ITEC 621 - Homework 2 - Regression Refresher and Data Pre-Processing

Kogod School of Business

O’Malley, Conie

2025-01-29

knitr::opts\_chunk$set(echo=T, warning=F, message=F)

## Submission

Download the **HW2\_YourLastName.Rmd** R Markdown file and save it with your own last name. Complete all your work in that template file, **Knit** the corresponding Word or PDF file. Your knitted document **must display your R commands**. Submit your knitted homework document. No need to submit the .Rmd file, just your knitted file.

Also, please prepare your R Markdown file with a **professional appearance**, as you would for top management or an important client.

Please, write all your interpretation narratives outside of the R code chunks, with the appropriate formatting and businesslike appearance. I write all my comments inside of the R code chunk to suppress their display until I print the solution, but you should not do this. I will read your submission as a report to a client or senior management. Anything unacceptable to that audience is unacceptable to me.

## A Few More Things About R Markdown

R Markdown can have various sections. We will use 4 of them:

**(1) The YAML Header**. YAML is a “recursive acronym that stands for”YAML Ain’t Markup Language”. What it means is that this section does not contain R Markup code yet, but only metadata about the file. It is delimited by triple dashes, and it contains things like your name, date, output type, etc.

**(2) The Global Options**. It is a code chunk at the top of the R Markdown file named global\_options, in which you can set global parameters affecting the entire R Markdown file. If you change an option (e.g., echo=T) in a specific code chunk below, this option will supersede the global option for that code chunk only.

**(3) Markdown Text Sections**. This is the area where you type all your text. This is where you can add “marked down” codes to do things like: change font types, colors and sizes; boldface, underline, etc. See the R Markdown cheat sheet on Blackboard or online.

**(4) Code Chunk** Sections. These are sections with **R code** you embed in between sections delimited between ```{r} and ```. Your R code inside these chunks must comform to the R language syntax. Code chunks can be nameless as above, or you can name them ```{r CodeChunkName}. If you decide to name your code chunks, the names must be unique within the R Markdown file. Code chunk names are useful when debugging complex scripts and also to identify scripts in R code libraries you may have created or downloaded. The R code in a chunk will only print if you set echo=T either in the global options or in the code chunk itself.

Please ensure that your text and R code are in the correct sections and use appropriate tags and formats.

## Specific Instructions

This HW has **6 multi-part questions** related to **refreshers** and data **pre-processing**. Some question are worth **15 points** and some **20 points**.

## Prep Work

* Load the {car} library
* Load the Salaries data set in {car} using data(Salaries)
* I suggest that you enter Enter ?Salaries and review the data set information. **DON’T do this in the script**, but just in the R Console. In this and all subsequent HW and exercises, you should always inspect the data documentation and browse the data, so that you become familiarized.
* This data were collected to monitor salary differences between male and female faculty members in U.S. colleges.

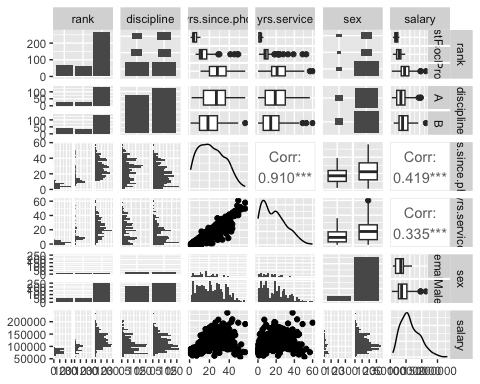
# Done for you  
library(car)  
data(Salaries)  
  
# Enter the commands below in the R Command window, but not in the R Markdown file (i.e., this is for you, not for the report)  
  
# ?Salaries  
# View(Salaries)

## 1. (15 pts.) Stats and Regression Refresher

Before you do any heavy duty predictive analysis, it is always good to do some descriptive analytics and simple OLS models to develop some familiarity with the data and the relationships among variables.

1.1 Using the ggpairs(){GGally} function, display a correlation chart with all variables in **Salaries**. Technical note: include the attribute upper=list(combo="box")) for better labeling of boxplot categories.

GGally::ggpairs(Salaries, upper=list(combo="box"))



1.2 Answer briefly: based on your review of the data, does it appear to be a salary gender gap? Why or why not? Identify on other promising predictor of salaries and briefly explain why is it promising. ### Analysis Based on the review of the data, it appears that there is a salary gender gap. The boxplot for the salary variable shows that the median salary for females is lower than the median salary for males. Additionally, the scatterplot matrix shows that there is a positive correlation between salary and rank, which is a promising predictor of salaries. This is because higher-ranked professors tend to have higher salaries, and rank is a categorical variable with clear order.

