Statistical Learning (Optional)

(Jing Li, Miami University)

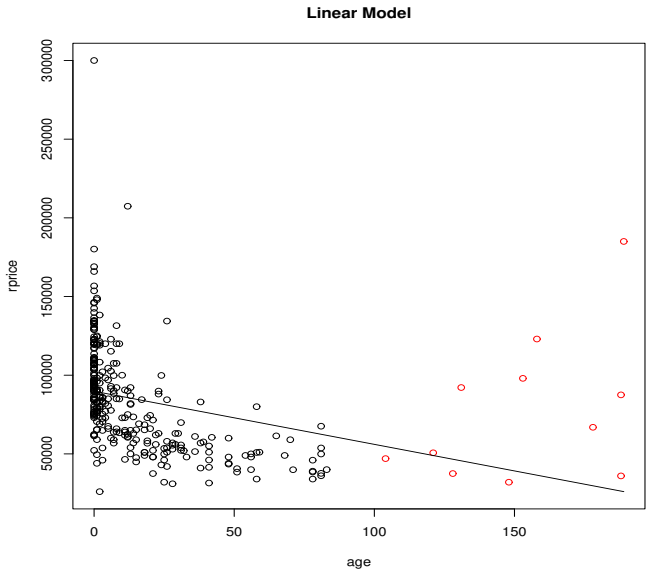
1. This note discusses the basic idea and terminology of Statistical Learning. To keep it simple, let us look into the problem of predicting house price using just one predictor, age of house[1]. Statistical learning is employed to choose the best function form, i.e., to determine how the predictor and price are related. The research question is, is the relationship linear or nonlinear? If nonlinear, how to model the nonlinearity? Can taking nonlinearity into account lead to better forecast?

Candidate Models

2. The starting point is a linear bivariate model

$$rprice = \beta_0 + \beta_1 age + u, \qquad (Linear\ Model) \tag{1}$$

based on which a downward-sloping OLS fitted line is shown below



It is obvious that this linear model does a bad job of predicting those old houses with age greater than 100 (highlighted in red color) —the model overpredicts 3 of them, but underpredicts 8. The uneven split of 3 vs 8 indicates systematic error, or the model

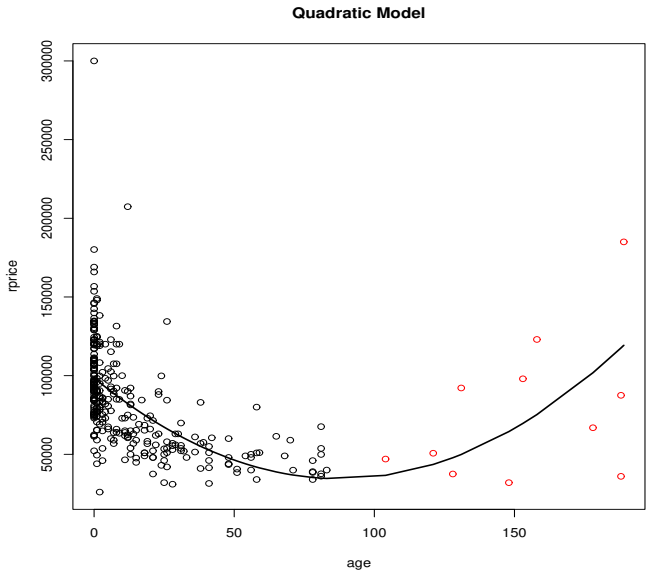
[1]House data is at `http://www.fsb.miamioh.edu/lij14/400_house.txt`

fails to pass the center of data. A better model should lead to more balanced prediction error.

