Regression Diagnostics

Dr. Saad Laouadi

Fitting a Multiple Linear Regression

```r
data(state)
df <- as.data.frame(state.x77[, c("Murder", "Population",
                          "Illiteracy", "Income", "Frost")])
model <- lm(Murder~Population + Illiteracy + Income + Frost, data = df)
summary(model)
```

```
|
| Call:
| lm(formula = Murder ~ Population + Illiteracy + Income + Frost,
|     data = df)
|
| Residuals:
|     Min      1Q  Median      3Q     Max
| -4.7960 -1.6495 -0.0811  1.4815  7.6210
|
| Coefficients:
|              Estimate Std. Error t value Pr(>|t|)
| (Intercept) 1.235e+00  3.866e+00   0.319   0.7510
| Population  2.237e-04  9.052e-05   2.471   0.0173 *
| Illiteracy  4.143e+00  8.744e-01   4.738 2.19e-05 ***
| Income      6.442e-05  6.837e-04   0.094   0.9253
| Frost       5.813e-04  1.005e-02   0.058   0.9541
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 2.535 on 45 degrees of freedom
| Multiple R-squared:  0.567,   Adjusted R-squared:  0.5285
| F-statistic: 14.73 on 4 and 45 DF,  p-value: 9.133e-08
```

Steps After Fitting Linear Regression Models

After fitting a regression model, then reading the summary statistics of this model; sure we will have an idea about significant and non-significant variables, along with some other information such as how well the model is.

However, **how do you know whether your model is appropriate?**

The answer would be whether the model has met the **OLS** assumptions or not.

Thus, testing whether the model has satisfied the statistical assumptions is a necessary step.

Failing to satisfy the statistical assumptions will lead to an inaccurate model, in other words **useless model**. How is that?

- You may conclude that the response and the predictor **are unrelated when they are**.
- Or, you may conclude they are **relate while they are not**
- Poor model generalization, **bad predictions in the real world**.

Confidence interval

Checking the confidence interval of estimates, using `confint()` function

```
confint(model)
```

```
|                    2.5 %       97.5 %
| (Intercept) -6.552191e+00 9.0213182149
| Population   4.136397e-05 0.0004059867
| Illiteracy   2.381799e+00 5.9038743192
| Income      -1.312611e-03 0.0014414600
| Frost       -1.966781e-02 0.0208304170
```

```
tidy(model, conf.int = T)[3,c('conf.low', 'conf.high')]
```

```
| # A tibble: 1 x 2
|   conf.low conf.high
|      <dbl>     <dbl>
| 1     2.38      5.90
```

This assumes that you can be **95%** confident that the interval **[2.38, 5.90]** contains the true change in the murder rate for a **1%** change in illiteracy rate.

Note:

if a confidence interval contains **0** it means that the variable is not significant. Thus, it is not related to the response.

Regression Diagnostics

Definition: Regression diagnostics is a set of techniques which provides the necessary tools for evaluating the appropriateness of a regression model and to detect the underlying problems.

- **What are these techniques?**
  - **Visual Examination**: Plots are the first step, and they give useful insights fast.
  - **Statistical metrics or formal examination**: These metrics confirm what plots give.

It is a good practice to use both techniques to be sure of the results.

- R has many useful built-in functions such as `plot()` function.

this function has an additional argument called `which`. read the documentation for number of plots by applying **?plot.lm** in `R console`

Linear Model Assumptions

There are some assumptions that linear models should be checked for:

1- Linearity in coefficients:

The response variable is linearly related to predictors. This can be checked by plotting residuals versus fitted values. If no systematic relationship then model well fit the data. what is left is **noise**.

2- Normality:

The residuals are normally distributed. QQ-plot is usually used for checking this assumption. Note, this is useful for hypothesis testing and confidence interval.

3- Independence to errors:

This has a relationship how the data was collected, and also the understanding of the subject matter. It often exists in time series data. Another term is **autocorrelation**.

4- Homoskedasticity (constant variance):

When error variances for all observations are not the same, this is **heteroskedasticity** and the errors are **heteroskedastic** otherwise they are **homoskedastic**. Scale-Location graph is used in this case.

5 - Multi-collinearity:

This is not an assumption but we should check for it. It exists when two or more predictors are correlated. It has serious effects on results of regression models.

Outliers, High-leverage and Influencial Points

Outliers

An **outlier** or an **extreme point** is an observation whose **response variable value** is unusual, **"it does not follow the general trend of the rest of points"**. This can be clear in a simple linear regression visually.
These values, also, are not predicted well by the model, so they have large positive or negative residuals. $\mathbf{Y_i} - \widehat{\mathbf{Y}}_\mathbf{i}$ and often have **dramatic effects** on the fitted regression model. **Outliers** need to be investigated carefully, then decide later whether they should be retained or eliminated.
Generally, a data point may be an **outlier** or **extreme** with respect to its Y value, its X value, or **both**.

High-leverage

A data point has **high leverage** if it has an **extreme** predictor value, **unusual predictor values**.

Influential

A data point is deemed **influential** if it changes estimation results a great deal, such as estimates, and confidence intervals, predicted values.

Final note

Outliers and **high-leverage** points are candidates to be influential. **Not guaranteed** and must be examined to take a final decision.

