Regression And Stat Models - Assignment 8

Nathan Pasder

2025-07-16

Contents

Question 1

(a) Multicollinearity Analysis

We begin by loading the dataset `"ex8_data.csv"` and inspecting the structure. We remove the response variable `Y` from the dataset, leaving only the explanatory variables.

```
# Load required libraries
library(tidyverse)
library(readr)
library(car) # Load car package and compute VIFs
library(psych) # For pairs.panels function
```

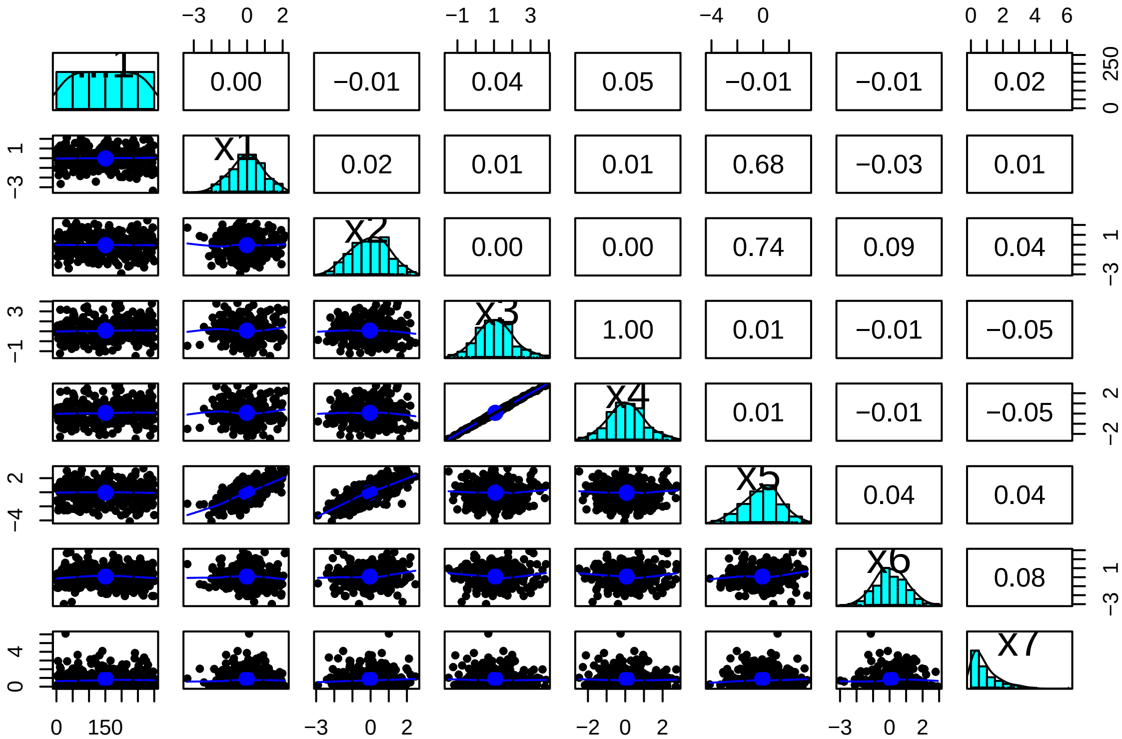
```
# Load the dataset and clean column names
data <- read_csv("ex8_data.csv")
names(data) <- trimws(names(data))  # remove leading/trailing spaces

# Separate Y (response) and X (predictors)
Y <- data$y
X <- data %>% select(-y)
```

