# Part 1

## Is there a significant relationship between the amount of time spent reading and the time spent watching television?

1. Covariance as a statistical tool is used in order to find the relationship between variables, whether positively or negatively correlated. When the covariance is positive, say with the variables TimeTV and Happiness, this means that there is a positive relationship between the two – as the time spent watching TV goes up, so does the reported happiness of the individual. When the covariance is negative, like with TimeReading and Happiness, the means that as the time spent reading goes up, the reported happiness goes down. While it does not indicate the strength of the relationship, it can help in predicting the direction of estimated values.

Covariance of data in Student Survey

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(ggm)

## Warning: package 'ggm' was built under R version 4.1.3

stud\_surv <- data.frame(read.csv("Student Survey.csv"))  
cov(stud\_surv)

## TimeReading TimeTV Happiness Gender  
## TimeReading 3.05454545 -20.36363636 -10.350091 -0.08181818  
## TimeTV -20.36363636 174.09090909 114.377273 0.04545455  
## Happiness -10.35009091 114.37727273 185.451422 1.11663636  
## Gender -0.08181818 0.04545455 1.116636 0.27272727

1. TimeReading is measured in hours, TimeTV is measured in minutes. While TimeReading and TimeTV are both measurements of time, there is a conversion that needs to be done in order to have them measured the same way. In respect to the relationships represented by covariance, this does not have an effect – the negative relationships stay negative, and positive relationships stay positive. What is will effect, however, is the magnitude of the covariance, which, as covariance is unscaled, can be difficult to interpret. For uniformity of data, as well as reducing the effects on future calculations, the best practice is to convert all similar variables (i.e. time, distance, temperature, etc) into the same measurements (i.e. hrs, km, Kelvins, etc). In this case, converting the TimeTV column to hours would be a better alternative to the current measurements. Happiness appears to be a percentage measure on a scale of 0 - 100%, and Gender is currently an integer. While Gender can not be quantified by numbers, factored variables cannot be included in a covariance calculation.

New data frame with converted values of TimeTV

stud\_surv\_hours <- stud\_surv %>% mutate(TimeTV = TimeTV/60)  
stud\_surv\_hours

## TimeReading TimeTV Happiness Gender  
## 1 1 1.5000000 86.20 1  
## 2 2 1.5833333 88.70 0  
## 3 2 1.4166667 70.17 0  
## 4 2 1.3333333 61.31 1  
## 5 3 1.2500000 89.52 1  
## 6 4 1.1666667 60.50 1  
## 7 4 1.2500000 81.46 0  
## 8 5 1.0000000 75.92 1  
## 9 5 1.0833333 69.37 0  
## 10 6 0.8333333 45.67 0  
## 11 6 1.1666667 77.56 1

Covariance of converted data

cov(stud\_surv\_hours)

## TimeReading TimeTV Happiness Gender  
## TimeReading 3.05454545 -0.3393939394 -10.350091 -0.0818181818  
## TimeTV -0.33939394 0.0483585859 1.906288 0.0007575758  
## Happiness -10.35009091 1.9062878788 185.451422 1.1166363636  
## Gender -0.08181818 0.0007575758 1.116636 0.2727272727

1. The correlation test chosen is the Pearson Correlation, as not only is its range specified between -1 and 1, but it can indicate the strength of relationships as well as direction. As the covariance calculation calculates the direction of correlation, positive or negative, I would expect the Pearson test to yield the same results as the calculation performed in section B.

D.1 Pearson Correlation of all variables

cor(stud\_surv\_hours, use = "everything", method = "pearson")

## TimeReading TimeTV Happiness Gender  
## TimeReading 1.00000000 -0.883067681 -0.4348663 -0.089642146  
## TimeTV -0.88306768 1.000000000 0.6365560 0.006596673  
## Happiness -0.43486633 0.636555986 1.0000000 0.157011838  
## Gender -0.08964215 0.006596673 0.1570118 1.000000000

D.2 Pearson Correlation of TimeTV vs. TimeReading

cor.test(stud\_surv\_hours$TimeTV, stud\_surv\_hours$TimeReading, use = "everything", method = "pearson")

##   
## Pearson's product-moment correlation  
##   
## data: stud\_surv\_hours$TimeTV and stud\_surv\_hours$TimeReading  
## t = -5.6457, df = 9, p-value = 0.0003153  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.9694145 -0.6021920  
## sample estimates:  
## cor   
## -0.8830677

D.3 Pearson Correlation of TimeTV vs. TimeReading with Confidence Interval at 99%

cor.test(stud\_surv\_hours$TimeTV, stud\_surv\_hours$TimeReading, use = "everything", method = "pearson", conf.level = .99)

##   
## Pearson's product-moment correlation  
##   
## data: stud\_surv\_hours$TimeTV and stud\_surv\_hours$TimeReading  
## t = -5.6457, df = 9, p-value = 0.0003153  
## alternative hypothesis: true correlation is not equal to 0  
## 99 percent confidence interval:  
## -0.9801052 -0.4453124  
## sample estimates:  
## cor   
## -0.8830677

D.4 The negative Pearson correlation coefficient points to a negative relationship between time spent watching TV and time spent reading - as one increases, the other decreases. The magnitude of the correlation is demonstrated by how close to -1 or 1 it is, with .5/-.5 being a benchmark for significance. With a coefficient of -0.8830677, this indicates a very significant relationship. Additionally, both the confidence intervals at 95% and 99% never cross 0, indicating a high confidence that the two are negatively correlated.