1.3 Fit an OLS regression model that predicts salaries using **ALL** variables as predictors and the results in an object named **fit.ols**. Use the summary() function to display the results.

lm(salary ~ ., data = Salaries) -> fit.ols # fit lm model  
  
summary(fit.ols) # view summary

Call:  
lm(formula = salary ~ ., data = Salaries)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-65248 -13211 -1775 10384 99592   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 65955.2 4588.6 14.374 < 2e-16 \*\*\*  
rankAssocProf 12907.6 4145.3 3.114 0.00198 \*\*   
rankProf 45066.0 4237.5 10.635 < 2e-16 \*\*\*  
disciplineB 14417.6 2342.9 6.154 1.88e-09 \*\*\*  
yrs.since.phd 535.1 241.0 2.220 0.02698 \*   
yrs.service -489.5 211.9 -2.310 0.02143 \*   
sexMale 4783.5 3858.7 1.240 0.21584   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 22540 on 390 degrees of freedom  
Multiple R-squared: 0.4547, Adjusted R-squared: 0.4463   
F-statistic: 54.2 on 6 and 390 DF, p-value: < 2.2e-16

1.4 If your predictive modeling goal is **inference** and you are testing the null hypothesis that there is no gender pay gap, what is your conclusion based on the **ols.fit** results?

### Analysis

Based on the results of the summary statistics of fit.ols and the p-value of the dummy variable sexMale is 0.215 which is greater than the level of significance (0.05), meaning that the variable is statistically insignificant to this model. This means according to this model, there is no gender pay gap.

## 2. (15 pts.) Heteroskedasticity and WLS

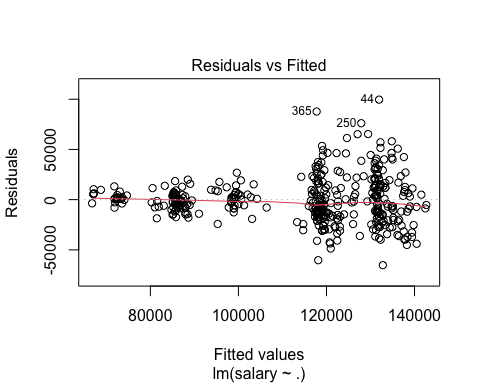
2.1 Conduct a **Breusch-Pagan** test for Heteroskedasticity for the **fit.ols** model above.

lmtest::bptest(fit.ols) # Breusch-Pagan Test

studentized Breusch-Pagan test  
  
data: fit.ols  
BP = 65.055, df = 6, p-value = 4.205e-12

2.2 Display the first residual plot for **fit.ols** by using which=1.

plot(fit.ols, which = 1) # first residual plot



2.3 Is there a problem with Heteroskedasticity? Why or why not? In your answer, please refer to **both**, the BP test and the residual plot.

### Analysis

Yes, there is a problem with heteroscedasticity in the model. The Breusch-Pagan test resulted in a p-value of 4.205e-12 < 0.05, meaning that we reject the that homoscedasticity is present in the model. The first residual plot shows a general cone shape distribution and an uneven distribution within the cone, confirming the presence of heteroscedasticity.

2.4 Fit a **WLS** model using residuals from the **fit.ols** model. Store this new model in an object named **fit.wls**. Display the summary() results fo your WLS model.

**Technical Note:** As I mentioned in class, by far the most common error when fitting WLS models is this: fit.wls=lm(fit.ols, data= ....). You cannot have an lm() fitted object inside an lm() function. It doesn’t make sense to have an object created by a function inside the same function. Instead of using **fit.ols**, type in the regression formula, like you do in any lm() model. Alternatively, store the formula in a formula object like **lm.formula** and enter enter it instead of **fit.all**.

fit\_abs\_res <- abs(resid(fit.ols)) # calculate absolute residuals   
fit\_fitted <- fitted(fit.ols) # calculate fitted values  
  
fit\_second\_ols <- lm(fit\_abs\_res ~ fit\_fitted) # fit second OLS model  
  
wts <- 1/fitted(fit\_second\_ols)^2 # calculate weights  
  
fit\_wls <- lm(salary ~ ., data = Salaries, weights = wts) # fit WLS model  
summary(fit\_wls) # view summary statistics

Call:  
lm(formula = salary ~ ., data = Salaries, weights = wts)  
  
Weighted Residuals:  
 Min 1Q Median 3Q Max   
-5.7085 -0.8737 -0.1912 0.7641 5.1454   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 71146.6 974.9 72.977 < 2e-16 \*\*\*  
rankAssocProf 12071.6 2020.5 5.974 5.21e-09 \*\*\*  
rankProf 48561.1 3030.1 16.026 < 2e-16 \*\*\*  
disciplineB 13168.9 1387.8 9.489 < 2e-16 \*\*\*  
yrs.since.phd 480.0 218.8 2.194 0.02882 \*   
yrs.service -660.9 206.6 -3.199 0.00149 \*\*   
sexMale 1838.3 1126.0 1.633 0.10336   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 1.313 on 390 degrees of freedom  
Multiple R-squared: 0.7187, Adjusted R-squared: 0.7144   
F-statistic: 166.1 on 6 and 390 DF, p-value: < 2.2e-16

2.5 Respond briefly: based on your WLS results, is there empirical evidence of gender salary inequality? Do you believe the WLS or the OLS model? Why?

