

# Day 3 : Statistical analysis using R

## Session 9: Descriptive statistics and hypothesis testing

### 9.1. Descriptive statistics

#### E031-descriptive\_statistics.R

```
data(mtcars) #from datasets package

vars <- c('mpg', 'cyl', 'disp')

#basic descriptive statistics from base package
summary(mtcars[vars])
      mpg          cyl          disp
Min.   :10.40   Min.   :4.000   Min.   :71.1
1st Qu.:15.43  1st Qu.:4.000   1st Qu.:120.8
Median :19.20  Median :6.000   Median :196.3
Mean   :20.09  Mean   :6.188   Mean   :230.7
3rd Qu.:22.80  3rd Qu.:8.000   3rd Qu.:326.0
Max.   :33.90  Max.   :8.000   Max.   :472.0

#descriptive statistics from other packages
Hmisc::describe(mtcars[vars])
mtcars[vars]

 3 Variables    32 Observations

mpg
  n missing distinct     Info      Mean   pMedian     Gmd     .05     .10     .25     .50     .75     .90     .95
 32      0        25   0.999   20.09    19.6   6.796   12.00   14.34   15.43   19.20   22.80   30.09   31.30

lowest : 10.4 13.3 14.3 14.7 15 , highest: 26  27.3 30.4 32.4 33.9
-----
cyl
  n missing distinct     Info      Mean   pMedian     Gmd
 32      0        3   0.866   6.188      6   1.948

Value      4       6       8
Frequency  11      7      14
Proportion 0.344  0.219  0.438

For the frequency table, variable is rounded to the nearest 0
-----
disp
  n missing distinct     Info      Mean   pMedian     Gmd     .05     .10     .25     .50     .75     .90     .95
 32      0        27   0.999  230.7   223.4  142.5   77.35   80.61  120.83  196.30  326.00  396.00  449.00

lowest : 71.1 75.7 78.7 79  95.1, highest: 360 400 440 460 472
-----

pastecs::stat.desc(mtcars[vars])
      mpg          cyl          disp
nbr.val     32.0000000 32.0000000 3.200000e+01
nbr.null    0.0000000 0.0000000 0.000000e+00
nbr.na      0.0000000 0.0000000 0.000000e+00
min        10.4000000 4.0000000 7.110000e+01
max        33.9000000 8.0000000 4.720000e+02
range      23.5000000 4.0000000 4.009000e+02
sum        642.9000000 198.0000000 7.383100e+03
median     19.2000000 6.0000000 1.963000e+02
mean       20.0906250 6.1875000 2.307219e+02
SE.mean    1.0654240 0.3157093 2.190947e+01
CI.mean.0.95 2.1729465 0.6438934 4.468466e+01
var        36.3241028 3.1895161 1.536080e+04
std.dev    6.0269481 1.7859216 1.239387e+02
coef.var   0.2999881 0.2886338 5.371779e-01

psych::describe(mtcars[vars])
  vars  n   mean     sd median trimmed     mad    min    max range skew kurtosis      se
mpg    1 32  20.09    6.03   19.2   19.70    5.41 10.4  33.9  23.5  0.61   -0.37  1.07
cyl    2 32   6.19    1.79    6.0    6.23    2.97  4.0   8.0   4.0  -0.17   -1.76  0.32
disp   3 32 230.72 123.94 196.3 222.52 140.48 71.1 472.0 400.9  0.38   -1.21 21.91

#-----
# Descriptive statistics by group
#-----
#grouping by one variable
by(mtcars[vars], #dataset
  list(Transmission = mtcars$am), #grouping variable: Transmission (0 = automatic, 1 = manual)
```

```

summary) #function
Transmission: 0
      mpg          cyl          disp
Min.  :10.40    Min.  :4.000    Min.  :120.1
1st Qu.:14.95   1st Qu.:6.000   1st Qu.:196.3
Median :17.30   Median :8.000   Median :275.8
Mean   :17.15   Mean   :6.947   Mean   :290.4
3rd Qu.:19.20   3rd Qu.:8.000   3rd Qu.:360.0
Max.   :24.40   Max.   :8.000   Max.   :472.0
-----
Transmission: 1
      mpg          cyl          disp
Min.  :15.00    Min.  :4.000    Min.  : 71.1
1st Qu.:21.00   1st Qu.:4.000   1st Qu.: 79.0
Median :22.80   Median :4.000   Median :120.3
Mean   :24.39   Mean   :5.077   Mean   :143.5
3rd Qu.:30.40   3rd Qu.:6.000   3rd Qu.:160.0
Max.   :33.90   Max.   :8.000   Max.   :351.0

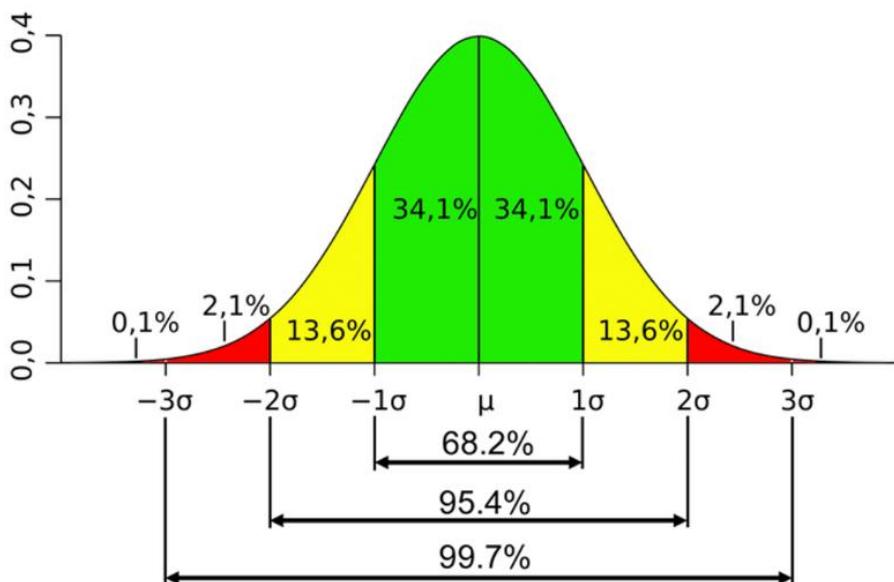
#grouping by multiple variables
by(mtcars$vars,
  list(Transmission = mtcars$am, Engine = mtcars$vs), # Transmission (0 = automatic, 1 = manual), Engine (0 = V-shaped, 1 = straight)
  summary)
Transmission: 0
Engine: 0
      mpg          cyl          disp
Min.  :10.40    Min.  :8       Min.  :275.8
1st Qu.:14.05   1st Qu.:8       1st Qu.:296.9
Median :15.20   Median :8       Median :355.0
Mean   :15.05   Mean   :8       Mean   :357.6
3rd Qu.:16.62   3rd Qu.:8       3rd Qu.:410.0
Max.   :19.20   Max.   :8       Max.   :472.0
-----
Transmission: 1
Engine: 0
      mpg          cyl          disp
Min.  :15.00    Min.  :4.000    Min.  :120.3
1st Qu.:16.77   1st Qu.:6.000   1st Qu.:148.8
Median :20.35   Median :6.000   Median :160.0
Mean   :19.75   Mean   :6.333   Mean   :206.2
3rd Qu.:21.00   3rd Qu.:7.500   3rd Qu.:265.8
Max.   :26.00   Max.   :8.000   Max.   :351.0
-----
Transmission: 0
Engine: 1
      mpg          cyl          disp
Min.  :17.80    Min.  :4.000    Min.  :120.1
1st Qu.:18.65   1st Qu.:4.000   1st Qu.:143.8
Median :21.40   Median :6.000   Median :167.6
Mean   :20.74   Mean   :5.143   Mean   :175.1
3rd Qu.:22.15   3rd Qu.:6.000   3rd Qu.:196.3
Max.   :24.40   Max.   :6.000   Max.   :258.0
-----
Transmission: 1
Engine: 1
      mpg          cyl          disp
Min.  :21.40    Min.  :4       Min.  : 71.1
1st Qu.:25.05   1st Qu.:4       1st Qu.: 77.2
Median :30.40   Median :4       Median : 79.0
Mean   :28.37   Mean   :4       Mean   : 89.8
3rd Qu.:31.40   3rd Qu.:4       3rd Qu.:101.5
Max.   :33.90   Max.   :4       Max.   :121.0

```

## 9.2. The concept of normal distribution

### a. What is a Normal Distribution?

- **Shape:** The normal distribution looks like a bell-shaped curve.
- **Symmetry:** It is perfectly symmetrical around the center.



#### b. Key Characteristics:

- **Mean (Average):** The center of the curve.
- **Standard Deviation:** Measures the spread of the data.
  - 68.2% of the data falls within 1 standard deviation of the mean.
  - 95.4% falls within 2 standard deviations.
  - 99.7% falls within 3 standard deviations.

