0.1 Multiple Linear Regression

- Simple linear regression is a useful approach when you have a single predictor variable.
  - In `Advertising`, we looked at the relationship between sales and TV advertising.
  - How to extend analysis to radio and newspaper?
- One approach is to run separate simple linear regressions.
  - For `Advertising`, this approach will lead us to believe all three forms of advertising are associated with increased sales.
  - How to combine the results?
  - Can be misleading when predictors are correlated, as we will see with `Advertising`.

- A better approach is to extend the linear regression model to accommodate multiple predictors:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \varepsilon.$$

  – We interprete β_j as the average effect on Y of one unit increase in X_j, *holding all other predictors fixed*.

0.2 Estimating the Coefficients

- As with the simple linear regression setting, we estimate the unknown regression coefficients $\beta_0, \beta_1, \ldots, \beta_p$ using our training data to get $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$, and make predictions

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \ldots + \hat{\beta}_p x_p.$$

  - This defines a line when $p = 1$ in the simple linear regression case.
  - This defines a plane when $p = 2$ as we saw the `Income` example.
  - For larger p, this defines a p-dimensional plane, and is difficult to visualize.

- To get the 'closest' fit, we again use the same least squares approach. We choose $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ that minimizes the residual sum of squares

$$\begin{aligned} \text{RSS} &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - \ldots - \hat{\beta}_p x_{ip})^2. \end{aligned}$$

- The minimizers are called the multiple least squares regression coefficient estimates.
  - Their explicit form is difficult to write down without the use of matrix algebra, due to a matrix inversion.
  - For the same reason, the computation cost is of the order of a matrix inversion, which can be done very quickly for not too large p.

- For `Advertising`, the R output for the full model is given here:

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.938889   0.311908   9.422   <2e-16 ***
TV           0.045765   0.001395  32.809   <2e-16 ***
radio        0.188530   0.008611  21.893   <2e-16 ***
newspaper   -0.001037   0.005871  -0.177     0.86
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.686 on 196 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8956
F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

  - Remember the interpretation for a coefficient now is the change in response associated with the corresponding predictor *holding the other predictors fixed*.

- For example, for a given amount of TV and newspaper advertising, spending an additional \$1,000 on radio advertising leads to an increase in sales by approximately 189 units.

- What happened to `newspaper`?

  - In the simple linear regression setting (not shown), `newspaper`'s coeffficient estimate was significantly non-zero.
  - The difference is that, in the simple linear regression setting, the coefficient estimate ignores the other predictors.
  - In this case, `newspaper` and `radio` are correlated.
  - `radio` drives `sales` so high `radio` is associated with high `sales`.
  - Because of the correlation, this also means high `newspaper`.
  - So absent data on `radio`, `newspaper` becomes a surrogate for `radio`, getting "credit" for `radio`'s effect on `sales`.

- A more absurd example: sharks and ice cream.
  - Running a regression of shark attacks on ice cream sales shows a positive relationship.
  - Ban ice cream to reduce shark attacks?
  - More sensible reason: high temperatures are associated with both.
  - Multiple regression of shark attacks including temperature will reveal ice cream sales is no longer significant.

0.3 Some Important Questions

1. Is at least one of the the predictors $X_1, X_2, \ldots, X_p$ useful in predicting the response?

2. Do all the predictors help to explain Y, or is only a subset of the predictors useful?

3. How well does the model fit the data?

4. Given a set of predictor values, what response value should we predict, and how accurate is our prediction?

Is There a Relationship Between the Response and Predictors?

- In simple linear regression, we only have $\beta_1 = 0$ to check.

- Here, we need to ask if *all* of the regression coefficients are zero, i.e. we like to test the null hypothesis,

$$H_0 : \beta_1 = \beta_2 = \ldots = \beta_p = 0$$

  versus the alternative

$$H_1 : \text{at least one } \beta_j \text{ is non-zero.}$$

- This hypothesis is performed by computing the *F-statistic*,

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

where, as with simple linear regression,

$$\text{TSS} = \sum_{i=1}^{n}(y_i - \bar{y})^2 \quad \text{and} \quad \text{RSS} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

– Under H_0, the F-statistic is around 1, but under H_1, we expect it to be larger than 1.

– Hence, we reject H_0 for large values of F.

– If ε_i are normally distributed, F follows an F-distribution. This assumption can be relaxed for large n.

– As shown in the output for `Advertising`, R will compute the p-value associated with the F-statistic.

  * The p-value there is essentially zero, so we have strong evidence at least one of the media is associated with increased `sales`.

- Notice in the `Advertising` R output, for each individual predictor a t-statistic and a p-value were reported.
  - These provide information about whether each individual predictor is related to the response, *after adjusting for the other predictors*.
  - e.g. as discussed earlier, `newspaper` not associated with sales, in presence of `TV` and `radio`.
- If one of the p-values is small, then the overall F-statistic must give small p-value?
  - It *seems* like if one of the p-values is small, then at least one of the predictors is related to the response.
  - However, this logic is flawed, especially when number of predictors p is large.

- Consider the case where $p = 100$, and $H_0 : \beta_1 = \beta_2 = \ldots = \beta_p = 0$ is true.
  - Under H_0, there is a 5% chance any one of the p-value is below 0.05.
  - We expect to see approximately five small p-values even though H_0 is true!
  - If we use individual t-statistics and associated p-values, we will likely incorrectly conclude there is a relationship.
  - The F-statistic, by virtue of taking into account all the predictors within one hypothesis test, does not suffer from this problem.