### Analysis

Based on the WLS results, there is no empirical evidence of gender salary inequality because the dummy variable sexMale is statistically insignificant to the model because p-value (0.103) > 0.05 confidence level. I believe the WLS model because there are more significant variables after adding the weights, the Adjusted R-squared has increased from 0.4463 to 0.7144 (OLS and WLS, respectively), and the model p value (2.2e-16) < 0.05 confidence level.

## 3. (15 pts.) Transformations: Categorical Data

3.1 Load the **{MASS}** library and use the levels() function to take a look at the levels of the **AirBags** factor variable in the Cars93 dataset. Then fit a regression model using the **Cars93** data set to predict **Price** (i.e., average car price in thousands of dollars) as a function of **Type, MPG.city, AirBags** and **Origin**. Store the results in an object named **lm.fit**. Then, display the summary() results of this model (you can try ?Cars93 at the console to get some information on the dataset).

**Caution:** A common mistake is to re-fit this model in **3.1** below after re-leveling in **3.3**. If you use the relevel() function in **3.3** below and then come back to **3.1**, the data set will be already re-leveled, so you won’t get the same results as the solution. If you knit the full R Markdown file, your results will be OK (because knitting starts from scratch), but if you run portions of the code, you may not get the correct results. For this part of the exercise, the reference level should be the first one alphabetically, that is **Driver & Passenger**. If you don’t get this, relevel() back to **Driver & Passenger** or click on the little broom on the right upper pane to clear your global environment and re-open your **MASS** library and the **Cars93** dataset fresh.

library(MASS)  
attach(Cars93)  
levels(Cars93$AirBags) # view the levels of AirBags variable

[1] "Driver & Passenger" "Driver only" "None"

lm.fit <- lm(Price ~ Type + MPG.city + AirBags + Origin, Cars93) # fit a linear regression model  
  
summary(lm.fit) # view summary statistics

Call:  
lm(formula = Price ~ Type + MPG.city + AirBags + Origin, data = Cars93)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-10.177 -3.853 -1.176 2.865 28.119   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 38.6020 4.7806 8.075 4.62e-12 \*\*\*  
TypeLarge 3.0755 2.7739 1.109 0.270739   
TypeMidsize 5.1573 2.1830 2.362 0.020496 \*   
TypeSmall -0.2819 2.5978 -0.109 0.913856   
TypeSporty 0.3151 2.3294 0.135 0.892722   
TypeVan -0.8718 2.9036 -0.300 0.764744   
MPG.city -0.7957 0.1912 -4.162 7.68e-05 \*\*\*  
AirBagsDriver only -4.3447 1.9076 -2.278 0.025322 \*   
AirBagsNone -8.9089 2.2844 -3.900 0.000195 \*\*\*  
Originnon-USA 5.1411 1.4387 3.573 0.000590 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 6.302 on 83 degrees of freedom  
Multiple R-squared: 0.616, Adjusted R-squared: 0.5744   
F-statistic: 14.79 on 9 and 83 DF, p-value: 5.166e-14

3.2 Provide a brief interpretation of the coefficient values and significance for the **AirBagsDriver only** and **AirBagsNone predictors**. In your answer, please identify the reference level Please remember to comment on the sign of the effect.

### Analysis

* AirBagDriver only: = -4.3447, meaning on average, holding everything else constant, there will be a -4.3447 change in price in relation to Driver & Passenger, which is the reference level of the dummy variable for AirBags. The negative coefficient value means that when the dummy variable for AirBagDriver only = 1, the price decreases, meaning this characteristic is less valuable in terms of price when compared to the reference level. The variable is significant to the model because p-value (0.025) < 0.05 confidence level.
* AirBagsNone: = -8.9089, meaning on average, holding everything else constant, there will be a -4.3447 change in price in relation to Driver & Passenger (reference level). The negative coefficient value means that when the dummy variable for AirBagsNone = 1, the price decreases, meaning this characteristic is less valuable in terms of price when compared to the reference level. The variable is statistically significant to the model because p-value (0.000590) < 0.05 confidence level.

3.3 Now, suppose that you want to compare prices of cars with air bags to those without airbags. Do this, please relevel() the **AirBags** factor variable so that the reference level is changed to **“None”**. Fit the regression model again after re-leveling the AirBags predictor. Store this re-leveled lm() object as **lm.fit.rlv**. Display the summary() results of this model.