1. The correlation coefficient is -0.8830677 as seen in D.3. As time spent reading itself can be influenced by a number of outside factors other than time spent watching TV, we can take the correlation coefficient one step further by squaring it and finding the coefficient of determination (R^2), which can give a measure of variability in one variable that is shared by the other. R^2 in this case is .779809, which, converted to a percentage, is 77.98%, meaning that time spent watching TV shares 77.98% of variability with time spent reading. The transalates into time spent watching TV accounting for 77.98% of the variation in time spent reading, and only 22.02% of the variation in time spent reading being from outside factors.
2. With the high coefficient of determination and strong relationship determined by the correlation coefficient, I am comfortable saying that the time spent reading was negatively impacted by time spent watching TV. While there are always be outside forces acting on the individuals, one explanation is the fact that there is a limited number of hours per day, and when there is more time spent watching TV, there is less time available for reading. Even with outside factors, though, the data shows that when time spent watching TV is increased, time spent reading readily decreases.
3. The partial correlation between time spent watching TV and Happiness is .6435158, which is not only positive, but is also about .5, pointing towards a strong relationship. This indicates that the more time spent watching TV, the happier an individual is, regardless of Gender. This could also be a determining factor in the negative relationship between time spent watching TV and time spent reading - if time spent watching TV increases happiness, a person will gravitate towards that. Further analysis between time spent reading and happiness could also support this idea.

Partial Correlation between TimeTV and Happiness, controlling for Gender

pcor(c("TimeTV", "Happiness", "Gender"), var(stud\_surv\_hours))

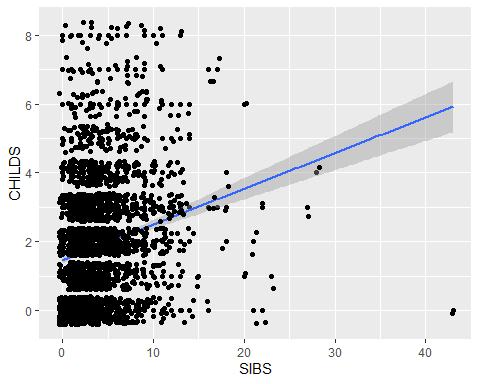
## [1] 0.6435158

library(ggplot2)  
library(ggm)

# Part 2

A. According to the below plot, we can see that as the number of siblings a respondent has (SIBS) increased, so does the number of children they have (CHILDS), shown by the positive linear relationship, indicating a positive correlation.

gss2016 <- data.frame(read.csv("gss-2016.csv"))  
ggplot(gss2016, aes(x = SIBS, y = CHILDS)) + geom\_point() + geom\_smooth(method = "lm") + geom\_jitter()



B. The covariance between SIBS and CHILDS, accounting for incomplete observations is calculated below:

cov(gss2016$SIBS, gss2016$CHILDS, use = "complete.obs")

## [1] 1.064853

The positive number confirms a positive relationship between SIBS and CHILDS, meaning, as one goes up, so does the other. It does not indicate the strength of the relationship, but it can help in predicting the direction of estimated values.

C. The correlation test chosen is the Pearson Correlation, as not only is its range specified between -1 and 1, but it can indicate the strength of relationships as well as direction. As the covariance calculation calculates the direction of correlation, positive or negative, I would expect the Pearson test to yield the same results as the calculation performed in section B.

D. The Pearson correlation test is calculated below:

cor(gss2016$SIBS, gss2016$CHILDS, use = "complete.obs", method = 'pearson')

## [1] 0.1988582

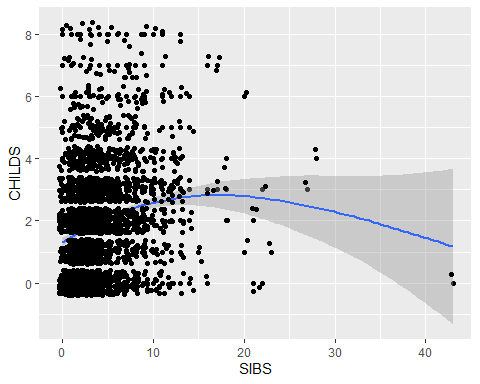
The positive number further confirms what the covariance indicated - a positive relationship between SIBS and CHILDS. However, it is a small number, closer to 0 than 1, indicating that the correlation is not strong. In order to have a significant relationship, the correlation would need to be .5 or larger - a correlation of .1988582 is weak to moderate at best.

E. The correlation coefficient is described above at .1988582. The coefficient of determination is simply the square of the correlation coefficient, which is calculated at .03954459. The coefficient of determination gives a measure of variability in one variable that is shared by the other, as outside factors can be affecting the results of how many children a person has. We can use the coefficient of determination to determine the percentage of affect that SIBS has on CHILDS but multiplying by 100%, giving us 3.95%, which means that the number of siblings a respondent can account for only 3.95% of the variability of the number of children a respondent had. This leaves 96.05% to outside factors.

F. Due to the positive correlation and covariance, I believe the relationship to be positive - as one increases, the other tends to as well; however, due to the small coefficient of determination, I believe that outside factors have a much larger effect on them rather than the number of siblings affecting the number of children a respondent would have. There is a relationship between the two, but it is weak.

G. Corresponding graph:

ggplot(gss2016, aes(x = SIBS, y = CHILDS)) + geom\_point() + geom\_smooth() + geom\_jitter()



# Part 3

A. Regression Analysis:

SIBMOD <- lm(CHILDS ~ SIBS, data = gss2016)  
summary(SIBMOD)

##   
## Call:  
## lm(formula = CHILDS ~ SIBS, data = gss2016)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -5.9216 -1.5713 0.0143 1.0143 6.5322   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.467767 0.046889 31.30 <2e-16 \*\*\*  
## SIBS 0.103577 0.009555 10.84 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.637 on 2854 degrees of freedom  
## (11 observations deleted due to missingness)  
## Multiple R-squared: 0.03954, Adjusted R-squared: 0.03921   
## F-statistic: 117.5 on 1 and 2854 DF, p-value: < 2.2e-16

B. From the regression analysis, we can see that the intercept is 1.467767, and the slope is .103577. The coefficient of determination is described above at .03954. Since the coefficient of determination is simply the square of the correlation coefficient, we can calculate the the correlation coefficient to be .198847.

