통계패키지 활용 자료분석 (2020 년 2 학기)

담 당 교 수 : 김 태 수

강좌번호	100982-31001	본인의 과제 자체 평가	80 점

과제명 : 중간고사

이름	김태형

제 출 일	2020 년 10 월 22 일
학 과	기초교육학부(교류학생)
학 번	20180088

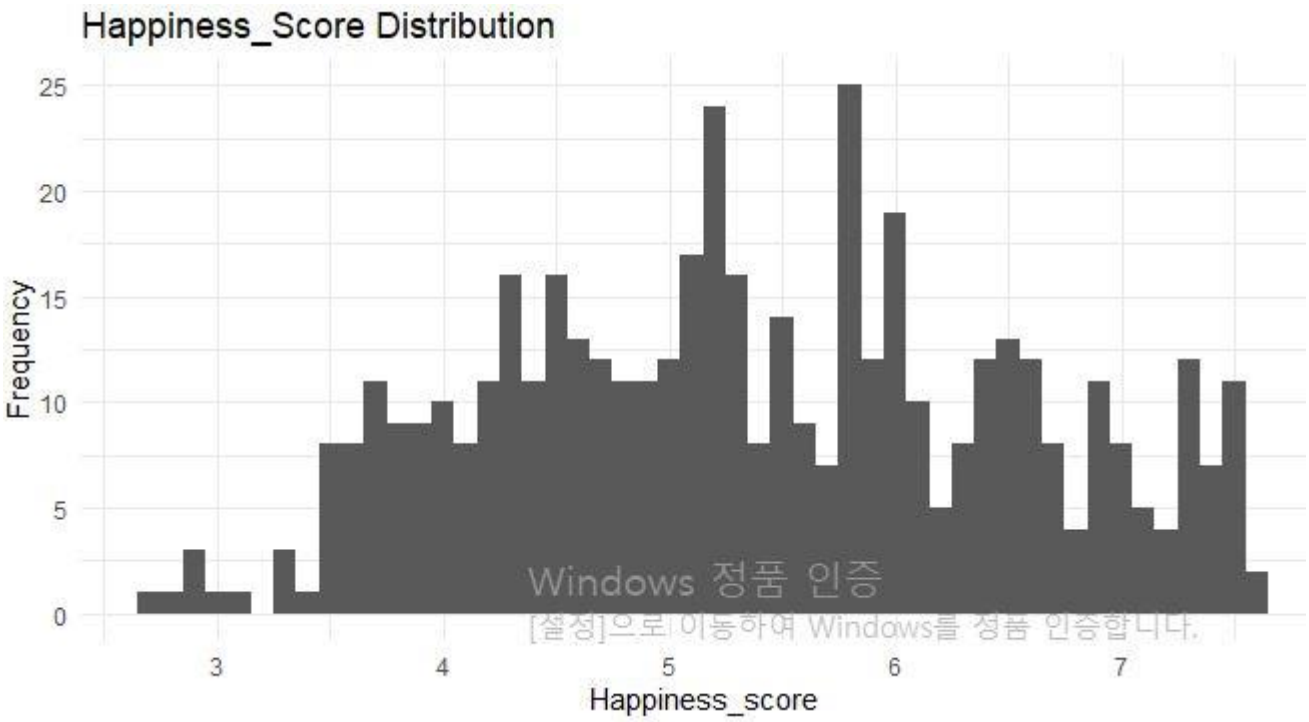
목차

1. 자료설명

제시된 데이터는 2015 년에서 2017 년까지 세계의 행복과 관련된 수치를 보여주는 데이터이다. 총 10 개의 행과 243 개의 열로 구성되어 있다.

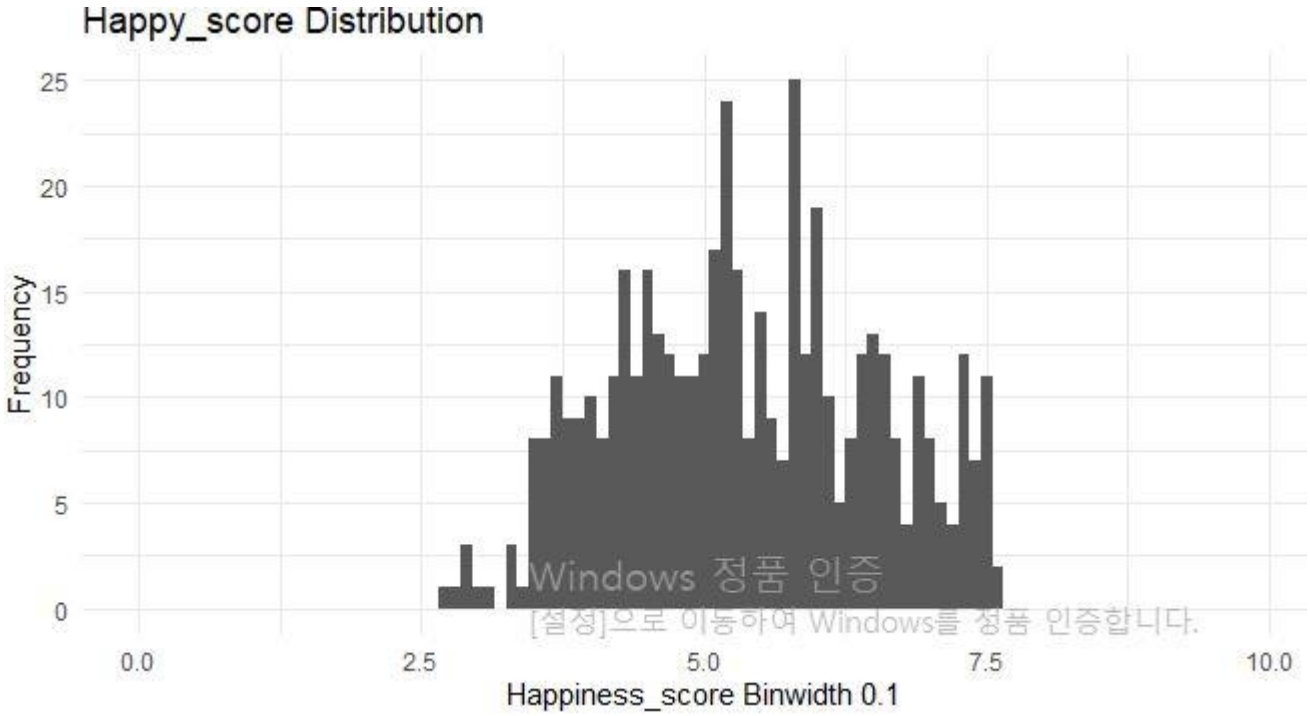
2. 시각화

```
ggplot(data=base01) +
  geom_histogram(binwidth=0.1, aes(x=base01$Happiness.Score)) +
  ggtitle("Happiness_Score Distribution") +
  xlab("Happiness_score") + ylab("Frequency") + theme_minimal()
```