#### c. Why is it Important?

- **Natural Occurrences:** Many natural phenomena follow this distribution (e.g., heights, test scores). For example, most students score around the average in a class, fewer scoring very high or very low.
- **Central Limit Theorem:** In large samples, the samples' mean tend to be normally distributed. ([Video](#))
- **Statistical Inferences:** Helps in making predictions and decisions based on data.

### 8.1. Hypothesis testing

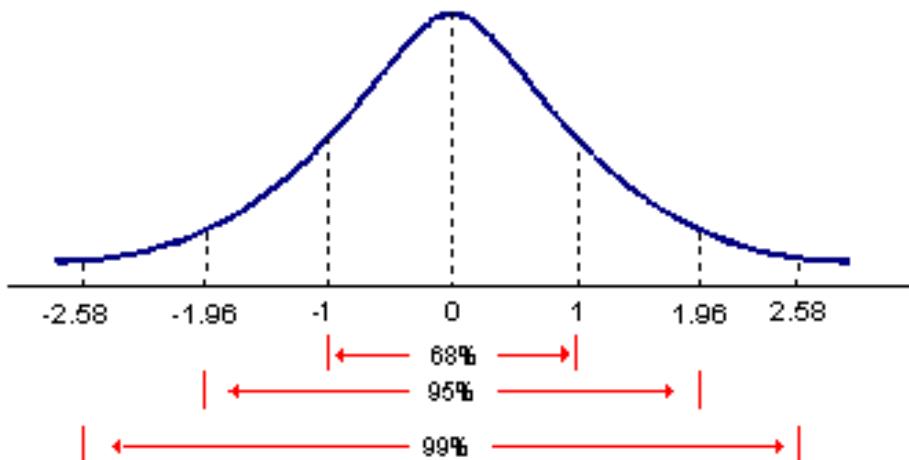
#### a. What is Hypothesis Testing?

- Hypothesis testing is a method used to decide whether there is enough evidence to support a particular claim about a population based on a sample of data.
- **Null Hypothesis ( $H_0$ ):** This is the default statement that there is no effect or no difference. It assumes that any observed differences are due to random chance.  
Example: "The average age is equal to 20."
- **Alternative Hypothesis ( $H_1$ ):** This is what you want to prove, stating there is an effect or a difference.  
Example: "The average age is not equal to 20."

#### b. Procedure of hypothesis testing

- State the null and alternative hypothesis. (e.g.  $H_0: \mu = 0, H_1: \mu \neq 0$ )
- Collect sample data.
- Calculate sample mean and standard error ( $\frac{s}{\sqrt{n}}$ ).

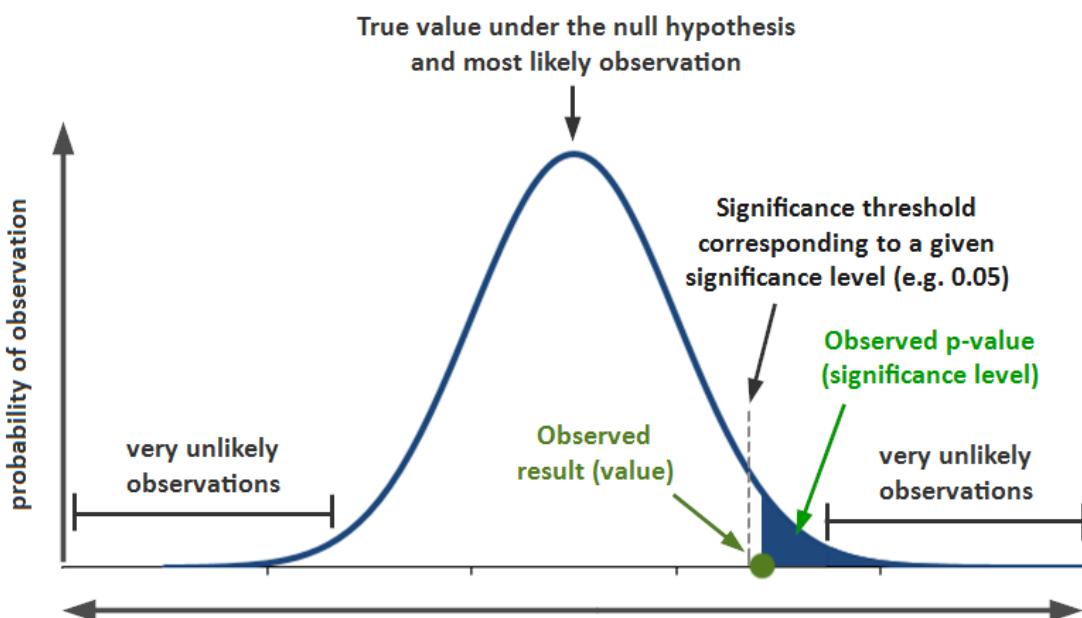
- Calculate t-statistics ( $t = \frac{\bar{X} - \mu}{Standard\ Error}$ ).
- Compare absolute value of t-statistics  $|t|$  with critical values for given level of significance ( $\alpha$ ). [1.65 (10% significance level), 1.96 (5%), 2.58 (1%)]



- Decision: reject null hypothesis if  $|t|$  exceeds critical value, otherwise fail to reject null hypothesis.

### c. Hypothesis testing with p-value

- p-value : probability (area under normal distribution) beyond  $|t|$  i.e., probability of rejecting  $H_0$  when its true.



- Decision : reject null hypothesis if p-value is lower than the significance level, otherwise fail to reject null hypothesis.
- Easier to conduct hypothesis testing with p-value. No need to calculate t-statistics and remember different critical values.

#### E033-hypothesis\_testing.R

```
library(dplyr)

# Set seed for reproducibility
set.seed(12345)

# Create a dummy dataset
n <- 1000
group <- rep(0:1, length.out = n)
score <- 50 + group * 10 + rnorm(n, mean = 0, sd = 10)
```

```

# Combine into a data frame
data <- data.frame(group = group, score = score)

# Conducting hypothesis testing (one-sample t-tests)
t.test(data$score, mu = 50) # H0: pop_mean = 50
    One Sample t-test

data: data$score
t = 15.613, df = 999, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 50
95 percent confidence interval:
54.77547 56.14850
sample estimates:
mean of x
55.46198

t.test(data$score, mu = 55) # H0: pop_mean = 55
    One Sample t-test

data: data$score
t = 1.3205, df = 999, p-value = 0.187
alternative hypothesis: true mean is not equal to 55
95 percent confidence interval:
54.77547 56.14850
sample estimates:
mean of x
55.46198

t.test(data$score, mu = 60) # H0: pop_mean = 60
    One Sample t-test

data: data$score
t = -12.972, df = 999, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 60
95 percent confidence interval:
54.77547 56.14850
sample estimates:
mean of x
55.46198

# -----
# Conducting two-sample t-test
# -----
data0 <- filter(data, group == 0)
data1 <- filter(data, group == 1)
t.test(data0$score, data1$score)
    Welch Two Sample t-test

data: data0$score and data1$score
t = -15.074, df = 997.82, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-10.763662 -8.284043
sample estimates:
mean of x mean of y
50.70006 60.22391

#OR

t.test(score ~ group, data = data) # H0: pop_mean_group1 = pop_mean_group2

```

```

Welch Two Sample t-test

data: score by group
t = -15.074, df = 997.82, p-value < 2.2e-16
alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
95 percent confidence interval:
-10.763662 -8.284043
sample estimates:
mean in group 0 mean in group 1
50.70006      60.22391

```

### Task 7:

Using NMICS6 data (009-hl.sav), conduct a hypothesis test whether average age between male and female is statistically different.

```

# Load necessary libraries
library(haven) # For reading SPSS files
library(dplyr)

# Import SPSS file from the URL
df <- read_spss('data/009-hl.sav')

# HL6 -> Age, HL4 -> Sex
# Summarize age for males (HL4 == 1) and females (HL4 == 2)
df_male <- filter(df, HL4 == 1)
df_female <- filter(df, HL4 == 2)
mean(df_male$HL6)
mean(df_female$HL6)
#OR
by(df$HL6, df$HL4, summary) #average age : male = 28.26, female = 28.83

# Conduct hypothesis testing (two-sample t-test)
t.test(HL6 ~ HL4, data = df)
Welch Two Sample t-test

data: HL6 by HL4
t = -3.1618, df = 54737, p-value = 0.001569
alternative hypothesis: true difference in means between group 1 and group 2 is not equal to 0
95 percent confidence interval:
-0.9125138 -0.2141086
sample estimates:
mean in group 1 mean in group 2
28.26344      28.82675

# Alternatively, use linear regression
regression_result <- lm(HL6 ~ HL4, data = df)
summary(regression_result)
Call:
lm(formula = HL6 ~ HL4, data = df)

Residuals:
    Min      1Q  Median      3Q     Max 
-28.827 -17.827 - 3.827 15.173 69.737 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 27.7001    0.2860 96.846 < 2e-16 ***
HL4         0.5633    0.1775   3.173 0.00151 **  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21.07 on 56593 degrees of freedom
Multiple R-squared:  0.0001779, Adjusted R-squared:  0.0001602 
F-statistic: 10.07 on 1 and 56593 DF,  p-value: 0.001509

