PSY 503: Lab 05 - Correlations, Regression with Dummy Coding

2025-10-08

PSY 503: Lab 05 - Correlations, Regression with Dummy Coding

Correlation and Regression

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   4.0.0     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(broom)
```

Let's use the galton's child parent height data as before. CSV file here: https://drive.google.com/file/d/1LK9jGBSpPr21S9_BzZhJi5Z68rB3JTEX/view?usp=sharing

Your tasks are:

###(1) Add two columns to the dataset: z_child_ht, and z_parent_ht which consists of standardized heights scores for the two variables

[Hint: Standardization = turning measurements into z-scores. z-scores represent measurements as standard deviations from the mean. z-score(x) = (x-mean(x))/sd(x)

```
main_dat <- read.csv('galton_child_parent_heights.csv')
head(main_dat)
```

```
##   family_id child_ht parent_ht
## 1        F1     72.2      74.5
## 2        F2     73.2      74.5
## 3        F3     73.2      74.5
## 4        F4     73.2      74.5
## 5        F5     68.2      73.5
## 6        F6     69.2      73.5
```

```r
z_score <- function(x) {(x-mean(x))/sd(x)}
main_dat$z_child_ht <- z_score(main_dat$child_ht)
main_dat$z_parent_ht <- z_score(main_dat$parent_ht)
head(main_dat)
```

```
##   family_id child_ht parent_ht z_child_ht z_parent_ht
## 1        F1     72.2      74.5 1.61521035    2.738661
## 2        F2     73.2      74.5 2.00852320    2.738661
## 3        F3     73.2      74.5 2.00852320    2.738661
## 4        F4     73.2      74.5 2.00852320    2.738661
## 5        F5     68.2      73.5 0.04195895    2.215723
## 6        F6     69.2      73.5 0.43527180    2.215723
```

###(2) Install the package "corrr" if you don't have it already. Use corrr::correlate() to compute the correlation matrix/ pair-wise correlations for the variables in the data-frame. Display the matrix and save it within a new dataframe. Explain the observed results. You can also use cor() to get the correlation matrix, but corrr::correlate() works better with tidyverse.

```r
#install.packages("corrr")
library(corrr)

correlate(main_dat)
```

```
## Non-numeric variables removed from input: ‘family_id‘
## Correlation computed with
## * Method: 'pearson'
## * Missing treated using: 'pairwise.complete.obs'
```

```
## # A tibble: 4 x 5
##   term        child_ht parent_ht z_child_ht z_parent_ht
##   <chr>          <dbl>     <dbl>      <dbl>       <dbl>
## 1 child_ht      NA         0.459      1           0.459
## 2 parent_ht      0.459    NA          0.459       1
## 3 z_child_ht     1         0.459     NA           0.459
## 4 z_parent_ht    0.459     1          0.459      NA
```

```r
cor_main_matrix <- correlate(main_dat)
```

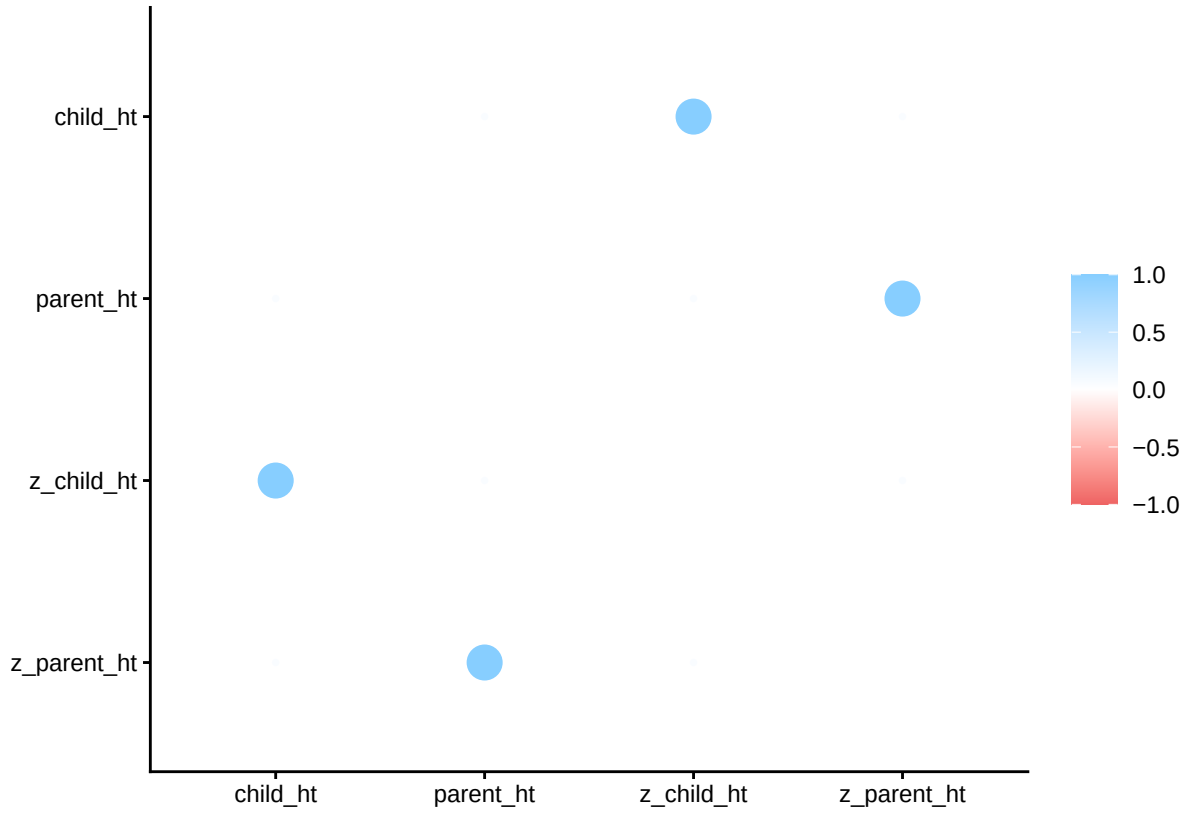
```
## Non-numeric variables removed from input: ‘family_id‘
## Correlation computed with
## * Method: 'pearson'
## * Missing treated using: 'pairwise.complete.obs'
```

A plot of the correlation matrix can be easily made by piping your correlation dataframe to rplot(), or calling rplot with the data as a parameter. Examples can be found here: https://corrr.tidymodels.org/reference/rplot.html

Plot your correlations below. Why are all the points of the same color?