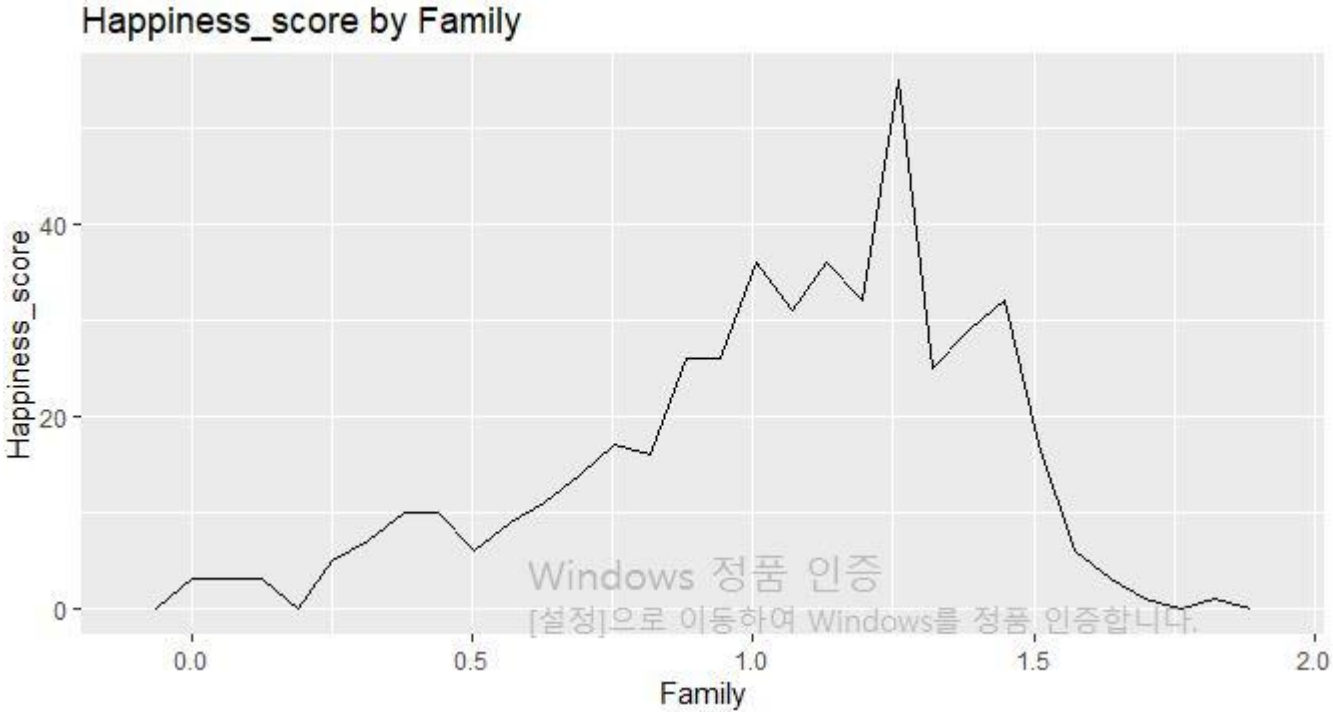
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```
ggplot(data=base01) +
  geom_histogram(binwidth=0.1, aes(x=base01$Happiness.Score)) +
  ggtitle("Happy_score Distribution") +
  xlab("Happiness_score Binwidth 0.1") +
  ylab("Frequency") + theme_minimal() + xlim(0,10)
```



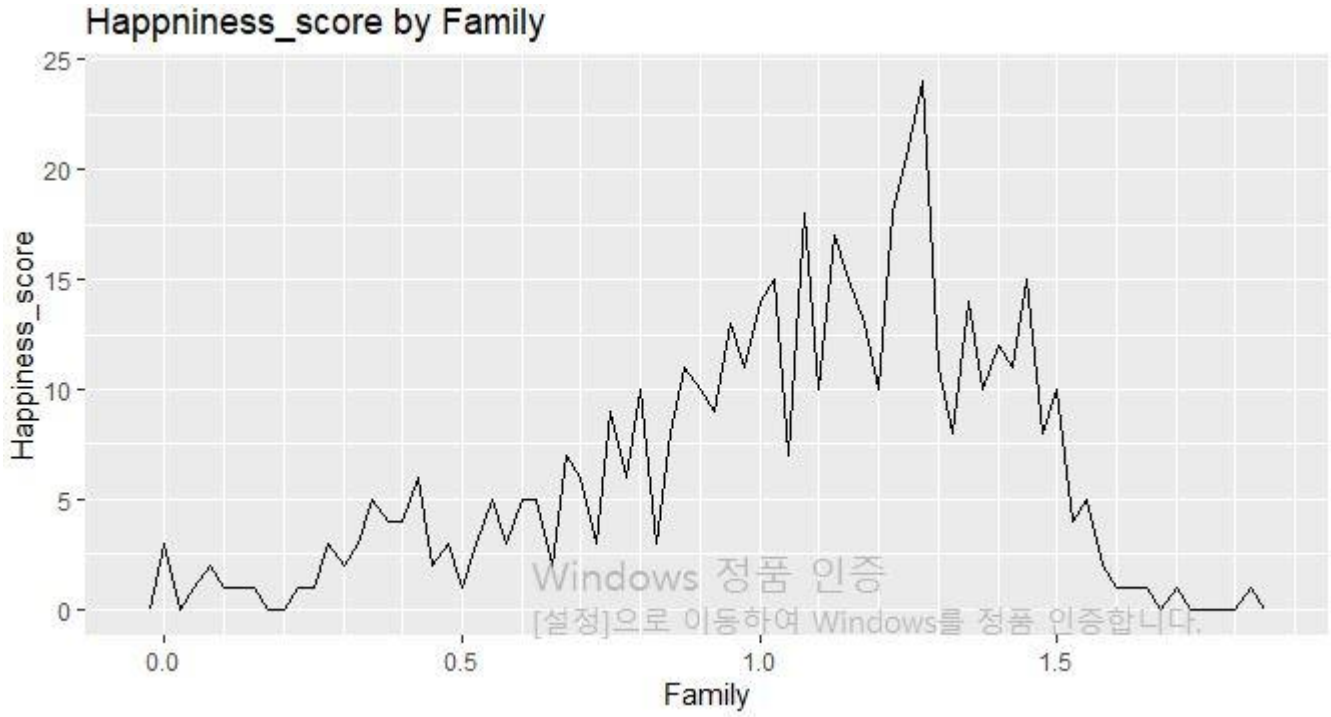
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```
ggplot(data=base01, aes(x=Family)) + geom_freqpoly() +
  ggtitle("Happiness_score by Family") + xlab("Family") + ylab("Happiness_score")
```



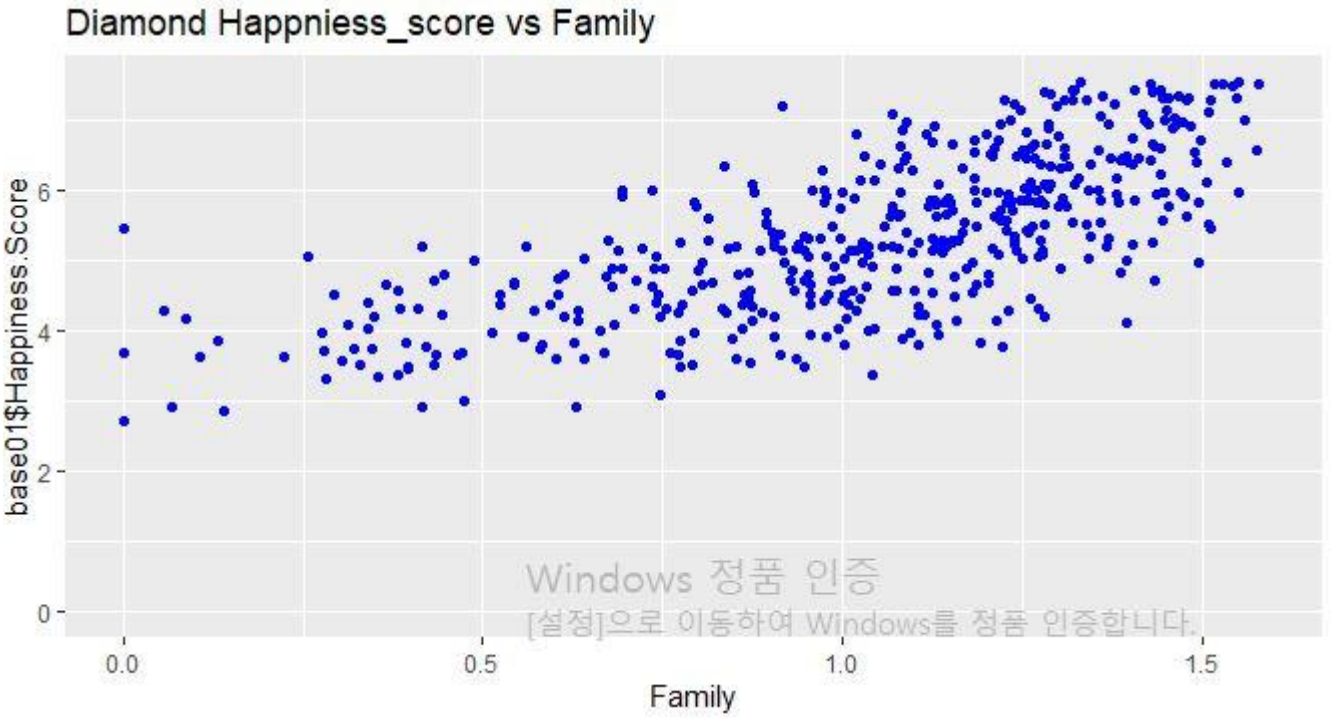
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```
ggplot(data=base01, aes(x=Family)) + geom_freqpoly(binwidth = 0.025) +
  ggtitle("Happniness_score by Family") + xlab("Family") +
  ylab("Happiness_score") + scale_x_continuous(minor_breaks = seq(0, 5.5, 0.1))
```



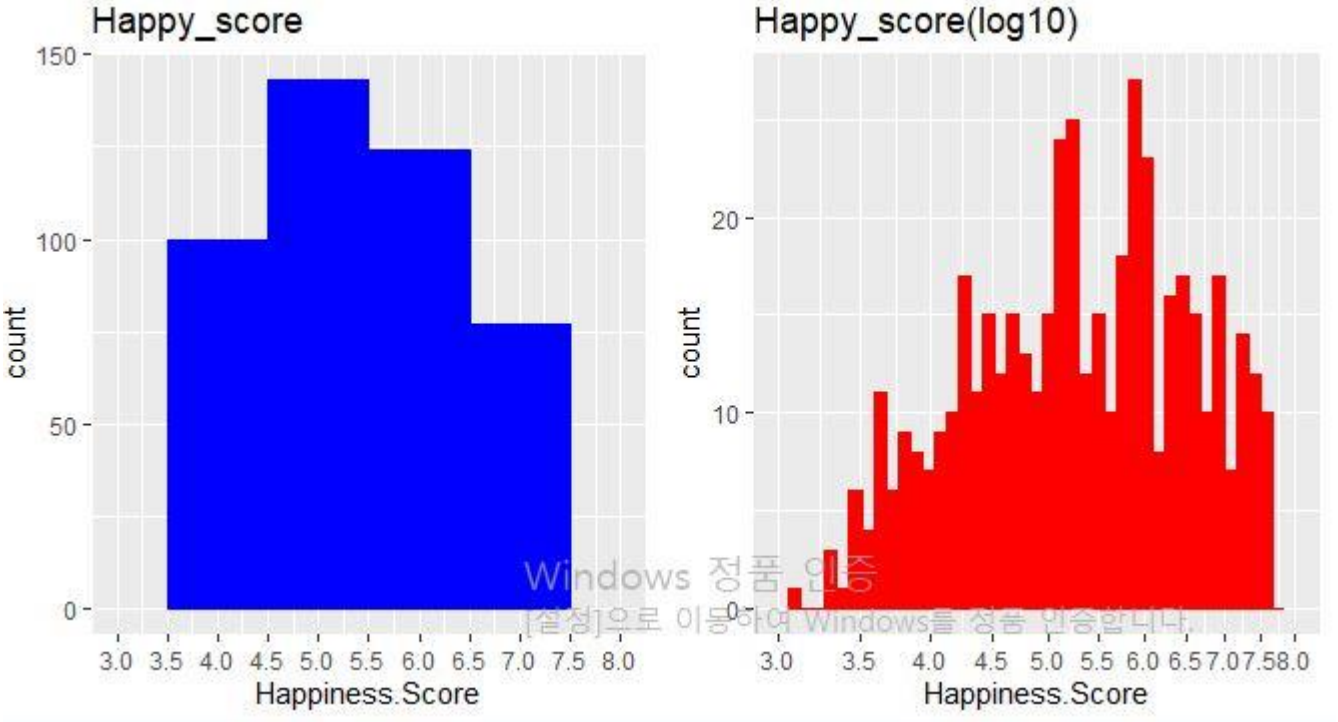
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```
ggplot(base01,aes(x=Family,y=base01$Happiness.Score))+
  geom_point(color='blue',fill='blue')+
  xlim(0,quantile(base01$Family,0.99))+
  ylim(0,quantile(base01$Happiness.Score,0.99))+
  ggtitle('Diamond Happniess_score vs Family')
```



