

Ali

2025-06-24

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

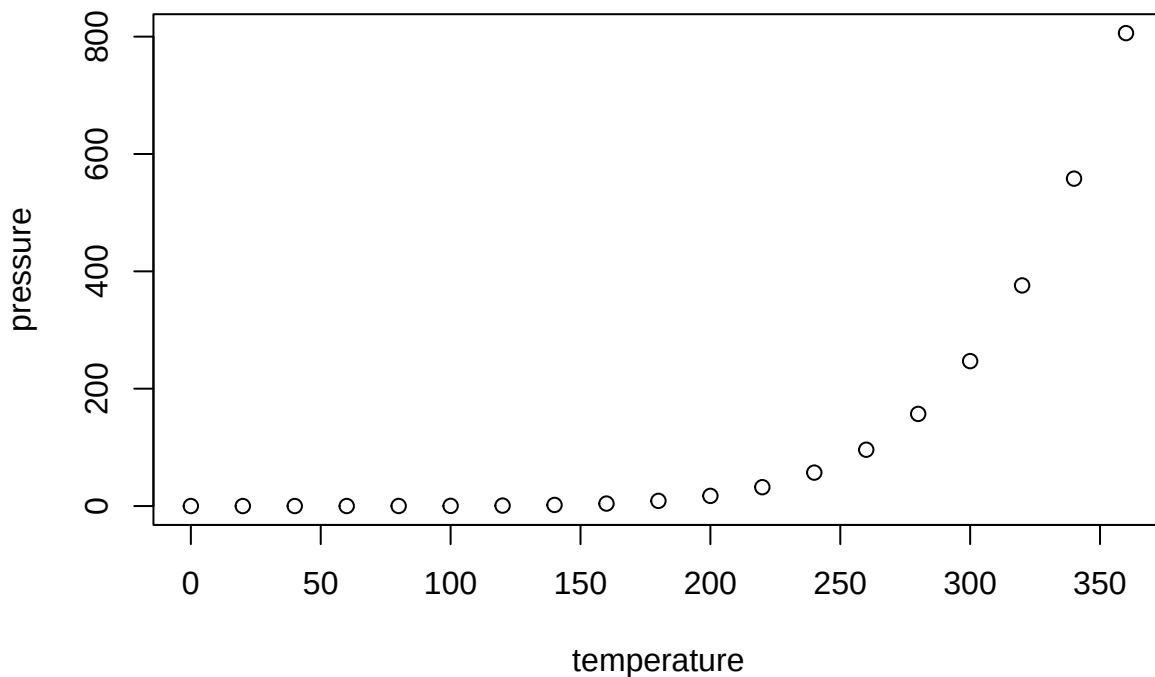
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
##  Min.   : 4.0    Min.   :  2.00
##  1st Qu.:12.0    1st Qu.: 26.00
##  Median :15.0    Median : 36.00
##  Mean   :15.4    Mean   : 42.98
##  3rd Qu.:19.0    3rd Qu.: 56.00
##  Max.   :25.0    Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```

install.packages("gapminder")

## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)

install.packages("dplyr")

## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)

library(gapminder)
library(dplyr)