```

## Session 10: Regression

### 10.1 Fitting regression models with lm()

Symbols commonly used in R formulas

Symbol	Usage
$\sim$	Separates response variables on the left from the explanatory variables on the right. For example, a prediction of $y$ from $x$ , $z$ , and $w$ would be coded $y \sim x + z + w$ .
$+$	Separates predictor variables
$:$	Denotes an interaction between predictor variables. A prediction of $y$ from $x$ , $z$ , and the interaction between $x$ and $z$ would be coded $y \sim x + z + x:z$ .
$*$	A shortcut for denoting all possible interactions. The code $y \sim x * z * w$ expands to $y \sim x + z + w + x:z + x:w + z:w$ .
$^n$	Denotes interactions up to a specified degree. The code $y \sim (x + z + w)^2$ expands to $y \sim x + z + w + x:z + x:w + z:w$ .
$.$	A placeholder for all other variables in the data frame except the dependent variable. For example, if a data frame contained the variables $x$ , $y$ , $z$ , and $w$ , then the code $y \sim .$ would expand to $y \sim x + z + w$ .
$-$	A minus sign removes a variable from the equation. For example, $y \sim (x + z + w)^2 - x:w$ expands to $y \sim x + z + w + x:z + z:w$ .
$-1$	Suppresses the intercept. For example, the formula $y \sim x -1$ fits a regression of $y$ on $x$ and forces the line through the origin at $x=0$ .
$I()$	Elements within the parentheses are interpreted arithmetically. For example, $y \sim x + (z + w)^2$ expands to $y \sim x + z + w + z:w$ . In contrast, the code $y \sim x + I((z + w)^2)$ expands to $y \sim x + h$ , where $h$ is a new variable created by squaring the sum of $z$ and $w$ .
function	Mathematical functions can be used in formulas. For example, $\log(y) \sim x + z + w$ predicts $\log(y)$ from $x$ , $z$ , and $w$ .

### E034-simple\_regression.R

```
# Set seed for reproducibility
set.seed(12345)

# Generate 1000 observations
n <- 1000

# Generate study_hours as uniform random numbers between 0 and 10
study_hours <- round(runif(n, min = 0, max = 10))

# Generate score as a linear function of study_hours with noise
score <- 50 + 5 * study_hours + rnorm(n, mean = 0, sd = 5)

# Combine into a data frame
df <- data.frame(study_hours, score)

# Perform linear regression
model <- lm(score ~ study_hours, data = df)

# Summarize the regression results
summary(model)
```

```

Call:
lm(formula = score ~ study_hours, data = df)

Residuals:
    Min      1Q  Median      3Q     Max 
-16.612 -3.334 -0.018  3.509 16.737 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 49.94446   0.32742 152.54 <2e-16 ***  
study_hours  4.99446   0.05571  89.64 <2e-16 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

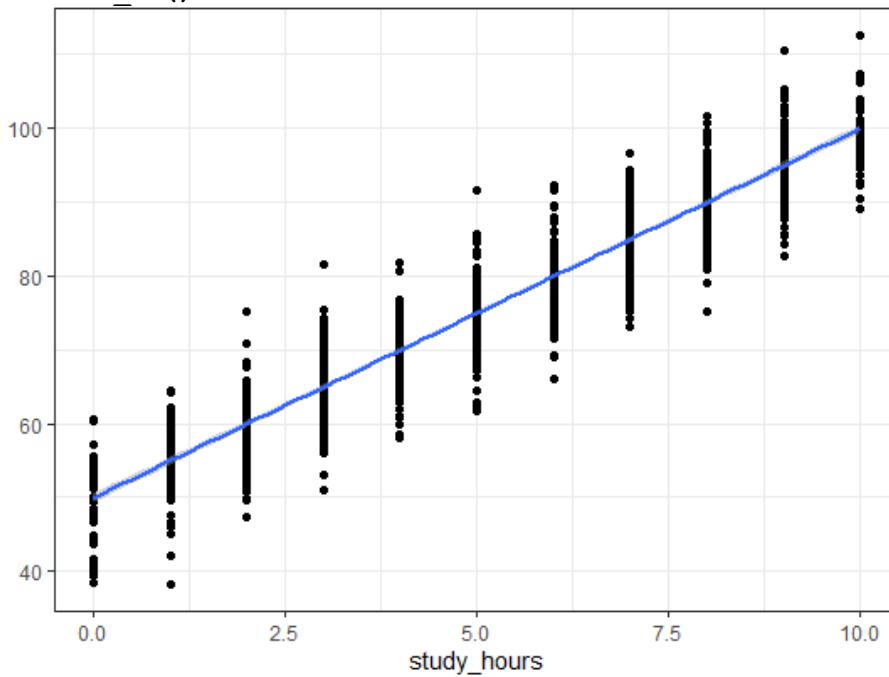
Residual standard error: 5.035 on 998 degrees of freedom
Multiple R-squared:  0.8895, Adjusted R-squared:  0.8894 
F-statistic: 8036 on 1 and 998 DF, p-value: < 2.2e-16

```

```

# visualizing simple regression
library(ggplot2)
ggplot(data = df, aes(x=study_hours, y=score)) +
  geom_point() +
  geom_smooth(method = 'lm') +
  theme_bw()

```



## 10.2 Multiple regression

### E035-multiple\_regression.R

```

# Set seed for reproducibility
set.seed(12345)

# Generate 200 observations
n <- 200

# Generate age variable (cycles from 18 to 69)
age <- (1:n %% 52) + 18

# Generate educ_year variable (cycles from 0 to 17)
educ_year <- (1:n %% 18)

# Generate income variable with a linear relationship to age and educ_year,
# plus noise
income <- 20000 + 800 * age + 3000 * educ_year + rnorm(n, mean = 0, sd =
2000)

```

```

# Combine into a data frame
df <- data.frame(age, educ_year, income)

# Regression with omitted variable
model_omitted <- lm(income ~ age, data = df)
summary(model_omitted)
Call:
lm(formula = income ~ age, data = df)

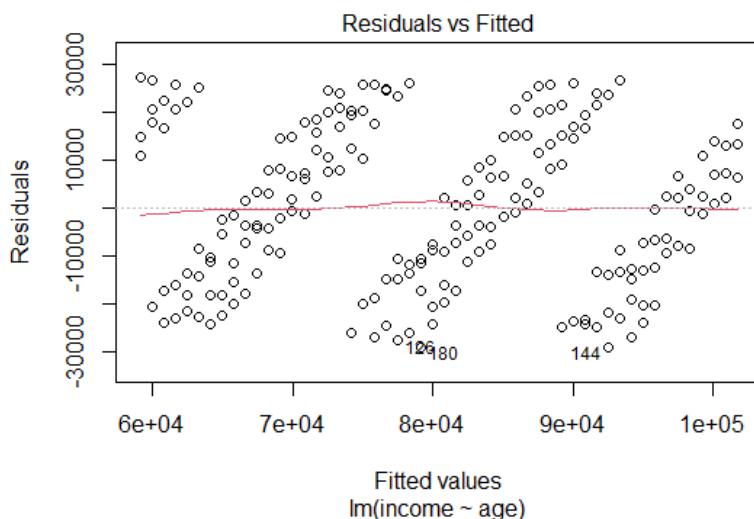
Residuals:
    Min      1Q  Median      3Q     Max 
-29181.6 -13567.7   363.5 14563.5 27208.4 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 44039.53   3535.95   12.46 <2e-16 ***
age          835.97    78.13    10.70 <2e-16 ***  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 16120 on 198 degrees of freedom
Multiple R-squared:  0.3663, Adjusted R-squared:  0.3631 
F-statistic: 114.5 on 1 and 198 DF,  p-value: < 2.2e-16

# Residual diagnostics for omitted variable model
plot(model_omitted, which = 1)  # Residual vs Fitted plot

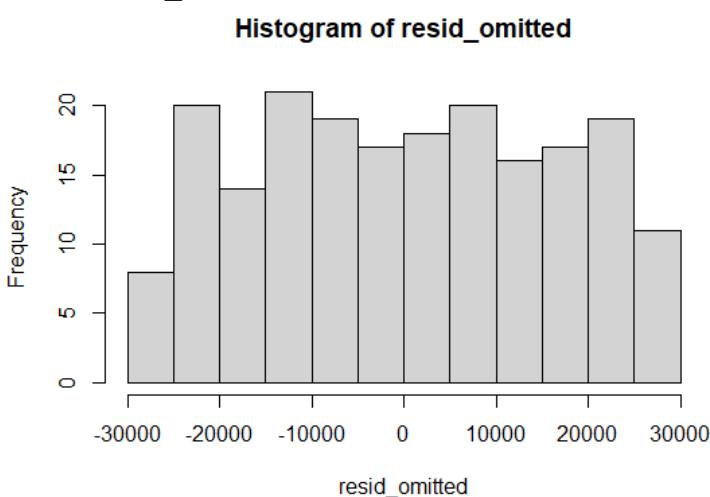
```



```

resid_omitted <- residuals(model_omitted)
hist(resid_omitted)

