A/B Testing of Web design

CrisprCat

Contents

Introduction

An e-commerce company designed a new app version, which they are thinking about to implement. It was hypothesized that the new design will increase the average revenue per user (ARPU). To test if this is indeed the case it was agreed to perform an A/B test to decide if the new app version will be implemented.

From previously monitoring the ARPU, it is known that the ARPU per month is 0.12 € +/- 1.4 €.

To justify the costs of implementing the new app version it was agreed that the the new app version should lead to an increase of ARPU to 0.15 €.

The minimum detectable Effect (MDE) is 0.07 €. It was agreed upon a statistical power of 70 %.

Data

Data source: https://www.kaggle.com/datasets/sergylog/ab-test-data?select=AB_Test_Results.csv

This dataset contains user information (USER_ID), the variant of the website the user saw (VARIANT_NAME, "variant" or "control") and the revenue created by the user (REVENUE). The data was published under the Community Data License Agreement - Sharing - Version 1.0.

Setup

Installing and loading required packages

```
# Store package names, required for the analysis in a vector
packages <- c("pwr", "tidyverse", "outliers", "EnvStats", "stats" , "visualize",
              "ggpubr")

# Install packages that are not yet installed
installed_packages <- packages %in% rownames(installed.packages())
if (any(installed_packages == FALSE)) {
  install.packages(packages[!installed_packages])
}

# Load packages
invisible(lapply(packages, library, character.only = TRUE))
```

A/B test design

The null hypothesis for the A/B test reads “there is no statistical significant difference in the ARPU between the two app variants”. As a numeric value will be analysed a t-test is planned to be used to compare the ARPU of the two app versions in the statistical analysis of the experiment results.

Power analysis

Power analysis is used to determine how big the sample size for the A/B experiment has to be to detect a statistical significant change in the outcome metric. The power analysis requires to define:

- the MDE (delta in the formula): 0.07 €
- the expected standard deviation: 1.4
- the power of test (probability that the test correctly rejects the null hypothesis): 0.70

```
# Power analysis
# Define the variables required for the power analysis
delta_1 = 0.07
sd_1 = 1.4
power_1 = 0.70

# Calculate the sample size required to detect the MDE with a 70 % probability
sample_size_1 = power.t.test(delta = delta_1, sd = sd_1, power = power_1)

# Define the width  of printed text
options(width = 60)

# Print a dynamic result statement
print(paste0("The sample size required to detect a lift by ", delta_1, " €, with an
             expected standard deviation of ", sd_1, " and a power of ", power_1,
             " is ", round(sample_size_1$n), " user per treatment."))
```

```
## [1] "The sample size required to detect a lift by 0.07 \200, with an \n             expected standar
```

Preparation for stakeholder discussion

As the sample size is highly influenced by the MDE and the desired power it can be beneficial to calculate additional sample sizes in preparation of a stakeholder discussion about the experimental design.

Thereby it can be shown that a smaller MDE or a higher power requires a much greater sample size.

```
# Generate alternative sample sizes

# Define the variables required for the power analysis with a smaller MDE
delta_2 = 0.04
sd_2 = 1.4
power_2 = 0.70

# Calculate the sample size required to detect a smaller MDE with a 70 % probability
sample_size_2 = power.t.test(delta = delta_2, sd = sd_2, power = power_2)

# Print a dynamic result statement
print(paste0("The sample size required to detect a lift by ", delta_2, " €, with an
             expected standard deviation of ", sd_2, " and a power of ", power_2,
             " is ", round(sample_size_2$n), " user per treatment, and is much
             bigger than the initial sample size of ", round(sample_size_1$n),
             " user per treatment."))
```

```
## [1] "The sample size required to detect a lift by 0.04 \200, with an\n             expected standard
```

```
# Define the variables required for the power analysis with a higher power
delta_3 = 0.07
sd_3 = 1.4
power_3 = 0.85

# Calculate the sample size required to detect a similar MDE with a higher
# probability of 85 %
sample_size_3 = power.t.test(delta = delta_3, sd = sd_3, power = power_3)

# Print a dynamic result statement
print(paste0("The sample size required to detect a lift by ", delta_3, " €, with an
             expected standard deviation of ", sd_3, " and a power of ", power_3,
             " is ", round(sample_size_3$n), " user per treatment, and is bigger
             than the initial sample size of ", round(sample_size_1$n), " user per
             treatment."))
```

```
## [1] "The sample size required to detect a lift by 0.07 \200, with an \n             expected standar
```

Random sampling

To ensure random user sampling, users have to be assigned to either the control or treatment group. In case of the new app design, this could happen before they open the app next time. However, if a certain page, for example during the buying/logout process is newly designed it is best to assign users to the control and treatment group as close as possible to when the user receives the treatment (sees the new page design). The information about the assignment of users to the control or variant is provided in the "VARIANT_NAME" variable in the dataset used for this analysis.

Run the experiment

During the experiment it should be constantly monitored to ensure a successful running experiment. On the one hand this means to monitor the random user assignment, as well as monitoring the outcome metric. In addition, further business KPIs should be monitored to ensure, that the treatment does not have an unexpected negative impact on the business.

Create visualizations to monitor the experiment

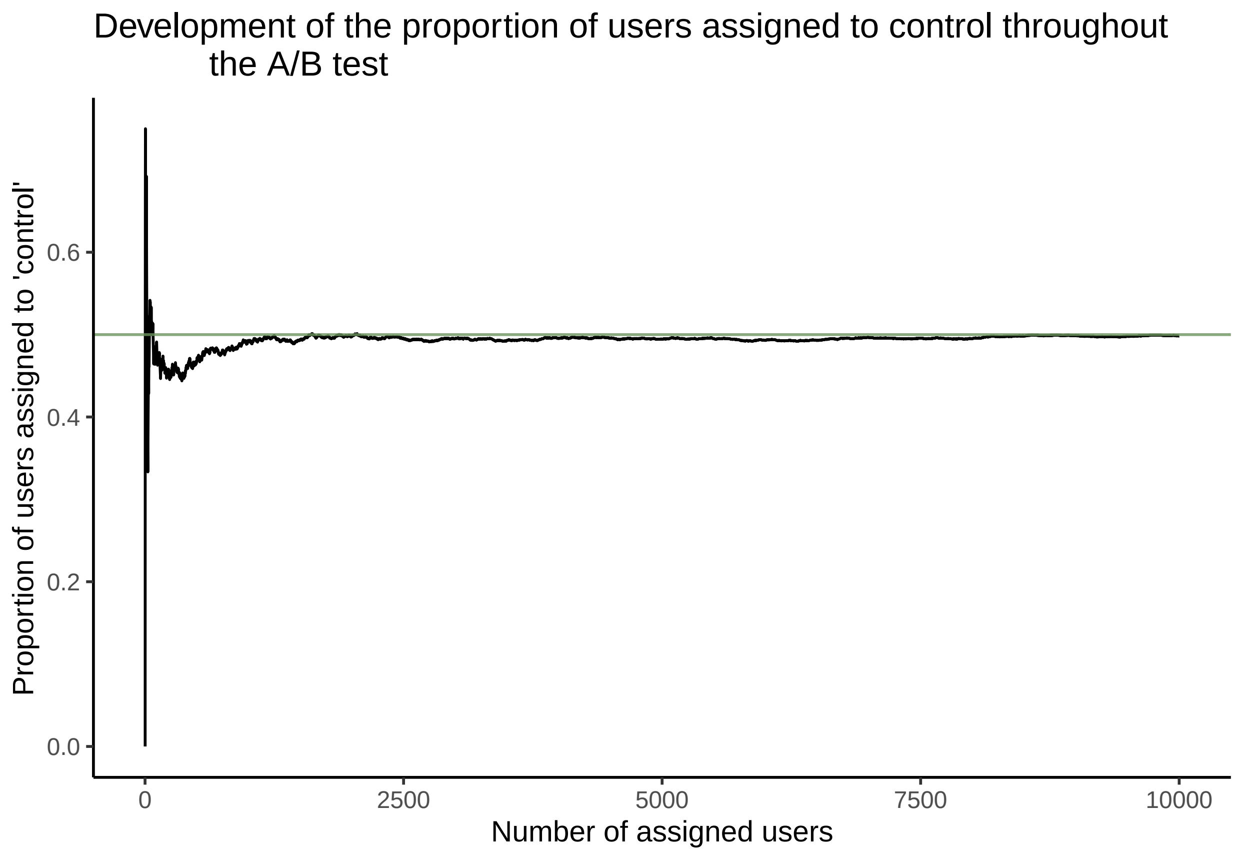
```r
# Load the data
AB_data = read.csv("AB_Test_Results.csv")
print(head(AB_data))
```

```
##   USER_ID VARIANT_NAME REVENUE
## 1     737      variant       0
## 2    2423      control       0
## 3    9411      control       0
## 4    7311      control       0
## 5    6174      variant       0
## 6    2380      variant       0
```