##
## Attaching package: 'dplyr'

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

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

# Extract life expectancy data
life_data <- gapminder$lifeExp

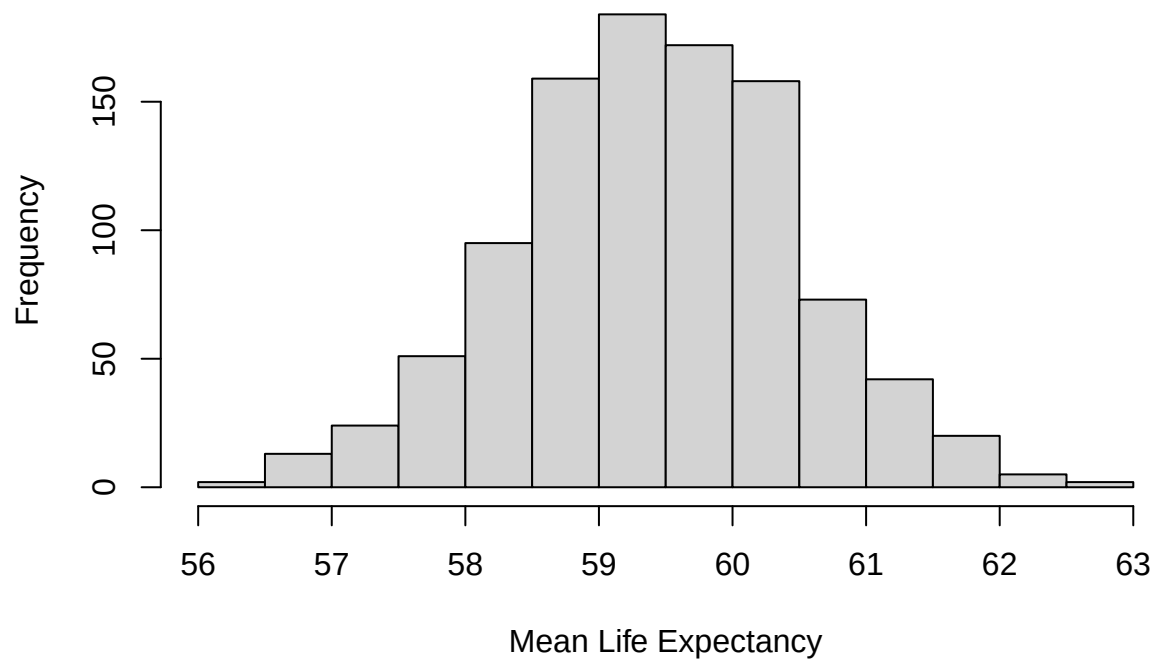
# Generate 1000 sample means
sample_means <- replicate(1000, mean(sample(life_data, size = 150, replace = TRUE)))

# Calculate standard error from the sampling distribution
se <- sd(sample_means)

# Create histogram and store histogram object
hist_obj <- hist(sample_means,
  main = "Sampling Distribution of Sample Mean",
  xlab = "Mean Life Expectancy",
  )

