Transformations and Cross-Validation

Jazib Jamal

June 8, 2017

## 1. More Transformations: Polynomial Models

1.1 Using the Salaries{car} data set, fit a **linear model** to predict **salary** as a function of **rank** and **yrs.since.phd**. Store the linear model in an object named **"fit.linear"**. Display the summary() results.

library(car)  
head(Salaries, 4) #Display first 4 rows for each column

## rank discipline yrs.since.phd yrs.service sex salary  
## 1 Prof B 19 18 Male 139750  
## 2 Prof B 20 16 Male 173200  
## 3 AsstProf B 4 3 Male 79750  
## 4 Prof B 45 39 Male 115000

fit.linear <- lm(salary~ rank + yrs.since.phd, data = Salaries)  
summary(fit.linear)

##   
## Call:  
## lm(formula = salary ~ rank + yrs.since.phd, data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -67148 -16683 -1323 11835 105552   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 81186.23 2964.17 27.389 < 2e-16 \*\*\*  
## rankAssocProf 13932.18 4345.76 3.206 0.00146 \*\*   
## rankProf 47860.42 4412.63 10.846 < 2e-16 \*\*\*  
## yrs.since.phd -80.37 129.47 -0.621 0.53510   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23650 on 393 degrees of freedom  
## Multiple R-squared: 0.3948, Adjusted R-squared: 0.3902   
## F-statistic: 85.47 on 3 and 393 DF, p-value: < 2.2e-16

1.2 What are the best predictors of faculty salaries? Why?

Looking at this model, the best predictors for faculty salaries is Rank of the faculty, which could be assistant professor, associate professor or full-professor

1.3 Who makes higher salaries, Assistant Professors, Associate Professors or Professors? How much more, on average?

Prof makes highest and assistant-Professor makes lowest. As compared to assistant professor keeping all other metrics constant, Associate professor makes $13,932 more on average and Professor makes $47,860 more.

1.4 Does the number of years since obtaining a PhD makes a difference in the salary? Why or why not?

Years.since.PhD does not make any difference according to this model. Because it is statistically insignificant as given by the p-value

1.5 Then fit a **polynomial** model of **power 4**. Leave rank alone (since it is Factor variable). Apply the polynomial transformation only on the yrs.since.phd variable using the poly() function. Store the resulting model in an object named **"fit.poly"**. Display the summary() results.

fit.poly <- lm(salary~ rank + poly(yrs.since.phd, 4), data = Salaries)  
summary(fit.poly)

##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, 4), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -54287 -16094 -1994 11395 103566   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 80921 5974 13.545 < 2e-16 \*\*\*  
## rankAssocProf 15277 6409 2.384 0.0176 \*   
## rankProf 45255 7420 6.099 2.57e-09 \*\*\*  
## poly(yrs.since.phd, 4)1 -2133 44221 -0.048 0.9615   
## poly(yrs.since.phd, 4)2 -44716 37340 -1.198 0.2318   
## poly(yrs.since.phd, 4)3 -55759 28253 -1.974 0.0491 \*   
## poly(yrs.since.phd, 4)4 23288 24699 0.943 0.3463   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23470 on 390 degrees of freedom  
## Multiple R-squared: 0.4086, Adjusted R-squared: 0.3995   
## F-statistic: 44.91 on 6 and 390 DF, p-value: < 2.2e-16

1.6 Conduct an **ANOVA** test to evaluate if the polynomial model has more predictive power than the linear model.

anova(fit.linear, fit.poly)

## Analysis of Variance Table  
##   
## Model 1: salary ~ rank + yrs.since.phd  
## Model 2: salary ~ rank + poly(yrs.since.phd, 4)  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 393 2.1985e+11   
## 2 390 2.1485e+11 3 4999068143 3.0247 0.02953 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

1.7 Does the polynomial model have more predictive power than the linear model? Why or why not?

The polynomial model has slightly more predictive model than linear model. It explains more variables (hence variance) and the r-square is higher. It explains the behavior of metric 'years.since.phd" and produces a fitting model

1.8 Based on these polynomial regression results, how would you interpret the effct of yrs.since.phd? (As we discussed in class, polynomials of power>2 are very hard to interpret, but give it a try)? In your answer, please look at the coefficients of all the different polynomial terms and provide a general interpretation.

The effect of yrs.since.PhD is not significant and the model we get, cannot be used to predict. By raising it to power of 3, we are only able to make the co-efficients statistically significant and make a better fitting model.

