Exercise 4

Miriam Fischer

12 10 2020

We first load our data

```
load("C:/Users/Miriam/Box Sync/Masters/IncompleteDataAnalysis/Assignment1_MF/databp.Rdata")
```

4a: Complete case analysis

We create a dataset only of complete observations

```
data_complete <- databp[complete.cases(databp), ]
```

We calculate the mean and standard error of the complete data For the standard error, we define a function calculating the error.

```
mean(data_complete$recovtime)
```

```
## [1] 19.27273
```

```
# sterror = st deviation divided by squareroot of length of data
sterror <- function(x) sd(x)/sqrt(length(x))
sterror(data_complete$recovtime)
```

```
## [1] 2.603013
```

To calculate the pearson correlation, we need to first calculate the correlation matrix (excluding missing cases). We then divide the corresponding entry by the product of the two variables standard deviation. We note than when calculating their standard deviation, we must use only the entries included in our complete case analysis.

```
covcc <- cov(databp, use = "complete")
```

correlation between recovery and dose

```
covcc["logdose", "recovtime"]/(sd(data_complete$logdose) * sd(data_complete$recovtime))
```

```
## [1] 0.2391256
```

correlation between time and bpressure

```
covcc["bloodp", "recovtime"]/(sd(data_complete$bloodp) * sd(data_complete$recovtime))
```

```
## [1] -0.01952862
```

4b: Mean imputation

We first observe that the time is the only variable with missing observations, we thus only need to perform imputation on this variable

```
mean_recov <- mean(databp$recovtime, na.rm = TRUE)
recov_mi <- ifelse(is.na(databp$recovtime), mean_recov, databp$recovtime)
```

Mean and st error of recovtime with mean imputation

```
data_mi <- data.frame(logdose = databp$logdose, bloodp = databp$bloodp, recovtime = recov_mi)

mean(data_mi$recovtime)
```

```
## [1] 19.27273
```

```
sterror(data_mi$recovtime)
```

```
## [1] 2.284135
```

We again calculate the pearson correlation, this time on our new dataset with the missing values filled in with the mean (and we don't need to say we only want complete observations as the data is complete)

```
covmi <- cov(data_mi)
```

correlation between recovery and dose

```
covmi["logdose", "recovtime"]/(sd(data_mi$logdose) * sd(data_mi$recovtime))
```

```
## [1] 0.2150612
```

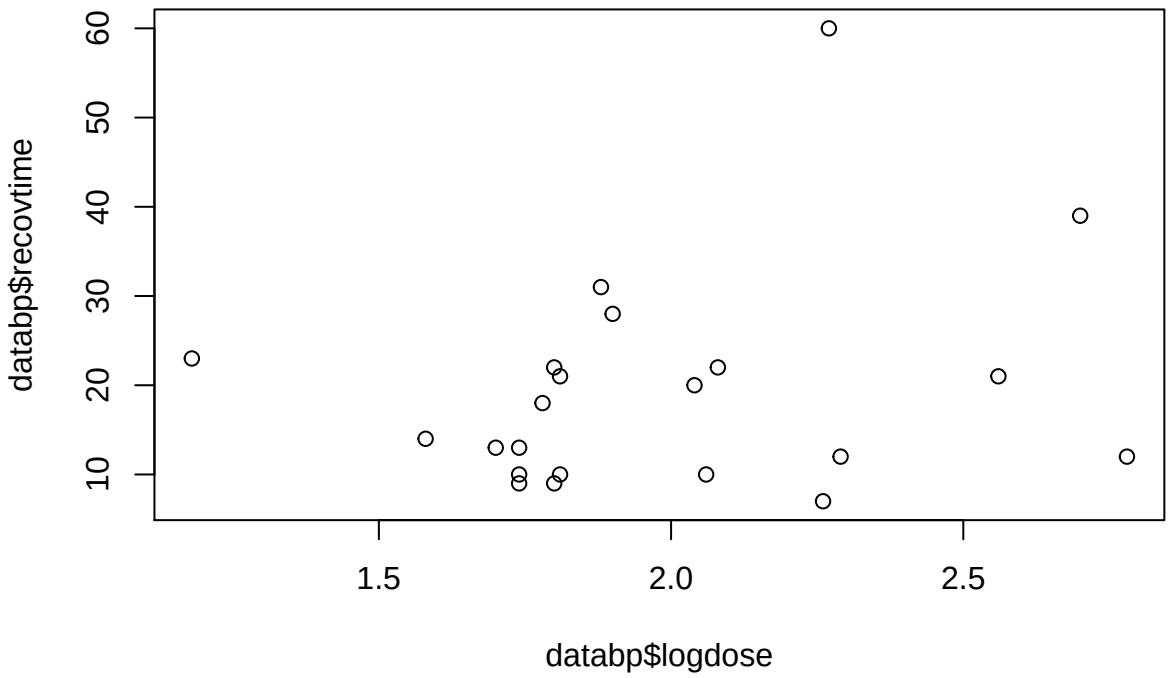
correlation between time and bpressure

```
covmi["bloodp", "recovtime"]/(sd(data_mi$bloodp) * sd(data_mi$recovtime))
```

