Resolving Discrepancies in R-Squared and F-Statistic Using Design Matrices in R Linear Models

Yichen Han, Eugen Gorich

2024-06-11

TL;DR

In this article, we want to dig deeper into an intricate problem we ran into while working on the exercise sheet 4. We used **design matrices** as input for linear models and noticed very high R-squared and F-statistic. We found that the `lm()` function in R does not handle design matrices automatically. Similar issue occurs when using **mean value models**. Based on *slide 64 and 66*, it is due to a different formula applied to SSM when the model is not set to include intercept. In this case, the resulting R-squared and F-Statistic are not interpretable. We then propose a solution to achieve the R-squared and F-Statistic that meaningfully describe the model fit.

We first make some necessary preparations and load the data set.

```r
library(dplyr)
load("wdi.Rdata")
# log CO2emission
wdi$CO2emission <- log10(wdi$CO2emission)
wdi2013 <- wdi[wdi$year == 2013,]
# delete row if continent or CO2emission is NA
wdi2013 <- wdi2013[complete.cases(wdi2013[,c('continent', 'CO2emission')]),]
wdi2013$continent <- as.factor(wdi2013$continent)
# relevel, continent == Europe as reference level
wdi2013_ref <- wdi2013
wdi2013_ref$continent <- relevel(wdi2013_ref$continent, ref = "Europe")
```

Problem 1: Discrepancies in R^2 and F-Statistic with Design Matrix Input

When dealing with, for example, reference coding, we have proposed two inherently identical approaches to model the data.

Model Without Custom Coding

The first approach is to set the independent categorical variable directly into the model formula. If not otherwise leveled, e.g. with `relevel()`, the first level of the factor is taken as the reference level.

In the following, we use the `continent` variable as an example to model the `log10`-transformed `CO2emission` data. We set our reference level to `Europe` for all models.

```r
lm_nocoding <- lm(CO2emission ~ continent, data = wdi2013_ref)
summary(lm_nocoding)
```

```
##
## Call:
## lm(formula = CO2emission ~ continent, data = wdi2013_ref)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.72508 -0.68659  0.00954  0.55617  2.93454
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)          4.4355     0.1469  30.191  < 2e-16 ***
## continentAfrica     -0.8831     0.1969  -4.485 1.23e-05 ***
## continentAmericas   -0.6575     0.2078  -3.164  0.00179 **
## continentAsia        0.1753     0.2004   0.875  0.38261
## continentOceania    -1.7337     0.2821  -6.145 4.22e-09 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.9634 on 201 degrees of freedom
## Multiple R-squared:  0.263,  Adjusted R-squared:  0.2484
## F-statistic: 17.93 on 4 and 201 DF,  p-value: 1.311e-12
```

Note that the `lm_noncoding` model automatically uses reference coding, with Intercept showing the mean of group `Europe`. The R^2 and F-statistic are calculated according to *slide 64* as 0.263 and 17.93 respectively.

Model Using Design Matrix

The second approach is to use a design matrix as input for the model. We can create the design matrix with the `model.matrix()` function. The design matrix is then used as input for the model.

```r
# define the design matrix
desmat_ref <- model.matrix(~continent, data = wdi2013_ref)
colnames(desmat_ref) <- sub("continent", "", colnames(desmat_ref))
# fit the model
lm_ref <- lm(CO2emission ~ desmat_ref - 1, data = wdi2013_ref)
## -1 to avoid repeating intercept
summary(lm_ref)
```

```
##
## Call:
## lm(formula = CO2emission ~ desmat_ref - 1, data = wdi2013_ref)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.72508 -0.68659  0.00954  0.55617  2.93454
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## desmat_ref(Intercept)   4.4355     0.1469  30.191  < 2e-16 ***
## desmat_refAfrica       -0.8831     0.1969  -4.485 1.23e-05 ***
## desmat_refAmericas     -0.6575     0.2078  -3.164  0.00179 **
## desmat_refAsia          0.1753     0.2004   0.875  0.38261
## desmat_refOceania      -1.7337     0.2821  -6.145 4.22e-09 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.9634 on 201 degrees of freedom
## Multiple R-squared:  0.9468, Adjusted R-squared:  0.9455
## F-statistic: 715.7 on 5 and 201 DF,  p-value: < 2.2e-16
```

Note that we get exactly identical parameter estimates and standard errors as in the model before. However, the R^2 and F-statistic are now calculated as 0.9468 and 715.7 respectively. This is not the expected result, as the R^2 and F-statistic should be the same as in the first model. The degrees of freedom is one higher here, set as 5 instead of 4, indicating the intercept column in the design matrix is treated as a variable.

