BTRY 4030 - Fall 2018 - Homework 1 Q1

Put Your Name and NetID Here

Due Friday, September 14, 2018

Instructions:

You may either respond to the questions below by editing the hw1_2018_q1.Rmd to include your answers and compiling it, or by handwriting your answers and scanning these in.

You may discuss the homework problems and computing issues with other students in the class. However, you must write up your homework solution on your own. In particular, do not share your homework RMarkdown file with other students.

Question 1

a. Consider the matrix:

$$A = \begin{pmatrix} 3 & 6 & 0 \\ 2 & 0 & 1 \\ -3 & 0 & 2 \end{pmatrix}.$$

Show that $\mathbf{x}^T A\mathbf{x} = \frac{1}{2}\mathbf{x}^T(A + A^T)\mathbf{x}$ for *all* values of $\mathbf{x} = (x_1, x_2, x_3)^T$.

For this specific matrix, we have that

$$A\mathbf{x} = \begin{pmatrix} 3x_1 + 6x_2 \\ 2x_1 - x_3 \\ -3x_1 + 2x_3 \end{pmatrix}$$

and

$$\mathbf{x}^T A\mathbf{x} = 3x_1^2 + 6x_1x_2 + 2x_1x_2 - x_2x_3 - 3x_1x_3 + 2x_3^2$$

similarly,

$$\mathbf{x}^T A^T\mathbf{x} = 3x_1^2 + 2x_1x_2 - 3x_1x_3 + 6x_1x_2 + x_2x_3 + 2x_3^2$$

which has all the same terms.

More generally, if we look at the dimension of the quantity $\mathbf{x}^T A\mathbf{x}$ it is seen that we are multiplying objects with dimensions $(1 \times 3) \times (3 \times 3) \times (3 \times 1)$, which leaves us with a 1×1 scalar term. Therefore $\mathbf{x}^T A\mathbf{x} = (\mathbf{x}^T A\mathbf{x})^T = \mathbf{x}^T A^T\mathbf{x}$. Using this result and substituting into the above gives $\frac{1}{2}\mathbf{x}^T(A + A^T)\mathbf{x} = \frac{1}{2}(\mathbf{x}^T A\mathbf{x} + \mathbf{x}^T A^T\mathbf{x}) = \frac{1}{2}(2\mathbf{x}^T A\mathbf{x}) = \mathbf{x}A^T\mathbf{x}$ which is the desired result.

b. Show that this is true for any square matrix A.

For any $n \times n$ square matrix A we have the same situation as in part 1a but the dimensions of the quantity $\mathbf{x}^T A\mathbf{x}$ is now $(1 \times n) \times (n \times n) \times (n \times 1)$, which leaves us with a 1×1 scalar, and all the results from 1a hold.

c. Let D be the matrix

$$D = \text{diag}(\mathbf{d}) = \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{pmatrix}.$$

Show that DA above multiplies the rows of A (given in part a) by the corresponding entry of $\mathbf{d}$ and that AD multiplies the columns.

Consider some square 3 × 3 matrix A.

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}.$$

We can carry out the matrix multiplication DA and look entry by entry, we get

$$DA = \begin{pmatrix} d_1a_{11} + 0a_{21} + 0a_{31} & d_1a_{12} + 0a_{22} + 0a_{32} & d_1a_{13} + 0a_{23} + 0a_{33} \\ 0a_{11} + d_2a_{21} + 0a_{31} & 0a_{12} + d_2a_{22} + 0a_{23} & 0a_{13} + d_2a_{23} + 0a_{33} \\ 0a_{11} + 0a_{21} + d_3a_{31} & 0a_{12} + 0a_{22} + d_3a_{32} & 0a_{13} + 0a_{23} + d_3a_{33} \end{pmatrix},$$

or more simply

$$DA = \begin{pmatrix} d_1a_{11} & d_1a_{22} & d_1a_{13} \\ d_2a_{21} & d_2a_{22} & d_2a_{23} \\ d_3a_{31} & d_3a_{32} & d_3a_{33} \end{pmatrix},$$

which is recognizable as A with rows multiplied by corresponding entries in D.