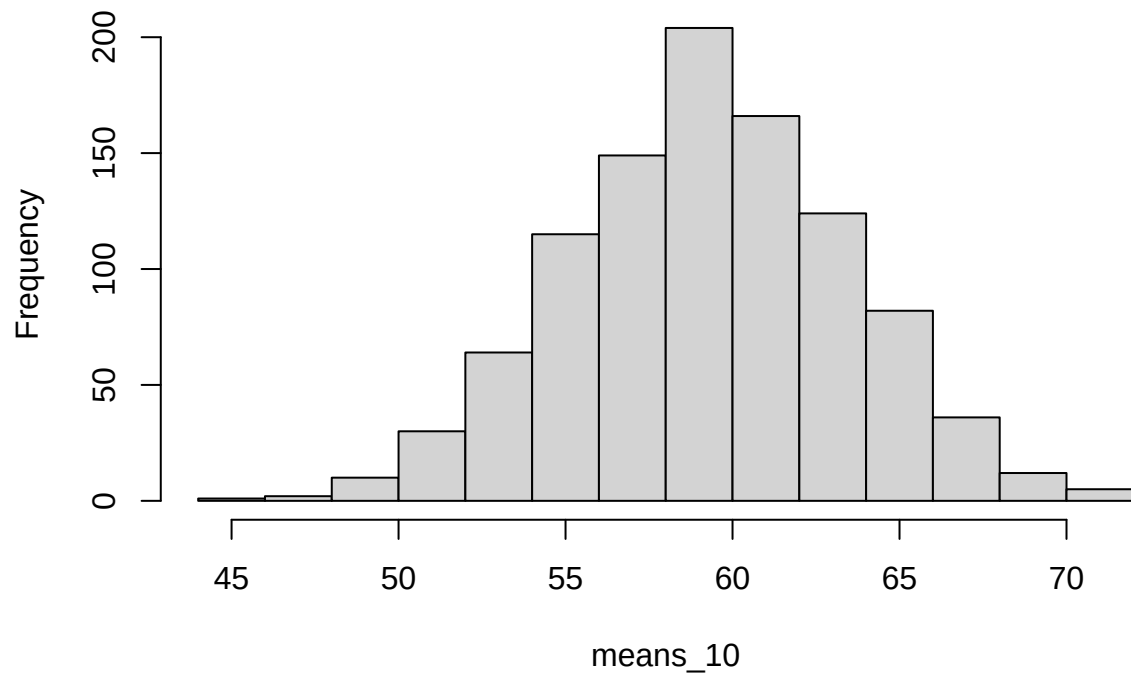
```

Sampling Distribution of Sample Mean



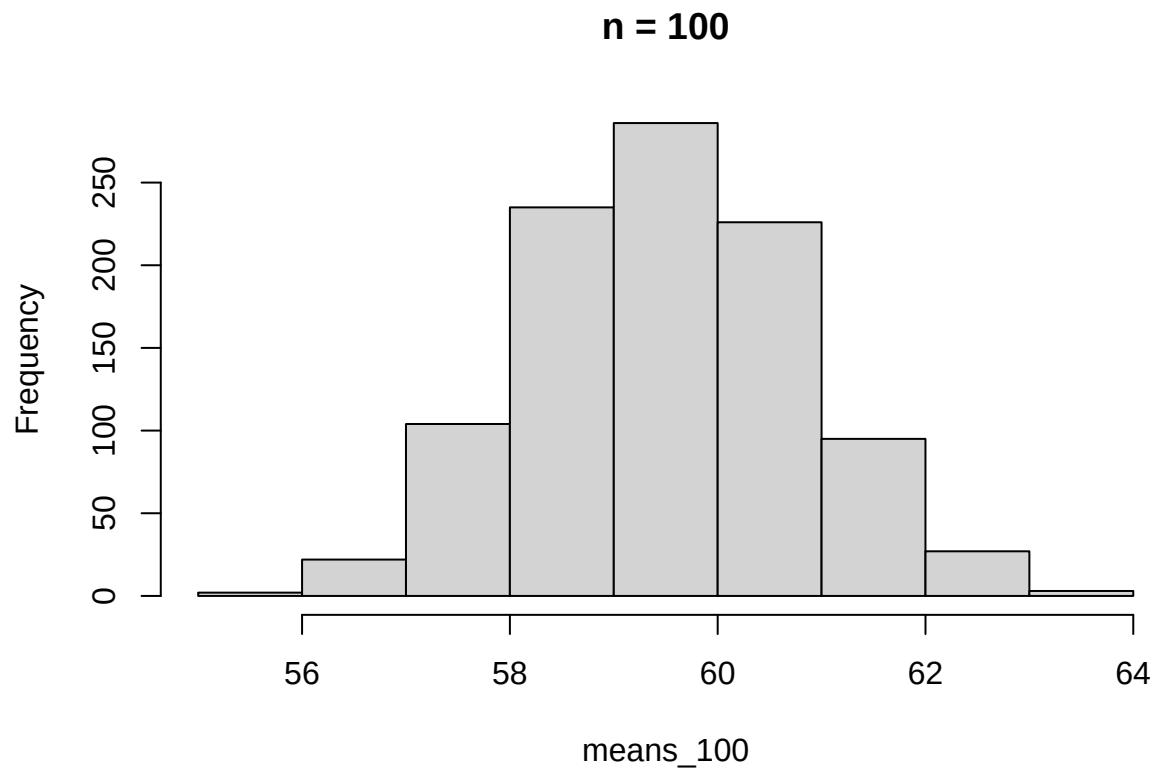
```
# Try with n = 10 and n = 100  
means_10 <- replicate(1000, mean(sample(life_data, 