Multicollinearity Issues in Linear Regression

Predictive Modeling & Statistical Learning

Gaston Sanchez

Multicollinearity Issues

Caveat

In these slides, I'm assuming that all variables (predictors and response) are centered (mean $= 0$)!

Linear Regression

Assuming centered data, the multiple regression model is:

$$Y = \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon$$

In matrix notation

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

Least Squares Solution

The OLS solution is given by:

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}\mathbf{b}$$

where:

$$\mathbf{b} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

Introduction

One of the issues when fitting regression models is due to multicollinearity: the condition that arises when two or more predictors are highly correlated.

How does this affect OLS regression?

Exact Collinearity

When one or more predictors are linear combinations of other predictors, then $\mathbf{X}^\mathsf{T}\mathbf{X}$ is singular.

This is known as *exact collinearity*.

There is no unique LS estimate $\hat{\boldsymbol{\beta}}$

Multicollinearity: Near-exact Collinearity

A more challenging problem arises when $\mathbf{X}^\top\mathbf{X}$ is close to singular but not exactly.

This is usually referred to as *multicollinearity*

Multicollinearity leads to imprecise (unstable) estimates $\hat{\boldsymbol{\beta}}$

What causes multicollinearity?

- One or more predictors are linear combinations of other predictors
- One or more predictors are almost perfect linear combinations of other predictors
- More predictors than observations $p > n$

Let's play with `mtcars`

Data set `mtcars`

First 10 rows:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

Let's use `mpg` as response, and `disp`, `hp`, and `wt` as predictors.

Data set mtcars

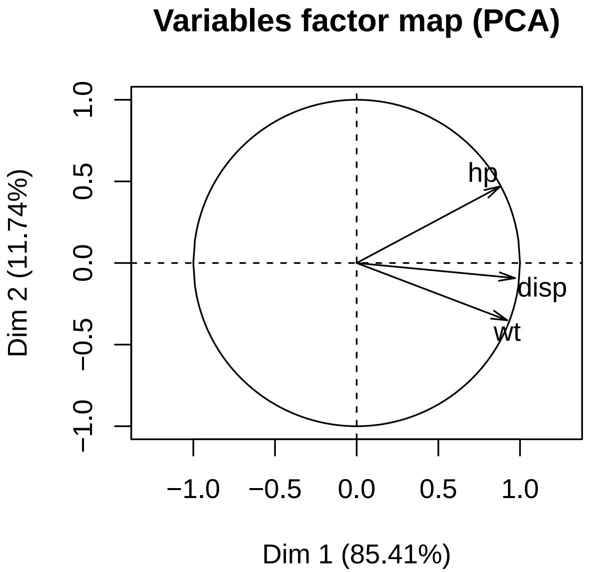
```
# response
mpg <- mtcars$mpg

# predictors
disp <- mtcars$disp
hp <- mtcars$hp
wt <- mtcars$wt

# standardized responses, and correlation matrix
X <- scale(cbind(disp, hp, wt))
```

Correlation matrix

$$\mathbf{R} = \frac{1}{n-1}\mathbf{X}^{\intercal}\mathbf{X}$$

```r
# correlation matrix
cor(X)
```

```
##           disp        hp        wt
## disp 1.0000000 0.7909486 0.8879799
## hp   0.7909486 1.0000000 0.6587479
## wt   0.8879799 0.6587479 1.0000000
```


Variables factor map (PCA)
hp
disp
wt
Dim 1 (85.41%)
Dim 2 (11.74%)
−1.0
−0.5
0.0
0.5
1.0

LS Regression

```r
# LS regression
reg <- lm(mpg ~ disp + hp + wt)
reg

##
## Call:
## lm(formula = mpg ~ disp + hp + wt)
##
## Coefficients:
## (Intercept)         disp           hp           wt
##   37.105505    -0.000937    -0.031157    -3.800891

# regression summary
reg_sum <- summary(reg)
```