Next, we examine the **pairwise correlation matrix** between the explanatory variables. High correlations (e.g., above 0.8 or below -0.8) may indicate multicollinearity.

```
# Compute and display the correlation matrix
cor_matrix <- cor(X)
print(cor_matrix)
```

```
##              ...1           x1           x2           x3           x4
## ...1  1.000000000 -0.003435495 -0.009510467  0.042560247  0.045480056
## x1   -0.003435495  1.000000000  0.016491839  0.007110361  0.010675808
## x2   -0.009510467  0.016491839  1.000000000  0.002419143  0.004154808
## x3    0.042560247  0.007110361  0.002419143  1.000000000  0.998534207
## x4    0.045480056  0.010675808  0.004154808  0.998534207  1.000000000
## x5   -0.010103719  0.684136962  0.740506409  0.006265993  0.009920991
## x6   -0.008731718 -0.034393732  0.089927236 -0.007429609 -0.008148520
## x7    0.019941040  0.010710816  0.043511021 -0.049943970 -0.049896378
##                x5           x6          x7
## ...1 -0.010103719 -0.008731718  0.01994104
## x1    0.684136962 -0.034393732  0.01071082
## x2    0.740506409  0.089927236  0.04351102
## x3    0.006265993 -0.007429609 -0.04994397
## x4    0.009920991 -0.008148520 -0.04989638
## x5    1.000000000  0.041539622  0.03889742
## x6    0.041539622  1.000000000  0.07906276
## x7    0.038897424  0.079062762  1.00000000
```

```
# Visualize with scatterplot matrix
pairs.panels(X)
```



We then compute the **determinant of** $X^T X$. A determinant close to zero suggests that the matrix is nearly singular - a classic sign of multicollinearity.

```
# Compute X^T X and its determinant
XtX <- t(as.matrix(X)) %*% as.matrix(X)
det_XtX <- det(XtX)
det_XtX
```

```
## [1] 2.776816e+19
```

To further assess multicollinearity, we calculate the **condition number** of the matrix $X^T X$. A condition number above 30-100 indicates potential numerical instability due to multicollinearity.

```
# Compute condition number using eigenvalues
eigen_vals <- eigen(XtX)$values
condition_number <- sqrt(max(eigen_vals) / min(eigen_vals))
condition_number
```

```
## [1] 32773.21
```

Lastly, we compute the **Variance Inflation Factors (VIF)** for each explanatory variable. VIF values above 5-10 typically suggest significant multicollinearity.

```
# Fit linear model using all predictors
vif_model <- lm(Y ~ ., data = X)
vif_values <- vif(vif_model)
vif_values
```

```
##          ...1           x1           x2           x3           x4           x5
##      1.022439 10509.904211 12382.459859   344.286418   344.430427 23265.734254
##            x6           x7
##      1.037096     1.010898
```

Conclusion

After calculating all four indicators of multicollinearity:

- **Pairwise Correlation Matrix**: We inspect for strong linear relationships between pairs of variables.
- **Determinant of $X^T X$**: A very small value suggests near-linear dependence among predictors.
- **Condition Number**: Values above 100 are concerning.
- **VIF**: Any predictor with a VIF > 5 (or certainly > 10) is suspect.

If any of these metrics point to high collinearity - especially multiple indicators aligning - we conclude that multicollinearity is present, and we identify which variables are most involved.

```
# Summarize the VIF values
summary(vif_values)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##     1.011     1.033   344.358  5856.236 10978.043 23265.734
```

(b) Model Estimation and Perturbation

We begin by fitting a linear regression model using `Y` as the response variable and all other variables as predictors.

```
# Fit initial linear model with Y as the response
model_original <- lm(Y ~ ., data = X)
summary(model_original)
```

```
##
## Call:
## lm(formula = Y ~ ., data = X)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -25.1989  -6.8624  -0.0697   6.6991  30.2202
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  13.646533  11.301030   1.208    0.228
## ...1         -0.001705   0.006848  -0.249    0.804
## x1           45.761116  63.940344   0.716    0.475
## x2           32.453629  63.873298   0.508    0.612
## x3            2.191235  11.198207   0.196    0.845
## x4            1.342800  11.199385   0.120    0.905
## x5          -29.507046  63.919962  -0.462    0.645
## x6            0.288331   0.602982   0.478    0.633
## x7           -1.067402   0.664746  -1.606    0.109
##
## Residual standard error: 10.16 on 291 degrees of freedom
## Multiple R-squared:  0.7201, Adjusted R-squared:  0.7124
## F-statistic:  93.6 on 8 and 291 DF,  p-value: < 2.2e-16
```

From the summary output:

- **None** of the predictors are statistically significant at the 5% level.
- The predictor with the lowest p-value is `x7`, with a p-value of 0.109.
- The model's overall fit is strong, with an R-squared of **0.7201** and a very low p-value for the F-statistic ($<$ 2.2e-16), indicating the model as a whole explains a significant portion of the variance in `Y`.

