Exercise 5 - Variable Selection

Multicollinearity and Variable Selection

O’Malley, Conie

2025-02-02

Table of Contents

[Preparation 1](#_Toc189385515)

[1. Read the Data 1](#_Toc189385516)

[2. OLS 2](#_Toc189385517)

[3. Multicollinearity Testing 2](#_Toc189385518)

[4. Stepwise Variable Selection 4](#_Toc189385519)

[5. Re-testing for Multicollinearity Testing 9](#_Toc189385520)

## Preparation

Download the **Ex4\_YourLastName.Rmd** R Markdown file to your working directory, rename with your last name and follow the instructions below. When you finish, upload onto Blackboard the .Rmd file or the knitted file as a Word or PDF file.

## 1. Read the Data

1.1 Use the **read.table()** function to retrieve columns **3 to 10** from this data set from Blackboard and store it in your R working folder for this class: **PizzaCal.csv** . Name your dataset **pizza**.

This dataset contains nutrition data on a sample of 300 slices of frozen pizza (100 grams each):

* **import**: binary variable, 0 if domestic, 1 if imported
* **cal**: number of calories in the slice
* **mois**: moisture content (grams per slice)
* **prot**: amount of protein (grams per slice)
* **fat**: fat content (grams per slice)
* **ash**: ash content (grams per slice)
* **sodium**: amount of sodium (grams per slice)
* **carb**: amount of carbohydrates (grams per slice)

# Done for you, but please ensure your data set has the columns listed below  
pizza <- read.table("/Users/coniecakes/Library/CloudStorage/OneDrive-Personal/001. Documents - Main/023. Programming Tools/R Studio/PredictiveAnalytics/R\_Exercises/PizzaCal.csv", header=TRUE, sep=",")[,3:10]

## 2. OLS

2.1 Fit an OLS model to predict **cal** with all other variables as predictors. Store the results in an object named **fit.full**. Display the summary() results for this model.

fit.full <- lm(cal ~ ., pizza) # fit an OLS model  
summary(fit.full) # view summary statistics

##   
## Call:  
## lm(formula = cal ~ ., data = pizza)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.7246 -0.3716 -0.0677 0.2065 25.5575   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 608.7980 127.3020 4.782 2.75e-06 \*\*\*  
## import 0.4670 0.3579 1.305 0.1930   
## mois -6.0868 1.2734 -4.780 2.78e-06 \*\*\*  
## prot -2.0888 1.2724 -1.642 0.1017   
## fat 2.9244 1.2736 2.296 0.0224 \*   
## ash -6.0664 1.3219 -4.589 6.62e-06 \*\*\*  
## sodium -0.5532 1.0002 -0.553 0.5806   
## carb -2.0881 1.2729 -1.640 0.1020   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.584 on 292 degrees of freedom  
## Multiple R-squared: 0.9994, Adjusted R-squared: 0.9993   
## F-statistic: 6.542e+04 on 7 and 292 DF, p-value: < 2.2e-16

## 3. Multicollinearity Testing

3.1 First, load the {perturb} (for CI) and {car} (for VIF’s) libraryies. Obtain the **Condition Index (CI)** for this model using the colldiag() function. For now, use the raw predictors with attributes scale=T, center=T to get the same results I got. Also, obtain the **Variance Inflation Factors (VIFs)** for the predictors in the model, using the vif() function.

**Technical Note:** scale=T standardizes the predictors before the CI computation. In essence, it causes the CI to be based on the correlation matrix rather than the covariance matrix, which is important when variables are in different scales. The default value or scale is T, so if you omit it, it still works fine. center=T causes all the variables to be centered at their means. This forces the regression line to go through the origin without intercept. This is important to avoid high correlation between predictors and the intercept to influence the CI calculation. The default for center is F, so it is important to change this to T. You can also add the parameter add.intercept=F to force the regression line to go through the origing, but when you set center=T it is not necessary.

library(perturb)  
library(car)  
  
cond\_indx <- colldiag(fit.full, scale = TRUE, center = TRUE)  
cond\_indx

## Condition  
## Index Variance Decomposition Proportions  
## import mois prot fat ash sodium carb   
## 1 1.000 0.000 0.000 0.000 0.000 0.000 0.003 0.000  
## 2 1.336 0.053 0.000 0.000 0.000 0.000 0.002 0.000  
## 3 2.927 0.085 0.000 0.000 0.000 0.000 0.029 0.000  
## 4 4.629 0.826 0.000 0.000 0.000 0.000 0.001 0.000  
## 5 6.987 0.015 0.000 0.000 0.000 0.012 0.107 0.000  
## 6 12.192 0.018 0.000 0.001 0.000 0.046 0.855 0.000  
## 7 645.742 0.002 1.000 0.999 1.000 0.942 0.003 1.000

vif(fit.full)