C. The coefficient of determination gives a measure of variability in one variable that is shared by the other, as outside factors can be affecting the results of how many children a person has. We can use the coefficient of determination to determine the percentage of affect that SIBS has on CHILDS but multiplying by 100%, giving us 3.95%, which means that the number of siblings a respondent can account for only 3.95% of the variability of the number of children a respondent had. This leaves 96.05% to outside factors.

D. With an F-Ratio of 117.5, this points to a much better model, as we tend to look for an F-Ratio at least higher than 1, and the p-value is less than .001.

E. Using the model, CHILDS = slope(SIBS) + intercept, we can calculate that if a person were to have three siblings, they would be predicted to have 1.778498 children. But as we know that it is not possible to have a percentage of a child, it would round to 2.

F. Using the same model again, we can calculate that if a person has no siblings, they are predicted to have at least 1.467767 children, the intercept. Again, as we cannot have a percentage of a child, it would be rounded to 1.

# Part 4

library(dplyr)  
library(tidyr)  
library(readr)  
parse\_date("01/02/2010", "%d/%m/%Y")

## [1] "2010-02-01"

week7 <- read\_csv("week-7-housing.csv",  
 na = c("", "NA", "N/A"),  
 col\_types = cols(  
 zip5 = col\_factor(levels = NULL),  
 building\_grade = col\_factor(levels = NULL),  
 present\_use = col\_factor(levels = NULL),  
 Sale\_Date = col\_date(format = "%m/%d/%Y"),  
 sale\_instrument = col\_factor(levels = NULL),  
 sale\_warning = col\_factor(levels = NULL),  
 sitetype = col\_factor(levels = NULL),  
 #year\_renovated = col\_factor(levels = c(0,1970:1979,1980:1989,1990:1999,2000:2009,2010:2019))  
 ))

## Week 7 Housing

A. I have not removed any data points as of yet.

B. Models for Sale Price vs Square Footage of the Lot, and for Sale Price vs Square Footage of the Lot plus number of Bedrooms and number of Full Bathrooms

Price\_SqFoot <- lm(Sale\_Price ~ sq\_ft\_lot, data = week7)  
Price\_BR\_FullB <- lm(Sale\_Price ~ sq\_ft\_lot + bedrooms + bath\_full\_count, data = week7)

C. Summaries for Price\_SqFoot and Price\_BR\_FullB:

##   
## Call:  
## lm(formula = Sale\_Price ~ sq\_ft\_lot, data = week7)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2016064 -194842 -63293 91565 3735109   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.418e+05 3.800e+03 168.90 <2e-16 \*\*\*  
## sq\_ft\_lot 8.510e-01 6.217e-02 13.69 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 401500 on 12863 degrees of freedom  
## Multiple R-squared: 0.01435, Adjusted R-squared: 0.01428   
## F-statistic: 187.3 on 1 and 12863 DF, p-value: < 2.2e-16

##   
## Call:  
## lm(formula = Sale\_Price ~ sq\_ft\_lot + bedrooms + bath\_full\_count,   
## data = week7)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3566287 -153218 -51375 69545 3736381   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.429e+05 1.481e+04 9.649 <2e-16 \*\*\*  
## sq\_ft\_lot 7.227e-01 5.910e-02 12.229 <2e-16 \*\*\*  
## bedrooms 6.887e+04 4.027e+03 17.099 <2e-16 \*\*\*  
## bath\_full\_count 1.458e+05 5.422e+03 26.888 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 381000 on 12861 degrees of freedom  
## Multiple R-squared: 0.1127, Adjusted R-squared: 0.1125   
## F-statistic: 544.3 on 3 and 12861 DF, p-value: < 2.2e-16

The R2 and Adjusted R2 for Price\_SqFoot are .01435 and .01428, respectively, while the R2 and Adjusted R2 for Price\_BR\_FullB are .1127 and .1125, respectively. The decreases from R2 to Adjusted R2 are indicators of how well a model fits a curve while adjusting for the number of variables in a model, and the more useless or unrelated an added variable is, the more the Adjusted R2 decreases. As the R2 for Price\_SqFoot decreased by .00003 and the R2 for Price\_BR\_FullB decreased by only .0002, we can tell that as the number of Bedrooms and Full Bathrooms are added to the model, it begins to not fit as well, indicating that the number of Bedrooms and Full Bathrooms are not as related as Square Footage of the Lot is to the Sale Price.

D. The standardized betas for Price\_SqFoot and Price\_BR\_FullB:

library(lm.beta)  
lm.beta(Price\_SqFoot)

##   
## Call:  
## lm(formula = Sale\_Price ~ sq\_ft\_lot, data = week7)  
##   
## Standardized Coefficients::  
## (Intercept) sq\_ft\_lot   
## 0.0000000 0.1198122

lm.beta(Price\_BR\_FullB)

##   
## Call:  
## lm(formula = Sale\_Price ~ sq\_ft\_lot + bedrooms + bath\_full\_count,   
## data = week7)  
##   
## Standardized Coefficients::  
## (Intercept) sq\_ft\_lot bedrooms bath\_full\_count   
## 0.0000000 0.1017484 0.1492025 0.2346207

The Standard Deviations for Sale Price, Square Footage of the lot, Number of Bedrooms, and Number of Full Baths:

sd(week7$Sale\_Price, na.rm = TRUE)

## [1] 404381.1

sd(week7$sq\_ft\_lot, na.rm = TRUE)

## [1] 56933.29

sd(week7$bedrooms, na.rm = TRUE)

## [1] 0.8761273

sd(week7$bath\_full\_count, na.rm = TRUE)

## [1] 0.6507965

Standardized betas tell us about the relationship between Sale Price and it’s predictors. Much like correlation, the positive values indicate positive relationships between Sale Price and Square Footage of the lot, as well as Sale Price and Bedrooms and Full Baths. However, unlike correlation, we can determine the degree that “each predictor affects the outcome if the effects of all other predictors are held constant.” Therefore, the standardized beta value of .1198122 for Square Footage of the lot means that for every standard deviation that Square Footage increases, Sale Price increases .1198122 standard deviations. Since one standard deviation of Square Footage of the lot is 56,933.29 feet and one standard deviation of Sale Price is $404,381.10, we know that for ever 56,933.29 increase in Square Footage of the lot, the Sale Price increases 0.1198122($404,381.10) = $48,449.79.

