Project 01 - Phát triển bán hàng trên app mobile hay website

Nhóm 15

2024-12-02

Contents

1 Các thư viện cần thiết

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
```

```
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ISLR2)
library(janitor)
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(boot)
library(lmPerm)
library(ipred)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```
library(leaps)
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:boot':
##
##     logit
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(broom)
setwd("D:/XLSLTK/Mini Project/Project_Bai_2")
```

2 Đọc dữ liệu.

2.1 Đọc dữ liệu bằng hàm read_csv()

```
df = read_csv("./data/Ecommerce_Customers.csv")
```

```
## Rows: 500 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): Email, Address, Avatar
## dbl (5): Avg. Session Length, Time on App, Time on Website, Length of Member...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

2.2 Kiểm tra các cột của dữ liệu

```
colnames(df)
```

```
## [1] "Email"                "Address"              "Avatar"
## [4] "Avg. Session Length"  "Time on App"          "Time on Website"
## [7] "Length of Membership" "Yearly Amount Spent"
```

Kiểm tra dữ liệu.

```
glimpse(df)
```

```
## Rows: 500
## Columns: 8
## $ Email                  <chr> "mstephenson@fernandez.com", "hduke@hotmail.com~
## $ Address                <chr> "835 Frank Tunnel\nWrightmouth, MI 82180-9605",~
## $ Avatar                 <chr> "Violet", "DarkGreen", "Bisque", "SaddleBrown",~
## $ `Avg. Session Length`  <dbl> 34.49727, 31.92627, 33.00091, 34.30556, 33.3306~
## $ `Time on App`          <dbl> 12.65565, 11.10946, 11.33028, 13.71751, 12.7951~
## $ `Time on Website`      <dbl> 39.57767, 37.26896, 37.11060, 36.72128, 37.5366~
## $ `Length of Membership` <dbl> 4.082621, 2.664034, 4.104543, 3.120179, 4.44630~
## $ `Yearly Amount Spent`  <dbl> 587.9511, 392.2049, 487.5475, 581.8523, 599.406~
```

2.3 Kiểm soát tên và dạng dữ liệu

Sử dụng hàm `clean_names()` để xử lý tên của biến chưa đúng quy tắc.

```
df = df |> clean_names()
```

Sử dụng `glimpse()` để kiểm tra lại dữ liệu sau khi đã xử lý tên biến.

```
glimpse(df)
```

```
## Rows: 500
## Columns: 8
## $ email                  <chr> "mstephenson@fernandez.com", "hduke@hotmail.com",~
```

```
## $ address              <chr> "835 Frank Tunnel\nWrightmouth, MI 82180-9605", "~
## $ avatar               <chr> "Violet", "DarkGreen", "Bisque", "SaddleBrown", "~
## $ avg_session_length   <dbl> 34.49727, 31.92627, 33.00091, 34.30556, 33.33067,~
## $ time_on_app          <dbl> 12.65565, 11.10946, 11.33028, 13.71751, 12.79519,~
## $ time_on_website      <dbl> 39.57767, 37.26896, 37.11060, 36.72128, 37.53665,~
## $ length_of_membership <dbl> 4.082621, 2.664034, 4.104543, 3.120179, 4.446308,~
## $ yearly_amount_spent  <dbl> 587.9511, 392.2049, 487.5475, 581.8523, 599.4061,~
```

Các dạng dữ liệu của các cộng đã đúng định dạng.

2.4 Kiểm tra NA

Kiểm tra xem dữ liệu có chứa các giá trị `NA` hay không. Nếu có thì thực hiểm kiểm tra xem dữ liệu bị khuyết bao nhiêu %.

Để thực hiện tính số lượng giá trị `NA` của các cột trong dữ liệu ta sử dụng hàm `colSums()` kết hợp với hàm `is.na()`.

```
colSums(is.na(df))
```

```
##                email              address               avatar
##                    0                    0                    0
##   avg_session_length          time_on_app      time_on_website
##                    0                    0                    0
## length_of_membership  yearly_amount_spent
##                    0                    0
```

Như vậy thì các cột của dữ liệu không chứa các giá trị `NA`. Vậy thì ta không cần kiểm tra tỷ lệ bị khuyết của dữ liệu.

2.5 Kiểm tra dữ liệu bị trùng lặp

Để kiểm tra xem dữ liệu có bị trùng lặp hay không ta sử dụng hàm `get_dupes()` từ thư viện `janitor`.

```
df |> get_dupes()
```

```
## No variable names specified - using all columns.
```

```
## No duplicate combinations found of: email, address, avatar, avg_session_length, time_on_app, time_on
```

```
## # A tibble: 0 x 9
## # i 9 variables: email <chr>, address <chr>, avatar <chr>,
## #   avg_session_length <dbl>, time_on_app <dbl>, time_on_website <dbl>,
## #   length_of_membership <dbl>, yearly_amount_spent <dbl>, dupe_count <int>
```

Như vậy thì dữ liệu trên không có các giá trị trùng lặp.

2.6 Thống kê giá trị

Sử dụng `summary()` để thống kê các giá trị

```
summary(df)
```

```
##     email               address             avatar          avg_session_length
##  Length:500         Length:500         Length:500         Min.   :29.53
##  Class :character   Class :character   Class :character   1st Qu.:32.34
##  Mode  :character   Mode  :character   Mode  :character   Median :33.08
##                                                           Mean   :33.05
##                                                           3rd Qu.:33.71
##                                                           Max.   :36.14
##   time_on_app     time_on_website length_of_membership yearly_amount_spent
##  Min.   : 8.508   Min.   :33.91   Min.   :0.2699       Min.   :256.7
##  1st Qu.:11.388   1st Qu.:36.35   1st Qu.:2.9304       1st Qu.:445.0
##  Median :11.983   Median :37.07   Median :3.5340       Median :498.9
##  Mean   :12.052   Mean   :37.06   Mean   :3.5335       Mean   :499.3
##  3rd Qu.:12.754   3rd Qu.:37.72   3rd Qu.:4.1265       3rd Qu.:549.3
##  Max.   :15.127   Max.   :40.01   Max.   :6.9227       Max.   :765.5
```

Ta có thể tạo ra các hàm sau để tạo 1 bảng dữ liệu tổng hợp cho các biến trong dữ liệu.

Bước 1: Sử dụng hàm `summarise()` kết hợp với `across()` để tính các ước lượng thống kê, kết quả sau đó được lưu lại trong một biến là `df_sum`

Vì trong dữ liệu chỉ có các biến `avg_session_length`, `time_on_app`, `time_on_website`, `length_of_membership` và `yearly_amount_spent` là các biến định lượng nên trong hàm `across()` ta chỉ lấy 5 biến đó.

```
df_sum = df |>
    summarise(across(c("avg_session_length",
                       "time_on_app",
                       "time_on_website",
                       "length_of_membership",
                       "yearly_amount_spent"),
         list(gtnn = min, gtln = max, tv = median,
              tb = mean, dlc = sd, q1 = ~ quantile(., 0.25),
              q3 = ~ quantile(., 0.75)),
                    .names = "{.col}.{.fn}"))
```

```
df_sum = df_sum |>
  mutate(across(everything(), ~ {
    attr(.x, "names") = NULL
    .x
  }))
```

Bước 2: Để tạo ra một bảng kết quả dễ nhìn hơn, ta dùng đoạn lệnh sau:

```
df_stats_tidy = df_sum |>
  gather(ten, gt) |>
  separate(ten, into = c("ten", "tk"), sep = "\\.") |>
  spread(tk, gt) |>
  select(ten, gtnn, q1, tv, tb, dlc, q3, gtln)
print.data.frame(df_stats_tidy, digits = 4)
```

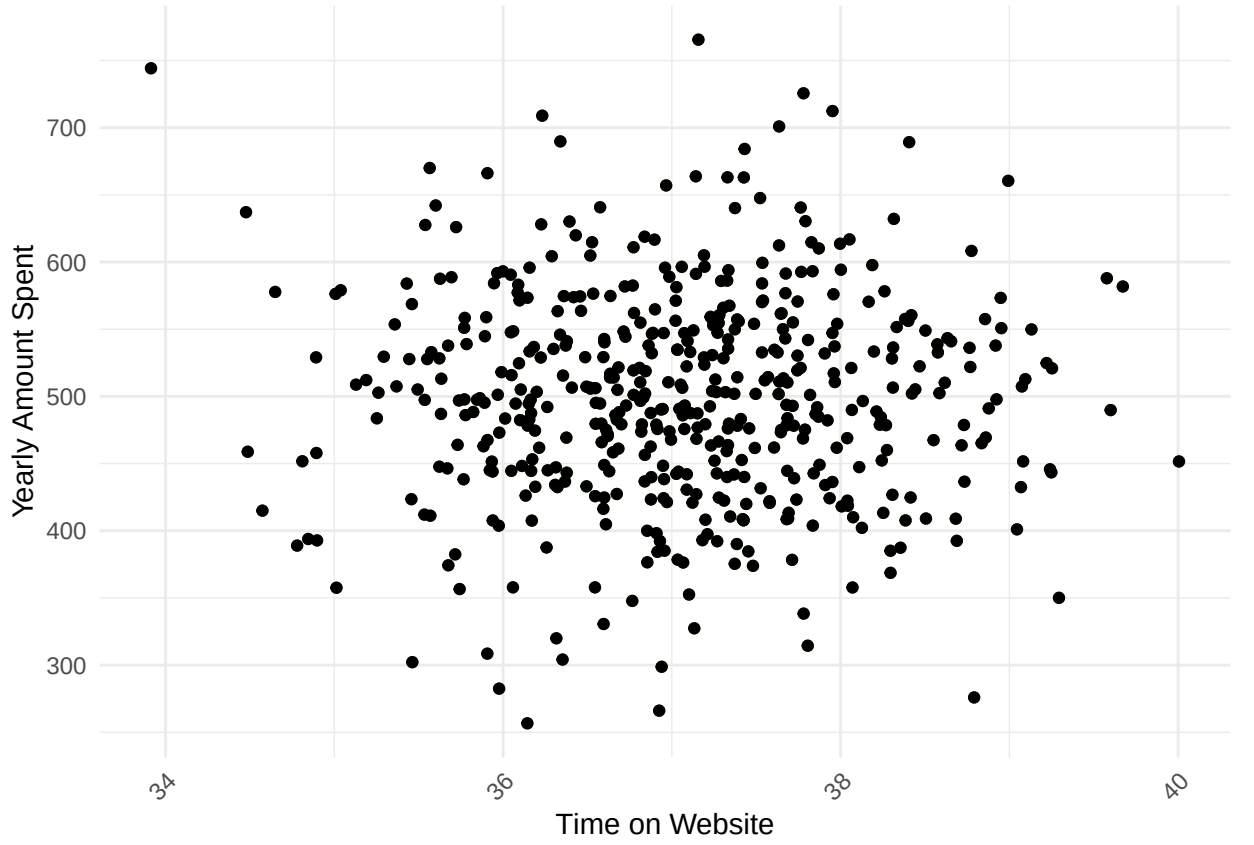
```
##                   ten     gtnn     q1      tv      tb     dlc      q3    gtln
```

```
## 1   avg_session_length  29.5324  32.34  33.082  33.053  0.9926  33.712  36.140
## 2 length_of_membership   0.2699   2.93   3.534   3.533  0.9993   4.127   6.923
## 3          time_on_app   8.5082  11.39  11.983  12.052  0.9942  12.754  15.127
## 4      time_on_website  33.9138  36.35  37.069  37.060  1.0105  37.716  40.005
## 5  yearly_amount_spent 256.6706 445.04 498.888 499.314 79.3148 549.314 765.518
```

3 Khám phá dữ liệu

Ta sẽ vẽ biểu đồ phân tán (*scatter plot*) của 2 biến `time_on_web` và `yearly_amount_spent` để mô tả sự mối liên hệ giữa hai biến. Để vẽ biểu đồ phân tán ta sử dụng hàm `geom_point()` trong thư viện `ggplot`.