```



```

shapiro.test(resid_omitted)  # Shapiro-Wilk test for normality [H0: normally distributed]

```

```

Shapiro-Wilk normality test

data: resid_omitted
W = 0.95517, p-value = 6.14e-06

#-----#
# Multiple regression with correct specification
model_correct <- lm(income ~ age + educ_year, data = df)
summary(model_correct)

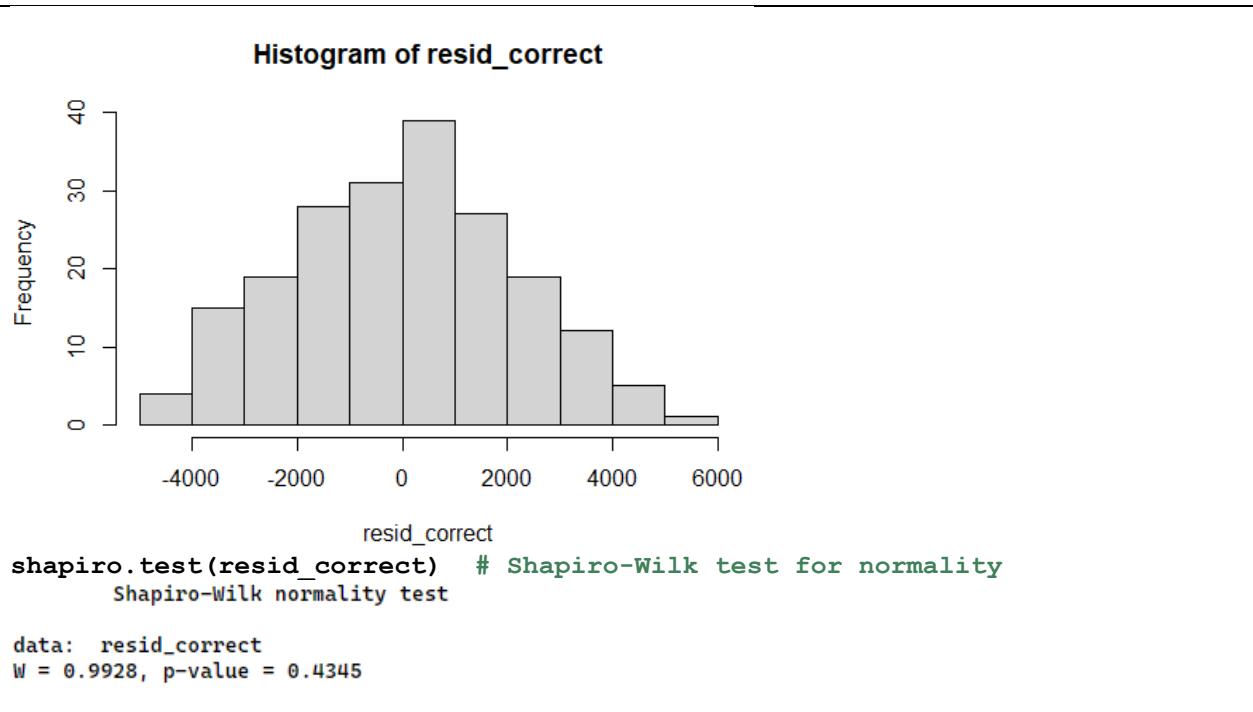
Histogram of resid_omitted

Frequency
0 5 10 15 20
-30000 -20000 -10000 0 10000 20000 30000
resid_omitted

# Residual diagnostics for correctly specified model
plot(model_correct, which = 1) # Residual vs Fitted plot

Residuals vs Fitted
6000
2000
-2000
-6000
40000 60000 80000 100000 120000
Fitted values
lm(income ~ age + educ_year)
resid_correct <- residuals(model_correct)
hist(resid_correct)

```



### 10.3 Polynomial regression

**E036-polynomial\_regression.R**

```

library(ggplot2)

mtcars <- datasets::mtcars

#-----
#simple regression
#-----
fit <- lm(data = mtcars, formula = mpg ~ hp) # mpg: Miles/(US) gallon, hp:
Gross horsepower
summary(fit) #R-squared : 0.6024, Residual standard error: 3.863
Call:
lm(formula = mpg ~ hp, data = mtcars)

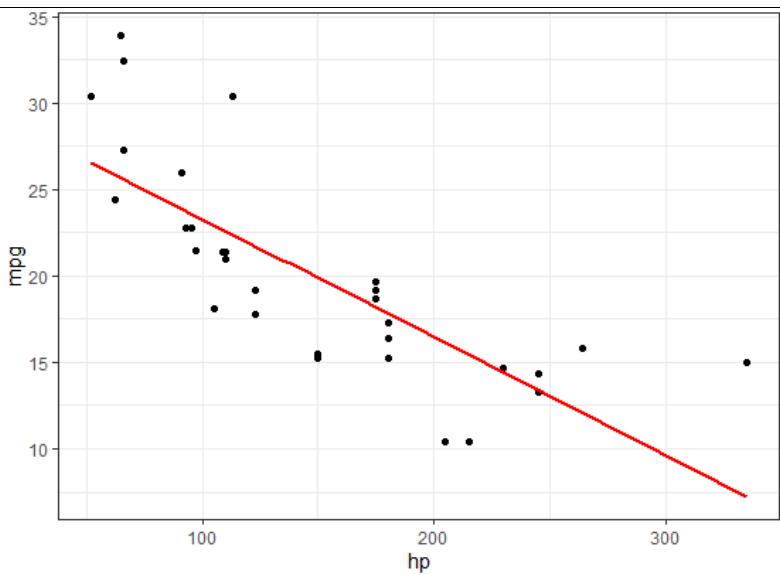
Residuals:
    Min      1Q  Median      3Q     Max 
-5.7121 -2.1122 -0.8854  1.5819  8.2360 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 30.09886   1.63392  18.421 < 2e-16 ***
hp         -0.06823   0.01012  -6.742 1.79e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.863 on 30 degrees of freedom
Multiple R-squared:  0.6024, Adjusted R-squared:  0.5892 
F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07

ggplot(mtcars, aes(x = hp, y = mpg)) +
  geom_point() +
  stat_smooth(method = 'lm', formula = y ~ x, color = 'red', se = FALSE) +
  theme_bw()

```



```

#-----
#Polynomial regression regression
#-----
fit <- lm(data = mtcars, formula = mpg ~ hp + I(hp^2))
summary(fit) #R-squared : 0.7561, Residual standard error: 3.077
Call:
lm(formula = mpg ~ hp + I(hp^2), data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.5512 -1.6027 -0.6977  1.5509  8.7213 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 4.041e+01 2.741e+00 14.744 5.23e-15 ***
hp         -2.133e-01 3.488e-02 -6.115 1.16e-06 ***
I(hp^2)      4.208e-04 9.844e-05  4.275 0.000189 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

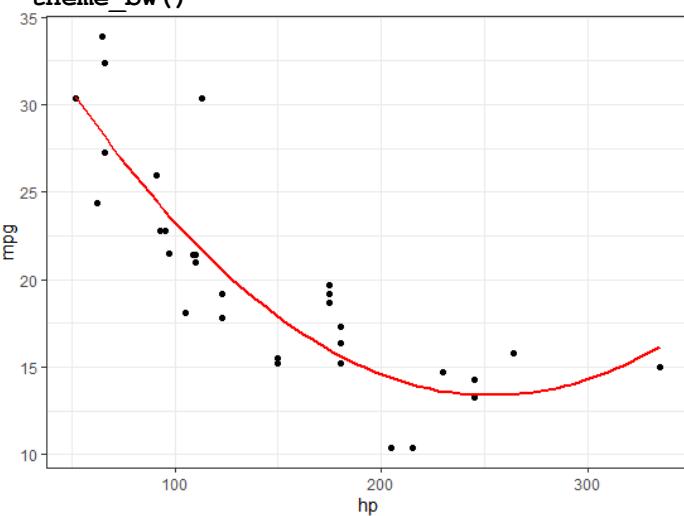
Residual standard error: 3.077 on 29 degrees of freedom
Multiple R-squared:  0.7561, Adjusted R-squared:  0.7393 
F-statistic: 44.95 on 2 and 29 DF,  p-value: 1.301e-09

```

```

ggplot(mtcars, aes(x = hp, y = mpg)) +
  geom_point() +
  stat_smooth(method = 'lm', formula = y ~ x + I(x^2), color = 'red', se =
FALSE) +
  theme_bw()