However, when Bedrooms and Full Bathrooms are added to the equation, the standardized beta value of the Square Footage of the lot becomes 0.1017484. While still indicating a positive relationship, the degree of change is now different. As the Square Footage of the lot increases by one standard deviation - 56,933.29 - and the number of bedrooms and full bathrooms stay constant, the Sale Price increases 0.1017484 standard deviation, which is calculated as 0.1017484($404,381.10) = $41,145.13.

Similarly, the standardized beta value of Bedrooms is 0.1492025, indicating a positive relationship where the number of Bedrooms increases by 1 and the Square Footage of the lot and Full Bathrooms stay constant. Using the calculated standard deviation value for Bedrooms, we can see that for every .8761273 increase in Bedrooms while the Square Footage of the lot and number of Full Bathrooms stays constant, the Sale Price increases by 0.1492025($404,381.10)= $60,334.67.

The standardized beta value of Full Bathrooms is 0.2346207, indicating a positive relationship where the number of Full Bathrooms increases by 1 and the Square Footage of the lot and Bedrooms stay constant. Using the calculated standard deviation value for Full Baths, we can see that for every .6507965 increase in Full Baths while the Square Footage of the lot and number of Bedrooms stay constant, the Sale Price increases by 0.2346207($404,381.10) = $94,876.18.

E. The Confidence Intervals for Sale Price, Square Footage of the lot, Number of Bedrooms, and Number of Full Baths:

library(Rmisc)

## Warning: package 'Rmisc' was built under R version 4.1.3

## Loading required package: lattice

## Loading required package: plyr

## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.  
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:  
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##   
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

CI(week7$Sale\_Price)

## upper mean lower   
## 667726.1 660737.7 653749.4

CI(week7$sq\_ft\_lot)

## upper mean lower   
## 23212.47 22228.57 21244.67

CI(week7$bedrooms)

## upper mean lower   
## 3.493804 3.478663 3.463522

CI(week7$bath\_full\_count)

## upper mean lower   
## 1.809692 1.798445 1.787199

F. The analysis of variance between Price\_SqFoot and Price\_BR\_FullB:

anova(Price\_SqFoot, Price\_BR\_FullB)

## Analysis of Variance Table  
##   
## Model 1: Sale\_Price ~ sq\_ft\_lot  
## Model 2: Sale\_Price ~ sq\_ft\_lot + bedrooms + bath\_full\_count  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 12863 2.0734e+15   
## 2 12861 1.8666e+15 2 2.068e+14 712.43 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

With a Pr(>F) value of 2.2e-16 being a very small number, we can conclude that the second model, Price\_BR\_FullB significantly improved the fit of the model to the data.

week7\_nooutliers <- week7 %>%  
 filter(sq\_ft\_lot < 1250000, bath\_full\_count < 10, bedrooms < 10)

H. The standardized residuals of Price\_SqFoot and Price\_BR\_FullB:

week7$stresids\_Price\_SqFoot <- rstandard(Price\_SqFoot)  
week7$stresids\_Price\_BR\_FullB <- rstandard(Price\_BR\_FullB)  
week7$LGresids\_Price\_SqFoot <- rstandard(Price\_SqFoot) < -2 |rstandard(Price\_SqFoot) > 2  
week7$LGresids\_Price\_BR\_FullB <- rstandard(Price\_BR\_FullB) < -2|rstandard(Price\_BR\_FullB) > 2

I. The number of variables with large residuals for Price\_SqFoot:

## [1] 334

The number of variables with large residuals for Price\_BR\_FullB:

## [1] 329

J. The variables with large residuals for Price\_SqFoot are:

## # A tibble: 334 x 3  
## Sale\_Price sq\_ft\_lot stresids\_Price\_SqFoot  
## <dbl> <dbl> <dbl>  
## 1 1900000 37017 3.06  
## 2 1520000 19173 2.15  
## 3 1588359 8752 2.34  
## 4 275000 532739 -2.05  
## 5 90000 574992 -2.60  
## 6 90000 574992 -2.60  
## 7 2500000 36362 4.55  
## 8 2169000 180774 3.42  
## 9 1534000 10247 2.20  
## 10 65000 305173 -2.09  
## # ... with 324 more rows

The variables with large residuals for Price\_BR\_FullB are:

## # A tibble: 329 x 5  
## Sale\_Price sq\_ft\_lot bedrooms bath\_full\_count stresids\_Price\_BR\_FullB  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 265000 112650 4 4 -2.15  
## 2 1900000 37017 4 3 2.67  
## 3 1390000 225640 0 1 2.47  
## 4 1588359 8752 2 2 2.65  
## 5 1450000 14043 3 2 2.10  
## 6 1450000 14043 2 1 2.66  
## 7 270000 89734 4 23 -9.81  
## 8 90000 574992 3 1 -2.16  
## 9 90000 574992 3 1 -2.16  
## 10 2500000 36362 4 2 4.63  
## # ... with 319 more rows

K. The leverage of Price\_SqFoot and Price\_BR\_FullB, respectively:

week7$lev\_Price\_SqFoot <- hatvalues(Price\_SqFoot)  
week7$lev\_Price\_BR\_FullB <- hatvalues(Price\_BR\_FullB)

The Cook’s Distance of Price\_SqFoot and Price\_BR\_FullB, respectively:

week7$cd\_Price\_SqFoot <- cooks.distance(Price\_SqFoot)  
week7$cd\_Price\_BR\_FullB <- cooks.distance(Price\_BR\_FullB)

The covariance rations of Price\_SqFoot and Price\_BR\_FullB, respectively:

week7$cvratio\_Price\_SqFoot <- covratio(Price\_SqFoot)  
week7$cvratio\_Price\_BR\_FullB <- covratio(Price\_BR\_FullB)

L. The Durbin-Watson Test to assert independence of variables is calculated below for Price\_SqFoot and Price\_BR\_FullB, respectively:

## lag Autocorrelation D-W Statistic p-value  
## 1 0.6309692 0.7380424 0  
## Alternative hypothesis: rho != 0

## lag Autocorrelation D-W Statistic p-value  
## 1 0.6487202 0.7025582 0  
## Alternative hypothesis: rho != 0

As both DW values are .7 approximately, we know that the variables in each model are positively correlated and do not meet the condition of independence.