LS Regression

```
Call:
lm(formula = mpg ~ disp + hp + wt)

Residuals:
   Min     1Q Median     3Q    Max
-3.891 -1.640 -0.172  1.061  5.861

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 37.105505   2.110815  17.579  < 2e-16 ***
disp        -0.000937   0.010350  -0.091  0.92851
hp          -0.031157   0.011436  -2.724  0.01097 *
wt          -3.800891   1.066191  -3.565  0.00133 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.639 on 28 degrees of freedom
Multiple R-squared:  0.8268,Adjusted R-squared:  0.8083
F-statistic: 44.57 on 3 and 28 DF,  p-value: 8.65e-11
```

LS Regression

Ratio between std errors and coeffs

```
round(reg_sum$coefficients[,2] / reg_sum$coefficients[,1], 4)

(Intercept)        disp          hp          wt
     0.0569    -11.0455     -0.3670     -0.2805
```

`disp` has a large standard error compared to its estimate

Inverse of $(\mathbf{X}^\top \mathbf{X})$

What about $(\mathbf{X}^\top \mathbf{X})^{-1}$

```
solve(t(X) %*% X)
```

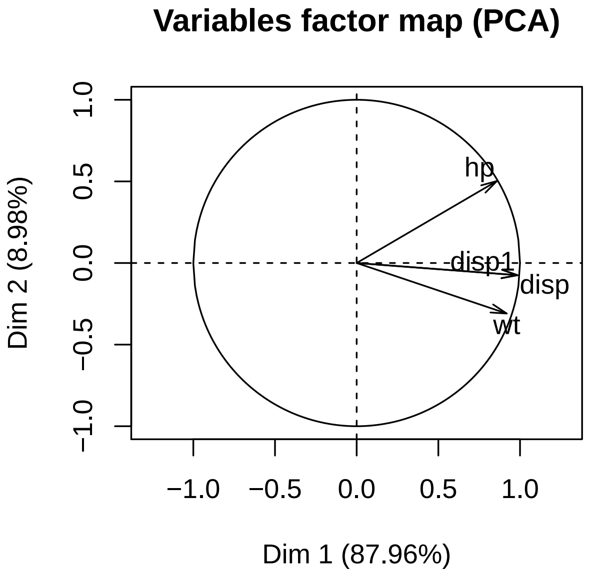
```
##              disp          hp          wt
## disp  0.23627475 -0.08598357 -0.15316573
## hp   -0.08598357  0.08827847  0.01819843
## wt   -0.15316573  0.01819843  0.15627798
```

Exact Collinearity

Let's introduce exact collinearity

```
disp1 <- 10 * disp

X1 <- scale(cbind(disp, disp1, hp, wt))

solve(t(X1) %*% X1)

Error in solve.default(t(X1) %*% X1):  system is
computationally singular:  reciprocal condition number =
1.55757e-17
```

Ooops!

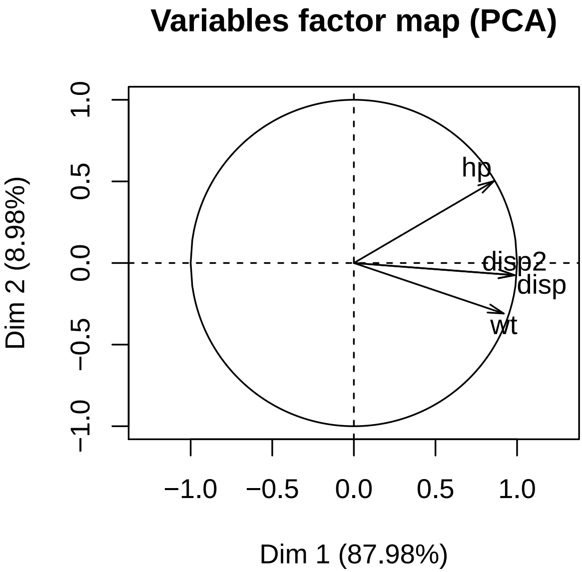

Variables factor map (PCA)
hp
disp1
disp
wt
Dim 1 (87.96%)
Dim 2 (8.98%)
−1.0
−0.5
0.0
0.5
1.0

