Simple Linear Model

Context

Suppose we have some observations, there are various ways we can find relationship within that data. Here, we will implement the most basic statistical learning method, a simple linear model to a dataset. The dataset we will use is the `student_exam_scores.csv`, in which we will analyze the relation of student exam scores with some features/covariates.

Formulation

Linear model has the following form for 1-dimensional input and one continuous output

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

where $\epsilon \sim N(0, \sigma^2)$. More generally, we can write it in vector notation for more than one input

$$Y = \vec{X}^T \vec{\beta} + \epsilon.$$

The idea here is to find a linear line that "best fit" the given data/observations. Here, "best fit" means the one that minimizes the residual, i.e. the sum of squared difference between the value of observation and it's associated fitted value. Fortunately, `R` has a tool to implement this, which is `lm`.

Implementation

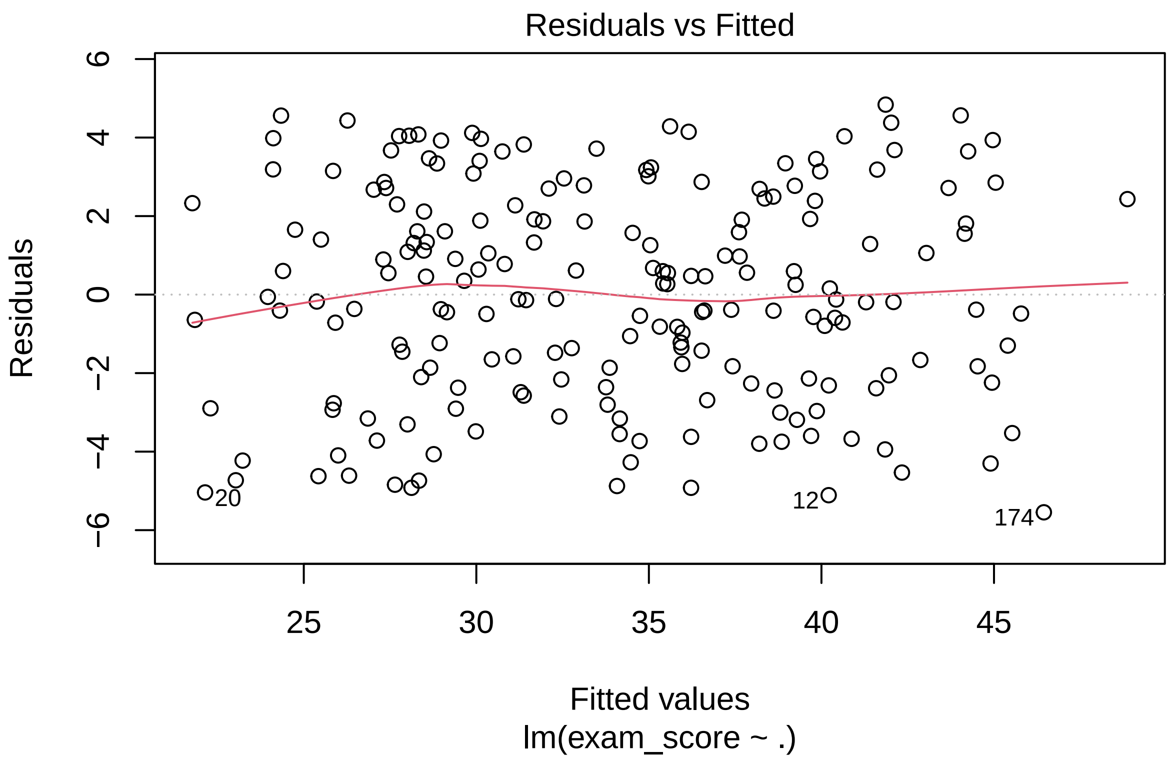
```r
student_data = read.csv2("student_exam_scores.csv", sep = ",")
important_data <- student_data[,-1]

important_data$hours_studied = as.numeric(important_data$hours_studied)
important_data$sleep_hours = as.numeric(important_data$sleep_hours)
important_data$attendance_percent = as.numeric(important_data$attendance_percent)
important_data$previous_scores = as.numeric(important_data$previous_scores)
important_data$exam_score = as.numeric(important_data$exam_score)

model = lm(exam_score ~ ., data = important_data)
summary(model)
```