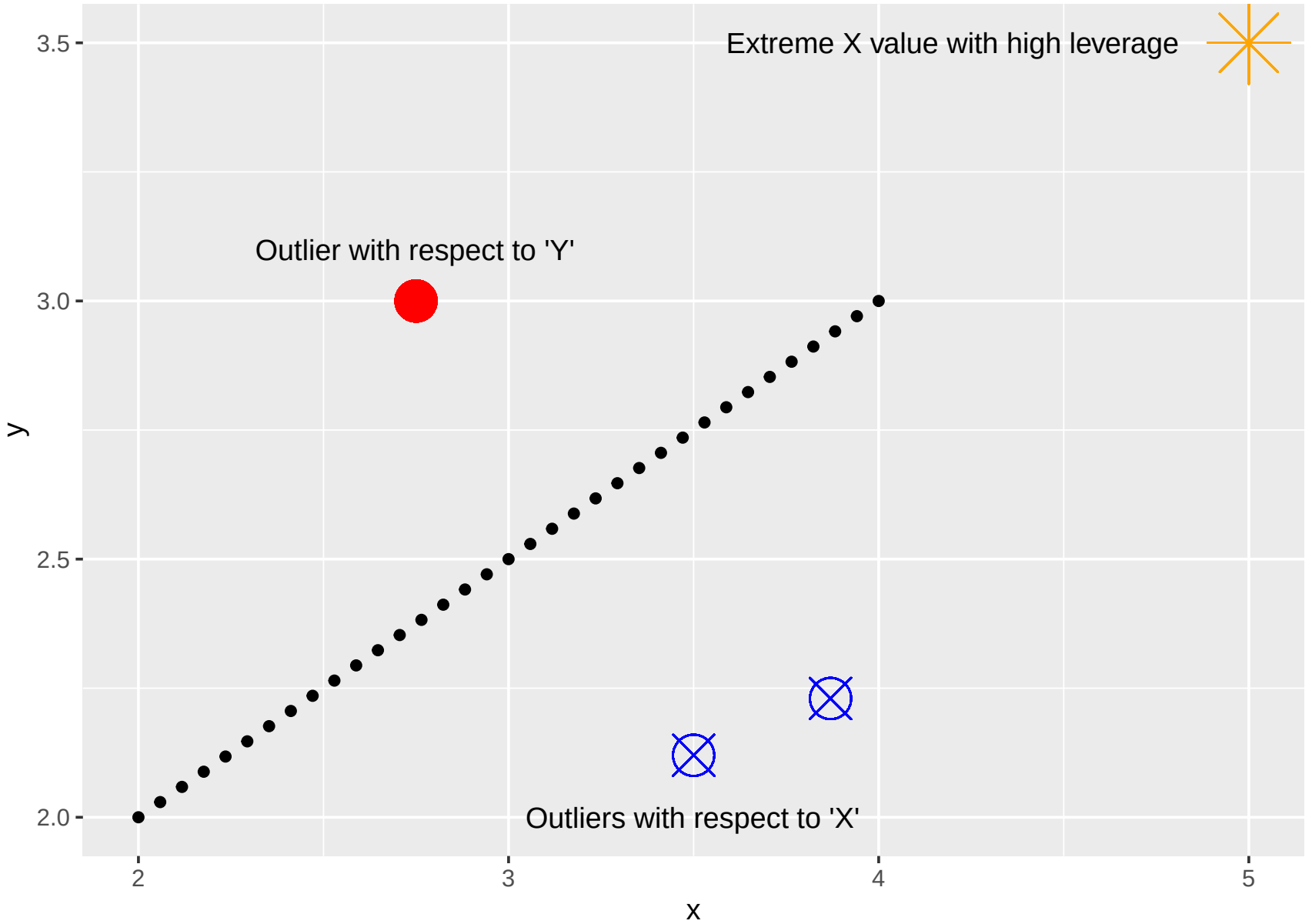
Outliers, High-leverage and Influencial Points "Continue"

```
df <- data.frame(x = c(seq(2, 4, len = 35)),
                 y = c(seq(2, 3, len = 35)))
```

```
g <- ggplot(df, aes(x, y))+
  geom_point() +
  geom_point(aes(x = 2.75, y =3), size = 7,
             shape = 19, color = "red") +
  geom_point(aes(x = 3.5, y = 2.12),
             shape = 13, size = 7, color = "blue") +
  geom_point(aes(x = 3.87, y = 2.23),
             shape = 13, size = 7, color = "blue") +
  geom_point(aes(x = 5, y = 3.5),
             shape = 8, size = 10, color = "orange") +
  annotate("text", x = 2.75, y = 3.1,
           label = "Outlier with respect to 'Y'") +
  annotate("text", x = 4.2, y = 3.5,
           label = "Extreme X value with high leverage") +
  annotate("text", x = 3.5, y =2,
           label = "Outliers with respect to 'X'")
```

Outliers, High-leverage and Influencial Points "Continue"

g



Detecting Outliers, High-leverage and Influencial Points

1. **Outlier** or **Unusual Y values**: We focus on one metric: the **Studentized Residuals**, whose principle is:
2. to delete observations one at a time
3. Refit the model on the remaining $n-1$ observations each time.
4. Compute the **deleted residuals** $d_i = y_i - \hat{y}_i$ where y_i is the actual response point for the i^{th} observation, and $\hat{y}_i$ is the predicted response for the i^{th} observation based on the estimated model with the i^{th} observation deleted.
5. Standardize the deleted residuals will give the **studentized residuals**

$$t_i = d_i / sd(d_i)$$

- The **cuttoff**: If studentized residuals of an observation greater than **2** or **3** in absolute value, we will consider this observation an outlier.

Note:

An outlier might be an **influential** point or might not.
Studentized residuals follow **t-distribution** with $n-p-1$ degrees of freedom.
If a data point has **extreme** studentized residuals value, then it is **influential**

2. **High-Leverage Points**:

Leverage is a measure of how **extreme** explanatory variable values are. It is called **hatvalues** metric as well because $\hat{y} = Hy$. H is a matrix contains only X_s. See, the H put a **hat** ˆ on Y, that is why they called it **hatvalues**.

The average hat values = (number of parameters)/sample size

$$\bar{\mathbf{h}} = \mathbf{p}/\mathbf{n} \ (p \ is \ number \ of \ parameters \ including \ the \ intercept)$$

or

$$\bar{\mathbf{h}} = \mathbf{p} + \mathbf{1}/\mathbf{n}$$

if the intercept is β_0

High-leverage cuttoff: An observation with **2** (in large samples) or **3** (in small samples) times the **average hat values** should be examined.

