: 0.6326 on 121 degrees of freedom
```

- 17 If you want to use the Huber loss, but choose a different value for  $k$ ,
- 18 then the syntax is

```
19 > holdrlm = rlm(PNCexret ~ Mkt.RF + SMB + HML,  
20   k=0.5, data=ffhold)
```

<sup>1</sup>  <sup>2</sup> **Prediction Intervals**

<sup>3</sup> Recall our discussion of **prediction intervals** within the context of  
<sup>4</sup> simple linear regression.

<sup>5</sup> Here we will rely on R to create the prediction intervals for us. For-  
<sup>6</sup> tunately, this is simple using the command `predict.lm()`.

<sup>7</sup> If I simply use

<sup>8</sup> `> predict.lm(fff3modPNC)`

<sup>9</sup> I will get back the fitted values for the model, i.e. the prediction is  
<sup>10</sup> made for each of the  $x$  that were used in the model.

<sup>11</sup> We are generally more interested in making predictions for **new** or  
<sup>12</sup> **future**  $x$ . We used the notation  $x^*$  for this previously.

<sup>13</sup> We use the `newdata` argument to `predict.lm()` to make such a  
<sup>14</sup> prediction. The trick here is that what is passed to `newdata` must be  
<sup>15</sup> a data frame with variables of the same name as was used in fitting  
<sup>16</sup> the model.

- 1 Suppose we want to predict the excess market return for PNC when
- 2 the excess market return is 0.01, the "Small minus Big" equals 0.1,
- 3 and the "High minus Low" equals 0.3.

- 4 The syntax I would use is as follows

```
5 > predict.lm( ff3modPNC,  
6   newdata=data.frame( Mkt.RF=0.01, SMB=0.1, HML=0.3),  
7   interval="prediction")
```

USE SAME NAMES

- 8 Note that I included interval="prediction" so that the output
- 9 will include the 95% prediction interval for the new observation. If
- 10 you want to change the level, use the argument level to the func-
- 11 tion.

- 12 The output appears as follows

```
13      fit      lwr      upr  
14 1 0.3267227 -1.473245 2.12669
```

- 15 So, a 95% prediction interval for the response in this case is found to
- 16 be  $(-1.47, 2.13)$ .

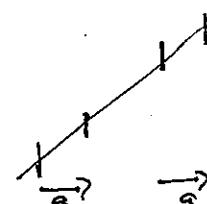
- 17 We could construct a data frame with multiple rows of data, and
- 18 construct prediction intervals for all of those "new" observations si-
- 19 multaneously.

## Part 3: Types of Predictors

- 2 Text references: §3.3.1 and 3.3.2 in ILS
- 3 In this part we will have a practical discussion of how different types of predictors are incorporated into models in R, and how the coefficients are appropriately interpreted.
- 6 We will utilize the standard notation that R uses for specifying models. In all cases let  $y$  denote the response variable; other variables will serve as predictors.
- 9 Recall that the simple form  $y \sim x$  specifies that there is a single predictor  $x$ . By default, an intercept term is included in the model, so the linear predictor in this specification is

$$\beta_0 + \beta_1 x \quad \text{"linear predictor"}$$

- 13 If you want to exclude the intercept, then use  $y \sim x - 1$ .



- 14 In either of case,  $x$  is treated as a **continuous predictor**. The assumption made here is that if this variable is increased by  $a$ , from  $x$  to  $x+a$ , then the linear predictor increases by  $\beta_1 a$ , regardless of  $x$ . This is a reasonable approximation to the truth in a wide range of situations, but one should be aware of what is being assumed.

1

---

## 2 Discrete Variables as Continuous Predictors

3 Of course, many variables are not *continuous quantities*, but that does  
4 not mean that including them as *continuous predictors* is a bad idea.

5 A primary example is that of a **count variable**, i.e., a variable which  
6 counts the number of times something occurs. For example, in Exam-  
7 ple 14.9 on page 391 in Ruppert, the response variable is

8 – whether or not an individual is approved for a credit card

9 and the predictors utilized include

10 – number of major derogatory reports on credit record  
11 – number of dependents  
12 – number of major credit cards held  
13 – number of active credit accounts

14

15 Such predictors **can** be included as continuous predictors; the inter-  
16 pretation remains the same: The linear predictor increases at a con-  
17 stant rate as the variable increases.

1

---

## 2 **Transformed Predictors**

- 3 In some cases one finds that it is better to apply a **transformation**
- 4 to a predictor before including it in a model. Leading examples of
- 5 transformations are replacing  $x$  with  $\log(x)$ , or replacing  $x$  with  $\sqrt{x}$ .
  
- 6 In R, applying a transformation is simple: Just include it in the model
- 7 specification:

8

$$y \sim \log(x)$$

- 9 If you want to include a polynomial term in the model, R requires
- 10 that you use the “as-is” function `I()` to wrap the predictor:

11

$$y \sim x + I(x^2)$$

`I`

- 1 The log transform is particularly useful: If the variable  $x$  is positive,
- 2 then  $\log(x)$  takes values on the entire real line. The log transform can
- 3 also take variables which are very skewed towards zero and give
- 4 them a “nicer” distribution. There are no concrete rules regarding
- 5 this, but this tends to yield better fits.
  
- 6 Returning to Example 14.9 in Ruppert: One of the predictors in that
- 7 application is named “share,” and equals the amount spent monthly
- 8 on credit card payments, relative to annual income. Figure 1 com-
- 9 pares the distribution of this variable, before and after the log trans-
- 10 form. The original variable has significant skew towards zero; this
- 11 is a strong indication that a log transform may be a good idea. The
- 12 transformed variable has a distribution with a good range of values.

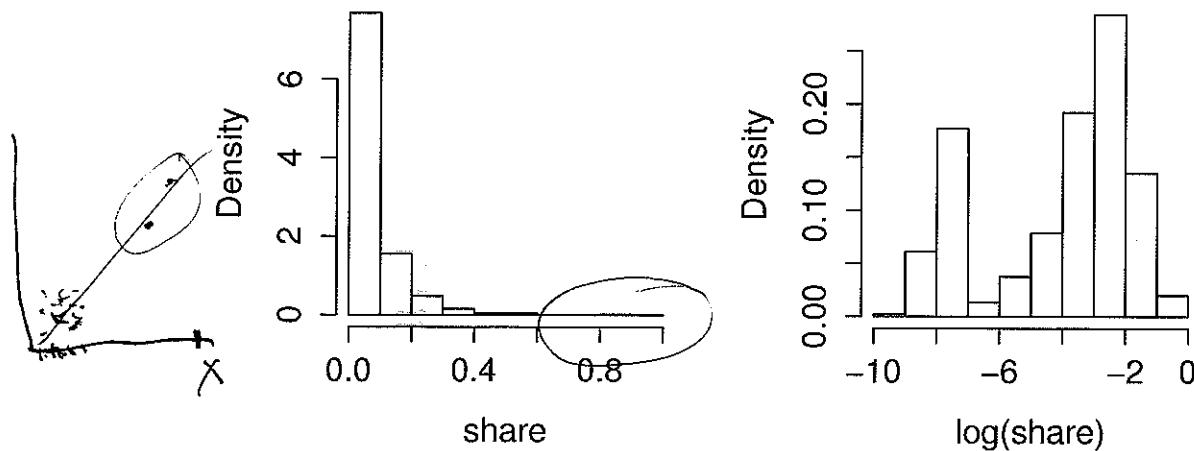


Figure 1: Histograms of the variable “share,” before and after transformation.

- 1 Log transforms have proven to also be useful with count data. If
- 2 there is a possibility that the count will be zero, use the transfor-
- 3 mation  $\log(1 + \text{count})$  instead. Figure 2 shows this transformation
- 4 applied to the "reports" variable.

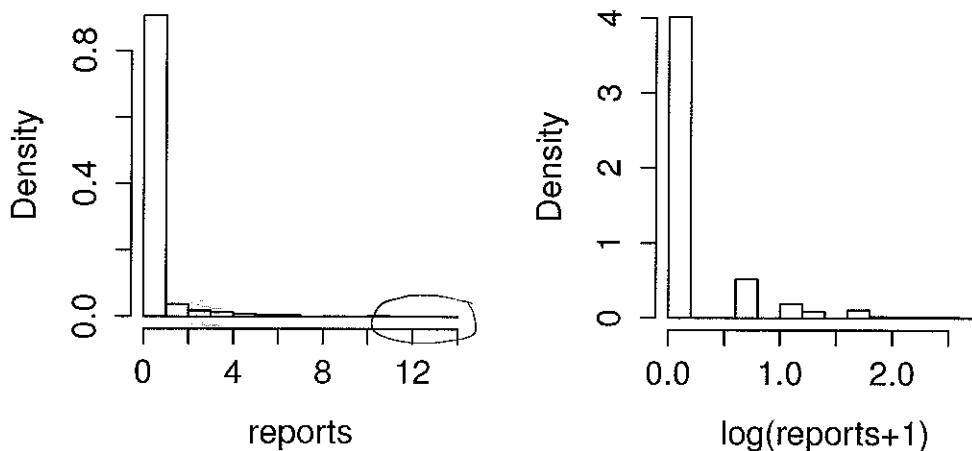


Figure 2: Histograms of the variable "reports," before and after transformation.

- 5 The following quote from Ruppert (page 393) is worth reproducing:
  - 6 There are no assumptions in regression about the distribu-
  - 7 tion of the predictors, so skewed predictor variables can, in
  - 8 principle, be used. However, highly skewed predictors have
  - 9 *high-leverage* points and are less likely to be linearly related
  - 10 to the response. It is a good idea at least to consider trans-
  - 11 formation of highly skewed predictors. In fact, the logistic
  - 12 model was also fit with `reports` and `share` untransformed,
  - 13 but this increased AIC by more than 3 compared to using the
  - 14 transformed predictors.
- h<sub>i</sub>*  
*strong potential*  
*to be influential*

- 1 For predictors which take negative values, one can consider transformations such as  $\log(X + k)$  where  $k$  is a suitable constant. Also, one
- 2 can use the Yeö-Johnson family of transformations.

## 2 Factors (Categorical Predictors)

- 3 We previously discussed using a discrete variable as a continuous
- 4 predictor. There are some discrete variables for which this is clearly
- 5 not a good idea, however. These are cases where the numbers are
- 6 merely labels for the values of a **categorical variable**. Such a variable
- 7 is called a **factor** within R.
  
- 8 In the context of our example, consider a variable which indicates
- 9 highest degree obtained:

Code	Value
0	Did not finish high school
1	Finished high school
2	Some college
3	Graduated college

- <sup>1</sup> **Exercise:** We would say that this variable is on the **ordinal scale**.
- <sup>2</sup> What does that mean? Does that mean that treating it as a continuous
- <sup>3</sup> predictor is a good idea?

<sup>4</sup> "Ordinal scale" - natural ordering to the 4  
<sup>5</sup> "levels" for the variable

<sup>6</sup>  
<sup>7</sup> Including as continuous predictor not a good  
<sup>8</sup> idea. If did: Assuming "Some college"  
<sup>9</sup> is "twice" the education of "finished high school"  
<sup>10</sup> Also assuming that going from "finished HS"  
<sup>11</sup> to "some college" has the "effect" as  
<sup>12</sup> going from "some college" to "graduated college"

- <sup>13</sup> This education variable is "a factor with four levels."

<sup>14</sup> **Here is the tricky part:** A factor with  $k$  levels adds  $k - 1$  terms into  
<sup>15</sup> the linear predictor.

- <sup>16</sup>  $\beta$  parameters
- <sup>17</sup> To see why this is the case, for now imagine that this variable is the
  - <sup>18</sup> only one in the model. The  $k - 1$  new terms correspond to "shifts" in
  - <sup>19</sup> the linear predictor relative to the "baseline" category. In particular,

when the code is...	the linear predictor is...
baseline $\rightarrow$ 0	$\beta_0$
1	$\beta_0 + \beta_1$
2	$\beta_0 + \beta_2$
3	$\beta_0 + \beta_3$

- 2 We will write this linear predictor more compactly as

3 
$$\beta_0 + \underbrace{\beta_1 \mathbf{1}_{\{edu=1\}} + \beta_2 \mathbf{1}_{\{edu=2\}} + \beta_3 \mathbf{1}_{\{edu=3\}}}_{}$$

- 4 where  $\mathbf{1}_A$  is the **indicator variable** for the event  $A$ .

- 5 **Exercise:** What would happen if the intercept term  $\beta_0$  was excluded  
6 from the model?

7 There will be a  $\beta$  for each level (no  
8  $\beta_0$  term)

9  
10  $\beta_1$  gives the expected resp. for when  $edu=0$   
11  $\beta_2$  " " " " " " " "  $edu=1$   
12  $\beta_3$  " " " " " " " "  $edu=2$   
13  $\beta_4$  " " " " " " " "  $edu=3$

14

15

Suppose I had two factors

$X_1$  has levels  $\{a, b, c, d\}$  } total of 5  $\beta$  parameters  
 $X_2$  has levels  $\{0, 1, 2\}$  } (in addition to  $\beta_0$ )  
↓  
baseline

$$\beta_0 \quad X_1 = a, X_2 = 0$$

$$\beta_0 + \beta_1 \quad X_1 = b, X_2 = 0$$

$$\beta_0 + \beta_2 \quad X_1 = c, X_2 = 0$$

$$\beta_0 + \beta_3 \quad X_1 = d, X_2 = 0$$

$$\beta_0 + \beta_4 \quad X_1 = a, X_2 = 1$$

$$\beta_0 + \beta_4 + \beta_1 \quad X_1 = b, X_2 = 1$$

$$\beta_0 + \beta_4 + \beta_2 \quad X_1 = c, X_2 = 1$$

$$\beta_0 + \beta_4 + \beta_3 \quad X_1 = d, X_2 = 1$$

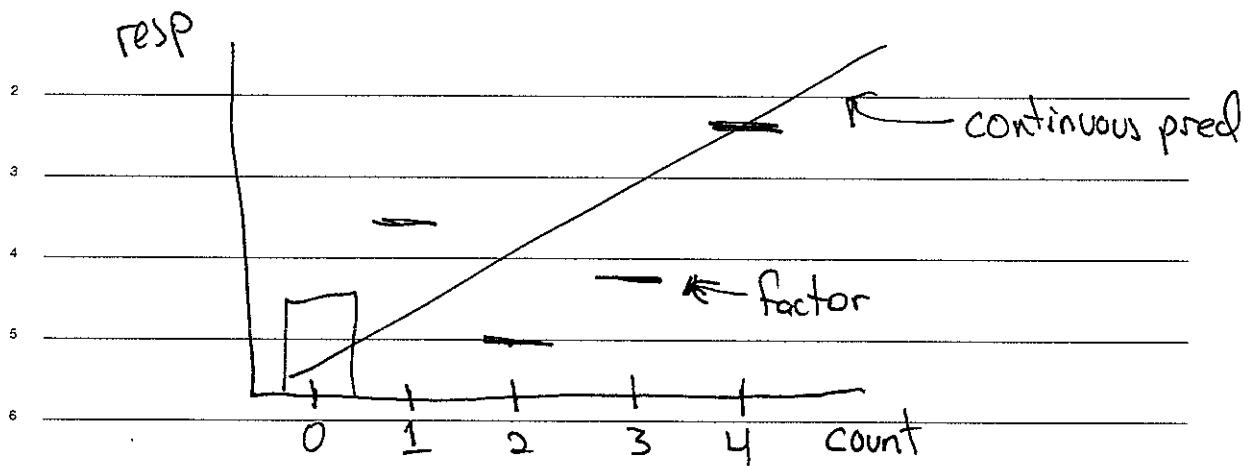
$$\beta_0 + \beta_5 \quad X_1 = a, X_2 = 2$$

$$\beta_0 + \beta_5 + \beta_1 \quad X_1 = b, X_2 = 2$$

$$\beta_0 + \beta_5 + \beta_2 \quad X_1 = c, X_2 = 2$$

$$\beta_0 + \beta_5 + \beta_3 \quad X_1 = d, X_2 = 2$$

- 1 Exercise: Why would we not just **every** discrete predictor as a factor?



8 If  $k$  levels, add  $k-1$   $\beta$ 's to model

9 The number of parameters can grow  
10 quickly in that case

11 12 More parameters requires more data to  
estimate well

Also leads to greater risk of overfitting

- 1 A variable in R can be made into a factor by using the `factor()` function. It is included in the model using the syntax

3

$$y \sim \text{factor}(\text{education}) + X_2 + X_3 + \text{factor}(X_4)$$

- 4 The three parameter estimates will appear in the R output named as
- 5 follows:

6 Coefficients:

		Estimate	Std. Error	t value	Pr(> t )
7					
8	(Intercept)	-0.1460	0.3166	-0.461	0.646
9	factor(education)1	0.1303	0.3550	0.367	0.714
10	factor(education)2	0.4130	0.3497	1.181	0.240
11	factor(education)3	0.2954	0.4005	0.737	0.463

Bo

1

## 2 Interactions

- 3 Interaction terms are added to the model by taking the product of  
4 other predictors.

- 5 In the absence of any interactions, we say that the model is additive  
6 in the predictors.

- 7 **Exercise:** Think carefully about what is being assumed when an additive model is used. In particular, what is the effect on the linear predictor if a covariate is varied?

10 Linear predictor is, say,  $\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$

11  
12 Increasing  $x_2$  by  $a$ , while holding  $x_1$  and  $x_3$   
13 constant, changes the linear pred. by  $\beta_2 a$   
14 regardless of the values for  $x_1$  and  $x_3$ .

15

---

16

---

17

---

18

- 1 Consider a model with two predictor variables  $x_1$  and  $x_2$ , but with
- 2 the interaction term  $x_1x_2$  included. So, the linear predictor is

$$\begin{aligned} 3 \quad & \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 && \leftarrow \text{interaction term} \\ & = \beta_0 + \beta_1 x_1 + (\beta_2 + \beta_3 x_1) x_2 \end{aligned}$$

- 4 Then, if  $x_1$  is held fixed, the effect of varying  $x_2$  is characterized by
- 5 the slope of  $\beta_2 + \beta_3 x_1$ . In other words, if  $x_2$  is increased by  $c$ , then the
- 6 linear predictor increases by  $(\beta_2 + \beta_3 x_1)c$ .

- 7 The key enhancement is that this slope now depends on  $x_1$ .
- 8 Of course, one may wonder: Why does it make sense to let the slope
- 9 depend on  $x_1$  **in this particular manner?**

- 10 The best way to answer this is to consider the interaction  $x_1x_2$  to be a
- 11 **first order approximation** to this dependence.
- 12 It is much like how we often use straight lines to approximate rela-
- 13 tionships which are not truly straight: These are useful approxima-
- 14 tions in the right circumstances.

- 1 In R, if you include  $x_1 * x_2$  in the model formula, it will include **all** of  $x_1$ ,  $x_2$ , **and** the interaction  $x_1 x_2$  in the model. If you only want to
- 2 include the interaction, use  $x_1 : x_2$

- 4 In other words, these two specifications will give the same result:

5 >  $\text{lm}(y \sim x_1 + x_2 + x_1 : x_2)$

6 >  $\text{lm}(y \sim x_1 * x_2)$

- 7 One can consider higher order interactions, such as  $x_1 x_2 x_3$  (**a three-way interaction**) in a model.

- 9 The R command

10 >  $\text{lm}(y \sim x_1 * x_2 * x_3)$

- 11 would include all the linear terms, the two-way interactions, along
- 12 with the three-way interaction.

The number of  $\beta$  parameters will grow rapidly as include higher order interactions

- 1 **Exercise:** What will happen if you include in the model an interaction between a continuous predictor and a factor?
- 2

3 Let  $x_1$  be a cont. pred,  $x_2$  be a factor with 3 levels  
4  $a, b, c$

5 Include  $x_1, x_2, x_1 x_2$  in the model

6 R output will include the intercept and

$x_1$	$\hat{B}_1$
$x_2 b$	$\hat{B}_2$
$x_2 c$	$\hat{B}_3$
$x_1 : x_2 b$	$\hat{B}_4$
$x_1 : x_2 c$	$\hat{B}_5$

12 What is the effect of increasing  $x_1$  by  $k$ ?

13 The linear predictor increases by

14  $\hat{B}_1 k$  when  $x_2 = a$

15  $(\hat{B}_1 + \hat{B}_4) k$  when  $x_2 = b$

16  $(\hat{B}_1 + \hat{B}_5) k$  when  $x_2 = c$

17 The slope for  $x_1$  depends on the level  
18 of the factor  $x_2$

What is the effect of switching from  $X_2 = a$  to  $X_2 = b$ ? (without interaction, would increase linear pred. by  $\hat{\beta}_2$ )

With interaction, linear predictor increase by  $\hat{\beta}_2 + \hat{\beta}_4 X_1$

Likewise, switching from  $X_2 = a$  to  $X_2 = c$ , linear pred. increases by  $\hat{\beta}_3 + \hat{\beta}_5 X_1$

$$c - b = (c - a) - (b - a)$$

$X_1$

$X_2$

$X_1 X_2$

## Part 4: Generalized Linear Models

- 2 Text references: §14.7 in Ruppert, §4.3 in ILS
- 3 The **Generalized Linear Model (GLM)** is a broadly-useful class of
- 4 models that allow for a range of assumptions on the distribution of
- 5 the response. This class includes the classic normal linear models,
- 6 logistic regression, Poisson regression, and more.
- 7 As before, the goal is to predict the response variable  $Y$  using co-
- 8 variates  $X_1, X_2, \dots, X_p$ . The term “linear” in the name “generalized
- 9 linear model” refers to the linear combination of the predictors which
- 10 is used for making predictions. In particular, we call

$$\beta_0 + \underbrace{\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p}_{\text{the linear predictor for the model.}} = \mathbf{x}^T \boldsymbol{\beta}$$

- 11
- 12 the **linear predictor** for the model.

1 In the classic linear regression model, the response is modelled as

2 
$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$$

3 where  $\epsilon$  is some random, mean-zero error. But, what if the response

4  $Y$  is binary (0/1) or if it is integer-valued (e.g., a count)? Fitting such

5 a model would not make sense.

6 The GLM framework allows us to extend to a wider range of distri-

7 butional assumptions for the response, while preserving the linear

8 predictor.

## 2 Basic Framework

3 A GLM has three key components:

4 1. An assumption regarding the distribution for the response, i.e.  
5 the  $Y_i$  are all assumed to be normal, or all binomial, or all Poisson,  
6 etc. This distribution is called the **family** of the model.

7 2. A collection of covariates  $x_i$  corresponding to each of the re-  
8 sponse variables. As usual, the goal is to use these covariates  
9 to model the response variable.

10 3. A monotonic function  $g()$ , called the **link function**, which is such  
11 that

$$g(E(Y_i)) = \mathbf{x}_i^T \boldsymbol{\beta} \quad \text{linear pred.}$$

13 where  $\boldsymbol{\beta}$  are a vector of parameters to be estimated.

- 1 **Exercise:** Explain how the normal linear model fits into the GLM framework.
- 2

3 
$$E(Y_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} = \underbrace{\mathbf{x}_i^T \boldsymbol{\beta}}$$

4  
5 So, the link function is the identity  
6 function,  $g(x) = x$

7 The "family" is the normal dist.  
8

- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16

- 17 In this part we will see other examples of GLM, starting with the
- 18 most important case, the **binary logistic regression model**.

- 1 One fits a GLM in R using the function `glm()`. The syntax is much
- 2 like that of `lm()` in the way the model is specified. An additional
- 3 argument `family` specifies the distribution for the response. Possi-
- 4 ble choices include `family = binomial, gaussian, Gamma,`
- 5 `poisson, inverse.gaussian`. The default is gaussian. *normal*
- 6 The link function is specified via the `family`; each of the families has
- 7 a default link. For now, we will not be concerned with using some-
- 8 thing other than the default link function.

## 2 Binary Logistic Regression

3 **Binary logistic regression** is a technique useful for predicting a binary response variable. Examples of binary responses include

- 5 • Whether or not an individual defaults on a loan.
- 6 • Whether or not an option finishes “in the money.”
- 7 • Whether or not a bank will fail.
- 8 • Whether or not a trade was fraudulent.

9 Such response variables are often **coded** using 0 (“failure”) and 1 (“success”) for the two possible outcomes, but it should be clear that 10 this is arbitrary.

12 The predictors one would use will be of the same type that we’ve 13 seen previously: A mix of continuous and categorical variables. But, 14 it should also be clear that when  $Y$  can equal either only 0 or 1, it is 15 not a good idea to fit a model of the form

$$16 Y = \beta_0 + \sum_{i=1}^p \beta_i x_i + \epsilon. \quad ]$$

17 (Although, keep in mind that software will not complain if you try 18 to fit such a model.)

## 1 The Model

- 2 Consider an event of interest with probability  $0 < p < 1$ . We define  
 3 the **odds** for that event as

4

$$\text{odds} = \frac{\text{Probability of success}}{\text{Probability of failure}} = \frac{p}{1-p}.$$

- 5 Naturally, the **log odds**<sup>1</sup> (also called the **logit function**) is defined as

6

$$\text{logit}(p) = \text{log odds} = \log\left(\frac{p}{1-p}\right).$$

- 7 Note that while the odds is strictly positive, the log odds can take  
 8 any real value. The log odds also behaves nicely as a function of  $p$ : It  
 9 is symmetric around  $p = 0.5$ ; see Figure 1.

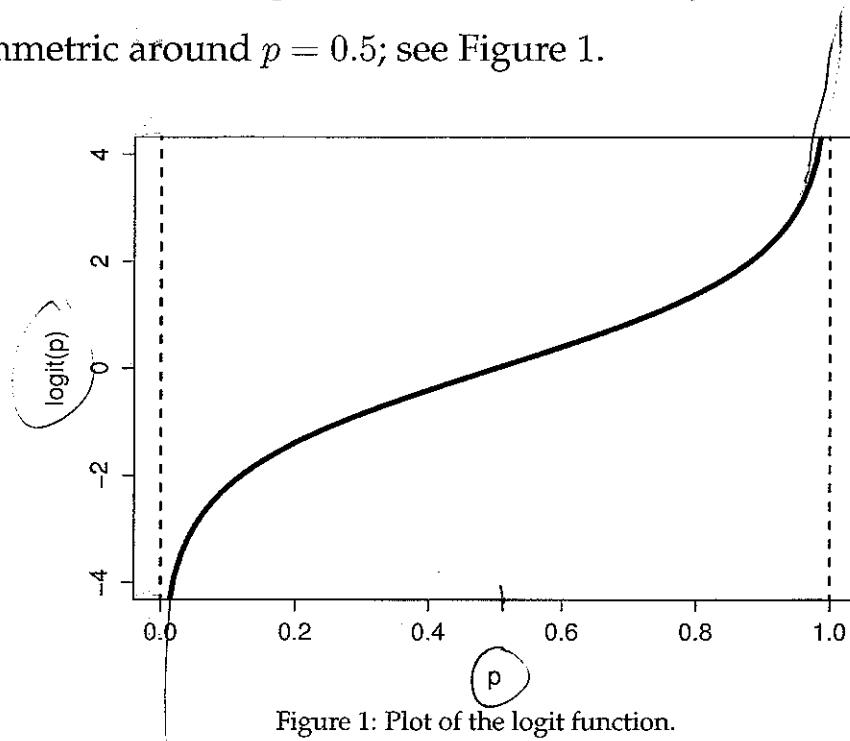


Figure 1: Plot of the logit function.

<sup>1</sup>Recall that we are always working with the "natural logarithm."

- 1 The logit function forms the basis for logistic regression. The model  
 2 can be stated as follows.

$$\text{logit}(P(Y=1)) = \mathbf{X}^T \boldsymbol{\beta}.$$

3 The binary response  $Y$  equals one with probability  $p$ , and

4 
$$\text{logit}(p) = \beta_0 + \sum_{i=1}^p \beta_i x_i.$$

*← # of predictors*

5 Responses from different objects are assumed to be independent.

- 6 It is important to note that the model is predicting the **logit of the probability that the response equals one**. This can be turned back into a probability by inverting the logit function. The inverse of the logit function is called the **logistic function**, and can be shown to be

7 
$$\text{logistic}(t) = \frac{1}{1 + e^{-t}}.$$

- 8 In other words, our model predicts that

9 
$$p = \frac{1}{1 + \exp(-(\beta_0 + \sum_{i=1}^p \beta_i x_i))}$$

10 
$$P = \frac{1}{1 + \exp(-\mathbf{X}^T \boldsymbol{\beta})}$$

- 11 and

12 
$$1 - p = \frac{\exp(-(\beta_0 + \sum_{i=1}^p \beta_i x_i))}{1 + \exp(-(\beta_0 + \sum_{i=1}^p \beta_i x_i))} = \frac{1}{1 + \exp(\beta_0 + \sum_{i=1}^p \beta_i x_i)}.$$

- 1 See Figure 2 for a plot of the logistic function. Note how the linear
- 2 predictor can take any value on the real line, and it is mapped back
- 3 into the interval  $(0, 1)$  to create the probability.

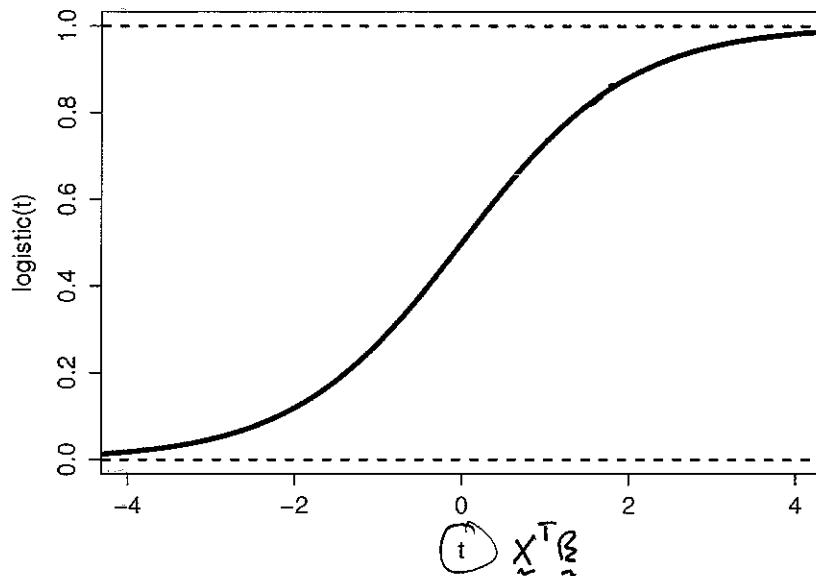


Figure 2: Plot of the logistic function.

- 4 **Exercise:** Explain how this model fits into the GLM framework.

5 \_\_\_\_\_

6 \_\_\_\_\_

7 \_\_\_\_\_ **Text**

8 \_\_\_\_\_

9 \_\_\_\_\_

10 \_\_\_\_\_

11 \_\_\_\_\_

12 \_\_\_\_\_

## AIC Calculations in R

$$Y_i \sim N(m(\beta, x_i), \sigma^2) \quad \text{e.g. } m(\beta, x_i) = x_i^T \beta$$

log likelihood:

$$-\frac{n}{2} \log(2\pi) - \frac{n}{2} \log(\sigma^2) - \sum_{i=1}^n \frac{(y_i - m(\beta, x_i))^2}{2\sigma^2}$$

-2 log-likelihood:

$$n \log(2\pi) + n \log(\sigma^2) + \sum_{i=1}^n \frac{(y_i - m(\beta, x_i))^2}{\sigma^2}$$

evaluated at MLE:

$$\underbrace{n \log(2\pi) + n \log\left(\frac{\text{RSS}}{n}\right) + n}_{F}$$

AIC() returns the full expression plus 2 times the number of parameters (including  $\sigma^2$ )

extract AIC() returns only  $n \log\left(\frac{\text{RSS}}{n}\right)$  plus 2 times the number of  $\beta$  parameters

Be careful! Make sure using consistent functions when comparing.

## The Design Matrix $\underline{X}$

Recall that the predictors are loaded into the columns of  $\underline{X}$ . Mathematically,  $\hat{Y} = \underline{X} \underline{B}$

$$\underline{B} = \begin{bmatrix} B_0 \\ B_1 \\ \vdots \\ B_p \end{bmatrix}$$

So, first column of  $\underline{X}$  is filled with 1's (unless excluding intercept)

What happens with factors?

A factor with  $k$  levels adds  $k-1$  columns to  $\underline{X}$

Assume "0" is the baseline category.

A column is added corresponding to level  $i=1, \dots, k-1$

For column  $i$ , the  $j$ th entry is ~~is~~ 1 if the  $j$ th obs. is level  $i$  for this factor. It's 0 otherwise

$$X = \begin{bmatrix} 1 & 1.2 & 0 & 0 \\ 1 & 3.4 & 0 & 0 \\ \vdots & \vdots & 0 & 0 \\ \vdots & \vdots & 1 & 0 \\ \vdots & \vdots & 1 & 0 \\ 1 & 7.8 & 0 & 1 \\ \vdots & \vdots & 0 & 1 \\ B_0 & B_1 & B_2 & B_3 \end{bmatrix} \quad \begin{array}{l} X_2 = 0 \\ X_2 = 1 \\ X_2 = 2 \end{array}$$

With `bestglm()`, could build design matrix (without column of 1's) and pass that in.

Would avoid slow calculations when using `factor()`.

In fact, `IC = "LOGCV"` did not even work.

But: `bestglm()` will not recognize certain columns are "tied together"

But: Can specify `TopModels` to be large, and hope that one of the models does not have this problem

- 1 See Figure 2 for a plot of the logistic function. Note how the linear predictor can take any value on the real line, and it is mapped back into the interval  $(0, 1)$  to create the probability.

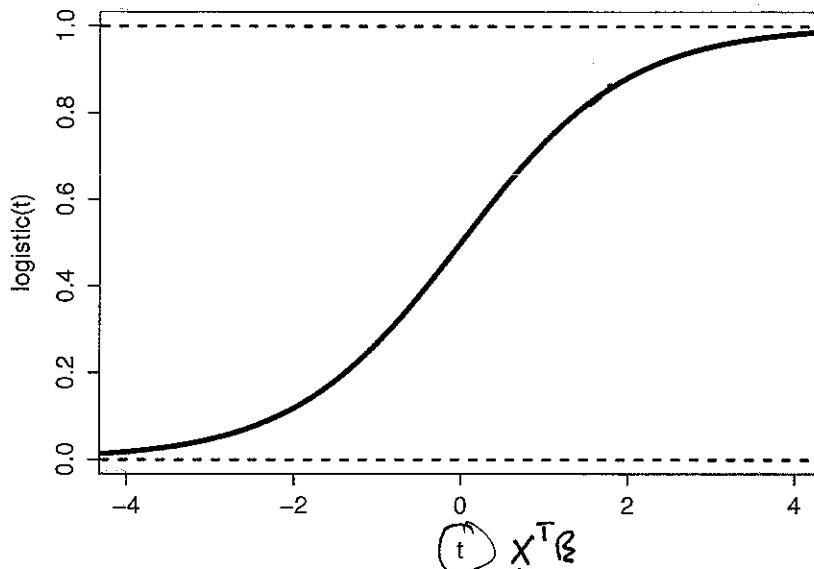


Figure 2: Plot of the logistic function.

- 4 **Exercise:** Explain how this model fits into the GLM framework.

5 we are assuming  $Y_i$  is binomial ( $n=1, p_i$ )  
 6 i.e. family = "binomial"

7 Assuming  $\text{logit}(p_i) = \tilde{X}_i^T \tilde{\beta}$   
 8  $\text{logit}(E(Y_i)) = \tilde{X}_i^T \tilde{\beta}$ , i.e.  $E(Y_i) = \text{logistic}(\tilde{X}_i^T \tilde{\beta})$

10 So,  $\text{logit}()$  is the link function

## Example: Predicting Bank Failures

- 2 This example is based on Olmeda and Fernandez (1997). In this pa-
- 3 per, the researchers utilized data on Spanish banks from 1977 to 1985.
- 4 During this period many banks failed; the goal is to relate the failure
- 5 to several economic health indicators. In particular, the authors use
- 6 the following predictors, each of which is a ratio:
  
- 7 1. current assets/total assets
- 8 2. (current assets - cash)/total assets
- 9 3. current assets/loans
- 10 4. reserves/loans
- 11 5. net income/total assets
- 12 6. net income/total equity capital
- 13 7. net income/loans
- 14 8. cost of sales/sales
- 15 9. cash flow/loans
  
- 16 The response variable is binary: failure or non-failure of the bank.  
"Success"  
↑

- It is a good idea to start by looking at the distribution for each of the predictors, and consider if any transformations would be suitable.
- This is shown as Figure 3.

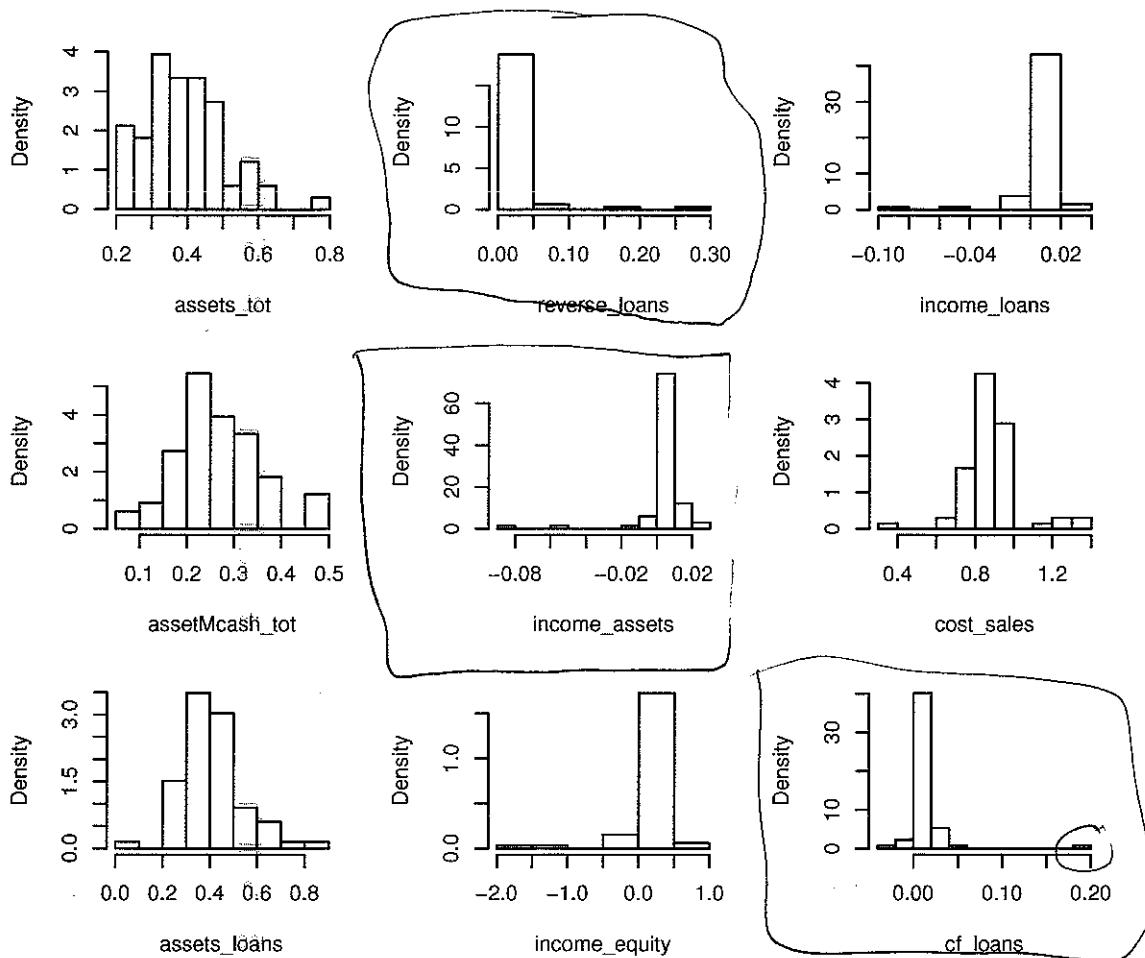


Figure 3: Histograms of the nine predictors.

- The main thing to look for are predictors with highly skewed distributions. Recall that we discussed that the extreme points in this distribution can become high leverage points in the regression. Ideally, these distributions would have a more “normal” shape.

- 1 For strictly positive variables, the log transform is the standard tool
- 2 for removing long right tails, as seen in variable `reverse_loans`.
  
- 3 For general variables, we can apply the Yeo-Johnson Transformation
- 4 described in our discussion of the linear model. There, we used it to
- 5 transform the response, but there is no reason why we could not also
- 6 use it to transform predictors.
  
- 7 In R, the command we need is `powerTransform()`, part of pack-
- 8 age `car`, with `family = "yjPower"`. In particular, we want the
- 9 `roundlam` component of the output, since this gives us the optimal
- 10 transform.
  
- 11 For instance, if we run

```
12 > lambda = powerTransform(income_assets,  
13           family="yjPower")$roundlam
```

- 14 we will get the optimal transformation power for transforming the
- 15 variable `income_assets`. (Here, by "optimal" it is meant that the
- 16 distribution is as close to normal as possible.)
  
- 17 This transformation is then applied using the function `yjPower()`:

```
18 > bankdata$income_assets_trans =  
19           yjPower(income_assets, lambda)
```

- 1 It was decided that the variables `reverse_loans`, `income_assets`,
- 2 `income_equity`, `income_loans`, and `cf_loans` would benefit from
- 3 transformation. Only the first of these is strictly positive.
- 4 This is the full R code used to construct the transformed variables.

```
5 # Read in the data
6
7 bankdata = read.table(
8     "http://www.stat.cmu.edu/~cschafer/MSCF/OlmedaData.txt")
9
10 # Name the variables
11
12 names(bankdata) = c('assets_tot','assetMcash_tot','assets_loans',
13                     'reverse_loans','income_assets','income_equity',
14                     'income_loans','cost_sales','cf_loans','default')
15
16 # Change the default variable into 0/1
17
18 bankdata$default = as.numeric(bankdata$default == "Failed")
19
20 # This function allows for direct reference to the named columns of
21 # the data frame bankdata. So, instead of writing bankdata$assets_tot,
22 # I can just write assets_tot
23
24 attach(bankdata)
25
26 # We need the package car for the Yeo-Johnson Transformation
27
28 library(car)
29
```

```
1 # Apply all of the transformations
2
3 bankdata$reverse_loans_trans = log(reverse_loans)
4
5 lambda = powerTransform(income_assets, family="yjPower")$roundlam
6 bankdata$income_assets_trans = yjPower(income_assets, lambda)
7
8 lambda = powerTransform(income_equity, family="yjPower")$roundlam
9 bankdata$income_equity_trans = yjPower(income_equity, lambda)
10
11 lambda = powerTransform(income_loans, family="yjPower")$roundlam
12 bankdata$income_loans_trans = yjPower(income_loans, lambda)
13
14 lambda = powerTransform(cf_loans, family="yjPower")$roundlam
15 bankdata$cf_loans_trans = yjPower(cf_loans, lambda)
```

16 **Exercise:** What is a drawback of using the Yeo-Johnson Transformation  
17 for this purpose?

18 It makes interpreting the predictors more difficult

19  
20 E.g. What is the effect on the expected  
21 response due to <sup>increasing</sup> changing income\_assets  
22 by 1?

23

24

25

Figure 4 shows the results of the transformations.

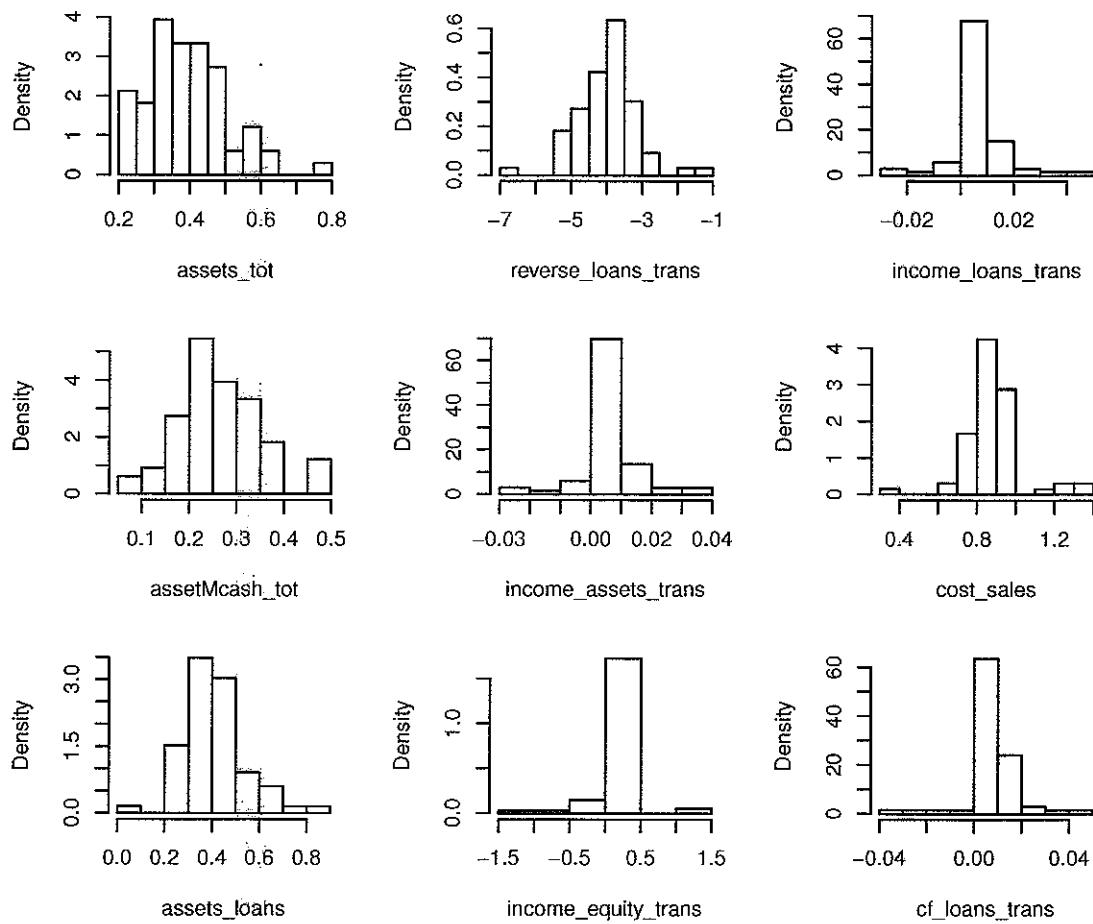


Figure 4: Histograms of the nine predictors following transformation.

It seems that we have achieved the objective that was set out: The distributions of the five transformed predictors are much less skewed.

## 1 Fitting the Model

- 2 In R, we use `glm()` with `family = binomial` for fitting the logistic regression model:

```
4 > bankglm = glm(default ~ assets_tot + assetMcash_tot  
5   + assets_loans + reverse_loans_trans  
6   + income_assets_trans + income_equity_trans  
7   + income_loans_trans + cost_sales  
8   + cf_loans_trans, family=binomial,  
9   data=bankdata)
```

, The output is as follows

```
2 Call:  
3 glm(formula = default ~ assets_tot + assetMcash_tot + assets_loans +  
4     reverse_loans_trans + income_assets_trans + income_equity_trans +  
5     income_loans_trans + cost_sales + cf_loans_trans, family = binomial,  
6     data = bankdata)  
7  
8 Deviance Residuals:  
9      Min        1Q     Median        3Q        Max  
10 -2.36368 -0.46052 -0.07788  0.65940  2.74881  
11  
12 Coefficients:  
13             Estimate Std. Error z value Pr(>|z|)  
14 (Intercept) -5.1381    6.6383 -0.774  0.43892  
15 assets_tot -193.4868   74.9281 -2.582  0.00981 **  
16 assetMcash_tot  8.8430   10.3954  0.851  0.39495  
17 assets_loans  175.3455   71.4758  2.453  0.01416 *  
18 reverse_loans_trans -1.2869   0.7419 -1.735  0.08281 .  
19 income_assets_trans 3710.9400  2178.9585  1.703  0.08855 .  
20 income_equity_trans -0.5040    2.7497 -0.183  0.85457  
21 income_loans_trans -3875.6648  2207.2168 -1.756  0.07910 .  
22 cost_sales       3.0938    5.9830  0.517  0.60509  
23 cf_loans_trans   -20.3434   91.8446 -0.221  0.82470  
24 ----  
25 Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1  
26  
27 (Dispersion parameter for binomial family taken to be 1)  
28  
29 Null deviance: 90.523 on 65 degrees of freedom  
30 Residual deviance: 47.215 on 56 degrees of freedom  
31 AIC: 67.215  
32  
33 Number of Fisher Scoring iterations: 8
```

## 1 Model Selection

2 It seems particularly important to perform variable selection in this  
3 example since the number of objects being used for fitting (66) is not  
4 that much larger than the number of parameters in the full model  
5 (10). Hence, there is more risk of overfitting.

6 We will use AIC as the criterion for choosing a model. As before, we  
7 will use the `step()` function:

8 > `finalmod = step(bankglm, direction="both")`

9 The result is that the model with lowest AIC is as follows

