Problem Set 4

Priyanshu Dey 703

2026-02-13

Problem 1: Demonstrating Multicollinearity

Consider the Credit data in the ISLR library. Choose `Balance` as the response and `Age`, `Limit`, and `Rating` as the predictors.

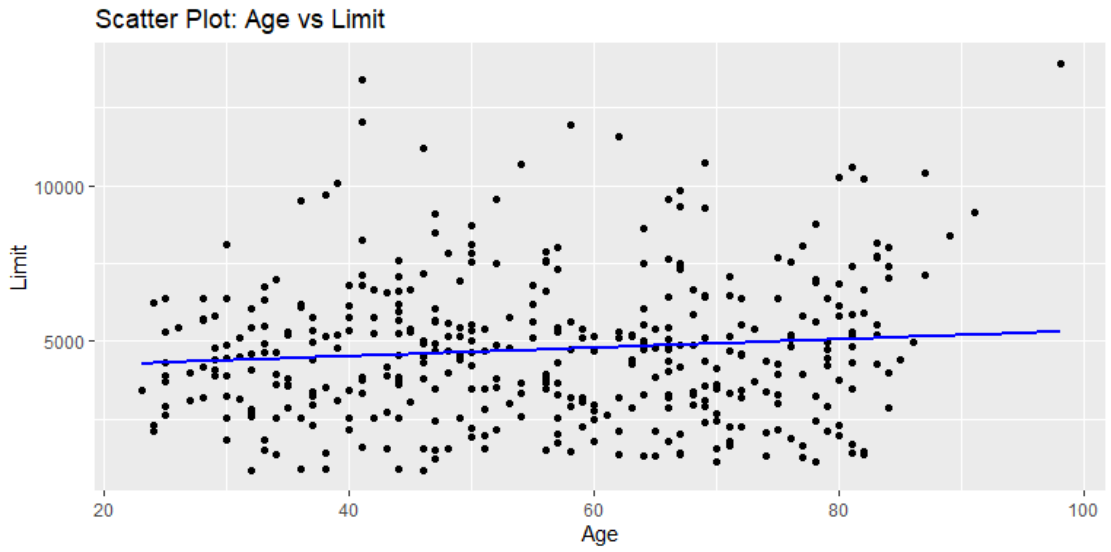
Data Preparation

```
data("Credit")
df = Credit
```

(a) Make a scatter plot of (i) Age versus Limit and (ii) Rating Versus Limit. Comment on the scatter plot.

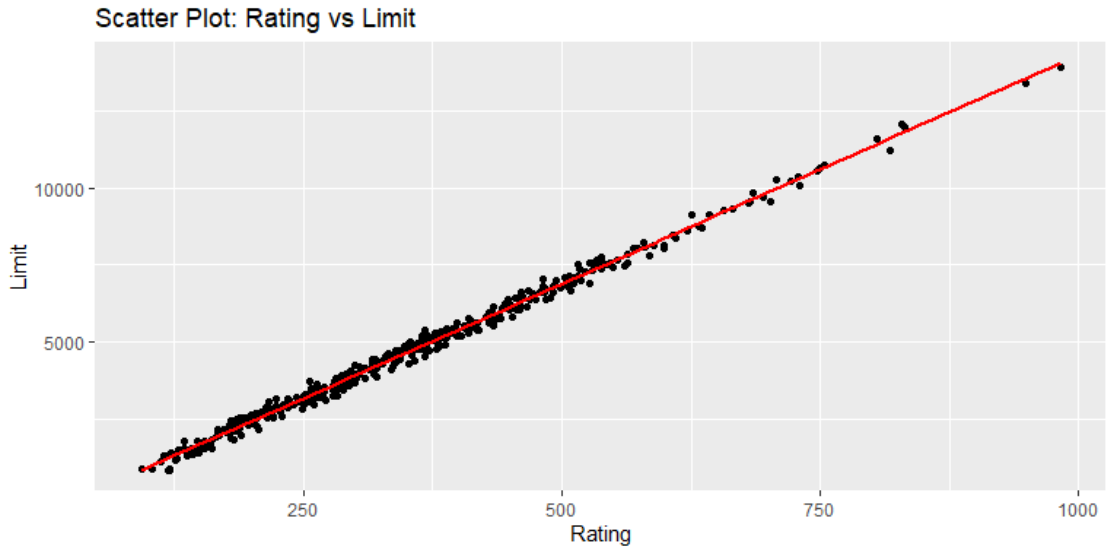
```
# Plot (i): Age vs Limit
p1 = ggplot(df, aes(x = Age, y = Limit)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "blue") +
  labs(title = "Scatter Plot: Age vs Limit")

# Plot (ii): Rating vs Limit
p2 = ggplot(df, aes(x = Rating, y = Limit)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Scatter Plot: Rating vs Limit")

# Display plots side by side (using gridExtra if available, or printing
sequentially)
# Here we print sequentially for simplicity
print(p1)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
print(p2)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
# Checking correlation
cor_age_limit = cor(df$Age, df$Limit)
cor_rating_limit = cor(df$Rating, df$Limit)
```