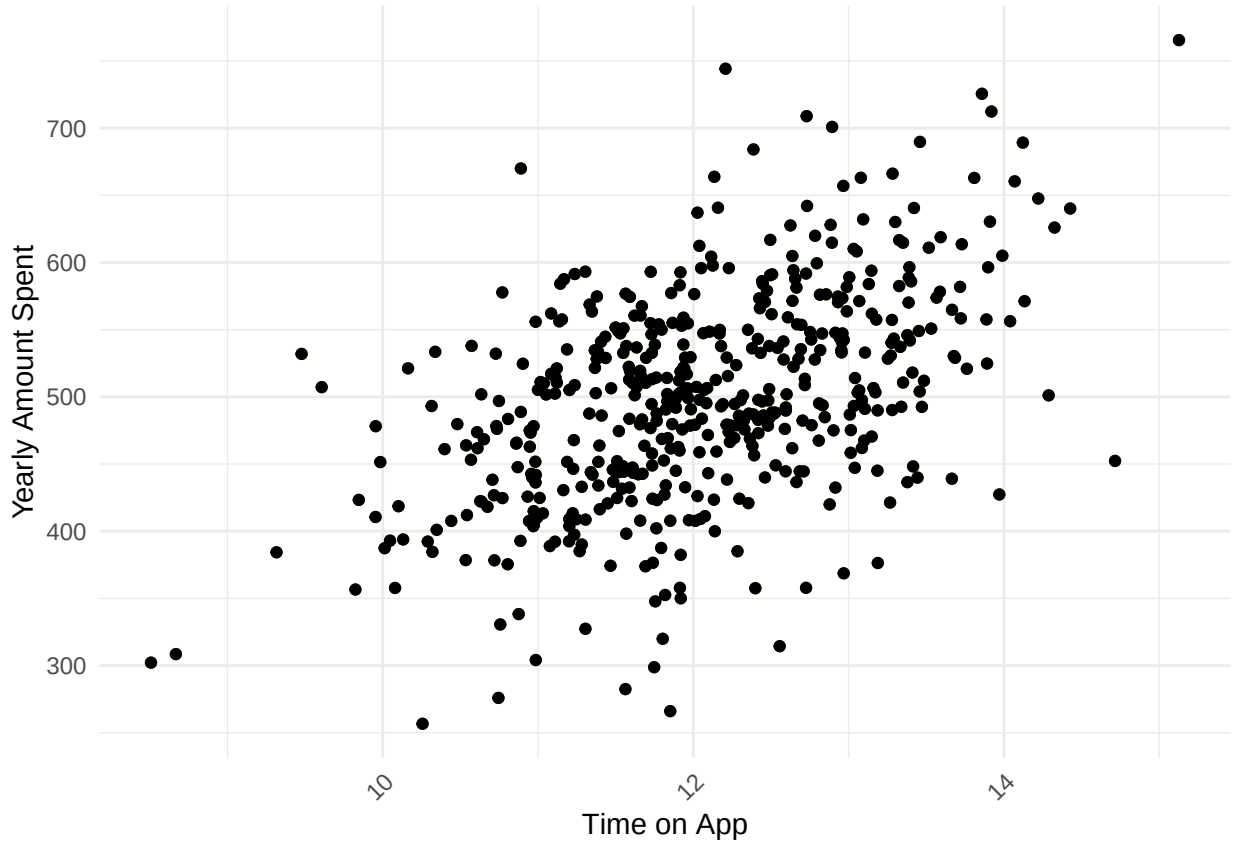
```
ggplot(df, aes(x = `time_on_website`, y = `yearly_amount_spent`)) +
  geom_point(color = "#4c9a2a") +
  labs(x = "Time on Website", y = "Yearly Amount Spent") +
  geom_point() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Qua biểu đồ phân tán trên, ta có thể đưa ra được nhận định là không có mối tương quan tuyến tính giữa hai biến `time_on_web` và `yearly_amount_spent`. Tức là việc dành nhiều thời gian hơn trên trang web không dẫn đến sự gia tăng hoặc giảm đáng kể trong chi tiêu năm.

Ta sẽ vẽ biểu đồ phân tán (*scatter plot*) của 2 biến `time_on_app` và `yearly_amount_spent` để mô tả sự mối liên hệ giữa hai biến. Để vẽ biểu đồ phân tán ta sử dụng hàm `geom_point()` trong thư viện `ggplot`.

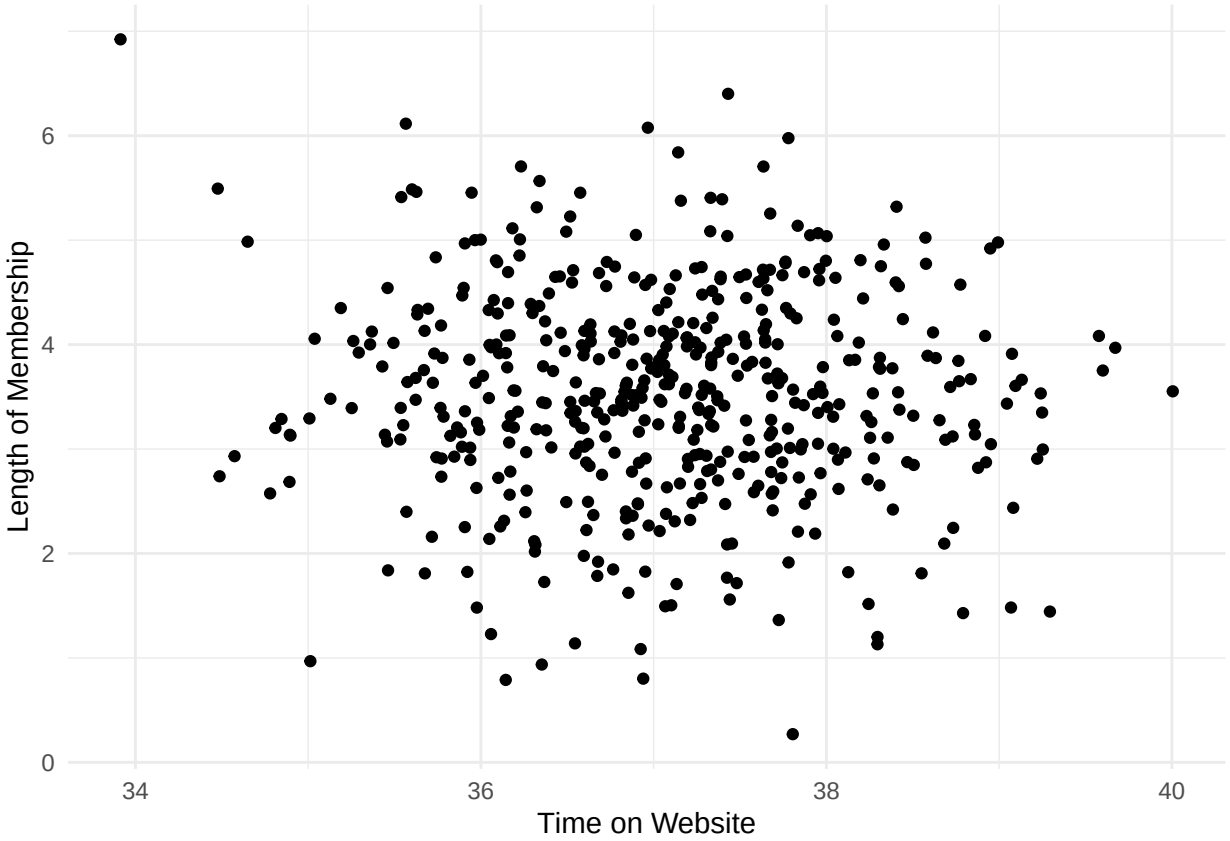
```r
ggplot(df, aes(x = `time_on_app`, y = `yearly_amount_spent`)) +
  geom_point(color = "#4c9a2a") +
  labs(x = "Time on App", y = "Yearly Amount Spent") +
  geom_point() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Dựa vào biểu đồ phân tán trên cho thấy được dữ liệu có xu hướng tăng nhẹ từ trái sang phải. Điều này cho thấy rằng, có một sự tương quan tuyến tính nhẹ giữa hai biến `time_on_app` và `yearly_amount_spent`. Tức là khi `time_on_app` (thời gian tương tác trên mobile app) tăng lên, `yearly_amount_spent` (số tiền chi tiêu trung bình năm) có thể cũng tăng, nhưng không tăng nhiều.

Kiểm tra thời gian tương tác trên website (`Time on Website`) với thời gian là khách hàng thành viên (`Length of Membership`).