Note

Leverage quantifies the potential for a data point to be influential

3 Influential Data Points

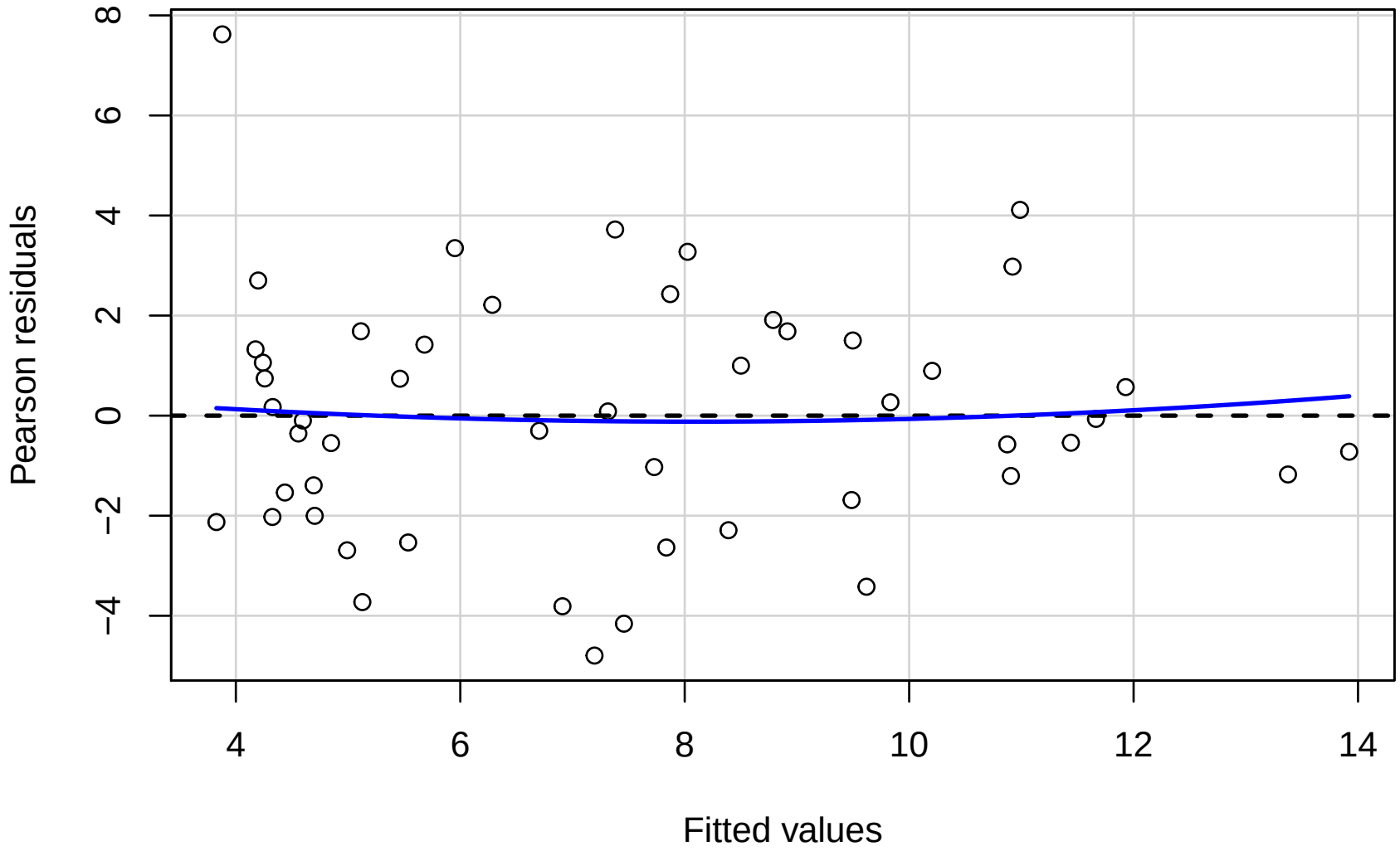
There are several metrics to identify influential observations. We focus on on statistical metrics and one plot.

To clarify more: **influence** means how much the model would change if we left one observation out of the dataset when modelling.

- **Cook's Distance** or **D statistic**: is a **leave-one-out** metric. It is a common metric to identify influential observations.
- **Cook's D cuttoff**: Values with cooks.d greater than **1** are likely to be influential.
- **Added-variable Plot** is another way to determine influential observations. This plot is generated as follows, for each predictor X_k, plot the residuals form regressing the response variable on the other $k-1$ predictors versus the residuals from regressing X_k on the other $k-1$ predictors. use `avPlots()` function from `car` package.