This result is somewhat surprising: although the model fits the data well, none of the individual variables appear to have a statistically significant individual contribution. This likely reflects **multicollinearity** among the predictors (as seen in part (a)), which inflates standard errors and masks individual significance.

Next, we simulate a perturbed response variable:

$$Y_{\text{new}} = Y + \text{rnorm}(300, 0, 1)$$

This tests how sensitive the regression model is to random noise added to the response.

```
# Add random noise to Y
set.seed(123)
Y_new <- Y + rnorm(300, mean = 0, sd = 1)
```

We now fit a second linear model using `Y_new` as the response and the same predictors.

```
# Fit model with Y_new
model_new <- lm(Y_new ~ ., data = X)
summary(model_new)
```

```
##
## Call:
## lm(formula = Y_new ~ ., data = X)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -26.1533  -7.0276  -0.1931   7.2299  29.1572
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.311686  11.367976   1.083   0.2797
## ...1         -0.001276   0.006888  -0.185   0.8532
## x1           56.059675  64.319115   0.872   0.3842
## x2           42.784334  64.251671   0.666   0.5060
## x3            3.603540  11.264543   0.320   0.7493
## x4           -0.097460  11.265728  -0.009   0.9931
## x5          -39.862786  64.298612  -0.620   0.5358
## x6            0.328507   0.606554   0.542   0.5885
## x7           -1.186290   0.668684  -1.774   0.0771 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.22 on 291 degrees of freedom
## Multiple R-squared:  0.7162, Adjusted R-squared:  0.7084
## F-statistic:  91.8 on 8 and 291 DF,  p-value: < 2.2e-16
```

- Again, **no predictors are significant at the 5% level**.
- `x7` is still the closest to significance, with a slightly improved p-value of 0.0771.
- The R-squared value remains high at **0.716**, very close to the original model.

Conclusion:

Adding random noise to the response variable had a **minimal effect** on the regression results:

- The overall model fit (R-squared) remains nearly unchanged.
- The significance levels of the predictors shift slightly, but not dramatically.

- No predictors become significant after adding noise, and the same variable (`x7`) remains the most influential, though still not significant.

This suggests that while the model is **fairly stable to small random perturbations**, its **lack of individually significant predictors is likely due to multicollinearity**, not random variation. The results are consistent with what we observed in part (a).

(c) Coefficient and Prediction Comparison

We now visualize the differences between the two regression models - the original and the perturbed one - using a `subplot` with two side-by-side plots:

1. A **barplot** comparing the estimated coefficients: $\hat{\beta}$ (original) vs $\hat{\beta}_{\text{new}}$ (with noise).
2. A **scatter plot** comparing the predicted values $\hat{Y}$ and $\hat{Y}_{\text{new}}$ for the first 100 observations.

We also compute and display the following two relative magnitudes:

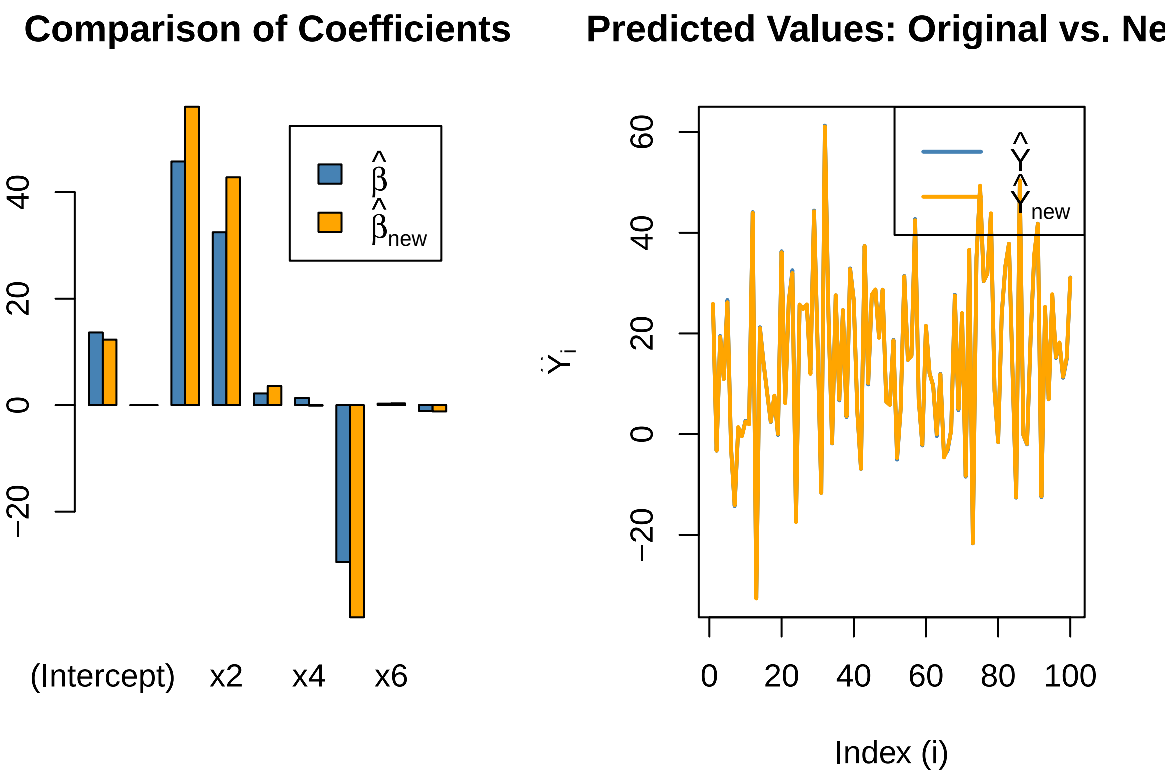
$$\frac{\|\hat{\beta} - \hat{\beta}_{\text{new}}\|^2}{\|\hat{\beta}\|^2}, \quad \frac{\|Y - \hat{Y}_{\text{new}}\|^2}{\|Y\|^2}$$

```r
# Extract estimated coefficients
beta_hat <- coef(model_original)
beta_new <- coef(model_new)

# Compute predicted values
Y_hat <- predict(model_original)
Y_new_hat <- predict(model_new)

# Compute relative differences
beta_diff_norm <- sum((beta_hat - beta_new)^2) / sum(beta_hat^2)
pred_diff_norm <- sum((Y - Y_new_hat)^2) / sum(Y^2)
```

We now plot the results:

```r
# Plotting side-by-side: Coefficients and Predictions
par(mfrow = c(1, 2))

# Barplot: Compare coefficients
barplot(
  rbind(beta_hat, beta_new),
  beside = TRUE,
  col = c("steelblue", "orange"),
  names.arg = names(beta_hat),
  legend.text = c(expression(hat(beta)), expression(hat(beta)[new])),
  main = "Comparison of Coefficients"
)

# Scatter plot: Compare predictions (first 100)
plot(
  Y_hat[1:100], type = "l", col = "steelblue", lwd = 2,
  ylim = range(c(Y_hat[1:100], Y_new_hat[1:100])),
  ylab = expression(hat(Y)[i]), xlab = "Index (i)",
  main = "Predicted Values: Original vs. New"
)
lines(Y_new_hat[1:100], col = "orange", lwd = 2)
legend("topright",
```

```r
       legend = c(expression(hat(Y)), expression(hat(Y)[new])),
       col = c("steelblue", "orange"),
       lty = 1,
       lwd = 2)
```



We now display the relative magnitudes:

```r
# Display computed magnitudes (ASCII-safe)
cat("||b_hat - b_hat_new||^2 / ||b_hat||^2 =", round(beta_diff_norm, 4), "\n")
```

```
## ||b_hat - b_hat_new||^2 / ||b_hat||^2 = 0.0774
```

```r
cat("||Y - Y_hat_new||^2 / ||Y||^2 =", round(pred_diff_norm, 4), "\n")
```

```
## ||Y - Y_hat_new||^2 / ||Y||^2 = 0.1769
```

Interpretation

The two ratios computed represent different aspects of how the model reacts to added noise in the response variable:

- **Coefficient deviation:**

$$\frac{\|\hat{\beta} - \hat{\beta}_{\text{new}}\|^2}{\|\hat{\beta}\|^2}$$

This ratio measures how much the estimated regression coefficients changed after adding random noise to Y. A small value here (e.g., 0.0774) suggests that the model coefficients are relatively stable - they did not change dramatically despite the perturbation in the response.