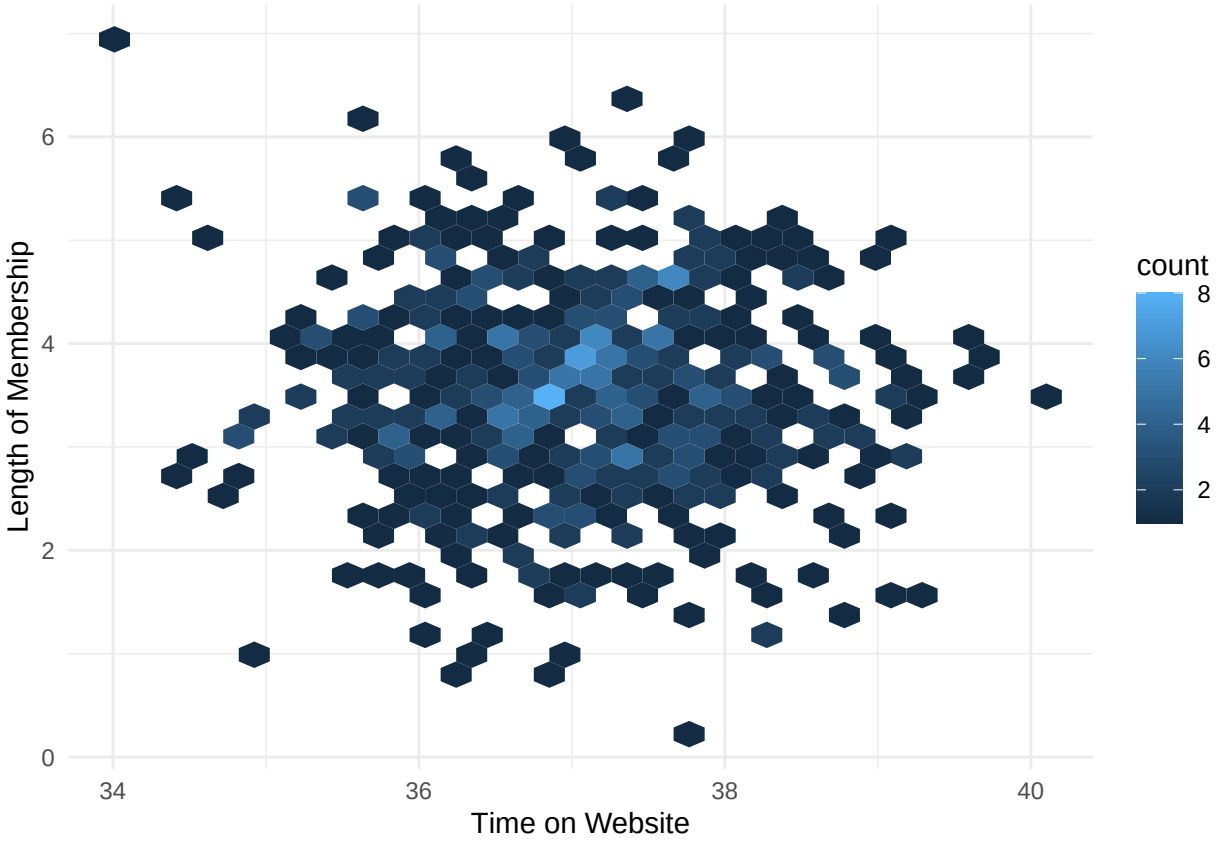
```r
ggplot(df, aes(x = `time_on_website`, y = `length_of_membership`)) +
       geom_point() +
       labs(y = "Length of Membership",
            x = "Time on Website") +
       theme_minimal()
```



Qua biểu đồ phân tán (*scatter plot*) trên thì cho ra được 1 biểu đồ rất khó để xác định tính tương quan. Do đó để biểu diễn mối quan hệ giữa hai biến này thì ta sử dụng biểu đồ hexbin khi đó dữ liệu sẽ được chia thành các ô lục giác trên không gian 2 chiều (**2D**) bằng cách sử dụng hàm `geom_hex()` trong thư viện `hexbin`.

```
# import library
library(hexbin)
```

```
ggplot(df, aes(x = `time_on_website`, y = `length_of_membership`)) +
       geom_hex(bins = 30) +
       labs(y = "Length of Membership",
            x = "Time on Website") +
       theme_minimal()
```

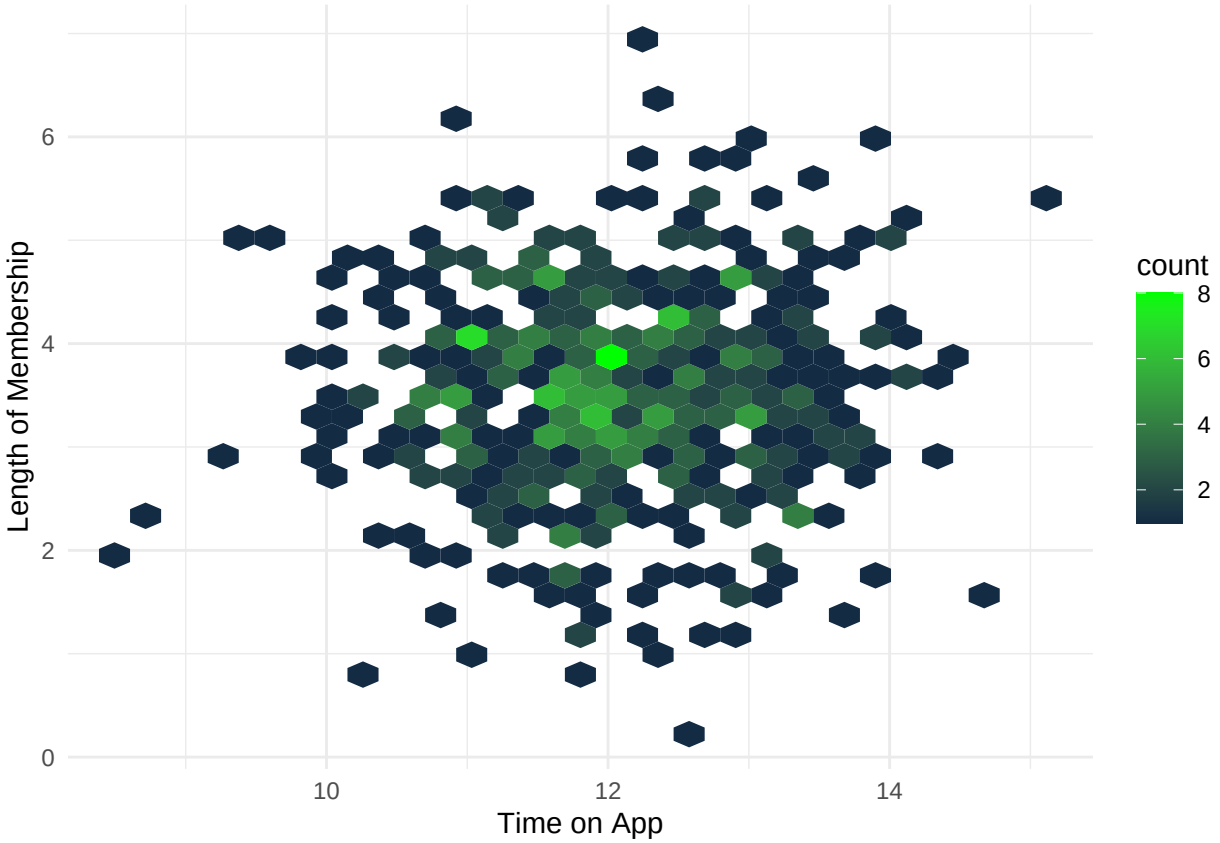


Dựa vào biểu đồ trên, ta có thể có được một số nhận xét sau:

- Phần lớn các điểm dữ liệu tập trung ở vùng trung tâm của biểu đồ, cụ thể là khoảng từ 36 đến 38 trên trục **"Time on Website"** và từ 3 đến 5 năm trên trục **"Length of Membership (years)"**.
- Điều này cho thấy mối quan hệ tập trung giữa thời gian tương tác trên website và thời gian là khách hàng thành viên.
- Một số điểm nằm ngoài khu vực tập trung, đặc biệt là ở vùng thời gian sử dụng website thấp hơn (34–35) và cao hơn (gần 40), nhưng số lượng rất ít.
- Dữ liệu hơi phân tán và có xu hướng tập trung ở giữa. Điều này cho thấy được là không có mối tương quan mạnh giữa hai biến này.

Kiểm tra thời gian tương tác trên mobile app (`Time on App`) với thời gian là khách hàng thành viên (`Length of Membership`).

```
ggplot(df, aes(x = `time_on_app`, y = `length_of_membership`)) +
      geom_hex(bins = 30) +
      scale_fill_gradient(high = "green") +
      theme_minimal() +
      labs(x = "Time on App",
           y = "Length of Membership")
```



Dựa vào biểu đồ trên, ta có thể có được một số nhận xét sau:

- Phần lớn các điểm dữ liệu tập trung ở vùng trung tâm của biểu đồ, cụ thể là khoảng từ 11 đến 13 trên trục **"Time on App"** và từ 3 đến 5 năm trên trục **"Length of Membership"**. Điều này cho thấy mối quan hệ tập trung giữa thời gian tương tác trên mobile app và thời gian là khách hàng thành viên.
- Khi rời xa trung tâm, mật độ dữ liệu giảm (màu xanh lá cây mờ dần, chuyển sang màu tối hơn).
- Dữ liệu phân bố rải rác và ngẫu nhiên, không có xu hướng tăng hay giảm đồng đều.
- Dữ liệu trải rộng trên một khoảng giá trị nhất định, nhưng phần lớn vẫn tập trung xung quanh trung tâm.

Để làm rõ hơn những khẳng định trên ta sẽ vẽ biểu đồ tương quan của tất cả các biến định lượng có trong dữ liệu.

