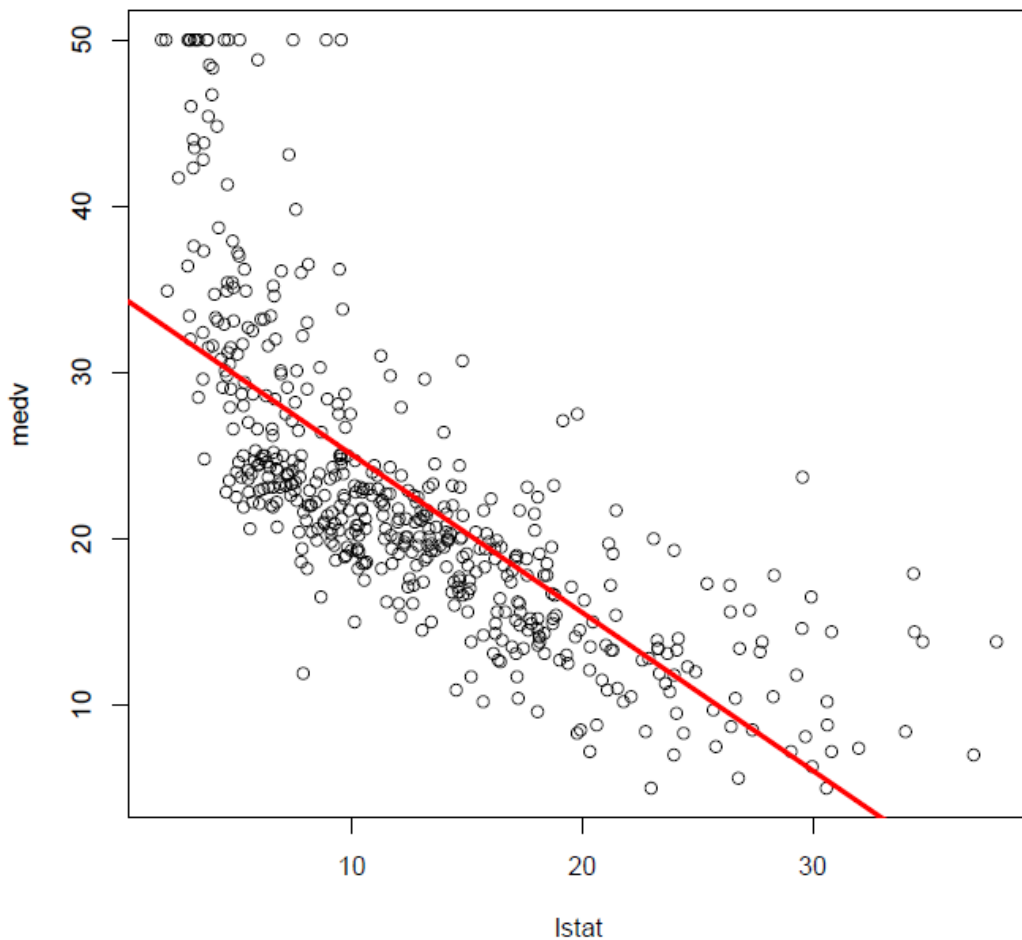# 2.1 Simple Linear Regression (SLR) analysis

Introduction

> **i** **Simple linear regression (SLR)** is a method to explain the relationship between two quantitative variables using a straight line. One variable is a **response** variable Y and the other one is a **predictor** variable X.

In this section, we will work with the `Boston` dataset (a famous housing dataset available in R) and show how to perform a simple linear regression analysis on this particular dataset. Each step of the analysis will be preceded with some theoretical background. We represent data as n pairs of observations:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n).$$

Each pair consists of measurements of variables X and Y.

We will describe in detail how to calculate the intercept and slope estimates in the simple linear regression problem. Additionally, we will discuss how to assess the accuracy of the parameter estimates and how to assess the accuracy of the SLR model itself. We will also identify some potential problems arising in linear regression in general.

> **i** The purpose of this section is not only to introduce simple linear regression but also to identify the main steps of statistical analysis, which can later be applied to more complex statistical models.

Example: Simple Linear Regression (SLR) analysis

The `MASS` R library contains the `Boston` dataset, which records `medv` (median house value) for 506 neighbourhoods around Boston. Fit a simple linear regression model using `medv` as your response variable and `lstat` (per cent of households with low socioeconomic status) as your predictor variable.

We will perform the simple **linear regression analysis** in the following **steps**:

Step 1: Inspect, summarise and visualise your dataset.

Step 2: Produce a scatter diagram of the response variable versus the explanatory variable. What is the relationship between these two variables?

Step 3: Fit the SLR model using the *lm()* function in R. Write down the resulting regression equation. What does this equation tell you?

Step 4: Assess the accuracy of the coefficient estimates using the R output.

Step 5: Assess the accuracy of the SLR model.

Step 6: Identify any potential problems in your analysis by using diagnostic plots.

Step 7: Use the regression equation to predict the value of `medv` for `lstat` values of 5, 10 and 15.

Below, the text in blue refers directly to our example while the text in black explains the theoretical background or presents other information needed for the analysis.

Step 1: Inspecting and summarising the data

> ℹ️ As the first step, we load the library MASS in R and display names of all the variables in the Boston dataset using the **names**() function. Type ?**Boston** to see the description of the Boston dataset in the help window.

Try entering the code into the R terminal to the right

```
> library(MASS)
> ?Boston
(press q to close it) ...
> names(Boston)
[1] "crim" "zn" "indus" "chas" "nox" "rm" "age"
[8] "dis" "rad" "tax" "ptratio" "black" "lstat" "medv"
```

> ℹ️ The **dim**() function tells us that the dataset has 506 rows, or *observations*, and 14 columns, or *variables*.

```
> dim(Boston)
[1] 506   14
```

> ℹ️ There are several options in R to view the data. The **head**() function returns the first n rows ($n = 6$ by default) of a dataset. The **fix**() function can be used to view the data in a spreadsheet like window. Note that the window must be closed before further R comments can be entered.

```
> head(Boston)
      crim zn indus chas   nox    rm  age    dis rad tax ptratio  black lstat medv
1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98 24.0
2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14 21.6
3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83  4.03 34.7
4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63  2.94 33.4
5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90  5.33 36.2
6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12  5.21 28.7
```

> ℹ️ The **summary**() function produces a numerical summary for each variable in our dataset. Simply typing **summary**(**medv**) or **summary**(**lstat**) will give an error message, because we have not told R to look in the **Boston** dataset for these variables. To refer to a variable, we must type the data set name followed by a $ symbol and the variable name.

```
> summary(Boston$medv)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   5.00   17.02   21.20   22.53   25.00   50.00

> attach(Boston)
> summary(lstat)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
   1.73    6.95    11.36    12.65    16.95    37.97
> detach(Boston)

> with(data=Boston, summary(medv))
   Min. 1st Qu.  Median     Mean 3rd Qu.     Max.
   5.00   17.02   21.20    22.53   25.00    50.00
```