Deciding on Important Variables

- Once we have established that at least one of the predictors is related to the response, it is natural to want to identify the responsible ones.
  - It is possible that all of the predictors are associated with the response, but often only a subset of predictors are.
  - Picking out these predictors is referred to *variable selection* or *feature selection*.
- Ideally we want to try out many different models, each containing a different subset of predictors.
  - For example, if $p = 2$, we can consider four models
    1. no predictors
    2. only X_1
    3. only X_2
    4. both X_1 and X_2

- We fit each of the four models and can compute one or more statistics to help us judge the quality of each model.
  * Popular criteria include *Mallow's* C_p, *Akaike information criterion* (AIC), *Bayesian information criterion* (BIC), and *adjusted* R^2. (More details in Ch06.)
- We can also plot various model outputs, such as residuals, to search for patterns.

• Unfortunately, there are a total of 2^p models that contain subsets of p variables.

  - Even with moderate p, e.g. $p = 30$, we end up with $2^{30} = 1,073,741,824$ models!

• We need a way to select a smaller set of models to consider. There are three classical approaches: Forward selection, Backward selection, mixed selection.

- *Forward Selection* (or Greedy approach)
  - We start with the null model (no predictors).
  - Fit p simple linear regressions,
  - add to the model the variable that results in the lowest RSS.
  - Iterate this process, always adding the variable that results in the lowest RSS, and never removing predictors that have been added.
  - This gives us $1 + \dfrac{p(p+1)}{2}$ models to choose from instead of 2^p.
  - You can also choose to have a stopping rule e.g. when new variable added has p-value too large.
- *Backward Selection* (or Backward Deletion)
  - We start with the full model.
  - remove the variable with the largest p-value.

  - Refit the model, and iterate.
  - Again, this gives us $1 + \frac{p(p+1)}{2}$ models to choose from.
  - You may also choose to have a stopping rule e.g. stop when the largest p-value is too small.

- *Mixed selection*

  - This is a combination of forward and backward selection.
  - We start with the null model, and do forward selection until the p-value is larger than some threshold.
  - We then do backward deletion until none of the remaining predictors have large p-values.
  - We iterate this forward and backward steps until all variables in the model have p-values smaller than the threshold, and any variable added to the model will have p-value larger than the threshold.

Model Fit

- RSE and R^2 are commonly used to quantify model fit.
- They are computed and interpreted in the same fashion as for simple linear regression.
- Recall R^2 is the proportion of explained variance in the response variable.
  - For `Advertising`, the full model has $R^2 = 0.8972$.
  - On the other hand, if we omit `newspaper`, $R^2 = 0.89719$.
  - Adding `newspaper` does (very slightly) increase R^2 even though we saw that the coefficient estimate is not significant.
- This is true in general: R^2 will always increase when we add variables since this allows us to fit the training data (but not necessarily the testing data) more closely.

  - Having only a tiny increase in R^2 is evidence that `newspaper` provides no real improvement to the model fit, and likely to lead to overfitting.
  - In contrast, the model with only `TV` has $R^2 = 0.6119$.
  - Adding `radio` to this model provides a much larger improvement.