Checking Assumptions: Linearity

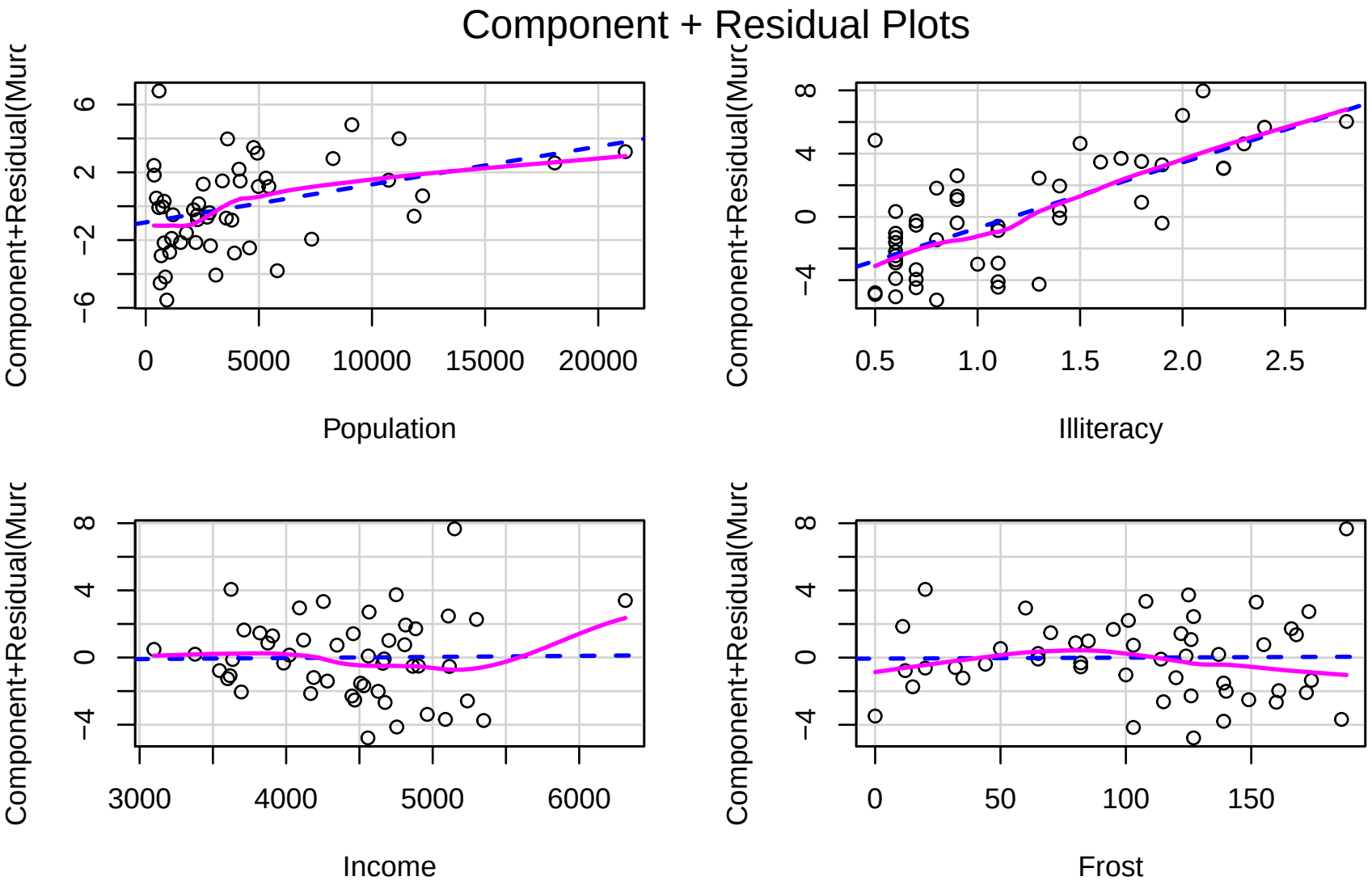
```
residualPlot(model)
```



If there is no systematic relationship (random spread) between the residuals and the predicted values then **linearity** assumption is met.

Checking for Linearity with component plus residual plots

```
crPlots(model)
```

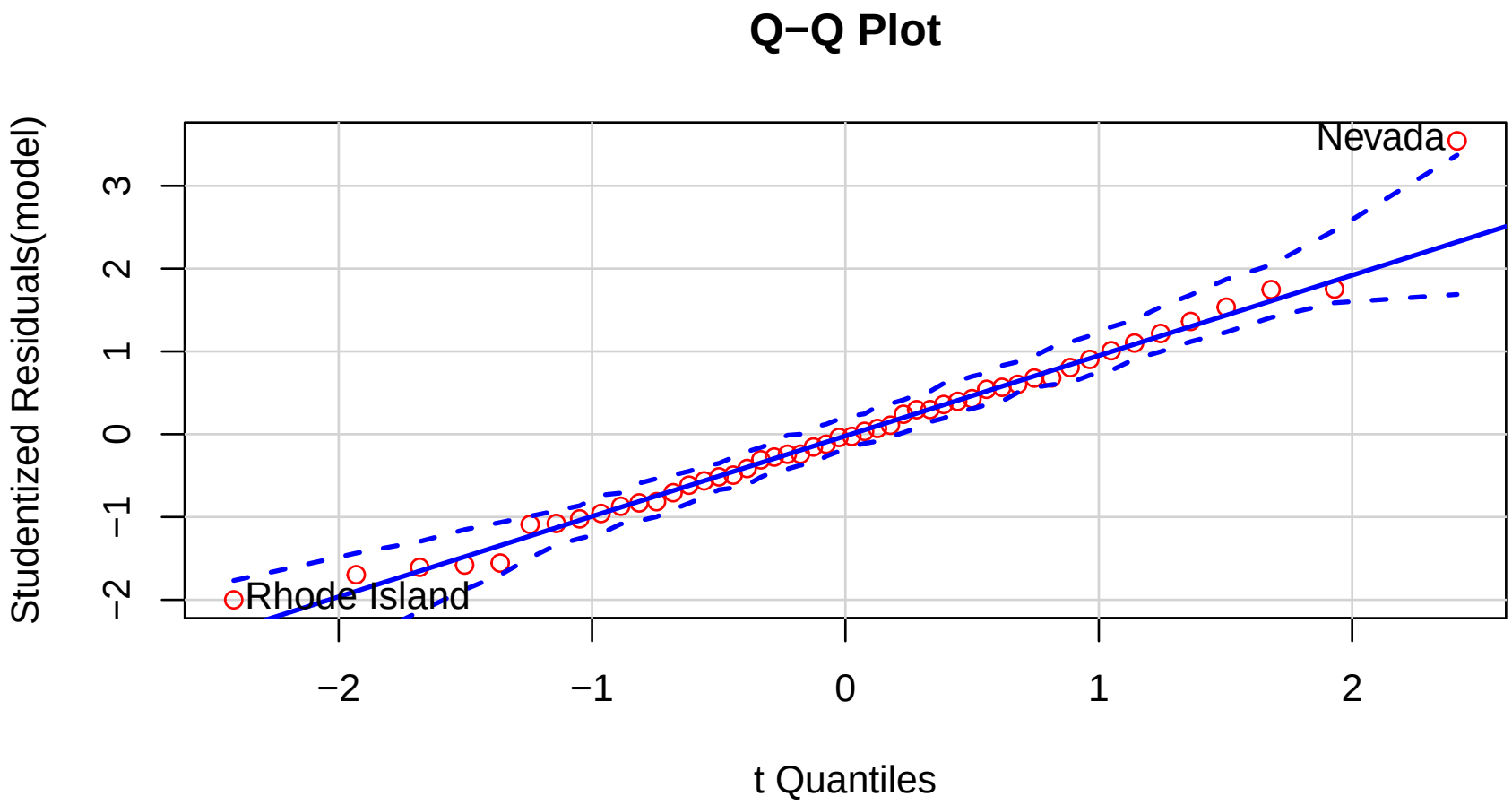


Nonlinearity in any plot suggests the wrong functional form of the predictor, thus, transformation might be needed such as $log(x)$ or adding x^2 or higher order.

