Domača naloga 8

Alen Kahteran

7. 11. 2020

Definicija funkcije

```
# function for generating 3 class normally distributted data with mean m1, m2 and m3,
# where standard deviation is 1.
generiraj_podatke <- function(n1, n2, n3, m1, m2, m3) {
    ### returns a data.frame nrow length = n1 + n2 + n3 with randomly distributed data
    ### such that m1, m2, m3 are population/sample means, while n1, n2, n3 is the size of
    ### vectors with corresponding means + white noise.
    ###
    ### Input:
    ###     n1 - Size of sample with m1 mean
    ###     n2 - Size of sample with m2 mean
    ###     n3 - Size of sample with m3 mean
    ###     m1 - mean of sample #1
    ###     m2 - mean of sample #2
    ###     m3 - mean of sample #3

    sample1 <- rnorm(n1, mean=m1, sd=1)
    sample2 <- rnorm(n2, mean=m2, sd=1)
    sample3 <- rnorm(n3, mean=m3, sd=1)
    sample1 <- cbind(sample1, m1, 1)
    sample2 <- cbind(sample2, m2, 2)
    sample3 <- cbind(sample3, m3, 3)

    full_data <- rbind(sample1, sample2, sample3)

    colnames(full_data) <- c("value", "m", "grp")

    return(tibble(data.frame(full_data)))
}
```

Določanje začetnih spremenljivk

```
# number of replicate repetitions
m_reps <- 1000

# size of each class data
n1 <- 100
n2 <- 100
n3 <- 100

# mean of each class data
```

```
m1 <- 0
m2 <- 0
m3 <- 0

# setting seed for reproducibility
set.seed(9)
```

Generiranje p vrednosti ko $m_1 = m_2 = m_3$

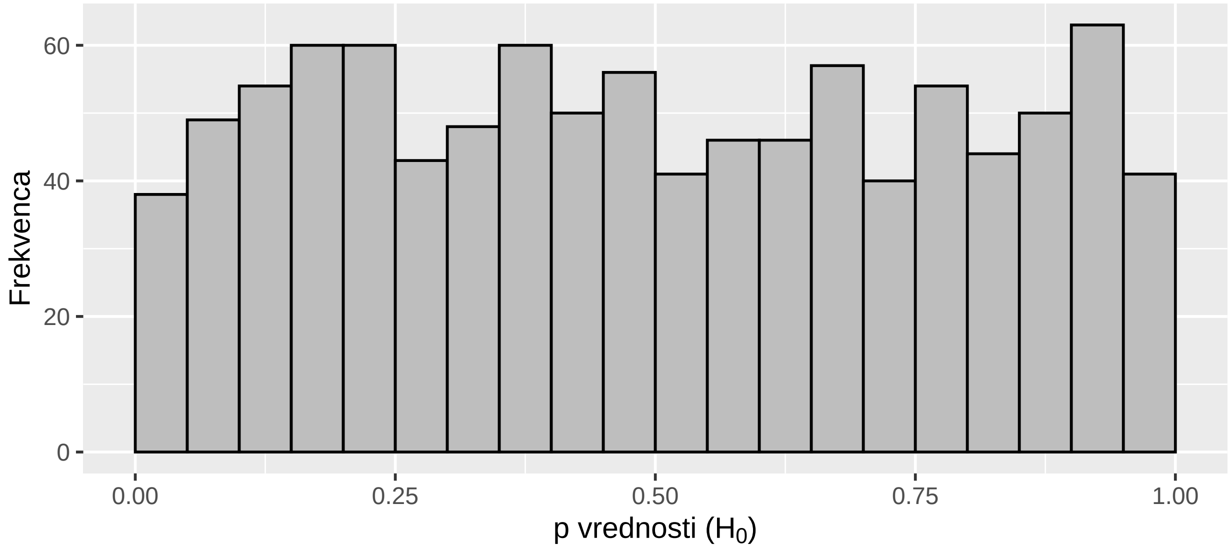
```
# repeat the process m_reps times to get m_reps number of p-values
# all means are the same
p_values_0 <- replicate(
    m_reps,
    summary(
        aov(value~as.factor(grp),
            data=generiraj_podatke(n1, n2, n3, m1, m2, m3)
            )
        )[[1]]$`Pr(>F)`[1])
```