3. To accomodate those 11 old houses, we hope to see a curve that moves upward for them. Toward that end, we add a squared term and try a quadratic model

$$rprice = \beta_0 + \beta_1 age + \beta_2 age^2 + u, \qquad (Quadratic\ Model) \tag{2}$$

We see three signals of improvement: (i) the t value of β_2 10.61565 exceeds 1.96, so the squared term is significant (thus the linear model is biased by omitting the squared term); (ii) the fitted curve passes the center of those old houses, and now the split is 5 below vs 6 above; (iii) R squared increases from 0.1101308 to 0.3429681



R codes are below

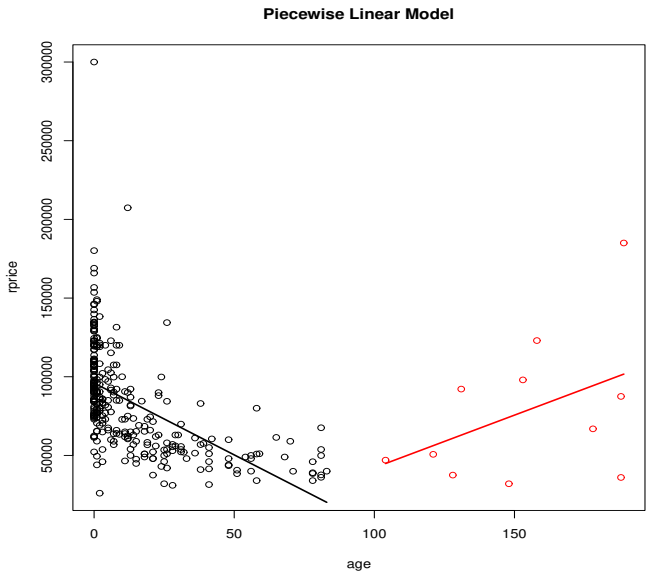
```
> ad = "https://www.fsb.miamioh.edu/lij14/400_house.txt"
> da = read.table(url(ad), header=T)
> m0 = lm(rprice~age,data=da)
> colors = ifelse(da$age > 100, "red", "black")
> plot(da$age,da$rprice,main="Linear Model",col=colors)
> lines(sort(da$age),predict(m0)[order(da$age)])

> da$agesq = da$age^2
> m1 = lm(rprice~age+agesq,data=da)
```

```
> summary(m1)$coef
                Estimate   Std. Error   t value      Pr(>|t|)
(Intercept) 98512.366586 1904.1852095  51.73466 6.691655e-157
age         -1457.472891  115.1854756 -12.65327  5.332914e-30
agesq           8.292846    0.7811906  10.61565  9.798541e-23
> plot(da$age,da$rprice,main="Quadratic Model",col=colors)
> lines(sort(da$age),predict(m1)[order(da$age)],lwd=2)
> summary(m0)$r.squared
[1] 0.1101308
> summary(m1)$r.squared
[1] 0.3429681
```

4. Alternatively, the presence of a turning point in the previous graph suggests a piecewise linear model—two linear models are fitted separately, one for houses younger than 100, and one for houses older than 100.

$$\begin{cases} rprice = \beta_0 + \beta_1 age + u & \text{if age} < 100 \\ rprice = \alpha_0 + \alpha_1 age + u & \text{if age} > 100 \end{cases} \quad (Piecewise\ Linear\ Model) \qquad (3)$$

where 100 is cutoff for two regimes. In the literature this is also called a model with structural break, or a model with unstable parameters. Graph below shows the two fitted lines.