```
10          Estimate Std. Error z value Pr(>|z|)  
11 (Intercept) -3.7657    3.1190 -1.207  0.22730  
12 assets_tot  -204.7483   72.4380 -2.827  0.00471 **  
13 assets_loans 192.2874   68.7388  2.797  0.00515 **  
14 reverse_loans_trans -1.6196    0.6784 -2.387  0.01697 *  
15 income_assets_trans 4695.1767  2052.2646  2.288  0.02215 *  
16 income_loans_trans -4891.4063  2105.4932 -2.323  0.02017 *  
17 ---  
18 Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1  
19  
20 (Dispersion parameter for binomial family taken to be 1)  
21  
22 Null deviance: 90.523  on 65  degrees of freedom  
23 Residual deviance: 48.887  on 60  degrees of freedom  
24 AIC: 60.887
```

## 1 Diagnostic Plots

- 2 For binary logistic regression the fitted values  $\hat{y}$  are the estimated
- 3 probabilities of "success" for each of the objects.
- 4 **Exercise:** What is  $\hat{y}$  as a function of the linear predictor?

5 
$$\hat{y} = \hat{p} = \text{logistic} \left( \hat{\beta}_0 + \hat{\beta}_1 x_1 + \dots + \hat{\beta}_p x_p \right)$$

6

7 
$$\hat{p} = \frac{1}{1 + \exp \left( -(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \dots + \hat{\beta}_p x_p) \right)}$$

8

9

10

- 11 Since there are only two possible outcomes, and there is no "model
- 12 error" term, residuals are not particularly useful for this model. The
- 13 most useful diagnostic plot is a plot of the observed responses versus
- 14 the fitted values, i.e.,  $y$  versus  $\hat{y}$ .

- 1 See Figure 5 for this plot for our model. Note that the response has  
 2 been "jittered" so that all of the points are not either zero or one; this  
 3 simply aids in reading the plot.

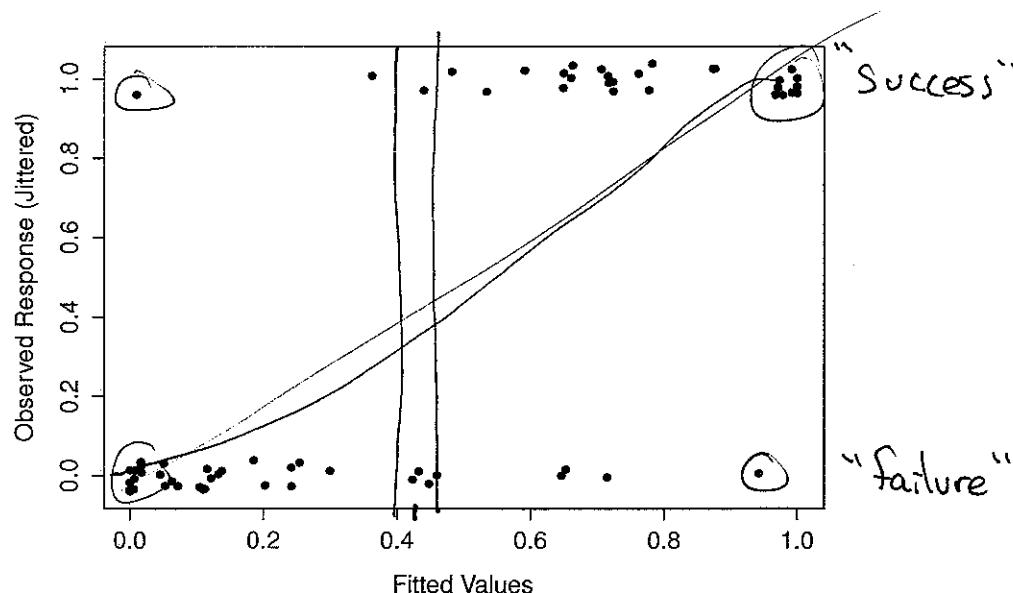


Figure 5: Plot of response versus fitted values for our model.

- 4 **Exercise:** If the model is correct, what should this plot look like?

5 Take any vertical slice. The proportion of  
 6 successes should approximately equal  
 7 the ~~center~~ center of that slice.

8 \_\_\_\_\_

9 \_\_\_\_\_

10 \_\_\_\_\_

11

1 This idea is formalized by the **Hosmer-Lemeshow test**. In this test,  
2 the range of fitted values if divided into  $b$  bins. (Often  $b = 10$ ). In  
3 each bin, one calculates both

- 4 1. the observed number of “successes”  $O_i$ , and  
5 2. the expected number of “successes” under the assumption that  
6 the model is correct  $E_i$ .

7 Then

8

$$\chi^2 = \sum_{i=1}^b \frac{(O_i - E_i)^2}{E_i}$$

9 has approximately the chi-squared distribution with  $b - 2$  degrees of  
10 freedom, under the null hypothesis that the model is correct.

11 If the model is incorrect, then the “observed” and “expected” will  
12 not agree, and  $\chi^2$  will be large.

13 Hence, the test rejects the null if  $\chi^2$  exceeds a threshold.

- 1 This test is implemented in R via the function `hoslem.test()` which
- 2 is part of the package `ResourceSelection`.

- 3 The syntax is

4 `> hoslem.test(bankdata$default,`

5 fitted.values(finalmod))

- 6 The output is as follows

7 Hosmer and Lemeshow goodness of fit (GOF) test

8

9 data: default, finalmod\$fit

10 X-squared = 7.7116, df = 8, p-value = 0.4621

$\chi^2$  b-2

- 11 **Exercise:** What is the conclusion?

12 Fail to reject  $H_0$ : model is correct.

13 "Fail to find evidence that model is incorrect."

16 Not: "Found evidence that model fits well."

17

- 1 We should also check for influential observations.
- 2 Figure 6 shows Cook's Distance for each of the 66 objects. There is
- 3 clearly one **very** influential observation: Object 35.

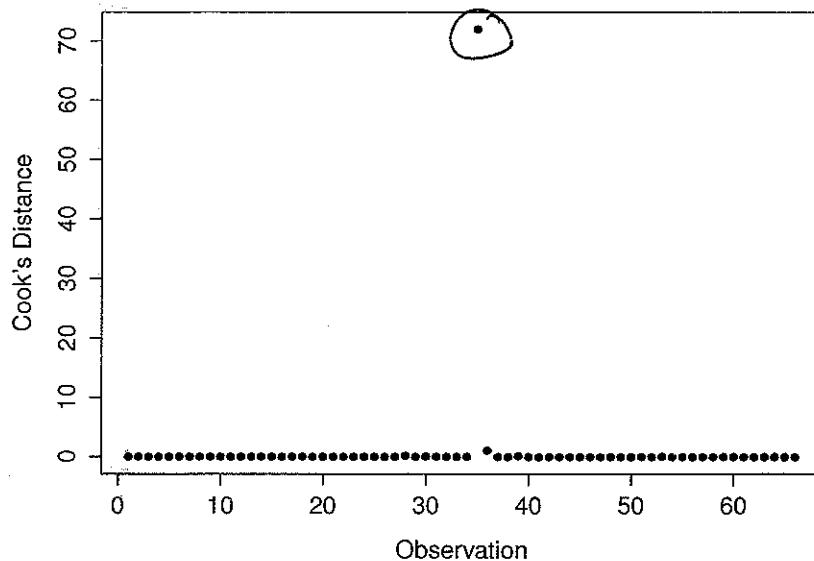


Figure 6: Plot of Cook's Distance for our model.

- 4 In order to identify the cause of the problem, one should inspect the
- 5 values for the predictors for this object. It is difficult visualize data
- 6 in five dimensions, but there are clever plots one can create, such as
- 7 the "star plot" shown as Figure 7.
- 8 The idea is simple: There is one star per object, and each star has five
- 9 "arms," one for each of the predictors. Such a plot can help us see
- 10 differences among the objects.

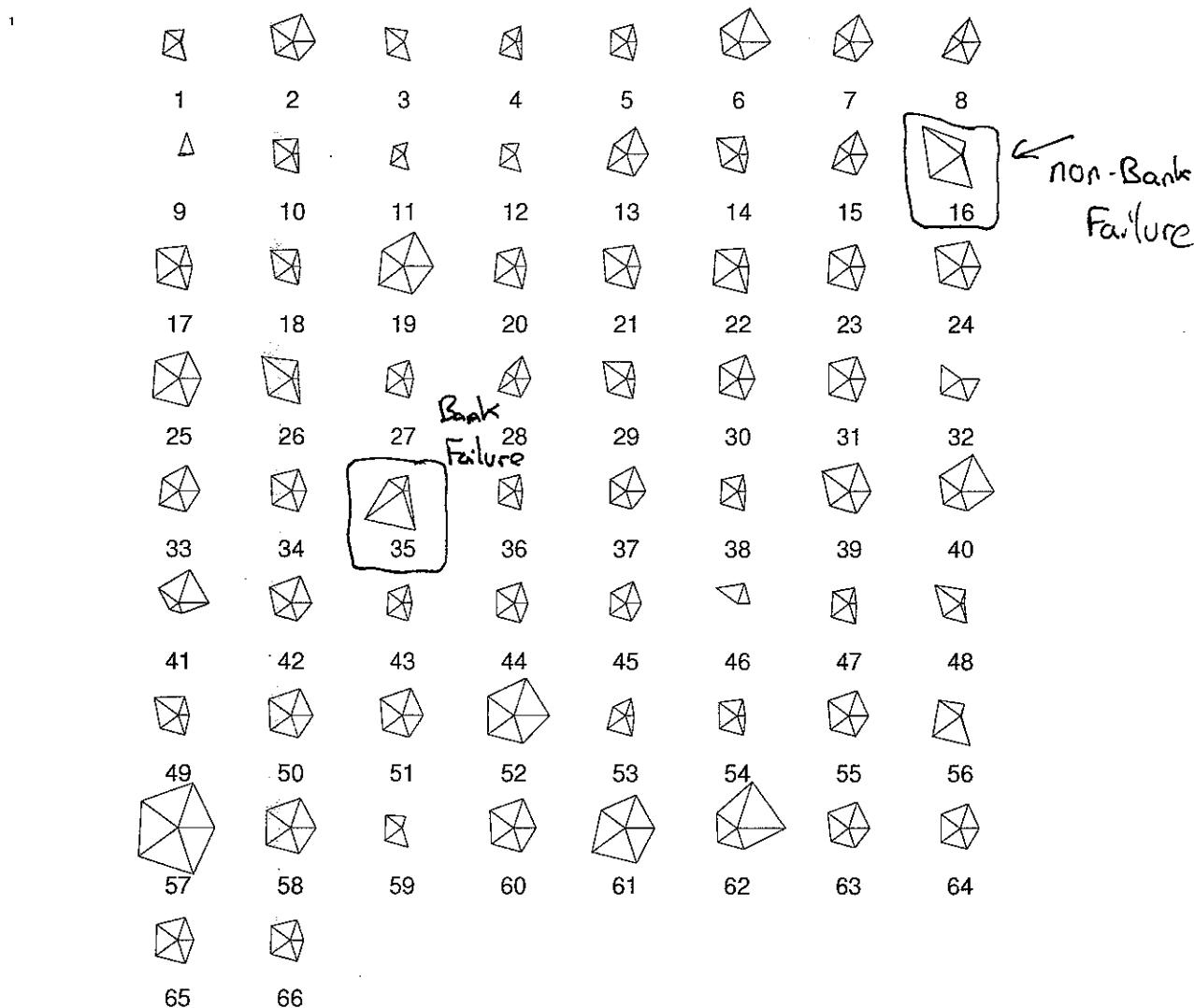


Figure 7: Star plot of predictors in the model. The length of the five "arms" indicate the size of the five predictors. The red stars are those banks that failed.

2 The R command used to create this plot is

3 > stars(bankdata[,c(1,3,11,12,14)],

4 labels=1:66, col.lines=(bankdata\$default+1))

faces() is an alternative

- 1 **Exercise:** Can you see anything unique about object 35?

2 It seems to be large on two predictors and  
3 small on another in a way unlike  
4 others that are bank failures. It appears  
5 to be different.

6

7

8

9

10

11

12

- 13 On the homework I will ask you to refit the model without this bank.

14 **An important point:** When the model is refit, the variable selection  
15 procedure should be redone. This object is clearly having a huge  
16 influence on the model, and that influence extends to the variables  
17 which are included. Indeed, you will find that the model is very  
18 different following the removal.

## 1 Interpreting the Parameters

- 2 In the normal linear model there is a very simple interpretation attached to each of the parameters: Increasing  $x_i$  by  $b$  adds  $b \times \hat{\beta}_i$  to the expected response.
- 5 In the binary logistic regression model, the interpretation is trickier.
- 6 Clearly, if  $\hat{\beta}_i > 0$  then the model is predicting that increasing  $x_i$  corresponds to increasing the probability of success. And if  $\hat{\beta}_i < 0$  then the model is predicting that decreasing  $x_i$  corresponds to increasing the probability of success.
- 10 More technically, we know that if we increase  $x_i$  by  $b$ , then the log odds of success increases by  $b \times \hat{\beta}_i$ . Or, we could say that increasing  $x_i$  by  $b$  multiplies the odds of success by  $\exp(b \times \hat{\beta}_i)$ .

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \underline{x}_i^T \underline{\beta}$$

$$\frac{p}{1-p} = \exp(\underline{x}_i^T \underline{\beta})$$

$$\# \exp((\underline{x}_i + b)^T \underline{\beta}) = \exp(\underline{x}_i^T \underline{\beta}) \exp(b^T \underline{\beta})$$

- 1 **Exercise:** Suppose that  $x_i$  is a binary **predictor** in a binary logistic
- 2 regression model. We find that  $\hat{\beta}_i = 2.3$ . What is a practical interpre-
- 3 tation of this estimate?

4 The odds of "success" is multiplied by  
5  $\exp(2.3)$  when switch from level 0 to  
6 level 1. (i.e.  $b=1$  in previous exercise)

7 If  $\hat{\beta}_i = 0$ ,  $\exp(\hat{\beta}_i) = 1$ , i.e. no change  
8 in odds.

9

---

10

---

11

---

12

---

13

---

14

## 1 Making Predictions

- 2 The logistic regression model yields predictions of the **probability of success** for a future observation.
- 4 We know that the actual observation will be either 0 or 1; it does not make sense to construct a "prediction interval" in this case.

$$(L, U)$$

- 6 Yet, there is still uncertainty in the estimate of the probability, due to uncertainty in the parameter estimates.

- 8 The R command:

9 > predict(finalmod, type="response", se.fit=T)

- 10 returns the vector of predicted probabilities, along with standard errors on those predictions.

- 12 The inclusion of type="response" is critical; otherwise, the function returns the estimate of the linear predictor.

---

## 2 Multinomial Logistic Regression



3 The **multinomial logistic regression** model is a generalization of the  
4 binary logistic regression model to the case where the response can  
5 be one of  $K$  possible outcomes.

6 To formulate this model, start by recalling that for the binary logistic  
7 regression model. There the possible outcomes were  $Y = 1$  ("success") and  $Y = 0$  ("failure"), the model was that

$$9 \quad \log\left(\frac{P(Y = 1)}{P(Y = 0)}\right) = \beta_0 + \sum_{j=1}^p \beta_j x_j.$$

10 Now let  $0, 1, \dots, (K - 1)$  denote the possible outcomes. Keep in mind  
11 that these are simply labels, the numbers carry no significance. Outcome  
12 "0" will serve as the "baseline outcome" against which all others are compared.

14 In particular, we assume that

$$15 \quad \log\left(\frac{P(Y = k)}{P(Y = 0)}\right) = \beta_{0k} + \sum_{j=1}^p \beta_{jk} x_j, \quad \text{for } k = 1, 2, \dots, (K - 1).$$

- 1 Note that there is a set of parameters for each of the outcomes  $k =$
  - 2  $1, 2, \dots, (K-1)$ . (With the appropriate use of indicator functions, this
  - 3 could be written as a single function, but it is messy.)
- 4 Hence, the total number of parameters can grow quite rapidly. Keep
  - 5 in mind that the sample size  $n$  should be somewhat larger than the
  - 6 number of parameters in order to have confidence that we are not
  - 7 overfitting.
- 8 **Exercise:** Suppose that all of the  $\beta$  parameters equal zero. What
  - 9 would that imply? What if  $\beta_{jk} = 0$  for  $j \geq 1$ , but  $\beta_{0k} \neq 0$ ?

10 \_\_\_\_\_

11 \_\_\_\_\_

12 \_\_\_\_\_

13 \_\_\_\_\_

14 \_\_\_\_\_

15 \_\_\_\_\_

16 \_\_\_\_\_

17 \_\_\_\_\_

18 \_\_\_\_\_

19 \_\_\_\_\_

20 \_\_\_\_\_

- 1 In R, one can fit this model using function `multinom()` from the
- 2 package `nnet`. `glm()` is **not** able to fit the model. The output is quite
- 3 similar.
  
- 4 An example of the use of this model can be found in "Forecasting
- 5 Stock Performance in Indian Market using Multinomial Logistic Re-
- 6 gression" by Upadhyay, et al.<sup>2</sup>
  
- 7 There, the authors divide the performance of a stock relative to the
- 8 market into categories "good," "average," and "poor." The category
- 9 is predicted using seven financial ratios, including book value, earn-
- 10 ings per share, and percentage change in operating profit.
  
- 11 This work illustrates that in some cases it can be advantageous to
- 12 take a response variable which is naturally continuous and translate
- 13 it into a categorical variable.

---

<sup>2</sup>*Journal of Business Studies Quarterly*, Volume 3, page 16.

- 1 **Exercise:** Discuss the advantages and drawbacks of taking this approach of converting a continuous response into a categorial variable.
- 2
- 3

4 \_\_\_\_\_

5 \_\_\_\_\_

6 \_\_\_\_\_

7 \_\_\_\_\_

8 \_\_\_\_\_

9 \_\_\_\_\_

10 \_\_\_\_\_

11 \_\_\_\_\_

12 \_\_\_\_\_

13 \_\_\_\_\_

14 \_\_\_\_\_

15 \_\_\_\_\_

16 \_\_\_\_\_

17 \_\_\_\_\_

18 \_\_\_\_\_

19 \_\_\_\_\_

**Poisson Regression**

- <sup>3</sup> In the **Poisson regression** model the response is assumed to follow
- <sup>4</sup> the Poisson distribution with mean which is a function of the covari-
- <sup>5</sup> ates.
- <sup>6</sup> This is useful for cases in which the response is a count.
- <sup>7</sup> Formally, we'd say the following:
- <sup>8</sup> In the Poisson regression model we assume that  $Y_i$  has the
- <sup>9</sup> Poisson distribution with mean  $\lambda_i = \exp(\beta^T \mathbf{x}_i)$ . We assume
- <sup>10</sup> that each of the  $Y_i$  are independent.
- <sup>11</sup> Clearly, this is a GLM with the log link function.
- <sup>12</sup> This can be fit in R using `family=poisson`.

- 1 In “Multiple Bids as a Consequence of Target Management Resis-  
 2 tance: A Count Data Approach,”<sup>3</sup> Jaggia and Thosar use this model  
 3 to predict the number of takeover bids after the initial bid aimed at  
 4 the firm. The predictors “comprise both target management actions  
 5 and firm specific characteristics.”
- 6 The results of the fit are shown in Table 3 in the paper, reproduced  
 7 here as Figure 8.

**Table 3. Poisson model estimation results.**

A Poisson count data model is estimated where the dependent variable represents the number of bids (count) after the initial bid received by the target firm. The explanatory variables considered are various target management actions and firm specific characteristics.

Variable	Coefficient	t-Statistics
Constant	0.986	1.846**
<i>Target Management Actions</i>		
Legal Defense	0.260	1.723**
Real Restructuring	-0.195	-1.015
Financial Restructuring	0.074	0.341
White Knight	0.481	3.030*
<i>Firm Specific Characteristics</i>		
Initial Bid Premium	-0.677	-1.798**
Institutional Holdings	-0.361	-0.853
Size	0.178	2.970*
Size Squared	-0.007	-2.416*
Regulation	-0.029	-0.183

\*represents significance at  $\alpha = 0.01$

\*\*represents significance at  $\alpha = 0.10$

**Tests of Poisson Specification (Mean-Variance Equality)**

Standard Test Statistic = -1.02 ( $p = 0.3078$ )

Dean-Lawless Adjusted Test Statistic = -0.30 ( $p = 0.7642$ )

Figure 8: Table 3 from Jaggia and Thosar (1993).

<sup>3</sup>Review of Quantitative Finance and Accounting, Volume 3, page 447.

## 1 Overdispersion in Poisson Regression

- 2 The Poisson distribution assumes equality of the mean and the variance; this is why this GLM does not have a dispersion parameter.
- 4 Nevertheless, this is a very restrictive assumption, and greatly diminishes the applicability of the model.
- 6 If it is the case that the variances for the  $Y_i$  are somewhat larger than their expected values, then there is said to be **overdispersion** in the model.
- 9 A standard test statistic for diagnosing this is the following:

$$10 T = \left( \sum_i [(y_i - \hat{y}_i)^2 - y_i] \right) / \left( 2 \sum_i \hat{y}_i^2 \right)^{1/2}$$

- 11 **Exercise:** How does the numerator of  $T$  behave with overdispersion?

12 \_\_\_\_\_

13 \_\_\_\_\_

14 \_\_\_\_\_

15 \_\_\_\_\_

16 \_\_\_\_\_

17 \_\_\_\_\_

- 1 If the Poisson model is correct then  $T$  has the chi-squared distribution with a single degree of freedom. Large values of  $T$  would lead one to conclude there is overdispersion.
- 4 If overdispersion is present, one can use `family=quasipoisson` instead. The details are technical, but this performs a MLE-like procedure in which an additional dispersion parameter is allowed in the model. The parameter estimates are not themselves changed, but the standard errors are adjusted to account for the overdispersion.
- 9 Another alternative is to assume that the response has the **negative binomial** distribution instead of the Poisson. This is implemented in R using the function `glm.nb()` which is part of the package MASS.

## **2 Gamma Regression**

- <sup>3</sup> One can assume that the response has the gamma distribution via
- <sup>4</sup> **gamma regression**. Recall that a gamma random variable has sup-
- <sup>5</sup> port on the entire positive real line. Hence, this model may make
- <sup>6</sup> sense in certain cases where the response is positive (e.g., volatility).
  
- <sup>7</sup> The standard recommendation is to use the `log` as the link function
- <sup>8</sup> in this case. In order to get R to use this link function (it is not the
- <sup>9</sup> default), you need to specify `family=Gamma(link="log")`.
  
- <sup>10</sup> Other choices for the link are `inverse` (the default) and `identity`,
- <sup>11</sup> but these both have the drawback of forcing the linear predictor to
- <sup>12</sup> be positive.
  
- <sup>13</sup> While this model may be useful in some cases, it performs quite sim-
- <sup>14</sup> ilarly to using a standard normal linear model with a log transform
- <sup>15</sup> of the response. Hence, it is not widely used.

<sup>1</sup> **Binomial Regression**

- <sup>3</sup> Binary logistic regression assumed that the response was Bernoulli,
- <sup>4</sup> but this could be generalized to a model which assumes that the re-
- <sup>5</sup> sponse is Binomial.
  
- <sup>6</sup> To get R to fit such a model, one would use `family=binomial`, but
- <sup>7</sup> with a couple adjustments:
  
- <sup>8</sup> 1. The response should be transformed into the proportion of suc-
- <sup>9</sup> cesses, not the count of successes. So, if `y` held the data assumed
- <sup>10</sup> to be from a  $\text{binomial}(n_i, p_i)$  distribution, then one would use
- <sup>11</sup> `y/n` as the response fed into the model object. Note that `n` can be
- <sup>12</sup> a vector, i.e., the number of trials can vary from one object to the
- <sup>13</sup> next.
  
- <sup>14</sup> 2. Specify `weights = n`, where again `n` holds the number of trials
- <sup>15</sup> for each object.
  
- <sup>16</sup> Overdispersion can also be a problem for this model. There is a
- <sup>17</sup> `family=quasibinomial` option.

## Part 5: High-Dimensional Variable Selection

- <sup>1</sup> Text references: §12.6 in Ruppert, §6.2 and 6.3.1 in ISL
- <sup>2</sup> This part is concerned with methods of doing variable selection which
- <sup>3</sup> are especially useful for cases in which there are a large number of
- <sup>4</sup> predictors.
- <sup>5</sup> We begin with some motivation.
- <sup>6</sup> As mentioned already, there is often a general preference for smaller
- <sup>7</sup> models, as we typically believe that a simple model is better than a
- <sup>8</sup> complex one. Simple models are less prone to overfitting.
- <sup>9</sup> In addition, as the number of predictors increases, there is more of a
- <sup>10</sup> risk of **multicollinearity**, a condition in which two or more predic-
- <sup>11</sup> tors are strongly linearly related to one another.
- <sup>12</sup> When multicollinearity is present it can be very difficult to distin-
- <sup>13</sup> guish their individual influence on the response. Also, in extreme
- <sup>14</sup> cases, multicollinearity produces numerical problems in the finding
- <sup>15</sup> of the parameter estimates.
- <sup>16</sup>

- 1 The following diagnostic is useful in this regard.
- 2 Suppose that there are  $p$  predictors in the model. In order to quan-
- 3 tify the relationship between  $x_k$  and the other  $p - 1$  predictors, we
- 4 fit a linear regression model with  $x_k$  as the response and the other
- 5 variables as the predictors:

$$6 \quad x_k = \beta_0 + \sum_{j \neq k} \beta_j x_j + \epsilon$$

- 7 If  $x_k$  can be well-predicted by the other variables, then we have mul-
- 8 ticollinearity.

- 9 Note that multicollinearity does not require  $x_k$  to be related to any
- 10 one of the other predictors, but merely related to a *linear combination*
- 11 of the other predictors.

- 12 Suppose this regression is fit, and the coefficient of determination is
- 13 obtained, call it  $R_k^2$ . It directly follows that  $R_k^2$  close to one is a sign of
- 14 multicollinearity.

- 1 We define the **variance inflation factor** for the  $k^{th}$  predictor to be

2

$$\text{VIF}_k = \frac{1}{1 - R_k^2}$$

3 A large VIF is a sign of multicollinearity.

- 4 Where does the name come from? In the case of the classic linear  
5 model, it can be shown that

6

$$V(\hat{\beta}_k) = \left( \frac{1}{1 - R_k^2} \right) \times \frac{\sigma^2}{(n - 1)s_{x_k}^2}$$

7

$$= \text{VIF}_k \times (V(\hat{\beta}_k) \text{ if } x_k \text{ was only predictor in model})$$

8

- 9
- 10 So,  $\text{VIF}_k$  is the amount by which the variance of the estimator for  
11  $\beta_k$  increases by including all of other predictors in the model. If  
12  $\text{VIF}_k = 4$ , for instance, then the variance of  $\hat{\beta}_k$  increases four times  
13 by including the other predictors.

- 14 VIF can be found in R by using the command `vif()` which is part  
15 of the package `car`. Apply `vif()` to the object returned by `lm()` or  
16 `glm()`.

- 17 VIF generalizes to the other GLM's in a natural way.

1 **Exercise:** Imagine that there is a pair of strongly correlated predictors,  $x_1$  and  $x_2$ . Why should the variance of  $\hat{\beta}_1$  increase greatly if  $x_2$  is added to the model?

2  $\wedge$

3 Note that the variance will be multiplied

4 by  $\frac{1}{1 - r_{12}^2}$   $\leftarrow$  This is  $R^2$  from model  $x_1 \sim x_2$

5 where  $r_{12}$  = sample correlation between  $x_1$  and  $x_2$

6 Heuristically, the model is "unsure" which  
7 of  $x_1$  and  $x_2$  to attribute the  
8 relationship with the response

9

10

11

- 1 If there are predictors with large VIF, then there is multicollinear-
- 2 ity, but it is not clear which predictor should be removed. A pair of
- 3 strongly correlated predictors will both have large VIF's. This does
- 4 not mean that both should be removed from the model.
  
- 5 Keep in mind that the VIF does not take into account the strength of
- 6 the relationship of either predictor with the response.
  
- 7 If two predictors are strongly correlated, then their respective corre-
- 8 lations with the response will be roughly equal, but one will have a
- 9 stronger relationship than the other. An intelligent method would
- 10 take this into consideration when making variable selection deci-
- 11 sions. Lasso
  
- 12 The two procedures we discuss here, principal components and lasso,
- 13 take very different approaches to dealing with a large number of pre-
- 14 dictors.

## 2 Principal Components Analysis (PCA)

- 3 Figure 1 shows three US Treasury **yield curves** from 2012.<sup>1</sup> Note that
- 4 there are 11 maturity dates.

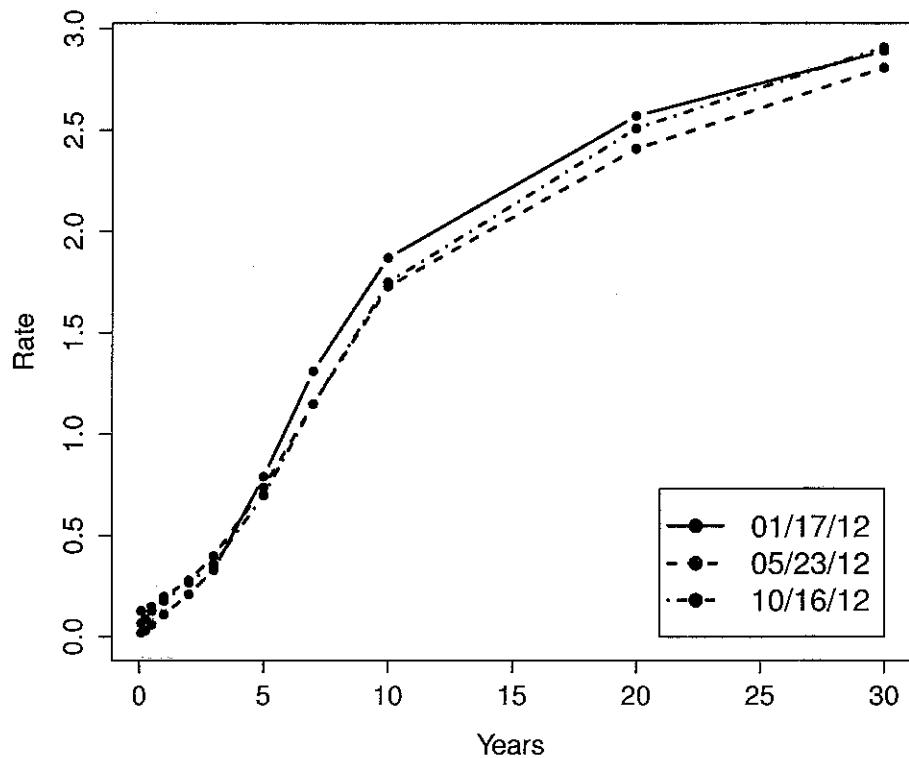


Figure 1: Three US Treasury yield curves from 2012.

- 5 There is important information encoded in these curves: one could
- 6 imagine, for instance, using the rates at a fixed point in time as pre-
- 7 dictors in a factor model.

---

<sup>1</sup>These data are from

<http://www.treasury.gov/resource-center/data-chart-center/interest-rates/Pages/TextView.aspx?data=yield>

- 1 Below are the VIFs for each of the 11 maturity dates. This is im-  
2 aging a regression model in which each of these 11 is included as a  
3 separate predictor in the regression. The response is irrelevant to this  
4 calculation.

```
5 yieldcurves[, 1] yieldcurves[, 2] yieldcurves[, 3] yieldcurves[, 4]
6 1.916965 5.403006 9.367883 7.877034
7 yieldcurves[, 5] yieldcurves[, 6] yieldcurves[, 7] yieldcurves[, 8]
8 12.530442 49.387423 259.817398 379.682987
9 yieldcurves[, 9] yieldcurves[, 10] yieldcurves[, 11]
10 550.377633 533.343288 165.581547
```

- 11 **Exercise:** Why is it not be a good idea to include the 11 different rates  
12 as individual predictors in a regression model?

13 Clear evidence of multicollinearity. Including  
14 all 11 is redundant, and also will  
15 lead to instability, (large variances,  
16 numerical problems)

17  
18 Want to consider "reducing the dimension  
19 of the predictor."

20

21

- 1 Instead of using the data in their raw form, we seek to construct a
- 2 new **representation** of the data, typically a **low-dimensional representation**,
- 3 that captures the important sources of variability.
  
- 4 A standard tool for this is **principal components analysis (PCA)**.

- 1 We'll start by considering a simple example with two predictors,  $x_1$   
 2 and  $x_2$ , and  $n = 100$ . Figure 2 shows the relationship between these  
 3 two predictors. The VIF for these predictors is 25.30376.

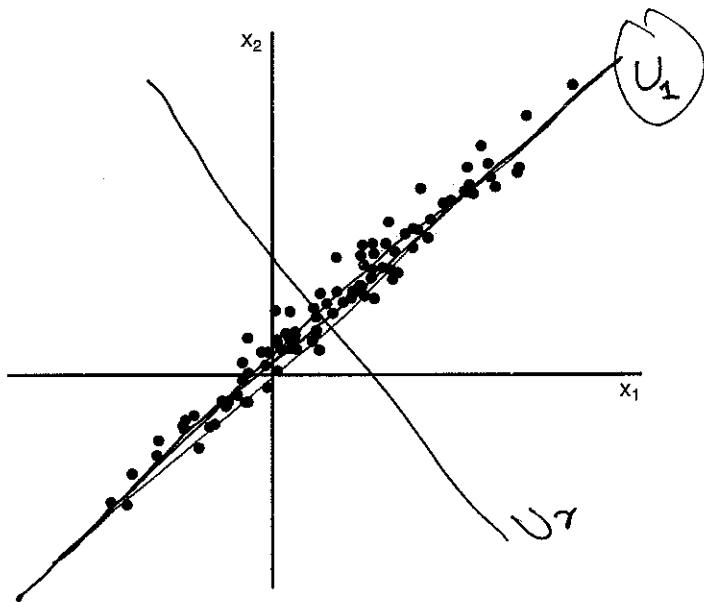


Figure 2: A sample of data in its original coordinate representation  $(x_1, x_2)$ .

- 4 **Exercise:** Explain why this coordinate system is not a "natural" way  
 5 of representing the data.

6 Clear relationship between predictors. Consider  
 7 a new axis system which reduces  
 8 correlation between predictors.

9 \_\_\_\_\_

10 \_\_\_\_\_

- 1 Figure 3 shows the same a sample as in Figure 2, but with a new axis
- 2 system shown as the red dashed line labelled  $u_1$  and  $u_2$ .
  
- 3 This new axis system is derived from the **principal components** of
- 4 the data.

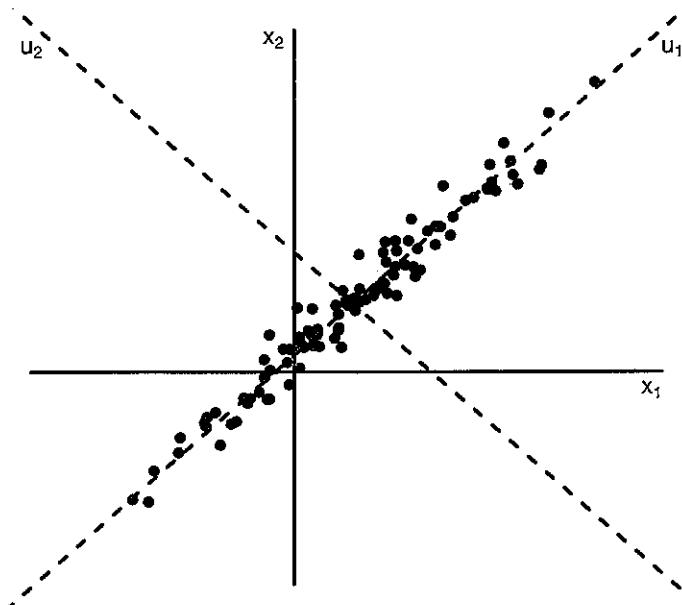


Figure 3: A sample of data in its original coordinate representation  $(x_1, x_2)$ , and in its transformed coordinate system found using PCA  $(u_1, u_2)$ .

- 1 The formal idea behind PCA is as follows.
- 2 Suppose we have observations of  $n$  of vectors  $\mathbf{x}_i$ , each of which is of dimension  $p$ :

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip})^T, \quad i = 1, 2, \dots, n.$$

In example,  
 $n = 250$   
 $p = 11$

- 3 Now we ask: What vector  $\mathbf{u}$  maximizes the sample variance of

$$v_i = \underbrace{\sum_{j=1}^p u_j x_{ij}}_{\text{subject to}} \quad i = 1, 2, \dots, n \quad v_1, v_2, \dots, v_{11}$$

- 4 subject to

$$\sum_{j=1}^p u_j^2 = 1. \quad \leftarrow \mathbf{u} \text{ is a direction} \quad u_1 = 1, u_2 = \dots = u_p = 0$$

- 5 This is the same as finding the first eigenvector of  $\hat{\Sigma}$ , the estimate of the population covariance matrix.

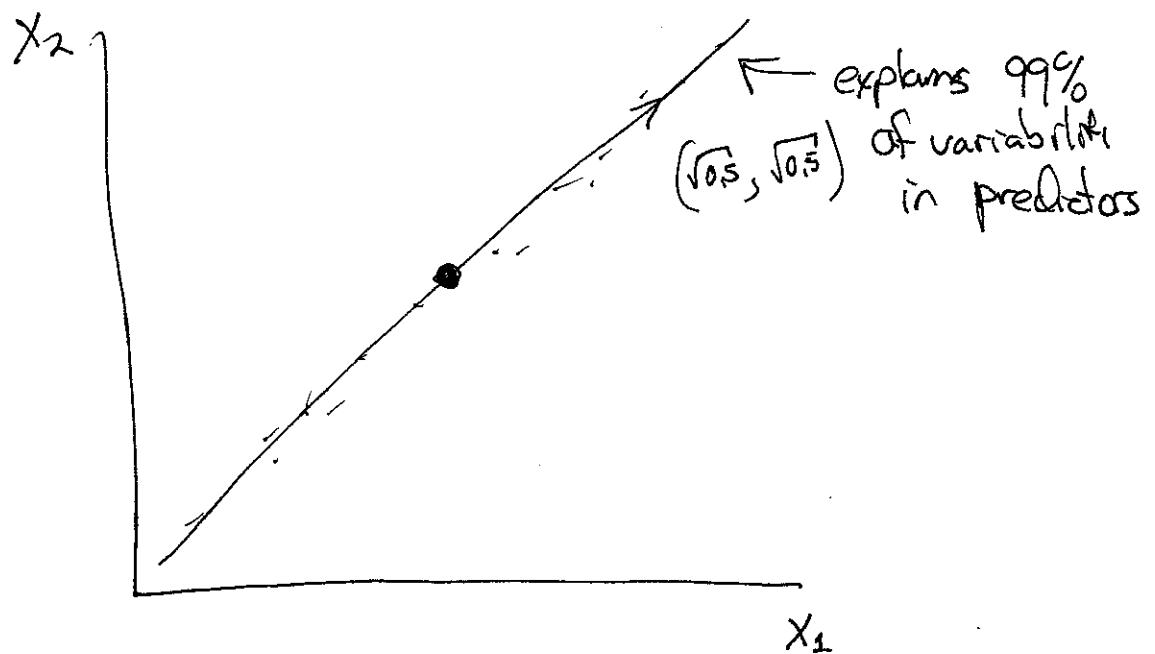
- 6 This  $\mathbf{u}$  is the first (sample) principal component. Denote it  $\mathbf{u}_1$ .

- 7 The additional principal components  $\mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_p$  are found by finding successive eigenvectors of  $\hat{\Sigma}$ .

$$\max_{\mathbf{U}} \frac{\mathbf{U}^T \hat{\Sigma} \mathbf{U}}{\mathbf{U}^T \mathbf{U}} = \frac{\text{sample variance of } v's}{\text{scaling to avoid } \mathbf{U} \rightarrow \infty}$$

$$\mathbf{V}(\mathbf{U}^T \mathbf{X}) = \mathbf{U}^T \Sigma \mathbf{X} \quad \text{where} \quad \mathbf{V}(\mathbf{X}) = \Sigma$$

- 1 It is useful to think about what is happening conceptually:
- 2    1. A new coordinate system is being constructed in which the  $p$ -  
3       dimensional data are represented. The center of this coordinate  
4       system is the sample mean of the data, but the axes are the prin-  
5       cipal components  $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_p$ .
- 6    2. When expressed in this new coordinate system, each of the di-  
7       mensions have sample correlation of zero.
- 8    3. The amount of “variance explained” by each principal compo-  
9       nent is less than each of the previous principal components.
- 10    4. In many situations, we will choose some cutoff  $\underline{q} \ll p$  such that  
11       we will only retain the first  $q$  axes in our new coordinate system.  
12       This will be because these first  $q$  components “capture” most of  
13       the variability in the data.  
  
14       For instance, in Figure 3, most of the variability in the data is  
15       “captured” by the first axis  $\mathbf{u}_1$ .  
  
16       In typical applications  $p$  will be very large, and a low-dimensional  
17       representation will be of great value.



- 1 **Exercise:** Explain how this is potentially a solution for multicollinearity, and also could yield a simpler model. Are there drawbacks?

3 We're constructing new predictors that are  
 4 by definition uncorrelated, hence no  
 5 problems with multicollinearity.

6 Also could use fewer predictors if a  
 7 few PC's describe most of the  
 8 variability in the original predictors.

10 A drawback is that the new predictors  
 11 are linear combinations of the  
 12 original predictors.

$$14 V_j = \sum_{k=1}^p u_k x_k$$

15 ↗ original predictors

17

18

## 1 The Example

- 2 Let's return to our original example, the consideration of the yield
- 3 curves for each of  $n = 250$  trading days of 2012.
  
- 4 Here,  $p = 11$  since each vector  $\mathbf{x}_i$ , for  $i = 1, 2, \dots, 250$ , has 11 dimensions.
  
- 6 From inspecting Figure 1 we get the hint that there is a common
- 7 shape to these 250 curves. One would guess that it does not really re-
- 8 quire 11 dimensions to describe the variations in the shapes of these
- 9 curves.
  
- 10 This is an ideal case in which one could use principal components
- 11 analysis. It is likely that these curves could be summarized using a
- 12 few well-chosen coordinates in a new axis system.

- 1 To perform the PCA in R, use the function `princomp()`. Here, our
- 2 yield curves are stored in a 250 by 11 matrix named `yieldcurves`;
- 3 each row is one date.

4 > `pcaout = princomp(yieldcurves)`

- 5 Let's now look at the components of `pcaout` and learn some of the
- 6 terminology associated with PCA.

- 7 • The center in the new coordinate system, which equals the sam-  
8 ple mean of the  $n$  vectors, is found at `pcaout$center`.
- 9 • The new coordinate axes are called the **loadings**; these are stored  
10 as the columns of `pcaout$loadings`. In other words, I can get  
11 the  $i^{th}$  principal component by looking at `pcaout$loadings[, i]`.
- 12 • The position of each of the 250 vectors in the new coordinate sys-  
13 tem are called the **scores**; these are found at `pcaout$scores`.  
14 The  $i^{th}$  row of this matrix returns the coordinates of the  $i^{th}$  yield  
15 curve in this new coordinate system.

- If you look at `summary(pcaout)`, you will see the amount of the original variance explained by each of the successive components:

```
4 > summary(pcaout)
5 Importance of components:
6
7 Standard deviation      Comp.1      Comp.2      Comp.3      Comp.4      Comp.5
8 Proportion of Variance 0.9541648 0.02486489 0.01405227 0.003459429 0.001420329
9 Cumulative Proportion  0.9541648 0.97902973 0.99308200 0.996541431 0.997961761
10
11 Standard deviation      Comp.6      Comp.7      Comp.8      Comp.9
12 Proportion of Variance 0.0006097279 0.0005004205 0.000333582 0.0002475158
13 Cumulative Proportion  0.9985714885 0.9990719090 0.999405491 0.9996530068
14
15 Standard deviation      Comp.10     Comp.11
16 Proportion of Variance 0.0002137419 0.0001332512
17 Cumulative Proportion  0.9998667488 1.0000000000
```

18 From the above output, we can immediately see that the first  
19 three components explain virtually all of the variability in the  
20 data. It seems that one only needs three coordinates to capture  
21 almost all of the variability in the yield curves.

- 1 Figure 4 shows the effect of varying along the first new coordinate  
 2 axis. The solid line in the plot is the center in our new coordinate  
 3 system. The dashed lines show the effect of going in either direction  
 4 away from this center.

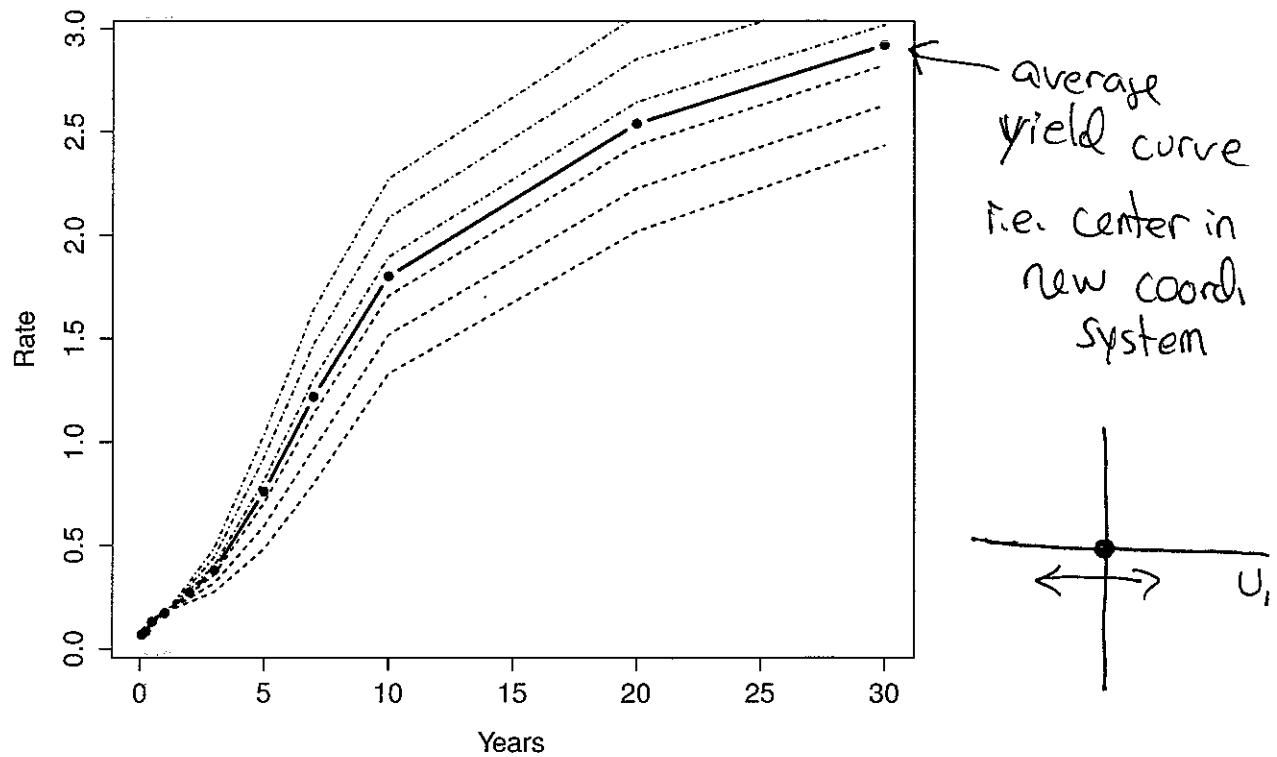


Figure 4: The effect of varying along the first principal component, i.e., along the first axis in our new coordinate system.

- 1 Figure 5 shows the effect of varying along the second new coordinate
- 2 axis.

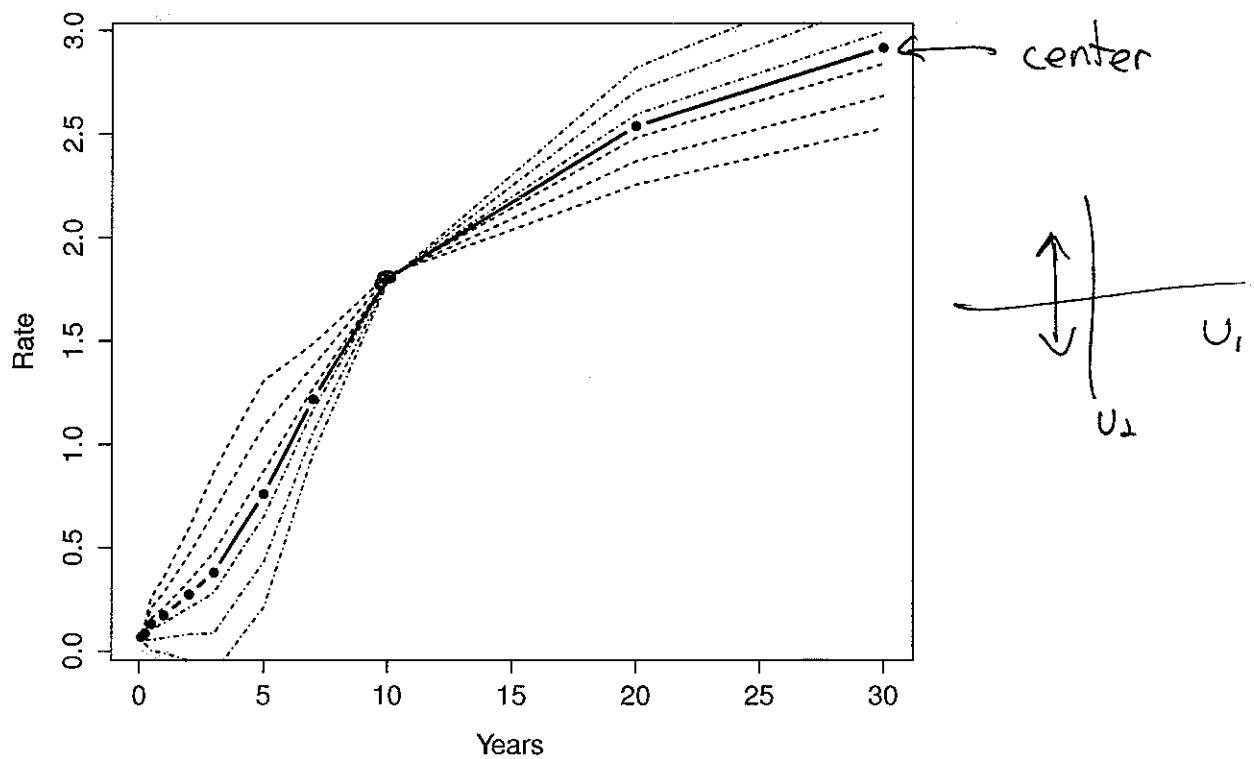


Figure 5: The effect of varying along the second principal component, i.e., along the second axis in our new coordinate system.

- 1 Figure 6 shows the effect of varying along the third new coordinate
- 2 axis.

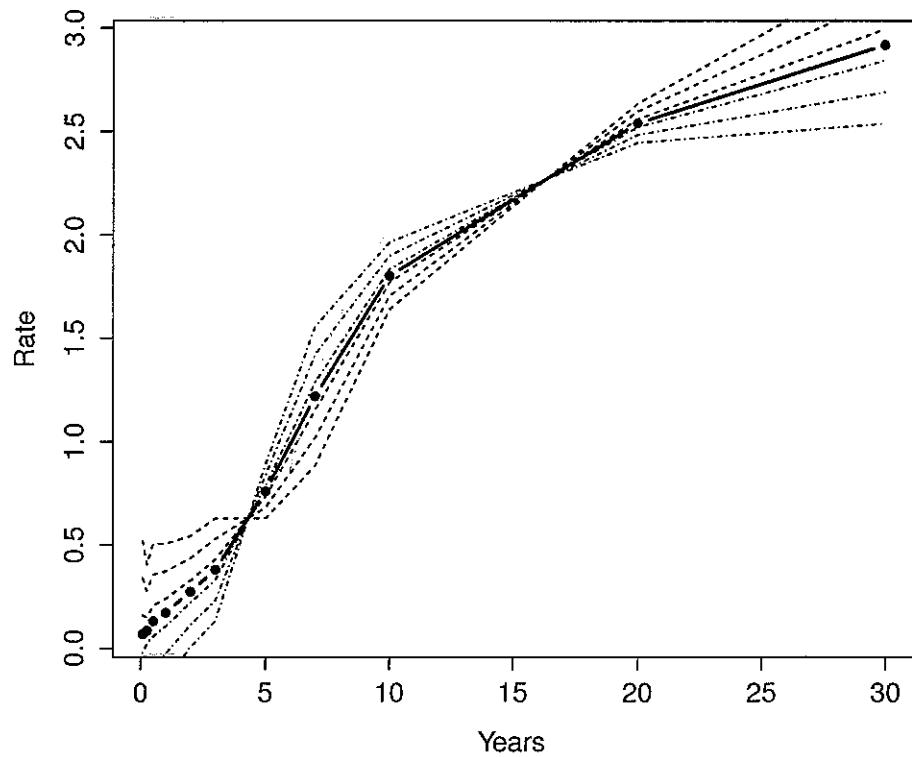


Figure 6: The effect of varying along the third principal component, i.e., along the third axis in our new coordinate system.

- 1 Our new representation of yield curves has many uses: For example,
  - 2 we have an easier way of communicating about shifts in the yield
  - 3 curve. We have a simpler way of recognizing novel, new behavior
  - 4 in the yield curve. Also, we could use these new coordinates as pre-
  - 5 dictors in a regression. Use of PCA for regression is called **Principal**
  - 6 **Components Regression.**
- 
- 7 **Exercise:** What, specifically, would you use from the PCA as the pre-
  - 8 dictors in the regression?

9 Use the scores, e.g. if want to include  $q$   
10 of the components ( $q = 3$ , say), then  
11 `pacout$scores[, 1:q]`

---

12

---

13

---

14

- 15 We should be mindful of the fact that the best coordinate system to
- 16 use for any given situation may change over time. What is best for
- 17 characterizing yield curves in 2012 may not be best in 2013. On the
- 18 other hand, constantly adjusting the coordinate system diminishes
- 19 the value of the approach. This is a tradeoff we must constantly man-
- 20 age.

1 **Should the Predictors be Scaled (Standardized)?**

- 2 PCA does not make sense when the different predictors are on very
- 3 different scales.
  
- 4 For example, suppose that one predictor under consideration was
- 5 “Volume.” Clearly, there is no practical difference between repre-
- 6 senting this variable in units of “shares” or in units of “thousands of
- 7 shares.” The only difference is a scaling.
  
- 8 But, that scaling would have a significant effect on PCA: If the pre-
- 9 dictor is in units of “shares,” then there is a much larger variance in
- 10 that predictor than there would be if the units were “thousands of
- 11 shares.”

- 1 For this reason, predictors are often individually “standardized” so
- 2 that they each have mean zero and variance one, prior to PCA.
  
- 3 **But: this can be risky.** The individual scaling can remove useful
- 4 information that may be present in the different variances.
  
- 5 So, don't standardize when the predictors are all in the same units,
- 6 or are otherwise comparable.
  
- 7 You should standardize when predictors are an assortment of differ-
- 8 ent variables on different scales.
  
- 9 In R, PCA with standardized predictors is easily accomplished by
- 10 using the argument cor = T to the function `princomp()`.

$\uparrow$   
 eigenvectors of correlation matrix instead  
 of covariance matrix

$$\begin{bmatrix} 1 & \rho(x_i, x_j) & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}$$

$$V(\hat{z}) = \hat{\sum} = \left[ \begin{array}{c} \\ \\ \\ \end{array} \right]$$

$$\boxed{x^T \hat{\sum} \hat{x}} = V(x^T z)$$

Find  $x$  such that  $V(x^T z)$  is maximized subject to

$$\|x\| = x^T x = 1$$

This is the eigenvector problem

1

## **The Lasso and Related Methods**

- 3 First, we need to establish the following idea: There is a direct re-  
4 lationship between overestimating the absolute coefficients  $|\beta_i|$  and  
5 overfitting the model.
- 6 Why is this the case? Simply put, the complete lack of a linear re-  
7 lationship between the response and  $x_k$  occurs when  $|\beta_k| = 0$ , and,  
8 as the strength of that relationship grows (holding everything else  
9 constant),  $|\beta_k|$  will also grow.
- 10 So, as our model predicts  $|\beta_k|$  larger, it is predicting a stronger rela-  
11 tionship between the response and  $x_k$ . But, if the estimate for  $|\beta_k|$  is  
12 too large, we will be overfitting, i.e., fitting a stronger relationship  
13 between the predictors and the response than is actually present.
- 14 **Exercise:** Within the context of this discussion, explain why remov-  
15 ing a predictor from the model is an effort to avoid overfitting.

17

18

- 1 Recognizing this, a long-standard approach to fitting a linear model
- 2 was to

3 find  $\hat{\beta}$  that minimizes RSS, subject to  $\sum_{j=1}^p \hat{\beta}_j^2 < t$ .

- 4 This procedure is called **ridge regression**.

- 5 It can be shown that this is equivalent to finding  $\hat{\beta}$  that minimizes

$$6 \quad \text{RSS} + \lambda \sum_{j=1}^p \beta_j^2 \quad (*)$$

- 7 There is a one-to-one connection between  $t$  and  $\lambda$ : Choosing  $t$  smaller
- 8 leads to larger  $\lambda$ .

- 9 It is more common to work in terms of  $\lambda$ ; this is a **regularization parameter** or **smoothing parameter**.

- 11 **Exercise:** Describe the effect of increasing  $\lambda$ .

12 As I increase  $\lambda$ , there is more of a penalty for  
 13 having  $\sum \hat{\beta}_j^2$  be large, so the procedure will  
 14 keep  $\sum \hat{\beta}_j^2$  small.

15 As  $\lambda \rightarrow 0$ , converge to the least squares  $\hat{\beta}$

- 1 The result of ridge regression is that the estimates  $\hat{\beta}_j$  are **shrunk** to-
  - 2 wards zero. This is the general idea of **shrinkage**.
  
  - 3 **Comment 1:** Note that the intercept  $\beta_0$  is **not** included in the sum
  - 4 that is added on to RSS in the equation above.
  
  - 5 **Comment 2:** The predictors are usually **standardized** prior to fitting
  - 6 the model. Here, standardization refers to centering and scaling the
  - 7 predictor such that it has sample mean zero and sample variance of
  - 8 one. This clearly is important; if it were not done, the relative sizes
  - 9 of the  $\beta_i$  would largely dictate the sum  $\sum_j \beta_j^2$ .
  
  - 10 **Comment 3:** In practice,  $\lambda$  will be chosen via cross-validation: You
  - 11 seek to find the value of  $\lambda$  that yields the best predictions of the re-
  - 12 sponse; by using cross-validation, you are adjusting for overfitting.

The key insight of Tibshirani (1996)<sup>2</sup> was to replace the restriction

$$\sum_{j=1}^p \hat{\beta}_j^2 \leq t \quad \text{ridge regression}$$

with

$$\sum_{j=1}^p |\hat{\beta}_j| \leq t.$$

- 1 The result is that the optimization problem changes from minimizing
- 2 Equation (\*) above to minimizing

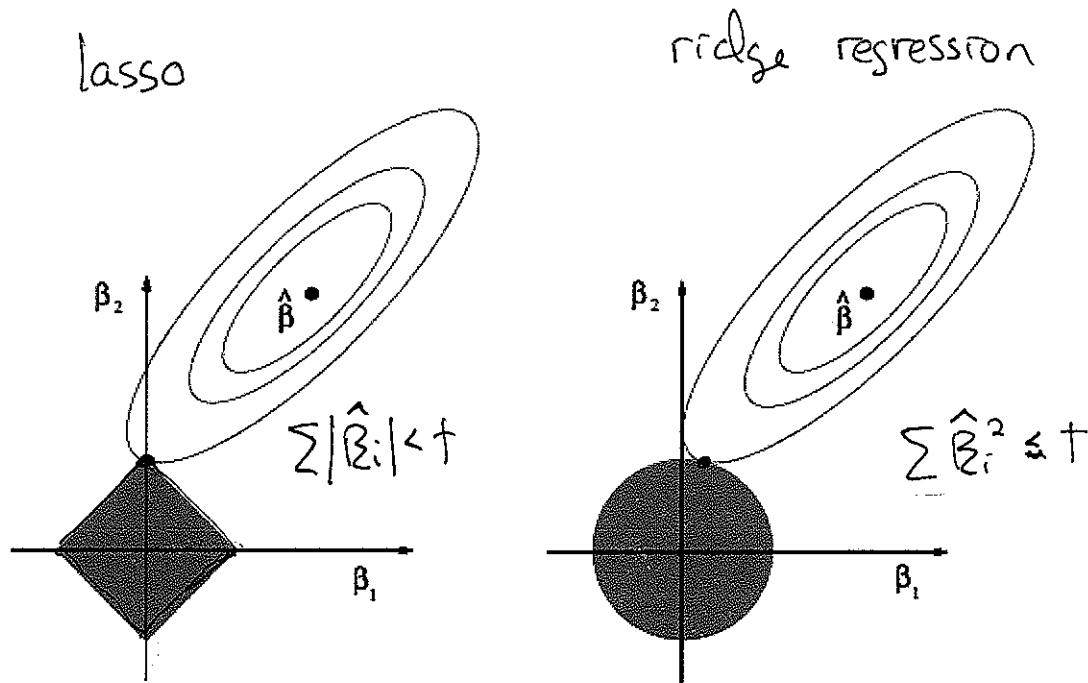
$$3 \quad \text{RSS} + \lambda \sum_{j=1}^p |\beta_j|$$

- 4 This seemingly simple adjustment has an incredible effect: Not only
- 5 is there still shrinkage of the estimates, but there is also variable se-
- 6 lection, since the solution to the optimization problem will naturally
- 7 force some <sup>often many</sup>  $\hat{\beta}_j$  to be equal to zero, i.e., it removes some of the
- 8 predictors from the model.
- 9 This procedure is called the **lasso**.

---

<sup>2</sup>"Regression Shrinkage and Selection via the Lasso," *JRSSB*, Volume 58, page 267.

- Figure 7, reproduced from Hastie, Tibshirani, and Friedman (2009),
- (and also appearing on page 222 of ISL) shows the effect clearly.
- The red contours are lines of constant RSS. The blue regions are the regions within which the  $\hat{\beta}_j$  are constrained.



**FIGURE 3.11.** *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions  $|\beta_1| + |\beta_2| \leq t$  and  $\beta_1^2 + \beta_2^2 \leq t^2$ , respectively, while the red ellipses are the contours of the least squares error function.*

Figure 7: Reproduced from HTF (2009, page 71). Also found on page 222 of ISL.

- Note that the  $\hat{\beta}$  shown in the figures are the estimates under standard least squares (neither ridge regression nor lasso is employed).

- <sup>1</sup> Ridge regression significantly shrinks  $\hat{\beta}_1$  towards zero, but it is not
- <sup>2</sup> completely removed, while with the lasso it is completely removed.

- <sup>3</sup> **Comment 1:** The lasso has become one of the most intense areas of
- <sup>4</sup> research within the field of Statistics. There are many generalizations
- <sup>5</sup> and extensions.

- <sup>6</sup> One such extension is the **elastic net**. Here, the optimization problem
- <sup>7</sup> is to find  $\beta$  that minimizes

$$\text{RSS} + \lambda \sum_{j=1}^p (\alpha |\beta_j| + (1 - \alpha) \beta_j^2)$$

*↖ lasso      ↘ ridge*

- <sup>8</sup>
- <sup>9</sup> where  $0 \leq \alpha \leq 1$ . Thus, this is a mixture of ridge regression ( $\alpha = 0$ )
- <sup>10</sup> and the lasso ( $\alpha = 1$ ).

- <sup>11</sup> **Comment 2:** This idea can be extended to any GLM by replacing RSS
- <sup>12</sup> with the negative of the log likelihood, e.g., using lasso with logistic
- <sup>13</sup> regression means minimizing

$$-\text{log likelihood} + \lambda \sum_{j=1}^p |\hat{\beta}_j|.$$

<sup>1</sup> **Example**

- <sup>2</sup> This example is based on material you will cover in 46-936, *Statis-*  
<sup>3</sup> *tical Arbitrage*. Much of this is adapted from the lecture notes of  
<sup>4</sup> Prof. Lehoczky and Prof. Schervish. The use of the lasso for this pur-  
<sup>5</sup> pose was their idea.
- <sup>6</sup> In “Statistical Arbitrage in the US Equities Market”<sup>3</sup>, Avellaneda and  
<sup>7</sup> Lee propose methods for **pairs trading** based on pairing stocks with  
<sup>8</sup> (suites of) exchange traded funds (ETFs).
- <sup>9</sup> Without getting into the details, a critical step in the process is to  
<sup>10</sup> identify ETFs which are strongly correlated with the particular stock  
<sup>11</sup> under consideration, but are not too strongly correlated with each  
<sup>12</sup> other. See the discussion at the start of §2.3 in the paper.

---

<sup>3</sup>Quantitative Finance, Volume 10, page 761

- <sup>1</sup> The underlying factor models which drive the trading strategy are of
- <sup>2</sup> the form

<sup>3</sup>

$$R_{it} = \sum_{j=1}^p \beta_{ij} x_j + \epsilon_{it}$$

- <sup>4</sup> where  $R_{it}$  is the return for stock  $\underline{i}$  for time period  $t$ , and  $x_j$  is the
- <sup>5</sup> return on ETF number  $j$ .

- <sup>6</sup> Note that there is a separate model fit for each stock  $i$ .

- <sup>7</sup> Here,  $p$  could be large: There are many potential ETFs. In this ex-
- <sup>8</sup> ample, we will start with a list of all of the NYSE ETFs as listed at

<sup>9</sup> <http://www.investorpoint.com/exchange/NYSE-New+York+Stock+Exchange/etf/>

- <sup>10</sup> You can read this list in using the command

```
11 > nyse_etfs = read.table(  
12 "http://www.stat.cmu.edu/~cschafer/MSCF/NYSEETFlist.txt", sep="")
```

- <sup>13</sup> There are 1427 ETFs listed in this file.

- 1 I constructed a matrix consisting of the daily returns for 2013 for
- 2 these ETFs. (After filtering those without data available for the entire
- 3 period, or other problems, there were 1124 remaining.) This data is
- 4 stored in the object `retmat`.
  
- 5 Next, I obtained the daily returns for 2013 for MSFT. This is stored in
- 6 the object `msft`.
  
- 7 **Exercise:** What will happen if I try to fit a classic linear model to
- 8 these data, with the MSFT returns as the response, and the 1124 ETFs
- 9 as predictors?

10  $n = 250$ ,  $p = 1124$ , so can't fit the model

11 Need  $p < n$  (want  $p \ll n$ )

12

13

14

15

16

17

18

19

20

- 1 The function `glmnet()`, part of the package `glmnet`, will quickly fit
- 2 the lasso for a sequence of different values for  $\lambda$ .

- 3 The syntax is as follows:

4 > `glmnetout = glmnet(retmat, as.numeric(msft))`

$X$        $Y$

- 5 The procedure chooses a reasonable range of 100 values for  $\lambda$ .

- 6 We can see a portion of the sequence as follows:

7 > `glmnetout`

8 Call: `glmnet(x = retmat, y = as.numeric(msft))`

	Df	%Dev	Lambda
11	[1, ]	0 0.00000	7.831e-03
12	[2, ]	2 0.02324	7.475e-03
13	[3, ]	2 0.04432	7.135e-03
14	[4, ]	2 0.06351	6.811e-03
15	[5, ]	2 0.08100	6.502e-03
16	[6, ]	3 0.09693	6.206e-03
17	[7, ]	3 0.11150	5.924e-03
18	[8, ]	3 0.12490	5.655e-03
19	[9, ]	3 0.13700	5.398e-03
20	[10, ]	5 0.14900	5.152e-03
21	....		

$\lambda$  is decreasing

% Deviance =  $R^2$

Deviance = RSS

Df = "degrees of freedom"

= # of non-zero  $\hat{\beta}$ 's

- 1 Each column of glmnetout\$beta holds the  $\hat{\beta}_{ij}$  for one particular
- 2 choice of  $\lambda$ . So, in this example this matrix is 1124 by 100.

$$\lambda = 0.006206$$

- 3 If I look at glmnetout\$beta[, 6], for instance, I will see mostly
- 4 zeros. In fact, from the table above, I know that there are only three
- 5 nonzero  $\hat{\beta}_{ij}$ . (Look at the column named Df.)
- 6 The following syntax will show the nonzero entries in column ten:

```

7 > glmnetout$beta[ (glmnetout$beta[, 6] != 0), 6]
8
9      IGM          ROM          XLK
10
11  0.11315185  0.00184535  0.11309398

```

*k-fold*

- 10 One could seek to find the optimal  $\lambda$  via <sup>^</sup>cross-validation.

- 11 In R, this is done simply using the command

```
12 > holdcvout = cv.glmnet(retmat, as.numeric(msft))
```

*X*      *y*

- 1 If one runs `plot(holdcvout)`, the result is shown as Figure 8.

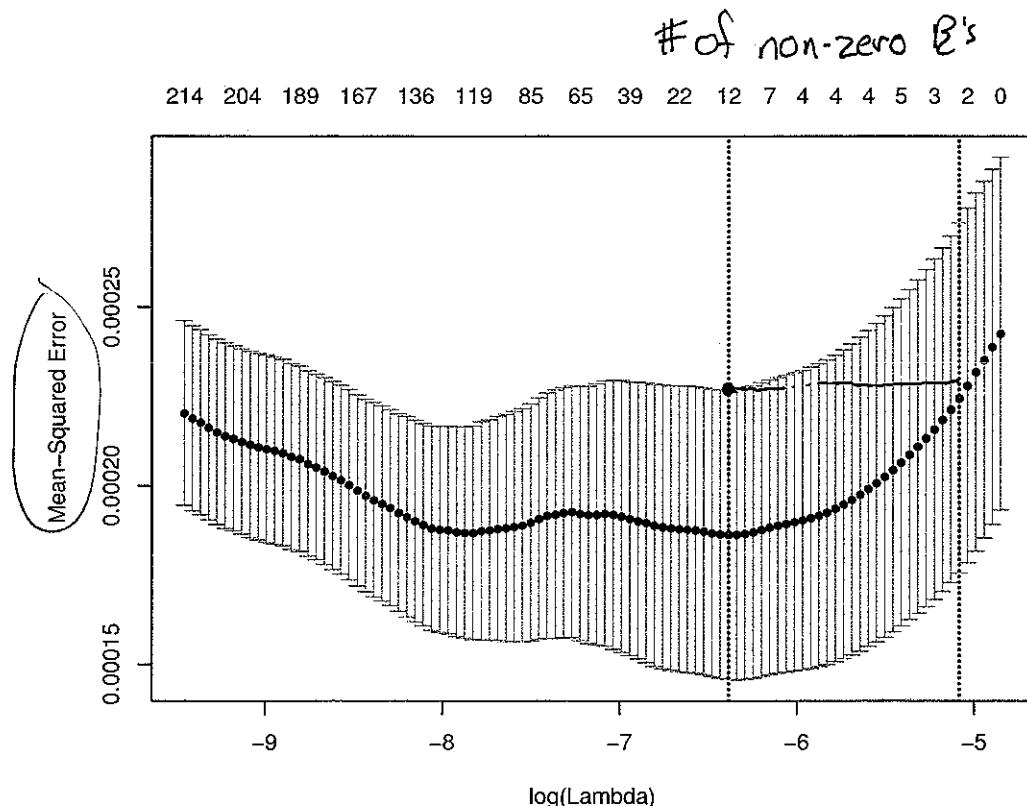


Figure 8: The plot produced by `cv.glmnet`.

- 2 The numbers across the top of the plot give the number of nonzero  
 3  $\hat{\beta}_{ij}$  for each value of  $\lambda$  (shown on the log-scale along the horizontal  
 4 axis).  
 5 The dotted vertical line on the left marks the point where the MSE  
 6 is minimized. But, the MSE is only estimated by cross-validation. In  
 7 recognition of this, another dotted line is drawn at the point which  
 8 is within one standard error of the minimum MSE; it is typical to use  
 9 this second  $\lambda$  as our optimal choice.

- 1 The incredible conclusion is that the optimal model has on the order
- 2 of two to 12 nonzero  $\hat{\beta}_{ij}$ .
  
- 3 Returning to our application, these few ETFs with nonzero coeffi-
- 4 cients could serve as the portfolio used in our pairs trading strategy.
- 5 It is worth noting that the authors recommend using 15 factors.
  
- 6 The final listing of parameter estimates for our optimal model are as
- 7 follows:

8	DBEM	DBIZ	DHS	EEMS	HYMB
9	5.582634e-02	6.873559e-03	3.068828e-01	3.060074e-05	6.855463e-02
10	IGU	IGV	MLN	MONY	ROM
11	-7.626999e-02	1.510050e-01	5.010218e-02	1.108840e-01	2.214586e-01
12	TDD	<del>TED</del>			
13	1.494576e-02	<del>7.870603e-17</del>			

- 1 To summarize what happened here via the lasso:
  
- 2     • We reduced the number of predictors from 1124 to close to 10.
- 3     • We dealt with the problem that the initial model was severely  
4         overparameterized ( $p \gg n$ ).
- 5     • The new set of predictors was constructed in a way that protects  
6         against multicollinearity.
- 7     • Shrinkage was applied to the non-zero coefficients that remain;  
8         the appropriate amount was applied to minimize the MSE.
- 9     • Implementation was fast and simple. *but maybe difficult to  
       choose  $\lambda$ .*

## 1 The Group Lasso

- 2 One variation on the lasso is the **group lasso**. In this version, one
- 3 is able to assign predictors to “groups” which are “tied together” in
- 4 the estimation procedure: Predictors in the same group are either all
- 5 kept, or all excluded.
- 6 One important use of this procedure is with categorical variables.
- 7 Recall that including a categorical variable with  $K$  levels introduces
- 8  $(K - 1)$  additional  $\beta$  parameters into the model. The user may wish
- 9 to “group” these  $(K - 1)$  parameters together so that the levels of the
- 10 factor are either all in or all out.
- 11 But there are other situations in which grouping predictors may be
- 12 sensible. For instance: There are groups of predictors, each corre-
- 13 sponding to a distinct event or point in time.

- <sup>1</sup> The group lasso is implemented in R using the function `grplasso()`,
  - <sup>2</sup> part of the package `grplasso`.
- 
- <sup>3</sup> The syntax is a little different than other functions we've considered.
  - <sup>4</sup> Note the following:
    - <sup>5</sup> 1. The function expects to be passed the **entire** matrix  $\mathbf{X}$ , where  $\mathbf{X}\beta$   
<sup>6</sup> is the linear predictor.  
<sup>7</sup> Hence, the number of columns in  $\mathbf{X}$  equals the total number of  
<sup>8</sup>  $\beta$  parameters, and the number of rows is  $n$ . (This is often called  
<sup>9</sup> the **design matrix**.) In particular, this means that if the intercept  
<sup>10</sup> term  $\beta_0$  is included in the model, the matrix  $\mathbf{X}$  must include a  
<sup>11</sup> column of 1's.  
<sup>12</sup> Also,  $\mathbf{X}$  must correctly reflect the inclusion of any categorical  
<sup>13</sup> variables. Each column corresponding to the  $(K - 1)$  levels will  
<sup>14</sup> be filled with zeros and ones, with the position of the ones deter-  
<sup>15</sup> mined by the level of the factor for those observations.
    - <sup>16</sup> 2. The response is passed in as the second argument.

- 1    3. The third argument to `grplasso` is a vector, equal in length to
  - 2    the number of columns of  $\mathbf{X}$ . If column  $i$  of  $\mathbf{X}$  corresponds to a
  - 3    predictor that is not a candidate to be dropped (e.g., the intercept
  - 4    term), then put a `NA` in that entry. Otherwise, predictors in the
  - 5    same group should be assigned the number. This is called the
  - 6    index vector by R.
  - 7    4. If you want to fit a standard linear model, then you must set
  - 8    `model=LinReg()`. The default is `model=LogReg()`, which is
  - 9    used to fit a binary logistic regression. This function does not
  - 10   have the flexibility to fit other GLMs.
  - 11   5. Unlike `glmnet()`, the user must specify a sequence of values for
  - 12    $\lambda$  to be tested. These are supplied via the argument `lambda` to
  - 13   `grplasso()`.
- 14   There is, however, a function `lambdamax()` which takes the same
- 15   arguments as `grplasso()` (except for `lambda`) and returns the
- 16   largest possible choice for  $\lambda$ . This could be used to form a se-
- 17   quence of values to try.

- 1 Let's do a toy example to illustrate use of the procedure.
- 2 Suppose we have a continuous response, stored in the vector  $y$ .
- 3 We have two predictors:  $x_1$  and  $x_2$ . Predictor  $x_2$  is categorical, with
- 4 three levels.
- 5 The sample size is  $n = 6$ , and the data are shown in the table below:

	y	9.18	10.16	18.01	13.13	12.59	6.13
6	$x_1$	21.79	27.21	29.58	24.76	26.44	22.78
	$x_2$	0	0	1	1	2	2

- 7 **Exercise:** Write out the design matrix  $X$  for this example.

8 \_\_\_\_\_

9 \_\_\_\_\_

10 \_\_\_\_\_

11 \_\_\_\_\_

12 \_\_\_\_\_

13 \_\_\_\_\_

14 \_\_\_\_\_

15 \_\_\_\_\_

- 1 This design matrix is loaded into a six-by-four matrix in R. Typically
- 2 this would be done by column:

```
3 X = matrix(0, nrow=6, ncol=4)
4 X[,1] = rep(1, 6)
5 X[,2] = x1
6 ...
```

- 7 Next, we create the `index` argument, which specifies the group for
- 8 each predictor. Here, we would use

```
9 ind = c(NA, 1, 2, 2)
```

- 10 Next, I want to find the largest possible value that  $\lambda$  could take in this
- 11 example. (In other words, for values of  $\lambda$  equal to or greater than this
- 12 maximum, all predictors are excluded.)

```
13 maxlam = lambdamax(X, y, ind, model=LinReg())
```

- 1 Then, I create a sequence of  $\lambda$  values. A recommended way to do this
- 2 is to make a geometric series of the form

3 
$$\lambda_{\max} \times (0.75)^i, \quad \text{for } i = 0, 1, \dots, m$$

- 4 In R, with  $m = 10$ :

5 `lamseq = maxlam * (0.75^(0:10))`

- 6 Finally, I make the call to `grlasso()`:

7 `holdout = grlasso(X, y, ind, lambda=lamseq,`  
8       `model=LinReg())`

- I see the estimated coefficients below. Note that it includes the coefficients for each value of  $\lambda$ . Also note that predictors  $x_2$  and  $x_3$  are either both included, or both excluded.

```
4 > holdout$coef
5      35.7115385947286 26.7836539460465 20.0877404595349 15.0658053446511
6  [1,]      11.53333     4.3945088     -0.9596093     -3.59201809
7  [2,]      0.00000     0.2807613      0.4913323      0.58607141
8  [3,]      0.00000     0.0000000     0.0000000     0.73392107
9  [4,]      0.00000     0.0000000     0.0000000     -0.06339512
10     11.2993540084884 8.47451550636627 6.3558866297747 4.76691497233103
11 [1,]     -5.1492332    -6.3158463     -7.1900695     -7.8453085
12 [2,]      0.6382087     0.6773524      0.7067369      0.7287916
13 [3,]     1.5011611     2.0740111     2.5020730     2.8221765
14 [4,]     -0.1360220    -0.1949118     -0.2417526     -0.2784737
15     3.57518622924827 2.6813896719362 2.01104225395215
16 [1,]     -8.3366801    -8.7048783     -8.9809461
17 [2,]      0.7453502     0.7577672      0.7670833
18 [3,]     3.0616779     3.2409998     3.3753037
19 [4,]     -0.3069485    -0.3288470     -0.3455837
```

- 1 Unfortunately, this package does not include a built-in cross-validation
- 2 procedure, so we'd need to write our own in order to choose the op-
- 3 timal  $\lambda$ .
  
- 4 This function does return the negative log likelihood via the compo-
- 5 nent `nloglik`, so we could calculate AIC simply using the following
- 6 R command:

```
7 > apply(holdout$coef!=0, 2, sum)*2 + holdout$nloglik
 8 35.7115385947286 26.7836539460465 20.0877404595349 15.0658053446511
 9      84.23333      62.98542      49.90847      40.74851
10 11.2993540084884 8.47451550636627 6.3558866297747 4.76691497233103
11      31.84380      26.83431      24.01619      22.43087
12 3.57518622924827 2.6813896719362 2.01104225395215
13      21.53904      21.03739      20.75520
```

## Part 6: Nonparametric Regression

- 2 Text references: §7.1 - 7.6 in ISL, §21.1 - 21.5 in Ruppert
- 3 Figure 1 shows the scatter plot of the relationship between the empirical forward rate and maturity date for zero-coupon bonds. Recall we introduced this example in Part 1, and that this is found in Ruppert, starting on page 381.

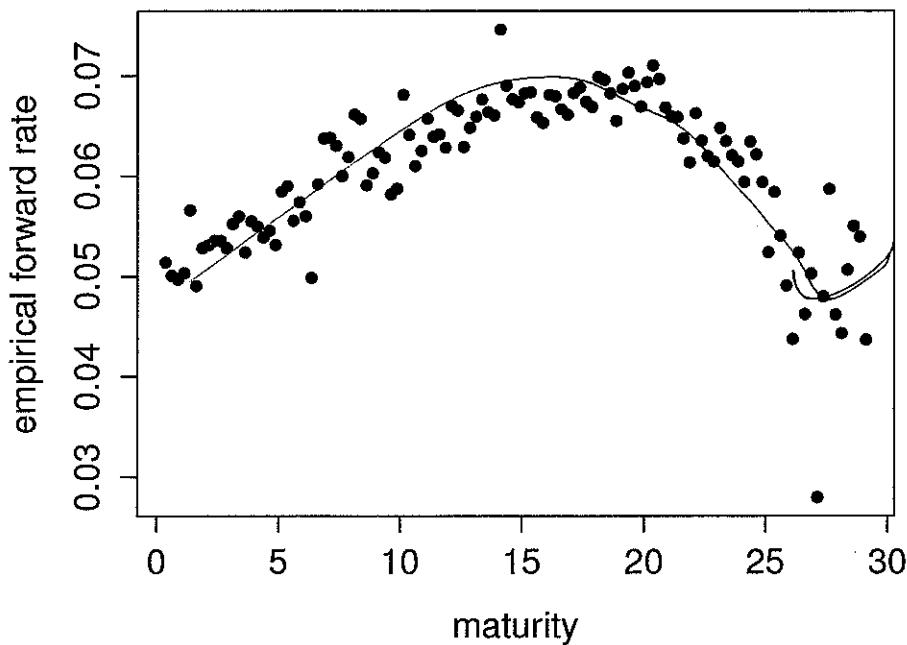


Figure 1: Empirical forward rate versus maturity for the estimation of the forward rate function.

- 7 We found there that a linear model that included higher-order powers of the variable “maturity” was a decent fit, but this was still unsatisfactory.

- 1 Here we will demonstrate **nonparametric regression** procedures for
  - 2 fitting this relationship.
  
  - 3 The key characteristic of nonparametric regression procedures is that
  - 4 they are able to adapt to the complexity of the relationships between
  - 5 the response and predictor(s).
  
  - 6 These procedures are controlled by a **smoothing parameter**, some-  
7 times called the **bandwidth**.
  
  - 8 As the smoothing parameter is chosen larger, a more “smooth” rela-  
9 tionship is fit. But, if it is necessary, it can be chosen smaller to fit to  
10 complexities in the relationship.
  
  - 11 The smoothing parameter is typically chosen via cross-validation or  
12 AIC.

- 1 It is useful to think of the “smoothing” which occurs as being as effort
  - 2 to “make up” for the lack of data.
  - 3 Indeed, there is a close relationship between the sample size  $n$  and
  - 4 the smoothing parameter: As  $n$  grows, we have more information,
  - 5 and do not need to perform as much smoothing, we can instead rely
  - 6 more on the data.
  - 7 This makes a good way to contrast parametric procedures with their
  - 8 nonparametric counterparts:
    - 9 – *In the parametric case* there is a fixed contribution from the assumptions that we make: If I incorrectly assume that a model is linear, no
    - 10 matter how large  $n$  is, I will not overcome that mistake.
    - 12 – *In the nonparametric case*, as  $n$  grows the amount of smoothing diminishes, and the data are allowed to completely dictate the shape
    - 13 of the relationship.

## 1 The (Residual) Deviance

2 As we consider a broader class of models, including those that make  
3 different distributional assumptions for the response and those that  
4 allow for nonparametric components, we see the concept of the **de-  
5 viance or residual deviance** of the fit.

6 This concept generalizes the residual sum of squares for the fit.

7 Hence, for models where the response is assumed to be normal, the  
8 deviance equals the RSS.

9 In the case of binary logistic regression, the deviance equals

$$10 \quad 2 \sum_{i=1}^n \left[ y_i \log\left(\frac{y_i}{\hat{p}_i}\right) + (1 - y_i) \log\left(\frac{1 - y_i}{1 - \hat{p}_i}\right) \right] = -\log \text{likelihood} + K$$

11 where  $y_i$  is the observed response and  $\hat{p}_i$  is the estimated probability  
12 from the model.

13 In all cases, maximizing the likelihood is equivalent to minimizing  
14 the deviance.

<sup>1</sup> **Degrees of Freedom**

- <sup>2</sup> Along with the deviance, we will see repeated references to the **de-**
- <sup>3</sup> **grees of freedom** for the model fit.
  
- <sup>4</sup> In a parametric model, the **(model) degrees of freedom** equals the
- <sup>5</sup> number of  $\beta$  parameters.
  
- <sup>6</sup> In the nonparametric case, there are ways of approximating the de-
- <sup>7</sup> grees of freedom, this is often referred to as the **effective degrees**
- <sup>8</sup> **of freedom** or **equivalent degrees of freedom**. The idea is that you
- <sup>9</sup> are approximating how many parameters would be necessary in ~~in~~
- <sup>10</sup> order to fit a parametric model of equal complexity.
  
- <sup>11</sup> See §21.3 in Ruppert for a discussion of how this approximation is
- <sup>12</sup> done in nonparametric regression.
  
- <sup>13</sup> The **residual degrees of freedom** equals  $n$  minus the model degrees
- <sup>14</sup> of freedom.

- 1 Figure 2 shows the effect of varying the smoothing parameter. In the  
2 top fit, there is insufficient smoothing, while in the bottom fit, the  
3 data have been oversmoothed.

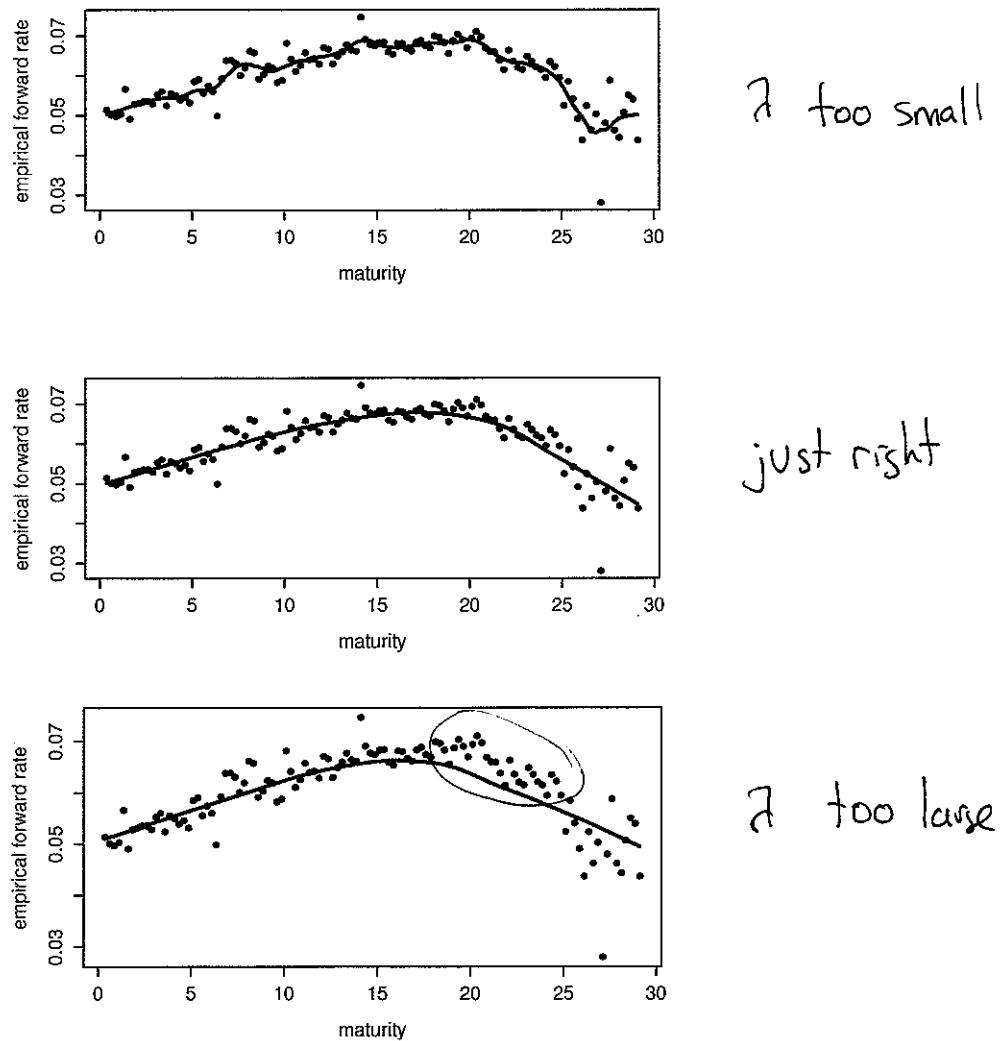


Figure 2: A demonstration of the effect of increasing the amount of smoothing in a nonparametric regression.

- 1 **Exercise:** Suppose a nonparametric regression model is utilized. What
- 2 is the relationship between the smoothing parameter, the deviance,
- 3 and the degrees of freedom?

4 As  $\lambda$  is chosen smaller (less smoothing), the  
5 deviance will decrease (there is more flexibility  
6 to come closer to each observations),  
7 and the model degrees of freedom will  
8 increase (need more parameters to define  
9 an equivalent parametric fit)

---

10

---

11

---

12

---

13

---

14

1

---

## 2 Local Linear Regression

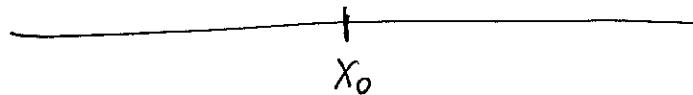
- 3 There are many variants of nonparametric regression, but there are a
- 4 couple procedures worth our focus.
- 5 The first is **local linear regression**, a version of **local polynomial re-**
- 6 **gression**. This approach has proven to be very powerful, and pos-
- 7 sses many strong theoretical properties to justify its use.
- 8 Briefly stated, this procedure works by fitting a sequence of linear
- 9 models: Each is fit not to the entire data set, but to only data within
- 10 a **neighborhood** of a target point.
- 11 The size of this neighborhood is our smoothing parameter: Large
- 12 neighborhoods yield a large degree of smoothing, while small neigh-
- 13 borhoods result in minimal smoothing.

1 Our model here is that we observe  $(x_i, Y_i)$  for  $i = 1, 2, \dots, n$  and that

2 
$$Y_i = f(x_i) + \epsilon_i$$

3 where the  $\epsilon_i$  are iid with mean zero and variance  $\sigma^2$ . Assuming that  
4 the  $\epsilon_i$  are normal will lead to further nice properties, but this devel-  
5 opment does not require that assumption.

6 In order to construct the local linear regression estimate of  $f(\cdot)$ , it is  
7 best to consider a sequence of steps **for each fixed  $x_0$  at which  $f(\cdot)$**   
8 **will be estimated.**



- 1 Step One: Fix the target point  $x_0$ .**
- 2 In other words, we are trying to estimate  $f(x_0)$ .**

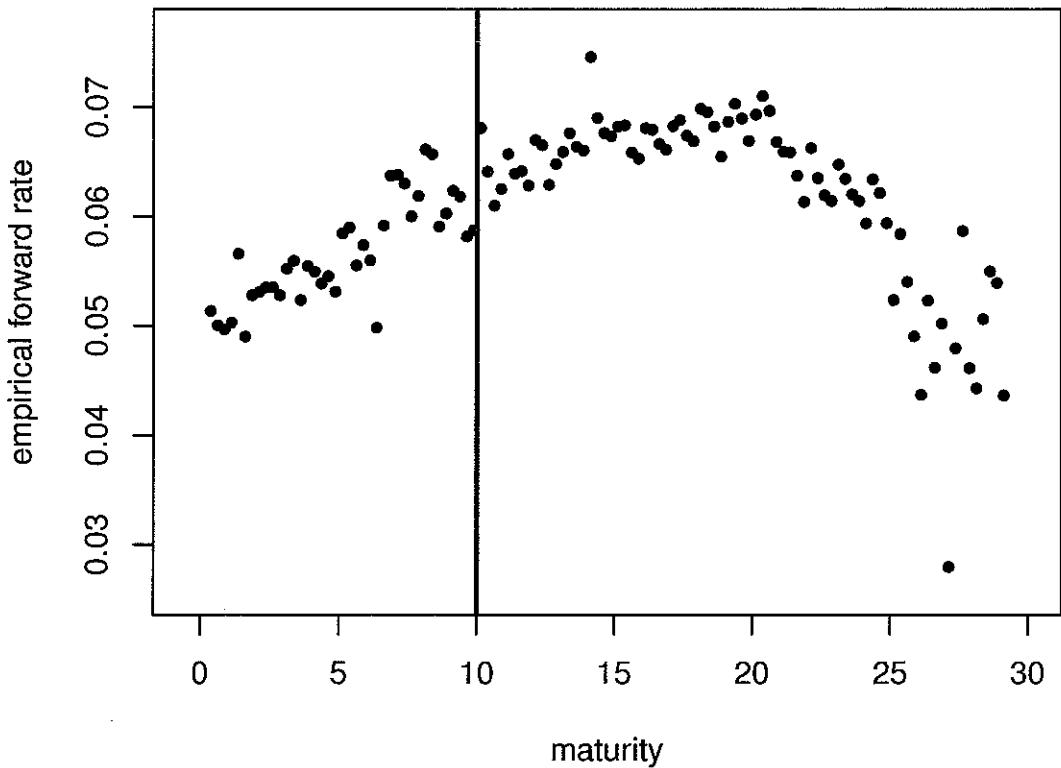


Figure 3: Step One, with  $x_0 = 10$ .

- 1 Step Two: Create the neighborhood around  $x_0$ .**
- 2 A common way to choose the neighborhood size is to choose is large enough to capture proportion  $\alpha$  of the data. This parameter  $\alpha$  is often 4 called the **span**. A typical choice is  $\alpha \approx 0.5$ .**

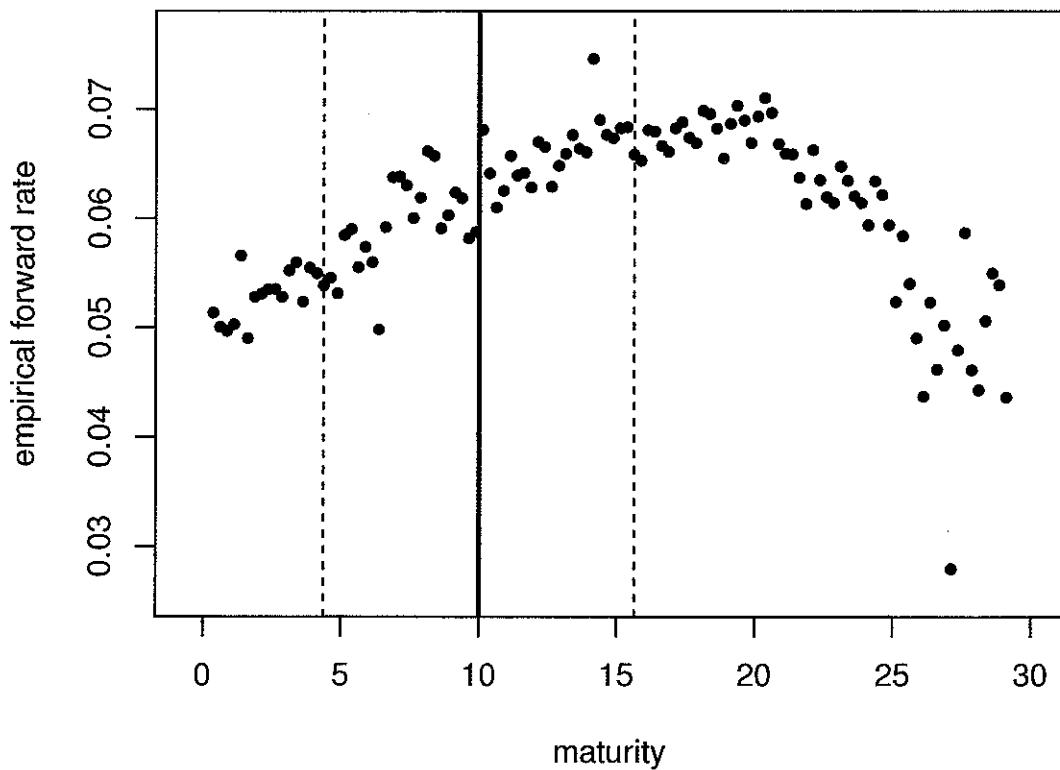


Figure 4: Step Two, with  $\alpha = 0.4$ .

1 Step Three: Weight the data in the neighborhood.

- 2 Values of  $x$  which are close  $x_0$  will receive a larger weight than those  
 3 far from  $x_0$ . Denote by  $w_i$  the weight placed on observation  $i$ . The  
 4 default choice is the **tri-cube weight function**:

$$5 \quad w_i = \begin{cases} \left(1 - \left|\frac{x_i - x_0}{\max \text{ dist}}\right|^3\right)^3, & \text{if } x_i \text{ in the neighborhood of } x_0 \\ 0, & \text{if } x_i \text{ is not in neighborhood of } x_0 \end{cases}$$

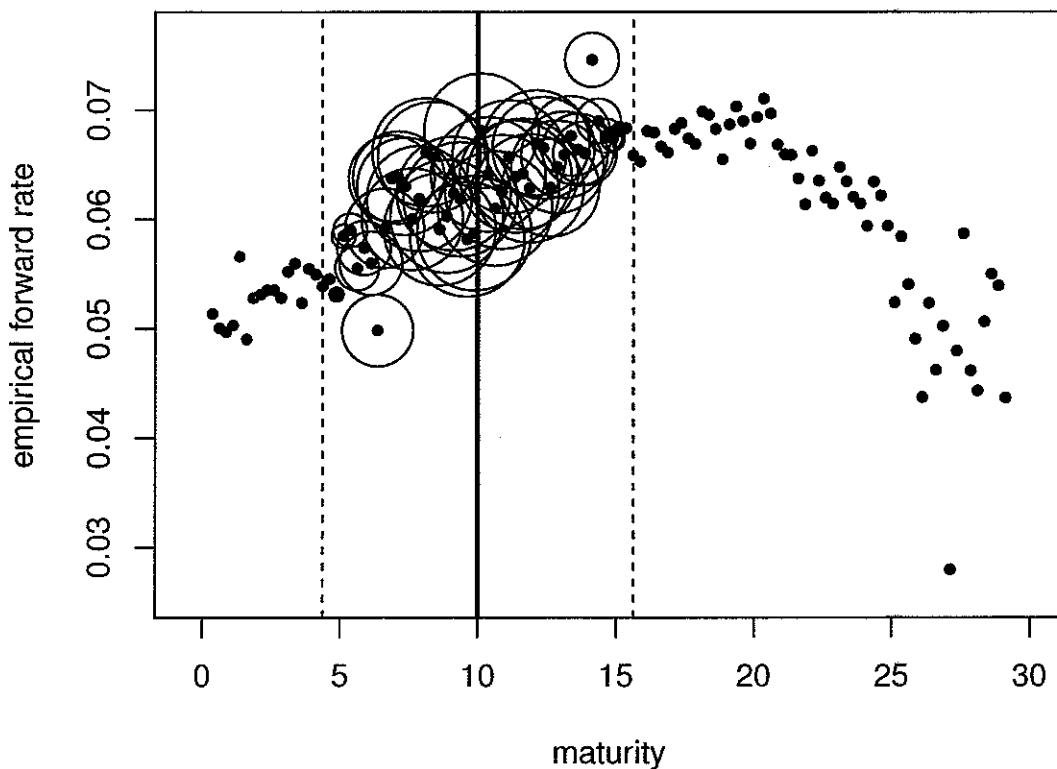
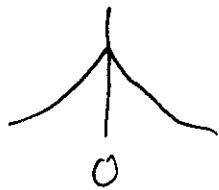


Figure 5: Step Three.

**Step Four: Fit the local regression line.**

- This is done by finding  $\beta_0$  and  $\beta_1$  to minimize the weighted sum of squares

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

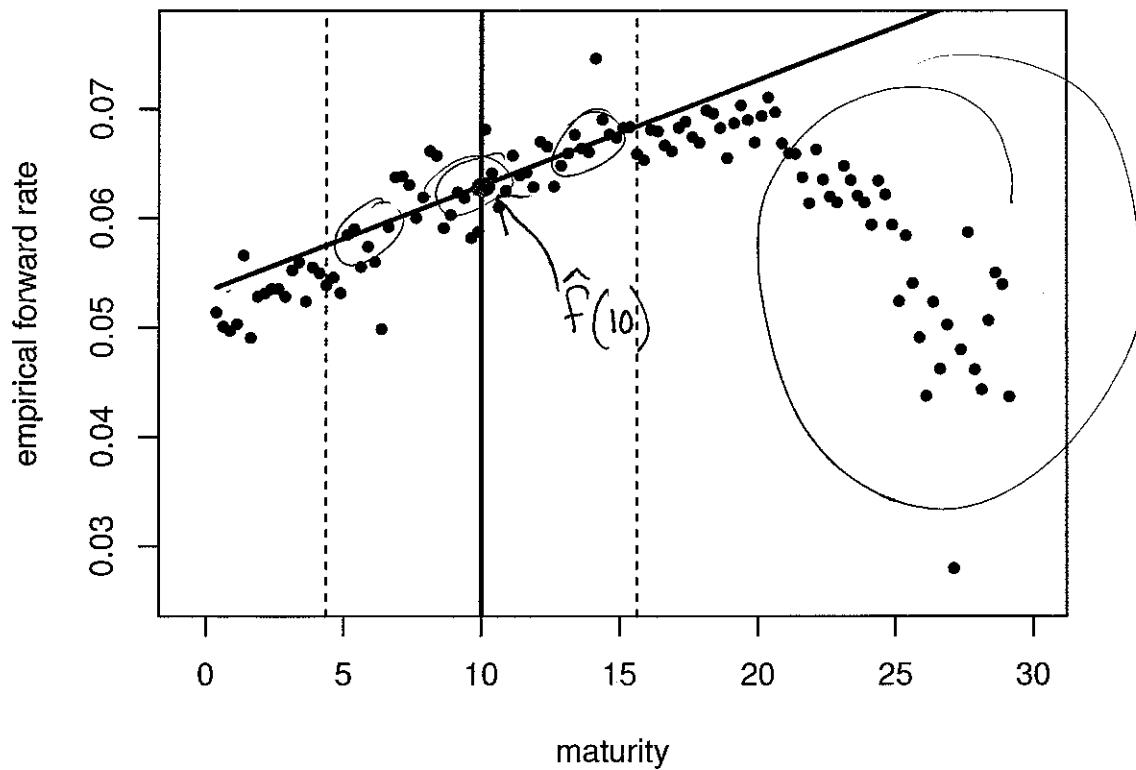


Figure 6: Step Four.

- 1 **Step Five: Estimate  $f(x_0)$ .**
- 2 This is done using the fitted regression line to estimate the regression
- 3 function at  $x_0$ :

- 4 
$$\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$$

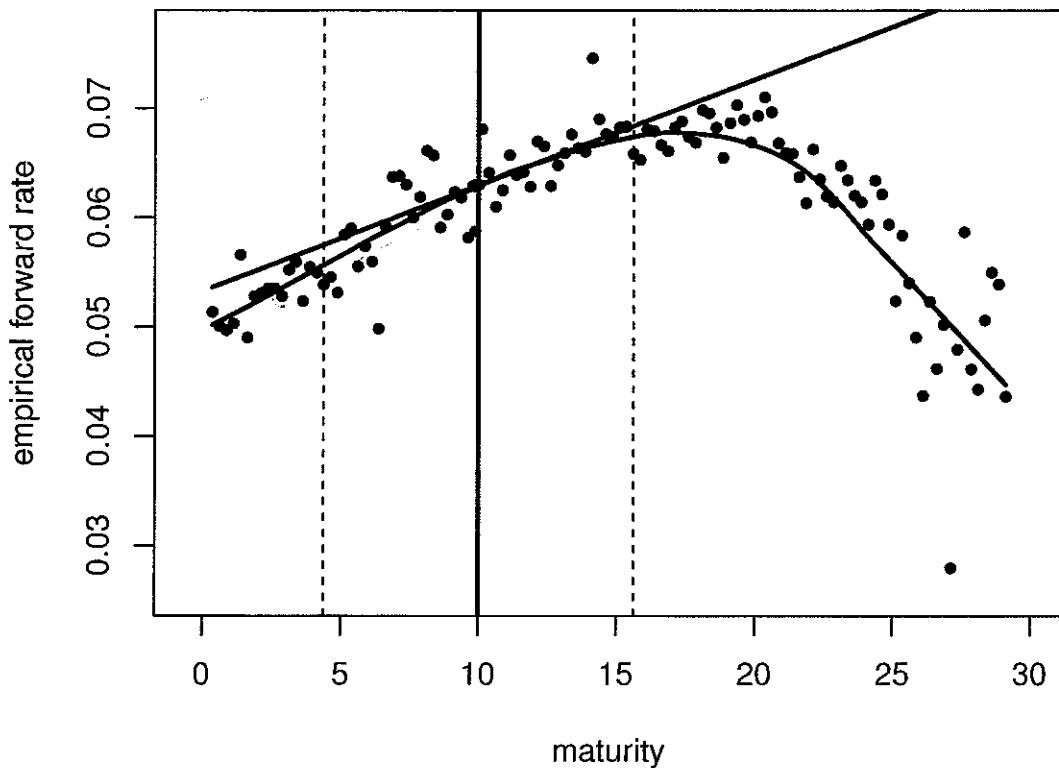


Figure 7: Step Five.

- 1 This is implemented in R using `loess()`:
- 2 > `holdlo = loess(emp ~ maturity, span = 0.4,`
- 3       `degree = 1)`
  
- 4 Setting `degree = 1` is necessary in order to get local linear models.
- 5 You can use the default `degree = 2` to get local quadratic fits, but
- 6 this is usually unnecessary, and is more difficult to interpret.

## 1 Smoothing Parameter Selection

2 There is another R function `loess.as()`, which is part of the pack-  
3 age `fANCOVA`, which has built in selection of the smoothing parame-  
4 ter  $\alpha$  (the span).

5 The syntax is

6 `> holdlo2 = loess.as(maturity, emp, criterion = "gcv")`

$\underbrace{\text{maturity}}_X, \underbrace{\text{emp}}_Y, \underbrace{\text{criterion = "gcv"}}_{\text{criterion}}$

7 With this function the default is `degree = 1`.

8 The choices for `criterion` are `aicc` and `gcv`. `GCV` is an approx-  
9 imation to leave-one-out cross-validation, i.e., an approximation to  
10 `PRESS` is used.

11 **Recommendation:** Use `AIC` (`aicc`) when the sample size is mod-  
12 erate ( $n < 500$ ), and cross-validation (`gcv`) when the sample size is  
13 larger.

14 If I use `AIC`, and I look at `summary(holdlo2)`, I see

15 `span : 0.4749899`

16 which is in line with what we saw as a good choice above.

$$\text{-log-likelihood} \quad \hat{f}(x_i) \quad \hat{\sigma}^2 \quad + 2edf$$

## 1 Making Predictions

- 2 The `predict()` function can be applied to the output of `loess()`.
- 3 The syntax is almost identical to `predict.lm()` used previously,
- 4 except that it does not allow for specifying interval="prediction".
- 5 Instead, to create a prediction interval, do the following:

- 6 1. Use the `predict` function with `se = T`:

7 > holdpreds = predict(holdlo, se = T) SE in  $\hat{f}(x)$

- 8 2. Construct the prediction interval half-width by using the fact that

10 prediction interval half-width =  $z_{\alpha/2} \sqrt{\hat{\sigma}^2 + (\text{SE of fit})^2}$

11 which can be found in R using "error"

12 > halfwidth = qnorm(1-alpha/2) \*

13       sqrt(holdpreds\$residual.scale^2 +

14       holdpreds\$se.fit^2)  $\hat{\sigma}$

- 15 3. Then construct the intervals, centered on the predictions:

16 > lwr = holdpreds\$fit - halfwidth

17 > upr = holdpreds\$fit + halfwidth

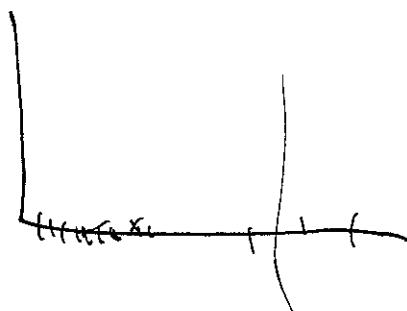
## 1 Comments

2 **Comment 1:** If you want a robust version of this fit, use `family =`  
3 "symmetric" instead of the default `family="gaussian"`. (This  
4 use of "family" is unfortunate. It has nothing to do with the family of  
5 a GLM.) This will work with both `loess()` and `loess.as()`.

6 **Comment 2:** Diagnostic plots based on residuals for this fit are just  
7 like with the classic linear models we fit previously. One can obtain  
8 the residuals and fitted values using `$residuals` and `$fitted`.

9 **Comment 3:** This is naturally extended to higher dimensions, just  
10 include the additional predictors in the model as before. **But:** There  
11 is a limit of four predictors with `loess()` and a limit of two with  
12 `loess.as()`. There is a good reason for this restriction, we will  
13 discuss in the next section.

14 **Comment 4:** The equivalent degrees of freedom can be found `$enp`.



1

## 2 Penalized (Smoothing) Splines

- 3 Another nonparametric approach to regression is the **penalized spline**  
4 or **smoothing spline**.

- 5 This approach starts with a natural optimization problem: Find the  
6 twice differentiable function  $f(\cdot)$  such that

7

$$\sum_{i=1}^n (y_i - f(x_i))^2 + \lambda \underbrace{\int_a^b [f^{(2)}(x)]^2 dx}_{J}$$

- 8 where  $f^{(2)}(x)$  indicates the second derivative of  $f$  evaluated at  $x$  and  
9  $\lambda > 0$  is the smoothing parameter.



- 10 Typically,  $a = \min\{x_i\}$  and  $b = \max\{x_i\}$ .



- 1 Note that the **penalty term**

2

$$\int_a^b [f^{(2)}(x)]^2$$

- 3 will be large if the function is "wiggly." Of course, if  $f(x) = a + bx$ ,  
4 then this penalty equals zero.

- 5 **Exercise:** What is the effect of varying  $\lambda$ ?

6 As  $\lambda$  chosen larger, more emphasis is placed on  
7 keeping penalty small, hence  $\hat{f}$  will be  
8 smoother.

9

10 As  $\lambda \rightarrow 0$ , come closer to interpolating the  
11 data points

12

13

- 14 As before,  $\lambda$  can be chosen via cross-validation or AIC.

- 1 This may appear to set up a difficult optimization problem, but, in  
 2 fact, the search for the optimal  $f(\cdot)$  can be transformed into con-  
 3 structing  $\hat{f}(\cdot)$  as the linear combination of a specially formed **basis**  
 4 of functions.

$$\phi_1(x), \phi_2(x), \dots, \phi_m(x) \quad \hat{f}(x) = \sum_{i=1}^m \hat{\beta}_i \phi_i(x)$$

B-spline basis

- 5 **Comment:** The number of basis functions utilized is equal to the  
 6 number of **knots** plus four. For computational reasons, the number  
 7 of knots is usually chosen much smaller than the number of data  
 8 points when  $n$  is large.

- 9 Figure 8 shows a simple example for the case where there are five  
 10 knots (shown as the vertical lines). See §21.4 and §21.5 in Ruppert for  
 11 details.

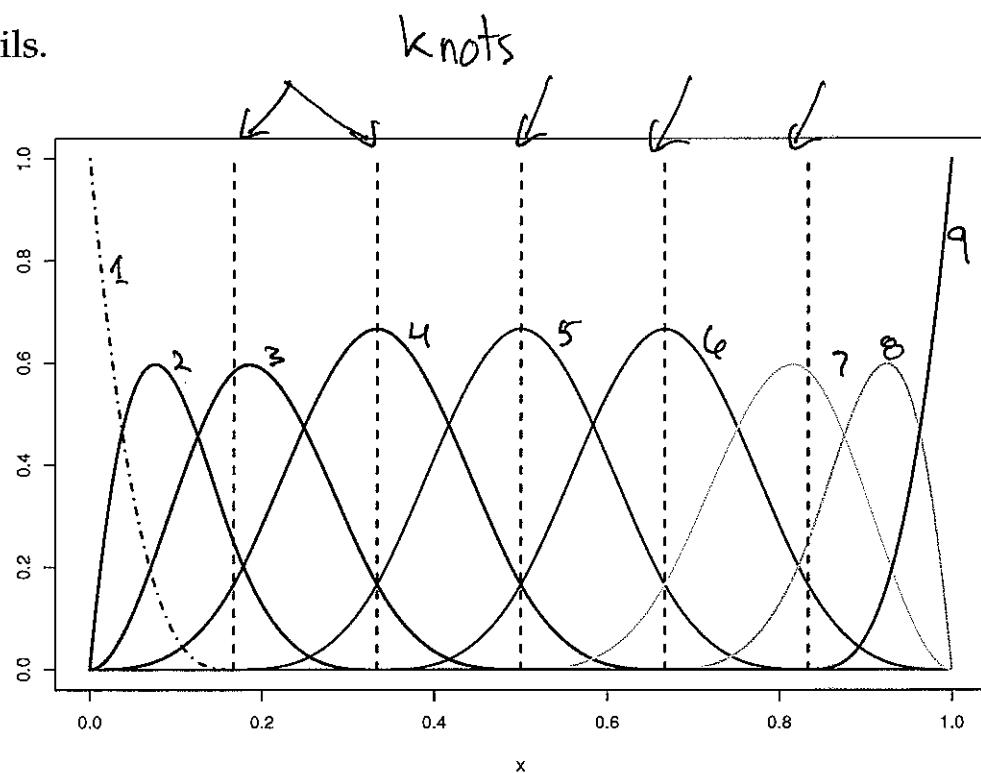


Figure 8: A spline basis (the B-spline basis) for the case with five knots.

- 1 In R, penalized splines are implemented via `smooth.spline()`.
- 2 The default is for `smooth.spline()` to choose  $\lambda$  using GCV, but if
- 3 you set `cv = TRUE`, then true leave-one-out cross-validation is used.
- 4 See Figure 9 for the result of the fit to our simple example.

