

Final Project (Group 2)

Group 2

2024-05-28

«««< HEAD - Research Question/Hypothesis: What variable in the world happiness report (family, health, trust, generosity, and economics) has the greatest effect on a nation's happiness score?
=====

- Research Question/Hypothesis: What variable in the world happiness report (family, health, trust, generosity, and economics) has the greatest effect on a nation's happiness score? »»»>
81e6af9ac23978bbafb85c9fa12c41d73c572ee5
- Hypothesis: Economics plays the largest role in a nation's happiness score.

```
library(readxl)
library(dplyr)
library(ggplot2)
library(tidyr)
```

```
data <- read_excel("2019.xls")
```

```
colnames(data)
```

```
## [1] "Overall rank"           "Country or region"
## [3] "Score"                  "GDP per capita"
## [5] "Social support"         "Healthy life expectancy"
## [7] "Freedom to make life choices" "Generosity"
## [9] "Perceptions of corruption"
```

```
library(readxl)
```

```
data <- read_excel("2019.xls")
```

```
print(colnames(data))
```

```
## [1] "Overall rank"           "Country or region"
## [3] "Score"                  "GDP per capita"
## [5] "Social support"         "Healthy life expectancy"
## [7] "Freedom to make life choices" "Generosity"
## [9] "Perceptions of corruption"
```

```
data <- data %>%
  rename(
    Economy = `GDP per capita`,
    Social = 'Social support',
    Health = `Healthy life expectancy`,
    Freedom = `Freedom to make life choices`,
    Corruption = 'Perceptions of corruption',
    Happiness_Score = `Score`
  )
print(colnames(data))
```

```
## [1] "Overall rank"      "Country or region" "Happiness_Score"
## [4] "Economy"           "Social"            "Health"
## [7] "Freedom"           "Generosity"        "Corruption"
```

```
head(
  select(data, Economy, Social, Health, Freedom, Corruption, Happiness_Score)
)
```

Economy	Social	Health	Freedom	Corruption	Happiness_Score
1.340	1.587	0.986	0.596	0.393	7.769
1.383	1.573	0.996	0.592	0.410	7.600
1.488	1.582	1.028	0.603	0.341	7.554
1.380	1.624	1.026	0.591	0.118	7.494
1.396	1.522	0.999	0.557	0.298	7.488
1.452	1.526	1.052	0.572	0.343	7.480

[Module 2: Junhyung Kim, Jiho Lee]

*Scatter Plot

```
qplot(x = Economy, y = Happiness_Score, data = data,
  geom = c("point", "smooth"), method = "lm") +
  labs(title =
    "Scatter Plot of Relationship Between
    Economy and Happiness Score",
    x = "Economy", y = "Happiness Score")
```

```
## Warning: 'qplot()' was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

```
## Warning in geom_point(method = "lm"): Ignoring unknown parameters: 'method'
```

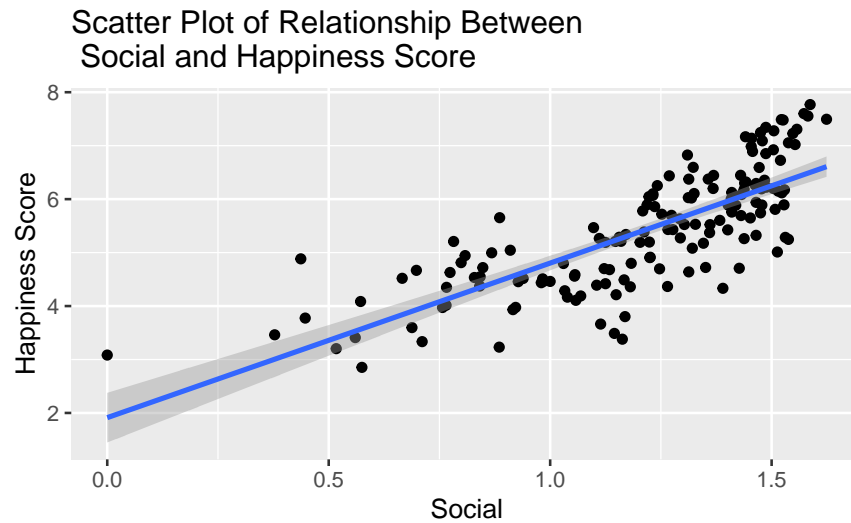
```
## 'geom_smooth()' using formula = 'y ~ x'
```



```
qplot(x= Social,y=Happiness_Score,data=data,  
geom=c("point","smooth"),method="lm")+  
labs(title =  
"Scatter Plot of Relationship Between  
Social and Happiness Score",  
x="Social",y="Happiness Score")
```

```
## Warning in geom_point(method = "lm"): Ignoring unknown parameters: 'method'
```

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