Now looking at AD we follow a similar procedure to get,

$$AD = \begin{pmatrix} d_1a_{11} + 0a_{12} + 0a_{13} & 0a_{11} + d_2a_{12} + 0a_{13} & 0a_{11} + 0a_{12} + d_3a_{13} \\ d_1a_{21} + 0a_{22} + 0a_{23} & 0a_{21} + d_2a_{22} + 0a_{23} & 0a_{21} + 0a_{22} + d_3a_{23} \\ d_1a_{31} + 0a_{32} + 0a_{33} & 0a_{31} + d_2a_{32} + 0a_{33} & 0a_{31} + 0a_{32} + d_3a_{33} \end{pmatrix},$$

and again simplifying yields,

$$AD = \begin{pmatrix} d_1a_{11} & d_2a_{12} & d_3a_{13} \\ d_1a_{21} & d_2a_{22} & d_3a_{23} \\ d_1a_{31} & d_2a_{32} & d_3a_{33} \end{pmatrix},$$

which can be seen as multiplying the columns of A by the respective entries in D.

d. Generalize the result in part c. to let D be a $n \times n$ diagonal matrix with diagonal elements d_i and $A = A_{ij}$ be a square matrix of dimension $n \times n$. Show that DA corresponds to multiplying the ith row of A by d_i and that AD is the corresponding operation on columns.

To generalize the results in 1c, consider looking at the i, j entry of DA, i.e.

$$DA_{i,j} = D_{i,:}A_{:,j},$$

where $D_{i,:}$ represents the i^{th} row in D and $A_{:,j}$ represents taking the j^{th} column of A. Knowing that $D_{i,:}$ is all zeros except for the i^{th} entry, d_i, means that this product of two vectors is equivalent to

$$DA_{i,j} = D_{i,:}A_{:,j} = d_iA_{ij}.$$

Since this is done for any row and column pair, we see that for every row i, the elements in that row will be the elements in that row of A, but multiplied by d_i, so the matrix product DA is equivalent to multiplying the rows of A by the corresponding entries in D.

Now to look at AD we consider a similar product,

$$AD_{i,j} = A_{i,:}D_{:,j},$$

and here we have that all entries of $D_{:,j}$ are zero except the j^{th} entry, d_j. Therefore we get

$$AD_{i,j} = A_{i,:}D_{:,j} = A_{ij}d_j.$$

This is equivalent to all entries column J of AD being the original elements of A but multiplied by d_j, giving that AD is equivalent to multiplying the columns of A by the corresponding entries in D. This gives that the result from the 3 × 3 case in 1c holds for any sized square matrices.