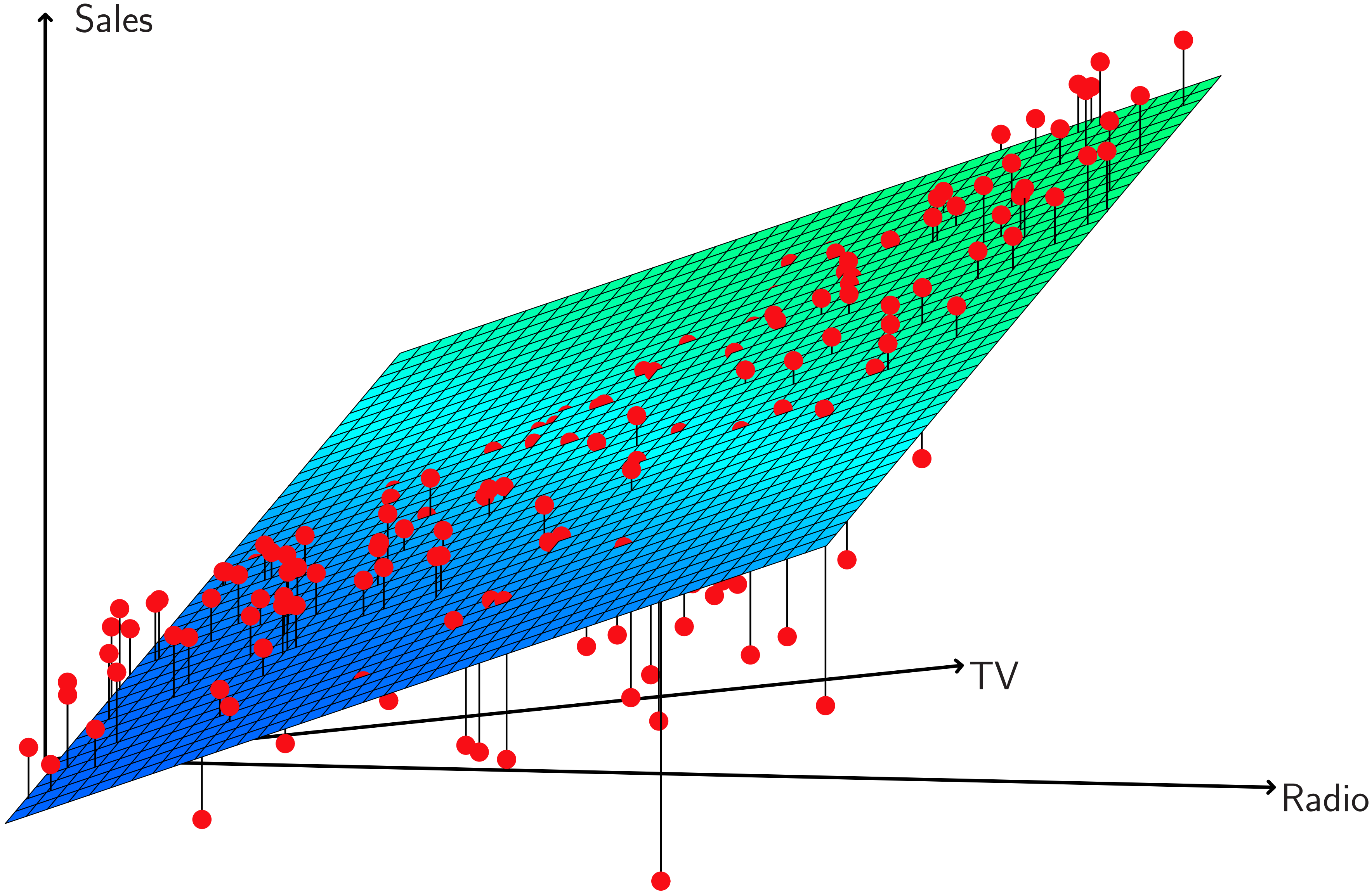
- The general definition for RSE is

$$\mathrm{RSE} = \sqrt{\frac{1}{n-p-1}\mathrm{RSS}}.$$

  - Computing the various RSEs for `Advertising` we get
    * full model has RSE $= 1.686$;
    * `TV` and `radio` has RSE $= 1.681$;
    * just `TV` has RSE $= 3.26$.

- Again this is evidence that, adding `radio` to `TV` improves our model, but further adding `newspaper` makes it worse.
- Note that RSE can increase for a larger model when the increase in RSS is small relative to the increase in p.

- In addition to RSE and R^2, we can also check the fit of the model by plotting the data.
  - Graphical summaries can sometimes reveal problems not visible from numerical statistics.
  - Plotting our best model with `TV` and `radio`, we notice a pattern:
    * the residuals tend to be negative when money was spent mostly on either `TV` or `radio`;
    * whereas they tend to be positive when money was spent on both `TV` and `radio`.


Sales
TV
Radio

- This suggests a *synergy* or *interaction* effect between the two advertising media: combining them gives a bigger boost than using them separately.
- This is a non-linear pattern, but can be accommodated within the linear model through the use of interaction terms.

Predictions

- Once we have fit the multiple regression model, it is straightforward to predict Y based on $X_1, X_2, \ldots, X_p$.
- We can break down the prediction error into three components:

1. How close our estimates are to the true parameters.
   - The *least squares plane*

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \ldots + \hat{\beta}_p X_p$$

     is an estimate for the *true population regression plane*

$$f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$$

   - The inaccuracy is part of the *reducible error*.
   - We can compute a *confidence interval* to estimate how close $\hat{Y}$ will be to $f(X)$.

2. How good of an approximation the linear model is for $f(X)$.
    - The linear model estimates the best linear approximation to the true surface.
    - The inaccuracy is a potential reducible error we call *model bias*.
    - Choosing to use a linear model assumes this error is small and we act as if the linear model was correct.
3. Even if we could estimate $f(X)$ perfectly, we still cannot predict the Y because of the random error ε.
    - This is the irreducible error.
    - We can use *prediction intervals* to take into account this variation from $f(X)$ to Y.

- Prediction intervals are always wider than confidence intervals.

    - Prediction intervals incorporate both the reducible error in the estimate for $f(X)$ as well as the irreducible error in how much an individual point will differ from the population regression plane.

– Prediction intervals are for individual observations, whereas confidence intervals are for the average response.

- For example, in `Advertising`, given we are interested in cities that each spent \$100,000 on `TV` and \$20,000 on `radio`,

  – the 95% confidence interval is $[10985, 11528]$.

  – This interval has a 95% probability to contain $f(100000, 20000)$ the true average `sales` for these cities.

  – On the other hand, the 95% prediction interval is $[7930, 14580]$.

  – This interval has a 95% probability to contain the `sales` in a *particular city* in this group of cities.

Three

Lab: Linear Regression

part (a): simple linear regression, multiple linear regression

3.1 LIBRARIES

- `library()` is used to load *libraries*, groups of functions and data sets, that are not included in the base R distribution.
- `MASS` is a large collection of data sets and functions.
- `ISLR2` includes data sets from the textbook.

```
> #library(MASS)
> library(ISLR2)
```

- Some libraries, such as `MASS`, comes with R.
- Others like `ISLR2` needs to be installed first.