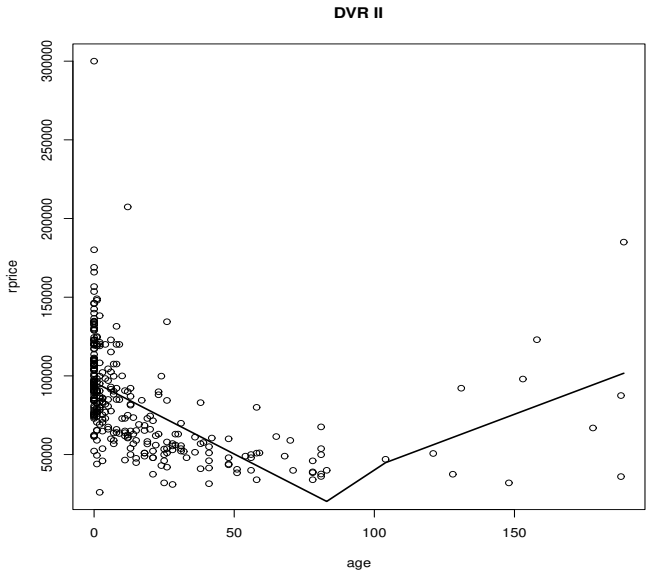


The piecewise linear model is the same as the dummy variable regression II we cover

in eco 311 class

$$rprice = \beta_0 + \beta_1 age + \beta_2 D + \beta_3 D \times age + u \qquad (DVR\ II) \tag{4}$$

where the dummy variable $D = 1$ for houses older than 100, and $D \times age$ is interaction term. In light of R-squared, this model produces worse in-sample fit than the quadratic model




```
> plot(age,rprice, main="Piecewise Linear Model",col=colors)
> m2a = lm(rprice[age<100]~age[age<100],data=da)
> lines(sort(age[age<100]),predict(m2a)[order(age[age<100])],lwd=2)
> m2b = lm(rprice[age>=100]~age[age>=100],data=da)
> lines(sort(age[age>=100]),predict(m2b)[order(age[age>=100])],lwd=2,col="red")

> da$du = as.numeric(da$age>=100)
> da$i = da$du*da$age
> m3 = lm(rprice~age+du+i,data=da)
> plot(age,rprice,main="DVR II")
> lines(sort(age),predict(m3)[order(age)], lwd=2)
> summary(m3)$r.squared
[1] 0.3022737
```

Using graphs to compare models is straightforward, but not rigorous. A formal approach is to compare the forecasting power of each model.

Forecast

5. Suppose we want to predict the price for a 48-year old house based on the quadratic model. Mathematically, the predicted price can be computed as

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x + \hat{\beta}_2 x^2 \tag{5}$$

where x is age and $x = 48$.

6. There are at least two ways to do that in R. First, we may just apply the formula (5)

```
> m1 = lm(rprice~age+agesq,data=da)
> coef(m1)[1]+coef(m1)[2]*48+coef(m1)[3]*48^2
(Intercept)
   47660.38
```

or we can use the predict function

```
> nda = data.frame(age=48,agesq=48^2)
> predict(m1,newdata=nda)
       1
47660.38
```

How about a 40-year old house?

```
> nda = data.frame(age=40,agesq=40^2)
> predict(m1,newdata=nda)
    1
53482
```

The key is to specify the observation that you want to predict using newdata option in the predict function.

7. Notice that the 48-year-old house is the first house in sample, but the 40-year-old house is out of sample. In terms of evaluating forecasting power, the 40-year-old house is more useful than the 48-year-old house because the model is estimated without it. However,

we do not know the actual price of the 40-year-old house, so we do not know whether the prediction 53482 is good or bad. By contrast, we know the actual price of the 48-year-old house, so we are able to compute its prediction error as $60000 - 47660.38$

```
> data[1,]
  age area baths rprice agesq
1  48 1660     1  60000  2304
> 48 %in% unique(age)
[1] TRUE
> 40 %in% unique(age)
[1] FALSE
```

Pseudo out-of-sample forecast

8. The last discussion suggests that we may randomly divide our sample into two non-overlapping subsets—the first subset is called training set, which is used to estimate or train the model. The second subset is testing set excluded in estimation. The purpose of testing set is evaluating the forecasting performance, as their actual y value will be compared to their predicted value obtained from the trained model. We refer to this approach of forecasting assessment as pseudo out-of-sample forecast.