Checking Assumptions: Normality

`qqPlot()` function plots **studentized residuals** vs. t-distribution with $n-k-1$ degrees of freedom.

```
qqPlot(model, labels = row.names(df),
       simulate = T, main = "Q-Q Plot", col = "red")
```




```
|        Nevada Rhode Island
|            28           39
```

Checking the Point 'Nevada'

```
df['Nevada', ]
```

```
|        Murder Population Illiteracy Income Frost
| Nevada   11.5        590        0.5   5149   188
```

```
cbind(fitted = fitted(model)['Nevada'],
      resid = residuals(model)['Nevada'],
      std_resid = rstudent(model)['Nevada'])
```

```
|           fitted    resid std_resid
| Nevada 3.878958 7.621042  3.542929
```

The model predicted **3.9%** murder rate while in fact it is **11.5%**. This point is an outlier and needs examination

Checking the Independence of Errors

Although correlated errors appear in time series more often, we can check for that in multiple-regression using **Durbin_Watson test** as follows

```
durbinWatsonTest(model)
```

```
|  lag Autocorrelation D-W Statistic p-value
|    1       -0.2006929      2.317691   0.286
|  Alternative hypothesis: rho != 0
```

Recall that:

- Null hypothesis: rho $= 0$ . . . . . No autocorrelation
- Alternative hypothesis: rho != 0 . . . . . Autocorrelation

In our case: p.value > 0.05, We cannot reject the null hypothesis of no autocorrelation.

Checking Homoskedasticity: Statistic test

Recall:

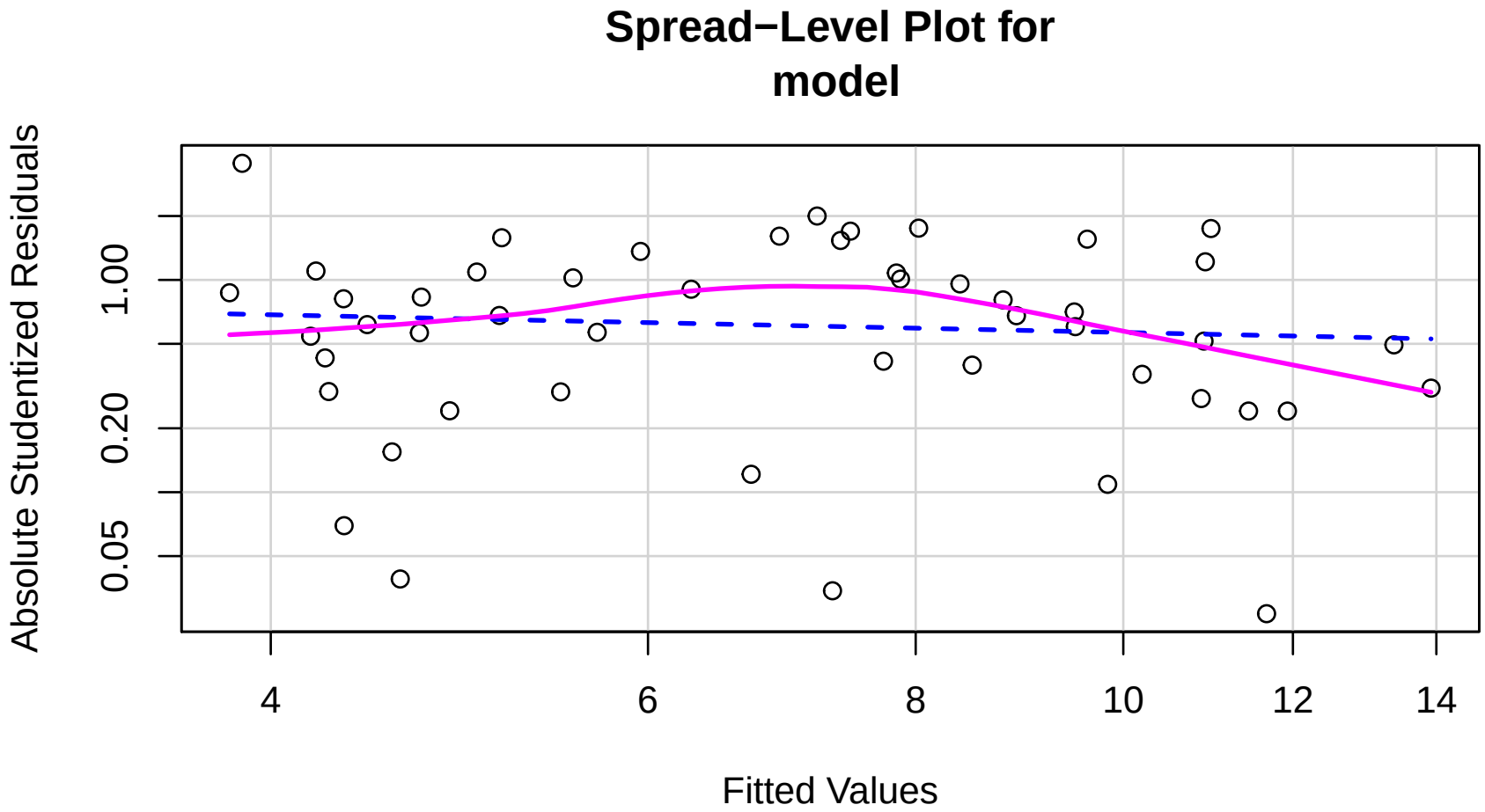
- Null H: constant variance
- Alt H: Heteroskedasticity

```
ncvTest(model)
```

```
| Non-constant Variance Score Test
| Variance formula: ~ fitted.values
| Chisquare = 1.746514, Df = 1, p = 0.18632
```

Checking Homoskedasticity: Checking by a plot

```
spreadLevelPlot(model)
```




```
|
| Suggested power transformation:  1.209626
```

Checking Multi-collinearity

Multicollinearity can be checked using **variance inflation factor (VIF)**. If $\sqrt{(vif)} > 2$ indicates a multicollinearity.

```
vif(model)
```

```
| Population Illiteracy     Income      Frost
|   1.245282   2.165848   1.345822   2.082547
```

```
sqrt(vif(model)) > 2
```

```
| Population Illiteracy     Income      Frost
|      FALSE      FALSE      FALSE      FALSE
```

Global Validation Linear Model Assumptions with gvlma() package

Global validation of linear model assumptions, kurtosis, skewness and heteroskedasticity.

```
gvlma(model)
```

```
|
| Call:
| lm(formula = Murder ~ Population + Illiteracy + Income + Frost,
|     data = df)
|
| Coefficients:
| (Intercept)   Population   Illiteracy       Income        Frost
|   1.235e+00    2.237e-04    4.143e+00    6.442e-05    5.813e-04
|
|
| ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
| USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
| Level of Significance =  0.05
|
| Call:
|  gvlma(x = model)
|
|                     Value p-value                Decision
| Global Stat        2.7728  0.5965 Assumptions acceptable.
| Skewness           1.5374  0.2150 Assumptions acceptable.
| Kurtosis           0.6376  0.4246 Assumptions acceptable.
| Link Function      0.1154  0.7341 Assumptions acceptable.
| Heteroscedasticity 0.4824  0.4873 Assumptions acceptable.
```

It seemed all assumptions are met (p=0.597) if ($p<0.05$) examine data.