```
qplot(x= Health,y=Happiness_Score,data=data,
      geom=c("point","smooth"),method="lm")+
labs(title =
      "Scatter Plot of Relationship Between
      Health and Happiness Score",
      x="Health",y="Happiness Score")
```

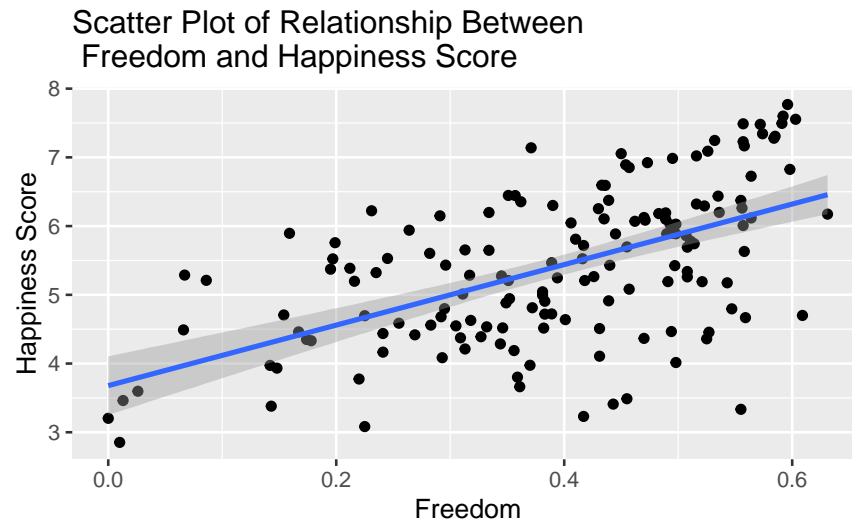
```
## Warning in geom_point(method = "lm"): Ignoring unknown parameters: 'method'
```

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



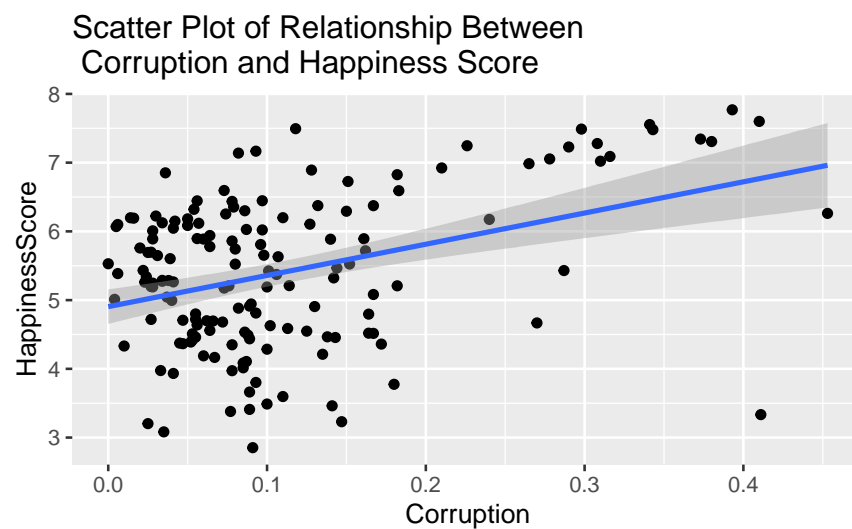
```
qplot(x= Freedom,y=Happiness_Score,data=data,
      geom=c("point","smooth"),method="lm")+
labs(title =
      "Scatter Plot of Relationship Between
      Freedom and Happiness Score",
      x="Freedom",y="Happiness Score")
```

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



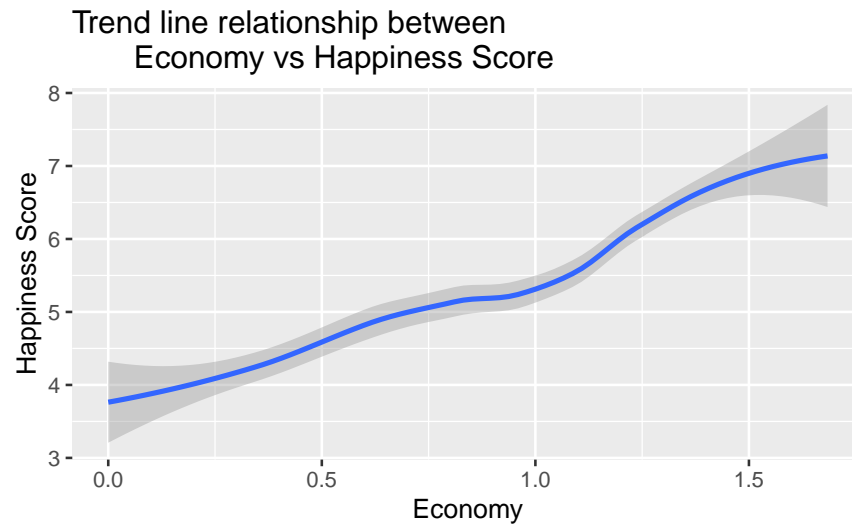
```
qplot(x= Corruption,y=Happiness_Score,data=data,
geom=c("point","smooth"),method="lm")+
labs(title ="Scatter Plot of Relationship Between
Corruption and Happiness Score",
x="Corruption",y="HappinessScore")
```

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



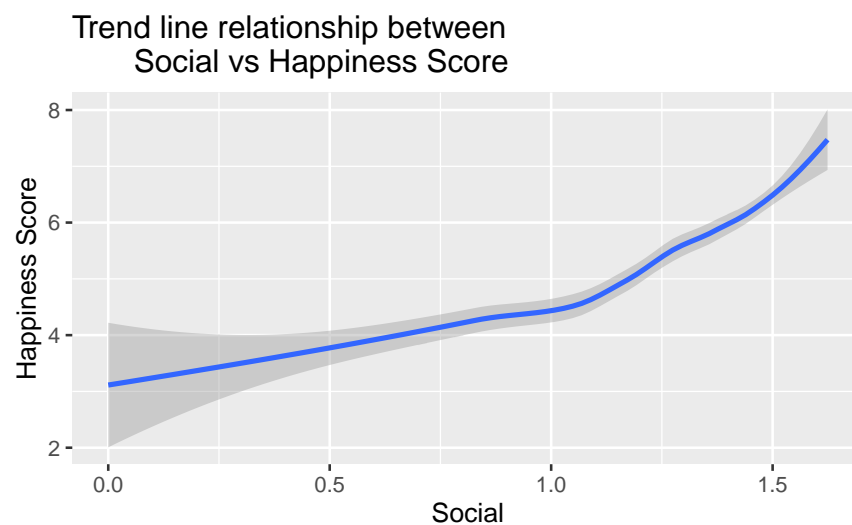
```
data %>%
  ggplot() +
  geom_smooth(mapping = aes(x = Economy, y = Happiness_Score)) +
  labs(x = "Economy", y = "Happiness Score",
  title="Trend line relationship between
  Economy vs Happiness Score")
```

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



```
data %>%  
  ggplot() +  
  geom_smooth(mapping = aes(x = Social, y = Happiness_Score)) +  
  labs(x = "Social", y = "Happiness Score",  
       title="Trend line relationship between  
       Social vs Happiness Score")
```

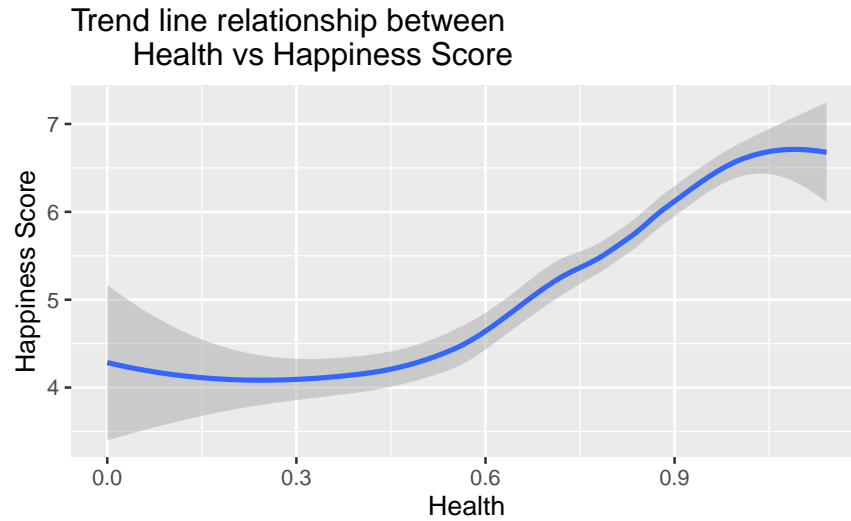
```
## 'geom_smooth()' using method = 'loess' and formula = 'y ~ x'
```