> i Alternatively, use the **attach**() function in order to tell R to make the variables in the attached dataset available by name. It is a good practice to detach the dataset after the task is completed. To avoid the **attach**() function, it is recommended to apply the **with**(**data**, ...) function.

```
> attach(Boston)
> summary(cbind(medv,lstat))
       medv               lstat
 Min.    : 5.00     Min.    : 1.73
 1st Qu. : 17.02    1st Qu. : 6.95
 Median  : 21.20    Median  : 11.36
 Mean    : 22.53    Mean    : 12.65
 3rd Qu. : 25.00    3rd Qu. : 16.95
 Max.    : 50.00    Max.    : 37.97
```

> i We can now produce the summary table for the variables of interest. The **cbind**() combines by columns the numerical summaries of `medv` and `lstat`.

```
> attach(Boston)
> summary(cbind(medv,lstat))
       medv               lstat
 Min.    : 5.00     Min.    : 1.73
 1st Qu. : 17.02    1st Qu. : 6.95
 Median  : 21.20    Median  : 11.36
 Mean    : 22.53    Mean    : 12.65
 3rd Qu. : 25.00    3rd Qu. : 16.95
 Max.    : 50.00    Max.    : 37.97
```

Code summary

```
library(MASS)

names(Boston)
dim(Boston)
head(Boston)

summary(Boston$medv)

attach(Boston)
summary(lstat)
summary(cbind(medv,lstat))
detach(Boston)
```
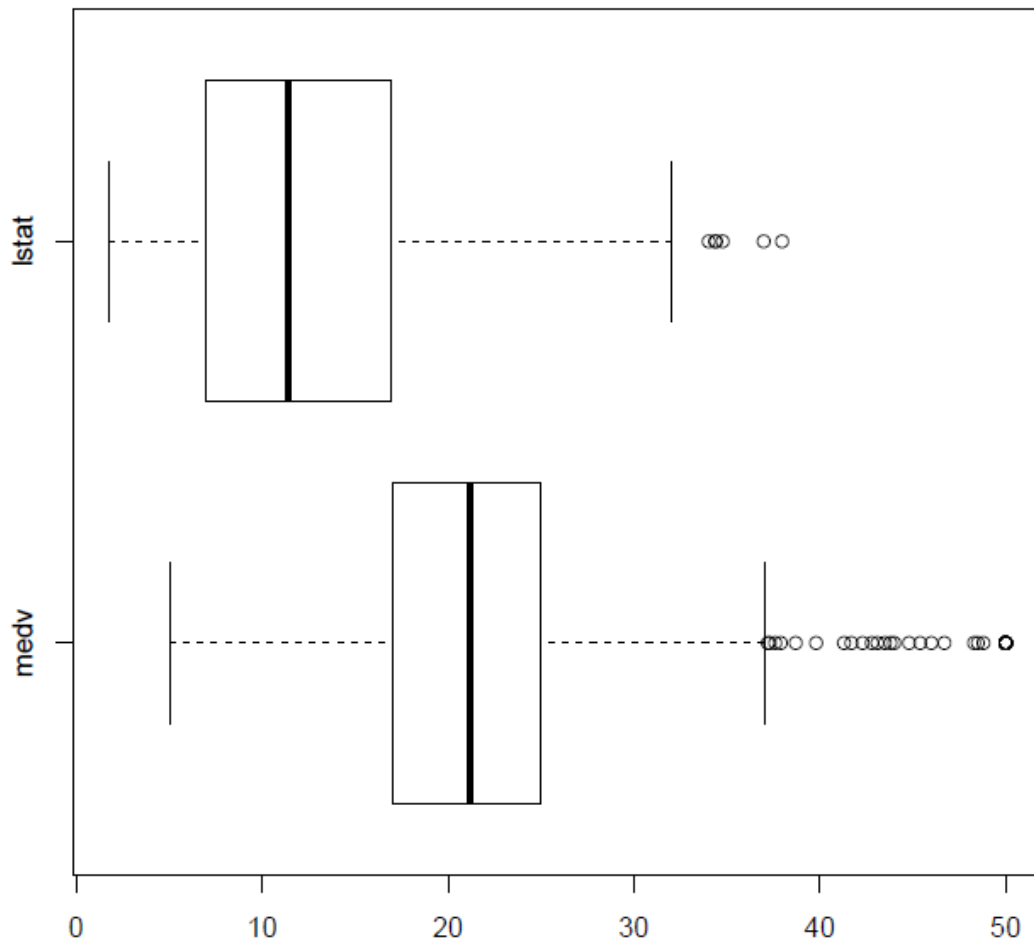
Step 2: Boxplot and Scatterplot

> **i** Comparative boxplots are a convenient way of graphically depicting groups of numerical data.

```
library(MASS)
attach(Boston)
boxplot(cbind(medv,lstat),horizontal = TRUE)
```


> **i** The boxplots suggest that there may exist some possible *outliers* affecting our analysis. If the outliers were removed the distribution of `lstat` would seem slightly right-skewed and the distribution of `medv` would seem symmetrical.

> **i** The scatterplot here is used to assess the relationship between the variables of interest.

```
library(MASS)

attach(Boston)
plot(lstat, medv)
```
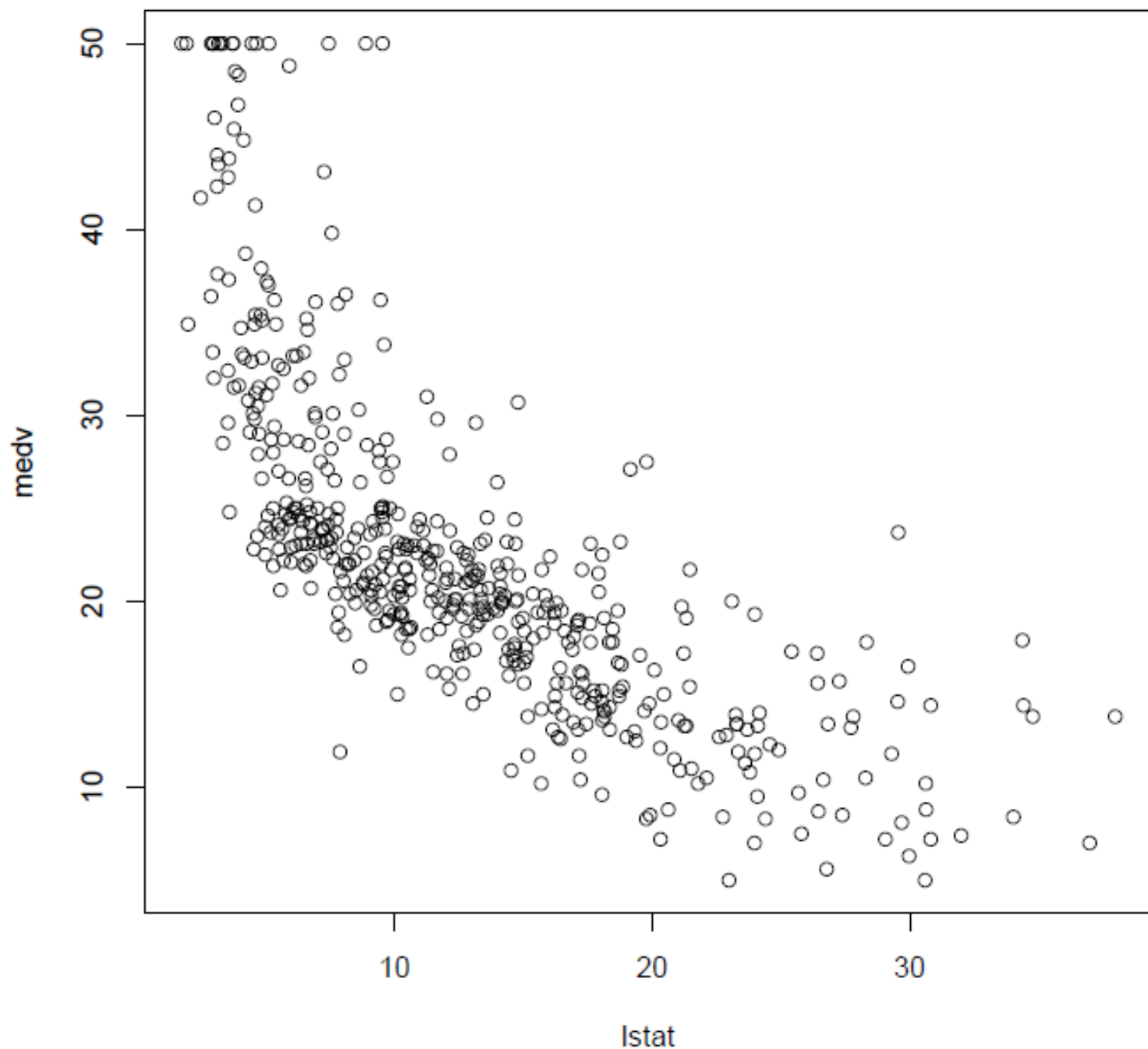
We can see that there exists a decreasing relationship between the `medv` and `lstat` variables. There seems to be some evidence of non-linearity in this relationship. On the other hand, it is likely that after removing the outliers the relationship would appear more linear. In this example, we will fit the linear model to the original dataset. However, you can later experiment with removing the outliers from your dataset.