- You can install packages using the menu items, or `install.packages()` through the console.

```
> install.packages("ISLR2")
trying URL 'https://cran.rstudio.com/bin/macosx/contrib/4.1/
ISLR2_1.3-1.tgz'
Content type 'application/x-gzip' length 4160940 bytes (4.0
MB)
==================================================
downloaded 4.0 MB
```

- `MASS` contains `Boston`, a data set with `medv` median house value for 506 neighbourhoods around Boston.
- `ISLR2` contains the same data set but with one less predictor. We will go ahead with the `ISLR2` version.
- There are 12 predictors and `?Boston` brings up descriptions for each variable.
- `head()` allows us to take a quick peek at the data.

```
> head(Boston)
     crim zn indus chas   nox    rm  age    dis rad tax
1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296
2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242
3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242
4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222
5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222
```

```
6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222
  ptratio lstat medv
1    15.3  4.98 24.0
2    17.8  9.14 21.6
3    17.8  4.03 34.7
4    18.7  2.94 33.4
5    18.7  5.33 36.2
6    18.7  5.21 28.7
```

- We use `lm()` to run a simple linear regression of `medv` on `lstat` (percentage of households with low socioeconomic status).

```
> lm.fit=lm(medv~lstat,data=Boston)
> attach(Boston)
> lm.fit=lm(medv~lstat)
```

- `lm.fit` is just the name of the variable we store the output from `lm()`. You can name it however you want.

- Calling `lm.fit` gives the basic coefficient estimates. We can use `summary(lm.fit)` for more details.

```
> lm.fit

Call:
lm(formula = medv ~ lstat)

Coefficients:
(Intercept)        lstat
```

```
      34.55        -0.95

> summary(lm.fit)

Call:
lm(formula = medv ~ lstat)

Residuals:
    Min       1Q   Median       3Q      Max
-15.168  -3.990  -1.318   2.034  24.500

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 34.55384    0.56263   61.41   <2e-16
lstat       -0.95005    0.03873  -24.53   <2e-16

(Intercept) ***
lstat       ***
---
Signif. codes:
```

```
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.216 on 504 degrees of freedom
Multiple R-squared:  0.5441,Adjusted R-squared:  0.5432
F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

- The output from `lm()`, in this case `lm.fit`, is an example of a *list*, a flexible R structure.
  - Each element of the list is a R object.
  - You can have a mixed list of vectors, matrices, and even lists.
  - Data frames are special lists, and as with data frames, you can use `names()` to get the names of all the variables a list contains.

```
> names(lm.fit)
 [1] "coefficients"  "residuals"     "effects"
 [4] "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "xlevels"
[10] "call"          "terms"         "model"
```

- `lm.fit$coefficients` will return the regression coefficient estimates.
  * For elements within a list, you can type just the first few letters provided there are no other elements sharing them.
  * You can also use the function `coef` to extract the coefficients.

```
> lm.fit$coefficients
(Intercept)       lstat
 34.5538409  -0.9500494
> lm.fit$co
(Intercept)       lstat
 34.5538409  -0.9500494
> coef(lm.fit)
(Intercept)       lstat
 34.5538409  -0.9500494
```

- `confint()` can be used to create confidence intervals for the coefficients.

```
> confint(lm.fit)
                2.5 %     97.5 %
(Intercept) 33.448457 35.6592247
lstat       -1.026148 -0.8739505
```

- For confidence intervals of $f(x)$ at various levels of x, you can use `predict()`. As the name suggests, you can also get prediction intervals.

