Unit 11 activities

Guilherme Amorim

2024-10-16

Unit 11 - Notes

R

Initial setup

```
# initial setup
library(haven)
library(skimr)
library(tidyverse)
library(here)
```

```
## Warning: package 'here' was built under R version 4.3.3
```

Data load

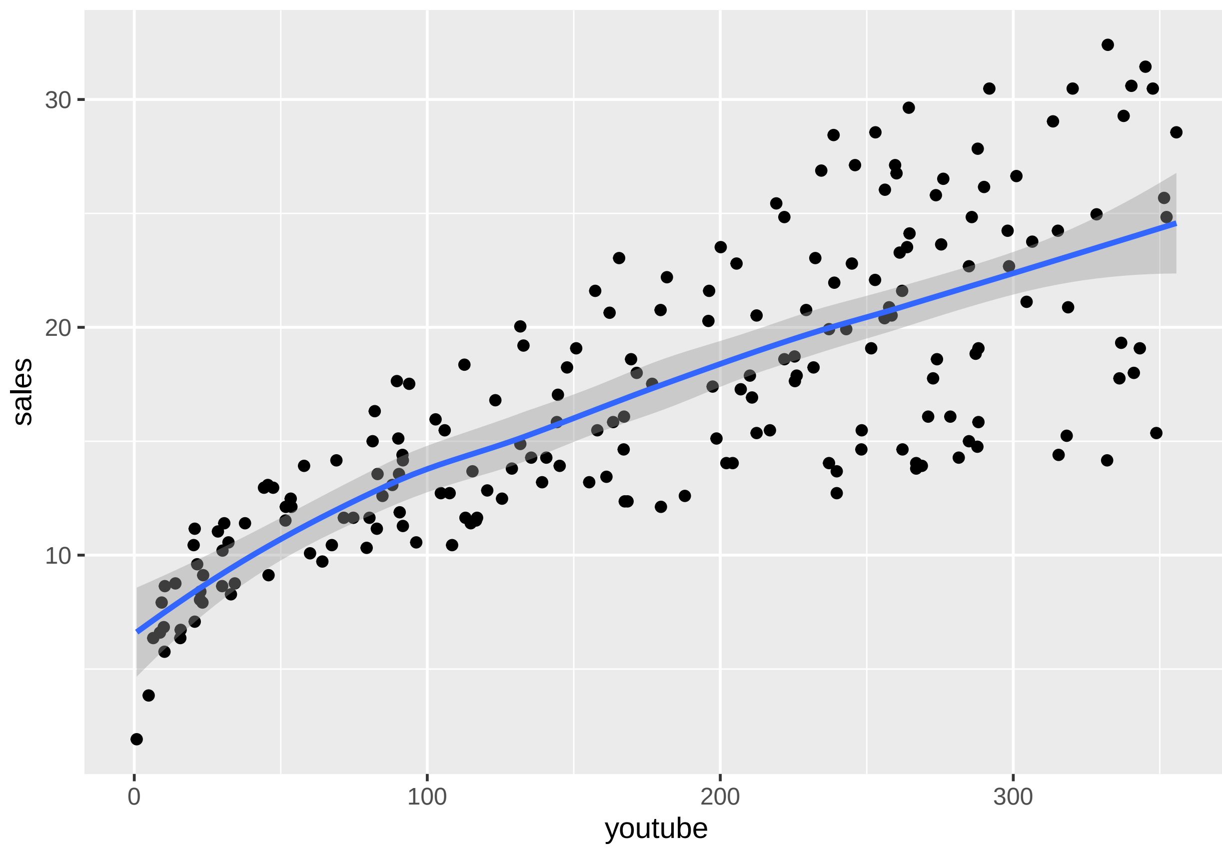
```
load(here("Datasets/marketing.rda"))

write_csv(marketing, here("Datasets/marketing.csv"))
```

Scatter plot and regression line (for youtube sales)

```
marketing%>%
  ggplot(aes(youtube, sales))+
  geom_point()+
  stat_smooth()
```