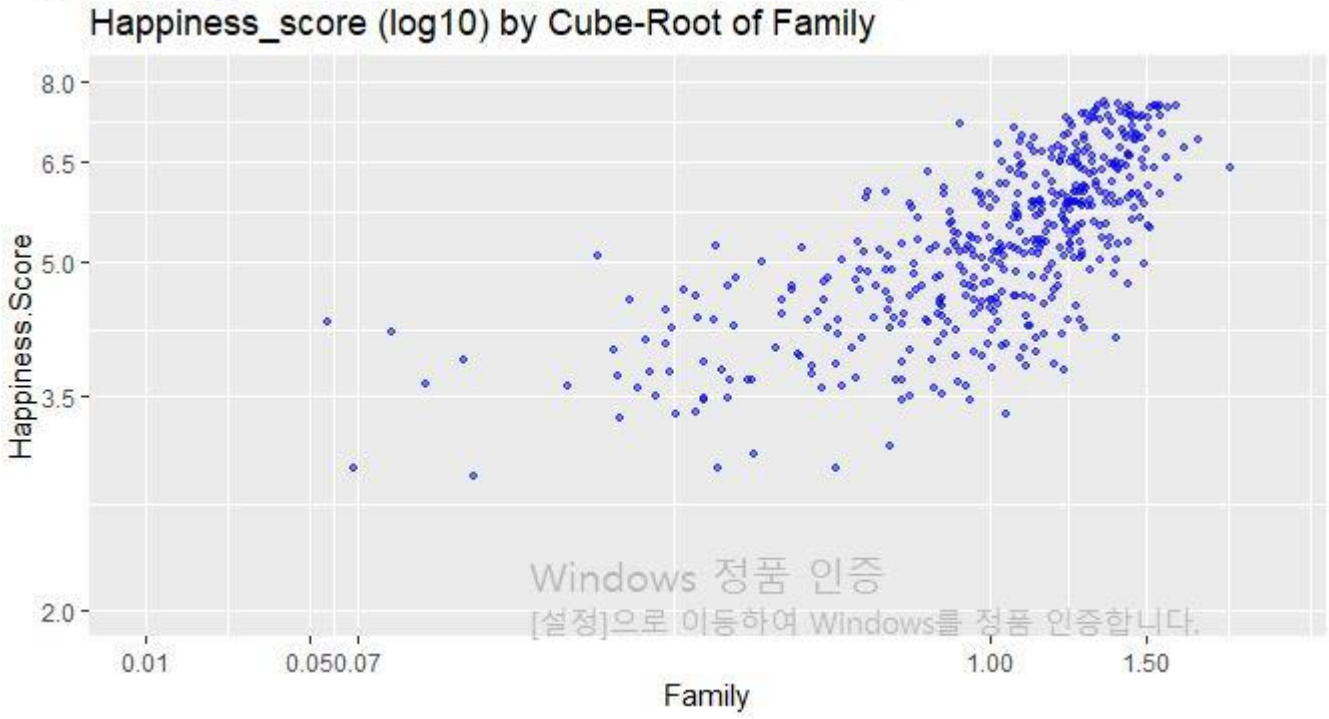
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```
library(gridExtra)
plot1 <- ggplot(base01,aes(x=Happiness.Score))+
  geom_histogram(color='blue',fill = 'blue',binwidth=1)+
  scale_x_continuous(breaks=seq(3,8,0.5),limit=c(3,8))+
  ggtitle('Happy_score')
plot2 <- ggplot(base01,aes(x=Happiness.Score))+
  geom_histogram(color='red',fill='red',binwidth=0.01)+
  scale_x_log10(breaks=seq(3,8,0.5),limit=c(3,8))+
  ggtitle('Happy_score(log10)')
grid.arrange(plot1,plot2,ncol=2)
```



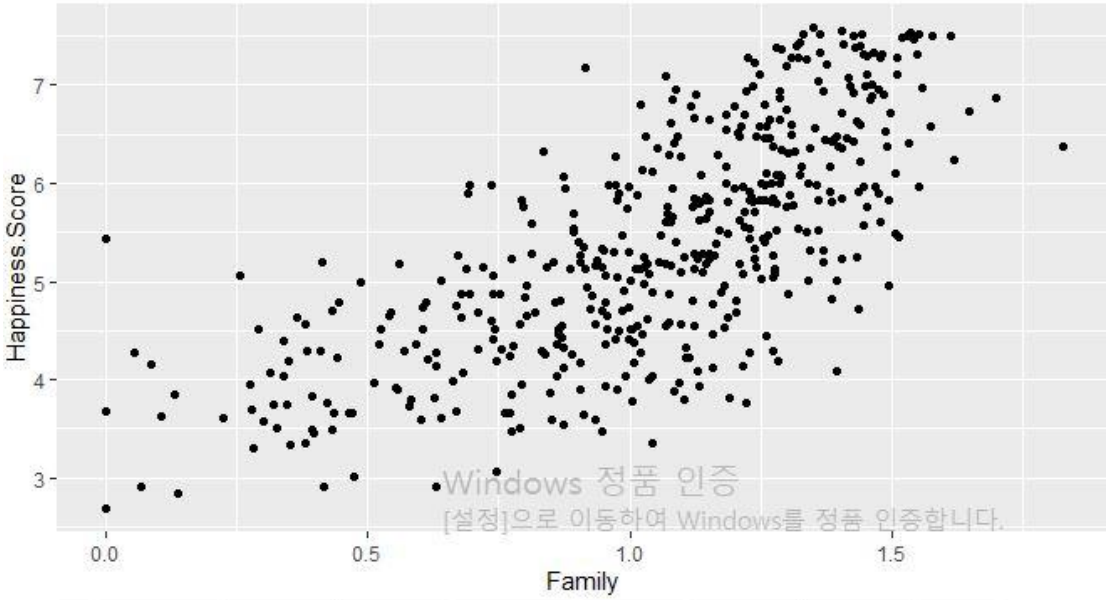
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```r
library(scales)
cuberoot_trans = function() trans_new('cuberoot',
                                      transform = function(x) x^(1/3),
                                      inverse = function(x) x^3)

## Use the cuberoot_trans function
library(ggplot2)
ggplot(aes(Family, Happiness.Score), data = base01) +
  geom_point(color='blue',fill='blue',alpha=1/2,size=1,position = 'jitter') +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.01, 2),
                     breaks = c(0.01, 0.05, 0.07, 1, 1.5)) +
  scale_y_continuous(trans = log10_trans(), limits = c(2, 8),
                     breaks = c(2, 3.5, 5, 6.5, 8)) +
  ggtitle('Happiness_score (log10) by Cube-Root of Family')
```