```
> predict(lm.fit,data.frame(lstat=(c(5,10,15))), interval="co
nfidence")
       fit      lwr      upr
1 29.80359 29.00741 30.59978
2 25.05335 24.47413 25.63256
3 20.30310 19.73159 20.87461
> predict(lm.fit,data.frame(lstat=(c(5,10,15))), interval="pr
ediction")
       fit       lwr      upr
1 29.80359 17.565675 42.04151
2 25.05335 12.827626 37.27907
3 20.30310  8.077742 32.52846
```

  - Note that the prediction interval is for predicting a single observation and is significantly wider than the confidence interval.

  - Here we specified `newdata` to be 3 points at 5,10 and 15.

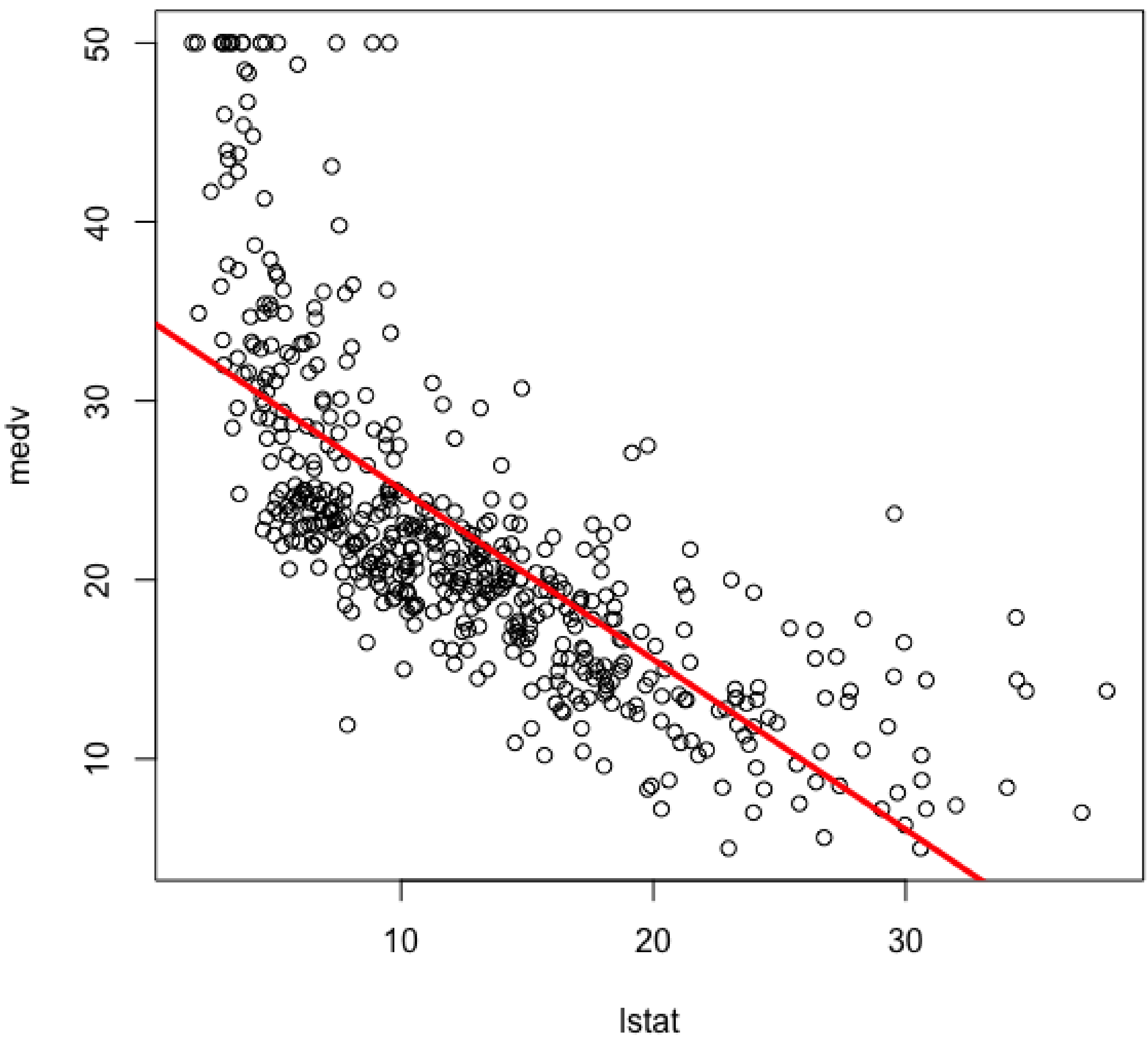
  - Note that the input argument needs to be in a data frame format, with the variable name matching the model.

  - If we omit this argument entirely, it will show the predictions and intervals for the training set.

- We can also directly plot the `lm()` output for simple linear regression since it returns the intercept and slope directly.

  - `abline()` in general adds a line to an existing plot.

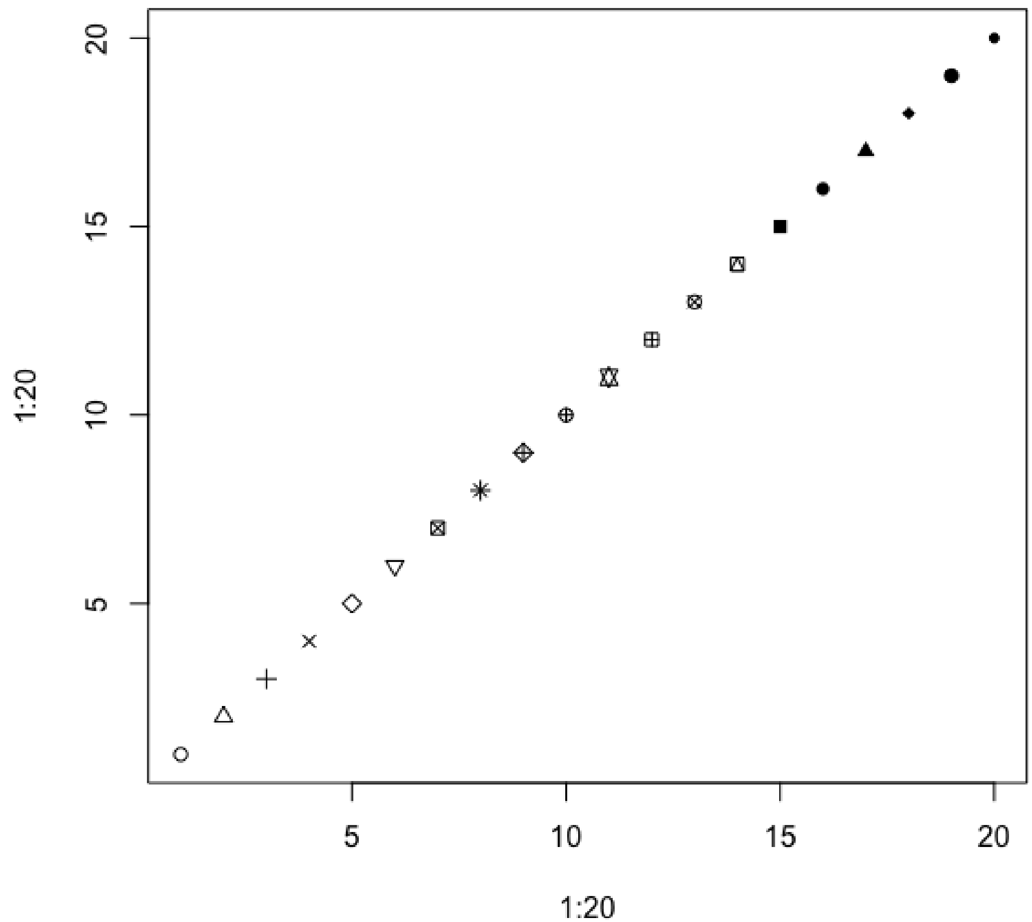
```
> plot(lstat,medv)
> abline(lm.fit,lwd=3,col="red")
```



medv
50
40
30
20
10
10
20
30
lstat

- The `pch` argument is a useful one, particularly if you want different symbols on the same plot.