```r
rplot(cor_main_matrix)
```

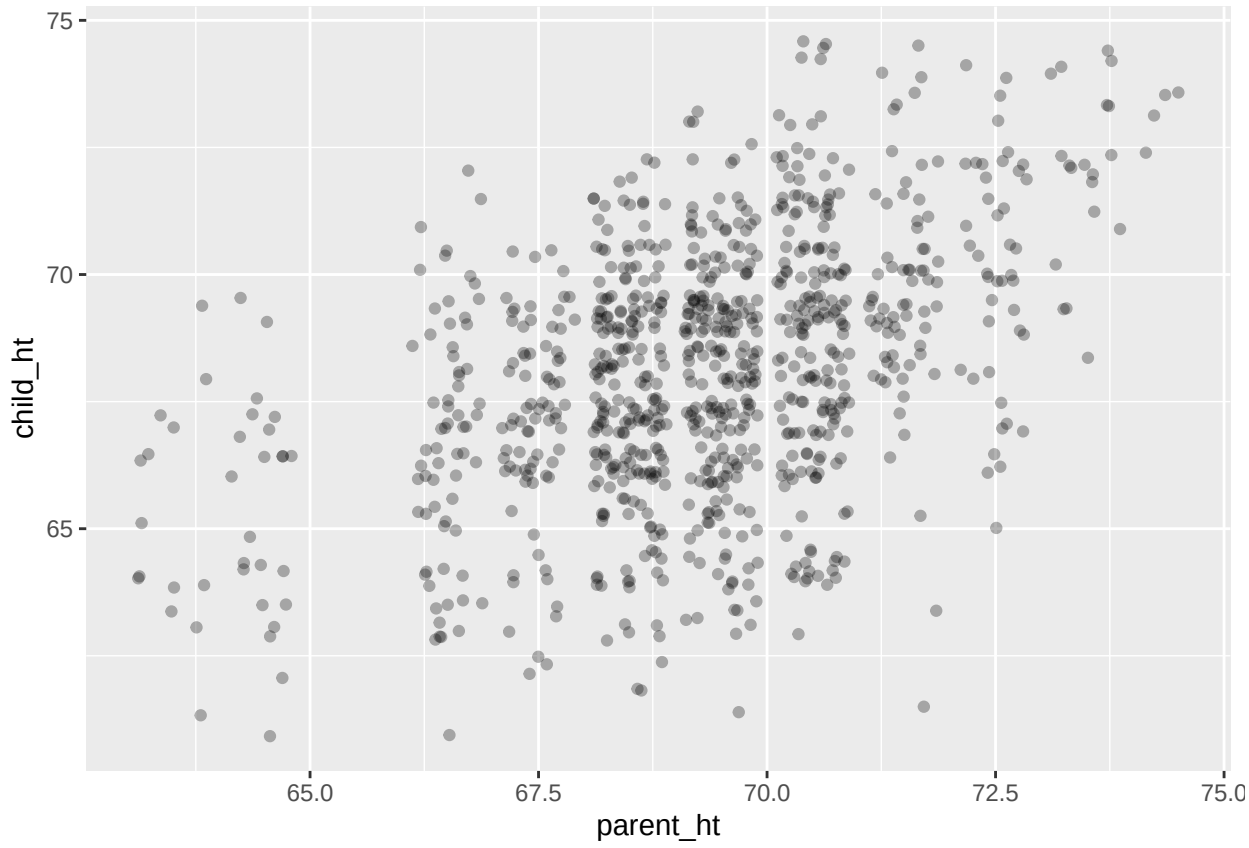
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## i The deprecated feature was likely used in the corrr package.
##   Please report the issue at <https://github.com/tidymodels/corrr/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Other than '1', basically all correlation efficients are showing 0.4587. I guess it's because standardizing variables doesn't change the correlations between the variables so that all the correlation results here are depicting how 'parent height' and 'child height' are correlated.

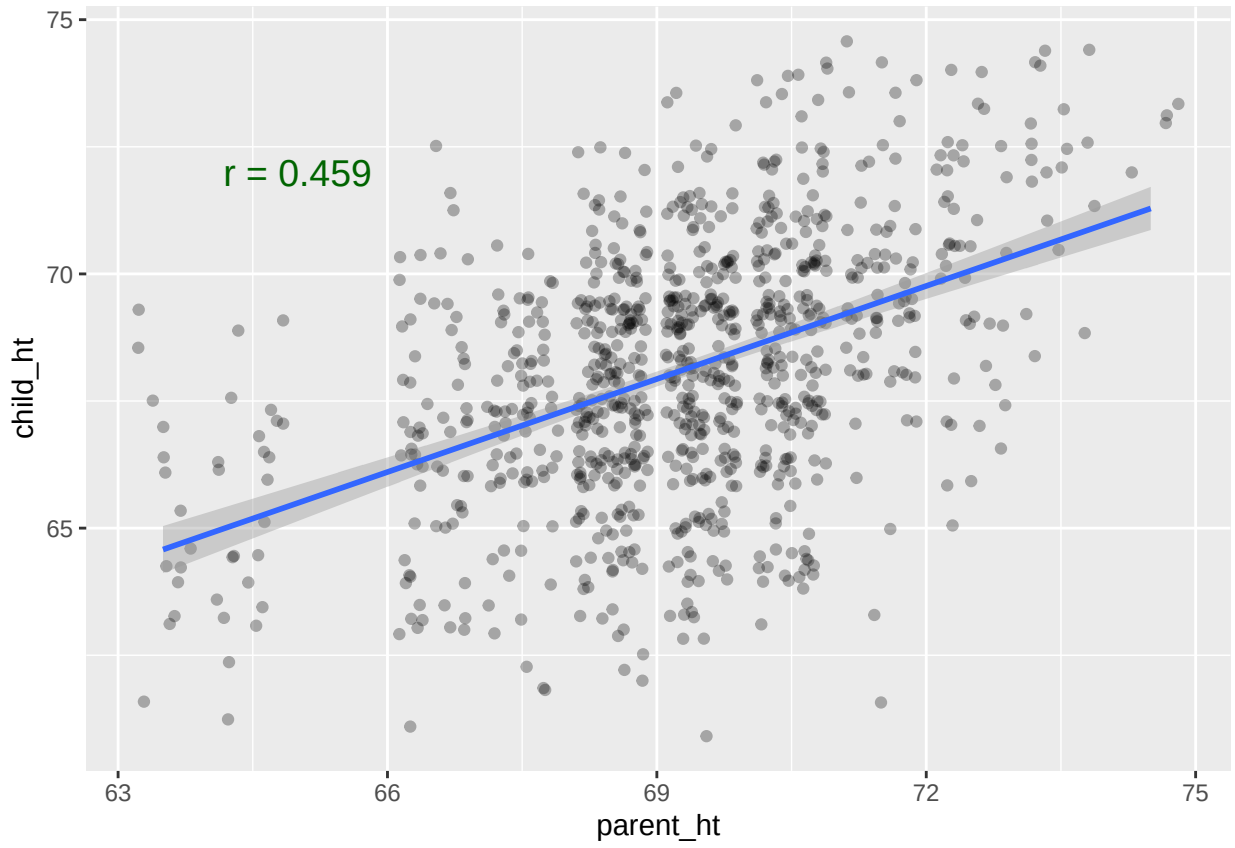
###(3) Create scatterplots of child_ht vs parent-ht, and z_child_ht vs z_parent_ht. Add a linear regression line to the plot (it is fine to use geom_smooth). Annotate the plot with the correlation value you had calculated earlier.