Step 3: Fitting the SLR

The simple linear regression model involves only one independent variable X. The functional relationship between the true mean of Y_i, that is $E(Y_i)$, and X_i is the equation of a straight line:

$$E(Y_i) = \beta_0 + \beta_1 X_i,$$

for $i = 1, \ldots, n$, with

- β_0 - intercept of the line - the value of $E(Y_i)$ when $X = 0$;
- β_1 - slope of the line - the rate of change in $E(Y_i)$ per unit change in X.

The deviation of the observation Y_i from its population mean $E(Y_i)$ is taken into account by adding a random error ε_i to give the statistical model:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

for $i = 1, 2, \ldots, n$.

> **i**   There are two important additional assumptions in the SLR analysis:

- X_i are measured without error, so x_i are fixed constants;
- The errors ε_i are independent from each other and are normally distributed with a mean of 0 and a common variance σ^2, that is, $\varepsilon_i \sim N(0, \sigma^2)$.

Let $\hat{\beta}_0$ and $\hat{\beta}_1$ be numerical estimates of the parameters β_0 and β_1 obtained from data, and let

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

be the estimated mean of Y_i, or prediction of Y_i, when $X_i = x_i$, for each $i = 1, \ldots, n$.

> **i**   The *least squares* principle chooses $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize the sum of squares of the residuals. The ith residual $e_i = y_i - \hat{y}_i$ represents the prediction error for data point i when we use $\hat{y}_i$ to predict the actual response y_i. The residual sum of squares (RSS) is then given by
>
> $$RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i))^2.$$
>
> The estimates of β_0 and β_1 are obtained by minimizing RSS. The derivatives of RSS with respect to $\hat{\beta}_0$ and $\hat{\beta}_1$ are set to zero, and as a result, the least squares estimates must satisfy the following *normal equations*:

$$n(\hat{\beta}_0) + \left(\sum_{i=1}^{n} x_i\right)\hat{\beta}_1 = \sum_{i=1}^{n} y_i$$

$$\left(\sum_{i=1}^{n} x_i\right)\hat{\beta}_0 + \left(\sum_{i=1}^{n} x_i^2\right)\hat{\beta}_1 = \sum_{i=1}^{n} x_i y_i.$$

(2.1.1)

Solving the above equations gives the *least squares estimates* for the *slope* and *intercept*:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

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

(2.1.2)

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ are the sample means.

The estimates form (2.1.2) give the equation of the *best fitting line*:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i.$$

Naturally, we still have to verify whether $\hat{\beta}_0$ and $\hat{\beta}_1$ really minimize RSS and satisfy the second order conditions of the minimizing problem. Thus we need the second derivatives of RSS with respect to $\hat{\beta}_0$ and $\hat{\beta}_1$, which are given by the so-called Hessian matrix (matrix of second derivatives)

$$H = 2\begin{pmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix}.$$

Then it remains to be shown that the Hessian matrix is positive definite.
Since $n > 0$ and $\det(H) = 4\left(n\left(\sum x_i^2\right) - \left(\sum x_i\right)^2\right) > 0$ from Hölder inequality, the Hessian matrix H is positive definite and therefore $\hat{\beta}_0$ and $\hat{\beta}_1$ minimize RSS.

We now fit the simple linear regression model to the data with `medv` as the response and `lstat` as the predictor as follows:

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
summary(lm.fit)
```

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

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 ***
```

```
---
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
```

From the above R output we can write down the obtained regression equation of the fitted line:

$$\widehat{\text{medv}} = 34.55 - 0.95 \times \text{lstat},$$

which shows that the `medv,` median house value, decreases as `lstat,` percent of households with low socioeconomic status, increases and the rate of decrease is equal to 0.95.