Near-exact Collinearity

Let's introduce near-exact collinearity

```r
set.seed(123)
disp2 <- disp + rnorm(length(disp))

X2 <- scale(cbind(disp, disp2, hp, wt))

solve(t(X2) %*% X2)
```

```
             disp       disp2          hp          wt
disp   588.167214 -590.721826  1.01055902  1.99383316
disp2 -590.721826  593.525960 -1.10174784 -2.15719062
hp       1.010559   -1.101748  0.09032362  0.02220277
wt       1.993833   -2.157191  0.02220277  0.16411837
```

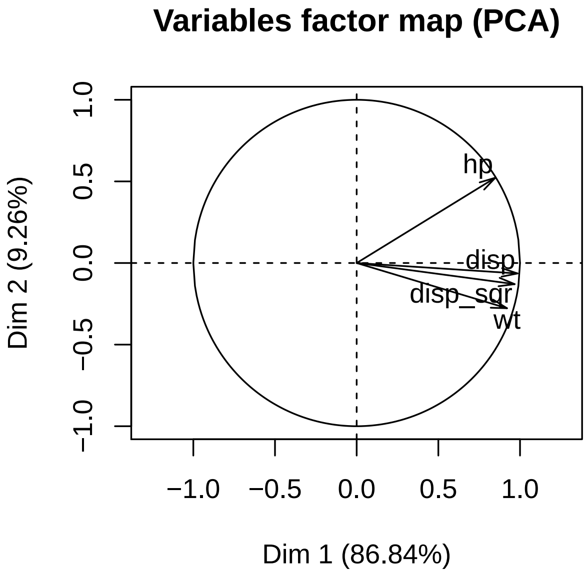

Variables factor map (PCA)
hp
disp2
disp
wt
1.0
0.5
0.0
−0.5
−1.0
Dim 2 (8.98%)
−1.0
−0.5
0.0
0.5
1.0
Dim 1 (87.98%)

Near-exact Collinearity

What about X_j and X_j^2?

```
disp_sqr <- disp^2

Xsqr <- scale(cbind(disp, disp_sqr, hp, wt))

solve(t(Xsqr) %*% Xsqr)
```

```
               disp    disp_sqr          hp          wt
disp      1.2157359 -0.91716163 -0.17616042 -0.16445287
disp_sqr -0.9171616  0.85882476  0.08444107  0.01056920
hp       -0.1761604  0.08444107  0.09658086  0.01923761
wt       -0.1644529  0.01056920  0.01923761  0.15640806
```


Variables factor map (PCA)
hp
disp
disp_sqr
wt
1.0
0.5
0.0
−0.5
−1.0
Dim 2 (9.26%)
−1.0
−0.5
0.0
0.5
1.0
Dim 1 (86.84%)

Multicollinearity

Let's examine correlations of `disp` and cousins

```
cor(cbind(disp, disp_sqr, hp, wt))

              disp  disp_sqr        hp        wt
disp     1.0000000 0.9792310 0.7909486 0.8879799
disp_sqr 0.9792310 1.0000000 0.7382463 0.8712214
hp       0.7909486 0.7382463 1.0000000 0.6587479
wt       0.8879799 0.8712214 0.6587479 1.0000000
```

Multicollinearity Issues

```r
solve(t(X2) %*% X2)
```

```
             disp       disp2          hp          wt
disp   588.167214 -590.721826  1.01055902  1.99383316
disp2 -590.721826  593.525960 -1.10174784 -2.15719062
hp       1.010559   -1.101748  0.09032362  0.02220277
wt       1.993833   -2.157191  0.02220277  0.16411837
```

```r
solve(t(Xsqr) %*% Xsqr)
```

```
               disp    disp_sqr          hp          wt
disp      1.2157359 -0.91716163 -0.17616042 -0.16445287
disp_sqr -0.9171616  0.85882476  0.08444107  0.01056920
hp       -0.1761604  0.08444107  0.09658086  0.01923761
wt       -0.1644529  0.01056920  0.01923761  0.15640806
```

Let's make it
more extreme!