Monitoring random sampling

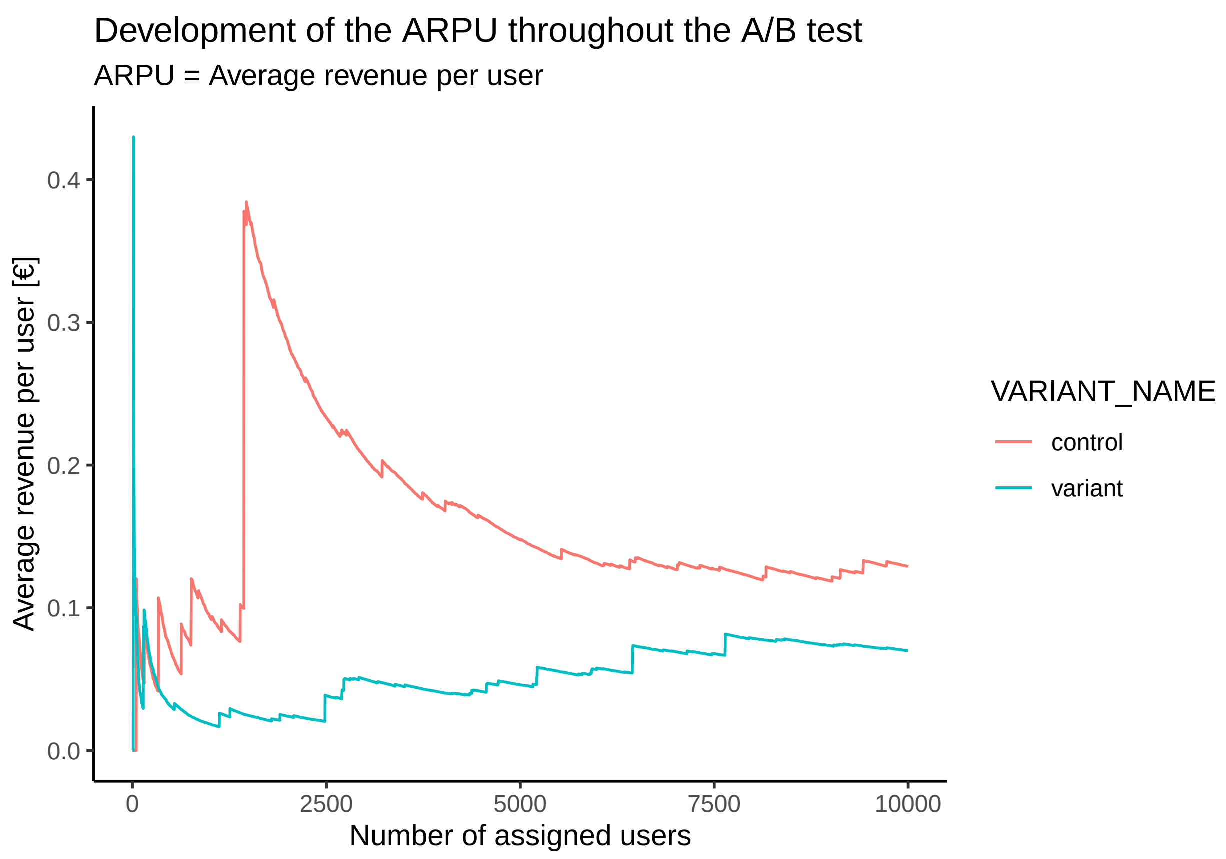
To monitor if the random sampling is working properly, the experiment could be set up as an A/A experiment (control and treatment are the same) for a few days and verify that there is no detectable statistical significant difference in the success metric. In addition, the assignment of customers to the control and treatment group can be monitored throughout the experiment. To monitor the assignment of users to the control and treatment group a new boolean value is created (1 for 'control', 0 for 'treatment'). Calculating the cumulative mean over the new variable represents the proportion of users assigned to the control group. In a **balanced experiment** this value should fluctuate around 0.5 throughout the experiment.

```r
# Calculate the proportion of Users assigned to control
# Assumption: the order of the users in the dataset is the order in which the
# observations were collected
variation = AB_data %>%
  # Create a new column and assign 1, when the user was assigned to control group
  mutate(is_control = case_when(VARIANT_NAME == "control" ~ 1,
                                TRUE ~ 0),
         # Create a new column and calculate the cumulative mean of 'is_control'
         proportion_control = cummean(is_control))

# Create a graph to visualize the cumulative mean of 'is_control' over time
proportion_control_plot = variation %>%
  ggplot(aes(x = as.numeric(row.names(variation)) , y = proportion_control)) +
    geom_line() +
    theme_classic() +
    xlab("Number of assigned users") +
    ylab("Proportion of users assigned to 'control'") +
    # add a green horizontal line at y=0.5
    geom_hline(yintercept = 0.5, color = "#6f9460", alpha = 0.8) +
    # add a title and a subtitle
    ggtitle("Development of the proportion of users assigned to control throughout
            the A/B test")

# Show the graph
print(proportion_control_plot)
```



Monitoring the success metric throughout the experiment

```
# Calculate the cumulative mean of the revenue within control and treatment group
# Assumption: the order of the users in the dataset is the order in which the
# observations were collected
ARPU = AB_data %>%
  group_by(VARIANT_NAME) %>%
  # Create a new column and calculate the cumulative mean of the 'REVENUE' column
  mutate(mean_revenue = cummean(REVENUE))

# Create a graph to visualize the cumulative mean of 'is_control' over time
ARPU_plot = ARPU %>%
  ggplot(aes(x = as.numeric(row.names(ARPU)) ,
             y = mean_revenue,
             color = VARIANT_NAME)) +
    geom_line() +
    theme_classic() +
    xlab("Number of assigned users") +
    ylab("Average revenue per user [€]") +
    # add a title and a subtitle
    ggtitle("Development of the ARPU throughout the A/B test",
            subtitle = "ARPU = Average revenue per user")

# Show the graph
print(ARPU_plot)
```



The monitoring of the ARPU throughout the experiment shows, that in each group there seems to be at least one user who made a big purchase, as seen by a strong increase in ARPU. However, there seem to be also a lot of users who don't spend any money as the ARPU goes down after the big increase again. In addition the ARPU of the control is higher than the ARPU of the new variant. However, whether this difference is statistically significant needs to be analysed after the experiment is finished.