9. For instance, codes below consider a testing set of 50 observations, and randomly select a training set to train the quadratic model and DVR II model[2]. The quadratic model is shown to generate superior forecasts because it has smaller Root Mean Squared Prediction Error (RMSPE) given as

$$RMSPE = \sqrt{\frac{\sum_{i=1}^{50}(y_i - \hat{y}_i^2)}{50}} \tag{6}$$

```
> set.seed(1234)
> test.set=50
> test.ind = sample(1:nrow(da),test.set)
> da.train = da[-test.ind,]
>
> m1t = lm(rprice~age+agesq,data=da.train)
```

[2]Ideally, each time the cutoff is not necessarily 100 for DVR II, and needs to be re-estimated.

```
> yhat_m1t = predict(m1t,newdata=da[test.ind,])
> RMSPE_m1 = sqrt(mean((da[test.ind,c("rprice")]-yhat_m1t)^2))
> RMSPE_m1
[1] 24263.98
>
> m3t = lm(rprice~age+du+i,data=da.train)
> yhat_m3t = predict(m3t,newdata=da[test.ind,])
> RMSPE_m3 = sqrt(mean((da[test.ind,c("rprice")]-yhat_m3t)^2))
> RMSPE_m3
[1] 24746.65
```

10. So far the comparison is based on just one testing set. To guard against sampling uncertainty, codes below make use of 20 testing sets, and show the average difference in RMSPE between two models

```
> set.seed(1234)
> test.set=50
> RMSPE_diff = NULL
> for (j in 1:20) {
+     test.ind = sample(1:nrow(da),test.set)
+     da.train = da[-test.ind,]
+     m1t = lm(rprice~age+agesq,data=da.train)
+     yhat_m1t = predict(m1t,newdata=da[test.ind,])
+     RMSPE_m1 = sqrt(mean((da[test.ind,c("rprice")]-yhat_m1t)^2))
+     m3t = lm(rprice~age+du+i,data=da.train)
+     yhat_m3t = predict(m3t,newdata=da[test.ind,])
+     RMSPE_m3 = sqrt(mean((da[test.ind,c("rprice")]-yhat_m3t)^2))
+     RMSPE_diff[j] = RMSPE_m1-RMSPE_m3
+ }
> RMSPE_diff
 [1]  -482.6628  -303.2989  -844.0142 -1336.8175  -178.5060  -798.7792 -1398.8123
 [9] -2035.2114  -101.6011  -942.2428 -1199.2838  1015.5549  -871.8921  -730.3075 -
[17]  -634.2383  -865.8733  -516.2100  -983.1747
> mean(RMSPE_diff)
[1] -846.6128
```

```
> 846.6128/mean(rprice)
[1] 0.01011227
```

We see that 19 of 20 testing sets show the quadratic model is better than DVR II. On average, the prediction error of quadratic model is less than DVR II by 846.6128. To put that number into perspective, it is about 1 percent of average house price.

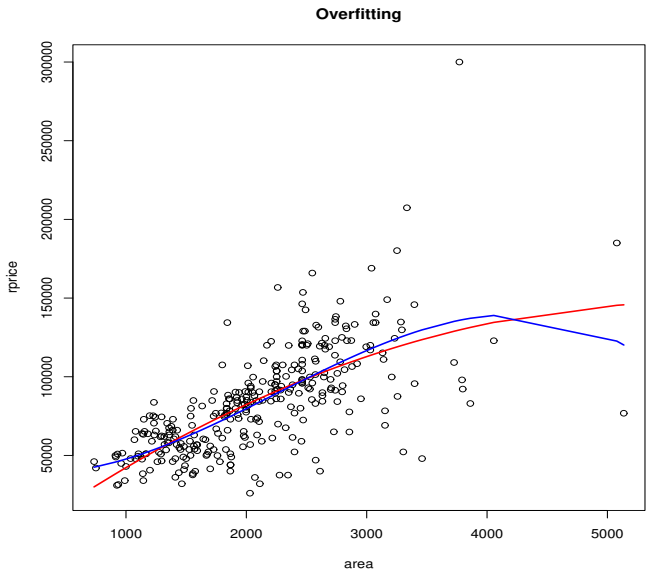
11. Based on the tool of pseudo out-of-sample-forecast, statistical learning concludes that the relationship between age and rprice is more likely to be quadratic as opposed to piecewise linear.