```
library(ggplot2)
ggplot(main_dat, aes(x = parent_ht, y = child_ht)) +
  geom_jitter(alpha = 0.3)
```



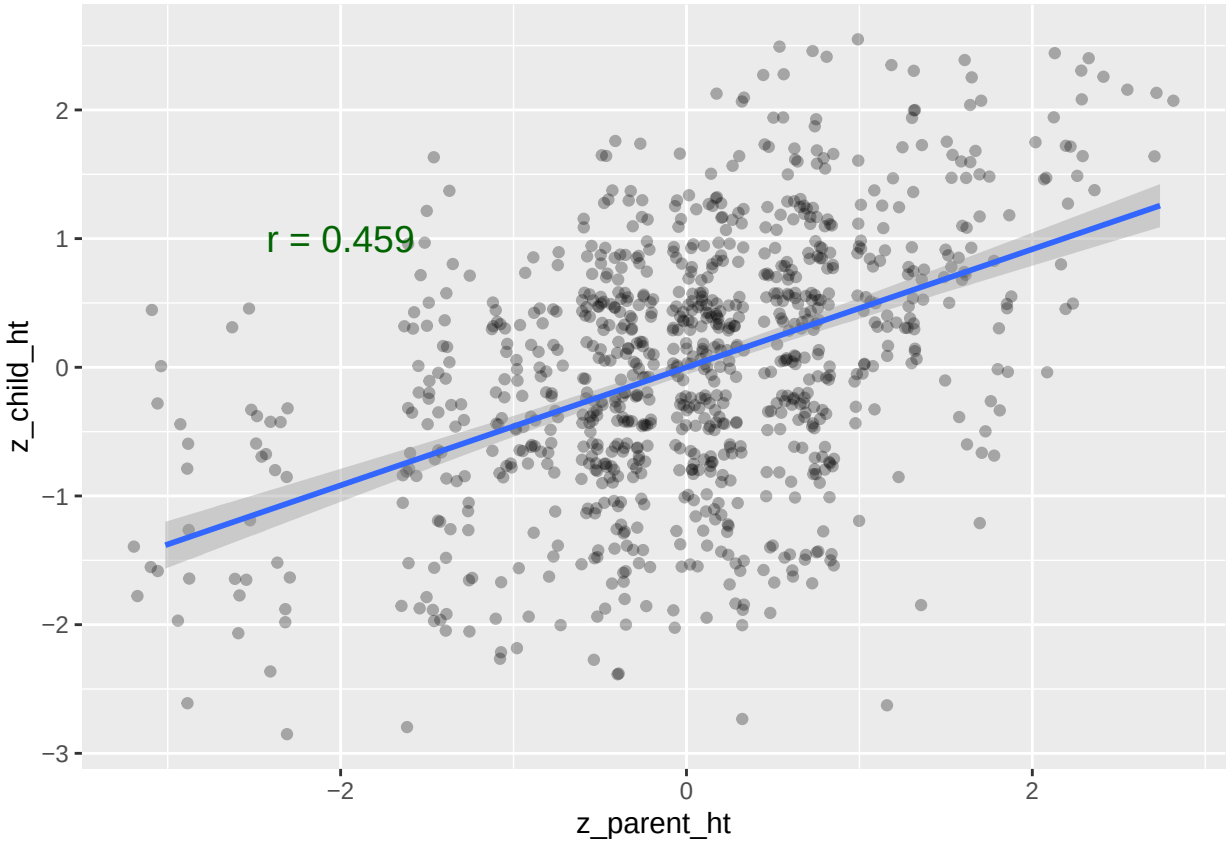