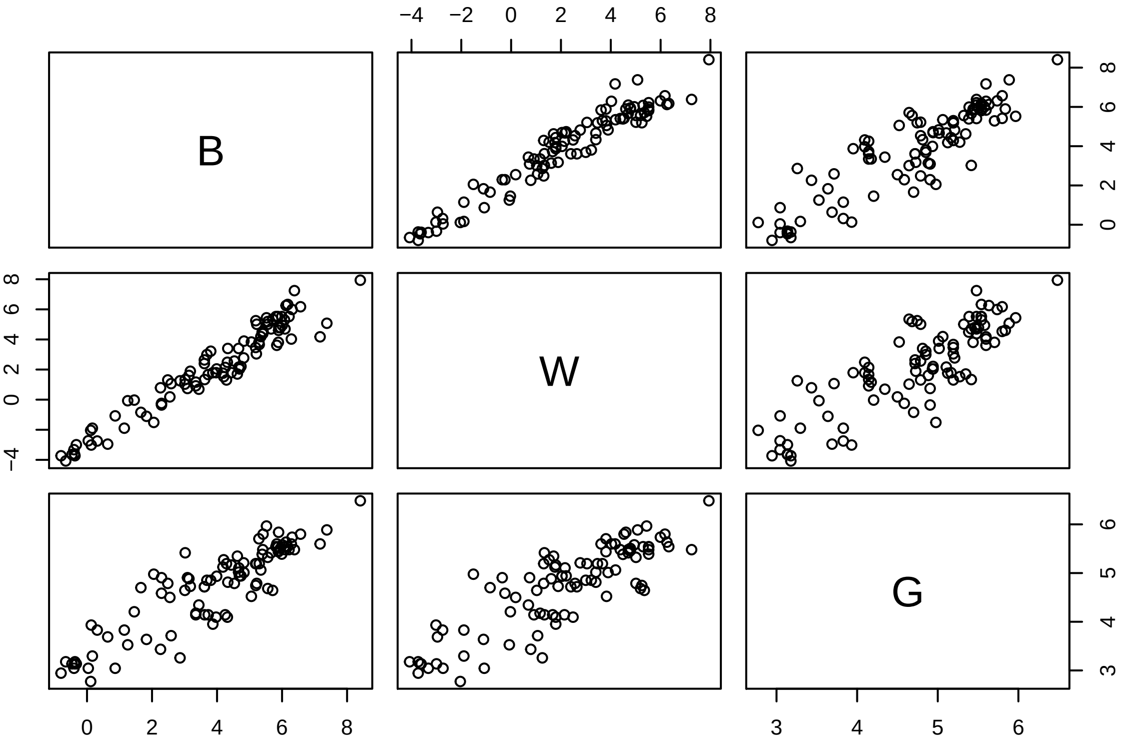
Question 2

The file “brainweight.txt” contains measurements of **breain weight** (B, in kilograms), **body weight** (W, in kilograms), and **gestation period** (G, in days) of a sample of 96 mammal species. The following commands can be used to read the data into R (although you will need point the **setwd** command to the directory on your computer that contains the data file.).

```
brain_data =read.table("BrainWeight.txt", stringsAsFactors = FALSE, header = TRUE)
colnames(brain_data)=c("B","W","G")
```

a. Use the **pairs** function to construct a scatterplot matrix of the logarithms of B, W, G.

```
pairs(log(brain_data))
```



b. Use the **cor** function to determine the correlation matrix for the three (logged) variables.

```
cor(log(brain_data))
```

```
##           B         W         G
## B 1.0000000 0.9642905 0.8912940
## W 0.9642905 1.0000000 0.8455317
## G 0.8912940 0.8455317 1.0000000
```

c. Use the **lm** function in R to fit the MLR model,

$$\ln B = \beta_0 + \beta_1 \ln W + \beta_2 \ln G + e$$

and print out the **summary** of the model fit.

```r
model = lm(B ~ ., data=log(brain_data))
summary(model)
```

```
##
## Call:
## lm(formula = B ~ ., data = log(brain_data))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00286 -0.30372 -0.05242  0.37851  1.58788
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.45728    0.45848  -0.997    0.321
## W            0.55117    0.03236  17.033   <2e-16 ***
## G            0.66782    0.10875   6.141    2e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4902 on 93 degrees of freedom
## Multiple R-squared:  0.9501, Adjusted R-squared:  0.949
## F-statistic: 885.2 on 2 and 93 DF,  p-value: < 2.2e-16
```

d. Create the design/covariate matrix, X, for the model in 1c and verify that the coefficient estimates in the summary ouput are given by the least squares formula: $(X^TX)^{-1}X^Ty$.

```r
X = cbind(rep(1, nrow(brain_data)),log(brain_data$W), log(brain_data$G))
beta = solve(t(X)%*%X)%*%t(X)%*%log(brain_data$B)
print(beta)
```

```
##            [,1]
## [1,] -0.4572826
## [2,]  0.5511654
## [3,]  0.6678215
```

e. Compute the hat-matrix and use it to calculate the fitted values and residual vector. (Include the relevant R code but don't print out these vectors or the hat matrix.) Determine the squared correlation between the fitted values and the response vector. What value in the summary output does this number correspond to?

```r
hat_mat = X %*% solve(t(X)%*%X) %*% t(X)
fit_vals = hat_mat %*% log(brain_data$B)
resids = log(brain_data$B) - fit_vals
sq_cor = cor(log(brain_data$B), model$fitted.values)^2
print(sq_cor)
```

```
## [1] 0.9500939
```