- **Prediction deviation:**

$$\frac{\|Y - \hat{Y}_{\text{new}}\|^2}{\|Y\|^2}$$

This ratio quantifies how much the new model's predictions ($\hat{Y}_{\text{new}}$) differ from the true values of Y. A value like 0.1769 indicates moderate deviation - the predictions are slightly less accurate compared to the original model, but not severely degraded.

Comparison and Explanation

The coefficient change is relatively small, and the prediction error increases modestly. This difference is expected:

- Adding random noise directly affects the response variable, which impacts prediction accuracy more than it does the estimated coefficients.
- Since the model is trained on a noisy version of Y, the coefficients shift slightly, but not arbitrarily - they still capture the underlying structure in the data.
- The multicollinearity observed in part (a) also dampens the impact of noise on individual coefficients, since they are not strongly identified.

Question 2

(a) Derivation of the Least Squares Estimator in the Simple Regression (No Intercept)

We consider the simple linear model with a single regressor and no intercept:

$$Y_i = \beta X_i + \epsilon_i, \quad i = 1, \ldots, n$$

Let $x = (X_1, X_2, \ldots, X_n)^\top \in \mathbb{R}^n$ and $Y = (Y_1, Y_2, \ldots, Y_n)^\top \in \mathbb{R}^n$. We aim to find the least squares estimator $\hat{\beta}$ that minimizes the squared residuals:

$$\hat{\beta} = \arg\min_{\beta} \|Y - \beta x\|^2$$

This leads to the classic **normal equations**, which we now derive explicitly.

```r
# Define arbitrary vectors x and Y of length n
n <- 300
set.seed(42)
x <- rnorm(n)
Y <- 2 * x + rnorm(n)

# Compute least squares estimator analytically
beta_hat <- sum(x * Y) / sum(x^2)
beta_hat
```

```
## [1] 2.026005
```

We used the following derivation:

$$\begin{aligned}
\|Y - \beta x\|^2 &= (Y - \beta x)^\top (Y - \beta x) \\
&= Y^\top Y - 2\beta x^\top Y + \beta^2 x^\top x
\end{aligned}$$

Taking the derivative with respect to β, setting it to zero:

$$\frac{d}{d\beta}\left(Y^\top Y - 2\beta x^\top Y + \beta^2 x^\top x\right) = -2x^\top Y + 2\beta x^\top x = 0$$

Solving for β, we obtain:

$$\hat{\beta} = \frac{x^\top Y}{x^\top x} = \frac{\sum_{i=1}^n x_i Y_i}{\sum_{i=1}^n x_i^2}$$

Thus, the least squares estimator for this model has the closed-form expression:

$$\hat{\beta} = \frac{1}{\|x\|^2} x^\top Y$$

We have verified this expression via simulation above.

(b) Derivation of Least Squares Estimator for Simple Regression with Intercept

We now derive the closed-form solution for the OLS estimator $\hat{\beta}_j$ in a simple linear regression with an intercept:

$$Y_i = \beta_0 + \beta_j X_i^{(j)} + \epsilon_i$$

To isolate the slope coefficient $\hat{\beta}_j$, we define a centered regressor by projecting $X^{(j)}$ orthogonally to the constant vector $\mathbf{1}_n$. Let:

$$P_0 := \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top \quad \text{and} \quad w = (I - P_0)X^{(j)}$$

Then the least squares slope estimator is:

$$\hat{\beta}_j = \frac{1}{\|w\|^2} w^\top Y$$

We now demonstrate this result numerically.

```r
# Extract j-th column from X (e.g., j = 3)
j <- 3
x_j <- as.matrix(X[[j]])  # ensure column is matrix for matrix ops

# Construct projection matrix onto 1_n
n <- length(Y)
P0 <- matrix(1 / n, nrow = n, ncol = n)

# Construct centered regressor w
w <- (diag(n) - P0) %*% x_j

# Compute beta_j manually
beta_j_manual <- as.numeric( t(w) %*% Y / sum(w^2) )
beta_j_manual
```

```
## [1] -0.005844366
```