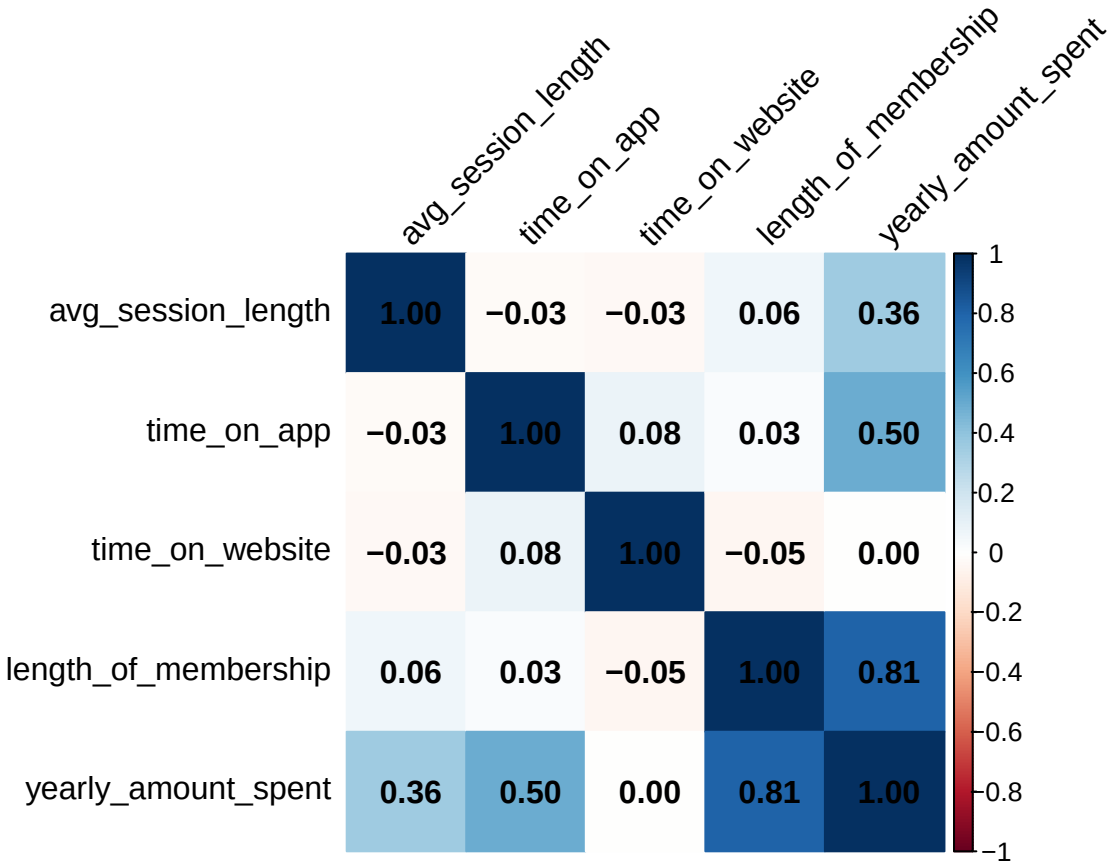
Tạo ma trận tương quan cho các biến định lượng trong dữ liệu.

```
numeric_data = df[, sapply(df, is.numeric)]
corr_matrix = cor(numeric_data, method = "pearson")
```

Vẽ biểu đồ tương quan của các biến định lượng để kiểm tra hệ số tương quan của từng cặp biến trong dữ liệu. Sử dụng hàm `corrplot()` trong thư viện `corrplot`.

```
corrplot(
  corr_matrix,
  method = "color",
```

```
  addCoef.col = "black",
  tl.col = "black",
  tl.srt = 45,
  is.corr = TRUE
)
```



Dựa vào biểu đồ tương quan của các biến `avg_session_length`, `time_on_app`, `time_on_website`, `length_of_membership` và `yearly_amount_spent` phía trên ta có được:

- Biến `length_of_membership` (Thời gian là khách hàng thành viên) có sự tương quan mạnh đối với biến `yearly_amount_spent` (Chi phí mua hàng trung bình năm) với 0.81
- Biến `time_on_app` (Thời gian tương tác trên mobile app) có sự tương quan mạnh đối với biến `yearly_amount_spent` (Chi phí mua hàng trung bình năm) với 0.50
- Biến `time_on_website` có sự tương quan rất thấp hoặc gần như không có (hệ số bằng 0 hoặc âm nhẹ). Điều này có thể chỉ ra rằng `time_on_website` không phải là một yếu tố quan trọng ảnh hưởng đến các yếu tố khác.

Nếu so sánh với thời gian tương tác trên website, có thể thấy rằng không có mối tương quan nào tức là việc khách hàng dành nhiều thời gian hơn trên website không nhất thiết dẫn đến việc sẽ mua hàng.

Vì biến `length_of_membership` (Thời gian là khách hàng thành viên) có sự tương quan mạnh đối với biến `yearly_amount_spent` (Chi phí mua hàng trung bình năm) nên ta sẽ mong muốn dự đoán được chi phí mua hàng trung bình năm `yearly_amount_spent` thông qua thời gian là khách thành viên `length_of_membership`.

```r
ggplot(df, mapping = aes(x = length_of_membership, y = yearly_amount_spent)) +
    geom_point() +
    geom_smooth(formula = y ~ x, method = "lm", se = TRUE, color = "blue") +
    theme_bw() +
    labs(x = "Length of Membership", y = "Yearly Amount Spent", title = "Scatterplot with Regression Li
```

Qua đó ta có thể đưa ra giả thuyết rằng công ty nên đầu tư vào nền tảng mobile app hơn là đầu tư vào nền tảng website nhưng cũng cần thực hiện xây dựng các mô hình hồi quy và chọn ra một mô hình tốt để có thể đưa ra quyết định cuối cùng.

4 Xây dựng mô hình hồi quy

Dựa vào các biến trên ta có được các mô hình sau:

1. Mô hình hồi quy tuyến tính đơn.
   - MH1: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{avg\_session\_length} + \epsilon$
   - MH2: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{time\_on\_app} + \epsilon$
   - MH3: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{length\_of\_membership} + \epsilon$
   - MH4: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{time\_on\_website} + \epsilon$
2. Mô hình hồi quy đa biến.
   - MH5: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{avg\_session\_length} + \beta_2 \times \text{time\_on\_app} + \epsilon$

- MH6: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{time\_on\_website} + \beta_2 \times \text{time\_on\_app} + \epsilon$
- MH7: $\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{avg\_session\_length} + \beta_2 \times \text{time\_on\_app} + \beta_3 \times \text{time\_on\_website} + \beta_4 \times \text{length\_of\_membership} + \epsilon$
- Và còn nhiều mô hình khác.

4.1 Mô hình hồi quy tuyến tính đơn

Vì biến `length_of_membership` (Thời gian là khách hàng thành viên) có sự tương quan mạnh đối với biến `yearly_amount_spent` (Chi phí mua hàng trung bình năm) nên chọn biến `length_of_membership` để dự đoán biến `yearly_amount_spent`.

Biến phản hồi được quan tâm là `yearly_amount_spent` - Chi phí mua hàng trung bình năm. Mô hình được đề xuất là:

$$\text{yearly\_amount\_spent}_i = \beta_0 + \beta_1 \times \text{length\_of\_membership}_i$$

Ta ước lượng mô hình này như sau:

```
md_df = lm(yearly_amount_spent ~ length_of_membership, data = df)
```

Kết quả tổng hợp

```
summary(md_df)
```

```
##
## Call:
## lm(formula = yearly_amount_spent ~ length_of_membership, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -125.975  -29.032   -0.494   33.033  147.777
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            272.400      7.675   35.49   <2e-16 ***
## length_of_membership    64.219      2.090   30.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 46.66 on 498 degrees of freedom
## Multiple R-squared:  0.6546, Adjusted R-squared:  0.6539
## F-statistic: 943.9 on 1 and 498 DF,  p-value: < 2.2e-16
```

Khi đó, ta được một số nhận xét sau:

- Nếu thời gian là khách thành viên tăng lên 10 đơn vị thì chi phí mua hàng trung bình năm sẽ tăng lên 642.19 đơn vị.
- Hệ số này có ý nghĩa thống kê vì giá trị `p-value` < 0.001.
- Giá trị R-squared (0.6546) cho thấy 65.46% sự biến động trong chi tiêu hàng năm của khách hàng có thể được giải thích bởi thời gian họ là khách thành viên. Điều này thể hiện rằng `length_of_membership` là một yếu tố quan trọng ảnh hưởng đến chi tiêu.

4.2 Mô hình hồi quy đa biến

Trước hết, chúng ta nên xây dựng một mô hình tuyến tính bao gồm tất cả các biến dự đoán hiện có, để có thể hiểu rõ hơn về mô hình cũng như sử dụng kết quả của mô hình này cho việc lựa chọn mô hình tiếp theo.

$$\text{yearly\_amount\_spent} = \beta_0 + \beta_1 \times \text{avg\_session\_length} + \beta_2 \times \text{time\_on\_app} + \beta_3 \times \text{time\_on\_website} + \beta_4 \times \text{length\_of\_menbersh}$$

Ta ước lượng mô hình này như sau:

```
model_df = lm(yearly_amount_spent ~ avg_session_length + time_on_app + time_on_website +
                length_of_membership, data = df)
```

Vì các cặp biến đã xét ở trên có biến `length_of_membership` có sự tương quan mạnh với biến `yearly_amount_spent` nên hiện tượng đa cộng tuyến có thể xảy ra nếu chọn cặp biến này.

Nên ta sẽ kiểm tra hiện tượng đa cộng tuyến bằng hàm `vif` trong gói `car`.

```
vif(model_df)
```

```
##   avg_session_length          time_on_app      time_on_website
##             1.005422             1.008684             1.010275
## length_of_membership
##             1.006949
```

Với $j = \overline{1,4}$, ta thấy rằng VIF_j gần bằng 1 nên không có hiện tượng đa cộng tuyến. Do đó việc lựa chọn mô hình này là hợp lý.

Kết quả tổng hợp.

```
summary(model_df)
```

```
##
## Call:
## lm(formula = yearly_amount_spent ~ avg_session_length + time_on_app +
##     time_on_website + length_of_membership, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -30.4059  -6.2191  -0.1364   6.6048  30.3085
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -1051.5943    22.9925 -45.736   <2e-16 ***
## avg_session_length      25.7343     0.4510  57.057   <2e-16 ***
## time_on_app             38.7092     0.4510  85.828   <2e-16 ***
## time_on_website          0.4367     0.4441   0.983    0.326
## length_of_membership    61.5773     0.4483 137.346   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.973 on 495 degrees of freedom
## Multiple R-squared:  0.9843, Adjusted R-squared:  0.9842
## F-statistic:  7766 on 4 and 495 DF,  p-value: < 2.2e-16
```

Khi đó, ta được một số nhận xét sau:

- Nếu trung bình thời gian tương tác tăng lên 10 đơn vị thì chi phí mua hàng trung bình năm sẽ tăng lên 257.343 đơn vị.
- Nếu thời gian tương tác trên website tăng lên 10 đơn vị thì chi phí mua hàng trung bình năm sẽ tăng lên 387.092 đơn vị.
- Nếu thời gian tương tác trên mobile app tăng lên 10 đơn vị thì chi phí mua hàng trung bình năm sẽ tăng lên 4.367 đơn vị.
- Nếu thời gian là khách thành viên tăng lên 10 đơn vị thì chi phí mua hàng trung bình năm sẽ tăng lên 615.773 đơn vị.
- Hệ số này có ý nghĩa thống kê vì giá trị `p-value` < 0.001.
- Giá trị R-squared (0.9843) cho thấy 98.43% sự biến động trong chi tiêu hàng năm của khách hàng có thể được giải thích bởi trung bình thời gian tương tác, thời gian tương tác trên website, thời gian tương tác trên mobile app và thời gian là khách hàng thành viên.