The term `sq_cor` above represents the squared correlation between the response vector and the fitted values, and we see that it is the same as the `R-squared` value in the model summary from part 1c.

f. Compute the mean squared error and its square root using the residual vector from 1e. What value in the summary output is equal to the root mean squared error?

```r
mse = sum(resids^2)/length(resids)
rt_mse = sqrt(mse)
rt_mse
```

```
## [1] 0.4824908
```

This value is close to the residual standard error, but not exact. If we reduce the denominator by the number of parameters in the model (to account for degrees of freedom) and redo the above computations we find

```
mse = sum(resids^2)/(length(resids)-length(model$coefficients))
rt_mse = sqrt(mse)
rt_mse
```

```
## [1] 0.4902111
```

which is exactly the residual squared error.

g. Compute the standard errors for the regression coefficients using the root mean squared error and the design matrix. Verify that your computed values agree with those given in the summary output.

The mean squared error from above serves as our estimator of the variance of e, $\hat{\sigma^2}$. This allows us to compute estimates of the standard errors of the coefficients as the square root of the diagonal entries of the matrix, $\hat{\sigma^2}(X^TX)^{-1}$. Computing these values gives,

```
sqrt(diag(mse*solve(t(X)%*%X)))
```

```
## [1] 0.45848472 0.03235852 0.10874659
```

which we see agrees with the values in the model summary.

h. Extract the residual vector (WG say) from the SLR model with y=ln(W) and x=ln(G). Rerun the MLR in 1c with ln(W) replaced by WG. What happens to the ANOVA decomposition if the order of the predictors is changed?

First we generate the SLR model and extract the residuals.

```
model = lm(W ~ G, data = log(brain_data))
WG = model$residuals
```

Now we recreate the model from 1c with WG as a predictor, checking the summary of the model, and comparing the ANOVA tables when the order of the predictors is changed.

```
model1 = lm(log(brain_data$B) ~ WG + log(brain_data$G))
summary(model1)
```

```
##
## Call:
## lm(formula = log(brain_data$B) ~ WG + log(brain_data$G))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00286 -0.30372 -0.05242  0.37851  1.58788
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -6.66476    0.27820  -23.96   <2e-16 ***
## WG                 0.55117    0.03236   17.03   <2e-16 ***
## log(brain_data$G)  2.23399    0.05806   38.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4902 on 93 degrees of freedom
## Multiple R-squared:  0.9501, Adjusted R-squared:  0.949
## F-statistic: 885.2 on 2 and 93 DF,  p-value: < 2.2e-16
```