```

5 > holdspline = smooth.spline(maturity, emp, cv=TRUE)
6 > holdspline
7 Call:
8 smooth.spline(x = maturity, y = emp, cv = TRUE)
9
10 Smoothing Parameter  spar= 0.6074297  lambda= 9.478851e-05 (10 iterations)
11 Equivalent Degrees of Freedom (Df): 12.80447
12 Penalized Criterion: 0.001213719
13 PRESS: 1.349458e-05

```

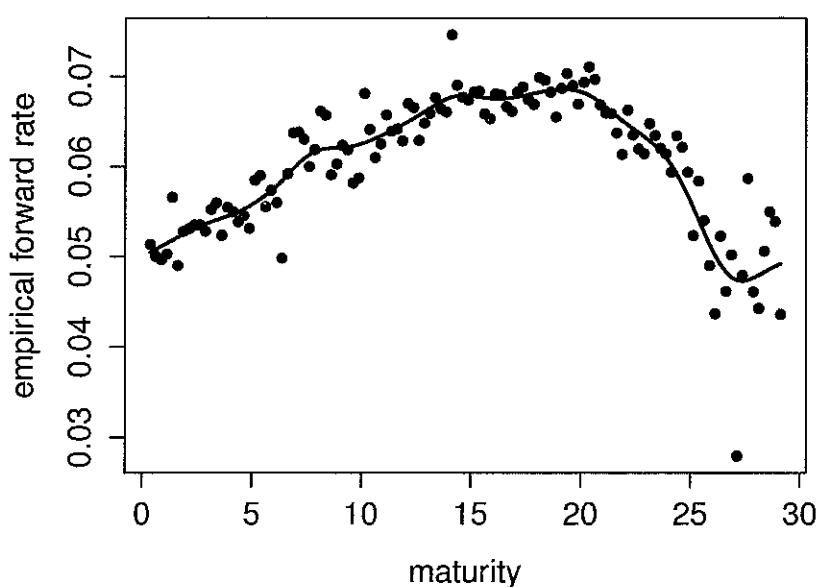


Figure 9: Result of using smoothing spline to fit the data.

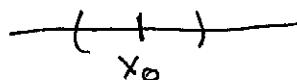
1

---

## 2 The Curse of Dimensionality

3 Despite the great promise of using nonparametric approaches to re-  
4 gression, there is one big challenge: **The Curse of Dimensionality.**

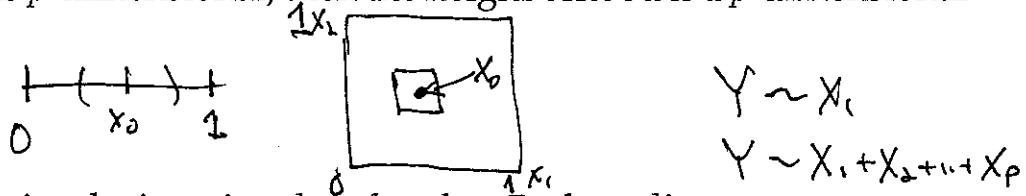
5 In order to see the “curse,” keep in mind that nonparametric regres-  
6 sion relies upon having ample data “in the neighborhood” of the tar-  
7 get  $x_0$ .



8 If there is limited data in a neighborhood, then there is little informa-  
9 tion on which to base a local estimate of the regression function.

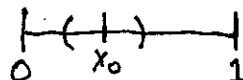
10 If one then chooses to make the neighborhood larger, then the value  
11 of the nonparametric procedure is being lost: The point is to make a  
12 local estimate of the complex function.

- If the predictor is  $p$ -dimensional, then the neighborhood is a  $p$ -dimensional hypercube.

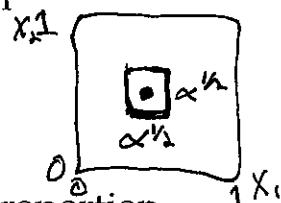


- To make things simple, imagine that the observed predictors are approximately uniformly distributed in a cube with sides  $(0, 1)$ , i.e., the **unit cube**.

- When  $p = 1$ , in order for your neighborhood to cover proportion  $\alpha$  of the data, the neighborhood has to have width  $\alpha$ .



- When  $p = 2$ , in order for your neighborhood to cover proportion  $\alpha$  of the data, it has to be a square with sides of length  $\alpha^{1/2}$ .



- For general  $p$ , in order for your neighborhood to cover proportion  $\alpha$  of the data, it has to be a hypercube with sides of length  $\alpha^{1/p}$ .

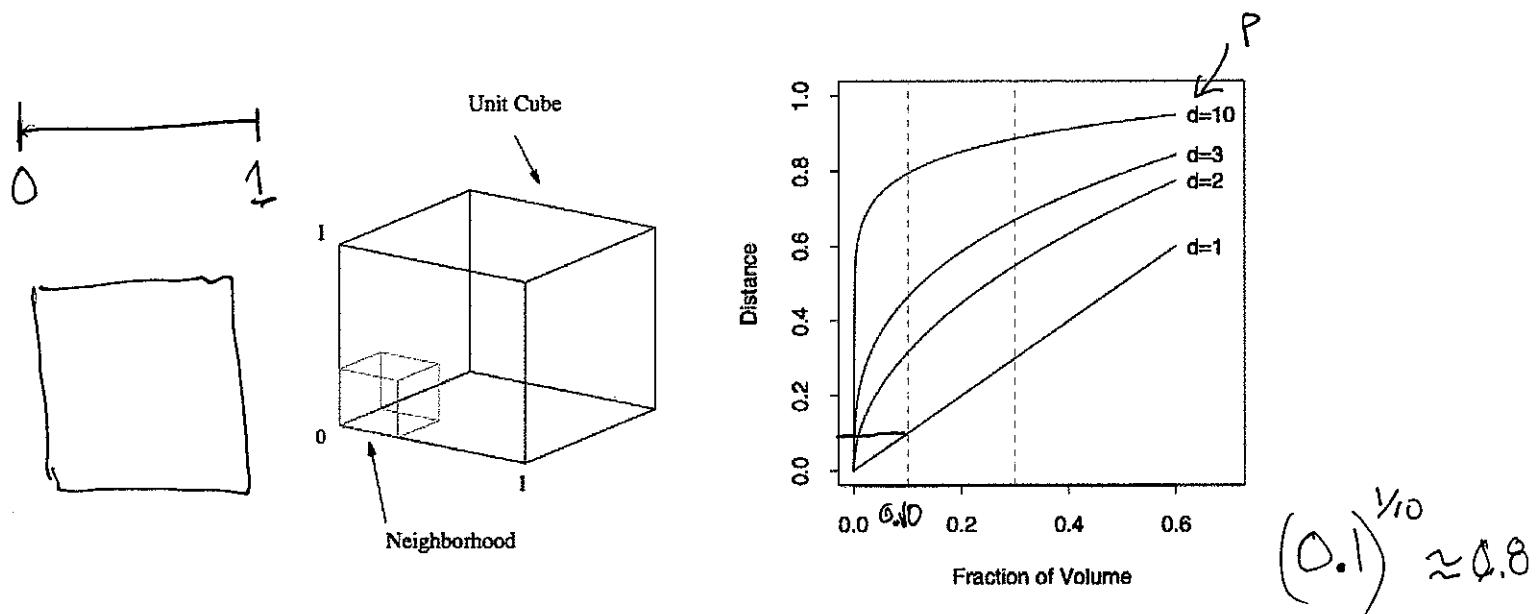
$$(\alpha^{1/p})^p = \alpha$$

- Exercise:** What is happening to the length of the sides of the neighborhoods as  $p$  increases?

As  $p \nearrow \alpha^{1/p} \nearrow 1$  ( $0 < \alpha < 1$ )

- The neighborhood is including a wider range of values for all predictors as  $p \nearrow$

- 1 Figure 2.6 from Hastie, Tibshirani, and Friedman (2009), reproduced
- 2 here as Figure 10 shows the curse.
- 3 Incredibly, when  $p \geq 10$ , the neighborhood has to cover 80% of the
- 4 values for each predictor in order for the neighborhood to include
- 5 just 10% of the data.



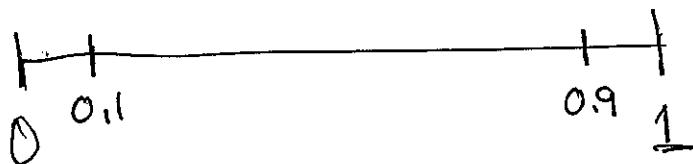
**FIGURE 2.6.** The curse of dimensionality is well illustrated by a subcubical neighborhood for uniform data in a unit cube. The figure on the right shows the side-length of the subcube needed to capture a fraction  $r$  of the volume of the data, for different dimensions  $p$ . In ten dimensions we need to cover 80% of the range of each coordinate to capture 10% of the data.

Figure 10: Figure 2.6 from Hastie, Tibshirani, and Friedman (2009).

Suppose  $p$ -dim predictor space, unit hypercube

Place a point (predictor value) uniformly in this hypercube, what is prob it's close to the edge?

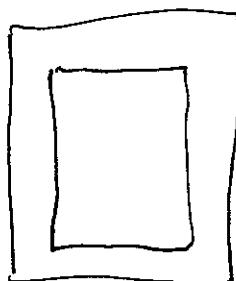
"Close to edge"  $\rightarrow$  within 0.1 of a side



If  $p=1$ , prob. = ~~0.2~~ 0.2

$$p=2, 1 - (0.8)^2 = 0.36$$

for general  $p$ ,  $1 - (0.8)^p$



$$p=10 \quad 0.89$$

$$p=20 \quad 0.99$$

$$p=50 \quad 0.99999$$

## Part 7: Additive Models

- 2 Text references: §7.7 in ISL, Chapter 11 in Hastie, Tibshirani, and  
 3 Friedman (2009)

- 4 A way of avoiding the curse of dimensionality is to make a simplifying  
 5 assumption: Assume that our regression function  $f(\cdot)$  is **additive**  
 6 in the individual predictors  $x_1, x_2, \dots, x_p$ .

- 7 In other words, we assume that

$$8 f(\mathbf{x}) = f_1(x_1) + f_2(x_2) + \dots + f_p(x_p)$$

$(x_1, x_2, \dots, x_p)$

- 9 **Exercise:** Explain why the classic linear model makes the additive  
 10 assumption.

11 
$$11 Y_i = \beta_0 + \beta_1 x_{1i} + \dots + \beta_p x_{pi} + \epsilon_i \quad i=1, 2, \dots, n$$

12 
$$13 f_j(x_j) = \beta_j x_j$$

14

15

16

17

- <sup>1</sup> The additive model that we use here is a compromise between the
- <sup>2</sup> linear model on one extreme and the completely nonparametric model
- <sup>3</sup> at the other extreme.

- <sup>4</sup> **The classic linear model:**

<sup>5</sup> 
$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$$

- <sup>6</sup> **The fully nonparametric model:**

<sup>7</sup> 
$$Y = f(x_1, x_2, \dots, x_p) + \epsilon$$

- <sup>8</sup> with the  $f$  estimated from the data.

- <sup>9</sup> **The additive nonparametric model:**

<sup>10</sup> 
$$Y = \widehat{\beta_0} + f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p) + \epsilon$$

- <sup>11</sup> with each of the  $f_i$  **estimated from the data.** (Each  $f_i$  is shifted so that
- <sup>12</sup> it is centered around zero. The intercept  $\beta_0$  accounts for the overall
- <sup>13</sup> mean of the response.)

- <sup>14</sup> Keep in mind that, in this notation, each of the  $x_i$  could be user-defined functions of the original predictors, e.g.  $x_i$  could be the interaction between two of the original variables.

- 1 We will now focus on the **additive nonparametric model**:

2 
$$Y = f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p) + \epsilon$$

- 3 with each of the  $f_i$  **estimated from the data**.

- 4 The general estimation strategy is called **backfitting**.

- 5 In this process, each  $f_k$  is estimated nonparametrically, in a rotation,  
6 and the process is repeated until there is convergence.

- 7 When estimating  $f_k(\cdot)$ , the other  $f_j(\cdot)$  are held fixed at their current  
8 best estimates, and we set up a one-dimensional nonparametric es-  
9 timation problem, on which one could use either local linear regres-  
10 sion, smoothing splines, or other approach.

- 11 So, when estimating  $f_k(\cdot)$ , the response is taken to be

12 
$$Y_i - \left[ \sum_{j \neq k} \hat{f}_j(x_{ij}) \right]$$

<sup>1</sup> **The Generalized Additive Model (GAM)**

- <sup>2</sup> This idea can be taken a step further. Recall that the generalized  
<sup>3</sup> linear model (GLM) is built around the **linear predictor** and the as-  
<sup>4</sup> sociated **link function**  $g(\cdot)$ :

<sup>5</sup> 
$$g(E(Y)) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p.$$

- <sup>6</sup> A natural extension is to consider models of the form

<sup>7</sup> 
$$g(E(Y)) = \beta_0 + f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p).$$

- <sup>8</sup> This is called the **generalized additive model (GAM)**.
- <sup>9</sup> With the GAM, we still have to specify the family of the response,  
<sup>10</sup> and in many ways the model is identical to the GLM.

## 1 The GAM in R

- 2 There are two packages which include a function `gam()`.
- 3 In both cases, the syntax is much like when using `glm()`. The key  
4 difference is that one must specify which of the predictors are to be  
5 fit nonparametrically. This is done by wrapping the predictor with a  
6 special function.
- 7 For the function `gam()` which is part of the package `gam`, there are  
8 two options: A function `s()` for fitting a smoothing spline, and a  
9 function `lo()` for fitting a local polynomial.
- 10 For the function `gam()` which is part of the package `mgcv`, there is  
11 only the option `s()`.

1 In either case, the syntax would appear simply as

2 > holdgam = gam(y ~ s(x1) + s(x2) + x3)

3 In this example, the model being fit is

4 
$$Y = \beta_0 + f_1(x_1) + f_2(x_2) + \beta_3 x_3 + \epsilon$$
  
$$\beta_0 x_0 \quad \beta_3 x_3$$

5 with  $f_1$  and  $f_2$  being estimated from the data using a smoothing  
6 spline.

7 Another example:

8 > holdgam = gam(y ~ lo(x1, x2) + lo(x3))

9 In this example, the model being fit is

10 
$$Y = \beta_0 + f_1(x_1, x_2) + f_3(x_3) + \epsilon$$

11 with  $f_1$  and  $f_3$  being estimated from the data using a local linear re-  
12 gression.

13 **The advantage to using `gam()` from `mgcv`** is that it incorporates  
14 automatic selection of the smoothing parameters for the individual  
15 smoothing splines.

<sup>1</sup> **Example**

<sup>2</sup> In this example we will predict the price of a call option using the  
<sup>3</sup> following predictors:

<sup>4</sup> 1. The time to expiration for the option.

<sup>5</sup> 2. The strike price for the option.

<sup>6</sup> 3. The historical volatility for the underlying equity, using the past  
<sup>7</sup> two months of data.

<sup>8</sup> 4. The current price of the equity.

<sup>9</sup> The data set consists of 1,339 call options that were sampled on the  
<sup>10</sup> evening of February 8, 2013. These are all from NYSE-listed stocks.

<sup>11</sup> Data were sampled randomly from all possible stocks and expiry  
<sup>12</sup> dates.

<sup>13</sup> Note that since the data were all taken from the same date, there is  
<sup>14</sup> serious question as to how well this model would extend to other  
<sup>15</sup> dates. Yet, the model could be useful, for example, to identify poten-  
<sup>16</sup> tially undervalued options. See the introduction to Hutchinson, Lo,  
<sup>17</sup> and Poggio (1994) for more justification for taking this approach to  
<sup>18</sup> options pricing.

- Figure 1 shows the distribution of the predictors.

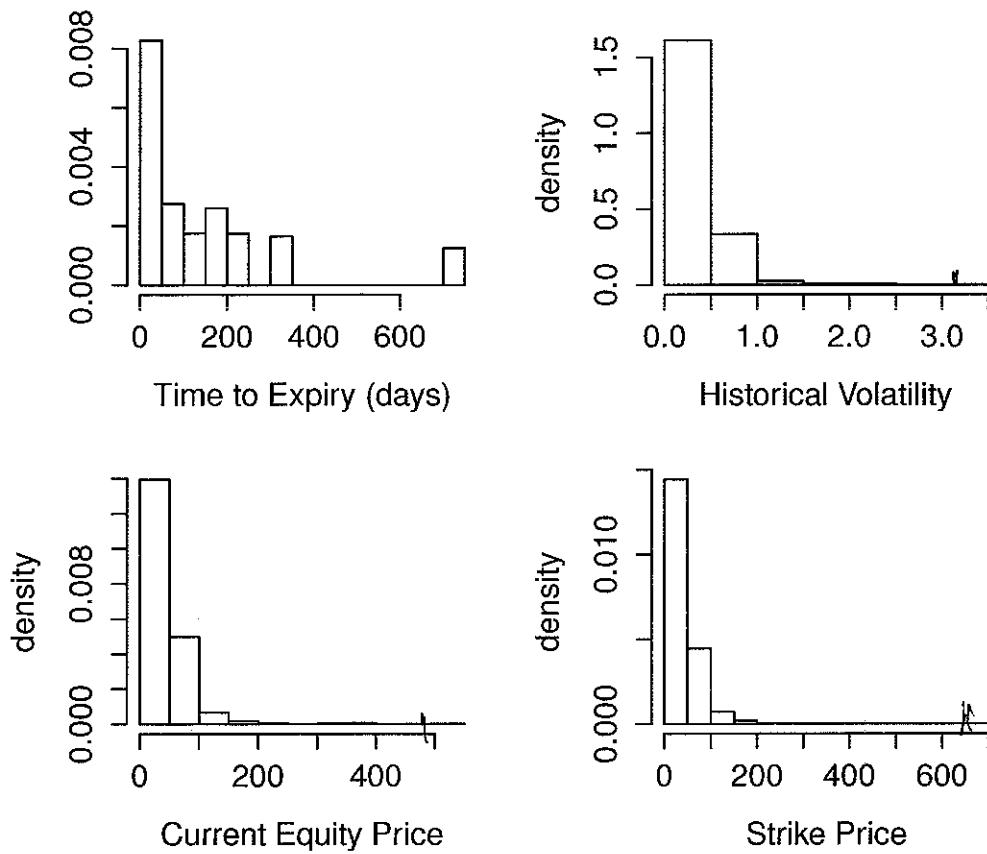


Figure 1: Histograms showing the distributions of the four predictors.

- Exercise:** Why is transforming these predictors a good idea?

Because of the skew, I will use log transform on each.

5 \_\_\_\_\_

6 \_\_\_\_\_

7 \_\_\_\_\_

8 \_\_\_\_\_

Figure 2 shows the result after log transformation.

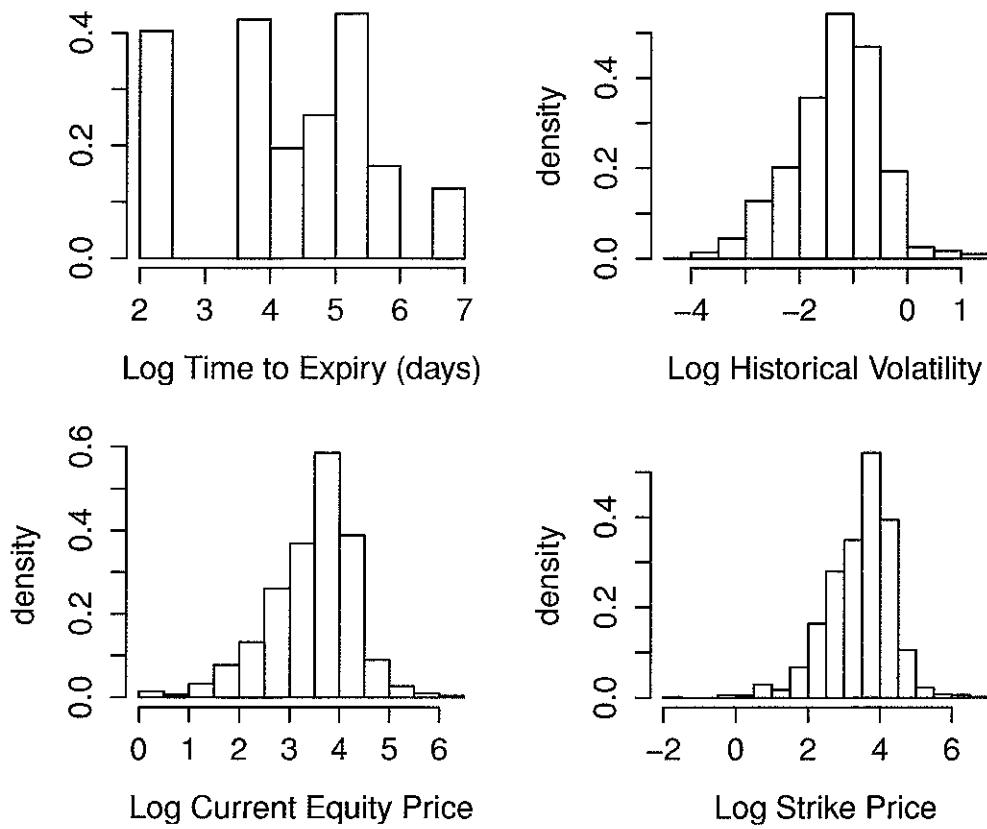


Figure 2: Histograms showing the distributions of the four predictors after log transformation.

- 1 We will now fit a sequence of models. It is usually a good idea to
- 2 start with a simple model, and then build up complexity as it is war-
- 3 ranted.
  
- 4 **Model One** includes the four log-transformed predictors in a linear
- 5 model to predict the price.
  
- 6 The result of the fit is shown below.

```
7 glm(formula = last ~ log(timetoexpiry) + log(strike) + log(curprice) +
8   log(histvol), data = alldat)
9
10 Deviance Residuals:
11      Min       1Q   Median       3Q      Max
12 -22.730   -2.231   -0.757    0.996   73.495
13
14 Coefficients:
15                               Estimate Std. Error t value Pr(>|t|)
16 (Intercept)                 -2.0092    1.4262 -1.409   0.159
17 log(timetoexpiry)          0.7499    0.1004  7.467 1.48e-13 ***
18 log(strike)                -9.4604   0.4950 -19.113 < 2e-16 ***
19 log(curprice)              10.7455   0.5890 18.243 < 2e-16 ***
20 log(histvol)                1.7006   0.3139  5.418 7.14e-08 ***
21 ---
22 Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
23
24 (Dispersion parameter for gaussian family taken to be 24.72016)
25
26 Null deviance: 50651  on 1338  degrees of freedom
27 Residual deviance: 32977  on 1334  degrees of freedom
28 AIC: 8101.9
29
30 Number of Fisher Scoring iterations: 2
```

- 1 The plot of residuals versus fitted values is shown as Figure 3.

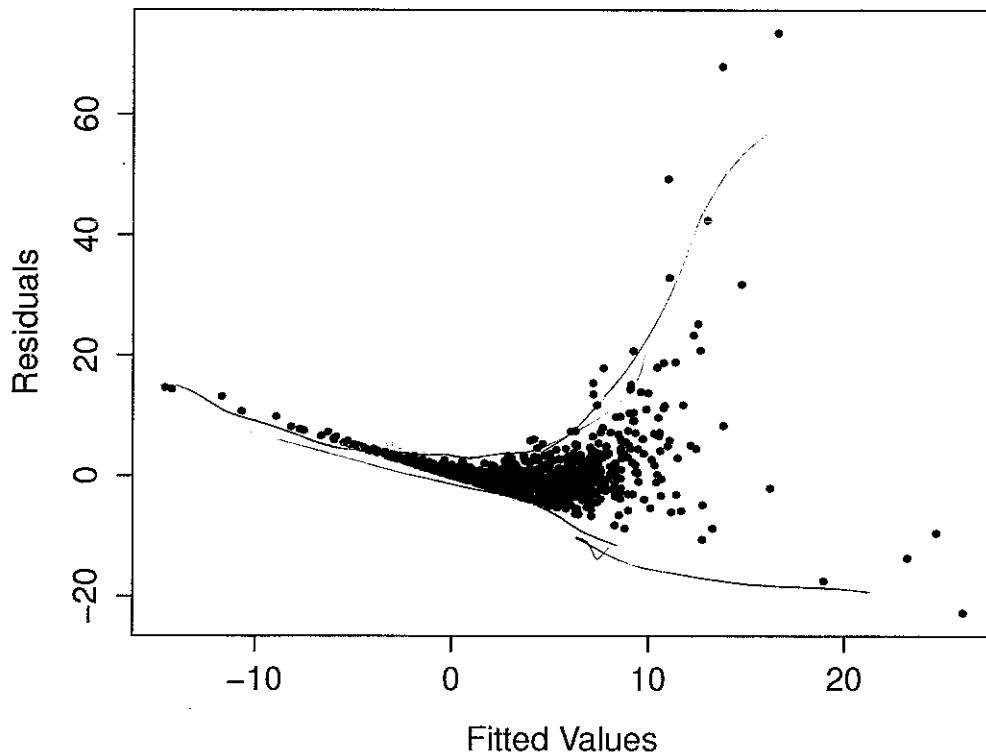


Figure 3: Plot of residuals versus fitted values for Model One.

- 2 **Exercise:** Comment on Figure 3.

3 Clear evidence of lack of fit.

---

4 \_\_\_\_\_

5 \_\_\_\_\_

6 \_\_\_\_\_

7 \_\_\_\_\_

8 \_\_\_\_\_

- 1 Next, I want to consider a transformation of the response. I use the
- 2 `boxCox()` function, and the resulting plot is shown as Figure 4.

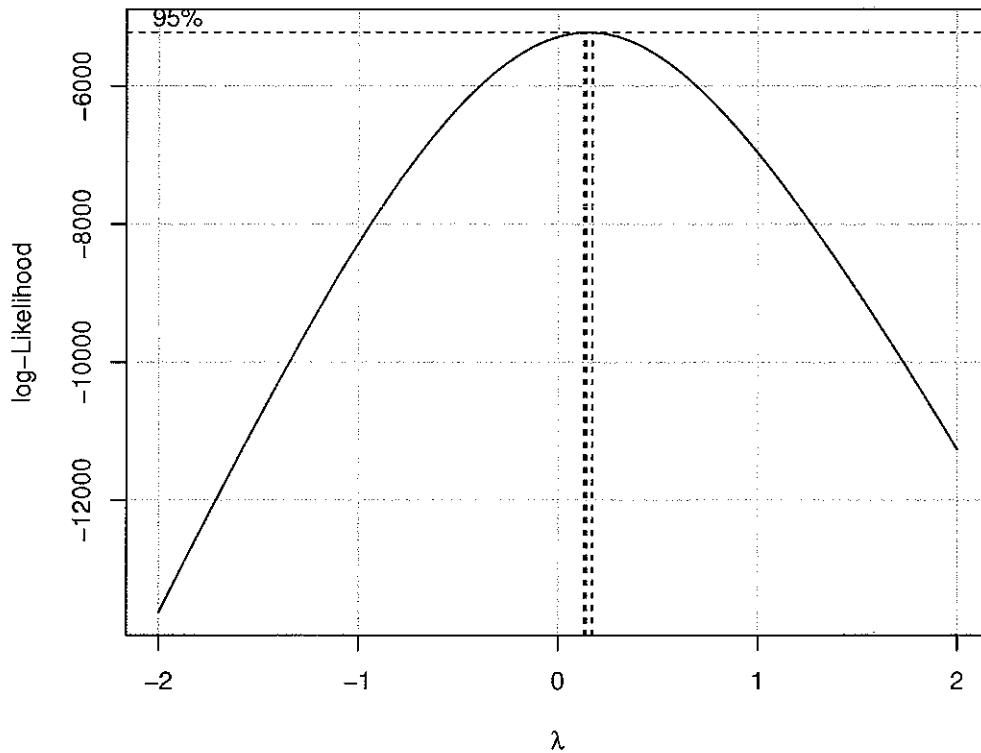


Figure 4: The plot created by `boxCox()` for our example.

- 3 **Exercise:** What conclusion do you draw from 4?

$\lambda \approx 0.25$  is best choice  $\Rightarrow y^{1/4}$  as new

response

- 4
- 5
- 6
- 7
- 8

- 1 Model Two is again a linear model, but with the response transformed by raising it to the power 1/4.
- 2
- 3 The result is shown below.

```

4 Call:
5   glm(formula = transresp ~ log(timetotoexpiry) + log(strike) + log(curprice) +
6       log(histvol), data = alldat)
7
8 Coefficients:
9
10              Estimate Std. Error t value Pr(>|t|)
11 (Intercept)      0.55051   0.08052   6.837 1.23e-11 ***
12 log(timetotoexpiry) 0.08462   0.00567  14.925 < 2e-16 ***
13 log(strike)      -0.92714   0.02795 -33.176 < 2e-16 ***
14 log(curprice)     1.03491   0.03325  31.121 < 2e-16 ***
15 log(histvol)      0.11066   0.01772   6.245 5.69e-10 ***
16 ---
17 Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
18
19 (Dispersion parameter for gaussian family taken to be 0.07879699)
20
21 Null deviance: 245.45 on 1338 degrees of freedom
22 Residual deviance: 105.12 on 1334 degrees of freedom
23 AIC: 404.67

```

Can't compare with Model 1 AIC because response has changed

- The plot of residuals versus fitted values is shown as Figure 5.

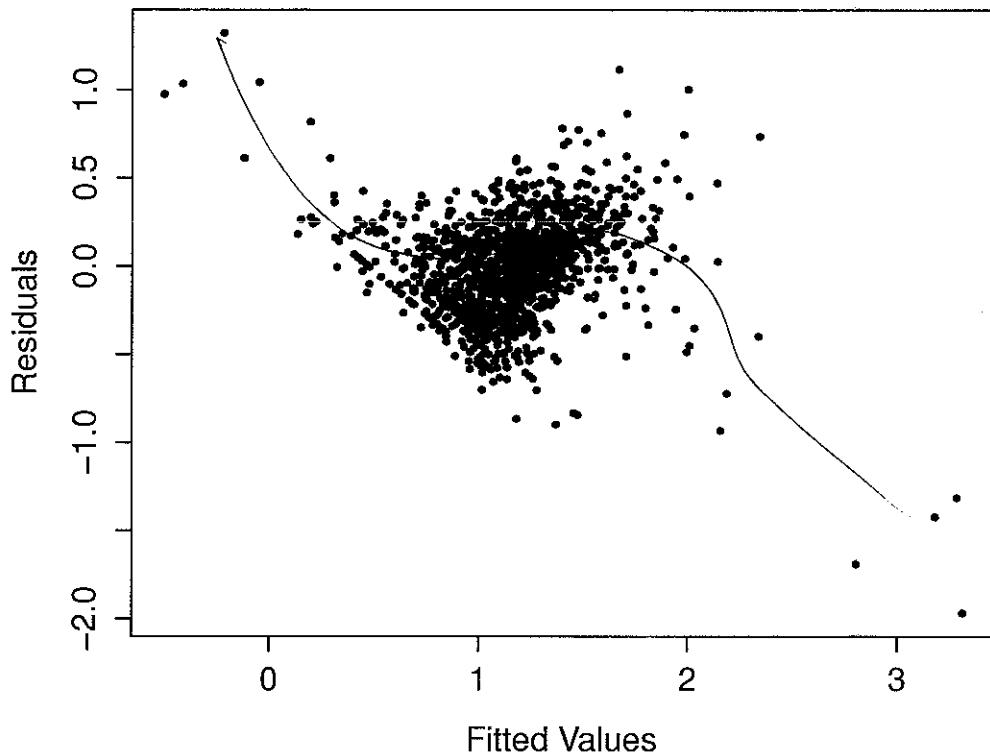


Figure 5: Plot of residuals versus fitted values for Model Two.

Seems better, but still evidence of lack of fit.

- 1 Figure 6 is also useful; it compares the predictions from the model
- 2 with the response for each of the observations.
- 3 Ideally, the points would scatter around the solid line.

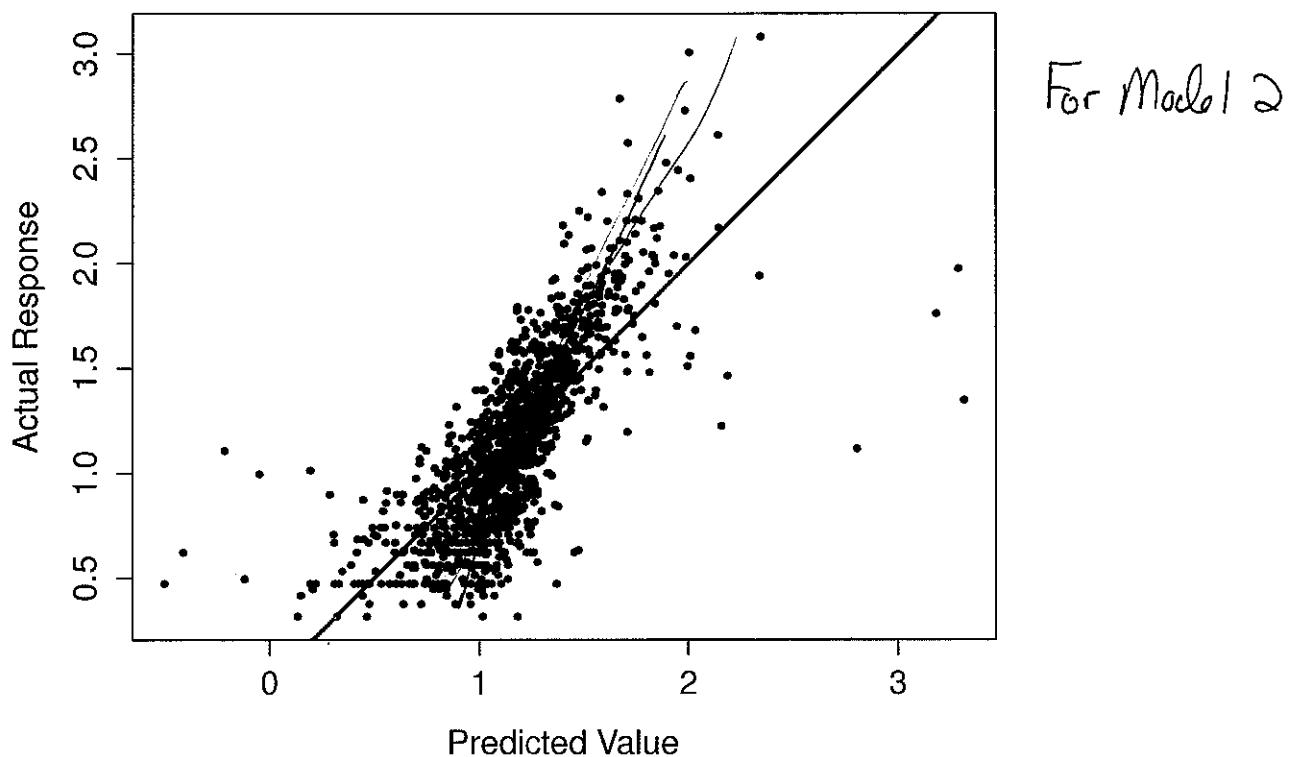


Figure 6: Plot of the response versus the predictions (fitted values) for Model Two. The solid line is the line of perfect agreement between the predictions and the observed responses.

- 4 These two plots reveal that further improvement is definitely needed.

- 1 **Model Three** is an additive model of the form
- 2 
$$\text{price}^{1/4} = \beta_0 + f_1(\log(\text{time to expiry})) + f_2(\log(\text{strike price}))$$
- 3 
$$+ f_3(\log(\text{current price})) + f_4(\log(\text{historical vol})) + \epsilon$$
- 4 **Exercise:** It is still a good idea to use the log-transformed predictors
- 5 in this additive model. Why?

6 It ~~removes~~ reduces regions in the predictor space  
7 with very few observations in the  
8 training set. It's difficult to estimate  
9 nonparametric regression in such regions.  
10 \_\_\_\_\_  
11 \_\_\_\_\_  
12 \_\_\_\_\_  
13 \_\_\_\_\_  
14 \_\_\_\_\_  
15 \_\_\_\_\_  
16 \_\_\_\_\_

1 The result of the fit is shown below