```
> plot(1:20,1:20,pch=1:20)
```



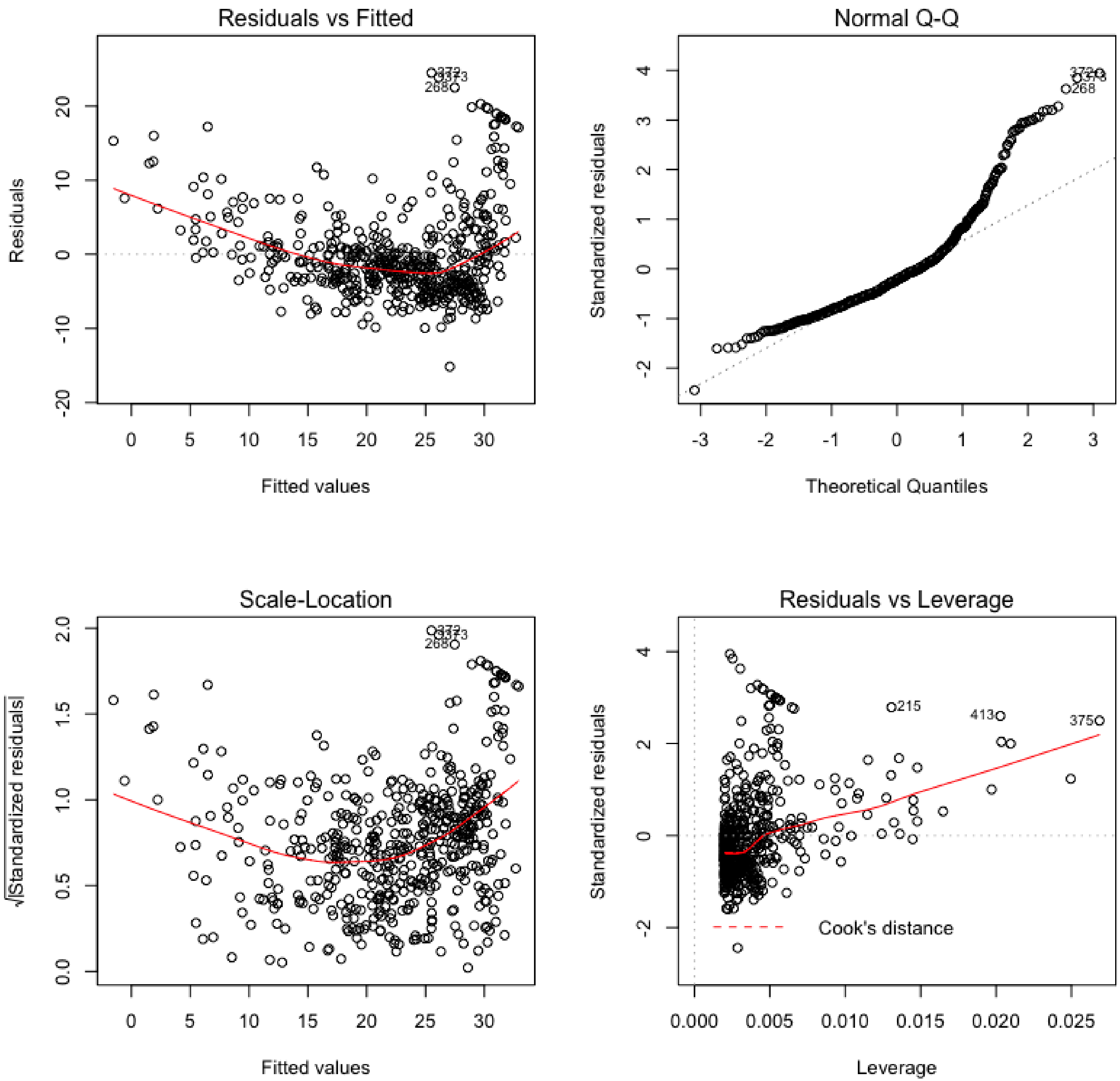
- Using `plot()` on the `lm()` output actually triggers `plot.lm()` which generates multiple plots that you can toggle through.

- If you like to view all four simultaneously, you can split up the plotting region into a grid of panels:

```
> par(mfrow=c(2,2))
> plot(lm.fit)
```

- You can change it back to the default `par(mfrow=c(1,1))` or simply close the plotting device