Overfitting and Bias

12. Everything equal, we prefer a parsimonious or simple model. The linear model is the simplest one, though it suffers from the bias of omitting the squared term.

13. We acquire better forecasting by augmenting the linear model with squared term. Then we wonder, can we do even better by adding the cubic term (the third-order polynomial)?

14. This time, we use area as the predictor, and re-do the pseudo out-of-sample forecast exercise. But first, the graph below compares the quadratic model (red color) and cubic model (blue color) when they are applied to the whole sample



We see an outlier near the top of graph, which is a house with price about 300K. The

blue line seems to be influenced by this outlier more than the red line. That is the first hint of overfitting—the cubic model is excessively complicated thanks to some extreme values. Put differently, a complex model tends to captures noise other than signal.

15. (Exercise) Use R to draw the graph above

16. Rather than in-sample-fit, the pseudo out-of-sample forecast emphasizes out-of-sample performance. The results are

```
> da$x = da$area
> da$xsq = da$x^2
> da$xcu = da$x^3
> set.seed(1234)
> test.set=50
> RMSPE_diff = NULL
> for (j in 1:20) {
+     test.ind = sample(1:nrow(da),test.set)
+     da.train = da[-test.ind,]
+     m1t = lm(rprice~x+xsq,data=da.train)
+     yhat_m1t = predict(m1t,newdata=da[test.ind,])
+     RMSPE_m1 = sqrt(mean((da[test.ind,c("rprice")]-yhat_m1t)^2))
+     m3t = lm(rprice~x+xsq+xcu,data=da.train)
+     yhat_m3t = predict(m3t,newdata=da[test.ind,])
+     RMSPE_m3 = sqrt(mean((da[test.ind,c("rprice")]-yhat_m3t)^2))
+     RMSPE_diff[j] = RMSPE_m1-RMSPE_m3
+ }
> RMSPE_diff
 [1]  -633.61887   267.46274  -550.76477  1066.46575 -1349.70990 -3948.34988   495.
 [9]   587.68973   418.17958  -402.04889  -998.56882  -821.43225   504.29137   231.
[17]   624.02395   738.94093  -743.53221    49.84382
> mean(RMSPE_diff)
[1] -452.1836
```

We see that on average the cubic model gives rise to worse forecast relative to the quadratic model. This is another hint that the cubic model suffers an overfitting issue.

17. Based on the tool of pseudo out-of-sample-forecast, statistical learning concludes that the relationship between area and rprice is more likely to be quadratic than cubic.

18. (Exercise) How about quadratic vs linear for the relationship between area and rprice?

Hyper-parameter

19. In general, consider a p-th order polynomial model

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + ... + \beta_p x^p + u \tag{7}$$

or a piece-wise model with k regimes

$$\begin{cases} y = \beta_{10} + \beta_{21} x + u & \text{in regime 1} \\ ... = ... \\ y = \beta_{k0} + \beta_{k1} x + u & \text{in regime k} \end{cases} \tag{8}$$

For given p and k, the unknown parameters are those betas. Nevertheless, p and k in practice are unknown as well, and they are called hyper-parameter. Statistical learning can be used to determine those hyper-parameters. For instance, choosing $p = 1$ vs $p = 2$ is essentially picking linear vs quadratic model.

Tree and Random Forest

20. An increasingly popular model in statistical learning is tree model, which replaces the linear function in the piece-wise linear model with step function of conditional mean

$$\begin{cases} y = E(y|x \in regime\ 1) + u & \text{in regime 1} \\ ... = ... \\ y = E(y|x \in regime\ k) + u & \text{in regime k} \end{cases} \tag{9}$$

where each regime is a branch of a tree. We have regression tree if y is numeric, and classification tree if y is qualitative such as a binary variable. The tree is building block for Random Forest.