```



## 10.4 Regression with interaction term

### E037-regression\_with\_interaction.R

```
mtcars <- datasets::mtcars

#generating a new interaction term hp * wt
mtcars$hp_wt <- mtcars$hp * mtcars$wt

fit <- lm(mpg ~ hp + wt + hp_wt, data=mtcars)
summary(fit)

Call:
lm(formula = mpg ~ hp + wt + hp_wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.0632 -1.6491 -0.7362  1.4211  4.5513 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 49.80842   3.60516 13.816 5.01e-14 ***
hp          -0.12010   0.02470 -4.863 4.04e-05 ***  
wt          -8.21662   1.26971 -6.471 5.20e-07 ***  
hp_wt        0.02785   0.00742  3.753 0.000811 *** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.153 on 28 degrees of freedom
Multiple R-squared:  0.8848, Adjusted R-squared:  0.8724 
F-statistic: 71.66 on 3 and 28 DF,  p-value: 2.981e-13

#OR

fit <- lm(mpg ~ hp + wt + hp:wt, data=mtcars)
summary(fit)

Call:
lm(formula = mpg ~ hp + wt + hp:wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.0632 -1.6491 -0.7362  1.4211  4.5513 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 49.80842   3.60516 13.816 5.01e-14 ***
hp          -0.12010   0.02470 -4.863 4.04e-05 ***  
wt          -8.21662   1.26971 -6.471 5.20e-07 ***  
hp:wt        0.02785   0.00742  3.753 0.000811 *** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.153 on 28 degrees of freedom
Multiple R-squared:  0.8848, Adjusted R-squared:  0.8724 
F-statistic: 71.66 on 3 and 28 DF,  p-value: 2.981e-13

#* -----
## A significant coefficient of interaction term indicates that
## the relationship between mpg and hp varies by wt. Similarly,
## the relationship between mpg and wt varies by hp.
## -----



# d(mpg)/d(hp) = - 0.12010 + 0.02785 * wt
wt = 1
print(- 0.12010 + 0.02785 * wt) #-0.09225

wt = 2
print(- 0.12010 + 0.02785 * wt) #-0.0644

wt = 3
print(- 0.12010 + 0.02785 * wt) #-0.03655

# d(mpg)/d(wt) = - 8.21662 + 0.02785 * hp
hp = 100
```

```

print(- 8.21662 + 0.02785 * hp) #-5.43162
hp = 150
print(- 8.21662 + 0.02785 * hp) #-4.03912

hp = 200
print(- 8.21662 + 0.02785 * hp) #-2.64662

```

## 10.5 Logarithmic regression

Model	Equation	Interpretation of $\beta_1$
Log-Log	$\log(y) = \beta_0 + \beta_1 \log(x)$	Elasticity: 1% change in $x$ leads to $\beta_1$ % change in $y$ .
Log-Linear	$\log(y) = \beta_0 + \beta_1 x$	Semi-elasticity: 1-unit change in $x$ leads to $(\exp(\beta_1) - 1) \times 100\%$ change in $y$ .
Linear-Log	$y = \beta_0 + \beta_1 \log(x)$	1% change in $x$ leads to $\beta_1/100$ unit change in $y$ .
Linear-Linear	$y = \beta_0 + \beta_1 x$	1-unit change in $x$ leads to $\beta_1$ unit change in $y$ .

### E038-logarithmic\_regression.R

```

#-----
# Log-Log Regression
#-----
# Load data
mtcars <- datasets::mtcars

# Log-log regression
model_loglog <- lm(log(mpg) ~ log(disp), data = mtcars)
summary(model_loglog)
Call:
lm(formula = log(mpg) ~ log(disp), data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.22758 -0.08874 -0.00791  0.07970  0.32143 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 5.38097   0.20803  25.87 < 2e-16 ***
log(disp)   -0.45857   0.03913 -11.72 1.01e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1282 on 30 degrees of freedom
Multiple R-squared:  0.8207,    Adjusted R-squared:  0.8148 
F-statistic: 137.3 on 1 and 30 DF,  p-value: 1.006e-12

# A 1% increase in Displacement (cu.in.) reduces Miles/(US) gallon by
# ~0.46%.
#-----
# Log-Linear Regression
#-----
# Log-linear regression
model_loglin <- lm(log(mpg) ~ hp, data = mtcars)
summary(model_loglin)

```

```

Call:
lm(formula = log(mpg) ~ hp, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.41577 -0.06583 -0.01737  0.09827  0.39621 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 3.4604669  0.0785838 44.035 < 2e-16 ***  
hp          -0.0034287  0.0004867 -7.045 7.85e-08 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 0.1858 on 30 degrees of freedom
Multiple R-squared:  0.6233, Adjusted R-squared:  0.6107 
F-statistic: 49.63 on 1 and 30 DF,  p-value: 7.853e-08

# A 1-unit increase in horsepower reduces MPG by ~0.34% ( $\exp(-0.0034287)$  - 1
# ≈ -0.003422829).

#-----
# Linear-Log Regression
#-----
# Load data
trees <- datasets::trees

# Linear-log regression
model_linlog <- lm(Volume ~ log(Girth), data = trees)
summary(model_linlog)
Call:
lm(formula = Volume ~ log(Girth), data = trees)

Residuals:
    Min      1Q  Median      3Q     Max 
-9.7246 -3.5312 -0.9174  3.2154 15.8780 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -138.973    11.439 -12.15 6.71e-13 ***  
log(Girth)   66.141     4.455  14.85 4.38e-15 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 5.701 on 29 degrees of freedom
Multiple R-squared:  0.8837, Adjusted R-squared:  0.8797 
F-statistic: 220.4 on 1 and 29 DF,  p-value: 4.381e-15

# A 1% increase in girth increases volume by ~ 0.66 units (66.141 / 100).

```

## Session 11: Logistic regression

### 11.1. Logistic regression

Logistic regression is useful when you're predicting a binary outcome from a set of continuous and/or categorical predictor variables.

#### E039-logistic\_regression.R

```

# Load necessary libraries
library(haven) # For reading SPSS files
library(dplyr) # For data manipulation
library(margins) # For calculating marginal effects

# Import SPSS file from the URL
data <- read_spss('data/010-hh.sav')

# Dropping missing values in HHSEX
data <- data %>% filter(!is.na(HHSEX))

# Creating new variables
data <- data %>%
  mutate(
    hh_size = HH48, # HH member size variable
    urb_rur = factor(HH6), # 1=Urban 2=Rural
    province = factor(HH7), # Province number

```

```

hhsex = factor(HHSEX) # 1=Male 2=Female
)

#setting 1=Urban as reference/base
data$urb_rur <- relevel(data$urb_rur, ref = '1')

#setting 2=Female as reference/base
data$hhsex <- relevel(data$hhsex, ref = '2')

#setting province 3 as base category/reference level
data$province <- relevel(data$province, ref = '3')

#-----
# Running logistic regression
#-----

logit_model <- glm(hhsex ~ hh_size + urb_rur + province,
                     data = data, family = binomial(link = "logit"))
summary(logit_model)
Call:
glm(formula = hhsex ~ hh_size + urb_rur + province, family = binomial(link = "logit"),
     data = data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.39332   0.06488 -6.062 1.35e-09 ***
hh_size      0.33308   0.01274 26.145 < 2e-16 ***
urb_rur2     0.16929   0.04273  3.962 7.44e-05 ***
province1    0.24989   0.07272  3.436 0.000590 ***
province2    0.46344   0.07949  5.830 5.53e-09 ***
province4    -0.47811   0.06912 -6.917 4.61e-12 ***
province5    -0.28721   0.06988 -4.110 3.95e-05 ***
province6    -0.22815   0.07692 -2.966 0.003017 **
province7    -0.25820   0.07476 -3.454 0.000553 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 14830  on 12654  degrees of freedom
Residual deviance: 13722  on 12646  degrees of freedom
AIC: 13740

Number of Fisher Scoring iterations: 4

# Calculating marginal effects for logistic regression
logit_margins <- margins(logit_model)
summary(logit_margins)
  factor      AME       SE      z      p    lower   upper
  hh_size  0.0606  0.0021 28.5476 0.0000  0.0564  0.0647
  province1 0.0424  0.0122  3.4806 0.0005  0.0185  0.0663
  province2 0.0747  0.0123  6.0485 0.0000  0.0505  0.0989
  province4 -0.0936 0.0137 -6.8339 0.0000 -0.1205 -0.0668
  province5 -0.0545 0.0134 -4.0804 0.0000 -0.0806 -0.0283
  province6 -0.0428 0.0146 -2.9299 0.0034 -0.0715 -0.0142
  province7 -0.0487 0.0143 -3.4141 0.0006 -0.0767 -0.0208
  urb_rur2  0.0307  0.0077  3.9854 0.0001  0.0156  0.0457
  ...

#-----
# Running probit regression
#-----

probit_model <- glm(hhsex ~ hh_size + urb_rur + province,
                     data = data, family = binomial(link = "probit"))
summary(probit_model)

```

```

Call:
glm(formula = hhsex ~ hh_size + urb_rur + province, family = binomial(link = "probit"),
     data = data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.184775  0.038365 -4.816 1.46e-06 ***
hh_size      0.188169  0.007198  26.142 < 2e-16 ***
urb_rur2    0.097061  0.025233   3.847 0.000120 ***
province1   0.156245  0.042600   3.668 0.000245 ***
province2   0.267450  0.045381   5.893 3.78e-09 ***
province4   -0.291513  0.041813  -6.972 3.13e-12 ***
province5   -0.165594  0.041717  -3.969 7.20e-05 ***
province6   -0.125738  0.045751  -2.748 0.005991 **
province7   -0.151142  0.044465  -3.399 0.000676 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 14830  on 12654  degrees of freedom
Residual deviance: 13737  on 12646  degrees of freedom
AIC: 13755

Number of Fisher Scoring iterations: 4

# Calculating marginal effects for probit regression
probit_margins <- margins(probit_model)
summary(probit_margins)

  factor      AME      SE      z      p    lower    upper
hh_size  0.0579 0.0021 28.1014 0.0000  0.0538  0.0619
province1 0.0453 0.0122 3.7069 0.0002  0.0214  0.0693
province2 0.0746 0.0123 6.0510 0.0000  0.0505  0.0988
province4 -0.0959 0.0139 -6.8939 0.0000 -0.1231 -0.0686
province5 -0.0529 0.0134 -3.9466 0.0001 -0.0791 -0.0266
province6 -0.0397 0.0146 -2.7226 0.0065 -0.0683 -0.0111
province7 -0.0481 0.0143 -3.3679 0.0008 -0.0761 -0.0201
urb_rur2  0.0298 0.0077  3.8642 0.0001  0.0147  0.0448

```

### Task 8:

Using NMICS6 data (011-Affairs.RData), complete the following tasks.