```

2 Family: gaussian
3 Link function: identity
4
5 Formula:
6 transresp ~ s(log(timetoexpiry)) + s(log(strike)) + s(log(curprice)) +
7 s(log(histvol))
8
9 Parametric coefficients:
10 Estimate Std. Error t value Pr(>|t|)
11 (Intercept) 1.153187 0.006278 183.7 <2e-16 ***
12 ---
13 Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
14
15 Approximate significance of smooth terms:
16 edf Ref.df F p-value
17 s(log(timetoexpiry)) 2.754 3.310 125.368 < 2e-16 ***
18 s(log(strike)) 8.396 8.866 249.566 < 2e-16 ***
19 s(log(curprice)) 8.161 8.809 221.922 < 2e-16 ***
20 s(log(histvol)) 5.340 6.593 5.688 3.72e-06 ***
21 ---
22 Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
23
24 R-sq.(adj) = 0.712 Deviance explained = 71.8%
25 GCV score = 0.0538 Scale est. = 0.05277 n = 1339

```

- 1 Comments on the output of `gam()`:
- 2 **Comment 1:** The table includes estimates of the degrees of freedom associated with each smoothed term (the column `edf`). These are each larger than one, because the smooths have more “freedom” than a line to fit to the patterns in the data.
- 6 **Comment 2:** The p-values in the table of smoothing terms are for the test of the null hypothesis that the associated function equals zero for all values of the predictor. In other words, it is testing if there is strong evidence of any relationship between the predictor and the response.

RSS

- 11 **Comment 3:** The residual deviance for the fit can be found from `model3$deviance`. The residual degrees of freedom for the fit is found at `model3$df.residual`. In this case, those quantities are 14 69.31 and 1313.349, respectively.

`` $n - \# \text{ of equivalent parameters}$

- 15 **Comment 4:** AIC for the model fit is found at `model3$aic`. This 16 value is this case is  $-111.7749$ .

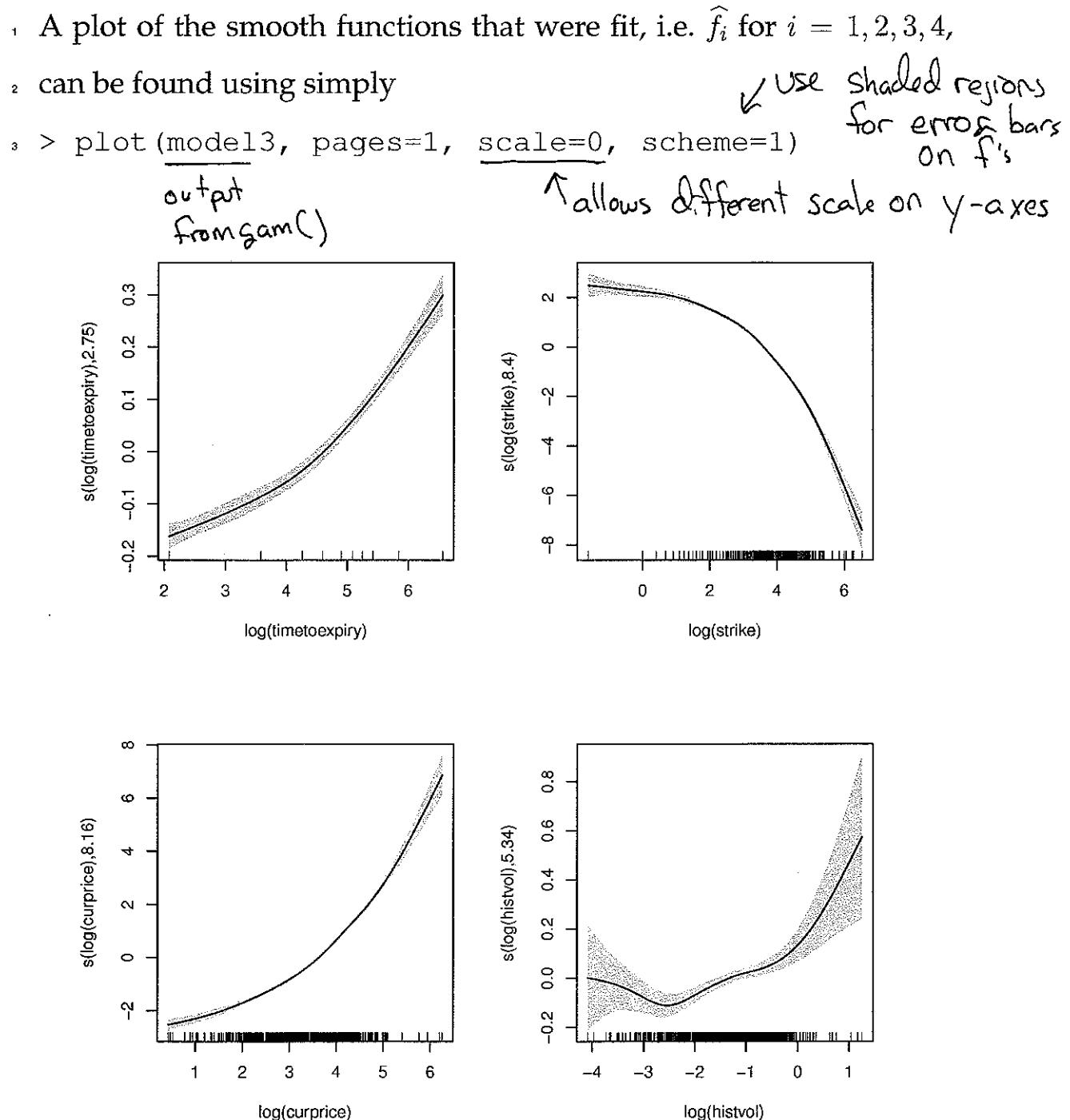
Figure 7: Plot of the estimated  $f_i$  functions.

Figure 8 shows the plot of residuals versus fitted values for this fit.

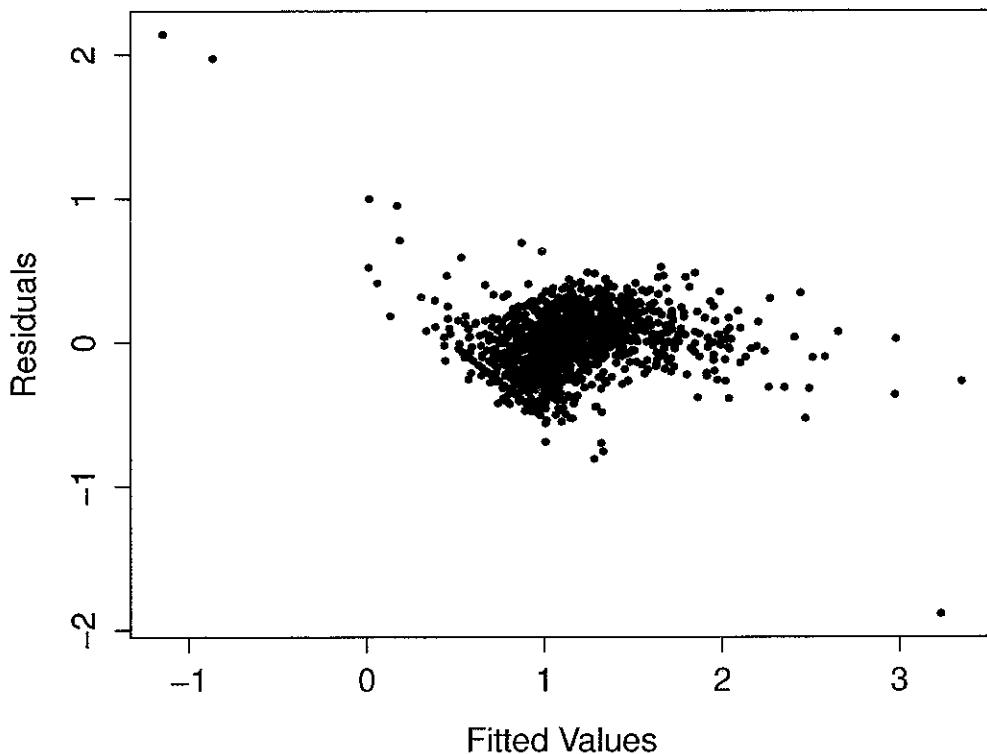


Figure 8: Plot of residuals versus fitted values for Model Three.

- Figure 9 shows the response versus the predictions for Model Three.

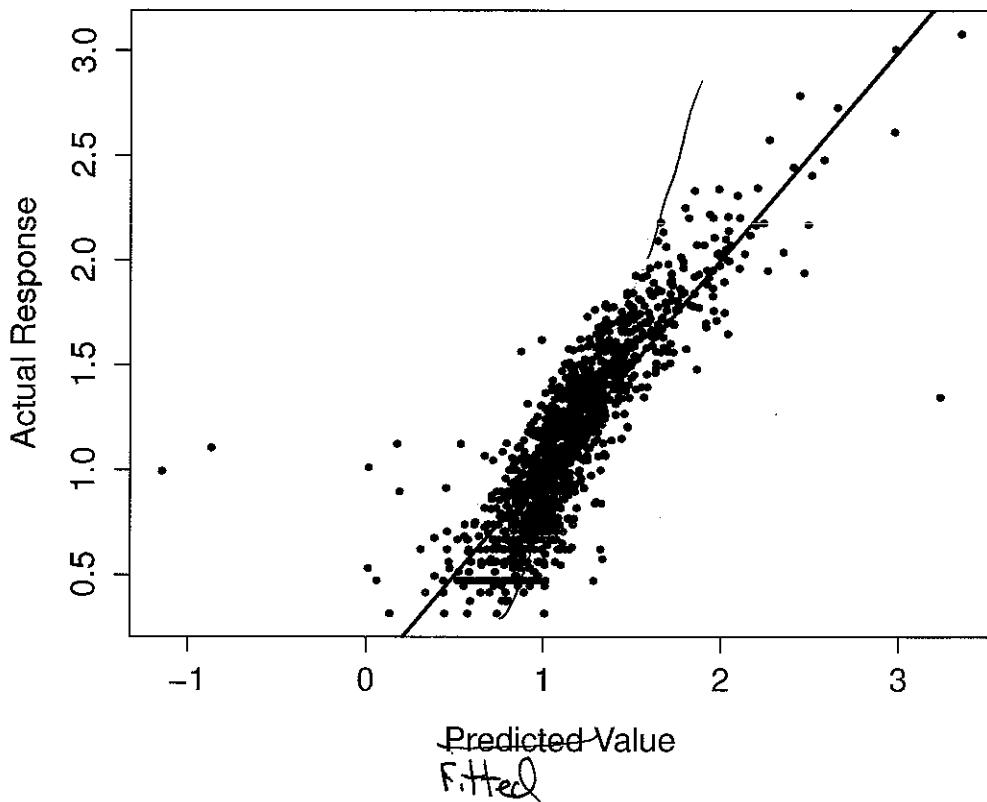


Figure 9: Plot of the response versus the predictions (fitted values) for Model Three. The solid line is the line of perfect agreement between the predictions and the observed responses.

- These two plots show that a lot of improvement has been made, but
  - there still seems like there is room for improving the model.

two-way nonparametric fits

- I considered all possible ~~interactions~~ between these four predictors,
  - and the only choice that reduced AIC was that between "current price" and "strike price."

1 **Model Four:** is an additive model of the form

2 
$$\text{price}^{1/4} = \beta_0 + f_1(\log(\text{time to expiry})) + f_2(\log(\text{strike price}))$$
  
3 
$$+ f_3(\log(\text{current price})) + f_4(\log(\text{historical vol}))$$
  
4 
$$+ f_5(\log(\text{strike price}), \log(\text{current price})) + \epsilon$$

5 The result of this fit is shown below

6 Formula:  
7 
$$\text{transresp} \sim s(\log(\text{timetoexpiry})) + s(\log(\text{strike})) + s(\log(\text{curprice})) +$$
  
8 
$$s(\log(\text{histvol})) + s(\log(\text{strike}), \log(\text{curprice}))$$
  
9  
10 Parametric coefficients:  
11 

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	1.153187	0.004848	237.9	<2e-16 ***

  
12 
$$---$$
  
13  
14  
15 Approximate significance of smooth terms:  
16 

	edf	Ref.df	F	p-value
s(log(timetoexpiry))	7.546	8.242	82.175	< 2e-16 ***
s(log(strike))	7.968	8.777	7.954	3.42e-11 ***
s(log(curprice))	8.619	8.937	10.191	3.60e-15 ***
s(log(histvol))	7.969	8.736	6.047	4.31e-08 ***
s(log(strike), log(curprice))	27.000	27.000	33.088	< 2e-16 ***

  
17 
$$---$$
  
18  
19  
20  
21  
22  
23 Signif. codes: 0 \*\*\* 0.001 \*\* 0.01 \* 0.05 . 0.1 1  
24  
25 R-sq. (adj) = 0.828 Deviance explained = 83.6%  
26 GCV score = 0.032955 Scale est. = 0.031475 n = 1339

Figure 10 shows the plot of residuals versus fitted values for this fit.

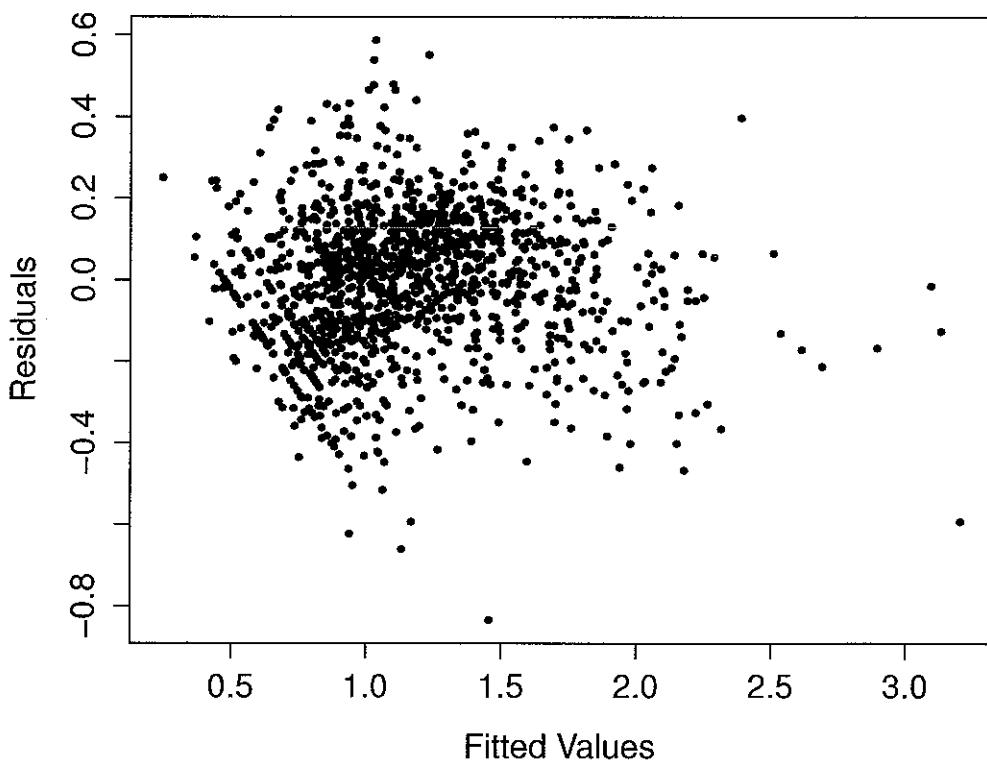


Figure 10: Plot of residuals versus fitted values for Model Four.

- Figure 11 shows the response versus the predictions for Model Four.

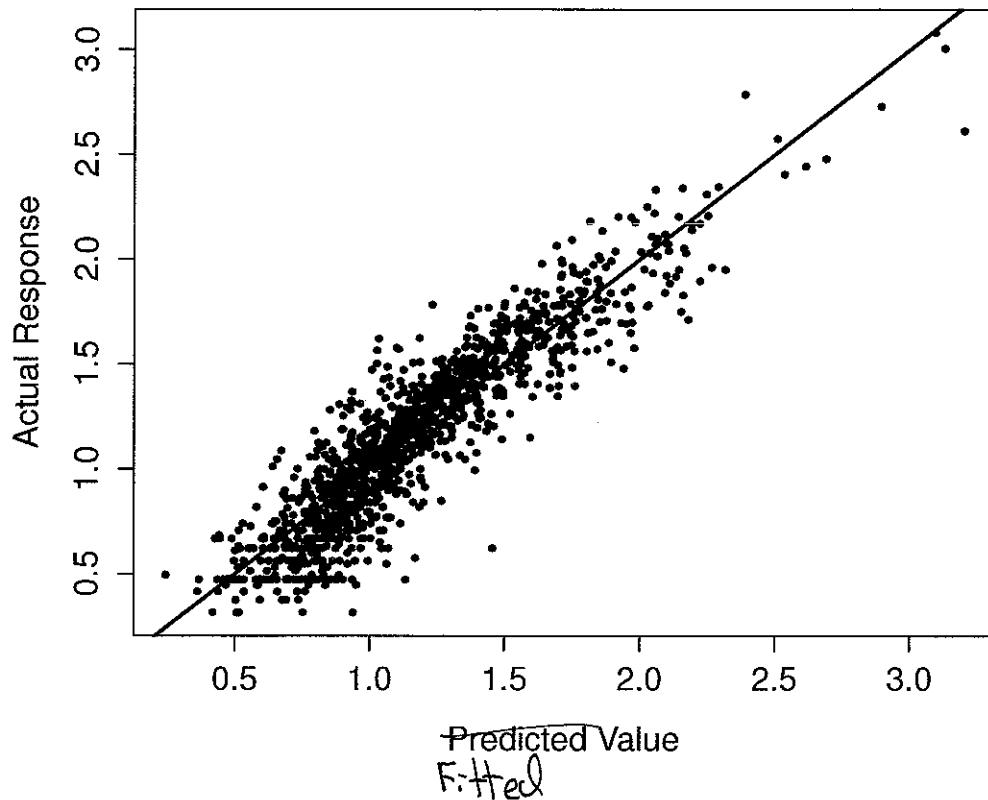


Figure 11: Plot of the response versus the predictions (fitted values) for Model Four. The solid line is the line of perfect agreement between the predictions and the observed responses.

- Figure 12 shows the functions that were fit in this case.

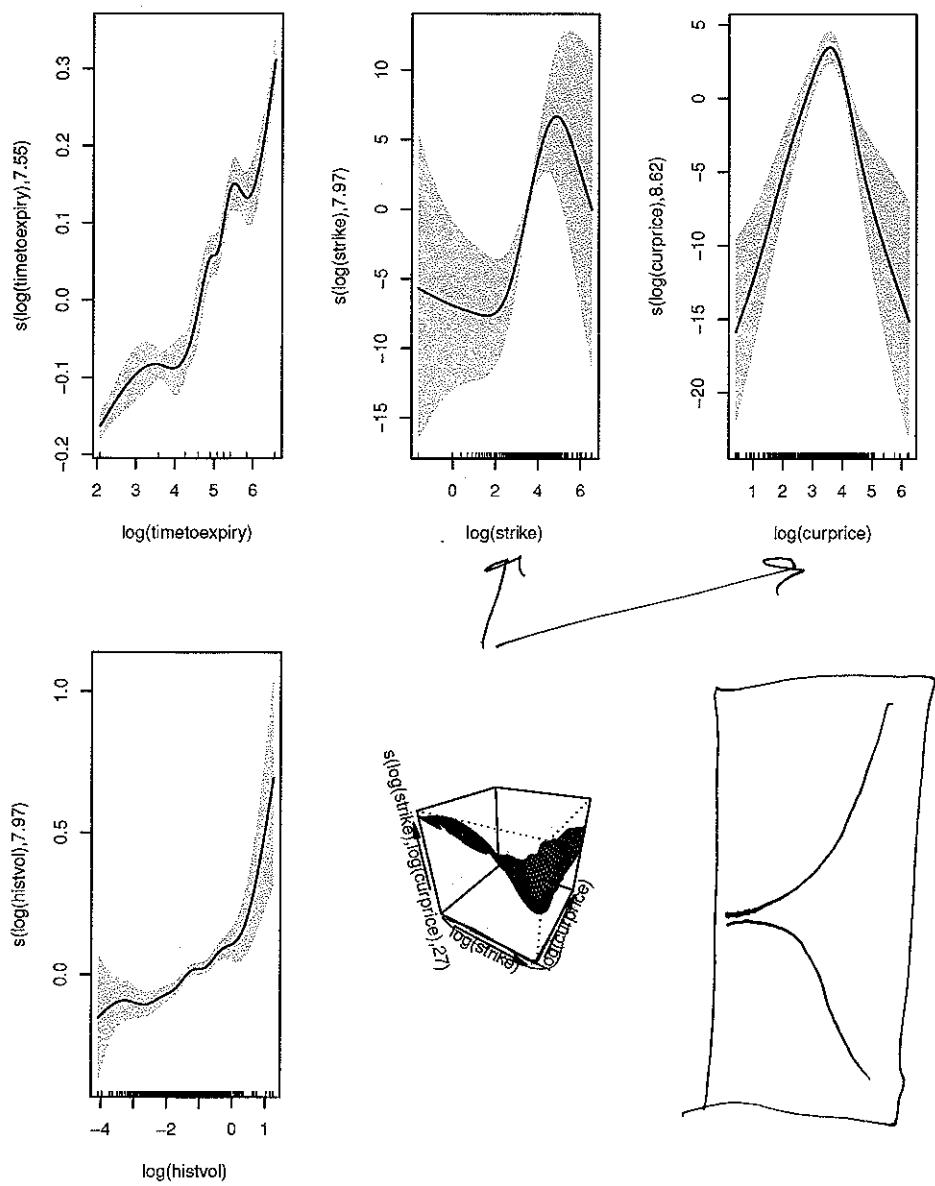
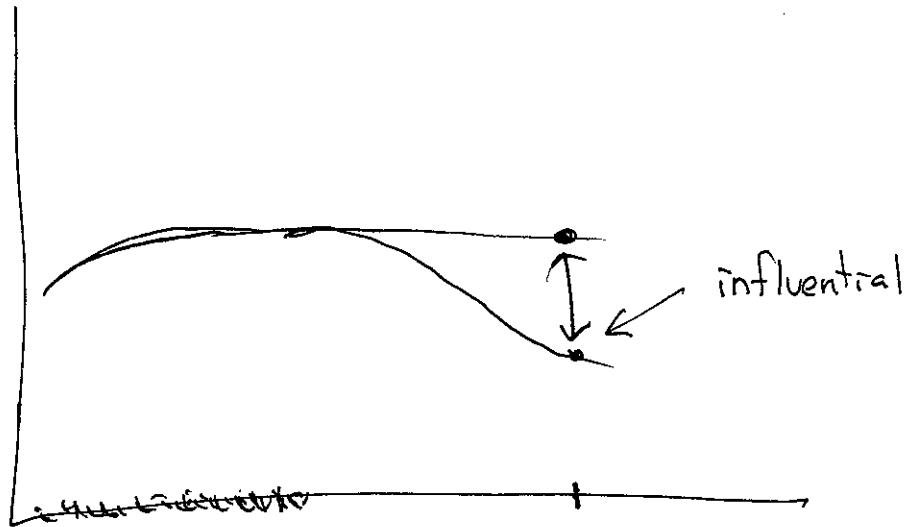


Figure 12: Plot of the estimated  $f_i$  functions for Model Four.



$$\sum_j \left( \hat{y}_{(-i)j} - \hat{y}_j \right)^2$$

## 1 Making Predictions

2 The process for creating prediction intervals with `gam()` is much like  
 3 that described above for use with `loess()`:

4 1. Use the `predict` function with `se = T`:

5 > `holdpreds = predict(model4, se = T)`

6 2. Construct the prediction interval half-width by using the fact  
 7 that

$$8 \text{ prediction interval half-width} = z_{\alpha/2} \sqrt{\hat{\sigma}^2 + (\text{SE of fit})^2}$$

9 which can be found in R using

10 > `halfwidth = qnorm(1-alpha/2) *`  
 11 `sqrt(model4$sig2 + holdpreds$se.fit^2)`

12 3. Then construct the intervals, centered on the predictions:

13 > `lwr = holdpreds$fit - halfwidth`  
 14 > `upr = holdpreds$fit + halfwidth`

- 1 **A final comment:** Keep in mind that this additive model form can
- 2 be used with any GLM. For instance, one could fit a logistic regres-
- 3 sion model in which the linear predictor is replaced with a sum of
- 4 nonparametric fits.

$\text{family} = \text{Binomial}$

1

## 2 Projection Pursuit Regression

- 3 If one fits an additive model using the function `gam()` with family  
4 = gaussian (which is the default), then the assumption is that

5

$$Y = \beta_0 + \sum_{j=1}^p f_j(x_j) + \epsilon$$

- 6 where  $\epsilon$  is normally distributed with mean zero and variance  $\sigma^2$ . Re-  
7 call that the  $f_j$  are estimated nonparametrically.

- 8 A generalization of this model is the **projection pursuit regression**  
9 model, which can be written as

10

$$Y = \widehat{\beta_0} + \sum_{k=1}^M \beta_k f_k(\alpha_k^T \mathbf{x}) + \epsilon$$

$\alpha_k, f_k, \beta_k$   
are all estimated  
from data

- 11 where each of the  $\alpha_k$  are a vector of length  $p$ . These  $\alpha_k$  are the **pro-**  
12 **jection direction vectors**.

- 13 The functions  $f_k$ , called the **ridge functions**, are estimated nonpara-  
14 metrically. These functions are scaled to have mean zero and vari-  
15 ance one when applied to the observed sample.

- 1 The projection pursuit model takes linear combinations of the pre-  
2 dictors, and then fits an additive model in these linear combinations.  
3 One can think of the  $\alpha_k^T x$  as being "new predictors" which are being  
4 utilized in an additive model.  $k=1, 2, \dots, m$
- 5 **Exercise:** Explain how the projection pursuit model generalizes the  
6 model used by `gam()`.

7 If  $M=p$  and  $\alpha_k = 1$  for only position  $k$ ,

8 0 otherwise. Then  $\alpha_k^T x = x_k$ , ie,

9  $f_k(\alpha_k^T x) = f_k(x_k)$

10

11

12

13

14

15

16

17

- 1 **Exercise:** How is the projection pursuit model related to using PCA
- 2 in conjunction with `gam()`?

3 With PCA, also creating predictors from linear  
4 combs. of original preds, but those  
5 combs. are formed without any influence  
6 or knowledge of the response.

7 With PPR, seek linear combs. that lead  
8 to a better model fit, i.e. lower  
9 deviance.

10

---

11

---

12

---

13

---

## 1 Fitting a Projection Pursuit Model

- 2 Projection pursuit regression models are fit using least squares, i.e., the
- 3 objective is to find parameters  $\beta_k$ , functions  $f_k$  and vectors  $\alpha_k$  to min-
- 4 imize

$$5 \quad RSS = \sum_{i=1}^n \left[ Y_i - \beta_0 - \sum_{k=1}^M \beta_k f_k(\alpha_k^T \mathbf{x}_i) \right]^2$$

- 6 An iterative algorithm is utilized: Start by assuming a value for  $\alpha_1$ ,
- 7 and then estimate  $\beta_1 f_1$  using a nonparametric regression technique
- 8 (often a smoothing spline). Then, for this estimate of  $\beta_1 f_1$ , determine
- 9 the  $\alpha_1$  that minimizes the sum of squares. This process is repeated
- 10 until convergence, and then the entire process is repeated for each of
- 11  $k = 2, 3, \dots, M$ .

- 12 Section 11.2 in HTF covers more of the technical details of the model.
- 13 Note that there  $\omega_k$  is used where I use  $\alpha_k$ , and  $g$  is used where I use
- 14  $\beta_k f_k$ .

1 **Projection Pursuit in R**

2 Projection pursuit is implemented in R using the function `ppr()`.

3 To continue with our options example, we could fit **Model Five**:

4 `> model5 = ppr(transresp ~ log(timetotoexpiry)`  
5      `+ log(strike) + log(curprice) + log(histvol),`  
6      `nterms = 4, data=alldat, sm.method="gcv spline")`  
       $M$

7 By setting `nterms = 4`, I am forcing  $M = 4$ .

8 The `sm.method` argument is set to `gcv spline` in order to get the  
9 function to use a smoothing spline to fit each  $f_k$ , with GCV used to  
10 choose the smoothing parameter.

11 Beyond these arguments, `ppr()` behaves much like other R regression functions. See `help(ppr)` for details.

1 Below is the output of

2 > summary(model5)

```

3 Call:
4 ppr(formula = transresp ~ log(timetoexpiry) + log(strike) + log(curprice) +
5      log(histvol), data = alldat, nterms = 4, sm.method = "gcvsppline")
6

```

7 Goodness of fit:

```

8   4 terms
9 33.101042 ← Deviance = RSS

```

```

10
11 Projection direction vectors:
12      term 1      term 2      term 3
13 log(timetoexpiry) 0.006272039644 0.399783868324 0.042947032997
14 log(strike)      -0.669815521832 0.633348669271 -0.691159488371
15 log(curprice)    0.742177094178 -0.522931039444 0.719827473136
16 log(histvol)     -0.021931463883 0.406921920945 -0.047984611187
17
18      term 4
19 log(timetoexpiry) -0.147578246841
20 log(strike)       -0.747048264676
21 log(curprice)    0.630938484390
22 log(histvol)      0.148513232475

```

23 Coefficients of ridge terms:

```

24      term 1      term 2      term 3      term 4
25 0.40943806041 0.18440265555 0.21039638238 0.08941710584
26 ↑β̂₁      ↑β̂₂      - - - -
```

27 Equivalent df for ridge terms:

```

28 term 1 term 2 term 3 term 4
29 10.32 14.00 8.78 14.53
```

$\hat{\beta}_0 = \text{model5}\$yb$   
 $= \text{Overall mean}$

- 1 The components of this output are as follows:
  - 2   • The quantity given as "Goodness of fit" is the residual sum of
  - 3    squares (deviance) for the fit.
  - 4   • The "Projection direction vectors" are the estimates of the vectors
  - 5     $\alpha_k$ . These can also be accessed using model15\$alpha.
  - 6   • The "Coefficients of ridge terms" are the estimates of the  $\beta_k$ . To
  - 7    find the estimate of  $\beta_0$ , look at model15\$yb.
  - 8   • The "Equivalent df for ridge terms" gives the equivalent degrees
  - 9    of freedom for each of the smoothing splines used in the model.
  - 10   To get the total (equivalent) degrees of freedom, count up all of
  - 11   the parameters using
- 12   > sum(model15\$edf) + length(model15\$beta) +  
13                    length(model15\$alpha) + 1  $\beta_0$

14   We see the result to be 68.62. Note that this is slightly more than  
15   Model Four.

- 1 To view the estimates of the ridge functions, simply use
- 2 > plot (model5)
- 3 This is shown as Figure 13 below.

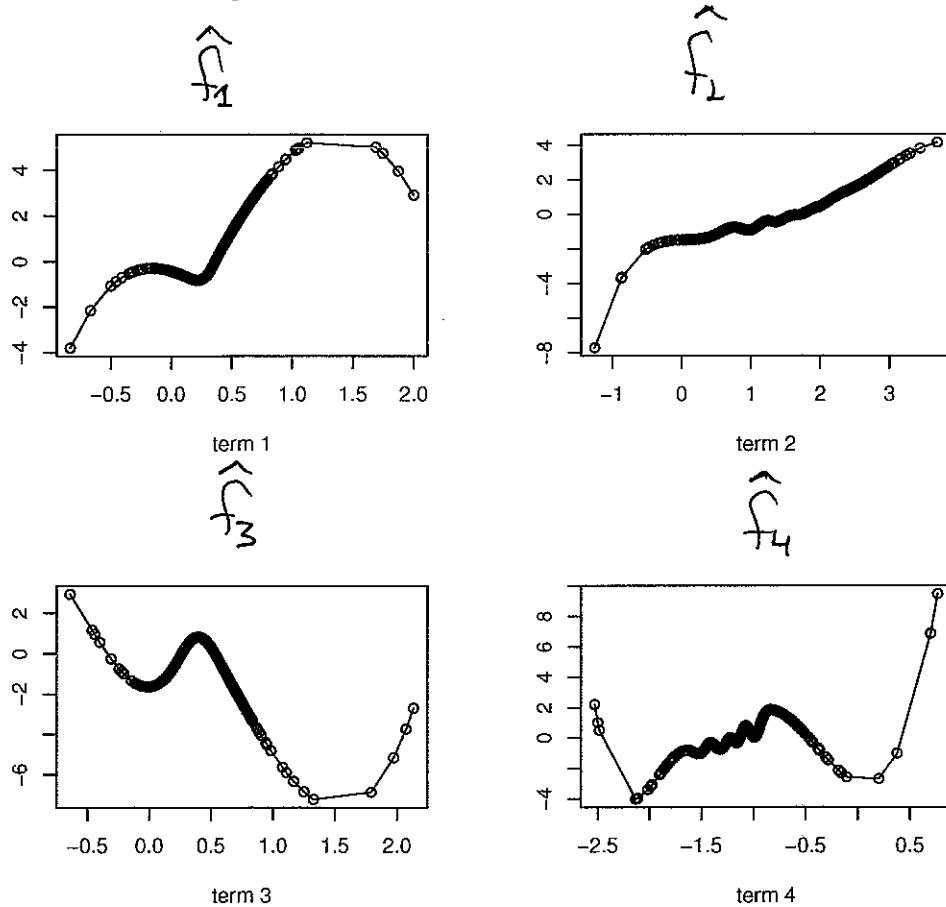


Figure 13: Plot of the estimated ridge functions for Model Five.

Figure 14 shows the plot of residuals versus fitted values for this fit.

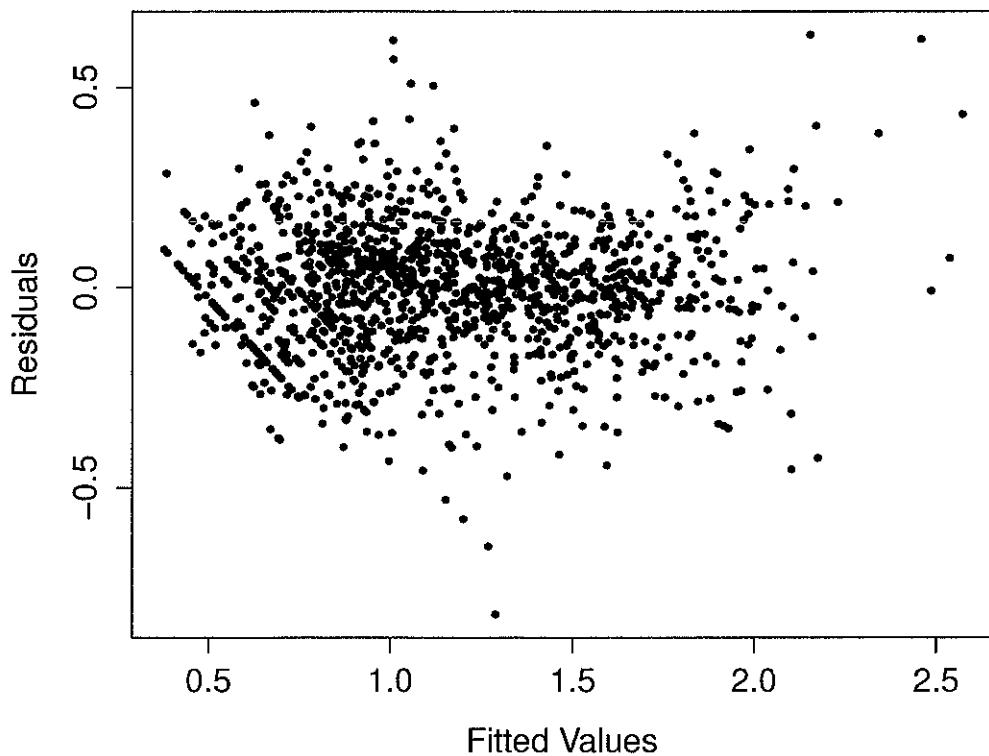


Figure 14: Plot of residuals versus fitted values for Model Five.

Figure 15 shows the response versus the predictions for Model Five.

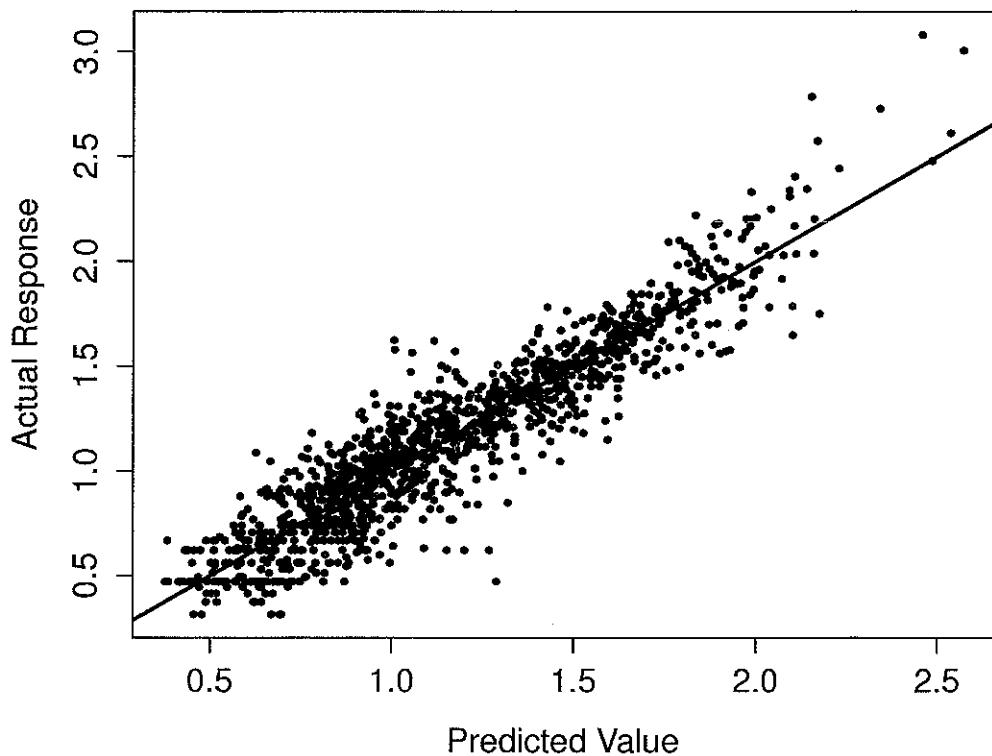


Figure 15: Plot of the response versus the predictions (fitted values) for Model Five. The solid line is the line of perfect agreement between the predictions and the observed responses.

- 1 **Exercise:** Compare Model Five with Model Four in terms of quality of fit.

3 It seems to be a slightly better fit,  
4 without much additional complexity

5 \_\_\_\_\_

6 \_\_\_\_\_

7 \_\_\_\_\_

8 \_\_\_\_\_

9 \_\_\_\_\_

10 \_\_\_\_\_

11 \_\_\_\_\_

12 \_\_\_\_\_

13 \_\_\_\_\_

## 1 Choosing the Number of Ridge Functions ( $M$ )

- 2 We have not considered to this point the issue of choosing  $M$ .
- 3 As is typically the case, we prefer a simpler model, so one would usually start with a small  $M$  and increase it until a suitable fit is found.
- 4
- 5 Cross-validation is also useful. In this case, I employed **k-fold cross-validation**, a procedure where the data are randomly divided into  $k$  groups (the “folds”) and all the model is refit  $k$  times, each time with
- 6 one of the folds excluded from the training sample. The remaining
- 7 fold is used as the test set on which prediction performance is measured.
- 8
- 9
- 10
- 11 **Exercise:** What is the relationship between k-fold cross-validation
- 12 and leave-one-out cross-validation?

~~Leave-one-out cross-validation is k-fold~~  
Leave-one-out cross-validation is k-fold CV with  $k=n$ . This is not practical when model actually has to be refit  $k$  times.

18

- I wrote a function, named `CVforppr()` for performing k-fold cross-validation with projection pursuit regression.

*output of `Apr()` or  
formula for model, i.e.  $y \sim X_1 + X_2$*

```

3  CVforppr = function(model, numfolds, nterms, data=model.frame(model), ...)
4  {
5      fold = sample(rep(1:numfolds, length.out = nrow(data)))
6
7      testerr = numeric(numfolds)
8
9      for(i in 1:numfolds)
10     {
11         holdppr = ppr(model, data=data[which(fold!=i),], nterms=nterms, ...)
12         testpreds = predict(holdppr, newdata=data[which(fold==i),])
13         testresp = data[which(fold==i), which(names(data)==all.vars(model)[1])]
14         testerr[i] = sum((testpreds - testresp)^2)
15     }
16     return(sum(testerr))
17 }
```

*model.frame  
returns data frame  
Used in fit*

- There are some advanced R concepts utilized in this function, worth studying if you are interested.

- The function can be accessed at

<http://www.stat.cmu.edu/~cschafer/MSCF/CVforppr.R>

- 1 Below is the code I used to run the cross-validation. I chose  $k = 10$ , a
- 2 fairly standard choice.
  
- 3 Note that in order to assess the stability of the cross-validation result
- 4 (keep in mind that it is random), I repeated it four times, each time
- 5 with a different seed for the random number generator.

```

6 modelformula = transresp ~ log(timetoexpiry) + log(strike) +
7           log(curprice) + log(histvol)
8
9 pprCV = matrix(0, nrow=10, ncol=4)
10
11 for(j in 1:ncol(pprCV)) {
12   set.seed(j)
13   for(i in 1:nrow(pprCV)) {
14     pprCV[i,j] = CVforppr(modelformula, nterms=i, numfolds=10,
15                           data=alldat, sm.method="gcvsspline")
16   }
17 }
18 }
```

4 iterations

$M = i$

- Figure 16 below shows how the error decreases as  $M$  is increased. We
- also see the variability across the four cross-validation repetitions.

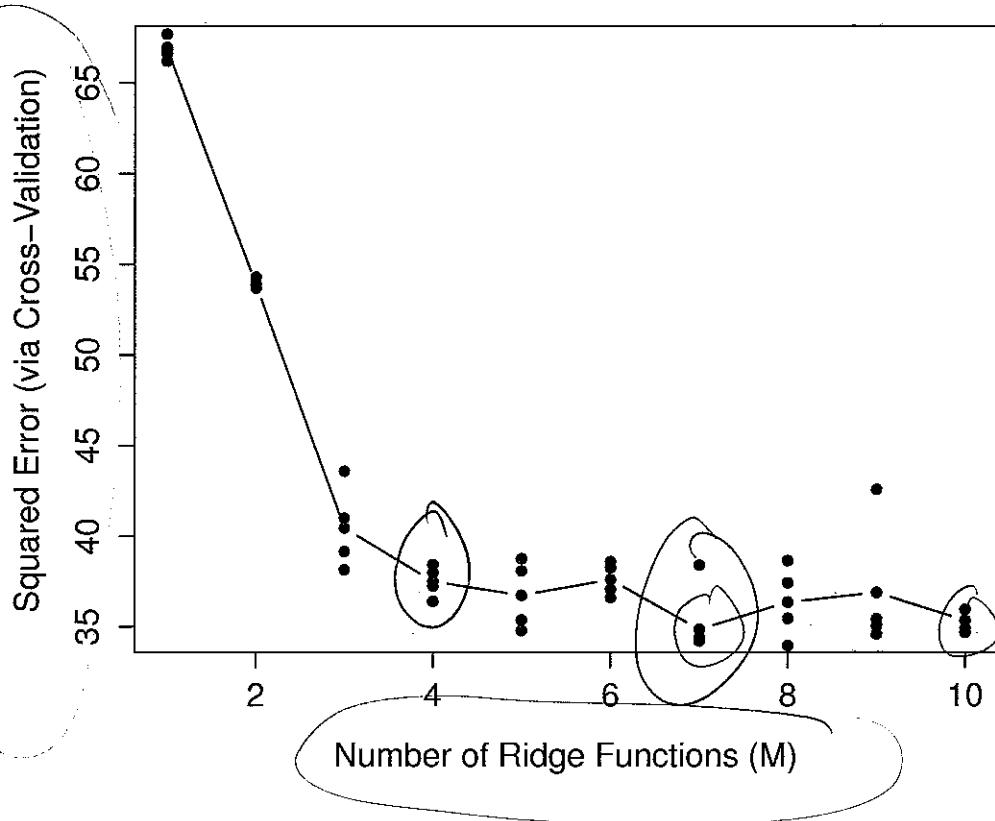


Figure 16: Plot showing the results of the cross-validation procedure. The line connects the average values across the four repetitions.

- The choice of  $M = 4$  follows from the fact that for  $M > 4$  there is
- not clear evidence of improved performance (along with the fact we
- found that  $M = 4$  yields a model which fits well).

1

2 **Neural Networks for Regression**

3 *ANN*

4 To quote from HTF (page 392):

5 The term **neural network** has evolved to encompass a large  
6 class of models and learning methods. Here we describe the  
7 most widely used “vanilla” neural net, sometimes called the  
8 **single hidden layer back-propogation network**, or **single**  
9 **layer perceptron**. There has been a great deal of *hype* sur-  
10 rounding neural networks, making them seem magical and  
11 mysterious. As we make clear in this section, ~~they~~ are just  
12 nonlinear statistical models, much like the projection pursuit  
regression model discussed above.

13 At one time, research into neural networks was very hot in machine  
14 learning, but that has largely died down. As is done in HTF, we will  
15 focus on the basic foundation of the approach.

16 I will further focus on the use of neural nets for predicting a contin-  
17 uous response, and leave the classification discussion for later.

- 1 The single hidden layer back-propogation network regression model  
 2 can be written as follows:

3

$$Y = \beta_0 + \sum_{k=1}^M \beta_k \phi(\alpha_{0k} + \alpha_k^T \mathbf{x}) + \epsilon$$

- 4 where the  $\beta$  and  $\alpha$  are parameters to be estimated, but the function  $\phi$   
 5 is **not** estimated.

- 6 **Exercise:** Compare this neural network model to the projection pur-  
 7 suit regression model:

8

$$Y = \beta_0 + \sum_{k=1}^M \beta_k f_k(\alpha_k^T \mathbf{x}) + \epsilon$$

9

10 With ANN, not estimating function of  
 11 the projection nonparametrically

12

13

14

15

16

17

18

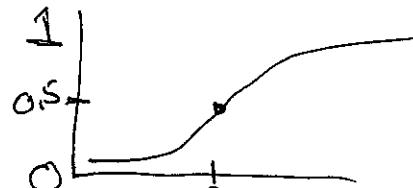
19

## 1 Terminology

2 Some terms commonly used in conjunction with neural networks:

- 3 • The function  $\phi$  is called the **activation function**. The standard  
4 choice is the **sigmoid function**

5 
$$\phi(u) = \frac{1}{1 + \exp(-u)}.$$



- 6 • The elements  $\phi(\alpha_{0k} + \alpha_k^T x)$  for  $k = 1, 2, \dots, M$  comprise the **hid-**  
7 **den layer**.
- 8 • The intercept terms  $\alpha_{0k}$  are called the **biases**.
- 9 • The entire collection of  $\alpha$  and  $\beta$  parameters are called the **weights**.

## 1 Fitting the Model

- 2 In the regression setting, the standard is to use least squares to esti-
- 3 mate the weights (parameters).
- 4 If least squares is used alone, however, the solution is unstable, and
- 5 the nonconvex optimization problem that is solved is sensitive to the
- 6 starting values used in the iterative search algorithm.
- 7 Hence, a **regularization penalty** is often added onto the residual sum
- 8 of squares. A standard choice is the same penalty used in ridge re-
- 9 gression, i.e., minimize

$$10 \quad \text{RSS} + \lambda \sum_{k=1}^M \left[ \beta_k^2 + \sum_{i=0}^p \alpha_{ik}^2 \right].$$

- 11 The parameter  $\lambda$  is commonly called the **decay parameter**.
- 12 This regularization becomes particularly important as  $M$  is increased.
- 13 Because of the regularization, it is important to transform the predic-
- 14 tors so that they are on comparable scales; often one forces each to
- 15 have mean zero and variance one. This can be accomplished easily
- 16 in R using the `scale()` function.

- <sup>1</sup> Of course, now we have two parameters that control the complex-
- <sup>2</sup> ity of the model:  $M$  and  $\lambda$ . These should both be chosen carefully
- <sup>3</sup> in order to avoid overfitting. Like other nonparametric estimation
- <sup>4</sup> methods, neural networks have tremendous capacity to overfit to
- <sup>5</sup> the observed data.

## 1 Neural Networks in R

- 2 The function `nnet()` that is part of the package MASS implements  
3 this version of neural networks.

- 4 Consider the syntax below:

```
5 > model6 = nnet(transresp ~  
6   scale(log(timetoxpiry)) +  
7   scale(log(strike)) + scale(log(curprice)) +  
8   scale(log(histvol)), data=alldat, size=10,  
9   linout=TRUE, decay=0.001, maxit=1000)
```

9  
fitting contr.  
response

2

M

1 Note the following:

2 • All of the predictors have been scaled to have mean zero and  
3 variance one.

4 • The argument `size=10` specifies the number of components in  
5 the hidden layer, i.e., the value of  $M$ .

6 • The argument `linout=TRUE` tells the function that a continuous  
7 response is being fit.

8 • The decay parameter  $\lambda$  is set to 0.001.

9 • By setting `maxit = 1000`, we increase the number of allowed  
10 iterations of the search algorithm above the default of 100.

11 • The residual sum of squares (deviance) can be found to be 24.73  
12 in this case using

13 > `sum(model6$residuals^2)`

14 • The values for all of the weights (parameters) can be found by  
15 looking at

16 > `summary(model6)`

17 • The weights are also stored in `model6$wts` and hence we can  
18 find the degrees of freedom for the fit to be 61 by looking at

19 > `length(model6$wts)`

## 1 Choosing the Tuning Parameters

- 2 We will use k-fold cross-validation to choose the tuning parameters
- 3  $M$  and  $\lambda$  (the decay parameter).
- 4 Since we have to loop over two parameters and still want to assess the stability in the procedure, the time required increases significantly. In the code below, only five possible values for each of  $M$  and  $\lambda$  are considered. This took approximately a half hour to run on my laptop.