Cars93$AirBags <- relevel(Cars93$AirBags, ref = "None") # relevel AirBags variable  
  
lm.fit.rlv <- lm(Price ~ Type + MPG.city + AirBags + Origin, Cars93) # fit new linear regression model  
  
summary(lm.fit.rlv) # view summary statistics

Call:  
lm(formula = Price ~ Type + MPG.city + AirBags + Origin, data = Cars93)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-10.177 -3.853 -1.176 2.865 28.119   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 29.6931 4.7225 6.288 1.43e-08 \*\*\*  
TypeLarge 3.0755 2.7739 1.109 0.270739   
TypeMidsize 5.1573 2.1830 2.362 0.020496 \*   
TypeSmall -0.2819 2.5978 -0.109 0.913856   
TypeSporty 0.3151 2.3294 0.135 0.892722   
TypeVan -0.8718 2.9036 -0.300 0.764744   
MPG.city -0.7957 0.1912 -4.162 7.68e-05 \*\*\*  
AirBagsDriver & Passenger 8.9089 2.2844 3.900 0.000195 \*\*\*  
AirBagsDriver only 4.5643 1.6720 2.730 0.007735 \*\*   
Originnon-USA 5.1411 1.4387 3.573 0.000590 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 6.302 on 83 degrees of freedom  
Multiple R-squared: 0.616, Adjusted R-squared: 0.5744   
F-statistic: 14.79 on 9 and 83 DF, p-value: 5.166e-14

3.4 Inspect the coefficients in the two models (before and after re-leveling) and answer briefly: What is the difference in interpretation for the effect of **AirBagsDrive only** between **lm.fit** and **lm.fit.rlv**? Did anything else change? Please explain briefly.

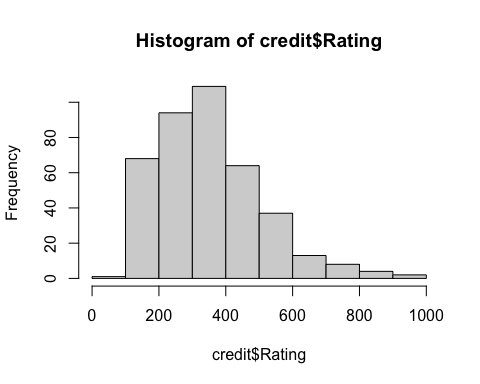
### Analysis

In the new model lm.fit.rlv, AirBagsDrive only had a change from -4.3447 to 4.5643 - meaning on average, everything else held constant, there is a price change of 4.5643 in relation to the reference level of None. This means that the characteristic associated with AirBagsDrive only increases the price in comparison to None (presumably that driver only air bags is more valuable than no airbags). The p values of the AirBags coefficients have changed since they are now being related to a new reference level - AirBagsNone. The rest of the model (p values, coefficients, Adjusted R-squared) all remain constant since the levels of the AirBag variable were only refactored and no data was added or removed from the overall model.

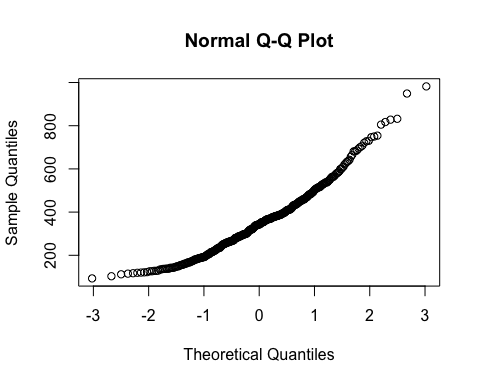
## 4. (20 pts.) Transformations: Log-Log Model

4.1 Using the read.table() function, read the **Credit.csv** data set into a data frame named **credit**. Ensure that you use header=T and sep=",". We want to use this data to predict credit **Rating**. First, display a histogram and a QQ-Plot for the **Rating** variable. It should be pretty obvious from the histogram that this variable is (skewed) not normal, although the QQ-Plot is borderline.

credit <- read.table("/Users/coniecakes/Library/CloudStorage/OneDrive-Personal/001. Documents - Main/023. Programming Tools/R Studio/Predictive Analytics/Homework/Credit.csv", header = T, sep = ",") # read data into data frame  
  
hist(credit$Rating) # plot histogram



qqnorm(credit$Rating) # plot normal qq plot

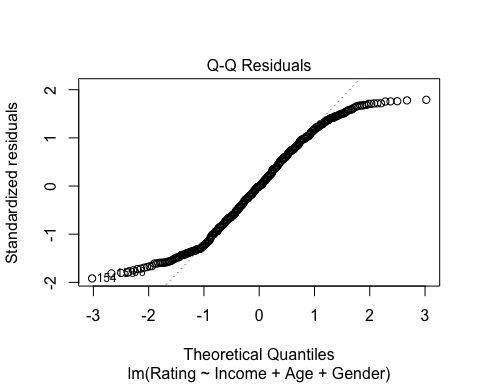


4.2 Even if the response variable were not normal, if the residual of the regression model is fairly normal, then it is OK to use the response variable without transformation. Let’s explore that. Fit a model called **fit.linear** to predict **Rating**, using **Income, Age** and **Gender** as predictors. Display a summary() of the results. Then plot() the resulting **fit.linear model**, but display only the residual plot, using the **which=2** parameter.