Tiếp theo, ta áp dụng phương pháp bootstrap để ước lượng khoảng tin cậy và kiểm định giả thuyết $\beta_j = 0$.

Trước tiên, ta định hàm `fun_boot_model()` để thực hiện ước tính mỗi lần lặp lấy mẫu:

```
fun_boot_model = function(data, ind, formula, ...)
{
    data_new = data[ind,]
    out_md = lm(formula = formula, data = data_new, ...)
    return(out_md$coefficients)
}
```

```
set.seed(84)
out_boot_model_df = boot(data = df, statistic = fun_boot_model, R = 1000,
                         formula = yearly_amount_spent ~ avg_session_length +
                           time_on_app +
                           time_on_website +
                           length_of_membership)
out_boot_model_df
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = df, statistic = fun_boot_model, R = 1000, formula = yearly_amount_spent ~
##     avg_session_length + time_on_app + time_on_website + length_of_membership)
##
##
## Bootstrap Statistics :
##          original       bias    std. error
## t1* -1051.5942550  0.650397159  23.4427912
## t2*    25.7342711 -0.005010213   0.4549877
## t3*    38.7091538  0.023586259   0.4613324
## t4*     0.4367388 -0.020559710   0.4431993
## t5*    61.5773237 -0.001257243   0.4335430
```

Sử dụng hàm `boot.ci()` để ước tính khoảng tin cậy 95%, của các hệ số, tương ứng với đối số `index`:

Hệ số thứ nhất.

```
boot.ci(out_boot_model_df, index = 1, type = "perc", conf = 0.95)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out_boot_model_df, conf = 0.95, type = "perc",
##     index = 1)
##
## Intervals :
## Level     Percentile
## 95%   (-1095,  -999 )
## Calculations and Intervals on Original Scale
```

Hệ số thứ hai.

```
boot.ci(out_boot_model_df, index = 2, type = "perc", conf = 0.95)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out_boot_model_df, conf = 0.95, type = "perc",
##     index = 2)
##
## Intervals :
## Level     Percentile
## 95%   (24.80, 26.63 )
## Calculations and Intervals on Original Scale
```

Hệ số thứ ba.

```
boot.ci(out_boot_model_df, index = 3, type = "perc", conf = 0.95)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out_boot_model_df, conf = 0.95, type = "perc",
##     index = 3)
##
## Intervals :
## Level     Percentile
## 95%   (37.84, 39.64 )
## Calculations and Intervals on Original Scale
```

Hệ số thứ tư.

```r
boot.ci(out_boot_model_df, index = 4, type = "perc", conf = 0.95)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out_boot_model_df, conf = 0.95, type = "perc",
##     index = 4)
##
## Intervals :
## Level     Percentile
## 95%   (-0.4850,  1.3399 )
## Calculations and Intervals on Original Scale
```

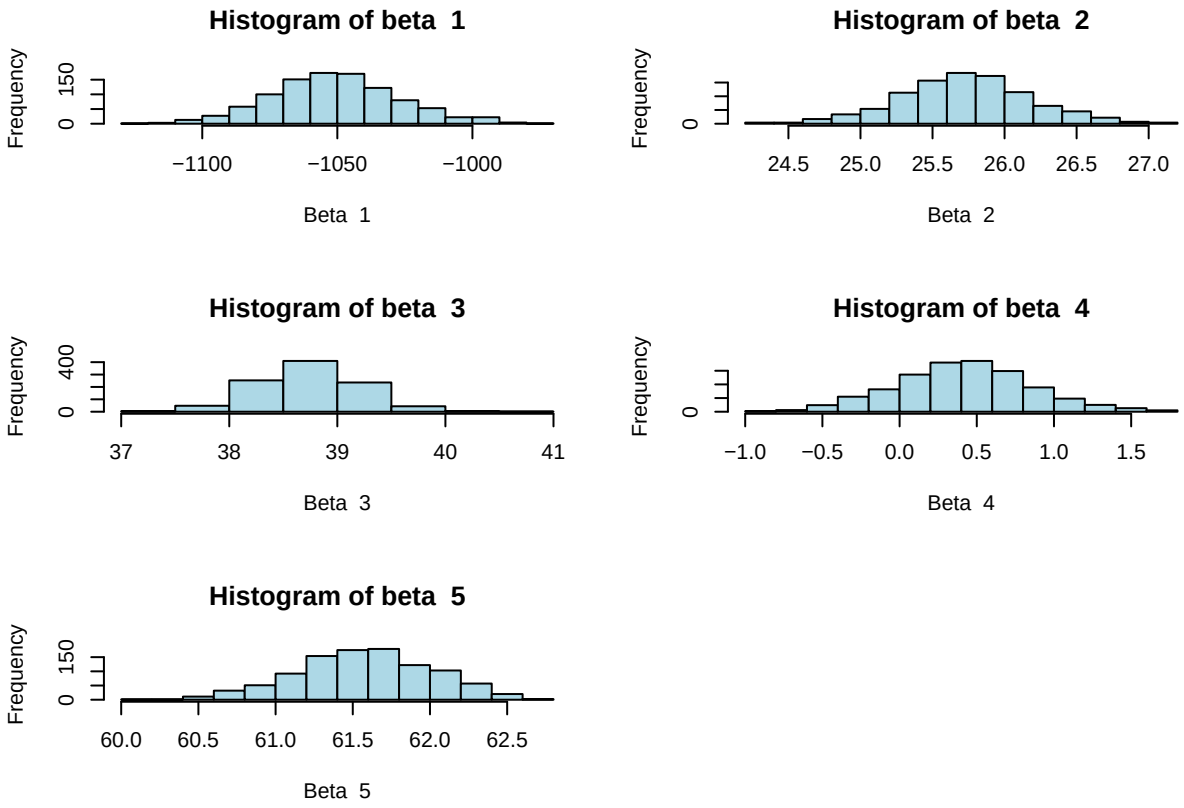
Hệ số thứ năm.

```r
boot.ci(out_boot_model_df, index = 5, type = "perc", conf = 0.95)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out_boot_model_df, conf = 0.95, type = "perc",
##     index = 5)
##
## Intervals :
## Level     Percentile
## 95%   (60.72, 62.40 )
## Calculations and Intervals on Original Scale
```

Histogram của chỉ số thứ nhất đến thứ năm.

```r
par(mfrow = c(3, 2))
for (i in 1:5)
{
    hist(out_boot_model_df$t[, i],
         col = "lightblue",
         main = paste("Histogram of beta ", i),
         xlab = paste("Beta ", i),
         ylab = "Frequency")
}
```



Tính toán giá trị p_value cho giả thuyết $\beta_j = 0$:

```
pval_adv = sapply(1:ncol(out_boot_model_df$t), function(i)
{
    qt = mean(out_boot_model_df$t[, i] <= 0)
    if (qt < 0.5)
    {
        return(2 * qt)
    } else
    {
        return(2 * (1 - qt))
    }
})

pval_adv
```

```
## [1] 0.000 0.000 0.000 0.338 0.000
```

Giả sử ta có thông tin rằng nếu trung bình thời gian tương tác tăng lên 10 đơn vị, thời gian tương tác trên website tăng lên 15 đơn vị, thời gian tương tác trên mobile app tăng lên 20 đơn vị, thời gian là khách thành viên tăng lên 25 đơn vị. Khi đó dự vào mô hình, ta có thể ước tính được trung bình chi phí mua hàng trung bình năm.

```
predict(model_df, newdata = data.frame(avg_session_length = 10,
                                       time_on_app = 20,
```

```
                                        time_on_website = 15,
                                        length_of_membership = 25))
```

```
##        1
## 1525.916
```

Ta sử dụng phương pháp bootstrap để tìm khoảng tin cậy cho chi phí mua hàng trung bình năm:

```
x_adv = c(1, 10, 20, 15, 25)
y_adv = apply(out_boot_model_df$t, 1, function(x)
{
    x_adv %*% x
})
quantile(y_adv, probs = c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 1490.130 1561.896
```

Như vậy, trung bình chi phí mua hàng trung bình năm được tiên lượng theo mô hình này, có giá trị thay đổi từ (1490.13, 1561.9) với độ tin cậy 95%.

• Khoảng tiên đoán cho trung bình chi phí mua hàng trung bình năm dựa vào mô hình.

Ta có thể sử dụng phương pháp bootstrap để ước lượng khoảng tiên đoán cho chi phí mua hàng trung bình năm dựa trên mô hình:

```
resid_adv = residuals(model_df)
y_adv_pc_pci = y_adv + sample(resid_adv, size = 1000, replace = TRUE)
quantile(y_adv_pc_pci, probs = c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 1488.108 1568.933
```

Như vậy, ta có thể ước lượng được khoảng tiên đoán cho trung bình chi phí mua hàng trung bình năm dựa vào mô hình là (1488.108, 1568.933) đơn vị.

• Tính **RMSE** bằng phương pháp *k-fold Cross-Validation*

Với $k = 10$:

```
set.seed(150)
k = 10
MSE.k = vector()
RMSE.k = vector()
n = nrow(df)
p = 2
lower = seq(1, n, by = n/k)
upper = c(lower[-1] - 1, n)

for(i in 1:k)
{
    fold = c(lower[i]: upper[i])
    train = df[-fold, ]
```

```
    test = df[fold, ]

    lm.fit = lm(yearly_amount_spent ~ avg_session_length +
                  time_on_app +
                  time_on_website +
                  length_of_membership, data = train)
    lm.pred = predict(lm.fit, newdata = test)
    MSE.k[i] = mean((lm.pred - test$yearly_amount_spent)^2)
    RMSE.k[i] = sqrt(mean((lm.pred - test$yearly_amount_spent)^2))
}
MSE.kfold = mean(MSE.k)
RMSE.kfold = mean(RMSE.k)
cat("MSE: ", sprintf("%.3f", MSE.kfold))
```

```
## MSE:  101.643
```

```
cat("\nRMSE: ", sprintf("%.3f", RMSE.kfold))
```

```
##
## RMSE:  10.070
```

5 Lựa chọn mô hình

Từ những mô hình trên, ta vẫn chưa thể kết luận được là những mô hình trên đâu là mô hình tốt nhất. Do đó ta sẽ thực hiện các phương pháp để tìm ra mô hình hồi quy tốt nhất.

Ta có thể sử dụng 1 trong 3 phương pháp sau:

- Hồi quy từng bước.
- Hồi quy từng bước và cross - validation.
- Phương pháp co hệ số.