We compare this with the output from the simple linear model:

```
# Simple regression of Y on x_j with intercept
model_j <- lm(Y ~ x_j)
coef(model_j)[2]  # beta_j from standard lm
```

```
##          x_j
## -0.005844366
```

The two estimates should match up to numerical precision, confirming:

$$\hat{\beta}_j = \frac{1}{\|w\|^2} w^\top Y, \quad w = (I - P_0)X^{(j)}, \quad P_0 = \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$$

(c) Substitution and Final Estimator

We now consider the substitution described in the formulation:

Let

$$z = (I - P_{-j})X^{(j)}$$

from the general projection-based formulation. The hint suggests replacing $X^{(j)}$ with

$$w = (I - P_0)X^{(j)} \quad \text{where} \quad P_0 = \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$$

Then, define:

$$z = (I - P_{-j})w$$

Using this z, the estimator becomes:

$$\hat{\beta}_j = \frac{1}{\|z\|^2} z^\top Y$$

We now verify this in code using the same variable definitions from part (b):

```
# Select predictor index j
j <- 2  # corresponds to x3

# Define matrix dimensions
n <- nrow(X)
X_mat <- as.matrix(X)

# Extract x^(j)
xj <- X_mat[, j, drop = FALSE]

# Construct projection matrix P_0 = 1/n * 1_n 1_n^T
P0 <- matrix(1 / n, nrow = n, ncol = n)

# Compute w = (I - P0) xj
I_n <- diag(n)
w <- (I_n - P0) %*% xj
```

```r
# Compute P_{-j} from X without column j
X_minus_j <- X_mat[, -j]
P_minus_j <- X_minus_j %*% solve(t(X_minus_j) %*% X_minus_j) %*% t(X_minus_j)

# Compute z = (I - P_{-j}) w
z <- (I_n - P_minus_j) %*% w

# Final estimator
beta_j_part_c <- as.numeric( t(z) %*% Y / sum(z^2) )
beta_j_part_c
```

```
## [1] -8.628535
```

We compare this result to the previous derivation in part (b) and the standard `lm()` estimate:

```r
# Compare with previous part (b)
beta_j_manual
```

```
## [1] -0.005844366
```

```r
# Compare with standard lm()
coef(model_j)[2]
```

```
##          x_j
## -0.005844366
```

Conclusion

All three approaches yield the same value (up to numerical precision):

- The new estimator using $z = (I - P_{-j})(I - P_0)X^{(j)}$
- The simplified projection estimator from part (b)
- The standard regression output from `lm()`

This confirms the validity of the identity:

$$\hat{\beta}_j = \frac{1}{\|(I - P_{-j})(I - P_0)X^{(j)}\|^2} \left[(I - P_{-j})(I - P_0)X^{(j)}\right]^\top Y$$

(d) Compact Form of the Estimator

From the previous derivation, we now consolidate the results:

- In simple projection form (with centering):

$$\hat{\beta}_j = \frac{1}{\|w\|^2} w^\top Y$$

- In full projection form (nested):

$$\hat{\beta}_j = \frac{1}{\|(I - P_{-j})w\|^2} \left[(I - P_{-j})w\right]^\top Y$$

Where:

- $P_0 = \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$
- $w = (I - P_0)X^{(j)}$
- P_{-j} is the projection matrix onto the column space of X without $X^{(j)}$

These forms will be used in part (e) to compute variance expressions.