```
ggplot(main_dat, aes(x = parent_ht, y = child_ht)) +
  geom_jitter(alpha = 0.3) +
  geom_smooth(method = 'lm') +
  annotate("text", x = 65, y = 72, label = "r = 0.459", color = "darkgreen", size = 5)
```

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




```
ggplot(main_dat, aes(x = z_parent_ht, y = z_child_ht)) +
  geom_jitter(alpha = 0.3) +
  geom_smooth(method = 'lm') +
  annotate("text", x = -2, y = 1, label = "r = 0.459", color = "darkgreen", size = 5)
```

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



###(4)
####(a) Fit a linear model to the data with i) child_ht as the outcome variable, and parent_ht as the predictor/explanatory variable. ii) z_child_ht the outcome variable, and z_parent_ht as the predictor/explanatory variable.

Save these results as new r objects.

```
parent2child <- lm(child_ht ~ parent_ht, data = main_dat)
print(parent2child)
```

```
##
## Call:
## lm(formula = child_ht ~ parent_ht, data = main_dat)
##
## Coefficients:
## (Intercept)    parent_ht
##     25.8486       0.6099
```

```
summary(parent2child)
```

```
##
## Call:
## lm(formula = child_ht ~ parent_ht, data = main_dat)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -8.2577 -1.4280  0.1323  1.5720  5.7918
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 25.84856    2.69009   9.609   <2e-16 ***
## parent_ht    0.60992    0.03882  15.710   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.26 on 926 degrees of freedom
## Multiple R-squared:  0.2104, Adjusted R-squared:  0.2096
## F-statistic: 246.8 on 1 and 926 DF,  p-value: < 2.2e-16
```

```
parent_z2child_z <- lm(z_child_ht ~ z_parent_ht, data = main_dat)
print(parent_z2child_z)
```

```
##
## Call:
## lm(formula = z_child_ht ~ z_parent_ht, data = main_dat)
##
## Coefficients:
## (Intercept)  z_parent_ht
##   2.611e-15    4.587e-01
```

```
summary(parent_z2child_z)
```

```
##
## Call:
## lm(formula = z_child_ht ~ z_parent_ht, data = main_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2479 -0.5616  0.0520  0.6183  2.2780
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.611e-15  2.918e-02    0.00        1
## z_parent_ht 4.587e-01  2.920e-02   15.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8891 on 926 degrees of freedom
## Multiple R-squared:  0.2104, Adjusted R-squared:  0.2096
## F-statistic: 246.8 on 1 and 926 DF,  p-value: < 2.2e-16
```

####(b) Compare the co-efficients you observe for the two cases, and explain anything salient about what you see.

```
library(broom)

tidy(parent2child)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   25.8      2.69        9.61 6.67e-21
## 2 parent_ht      0.610    0.0388     15.7  1.76e-49
```

```
tidy(parent_z2child_z)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept) 2.61e-15    0.0292  8.95e-14 1.000e+ 0
## 2 z_parent_ht 4.59e- 1    0.0292  1.57e+ 1 1.76 e-49
```

The coefficient value of the z-score model is lower than that of the original value model. The intercept coefficient value is extremely small in the z score model.

####(c) Use glance() from the broom package to look at the several summary statistics obtained for these regression fits. Explain anything salient about what you see here.

```
glance(parent2child)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.210         0.210  2.26      247. 1.76e-49     1 -2073. 4151. 4166.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
glance(parent_z2child_z)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.210         0.210 0.889      247. 1.76e-49     1 -1207. 2419. 2434.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Both models have the same r squared, adjusted r squared, statistic, and p values, while some other values are different.

####(d) We have come across the notion of total variance in the data, as well as variance that is explained or unexplained by a model. Calculate these three values for both set of models, and write about anything you find salient.