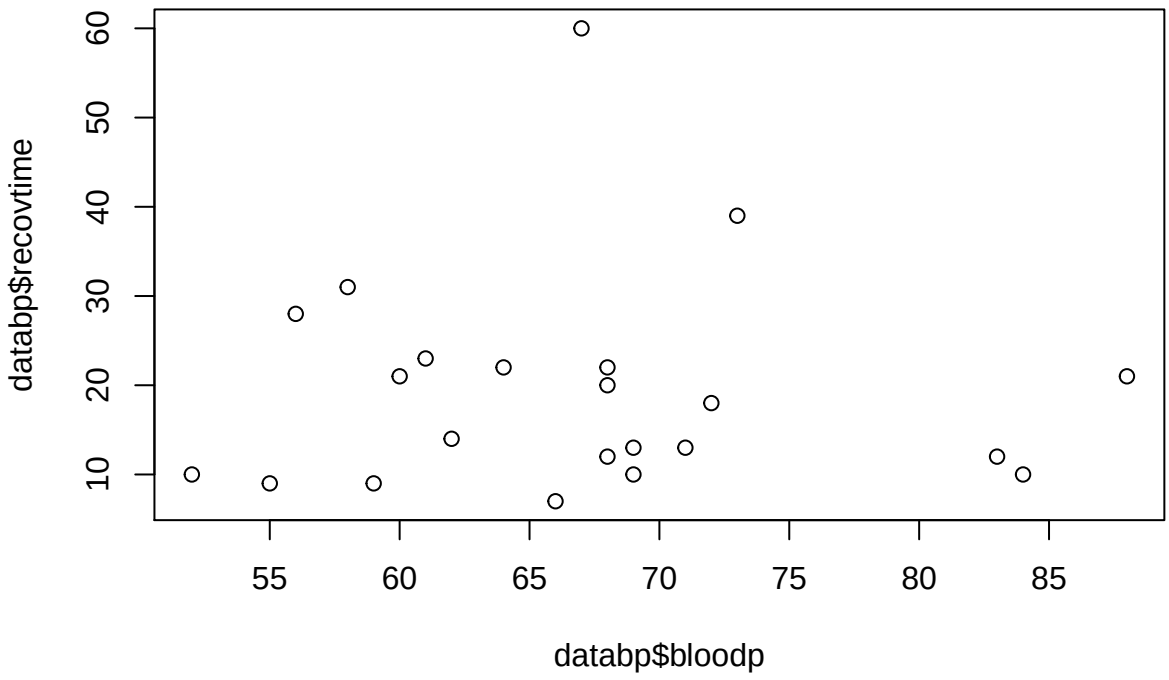
```
## [1] -0.01934126
```

4c: Mean regression imputation

```
plot(databp$logdose, databp$recovtime)
```




```
plot(databp$bloodp, databp$recovtime)
```