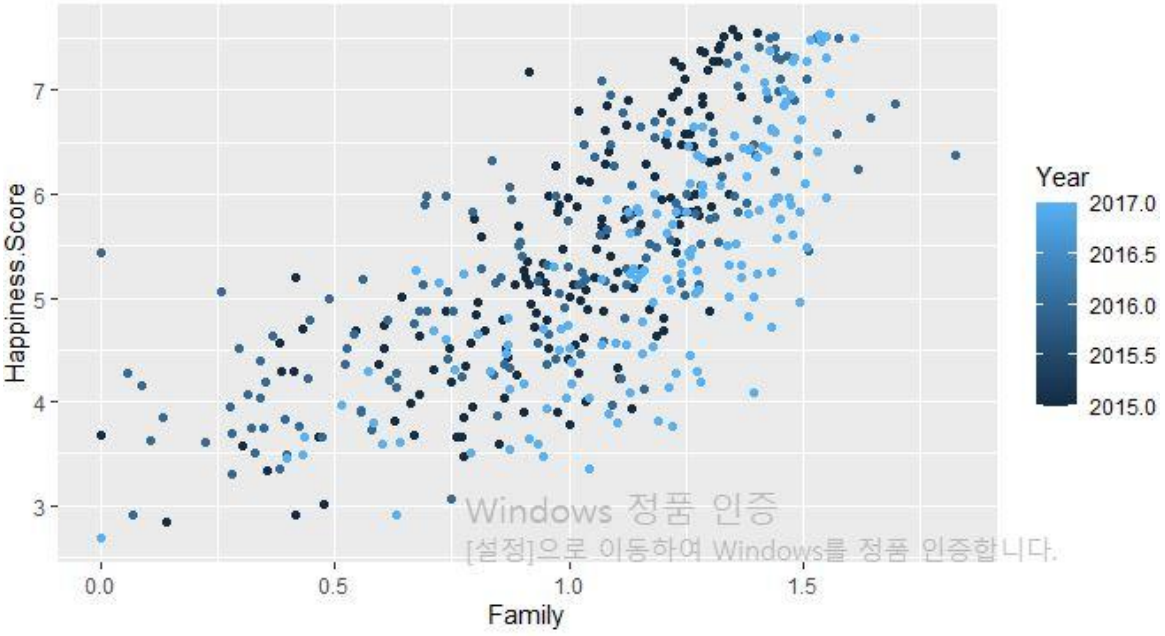


Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

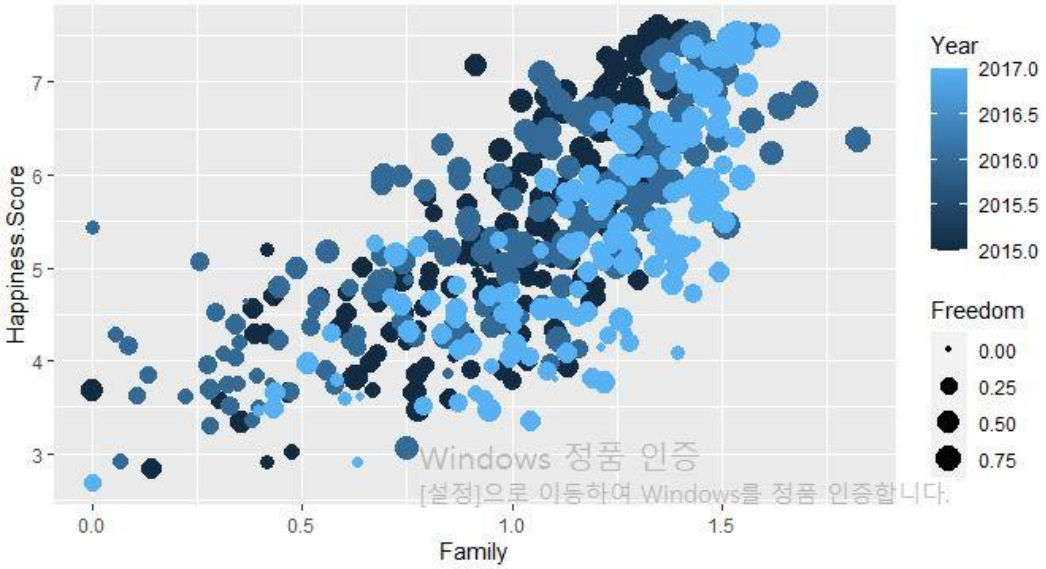
```r
##가로 축은 Family로, 세로축은 Happiness.Score로 설정, geom_point로 설정.
ggplot(base01, aes(Family, Happiness.Score))  + geom_point()

##색상추가
ggplot(base01, aes(Family, Happiness.Score))  + geom_point(aes(colour = Year))

##point의 크기는 꽃잎의  넓이(Happiness.Rank)에 따라 설정
ggplot(base01, aes(Family, Happiness.Score))  + geom_point(aes(colour = Year, size=Freedom))


##중복되어있는 점(겹쳐있는 점)을 표현
ggplot(base01, aes(Family, Happiness.Score))  + geom_point(aes(colour = Year, size=Happiness.Score), alpha=I(0.47))
```