Extreme Multicollinearity

```r
set.seed(123)
disp3 <- disp + rnorm(length(disp), mean = 0, sd = 0.1)

X3 <- scale(cbind(disp, disp3, hp, wt))

cor(disp, disp3)

[1] 0.9999997
```

Multicollinearity Issues

```
# small changes may have a "butterfly" effect
disp31 <- disp3

# change just one observation
disp31[1] <- disp3[1] * 1.01

X31 <- scale(cbind(disp, disp31, hp, wt))

cor(disp, disp31)

[1] 0.9999973
```

Multicollinearity Issues

```
solve(t(X3) %*% X3)

              disp        disp3           hp           wt
disp   59175.36325 -59202.97211  10.91501090  21.38646548
disp3 -59202.97211  59230.83035 -11.00617104 -21.54976679
hp        10.91501    -11.00617   0.09032362   0.02220277
wt        21.38647    -21.54977   0.02220277   0.16411837
```

```
solve(t(X31) %*% X31)

               disp       disp31          hp          wt
disp   5941.5946101 -5942.3977358  0.30661752  0.64947961
disp31 -5942.3977358  5943.4373181 -0.39266978 -0.80278577
hp        0.3066175   -0.3926698  0.08830442  0.01825147
wt        0.6494796   -0.8027858  0.01825147  0.15638642
```

Variance of Coefficients

Variance-Covariance matrix $Var(\hat{\boldsymbol{\beta}})$

$$Var(\hat{\boldsymbol{\beta}}) = \begin{bmatrix} Var(\hat{\beta}_1) & Cov(\hat{\beta}_1, \hat{\beta}_2) & \cdots & Cov(\hat{\beta}_1, \hat{\beta}_p) \\ Cov(\hat{\beta}_2, \hat{\beta}_1) & Var(\hat{\beta}_2) & \cdots & Cov(\hat{\beta}_2, \hat{\beta}_p) \\ \vdots & & \ddots & \vdots \\ Cov(\hat{\beta}_p, \hat{\beta}_1) & Cov(\hat{\beta}_p, \hat{\beta}_2) & \cdots & Var(\hat{\beta}_p) \end{bmatrix}$$

$$Var(\hat{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}$$

Variance of Estimates

$$Var(\hat{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}^\intercal\mathbf{X})^{-1}$$

The variance of a particular coefficient $\hat{\beta}_j$ is given by:

$$Var(\hat{\beta}_j) = \sigma^2\left[(\mathbf{X}^\intercal\mathbf{X})^{-1}\right]_{jj}$$

where $\left[(\mathbf{X}^\intercal\mathbf{X})^{-1}\right]_{jj}$ is the j-th diagonal element of $(\mathbf{X}^\intercal\mathbf{X})^{-1}$

Variance of Estimates

- Recall again that we don't know σ^2. How can we find an estimator $\hat{\sigma}^2$?
- We don't observe the error terms ε but we do have the residuals $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$
- As well as the Residual Sum of Squares (RSS)

$$RSS = \frac{1}{n}\sum_{i=1}^{n} e_i^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Unbiased Estimate of σ^2

To estimate σ^2 we use:

$$\hat{\sigma}^2 = \frac{RSS}{n-p-1} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-p-1} = s^2$$

The square root $\hat{\sigma} = \sqrt{\frac{RSS}{n-p-1}}$ is also known as the **Residual Standard Error** (reported by most software)

How to detect Multicollinearity?

How to detect Collinearity?

- Examine correlation matrix of predictors
- Check multiple correlation coefficients R_j^2
- Examine eigenvalues of $\mathbf{X}^\top\mathbf{X}$

Detecting collinearity pairwise correlations

Perhaps the most basic approach to start checking whether there is multicollinearity is to examine the correlation matrix of predictors

$$\mathbf{R} = \frac{1}{n}\mathbf{X}^\mathsf{T}\mathbf{X}$$

Detecting collinearity pairwise correlations

Examining the correlation matrix

- Examining the correlation matrix of predictors

$$\mathbf{R} = \frac{1}{n}\mathbf{X}^{\intercal}\mathbf{X}$$

- Correlation values close to -1 or $+1$ indicate large **pairwise** collinearities.
- However, there may be small correlations from highly correlated variables.