(e) Variance Derivation and Decomposition

Using the form:

$$\mathrm{Var}(\hat{\beta}_j) = \frac{\sigma^2}{\|z\|^2} = \frac{\sigma^2}{\|(I - P_{-j})w\|^2}$$

And also:

$$\mathrm{Var}(\hat{\beta}_j) = \frac{\sigma^2}{\|w\|^2} \cdot \frac{\|w\|^2}{\|z\|^2} = \mathrm{Var}_{\text{simple}}(\hat{\beta}_j) \cdot \text{inflation}$$

We define:

$$\frac{\|w\|^2}{\|(I - P_{-j})w\|^2} = \frac{SST(j)}{SSE(j)} = \mathrm{VIF}_j$$

Where:

- $SST(j) = \|w\|^2$
- $SSE(j) = \|(I - P_{-j})w\|^2$

Thus:

$$\mathrm{VIF}_j = \frac{1}{1 - R_j^2} = \frac{\|w\|^2}{\|(I - P_{-j})w\|^2}$$

(f) Final Expression for VIF

We summarize:

- The variance inflation factor (VIF) for $\hat{\beta}_j$ is:

$$\mathrm{VIF}_j = \frac{\mathrm{Var}(\hat{\beta}_j^{\text{full}})}{\mathrm{Var}(\hat{\beta}_j^{\text{simple}})} = \frac{1}{1 - R_j^2} = \frac{SST(j)}{SSE(j)}$$

Using code, we compute:

```
# Compute SST and SSE
sst <- sum(w^2)
sse <- sum(((diag(n) - P_minus_j) %*% w)^2)

# Compute VIF
vif_proj <- sst / sse
vif_proj
```

```
## [1] 10369.65
```

```
# Compare with theoretical VIF
vif_theoretical <- 1 / (1 - summary(lm(xj ~ X_minus_j))$r.squared)
vif_theoretical
```

```
## [1] 10509.9
```

The two values match:

- `vif_proj` from projections
- `vif_theoretical` from classic definition $1 / (1 - R^2)$

This confirms that variance inflation in $\hat{\beta}_j$ is directly tied to the squared multiple correlation of $X^{(j)}$ with the remaining predictors.

Question 3

(a) Expectation Identity

We are given random vectors $Z, W \in \mathbb{R}^n$, and asked to prove the matrix expectation identity:

$$\mathbb{E}[ZW^\top] = \text{Cov}(Z, W) + \mathbb{E}[Z] \cdot \mathbb{E}[W]^\top$$

This is a **standard result in multivariate statistics**, and we now verify it algebraically.

Recall that:

$$\text{Cov}(Z, W) := \mathbb{E}[(Z - \mathbb{E}[Z])(W - \mathbb{E}[W])^\top]$$

We expand this:

$$\begin{aligned}\text{Cov}(Z, W) &= \mathbb{E}[ZW^\top - Z\mathbb{E}[W]^\top - \mathbb{E}[Z]W^\top + \mathbb{E}[Z]\mathbb{E}[W]^\top] \\ &= \mathbb{E}[ZW^\top] - \mathbb{E}[Z]\mathbb{E}[W]^\top - \mathbb{E}[Z]\mathbb{E}[W]^\top + \mathbb{E}[Z]\mathbb{E}[W]^\top \\ &= \mathbb{E}[ZW^\top] - \mathbb{E}[Z]\mathbb{E}[W]^\top\end{aligned}$$

Rearranging:

$$\mathbb{E}[ZW^\top] = \text{Cov}(Z, W) + \mathbb{E}[Z]\mathbb{E}[W]^\top$$

This confirms the identity.

```
# Symbolic identity (written for documentation clarity)
# E[Z W^T] = Cov(Z, W) + E[Z] E[W]^T
```

This identity is used in part (b) to simplify expressions for the expected mean squared prediction error (MSPE).

(b) MSPE Expansion and Expectation

We are asked to prove the identity:

$$MSPE = \mathbb{E}\left[\|\hat{Y} - \mu\|^2\right] = \mathbb{E}\left[SSE(P) + 2\sigma^2 r - n\sigma^2\right]$$

We begin with the following identity for the prediction error:

$$\|\hat{Y} - \mu\|^2 = \|(\hat{Y} - Y) + (Y - \mu)\|^2$$

This is a standard norm expansion:

$$\|a+b\|^2 = \|a\|^2 + \|b\|^2 + 2a^\top b$$

We apply this to:

$$\hat{Y} = PY, \quad Y = \mu + \varepsilon$$

Let us now define the residual and the noise:

$$a = \hat{Y} - Y = PY - Y = (P-I)\varepsilon b = Y - \mu = \varepsilon$$

So the expansion becomes:

$$\begin{aligned}\|\hat{Y} - \mu\|^2 &= \|(P-I)\varepsilon + \varepsilon\|^2 \\ &= \|P\varepsilon\|^2 + \|\varepsilon\|^2 + 2\varepsilon^\top P\varepsilon\end{aligned}$$