As a hint, all of these can be found with the var command. The values you would be finding the variance of are either the (i) fitted data / predictions, (ii) observed data, or (iii) model residuals. You need to map these quantities to the different types of variance. Feel free to use augment() from broom, or referencing obtaining these variables from within the model object (lmfit_from_4a$fitted, lmfit_from_4a$resid)

```
augment(parent2child, data = main_dat)
```

```
## # A tibble: 928 x 11
##    family_id child_ht parent_ht z_child_ht z_parent_ht .fitted .resid    .hat
```

```
##    <chr>        <dbl>     <dbl>      <dbl>       <dbl>   <dbl>  <dbl>   <dbl>
##  1 F1            72.2      74.5     1.62         2.74    71.3  0.912 0.00917
##  2 F2            73.2      74.5     2.01         2.74    71.3  1.91  0.00917
##  3 F3            73.2      74.5     2.01         2.74    71.3  1.91  0.00917
##  4 F4            73.2      74.5     2.01         2.74    71.3  1.91  0.00917
##  5 F5            68.2      73.5     0.0420       2.22    70.7 -2.48  0.00637
##  6 F6            69.2      73.5     0.435        2.22    70.7 -1.48  0.00637
##  7 F7            69.2      73.5     0.435        2.22    70.7 -1.48  0.00637
##  8 F8            70.2      73.5     0.829        2.22    70.7 -0.478 0.00637
##  9 F9            71.2      73.5     1.22         2.22    70.7  0.522 0.00637
## 10 F10           71.2      73.5     1.22         2.22    70.7  0.522 0.00637
## # i 918 more rows
## # i 3 more variables: .sigma <dbl>, .cooksd <dbl>, .std.resid <dbl>
```

```
raw_var_fit <- var(parent2child$fitted) #fitted data
raw_var_obs <- var(main_dat$child_ht) #observed data
raw_var_res <- var(parent2child$resid) #residuals

augment(parent_z2child_z, data = main_dat)
```

```
## # A tibble: 928 x 11
##    family_id child_ht parent_ht z_child_ht z_parent_ht .fitted .resid    .hat
##    <chr>        <dbl>     <dbl>      <dbl>       <dbl>   <dbl>  <dbl>   <dbl>
##  1 F1            72.2      74.5     1.62         2.74    1.26  0.359 0.00917
##  2 F2            73.2      74.5     2.01         2.74    1.26  0.752 0.00917
##  3 F3            73.2      74.5     2.01         2.74    1.26  0.752 0.00917
##  4 F4            73.2      74.5     2.01         2.74    1.26  0.752 0.00917
##  5 F5            68.2      73.5     0.0420       2.22    1.02 -0.974 0.00637
##  6 F6            69.2      73.5     0.435        2.22    1.02 -0.581 0.00637
##  7 F7            69.2      73.5     0.435        2.22    1.02 -0.581 0.00637
##  8 F8            70.2      73.5     0.829        2.22    1.02 -0.188 0.00637
##  9 F9            71.2      73.5     1.22         2.22    1.02  0.205 0.00637
## 10 F10           71.2      73.5     1.22         2.22    1.02  0.205 0.00637
## # i 918 more rows
## # i 3 more variables: .sigma <dbl>, .cooksd <dbl>, .std.resid <dbl>
```

```
z_var_fit <- var(parent_z2child_z$fitted) #fitted data
z_var_obs <- var(main_dat$z_child_ht) #observed data
z_var_res <- var(parent_z2child_z$resid) #residuals
```

####(e) Calculate explained variance/ total variance for both sets of models. Connect results here with results from 4a & 4c..

```
raw_var_fit
```

```
## [1] 1.360329
```

```
raw_var_obs
```

```
## [1] 6.464333
```

```
raw_var_res
```

```
## [1] 5.104004
```

```
z_var_fit
```

```
## [1] 0.2104361
```

```
z_var_obs
```

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

```
z_var_res
```

```
## [1] 0.7895639
```

```
raw_var_fit/raw_var_obs
```

```
## [1] 0.2104361
```

```
z_var_fit/z_var_obs
```

```
## [1] 0.2104361
```

For both models, it's 0.2104361. This matches with the models' r squared values that we found in 4a and 4c.

Part 2. Regression Intro (Continuous outcome, categorical predictors)

Let's use the gapminder dataset that you have already worked with, and let's use the most recent year of data.

To motivate our analysis, fundamental question in economic development is "How does a country's geographic location relate to its economic prosperity?"

One interesting observation is that countries within the same continent often have similar levels of economic development. Do you guess differently about a country's economy knowing that it is in Africa or that it is in Europe?

Well, in addition to guessing, we can look at the data and attempt a regression analysis. gdpPercap could be a proxy for economic prosperity And we know the continents that each country belongs to.

```
library(gapminder)
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
```

```
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # i 1,694 more rows
```

```
data <- gapminder
head(data)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
## 5 Afghanistan Asia       1972    36.1 13079460      740.
## 6 Afghanistan Asia       1977    38.4 14880372      786.
```

With regression, we can ask questions like: 1. Is there a difference in GDP per capita across continents? 2. How much of the variation in countries' GDP per capita can be explained by which continent they're in? and so on.