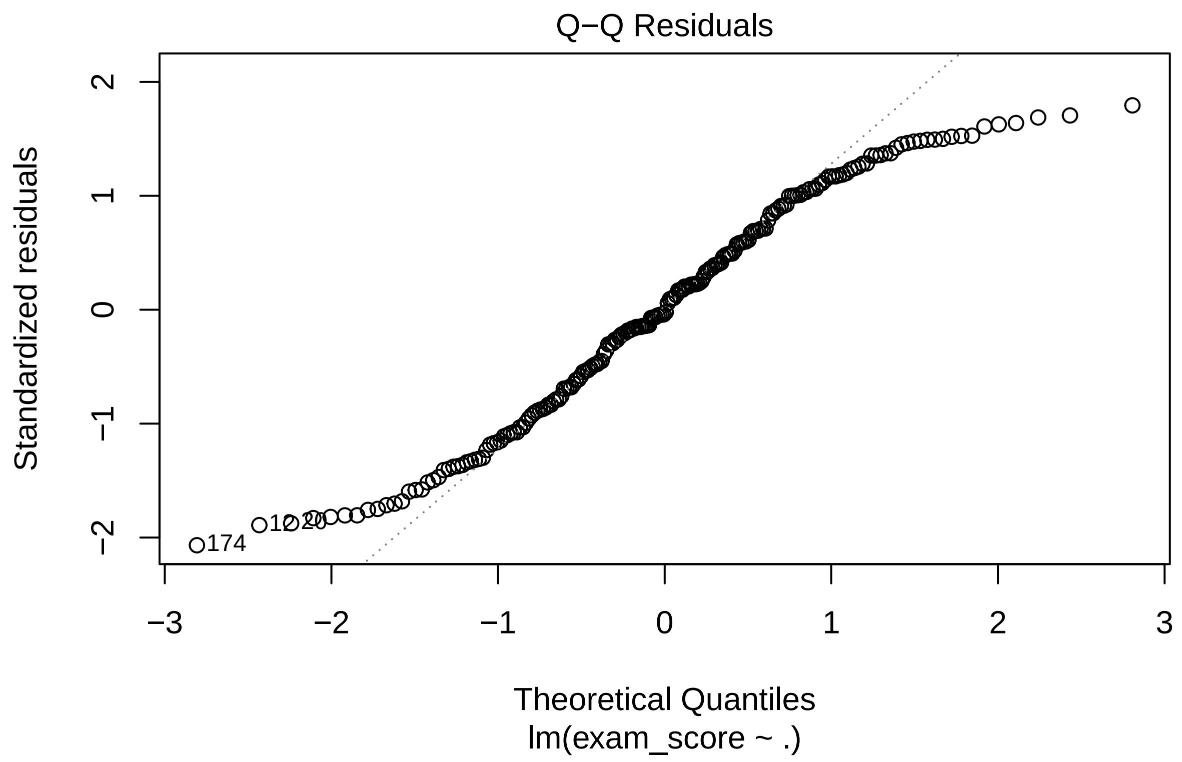
```
##
## Call:
## lm(formula = exam_score ~ ., data = important_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.5455 -2.1818 -0.0847  2.3436  4.8390
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.14209    1.67079  -1.282    0.201
## hours_studied        1.55526    0.06044  25.732  < 2e-16 ***
## sleep_hours          0.95226    0.13243   7.191 1.34e-11 ***
## attendance_percent   0.10839    0.01362   7.961 1.38e-13 ***
## previous_scores      0.17728    0.01267  13.995  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.731 on 195 degrees of freedom
## Multiple R-squared:  0.8414, Adjusted R-squared:  0.8382
## F-statistic: 258.7 on 4 and 195 DF,  p-value: < 2.2e-16
```

The table above gives information such as the estimated coefficient for the corresponding covariates, how much variation explained by the model, etc.

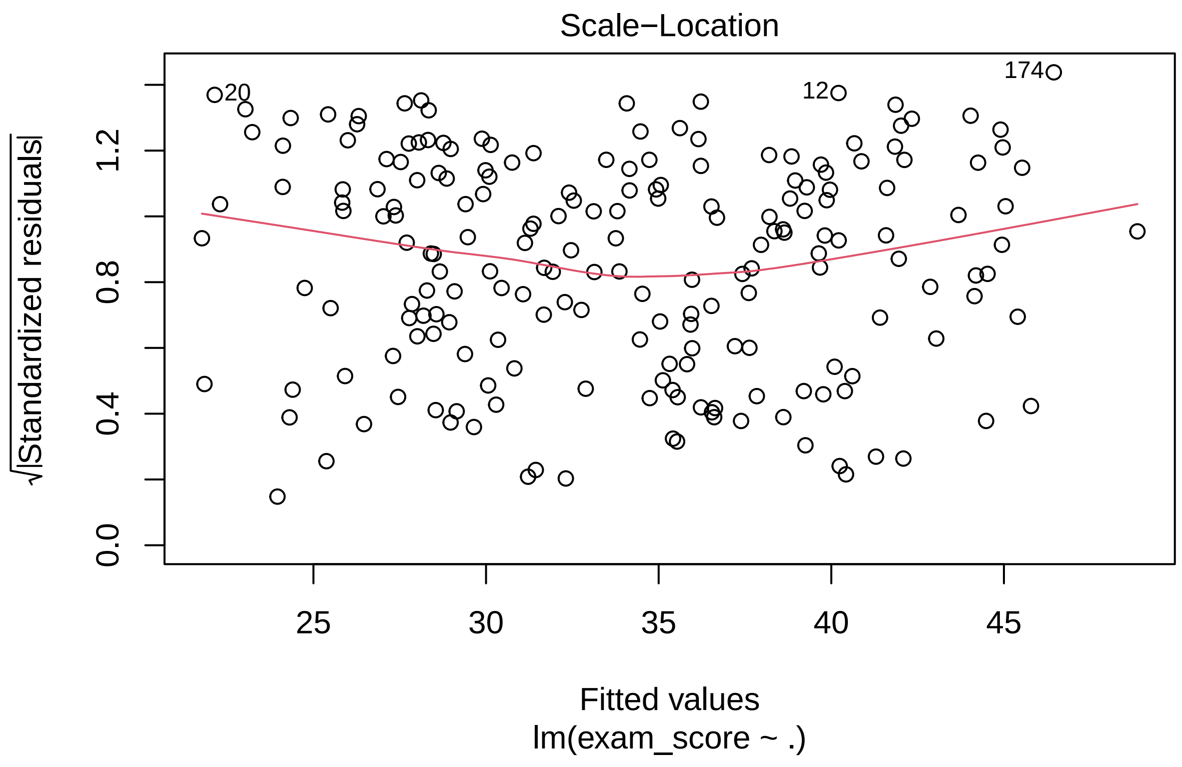
However, we need to check whether our assumption of the noise to follow a normal distribution.

```r
plot(model)
```