fit.linear <- lm(Rating ~ Income + Age + Gender, credit) # fit a linear model  
summary (fit.linear) # view summary statistics

Call:  
lm(formula = Rating ~ Income + Age + Gender, data = credit)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-180.226 -77.204 -0.342 78.129 169.052   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 212.0946 17.1029 12.401 <2e-16 \*\*\*  
Income 3.5034 0.1367 25.628 <2e-16 \*\*\*  
Age -0.3304 0.2793 -1.183 0.238   
GenderFemale 5.4432 9.4804 0.574 0.566   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 94.74 on 396 degrees of freedom  
Multiple R-squared: 0.6279, Adjusted R-squared: 0.6251   
F-statistic: 222.7 on 3 and 396 DF, p-value: < 2.2e-16

plot(fit.linear, which = 2) # plot residuals



4.3 The residuals look normally distributed in the center of the QQ-Plot and wagging some at the tails. Let’s fit a couple of log models to see if we can improve upon the linear model. Please fit both, a **log-linear model** (loging only the response variable **Rating**) and a **log-log** (loging only the response variable **Rating** and and the predictor **Income**). Store the results of the first model in an object named **fit.log.linear** and the second one in an object named **fit.log.log**. Display the summary() for both models.

fit.log.linear <- lm(log(Rating) ~ Income + Age + Gender, credit) # fit a log-linear model  
fit.log.log <- lm(log(Rating) ~ log(Income) + Age + Gender, credit) # fit a log-log model  
  
summary(fit.log.linear) # view summary statistics

Call:  
lm(formula = log(Rating) ~ Income + Age + Gender, data = credit)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.99344 -0.21076 0.04697 0.25875 0.52991   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 5.4381923 0.0599182 90.760 <2e-16 \*\*\*  
Income 0.0088430 0.0004789 18.465 <2e-16 \*\*\*  
Age -0.0013459 0.0009784 -1.376 0.17   
GenderFemale 0.0229726 0.0332137 0.692 0.49   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3319 on 396 degrees of freedom  
Multiple R-squared: 0.4654, Adjusted R-squared: 0.4614   
F-statistic: 114.9 on 3 and 396 DF, p-value: < 2.2e-16

summary(fit.log.log) # view summary statistics

Call:  
lm(formula = log(Rating) ~ log(Income) + Age + Gender, data = credit)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.9002 -0.2105 0.0400 0.2712 0.6775   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 4.2661905 0.0993755 42.930 <2e-16 \*\*\*  
log(Income) 0.4389052 0.0248821 17.639 <2e-16 \*\*\*  
Age -0.0011171 0.0009974 -1.120 0.263   
GenderFemale 0.0137189 0.0339043 0.405 0.686   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3388 on 396 degrees of freedom  
Multiple R-squared: 0.4429, Adjusted R-squared: 0.4387   
F-statistic: 104.9 on 3 and 396 DF, p-value: < 2.2e-16

4.4 Please provide a quick interpretation of the Income or log(Income) coefficient for each of the **three models** fitted above.

### Analysis

The income for fit.linear is the greatest (3.5034), meaning that Income has the greatest effect on Rating in this model. In fit.log.linear, the Income is drastically reduced (0.00884), meaning that Income has a very minor effect on log(Rating) in this model. In fit.log.log the log(Income) has increased to .4389, meaning there is a positive relationship between log(Income) and log(Rating).

4.5 Using the **Adjusted R-Square** as a guide, which of the three models is the best (please note that you **cannot** compare the 3 models with ANOVA because they are not nested)

### Analysis

The fit.linear model is the best because it has the highest Adjusted R-squared value of 0.6251. fit.log.linear and fit.log.log have Adjusted R-squared values of 0.4614 and 0.4387, respectively, meaning they explain less variation in the response variable.