Vì không có hiện tượng đa cộng tuyến nên ta sẽ sử dụng được phương pháp hồi quy từng bước và cross - validation hoặc hồi quy từng bước.

Ở đây ta sử dụng phương pháp hồi quy từng bước để lựa chọn mô hình tốt nhất.

Sử dụng hàm `regsubsets()` để tìm tập biến tốt nhất (chứa tối đa 5 biến) cho mô hình dự đoán chi phí mua hàng trung bình năm.

```
out_subset_df = regsubsets(yearly_amount_spent ~ . - email - avatar - address,
                           data = df,
                           nvmax = 5,

                           method = "exhaustive")
```

Để biết được tập nào tốt nhất trong số các tập từ $\mathcal{M}_1, ..., \mathcal{M}_5$, ta sử dụng chỉ số Mallow's Cp.

```r
sum_out_subset_df = summary(out_subset_df)
sum_out_subset_df$cp
```

```
## [1] 10404.275713  3257.028515     3.967151     5.000000
```

Kết quả cho thấy tập kết quả thứ 4 (chứa 4 biến) là tốt nhất. Ta có thể truy xuất kết quả các biến được lựa chọn:

```r
sum_out_subset_df$which[which.min(sum_out_subset_df$cp), ]
```

```
##          (Intercept)   avg_session_length          time_on_app
##                 TRUE                 TRUE                 TRUE
##      time_on_website length_of_membership
##                FALSE                 TRUE
```

`TRUE` có nghĩa là được chọn, `FALSE` có nghĩa là không được chọn.

Kết quả ước lượng hệ số của mô hình tương ứng với tập này là:

```r
coef(out_subset_df, which.min(sum_out_subset_df$cp))
```

```
##          (Intercept)   avg_session_length          time_on_app
##          -1035.33958            25.72103            38.74598
## length_of_membership
##             61.55603
```

Ta thấy rằng mô hình tốt nhất là mô hình chứa 4 biến: `avg_session_length`, `time_on_app`, `time_on_website`, `length_of_membership`.

6 Chuẩn đoán mô hình

```r
md = lm(yearly_amount_spent ~ avg_session_length +
          time_on_app +
          length_of_membership, data = df)

summary(md)
```

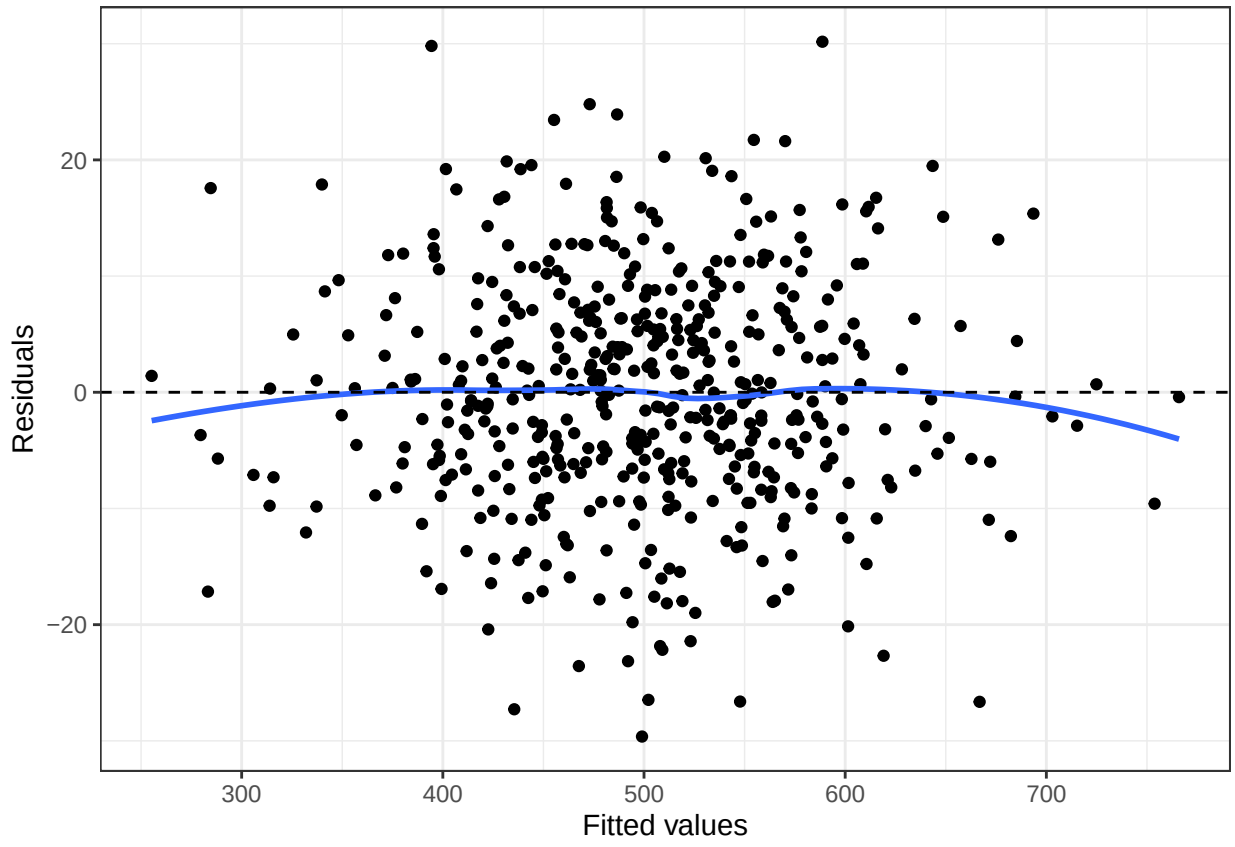
```
##
## Call:
## lm(formula = yearly_amount_spent ~ avg_session_length + time_on_app +
##     length_of_membership, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.628  -6.378  -0.135   6.351  30.169
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)          -1035.3396    15.9829  -64.78   <2e-16 ***
## avg_session_length      25.7210     0.4508   57.05   <2e-16 ***
## time_on_app             38.7460     0.4494   86.21   <2e-16 ***
## length_of_membership    61.5560     0.4478  137.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.973 on 496 degrees of freedom
## Multiple R-squared:  0.9843, Adjusted R-squared:  0.9842
## F-statistic: 1.036e+04 on 3 and 496 DF,  p-value: < 2.2e-16
```

Kiểm tra tính tuyến tính của mô hình

```
augmented_df = augment(md, data = df)
ggplot(data = augmented_df, mapping = aes(x=.fitted, y=.resid))+
  geom_point() +
  geom_smooth(method = "loess", se = FALSE) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  labs(x = "Fitted values", y = "Residuals") +
  theme_bw()
```

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



Nhận xét: Hình vẽ cho thấy không có xu hướng đường cong đáng kể.

⇒ Giảđịnh về tính tuyến tính của mô hình là phù hợp

Kiểm tra tính tuyến tính từng phần

```
terms = predict(md, type = "terms")
head(terms)
```

```
##   avg_session_length time_on_app length_of_membership
## 1          37.143080  23.3701502             33.80405
## 2         -28.985585 -36.5385140            -53.51850
## 3          -1.344664 -27.9827300             35.15352
## 4          32.212073  64.5130550            -25.44005
## 5           7.137046  28.7766637             56.19123
## 6          21.035802  -0.9904479            120.65264
```

```
partial_resid_md = residuals(md, type = "partial")
head(partial_resid_md)
```

```
##   avg_session_length time_on_app length_of_membership
## 1           31.46281   17.689881             28.12379
## 2          -17.05209  -24.605019            -41.58501
## 3          -18.93732  -45.575391             17.56086
## 4           43.46530   75.766282            -14.18682
## 5           15.12416   36.763781             64.17834
## 6           18.12622   -3.900029            117.74306
```

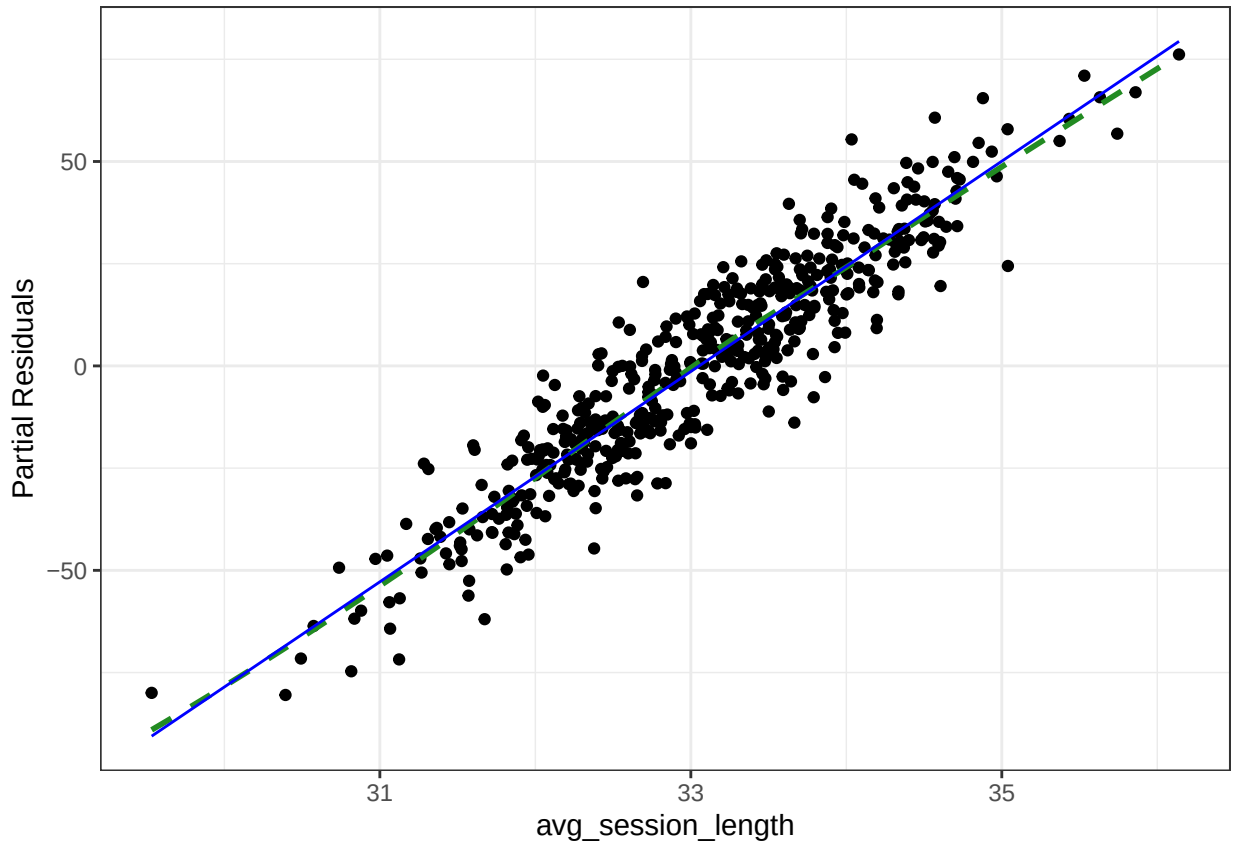
- avg_session_length

```
data_avg_session_length_md = tibble(
  avg_session_length = df$avg_session_length,
  terms_avg_session_length = terms[, "avg_session_length"],
  partial_resid_avg_session_length = partial_resid_md[, "avg_session_length"]
)
```

```
ggplot(data_avg_session_length_md,
       mapping = aes(avg_session_length,
                     partial_resid_avg_session_length)) +
  geom_point(fomular = y ~ x) +
  geom_smooth(method = "loess", se = FALSE, linetype = "dashed",
              color = "forestgreen") +
  geom_line(aes(x = avg_session_length, y = terms_avg_session_length),
            color = "blue") +
  labs(x = "avg_session_length", y = "Partial Residuals") +
  theme_bw()
```

```
## Warning in geom_point(fomular = y ~ x): Ignoring unknown parameters: `fomular`
```