All the information provided when we fit the SLR model to our data by using the $\text{lm}()$ function is now stored in the object lm.fit. When we type $\text{summary}(\text{lm.fit})$, it displays the detailed summary information about the fitted model. There are different ways to extract information from lm.fit. For example, $\text{coef}(\text{lm.fit})$ provides the estimates of the intercept and slope of the fitted model. We see from the output below that $\hat{\beta}_0 = 34.55$ and $\hat{\beta}_1 = -0.95$, so that the predictions are obtained by $\hat{y}_i = 34.55 - 0.95 x_i$.
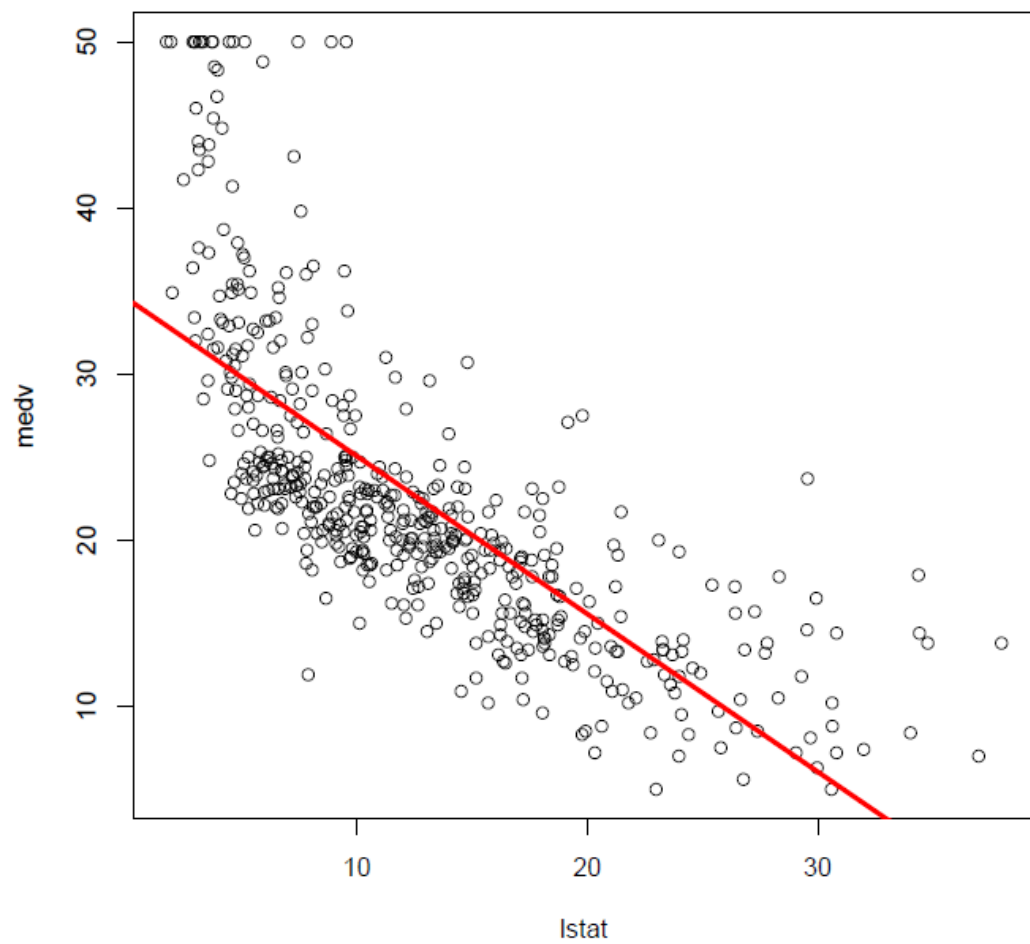
```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
coef(lm.fit)
```

```
(Intercept)        lstat
34.5538409   -0.9500494
```

We will now add the least squares regression line to the scatterplot:

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)

attach(Boston)
plot(lstat,medv)
abline(lm.fit,col='red',lwd=3)
```

Step 4: Assessing the accuracy of the coefficient estimates in SLR

Standard errors for coefficient estimates

The coefficient estimates given in Step 2 are *unbiased*, that is, $E(\hat{\beta}_0) = \beta_0$ and $E(\hat{\beta}_1) = \beta_1$. The standard errors of $\hat{\beta}_0$ and $\hat{\beta}_1$, written as $SE(\hat{\beta}_0)$ and $SE(\hat{\beta}_1)$ can be computed using the following formulas:

$$SE(\hat{\beta}_0)^2 = \sigma^2 \left(\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \right)$$

$$SE(\hat{\beta}_1)^2 = \frac{\sigma^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2},$$

where $\sigma^2 = \mathrm{Var}(\varepsilon)$.

In general, σ is not known, but can be estimated from the data. This estimate is known as the *residual standard error*, and is given by

$$\hat{\sigma} = RSE = \sqrt{\frac{RSS}{n-2}}.$$

When σ is estimated from data, we should write $\widehat{SE}(\hat{\beta}_0)$ and $\widehat{SE}(\hat{\beta}_1)$ using extra "hat", but we will not use this extra "hat" in our notations.

> **i** The standard errors for coefficient estimates in our example are $SE(\hat{\beta}_0) = 0.5626$ and $SE(\hat{\beta}_1) = 0.0387$, while the estimate of σ is 6.216. See `summary(lm.fit)` output.

Confidence Interval

Standard errors can be used to compute a $(1 - \alpha)100\%$ **confidence intervals** for β_0 and β_1 as:

$$[\hat{\beta}_k - t_{\alpha/2, n-2} SE(\hat{\beta}_k), \hat{\beta}_k + t_{\alpha/2, n-2} SE(\hat{\beta}_k)],$$

$k = 0, 1$, where $t_{\alpha/2, n-2}$ is $\alpha/2$ critical value of a Student-t distribution with $n - 2$ degrees of freedom.

> **i** In order to obtain confidence intervals for β_0 and β_1, we can use the `confint()` command. The level of confidence is 0.95 by default. From the R output below, the 95% confidence interval for the slope is

$$[-1.026, -0.874].$$

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
confint(lm.fit)
```

```
                 2.5 %      97.5 %
(Intercept) 33.448457  35.6592247
lstat       -1.026148  -0.8739505
```

> **i** You can also try to use the confidence interval formula above. There are $n = 506$ observations in the Boston dataset. The critical value $t_{1-\alpha/2,n-2}$ required for the 95% confidence interval, that is, for $\alpha = 0.05$ can be found using the quantile function for the t distribution `qt()`:

```
qt(0.025,504,lower.tail=FALSE)
```

```
[1] 1.964682
```

Hypothesis tests on the coefficients

Standard errors can also be used to perform **hypothesis tests** on the coefficients. The most common hypothesis test involves testing the **null hypothesis** of

$H_0 : \beta_1 = 0$ (there is no relationship between X and Y)

versus the **alternative hypothesis**

$H_1 : \beta_1 \neq 0$ (there is some relationship between X and Y).

For the purpose of testing the above hypothesis we compute a t-**statistic**, given by

$$t = \frac{\hat{\beta}_1 - 0}{SE(\hat{\beta}_1)},$$

which has a Student-t distribution with $n - 2$ degrees of freedom under H_0, that there is no relationship between X and Y.

When $|t| < t_{\alpha/2,n-2}$, we cannot reject the H_0 at the α level of significance.

This is equivalent to the decision based on a **p-value**, that is, we reject H_0 if p-value is small enough (p-value $< \alpha$).

> **i** In our example, see `summary(lm.fit)` output, at $\alpha = 0.05$ level of significance, we can reject the H_0 hypothesis of $\beta_1 = 0$ since p-value< 0.05. The observed value of t-statistic is -24.53 and the distribution of the test statistic is Student-t distribution with 504 degrees of freedom.

> **i** Note that p-values of the tests $H_0 : \beta_k = 0$ vs $H_1 : \beta_k \neq 0$, $k = 0, 1$ are given in the column of `Pr(> |t|)`.

Note also that the F statistic given in the R output corresponds to H_0 : there is no relationship between the response and the predictor ($\beta_1 = 0$).

Step 5: Assessing the accuracy of the SLR model

The quality of a linear regression fit is typically assessed using the residual standard error (RSE) and the R^2 statistic.

Residual Standard Error

Recall that the **residual standard error** (RSE) is an estimate of the standard deviation of ε and is given by

$$RSE = \sqrt{\frac{RSS}{n-2}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}.$$

The RSE is considered a measure of **lack of fit** of the model to the data. It is useful when fits from two different models are compared. The RSS will be small for the model which fits the data well, since the predictions $\hat{y}_i$ will be close to the observations y_i, resulting in the small RSE.

> **i** The residual standard error in our example is equal to 6.216.