Checking Outliers

`outlierTest()` function reports the Bonferroni adjusted p-value for the largest absolute studentized residuals.

```
outlierTest(model)
```

```
|        rstudent unadjusted p-value Bonferroni p
| Nevada 3.542929         0.00095088     0.047544
```

Null Hypothesis: A point is not outlier

Alt Hypothesis: A point is outlier

Nevada is identified as an outlier.

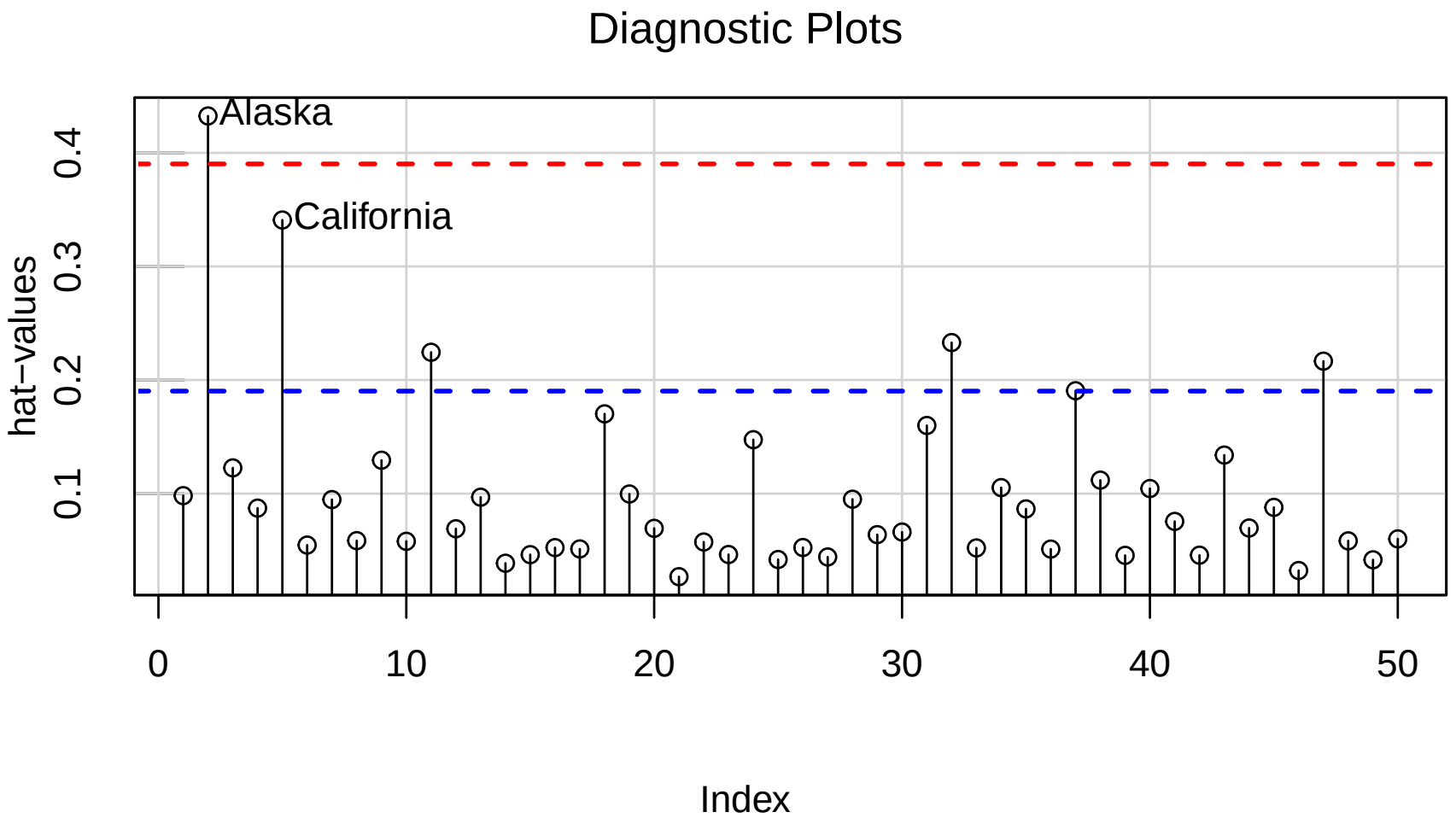
High-Leverage Points

Recall:

- The cutoff is : **+2** or **+3** times the average hatvalues.