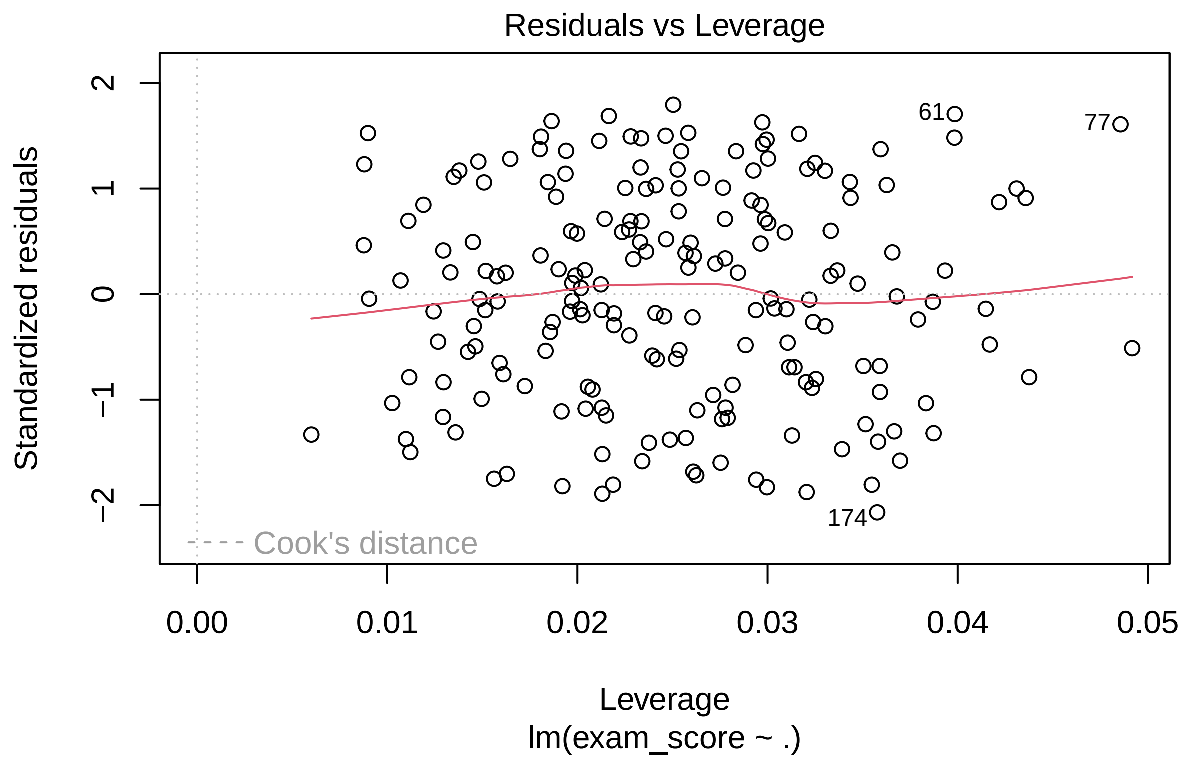




Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(exam_score ~ .)
174
12
20
−3
−2
−1
0
1
2
3



Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
25
30
35
40
45
Fitted values
lm(exam_score ~ .)
20
12
174



The residual vs fitted values plot looks fine, however the QQ-plot is somehow a bit worse.

We can also do prediction using the model we just created with example below

```r
input = data.frame(hours_studied = 12,
                   sleep_hours = 9,
                   attendance_percent = 95,
                   previous_scores = 95)
predict(model, input)
```

```
##        1
## 52.23045
```

The student is predicted to obtain 52.23 for the exam.

Limitation

- The model is too simple
-