

4.6: When Things Go Wrong

Sometimes when we try to develop a model using the backward elimination process, we get results that do not appear to make any sense. For an example, let's try to develop a multi-factor regression model for the Int1992 data using this process. As before, we begin by including all of the potential predictors from Table 4.1 in the model. When we try that for Int1992, however, we obtain the following result:

```
> int92.lm<-lm(nperf ~ clock + threads + cores + transistors + dieSize + voltage + fe
channel + F04delay + L1icache + sqrt(L1icache) + L1dcache + sqrt(L1dcache) + L2cache
> summary(int92.lm)
```

Call:

```
lm(formula = nperf ~ clock + threads + cores + transistors +
    dieSize + voltage + featureSize + channel + F04delay +
    L1icache + sqrt(L1icache) + L1dcache + sqrt(L1dcache) +
    L2cache + sqrt(L2cache))
```

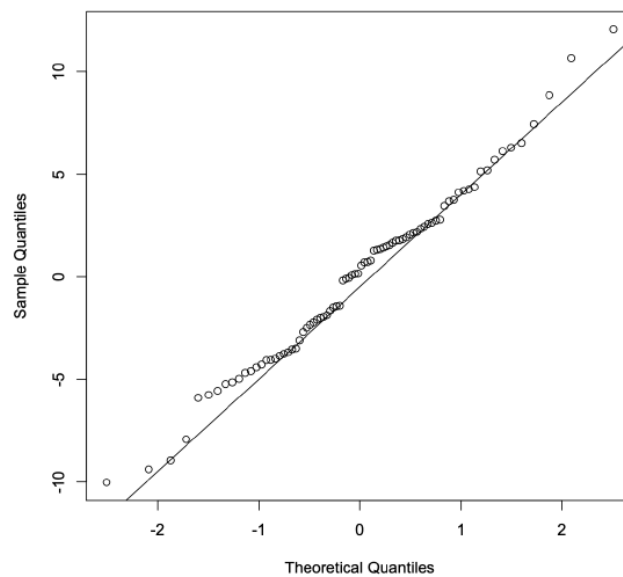
Residuals:

```
    14     15     16     17     18     19
0.4096 1.3957 -2.3612 0.1498 -1.5513 1.9575
```

Coefficients: (14 not defined because of singularities)

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-25.93278	6.56141	-3.952	0.0168 *
clock	0.35422	0.02184	16.215	8.46e-05 ***
threads	NA	NA	NA	NA
cores	NA	NA	NA	NA
transistors	NA	NA	NA	NA
dieSize	NA	NA	NA	NA
voltage	NA	NA	NA	NA
featureSize	NA	NA	NA	NA
channel	NA	NA	NA	NA
F04delay	NA	NA	NA	NA
L1icache	NA	NA	NA	NA
sqrt(L1icache)	NA	NA	NA	NA
L1dcache	NA	NA	NA	NA
sqrt(L1dcache)	NA	NA	NA	NA
L2cache	NA	NA	NA	NA
sqrt(L2cache)	NA	NA	NA	NA

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 1.868 on 4 degrees (72 observations deleted due to missingne
Multiple R-squared: 0.985, Adjusted R-squared: 0.9813 F-statistic: 262.9 on 1 and 4 I



Notice that every predictor but `clock` has `NA` for every entry. Furthermore, we see a line that says that fourteen coefficients were “not defined because of singularities.” This statement means that R could not compute a value for those coefficients because of some anomalies in the data. (More technically, it could not invert the matrix used in the least-squares minimization process.)

The first step toward resolving this problem is to notice that 72 observations were deleted due to “missingness,” leaving only four degrees of freedom. We use the function `nrow(int92.dat)` to determine that there are 78 total rows in this data frame. These 78 separate observations sum up to the two predictors used in the model, plus four degrees of freedom, plus 72 deleted rows. When we tried to develop the model using `lm()`, however, some of our data remained unused.

To determine why these rows were excluded, we must do a bit of sanity checking to see what data anomalies may be causing the problem. The function `table()` provides a quick way to summarize a data vector, to see if anything looks obviously out of place. Executing this function on the `clock` column, we obtain the following:

```
> table(clock)
clock
48  50  60  64  66  70  75  77  80  85  90  96  99 100 101 110
   118 120 125 133 150 166 175 180 190 200 225 231 233 250 266
   275 291 300 333 350
1   3   4   1   5   1   4   1   2   1   2   1   2  10   1   1
1   3   4   4   3   2   2   1   1   4   1   1   2   2   2   1   1   1   1
```

The top line shows the unique values that appear in the column. The list of numbers directly below that line is the count of how many times that particular value appeared in the column. For example, 48 appeared once, while 50 appeared three times and 60 appeared four times. We see a reasonable range of values with minimum (48) and maximum (350) values that are not unexpected. Some of the values occur only once; the most frequent value occurs ten times, which again does not seem unreasonable. In short, we do not see anything obviously amiss with these results. We conclude that the problem likely is with a different data column.