## import mois prot fat ash sodium   
## 3.175904 17637.546103 7989.535468 15575.156318 335.803636 16.354718   
## carb   
## 62784.245725

3.2 Interpretation. Are there problems with multi-collinearity? Please briefly explain in 1 or 2 lines why or why not.

#### Analysis

Yes there is a problem with multicollinearity. The Condition Index of the model is 645.742 which is much greater than the threshold of 50 and all predictors have Variance Inflation Factors > 10, which is the threshold for individual VIFs. This suggests that all predictor variables are correlated.

3.3 Suggestions. Do you have any suggestions on how to resolve the multicollinearity issue? Provide one sound suggestion

#### Suggestion

Perform a stepwise regression to remove variables that are highly correlated.

## 4. Stepwise Variable Selection

4.1 Fit the **null** model and name it **fit.null**. Then run a **stepwise** variable selection process in a **backwards** fashion, between **fit.full** and **fit.null**. So, **fit.full** should be your first argument in the step() function. Then, we want the scope parameter of the stepwise process to go from lower=fit.null to upper=fit.full. We also want the direction=both to proceed in a **stepwise** fashion.

**Technical Note:** the default test in step() is the **AIC**. But it is better to use a standard statistical test like the **F-test**, which is what we do with anova(). To change the default test to an **F-Test** (which uses a p<0.15 as the default criterion to add and remove variables), add the test="F" attribute. In sum, your scope() function should look like this: scope=list(lower=fit.null, upper=fit.full), direction="both", test="F".

Save the resulting step() object with the name **fit.step.15**. Then display it using the summary() function.

fit.null <- lm(cal ~ 1, pizza) # fit a null model  
step(fit.full, scope = list(lower = fit.null, upper = fit.full), # stepwise regression  
 direction = "both", test = "F") -> fit.step.15

## Start: AIC=283.8  
## cal ~ import + mois + prot + fat + ash + sodium + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## - sodium 1 0.767 733.26 282.12 0.3059 0.58064   
## - import 1 4.270 736.76 283.55 1.7022 0.19302   
## <none> 732.49 283.80   
## - carb 1 6.750 739.24 284.55 2.6909 0.10200   
## - prot 1 6.761 739.25 284.56 2.6950 0.10174   
## - fat 1 13.226 745.72 287.17 5.2725 0.02237 \*   
## - ash 1 52.829 785.32 302.69 21.0595 6.620e-06 \*\*\*  
## - mois 1 57.318 789.81 304.40 22.8490 2.781e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Step: AIC=282.12  
## cal ~ import + mois + prot + fat + ash + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## - import 1 3.767 737.03 281.65 1.5051 0.22087   
## <none> 733.26 282.12   
## - prot 1 6.906 740.17 282.93 2.7597 0.09774 .   
## - carb 1 7.002 740.26 282.97 2.7977 0.09547 .   
## + sodium 1 0.767 732.49 283.80 0.3059 0.58064   
## - fat 1 12.854 746.11 285.33 5.1363 0.02416 \*   
## - mois 1 58.154 791.41 303.01 23.2375 2.303e-06 \*\*\*  
## - ash 1 58.199 791.46 303.03 23.2555 2.283e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Step: AIC=281.65  
## cal ~ mois + prot + fat + ash + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## <none> 737.03 281.65   
## + import 1 3.767 733.26 282.12 1.5051 0.22087   
## - prot 1 7.296 744.32 282.61 2.9104 0.08907 .   
## - carb 1 7.392 744.42 282.65 2.9485 0.08701 .   
## + sodium 1 0.264 736.76 283.55 0.1049 0.74622   
## - fat 1 12.387 749.41 284.65 4.9410 0.02699 \*   
## - ash 1 58.904 795.93 302.72 23.4968 2.029e-06 \*\*\*  
## - mois 1 59.690 796.72 303.01 23.8105 1.745e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

summary(fit.step.15)