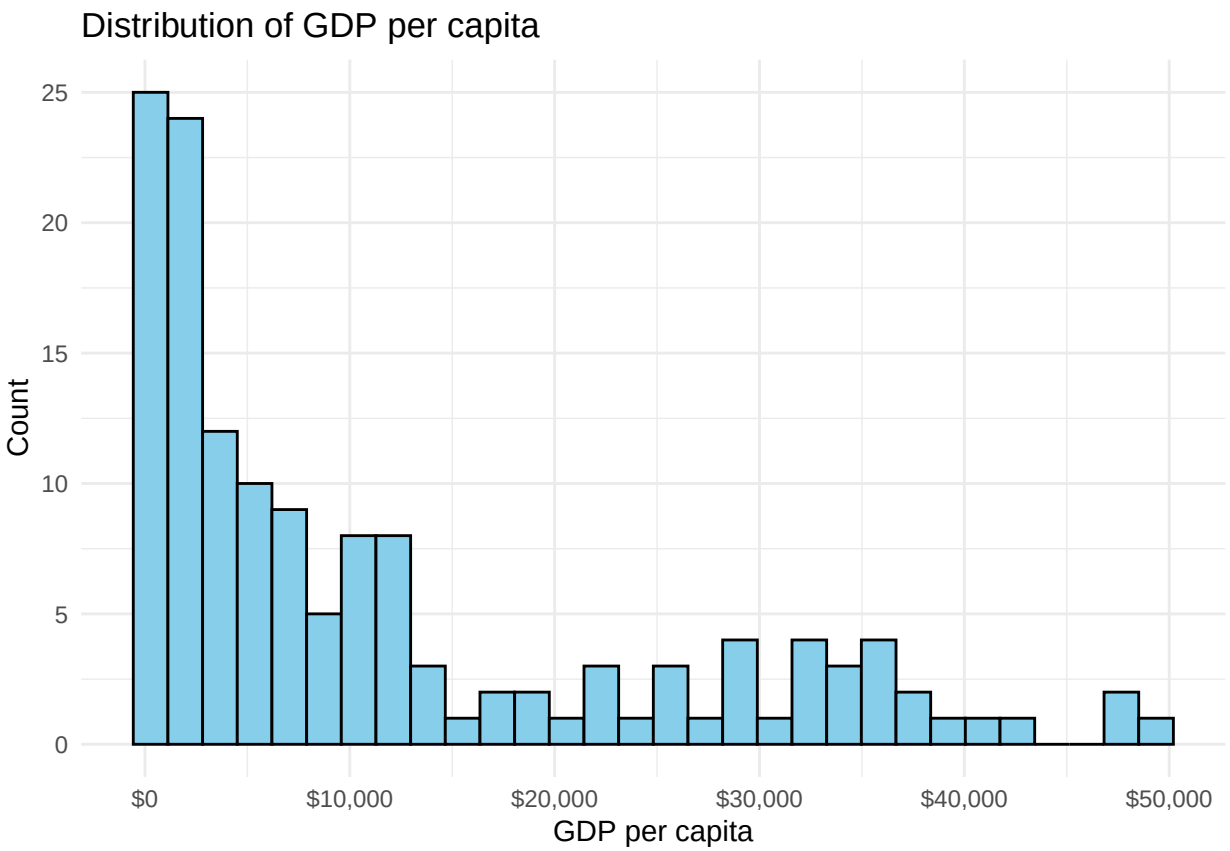
Step 1: Visualize data

```
library(gapminder)
library(tidyverse)

# Let's use the most recent year of data
gapminder_2007 <- gapminder %>%
  filter(year == 2007)

# let's look at the distribution of GDP per capita
p1<- ggplot(gapminder_2007, aes(x = gdpPercap)) +
  geom_histogram(bins = 30, fill = "skyblue", color = "black") +
  scale_x_continuous(labels = scales::dollar_format()) +
  labs(title = "Distribution of GDP per capita",
       x = "GDP per capita", y = "Count") +
  theme_minimal()

print (p1)
```



This data is right-skewed. It turns out that taking the log of per capita gdp in such scenarios helps with skewness. (It also helps economists talk more easily about rate of change of GDP and whether that is increasing or not)

Let's transform the variable and observe the results

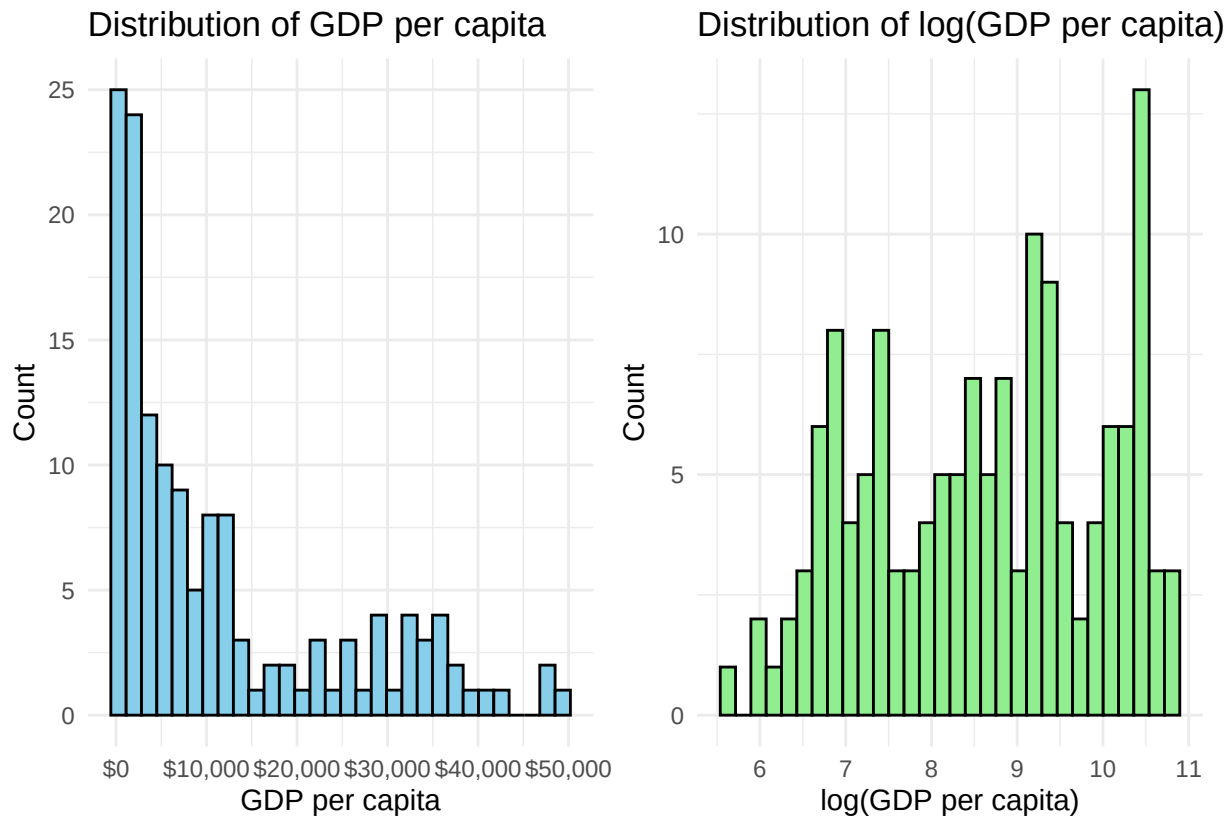
```
# Let's use the most recent year of data
gapminder_2007 <- gapminder %>%
  filter(year == 2007) %>%
  mutate(log_gdp = log(gdpPercap))

# Visualize the distribution of GDP per capita
p1 <- ggplot(gapminder_2007, aes(x = gdpPercap)) +
  geom_histogram(bins = 30, fill = "skyblue", color = "black") +
  scale_x_continuous(labels = scales::dollar_format()) +
  labs(title = "Distribution of GDP per capita",
       x = "GDP per capita", y = "Count") +
  theme_minimal()

# Visualize the distribution of log(GDP per capita)
p2 <- ggplot(gapminder_2007, aes(x = log_gdp)) +
  geom_histogram(bins = 30, fill = "lightgreen", color = "black") +
  labs(title = "Distribution of log(GDP per capita)",
       x = "log(GDP per capita)", y = "Count") +
  theme_minimal()

# Display plots side by side
```

```
library(patchwork)
p1 + p2
```




```
### ??? this doesn't work

#update.packages(ask = FALSE)
#install.packages("patchwork", dependencies = TRUE)
```