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



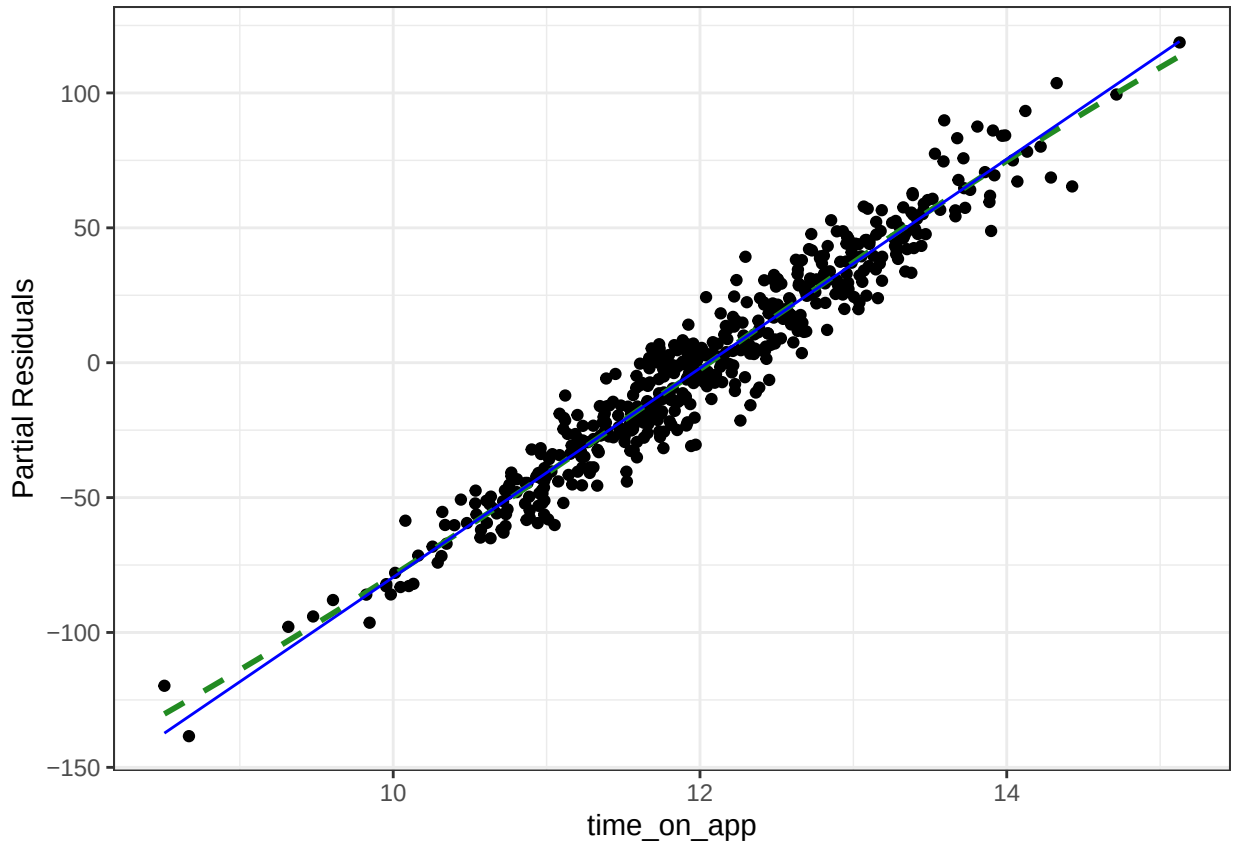
Nhận xét: Có sự tuyến tính từng phần của thành phần avg_session_length trong mô hình.

- time_on_app

```
data_time_on_app_md = tibble(
  time_on_app = df$time_on_app,
  terms_time_on_app = terms[, "time_on_app"],
  partial_resid_time_on_app = partial_resid_md[, "time_on_app"]
)
```

```
ggplot(data_time_on_app_md, mapping = aes(time_on_app, partial_resid_time_on_app)) +
  geom_point() +
  geom_smooth(method = "loess", se = FALSE, linetype = "dashed",
              color = "forestgreen") +
  geom_line(aes(x = time_on_app, y = terms_time_on_app), color = "blue") +
  labs(x = "time_on_app", y = "Partial Residuals") +
  theme_bw()
```

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



Nhận xét: Có sự tuyến tính từng phần của thành phần time_on_app trong mô hình

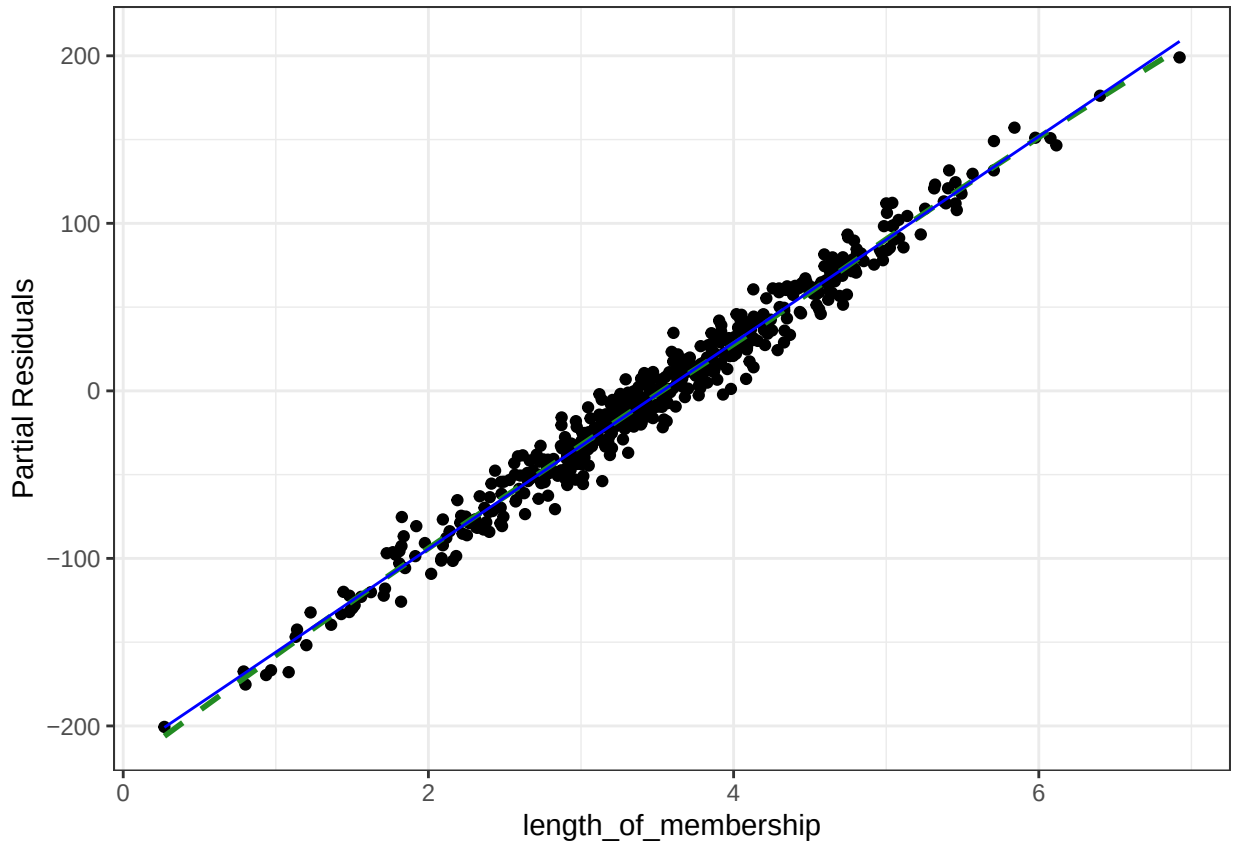
- length_of_membership

```
data_length_of_membership_md = tibble(
  length_of_membership = df$length_of_membership,
  terms_length_of_membership = terms[, "length_of_membership"],
  partial_resid_length_of_membership = partial_resid_md[, "length_of_membership"]
)
```

```
ggplot(data_length_of_membership_md,
       mapping = aes(length_of_membership,
                     partial_resid_length_of_membership)) +
  geom_point(fomular = y ~ x) +
  geom_smooth(method = "loess", se = FALSE, linetype = "dashed",
              color = "forestgreen") +
  geom_line(aes(x = length_of_membership, y = terms_length_of_membership),
            color = "blue") +
  labs(x = "length_of_membership", y = "Partial Residuals") +
  theme_bw()
```

```
## Warning in geom_point(fomular = y ~ x): Ignoring unknown parameters: `fomular`
```