M. As Price\_SqFoot only has one predictor, we are unable to run the VIF tests to determine multicollinearity. The tests for Price\_BR\_FullB are below:

vif(Price\_BR\_FullB)

## sq\_ft\_lot bedrooms bath\_full\_count   
## 1.003410 1.103585 1.103567

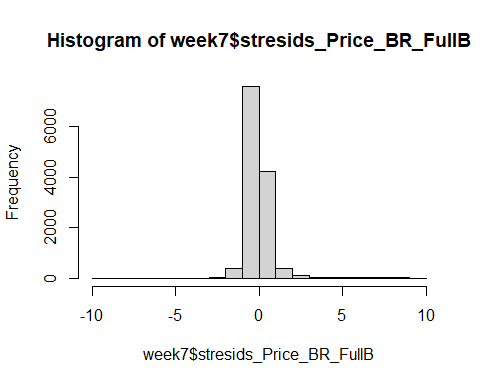
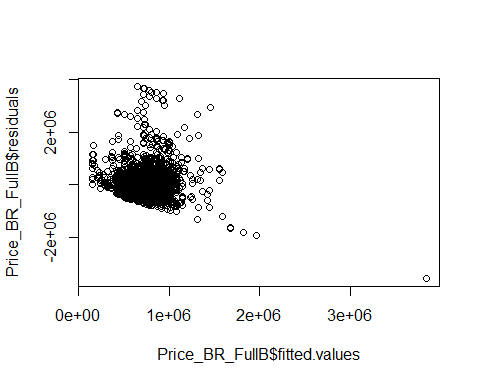
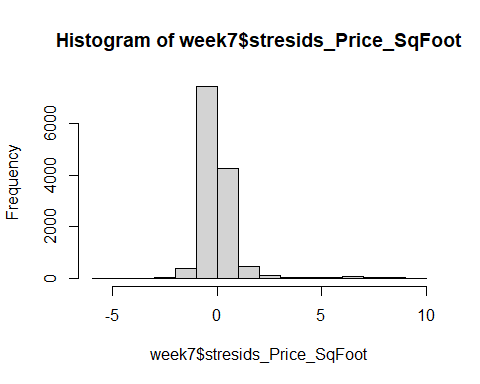
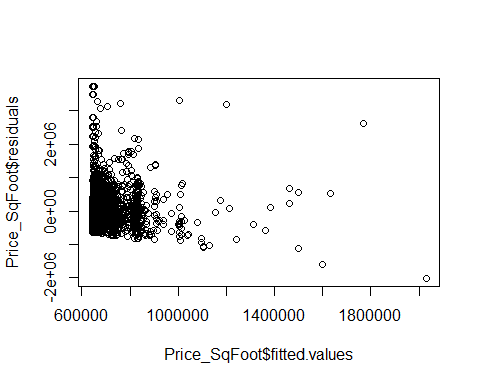
1/vif(Price\_BR\_FullB)

## sq\_ft\_lot bedrooms bath\_full\_count   
## 0.9966015 0.9061373 0.9061528

mean(vif(Price\_BR\_FullB))

## [1] 1.070187

As the VIF values are below 10, the tolerance values are above .2 and the average VIF is fairly close to 1, we can conclude that there is no multicollinearity in our data.

N. Plots and histograms of Price\_SqFoot and Price\_BR\_FullB, respectively 

In each plot function, you can see an outlier in the bottom right corner, indicating a very large fitted value, but a very small residual. This can affect the plot, causing the remaining values to be concentrated on the far left.

# Part 5

## University of California Irvine - Thoracic Surgery Data

A. The summary for the binary logistic regression model is shown below:

##   
## Call:  
## glm(formula = Risk1Yr ~ ., family = "binomial", data = ThoracicSurgery)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6084 -0.5439 -0.4199 -0.2762 2.4929   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.655e+01 2.400e+03 -0.007 0.99450   
## DGNDGN2 1.474e+01 2.400e+03 0.006 0.99510   
## DGNDGN3 1.418e+01 2.400e+03 0.006 0.99528   
## DGNDGN4 1.461e+01 2.400e+03 0.006 0.99514   
## DGNDGN5 1.638e+01 2.400e+03 0.007 0.99455   
## DGNDGN6 4.089e-01 2.673e+03 0.000 0.99988   
## DGNDGN8 1.803e+01 2.400e+03 0.008 0.99400   
## PRE4 -2.272e-01 1.849e-01 -1.229 0.21909   
## PRE5 -3.030e-02 1.786e-02 -1.697 0.08971 .   
## PRE6PRZ1 -4.427e-01 5.199e-01 -0.852 0.39448   
## PRE6PRZ2 -2.937e-01 7.907e-01 -0.371 0.71030   
## PRE7T 7.153e-01 5.556e-01 1.288 0.19788   
## PRE8T 1.743e-01 3.892e-01 0.448 0.65419   
## PRE9T 1.368e+00 4.868e-01 2.811 0.00494 \*\*  
## PRE10T 5.770e-01 4.826e-01 1.196 0.23185   
## PRE11T 5.162e-01 3.965e-01 1.302 0.19295   
## PRE14OC12 4.394e-01 3.301e-01 1.331 0.18318   
## PRE14OC13 1.179e+00 6.165e-01 1.913 0.05580 .   
## PRE14OC14 1.653e+00 6.094e-01 2.713 0.00668 \*\*  
## PRE17T 9.266e-01 4.445e-01 2.085 0.03709 \*   
## PRE19T -1.466e+01 1.654e+03 -0.009 0.99293   
## PRE25T -9.789e-02 1.003e+00 -0.098 0.92227   
## PRE30T 1.084e+00 4.990e-01 2.172 0.02984 \*   
## PRE32T -1.398e+01 1.645e+03 -0.008 0.99322   
## AGE -9.506e-03 1.810e-02 -0.525 0.59944   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 395.61 on 469 degrees of freedom  
## Residual deviance: 341.19 on 445 degrees of freedom  
## AIC: 391.19  
##   
## Number of Fisher Scoring iterations: 15