1.9 There is a well-known phenomenon in academics called **"salary compression"** in which newly minted PhD's command higher salaries in the market than older professors. Take a look at the coefficient values and significance levels of both, the rank and all the polynomial terms and discuss whether you see evidence of salary compresion or not. Please briefly explain your rationale.

The evidence does not exist for salary compression. The values are not statistically significant to allow us to make that conclusion. The rank tells us the increase in salary with rank, while the model does not tell us anything to study the variable 'years.since.PhD'.

1.10 **Bonus Question** (Optional, 3 extra points if you get the loop to work correctly)

Your fit.poly model only tests the 4th polynomial. But what if you wanted to test the first 6 polynomials, one by one? Write a **loop** to fit all **6 polynomials**, from poly(yrs.since.phd,1) to poly(yrs.since.phd,6). Include the **rank variable** in your models like you did in 1.5 above. In each pass of the loop, store your model in an object named \***fit.poly**. In each loop also, after fitting the model, display its summary() results.

for (i in 1:6) {  
 print(summary(lm(salary~ rank + poly(yrs.since.phd, i), data = Salaries)))  
}

##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, i), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -67148 -16683 -1323 11835 105552   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 79393 3649 21.758 < 2e-16 \*\*\*  
## rankAssocProf 13932 4346 3.206 0.00146 \*\*   
## rankProf 47860 4413 10.846 < 2e-16 \*\*\*  
## poly(yrs.since.phd, i) -20611 33201 -0.621 0.53510   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23650 on 393 degrees of freedom  
## Multiple R-squared: 0.3948, Adjusted R-squared: 0.3902   
## F-statistic: 85.47 on 3 and 393 DF, p-value: < 2.2e-16  
##   
##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, i), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -59004 -16303 -2330 12147 105111   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 87177 5132 16.987 < 2e-16 \*\*\*  
## rankAssocProf 7520 5257 1.430 0.1534   
## rankProf 37786 6428 5.879 8.86e-09 \*\*\*  
## poly(yrs.since.phd, i)1 30961 40857 0.758 0.4490   
## poly(yrs.since.phd, i)2 -73968 34453 -2.147 0.0324 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23540 on 392 degrees of freedom  
## Multiple R-squared: 0.4019, Adjusted R-squared: 0.3958   
## F-statistic: 65.85 on 4 and 392 DF, p-value: < 2.2e-16  
##   
##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, i), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -53812 -16485 -2320 11497 102869   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 81473 5945 13.705 < 2e-16 \*\*\*  
## rankAssocProf 13900 6239 2.228 0.0265 \*   
## rankProf 44764 7401 6.049 3.42e-09 \*\*\*  
## poly(yrs.since.phd, i)1 -1440 44208 -0.033 0.9740   
## poly(yrs.since.phd, i)2 -46646 37279 -1.251 0.2116   
## poly(yrs.since.phd, i)3 -52907 28087 -1.884 0.0603 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23470 on 391 degrees of freedom  
## Multiple R-squared: 0.4073, Adjusted R-squared: 0.3997   
## F-statistic: 53.73 on 5 and 391 DF, p-value: < 2.2e-16  
##   
##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, i), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -54287 -16094 -1994 11395 103566   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 80921 5974 13.545 < 2e-16 \*\*\*  
## rankAssocProf 15277 6409 2.384 0.0176 \*   
## rankProf 45255 7420 6.099 2.57e-09 \*\*\*  
## poly(yrs.since.phd, i)1 -2133 44221 -0.048 0.9615   
## poly(yrs.since.phd, i)2 -44716 37340 -1.198 0.2318   
## poly(yrs.since.phd, i)3 -55759 28253 -1.974 0.0491 \*   
## poly(yrs.since.phd, i)4 23288 24699 0.943 0.3463   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23470 on 390 degrees of freedom  
## Multiple R-squared: 0.4086, Adjusted R-squared: 0.3995   
## F-statistic: 44.91 on 6 and 390 DF, p-value: < 2.2e-16  
##   
##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, i), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -51224 -15786 -1914 11415 102310   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 79149 6098 12.979 < 2e-16 \*\*\*  
## rankAssocProf 16423 6452 2.545 0.0113 \*   
## rankProf 47625 7600 6.266 9.81e-10 \*\*\*  
## poly(yrs.since.phd, i)1 -14886 45087 -0.330 0.7414   
## poly(yrs.since.phd, i)2 -35444 37872 -0.936 0.3499   
## poly(yrs.since.phd, i)3 -59280 28328 -2.093 0.0370 \*   
## poly(yrs.since.phd, i)4 22689 24672 0.920 0.3583   
## poly(yrs.since.phd, i)5 34003 24177 1.406 0.1604   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23440 on 389 degrees of freedom  
## Multiple R-squared: 0.4116, Adjusted R-squared: 0.401   
## F-statistic: 38.87 on 7 and 389 DF, p-value: < 2.2e-16  
##   
##   
## Call:  
## lm(formula = salary ~ rank + poly(yrs.since.phd, i), data = Salaries)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -48277 -16312 -2113 11684 101473   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 80700 6526 12.367 < 2e-16 \*\*\*  
## rankAssocProf 15036 6780 2.218 0.0271 \*   
## rankProf 45643 8158 5.595 4.18e-08 \*\*\*  
## poly(yrs.since.phd, i)1 -4956 47480 -0.104 0.9169   
## poly(yrs.since.phd, i)2 -43201 39620 -1.090 0.2762   
## poly(yrs.since.phd, i)3 -55536 28892 -1.922 0.0553 .   
## poly(yrs.since.phd, i)4 22293 24696 0.903 0.3673   
## poly(yrs.since.phd, i)5 32599 24284 1.342 0.1803   
## poly(yrs.since.phd, i)6 16897 25164 0.671 0.5023   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23460 on 388 degrees of freedom  
## Multiple R-squared: 0.4123, Adjusted R-squared: 0.4002   
## F-statistic: 34.02 on 8 and 388 DF, p-value: < 2.2e-16