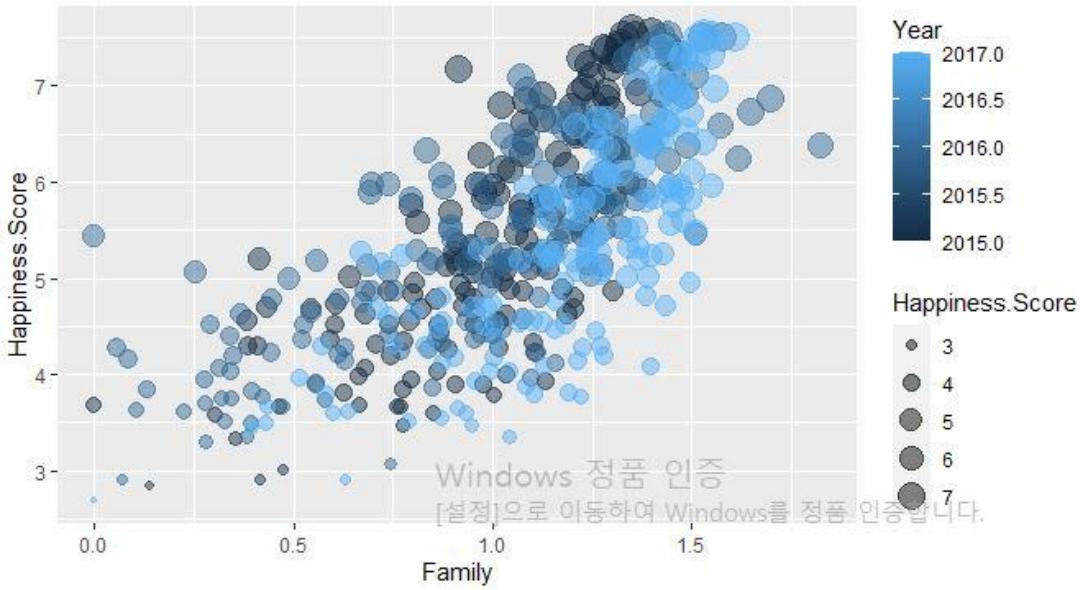


Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

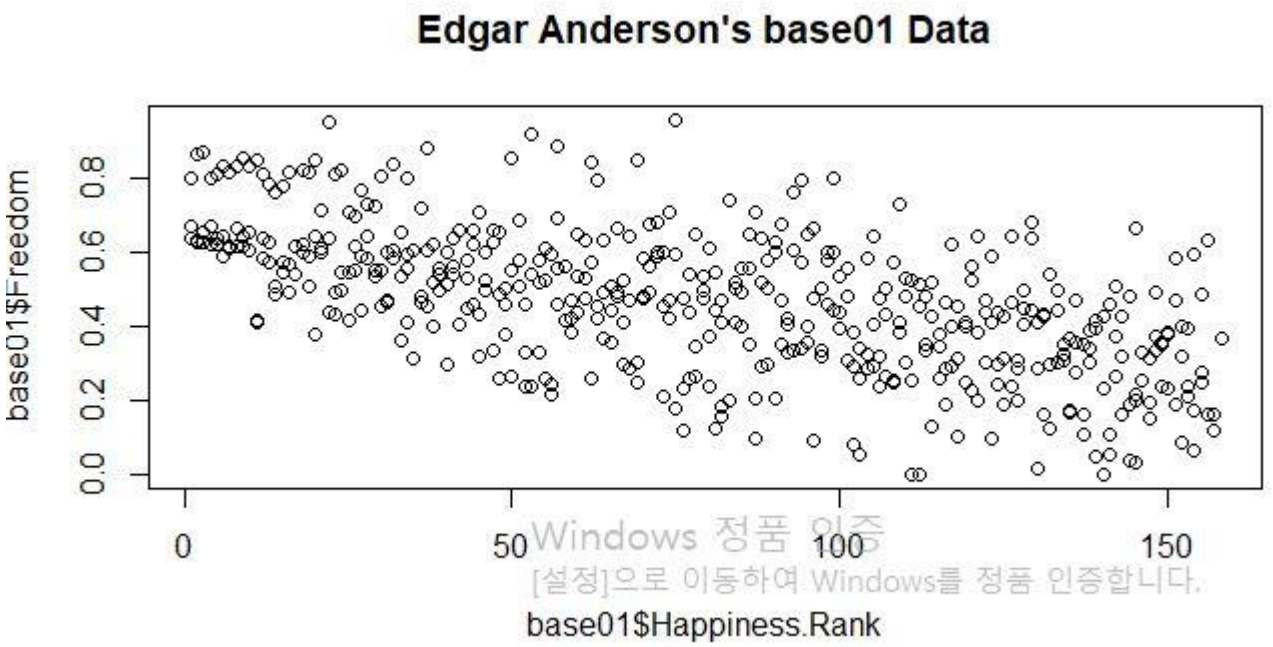


Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

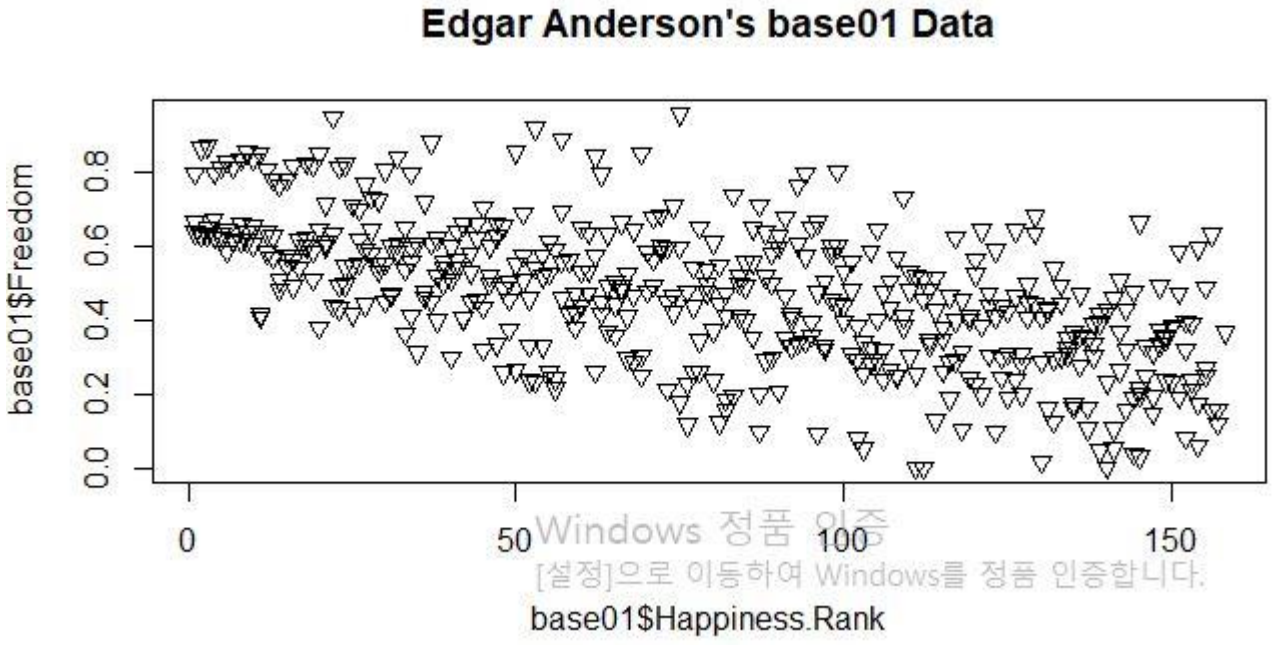

Happiness.Score
Family
Year
2017.0
2016.5
2016.0
2015.5
2015.0
Freedom
0.00
0.25
0.50
0.75
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.


Happiness.Score
Family
Year
2017.0
2016.5
2016.0
2015.5
2015.0
Happiness.Score
3
4
5
6
7
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

```r
##Simple Scatter Plots
plot(base01$Happiness.Rank, base01$Freedom, main="Edgar Anderson's base01 Data")


plot(base01$Happiness.Rank, base01$Freedom, pch=c(21,22,23)[unclass(base01$Year)],
     main="Edgar Anderson's base01 Data")

plot(base01$Happiness.Rank, base01$Freedom, pch=25, bg=c("red","green3","blue")
     [unclass(base01$Year)], main="Edgar Anderson's base01 Data")
```



Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.



Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.

3. 결론

주어진 데이터를 시각화한 결과, 행복 점수(Happiness Score)는 가족(Family)와 양의 상관관계를 가지고 있음을 알 수 있다. 또한 행복 점수의 분포는 전반적으로 고르게 분포되어 있으나, 중위값에 대한 집중도가 좀 더 높다고 볼 수 있다.