```

9  modelformula = transresp ~ scale(log(timetoxpiry)) + scale(log(strike)) +
10    scale(log(curprice)) + scale(log(histvol))
11
12 nnetCV = array(0, dim=c(5, 5, 4))
13 decaylist = c(0, 0.0001, 0.001, 0.01, 0.02)
14 Mlist = c(5, 10, 15, 25, 50)
15
16 for(k in 1:dim(nnetCV)[[3]]) ← loop over different seeds
17 {
18   set.seed(k)
19   for(i in 1:length(decaylist)) ← choice of λ
20   {
21     for(j in 1:length(Mlist)) ← choice of M
22     {
23       nnetCV[i, j, k] = CVfornnet(modelformula, size=Mlist[j],
24                                     decay=decaylist[i], numfolds=10, data=alldat)
25     }
26   }
27 }
```

- 1 **Exercise:** Look at the R output below. Can you see how I chose  $M =$   
 2 10 and  $\lambda = 0.001$ ?

3 > apply(nnetCV, c(1,2), mean) average over 4 seeds

4 [1,] 36.22044 31.18069 35.69841 55.89856 [5]  $\lambda = 0$

5 [2,] 33.36599 31.33295 33.25801 35.24420 66.86226  $\lambda = 0.001$

6 [3,] 33.60559 30.60872 32.92143 37.91271 63.38938  $\lambda = 0.001$

7 [4,] 33.41259 32.00541 31.20883 32.75816 33.53889  $\lambda = 0.01$

8 [5,] 33.45401 32.56375 32.26354 31.19979 31.92250  $\lambda = 0.02$

9 M=5 10 15 25 50

10 > apply(nnetCV, c(1,2), sd) significant instability

11 [1,] 2.6901061 1.3356487 3.9312762 31.0615974 [5]  $\lambda = 0$

12 [2,] 1.6763893 2.6997057 1.6193681 1.9805685 8.2166714

13 [3,] 1.3929055 0.7022319 0.9256333 1.3750534 7.9213219

14 [4,] 0.9455172 1.5102847 0.7075583 1.0047759 1.3388263

15 [5,] 0.8700113 0.3257760 0.8382898 0.6748657 0.6410868

16 sd over 4 seeds

17 This choice minimized the MSE, it is  
 18 a relatively simple model, and the  
 19 MSE estimate is quite stable

20

21

22

23

24

25

26

- Figure 17 shows the plot of residuals versus fitted values for this fit.

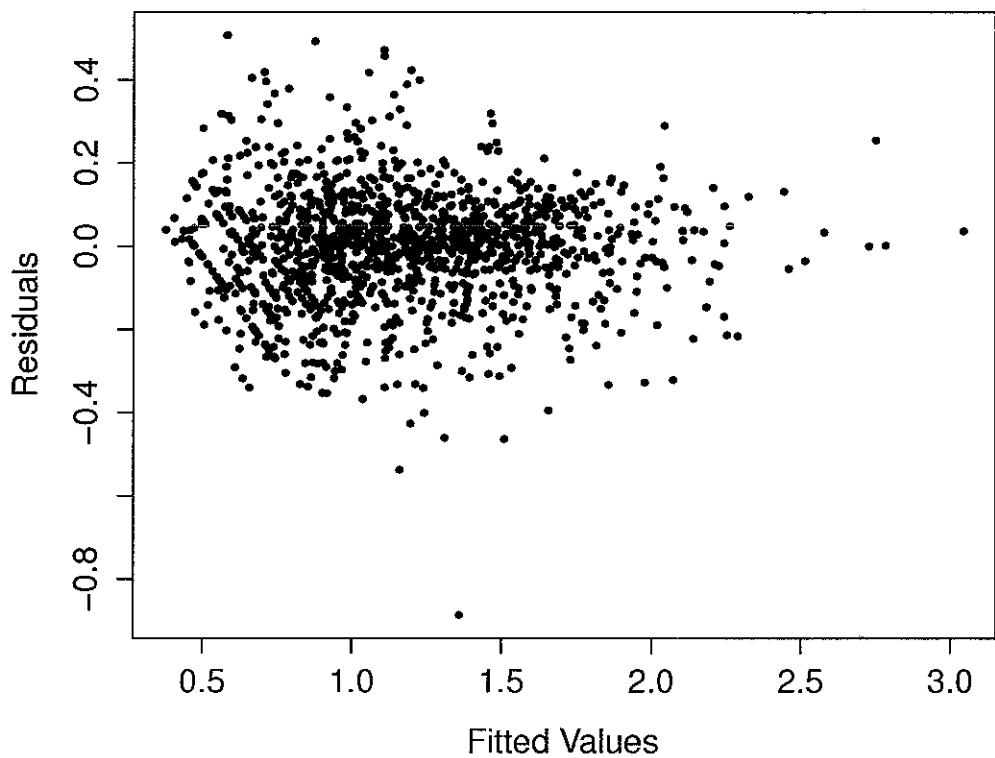


Figure 17: Plot of residuals versus fitted values for Model Six.

Figure 18 shows the response versus the predictions for Model Six.

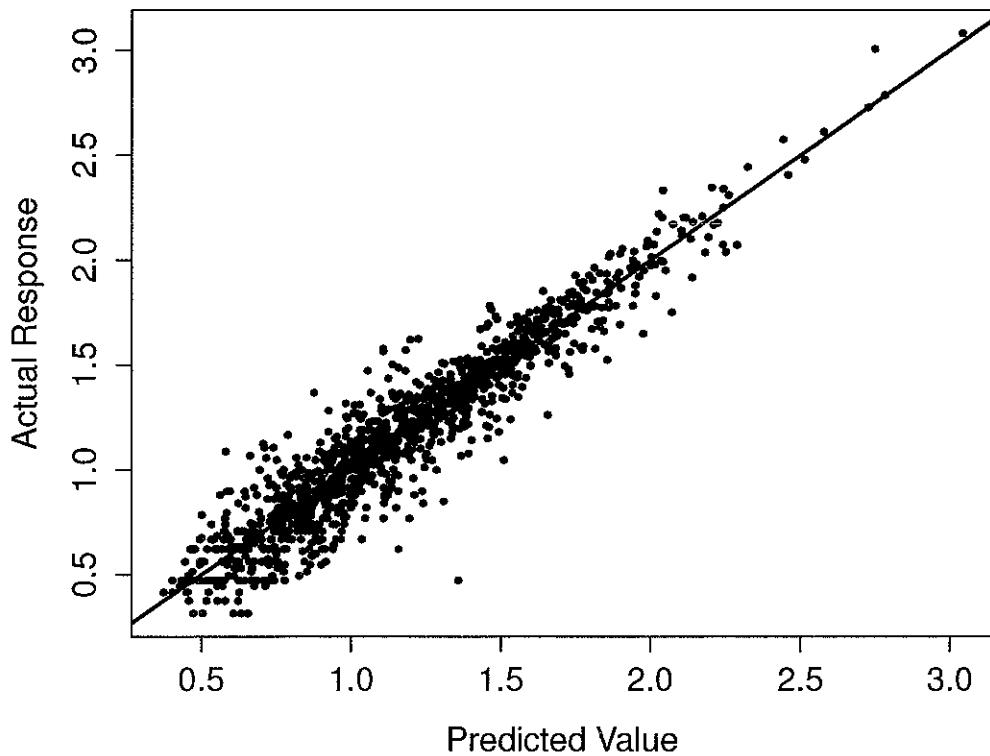


Figure 18: Plot of the response versus the predictions (fitted values) for Model Six. The solid line is the line of perfect agreement between the predictions and the observed responses.

## 1 Comparing the Models

- 2 The table below compares the five models that used the transformed  
 3 response based on their deviance and their degrees of freedom.

Model	Deviance	d.f.	$\hat{\sigma}^2$	
Two	105.12	5	0.0788	
Three	69.31	25.65	0.0528	
4 -4572 Four	40.25	60.10	0.0315	GAM
5 -4817 Five	33.10	68.62	0.0247	PIR
6 -5222.834 Six	24.73	61	0.0185	NNET

- 5 **Exercise:** Model Six has fewer degrees of freedom than does Model  
 6 Five. Why do you think Model Six was able to achieve a better fit?

7 The additional ~~fit~~ projections (going from  
 8  $M=4$  to  $M=10$ ) were more valuable  
 9 than the additional flexibility from  
 10 using  $f_k$  instead of  $\phi$

11 Note: Recall from start of Lecture 3

$$13 AIC = n \log \left( \frac{RSS}{n} \right) + 2 \# \text{ of params} + k$$

## Part 8: Tree Models

- 2 Text references: §8.1 – 8.2.2 in ISL
  - 3 **Tree-based models**, which include both **regression trees** and **classification trees**, create a simple-to-interpret tree structure that leads to a prediction as a function of covariates. **CART**
  - 6 Consider Figure 1. This utilizes the same data set on call options as was used in the previous Part of the notes. At each “split,” the criterion tells you whether you should take the left or right branch based on whether or not the stated condition is true or false, respectively.
  - 10 Note that in this example, the response is the price to the power 0.25.
  - 11 The predictors are untransformed.

- 1 **Exercise:** Using this tree model, what would be the predicted price
  - 2 for a call option with strike price of \$40 and current asset price of
  - 3 \$40?

Predict (price<sup>'14</sup>) is 14730

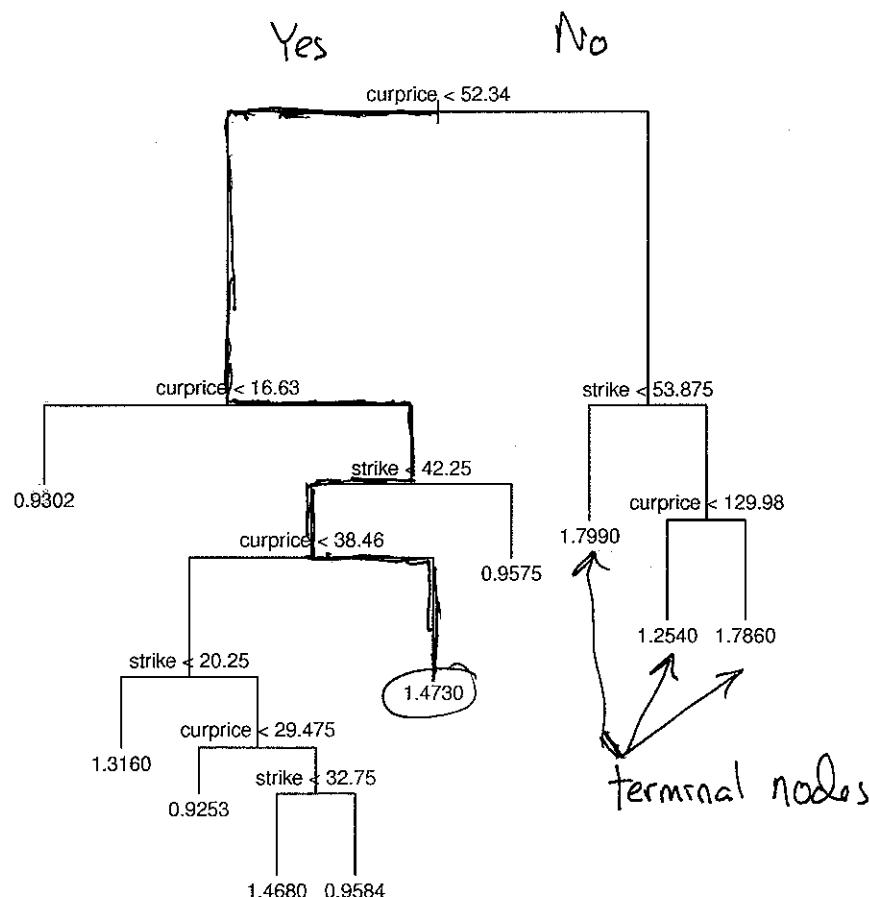


Figure 1: A simple example of a tree for the call options example. In this example, there are only two predictors: current asset price and strike price.

<sup>1</sup> **Regression Trees**

- <sup>3</sup> Here we will consider the construction of these trees in the case where  
<sup>4</sup> the response is a continuous-valued quantity. These are called **re-  
5 gression trees**.
- <sup>6</sup> As with previous regression models, any regression tree makes a pre-  
<sup>7</sup> diction for every predictor vector  $\mathbf{x}_i$ , as before these are the fitted  
<sup>8</sup> values  $\hat{y}_i$ .
- <sup>9</sup> The deviance for this model is equivalent to the RSS:

<sup>10</sup> 
$$\text{deviance} = \text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- <sup>11</sup> The degrees of freedom associated with a regression tree is simply  
<sup>12</sup> equal to the number of terminal nodes in that tree.

## 1 Fitting the Model

- <sup>2</sup> In an ideal world, we could search over all possible trees (maybe with a limit on the number of degrees of freedom) and find the one with the smallest deviance.
  - <sup>5</sup> Unfortunately, this is not feasible given the number of possible ways of splitting the predictors.
  - <sup>7</sup> Instead, a two-stage process is recommended for fitting tree models.

- 1 In the **first stage**, a **greedy algorithm** is utilized to “grow” the tree
- 2 model. Decisions are made one split at a time: Initially, one asks,
- 3 what split would create a model with the smallest deviance?
  
- 4 Keep in mind that a single split can only involve one predictor. So,
- 5 the space of possible splits is not that difficult to search over.
  
- 6 This process is repeated several times: Whatever the current tree,
- 7 one can ask: What additional split, added to the current tree, would
- 8 create a model with the smallest deviance?

Figure 2 shows this evolution for the simple model described above.

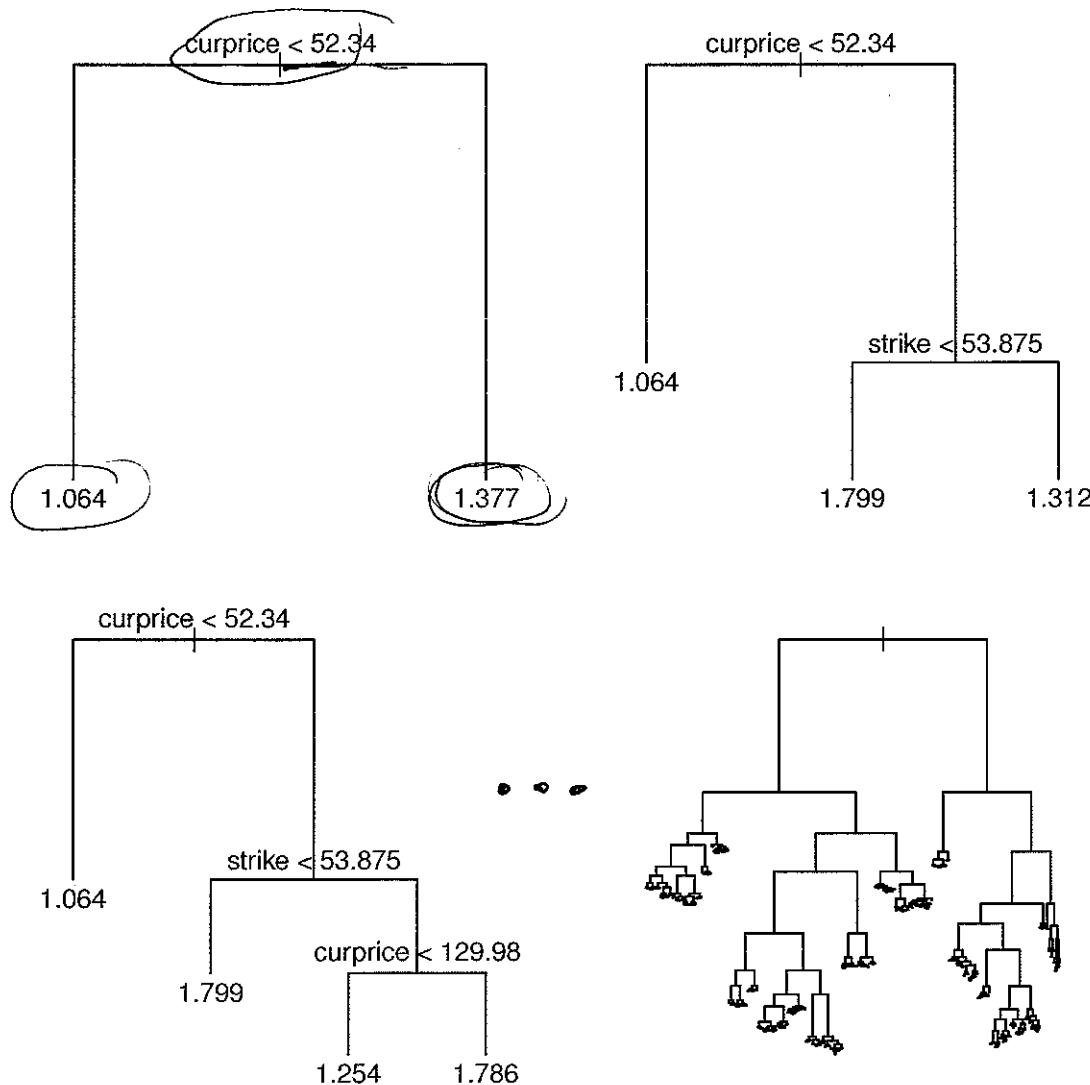


Figure 2: An example of the growing of a tree model. At the last stage, the tree is saturated.

- 2 The tree can be grown out until every unique value of the predictor
- 3 vector has its own terminal node. In Figure 2, this is shown as the
- 4 bottom right tree.

- <sup>1</sup> Clearly, the result of the first stage is a severely overfitted model.
- <sup>2</sup> So, in the **second stage**, the full tree is **pruned** back to an optimal
- <sup>3</sup> choice for the model.
- <sup>4</sup> In this process, one imagines searching over all subtrees of the full
- <sup>5</sup> tree that was constructed in order to find that tree which minimizes
- <sup>6</sup> the **cost complexity value** for the tree, defined as

<sup>7</sup> 
$$\underline{\text{deviance}} + \underline{\alpha \times \text{number of terminal nodes in tree}}$$

- <sup>8</sup> Here,  $\alpha$  is a smoothing parameter.
- <sup>9</sup> Note how growing the full tree first has greatly reduced the number
- <sup>10</sup> of possible trees that we have to search over.

- <sup>11</sup> **Exercise:** Describe the effect of varying  $\alpha$ .

<sup>12</sup> If  $\alpha$  large, then significant penalty for having  
a large # of terminal nodes. Hence,  
tree will be simpler. (fewer nodes)

<sup>15</sup> If  $\alpha$  small, have more terminal nodes.

<sup>17</sup> \_\_\_\_\_

<sup>18</sup> \_\_\_\_\_

1 The smoothing parameter  $\alpha$  is typically chosen via  $K$ -fold cross vali-  
 2 dation. A recommended choice is  $K = 10$ . It is instructive to consider  
 3 the steps of this procedure as applied to this case:

- 4 1. Fix a particular candidate value for  $\alpha$ .  
 5 2. Divide the data set into  $K$  folds.  
 6 3. For each of  $i = 1, 2, \dots, K$ , do the following:  
 7 (a) Using all of the data except those observations in fold  $i$ , re-  
 8 grow a full tree.  
 9 (b) Prune the tree using the candidate  $\alpha$ .  
 10 (c) Using this pruned tree, predict the response for each of ob-  
 11 servations in fold  $i$ .  $\leftarrow$  left-out fold  
 12 (d) Calculate the deviance for these predictions, i.e. find

$$13 \text{deviance}_i = \sum_{j \in \text{fold } i} (y_j - \hat{y}_j)^2$$

- 14 4. Calculate

$$15 \text{deviance}_{\text{CV}}(\alpha) = \sum_{i=1}^K \text{deviance}_i$$

- 16 The above process is repeated for many candidate  $\alpha$  values, and we  
 17 choose the  $\alpha$  that minimizes  $\text{deviance}_{\text{CV}}(\alpha)$ . Once we have our cross-  
 18 validated choice for  $\alpha$ , then we can go back to our original, full tree  
 19 and prune using that  $\alpha$ .

## 1 Regression Trees in R

rpart

- 2 The R package `tree` has built-in functions for growing and pruning trees. We will work through the steps here.
- 4 For now, we will just use two predictors, strike price and current asset price.
- 6 First, to grow the full tree, one would use
  - 7 `> fulltree = tree(transresp ~ strike + curprice,`
  - 8 `data=alldat, mindev=0, minsize=2)`
- 9 The use of `mindev=0` and `minsize=2` are important. These arguments control the **stopping rule** to determine when the growing of the tree should cease.
- 12 – If I set `mindev=0.01` (the default), then an existing node will not be split if its within-node deviance is less than 1% of the overall variance of the response.

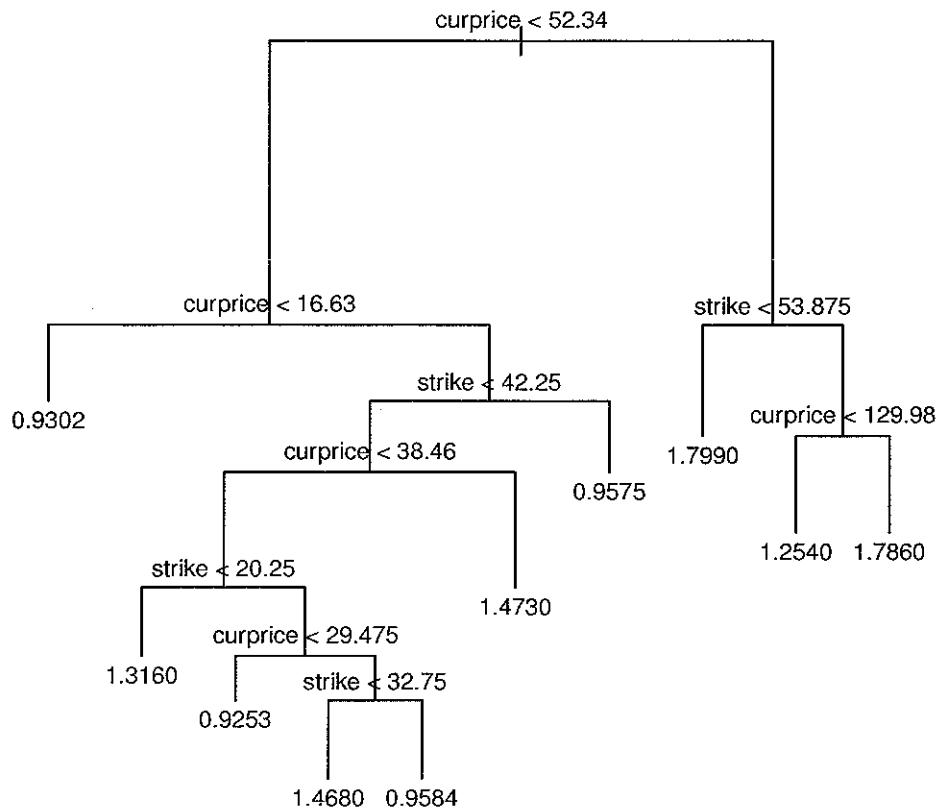


- 15 – If I set `minsize = 10` (the default), then nodes which have fewer than 10 observations will not be candidates for further splitting.

- 1 If I want to prune a tree to a particular choice for the cost complexity
- 2 parameter  $\alpha$ , (say,  $\alpha = 5$ ), then I would use
- 3 > `prunedtree = prune.tree(fulltree, k=5)`  

- 4 (Note that R uses  $k$  to denote this. I prefer to stick with the HTF
- 5 notation of  $\alpha$ .)
- 6 One can also use
- 7 > `prunedtree2 = prune.tree(fulltree, best=10)`
- 8 This will return the optimal pruning of the full tree which has ten
- 9 terminal nodes.

- 1 The output of `prune.tree()` is simply another object of the same
- 2 type as the output of `tree()`.
  
- 3 To look at the tree, one simply uses
- 4 `> plot(prunedtree)`
- 5 `> text(prunedtree, cex=0.75)`  $\swarrow$  font size
  
- 6 Without the call to `text()`, you will only see the structure of the
- 7 tree, not the labels on the splits. The argument `cex` controls the font
- 8 size. The default of `cex=1` is often too large.

Figure 3: The full tree pruned to  $\alpha = 5$ .

- <sup>1</sup> Of course, we want to choose  $\alpha$  using cross validation.
- <sup>2</sup> Unfortunately, although the package `tree` has a built-in function `cv.tree()`,
- <sup>3</sup> this is not implemented appropriately for our purpose. In particular,
- <sup>4</sup> it does not allow for deviation from the default settings of `mindev` and
- <sup>5</sup> `minsize` when growing the full tree.

- 1 So, I made a couple simple edits to the function and posted it at
- 2 <http://www.stat.cmu.edu/~cschafer/MSCF/cv.tree.full.txt>
- 3 The name of the function is `cv.tree.full()`. The call is simply
- 4 `> cvout = cv.tree.full(fulltree)`

default values of `mindiv` and `minsize` are  
0 and 2, respectively

- 1 To create Figure 4, the call is
- 2 > plot (cvout)

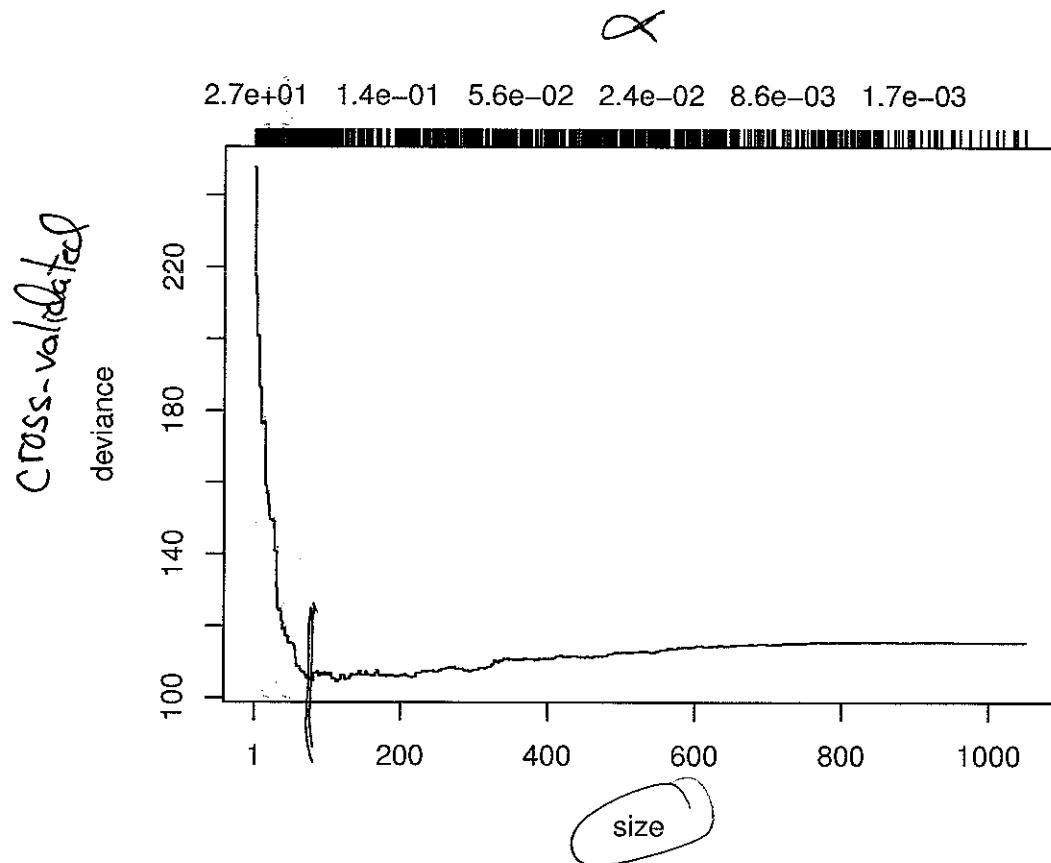


Figure 4: The output of `plot (cvout)`.

- 3 The numbers across the top of the plot are the different values for  $\alpha$ ;
- 4 the numbers on the lower axis are the number of nodes in the pruned
- 5 tree corresponding that choice of  $\alpha$ .

1 Consider these commands:

```

2 > optalpha = cvout$k[which.min(cvout$dev) ]
3 0.2876668
4 > cvout$size[which.min(cvout$dev) ]
5 110
6 > opttree = prune.tree(fulltree, k=optalpha)

```

7 We see that the optimal choice of  $\alpha$  is 0.29, which corresponds to a  
8 tree with 110 terminal nodes. This tree is shown as Figure 5.

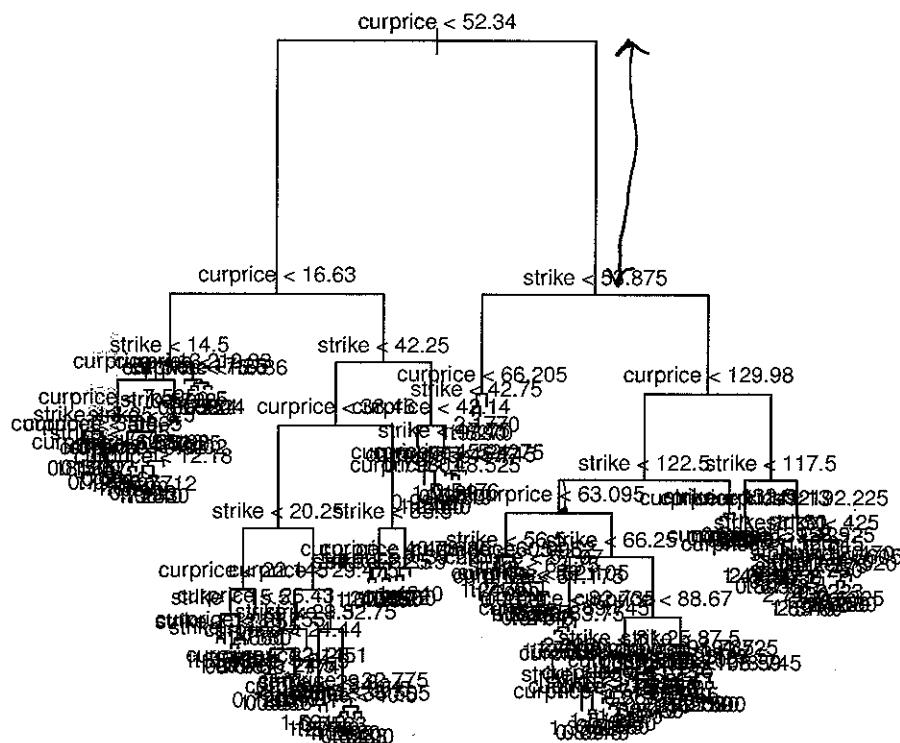


Figure 5: The full tree pruned to  $\alpha = 0.29$ .

## 1 Making Predictions

- 2 Making predictions with the tree is simple: You take your new co-
- 3 variate vector  $\underline{x}^*$  and run through the steps of the decision tree until
- 4 you reach a terminal node.
  
- 5 In R, one can use `predict()` on the output of `tree()` to get the
- 6 fitted values, or make predictions for new observations. The syntax
- 7 is much like `predict.lm()`.

## 1 Return to Example

- 2 Let's now fit a larger model to these data, much like the ones we fit
- 3 with the additive models earlier.
- 4 We will add into our model the predictors "time to expiration" and
- 5 "historical volatility."
- 6 We will call this **Model Seven** to fit in the earlier sequence of models.
- 7 Otherwise, the R code to fit the model and find the optimal  $\alpha$  is much
- 8 like above:

```
9 model7fulltree = tree(transresp ~ timetoexpiry + strike + ]  
10      curprice + histvol, data=alldat, mindev=0, minsize=2) ]  
11  
12 cvmodel7 = cv.tree.full(model7fulltree) ]  
13  
14 model7optalpha = cvmodel7$k [which.min(cvmodel7$dev) ]  
15 model7opttree = prune.tree(model7fulltree, k=model7optalpha)
```

- Exercise: Should I log-transform the predictors in this case?

It does not make a difference. Any monotonic trans.  
 will yield the same result. Stick with  
 original predictors for improved interpretability.

- Figure 6 shows the resulting regression tree. This model has 288 terminal nodes, and a deviance of 12.61. Recall that  $n = 1339$ .

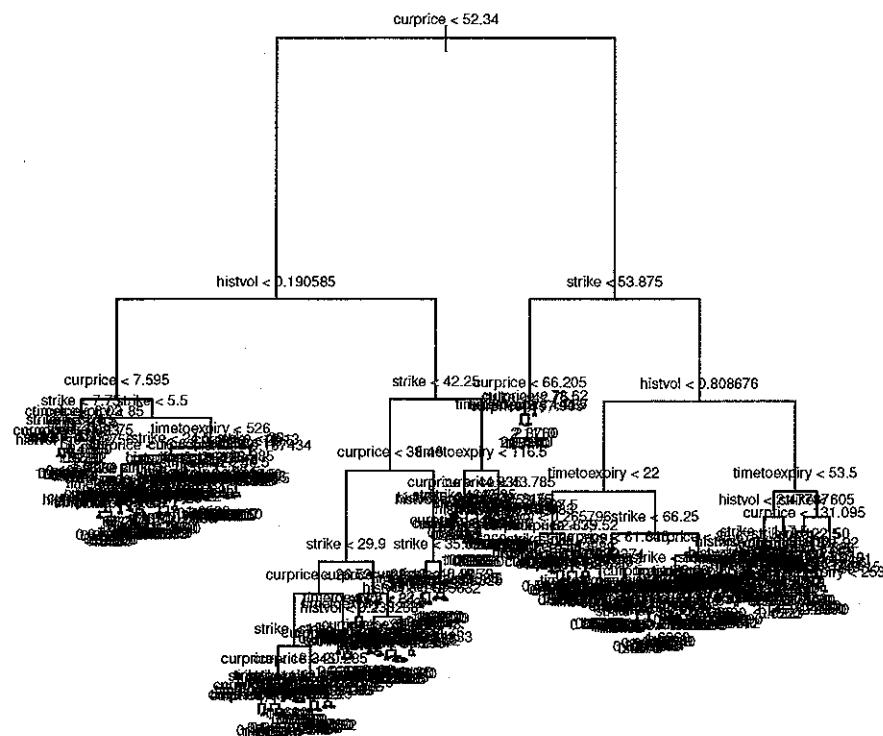


Figure 6: The optimal tree for Model Seven.

Figure 7 is the plot of residuals versus fitted values in this case.

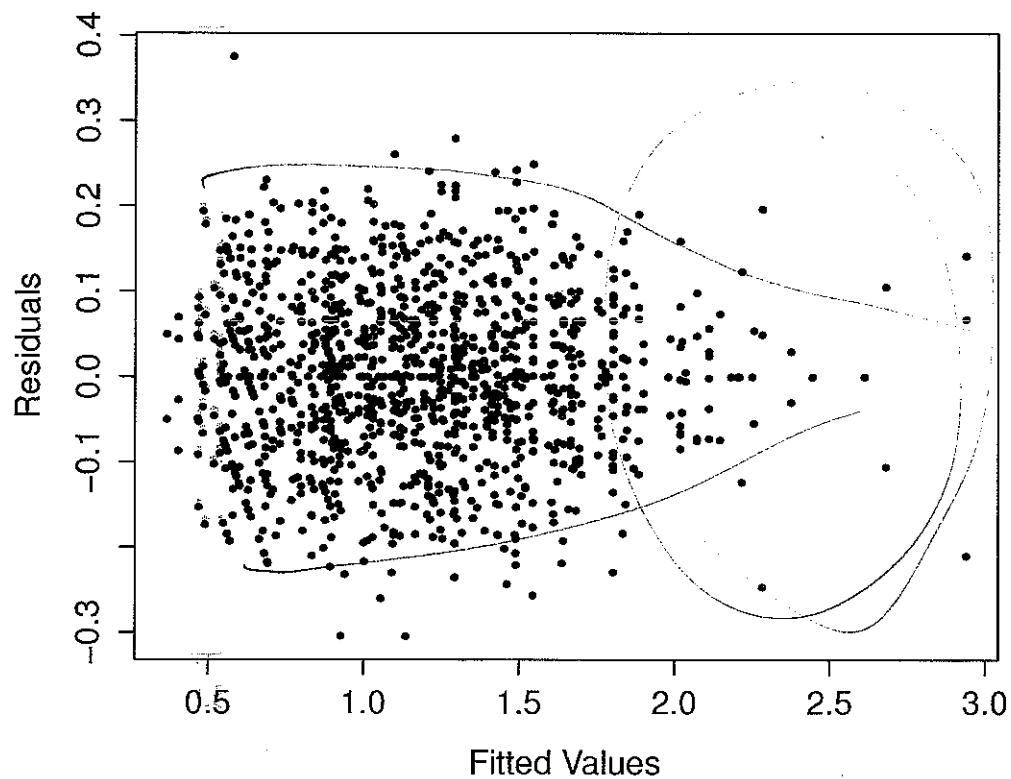


Figure 7: The plot of residuals versus fitted values for Model Seven.

- 1 Figure 8 shows the actual response values versus the predicted val-
- 2 ues.

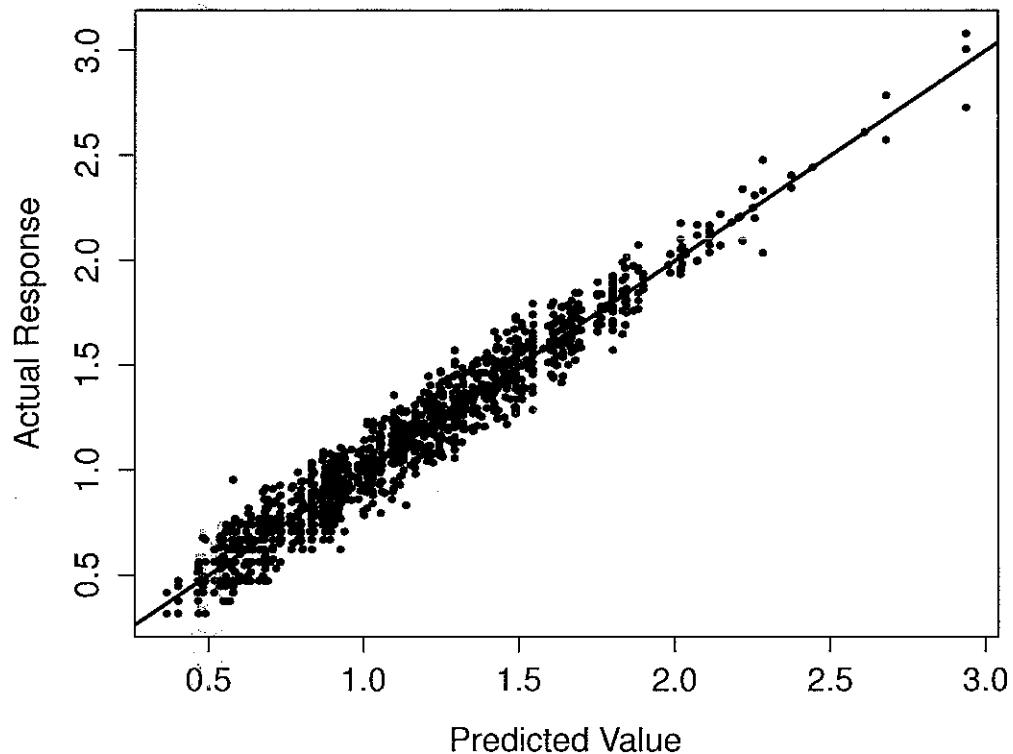


Figure 8: The plot of observed responses versus predicted values for Model Seven. The solid line indicates perfect agreement between the two.

- 3 **Exercise:** Comment on the results of this model fit.

4 Appears to be evidence of overfitting. There

5 are a large number of d.f. (288) relative  
6 to n (1339)

7

8

1 There is some concern of overfitting with these models. A source of  
2 this concern is the variability that is inherent in the cross-validation  
3 procedure. In theory, cross-validation should protect against overfit-  
4 ting, but in practice it can be tricky.

5 Consider Figure 9.

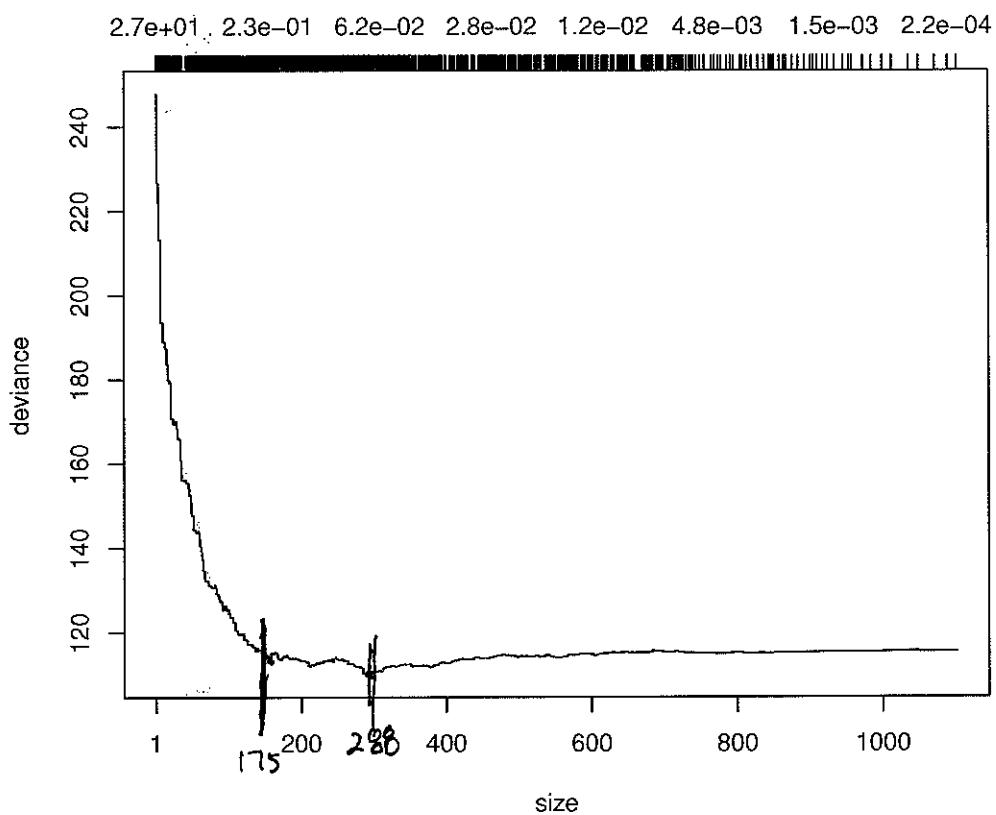


Figure 9: The output of `cv.tree.full()` for the case of Model Seven.

- 1 We note that in this Figure, although the deviance is minimized at a
- 2 size of 288, there is much not difference as compared to a model with
- 3 175 terminal nodes.
  
- 4 Given that we can't expect that much precision from the cross-validation
- 5 procedure, along with the fact that we prefer simpler models, we
- 6 would often switch to using a model of size 175.
  
- 7 I will call this **Model Eight**. In R:
- 8 > model8opttree = prune.tree(model7fulltree, best=175)

- Figure 10 shows the resulting regression tree. The deviance is now 25.99.

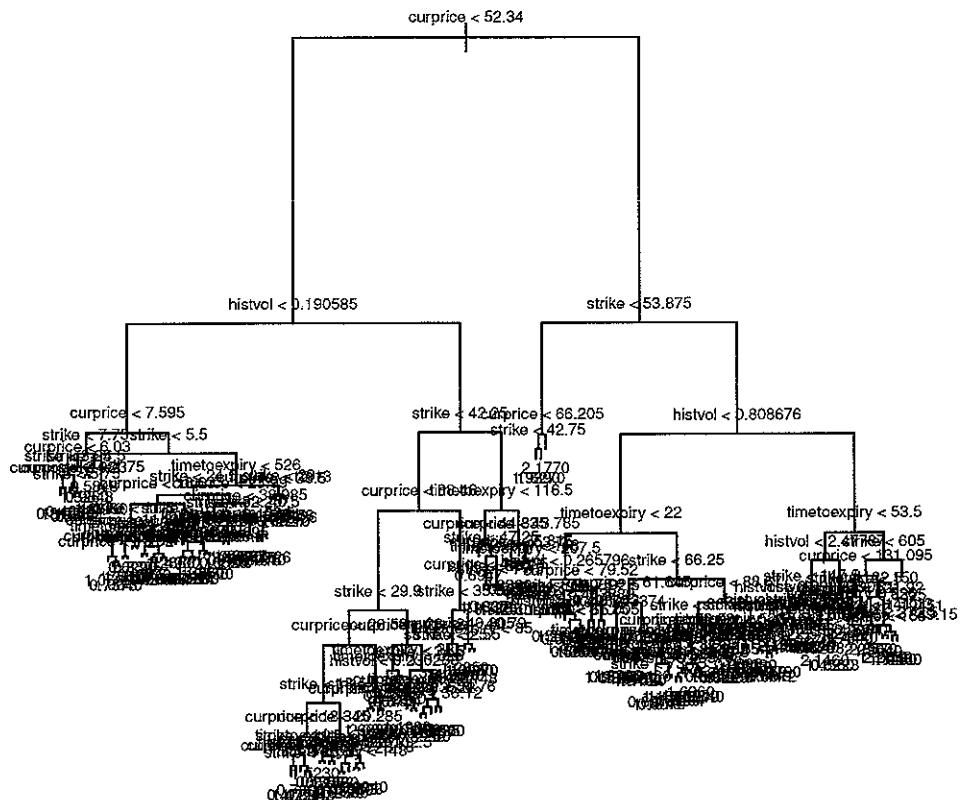


Figure 10: The optimal tree for Model Eight.

Figure 11 is the plot of residuals versus fitted values in this case.

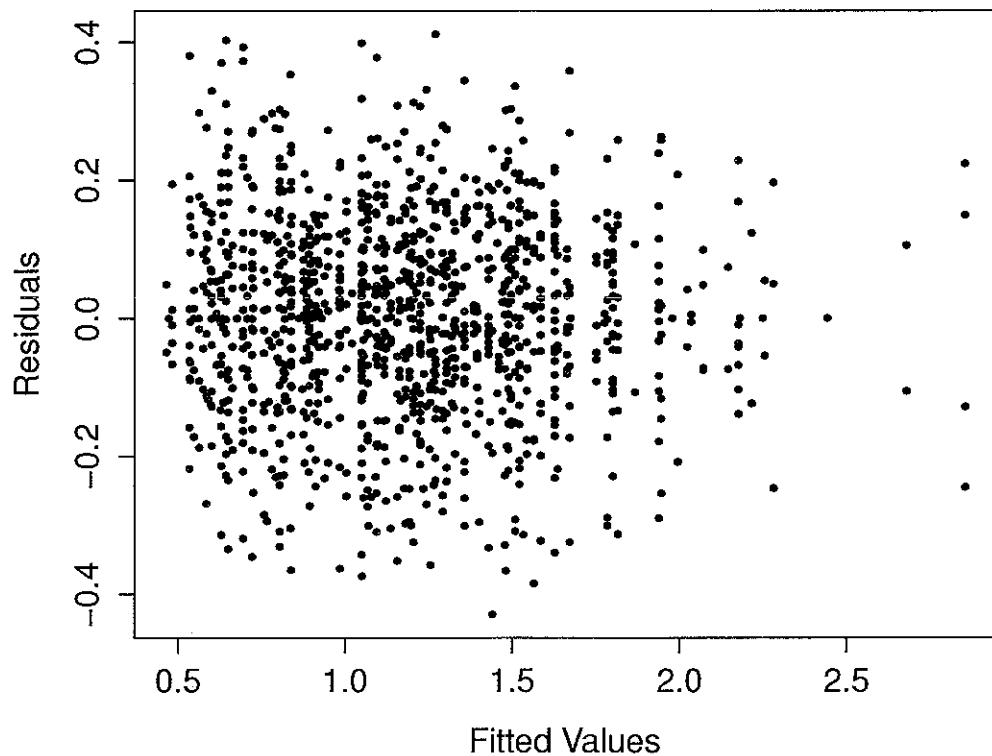


Figure 11: The plot of residuals versus fitted values for Model Eight.

- Figure 12 shows the actual response values versus the predicted values.

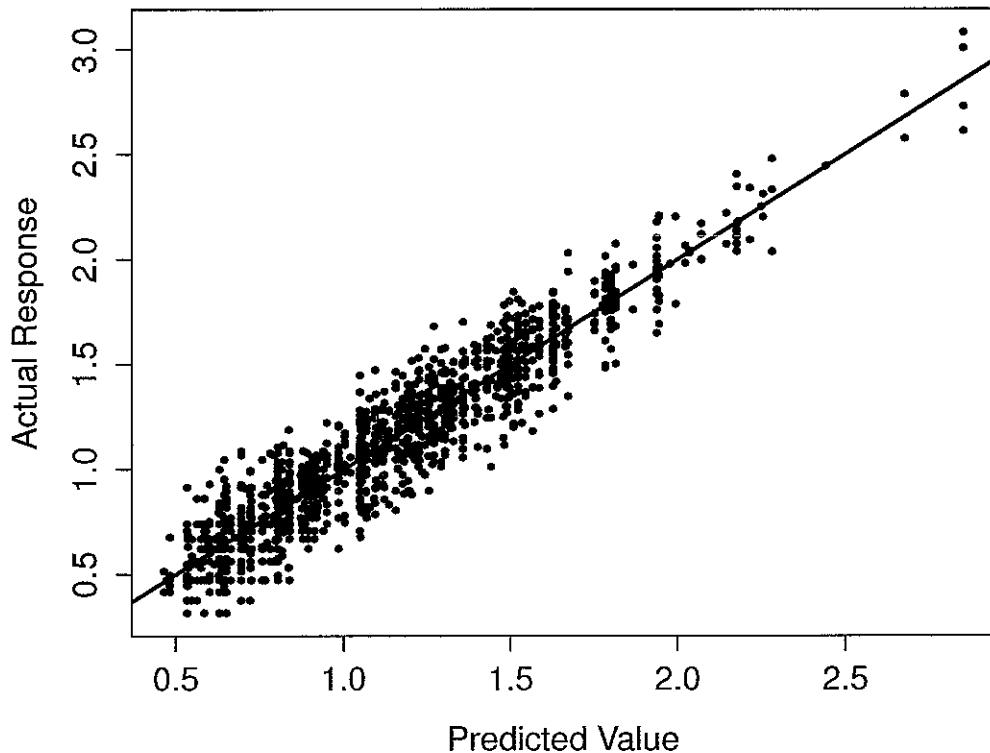


Figure 12: The plot of observed responses versus predicted values for Model Eight. The solid line indicates perfect agreement between the two.

1 I have fit this model, but I would also like to report the generaliza-  
2 tion error for the procedure. This would be the ideal way of compar-  
3 ing the performance of this model with models we fit in the previous  
4 Part.

5 **Exercise:** How would I estimate the generalization error for this (or  
6 any other) model?

7 Obtain a "test set", ~~is~~ a data set not used  
8 in the fitting procedure. Use all of  
9 the models to predict for these obs. and  
10 calculate deviance.

11

---

12

<sup>1</sup> 

---

<sup>2</sup> **Classification Trees**

<sup>3</sup> One of the main positive features of tree models is the natural way  
<sup>4</sup> in which they can handle a categorical response.

<sup>5</sup> A model for such a case is called a **classification tree**.

<sup>6</sup> In most ways fitting these models is identical to fitting a regression  
<sup>7</sup> tree. There are important differences, however.

<sup>8</sup> The first difference is practical: When we call `tree()` we must spec-  
<sup>9</sup> ify that the response is a factor by wrapping the response with the  
<sup>10</sup> `factor()` function.

<sup>11</sup> `> tree(factor(resp) ~ x1 + x2, ...)`

<sup>12</sup> This is how R knows you are fitting a classification tree instead of a  
<sup>13</sup> regression tree.

## 1 The Cross-Entropy

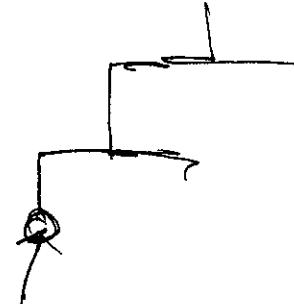
2 The second difference is that the definition of the deviance can no  
 3 longer be the residual sum of squares. Instead, we define the de-  
 4 viance to be the cross-entropy.

5 To derive the cross-entropy, first note that, even though the tree is  
 6 displayed as having a single prediction at the end of each terminal  
 7 node, the model actually fits a **distribution** over the  $K$  possible val-  
 8 ues for the response.

9 For example, if  $K = 4$ , when I use `predict()` on the output of  
 10 `tree()` in the case of a classification tree, I see

four levels

	0	1	2	3
11	0			
12	1	0.0000000 0.5714286 0.0000000 0.4285714		
13	2	0.1000000 0.6000000 0.3000000 0.0000000		
14	3	0.2857143 0.3571429 0.3571429 0.0000000		
15	4	0.0000000 0.6000000 0.2000000 0.2000000		
16	5	0.0000000 0.0000000 0.4000000 0.6000000		
17	...			



18 Each row of this matrix sums to one.

19 The labels that are drawn on the tree represent the **most likely** among  
 20 the  $K$  possibilities.

- 1 If  $\hat{p}_{ik}$  is the probability that observation  $i$  is of class  $k$ , then

2 cross-entropy =  $-2 \times \sum_{i=1}^n \left[ \sum_{k=0}^{K-1} \hat{p}_{ik} \log \hat{p}_{ik} \right]$  = deviance

- 3 Note that, by convention,  $0 \times \log(0) = 0$ .

- 4 **Exercise:** What happens to the cross-entropy as for each  $i$ , one of the  
 5  $\hat{p}_{ik}$  converges to one?

6  $-\hat{p}_{ik} \log \hat{p}_{ik} \rightarrow 0 \text{ as } \hat{p}_{ik} \rightarrow 1$

7  $-\hat{p}_{ik} \log \hat{p}_{ik} \rightarrow 0 \text{ as } \hat{p}_{ik} \rightarrow 0$

8 So, crossentropy converges to 0

- 9
- 10 **Exercise:** What happens to the cross-entropy as  $\hat{p}_{ik}$  converges to  $1/K$   
 11 for all  $i$  and  $k$ ?

12 If  $\hat{p}_{ik} = 1/K$  for all  $i, k$ , the model  
 13 has no idea as to the appropriate level  
 14 for each response

15

16 The cross-entropy is maximized in this case

17

- 1 The motivation for using a measure like the cross-entropy is to re-
- 2 spect the continuous nature of the predictions.
  
- 3 First, note that predictions are made based strictly on the observed
- 4 data: For any terminal node, there will be  $m$  observations which
- 5 "land" in that node. The distribution over the classes will determine
- 6 the prediction for that node.
  
- 7 Imagine that  $K = 4$ , and consider two cases for a particular node
- 8 with  $m = 10$ :

9 **CASE I:**  $\hat{p}_{i0} = 0.4, \hat{p}_{i1} = 0.3, \hat{p}_{i2} = 0.2, \hat{p}_{i3} = 0.1$

10 and

11 **CASE II:**  $\hat{p}_{i0} = 0.9, \hat{p}_{i1} = 0.1, \hat{p}_{i2} = 0, \hat{p}_{i3} = 0$

- 12 In both cases, the "most likely" class is "0," but in CASE II the model
- 13 is much more confident in that prediction.

- 14 CASE I contributes 2.56 to the deviance, while CASE II contributes
- 15 0.65.

$$-\sum_{k=0}^{K-1} \hat{p}_{ik} \log(\hat{p}_{ik})$$

## **1 Example: Predicting Credit Ratings**

- 2 This example was the basis of the summer internship for a recent**
- 3 MScF graduate at a major investment bank. The goal was to pre-**
- 4 dict counterparty ratings of firms using a set of quantitative factors,**
- 5 including profit, capital, a measure of liquidity, and so forth.**
  
- 6 The response variable is the rating, which takes one of three values:**
- 7 “Low Risk,” “Moderate Risk,” and “High Risk.” This is a clear op-**
- 8 portunity to try a classification tree.**

1 These data are available on my web site:

```
2 > ratedata =  
3   read.table("http://www.stat.cmu.edu/~cschafer/MSCF/RateData.txt")
```

4 Load the needed functions

```
5 > library(tree)  
6 > source("http://www.stat.cmu.edu/~cschafer/MSCF/cv.tree.full.txt")
```

7 Now, create the full tree. Note how the categorical predictors are  
8 correctly wrapped with the `factor()` function.