Coefficient of Determination R^2

A measure of the contribution of the independent variable(s) in the model is R^2 statistic, known as the **coefficient of determination**:

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

where

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

is the **total sum of squares** and

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

the residual sum of squares, was defined earlier.

The coefficient of determination can be interpreted in the following way:

- **TSS** is the amount of variability inherent in the response before the regression is performed;
- **RSS** is the amount of variability that is left unexplained after performing the regression;
- **TSS-RSS** is the amount of variability in the response that is explained (or removed) by performing the regression;
- R^2 is the proportion of variability in Y that can be explained using X; it's value always lies between 0 and 1.
- In the simple linear regression setting, $R^2 = r^2$, where r is the correlation coefficient.

> i The R^2 statistic is rather low in our example. The $R^2 = 0.5441$ suggests that only 54% of the variability in `medv` can be explained using `lstat`. More complex model or removing of outliers might be needed to improve R^2.

Step 6: Diagnostic plots

Some potential problems may arise in linear regression. Below we list such possible issues and suggest some diagnostic plots that can be used to identify them.

1. Non-linearity of the response-predictor relationship:
    Residual plots - we plot residuals $e_i = y_i - \hat{y}_i$ versus x_i or versus fitted values $\hat{y}_i$. Non-linearity can be seen in the presence of a pattern, such as U-shape.

2. Correlation of error terms:
    If there is a time component in the data, we plot the residuals as a function of time (when data is time dependent).

3. Non-constant variance of error terms:
    Residual plots - heteroscedasticity can be seen in the form of a funnel shape in the residuals versus fitted values plot.

4. Outliers:
    Outliers are observations for which the response y_i is unusually far from the predicted value. Use a plot of studentized residuals, computed by dividing each residual by its estimated standard error (RSE). Observations whose studentized residuals are greater than 3 in absolute value are possible outliers.

1. High-leverage points:
    Observations with high-leverage have an unusual value for x_i. Plot studentized residuals versus the leverage statistic defined by

    $$h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{k=1}^{n}(x_k - \bar{x})^2}.$$

    The leverage statistic has values between $1/n$ and 1, with average $2/n$. If given observation has h_i that exceeds 2 or 3 times the average $2/n$, then we may suspect the corresponding point has high leverage.
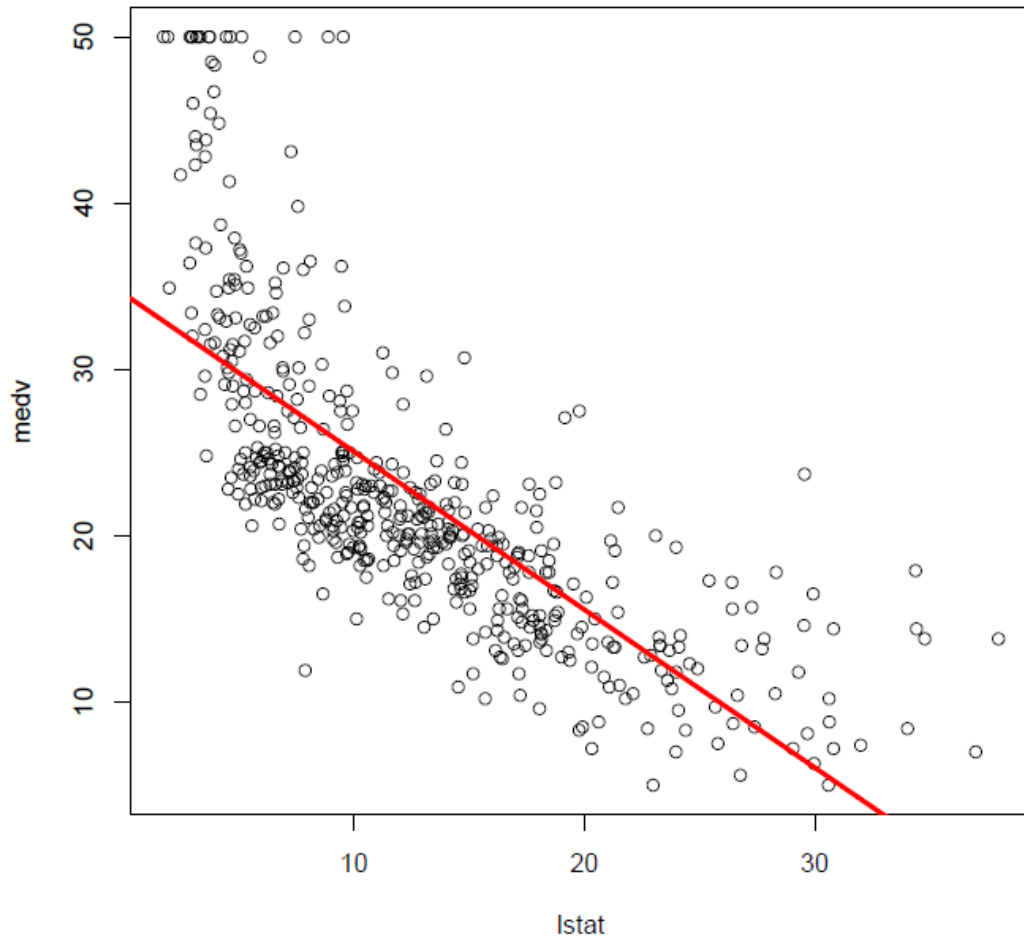
2. Collinearity.
    Collinearity refers to the situation in which two or more predictor variables are closely related to one another (not the case in SLR).