4. 사용된 코드 일체

```
#install.packages("ggplot2")
library(ggplot2)

#base01


##가로 축은 Family 로, 세로축은 Happiness.Score 로 설정, geom_point 로 설정.
ggplot(base01, aes(Family, Happiness.Score))  + geom_point()

##색상추가
ggplot(base01, aes(Family, Happiness.Score))  + geom_point(aes(colour = Year))

##point 의 크기는 꽃잎의  넓이(Happiness.Rank)에 따라 설정
ggplot(base01, aes(Family, Happiness.Score))  + geom_point(aes(colour = Year, size=Freedom))


##중복되어있는 점(겹쳐있는 점)을 표현
ggplot(base01, aes(Family, Happiness.Score))  + geom_point(aes(colour = Year, size=Happiness.Score), alpha=I(0.47))

iris
#############################################################################################
##Simple Scatter Plots
plot(base01$Happiness.Rank, base01$Freedom, main="Edgar Anderson's base01 Data")


plot(base01$Happiness.Rank, base01$Freedom, pch=c(21,22,23)[unclass(base01$Year)],
    main="Edgar Anderson's base01 Data")

plot(base01$Happiness.Rank, base01$Freedom, pch=25, bg=c("red","green3","blue")
    [unclass(base01$Year)], main="Edgar Anderson's base01 Data")

##Draftsman's or Pairs Scatter Plots
pairs(base01[1:4], main = "Edgar Anderson's base01 Data", pch = 21, bg = c("red", "green3", "blue")
     [unclass(base01$Year)])

panel.pearson <- function(x, y, ...) {
  horizontal <- (par("usr")[1] + par("usr")[2]) / 2;
  vertical <- (par("usr")[3] + par("usr")[4]) / 2;
  text(horizontal, vertical, format(abs(cor(x,y)), digits=2))
}
pairs(base01[1:4], main = "Edgar Anderson's base01 Data", pch = 23,
     bg = c("red","green3","blue")
     [unclass(base01$Year)], upper.panel=panel.pearson)

pairs(base01[1:4], main = "Anderson's base01 Data -- 3 Year",
     pch = 21, bg = c("red", "green3", "blue")
     [unclass(base01$Year)], lower.panel=NULL,
     labels=c("SL","SW","PL","PW"), font.labels=2, cex.labels=4.5)


#########################################################################################################################

#install.packages("gridExtra")
library(gridExtra)
#install.packages("grid")
library(grid)
#install.packages("plyr")
library(plyr)


## First let's get a random sampling of the data
base01[sample(nrow(base01),10),]


## Density & Frequency analysis with the Histogram,

## Family
HisSl <- ggplot(data=base01, aes(x=Family))+
```

```
  geom_histogram(binwidth=0.2, color="black", aes(fill=Year)) +
  xlab("Sepal Length (cm)") +
  ylab("Frequency") +
  theme(legend.position="none")+
  ggtitle("Histogram of Family")+
  geom_vline(data=base01, aes(xintercept = mean(Family)),linetype="dashed",color="grey")


## Happy score
HistSw <- ggplot(data=base01, aes(x=Happiness.Score)) +
  geom_histogram(binwidth=0.2, color="black", aes(fill=Year)) +
  xlab("Sepal Width (cm)") +
  ylab("Frequency") +
  theme(legend.position="none")+
  ggtitle("Histogram of happy score")+
  geom_vline(data=base01, aes(xintercept = mean(Happiness.Score)),linetype="dashed",color="grey")


## Freedom
HistPl <- ggplot(data=base01, aes(x=Happiness.Rank))+
  geom_histogram(binwidth=0.2, color="black", aes(fill=Year)) +
  xlab("Petal Length (cm)") +
  ylab("Frequency") +
  theme(legend.position="none")+
  ggtitle("Histogram of Petal Length")+
  geom_vline(data=base01, aes(xintercept = mean(Happiness.Rank)),
          linetype="dashed",color="grey")


## happiness rank
HistPw <- ggplot(data=base01, aes(x=Freedom))+
  geom_histogram(binwidth=0.2, color="black", aes(fill=Year)) +
  xlab("happiness rank") +
  ylab("Frequency") +
  theme(legend.position="right" )+
  ggtitle("Histogram of happy rank")+
  geom_vline(data=base01, aes(xintercept = mean(Freedom)),linetype="dashed",color="grey")


## Plot all visualizations
grid.arrange(HisSl + ggtitle(""),
          HistSw + ggtitle(""),
          HistPl + ggtitle(""),
          HistPw  + ggtitle(""),
          nrow = 2,
          top = textGrob("base01 Frequency Histogram",
                  gp=gpar(fontsize=15))
)



## Notice the shape of the data, most attributes exhibit a normal distribution.
## You can see the measurements of very small flowers in the Petal width and length column.

## We can review the density distribution of each attribute broken down by class value.
## Like the scatterplot matrix, the density plot by class can help see the separation of classes.
## It can also help to understand the overlap in class values for an attribute.

DhistPl <- ggplot(base01, aes(x=Happiness.Rank, colour=Year, fill=Year)) +
  geom_density(alpha=.3) +
  geom_vline(aes(xintercept=mean(Happiness.Rank),  colour=Year),linetype="dashed",color="grey", size=1)+
  xlab("Freedom") +
  ylab("Density")+
  theme(legend.position="none")

DhistPw <- ggplot(base01, aes(x=Freedom, colour=Year, fill=Year)) +
  geom_density(alpha=.3) +
  geom_vline(aes(xintercept=mean(Freedom),  colour=Year),linetype="dashed",color="grey", size=1)+
  xlab("happy score") +
  ylab("Density")

DhistSw <- ggplot(base01, aes(x=Happiness.Score, colour=Year, fill=Year)) +
  geom_density(alpha=.3) +
  geom_vline(aes(xintercept=mean(Happiness.Score),  colour=Year), linetype="dashed",color="grey", size=1)+
```

```
  xlab("Happiness score") +
  ylab("Density")+
  theme(legend.position="none")

DhistSl <- ggplot(base01, aes(x=Family, colour=Year, fill=Year)) +
  geom_density(alpha=.3) +
  geom_vline(aes(xintercept=mean(Family),  colour=Year),linetype="dashed", color="grey", size=1)+
  xlab("Family") +
  ylab("Density")+
  theme(legend.position="none")

## Plot all density visualizations
grid.arrange(DhistSl + ggtitle(""),
             DhistSw  + ggtitle(""),
             DhistPl + ggtitle(""),
             DhistPw  + ggtitle(""),
             nrow = 2,
             top = textGrob("base01 Density Plot",
                            gp=gpar(fontsize=15))
)

## Next with the bloxplot we will identify some outliers. As you can see some classes
## do not overlap at all (e.g. Petal Length)
## where as with other attributes there are hard to tease apart (Sepal Width).

ggplot(base01, aes(Year, Happiness.Rank, fill=Year)) +
  geom_boxplot()+
  scale_y_continuous("Freedom", breaks= seq(0,30, by=.5))+
  labs(title = "base01 Freedom Box Plot", x = "Year")



## Let's plot all the variables in a single visualization that will contain
## all the boxplots

BpSl <- ggplot(base01, aes(Year, Family, fill=Year)) +
  geom_boxplot()+
  scale_y_continuous("Family", breaks= seq(0,30, by=.5))+
  theme(legend.position="none")

BpSw <-  ggplot(base01, aes(Year, Happiness.Score, fill=Year)) +
  geom_boxplot()+
  scale_y_continuous("Happiness score", breaks= seq(0,30, by=.5))+
  theme(legend.position="none")

BpPl <- ggplot(base01, aes(Year, Happiness.Rank, fill=Year)) +
  geom_boxplot()+
  scale_y_continuous("Happiness rank", breaks= seq(0,30, by=.5))+
  theme(legend.position="none")

BpPw <-  ggplot(base01, aes(Year, Freedom, fill=Year)) +
  geom_boxplot()+
  scale_y_continuous("Freedom", breaks= seq(0,30, by=.5))+
  labs(title = "base01 Box Plot", x = "Year")

## Plot all visualizations
grid.arrange(BpSl  + ggtitle(""),
             BpSw  + ggtitle(""),
             BpPl + ggtitle(""),
             BpPw + ggtitle(""),
             nrow = 2,
             top = textGrob("Sepal and Petal Box Plot",
                            gp=gpar(fontsize=15))
)


## You can also visualize the data using the violin plots. They are similar to
## the Box Plots but they
## show the number of points at a particular value by the width of the shapes.
## The can also include the marker for the median and a box for the interquartile range.

VpSl <-  ggplot(base01, aes(Year, Family, fill=Year)) +
  geom_violin(aes(color = Year), trim = T)+
  scale_y_continuous("Family", breaks= seq(0,30, by=.5))+
```

```
  geom_boxplot(width=0.1)+
  theme(legend.position="none")

VpSw <-  ggplot(base01, aes(Year, Happiness.Score, fill=Year)) +
  geom_violin(aes(color = Year), trim = T)+
  scale_y_continuous("Happiness.Score", breaks= seq(0,30, by=.5))+
  geom_boxplot(width=0.1)+
  theme(legend.position="none")

VpPl <-  ggplot(base01, aes(Year, Happiness.Rank, fill=Year)) +
  geom_violin(aes(color = Year), trim = T)+
  scale_y_continuous("Happiness.Rank", breaks= seq(0,30, by=.5))+
  geom_boxplot(width=0.1)+
  theme(legend.position="none")

VpPw <-  ggplot(base01, aes(Year, Freedom, fill=Year)) +
  geom_violin(aes(color = Year), trim = T)+
  scale_y_continuous("Freedom", breaks= seq(0,30, by=.5))+
  geom_boxplot(width=0.1)+
  labs(title = "base01 Box Plot", x = "Year")

## Plot all visualizations
grid.arrange(VpSl  + ggtitle(""),
             VpSw  + ggtitle(""),
             VpPl + ggtitle(""),
             VpPw + ggtitle(""),
             nrow = 2,
             top = textGrob("Violin Plot",
                            gp=gpar(fontsize=15))
)


## Now let's create a scatterplot of petal lengths versus petal widths with the color & shape by Year.
## There is also a regression line with a 95% confidence band.
## Notice the petal length of the setosa is clearly a differenciated cluster
## so it will be a good predictor for ML.

ggplot(data = base01, aes(x = Happiness.Rank, y = Freedom))+
  xlab("Happiness.Rank")+
  ylab("Freedom") +
  geom_point(aes(color = Year,shape=Year))+
  geom_smooth(method='lm')+
  ggtitle("Happiness.Rank vs Freedom")


## Here is a similar plot with more details on the regression line.
#install.packages("car")
library(car)
scatterplot(base01$Happiness.Rank,base01$Freedom)


## Now check the Sepal Length vs Width. Notice the sepal of the Virginica and
## Versicolor Year is more mixed, this feature might not be a good predictor.
ggplot(data=base01, aes(x = Family, y = Happiness.Score)) +
  geom_point(aes(color=Year, shape=Year)) +
  xlab("Family") +
  ylab("Happiness.Score") +
  ggtitle("Family vs Happiness.Score")


## Based on all the plots we have done we can see there is certain correlation.
## Let's take a look at the pairwise correlation numerical values to
## ascertain the relationships in more detail.

#install.packages("GGally")
library(GGally)
ggpairs(data = base01[1:4],
        title = "base01 Correlation Plot",
        upper = list(continuous = wrap("cor", size = 5)),
        lower = list(continuous = "smooth")
)

## The examination of the plot revelas a strong correlation between
## the variables Petal Width and the Petal Length (96%) as well as
```

The Sepal Length and Petal Length (87%).
The heatmat is another useful exploratory plot. It is like
a two dimentional histogram and it works by using color
intensity to represent how large the data value is.
The brighter the color the larger the value.
For example the color white represents the largest value
while the red represent the smallest one
with different colors which represent the different values in betwwen.