Monitoring further Business KPIs

In parallel to the success metric other important business KPIs should be monitored throughout the experiment. This allows to detect any harm that the A/B test (new app variant) could have on a global business scale as soon as possible. In case that the harm detected gets unbearable it might be an option to preliminary terminate the experiment to not repel users from the app.

Analyzing the test results

Inspecting the data

The results of the experiment were saved in one .csv file containing the UserID, the variant of the app they have seen (control or treatment) and the revenue they generated.

```
# Show the first lines of the dataframe
print(head(AB_data))
```

```
##   USER_ID VARIANT_NAME REVENUE
## 1     737      variant       0
## 2    2423      control       0
## 3    9411      control       0
## 4    7311      control       0
## 5    6174      variant       0
## 6    2380      variant       0
```

Data cleaning

The dataset from an A/B experiment might require data cleaning, as mistakes might appear due to technical issues during the experiment or data retrieval from an SQL database. The steps of data cleaning include:

- checking for missing values
    - check if they occur at random
    - check if they can be filled in
    - if they cannot be filled in remove them for analysis
- checking for duplicates
    - remove complete duplicates
    - replace incomplete duplicates with a statistical summary of those duplicates, depending on the context
- checking for structural errors
    - in categorical data
    - find and replace inconsistent naming conventions / typos / incorrect capitalization
- checking for outliers
    - remove or correct them when they result from improper data-entry
- checking for data types
    - check if the columns contain the expected data type
    - if not, find out why, fix the issues and transform data to the correct data type

```
# Check for missing (NULL) values
number_of_NA = sum(is.na(AB_data))
# Print a dynamic result statement
print(paste0("The dataset contains ", number_of_NA, " missing values."))
```

```
## [1] "The dataset contains 0 missing values."
```

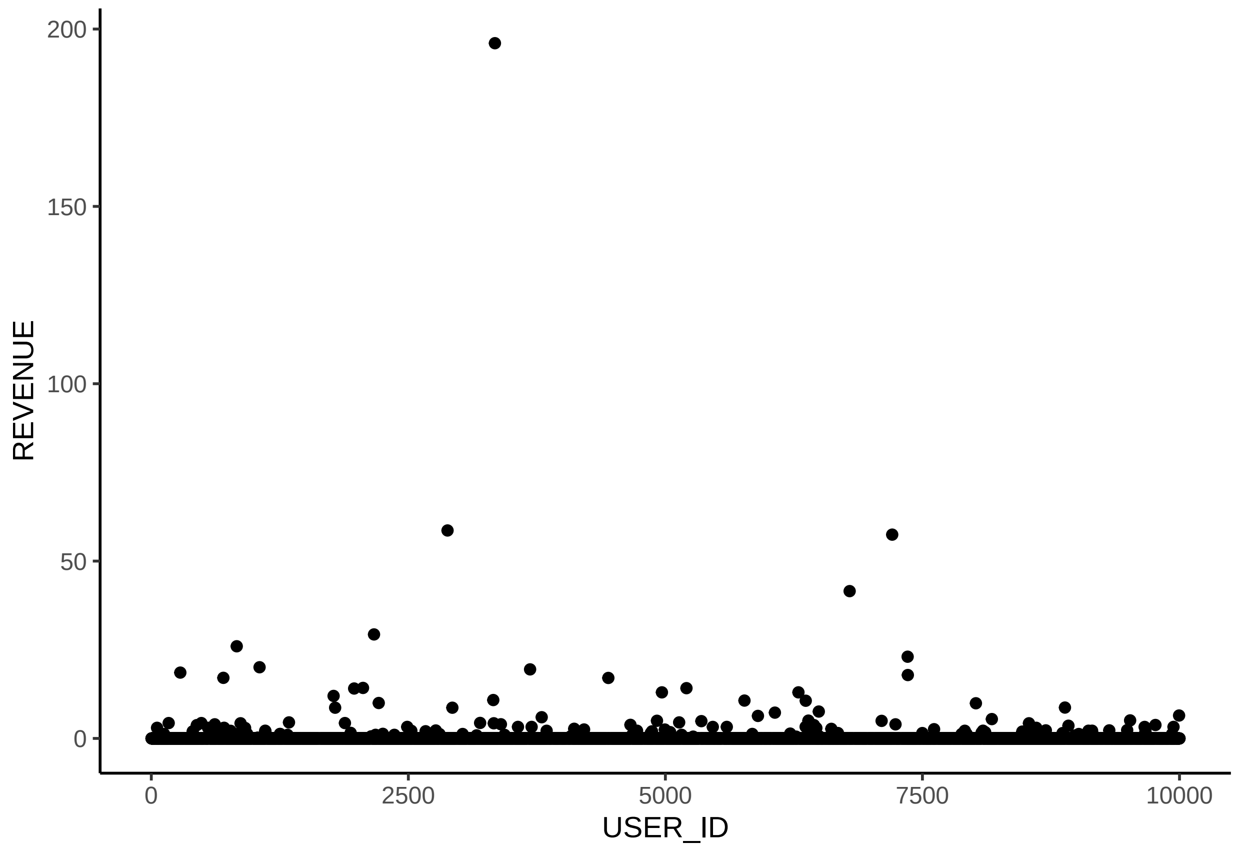
```
# Remove missing values
# For this dataset not required!
# rm_NA_data = AB_data %>%
#   na.omit() # drop rows that contain NA values
# # Print a dynamic result statement
# print(paste0(count(AB_data) - count(rm_NA_data),
# " row(s) with NA values were removed."))
```

```
# Check for duplicates
rm_duplicate_data = AB_data %>%
  distinct() # remove duplicated rows

# Print a dynamic result statement
print(paste0(count(AB_data) - count(rm_duplicate_data),
             " duplicated rows were removed."))
```

```
## [1] "2067 duplicated rows were removed."
```

```
# Check for outliers
# Graphically, create a scatterplot
scatter_plot = rm_duplicate_data %>%
  ggplot(aes(x = USER_ID, y = REVENUE)) +
  geom_point() +
  theme_classic()
```

```
# Show the graph
print(scatter_plot)
```