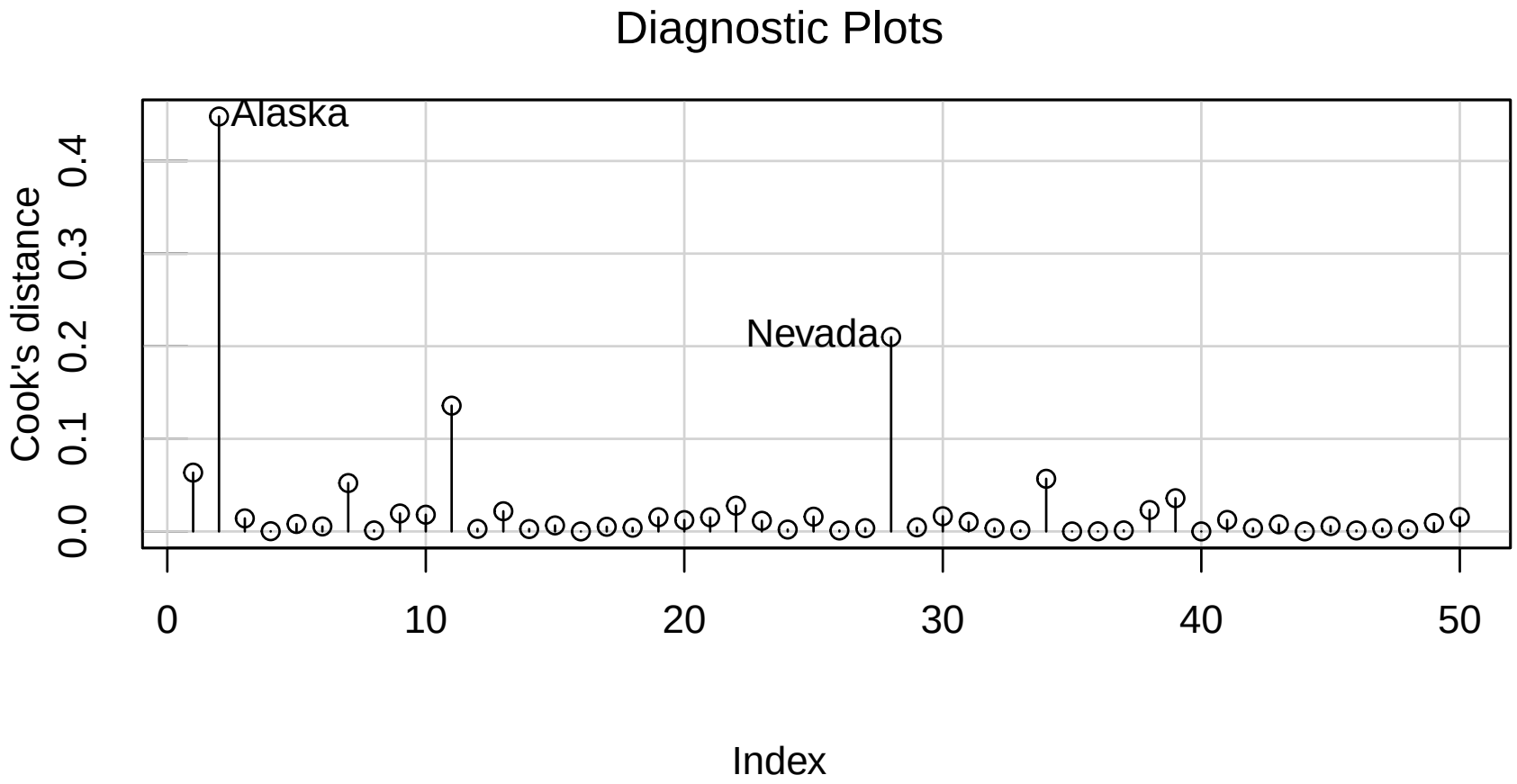
we will be using `influenceIndexPlot()` function

```
influenceIndexPlot(model, vars = "hat")
abline(h = c(2,3) * mean(hatvalues(model)), col = c("blue", "red"),
       lty = 2, lwd = 2)
```



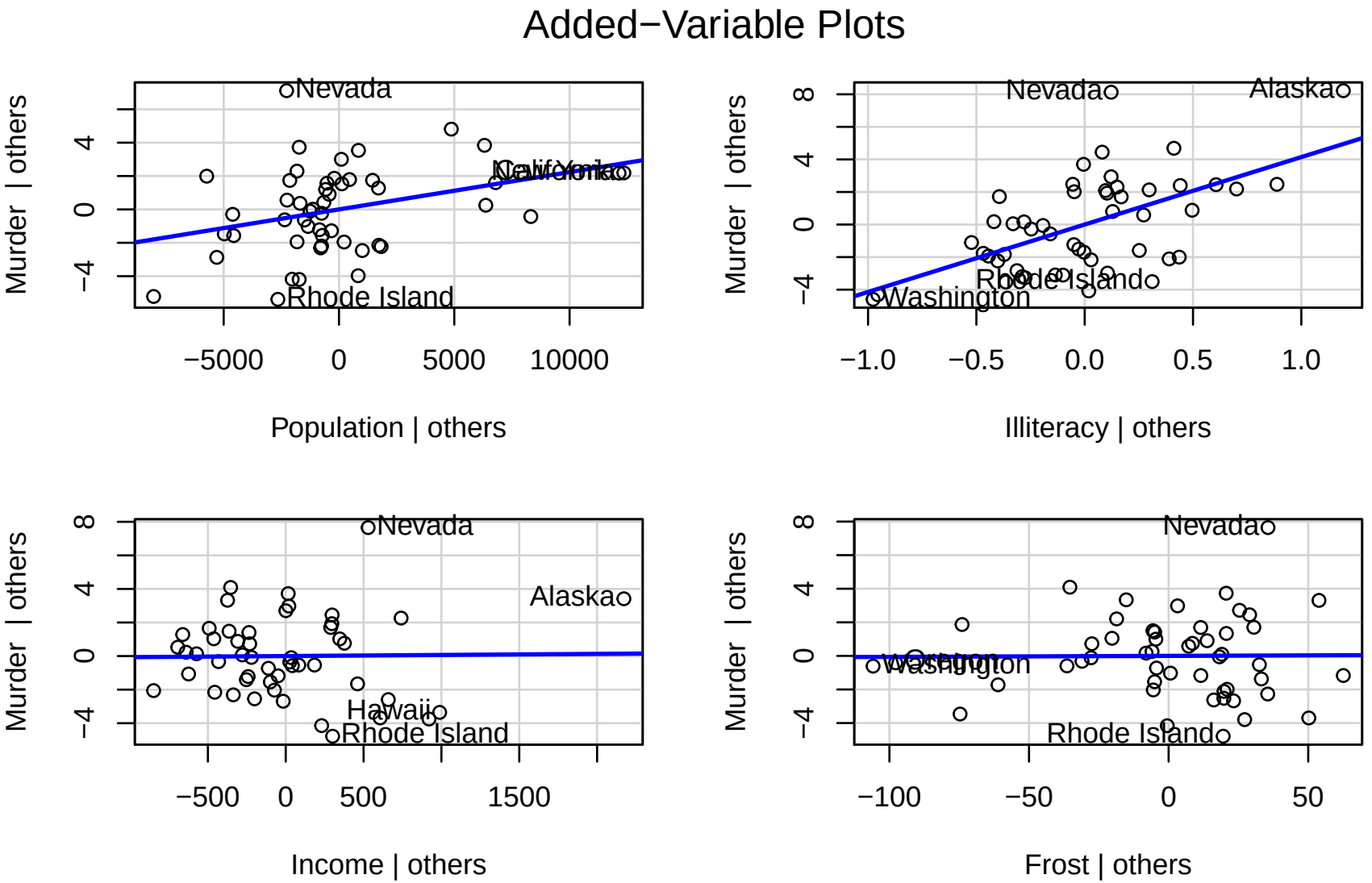
Influential Data Points

```
influenceIndexPlot(model, vars = "Cook")
```