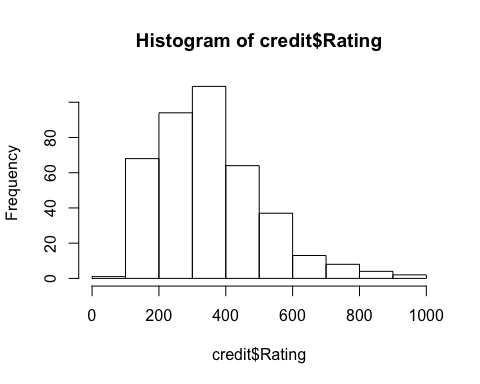
## 2. Log Models

2.1 Using the read.table() function, read the **Credit.csv** data set into a data frame named **credit**. Ensure that you use header=TRUE. We want to use this data to predict credit **Rating**. Then display a histogram and a qq-plot for the Rating variable. It should be pretty obvious from the histogram that this variable is not normal, although the qq-plot is borderline.

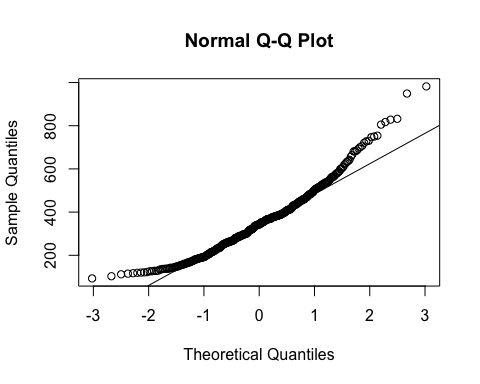
library(readr)  
credit <- read.table("~/Documents/Predictive Analytics/data sets/621 Pred Anal/Credit.csv", header = TRUE, sep = ",")  
head(credit, 4)

## X Income Limit Rating Cards Age Education Gender Student Married  
## 1 1 14.891 3606 283 2 34 11 Male No Yes  
## 2 2 106.025 6645 483 3 82 15 Female Yes Yes  
## 3 3 104.593 7075 514 4 71 11 Male No No  
## 4 4 148.924 9504 681 3 36 11 Female No No  
## Ethnicity Balance  
## 1 Caucasian 333  
## 2 Asian 903  
## 3 Asian 580  
## 4 Asian 964

hist(credit$Rating)



qqnorm(credit$Rating)  
qqline(credit$Rating)