##   
## Call:  
## lm(formula = cal ~ mois + prot + fat + ash + carb, data = pizza)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.5961 -0.3576 -0.0795 0.1952 25.7092   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 618.801 126.986 4.873 1.80e-06 \*\*\*  
## mois -6.196 1.270 -4.880 1.75e-06 \*\*\*  
## prot -2.168 1.271 -1.706 0.0891 .   
## fat 2.822 1.269 2.223 0.0270 \*   
## ash -6.258 1.291 -4.847 2.03e-06 \*\*\*  
## carb -2.181 1.270 -1.717 0.0870 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.583 on 294 degrees of freedom  
## Multiple R-squared: 0.9994, Adjusted R-squared: 0.9993   
## F-statistic: 9.165e+04 on 5 and 294 DF, p-value: < 2.2e-16

**Technical Note:** which p-value to use to include and remove variables? A larger p-value will yield more predictors in the model, whereas a smaller p-value will be more restrictive and only retain highly significant predictors. You can control this with a parameter **k**. k is the chi-square equivalent to the p-value. The default in step() is k=2, which is equivalent to p<0.15.

4.2 To use p<0.05 as the variable inclusion/removal criteria, use qchisq(0.05, 1, lower.tail=F) to find the **k** value corresponding to p<0.05. Save this result in a variable named **kval** and display its result. You should get k=3.84. Then either add at the end of the scope() function the parameter k=3.84, or better yet, use k=kval. And re-fit **fit.step** but this time save the model as **fit.step.05**

kval <- qchisq(0.05, 1, lower.tail = FALSE) # assign new chi square value to variable object  
kval

## [1] 3.841459

fit.step.05 <- step(fit.full, scope = list(lower = fit.null, upper = fit.full),  
 direction = "both", test = "F", k = kval) # stepwise regression

## Start: AIC=298.53  
## cal ~ import + mois + prot + fat + ash + sodium + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## - sodium 1 0.767 733.26 295.01 0.3059 0.58064   
## - import 1 4.270 736.76 296.44 1.7022 0.19302   
## - carb 1 6.750 739.24 297.44 2.6909 0.10200   
## - prot 1 6.761 739.25 297.45 2.6950 0.10174   
## <none> 732.49 298.53   
## - fat 1 13.226 745.72 300.06 5.2725 0.02237 \*   
## - ash 1 52.829 785.32 315.58 21.0595 6.620e-06 \*\*\*  
## - mois 1 57.318 789.81 317.29 22.8490 2.781e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Step: AIC=295.01  
## cal ~ import + mois + prot + fat + ash + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## - import 1 3.767 737.03 292.70 1.5051 0.22087   
## - prot 1 6.906 740.17 293.98 2.7597 0.09774 .   
## - carb 1 7.002 740.26 294.01 2.7977 0.09547 .   
## <none> 733.26 295.01   
## - fat 1 12.854 746.11 296.38 5.1363 0.02416 \*   
## + sodium 1 0.767 732.49 298.53 0.3059 0.58064   
## - mois 1 58.154 791.41 314.06 23.2375 2.303e-06 \*\*\*  
## - ash 1 58.199 791.46 314.08 23.2555 2.283e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Step: AIC=292.7  
## cal ~ mois + prot + fat + ash + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## - prot 1 7.296 744.32 291.81 2.9104 0.08907 .   
## - carb 1 7.392 744.42 291.85 2.9485 0.08701 .   
## <none> 737.03 292.70   
## - fat 1 12.387 749.41 293.86 4.9410 0.02699 \*   
## + import 1 3.767 733.26 295.01 1.5051 0.22087   
## + sodium 1 0.264 736.76 296.44 0.1049 0.74622   
## - ash 1 58.904 795.93 311.93 23.4968 2.029e-06 \*\*\*  
## - mois 1 59.690 796.72 312.22 23.8105 1.745e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Step: AIC=291.81  
## cal ~ mois + fat + ash + carb  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## - carb 1 1 745 288.25 0.2722 0.60227   
## <none> 744 291.81   
## + prot 1 7 737 292.70 2.9104 0.08907 .   
## + import 1 4 740 293.98 1.6509 0.19985   
## + sodium 1 0 744 295.52 0.1315 0.71719   
## - ash 1 662 1407 478.89 262.4441 < 2e-16 \*\*\*  
## - mois 1 44000 44745 1516.85 17438.7885 < 2e-16 \*\*\*  
## - fat 1 73723 74467 1669.67 29218.8368 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Step: AIC=288.25  
## cal ~ mois + fat + ash  
##   
## Df Sum of Sq RSS AIC F value Pr(>F)   
## <none> 745 288.25   
## + import 1 4 741 290.42 1.6434 0.2009   
## + sodium 1 1 744 291.70 0.3841 0.5359   
## + carb 1 1 744 291.81 0.2722 0.6023   
## + prot 1 1 744 291.85 0.2342 0.6288   
## - ash 1 1597 2342 627.96 634.3175 <2e-16 \*\*\*  
## - fat 1 130323 131068 1835.43 51778.5060 <2e-16 \*\*\*  
## - mois 1 237474 238219 2014.67 94350.9013 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

