Module 4 Lab

This lab covers a number of topics, and is divided into two sections. The first section uses a baseball dataset to proceed through the steps associated with fitting a multiple linear regression (MLR) model. These steps include exploratory data analysis, fitting the model, assessing the validity of the regression assumptions, and calculating case influence statistics. The second section is a simulation study that examines the impact of multicollinearity in the explanatory variables in a MLR model.

Section 1 - Major League Baseball Data

The Data and Initial Plots

The dataset we will use is a ficticious random sample of 28 Major League Baseball (MLB) teams season statistics. Each observation is one season for one team, and the variables are team totals in four offensive categories. The dataset `MLB.csv` is in the Module 4 Lab folder. Go ahead and load it in.

```r
MLB <- read.csv("MLB.csv")
head(MLB)
```

```
##     R  HR  BB   SO
## 1 685 130 519 1142
## 2 688 181 542 1384
## 3 745 212 416 1125
## 4 602 172 439 1230
## 5 598 148 411 1207
## 6 698 155 585 1245
```

Our question of interest is: what is the relationship between the response variable runs (`R`), and the explanatory variables home run (`HR`), base on balls (`BB`), and strikeout (`SO`). A good place to start is scatter plots of runs against each of the three individual predictors. Use this code to take a look at the plots.

```r
qplot(HR, R, data = MLB)
qplot(BB, R, data = MLB)
qplot(SO, R, data = MLB)
```

The relationships between (1) runs and home runs, and (2) runs and base on balls, looks positive and linear, with no noteworthy outliers. The relationship between runs and strikeouts also looks approximately linear, but negative, and also with no outliers.

Without further ado, let's go ahead and fit a model.

Multiple Linear Regression

The R code for multiple linear regression (MLR) is very similar to the R code for simple linear regression (SLR). We again use `lm()`, but we need to be sure to specify each explanatory variable we want to include in the model. The model we are fitting is:

$$Runs = \beta_0 + \beta_1 * HR + \beta_2 * BB + \beta_2 * SO + \epsilon_i,$$

where $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$.

```r
fit <- lm(R ~ HR + BB + SO, data = MLB)
```

Looking at the output, we see that all three explanatory variables are significant, with p-values less than 0.01. The next step is assessing the validity of the model assumptions.

Residual Diagnostics

One way to evaluate the validity of our assumptions is by looking at residual plots. With more explanatory variables, we have more residual plots to look at.

```r
qplot(HR, residuals(fit), data = MLB) + geom_hline(aes(yintercept=0))
qplot(BB, residuals(fit), data = MLB) + geom_hline(aes(yintercept=0))
qplot(SO, residuals(fit), data = MLB) + geom_hline(aes(yintercept=0))

qplot(fitted(fit), residuals(fit), data = fit) + geom_hline(aes(yintercept=0))
```

The residuals versus explanatory variable plots give no indication of non-linearity. The residuals versus fitted values plot also does not raise any red flags: no dramatic indication of non-normality, non-constant variance, or non-independence.

Case-Influence Statistics

In general, one of the concerns with outliers is that an invalid observation could inappropriately have a large impact on inference. In the case of suspicious outliers, we would want to know the effect of the observation on the MLR, and case influence statistics help analyze that effect. We look at two case influence statistics here— leverage and Cook's distance.

Leverage

One category of influential observations, in terms of the β parameter estimates, are those far from the "center" of the explanatory variables. Such a point is said to have "high leverage," which can be quantified with the corresponding diagonal element of something called the hat matrix. The function `influence()` calculates, among other statistics, the diagonal elements of the hat matrix.

```r
CIS <- data.frame(influence(fit))
names(CIS) # Notice the element "hat"
qplot(1:28, hat, data = CIS)
MLB[CIS$hat > 0.2,]
MLB[CIS$hat > 0.3,]
```

Assessing the plot visually, four observations stand out as having greater (two much greater) leverage than the rest. The last two lines of code identify the high leverage observations. If we had noticed outliers in our initial plots of the explanatory variables against the response, then we would want to know if the outliers had high leverage.