From the scatterplot it seems that the maximum revenue value might be an outlier. In this case one could check if there was a mistake during data acquisition. If the amount spend on this order is totally out of the normally expected range it might be worth to track this one order down in the database and see if there might be more information that would support the decision to remove or correct this one value from the data set. However, without further information this datapoint will be retained for the analysis.

```
# Check for data types
print(str(AB_data))
```

```
## 'data.frame':    10000 obs. of  3 variables:
##  $ USER_ID     : int  737 2423 9411 7311 6174 2380 2849 9168 6205 7548 ...
##  $ VARIANT_NAME: chr  "variant" "control" "control" "control" ...
##  $ REVENUE     : num  0 0 0 0 0 0 0 0 0 0 ...
## NULL
```

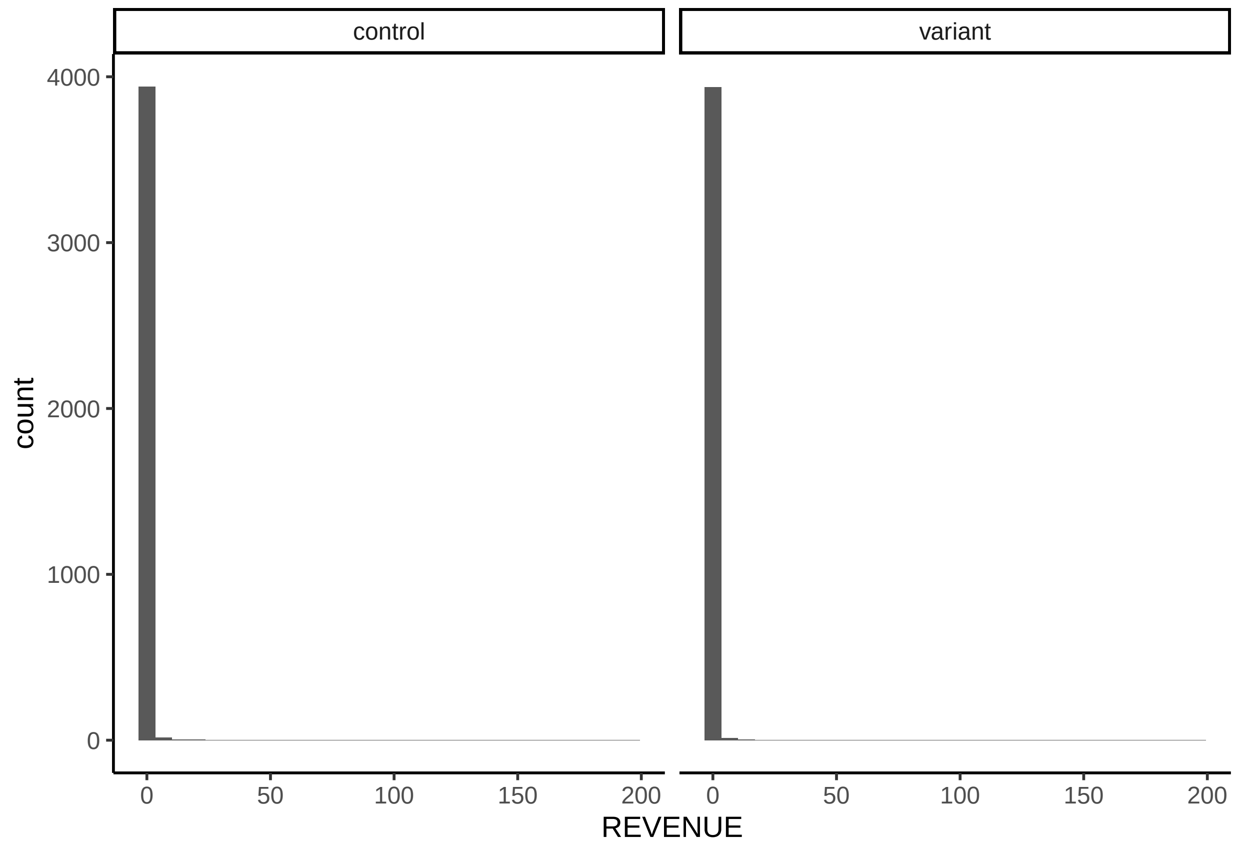
```
# Change the type of VARIANT_NAME column to 'factor'
AB_data$VARIANT_NAME = as.factor(AB_data$VARIANT_NAME)

# Check for data types again
print(str(AB_data))
```

```
## 'data.frame':    10000 obs. of  3 variables:
##  $ USER_ID     : int  737 2423 9411 7311 6174 2380 2849 9168 6205 7548 ...
##  $ VARIANT_NAME: Factor w/ 2 levels "control","variant": 2 1 1 1 2 2 1 1 2 1 ...
##  $ REVENUE     : num  0 0 0 0 0 0 0 0 0 0 ...
## NULL
```

Investigating the data distribution

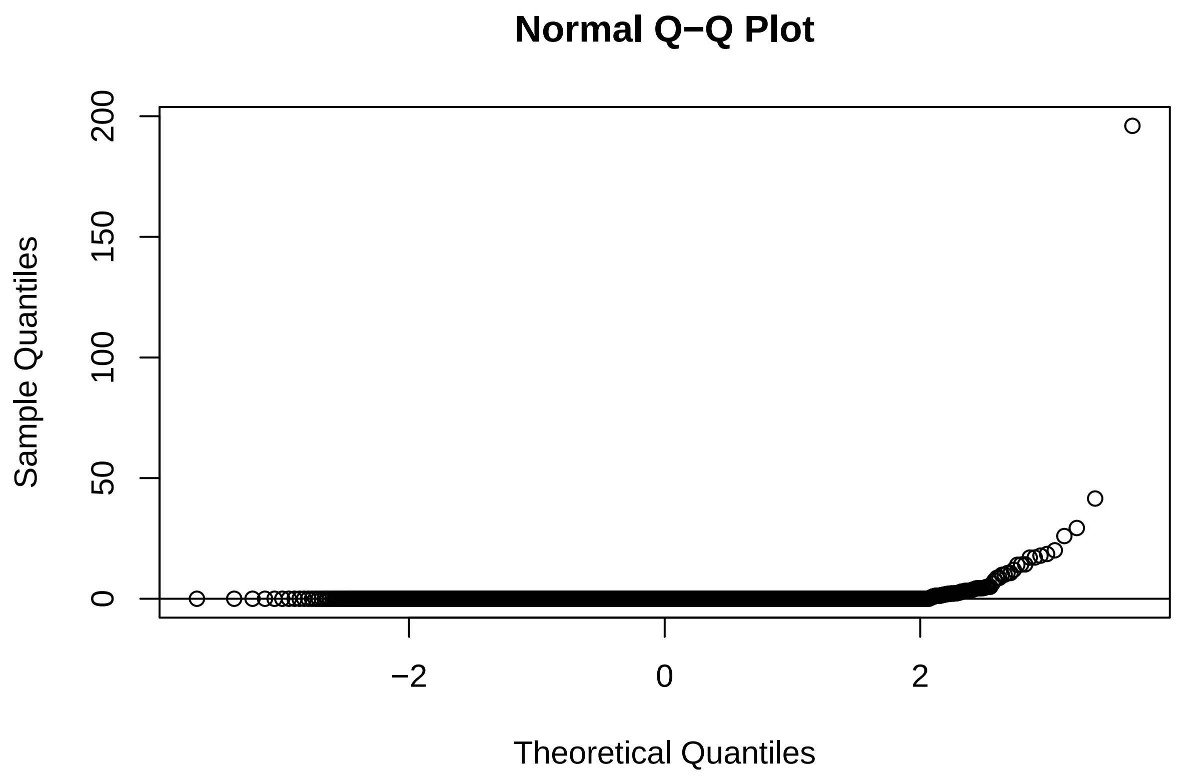
```
# Create a histogram of the data
ggplot(rm_duplicate_data, aes(x = REVENUE)) +
  geom_histogram() +
  facet_wrap("VARIANT_NAME") +
  theme_classic()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with
## `binwidth`.
```