We can easily prove the mathematical equivalence of the two models by comparing the design matrices of the two models.

```
dm_dm <- model.matrix(lm_ref)
dm_dm <- unname(dm_dm)
dm_nocoding <- model.matrix(lm_nocoding)
dm_nocoding <- unname(dm_nocoding)

all(dm_dm == dm_nocoding)
```

```
## [1] TRUE
```

Proposed Solution

We have made our way to avoid this problem by removing the intercept column from our design matrix and avoiding `-1` in our model formula.

```
# define the design matrix
desmat_ref <- model.matrix(~continent, data = wdi2013_ref)
colnames(desmat_ref) <- sub("continent", "", colnames(desmat_ref))
desmat_ref <- desmat_ref[,-1]
# fit the model
lm_ref <- lm(CO2emission ~ desmat_ref, data = wdi2013_ref)
summary(lm_ref)
```

```
##
## Call:
```

```
## lm(formula = CO2emission ~ desmat_ref, data = wdi2013_ref)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.72508 -0.68659  0.00954  0.55617  2.93454
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4.4355     0.1469  30.191  < 2e-16 ***
## desmat_refAfrica     -0.8831     0.1969  -4.485 1.23e-05 ***
## desmat_refAmericas  -0.6575     0.2078  -3.164  0.00179 **
## desmat_refAsia        0.1753     0.2004   0.875  0.38261
## desmat_refOceania    -1.7337     0.2821  -6.145 4.22e-09 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.9634 on 201 degrees of freedom
## Multiple R-squared:  0.263,  Adjusted R-squared:  0.2484
## F-statistic: 17.93 on 4 and 201 DF,  p-value: 1.311e-12
```

We see exactly the same model output as in the first model. The R^2 and F-statistic are now corrected as 0.263 and 17.93 respectively.

NB: this is a different approach than removing intercept when the design matrix is created (`model.matrix(~continent - 1,...)`), which yields mean value model.

We now move on to the case where the above proposed fix does not work.

Problem 2: Anomalies in Mean Value Models

The sample solution provided in Moodle was able to reach a noticeably high R^2 in its mean value model, which is 0.9468, same as our second, false model.

The model output is reproduced here:

```r
# we refrain from other coding options here, many approaches are possible
desmat_mean <- model.matrix(~continent - 1, data = wdi2013_ref)
colnames(desmat_mean) <- sub("continent", "", colnames(desmat_mean))
lm_mean <- lm(CO2emission ~ desmat_mean - 1, data = wdi2013_ref)
summary(lm_mean)
```