In the above we see that when `ln(W)` is replaced with `WG`, the three coefficient estimates change, but the overall model fit (as determined by R^2) remains the same. Furthermore the standard error of the coefficient for `WG` is the same as the coefficient for `ln(W)`. These results are somewhat expected, as the new predictor does not really add any 'new' information to the model.

```
model2 = lm(log(brain_data$B) ~ log(brain_data$G) + WG)
anova(model1)
```

```
## Analysis of Variance Table
##
## Response: log(brain_data$B)
##                   Df  Sum Sq Mean Sq F value    Pr(>F)
## WG                 1  69.72   69.72  290.13 < 2.2e-16 ***
## log(brain_data$G)  1 355.74  355.74 1480.37 < 2.2e-16 ***
## Residuals         93  22.35    0.24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(model2)
```

```
## Analysis of Variance Table
##
## Response: log(brain_data$B)
##                   Df  Sum Sq Mean Sq F value    Pr(>F)
## log(brain_data$G)  1 355.74  355.74 1480.37 < 2.2e-16 ***
## WG                 1  69.72   69.72  290.13 < 2.2e-16 ***
## Residuals         93  22.35    0.24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Comparing the ANOVA tables for the two models we see the order of the predictors does not effect any component of the table (the same goes for the model summaries).

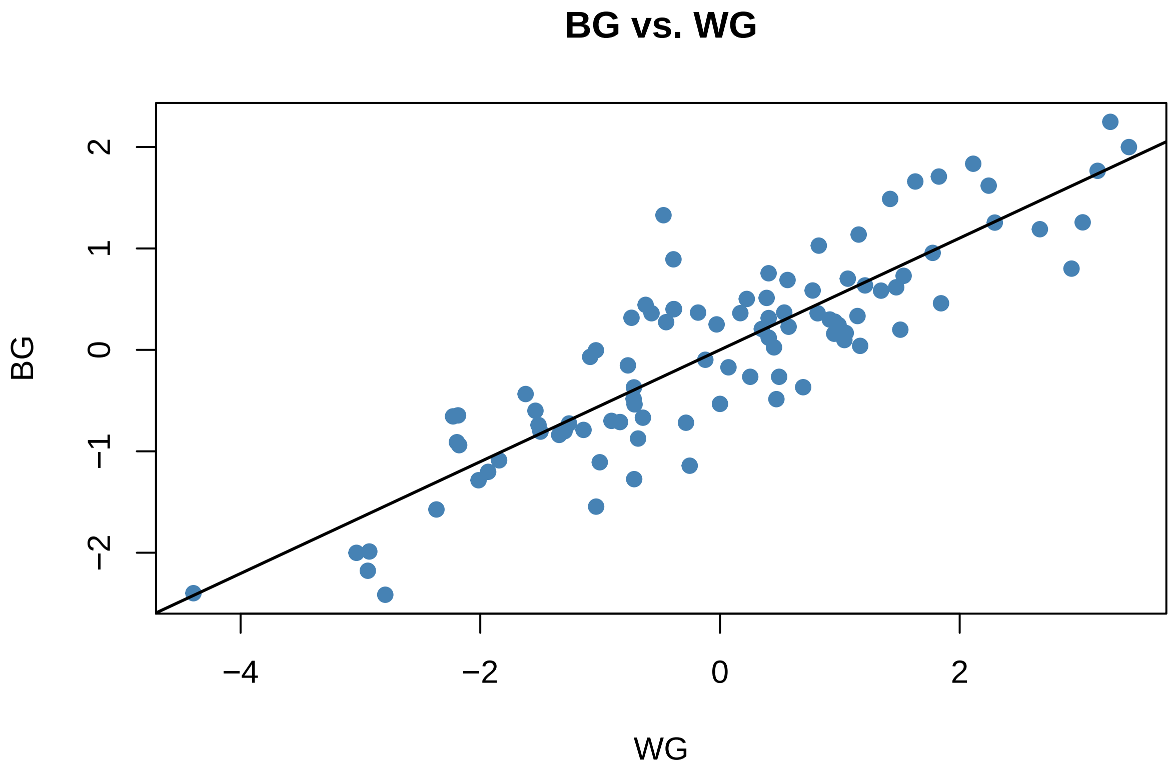
i. Extract the residuals (BG say) from SLR fit with y=ln(B) and x=ln(G). Construct a scatterplot of BG versus WG and add the regression line.

First construct the SLR model and extract the residuals.

```
model = lm(B ~ G, data = log(brain_data))
BG = model$residuals
```

Now construct the scatter plot and add regression line.

```
plot(BG ~ WG, pch = 19, col = "steelblue",
     main = "BG vs. WG")
bg_wg_model = lm(BG ~ WG)
abline(bg_wg_model, lwd=1.5)
```



Looking at the summary of this new model we see that the slope of the regression line (i.e. the coefficient of the `WG` term) is the same as the coefficient of the `W` in the model from 1c.

j. Compare the slope of the regression of BG on WG to the estimated regression coefficients from the MLR in 1c.