10, replace = TRUE)))  
hist(means_10, main="n = 10")
```

n = 10



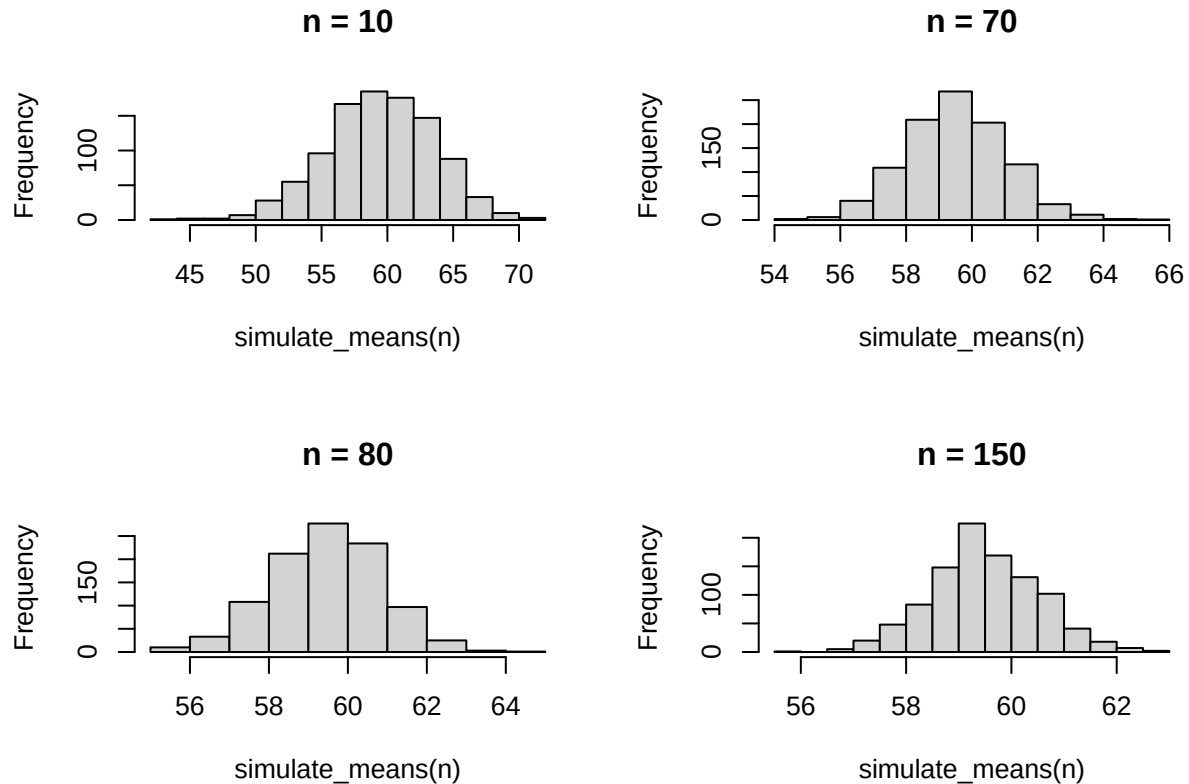
```
means_100 <- replicate(1000, mean(sample(life_data, 100, replace = TRUE)))
```