```
9 > fulltree = tree(factor(Rating) ~ factor(ext.source) +  
10   factor(EDF.source) + Profit + Capital +  
11   Excess.Net.Capital + Liab.Capital + Leverage +  
12   Liquidity.1 + Liquidity + factor(Descr),  
13   data=ratedata, mindev=0, minsize=2)
```

14 Next, run cross-validation and prune the tree:

```
15 > cvout = cv.tree.full(fulltree)  
16 > alphaopt = cvout$k[which.min(cvout$dev)]  
17 > opttree = prune.tree(fulltree, k=alphaopt)
```

- The resulting tree is shown as Figure 13.

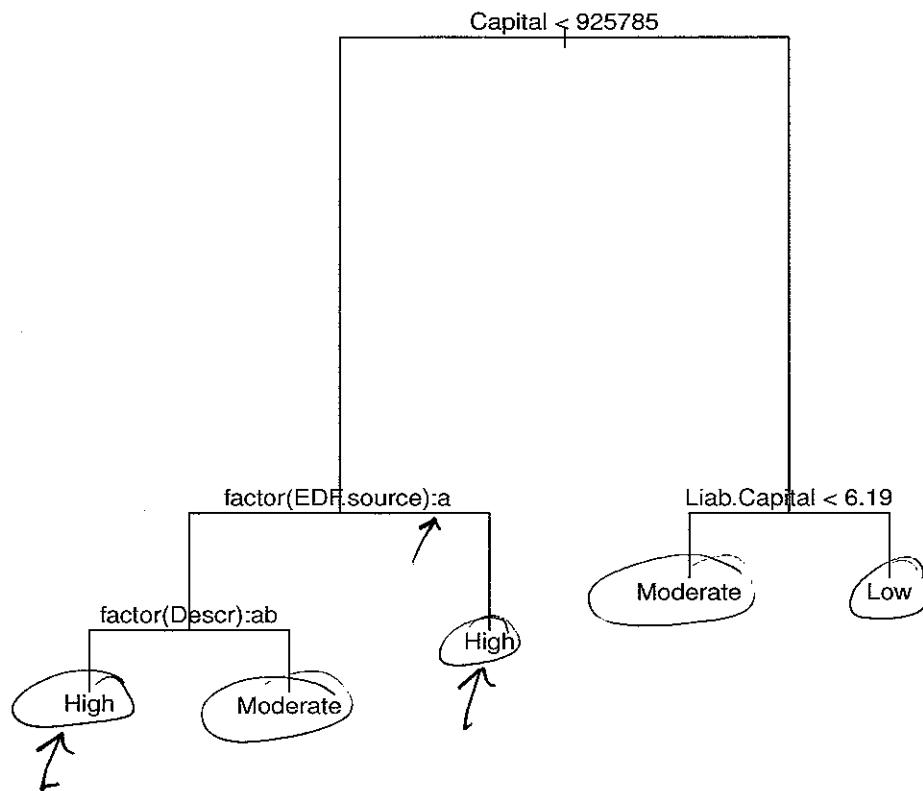


Figure 13: The optimal tree for the credit rating example.

- While it is the case that a more complicated model could have reduced the error rate, the cross-validation procedure judged that such a model would not extend outside of the training sample sufficiently well in order to justify the additional complexity.

- 1 The predict() function has argument type that changes the na-
  - 2 ture of the prediction.
  
  - 3 The default is type = "vector" which gives the fitted probabili-
  - 4 ties of being in each of the classes.
  
  - 5 For example, the first few rows of predict(opttree, ratedata)
  - 6 are shown below

- 1 One can also see the predicted class by using `type="class"`. This
- 2 is simply the class with the highest probability.

```
3 > predict(opttree, ratedata, type="class")  
4 [1] High      High      High      High      Moderate ...
```

- 5 A useful application of the `table()` function is below.

```
6 > table(ratedata$Rating,  
7 predict(opttree, ratedata, type="class"))
```

- 8 The resulting two-way table shows the true class as row, and the
- 9 predicted class as column.

Model

	High	Low	Moderate
High	86	0	4
Low	7	65	4
Moderate	4	19	31

- 14 **Exercise:** The first row of this table shows a positive feature of this
- 15 model fit. What is it?

16 Firms that are truly high risk are not being  
17 classified as low risk

18 \_\_\_\_\_

19 \_\_\_\_\_

<sup>1</sup> 

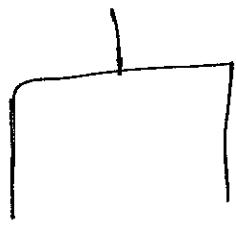
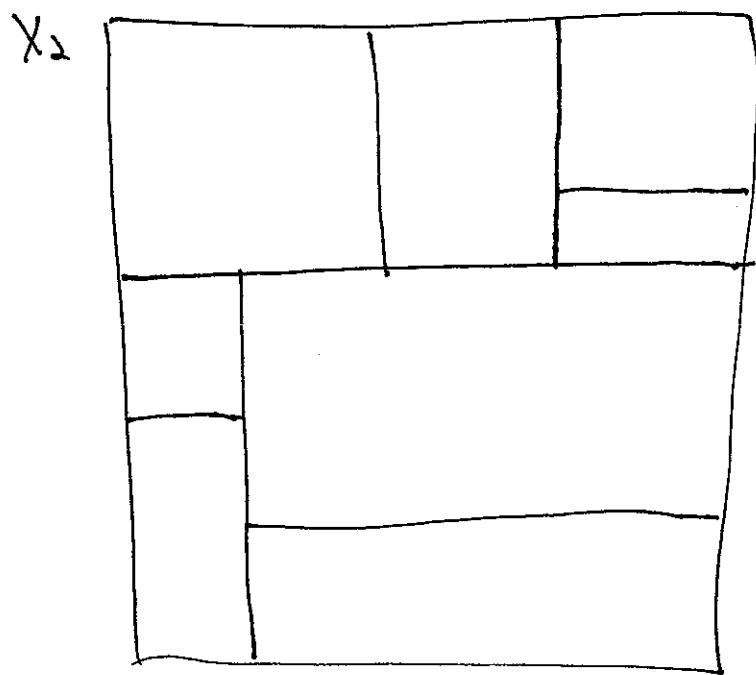
---

<sup>2</sup> **Performance of CART**

<sup>3</sup> **Advantages to Classification and Regression Trees (CART)**

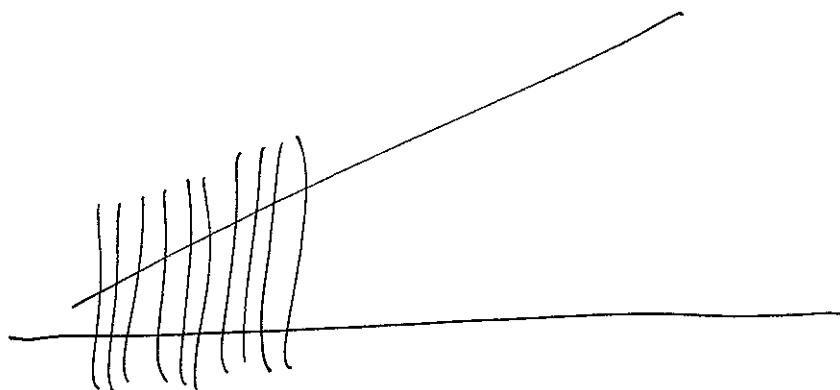
- <sup>4</sup> 1. The interpretation is very intuitive.
- <sup>5</sup> 2. These models are able to fit to complex interactions between the
- <sup>6</sup> predictors and their relationship with the response.
- <sup>7</sup> 3. Variable selection is a by-product of the cross-validation proce-
- <sup>8</sup> dure: Unimportant predictors do not appear in the tree.
- <sup>9</sup> 4. They have a very natural way of handling categorical response
- <sup>10</sup> and predictors.
- <sup>11</sup> 5. There are natural ways of dealing with missing values for the
- <sup>12</sup> predictors. See page 311 in Hastie, Tibshirani, and Friedman
- <sup>13</sup> (2009).
- <sup>14</sup> 6. There is no concern over transforming predictors. In particular,
- <sup>15</sup> extreme values of predictors are no problem.
- <sup>16</sup> 7. The computation cost is minimal.

~~unimportant~~



## 1 Disadvantages to CART

- 2 1. In cases where the relationship is smooth and simple, for exam-
- 3 ple  $Y = \beta_0 + \beta_1 x + \epsilon$ , it can take many more degrees of freedom
- 4 (parameters) to come close to the quality of fit achieved by linear
- 5 regression or nonparametric regression.
- 6 2. The predictions are not continuous; a small change in a predictor
- 7 can result in a significantly different prediction.
- 8 3. The trees can be very unstable. For instance, repeating the cross-
- 9 validation procedure several times will reveal many changes in
- 10 the size and structure of the tree.



2 **Random Forests**

- 3 The instability of trees has led to the consideration of modifications
- 4 with less variance in their predictions.
  
- 5 A simple extension is **bagging**.
  
- 6 The term "bagging" is a shortening of "Bootstrap Aggregation."
  
- 7 Our introduction to the bootstrap described it as a means for estimat-
- 8 ing standard errors via repeated resampling of the data.
  
- 9 Here, we also again construct  $B$  new data sets by drawing with re-
- 10 placement from the original data, but now we construct a tree using
- 11 each of these data sets, use each tree to make a prediction for the
- 12 covariate vector of interest.  $X^*$
  
- 13 Ultimately, all of the predictions are averaged (in the case of regres-
- 14 sion) or a majority vote is taken (in the case of classification) to de-
- 15 termine the final prediction.

"bootstrap samples"

- 1 Bagging illustrates a general concept that has proven very useful:
- 2 The aggregation of many simple, rough estimators can lead to an
- 3 estimator with both high precision and accuracy.
  
- 4 Such techniques are under the framework of **ensemble learning**.
  
- 5 Here we will talk about one very powerful ensemble learning tech-
- 6 nique for trees, called **random forests**.
  
- 7 Of course, the best tool will vary from problem to problem, but var-
- 8 ious assessments of the quality of machine learning classification
- 9 techniques have earned random forests the title of the “best off-the-
- 10 shelf classifier.”
  
- 11 Random forests are a generalization of bagging applied to trees.

## 1 The Random Forest Algorithm

2 From HTF, page 588:

3

4 1. For  $b = 1, 2, \dots, B$ :

5 (a) Draw a bootstrap sample of size  $n$  from the training data.

6 (b) Grow a tree using the data in the bootstrap sample.  $\cap^n$

7 **But:** After each split, choose  $m$  of the  $p$  predictors at ran-  $m \leq p$   
8 dom, and only consider splits from among those  $m$  predic-  
9 tors. Grow the tree until no further splits can be made with-  
10 out the minimum node size dropping below  $s$ .  $\Rightarrow T_b$

11 2. Output all of the trees  $T_1, T_2, \dots, T_B$ .

12 **For regression**, the prediction at  $x$  is found by averaging the predic-  
13 tion from all  $B$  trees.

14 **For classification**, the prediction at  $x$  is found by taking a vote of the  
15  $B$  trees.

---

**Comments:**

- Comment 1:** The default choice for  $B$  is 500.
- Comment 2:** The default choice for  $m$  is  $\lfloor p/3 \rfloor$  in the case of regression and  $\lfloor \sqrt{p} \rfloor$  in the case of classification. In practice, different values of  $m$  should be tried, and performance compared.
- Comment 3:** The default choice for  $s$  is 5 in the case of regression and 1 in the case of classification, but, again, one should consider the effects of varying this setting.
- Comment 4:** There is no need to do cross-validation with these trees in order to set these tuning parameters. With each of the  $B$  bootstrap samples, there will be some of the original  $n$  observations which are left out; these form the **out-of-bag (OOB)** sample.  
After a tree is fit, the OOB sample is used as a test set: predictions are made, and the deviance is calculated. This quantity is accumulated over all  $B$  trees.

## 1 Implementation in R

- 2 Random forests are implemented in the R package `randomForest`.
- 3 The main function is `randomForest()`; the syntax is much like
- 4 other model fitting functions we have used.

```
5 > model9 = randomForest(transresp ~ timetoexpiry  
6           + strike + curprice + histvol, data=alldat)
```

- 7 If one wants to change  $B$ , this is done via the argument `ntree`.
- 8 If one wants to change  $s$ , this is done via the argument `nodesize`.
- 9 If one wants to change  $m$ , this is done via the argument `mtry`.

- If one runs `plot (model9)`, the result is the plot shown as Figure 14.

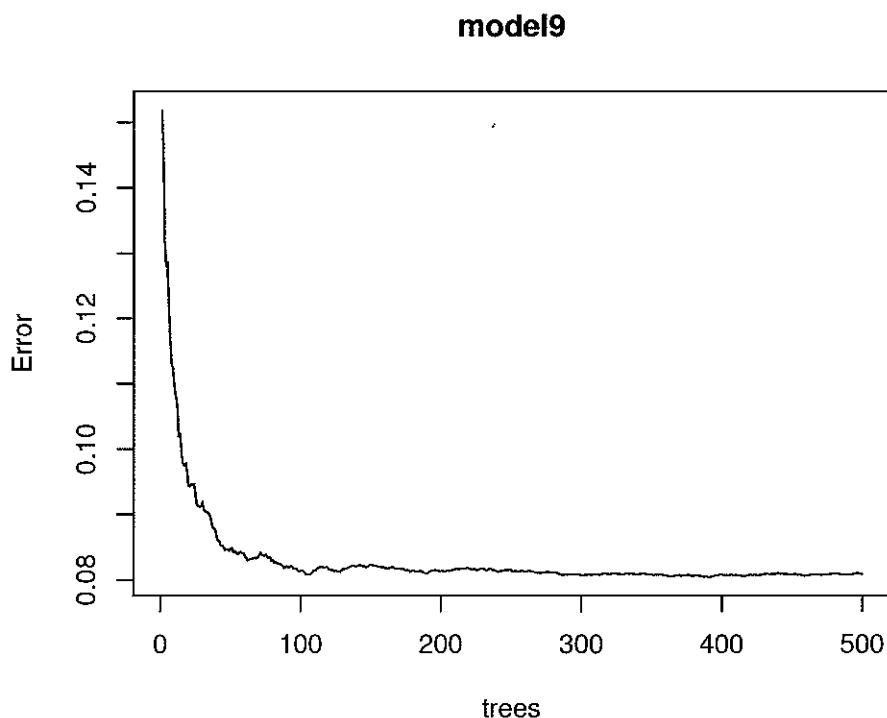


Figure 14: The result of `plot (model9)`.

- This figure shows how the OOB sample error decreases as more trees are included.
- Exercise:** Do you believe that a sufficient number of trees were included? In other words, was  $B$  chosen large enough?

$B$  is larger than it needed to be. It seems that  $B \approx 100$  would give same OOB error.

, Figure 15 is the plot of residuals versus fitted values in this case.

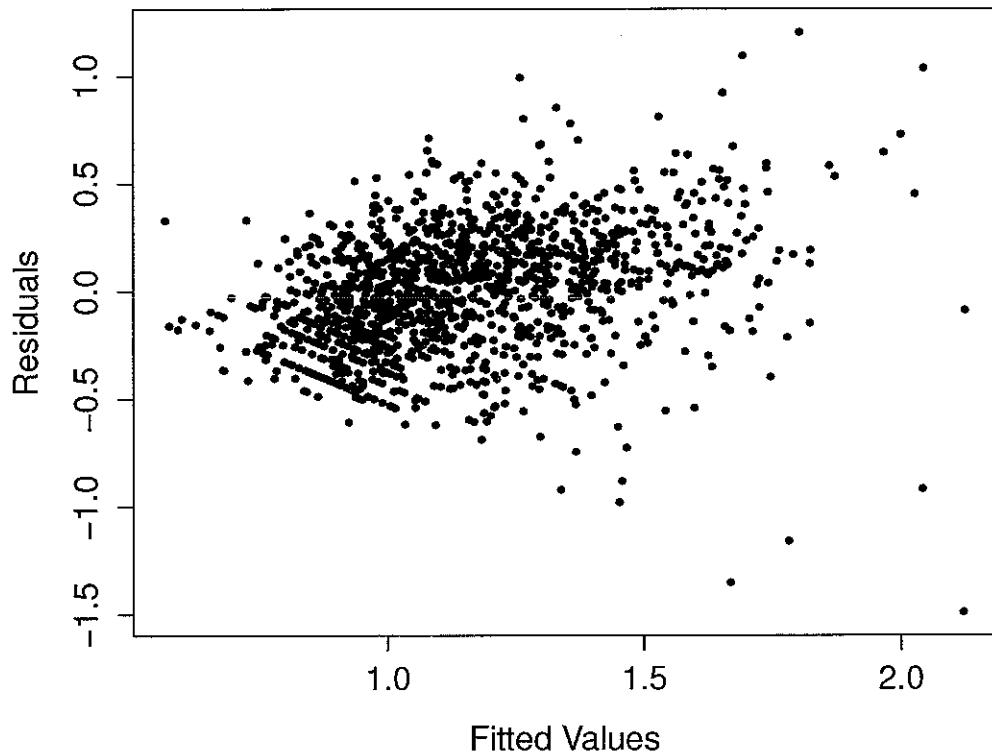


Figure 15: The plot of residuals versus fitted values for Model Nine.

- 1 Figure 16 shows the actual response values versus the predicted val-
- 2 ues.

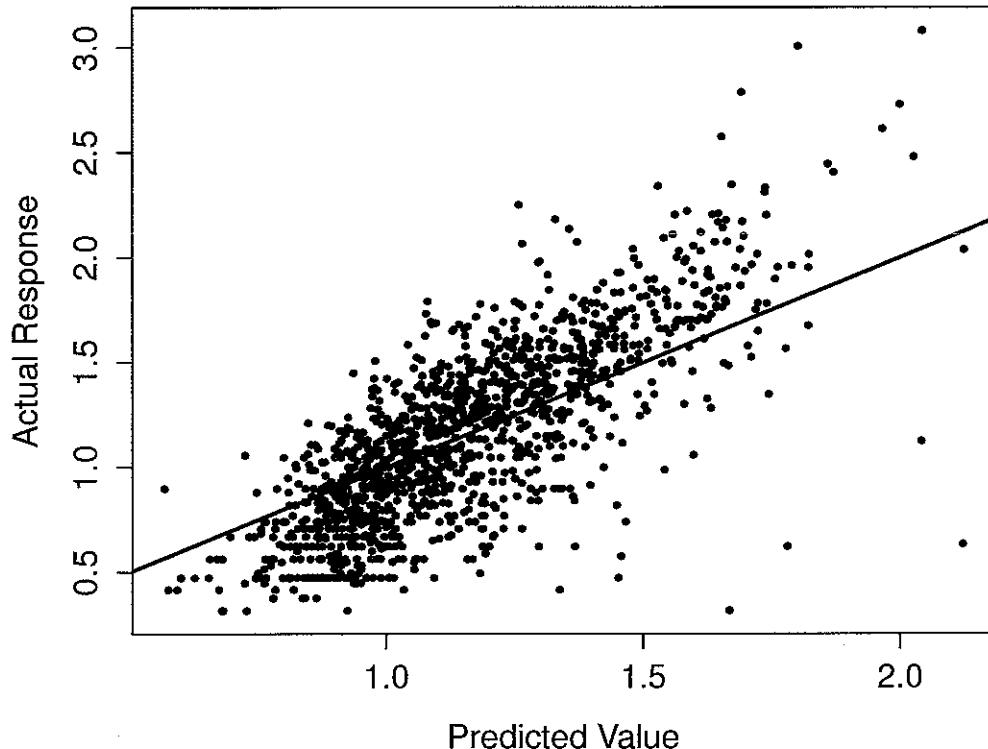


Figure 16: The plot of observed responses versus predicted values for Model Nine. The solid line indicates perfect agreement between the two.

- 3 The fit of this model is clearly not acceptable.
- 4 The main tuning parameter for Random Forests is  $\underline{m}$ , the number
- 5 of predictors which are sampled at each step in order to determine
- 6 which set of cuts to search over.

1 The package `randomForest` includes function `tuneRF()` for searching  
2 for the optimal choice of  $m$ ; this decision is based on OOB error  
3 performance.

4 In order to use this function, one needs to construct a matrix which  
5 contains only the predictors:

6 > xmat = alldat[,c(1,2,8,9)]

7 Next, call `tuneRF()`:

8 > `tuneRF(xmat, alldat$transresp)`

9 The output will appear as follows:

```
10 mtry = 1  OOB error = 0.08497548
11 Searching left ...
12 Searching right ...
13 mtry = 2  OOB error = 0.06243305
14 0.2652816 0.05
15 mtry = 4  OOB error = 0.04697615
16 0.2475755 0.05
17 mtry      OOBError
18 1      1 0.08497548
19 2      2 0.06243305
20 4      4 0.04697615
```

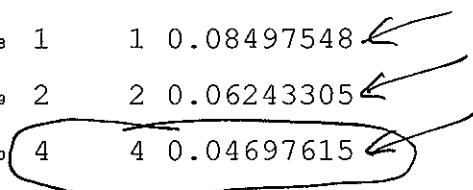


Figure 17 is the plot of residuals versus fitted values in this case.

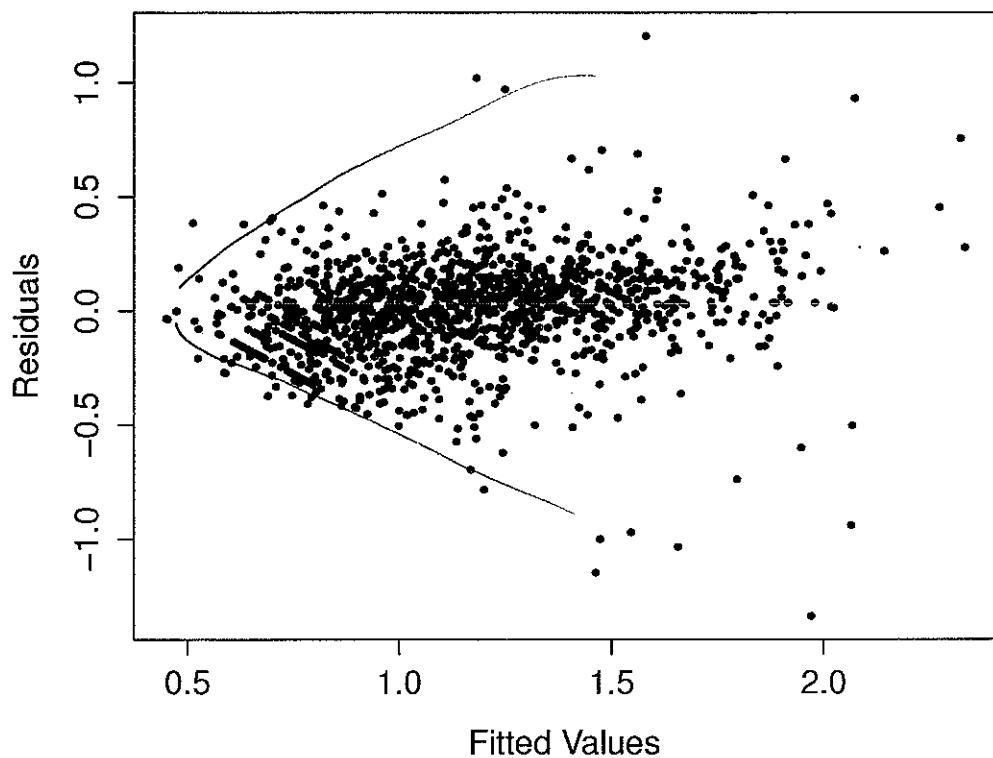


Figure 17: The plot of residuals versus fitted values for Model Ten.

- 1 Figure 18 shows the actual response values versus the predicted val-
- 2 ues.

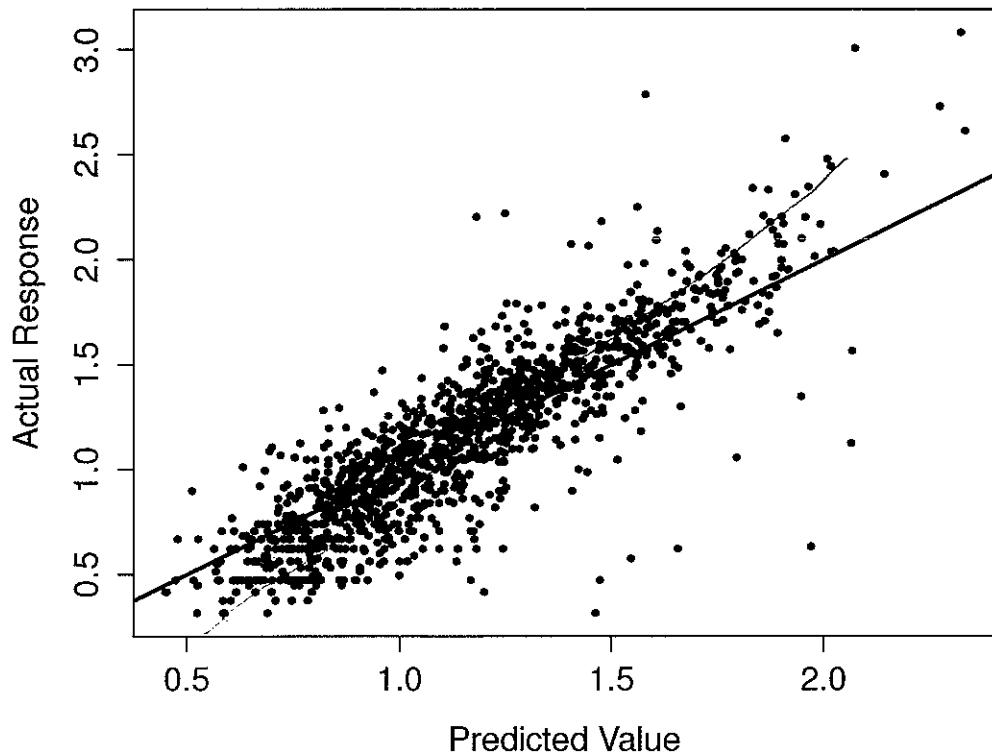


Figure 18: The plot of observed responses versus predicted values for Model Ten. The solid line indicates perfect agreement between the two.

- 3 **Exercise:** Comment on the quality of the fit of this model.

4 In this example, does not give great result.

5 Model does not fit well.

6

7

8

- Exercise:** If we wanted to truly make a decision between these ten models, what steps should we take?

3 Generate a test set and compare  
4 estimates of generalization error.

5

---

6

---

7

---

8

---

9

---

10

---

11

---

12

---

13

## Part 9: Boosting

- <sup>2</sup> Text references: §8.2.3 in ISL, Chapter 10 in HTF
- <sup>3</sup> Another approach to ensemble learning is **boosting**.
- <sup>4</sup> While bagging is built on combining weak classification/regression
- <sup>5</sup> models, each fit to a bootstrap sample, boosting uses a *sequence* of
- <sup>6</sup> weak models that *adapt* in an attempt to improve the overall predic-
- <sup>7</sup> tion. The final model is a weighted combination of each model in the
- <sup>8</sup> sequence.
- <sup>9</sup> There are a range of boosting algorithms, and they can be used in
- <sup>10</sup> conjunction with almost any regression or classification method.
- <sup>11</sup> We will start by demonstrating boosting with trees. The develop-
- <sup>12</sup> ment follows that of Section 8.2.3 in ISL.

1

## 2 The Boosting Algorithm for Regression Trees

3 Below is a boosting algorithm for use with regression trees. As be-  
4 fore, the objective is to estimate  $f(\mathbf{x})$ . In what follows, think of the  $r_i$   
5 as the residuals under the current model.<sup>1</sup>

---

6

- 7 1. Set  $\hat{f}(\mathbf{x}) = 0$  and  $r_i = y_i$  for all  $i$ .  
8 2. For  $b = 1, 2, \dots, B$ , repeat the following:

9 (a) Fit a regression tree  $\hat{f}^b$  with  $d$  splits ( $d + 1$  terminal nodes) to  
10 the full data  $(\mathbf{x}_i, y_i)$ .

11 (b) Update  $\hat{f}$  via

$$12 \hat{f}(\mathbf{x}) \leftarrow \hat{f}(\mathbf{x}) + \lambda \hat{f}^b(\mathbf{x}) \quad \underbrace{\lambda}_{\text{shrinkage parameter}}$$

13 (c) Update the residuals via

$$14 r_i \leftarrow r_i - \lambda \hat{f}^b(\mathbf{x}_i) \quad \text{for all } i$$

15 3. Output the final model:

$$16 \hat{f}(\mathbf{x}) = \sum_{b=1}^B \lambda \hat{f}^b(\mathbf{x})$$

---

<sup>1</sup>This is Algorithm 8.2 in ISL.

- 1 **Exercise:** What is accomplished by fitting each successive tree to the
- 2 *residuals* of the current model?

3 The successive models are fit to the residuals  
4 in an effort to explain the "yet-to-be  
5 explained" variability in the response.

6 The model is "Learned slowly" p.322 in ISL

7 "We slowly improve  $\hat{f}$  in areas that it  
8 does not fit well."

9 With each iteration, RSS will decrease.

10 The algorithm can be viewed as a  
11 search for the optimum.

12 "Gradient Boosting Machines" (`gbm()`)

## 1 The Tuning Parameters

- 2 There are three tuning parameters for this algorithm:  $\lambda$ ,  $d$ , and  $B$ .
- 3 ISL cites “typical” values of  $\lambda$  as 0.01 and 0.001. This means that each
- 4 individual model has a small influence on the final model.
- 5 The number of splits,  $d$ , is usually set to be very small, even  $d =$
- 6 1 is used. The point is that each of the component models should
- 7 be simple and hence fit to “local” features in the relationship. As is
- 8 the theme with ensemble learning, each component model should
- 9 be simple, with low bias and high variance. Averaging these models
- 10 should result in a model with lower variance, but still with low bias.
- 11 The value of  $B$  is set using cross-validation; it may be necessary to
- 12 set  $B$  to be very large in order to obtain a good model.

## 1 Boosting Trees in R

- 2 We will use the function `gbm()` as part of package `gbm`. The syntax  
3 for the call is as follows:

```
4 > model11 = gbm(transresp ~ timetoexpiry + strike +  
5   curprice + histvol, data = alldatsub, bag.fraction = 0.5,  
6   n.trees = 200000, distribution = "gaussian",  
7   interaction.depth = 3, shrinkage = 0.001, cv.folds=3)
```

- 8 • The value of  $d$  is controlled by `interaction.depth`. Note that  
9 the default value is one.
- 10 • The `distribution` argument sets the *loss function* that is used  
11 in the fitting. Using ``gaussian'' specifies squared error loss.  
12 See the `help()` file for the choices.
- 13 • The argument `bag.fraction` controls what proportion of the  
14 data are used in the fitting of each tree model in the sequence.  
15 The default value is 0.5. *Downsampling*  
16 • The argument `n.trees` sets  $\lambda$ , the maximum number of tree  
17 models. The default value is 100, which is too small for most  
18 practical applications. Here, I have set it to 200000.  
19 • The argument `shrinkage` sets  $\lambda$ . Here I am using the default  
20 value of 0.001.

1 Cross-Validation to Choose  $B$ 

- 2 By setting `cv.folds = 3`, I am asking the procedure to use three-  
 3 fold cross-validation. Due the computational demands, it is difficult  
 4 to use many folds.
- 5 The function runs cross-validation for each choice of  $B$  as it pro-  
 6 gresses through the sequence. There is a built-in function, `gbm.perf()`  
 7 which returns the plot shown below as Figure 1. Based on this, I  
 8 choose to use  $B = 50000$ , as the improvement beyond that choice  
 9 seems minimal.

10 > `gbm.perf(model11, method="cv")`

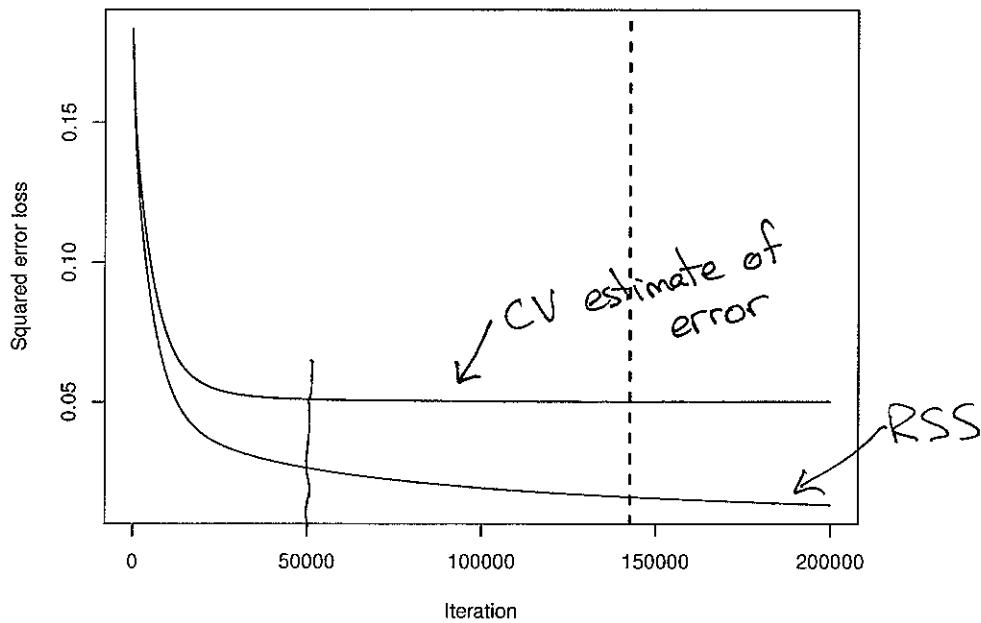


Figure 1: The output of `gbm.perf()` in this case. The top curve shows how the cross-validated estimate of the error changes with  $B$ . The bottom curve shows how RSS is decreasing with  $B$ .

- **Important Note on use of gbm():**
- There is a bug in the code for gbm() if using cv.folds greater than zero.
- For reasons that are not clear, it is necessary that all columns of the provided data frame must be used in the model (either as predictors or as the response).
- Hence, in our case, I used the command
- > alldatsub = alldat[,c(1,2,8,9,11)]
- and then used alldatsub in the call to gbm().

Figure 2 is the plot of residuals versus fitted values in this case.

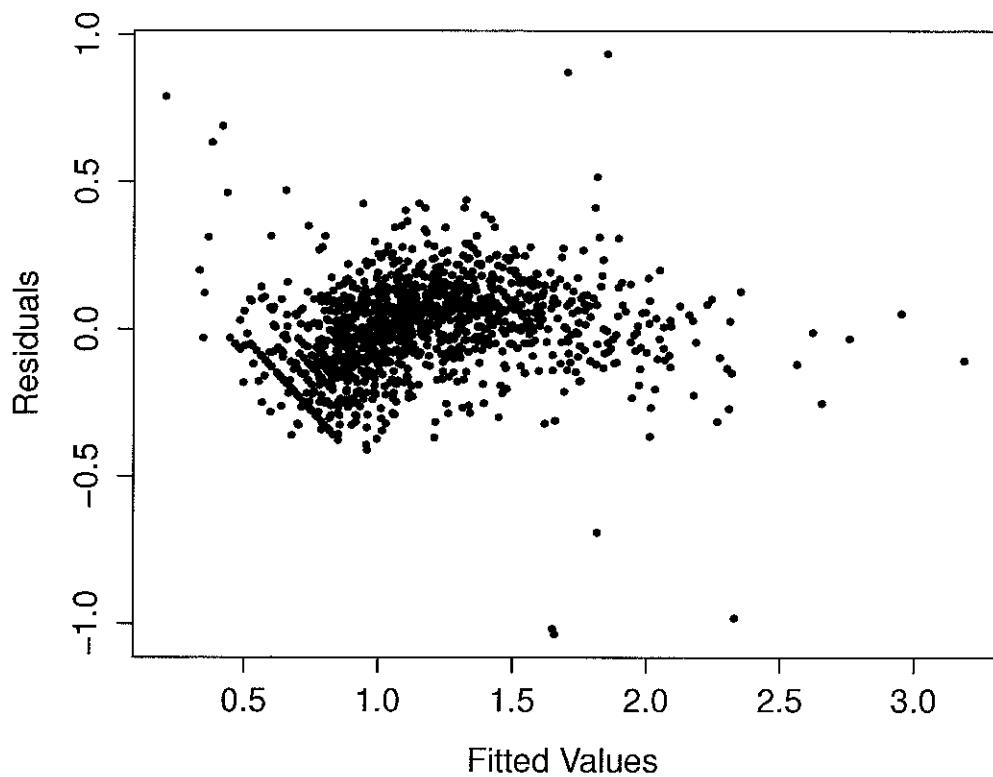


Figure 2: The plot of residuals versus fitted values for Model Eleven.

Figure 3 shows the actual response values versus the predicted values. The deviance for this model is 35.47.

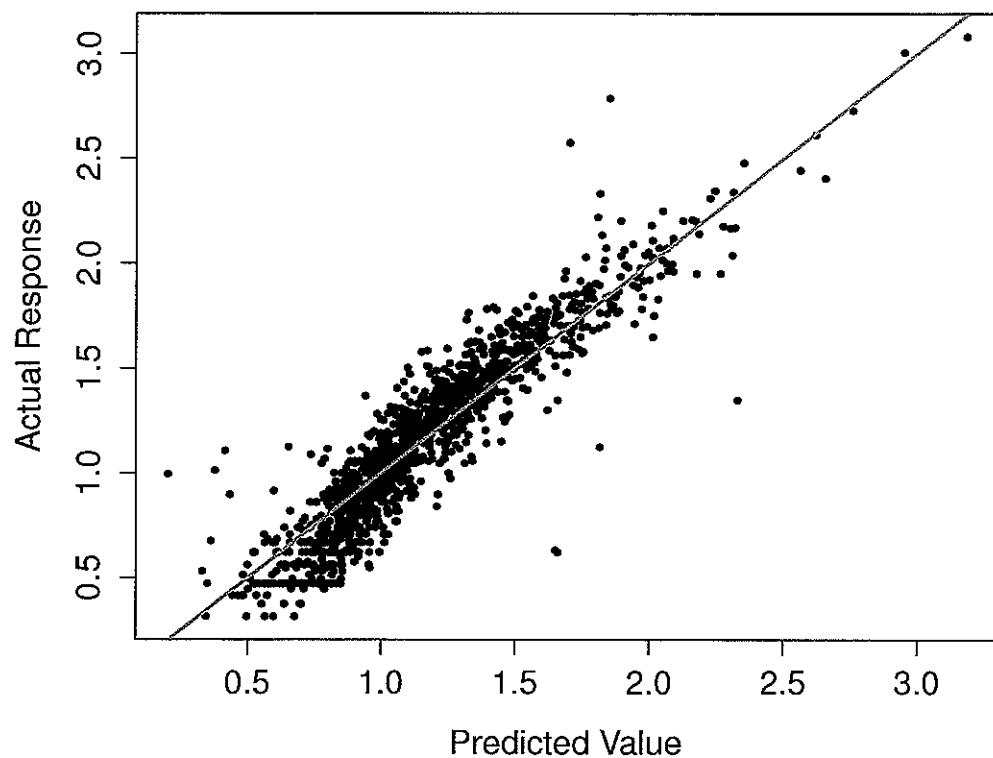


Figure 3: The plot of observed responses versus predicted values for Model Eleven. The solid line indicates perfect agreement between the two.

1

---

## 2 Boosting for Classification

- 3 The regression algorithm presented above could be extended beyond
- 4 regression trees by replacing the trees models with another regres-
- 5 sion procedure.
  
- 6 This algorithm could also be applied to the classification setting; in-
- 7 deed, this was the original setting in which boosting was considered.
  
- 8 Boosting methods for classification build a sequence of *weak classi-*
- 9 *fiers* which are then combined to create a single, more-powerful clas-
- 10 *sifier*. Instead of fitting the models to residuals (as done in the regres-
- 11 *sion case*), the *weights* placed on each observation are adjusted with
- 12 each classifier.
  
- 13 **Exercise:** How is adjusting the weights analogous to fitting to the
- 14 residuals?

15 By putting more weight on an observation, you're  
16 "focusing the effort" of the classifier on  
17 that observation

18

## The AdaBoost Algorithm

- 2 Below is Algorithm 10.1 in HTF, the *AdaBoost.M1* approach.<sup>2</sup> The  
3 setup here is that the goal is to classify into one of two groups with  
4 labels  $y_i \in \{-1, 1\}$  using predictors  $\mathbf{x}_i$ .
- 

6 1. Initialize the observations weights  $w_i = 1/n$  for  $i = 1, 2, \dots, n$ .

7 2. For  $b = 1, 2, \dots, B$ :

8 (a) Fit a classifier  $G_b(\mathbf{x})$  to the training data using weights  $w_i$ .

9 (b) Compute

$$10 \quad \text{err}_b = \frac{\sum_{i=1}^n w_i I(y_i \neq G_b(\mathbf{x}_i))}{\sum_{i=1}^n w_i} \quad \text{Indicator}$$

11 (c) Compute

$$12 \quad \alpha_b = \log((1 - \text{err}_b)/\text{err}_b) \quad \text{weight (AdaBoost)}$$

13 (d) Update the weights to

$$14 \quad w_i \leftarrow w_i \exp[\alpha_b I(y_i \neq G_b(\mathbf{x}_i))]$$

15 3. Output the final classifier:

1, -1

$$16 \quad G(\mathbf{x}) = \text{sign} \left[ \sum_{b=1}^B \alpha_b G_b(\mathbf{x}) \right]$$

---

<sup>2</sup>Y. Freund, and R. Shapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Proceedings of the Second European Conference on Computational Learning Theory*, 1995, pp. 2337.

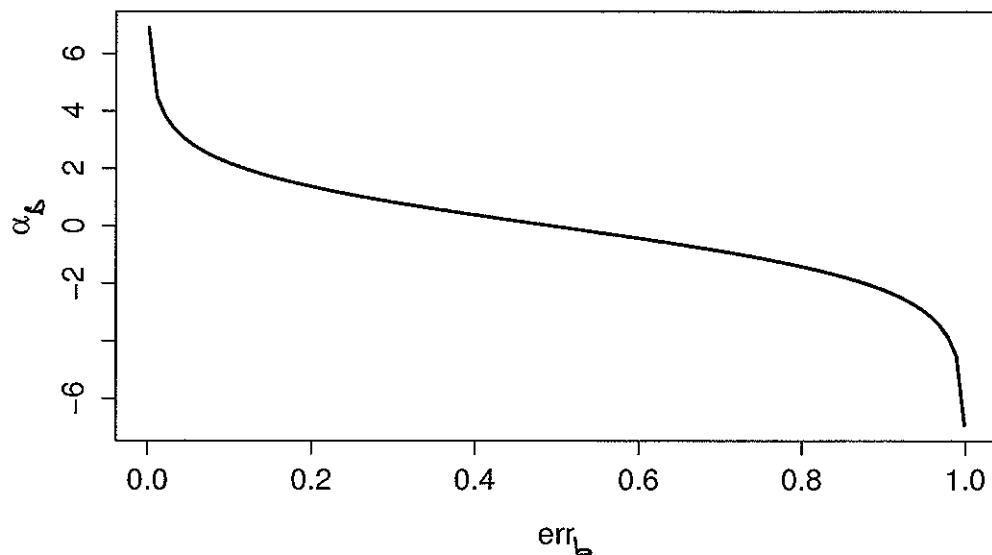


Figure 4: The AdaBoost weighting function.

- 2 **Exercise:** Under what circumstances would  $\alpha_b$  be large? Under what  
 3 circumstances would  $w_i$  be large?

4  $\alpha_b$  large when error for  $G_b$  is small

5  
 6  $w_i$  increases a relatively large amount when  
 7  $\exp(\alpha_b I(y_i \neq G_b(x_i)))$  is large. This  
 8 occurs when  $\alpha_b$  is large and  ~~$y_i \neq G_b(x_i)$~~ ,  
 9 i.e. when  $G_b$  is a good classifier but it  
 10 got obs.  $i$  wrong

11

12

## 1 AdaBoost in R

- 2 One can use the above AdaBoost algorithm with gbm() by specifying  
3 distribution="adaboost". Note that the response should  
4 be 0/1.