```
summary(bg_wg_model)
```

```
##
## Call:
## lm(formula = BG ~ WG)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00286 -0.30372 -0.05242  0.37851  1.58788
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.802e-17  4.977e-02    0.00        1
## WG          5.512e-01  3.219e-02   17.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4876 on 94 degrees of freedom
## Multiple R-squared:  0.7573, Adjusted R-squared:  0.7547
## F-statistic: 293.2 on 1 and 94 DF,  p-value: < 2.2e-16
```

Looking at the summary of this new model we see that the slope of the regression line (i.e. the coefficient of the `WG` term) is the same as the coefficient of the `W` in the model from 1c.

Question 3

Here we will consider data that come from categorical covariates. Specifically, if we assume we have pairs (y_i, X_i) of observations where y_i is an outcome and x_i is a label indicating category of observations (eg, undergraduate, masters, PhD level of eduction) but here we will just use A, B or C.

Since we can't sum up letters, in order to deal with labels, we unfold x_i into three dummy variables

$$x_i \to (x_{iA}, x_{iB}, x_{iC})$$

where x_{iA} is 1 if x_i had label A, and x_{iB} is 1 if it had label B etc; the values are 0 otherwise.

We now create X as a $n \times 3$ matrix. It might help to think of an example where $n = 6$ with 2 observations from each category. E.g.

$$X = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

We will assume that there are k observations (rather than 2) from each category.

We can then write the model

$$y_i = \beta_A x_{iA} + \beta_B x_{iB} + \beta_C x_{iC} + \epsilon_i$$

a. What prediction does this model make for an observation from each category?

For an observation from category k the model predicts $y = \beta_k$.

b. Show that $\sum_{i=1}^{k}(y_i - b)^2$ is minimized at $b = \bar{y}$; hence find the least-squares estimates of β_A, β_B and β_C.

To show that this is minimized by $\bar{y}$ we take a deriviative with respect to b and set the resulting expression equal to 0.

$$\frac{\partial}{\partial b}\sum_{i=1}^{k}(y_i - b)^2 = -2\sum_{i=1}^{k}(y_i - b) \overset{set}{=} 0$$

Carrying the sum through gives,

$$\sum_{i=1}^{k} y_i = kb$$

or

$$b = \frac{1}{k}\sum_{i=1}^{k} y_i = \bar{y}.$$

And note that the second derivative with respect to b of the above expression is just 2, indicating that the function is convex, and our optimizer of $\bar{b}$ is in fact a minimizer.

This result indicates that the estimates for the β coefficients should be the mean of the response variable across each category. That is, β_k will be the average of the y values for which y is in category k, $\beta_k = \frac{1}{k}\sum_{y_i \in k} y_i$, where n_k is the number of observations in category k and $y_i \in k$ is the set of observations from category k.

c. Give an expression for $X^T X$ in terms of special matrices (it may help to work this out for the specific case of the example above, first)

First note that we can represent the entries of $X^T X$ as dot products of the columns of X, i.e. $X^T X_{ij} =< X_{:,i}, X_{:,j} >$. Furthermore, since each column has zero entries wherever another has non-zero entries we have that the columns of X are mutually orthogonal, thus $X^T X$ will be a strictly diagonal matrix ($< X_{:,i}, X_{:,j} >= 0$ when $i \neq j$).

Now consider what the diagonal entries will be; $X^T X_{ii}$ will be the inner product of the i^{th} column with itself, which is just the sum of the squared entries. So $X^T X_{ii} = \sum_{j=1}^{k} X_{ji}^2$ and since $X_{ji} = 1$ when y_j is from category i and 0 otherwise this is really just a sum of indicator functions of whether y_j is in category i, which gives the number of observations from category i. More explicitly,

$$X^T X = \text{diag}(\#\text{ obs. in category } A, \ldots, \#\text{ obs. in category } N) = \text{diag}(n_A, \ldots, n_N),$$

where n_A represents the number of observations in category A.

d. Give an expression for $X^T\mathbf{y}$ and show that the formula $(X^T X)^{-1} X^T\mathbf{y}$ gives you the same answer as in 3b.

$X^T y$ will be a vector where entry i contains the sum of the observations of y in category i. If we say $y_i \in A$ means observation y_i is from category A we can succinctly write this vector as,

$$X^T y = (\sum_{y_i \in A} y_i, \ldots, \sum_{y_i \in N} y_i)^T.$$

Now note that since $X^T X$ is diagonal the inverse will just be a diagonal matrix with the reciprocals of the entries of $X^T X$. The resulting vector $(X^T X)^{-1} X^T y$ will have entries that represent the dot products of the columns of $(X^T X)^{-1}$ and $X^T y$, which will give

$$(X^T X)^{-1} X^T y = (\frac{1}{n_A} \sum_{y_i \in A} y_i, \ldots, \frac{1}{n_N} \sum_{y_i \in N} y_i)^T,$$

confirming the results in 3b.

e. How do things change if you have different numbers of observations in different categories? Eg k_A observations in category A, k_B in category B and k_C in category C. What if I have more than three categories?

The results found in the previous parts have no restriction on the number of categories or the number of observations in each category, so the form of the estimator in 3d will hold for arbitrary numbers of categories/observations.

f. Our original model has no intercept, suppose we fit the model

$$y_i = \beta_0 + \beta_A x_{iA} + \beta_B x_{iB} + \beta_C x_{iC} + \epsilon_i$$

If we set $\beta_0 = 10$, what are the least squares estimates of β_A, β_B and β_C in the new model?

If we set $\beta_0 = 10$ we can reinterpret this model as the same model from part 1a with a shifted set of y_i values. Define $\tilde{y}_i = y_i - 10$ to get

$$\tilde{y}_i = y_i - 10 + \beta_0 + \beta_A x_{iA} + \beta_B x_{iB} + \beta_C x_{iC} + \epsilon_i.$$

Then we can get the least square estimates for $\hat{\beta}_p = \frac{1}{n_p} \sum_{\tilde{y}_i \in p} \tilde{y}_i = \frac{1}{n_p} \sum_{y_i \in p} (y_i - 10)$.

g. For this new model, we add a column of 1's to X to get a new matrix

$$\tilde{X} = [\mathbf{1}_n, X]$$

as a working examle with $k = 2$ we have

$$\tilde{X} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Set $\alpha = (1, -1, -1, -1)$ show that $\tilde{X}\alpha = 0$ and that we can add $g\alpha$ to $(\beta_0, \beta_A, \beta_B, \beta_C)$ for any g without changing the predictions.

With a small snippet of code it is easy to see that $\tilde{X}\alpha = 0$.