```
hist(means_100, main="n = 100")
```



```
simulate_means <- function(n, reps = 1000) {  
  replicate(reps, mean(sample(life_data, size = n, replace = TRUE)))  
}
```

```
sample_sizes <- c(10, 70, 80, 150)  
par(mfrow = c(2,2))  
for (n in sample_sizes) {  
  hist(simulate_means(n), main = paste("n =", n))  
}
```



```
# Extract life expectancy data
life_data <- gapminder$lifeExp

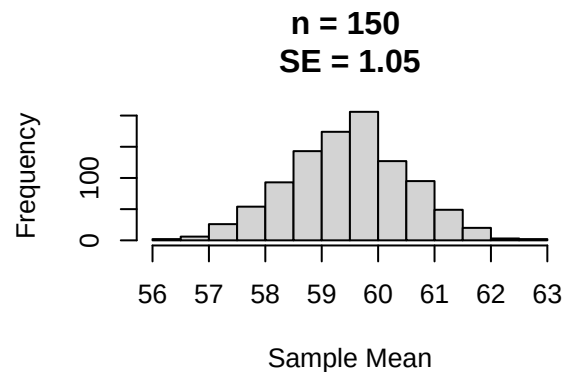
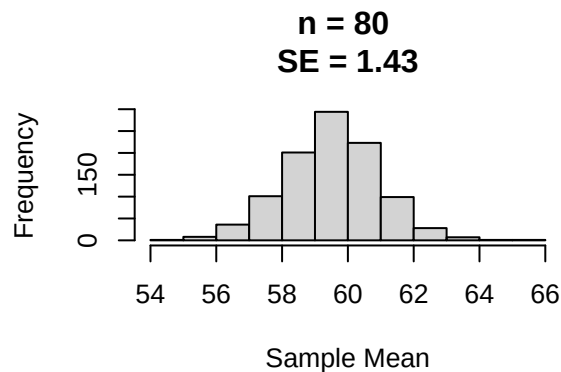
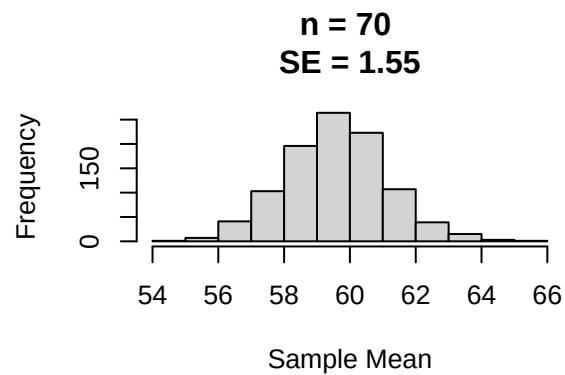
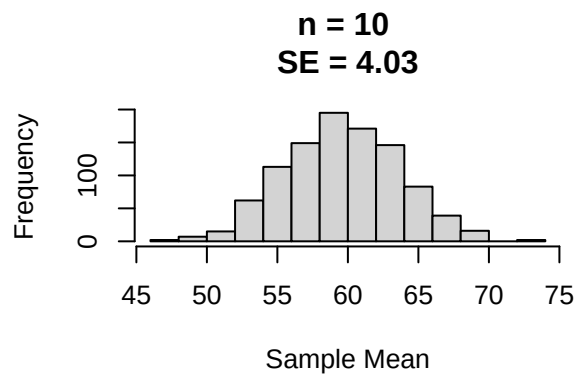
# Function to simulate sample means
simulate_means <- function(n, reps = 1000) {
  replicate(reps, mean(sample(life_data, size = n, replace = TRUE)))
}

# Define sample sizes
sample_sizes <- c(10, 70, 80, 150)

# Set plotting layout: 2 rows, 2 columns
par(mfrow = c(2, 2))

# Loop through each sample size and plot histograms with SE
for (n in sample_sizes) {
  sample_means <- simulate_means(n)
  se <- sd(sample_means) # Calculate standard error

  # Create histogram and store object to get y-axis height
  hist_obj <- hist(sample_means,
    main = paste("n =", n, "\nSE =", round(se, 2)),
    xlab = "Sample Mean",
  )
}
```



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

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```
library(gapminder)
library(ggplot2)
head(gapminder)
```