- Load the **011-Affairs.RData**
- Tabulate the frequency of **affairs** variable from **Affairs** dataframe.
- Create a variable **ynaffairs** in **Affairs** dataframe such that the variable takes value 0 if no affairs and 1 if the person is involved in affairs.
- Set **ynaffairs** and **rating** variables as factor variables.
- Set '0' as reference for **ynaffairs** variable, '5' for **rating**, 'no' for **children**, and 'female' for **gender** variables.
- Fit a logistic regression model with **ynaffairs** as dependent variable and **gender, age, yearsmarried, children, rating** as independent variable.
- Calculate average marginal effect for each variables using the **margins()** function.

```

library(dplyr)
library(margins)

load('data/011-Affairs.RData')
table(Affairs$affairs)
  0   1   2   3   7  12
451  34  17  19  42  38
.

Affairs <- Affairs %>% mutate(ynaffair = case_when(affairs > 0 ~ 1, TRUE ~ 0),
                                ynaffair = factor(ynaffair),
                                rating = factor(rating))
table(Affairs$ynaffair)
  0   1
451 150
.

#setting 0 : No-Affairs as base/reference
Affairs$ynaffair <- relevel(Affairs$ynaffair, ref = '0')

```

```

#setting 5 : Very happy as base/reference
# 1 = very unhappy, 2 = somewhat unhappy, 3 = average, 4 = happier than
average, 5 = very happy.
Affairs$rating <- relevel(Affairs$rating, ref = '5')

#setting no children as base/reference
Affairs$children <- relevel(Affairs$children, ref = 'no')

#setting female as base/reference
Affairs$gender <- relevel(Affairs$gender, ref = 'female')

fit <- glm(ynaffair ~ gender
           + age
           + yearsmarried
           + children
           + rating,
           data=Affairs,
           family = binomial(link = "logit"))

summary(fit)
Call:
glm(formula = ynaffair ~ gender + age + yearsmarried + children +
    rating, family = binomial(link = "logit"), data = Affairs)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.36331   0.45275 -3.011  0.00260 **
gendermale   0.38018   0.20644  1.842  0.06553 .
age          -0.04432   0.01789 -2.477  0.01324 *
yearsmarried 0.08127   0.03154  2.577  0.00997 **
childrenyes  0.32477   0.28716  1.131  0.25807
rating1       1.66252   0.55213  3.011  0.00260 **
rating2       1.64220   0.31872  5.152  2.57e-07 ***
rating3       0.76132   0.30044  2.534  0.01128 *
rating4       0.52336   0.25641  2.041  0.04124 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 675.38 on 600 degrees of freedom
Residual deviance: 622.26 on 592 degrees of freedom
AIC: 640.26

Number of Fisher Scoring iterations: 4

summary(margins(fit))
  factor     AME      SE      z      p    lower    upper
  age -0.0075 0.0030 -2.5155 0.0119 -0.0134 -0.0017
  childrenyes 0.0537 0.0459  1.1698 0.2421 -0.0363 0.1437
  gendermale  0.0648 0.0350  1.8511 0.0642 -0.0038 0.1334
  rating1    0.3274 0.1273  2.5714 0.0101  0.0778 0.5769
  rating2    0.3225 0.0666  4.8427 0.0000  0.1920 0.4530
  rating3    0.1248 0.0522  2.3916 0.0168  0.0225 0.2271
  rating4    0.0803 0.0392  2.0468 0.0407  0.0034 0.1572
  yearsmarried 0.0138 0.0053  2.6201 0.0088  0.0035 0.0242

```

## Session 12: R notebook with R-markdown

## Session 11: Time-series analysis

### 11.1. Stationarity concept

- Stationarity refers to a time series whose statistical properties, such as mean, variance, and autocorrelation, remain constant over time.
- Non-stationary series are prone to spurious relationships.