```
> dev.off()
null device
          1
```


Residuals vs Fitted
Residuals
Fitted values
372
373
268
Normal Q-Q
Standardized residuals
Theoretical Quantiles
268
Scale-Location
√|Standardized residuals|
Fitted values
372
373
268
Residuals vs Leverage
Standardized residuals
Leverage
215
413
375
Cook's distance

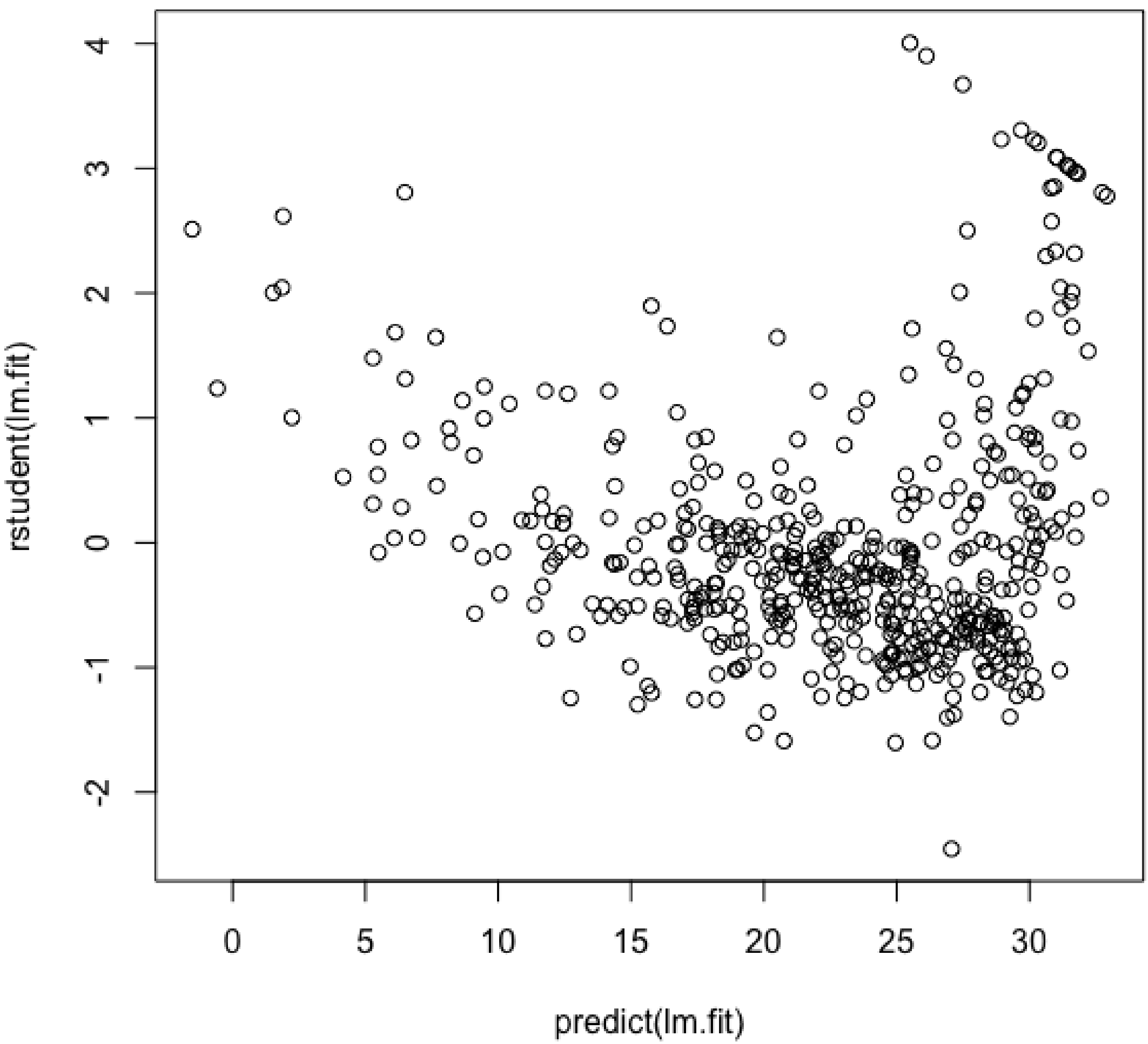
- We can also get residuals plot directly.
  - `residuals()` extract the residuals from `lm()` output
  - `rstudent()` gets the studentized residuals.

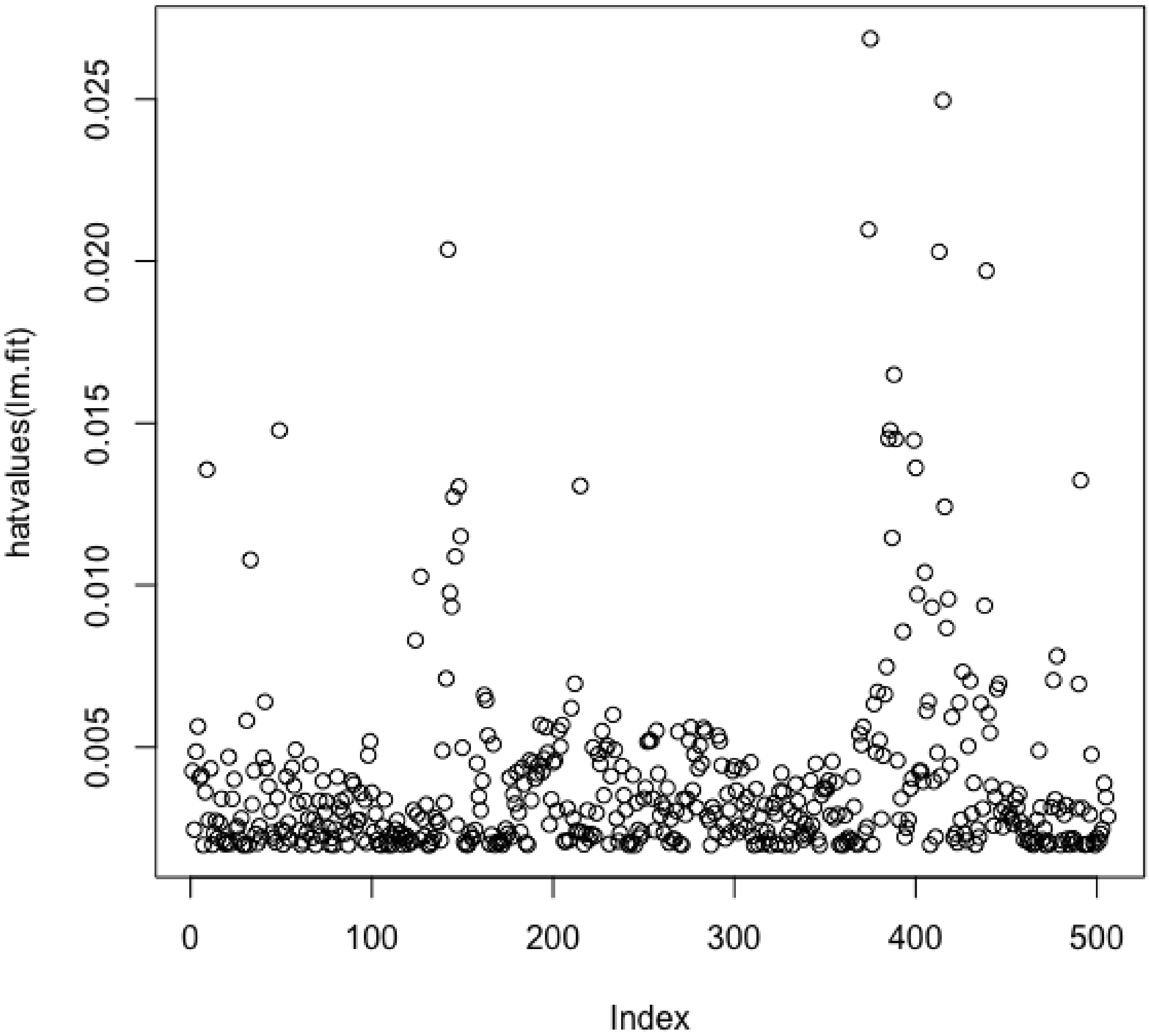
```
> plot(predict(lm.fit), rstudent(lm.fit))
```

  - From the residual plot, it is quite clear there is some non-linearity.

- `hatvalues()` is used to compute leverage statistics.
  - This works for multiple linear regression objects as well.
  - `which.max()` returns the index of the largest element: useful for identifying the largest leverage point.