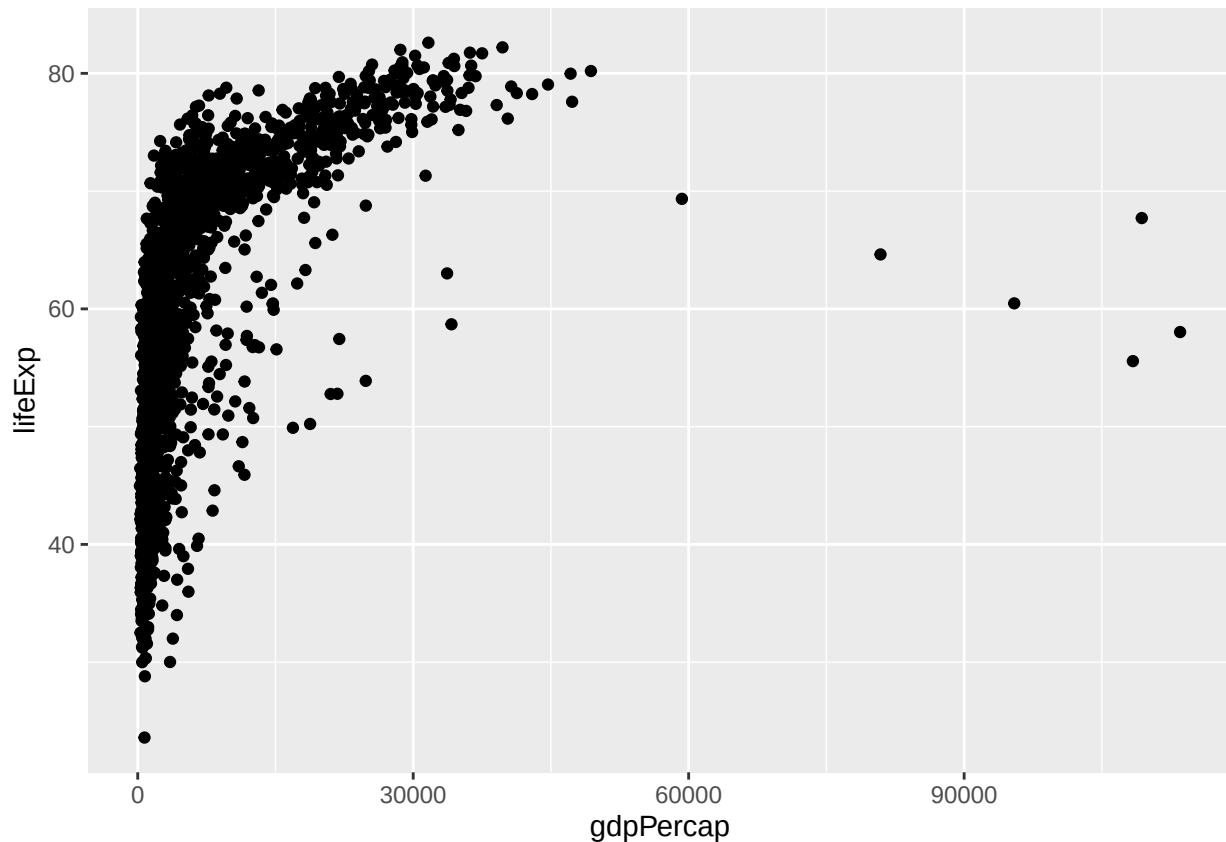
```
## # A tibble: 6 x 6
##   country    continent  year lifeExp      pop gdpPercap
##   <fct>      <fct>    <int> <dbl>    <int>    <dbl>
## 1 Afghanistan Asia      1952  28.8  8425333    779.
## 2 Afghanistan Asia      1957  30.3  9240934    821.
## 3 Afghanistan Asia      1962  32.0 10267083    853.
## 4 Afghanistan Asia      1967  34.0 11537966    836.
## 5 Afghanistan Asia      1972  36.1 13079460    740.
## 6 Afghanistan Asia      1977  38.4 14880372    786.
```

```
summary(gapminder)
```

```
##           country      continent      year      lifeExp
## Afghanistan: 12 Africa :624   Min.   :1952   Min.   :23.60
## Albania      : 12 Americas:300 1st Qu.:1966 1st Qu.:48.20
## Algeria       : 12 Asia   :396 Median :1980 Median :60.71
## Angola        : 12 Europe :360 Mean    :1980 Mean   :59.47
## Argentina     : 12 Oceania: 24 3rd Qu.:1993 3rd Qu.:70.85
## Australia     : 12                Max.   :2007 Max.   :82.60
## (Other)       :1632
##           pop           gdpPercap
```

```
## Min.      :6.001e+04   Min.      : 241.2
## 1st Qu.:2.794e+06   1st Qu.: 1202.1
## Median :7.024e+06   Median : 3531.8
## Mean    :2.960e+07   Mean     : 7215.3
## 3rd Qu.:1.959e+07   3rd Qu.: 9325.5
## Max.    :1.319e+09   Max.     :113523.1
##
```