```
5 ratedata = read.table(  
6   "http://www.stat.cmu.edu/~cschafer/MSCF/RateData.txt",  
7   header=T)  
8  
9 # Create new binary response  
10  
11 ratedata$lowhi = as.numeric(ratedata$Rating == "High")  
12  
13 # Fit using AdaBoost  
14  
15 set.seed(0)  
16  
17 rateadaboost = gbm(lowhi ~ factor(ext.source) +  
18   factor(EDF.source) + Profit + Capital +  
19   Excess.Net.Capital + Liab.Capital + Leverage +  
20   Liquidity.1 + Liquidity + factor(Descr),  
21   data = ratedata, distribution = "adaboost",  
22   n.trees = 20000, cv.folds = 3)
```

## Part 10: Support Vector Machines

- 2 Text references: Chapter 9 in ISL and Chapter 12 in HTF
- 3 **Support Vector Machines** are a class of methods useful for both clas-
  - 4 sification and regression (but more commonly used for classification)
  - 5 based on separating hyperplanes.
- 6 As with all methods we consider, there are a wide range of gener-
  - 7 alizations and extensions, but here we will focus on the basic, most
  - 8 useful elements.
- 9 Our development will follow closely that of the texts cited above
  - 10 (which are quite similar).

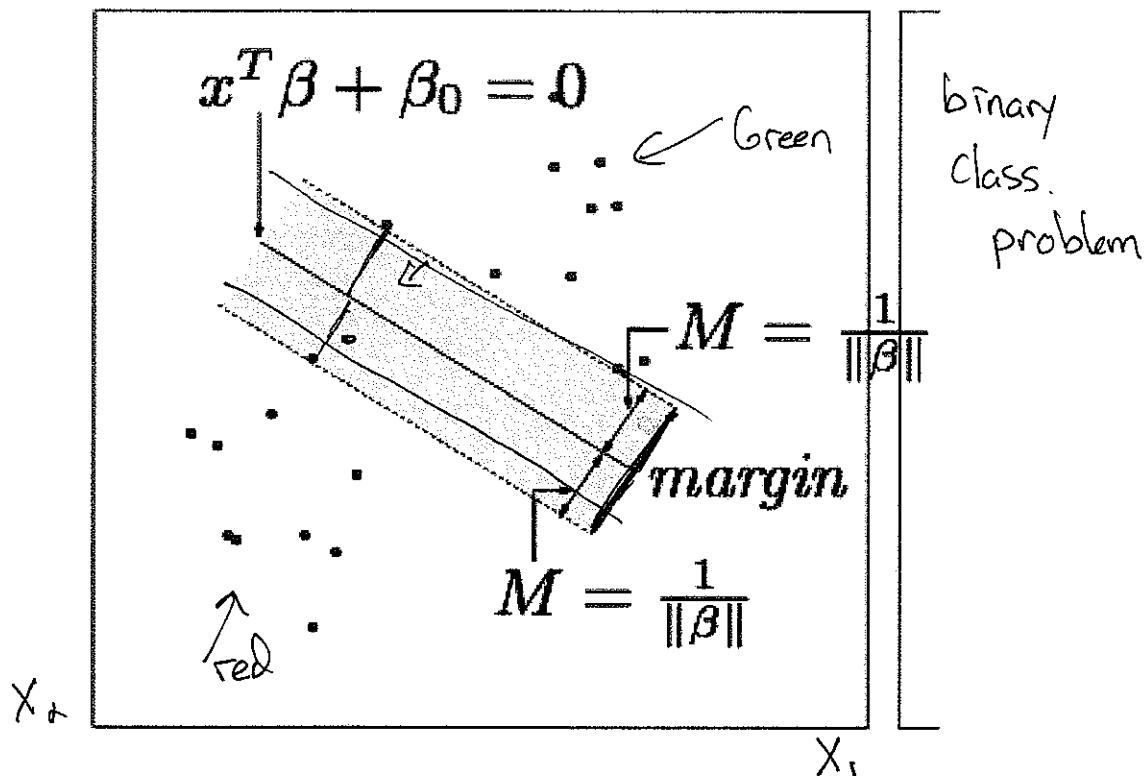


Figure 1: The linearly separable case. This is from Figure 12.1 in HTF.

- 1 The simplest case is illustrated in Figure 1. In this case the response
- 2 belongs to one of two classes, with labels given by the colors of the
- 3 points. There are two predictors, represented by the two axes in the
- 4 plot.
  
- 5 We start by asking: Is there a line that separates the two classes? In
- 6 this case, it is clear that there are many such lines. This is the **linearly**
- 7 **separable** case.

- 1 The approach taken is to find the one such separating line that max-
      - 2 imizes the **margin** of the classifier. The margin is defined to be twice
      - 3 the distance from the separating line to the closest member of each
      - 4 class.
    - 5 By construction, there will be some of the data points which lie on
      - 6 the boundary of the marginal region (the shaded region in Figure 1).
      - 7 These are called the **support vectors**.
    - 8 **Exercise:** What is the justification for choosing the separating line
      - 9 that maximizes the margin?

<sup>10</sup> Our objective is to extend classifier to the  
<sup>11</sup> larger population. The hope is that we'll  
<sup>12</sup> minimize the error rate by maximizing  
<sup>13</sup> the space between the line and the  
<sup>14</sup> observed points in the training sample.

<sup>15</sup>

- 1 Of course, expecting classes to be linearly separable is not realistic.
- 2 To account for this, the idea of **slack variables**, denoted  $\xi_i^*$ , are included.
- 3
- 4 As shown in Figure 2, some observations are allowed to fall into the
- 5 marginal region, or even be on the incorrect side of the classification
- 6 boundary.
- 7 The slack for observation  $i$  is equal to how far within the marginal
- 8 region that observation falls. The slack is defined to be zero for an
- 9 observation that is on the correct side of the marginal region.

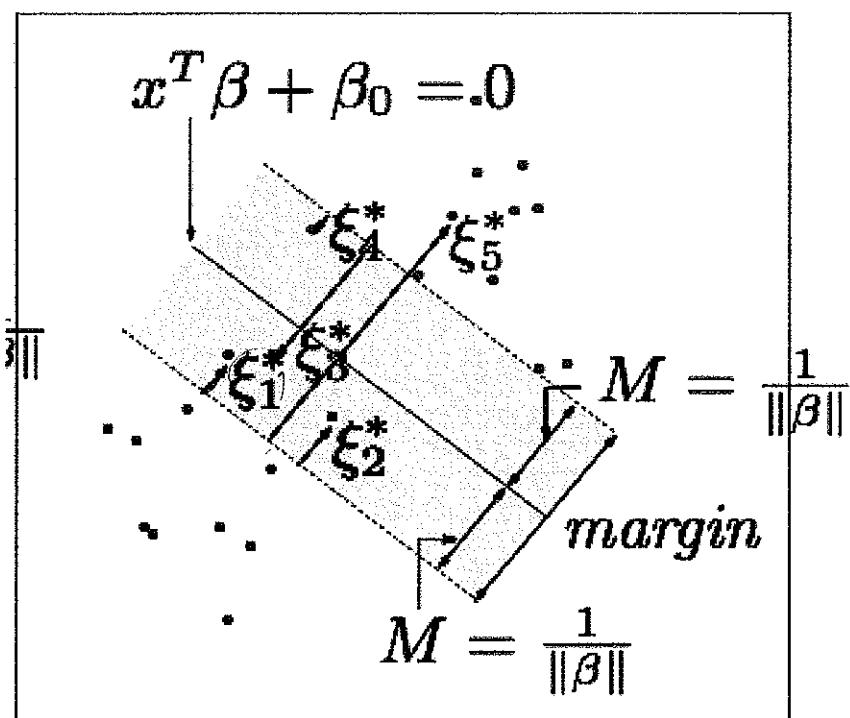


Figure 2: Case where the classes are not linearly separable. This is from Figure 12.1 in HTF.

## 1 The Optimization Problem

- 2 The new optimization problem is to maximize the margin subject
- 3 to the sum of the slack variables (normalized by  $M$ ) being less than
- 4 some chosen constant.
- 5 Formally, this can be shown to be equivalent to minimizing

$$\frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^n \xi_i$$

- 6
- 7 where  $\xi_i \geq 0$  and  $y_i(\mathbf{x}_i^T \beta + \beta_0) \geq 1 - \xi_i$  for all  $i$ . Here,  $\xi_i = \xi_i^*/M$  and
- 8  $C$  is the **cost parameter**. Also, note that  $y_i$  equals either 1 or -1.

- 9 This can further be shown to be equivalent to maximizing over  $\alpha_i$
- 10 (the **dual problem**):

$$\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{i'=1}^n \alpha_i \alpha_{i'} y_i y_{i'} \mathbf{x}_i^T \mathbf{x}_{i'}$$

- 11
- 12 subject to  $0 \leq \alpha_i \leq C$  and  $\sum_{i=1}^n \alpha_i y_i = 0$ .

- 13 See Section 12.2.1 in HTF for more details on the optimization prob-
- 14 lem.

---

## <sup>2</sup> Moving to Higher Dimensions

- <sup>3</sup> Even with the inclusion of the slack variables, you could probably
- <sup>4</sup> imagine that linear separators will not do a good job of classifying
- <sup>5</sup> in complex, real-world problems. The relationships we see are not
- <sup>6</sup> typically that simple.

- <sup>7</sup> The solution is transform the original predictor vector  $\mathbf{x}$  into an  $M$ -dimensional space using a function  $\mathbf{h}$ . So, the new predictor vector
- <sup>8</sup> would be

$$\in \mathbb{R}^M$$

<sup>10</sup> 
$$\underline{\mathbf{h}(\mathbf{x}_i)} = (h_1(\mathbf{x}_i), h_2(\mathbf{x}_i), \dots, h_M(\mathbf{x}_i))$$

- <sup>11</sup> The point is that with  $M$  sufficiently large, and  $\mathbf{h}$  well-chosen, the
- <sup>12</sup> original data is represented in a higher-dimensional space such that
- <sup>13</sup> a linear separator does work well.

- 1 Figure 3 shows a simple example. Here, a standard, polynomial  
 2 mapping is used from two to three dimensions. Specifically,

3  $z_1 = x_1^2, z_2 = \sqrt{2}x_1x_2, z_3 = x_2^2$   
 $h_1(\underline{x}) \quad h_2(\underline{x}) \quad h_3(\underline{x})$

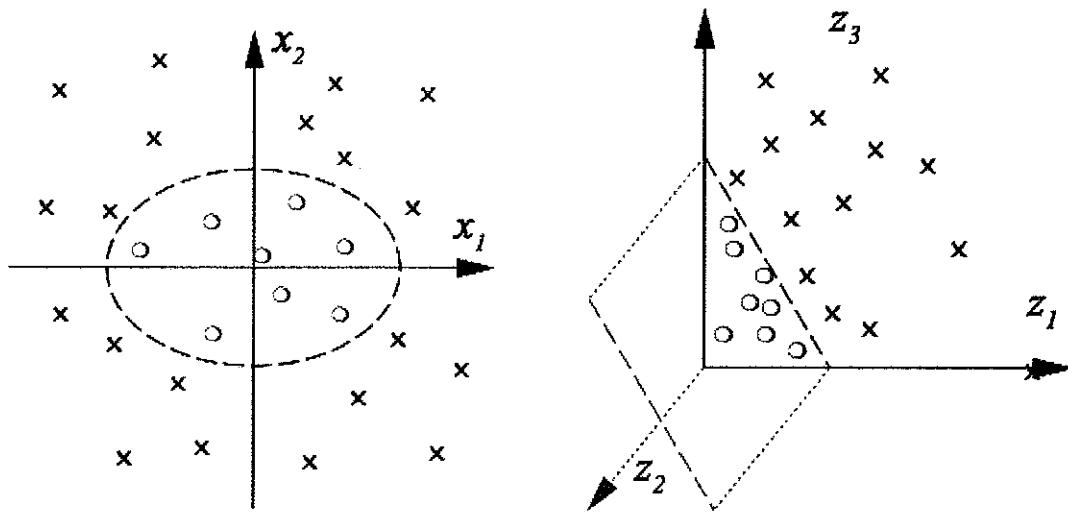


Figure 3: Not linearly separable before transformation, but separable afterwards.

- 4 It turned out that the second dimension ( $z_2$ ) was not even important,  
 5 but that is fine. In practice, data will be mapped into a much higher  
 6 dimension than their original form, so that there are many “oppor-  
 7 tunities” for linear separators to be found.

<sup>1</sup> **The “Kernel Trick”**

<sup>2</sup> Now the optimization problem would look like

<sup>3</sup> 
$$\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{i'=1}^n \alpha_i \alpha_{i'} y_i y_{i'} \mathbf{h}(\mathbf{x}_i)^T \mathbf{h}(\mathbf{x}_{i'})$$
  
<sup>4</sup> subject to  $0 \leq \alpha_i \leq C$  and  $\sum_{i=1}^n \alpha_i y_i = 0$ .

$K(\mathbf{x}_i, \mathbf{x}_{i'})$

<sup>5</sup> We immediately note that the optimization only depends on  $\mathbf{h}$  through  
<sup>6</sup> the inner product, which we'll denote

<sup>7</sup> 
$$K(\mathbf{x}_i, \mathbf{x}_{i'}) = \mathbf{h}(\mathbf{x}_i)^T \mathbf{h}(\mathbf{x}_{i'})$$

<sup>8</sup> and we'll call  $K$  the **kernel function**.

<sup>9</sup> We have replaced the need to choose  $\mathbf{h}$  with the need to choose the  
<sup>10</sup> kernel  $K$ .

<sup>11</sup> For any choice of kernel, there are corresponding functions  $\mathbf{h}$ , but we  
<sup>12</sup> don't actually need to find them, and we typically use kernels that  
<sup>13</sup> would make such derivation difficult or impossible.

<sup>14</sup> This substitution is called the **kernel trick**.

- 1 The kernel is a function which returns a measure of **similarity** between pairs of data. Popular choices include:

- 3 1. *The linear kernel:*

4  $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}' \leftarrow$  our first examples

- 5 2. *The  $d^{th}$ -degree polynomial:*

6  $K(\mathbf{x}, \mathbf{x}') = (\tau + \gamma \mathbf{x}^T \mathbf{x}')^d$

- 7 3. *The radial basis:*

8  $K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$

gets larger  
as  $\mathbf{x}, \mathbf{x}'$  are  
more similar

- 9 4. *The neural network (sigmoid):*

10  $K(\mathbf{x}, \mathbf{x}') = \tanh(\gamma \mathbf{x}^T \mathbf{x}' + \tau)$

- 11 Note that it is standard to scale predictors so that they have mean zero and variance one. Hence, the inner product is larger for more similar vectors  $\mathbf{x}$  and  $\mathbf{x}'$ .

- 14 The radial basis is a very standard choice, and often serves as the default by software.

- 1 Part of the beauty of this is that we could take an object  $x$  to be just
      - 2 about anything, of any dimension. All we need is some way of mea-
      - 3 suring the similarity of a pair of such objects, i.e., we need to con-
      - 4 struct a kernel function.
  - 5 For example,  $x$  can be a long time series, an image, a document, etc.
  - 6 For a range of examples of applications of SVMs in finance, see the
    - 7 (out-of-date) list at <http://www.svms.org/finance/>

## 1 Constructing the Boundary

- 2 The function  $f$  that defines the boundary between the two classes is  
 3 of the form

4 
$$f(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T \beta + \beta_0$$

5 for some vector  $\beta$ .

- 6 The boundary is the class of  $\mathbf{x}$  such that  $f(\mathbf{x}) = 0$ .

- 7 It is not difficult to show that this can be rewritten, however, as

8 
$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + \beta_0$$

- 9 and it is also possible to show that  $y_i f(\mathbf{x}_i) = 1$  for any  $i$  for which  
 10  $0 < \alpha_i < C$ . This yields a simple way to find  $\beta_0$ .

- 11 **Exercise:** What happens to the value of  $f(\mathbf{x})$  when  $\mathbf{x}$  is similar to

12 many of the  $\mathbf{x}_i$  for which  $y_i = 1$ ? We would hope that  $f(\mathbf{x})$  would  
 be positive.

13 If  $\mathbf{x}$  is "close to"  $\mathbf{x}_i$ , then  $K(\mathbf{x}, \mathbf{x}_i)$  is large

14  $\sum_{i=1}^n \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)$  will be large ~~positive~~ when  
 15  $\mathbf{x}$  is similar to  $\mathbf{x}_i$  for which  $y_i = 1$

16 this will force  $f(\mathbf{x})$  to be large, positive

17 when  $\mathbf{x}$  is close to  $\mathbf{x}_i$  for which  $y_i = 1$

18 Hence,  $\mathbf{x}$  is classified as "1"

1

## 2 SVM in R

3 We will use the function `svm()` which is part of the package e1071.

4 The syntax is very simple, especially if you are willing to utilize the  
5 default settings:

6 > `svmout = svm(factor(change) ~ ., data=trainset)`

7 Recall that the `~ .` notation means to use all of the other variables in  
8 the data frame `trainset` as predictors.

9 Some additional arguments, and their default values:

10 • `scale = TRUE`: Specifies whether or not the predictors should  
11 be scaled prior to the analysis.

12 • `kernel = "radial"`: The kernel. The choices include `linear`,  
13 `polynomial`, `radial` and `sigmoid`.

14 • `cost = 1`: The value of  $C$ , the cost parameter.

15 • `gamma`: The  $\gamma$  parameter used in the kernels. The default value  
16 is one over the dimension of the predictor.

- <sup>1</sup> The output object contains many components, including:
  - <sup>2</sup> • `svmout$SV`: All of the support vectors.
  - <sup>3</sup> • `svmout$decision.values`: The fitted function  $\hat{f}$  evaluated at
  - <sup>4</sup> all of the data points.
  - <sup>5</sup> • `svmout$fitted`: The fitted values from the model.
  - <sup>6</sup> An additional function `tune.svm()` is useful for choosing tuning
  - <sup>7</sup> parameters. Usually, the focus is placed on the choice of the cost
  - <sup>8</sup> parameter  $C$ .
  - <sup>9</sup> The syntax is identical to the above, but with `cost` being provided
  - <sup>10</sup> as a list of values to be compared:
- ```
11 > svmout = tune.svm(factor(change) ~ .,  
12 data=trainset, cost=c(5,10,25,50))
```
- <sup>13</sup> After this command executes, the object `svmout$best.model` will
  - <sup>14</sup> be of the same type as the output of `svm()`, and will hold the best of
  - <sup>15</sup> models under comparison, as judged using ten-fold cross-validation.

## Example: High Frequency Trading

2 Here we will consider a recent high frequency trading challenge put  
3 forth here:

4 <http://www.circulumvite.com/home/trading-competition>

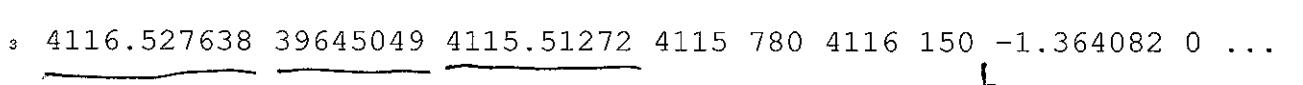
5 Briefly stated, the challenge was to predict the future value of the  
6 USD/BRL exchange rate using some proprietary "indicators."

7 Training data were provided, from which one could build a model.  
8 Here we will focus on the problem of predicting the direction of the  
9 change in the rate 120 seconds in the future.

10 I took the first day of the data from the challenge and created a file  
11 located at

12 [www.stat.cmu.edu/~cschafer/MSCF/cprod\\_data\\_20130103v.txt](http://www.stat.cmu.edu/~cschafer/MSCF/cprod_data_20130103v.txt)

13 These data are directly from the challenge; the only change I made  
14 was to add the first column which gives the rate 120 seconds in the  
15 future.

- 1 The full training data consisted of 118 days of trading.
- 2 First, consider the format of one line of one of these files:  
  
4 The columns are as follows:
  - 5 1. The price 120 seconds in the future. (This is the “weighted aver-
  - 6 age price” mentioned below at that time.)
  - 7 2. Time in milliseconds since midnight UTC. So, 39645049 would
  - 8 be a little after 6:00 AM in New York and Pittsburgh.
  - 9 3. A weighted average of the current best bid and ask prices, with
  - 10 the weights provided by the sizes at these prices. We will treat
  - 11 this as the price at this point in time. Note that this is USD/BRL
  - 12 exchange rate multiplied by 2000.
  - 13 4. The current best bid price.
  - 14 5. The size at this bid price.
  - 15 6. The current best ask price.
  - 16 7. The size at this ask price.
  - 17 8. (and beyond) The market indicators.

1 To quote:

2 This set of indicators is from a model that we are actually us-  
3 ing on a daily basis in trading. The majority of the indicators  
4 in the dataset are “Trend” indicators, that merely echo cur-  
5 rent price - some moving average of past prices. Apart from  
6 this there are some “Book” indicators that reflect the differ-  
7 ence in limit orders being shown on the buy side versus sell  
8 side. Then there are a few “Trade” indicators that look at  
9 trades that have happened in the immediate past and try to  
10 infer a momentum in that direction. People are free to make  
11 their own indicators from the price series.

- 1 See the sequence of commands below.

```
2 library(e1071)
3
4 oneday = read.table(
5   "http://www.stat.cmu.edu/~cschafer/MSCF/cprod_data_20130103v.txt")
6
7 oneday$change = as.numeric((oneday$V1 - oneday$V3) > 0)
8
9 # Divide the data into training and test sets
10
11 set.seed(0)
12
13 trainrows = sample(1:nrow(oneday), replace=F, size=10000) ] training set
14 trainset = oneday[trainrows,8:35]
15 testset = oneday[-trainrows,8:35] ] ]
16
17 svmout = tune.svm(factor(change) ~ ., data=trainset,
18   cost=c(5,10,25,50))      53,352 total obs
```

1 The cross-validation result suggests that  $C = 25$  is the best choice.

2 > summary(svmout)

3

4 Parameter tuning of svm:

5

6 - sampling method: 10-fold cross validation

7

8 - best parameters:

9 cost

10 25 

11

12 - best performance: 0.04579108

13

14 - Detailed performance results:

15 cost error dispersion

16 1 5 0.06748901 0.008798184

17 2 10 0.05399483 0.005428649

18 3 25 0.04579108 0.004018787

19 4 50 0.04629008 0.003640752

- 1 R does provide some capacity to view the results of the SVM fit, but
- 2 it is not easy to interpret. See Figure 4. Here, the decision made
- 3 by SVM is shown in two colors. This is a **slice** of the region: All
- 4 predictors except V9 and V10 are held constant (at their means, but
- 5 that can be changed).
  
- 6 The data are plotted in this two-dimensional space, regardless of
- 7 their values of the other predictors. As a result, it is difficult to make
- 8 much of the comparison between the data points and the classifica-
- 9 tion region.
  
- 10 The command to create this plot is
- 11 > plot(svmout\$best.model, trainset, V10 ~ V9)

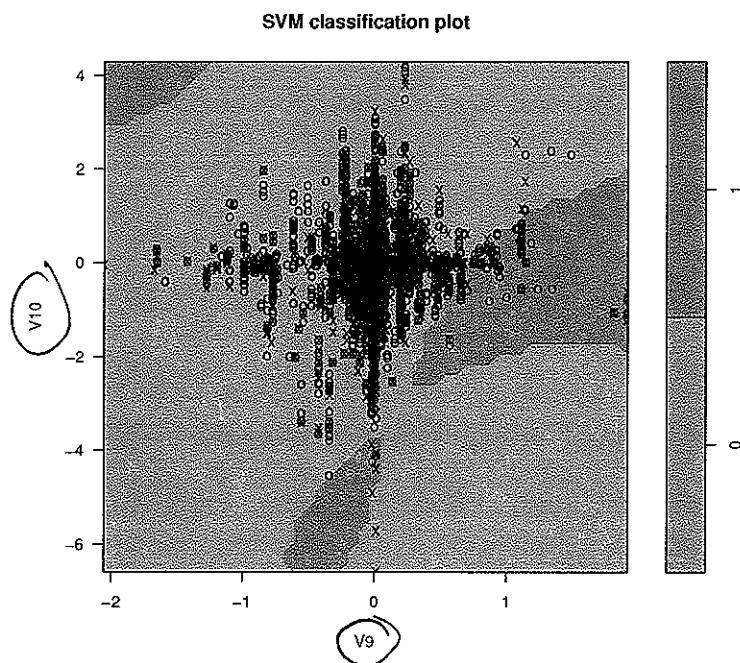


Figure 4: Plot attempting to show the SVM classification decision boundary.

## Comparing Methods

- 2 It is interesting to compare the results of this fit with other methods
- 3 we have considered. The natural candidates for comparison are lo-
- 4 gistic regression, classification trees, and random forests.
  
- 5 The same 10,000 rows of the original data set were used to train all
- 6 four models. The remaining 43,352 rows serve as a test set.

| Method              | Percent Correct |
|---------------------|-----------------|
| Logistic Regression | 0.617           |
| Classification Tree | 0.928           |
| Random Forests      | 0.973           |
| SVM                 | 0.953           |

- 1 **Exercise:** Do you think that the above classification results are of  
2 practical value? What would be a more realistic way to conduct the  
3 comparison?

4 Not of practical value. The training set  
5 was constructed by choosing random  
6 times throughout the day. Won't be  
7 able to do that in practice.

8 Use the first 10,000 as training set, then  
9 predict the rest.

10 LR 0.496

11 CT 0.547

12 RF 0.547

13 SVM 0.535

## Part 11: Clustering

- 2 Text references: Chapter 10.3 in ISL
- 3 To this point in the course, we have focused on the **supervised learning** problem, i.e. trying to predict a response  $Y$  given predictors  $\mathbf{x}$ .
- 5 Another important class of problems consist of **unsupervised learning**.
- 6 Here, the objective is to find (learn) relationships and similarities
- 7 between a collection of objects. One can view this like the setup
- 8 considered previously, except there is no response  $Y$  to predict.
- 9 **Principal Components Analysis (PCA)** is an unsupervised learning
- 10 approach: Low-dimensional structure in  $\mathbf{x}$  is “learned” by finding
- 11 linear combinations of the (high-dimensional)  $\mathbf{x}$  that captures a large
- 12 proportion of its variability.
- 13 Here, we will focus on the general problem of **clustering**, the chal-
- 14 lenge of dividing objects into  $K$  groups such that similar objects are
- 15 assigned to the same group.

<sup>1</sup> **Why Cluster?**

- <sup>2</sup> Making predictions is a very natural problem; most people can understand why it is useful to study methods for supervised learning.
- <sup>3</sup> Applications for unsupervised learning can be less obvious.
  
- <sup>4</sup> There can be distinct groups present in data sets. This grouping can provide insight into the structure of the data, and guide further analyses (in the same way that stocks in different “sectors” may be best modeled separately).
  
- <sup>5</sup> Clustering can also help in the identification of outliers.
  
- <sup>6</sup> The following examples are from finance. Note that I am not making any claims as to the legitimacy and/or accuracy of their analyses; I am just providing examples of ways in which clustering techniques could be used.

- 1 Pavlidis, et al. (2006): "Financial Forecasting through Unsupervised
  - 2 Clustering and Neural Networks," *Operational Research*.
- 
- 3 The authors use clustering to divide the predictor space into regions
  - 4 of greater homogeniety, and fit distinct (neural network) models in
  - 5 each of these.
- 
- 6 "Although global approximation methods can be applied to model
  - 7 and forecast time series . . . , it is reasonable to expect that forecasting
  - 8 accuracy can be improved if regions of the input space exhibiting
  - 9 similar dynamics are identified and subsequently a local model is
  - 10 constructed for each of them."

- 1 Lee, et al. (2010): “An Effective Clustering Approach to Stock Market
- 2 Prediction,” *PACIS 2010 Proceedings*
  
- 3 The authors use clustering to divide the financial reports into homo-
- 4 geneous groups, and then take the cluster center as a “representa-
- 5 tive.”
  
- 6 “When a financial report is released, we will transform it into a fea-
- 7 ture vector . . . [and] we assign [it] to the nearest representative fea-
- 8 ture vector. Then, we predict the direction of the stock price move-
- 9 ment according to the class label of the nearest representative feature
- 10 vector.”
  
- 11 In this framework, the clustering serves as a “smoothing” operation
- 12 on “nonstandard” data (the financial report).

| Financial report      |                      | $d_1$ | $d_2$ | $d_3$ | $d_4$ | $d_5$ | $d_6$ | $d_7$ | $d_8$ | $d_9$ |
|-----------------------|----------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Qualitative features  | efficient            | 1     | 1     | 0     | 1     | 0     | 0     | 1     | 0     | 0     |
|                       | growth               | 1     | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |
|                       | advantage            | 1     | 1     | 1     | 0     | 0     | 0     | 0     | 0     | 0     |
|                       | improvement          | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |
|                       | deficient            | 0     | 0     | 0     | 1     | 1     | 0     | 0     | 0     | 0     |
|                       | reorganize           | 0     | 0     | 0     | 1     | 1     | 0     | 0     | 1     | 1     |
|                       | difficulty           | 0     | 0     | 0     | 1     | 1     | 1     | 1     | 0     | 0     |
|                       | complaint            | 0     | 0     | 0     | 1     | 0     | 0     | 0     | 0     | 1     |
| Quantitative features | operating margin     | 0.4   | 0.38  | 0.37  | 0.1   | 0.07  | 0.08  | 0.05  | 0.06  | 0.03  |
|                       | ROE                  | 0.3   | 0.28  | 0.27  | 0.01  | 0.04  | 0.04  | 0.07  | 0.02  | 0.05  |
|                       | ROTA                 | 0.25  | 0.23  | 0.22  | 0.02  | 0.05  | 0.07  | 0.04  | 0.01  | 0.04  |
|                       | equity to capital    | 0.8   | 0.78  | 0.77  | 0.45  | 0.4   | 0.5   | 0.45  | 0.5   | 0.55  |
|                       | receivables turnover | 2.5   | 2.4   | 2.45  | 1.4   | 1.3   | 1.5   | 1.2   | 1.1   | 1.5   |
| Class label           |                      | 1     | 1     | 0     | -1    | -1    | -1    | 0     | -1    | -1    |

Table 1. An example dataset.

Figure 1: Table 1 from Lee, et al. (2010).

1 **Miceli and Susinno (2003): "Using Trees to Grow Money," *Risk***

- 2 The authors cluster hedge fund performance time series in order to  
 3 "visualize a taxonomy embedded in synchronous historical data."  
 4 They state that "[i]n a universe with low transparency from an in-  
 5 vestor's point of view, and where operating strategies are protected  
 6 from full disclosure ... instruments of classification ... would allow  
 7 the investor to verify the declared strategies in the statements issued  
 8 by hedge funds. In particular, nodes anomalous to a reference cluster  
 9 help to identify funds that require a deeper investigation."

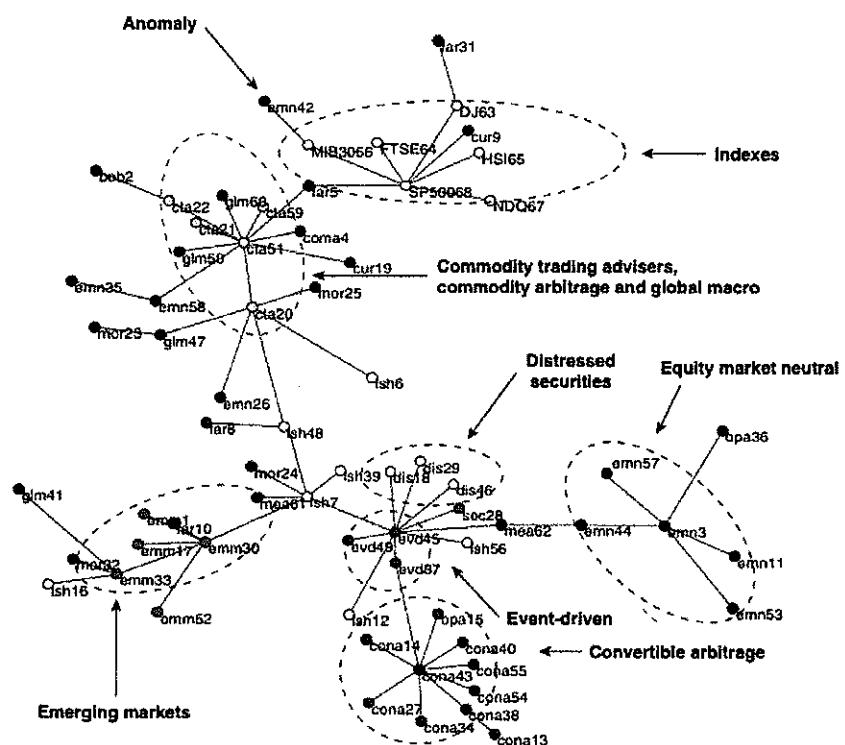


Figure 2: Figure from Miceli and Susinno (2003).

- 1 Das (2003): "Hedge Fund Classification using K-means Clustering
  - 2 Method," *9th International Conference on Computing in Economics and*
  - 3 *Finance.*
  - 4 The author proposes using clustering as a means of categorizing hedge
  - 5 funds, hence improving on the existing "myriad of classifications,
  - 6 some overlapping and some mutually exclusive."
  - 7 Craighead and Klemesrud (2002): "Stock Selection Based on Cluster
  - 8 and Outlier Analysis," Nationwide Financial, Columbus, OH.
  - 9 The authors use clustering to identify outliers that could potentially
  - 10 create problems with portfolio selection algorithms.

1

---

## 2 K-Means Clustering

- 3 The **K-means** clustering algorithm is a relatively simple and intuitive  
4 approach to dividing the sample  $x_1, x_2, \dots, x_n$  into  $K$  distinct groups.
- 5 Despite repeated extensions and enhancements of the method and  
6 the growth of other clustering methods, K-means remains popular  
7 and useful. At a minimum, it is important to understand K-means  
8 prior to exploring other methods.
- 9 The algorithm is based on a very natural measure of within-cluster  
10 heterogeneity. Observations are assigned to clusters in an effort to  
11 minimize this measure.
- 12 It is assumed that  $K$  is fixed by the user; the choice of  $K$  is a tricky  
13 issue.
- 14 I will adopt the notation used by ISL, Section 10.3.1.

- 1 First, define  $C_k$  to be the set of indices that belong to cluster  $k$ , for
- 2  $k = 1, 2, \dots, K$ .
- 3 Each observation is assigned to exactly one cluster. In other words,
- 4  $C_k \subseteq \{1, 2, \dots, n\}$  with

- 5 
$$\bigcup_{k=1}^K C_k = \{1, 2, \dots, n\} \text{ and } C_k, C_\ell \text{ disjoint for } k \neq \ell.$$

- 6 With K-means, the <sup>heterogeneity</sup> similarity within cluster  $k$  is measured by

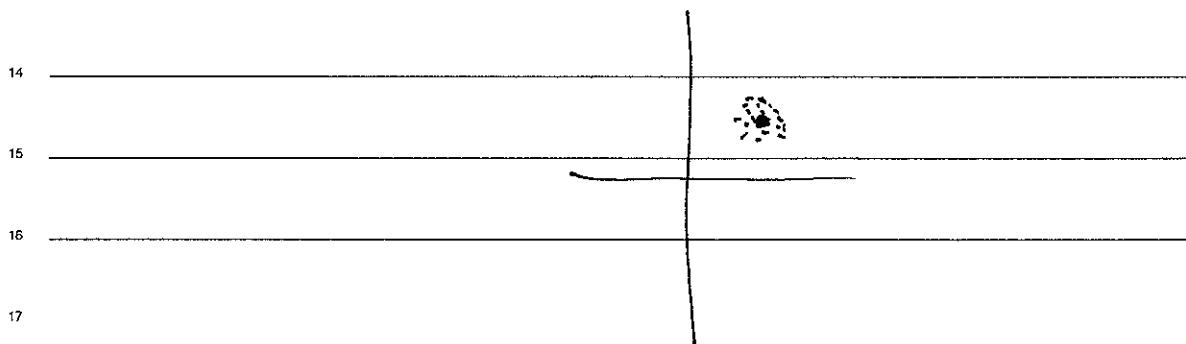
- 7 
$$W(C_k) = \sum_{i \in C_k} \sum_{j=1}^p (x_{ij} - \bar{x}_{kj})^2$$

- 8 where  $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip})$  and  $\bar{\mathbf{x}}_k$  is the cluster **centroid**, calculated
- 9 as the sample mean of the cluster members.

- 10 **Exercise:** Under what circumstance(s) will  $W(C_k)$  be small?

11 Small if all of  $\mathbf{x}_i$  in cluster  $k$

12 are close to the centroid, ie. they  
13 are close to one another



- <sup>1</sup> K-means clustering seeks to find the allocation of the observations
- <sup>2</sup> that minimizes

$$\sum_{k=1}^K W(C_k)$$

- <sup>4</sup> Comparing every possible allocation is not realistic. (Note that there
- <sup>5</sup> is no restriction on the sizes of the clusters.)
- <sup>6</sup> Instead, the following algorithm is utilized. (Algorithm 10.1 in ISL.)

- 
- <sup>8</sup> 1. Randomly assign a number from 1 to  $K$  to each fo the observa-  
<sup>9</sup> tions. These are the initial cluster assignments.
  - <sup>10</sup> 2. Repeat until no changes are made to cluster assignments:
    - <sup>11</sup> (a) Find the centroid for each of the  $K$  clusters.
    - <sup>12</sup> (b) Re-assign each observation to the cluster whose centroid is  
<sup>13</sup> closest as measured by Euclidean distance.

---

- 1 Note that the criterion is guaranteed to not increase with each itera-
  - 2 tion of the algorithm, but the search could get caught in a local min-
  - 3 imum (and not the global minimum).
  
  - 4 To address this problem, it is standard to repeat the algorithm several
  - 5 times with different (randomly chosen) starting allocations. If you
  - 6 keep track of the criterion achieved from each repetition, you can
  - 7 easily determine which allocation is the best.

## 1 K-means in R

- <sup>2</sup> K-means is implemented in R using the function `kmeans()`.

- 3 The syntax is simple:

```
4 > kmout = kmeans(datamat, centers = 5, nstart = 10)
```

- 5 Here I have specified that I want  $K = 5$  clusters, and that I want the  
6 basic algorithm repeated ten times in order to increase the chances of  
7 finding the global minimum.

- 8 Note that `datamat` must be formatted so that each of the vectors to
  - 9 be clustered are stored in its **rows**. Use the transpose function `t()`,
  - 10 if needed.

## Example: Clustering Stock Time Series

- 2 In this example, we will cluster the time series of 70 randomly-chosen
- 3 NYSE stocks, with data taken from December 1, 2014 to January 31,
- 4 2015. This is a total of 42 trading days.

- 5 The data can be read in using the command

```
42 by 70
6 stockmat = read.table(
7   "http://www.stat.cmu.edu/~cschafer/MSCF/stockdata.txt",
8   header=T)
```

Note time series stored as  
columns

- 9 Note that the data are formatted such that each column is a different
- 10 stock; the rows are the different days. Also, each time series has been
- 11 scaled so that it has a mean of zero and variance of one. It would be
- 12 difficult to compare the series without this transformation.

- 13 The column names give the different ticker symbols.

- 1 The call to kmeans () is below. Note that I had to transpose stockmat
- 2 to get it into the proper format.

3 `kmout = kmeans(t(stockmat), centers = 6, nstart = 10)`

---

- 4 Some comments on the output of kmeans ():

5 1. The rows of kmout\$centers holds the <sup>Centroids</sup> ~~centers~~ of the  $K$  clusters.

6 In our example, each center is a vector of length 42.

7 2. kmout\$cluster specifies the cluster that each of the observa-

8 tions has been assigned to.

9 3. The value of  $W(C_k)$  for each of the clusters can be found from

10 kmout\$withinss.

11 4. The size of each cluster is found from kmout\$size.

12 Plots showing the six cluster centers, along with the members, are

13 shown on the following pages.

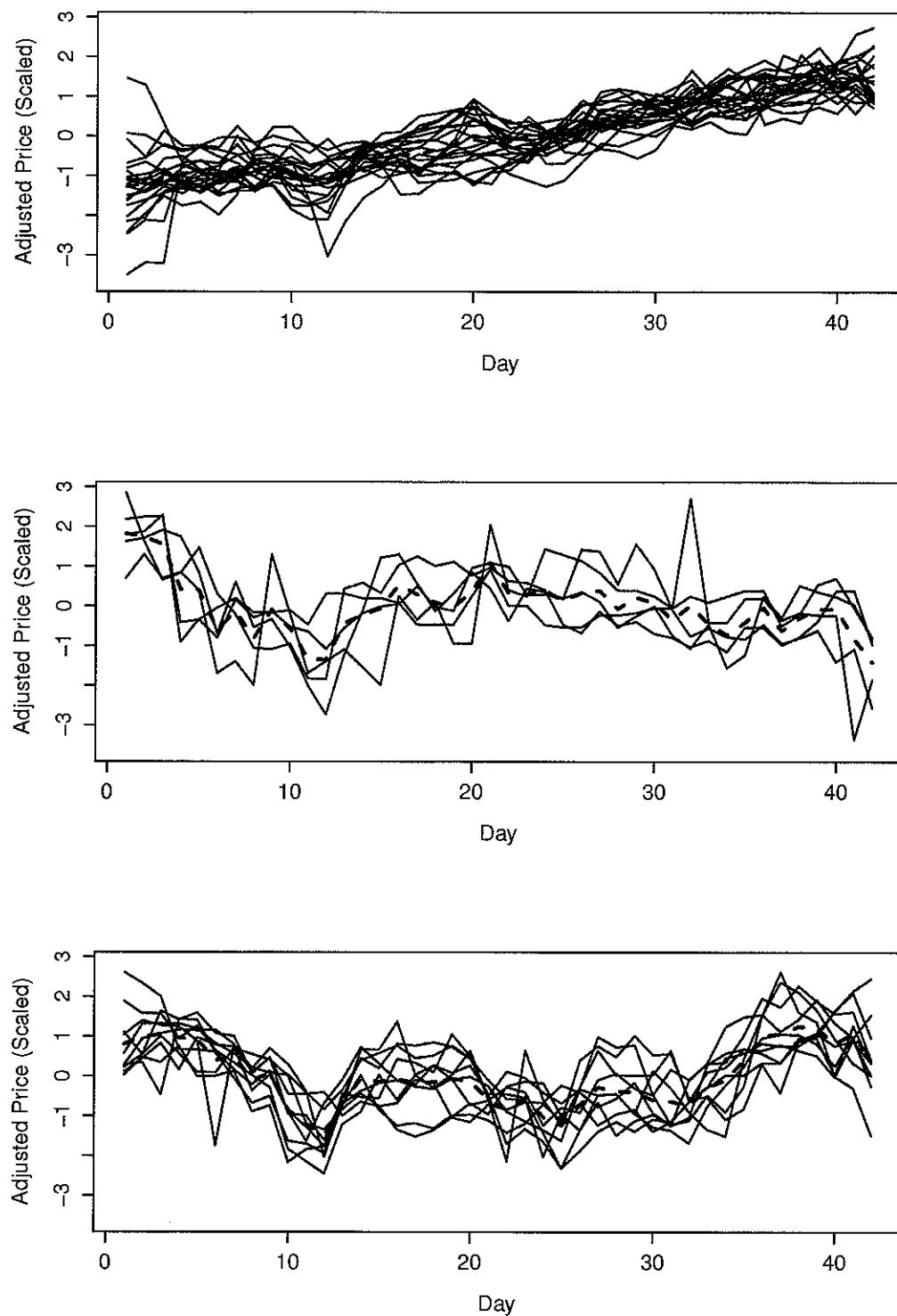


Figure 3: The first three clusters. Centers are shown as dashed, red lines.

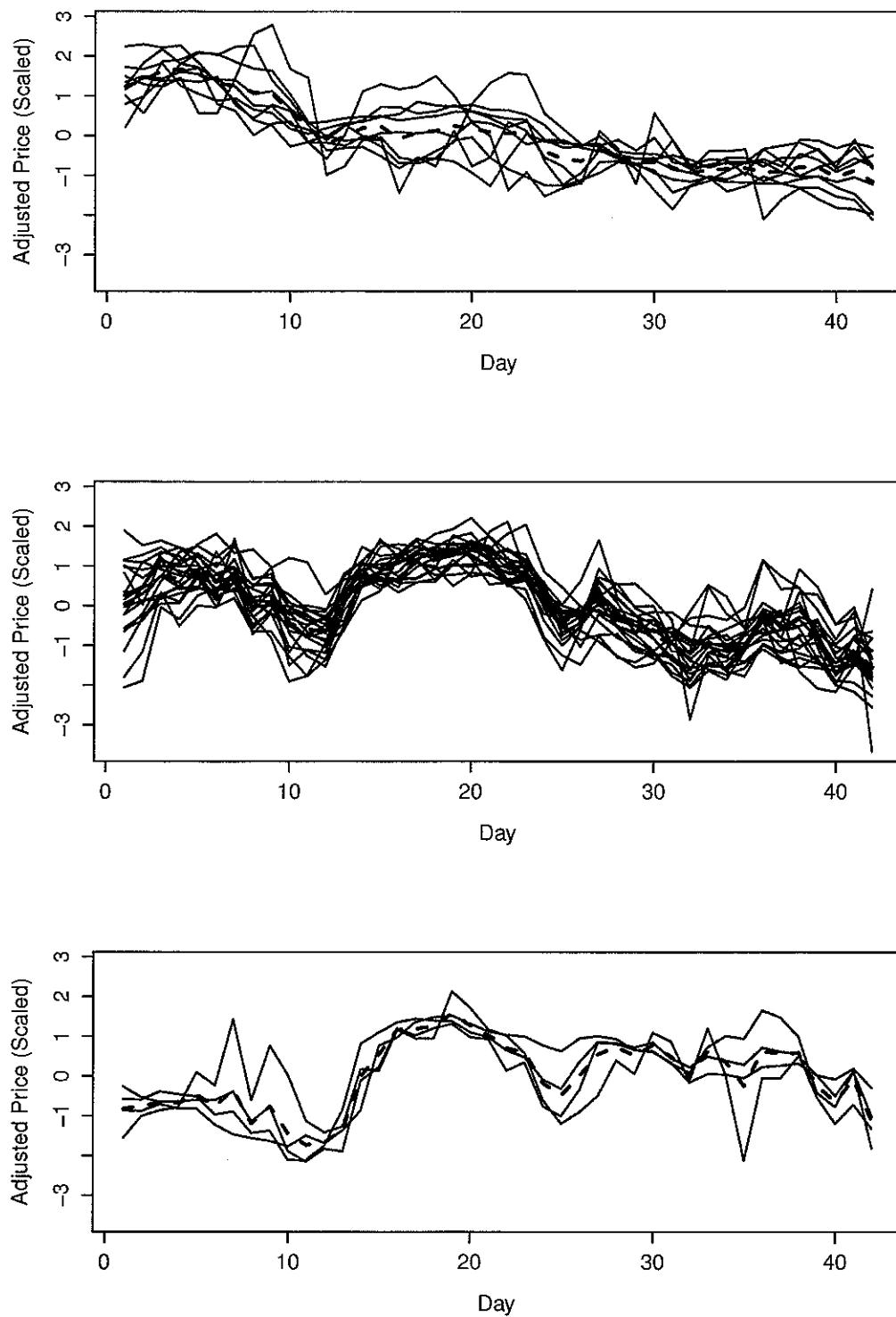


Figure 4: The second three clusters. Centers are shown as dashed, red lines.

- 1 **Exercise:** What is a way that the cluster information could be incorporated into a regression model that attempts to use these stock time series as predictors?

4 Create a factor which equals the cluster

5 membership, ie.  $X_i = 1, 2, 3, 4, 5, 6$

6

7 Include in my model interactions between

8  $X_i$  and <sup>some</sup> of the other predictors.

9

10 The model will fit a separate slope

11 for each of the 6 clusters

12

13 Or: Could fit a separate model in each of the

14 6 clusters

15

16

17

18

19

1

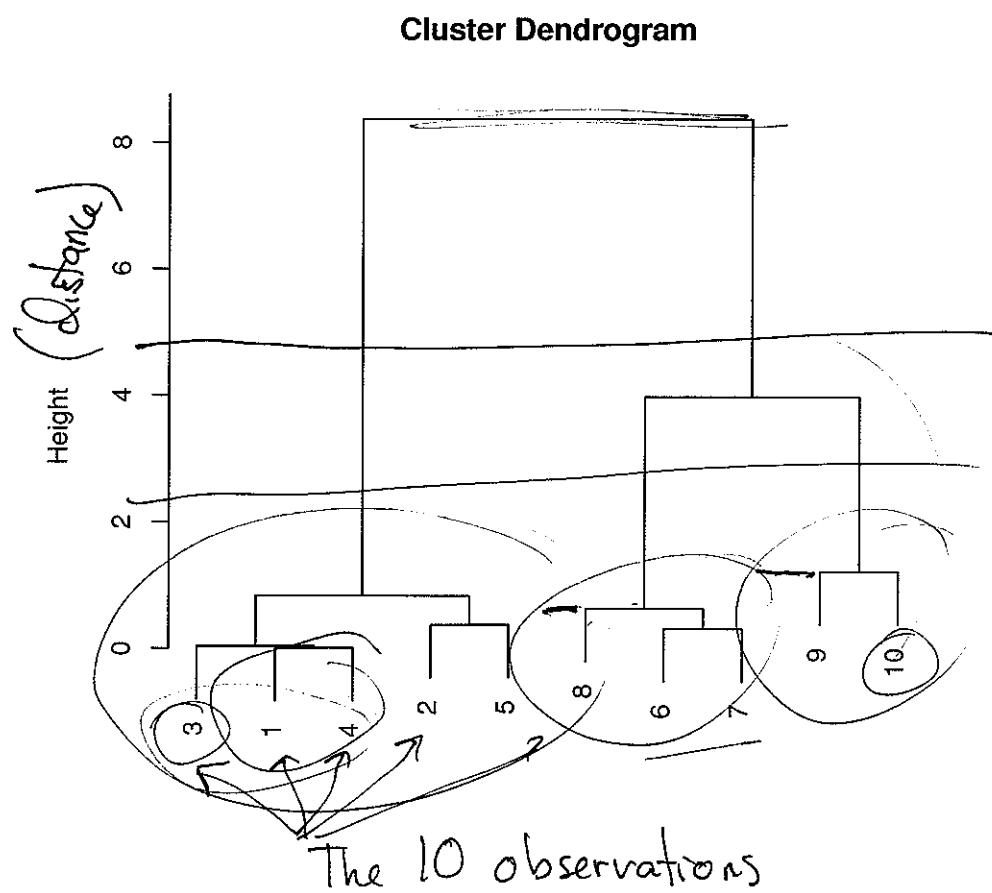
---

## 2 Hierarchical Clustering

- 3 Another class of clustering procedures, called **hierarchical clustering**, are characterized by the distinctive **dendograms** they create to
- 4 depict the relationships between observations.
  
- 6 An appealing feature of these methods is that the creation of the dendrogram does not require the specification of the number of clusters.
- 7
- 8 Sometimes, the dendrogram may reveal an “obvious” division among
- 9 the observations, and hence suggest the number of clusters needed.
  
- 10 This clustering method allows the user to specify a measure of dis-
- 11 similarity. Often, Euclidean distance is used (just as in K-means).



- 1 It is useful to begin by understanding how to interpret the dendrogram. An example is given below.
- 2
- 3 Observations which are similar are **linked** at a low **height** in the dendrogram. For instance, to link observations 6 and 7, you need only go up to a height of approximately 0.5. These two observations are quite similar. But, in order to link observations 2 and 10, you need to go to a height of over 8. These are quite dissimilar.
- 4
- 5
- 6
- 7
- 8 The units for “height” match the units of the dissimilarity metric that
- 9 the user provides.



10

- <sup>1</sup> **Exercise:** Which pair of observations are more similar: 8 and 6, or 9 and 10?

<sup>3</sup> 8 and 6 are more similar than  
<sup>4</sup> 9 and 10. Don't be fooled by fact  
<sup>5</sup> that 9 and 10 share a "branch"

- <sup>9</sup> Clusters can now be formed by "cutting" the dendrogram at a user-chosen height. It may seem "clear" by looking at the dendrogram that cutting at around a height of 3 would be a good choice. The results is three clusters.

- <sup>13</sup> Note that in this case I generated the ten observations from the normal distribution with a standard deviation of 0.5. The mean for observations 1 thru 5 was 0, for observations 6 thru 8 was 5, and for observations 9 and 10 was 7. This explains the structure in the dendrogram.

## Building the Dendrogram

- 2 The process of constructing the dendrogram is quite intuitive.
- 3 First, the two observations which are the most similar (as measured
- 4 by the chosen dissimilarity metric) and joined together, at a height
- 5 equal to their dissimilarity. In our example above, observations 1
- 6 and 4 were the first to be joined, at a height very close to zero.
- 7 This process then continues, with the most similar objects joined to-
- 8 gether at the appropriate height. **But:** Once observations are joined,
- 9 they are **treated as a unit** from that point on. For instance, in the sec-
- 10 ond step in the construction of the dendrogram above, observation 3
- 11 was joined with the **unit consisting of 1 and 4**.
- 12 This process continues until all observations are joined, i.e., there are
- 13 no disconnected observations in the dendrogram.

- 1 The obvious question is thus: **How is the dissimilarity of collections**
  - 2 **of observations calculated?** There are a few different approaches.
  - 3 Two of the standard ones are as follows:
  - 4 With **complete linkage**, the distance between sets of observations,  $S_i$
  - 5 and  $S_j$ , is defined to be the **maximal** dissimilarity between all pairs
  - 6  $x \in S_i$  and  $y \in S_j$ .

↑

  - 7 With **single linkage**, the distance between sets of observations,  $S_i$
  - 8 and  $S_j$ , is defined to be the **minimal** dissimilarity between all pairs
  - 9  $x \in S_i$  and  $y \in S_j$ .  
  - 10 Less-commonly used options are **centroid linkage**, in which the dis-
  - 11 tance between  $S_i$  and  $S_j$  is calculated as the dissimilarity between the
  - 12 centroids of the two groups, and **average linkage**, calculated as the
  - 13 average distance between all pairs of elements in  $S_i$  and  $S_j$ .  
  - 14 Since there are not really “correct” clusters, one typically chooses the
  - 15 approach that yields the most useful results.

## 1. Hierarchical Clustering in R

- 2 The function `hclust()` performs hierarchical clustering in R.
- 3 Remember that this method is built around a user-chosen measure
- 4 of dissimilarity. Hence, the primary input argument to `hclust()` is
- 5 a symmetric  $n$  by  $n$  **matrix** where the  $(i, j)$  entry provides the dissim-
- 6 ilarity between  $\mathbf{x}_i$  and  $\mathbf{x}_j$ . The diagonal of this matrix should consist
- 7 of zeros.
- 8 R has a function `dist()` which calculates the Euclidean distance be-
- 9 tween the rows of the provided object. For instance,

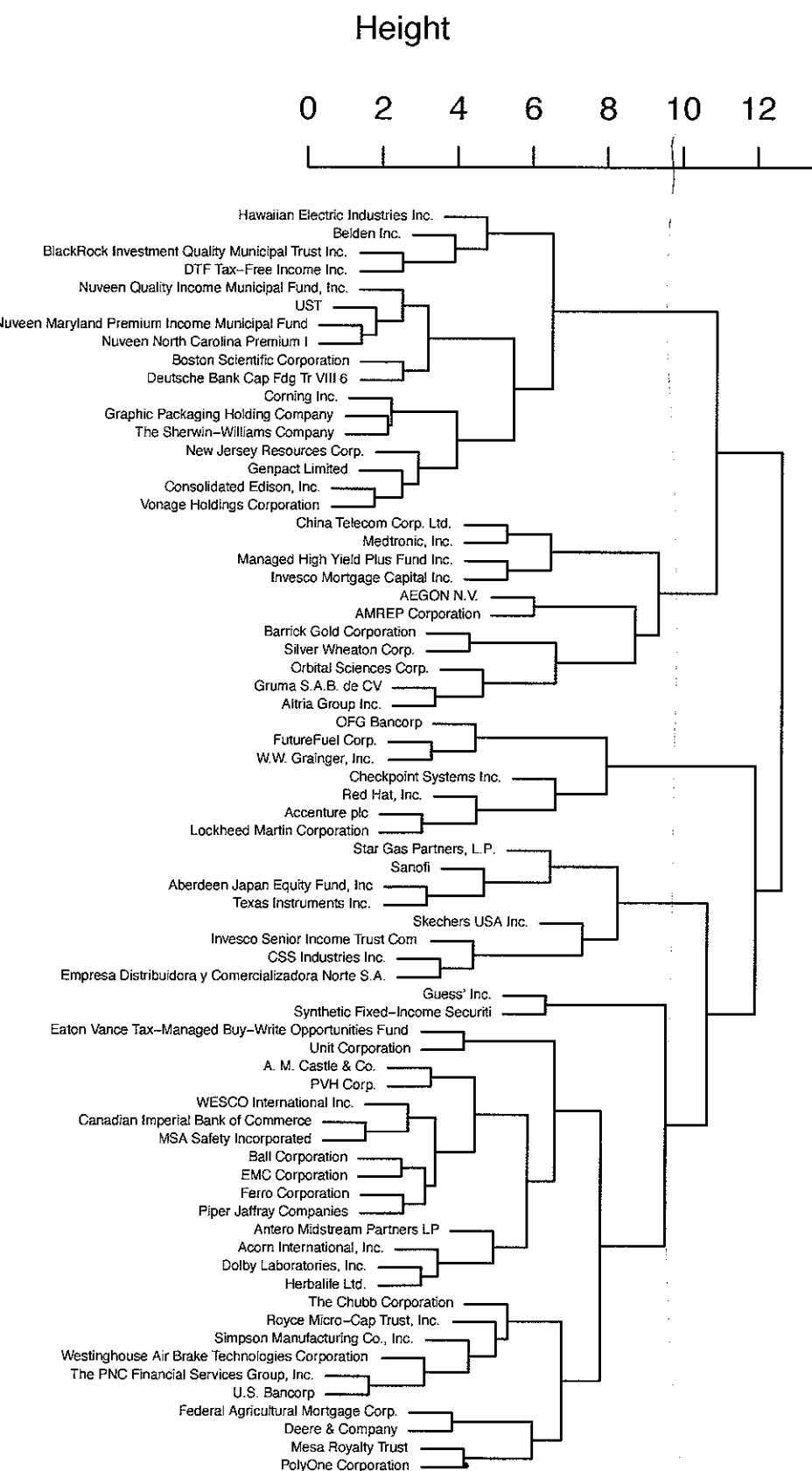
10 > `dist(t(stockmat))`

11 will calculate the Euclidean distance between each of the time series  
12 stored as the columns of `stockmat`.

*n x n matrix*

- 1 The syntax for complete linkage clustering is as follows:
- 2 > `hcout = hclust(dist(t(stockmat)), method="complete")`
- 3 Other choices for method include "single", "centroid", and
- 4 "average".
- 5 The `plot()` function, when applied to the output of `hclust()`, pro-
- 6 duces the dendrogram:
- 7 > `plot(hcout, labels=fullname, cex=0.45,`
- 8       `sub="", xlab=""`)
- 9 Here, I have created `fullname` to consist of the full names corre-
- 10 sponding to each of the 70 ticker symbols in our data set. The result
- 11 is shown on the following page.

## Cluster Dendrogram



- 1 Another useful function is `cutree()`. This will create return cluster
- 2 membership for each of the observation, based on a desired number
- 3 of clusters, or on a desired height.
- 4 For instance, to cut our dendrogram into six clusters, we specify `k=6`:

```
5 > cutree(hcout, k=6)
6 ABX ACN AEB AGM AM ATV AXR BDC BKN BLL BSX CAS CB CHA ...
7 1 2 1 3 3 1 4 4 3 4 3 3 1 ...
```

- 8 And to cut our our dendrogram at a height of 10, specify `h=10`:

```
9 > cutree(hcout, h=10)
10 ABX ACN AEB AGM AM ATV AXR BDC BKN BLL BSX CAS CB CHA ...
11 1 2 1 3 3 1 4 4 3 4 3 3 1 ...
```