Detecting collinearity with R_j^2 coefficients

Multiple correlation coefficients

- Another way to check for collinearity is to calculate multiple correlation coefficients R_j^2 for each predictor.
- Regress X_j on all other predictors X_h $(h \neq j)$.
- If R_j^2 is close to one, it means that this predictor can almost be predicted exactly by a linear combination of other predictors.

Detecting collinearity with Eigenvalues

Eigenvalues

- A third approch is to examine the eigenvalues of $\mathbf{X}^\top\mathbf{X}$
- Eigenvalues equal to zero denote exact collinearity
- Small eigenvalues (close to zero) indicate multicollinearity. But how small?

Detecting collinearity with Eigenvalues

Some authors propose to use the **condition number** κ

$$\kappa = \sqrt{\frac{\lambda_1}{\lambda_k}}$$

to determine if a given eigenvalue λ_k is “sufficiently” small enough.

A condition number $\kappa \geq 30$ is considered to indicate small λ_k.

Effect of Multicollinearity

Variance Inflation Factor (VIF)

Assuming standardized variables, $\mathbf{X}^\mathsf{T}\mathbf{X} = n\mathbf{R}$
It can be shown that

$$Var(\hat{\boldsymbol{\beta}}) = \sigma^2 \left(\frac{\mathbf{R}^{-1}}{n} \right)$$

and $Var(\hat{\beta}_j)$ can then be expressed as:

$$Var(\hat{\beta}_j) = \frac{\sigma^2}{n} [\mathbf{R}^{-1}]_{jj}$$

Variance Inflation Factor (VIF)

$$Var(\hat{\beta}_j) = \frac{\sigma^2}{n}[\mathbf{R}^{-1}]_{jj}$$

It turns out that:

$$[\mathbf{R}^{-1}]_{jj} = \frac{1}{1 - R_j^2}$$

is known as the **Variance Inflation Factor** or VIF

Effects of Collinearity

The effect of collinearity can be seen by examining $Var(\hat{\boldsymbol{\beta}})$

$$Var(\hat{\boldsymbol{\beta}}) = \left[\frac{\sigma^2}{1 - R_j^2}\right] \left(\frac{1}{\sum_{i=1}^{n}(x_{ij} - \bar{x})^2}\right)$$

- If a predictor X_j does not vary much, then $Var(\hat{\boldsymbol{\beta}})$ will be large.
- If R_j^2 is close to 1, then VIF will be large, and so $Var(\hat{\boldsymbol{\beta}})$ will also be large.

Role of eigenvalues of matrix $\mathbf{R}$

If we write the eigenvalue decomposition of $\mathbf{R}$ as:

$$\mathbf{R} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\mathsf{T}$$

Role of eigenvalues of matrix $\mathbf{R}$

If we write the eigenvalue decomposition of $\mathbf{R}$ as:

$$\mathbf{R} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathsf{T}}$$

then the inverse of $\mathbf{R}$ becomes:

$$\mathbf{R}^{-1} = \mathbf{U}\mathbf{\Lambda}^{-1}\mathbf{U}^{\mathsf{T}}$$

Role of eigenvalues of matrix $\mathbf{R}$

It can be shown that

$$Var(\hat{\boldsymbol{\beta}}) = \left(\frac{\sigma^2}{n}\right) \sum_{l=1}^{p} \frac{u_{jl}}{\lambda_l}$$

As you can tell, the variance of the estimators depends on the inverses of the eigenvalues of $\mathbf{R}$

With very small eigenvalues, the larger the variance of the estimates.

In Summary

With multicollinearity ...

- the standard errors of $\hat{\beta}_j$ are inflated
- the fit is unstable, and becomes very sensitive to small perturbations
- small changes in Y can lead to large changes in the coefficients

Think about it

What would you do to overcome multicollinearity?

Some suggestions

- Reduce number of predictors
- If $p > n$, then try to get more observations (increase n)
- Find a basis for the predictors
- Impose constraints on the estimated coefficients
- A mix of all of the above?
- *Other ideas?*