### 11.2. Spurious relationships

#### E040-spurious\_regression.R

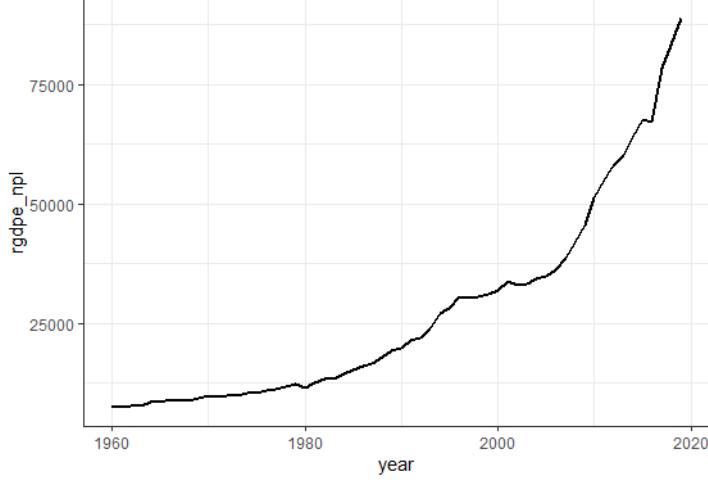
```
library(haven)
library(dplyr)
library(tseries)
library(ggplot2)

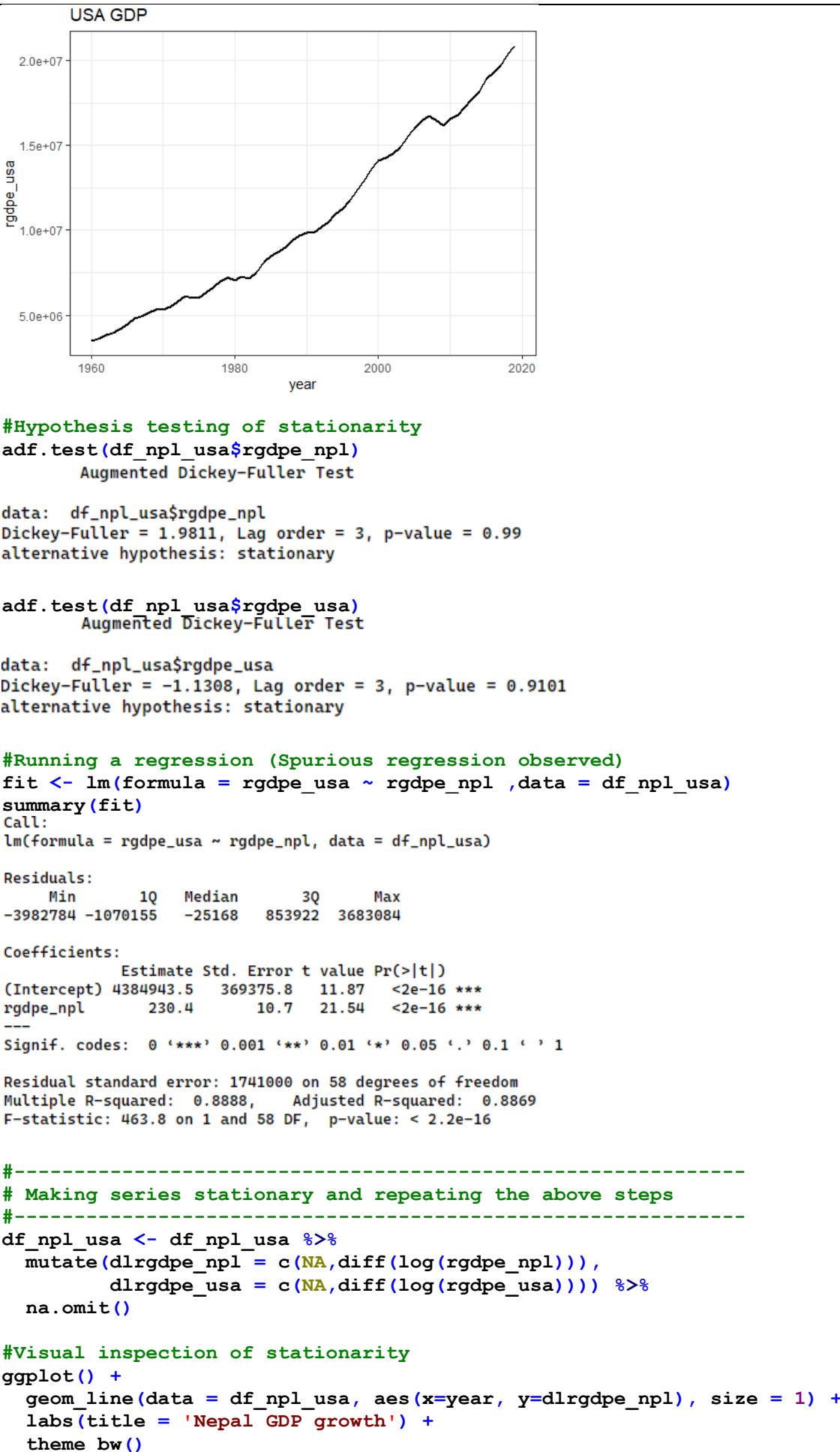
df <- read_dta('data/012-pwt1001.dta')

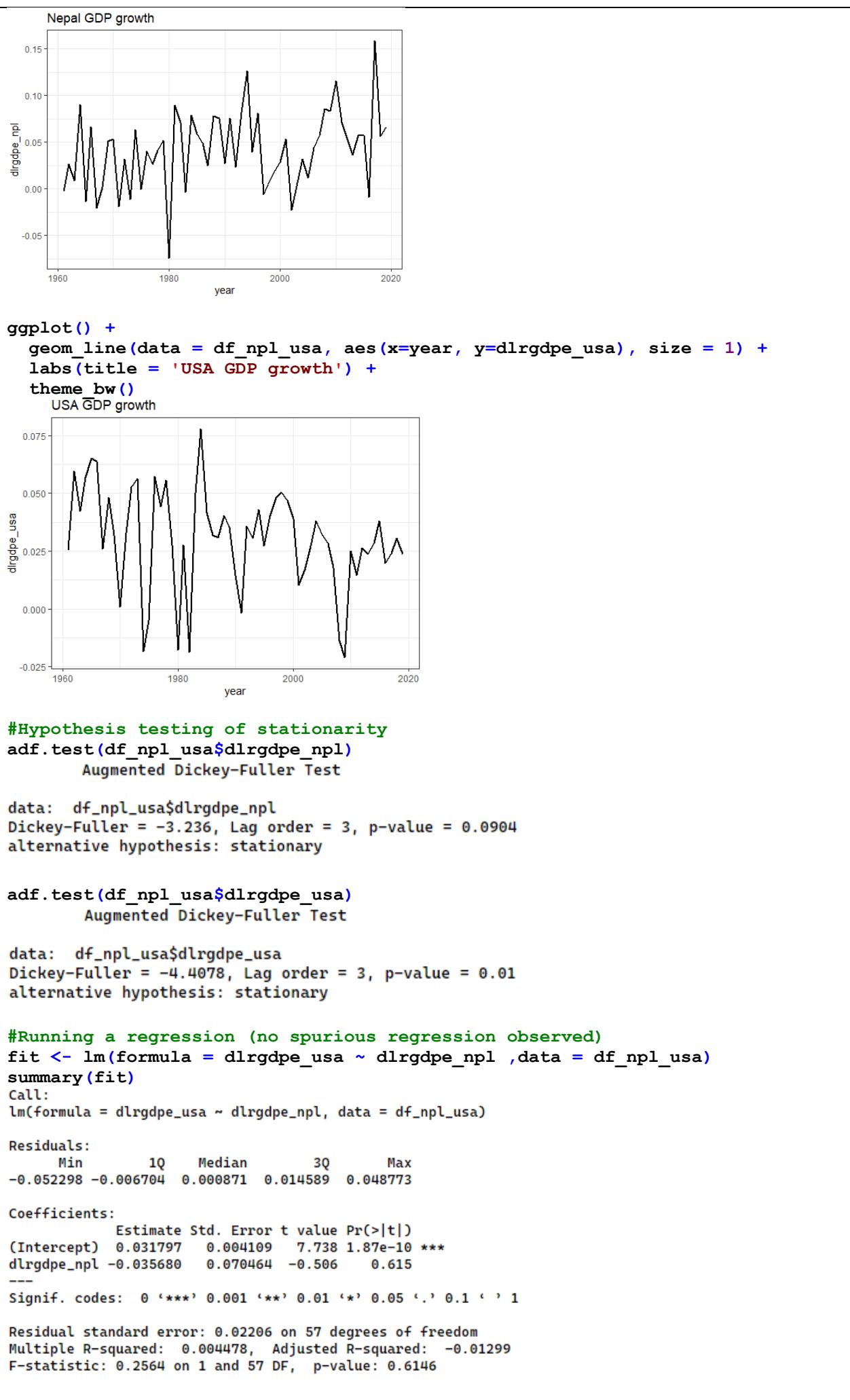
#keeping real GDP of Nepal from 1960 onwards
npl <- df %>%
  filter(countrycode == 'NPL' & year >= 1960) %>%
  select(year, rgdpe) %>%
  rename(rgdpe_npl = rgdpe)

#keeping real GDP of USA from 1960 onwards
usa <- df %>%
  filter(countrycode == 'USA' & year >= 1960) %>%
  select(year, rgdpe) %>%
  rename(rgdpe_usa = rgdpe)

#joining Nepal and USA data into one dataframe
df_npl_usa <- full_join(npl, usa, by = 'year')

#Visual inspection of stationarity
ggplot() +
  geom_line(data = df_npl_usa, aes(x=year, y=rgdpe_npl), size = 1) +
  labs(title = 'Nepal GDP') +
  theme_bw()
  Nepal GDP

ggplot() +
  geom_line(data = df_npl_usa, aes(x=year, y=rgdpe_usa), size = 1) +
  labs(title = 'USA GDP') +
  theme_bw()
```





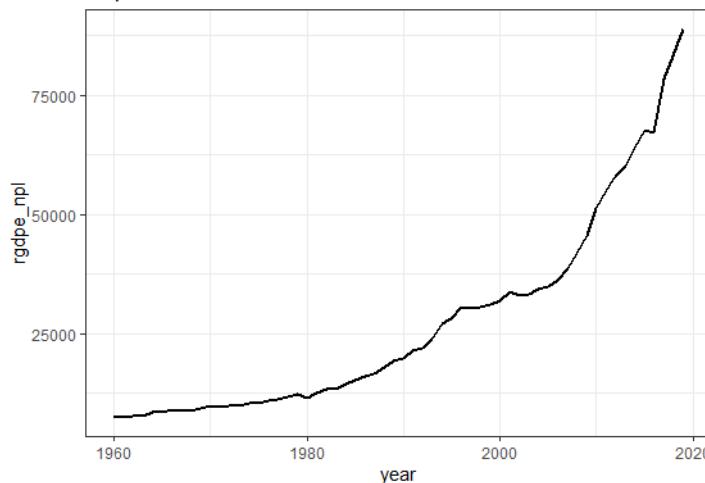
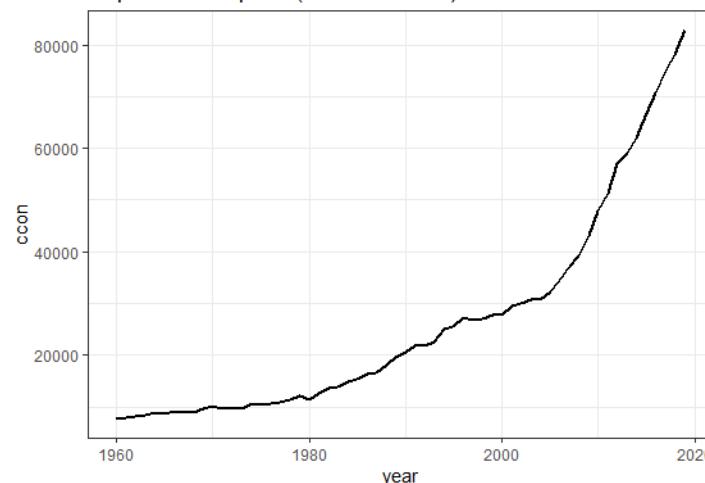
### 11.3. True relationships