## 5. (15 pts.) Transformations: Standardization

5.1 Using the **Cars93{MASS}** data set, fit a model to predict a car’s **price** as a function of the car’s **type, city miles per gallon, air bags** and **origin**. Store the results in an object named **fit.unstd** and display the summary() results for this linear model object.

colnames(Cars93)

[1] "Manufacturer" "Model" "Type"   
 [4] "Min.Price" "Price" "Max.Price"   
 [7] "MPG.city" "MPG.highway" "AirBags"   
[10] "DriveTrain" "Cylinders" "EngineSize"   
[13] "Horsepower" "RPM" "Rev.per.mile"   
[16] "Man.trans.avail" "Fuel.tank.capacity" "Passengers"   
[19] "Length" "Wheelbase" "Width"   
[22] "Turn.circle" "Rear.seat.room" "Luggage.room"   
[25] "Weight" "Origin" "Make"

fit.unstd <- lm(Price ~ Type + MPG.city + AirBags + Origin, Cars93) # fit a linear model  
summary(fit.unstd) # view summary

Call:  
lm(formula = Price ~ Type + MPG.city + AirBags + Origin, data = Cars93)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-10.177 -3.853 -1.176 2.865 28.119   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 29.6931 4.7225 6.288 1.43e-08 \*\*\*  
TypeLarge 3.0755 2.7739 1.109 0.270739   
TypeMidsize 5.1573 2.1830 2.362 0.020496 \*   
TypeSmall -0.2819 2.5978 -0.109 0.913856   
TypeSporty 0.3151 2.3294 0.135 0.892722   
TypeVan -0.8718 2.9036 -0.300 0.764744   
MPG.city -0.7957 0.1912 -4.162 7.68e-05 \*\*\*  
AirBagsDriver & Passenger 8.9089 2.2844 3.900 0.000195 \*\*\*  
AirBagsDriver only 4.5643 1.6720 2.730 0.007735 \*\*   
Originnon-USA 5.1411 1.4387 3.573 0.000590 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 6.302 on 83 degrees of freedom  
Multiple R-squared: 0.616, Adjusted R-squared: 0.5744   
F-statistic: 14.79 on 9 and 83 DF, p-value: 5.166e-14

5.2 Then, using the **lm.beta(){lm.beta}** function, extract and the standardized regression coefficients for this model and display the results. Store the results in an object named **lm.std** and display its summary().

lm.beta::lm.beta(fit.unstd) -> lm.std  
summary(lm.std)

Call:  
lm(formula = Price ~ Type + MPG.city + AirBags + Origin, data = Cars93)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-10.177 -3.853 -1.176 2.865 28.119   
  
Coefficients:  
 Estimate Standardized Std. Error t value Pr(>|t|)   
(Intercept) 29.69310 NA 4.72251 6.288 1.43e-08 \*\*\*  
TypeLarge 3.07554 0.10338 2.77387 1.109 0.270739   
TypeMidsize 5.15727 0.22813 2.18304 2.362 0.020496 \*   
TypeSmall -0.28187 -0.01227 2.59777 -0.109 0.913856   
TypeSporty 0.31511 0.01173 2.32941 0.135 0.892722   
TypeVan -0.87178 -0.02683 2.90359 -0.300 0.764744   
MPG.city -0.79575 -0.46296 0.19121 -4.162 7.68e-05 \*\*\*  
AirBagsDriver & Passenger 8.90892 0.34998 2.28440 3.900 0.000195 \*\*\*  
AirBagsDriver only 4.56426 0.23687 1.67198 2.730 0.007735 \*\*   
Originnon-USA 5.14108 0.26742 1.43868 3.573 0.000590 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 6.302 on 83 degrees of freedom  
Multiple R-squared: 0.616, Adjusted R-squared: 0.5744   
F-statistic: 14.79 on 9 and 83 DF, p-value: 5.166e-14

5.3 Answer briefly: what is the difference between the unstandardized and standardized regression results? Why would you use standardized variables or coefficients?

### Analysis

* Unstandardized: coefficients reflect the change in the response variable for a one unit change in predictor variables, holding all other variables constant
* Standardized: variables are transformed to have a = 0 and = 1. Coefficients now represent the change in the response variable in units of standard deviation for a single standard deviation change in the predictor variables, allowing for easier strength of relationship comparison.

Using standardized variables makes it easier to interpret the strength of relationships across variables and makes interpreting variables with different scales easier.

5.4 Answer briefly: is it OK to standardize binary or categorical variables like “Type” or “AirBags”? How would you get around this issue?

### Analysis

It’s generally unacceptable to standardize binary or categorical variables because they are not continuous variables, rather discrete categories. Leaving categorical or binary variables as is for analysis or using dummy variables are two ways to get around the issue.

## 6. (20 pts.) Transformations: Lagged Variables and Serial Correlation

Somtimes data sets contain more complex data structures within them. This is the case with the **economics** data set contained in the **{ggplot2}** library, which we will use for this exercise. Unfortunately, there is a small glitch in this dataset (it has a data frame inside one of the columns), which causes the **slide() function to give an error**. Fortunately, there is a simple fix for this by just re-creating the data frame. I have done this for you already in the script.

# Done for you  
library(ggplot2)  
data(economics)  
economics = as.data.frame(economics) # To fix the dataset glitch

Now, from the **R Console** (NOT in the script), enter ?economics to view the explanation of the variables in the data set. Familiarize yourself with the variables and their units, so that you can interpret results correctly. You will be developing a predictive model for **unemployment**.