Cook's Distance

The "Cook's D" statistic measures the overall impact of an observation on the $\hat{\beta}$ values. In other words, an observation's Cook's D will be large if its removal would result in a large change in the β estimates.

```
Cook <- cooks.distance(fit)
qplot(1:28, Cook)
MLB[Cook > 0.6,]
```

Notice that observation 10 is among the most influential observations in terms of β estimates (Cook's Distance), and in terms of leverage. If observation ten had been an outlier, then it would be up to the researcher to decide if observation ten is valid. It is very important to note–an observation **should not be removed** simply because it is influential.

Section Two - Multicollinearity

When the explanatory variables are correlated, it inflates the estimation variance of the β parameters. This section demonstrates this fact through simulations.

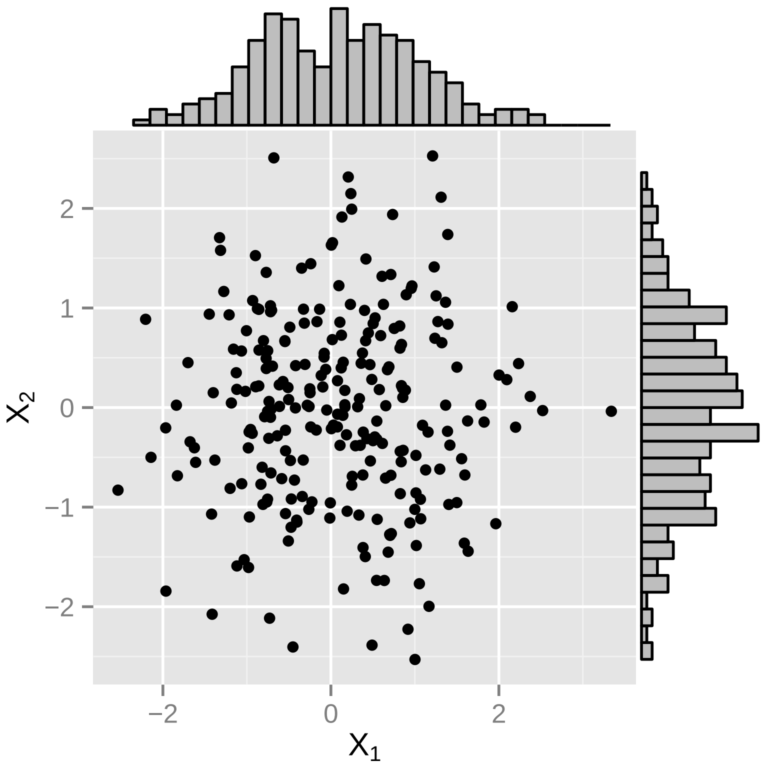
Simulation Study

First, define the following model:

$$Y_i = 0.5 + 0.3X_{1i} + 0.7X_{2i} + \epsilon_i,$$

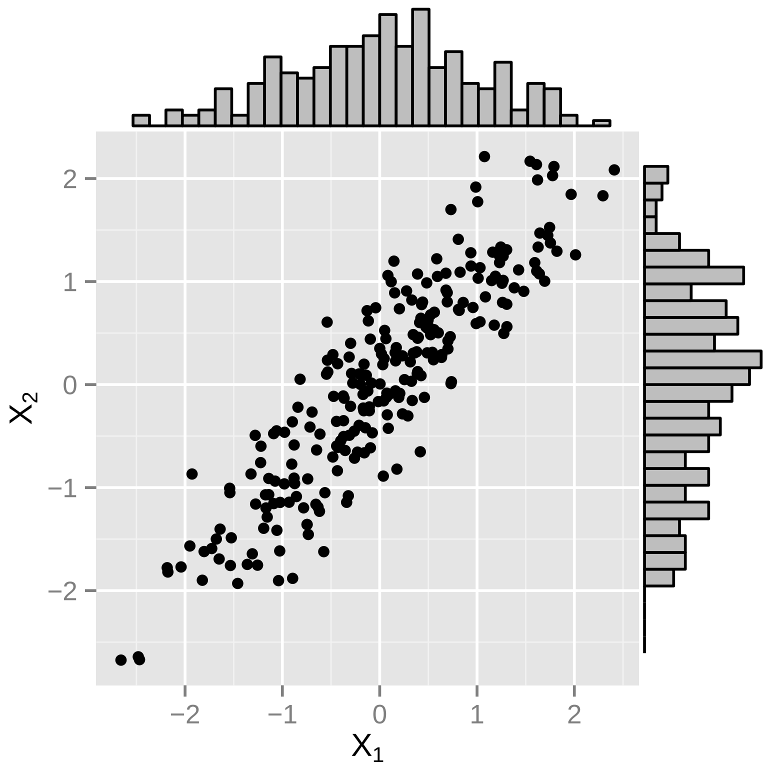
where $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$. The choice of coefficients (0.5, 0.3, 0.7) was arbitrary; we just need a model to work with. Now, consider two scenarios:

- X_{1i} and X_{2i} are uncorrelated—the scenario we are used to.



Notice that the individual histograms for X_1 and X_2 are very similar, reflecting the fact that, when considered separately, they have identical distributions to one another.

- X_{1i} and X_{2i} are highly correlated—multicollinear, the scenario we are learning about!



Notice the histograms in the margins again look very similar. In fact, despite begin correlated, when considered separately each distribution is N(0,1).

The issue of interest here is: what happens to the variance of the estimates of β_0, β_1, and β_2 when X_1 and X_2 are correlated? In other words, is it harder to estimate the β coefficients when X_1 and X_2 are correlated? Let's run a simulation to find out. Here are the steps.

1. Define $\beta_0 = 0.5$, $\beta_1 = 0.3$, and $\beta_2 = 0.7$
2. Define the mean of X_1 and X_2
3. Generate correlated/uncorrelated X_1 and X_2 data
4. Generate the response variable; use model equation and add N(0,1) noise
5. Fit a MLR model
6. Extract the coefficient estimate; $\hat{\beta}_0$, $\hat{\beta}_1$, or $\hat{\beta}_2$
7. Repeat steps (3) through (6) many times.

Let's start by writing a function that will accomplish steps (3) through (6).

```r
fitmodel <- function(beta, cov){ # Beta = 1, 2, 3 - specify coefficient
  X <- mvrnorm(n = 75, mu = mu, Sigma = cov) # Generate uncorrelated/correlated data
  X1 <- X[,1] # First column of X
  X2 <- X[,2] # Second column of X
  Y <- beta0 + beta1*X1 + beta2*X2 + rnorm(75, 0, 1) # Generate/calculate response
  fit <- lm(Y ~ X1 + X2) # Fit the model
  fit$coefficients[beta] # Return specified coef - "beta" argument
}
```

Our function will take two arguments: the β coefficient ($1 = \beta_0$, $2 = \beta_1$, or $3 = \beta_2$) we want to estimate, and a covariance matrix. The second line generates bivariate Normal data, using the specified covariance matrix. I chose n = 75 to provide enough length in each set of observations to create reasonable estimates. The third line creates the response variable by calculating $Y_i = 0.5 + 0.3X_{1i} + 0.7X_{2i}$, and then adding a randomly generated N(0,1) number to it. The fourth line fits a MLR model, and the fifth line extracts $\hat{\beta}_0$, $\hat{\beta}_1$, or $\hat{\beta}_2$ from the model, depending on whether 1, 2, or 3 was specified in the `fitmodel(beta = ?, cov)` function call.