B. According to the summary, the variables with the greatest effect on the survival rate were DGNDGN8 with a slope of 18.03, DGNDGN4 with a slope of 16.38, and DGNDGN2 with a slope of 14.74.

C. Using the dataset provided, we can create training and test subsets of the data and train a model to predict the accuracy of our model that predicts whether or not a patient survived after one year, given all of the variables:

#split the data  
split <- sample.split(ThoracicSurgery, SplitRatio = .8)  
#create training data subset  
train <- subset(ThoracicSurgery, split == "TRUE")  
#create test data subset  
test <- subset(ThoracicSurgery, split == "FALSE")  
  
#create model of new training data subset  
trainmodel <- glm(Risk1Yr ~ ., data = train, family = 'binomial')  
  
#use test data on training model  
testpredict <- predict(trainmodel, test, type = "response")   
testpredict <- predict(trainmodel, train, type = "response")  
  
#use confustion matrix to compute accuracy of model  
confmatrix <- table(Actual\_Value = train$Risk1Yr, Predicted\_value = testpredict > .5)  
confmatrix

## Predicted\_value  
## Actual\_Value FALSE TRUE  
## F 290 12  
## T 51 7

((confmatrix[[1,1]] + confmatrix[[2,2]]) / sum(confmatrix)) \* 100

## [1] 82.5

According to the confusion matrix, our model is approximately 85% accurate.

# Part 6

## Binary Classifier Data

train\_index <- sample(1:nrow(binaryclass), (nrow(binaryclass)\*.8))  
test\_index <- setdiff(1:nrow(binaryclass), train\_index)  
  
#create training data subset  
train <- binaryclass[train\_index, ]  
#create test data subset  
test <- binaryclass[test\_index, ]  
  
#create model of new training data subset  
trainmodel <- glm(label ~ ., data = train, family = 'binomial')  
  
#use test data on training model  
testpredict <- predict(trainmodel, test, type = "response")   
testpredict <- predict(trainmodel, train, type = "response")  
  
#use confustion matrix to compute accuracy of model  
confmatrix <- table(Actual\_Value = train$label, Predicted\_value = testpredict > .5)  
confmatrix

## Predicted\_value  
## Actual\_Value FALSE TRUE  
## 0 348 268  
## 1 234 348

((confmatrix[[1,1]] + confmatrix[[2,2]]) / sum(confmatrix)) \* 100

## [1] 58.09683

B. The accuracy of the nearest neighbors algorithm is calculated below:

train\_labels <- binaryclass[train\_index, 1, drop = TRUE]  
test\_labels <- binaryclass[test\_index, 1, drop = TRUE]  
  
binclasspred <- knn(train = train, test = test, cl = train\_labels, k = 39)  
library(gmodels)

## Warning: package 'gmodels' was built under R version 4.1.3

CrossTable(x = test\_labels, y = binclasspred, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 300   
##   
##   
## | binclasspred   
## test\_labels | 0 | 1 | Row Total |   
## -------------|-----------|-----------|-----------|  
## 0 | 146 | 5 | 151 |   
## | 0.967 | 0.033 | 0.503 |   
## | 0.961 | 0.034 | |   
## | 0.487 | 0.017 | |   
## -------------|-----------|-----------|-----------|  
## 1 | 6 | 143 | 149 |   
## | 0.040 | 0.960 | 0.497 |   
## | 0.039 | 0.966 | |   
## | 0.020 | 0.477 | |   
## -------------|-----------|-----------|-----------|  
## Column Total | 152 | 148 | 300 |   
## | 0.507 | 0.493 | |   
## -------------|-----------|-----------|-----------|  
##   
##

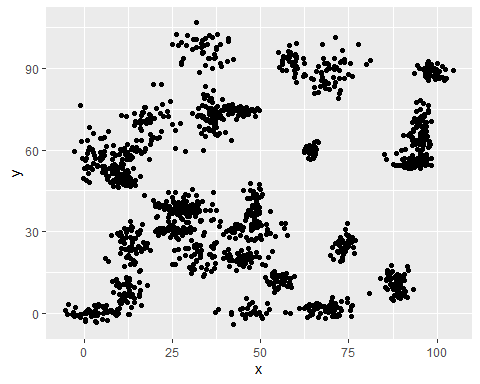
We can see that 128 cases were accurately predicted as “0”, and 160 cases were accurately predicted as “1”, which, for a total amount of cases of 300, is an accurary of 128+160/300 = 96% accuracy. When compared to the regression model, we can see that the nearest neighbor algorithm is much more accurate for this data set, as the regression model was only 50%-55% accuarate.

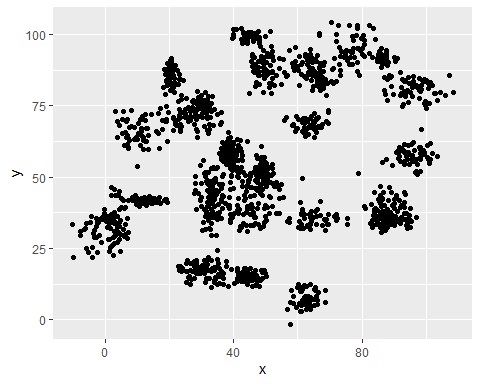
C. We’re seeing an improvement in accuracy from the KNN model versus the logistic regression model for a few reasons. In this data set, we only had two input variables, and the KNN tends to work better on a small number of inputs. Additionally, the KNN does not make assumptions about the functional form of the problem being solved, while logistic regression will always be a Sigmoid curve. Lastly, while it was not specifically analyzed, colinearity and outliers tend to interfere with the accuracy of a logistic regression model more than with a KNN.