```
> plot(hatvalues(lm.fit))
> which.max(hatvalues(lm.fit))
375
```


rstudent(lm.fit)
4
3
2
1
0
-1
-2
0
5
10
15
20
25
30
predict(lm.fit)



hatvalues(lm.fit)
0.005
0.010
0.015
0.020
0.025
0
100
200
300
400
500
Index

3.3 Multiple Linear Regression

- We use the same function `lm()` to fit multiple regression.
- The syntax `lm(y~x1+x2+x3)` is used to fit a model with three predictors, `x1`, `x2`, and `x3`.

```
> lm.fit=lm(medv~lstat+age,data=Boston)
> summary(lm.fit)

Call:
lm(formula = medv ~ lstat + age, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-15.981  -3.978  -1.283   1.968  23.158

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 33.22276    0.73085  45.458  < 2e-16 ***
```

```
lstat        -1.03207    0.04819 -21.416  < 2e-16 ***
age           0.03454    0.01223   2.826  0.00491 **
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.173 on 503 degrees of freedom
Multiple R-squared:  0.5513,Adjusted R-squared:  0.5495
F-statistic:   309 on 2 and 503 DF,  p-value: < 2.2e-16
```

- You can also regress on all the variables with the following short-hand:

```
> lm.fit=lm(medv~.,data=Boston)
> summary(lm.fit)

Call:
lm(formula = medv ~ ., data = Boston)

Residuals:
```

```
     Min       1Q   Median       3Q      Max
-15.1304  -2.7673  -0.5814   1.9414  26.2526

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  41.617270   4.936039   8.431 3.79e-16 ***
crim         -0.121389   0.033000  -3.678 0.000261 ***
zn            0.046963   0.013879   3.384 0.000772 ***
indus         0.013468   0.062145   0.217 0.828520
chas          2.839993   0.870007   3.264 0.001173 **
nox         -18.758022   3.851355  -4.870 1.50e-06 ***
rm            3.658119   0.420246   8.705  < 2e-16 ***
age           0.003611   0.013329   0.271 0.786595
dis          -1.490754   0.201623  -7.394 6.17e-13 ***
rad           0.289405   0.066908   4.325 1.84e-05 ***
tax          -0.012682   0.003801  -3.337 0.000912 ***
ptratio      -0.937533   0.132206  -7.091 4.63e-12 ***
lstat        -0.552019   0.050659 -10.897  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 4.798 on 493 degrees of freedom
Multiple R-squared:  0.7343,Adjusted R-squared:  0.7278
F-statistic: 113.5 on 12 and 493 DF,  p-value: < 2.2e-16
```

- Just as you can access individual components of the `lm()`, you can also access individual components of `summary()`.

```
> names(summary(lm.fit))
 [1] "call"          "terms"         "residuals"
 [4] "coefficients"  "aliased"       "sigma"
 [7] "df"            "r.squared"     "adj.r.squared"
[10] "fstatistic"    "cov.unscaled"
> summary(lm.fit)$r.squared
[1] 0.734307
> summary(lm.fit)$sigma
[1] 4.798034
```