Executing the `table()` function on the next column in the data frame threads produces this output:

```
> table(threads)
threads
1
78
```

Aha! Now we are getting somewhere. This result shows that all of the 78 entries in this column contain the same value: 1. An input factor in which all of the elements are the same value has no predictive power in a regression model. If every row has the same value, we have no way to distinguish one row from another. Thus, we conclude that `threads` is not a useful predictor for our model and we eliminate it as a potential predictor as we continue to develop our Int1992 regression model.

We continue by executing `table()` on the column labeled `cores`. This operation shows that this column also consists of only a single value, 1. Using the `update()` function to eliminate these two predictors from the model gives the following:

```
> int92.lm <- update(int92.lm, ~. threads cores)
> summary(int92.lm)
Call:
lm(formula = nperf ~ clock + transistors + dieSize + voltage +
    featureSize + channel + F04delay + L1icache + sqrt(L1icache) +
    L1dcache + sqrt(L1dcache) + L2cache + sqrt(L2cache))

Residuals:
    14     15     16     17     18     19 
0.4096  1.3957 -2.3612  0.1498 -1.5513  1.9575 

Coefficients: (12 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -25.93278    6.56141   -3.952   0.0168 *
clock          0.35422    0.02184   16.215 8.46e-05 ***
transistors           NA          NA      NA      NA
dieSize              NA          NA      NA      NA
voltage              NA          NA      NA      NA
featureSize          NA          NA      NA      NA
channel              NA          NA      NA      NA
F04delay             NA          NA      NA      NA
L1icache             NA          NA      NA      NA
sqrt(L1icache)       NA          NA      NA      NA
L1dcache             NA          NA      NA      NA
sqrt(L1dcache)       NA          NA      NA      NA
L2cache            NA          NA      NA      NA
sqrt(L2cache)       NA          NA      NA      NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.868 on 4 degrees of freedom (72 observations deleted due to missing values)
Multiple R-squared:  0.985, Adjusted R-squared:  0.9813 F-statistic: 262.9 on 1 and 4 DF, p-value: 1.11e-10
```

Unfortunately, eliminating these two predictors from consideration has not solved the problem. Notice that we still have only four degrees of freedom, because 72 observations were again eliminated. This small number of degrees of freedom indicates that there must be at least one more column with insufficient data.

By executing `table()` on the remaining columns, we find that the column labeled `L2cache` has only three unique values, and that these appear in a total of only ten rows:

```
> table(L2cache)
L2cache
96 256 512
 6   2   2
```

Although these specific data values do not look out of place, having only three unique values can make it impossible for `lm()` to compute the model coefficients. Dropping `L2cache` and `sqrt(L2cache)` as potential predictors finally produces the type of result we expect:

```
> int92.lm <- update(int92.lm, ~. L2cache sqrt(L2cache))
> summary(int92.lm)
```

Call:

```
lm(formula = nperf ~ clock + transistors + dieSize + voltage +
    featureSize + channel + F04delay + L1icache + sqrt(L1icache) +
    L1dcache + sqrt(L1dcache))
```

Residuals:

Min	1Q	Median	3Q	Max
-7.3233	-1.1756	0.2151	1.0157	8.0634

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-58.51730	17.70879	-3.304	0.00278 **
clock	0.23444	0.01792	13.084	6.03e-13 ***
transistors	-0.32032	1.13593	-0.282	0.78018
dieSize	0.25550	0.04800	5.323	1.44e-05 ***
voltage	1.66368	1.61147	1.032	0.31139
featureSize	377.84287	69.85249	5.409	1.15e-05 ***
channel	-493.84797	88.12198	-5.604	6.88e-06 ***
F04delay	0.14082	0.08581	1.641	0.11283
L1icache	4.21569	1.74565	2.415	0.02307 *
sqrt(L1icache)	-12.33773	7.76656	-1.589	0.12425
L1dcache	-5.53450	2.10354	-2.631	0.01412 *
sqrt(L1dcache)	23.89764	7.98986	2.991	0.00602 **

 Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
 Residual standard error: 3.68 on 26 degrees of freedom (40 observations deleted due to missing values)
 Multiple R-squared: 0.985, Adjusted R-squared: 0.9786 F-statistic: 155 on 11 and 26 Df, p-value: 1.1e-26

We now can proceed with the normal backward elimination process. We begin by eliminating the predictor that has the largest p-value above our preselected threshold, which is `transistors` in this case. Eliminating this predictor gives the following:

```
> int92.lm <update(int92.lm, .~. -transistors)
> summary(int92.lm)

Call:
lm(formula = nperf ~ clock + dieSize + voltage + featureSize +
    channel + F04delay + L1icache + sqrt(L1icache) + L1dcache +
    sqrt(L1dcache))

Residuals:
    Min       1Q   Median       3Q      Max
-13.2935  -3.6068   -0.3808   2.4535  19.9617

Coefficients:
              Estimate      Std. Error    t value    Pr(>|t|)
(Intercept)  -16.73899       24.50101    -0.683    0.499726
clock         0.19330        0.02091     9.243    2.77e-10 ***
dieSize       0.11457        0.02728     4.201    0.000219 ***
voltage       0.40317        2.85990     0.141    0.888834
featureSize   11.08190       104.66780     0.106    0.916385
channel      -37.23928       104.22834    -0.357    0.723379
F04delay     -0.13803        0.14809    -0.932    0.358763
L1icache      7.84707        3.33619     2.352    0.025425 *
sqrt(L1icache) -16.28582      15.38525    -1.059    0.298261
L1dcache     -14.31871       2.94480    -4.862    3.44e-05 ***
sqrt(L1dcache) 48.26276       9.41996     5.123    1.64e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.528 on 30 degrees of freedom (37 observations deleted due
Multiple R-squared: 0.9288, Adjusted R-squared: 0.9051 F-statistic: 39.13 on 10 and 30
```