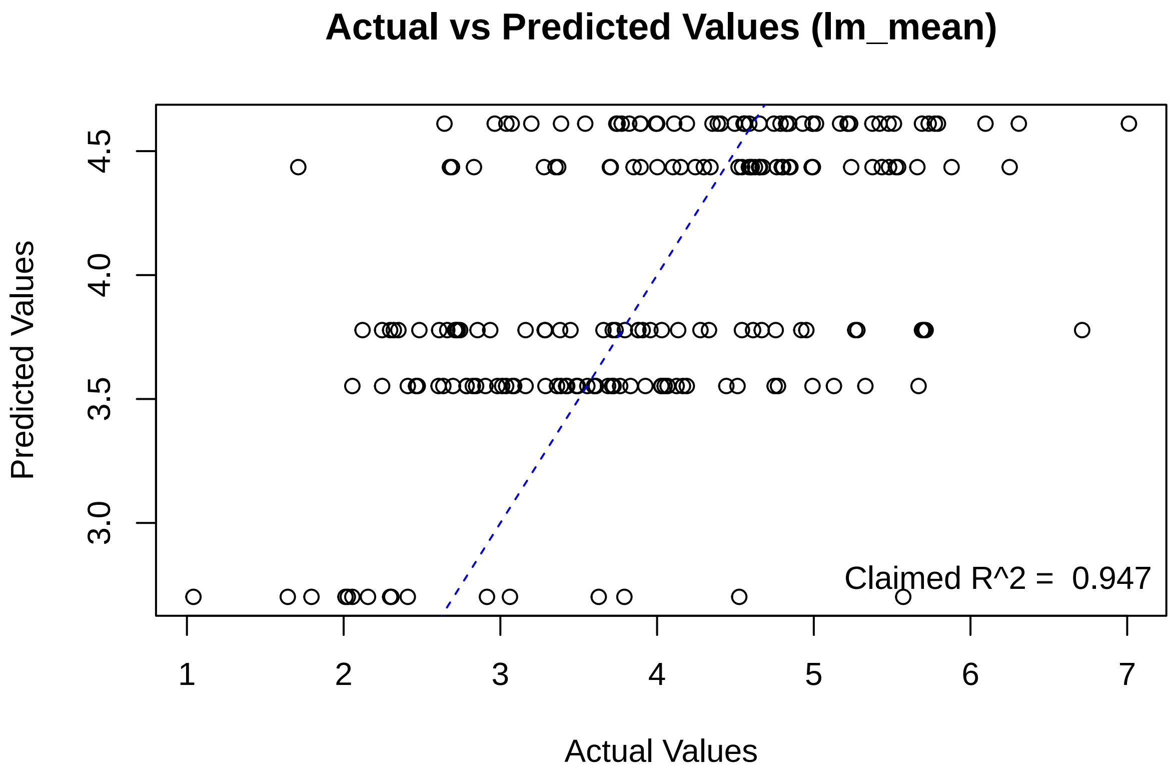
```
##
## Call:
## lm(formula = CO2emission ~ desmat_mean - 1, data = wdi2013_ref)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.72508 -0.68659  0.00954  0.55617  2.93454
##
## Coefficients:
```

```
##                        Estimate Std. Error t value Pr(>|t|)    
## desmat_meanEurope        4.4355     0.1469   30.19   <2e-16 ***
## desmat_meanAfrica        3.5525     0.1311   27.10   <2e-16 ***
## desmat_meanAmericas      3.7780     0.1469   25.72   <2e-16 ***
## desmat_meanAsia          4.6108     0.1362   33.84   <2e-16 ***
## desmat_meanOceania       2.7019     0.2408   11.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9634 on 201 degrees of freedom
## Multiple R-squared:  0.9468, Adjusted R-squared:  0.9455 
## F-statistic: 715.7 on 5 and 201 DF,  p-value: < 2.2e-16
```

This high R^2 does not explain exact model fit, as we visualize it in an actual vs. predicted plot.

```r
# plot actual vs. predicted values
plot(wdi2013_ref$CO2emission, lm_mean$fitted.values,
     xlab = "Actual Values", ylab = "Predicted Values",
     main = "Actual vs Predicted Values (lm_mean)")
abline(a = 0, b = 1, col = "blue", lty = 2)

# add claimed R^2 to the plot
r_squared <- summary(lm_mean)$r.squared
legend("bottomright", bty = "n", legend = paste("Claimed R^2 = ",
                                                round(r_squared, 3)))
```



Obviously, it is not possible to correct the statistics by removing a column from the design matrix, because this will be interpreted by R as another reference coded model.

Further Examination and Root Cause Analysis

To understand the cause of the difference, we need to understand how `lm` approaches the calculation of those statistics. Using `View(summary.lm)`, we can see the source code of the `summary.lm` function. The R^2 and F-statistic are calculated as follows:

```r
### all comments are added by authors of this article
# initial definition
z <- object # object is a lm-model
p <- z$rank
rdf <- z$df.residual
w <- z$weights
r <- z$residuals
f <- z$fitted.values
# mss and rss
if (is.null(w)) {
  mss <- if (attr(z$terms, "intercept"))
      sum((f - mean(f))^2)
  else sum(f^2)
  rss <- sum(r^2)
}
# ... truncated ...
# calculation of r.squared, adj.r.squared and fstatistic
if (p != attr(z$terms, "intercept")) {
  df.int <- if (attr(z$terms, "intercept"))
      1L
    else 0L
    ans$r.squared <- mss/(mss + rss)
    ans$adj.r.squared <- 1 - (1 - ans$r.squared) * ((n -
      df.int)/rdf)
    ans$fstatistic <- c(value = (mss/(p - df.int))/resvar,
      numdf = p - df.int, dendf = rdf)
}
# ... truncated ...
```

We can prove that the difference is mainly generated by the conditional term `if attr(z$terms, "intercept")`. This term is `TRUE` if the model is to automatically include an intercept in addition to variables given, and `FALSE` if not.