```
data %>%  
  ggplot() +  
  geom_smooth(mapping = aes(x = Health, y = Happiness_Score)) +
```

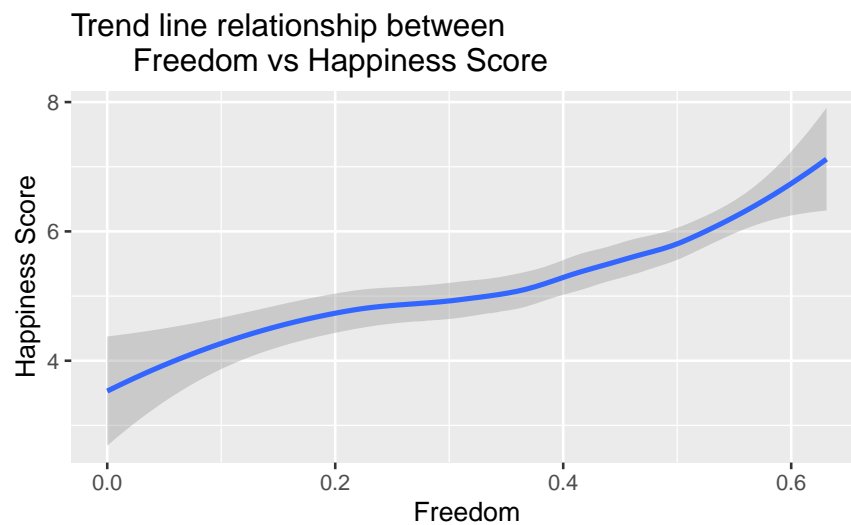
```
labs(x = "Health", y = "Happiness Score",
     title="Trend line relationship between
     Health vs Happiness Score")
```

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



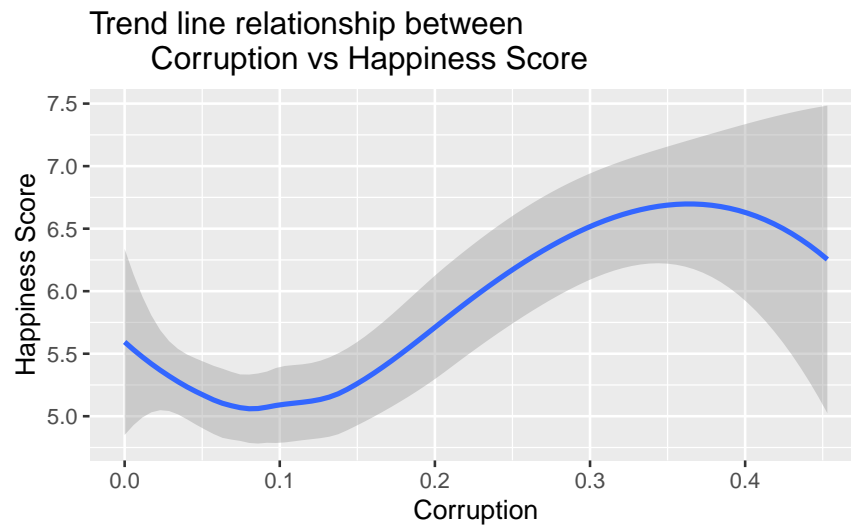
```
data %>%
  ggplot() +
  geom_smooth(mapping = aes(x = Freedom, y = Happiness_Score)) +
  labs(x = "Freedom", y = "Happiness Score",
       title="Trend line relationship between
       Freedom vs Happiness Score")
```

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



```
data %>%
  ggplot() +
  geom_smooth(mapping = aes(x = Corruption, y = Happiness_Score)) +
  labs(x = "Corruption", y = "Happiness Score",
       title="Trend line relationship between
              Corruption vs Happiness Score")
```

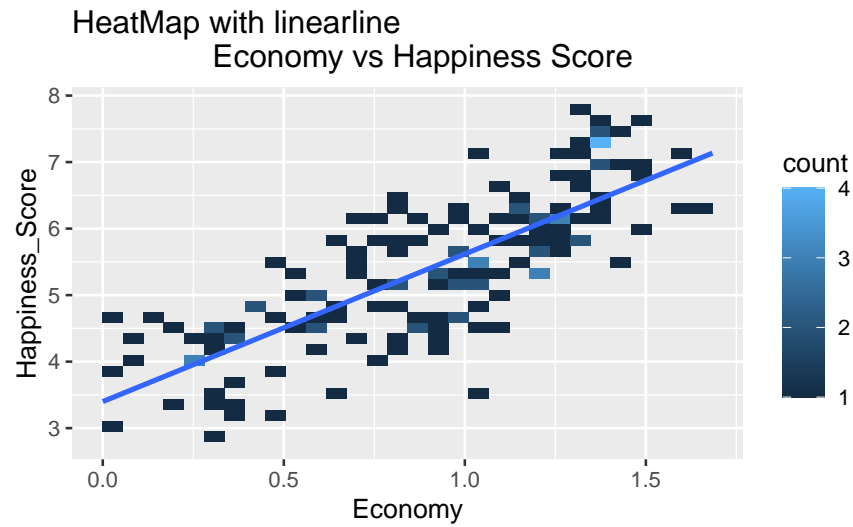
```
## 'geom_smooth()' using method = 'loess' and formula = 'y ~ x'
```



*HeatMap

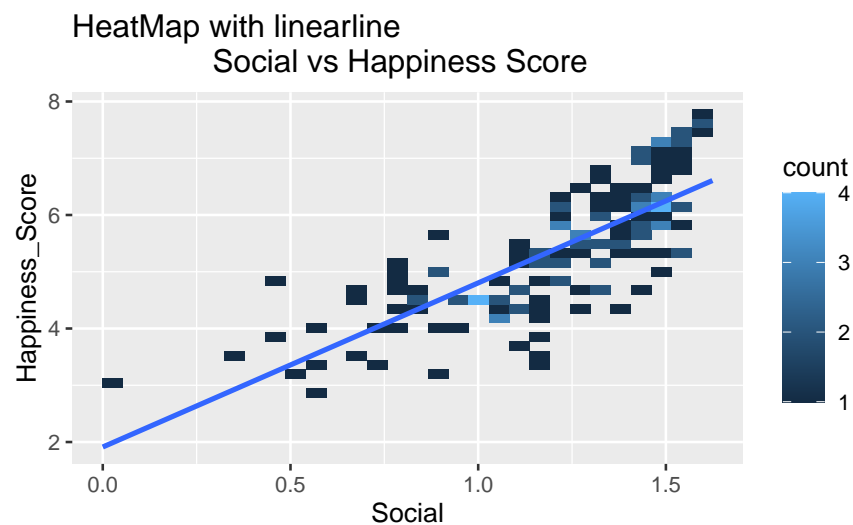
```
data %>%
  ggplot(aes(x = Economy, y = Happiness_Score)) +
  geom_bin2d(bins = 30) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "HeatMap with linearline
              Economy vs Happiness Score",
       x = "Economy", y = "Happiness_Score")
```

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