Generiranje p vrednosti ko so m_1 in m_2 in m_3 različni

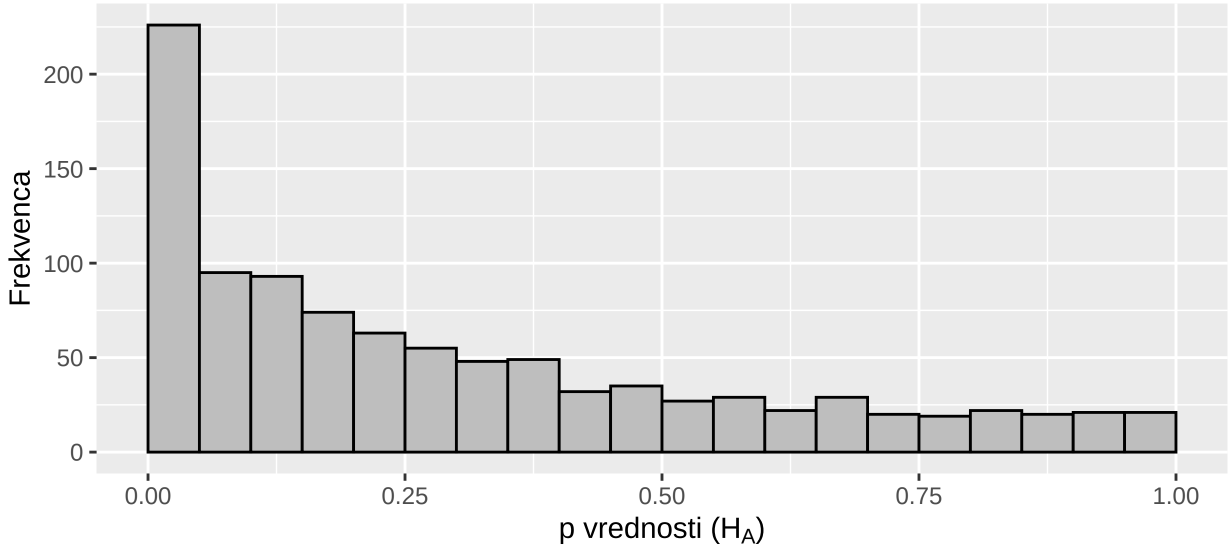
```
# changing the mean to not be the same
m1 <- -0.1
m2 <- 0
m3 <- 0.1

# repeat the process m_reps times to get m_reps number of p-values
# all means are NOT the same
p_values_a <- replicate(
    m_reps,
    summary(
        aov(value~as.factor(grp),
            data=generiraj_podatke(n1, n2, n3, m1, m2, m3)
            )
        )[[1]]$`Pr(>F)`[1])
```

Izris histogramov

```
# displaying histogram of H_0
ggplot(NULL, aes(x=p_values_0)) +
    geom_histogram(binwidth=0.05,
                   boundary=0,
                   col="black",
                   fill="gray") +
    labs(x=bquote("p vrednosti ("*H[0]*")"), y="Frekvenca")
```



```
# displaying histogram of H_A
ggplot(NULL, aes(x=p_values_a)) +
    geom_histogram(binwidth=0.05,
                   boundary=0,
                   col="black",
                   fill="gray") +
    labs(x=bquote("p vrednosti ("*H[A]*")"), y="Frekvenca")
```



Vidimo, da v prvem primeru ko m vrednosti spadajo pod ničelno hipotezo, je p vrednost uniformno porazdeljena, saj se zaradi naključnosti zna zgoditi da ničelno hipotezo kdaj zavrnemo. Ravno tako je razvidno, da v večini primerov ne moremo zavrniti ničelne hipoteze. Pri drugem grafu je pa ravno obratno. Videti je da je približno četrtina primerov v delu kjer lahko zavrnemo hipotezo ($p \in [0, 0.5]$). Videti je tudi, da se višja p vrednost manjkrat pojavi.