Comments:

1. **Age vs Limit:** The scatter plot shows a random cloud of points with no clear pattern. The correlation coefficient is very low (approx 0.1), indicating that Age and `Limit` are **uncorrelated**.
2. **Rating vs Limit:** The scatter plot shows an extremely strong positive linear relationship. The points lie almost perfectly on a straight line. The correlation coefficient is extremely high (approx 1), indicating **severe multicollinearity**

between Rating and Limit. This makes sense conceptually, as credit ratings are directly determined by credit limits.

(b) Run three separate regressions: (i) Balance on Age and Limit (ii) Balance on Age, Rating and Limit (iii) Balance on Rating and Limit. Present all the regression output in a single table using stargazer. What is the marked difference that you can observe from the output?

```
# Model (i): Balance ~ Age + Limit
model_1 = lm(Balance ~ Age + Limit, data = df)

# Model (ii): Balance ~ Age + Rating + Limit
model_2 = lm(Balance ~ Age + Rating + Limit, data = df)

# Model (iii): Balance ~ Rating + Limit
model_3 = lm(Balance ~ Rating + Limit, data = df)

stargazer(model_1, model_2, model_3,
          type = "text",
          title = "Regression Results: Multicollinearity Detection",
          column.labels = c("Age+Limit", "Age+Rating+Limit", "Rating+Limit"),
          keep.stat = c("n", "rsq", "adj.rsq", "ser", "f"))
```

```
##
## Regression Results: Multicollinearity Detection
##
================================================================================
=================
##                                          Dependent variable:
##                     ---------------------------------------------------
--------------------
##                                                Balance
##                            Age+Limit            Age+Rating+Limit
Rating+Limit
##                               (1)                    (2)
(3)
## --------------------------------------------------------------------------
--------------------
## Age                         -2.291***              -2.346***
##                              (0.672)                (0.669)
##
## Rating                                              2.310**
2.202**
##                                                     (0.940)
(0.952)
##
## Limit                        0.173***               0.019
0.025
##                              (0.005)                (0.063)
```

```
(0.064)
##
## Constant                   -173.411***             -259.518***
-377.537***
##                              (43.828)                (55.882)
(45.254)
##
## ---------------------------------------------------------------------------
--------------------
## Observations                  400                     400
400
## R2                           0.750                   0.754
0.746
## Adjusted R2                  0.749                   0.752
0.745
## Residual Std. Error    230.532 (df = 397)       229.080 (df = 396)
232.320 (df = 397)
## F Statistic         594.988*** (df = 2; 397) 403.718*** (df = 3; 396)
582.820*** (df = 2; 397)
##
===========================================================================
=================
## Note:                                                         *p<0.1;
**p<0.05; ***p<0.01
```

Marked Difference Observed:

- In **Model (i)**, `Limit` is highly significant and has a positive coefficient (approx 0.17).
- In **Model (ii)** and **Model (iii)**, when `Rating` is added, the standard error for `Limit` increases drastically (inflated variance).
- Crucially, in Model (ii), `Limit` changes from being the primary driver to potentially being insignificant or having its coefficient shift drastically (due to `Rating` absorbing the variance). This instability in the coefficients for `Limit` and `Rating` (despite them both being correlated with Balance) is a hallmark of multicollinearity.

(c) Calculate the variance inflation factor (VIF) and comment on multicollinearity.

```r
vif_values = vif(model_1)
vif_values3 = vif(model_3)
print(vif_values)
```

```
##      Age    Limit
## 1.010283 1.010283
```

```r
print(vif_values3)
```

```
##   Rating    Limit
## 160.4933 160.4933
```

Comment on Multicollinearity:

- **Age:** The VIF for `Age` is close to 1, indicating no multicollinearity with the other predictors.
- **Limit and Rating:** The VIF values for `Limit` and `Rating` are extremely high (typically > 160). A VIF exceeding 5 or 10 is considered problematic.
- **Conclusion:** There is **severe multicollinearity** between `Limit` and `Rating`. This confirms the observation from the scatter plot in part (a). Including both in the same model leads to unstable coefficient estimates and inflated standard errors, making it difficult to isolate the individual effect of each predictor on `Balance`.