```r
X_tilde = cbind(rep(1, 6), c(1,1,0,0,0,0), c(0,0,1,1,0,0), c(0,0,0,0,1,1))
alpha = c(1,-1,-1,-1)
print(X_tilde %*% alpha) ## produces the zero vector
```

```
##      [,1]
## [1,]    0
## [2,]    0
## [3,]    0
## [4,]    0
## [5,]    0
## [6,]    0
```

From this we can compute the matrix product for predictions with $g\alpha$ added to the vector of coefficients (here called β) to get,

$$X(g\alpha + \beta) = gX\alpha + X\beta = g0 + \hat{y} = \hat{y}.$$

Showing that predictions are unchanged. (Note: this is due to the column vectors of X being linearly dependent and thus producing a singular gram matrix)

h. To deal with the issue above, we usually drop one of the categories and fit the model

$$y_i = \beta_0 + \beta_B x_{iB} + \beta_C x_{iC} + \epsilon_i$$

What predictions does this model make for observations in each category?

This model will estimate $\beta_0 = \frac{1}{n_A}\sum_{y_i \in A} y_i$, the estimate for β_A from 3b/d, while estimating $\beta_B = \frac{1}{n_B}\sum_{y_i \in B} y_i - \frac{1}{n_A}\sum_{y_i \in A} y_i$ (likewise for β_C). This ensures that predictions remain the same as in the previous model, however the gram matrix $X^T X$ is no longer singular.

i. With this representation, what is the X matrix for this model using our $k = 2$ example above? Write down an expression for $X^T X$.

In this model we have

$$X = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

and (for this example) the matrix $X^T X$ becomes

$$\begin{pmatrix} 6 & 2 & 2 \\ 2 & 2 & 0 \\ 2 & 0 & 2 \end{pmatrix}.$$

In general we will have,

$$X^T X = \begin{pmatrix} n_{total} & n_B & \dots & n_N \\ n_B & n_B & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ n_N & 0 & \dots & n_N \end{pmatrix}$$