summary(fit.step.05)

##   
## Call:  
## lm(formula = cal ~ mois + fat + ash, data = pizza)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.4423 -0.3469 -0.0701 0.2075 25.7179   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 400.81798 0.59090 678.32 <2e-16 \*\*\*  
## mois -4.01628 0.01308 -307.17 <2e-16 \*\*\*  
## fat 4.99667 0.02196 227.55 <2e-16 \*\*\*  
## ash -3.99510 0.15863 -25.19 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.586 on 296 degrees of freedom  
## Multiple R-squared: 0.9994, Adjusted R-squared: 0.9993   
## F-statistic: 1.521e+05 on 3 and 296 DF, p-value: < 2.2e-16

summary(fit.step.15)

##   
## Call:  
## lm(formula = cal ~ mois + prot + fat + ash + carb, data = pizza)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.5961 -0.3576 -0.0795 0.1952 25.7092   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 618.801 126.986 4.873 1.80e-06 \*\*\*  
## mois -6.196 1.270 -4.880 1.75e-06 \*\*\*  
## prot -2.168 1.271 -1.706 0.0891 .   
## fat 2.822 1.269 2.223 0.0270 \*   
## ash -6.258 1.291 -4.847 2.03e-06 \*\*\*  
## carb -2.181 1.270 -1.717 0.0870 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.583 on 294 degrees of freedom  
## Multiple R-squared: 0.9994, Adjusted R-squared: 0.9993   
## F-statistic: 9.165e+04 on 5 and 294 DF, p-value: < 2.2e-16

4.3 Answer briefly: what is the difference in the final model when using p<0.05 rather than p<0.15 as the inclusion/removal criterion?

#### Analysis

The final model is a further reduced model when because the confidence level for the F-test is reduced. As we see the variables prot and carb were significant at a level, but no longer at a level.

## 5. Re-testing for Multicollinearity Testing

5.1 Now that we have 2 reduced models identified by **stepwise** regression, compute the CI (colldiag()) and VIF for both models. For colldiag() use scale=T, center=T.

col\_indx\_15 <- colldiag(fit.step.15, scale = TRUE, center = TRUE) # CI for reduced model 1  
col\_indx\_15

## Condition  
## Index Variance Decomposition Proportions  
## mois prot fat ash carb   
## 1 1.000 0.000 0.000 0.000 0.000 0.000  
## 2 1.671 0.000 0.000 0.000 0.000 0.000  
## 3 3.292 0.000 0.000 0.000 0.000 0.000  
## 4 7.173 0.000 0.000 0.000 0.033 0.000  
## 5 594.374 1.000 1.000 1.000 0.967 1.000

vif(fit.step.15) # VIFs for reduced model 1

## mois prot fat ash carb   
## 17551.9623 7971.3499 15483.2329 320.4669 62538.6785

col\_indx\_05 <- colldiag(fit.step.05, scale = TRUE, center = TRUE) # CI for reduced model 2  
col\_indx\_05

## Condition  
## Index Variance Decomposition Proportions  
## mois fat ash   
## 1 1.000 0.002 0.058 0.059  
## 2 1.278 0.434 0.017 0.005  
## 3 4.202 0.563 0.925 0.936

vif(fit.step.05) # VIFs for reduced model 2

## mois fat ash   
## 1.853456 4.614729 4.819130

5.2 What is your general conclusion from this analysis? Which model specification would you select?

#### Analysis

The variables in fit.step.15 are still highly correlated. The Condition Index = 594.374 and VIF > 320 for all variables - both well outside of acceptable ranges suggesting high multicollinearity. I would select the second reduced model fit.step.05 because the Condition Index = 4.202 is well below the threshold of 50 and all VIFs < 10 - suggesting that multicollinearity has been addressed.