Problem 2: Detection of Outliers, Leverage, and Influential Points

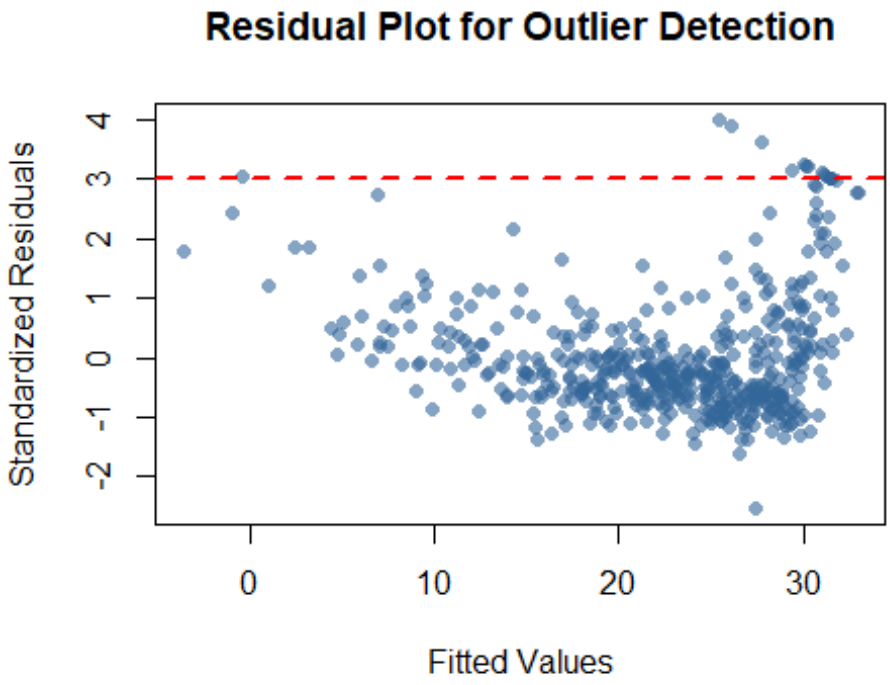
1. Multiple Linear Regression

```r
library("MASS")
data("Boston")

model = lm(medv ~ crim + nox + black + lstat, data = Boston)
summary(model)
```

```
##
## Call:
## lm(formula = medv ~ crim + nox + black + lstat, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.564  -4.004  -1.504   2.178  24.608
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.053584   2.170839  13.844   <2e-16 ***
## crim        -0.059424   0.037755  -1.574    0.116
## nox          3.415809   3.056602   1.118    0.264
## black        0.006785   0.003408   1.991    0.047 *
## lstat       -0.918431   0.050167 -18.307   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.183 on 501 degrees of freedom
## Multiple R-squared:  0.5517, Adjusted R-squared:  0.5481
## F-statistic: 154.1 on 4 and 501 DF,  p-value: < 2.2e-16
```

2. Residual Plot & Outlier Detection

```r
stand_res = rstandard(model)
fitted_vals = fitted(model)


plot(fitted_vals, stand_res,
     xlab = "Fitted Values",
     ylab = "Standardized Residuals",
     main = "Residual Plot for Outlier Detection",
     pch = 16, col = rgb(0.2, 0.4, 0.6, 0.6))

abline(h = c(-3, 3), col = "red", lty = 2, lwd = 2)
```