Let's try it for β_0 in the uncorrelated explanatory variables scenario. Remember, we need to complete steps (1), (2), and (7) with the function we just created.

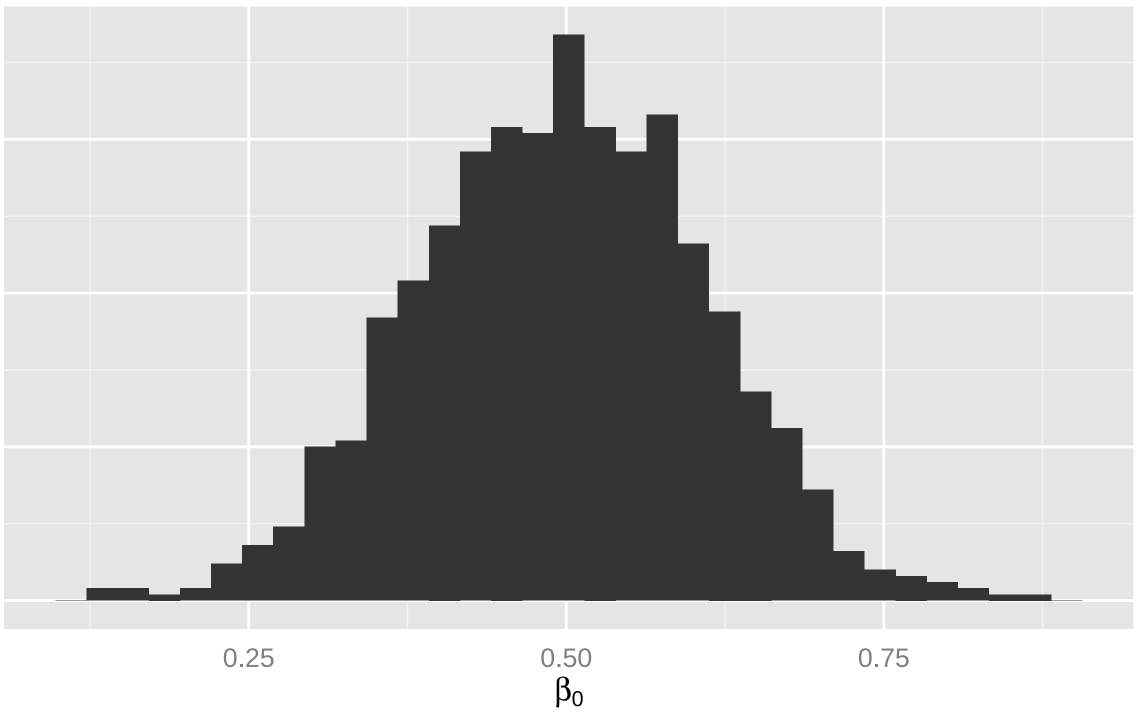
```r
# Step 1
beta0 <- 0.5 # define beta_0
beta1 <- 0.3 # define beta_1,
beta2 <- 0.7 # define beta_2

# Step 2
mu <- matrix(c(0,0)) # Set means for X_1, X_2
sigma1 <- matrix(c(1, 0, 0, 1), ncol = 2) # Cov Matrix: Cov(X_1, X_2) = 0

# Step 7
set.seed(1822) # Francis Galton born, invented regression concept
beta0_estimates <- replicate(1000, fitmodel(1, sigma1)) # 1st coef --> beta_0
sd(beta0_estimates)
```

```
## [1] 0.1132952
```

```r
# Display results
qplot(beta0_estimates) + ylab("") +
  theme(axis.ticks = element_blank(), axis.text.y = element_blank()) +
  xlab(expression(beta[0]))
```