Let's Ccreate the matrix and transpose it before using it for the heatmap to
ensure the columns corresponds to the features and the rows correspond to
the observations.

```
base01Matix <- as.matrix(base01[1:150, 1:4])
base01TransposedMatrix <- t(base01Matix)[,nrow(base01Matix):1]
image(1:4, 1:150, base01TransposedMatrix)
```

```
base01 <- read.csv(file="C:\\Users\\PC\\Desktop\\base01.csv",head=T)
##Now let us test the power of RStudio with the base01 set.
##This is a good example of how powerful R can be running large sets of data.

library(ggplot2)

summary(base01)

dim(base01)

##We can answer how many observations, how many variables (only 10),
##and how many of those variables are ordered factor (only 3).
##We can also learn about the happieest country by using the command ?base01.

##Let's test the power now of GGplot2 with a simple histogram of diamond Happniess.Scores.

ggplot(data=base01) +
  geom_histogram(binwidth=0.1, aes(x=base01$Happiness.Score)) +
  ggtitle("Happiness_Score Distribution") +
  xlab("Happiness_score") + ylab("Frequency") + theme_minimal()


##This is a long tail distribution, with a high concentration of observations below
##We can get mean and median by running simple R commands:

mean(base01$Happiness.Score)

median(base01$Happiness.Score)

##Supposed we want to know the following:
##How many score less than 3?
##How many score less than 5?
##How many score equal to 7 or more?

sum(base01$Happiness.Score < 3)

sum(base01$Happiness.Score < 5)

sum(base01$Happiness.Score >= 7)

##Let's get closer to that peak
##There is a very visible peak in the histogram.
##It comes very early in the analysis #of the chart.
##Let's get very close to observe the higher than expected frequency.

##This is a first attempt:

ggplot(data=base01) +
  geom_histogram(binwidth=0.1, aes(x=base01$Happiness.Score)) +
  ggtitle("Happy_score Distribution") +
  xlab("Happiness_score Binwidth 0.1") +
  ylab("Frequency") + theme_minimal() + xlim(0,10)

##Another drop in frequency, this time to no more than 1,500.
##That is the reason Data Scientist are actual people and RStudio and
##R haven't taken over!
##Bin selection will play a significant role in visualizations,
##with a possible change in frequency readouts and shape of the curve or function.
##UDACITY thinks that asking for five histograms, broken down by cut,
##was going to be a challenge.
##Sadly for them, and luckily for aspiring Data Scientits like us,
##this is not the case with R and ggplot2.
##Using the facet_wrap(~cut) command is almost too easy to produce the graphs:



ggplot(data=base01) +
  geom_histogram(binwidth=100, aes(x=base01$Happiness.Score)) +
  ggtitle("Happiness_score by Cut") +
  xlab("Happiess_score") + ylab("Frequency") + theme_minimal() + facet_wrap(~Freedom)


##What if we want to see the cut for the highest Happniess.Scored diamond?
##This one is a little tricky, but the easiest way is to subset the base01 data
```

```
##using as the filter the logical expression Happniess.Score == max(Happniess.Score).
##It is unusual but it works.

subset(base01, Freedom == max(Freedom))

##And the answer is Premium cut for a diamond of 2.29 Family that sold at U$18,823!
##Getting the cut of the lowest Happniess.Scored diamond is a similar task.

subset(base01, Freedom == min(Freedom))

##Looks like we have a tie between to units, both sold at U$326,
##one of 0.23 Family and Ideal cut, and another of 0.21 Familys and Premium cut.

##The last question is which cut has the lowest median Happniess.Score.
##This one is VERY tricky, since it involves lots of query in the data.
##The long and easy way is to use the which command to subset data vectors and
##then get the median of those:

a = base01[which(base01$cut == "Fair"),]
b = base01[which(base01$cut == "Good"),]
c = base01[which(base01$cut == "Very Good"),]
d = base01[which(base01$cut == "Premium"),]
e = base01[which(base01$cut == "Ideal"),]

median(a$Happniess.Score)

median(b$Happniess.Score)

median(c$Happniess.Score)

median(d$Happniess.Score)

median(e$Happniess.Score)

##Going back to our grid histogram, let's get different frequency scales (the y axis)
##to accomodate for specific patterns.
##It's harder to see patterns if all five charts use the same scale
##for comparison and patterns become lost in the translation.
##The command is very easy.

ggplot(data=base01) +
  geom_histogram(binwidth=100, aes(x=base01$Happniess.Score)) +
  ggtitle("Diamond Happniess.Score Distribution by Cut") +
  xlab("Diamond Happniess.Score U$") + ylab("Frequency") +
  theme_minimal() + facet_wrap(~cut, scales="free_y")


##You can now see how different graphs have different Y scales.
##For example Fair cut base01 have a Y scale maximizing at 600,
##while Ideal base01 have a Y scale topping at 2,500.
##This is just the effect of using scale="free_y" in the facet_wrap layer.
##Let's work now on plotting Happniess.Score per Family of different cuts.

ggplot(data=base01) +
  geom_histogram(binwidth=50, aes(x=base01$Happniess.Score/base01$Family)) +
  ggtitle("Diamond Happniess.Score per Family Distribution by Cut") +
  xlab("Diamond Happniess.Score per Family U$") + ylab("Frequency") +
  theme_minimal() + facet_wrap(~cut)


##UDACITY also asks for log10 scale.
##Let's work now on plotting Happniess.Score per Family of different cuts and using Log10.

ggplot(data=base01) +
  geom_histogram(binwidth=0.01, aes(x=base01$Happniess.Score/base01$Family)) +
  ggtitle("Diamond Happniess.Score per Family Distribution by Cut") +
  xlab("Diamond Happniess.Score per Family U$ - LOG 10 Scale") +
  ylab("Frequency") + theme_minimal() + facet_wrap(~cut) + scale_x_log10()


##It's easier to see how Happniess.Score per Family raises with cut quality.
##Notice how I change the bin size to make sense on Log10 scale (else it look terrible…)

##A loook into boxplots
```

```
##The next assignmen is about investigating the Happniess.Score of base01 using box plots,
##numerical summaries, and one of the following categorical variables:
##cut, clarity, or color.

ggplot(base01, aes(factor(cut), Happniess.Score, fill=cut)) +
  geom_boxplot() + ggtitle("Diamond Happniess.Score according Cut") +
  xlab("Type of Cut") + ylab("Diamond Happniess.Score U$") + coord_cartesian(ylim=c(0,7500))


##It’s hard to draw conclusions;
##it seems that cut of all types carry Happniess.Scores of all types,
##not really a way to determine how good or expensive a diamond is.
##I suspect people never take a magnifying glass and really look at
##the cut when they choose a diamond unless they are true proffesionals.
##Let’s see the same chart using clarity

ggplot(base01, aes(factor(clarity), Happniess.Score, fill=clarity)) +
  geom_boxplot() + ggtitle("Diamond Happniess.Score according Clarity") +
  xlab("Clarity") + ylab("Diamond Happniess.Score U$") + coord_cartesian(ylim=c(0,7500))


##OK! This is more meaningful, we even get a few outliers
##(I limited the number of outliers by using xlim=c(0,7500)) or no more than
##U$7,500 dollars. So clarity is a meaningful variable where cut is not.
##We can conclude people see and appreciate more shiny things?

##The next part is answering four questions about color and Happniess.Score range inside
##a IQR range. These are the following.

##What is the Happniess.Score range for the middle 50% of base01 with color D (best color)?
##What is the Happniess.Score range for the middle 50% of base01 with color J (worst color)?
##What is the IQR for base01 with the best color (color D)?
##What is the IQR for base01 with the worst color (color J)?
##To use some instructor notes,we can use the function IQR() to find
##the interquartile range. Pass it a subset of the base01 data frame.
##For example IQR(subset(base01, Happniess.Score <1000)$Happniess.Score) subset returns
##a data frame (so we need to use $Happniess.Score on the end to access that variable.)

##However for some reason this method did not work for me,
##so I decided to simply create a subset and take it from there.

d = subset(base01, base01$color == 'D')
j = subset(base01, base01$color == 'J')

summary(d)

IQR(d$Happniess.Score)

summary(j)

IQR(j$Happniess.Score)


##This is very strange (the results are fine),
##people actually pay more on average for a J color base01 (worst color) than
##for a D color diamond (best color)!

##How about we investigate the Happniess.Score per Family of base01 across
##the different colors of base01 using boxplots?
##This sounds like a big effort but it’s actually just a little change of code.

ggplot(base01, aes(factor(color), (Happniess.Score/Family), fill=color)) +
  geom_boxplot() + ggtitle("Diamond Happniess.Score per Family according Color") +
  xlab("Color") + ylab("Diamond Happniess.Score per Family U$")


##**Now that is a big quantity of outliers for color D.
##I can see where people spend their money, not on the under U$7,500 range,
##but rather on the most unique rocks.
##We can limit the Happniess.Score range to under U$7,500 and see a smaller picture.