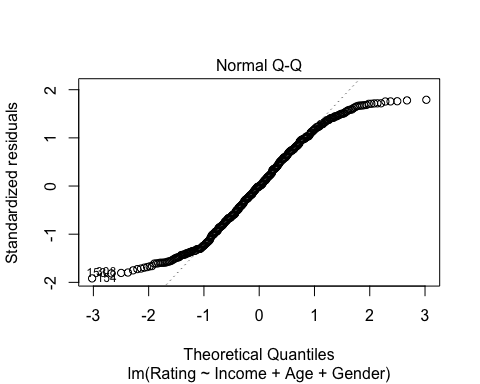


2.2 Even if the response variable is not normal, if the residual of the regression model is fairly normal, then it is OK to use the response variable without transformation. Let's explore that. Fit a model called **fit.linear** to predict **Rating**, using **Income, Age** and **Gender** as predictors.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

fit.linear <- lm(Rating~ Income + Age + Gender, data = credit)  
summary(fit.linear)

##   
## 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)



2.3 After inspecting the residual plot, do you think that this model is OK? Or do you think that you need to **log-transform** the Rating variable?

We do not need to log-transform the 'Rating variable' because it is normally distributed

2.4 Regardless of your answer to 2.2, it would not be a bad idea to test a few log tranformations to see if we get better predictive accuracy. Please fit both, a **log-linear model** (loging only the response variable **Rating**) and a **log-log** (loging only the response variable **Rating** and **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, data = credit)  
summary(fit.log.linear)

##   
## 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

fit.log.log <- lm(log(Rating)~ log(Income) + Age + Gender, data = credit)  
summary(fit.log.log)

##   
## 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

2.5 Interpretation of the Income or log(Income) coefficient for each of the three fitted models.

The income variable is significant for all three models. All models tell us that an increase in income (keeping all else constant), increases the Rating.

2.6 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)

The first model is the best because it did not require any transformation since it was normal. Its adjusted-r-square value is the highest.

## 3. Standardized Coefficients

3.1 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.

library(MASS)  
data("Cars93")  
  
fit.unstd <- lm(Price~ Type + MPG.city + Origin + AirBags, data = Cars93)  
summary(fit.unstd)

##   
## Call:  
## lm(formula = Price ~ Type + MPG.city + Origin + AirBags, 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 \*\*\*  
## Originnon-USA 5.1411 1.4387 3.573 0.000590 \*\*\*  
## AirBagsDriver only -4.3447 1.9076 -2.278 0.025322 \*   
## AirBagsNone -8.9089 2.2844 -3.900 0.000195 \*\*\*  
## ---  
## 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 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().

require(lm.beta)  
lm.std <- lm.beta(fit.unstd)  
summary(lm.std)

##   
## Call:  
## lm(formula = Price ~ Type + MPG.city + Origin + AirBags, 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) 38.60202 0.00000 4.78065 8.075 4.62e-12 \*\*\*  
## 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 \*\*\*  
## Originnon-USA 5.14108 0.26742 1.43868 3.573 0.000590 \*\*\*  
## AirBagsDriver only -4.34466 -0.22547 1.90757 -2.278 0.025322 \*   
## AirBagsNone -8.90892 -0.44658 2.28440 -3.900 0.000195 \*\*\*  
## ---  
## 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

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

There is no difference in the results even after standardizing. We will use standardized variables when we want to compare the metrics in the regression output.

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

It is not okay to standardize binary variables. The standardized variables would give us extreme values on either end.

## 4. Lagged Variables

4.1 Do the following in the R Console (don't write script commands for this) **[For you only]**: load and inspect the **economics{ggplot2}** data set. Use ?economics to view the explanation of the variables.

4.2 We will be lagging variables shortly using the **slide(){DataCombine}** function. However, sometimes data sets contain other more complex data structures within it. This is one of these cases. **The slide() function will give you an error if you lag variables in this data set. This can be corrected by converting the more complex data frame "economics" into a simpler data frame. Just add this command in your script before you do anything else with the economics data set:** economics = as.data.frame(economics)\*\*

library(ggplot2)  
require(DataCombine)  
economics <- as.data.frame(economics)

4.3 Now fit a linear model that predicts personal consumption expenditures (**pce**) as a function of date, personal savings, unemployment and total population. Name this model **"fit.no.lag"**. Display the summary result for the resulting linear model. Think but don't respond yet: is this a good model? does it have a good fit?

fit.no.lag <- lm(pce~ date + psavert + unemploy + pop, data = economics)  
summary(fit.no.lag)