```
data %>%
  ggplot(aes(x = Social, y = Happiness_Score)) +
    geom_bin2d(bins = 30) +
    geom_smooth(method = "lm", se = FALSE) +
    labs(title = "HeatMap with linearline
               Social vs Happiness Score",
         x = "Social", y = "Happiness_Score")
```

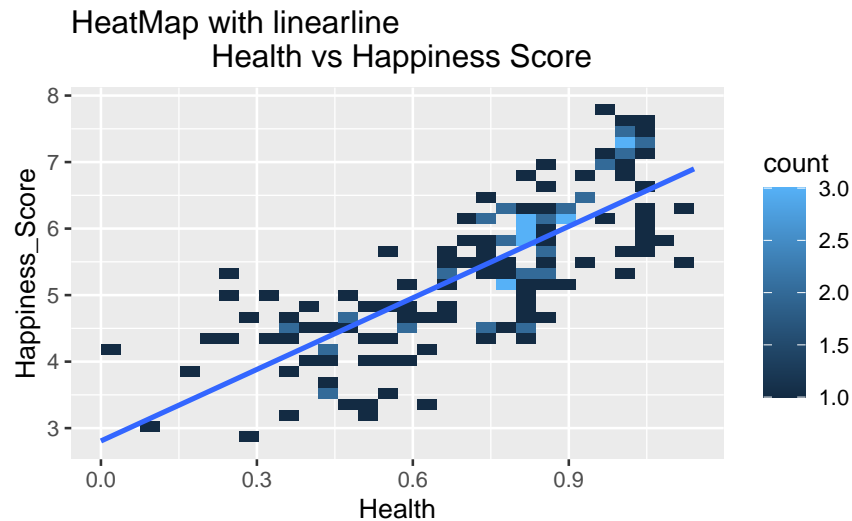
'geom_smooth()' using formula = 'y ~ x'



```
data %>%
  ggplot(aes(x = Health, y = Happiness_Score)) +
    geom_bin2d(bins = 30) +
    geom_smooth(method = "lm", se = FALSE) +
```

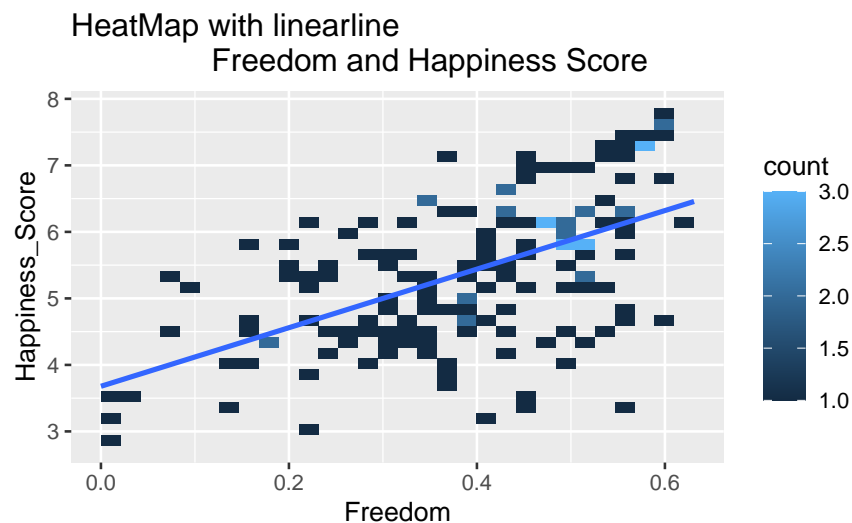
```
labs(title = "HeatMap with linearline  
Health vs Happiness Score",  
x = "Health", y = "Happiness_Score")
```

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



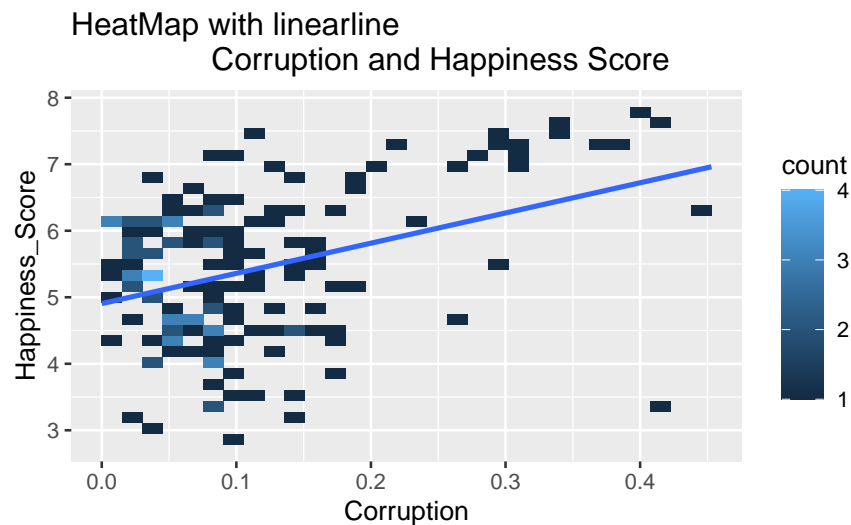
```
data %>%  
ggplot(aes(x = Freedom, y = Happiness_Score)) +  
  geom_bin2d(bins = 30) +  
  geom_smooth(method = "lm", se = FALSE) +  
  labs(title = "HeatMap with linearline  
Freedom and Happiness Score",  
x = "Freedom", y = "Happiness_Score")
```

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