As we have seen a positive correlation between time and dose and a negative correlation between time and bpressure, we include both variables into our regression. Our regressionmodel is

$$\text{Recovtime}_i = \beta_0 + \beta_1 * \text{Logdose}_i + \beta_2 * \text{Bloodpressure}_i + \varepsilon_i$$

assuming $\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$. We expect $\beta_1 > 0$ and $\beta_2 < 0$.

```
fit_mr <- lm(databp$recovtime ~ databp$logdose + databp$bloodp)
fit_mr$coefficients
```

```
##    (Intercept) databp$logdose  databp$bloodp
##     15.2159065     11.4290287     -0.2769265
```

We create a new vector with the predicted times and include it to a new dataset with our new predicted values.

```
predicted_time <- predict(fit_mr, newdata = databp)
recov_predict <- ifelse(is.na(databp$recovtime), predicted_time, databp$recovtime)
data_mr <- data.frame(logdose = databp$logdose, bloodp = databp$bloodp, recovtime = recov_predict)
```

Mean and standard error of the predicted dataset

```
mean(data_mr$recovtime)
```

```
## [1] 19.44428
```

```
sterror(data_mr$recovtime)
```

```
## [1] 2.312845
```

Calculating the correlation matrix

```
covmr <- cov(data_mr)
```

correlation between recovery and dose

```
covmr["logdose", "recovtime"]/(sd(data_mr$logdose) * sd(data_mr$recovtime))
```

```
## [1] 0.2801835
```

correlation between time and bpressure

```
covmr["bloodp", "recovtime"]/(sd(data_mr$bloodp) * sd(data_mr$recovtime))
```

```
## [1] -0.0111364
```

4d:

We use the model of 3c and add a random number

```
set.seed(49)

predicted_sri <- predict(fit_mr, newdata = databp) + rnorm(nrow(databp), 0, sigma(fit_mr))
recov_sri <- ifelse(is.na(databp$recovtime), predicted_sri, databp$recovtime)
```

We calculate the mean and standard error

```
data_sri <- data.frame(logdose = databp$logdose, bloodp = databp$bloodp, recovtime = recov_sri)
mean(data_sri$recovtime)
```

```
## [1] 18.72278
```

```
sterror(data_sri$recovtime)
```

```
## [1] 2.341176
```

Calculating the correlation matrix

```
covsri <- cov(data_sri)
```

correlation between recovery and dose

```
covsri["logdose", "recovtime"]/(sd(data_sri$logdose) * sd(data_sri$recovtime))
```

```
## [1] 0.251039
```

correlation between time and bpressure

```
covsri["bloodp", "recovtime"]/(sd(data_sri$bloodp) * sd(data_sri$recovtime))
```

```
## [1] -0.01514032
```

As we include a random error term, we need to make sure that we don't predict negative times. Further, we might consider to limit the fractional part of the predicted values.

4e: predictive mean

```
mis_index <- which(is.na(databp$recovtime))

best_replace <- function(val) {

    min <- .Machine$integer.max
```

```
    ind <- -1
    available_data <- which(is.na(databp$recovtime) == FALSE)

    for (i in available_data) {
        squaredis <- (val - databp$recovtime[i])^2
        if (squaredis < min) {
            min <- squaredis
            ind <- i
        }
    }
    return(ind)
}

recov_prmtp <- 1:length(databp$recovtime)

for (inde in mis_index) {
    best <- best_replace(predicted_sri[inde])
    recov_prmtp[inde] <- databp$recovtime[best]
    cat("For datapoint ", inde, " we take the value of ", best, "\n")
}
```

```
## For datapoint  4  we take the value of  1
## For datapoint  10  we take the value of  7
## For datapoint  22  we take the value of  3
```

```
recov_prm <- ifelse(is.na(databp$recovtime), recov_prmtp, databp$recovtime)
```

4f:

One advantage of predictive mean matching is that we can be certain that our predicted values are values which comply with the pattern of the given variable. With that I mean that if our variable is time, and has only integer values, predictive mean matching will also only predict integer values (as it simply takes already given values). Stochastic regression imputation on the other side calculates the value with regression, and might predict non-integer values. However, predictive mean matching can have similar disadvantages than mean imputation: using already existing values makes the data less variable, leading to changes in the standard deviation of the data.