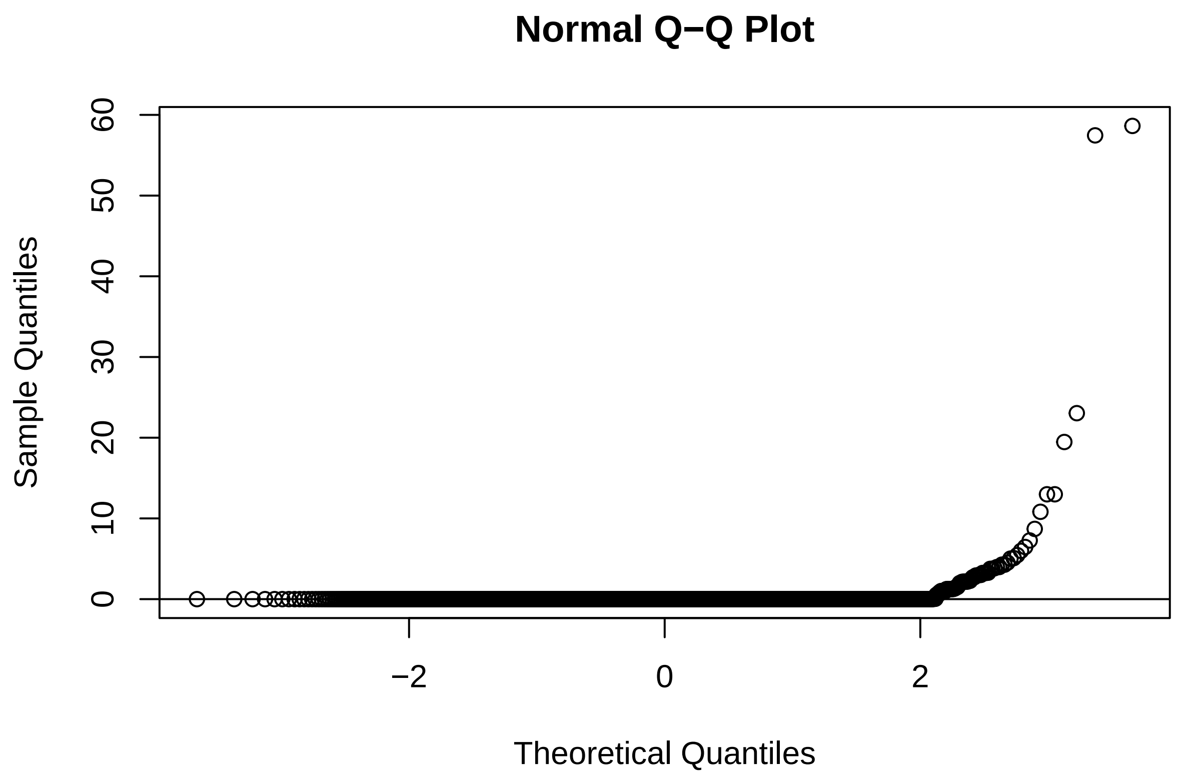


The histogram suggests that due to the high number of 0 € values in revenue the data does not follow a normal distribution. With a graphical approach (creation of a QQplot) and statistical testing we can test this hypothesis.

```
# Create a QQ-plot for the control group
qqnorm(rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == 'control'])
qqline(rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == 'control'])
```



```
# Create a QQ-plot for the treatment group
qqnorm(rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == 'variant'])
qqline(rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == 'variant'])
```



```r
# test for normal distribution
print(shapiro.test(rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == 'control']))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == "control"]
## W = 0.022009, p-value < 2.2e-16
```

```r
print(shapiro.test(rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == 'variant']))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rm_duplicate_data$REVENUE[rm_duplicate_data$VARIANT_NAME == "variant"]
## W = 0.032665, p-value < 2.2e-16
```

As the p-value in each group is < 0.05, the null hypothesis, that the samples data distribution is not different from a normal distribution can be rejected. In conclusion the data does not follow a normal distribution, so to compare the mean ARPU of the control and treatment version we will use the Mann-Whitney U test that does not assume a normal distribution.

Calculating summary statistics

```r
# calculate some descriptive statistics
summary_data = rm_duplicate_data %>%
  group_by(VARIANT_NAME) %>%
  summarise(mean = mean(REVENUE),
            sd = sd(REVENUE),
            n = n(),
            median = median(REVENUE),
            min = min(REVENUE),
            max = max(REVENUE))

print(summary_data)
```

```
##   VARIANT_NAME       mean       sd    n median min    max
## 1      control 0.16184244 3.367815 3973      0   0 196.01
## 2      variant 0.08875505 1.479239 3960      0   0  58.63
```

The calculations shows, that the average revenue per user (ARPU) is lower with the new app variant. But is this difference also statistically significant?

Mann-Whitney U test

```
# Mann-Whitney U test
res = wilcox.test(REVENUE ~ VARIANT_NAME, data = rm_duplicate_data)

print(res)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  REVENUE by VARIANT_NAME
## W = 7882388, p-value = 0.5129
## alternative hypothesis: true location shift is not equal to 0
```

As the p-value is > 0.3 the null hypothesis, that there is no difference between the ARPU of control and treatment variant of the app cannot be rejected. As we could not find a statistically significant increase of the ARPU, as agreed upon before, we cannot justify the implementation of the new app version. Instead it is advised to design a new app variant, that can be tested in a follow-up A/B test.

But is there maybe a difference in the conversion rate of users? Although this was originally not the success metric that was defined for this experiment, we can calculate it and check if there is a difference between the control and variant version.

```
# Calculate the conversion rate
conversion_rate = rm_duplicate_data %>%
  group_by(VARIANT_NAME) %>%
  mutate(conversion = case_when(REVENUE > 0 ~ 1,
                                REVENUE == 0 ~ 0)) %>%
  summarise(cr = sum(conversion) / n())

print(conversion_rate)
```

```
##   VARIANT_NAME         cr
## 1      control 0.02013592
## 2      variant 0.01818182
```

The conversion rate is lower for the variant. Is this difference statistically significant? The null hypothesis is, that there is no difference in the conversion rate between the two app version, and this can be tested with a chi squared test.

```
# perform chi squared test on conversion rate
chi_sqrd_test = rm_duplicate_data %>%
  group_by(VARIANT_NAME) %>%
  mutate(conversion = case_when(REVENUE > 0 ~ 1,
                                REVENUE == 0 ~ 0)) %>%
  summarise(didConvert = sum(conversion),
            notConvert = n() - didConvert)
print(chi_sqrd_test)
```

```
##   VARIANT_NAME didConvert notConvert
## 1      control         80       3893
## 2      variant         72       3888
```