```
data %>%
  ggplot(aes(x = Corruption, y = Happiness_Score)) +
    geom_bin2d(bins = 30) +
    geom_smooth(method = "lm", se = FALSE) +
    labs(title = "HeatMap with linearline",
         "Corruption and Happiness Score",
         x = "Corruption", y = "Happiness_Score")
```

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



===== [Module 4: Eugene Kim, Harold Lee - Explanatory Data Analysis]

```
str(data, vec.len = 2)
```

```
## tibble [156 x 9] (S3: tbl_df/tbl/data.frame)
## $ Overall rank      : num [1:156] 1 2 3 4 5 ...
## $ Country or region: chr [1:156] "Finland" "Denmark" ...
## $ Happiness_Score  : num [1:156] 7.77 7.6 ...
## $ Economy          : num [1:156] 1.34 1.38 ...
## $ Social           : num [1:156] 1.59 1.57 ...
## $ Health           : num [1:156] 0.986 0.996 ...
## $ Freedom          : num [1:156] 0.596 0.592 0.603 0.591 0.557 ...
## $ Generosity        : num [1:156] 0.153 0.252 0.271 0.354 0.322 ...
## $ Corruption        : num [1:156] 0.393 0.41 0.341 0.118 0.298 ...
```

```
head(
  select(data, Economy, Social, Health, Freedom, Corruption,
         Happiness_Score)
)
```

Economy	Social	Health	Freedom	Corruption	Happiness_Score
1.340	1.587	0.986	0.596	0.393	7.769
1.383	1.573	0.996	0.592	0.410	7.600
1.488	1.582	1.028	0.603	0.341	7.554
1.380	1.624	1.026	0.591	0.118	7.494
1.396	1.522	0.999	0.557	0.298	7.488
1.452	1.526	1.052	0.572	0.343	7.480

```
tail(select(data, Economy, Social, Health, Freedom, Corruption,
            Happiness_Score)
)
```

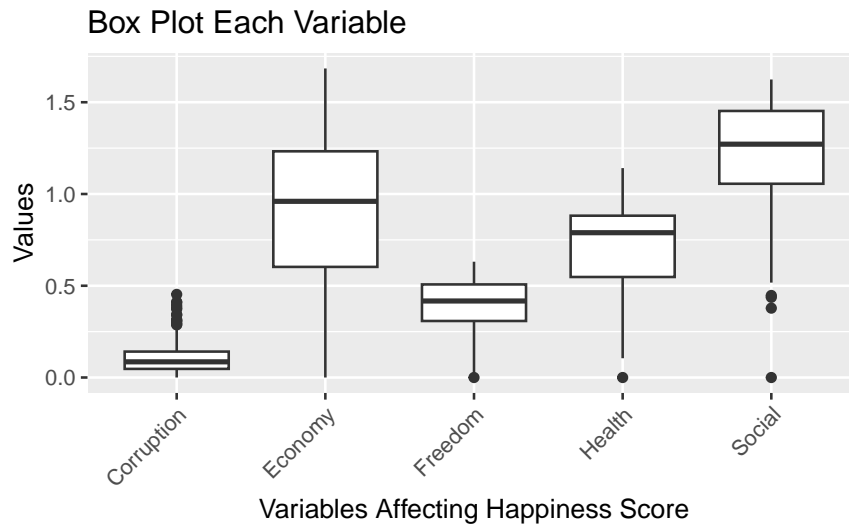
Economy	Social	Health	Freedom	Corruption	Happiness_Score
0.287	1.163	0.463	0.143	0.077	3.380
0.359	0.711	0.614	0.555	0.411	3.334
0.476	0.885	0.499	0.417	0.147	3.231
0.350	0.517	0.361	0.000	0.025	3.203
0.026	0.000	0.105	0.225	0.035	3.083
0.306	0.575	0.295	0.010	0.091	2.853

*Summary statistics

*Box Plot

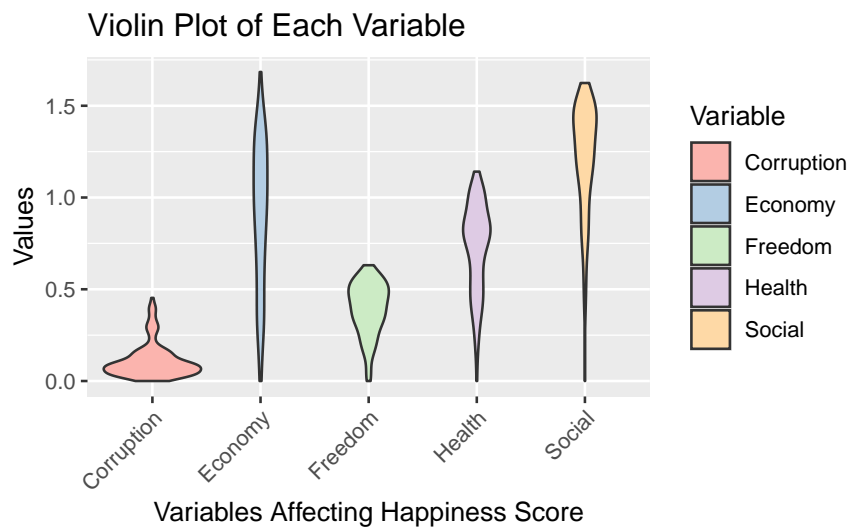
```
data_long <- data %>%
  gather(key = "Variable", value = "Score", Economy, Social, Health,
          Freedom, Corruption)

ggplot(data_long, aes(x = Variable, y = Score)) +
  geom_boxplot(width = 0.7) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Box Plot Each Variable",
       x = "Variables Affecting Happiness Score", y = "Values")
```



*Violin Plot

```
ggplot(data_long, aes(x = Variable, y = Score, fill = Variable)) +
  geom_violin(trim = TRUE) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Violin Plot of Each Variable",
       x = "Variables Affecting Happiness Score", y = "Values") +
  scale_fill_brewer(palette = "Pastel1")
```



*Summary

```
data %>%
  summarize(
    mean= mean(Economy),
```

```

median = median(Economy),
sd = sd(Economy),
iqr = IQR(Economy),
min = min(Economy),
max = max(Economy)
)

```

mean	median	sd	iqr	min	max
0.9051474	0.96	0.3983895	0.62975	0	1.684