After eliminating this predictor, however, we see something unexpected. The p-values for `voltage` and `featureSize` increased dramatically. Furthermore, the adjusted R-squared value dropped substantially, from 0.9786 to 0.9051. These unexpectedly large changes make us suspect that `transistors` may actually be a useful predictor in the model even though at this stage of the backward elimination process it has a high p-value. So, we decide to put `transistors` back into the model and instead drop `voltage`, which has the next highest p-value. These changes produce the following result:

```
> int92.lm <update(int92.lm, .~. +transistors -voltage)
> summary(int92.lm)

Call:
lm(formula = nperf ~ clock + dieSize + featureSize + channel +
    F04delay + L1icache + sqrt(L1icache) + L1dcache +
    sqrt(L1dcache) +
    transistors)

Residuals:
    Min       1Q   Median       3Q      Max
-10.0828  -1.3106   0.1447   1.5501   8.7589
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-50.28514	15.27839	-3.291	0.002700	**
clock	0.21854	0.01718	12.722	3.71e-13	***
dieSize	0.20348	0.04401	4.623	7.77e-05	***
featureSize	409.68604	67.00007	6.115	1.34e-06	***
channel	-490.99083	86.23288	-5.694	4.18e-06	***
F04delay	0.12986	0.09159	1.418	0.167264	
L1icache	1.48070	1.21941	1.214	0.234784	
sqrt(L1icache)	-5.15568	7.06192	-0.730	0.471413	
L1dcache	-0.45668	0.10589	-4.313	0.000181	***
sqrt(L1dcache)	4.77962	2.45951	1.943	0.062092	.
transistors	1.54264	0.88345	1.746	0.091750	.

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.96 on 28 degrees of freedom (39 observations deleted due to missing values)
Multiple R-squared: 0.9813, Adjusted R-squared: 0.9746 F-statistic: 146.9 on 10 and 28 df, p-value: 1.49e-26

The adjusted R-squared value now is 0.9746, which is much closer to the adjusted R-squared value we had before dropping `transistors`. Continuing with the backward elimination process, we first drop `sqrt(L1icache)` with a p-value of 0.471413, then `F04delay` with a p-value of 0.180836, and finally `sqrt(L1dcache)` with a p-value of 0.071730.

After completing this backward elimination process, we find that the following predictors belong in the final model for Int1992:

clock transistors dieSize featureSize
channel L1icache L1dcache

As shown below, all of these predictors have p-values below our threshold of 0.05. Additionally, the adjusted R-square looks quite good at 0.9722.

```
> int92.lm <update(int92.lm, .~. -sqrt(L1dcache))
> summary(int92.lm)

Call:
lm(formula = nperf ~ clock + dieSize + featureSize + channel +
    L1icache + L1dcache + transistors, data = int92.dat)

Residuals:
    Min       1Q   Median       3Q      Max
-10.1742  -1.5180   0.1324   1.9967  10.1737

Coefficients:
              Estimate      Std. Error    t value    Pr(>|t|)
(Intercept)   -34.17260         5.47413     -6.243    6.16e-07 ***
clock           0.18973         0.01265     15.004    9.21e-16 ***
dieSize        0.11751         0.02034      5.778    2.31e-06 ***
featureSize   305.79593        52.76134      5.796    2.20e-06 ***
channel       -328.13544        53.04160     -6.186    7.23e-07 ***
L1icache       0.78911         0.16045      4.918    2.72e-05 ***
L1dcache      -0.23335         0.03222     -7.242    3.80e-08 ***
transistors    3.13795         0.51450      6.099    9.26e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.141 on 31 degrees of freedom (39 observations deleted due
Multiple R-squared: 0.9773, Adjusted R-squared: 0.9722 F-statistic: 191 on 7 and 31 Df
```

This example illustrates that you cannot always look at only the p-values to determine which potential predictors to eliminate in each step of the backward elimination process. You also must be careful to look at the broader picture, such as changes in the adjusted R-squared value and large changes in the p-values of other predictors, after each change to the model.

This page titled [4.6: When Things Go Wrong](#) is shared under a [CC BY-NC 4.0](#) license and was authored, remixed, and/or curated by [David Lilja](#) ([University of Minnesota Libraries Publishing](#)) via [source content](#) that was edited to the style and standards of the LibreTexts platform.