#### E041-actual\_relationship.R

```
library(haven)
library(dplyr)
library(tseries)
library(ggplot2)

df <- read_dta('data/012-pwt1001.dta')

df <- filter(df, countrycode == 'NPL' & year >= 1960) %>% select(year,
rgdpe, ccon)

#Visual inspection of stationarity
ggplot() +
  geom_line(data = df, aes(x=year, y=rgdpe), size = 1) +
  labs(title = 'Nepal GDP') +
  theme_bw()
  Nepal GDP
  
  
ggplot() +
  geom_line(data = df, aes(x=year, y=ccon), size = 1) +
  labs(title = 'Nepal Consumption (Private + Govt)') +
  theme_bw()
  Nepal Consumption (Private + Govt)
  
  
#Hypothesis testing of stationarity
adf.test(df$rgdpe)
  Augmented Dickey-Fuller Test

data: df$rgdpe
Dickey-Fuller = 1.9811, Lag order = 3, p-value = 0.99
alternative hypothesis: stationary

adf.test(df$ccon)
```

```

Augmented Dickey-Fuller Test

data: df$ccon
Dickey-Fuller = 0.87741, Lag order = 3, p-value = 0.99
alternative hypothesis: stationary

#Running a regression
fit <- lm(formula = rgdpe ~ ccon ,data = df)
summary(fit)
Call:
lm(formula = rgdpe ~ ccon, data = df)

Residuals:
    Min      1Q  Median      3Q     Max 
-7032.0 -695.9 -295.4 1007.2 3166.8 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -157.77337  331.29087 -0.476   0.636    
ccon         1.04900   0.01004 104.504  <2e-16 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1554 on 58 degrees of freedom
Multiple R-squared:  0.9947,    Adjusted R-squared:  0.9946 
F-statistic: 1.092e+04 on 1 and 58 DF,  p-value: < 2.2e-16

#-----#
# Making series stationary and repeating the above steps
#-----#
df <- df %>%
  mutate(dlrgdpe = c(NA,diff(log(rgdpe))), 
        dlccon = c(NA,diff(log(ccon)))) %>%
  na.omit()

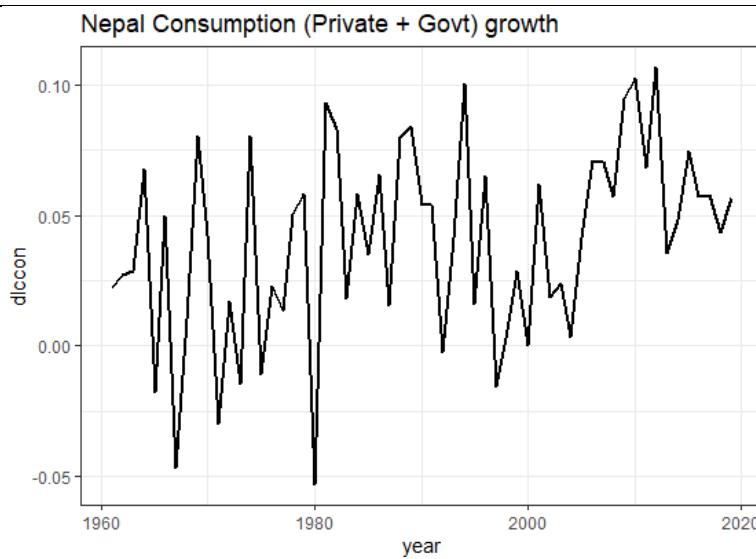
#Visual inspection of stationarity
ggplot() +
  geom_line(data = df, aes(x=year, y=dlrgdpe), size = 1) +
  labs(title = 'Nepal GDP growth') +
  theme_bw()
  Nepal GDP growth


```

```

ggplot() +
  geom_line(data = df, aes(x=year, y=dlccon), size = 1) +
  labs(title = 'Nepal Consumption (Private + Govt) growth') +
  theme_bw()

```



```
#Hypothesis testing of stationarity
adf.test(df_npl_usa$dlrgdpe_npl)
  Augmented Dickey-Fuller Test

data: df_npl_usa$dlrgdpe_npl
Dickey-Fuller = -3.236, Lag order = 3, p-value = 0.0904
alternative hypothesis: stationary

adf.test(df_npl_usa$dlrgdpe_usa)
  Augmented Dickey-Fuller Test

data: df_npl_usa$dlrgdpe_usa
Dickey-Fuller = -4.4078, Lag order = 3, p-value = 0.01
alternative hypothesis: stationary

#Running a regression
fit <- lm(formula = dlrgdpe ~ dlccon ,data = df)
summary(fit)

Call:
lm(formula = dlrgdpe ~ dlccon, data = df)

Residuals:
    Min      1Q      Median      3Q      Max 
-0.065718 -0.013749  0.000986  0.015554  0.101673 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 0.006254   0.004832   1.294   0.201    
dlccon       0.880921   0.088548   9.949 4.54e-14 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02507 on 57 degrees of freedom
Multiple R-squared:  0.6346,    Adjusted R-squared:  0.6281 
F-statistic: 98.97 on 1 and 57 DF,  p-value: 4.542e-14
```

## Session 12: Stargazer for reporting regression results and the project work

### 12.1. stargazer

```
E042-stargazer.R
mtcars <- datasets::mtcars

model1 <- lm(mpg ~ hp, data = mtcars)
model2 <- lm(mpg ~ hp + drat, data = mtcars)
model3 <- lm(mpg ~ hp + drat + cyl + wt, data = mtcars)
model4 <- lm(hp ~ disp + carb, data = mtcars)

library(stargazer)

#descriptive statistics table
stargazer(mtcars, type = 'text')
```

```

=====
Statistic N Mean St. Dev. Min Max
-----
mpg      32 20.091  6.027  10.400 33.900
cyl      32 6.188   1.786    4       8
disp     32 230.722 123.939  71.100 472.000
hp       32 146.688  68.563   52      335
drat     32 3.597   0.535   2.760   4.930
wt       32 3.217   0.978   1.513   5.424
qsec     32 17.849  1.787   14.500 22.900
vs       32 0.438   0.504    0       1
am       32 0.406   0.499    0       1
gear     32 3.688   0.738    3       5
carb     32 2.812   1.615    1       8
-----

#displaying regression models results in a single table
stargazer(model1, model2, model3, model4, type = "text")
=====
Dependent variable:

```

	(1)	mpg (2)	(3)	hp (4)
hp	-0.068*** (0.010)	-0.052*** (0.009)	-0.021 (0.013)	
drat		4.698*** (1.192)	0.818 (1.387)	
cyl			-0.762 (0.635)	
wt			-2.973*** (0.818)	
disp				0.324*** (0.043)
carb				21.999*** (3.298)
Constant	30.099*** (1.634)	10.790** (5.078)	34.496*** (7.441)	9.988 (11.614)
Observations	32	32	32	32
R2	0.602	0.741	0.845	0.852
Adjusted R2	0.589	0.723	0.822	0.842
Residual Std. Error	3.863 (df = 30)	3.170 (df = 29)	2.541 (df = 27)	27.244 (df = 29)
F Statistic	45.460*** (df = 1; 30)	41.522*** (df = 2; 29)	36.839*** (df = 4; 27)	83.665*** (df = 2; 29)

```

=====
Note: *p<0.1; **p<0.05; ***p<0.01

```

```

#defining the covariate and variable labels
stargazer(model1, model2, model3, model4, type = "text",
           digits = 2,
           covariate.labels = c('Gross horsepower (hp)',
                                'Rear axle ratio (drat)',
                                'Number of cylinders (cyl)',
                                'Weight (1000 lbs) (wt)',
                                'Displacement (cu.in.) (disp)',
                                'Number of carburetors (carb)'),
           dep.var.labels = c("Miles/(US) gallon (mpg)", "Gross horsepower
(hp)"),
           notes = "Standard errors are in parentheses.")

#export and save the result as html
stargazer(model1, model2, model3, model4, type = "html", out =
'model_results.html',
           digits = 2,
           covariate.labels = c('Gross horsepower (hp)',
                                'Rear axle ratio (drat)',
                                'Number of cylinders (cyl)',
                                'Weight (1000 lbs) (wt)',
                                'Displacement (cu.in.) (disp)',
                                'Number of carburetors (carb)'),
           dep.var.labels = c("Miles/(US) gallon (mpg)", "Gross horsepower
(hp)"),
           notes = "Standard errors are in parentheses.")

```

	<i>Dependent variable:</i>			
	Miles/(US) gallon (mpg)		Gross horsepower (hp)	
	(1)	(2)	(3)	(4)
Gross horsepower (hp)	-0.07*** (0.01)	-0.05*** (0.01)	-0.02 (0.01)	
Rear axle ratio (dart)		4.70*** (1.19)	0.82 (1.39)	
Number of cylinders (cyl)			-0.76 (0.64)	
Weight (1000 lbs) (wt)			-2.97*** (0.82)	
Displacement (cu.in.) (disp)				0.32*** (0.04)
Number of carburetors (carb)				22.00*** (3.30)
Constant	30.10*** (1.63)	10.79** (5.08)	34.50*** (7.44)	9.99 (11.61)
Observations	32	32	32	32
R <sup>2</sup>	0.60	0.74	0.85	0.85
Adjusted R <sup>2</sup>	0.59	0.72	0.82	0.84
Residual Std. Error	3.86 (df = 30)	3.17 (df = 29)	2.54 (df = 27)	27.24 (df = 29)
F Statistic	45.46*** (df = 1; 30)	41.52*** (df = 2; 29)	36.84*** (df = 4; 27)	83.66*** (df = 2; 29)
<i>Note:</i>	* p<0.1; ** p<0.05; *** p<0.01 Standard errors are in parentheses.			