# Part 7

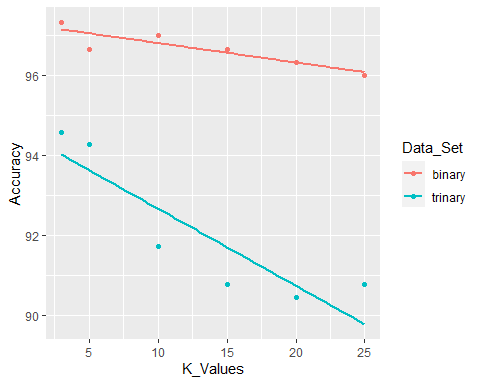
## Binary vs. Trinary

A. Scatterplot graph of the binary data:



Scatterplot graph of the trinary data: 

B.Graph of the Accuracy of the Binary and Trinary data with different KNN values:

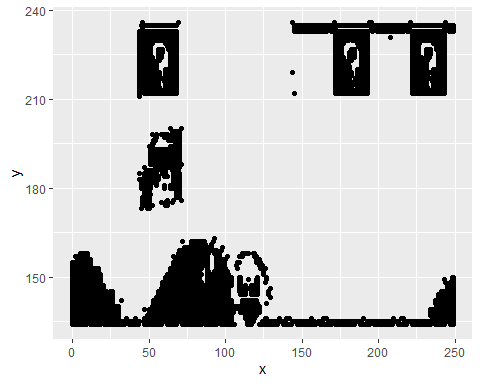


C. I don’t believe that a linear classifier would work well on these data sets, as they each have plots that could overlap each other, rather than be segregated by a line to classify each set.

# Part 8

## Clustering Data

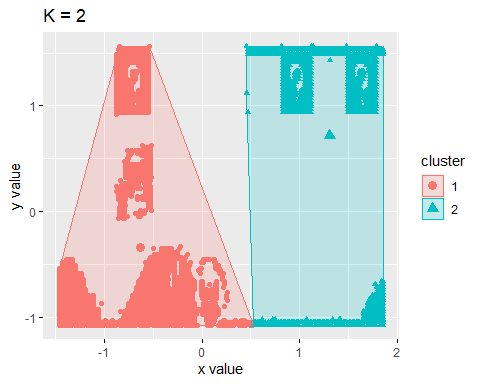
A. Scatter plot of the cluster data:



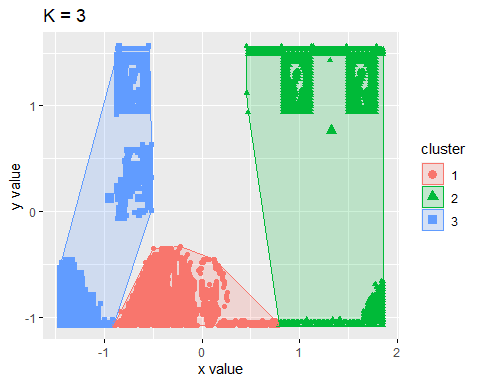
B. Scatter Plots of data fitted with k-means algorithm using K values 2 through 12:

K2 <- kmeans(x = cluster, 2)  
K3 <- kmeans(x = cluster, 3)  
K4 <- kmeans(x = cluster, 4)  
K5 <- kmeans(x = cluster, 5)  
K6 <- kmeans(x = cluster, 6)  
K7 <- kmeans(x = cluster, 7)  
K8 <- kmeans(x = cluster, 8)  
K9 <- kmeans(x = cluster, 9)  
K10 <- kmeans(x = cluster, 10)  
K11 <- kmeans(x = cluster, 11)  
K12 <- kmeans(x = cluster, 12)

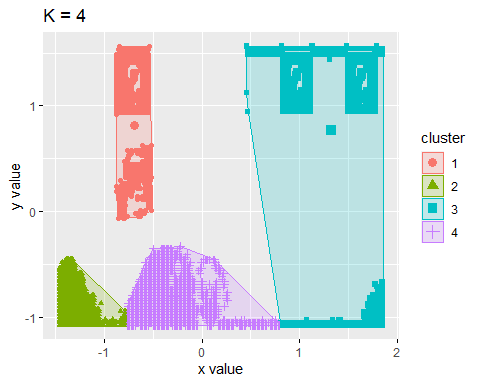
fviz\_cluster(K2, geom = "point", data = cluster) + ggtitle("K = 2")



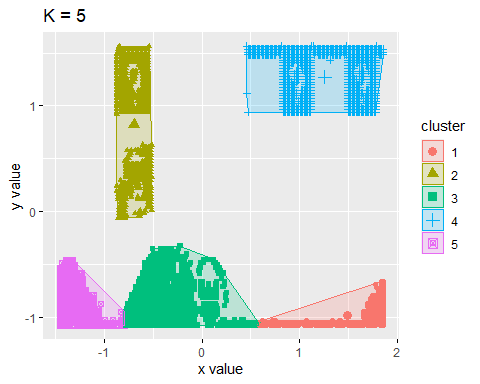
fviz\_cluster(K3, geom = "point", data = cluster) + ggtitle("K = 3")



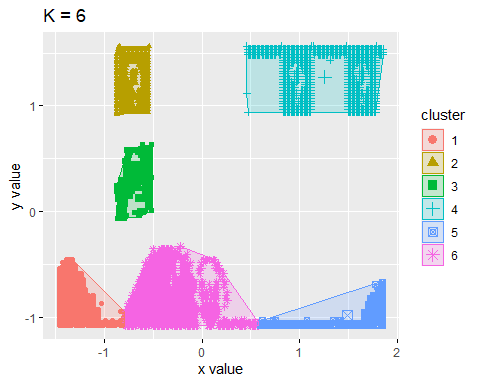
fviz\_cluster(K4, geom = "point", data = cluster) + ggtitle("K = 4")