6.1 First, use options(scipen=4) to limit the display of scientific notation. Then fit a linear model to predict unemployment (**unemploy**) as a function of date (**date**), personal consumption expenditures (**pce**), duration of unemployment (**uempmed**), personal savings (**psavert**), and total population (**pop**). Name this model **fit.no.lag**. Display the summary() result for the resulting linear model.

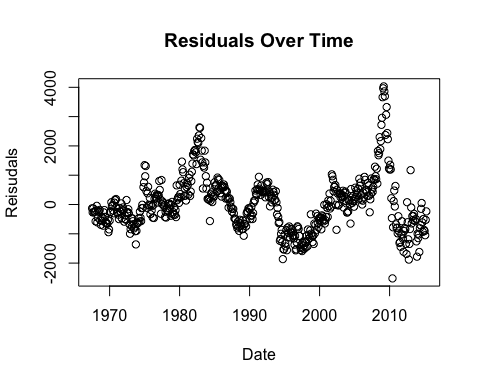
options(scipen = 4) # change scientific notation  
  
fit.no.lag <- lm(unemploy ~ date + pce + uempmed + psavert + pop, economics) # fit a linear regression model  
summary(fit.no.lag) # view summary statistics

Call:  
lm(formula = unemploy ~ date + pce + uempmed + psavert + pop,   
 data = economics)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-2522.2 -586.2 -76.0 439.9 4031.1   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 26896.43520 6200.89392 4.338 0.0000171 \*\*\*  
date 1.64376 0.16036 10.250 < 2e-16 \*\*\*  
pce -0.89754 0.10103 -8.884 < 2e-16 \*\*\*  
uempmed 581.60772 17.46624 33.299 < 2e-16 \*\*\*  
psavert 123.78794 32.45718 3.814 0.000152 \*\*\*  
pop -0.13104 0.03004 -4.362 0.0000153 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 926.3 on 568 degrees of freedom  
Multiple R-squared: 0.8782, Adjusted R-squared: 0.8771   
F-statistic: 818.7 on 5 and 568 DF, p-value: < 2.2e-16

6.2 It should be obvious from the results above that this appears to be a good model. But unemployment in one period may affect unemployment in subsequent periods, so we need to inspect for serial correlation. Display a scatter plot with economics$date (month of the observation) in the horizontal axis and the **residuals** of **fit.no.lag** in the vertical axis.

Then, briefly comment if you suspect serial correlation and why (1 or 2 lines), based on what you see on this plot.

residuals\_no\_lag <- residuals(fit.no.lag) # calculate residuals  
  
plot(economics$date, residuals\_no\_lag,  
 main = "Residuals Over Time",  
 xlab = "Date",  
 ylab = "Reisudals")



### Analysis

There is a clear pattern in this data, showing some sort of abnormal distribution of residuals. This signals there is a high likelihood of serial correlation in this model.

6.3 Now load the **{lmtest}** library and run a Durbin-Wastson test to confirm or not that the model suffers from serial correlation.

lmtest::dwtest(fit.no.lag) # conduct Durbin-Watson test

Durbin-Watson test  
  
data: fit.no.lag  
DW = 0.18485, p-value < 2.2e-16  
alternative hypothesis: true autocorrelation is greater than 0

Then, briefly comment if the DW test confirms or not the presence of serial correlation, whether it is positive or negative and why or why not.

### Analysis

The DW Statistic = 0.18485, meaning there is a strong, positive serial correlation and the p-value (2.2e-16) < 0.05 confidence level means we reject that there is no serial correlation.

6.4 Regardless of your answer above, go ahead and correct for serial correlation. My intuition tells me that unemployment in the previous month is a strong predictor of the unemployment this month. Also, I suspect that the unemployment on the same month a year ago may also influence unemployment this month.

So, let’s go ahead and load the **{DataCombine}** library and use the slide() function to create 2 lagged variables called **unemploy.L1** (lagged 1 month) and **unemploy.L12** (lagged 12 months).

Also, display all columns of the first **15 rows (only)** of the **date** and all three **unemploy** variables and observe how the lag columns were created. Tip, use economics[1:15,c("date", "unemploy", "unemploy.L1", "unemploy.L12")]

economics <- DataCombine::slide(economics, Var = "unemploy", # 1 month lag  
 NewVar = "unemploy.L1",   
 slideBy = -1)  
  
economics <- DataCombine::slide(economics, Var = "unemploy", # 12 month lag  
 NewVar = "unemploy.L12",   
 slideBy = -12)  
  
economics[1:15, c("date", "unemploy", "unemploy.L1", "unemploy.L12")] # sanity check

date unemploy unemploy.L1 unemploy.L12  
1 1967-07-01 2944 NA NA  
2 1967-08-01 2945 2944 NA  
3 1967-09-01 2958 2945 NA  
4 1967-10-01 3143 2958 NA  
5 1967-11-01 3066 3143 NA  
6 1967-12-01 3018 3066 NA  
7 1968-01-01 2878 3018 NA  
8 1968-02-01 3001 2878 NA  
9 1968-03-01 2877 3001 NA  
10 1968-04-01 2709 2877 NA  
11 1968-05-01 2740 2709 NA  
12 1968-06-01 2938 2740 NA  
13 1968-07-01 2883 2938 2944  
14 1968-08-01 2768 2883 2945  
15 1968-09-01 2686 2768 2958

6.5 Fit the same linear model above, but add the predictors **unemploy.L1** and **unemploy.L12**. Store the results of this model in an object named **fit.lag** Display the linear model summary() results.