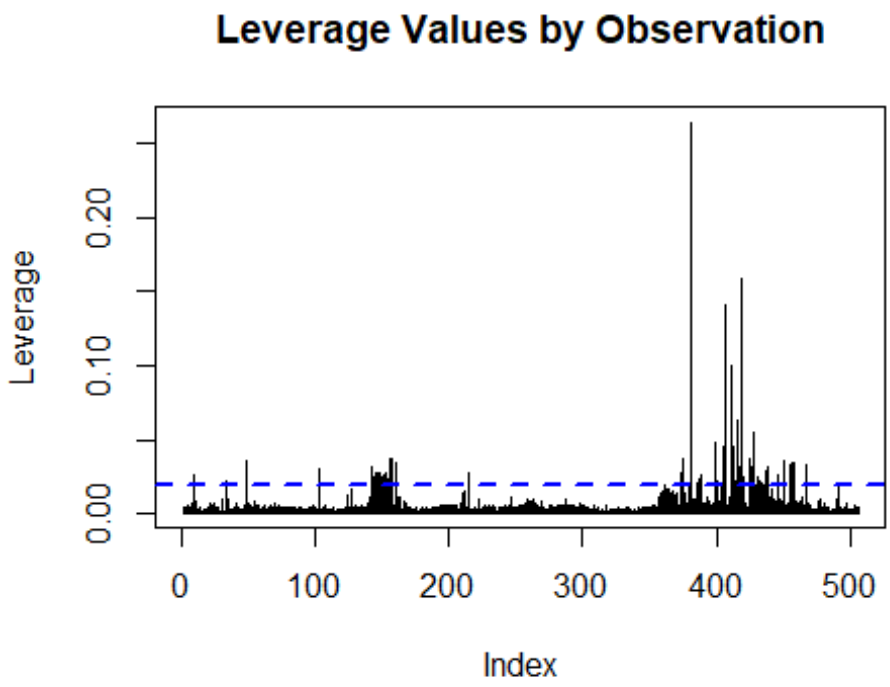



```r
outliers <- which(abs(stand_res) > 3)
cat("Observations identified as outliers:\n")

## Observations identified as outliers:

print(outliers)

## 167 187 196 205 226 258 263 268 284 369 370 372 373 413
## 167 187 196 205 226 258 263 268 284 369 370 372 373 413
```

Consistency Check: By looking at the graph, any points falling above the top red dashed linecorrespond exactly to the indices identified in the `outliers` calculation.

3. High Leverage Points (Hat Matrix)

```
p = 4
n = nrow(Boston)
leverage_threshold = 2 * (p + 1) / n


lev = hatvalues(model)


plot(lev, type = "h",
     main = "Leverage Values by Observation",
     ylab = "Leverage", xlab = "Index")
abline(h = leverage_threshold, col = "blue", lty = 2, lwd = 2)
```

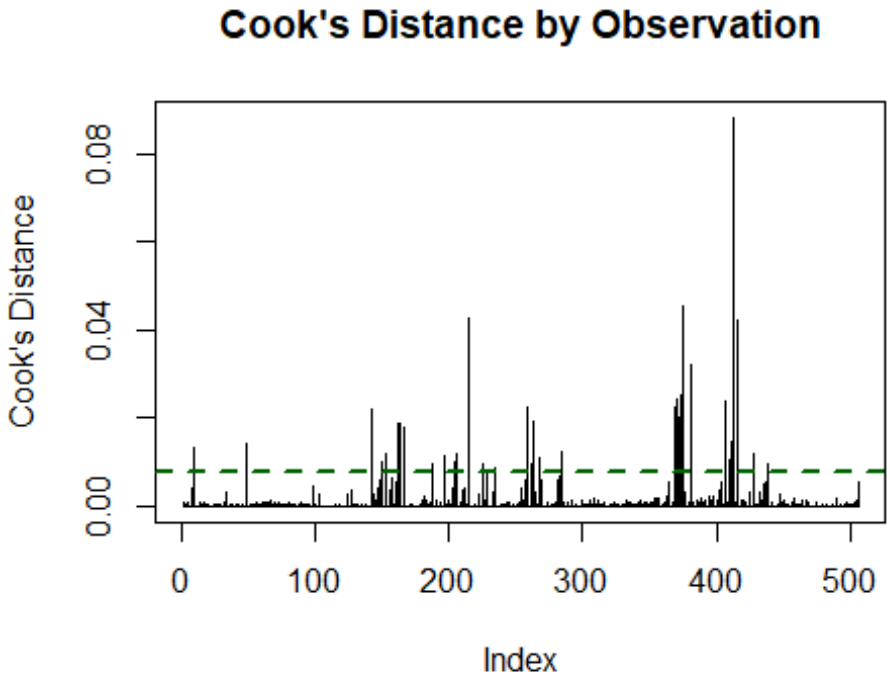



```
high_leverage = which(lev > leverage_threshold)
cat("Number of high leverage points detected:", length(high_leverage), "\n")

## Number of high leverage points detected: 64
```

4. Influential Points (Cook's Distance)

```
cooks_d = cooks.distance(model)
cooks_threshold = 4 / n


plot(cooks_d, type = "h",
```

```
     main = "Cook's Distance by Observation",
     ylab = "Cook's Distance", xlab = "Index")
abline(h = cooks_threshold, col = "darkgreen", lty = 2, lwd = 2)
```




```
influential_points <- which(cooks_d > cooks_threshold)
cat("Number of influential points detected:", length(influential_points),
"\n")

## Number of influential points detected: 37
```