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) + geom_point()
```



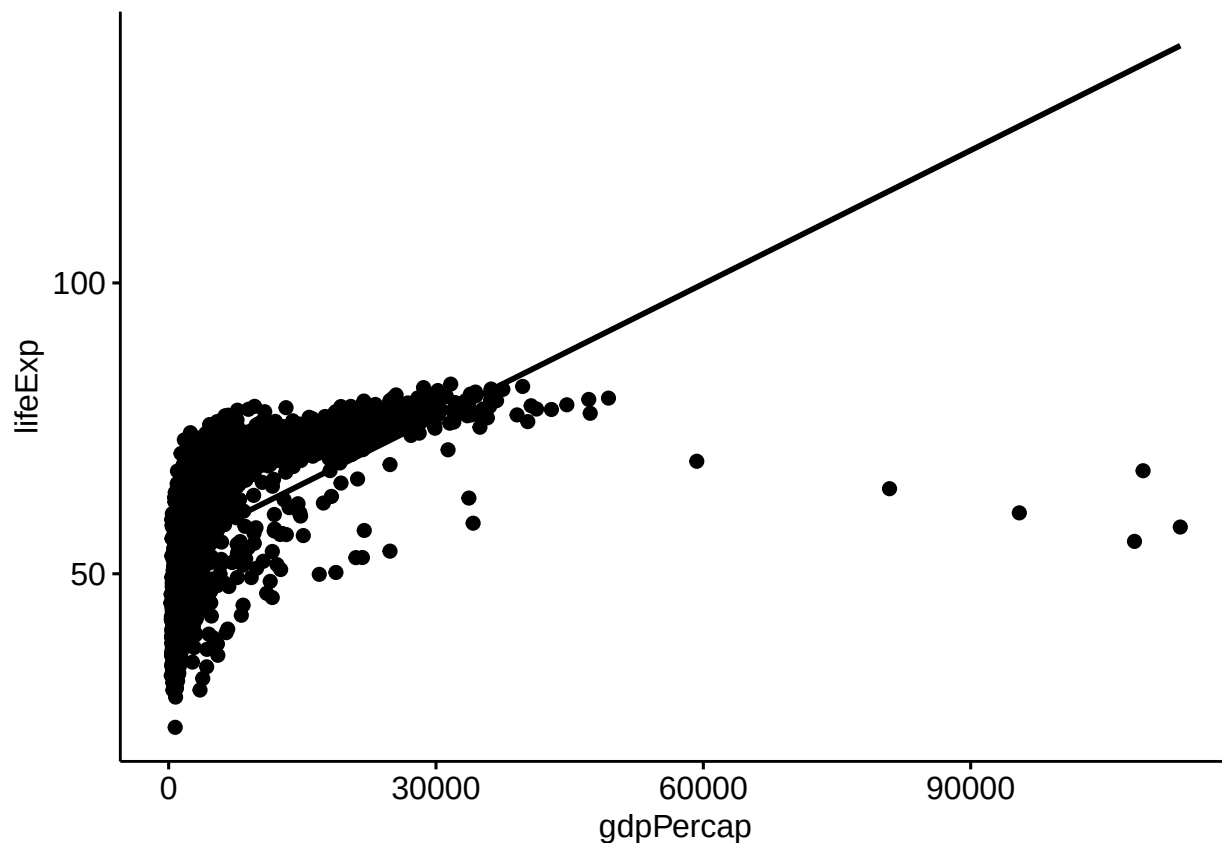
```
# R Code
cor(gapminder$lifeExp, gapminder$gdpPercap, method = "pearson")
```

```
## [1] 0.5837062
```

```
# Visualise
install.packages("ggpubr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```
library(ggpubr)
ggscatter(gapminder, x = "gdpPercap", y = "lifeExp", add = "reg.line")
```



```
# R Code for correlation
cor(gapminder$lifeExp, gapminder$gdpPercap, method = "spearman")
```

```
## [1] 0.8264712
```

```
# Extract variables
```

```
life_exp <- gapminder$lifeExp
```

```
population <- gapminder$pop
```

```
# Pearson correlation
```

```
cor_pearson <- cor(life_exp, population, method = "pearson")
```

```
# Spearman correlation
```

```
cor_spearman <- cor(life_exp, population, method = "spearman")
```

```
# Display results
```

```
cat("Pearson correlation:", round(cor_pearson, 3), "\n")
```

```
## Pearson correlation: 0.065
```

```
cat("Spearman correlation:", round(cor_spearman, 3), "\n")
```

```
## Spearman correlation: 0.181
```

Spearman correlation is better for analyzing the relationship between lifeExp (life expectancy) and pop (population) in the gapminder dataset.