Power Analysis

12/11/2023

```
# install.packages("pwr") need to install if you don't have it yet
library(pwr)
library(data.table)
library(sandwich)
library(lmtest)

library(ggplot2)
library(knitr)
```

1 Exploring the Influence of Personality Profiles on User Engagement in Dating Apps

1.1 Project

This study investigates the user personalities displayed on dating app profiles and their subsequent engagement levels. By comparing the effects of creating a neutral personality profile with crafting a highly humorous and intriguing personality description, we aim to examine the impact of varying personality portrayals in dating app profiles (independent of accompanying photos) on user interactions and engagement.

1.2 Samples

Our study utilized samples drawn from five distinct account profiles, each created by individual team members. Each team member generated two account profiles featuring the same photo: one with a factual description (control) and another with a humorous description (treatment). The team members, on a daily basis, swiped right on 10 candidates using the app and recorded the number of matches as our treatment effect. Consequently, our daily samples consisted of 10 profiles in the control group and 10 profiles in the treatment group, summing up to 10 * 5 profiles for each category.

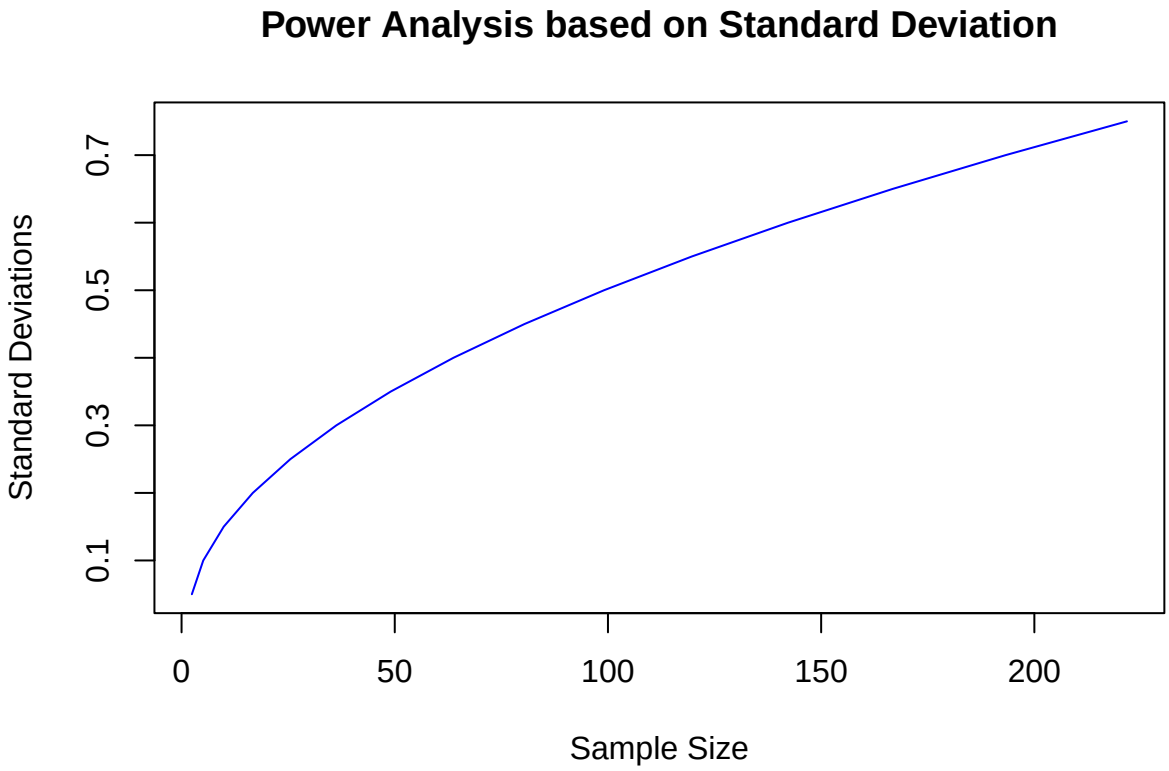
1.3 To simulate a power analysis based on data dispersion

(Need to look at the other research paper)

The provided code and plot are derived from simulating various dispersions (standard deviation), ranging from 5% to 75% with a 5% incremental increase, while aiming to achieve a consistent power level of 0.8. We assumed an ATE of .2 for this simulation, as per the previous simulation. This illustrates how the necessary sample size increases if dispersion is higher.

```
# Install and load the necessary library

# Set parameters
alpha <- 0.05           # Significance level
power <- 0.8            # Desired power
mean1 <- 0.5            # Mean of group 1(control)
mean2 <- 0.7           # Mean of group 2(humor)
std_devs <- seq(0.05, .75, by = 0.05)  # Range of standard deviations to explore
```

```r
# Initialize variables
target_sample_size <- NA
target_std_dev <- NA

# Function to calculate power for a given standard deviation and sample size
calculate_power <- function(std_dev, sample_size) {
  d <- (mean2 - mean1) / std_dev  # Cohen's d effect size
  pwr.t.test(d = d, n = sample_size, sig.level = alpha, type = "two.sample")$power
}

# Iterate through standard deviations and find the combination that achieves the desired power
for (std_dev in std_devs) {
  # Use a range of sample sizes for exploration
  for (sample_size in seq(10, 500, by = 10)) {
    current_power <- calculate_power(std_dev, sample_size)

    if (current_power >= power) {
      target_sample_size <- sample_size
      target_std_dev <- std_dev
      break
    }
  }

  if (!is.na(target_sample_size)) {
    break
  }
}

# Function to calculate sample size for a given standard deviation
calculate_sample_size <- function(std_dev, alpha, power) {
  d <- (mean2 - mean1) / std_dev  # Cohen's d effect size
  pwr.t.test(d = d, sig.level = alpha, power = power, type = "two.sample")$n
}

# Calculate sample sizes for each standard deviation
sample_sizes <- sapply(std_devs, calculate_sample_size, alpha = alpha, power = power)

# Plot the results
plot(sample_sizes, std_devs, type = "l", col = "blue",
     xlab = "Sample Size", ylab = "Standard Deviations",
     main = "Power Analysis based on Standard Deviation")
```



The blue line shows the sample sizes need to uphold an 80% power based on varying amounts of standard deviations. It makes sense that as dispersion increases, you would need increasing amounts of sample sizes to ensure the ATE is not due to randomization.