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



The scatter plot indicates a linear relationship between both variables ### Correlation

```
cor.test(marketing$youtube, marketing$sales)
```

```
##
##  Pearson's product-moment correlation
##
## data:  marketing$youtube and marketing$sales
## t = 17.668, df = 198, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7218201 0.8308014
## sample estimates:
##       cor
## 0.7822244
```

From the correlation coefficient, we can say that there is a strong, positive, statistically significant association between budget spent on Youtube marketing and sales

Linear regression model

```
model<-lm(sales~youtube, marketing)

model
```

```
##
## Call:
## lm(formula = sales ~ youtube, data = marketing)
##
## Coefficients:
## (Intercept)      youtube
##     8.43911      0.04754
```

```r
summary(model)
```

```
##
## Call:
## lm(formula = sales ~ youtube, data = marketing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0632  -2.3454  -0.2295   2.4805   8.6548
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.439112   0.549412   15.36   <2e-16 ***
## youtube     0.047537   0.002691   17.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.91 on 198 degrees of freedom
## Multiple R-squared:  0.6119, Adjusted R-squared:  0.6099
## F-statistic: 312.1 on 1 and 198 DF,  p-value: < 2.2e-16
```

The linear regression model can be expressed as: Sales = 8.43911 + 0.04754 * (Youtube). The p-values for both the intercept and the slope of the predictor variable are <0.001 and therefore statistically significant (i.e. not generated by a random distribution of the underlying variables). The residual standard error is 3.91, which provides a measure of the average absolute error associated with each prediction. The adjusted R-squared is 0.6099, indicating that 60% of the variance in the dependent variable (sales) is explained by the independent variable (Youtube marketing budget). This measure is quite similar to the crude R-squared as there is only one dependent variable.

Python

Initial setup

```python
# first install packages (from terminal)

# pip3 install numpy

# pip3 install pandas

# pip3 install matplotlib

# pip3 install pyreadstat
```

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from scipy import stats

import pyreadr

import sklearn

from sklearn.linear_model import LinearRegression
```

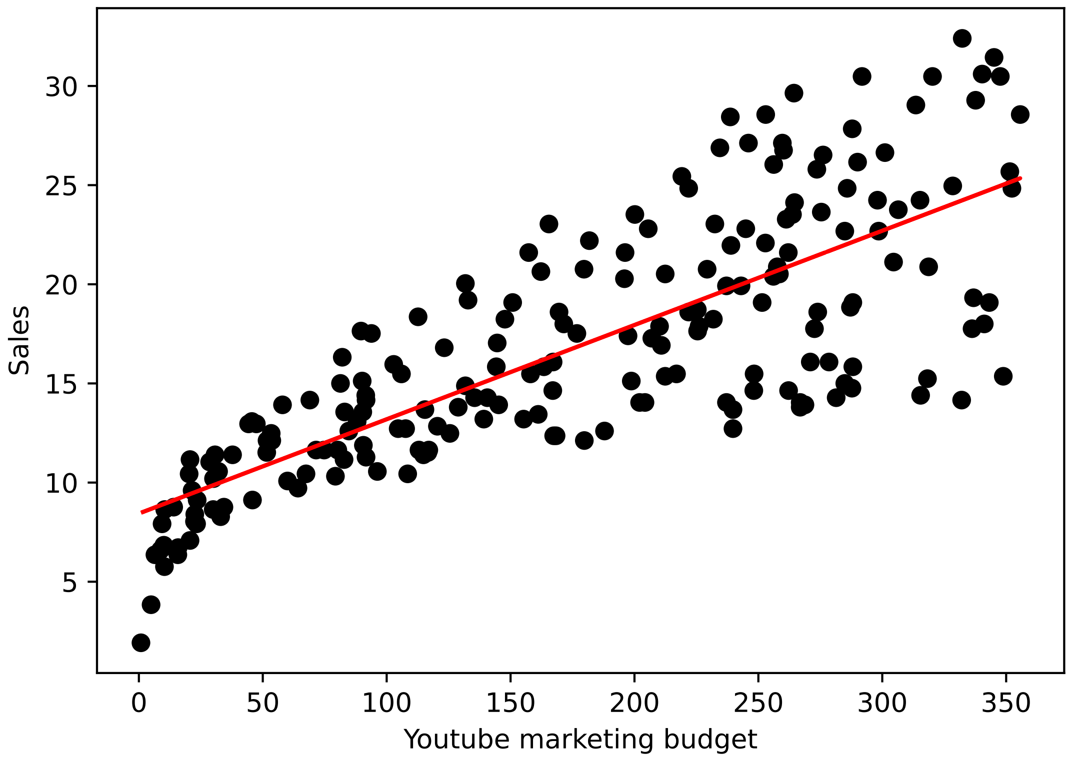
Data load