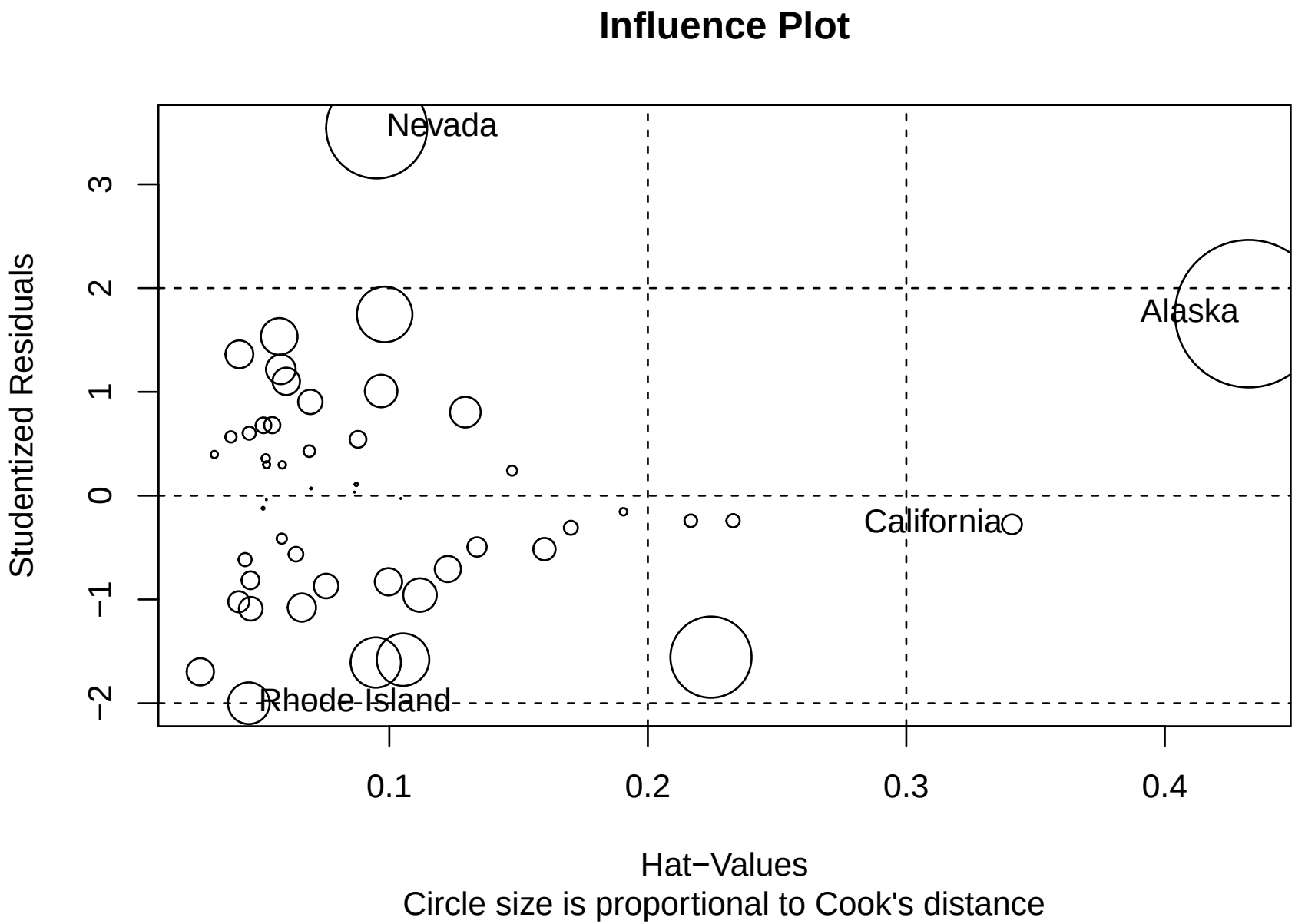
Influential Data Points (Continue): Added Plot

```
avPlots(model, ask = F)
```



Influential Data Points (Continue): Influential Plot

```
influencePlot(model, main = "Influence Plot",
              sub = "Circle size is proportional to Cook's distance")
```



Reffiting the Model Without the High-leverage Point

```
whichNames("Nevada", df) # 28
```

```
| Nevada
|     28
```

```
model_2 <- update(model, subset=-c(28))
cbind(tidy(model)$estimate, tidy(model_2)$estimate)
```

```
|                [,1]           [,2]
| [1,] 1.2345634112  3.0369896814
| [2,] 0.0002236754  0.0002480662
| [3,] 4.1428365903  4.0204779097
| [4,] 0.0000644247 -0.0002599328
| [5,] 0.0005813055 -0.0041333484
```

Compare Coefficients

```
compareCoefs(model,model_2)
```

```
| Calls:
| 1: lm(formula = Murder ~ Population + Illiteracy + Income + Frost, data =
|   df)
| 2: lm(formula = Murder ~ Population + Illiteracy + Income + Frost, data =
|   df, subset = -c(28))
|
|               Model 1   Model 2
| (Intercept)      1.23      3.04
| SE               3.87      3.49
|
| Population   2.24e-04  2.48e-04
| SE           9.05e-05  8.10e-05
|
| Illiteracy      4.143     4.020
| SE              0.874     0.781
|
| Income       6.44e-05 -2.60e-04
| SE           6.84e-04  6.17e-04
|
| Frost        0.000581 -0.004133
| SE           0.010054  0.009066
|
```