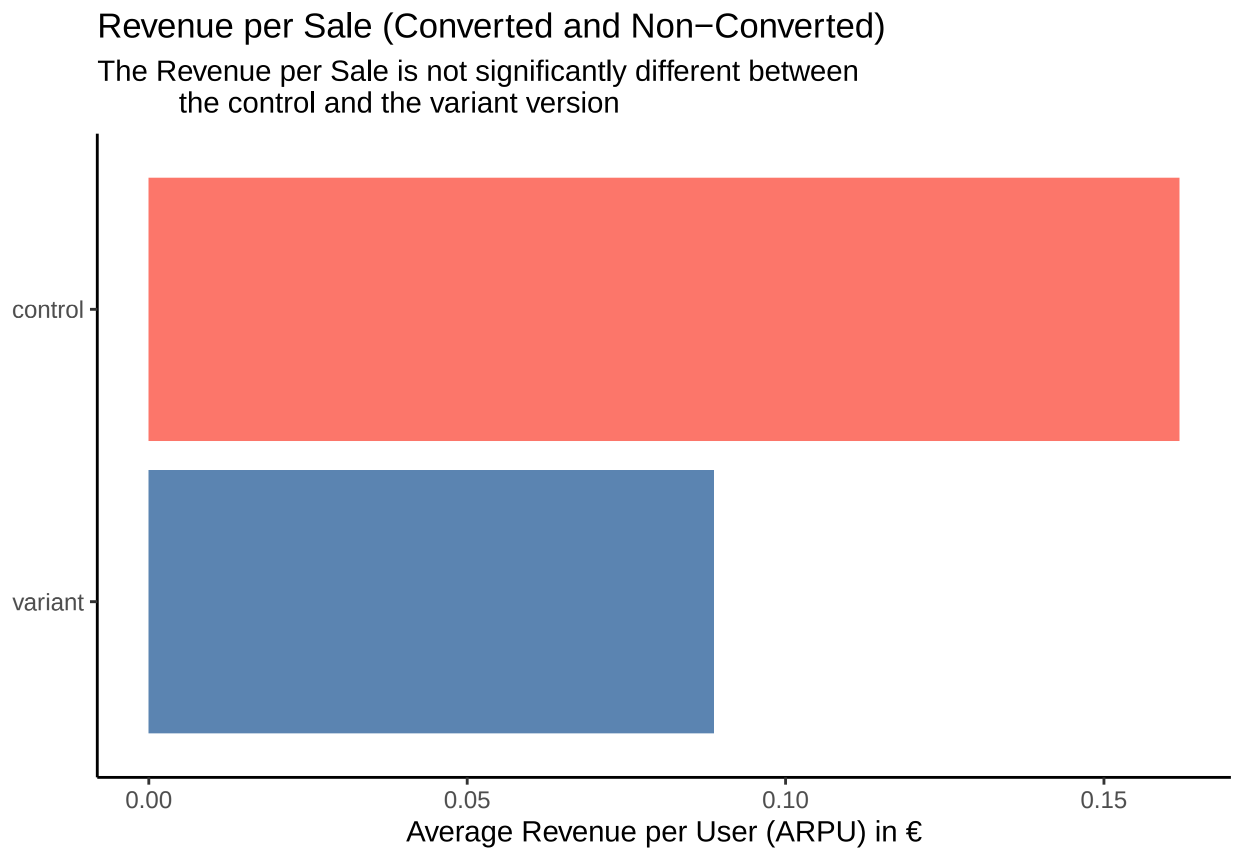
```
print(chisq.test(chi_sqrd_test[, -1]))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity
##  correction
##
## data:  chi_sqrd_test[, -1]
## X-squared = 0.30569, df = 1, p-value = 0.5803
```

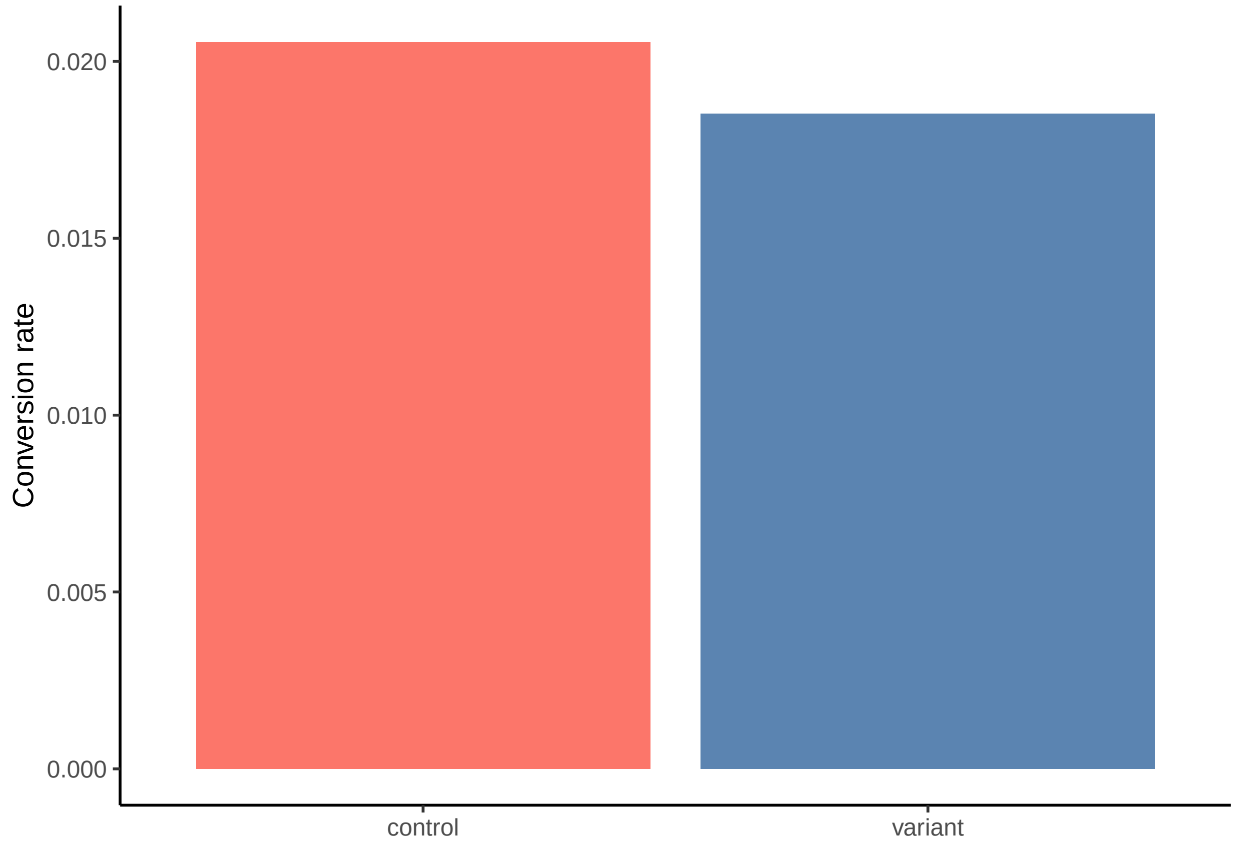
With a p-value > 0.3 the null hypothesis, that there is no difference between the conversion rate of control and treatment variant of the app cannot be rejected.