```

data %>%
  summarize(
    mean= mean(Social),
    median = median(Social),
    sd = sd(Social),
    iqr = IQR(Social),
    min = min(Social),
    max = max(Social)
  )

```

mean	median	sd	iqr	min	max
1.208814	1.2715	0.2991914	0.39675	0	1.624

```

data %>%
  summarize(
    mean= mean(Health),
    median = median(Health),
    sd = sd(Health),
    iqr = IQR(Health),
    min = min(Health),
    max = max(Health)
  )

```

mean	median	sd	iqr	min	max
0.7252436	0.789	0.242124	0.334	0	1.141

```

data %>%
  summarize(
    mean= mean(Freedom),
    median = median(Freedom),
    sd = sd(Freedom),
    iqr = IQR(Freedom),
  )

```

```

min = min(Freedom),
max = max(Freedom)
)

```

mean	median	sd	iqr	min	max
0.3925705	0.417	0.1432895	0.19925	0	0.631

```

data %>%
  summarize(
    mean= mean(Corruption),
    median = median(Corruption),
    sd = sd(Corruption),
    iqr = IQR(Corruption),
    min = min(Corruption),
    max = max(Corruption)
  )

```

mean	median	sd	iqr	min	max
0.1106026	0.0855	0.0945378	0.09425	0	0.453

```

data_E <- data %>%
  summarize(
    mean= mean(Economy),
    median = median(Economy),
    sd = sd(Economy),
    iqr = IQR(Economy),
    min = min(Economy),
    max = max(Economy)
  )

```

```

data_S <- data %>%
  summarize(
    mean= mean(Social),
    median = median(Social),
    sd = sd(Social),
    iqr = IQR(Social),
    min = min(Social),
    max = max(Social)
  )

```

```

data_H <- data %>%
  summarize(
    mean= mean(Health),

```

```

    median = median(Health),
    sd = sd(Health),
    iqr = IQR(Health),
    min = min(Health),
    max = max(Health)
  )

```

```

data_F <- data %>%
  summarize(
    mean= mean(Freedom),
    median = median(Freedom),
    sd = sd(Freedom),
    iqr = IQR(Freedom),
    min = min(Freedom),
    max = max(Freedom)
  )

```

```

data_C <- data %>%
  summarize(
    mean= mean(Corruption),
    median = median(Corruption),
    sd = sd(Corruption),
    iqr = IQR(Corruption),
    min = min(Corruption),
    max = max(Corruption)
  )

```

```

combined_data <- rbind(data_E, data_S, data_H, data_F, data_C)
row.names(combined_data) <- c("Economy", "Social", "Health", "Freedom", "Corruption")

```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
print(combined_data)
```

```

## # A tibble: 5 x 6
##   mean median    sd    iqr  min  max
## * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 0.905 0.96  0.398 0.630    0 1.68
## 2 1.21  1.27  0.299 0.397    0 1.62
## 3 0.725 0.789 0.242 0.334    0 1.14
## 4 0.393 0.417 0.143 0.199    0 0.631
## 5 0.111 0.0855 0.0945 0.0942    0 0.453

```


[Module 5: Chun Jin Park - Modeling].

linear model using lm

```
model <- lm(Happiness_Score ~ Economy + Social + Health + Freedom + Generosity
            + Corruption, data = data)

summary(model)
```

```
##
## Call:
## lm(formula = Happiness_Score ~ Economy + Social + Health + Freedom +
##     Generosity + Corruption, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.75304 -0.35306  0.05703  0.36695  1.19059
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.7952     0.2111   8.505 1.77e-14 ***
## Economy       0.7754     0.2182   3.553 0.000510 ***
## Social        1.1242     0.2369   4.745 4.83e-06 ***
## Health        1.0781     0.3345   3.223 0.001560 **
## Freedom       1.4548     0.3753   3.876 0.000159 ***
## Generosity     0.4898     0.4977   0.984 0.326709
## Corruption     0.9723     0.5424   1.793 0.075053 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5335 on 149 degrees of freedom
## Multiple R-squared:  0.7792, Adjusted R-squared:  0.7703
## F-statistic: 87.62 on 6 and 149 DF,  p-value: < 2.2e-16
```

tidy to get the model coefficients

```
coefficients <- tidy(model)
print(coefficients)
```

```
## # A tibble: 7 x 5
##   term          estimate std.error statistic  p.value
##   <chr>         <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    1.80      0.211      8.51 1.77e-14
```

```
## 2 Economy      0.775      0.218      3.55 5.10e- 4
## 3 Social       1.12       0.237      4.75 4.83e- 6
## 4 Health       1.08       0.335      3.22 1.56e- 3
## 5 Freedom      1.45       0.375      3.88 1.59e- 4
## 6 Generosity    0.490      0.498      0.984 3.27e- 1
## 7 Corruption    0.972      0.542      1.79 7.51e- 2
```

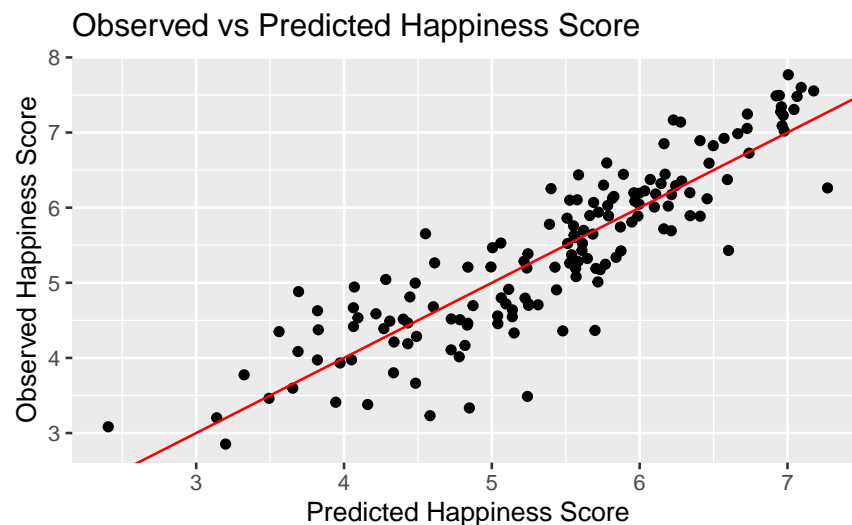
glance to get the model's performance metrics

```
performance <- glance(model)
print(performance)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
##   <dbl>      <dbl> <dbl>      <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1    0.779        0.770 0.534        87.6 2.40e-46     6  -120.  256.  280.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

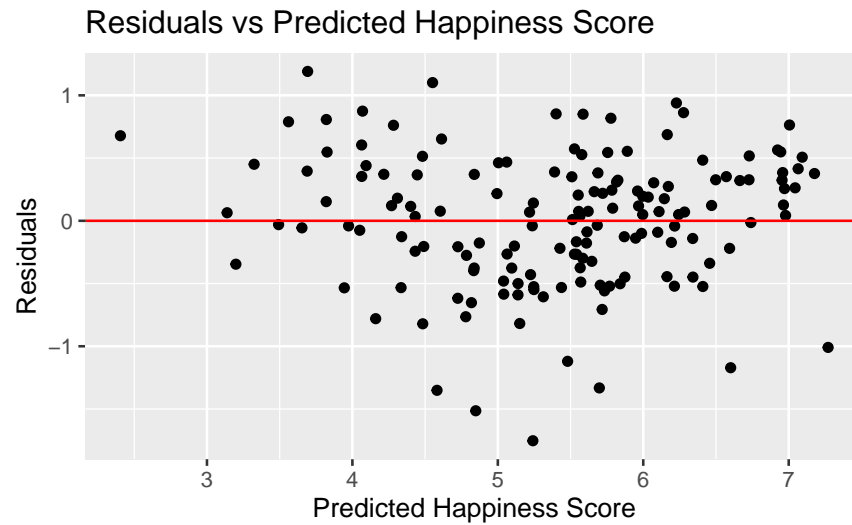
Observed vs Predicted plot