ggplot(base01, aes(factor(color), (Happniess.Score/Family), fill=color)) +
  geom_boxplot() + ggtitle("Diamond Happniess.Score per Family according Color")+
  xlab("Color") + ylab("Diamond Happniess.Score per Family U$") +
```

```
  coord_cartesian(ylim=c(0,7500))


##How strange, under the U$7,500 range the Happniess.Score per Family of base01 is
##actually more expensive on color G (medium quality on the scale) than
##any other color.

##The more I study these charts, the more I see that people know very little
##about base01, and pay way more for medium-quality rocks
##because cut, color and clarity are still very hard to define and detect
##for the untrained eye.

##For the next question, we will investigate Family weight using
##a frequency polygon chart.
##I have not used many of this type of chart, so let's see what we get.

ggplot(data=base01, aes(x=Family)) + geom_freqpoly() +
  ggtitle("Happiness_score by Family") + xlab("Family") + ylab("Happiness_score")

## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.

##RStudio complains about binwidth not being accurate.
##Let's adjust (quick Google of where to change binwidth, I keep forgetting…)

ggplot(data=base01, aes(x=Family)) + geom_freqpoly(binwidth = 0.025) +
  ggtitle("Happniness_score by Family") + xlab("Family") +
  ylab("Happiness_score") + scale_x_continuous(minor_breaks = seq(0, 5.5, 0.1))


##Mind you, this is counting occurrences and not Happniess.Score per Family.
##I tried that and got a very criptice message Error :
## Mapping a variable to y and also using stat="bin". With stat="bin",
##it will attempt to set the y value to the count of cases in each group.
##This can result in unexpected behavior and will not be allowed in a future version
##of ggplot2. If you want y to represent counts of cases,
##use stat="bin" and don't map a variable to y.
##If you want y to represent values in the data, use stat="identity".
##See ?geom_bar for examples. (Defunct; last used in version 0.9.2)
##But as you can see, bigger rocks are hard to come by.
##Well, this has been probably my biggest work to-date with R Markdown.
##What can I say? I love it!



######################################################################
######################################################################


##1. Let's consider the Happniess.Score of a diamond and it's Family weight.
##   Create a scatterplot of Happniess.Score (y) vs Family weight (x),
##   and limit the x-axis and y-axis to omit the top 1% of values.

ggplot(base01,aes(x=Family,y=base01$Happiness.Score))+
  geom_point(color='blue',fill='blue')+
  xlim(0,quantile(base01$Family,0.99))+
  ylim(0,quantile(base01$Happiness.Score,0.99))+
  ggtitle('Diamond Happniess_score vs Family')

## Notice it looks like the Happniess.Score increases kind of exponentially against Family,
##       but it gets diverse when the Family increases.

## 2. Let's sample 10,000 from base01 data set and then use
##    ggpairs to generate the pair-wise variables relationship

install.packages("GGally")
library(GGally)
set.seed(20180088)
base01_samp <- base01[sample(1:length(base01$Happiness.Score),1000)]
ggpairs(base01_samp)

ggpairs(base01_samp, params = c(shape=I('.'),outlier.shape=I('.')))


## We notice the followings:
```

```
## - diamond Happniess.Score is almost linearly correlated with x, y, z and Family;
##    These are the critical factors driving Happniess.Score
## - diamond Happniess.Score seems related to cut/color/clarity
##     but is not very clear from this plot
## - diamond Happniess.Score seems not directly related to depth and table


## 3. Create two histograms of the Happniess.Score variable, one is of Happniess.Score and
##    the 2nd is log10 transformation of Happniess.Score, and place them side by side
##    on one output image.

library(gridExtra)
plot1 <- ggplot(base01,aes(x=Happiness.Score))+
  geom_histogram(color='blue',fill = 'blue',binwidth=1)+
  scale_x_continuous(breaks=seq(3,8,0.5),limit=c(3,8))+
  ggtitle('Happy_score')
plot2 <- ggplot(base01,aes(x=Happiness.Score))+
  geom_histogram(color='red',fill='red',binwidth=0.01)+
  scale_x_log10(breaks=seq(3,8,0.5),limit=c(3,8))+
  ggtitle('Happy_score(log10)')
grid.arrange(plot1,plot2,ncol=2)

##  It's obviously that the Happniess.Score histogram is skewed to the left side
##  while the log10(Happniess.Score) tends to bell curve distributed.
##  Also, the two peaks in the log10(Happniess.Score) plot coincides
##  with the 1st and 3rd quantile of Happniess.Score.

## 4. Create scatter plot by log10 transforming Happniess.Score on y axis
##    and cuberoot transforming Family on x axis

### Create a new function to transform the Family variable

library(scales)
cuberoot_trans = function() trans_new('cuberoot',
                                      transform = function(x) x^(1/3),
                                      inverse = function(x) x^3)

## Use the cuberoot_trans function
library(ggplot2)
ggplot(aes(Family, Happiness.Score), data = base01) +
  geom_point(color='blue',fill='blue',alpha=1/2,size=1,position = 'jitter') +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.01, 2),
                     breaks = c(0.01, 0.05, 0.07, 1, 1.5)) +
  scale_y_continuous(trans = log10_trans(), limits = c(2, 8),
                     breaks = c(2, 3.5, 5, 6.5, 8)) +
  ggtitle('Happiness_score (log10) by Cube-Root of Family')


## Now the log10(Happniess.Score) is almost linear with cuberoot of Family
## - we can move on to the modeling.

## 5. Let's see if other factors have some impact on Happniess.Score. Clarity first.

ggplot(aes(x = Family, y = Happniess.Score), data = base01) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter',aes(color=clarity)) +
  scale_color_brewer(type = 'div',
                     guide = guide_legend(title = 'Clarity', reverse = T,
                                          override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Happniess.Score (log10) by Cube-Root of Family and Clarity')


## It's clear that Clarity factors into the diamond Happniess.Score
## - a better clarity almost always has higher Happniess.Score than lower end clarity.

## 6. Now let's see if cut has similar impact on diamond Happniess.Score

ggplot(aes(x = Family, y = Happiness.Score), data = base01) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter',aes(color=cut)) +
  scale_color_brewer(type = 'div',
                     guide = guide_legend(title = 'Cut', reverse = T,
```

```
                                override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Happniess.Score (log10) by Cube-Root of Family and Cut')

## While cut plot does not show as obvious pattern as Clarity,
## it's still clear that with the smae Family the base01 with
## the best cut are Happniess.Scored higher. Hence, I think cut should be also included
## in the Happniess.Score prediction algorithm.

## 7. Now let's see if color accounts for any variance of diamond Happniess.Score
ggplot(aes(x = Family, y = Happniess.Score), data = base01) +
  geom_point(alpha = 0.5, size = 1, position = 'jitter',aes(color=color)) +
  scale_color_brewer(type = 'div',
                     guide = guide_legend(title = 'Color', reverse = F,
                                override.aes = list(alpha = 1, size = 2))) +
  scale_x_continuous(trans = cuberoot_trans(), limits = c(0.2, 3),
                     breaks = c(0.2, 0.5, 1, 2, 3)) +
  scale_y_continuous(trans = log10_trans(), limits = c(350, 15000),
                     breaks = c(350, 1000, 5000, 10000, 15000)) +
  ggtitle('Happniess.Score (log10) by Cube-Root of Family and Color')


## This looks similar with previous Clarity plot and Color should be also
## considered as an factor for Happniess.Score.

## 8. it's time to build out the Happniess.Score prediction model!
m1 <- lm(I(log10(Happiness.Score)) ~ I(Family^(1/3)), data = base01)
m1
m2 <- update(m1,~ . +Family)
m2
m3 <- update(m2,~ . +cut)
m3
m4 <- update(m3,~ . +color)
m4
m5 <- update(m4,~ . +clarity)
m5
##mtable(m1,m2,m3,m4,m5)




## the linear model for the diamond Happniess.Score is:
## Log(Happniess.Score)=
##  0.18+3.97Family1/3???0.474Family+pc5???cutcoef+pc7???colorcoef+pc8???claritycoef
## where pc5, pc7 and pc8 are polynomial contrast with n=5, 7, 8, respectively.
## You can check the detail by applying contr.poly(n).

## 9. Pick a diamond and predict the Happniess.Score by using our new model.
##   I randomly take this diamond in this example:

##       Family       cut color clarity depth table Happniess.Score    x    y    z
## 13696     1 Very Good     G     VS2  61.8    59  5600 6.29 6.37 3.91

## Here is the code to get the predicted (fitted) Happniess.Score of this diamond
## by using the linear model:

thisDiamond <- data.frame(Family=1, cut='Very Good',
                          color='G',clarity='VS2')
modelEstimate <- predict(m5,newdata = thisDiamond,
                         interval = "prediction",level = .95)
10^modelEstimate

## The predicted Happniess.Score is $5,232.11 vs. actual Happniess.Score $5,600.
```