Data visualization of the test results

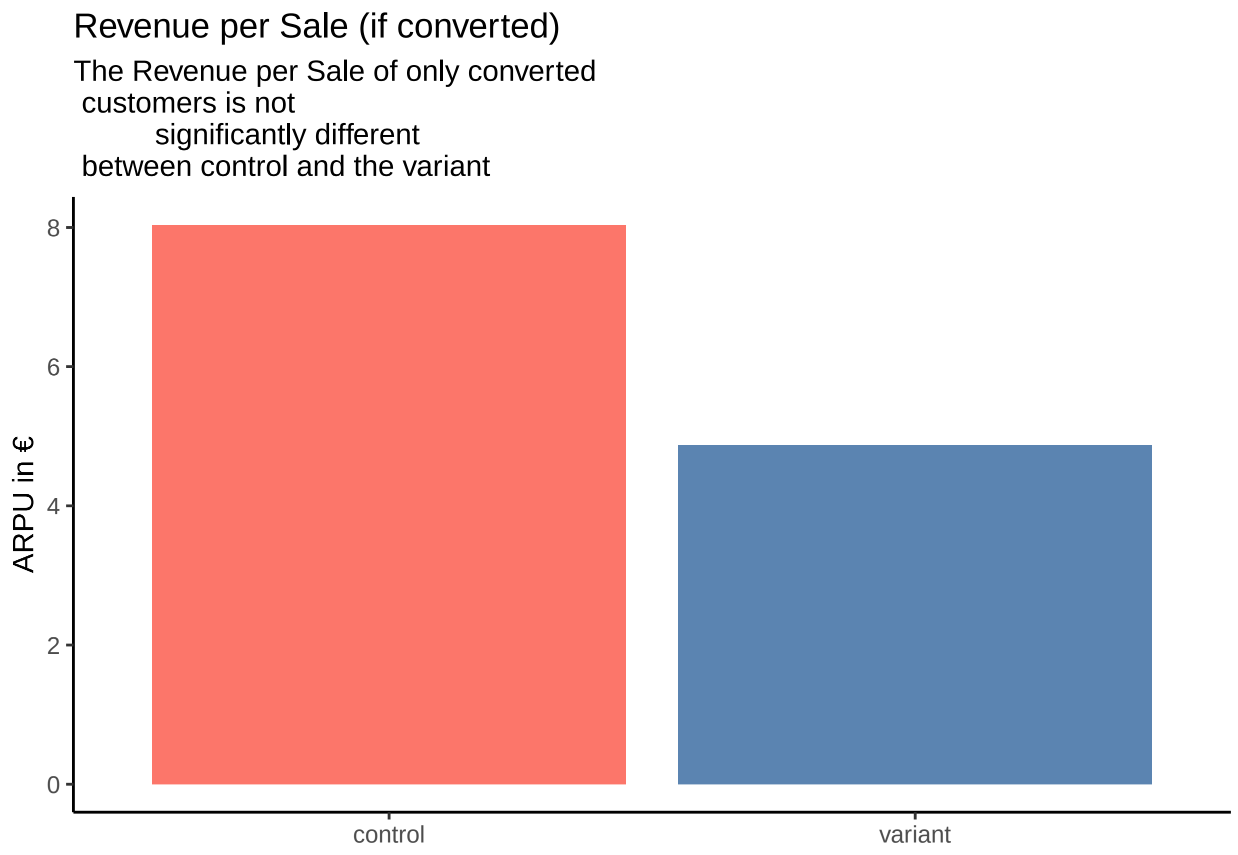
```
# Create a bar plot to visualize the ARPU
ARPU_bar_plot = summary_data %>%
  ggplot(., aes(x= VARIANT_NAME, y = mean)) +
  geom_bar(stat = "identity", fill = c("#FC766AFF", "#5B84B1FF")) +
  theme_classic() +
  xlab(NULL) +
  ylab("Average Revenue per User (ARPU) in €") +
  scale_x_discrete(limits = c("variant", "control")) +
  coord_flip() +
  ggtitle("Revenue per Sale (Converted and Non-Converted)",
          subtitle = "The Revenue per Sale is not significantly different between
          the control and the variant version")

print(ARPU_bar_plot)
```



```r
# Create a bar plot to visualize the conversion rate
conversion_rate_plot = rm_duplicate_data %>%
  group_by(VARIANT_NAME) %>%
  mutate(conversion = case_when(REVENUE > 0 ~ 1,
                               REVENUE == 0 ~ 0)) %>%
  summarise(didConvert = sum(conversion),
            notConvert = n() - didConvert) %>%
  ggplot(., aes(x = VARIANT_NAME, y = didConvert / notConvert)) +
  geom_bar(stat = "identity", fill = c("#FC766AFF", "#5B84B1FF")) +
  theme_classic() +
  xlab(NULL) +
  ylab("Conversion rate")

print(conversion_rate_plot)
```