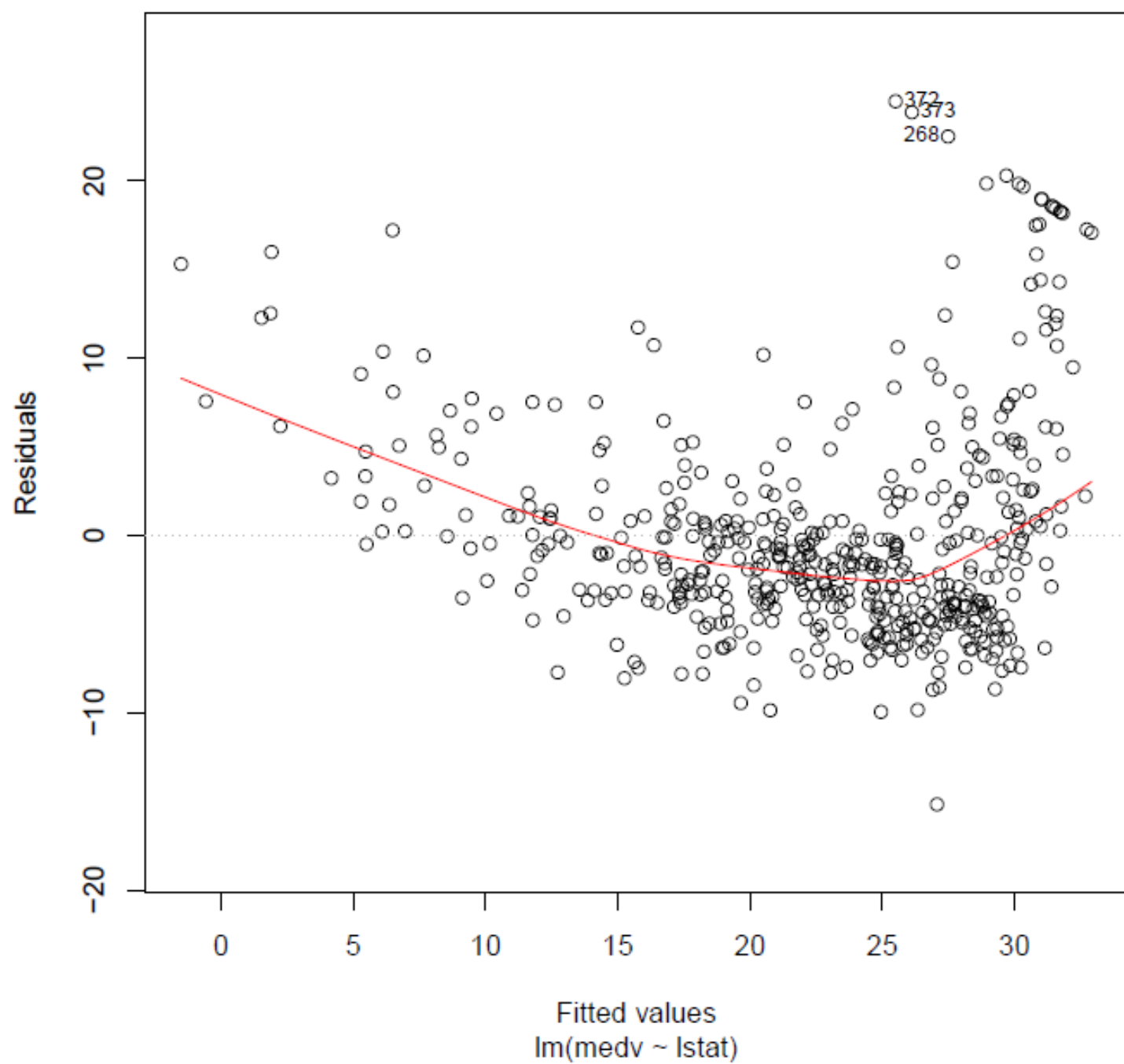> **i** We will now plot `medv` and `lstat` along with the least squares regression line:

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
attach(Boston)
plot(lstat, medv)
abline(lm.fit,col='red',lwd=3)
```
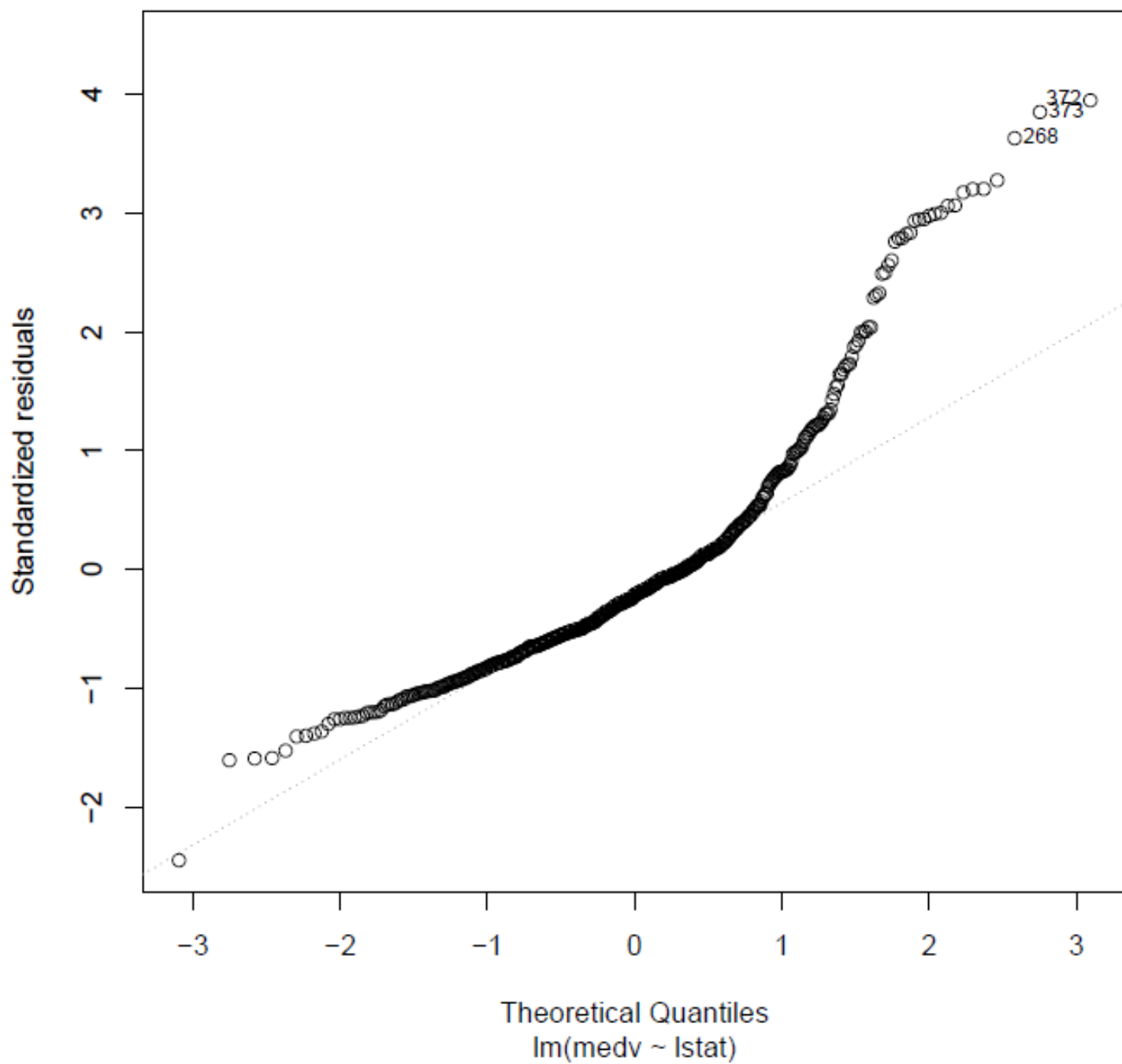