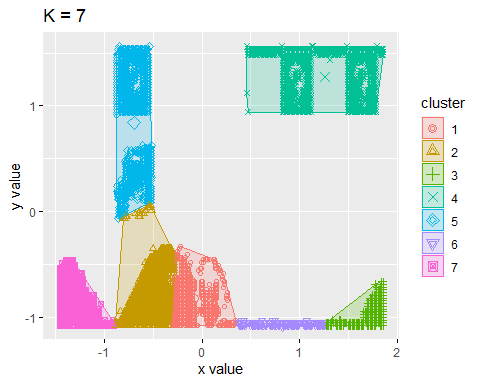
fviz\_cluster(K5, geom = "point", data = cluster) + ggtitle("K = 5")



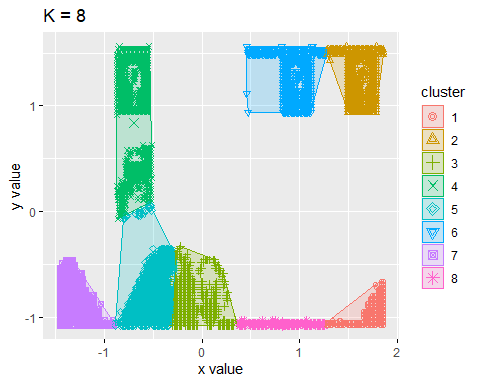
fviz\_cluster(K6, geom = "point", data = cluster) + ggtitle("K = 6")



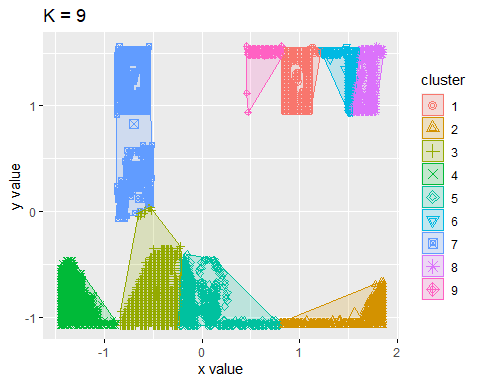
fviz\_cluster(K7, geom = "point", data = cluster) + ggtitle("K = 7")



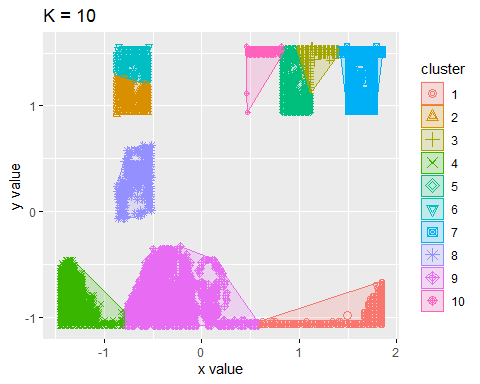
fviz\_cluster(K8, geom = "point", data = cluster) + ggtitle("K = 8")



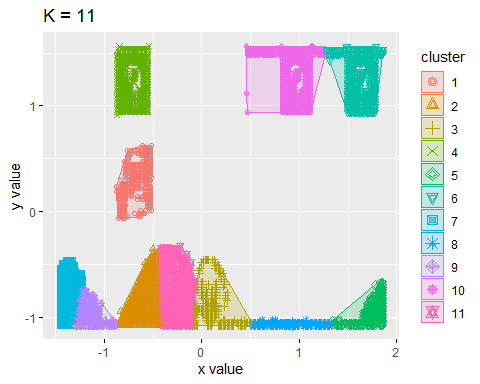
fviz\_cluster(K9, geom = "point", data = cluster) + ggtitle("K = 9")



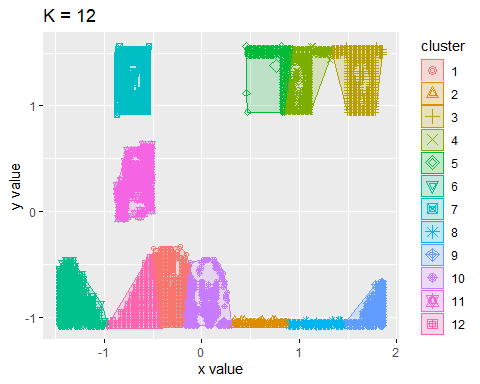
fviz\_cluster(K10, geom = "point", data = cluster) + ggtitle("K = 10")



fviz\_cluster(K11, geom = "point", data = cluster) + ggtitle("K = 11")

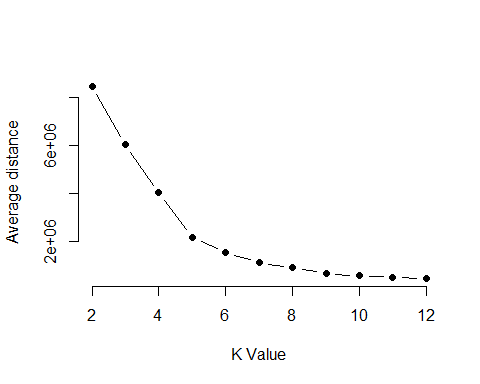


fviz\_cluster(K12, geom = "point", data = cluster) + ggtitle("K = 12")



C. Line chart of average distances from center for each K Value:

wss <- function(k) {kmeans(cluster, k, nstart = 12)$tot.withinss}  
k.values <- 2:12  
wss\_values <- map\_dbl(k.values, wss)  
plot(k.values, wss\_values, type = "b", pch = 19, frame = FALSE, xlab = "K Value", ylab = "Average distance")



D. Using the above graph, the elbow point appears to be at the K Value point of 5. This means that the optimal number of clusters to calculate when using the K-means algorithm is 5.