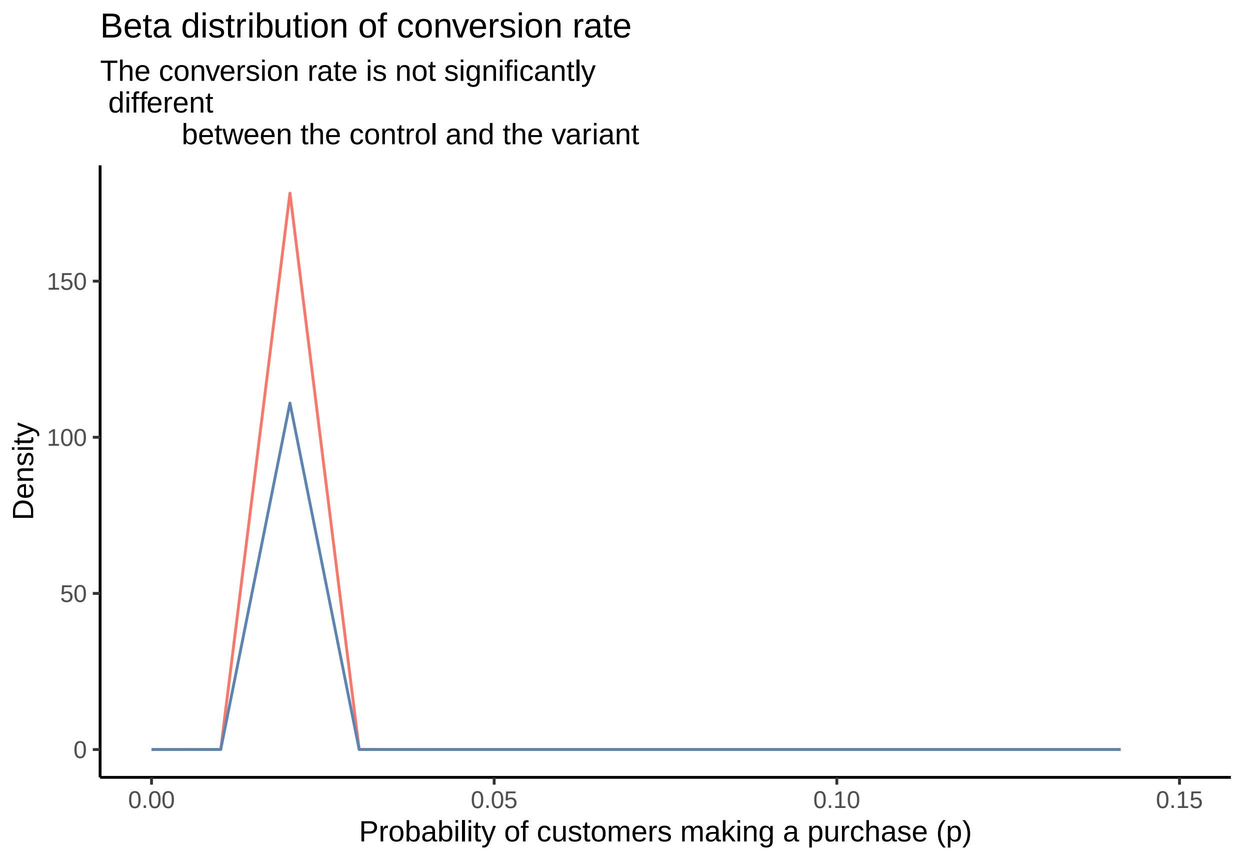


```r
# Create a bar plot to visualize the revenue per user only for converted user
ARPU_converted_only = rm_duplicate_data %>%
  subset(REVENUE > 0) %>%
  group_by(VARIANT_NAME) %>%
  summarise(mean = mean(REVENUE)) %>%
  ggplot(., aes(x= VARIANT_NAME, y = mean)) +
  geom_bar(stat = "identity", fill = c("#FC766AFF", "#5B84B1FF")) +
  theme_classic() +
  xlab(NULL) +
  ylab("ARPU in €") +
  ggtitle("Revenue per Sale (if converted)",
          subtitle = "The Revenue per Sale of only converted \n customers is not
          significantly different \n between control and the variant")

print(ARPU_converted_only)
```



```r
# Visualize the beta distribution
#define range, representing the probability of the events
p = seq(0,1, length=100)

# Define the beta distributions for Variant A and Variant B
beta_A = dbeta(p, 80, 3893)
beta_B = dbeta(p, 72, 3888)

# Create a ggplot of both beta functions
beta_plot = ggplot() +
  geom_line(aes(p, beta_A), color = "#FC766AFF") +
  geom_line(aes(p, beta_B), color = "#5B84B1FF") +
  theme_classic() +
  xlim(0, 0.15) + # change limits of x-axis
  xlab("Probability of customers making a purchase (p)") +
  ylab("Density") +
  ggtitle("Beta distribution of conversion rate",
          subtitle = "The conversion rate is not significantly \n different
          between the control and the variant")

print(beta_plot)
```

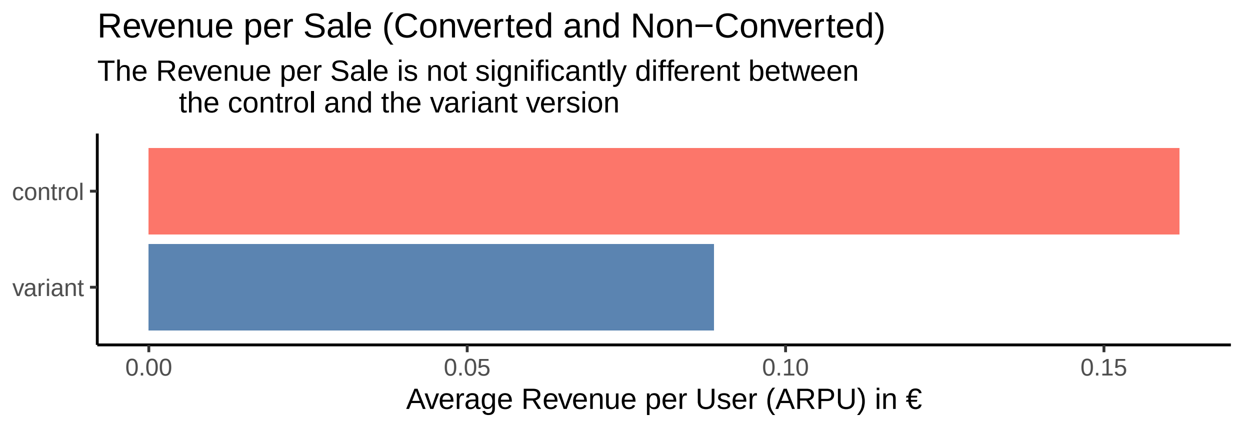
```
## Warning: Removed 85 row(s) containing missing values (geom_path).
## Removed 85 row(s) containing missing values (geom_path).
```



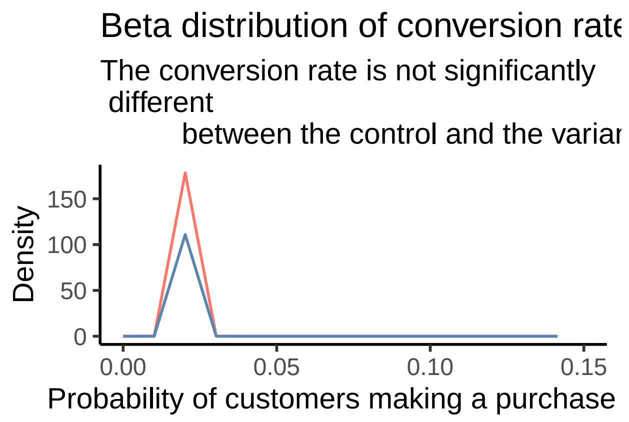
```r
# Combine the graphs to one final figure
final_figure = ggarrange(ARPU_bar_plot,
                         ggarrange(beta_plot, ARPU_converted_only, ncol = 2),
                         nrow = 2)
```

```
## Warning: Removed 85 row(s) containing missing values (geom_path).
## Removed 85 row(s) containing missing values (geom_path).
```

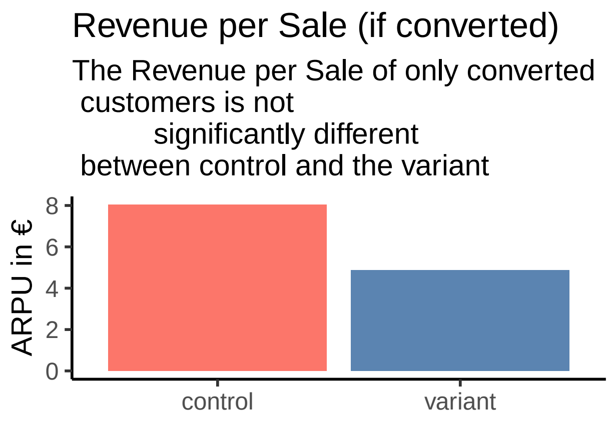
```r
print(final_figure)
```



Revenue per Sale (Converted and Non−Converted)
The Revenue per Sale is not significantly different between
the control and the variant version
control
variant
0.00
0.05
0.10
0.15
Average Revenue per User (ARPU) in €



Beta distribution of conversion rate
The conversion rate is not significantly
different
between the control and the variar
Density
150
100
50
0
0.00
0.05
0.10
0.15
Probability of customers making a purchase



Revenue per Sale (if converted)
The Revenue per Sale of only converted
customers is not
significantly different
between control and the variant
ARPU in €
8
6
4
2
0
control
variant