```
data = pd.read_csv("C:/Users/guilhermep/Documents/PgDip/Coding/Module 2/pgdip_module2_practice/Datasets/
```

Scatter plot and regression line

```
m, b = np.polyfit(data.youtube, data.sales, 1)

plt.scatter(data.youtube, data.sales, color="black")
plt.plot(data.youtube, m*data.youtube+b, color='red')
plt.xlabel("Youtube marketing budget")
plt.ylabel("Sales")
plt.show()
```



Correlation

```
stats.pearsonr(data.youtube, data.sales)
```

```
## PearsonRResult(statistic=0.7822244248616065, pvalue=1.467389700194595e-42)
```

Linear regression model

```
x=np.array(data.youtube).reshape(-1, 1)

y=np.array(data.sales).reshape(-1, 1)

model = LinearRegression().fit(x, y)

r_sq = model.score(x, y)
print(f"intercept: {model.intercept_}")
```

```
## intercept: [8.43911226]
```

```
print(f"slope: {model.coef_}")
```

```
## slope: [[0.04753664]]
```

```
print(f"R-squared: {r_sq}")
```

```
## R-squared: 0.611875050850071
```

```
print(f"adjusted R-squared: {1 - ( 1-model.score(x, y) ) * ( len(y) - 1 ) / ( len(y) - x.shape[1] - 1 )
```

```
## adjusted R-squared: 0.6099148238341623
```

Unit 11 - data activity

Using the Health_Data, please perform the following functions in R:

Perform simple linear regression analysis to find the population regression equation to predict the diastolic BP by systolic BP. Interpret the findings of regression analysis at 5% level of significance.

R

load data

```
data<-read_sav(here("Datasets/Health Data.sav"))
```

linear regression model

```
model<-lm(dbp~sbp, data)

model
```

```
##
## Call:
## lm(formula = dbp ~ sbp, data = data)
##
## Coefficients:
## (Intercept)          sbp
##      19.407        0.496
```

```
summary(model)
```

```
##
## Call:
## lm(formula = dbp ~ sbp, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.7958  -3.9366   0.1804   3.6685  19.2042
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.4068     2.7931   6.948 4.67e-11 ***
## sbp           0.4960     0.0216  22.961  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.264 on 208 degrees of freedom
## Multiple R-squared:  0.7171, Adjusted R-squared:  0.7157
## F-statistic: 527.2 on 1 and 208 DF,  p-value: < 2.2e-16
```

The regression equation for dbp as a function of sbp is dbp = 19.4068 + 0.496*sbp.

The p-values for the intercept and the coefficient of the predictor variable are both <0.001, and are therefore highly statistically significant at the 0.05 significance level. Therefore, the null hypothesis of no significant association between the predictor (sbp) and outcome variables (dbp) and conclude that there is a statistically significant association between them.

Python

load data

```
data = pd.read_spss("C:/Users/guilhermep/Documents/PgDip/Coding/Module 2/pgdip_module2_practice/Datasets
```

linear regression model

```
x=np.array(data.sbp).reshape(-1, 1)

y=np.array(data.dbp).reshape(-1, 1)

model = LinearRegression().fit(x, y)

r_sq = model.score(x, y)
print(f"intercept: {model.intercept_}")
```

```
## intercept: [19.4067706]
```

```
print(f"slope: {model.coef_}")
```

```
## slope: [[0.49603259]]
```

```
print(f"R-squared: {r_sq}")
```

```
## R-squared: 0.7170837699710062
```

```
print(f"adjusted R-squared: {1 - ( 1-model.score(x, y) ) * ( len(y) - 1 ) / ( len(y) - x.shape[1] - 1 )
```

```
## adjusted R-squared: 0.7157235957881745
```