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

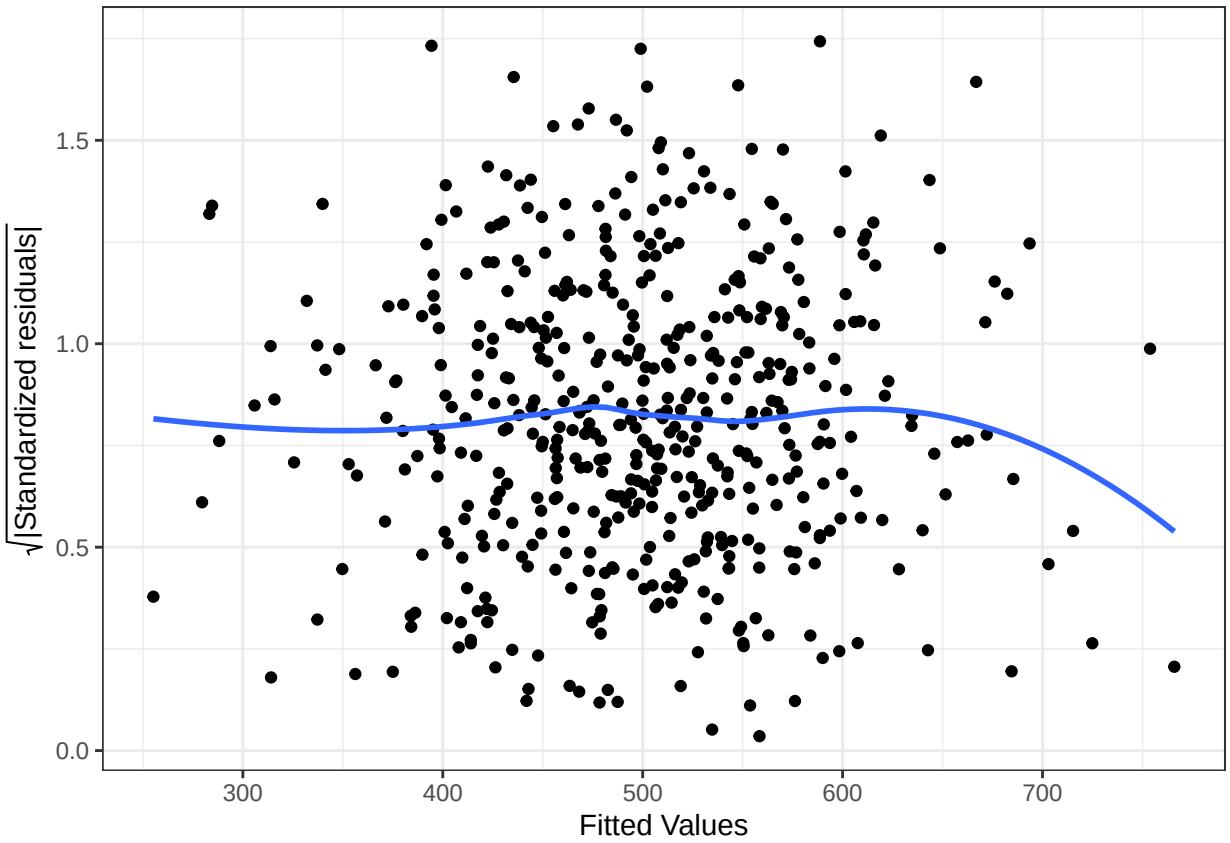


Nhận xét: Có sự tuyến tính từng phần của thành phần `length_of_membership` trong mô hình

Kiểm tra tính đồng nhất phương sai

```
ggplot(md, aes(.fitted, sqrt(abs(.stdresid)))) +
  geom_point(na.rm = TRUE) +
  geom_smooth(method = "loess", na.rm = TRUE, se = FALSE) +
  labs(x = "Fitted Values", y = expression(sqrt("|Standardized residuals|"))) +
  theme_bw()
```

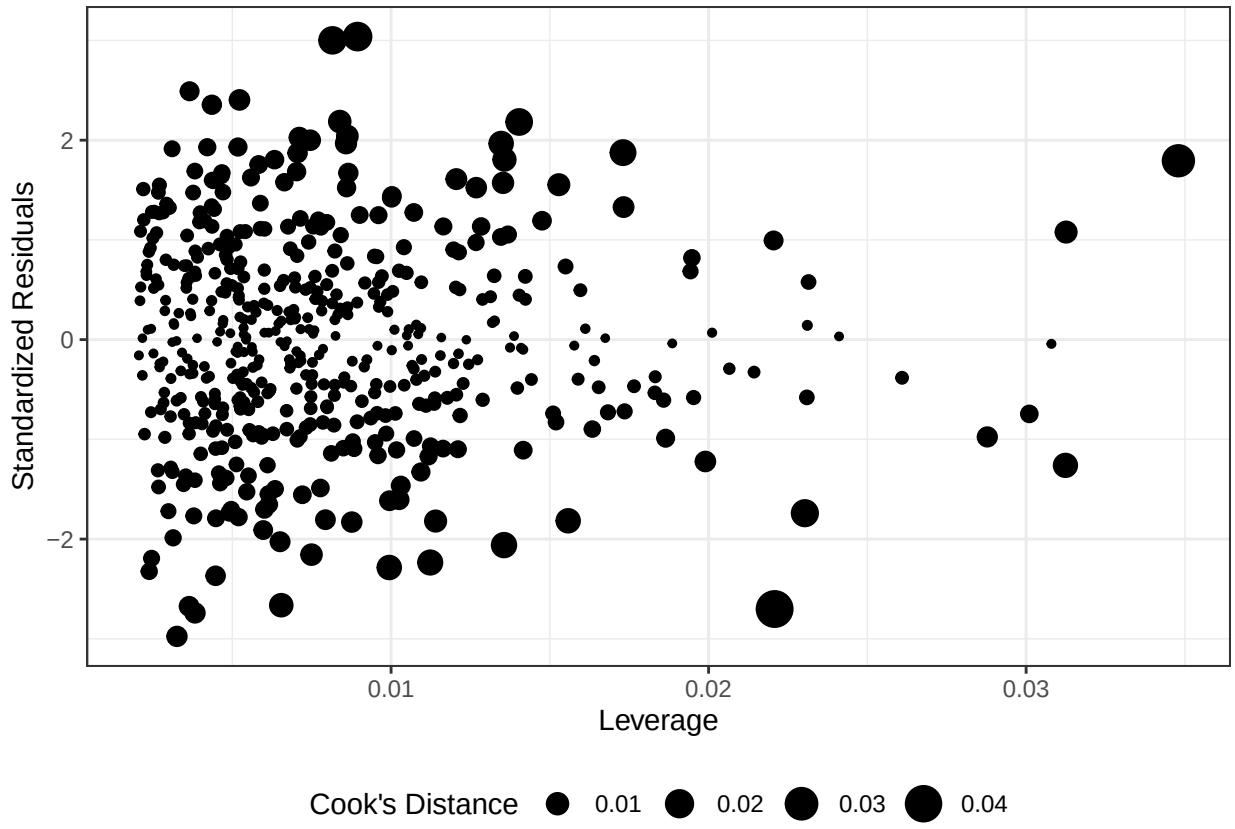
```
## `geom_smooth()` using formula = 'y ~ x'
```



Nhận xét: Hình vẽ cho thấy sự đồng nhất phương sai của mô hình

Kiểm tra điểm ngoại lai trong mô hình

```
ggplot(md, aes(.hat, .stdresid)) +
  geom_point(aes(size = .cooksd)) +
  xlab("Leverage") + ylab("Standardized Residuals") +
  scale_size_continuous("Cook's Distance", range = c(1, 6)) +
  theme_bw() +
  theme(legend.position = "bottom")
```




```
std_resid_md = rstandard(md)
hat_values_md = hatvalues(md)
cooks_D_md = cooks.distance(md)

data_cooks_md = tibble(id_point = 1:nrow(df),
                           rstand = std_resid_md, hats = hat_values_md,
                           cooks = cooks_D_md,
                       yearly_amount_spent = df$yearly_amount_spent)
data_cooks_md |> arrange(desc(cooks))
```

```
## # A tibble: 500 x 5
##    id_point rstand    hats  cooks yearly_amount_spent
##       <int>  <dbl>   <dbl>  <dbl>               <dbl>
##  1      261  -2.70 0.0221  0.0412                640.
##  2      310   1.79 0.0348  0.0290                302.
##  3      461   3.04 0.00894 0.0208                619.
##  4      149   3.00 0.00815 0.0185                424.
##  5      304  -1.74 0.0230  0.0179                266.
##  6      245   2.18 0.0140  0.0170                592.
##  7      201   1.88 0.0173  0.0155                505.
##  8      480  -2.06 0.0136  0.0146                402.
##  9      433  -2.24 0.0112  0.0142                487.
## 10      198   1.97 0.0135  0.0132                663.
## # i 490 more rows
```

Nhận xét: Vì các `Cook's Distance` của từng điểm dữ liệu trên đều nhỏ hơn 0.5 nên không có giá trị ngoại

lai.

7 Kết luận

Vì mô hình hồi quy tốt nhất là gồm 3 biến độc lập `avg_session_length`, `time_on_app`, `length_of_membership` nên ta sẽ xây dựng mô hình hồi quy tuyến tính và tính hệ số tương quan. Điều đó càng thêm khẳng định được cho giả thuyết rằng công ty nên đầu tư vào nền tảng mobile app hơn là đầu tư vào nền tảng website.

```
model = lm(yearly_amount_spent ~ avg_session_length +
             time_on_app +
             length_of_membership,
           data = df)
coefficients = coef(model)
print(coefficients)
```

```
##          (Intercept)   avg_session_length          time_on_app
##          -1035.33958             25.72103             38.74598
## length_of_membership
##              61.55603
```

Diễn giải các hệ số hồi quy:

- Khi giữ cố định các yếu tố khác, việc tăng 1 đơn vị trong Trung bình thời gian tương tác (Avg. Session Length) sẽ dẫn tới việc tăng 25,72 đơn vị của chi phí mua hàng trung bình năm.
- Khi giữ cố định các yếu tố khác, việc tăng 1 đơn vị trong thời gian tương tác trên mobile app (Time on App) sẽ dẫn tới việc tăng 38,74 đơn vị của chi phí mua hàng trung bình năm.
- Khi giữ cố định các yếu tố khác, việc tăng 1 đơn vị trong Thời gian là khách hàng thành viên (Length of Membership) sẽ dẫn tới việc tăng 61,56 đơn vị của chi phí mua hàng trung bình năm.

Hệ số càng cao, mối quan hệ giữa biến độc lập và biến mục tiêu càng mạnh, trong trường hợp này là Chi phí mua hàng trung bình năm (**Yearly Amount Spent**).

Kết luận: Công ty nên tập trung vào việc phát triển ứng dụng di động hơn là trang web. Điều này có thể được hiểu qua việc thời gian tương tác trên ứng dụng di động có mối quan hệ mạnh mẽ với chi phí mua hàng trung bình năm. Đồng thời, thời gian là khách hàng thành viên cũng ảnh hưởng mạnh mẽ đến chi phí mua hàng trung bình năm.