```
ggplot(data, aes(x = predict(model), y = Happiness_Score)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1, col = "red") +
  labs(title =
    "Observed vs Predicted Happiness Score",
    x = "Predicted Happiness Score",
    y = "Observed Happiness Score")
```



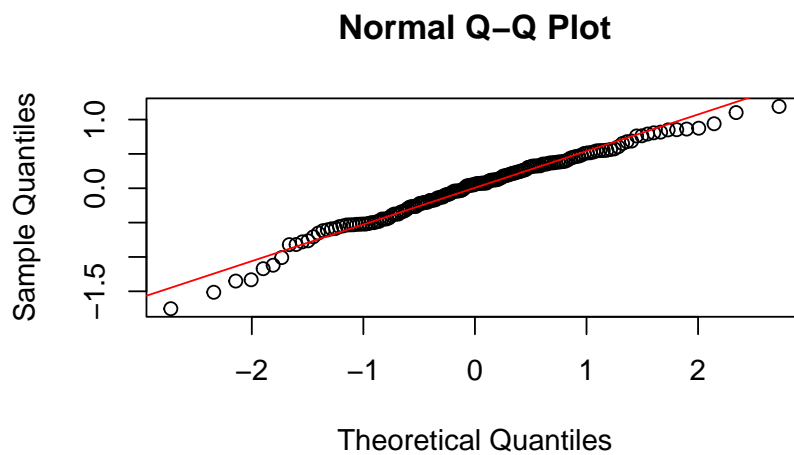
Residuals vs Predicted plot

```
ggplot(data, aes(x = predict(model), y = residuals(model))) +  
  geom_point() +  
  geom_hline(yintercept = 0, col = "red") +  
  labs(title = "Residuals vs Predicted Happiness Score",  
       x = "Predicted Happiness Score",  
       y = "Residuals")
```



Q-Q plot

```
qqnorm(residuals(model))  
qqline(residuals(model), col = "red")
```



« « « < HEAD

===== [Module 6: SeNa Julsdorf, Hyeongseok Sim]

```
null_distribution_correlation <- data %>%  
  specify(Happiness_Score ~ Social) %>%  
  hypothesize(null = "independence") %>%  
  generate(reps = 10000, type = "permute") %>%  
  calculate(stat = "correlation")
```

```
happiness_obs_stat <- data %>%  
  specify(Happiness_Score ~ Social) %>%  
  calculate(stat = "correlation")
```

```
p_value <- null_distribution_correlation %>%  
  get_p_value(obs_stat = happiness_obs_stat, direction = "both")
```

```
## Warning: Please be cautious in reporting a p-value of 0. This result is an approximation  
## based on the number of 'reps' chosen in the 'generate()' step.  
## i See 'get_p_value()' ('?infer::get_p_value()') for more information.
```

```
print(p_value)
```

```
## # A tibble: 1 x 1  
##   p_value  
##   <dbl>  
## 1      0
```

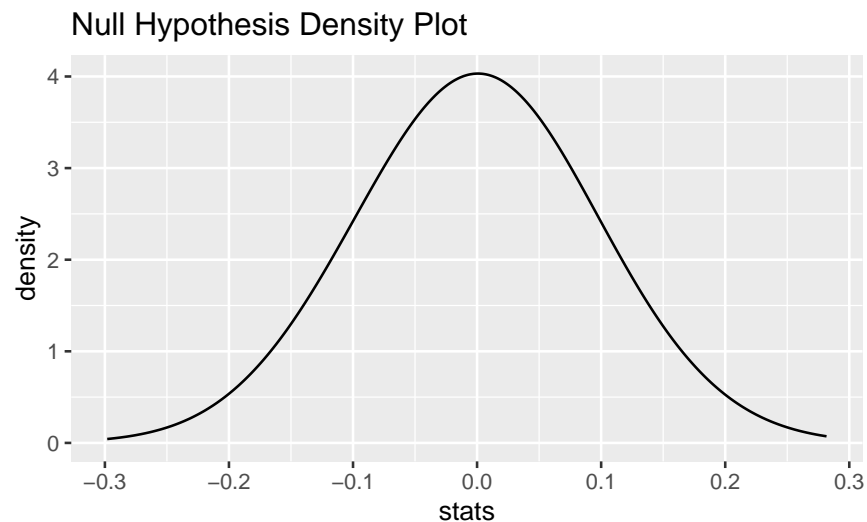
```
visualize(null_distribution_correlation) +  
  shade_p_value(obs_stat = happiness_obs_stat, direction = "two_sided") +  
  labs(title = "Null Distribution of Happiness Score", x = "Correlation",  
        y = "Count")
```



```

null_distribution_correlation %>%
  ggplot() +
  geom_density(mapping = aes(x = stat), adjust = 5) +
  labs(title = "Null Hypothesis Density Plot", x = "stats", y = "density")