> **i** There is some evidence of non-linearity in the relationship between `lstat` and `medv`. Four diagnostic plots can be produced by applying the **plot**() function directly on the output from **m**()*:*

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
plot(lm.fit)
```
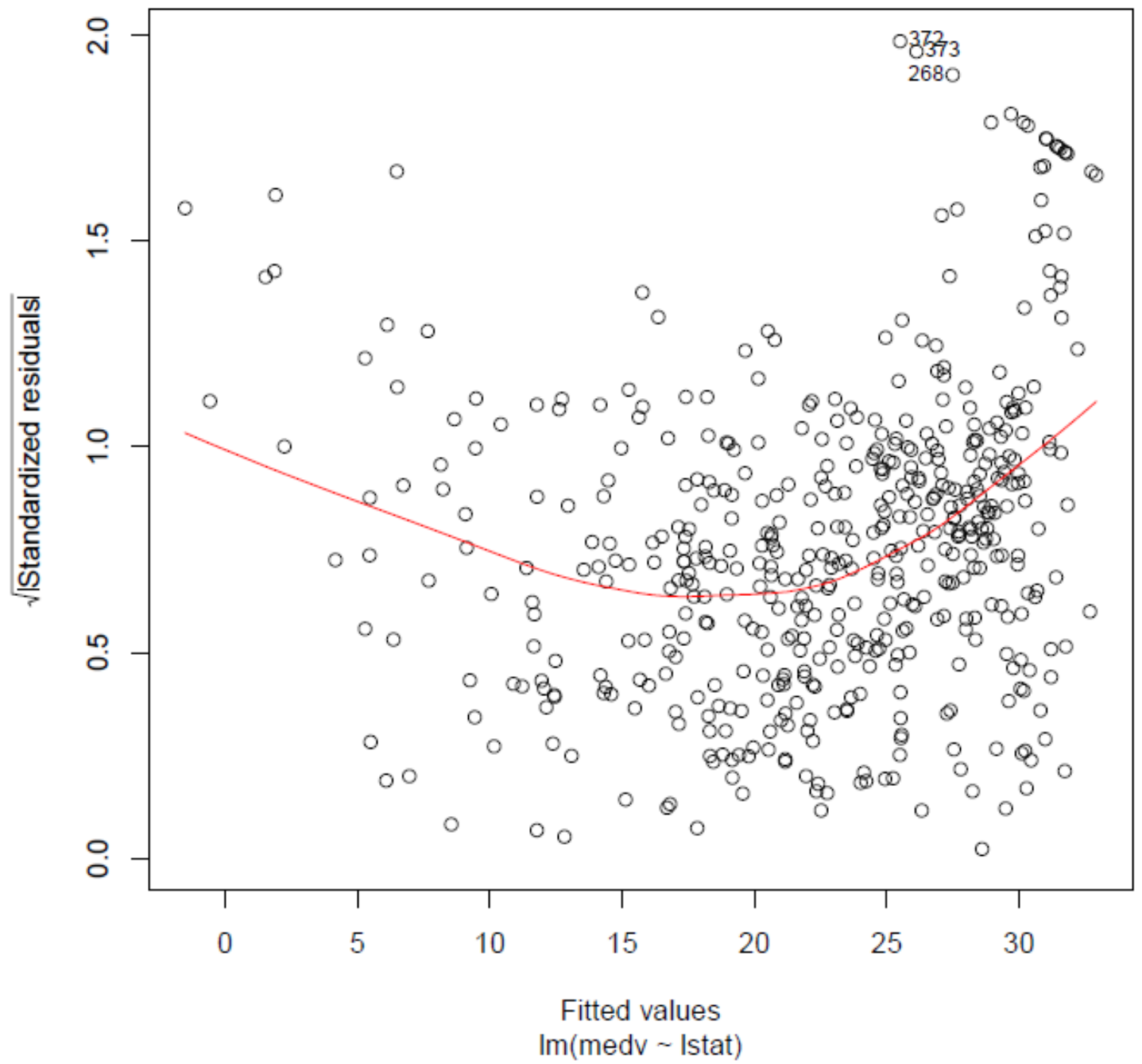
Residuals vs Fitted