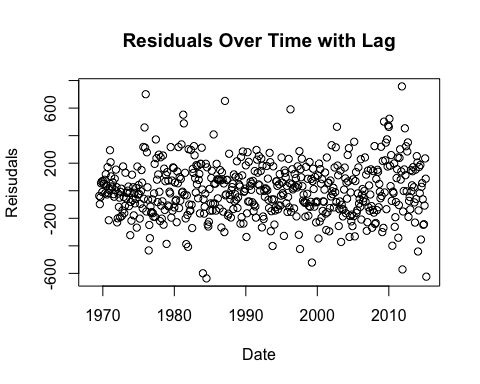
Then test this model for serial correlation with a **Durbin-Watson** test.

fit.lag <- lm(unemploy ~ date + pce + uempmed + psavert +   
 pop + unemploy.L1 + unemploy.L12, economics) # fit a linear regression model  
summary(fit.lag) # view summary statistics

Call:  
lm(formula = unemploy ~ date + pce + uempmed + psavert + pop +   
 unemploy.L1 + unemploy.L12, data = economics)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-636.48 -126.15 -7.57 127.65 757.56   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) -2172.974745 1578.301787 -1.377 0.169136   
date -0.052880 0.047746 -1.108 0.268544   
pce 0.016157 0.023327 0.693 0.488837   
uempmed -31.500533 8.560124 -3.680 0.000256 \*\*\*  
psavert 29.676306 6.969609 4.258 2.42e-05 \*\*\*  
pop 0.009526 0.007643 1.246 0.213176   
unemploy.L1 1.065303 0.009687 109.972 < 2e-16 \*\*\*  
unemploy.L12 -0.055135 0.009076 -6.075 2.31e-09 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 196 on 554 degrees of freedom  
 (12 observations deleted due to missingness)  
Multiple R-squared: 0.9943, Adjusted R-squared: 0.9942   
F-statistic: 1.373e+04 on 7 and 554 DF, p-value: < 2.2e-16

6.6 Was serial correlation corrected with the lagged model? Why or why not?

residuals\_lag <- residuals(fit.lag) # calculate residuals with lag  
  
lagged\_residuals <- c(rep(NA, 12), residuals\_lag[1:(length(residuals\_lag) - 12)])  
  
plot(economics$date[13:length(economics$date)], lagged\_residuals,  
 main = "Residuals Over Time with Lag",  
 xlab = "Date",  
 ylab = "Reisudals")



### Analysis

By looking at the Residuals Over Time with Lag plot, we can see that the distribution is appropriate to assume a random distribution with no underlying, unaccounted for relationships. The correlation has been corrected with the lagged model (particularly the 12 month). Reviewing the summary statistics, the Adjusted R-squared value has increased from 0.8771 to 0.9942, the model p-value (2.2e-16) < 0.05 confidence level, and both lagged variables are statistically significant with p-value < 0.05 confidence level.

6.7 Run ?economics in the console and take note of the description and units of all variables in this dataset. Then briefly discuss the difference in significant predictors (only) between the **fit.no.lag** and **fit.lag** models. Then provide a well-articulated interpretation of the coefficients of the 2 lagged variables in **fit.lag**.

### Analysis

* fit.no.lag: the significant predictors are: date, pce (personal consumption expenditures in billions USD), unempmed (median duration of unemployment in weeks), psavert (personal savings rate), and pop (total population in thousands).
* fit.lag: the significant predictors are: unempmed, psavert, unemploy.L1 (1 month unemployment lag), unemploy.L12 (12 month unemployment lag).
* unemploy.L1: = 1.065, meaning that on average, everything else held constant, for every one unit increase in the previous month’s unemployment, there is a 1.065 unit increase in the current month’s unemployment. This suggests there is a relationship in unemployment from month to month. This variable is statistically significant because p-value (2e-16) < 0.05 confidence level.
* unemploy.L2: = -0.055, meaning that on average, everything else held constant, for every one unit increase in the employment of the month 12 month’s prior, there is a 0.055 unit decrease in the current month’s unemployment. This means there is a weak relationship between unemployment from 12 months prior to the current month. This relationship is still statistically significant to the model with p-value (2.31e-09) < 0.05 confidence level.