This can be examined using the following code:

```r
lm_ref$terms
lm_nocoding$terms
```

For clarity of the page, we simply conclude that `attr(z$terms, "intercept")` is `FALSE` for `lm_ref` and `TRUE` for `lm_nocoding`. This leads to different formulas used for `MSS` and `df.int`. The former is used for the calculation of R^2 and F, the latter is the degrees of freedom.

SSM of model with intercept: (slide 64)

$$SSM := (\hat{\mathbf{Y}} - \bar{\mathbf{Y}})'(\hat{\mathbf{Y}} - \bar{\mathbf{Y}}) = \sum(\hat{y}_i - \bar{y})^2$$

SSM of model without intercept: (slide 66)

$$SSM^* := \hat{\mathbf{Y}}'\hat{\mathbf{Y}} = \sum \hat{y}_i^2$$

This prevents us from getting the intended "corrected" SSM if the model has excluded automatic intercept. It could be resolved if another argument was added to the `lm` function, which would allow the user to specify whether the intercept term is `TRUE` or `FALSE`, allowing easier manual correction of the statistics.

Correcting the Mean Value Model

However, the discrepancy is not totally nonsense if we consider what R^2 and F-statistic are supposed to measure.

Using these statistics, we compare the fitted model with the predictive power of the **intercept-only model**. In the regular case, the intercept stands for the mean of the response variable. In the case of the mean value model, we are comparing the fitted model with **a model of 0**. Clearly, the fitted model must be much better than the 0-model, which is why the R^2 is so high.

Keeping this in mind, the mean value model can be manually corrected by subtracting the mean of the response variable from the response variable itself. This way, the intercept of the model is 0, and the R^2 and F are calculated correctly.

We need to note that this correction leads to parameter estimates as in the **effect-coding scenario**! To get the true means as coefficients, we simply need to add the mean of the response variable to the parameter estimates.

The corrected model is as follows:

```r
desmat_mean <- model.matrix(~continent - 1, data = wdi2013_ref)
colnames(desmat_mean) <- sub("continent", "", colnames(desmat_mean))
lm_mean_cor <- lm((CO2emission - mean(CO2emission)) ~ desmat_mean - 1,
              data = wdi2013_ref)
summary(lm_mean_cor)
```

```
##
## Call:
## lm(formula = (CO2emission - mean(CO2emission)) ~ desmat_mean -
##     1, data = wdi2013_ref)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.72508 -0.68659  0.00954  0.55617  2.93454
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## desmat_meanEurope    0.4608     0.1469   3.137  0.00197 **
## desmat_meanAfrica    -0.4222     0.1311  -3.221  0.00149 **
## desmat_meanAmericas  -0.1967     0.1469  -1.339  0.18222
## desmat_meanAsia       0.6361     0.1362   4.669 5.52e-06 ***
## desmat_meanOceania   -1.2728     0.2408  -5.285 3.26e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9634 on 201 degrees of freedom
## Multiple R-squared:  0.263,  Adjusted R-squared:  0.2447
## F-statistic: 14.35 on 5 and 201 DF,  p-value: 5.186e-12
```

```
coeffs <- coef(lm_mean_cor) + mean(wdi2013_ref$CO2emission)
print(coeffs)
```

```
##   desmat_meanEurope   desmat_meanAfrica desmat_meanAmericas     desmat_meanAsia
##            4.435520            3.552465            3.778044            4.610842
##  desmat_meanOceania
##            2.701870
```

End Note

In conclusion, while mean value models and the use of design matrices in R modelling may not be common practice, this discussion illuminates important aspects of R's linear modeling functions. By understanding the underlying mechanics of R-squared and the F-statistic calculations, we can ensure accurate model evaluations. This exploration not only resolves specific issues but also enhances our comprehension of statistical modeling in R, enabling us to apply these insights to more complex scenarios.