372
373
268

Residuals

Fitted values
lm(medv ~ lstat)

Normal Q–Q

Im(medv ~ lstat)

Scale–Location

372
373
268

√|Standardized residuals|

Fitted values
lm(medv ~ lstat)

Residuals vs Leverage

lm(medv ~ lstat)
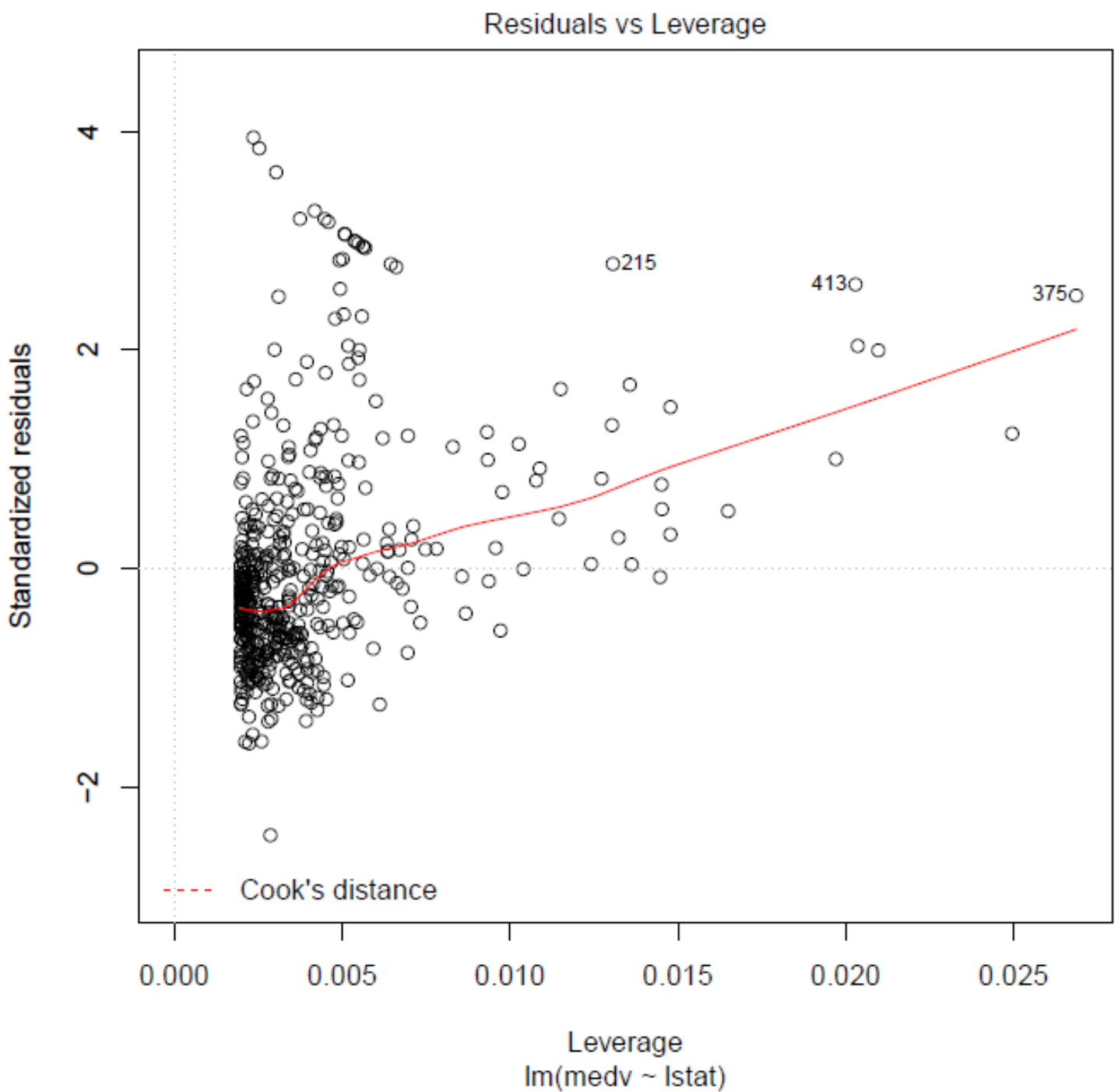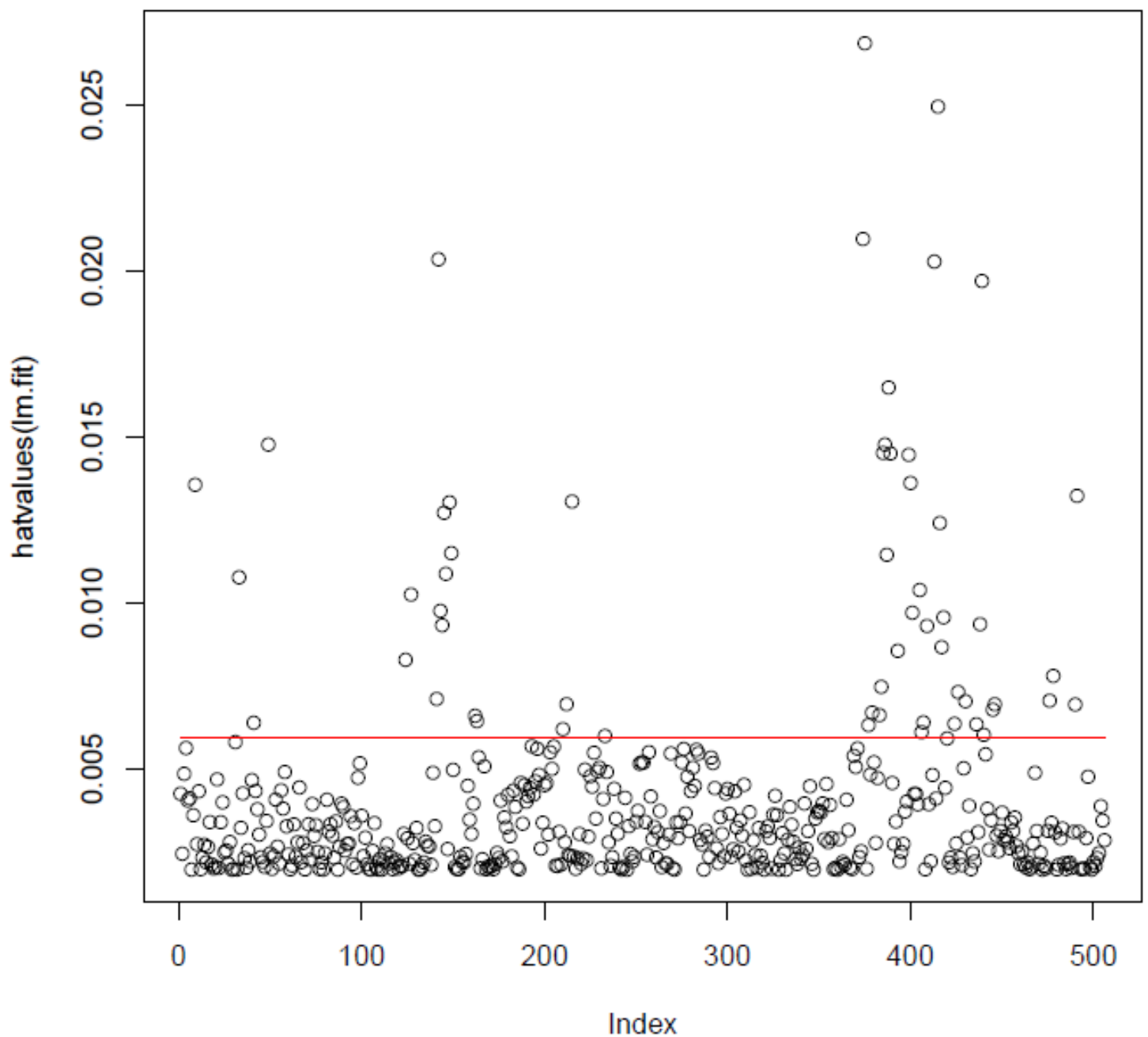
> On the basis of the residual plots, there is some evidence of non-linearity. Leverage statistics can be computed for any number of predictors using the `hatvalues()` function:

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
plot(hatvalues(lm.fit))
lines(rep(3*2/506,506),col=2)
```

> The red line in the plot indicates the $3 \times 2/n$ level. To identify the index of the largest element of a vector of leverage statistics we used the function `which.max()`.

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
which.max(hatvalues(lm.fit))
```

```
375
375
```

> Note also that we can use the **names()** function to show what other information is stored in `lm.fit` :

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)
```

```
names(lm.fit)

lm.fit$coefficients
```

```
[1] "coefficients" "residuals" "effects" "rank"
[5] "fitted.values" "assign" "qr" "df.residual"
[9] "xlevels" "call" "terms" "model"


(Intercept)       lstat
34.5538409 -0.9500494
```

Step 7: Prediction

> ℹ️ Let us now use the **predict**() function to calculate the predicted value of `medv` for `lstat` equal to 5, 10 and 15. For example, predicted value of `medv` is 25.05 when `lstat` equals 10.

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)

predict(lm.fit, data.frame(lstat=c(5,10,15)))
```

```
       1        2        3
29.80359 25.05335 20.30310
```

> ℹ️ The **predict**() function can also be used to produce **confidence intervals** and **prediction intervals** for the predicted value of `medv` for `lstat` equal to 5, 10 and 15. For example, the 95% confidence interval associated with a `lstat` value of 10 is [24.47, 25.63], and the 95% prediction interval is [12.83, 37.28]. Both intervals are centered around 25.05, which is the predicted value of `medv` when `lstat` equals 10.

```
library(MASS)
lm.fit<-lm(medv~lstat,data=Boston)

predict(lm.fit, data.frame(lstat=c(5,10,15)))
predict(lm.fit, data.frame(lstat=c(5,10,15)),interval="confidence")
predict(lm.fit, data.frame(lstat=c(5,10,15)),interval="prediction")
```

```
> predict(lm.fit, data.frame(lstat=c(5,10,15)))
       1        2        3
29.80359 25.05335 20.30310

> predict(lm.fit, data.frame(lstat=c(5,10,15)),interval="confidence")
       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="prediction")
       fit       lwr      upr
1 29.80359 17.565675 42.04151
2 25.05335 12.827626 37.27907
3 20.30310  8.077742 32.52846
```

Additional Activity

Question 1 *Submitted Mar 16th 2023 at 9:04:16 pm*

Which are true of simple linear regression analysis.

- [ ] the simple linear model involves only two explanatory variables;

- [x] the random errors ε_i in the simple linear model are normally and identically distributed;

- [ ] the least squares principle in the simple linear regression problem chooses $\hat{\beta}_0$ *and* $\hat{\beta}_1$ that maximize the sum of squares of the residuals $\sum_{i=1}^{n}[y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)^2]$;

- [x] the standard error for the coefficient estimate $SE(\hat{\beta}_1)$ is needed for the calculation of the test statistic value corresponding to the H_0 : there is no relationship between X and Y.

Question 2 *Submitted Mar 16th 2023 at 9:04:38 pm*

Which of the following measures or diagnostic plots can be used in assessing the accuracy of the SLR model:

- [ ] total sum of squares (TSS);

- [x] the proportion of variability in Y that can be explained using X;

- [x] residual plots;

- [ ] plot of residuals as a function of time.