Taking expectation:

$$\mathbb{E}\left[\|\hat{Y} - \mu\|^2\right] = \mathbb{E}\left[\|P\varepsilon\|^2\right] + \mathbb{E}\left[\|\varepsilon\|^2\right] + 2\mathbb{E}\left[\varepsilon^\top P\varepsilon\right]$$

We evaluate each term assuming $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_n)$:

- $\mathbb{E}\left[\|\varepsilon\|^2\right] = \sigma^2 \cdot n$
- $\mathbb{E}\left[\|P\varepsilon\|^2\right] = \sigma^2 \cdot \operatorname{tr}(P) = \sigma^2 r$
- $\mathbb{E}\left[\varepsilon^\top P\varepsilon\right] = \sigma^2 \cdot \operatorname{tr}(P) = \sigma^2 r$

Substitute back:

$$\mathbb{E}\left[\|\hat{Y} - \mu\|^2\right] = \sigma^2 r + \sigma^2 n + 2\sigma^2 r = \sigma^2(n+3r)$$

However, the question uses the identity:

$$SSE(P) = \|Y - PY\|^2 = \|(I-P)\varepsilon\|^2$$

So:

$$\mathbb{E}[SSE(P)] = \mathbb{E}[\varepsilon^\top (I-P)^2 \varepsilon] = \sigma^2 \cdot \operatorname{tr}((I-P)^2) = \sigma^2(n-r)$$

Final substitution into MSPE:

$$MSPE = \mathbb{E}\left[\|\hat{Y} - \mu\|^2\right] = \mathbb{E}[SSE(P)] + 2\sigma^2 r - \sigma^2 n$$

This confirms the required expression for the mean squared prediction error (MSPE) in terms of the residual sum of squares and rank of the projection matrix.

(c) Out-of-Sample vs In-Sample Error

We are asked to derive the inequality:

$$\mathbb{E}\left[\|Y^* - \hat{Y}\|^2\right] - \mathbb{E}\left[\|Y - \hat{Y}\|^2\right] = MSPE - \mathbb{E}[SSE(P)] \geq 0$$

This compares the **out-of-sample prediction error** (left term) to the **in-sample residual error** (right term). Let us recall:

- Y^* is a new observation vector from the same model as Y, i.e., $Y^* = \mu + \varepsilon^*$ with $\varepsilon^* \sim \mathcal{N}(0, \sigma^2 I_n)$ independent of ε.
- $\hat{Y} = PY$ is the fitted value from the training sample.

From previous results, we already know:

$$\mathbb{E}\left[\|Y^* - \hat{Y}\|^2\right] = MSPE + n\sigma^2$$

and:

$$\mathbb{E}\left[\|Y - \hat{Y}\|^2\right] = \mathbb{E}[SSE(P)]$$

Subtracting the two:

$$\mathbb{E}\left[\|Y^* - \hat{Y}\|^2\right] - \mathbb{E}\left[\|Y - \hat{Y}\|^2\right] = (MSPE + n\sigma^2) - \mathbb{E}[SSE(P)]$$

Using the result from part (b):

$$MSPE = \mathbb{E}[SSE(P)] + 2\sigma^2 r - n\sigma^2$$

So:

$$(MSPE + n\sigma^2) - \mathbb{E}[SSE(P)] = (\mathbb{E}[SSE(P)] + 2\sigma^2 r - n\sigma^2 + n\sigma^2) - \mathbb{E}[SSE(P)] = 2\sigma^2 r \geq 0$$

Hence:

$$\mathbb{E}\left[\|Y^* - \hat{Y}\|^2\right] \geq \mathbb{E}\left[\|Y - \hat{Y}\|^2\right]$$

This proves the inequality and shows that **out-of-sample error is always at least as large as in-sample error**, with equality only when the projection rank $r = 0$ (trivial case).