The transformed values seem more like a normal distribution and with less skew. So let's use log_gdp as our outcome variable.

###(5) Create dummy variables for the continent variable. Use the mutate() function to add new columns, one for each continent except one (which will serve as the reference category). Use Africa as the reference category.

[Hint: (1) "as.integer(continent ==")Americas")" returns the value of 1 when the continent is Americas, and 0 otherwise (2) Note again, that you'll need one column less than all the possible continent columns]

Show the first few rows of this new dataset.

```
gapminder_dummy <- gapminder_2007 %>%
  mutate(log_gdp = log(gdpPercap),
         continent_America = as.integer(continent == "Americas"),
         continent_Asia    = as.integer(continent == "Asia"),
         continent_Europe  = as.integer(continent == "Europe"),
         continent_Oceania = as.integer(continent == "Oceania"))
```

```r
head(gapminder_dummy)
```

```
## # A tibble: 6 x 11
##   country     continent  year lifeExp    pop gdpPercap log_gdp continent_America
##   <fct>       <fct>     <int>   <dbl>  <int>     <dbl>   <dbl>              <int>
## 1 Afghanistan Asia       2007    43.8 3.19e7      975.    6.88                  0
## 2 Albania     Europe     2007    76.4 3.60e6     5937.    8.69                  0
## 3 Algeria     Africa     2007    72.3 3.33e7     6223.    8.74                  0
## 4 Angola      Africa     2007    42.7 1.24e7     4797.    8.48                  0
## 5 Argentina   Americas   2007    75.3 4.03e7    12779.    9.46                  1
## 6 Australia   Oceania    2007    81.2 2.04e7    34435.   10.4                   0
## # i 3 more variables: continent_Asia <int>, continent_Europe <int>,
## #   continent_Oceania <int>
```

###(6) Fit two models Model A is the empty model that predicts log_gdp by only includes an intercept. Model B is the model that predicts log_gdp based on continent.

```r
model_A <- lm(log_gdp ~ 1, data = gapminder_dummy)
summary(model_A)
```

```
##
## Call:
## lm(formula = log_gdp ~ 1, data = gapminder_dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9898 -1.2230  0.1041  1.1828  2.1911
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.6158     0.1138   75.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.356 on 141 degrees of freedom
```

```r
model_B <- lm(log_gdp ~ continent_America + continent_Asia + continent_Europe + continent_Oceania,
              data = gapminder_dummy)
summary(model_B)
```

```
##
## Call:
## lm(formula = log_gdp ~ continent_America + continent_Asia + continent_Europe +
##     continent_Oceania, data = gapminder_dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9300 -0.7032 -0.1170  0.5136  2.0236
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)         7.4865     0.1332  56.208  < 2e-16 ***
## continent_America   1.5349     0.2338   6.566 9.85e-10 ***
## continent_Asia      1.2542     0.2138   5.867 3.17e-08 ***
## continent_Europe    2.4994     0.2202  11.351  < 2e-16 ***
## continent_Oceania   2.8039     0.6921   4.051 8.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9605 on 137 degrees of freedom
## Multiple R-squared:  0.5125, Adjusted R-squared:  0.4983
## F-statistic: 36.01 on 4 and 137 DF,  p-value: < 2.2e-16
```

```
#model_B <- lm(log_gdp ~ continent, data = gapminder_2007)
#summary(model_B)
```

###(7) Examine the results of the two models. What do you notice/ infer from it? Try using glance() for this model too, and elaborate on what you think is salient.

```
glance(model_A)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1         0             0  1.36        NA      NA    NA  -244.  492.  498.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
glance(model_B)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.513         0.498 0.960      36.0 1.52e-20     4  -193.  398.  416.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Model A(null model)'s r squared value is 0, which means it doesn't explain any variance in log_gdp.