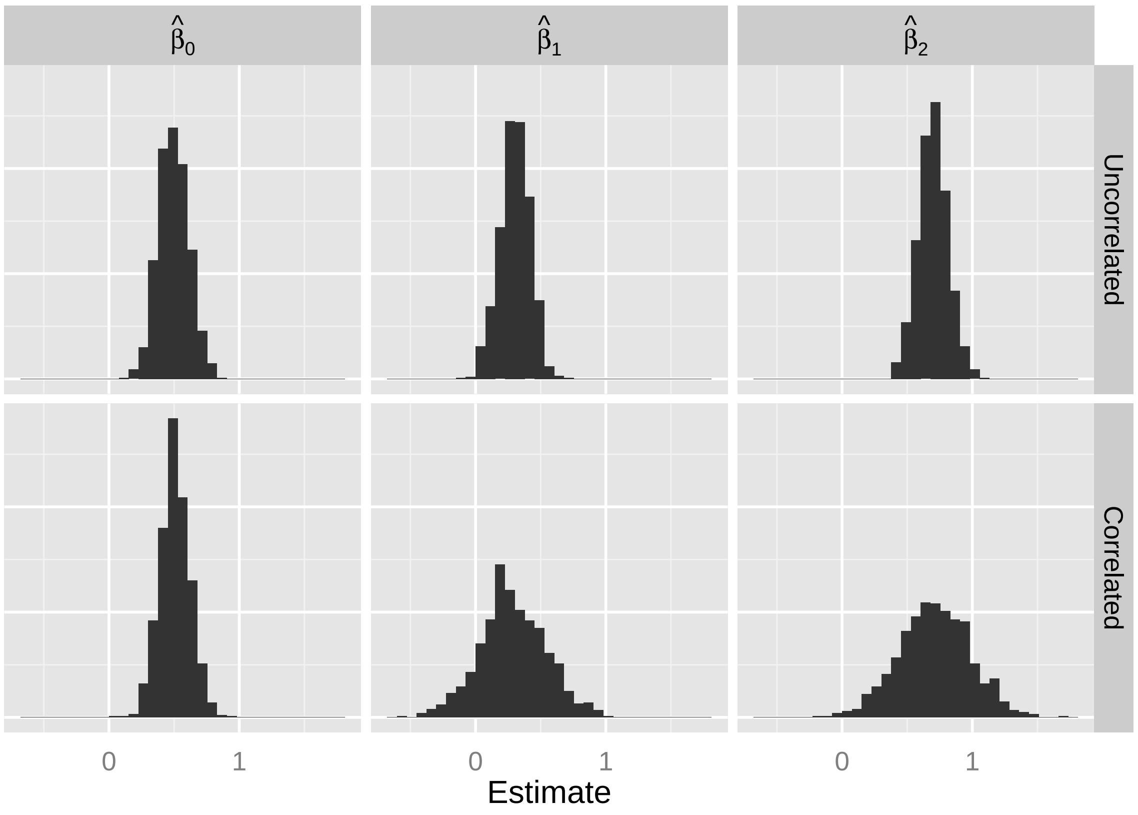


Notice that the covariance matrix we specified in "Step 2" gives independent N(0,1) random variables for X_1 and X_2. The function `sd()` gives the standard error of our estimates, about 0.11. We see in the histogram that our estimates are clustered around 0.5, the true value of β_0.

Now it is your turn to do the same thing for β_1 and β_2 in the uncorrelated case; and β_0, β_1, and β_2 in the correlated case. As you run the simulations, fill in the standard errors in the table below. Note: In the correlated case, use `sigma2 <- matrix(c(1, 0.9, 0.9, 1), ncol = 2)` to define the covariance matrix.

Parameter	$SE(\hat{\beta}_i)$
Uncorrelated	
β_0	0.11
β_1	
β_2	
Correlated	
β_0	
β_1	
β_2	

If you plot the results, you should see something similar to my results below.



The variances of $\hat{\beta}_1$ and $\hat{\beta}_2$ are much larger when they are correlated than when they are uncorrelated. Does it make sense that $\hat{\beta}_0$ is unaffected?