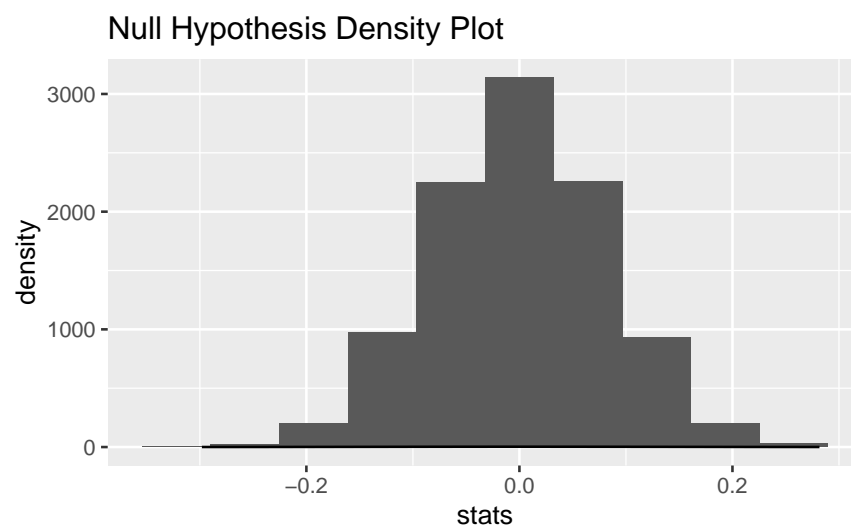
```



```

null_distribution_correlation %>%
  ggplot() +
  geom_histogram(mapping = aes(x = stat), bins = 10) +
  geom_density(mapping = aes(x = stat), adjust = 10) +
  labs(title = "Null Hypothesis Density Plot", x = "stats", y = "density")

```



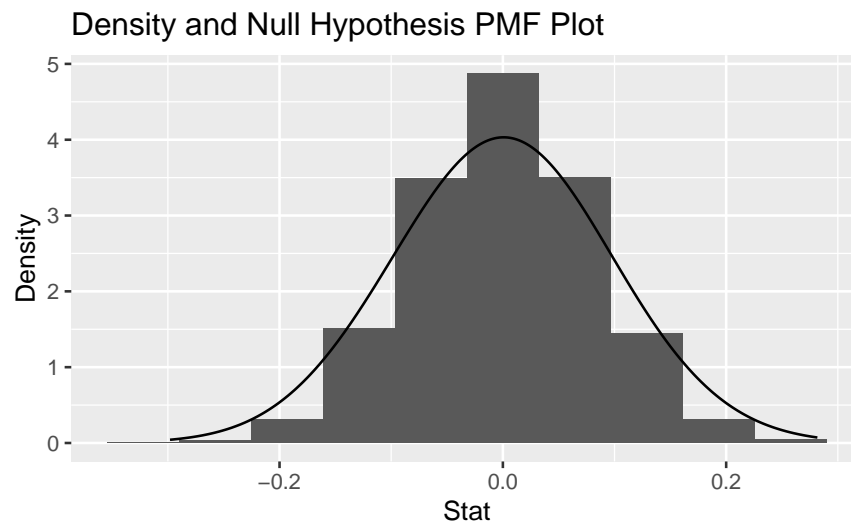
```

null_distribution_correlation %>%
  ggplot() +
  geom_histogram(mapping = aes(x = stat, y = ..density..), bins = 10) +

```

```
geom_density(mapping = aes(x = stat), adjust = 5) +
labs(title = "Density and Null Hypothesis PMF Plot",
x = "Stat", y = "Density")
```

```
## Warning: The dot-dot notation ('..density..') was deprecated in ggplot2 3.4.0.
## i Please use 'after_stat(density)' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```



===== [Module 7: Jeonghwa Cho]

```
bootstraps_distribution_correlation <- data %>%
specify(Happiness_Score ~ Social) %>%
generate( reps = 10000, type = "bootstrap") %>%
calculate(stat = "correlation")
```

```
bootstrap_ci <- bootstraps_distribution_correlation %>%
get_confidence_interval()
```

```
## Using 'level = 0.95' to compute confidence interval.
```

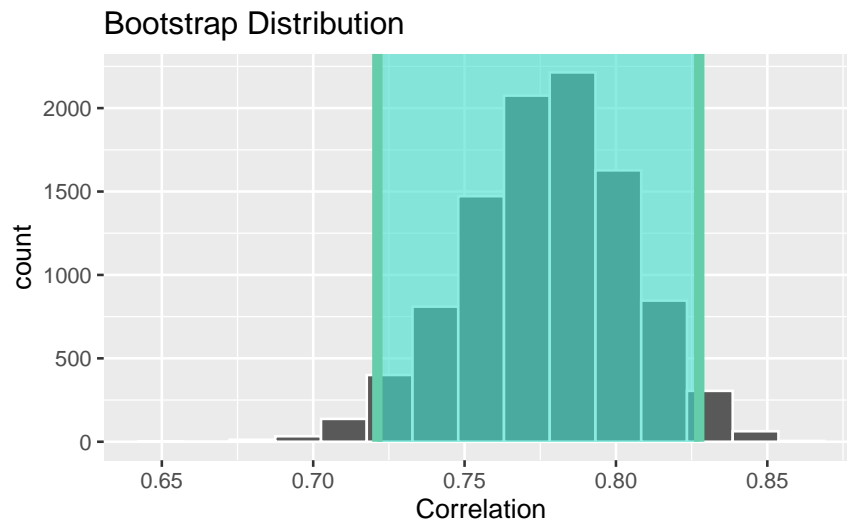
```
bootstrap_ci
```

lower_ci	upper_ci
0.7211866	0.8275057

```

bootstraps_distribution_correlation %>%
visualize()+
shade_confidence_interval(bootstrap_ci)+
ggtitle("Bootstrap Distribution")+
xlab("Correlation")+
ylab("count")

```



```

bootstrap_results <- cohens_d_bootstrap(
data = data_long,
model = Happiness_Score ~ Variable)

```

```

bootstrap_report(bootstrap_results)

```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 5000 bootstrap replicates
##
## CALL :
## boot::boot.ci(boot.out = cohens_d_bootstrap_sim, type = c("perc"))
##
## Intervals :
## Level      Percentile
## 95%      (-0.2207,  0.2228 )
## Calculations and Intervals on Original Scale
##
## Response variable
## Happiness_Score
##
## Explanatory variable
## Variable
##

```

```
## Explanatory category with larger mean
## Economy
##
## Explanatory category with smaller mean
## Social
##
## Cohen's d observed value
## 0
```

```
plot_ci(bootstrap_results)
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