```
#added
model_C <-  lm(log_gdp ~ continent_America + continent_Asia + continent_Europe + continent_Oceania, data
print (model_C)
```

```
##
## Call:
## lm(formula = log_gdp ~ continent_America + continent_Asia + continent_Europe +
##     continent_Oceania, data = gapminder_dummy)
##
## Coefficients:
##       (Intercept)  continent_America     continent_Asia   continent_Europe
##             7.487              1.535              1.254              2.499
## continent_Oceania
##             2.804
```

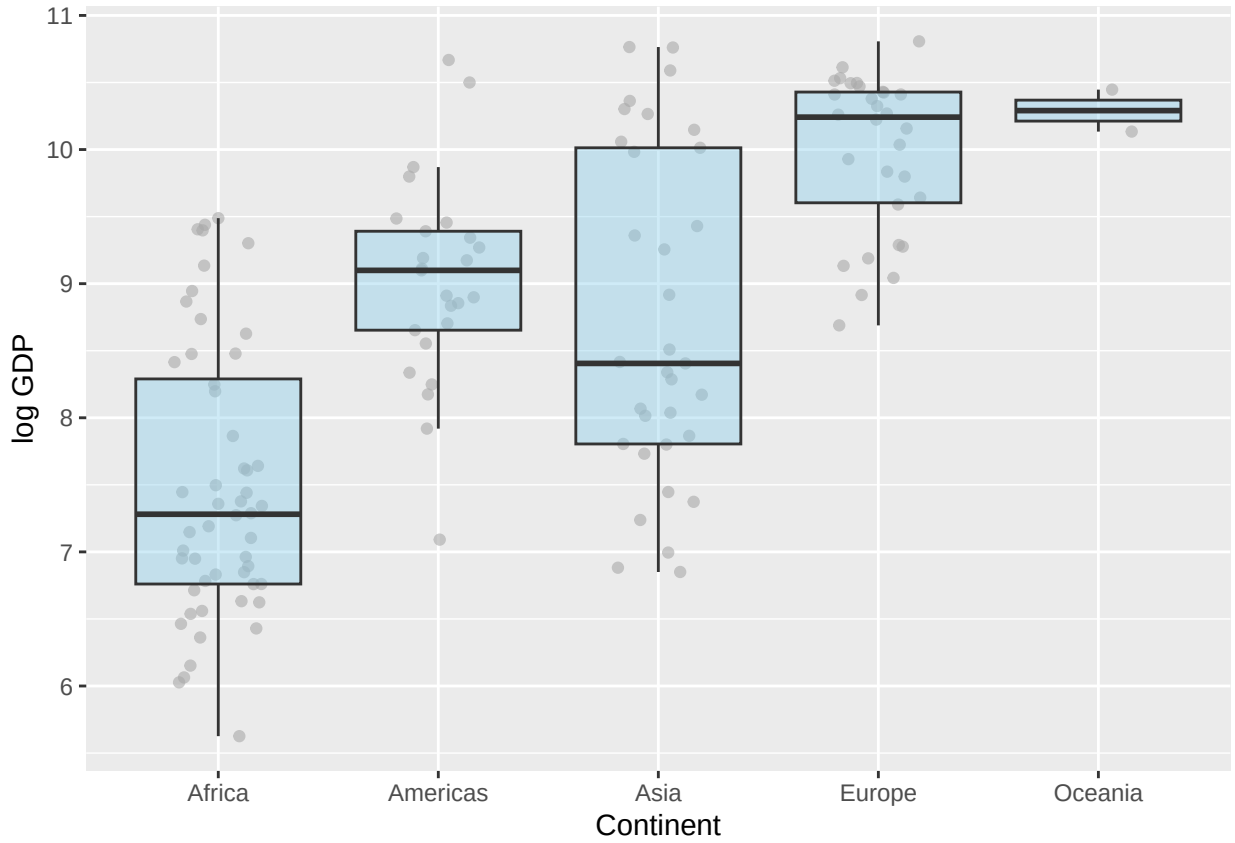
```
print (summary(model_C))
```

```
##
## Call:
## lm(formula = log_gdp ~ continent_America + continent_Asia + continent_Europe +
##     continent_Oceania, data = gapminder_dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9300 -0.7032 -0.1170  0.5136  2.0236
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         7.4865     0.1332  56.208  < 2e-16 ***
## continent_America   1.5349     0.2338   6.566 9.85e-10 ***
## continent_Asia      1.2542     0.2138   5.867 3.17e-08 ***
## continent_Europe    2.4994     0.2202  11.351  < 2e-16 ***
## continent_Oceania   2.8039     0.6921   4.051 8.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9605 on 137 degrees of freedom
## Multiple R-squared:  0.5125, Adjusted R-squared:  0.4983
## F-statistic: 36.01 on 4 and 137 DF,  p-value: < 2.2e-16
```

###(8) Create a strip plot of log_gdp for each continent, with a box plot overlaid. This will show both the individual country data points and the overall distribution for each continent. Use ggplot2 to create this visualization. Does this visual support the conclusions you drew from the R-squared value?

[Hint: Strip plots are generated by using geom_jitter(); Given that you are overlaying the boxplot over the strip-plot, use the alpha parameter to ensure that the boxplot is not opaque and obscuring the datapoints. For the boxplot, set outlier.shape = NA, so that outliers are not plotted — we already have the data visualized including outliers]

```
ggplot(gapminder_2007, aes(x = continent, y = log_gdp)) +
  geom_jitter(width = 0.2, alpha = 0.6, color = "darkgray") +
  geom_boxplot(alpha = 0.4, outlier.shape = NA, fill = "skyblue") +
  labs(x = "Continent",
       y = "log GDP")
```



The continents that showed high coefficient values (e.g., Europe, Oceania) actually show very high values in GDP!