##   
## Call:  
## lm(formula = pce ~ date + psavert + unemploy + pop, data = economics)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -833.2 -279.7 -38.2 313.1 1039.8   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -5.158e+04 1.431e+03 -36.058 < 2e-16 \*\*\*  
## date -1.046e+00 5.672e-02 -18.442 < 2e-16 \*\*\*  
## psavert 1.060e+02 1.422e+01 7.457 3.33e-13 \*\*\*  
## unemploy -2.208e-02 1.074e-02 -2.057 0.0402 \*   
## pop 2.485e-01 7.232e-03 34.366 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 393.2 on 569 degrees of freedom  
## Multiple R-squared: 0.988, Adjusted R-squared: 0.9879   
## F-statistic: 1.173e+04 on 4 and 569 DF, p-value: < 2.2e-16

4.4 We already have a time-ordered variable called "date" and the data is already sorted by date, so there is no need to sort the data this time. We just need to inspect the model for serial correlation using a **Durbin-Watson** test (of course, there are other tests you could use). Run a **DW test** and determine if there is **serial correlation** in the model and provide a brief interpretation of the test results.

durbinWatsonTest(fit.no.lag)

## lag Autocorrelation D-W Statistic p-value  
## 1 0.9665988 0.05553898 0  
## Alternative hypothesis: rho != 0

4.5 Regardless of your answer above, let's go ahead and correct for serial correlation. My own intuition tells me that personal consumption in one period will be influenced by the same personal consumption the month before. But it may also be influenced by personal consumption a year before on the same month (i.e., 12 months back). So, go ahead and use the slide(){DataCombine} function to create 2 lagged variables called **"pce.L1"** (lagged 1 month) and **"pce.L12"** (lagged 12 months). The data is already sorted by date, so you don't need to sort it. Otherwise you would have to.

require(DataCombine)  
economics = slide(economics, Var = "pce", NewVar = "pce.L1", slideBy = -1)  
economics = slide(economics, Var = "pce", NewVar = "pce.L12", slideBy = -12)  
head(economics)

## date pce pop psavert uempmed unemploy pce.L1 pce.L12  
## 1 1967-07-01 507.4 198712 12.5 4.5 2944 NA NA  
## 2 1967-08-01 510.5 198911 12.5 4.7 2945 507.4 NA  
## 3 1967-09-01 516.3 199113 11.7 4.6 2958 510.5 NA  
## 4 1967-10-01 512.9 199311 12.5 4.9 3143 516.3 NA  
## 5 1967-11-01 518.1 199498 12.5 4.7 3066 512.9 NA  
## 6 1967-12-01 525.8 199657 12.1 4.8 3018 518.1 NA

4.6 Fit the same linear model above, but add the predictors **pce.L1** and **pce.L12**. Store the resulst of this model in an object named **fit.lag** Display the linear model summary() results. Then also test this model for serial correlation with a **Durbin-Watson** test.

fit.lag <- lm(pce~ date + psavert + unemploy + pop + pce.L1 + pce.L12, data = economics)  
summary(fit.lag)

##   
## Call:  
## lm(formula = pce ~ date + psavert + unemploy + pop + pce.L1 +   
## pce.L12, data = economics)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -149.710 -8.818 0.895 10.133 160.646   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.798e+02 1.647e+02 -1.092 0.2753   
## date -7.250e-03 4.593e-03 -1.579 0.1150   
## psavert -3.770e+00 9.527e-01 -3.957 8.56e-05 \*\*\*  
## unemploy 1.731e-03 7.087e-04 2.442 0.0149 \*   
## pop 1.095e-03 8.027e-04 1.364 0.1731   
## pce.L1 1.026e+00 1.123e-02 91.312 < 2e-16 \*\*\*  
## pce.L12 -2.858e-02 1.111e-02 -2.572 0.0104 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 24.7 on 555 degrees of freedom  
## (12 observations deleted due to missingness)  
## Multiple R-squared: 1, Adjusted R-squared: 1   
## F-statistic: 1.943e+06 on 6 and 555 DF, p-value: < 2.2e-16

dwt(fit.lag)

## lag Autocorrelation D-W Statistic p-value  
## 1 -0.08263869 2.164049 0.086  
## Alternative hypothesis: rho != 0

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

As given by the dw test there was a coorelation correction since the value changed to 2.16

4.8 How did the results change from **fit.no.lag** to **fit.lag**.

Based on the DW test, there is no serial correlation, which means that generally lagged model is OK and the serial correlation was corrected.