Title

Course

Christian Oppegård Moen

DD MM YYYY

Contents

Data 1
Load and reformat . . . 1
Make response . . . 5
Response (empirical mean and sd) . . . 5

Make building data 7

GLMM fit on errytin' 8

GLM fit on Svartid 8

GLM fit on noise types 9

Initial data observations 10

Linear models 12

Building relations 18

```
library(glmm)
library(dplyr)
```

Data

Dataformat is of tab separated values regarding noise on campus. Some variables are shown in the printout below. The response is `spm3` and `spm4` which are "trivsel" and "effektivitet", respectively. The response is evaluated for each building block.

Load and reformat

```
pathData = "./data"
d = read.delim("./data/data-315297-2023-02-22-0953-utf.txt", header = T)

# Time
formatTime <- function(t) {
    tSplit = strsplit(t, " ")[[1]]
    s = 0
    for (i in seq(1, length(tSplit), 2)) {
        s = s + switch(tSplit[i + 1], dager = strtoi(tSplit[i]) * 24 * 3600, dag = strtoi(tSplit[i]) *
            24 * 3600, timer = strtoi(tSplit[i]) * 3600, time = strtoi(tSplit[i]) *
            3600, minutt = strtoi(tSplit[i]) * 60, minutter = strtoi(tSplit[i]) *
            60, sekunder = strtoi(tSplit[i]), sekund = strtoi(tSplit[i]), 0)
    }
    return(s)
}

ftimes = unlist(lapply(d$Svartid, formatTime))
cbind(d$Svartid, ftimes)
```

```
##                                         ftimes
##   [1,] "2 minutter 32 sekunder"         "152"
##   [2,] "1 minutt 58 sekunder"           "118"
##   [3,] "2 minutter 32 sekunder"         "152"
##   [4,] "4 minutter 28 sekunder"         "268"
##   [5,] "2 minutter 45 sekunder"         "165"
##   [6,] "2 minutter 41 sekunder"         "161"
##   [7,] "11 minutter 46 sekunder"        "706"
##   [8,] "1 minutt 5 sekunder"            "65"
##   [9,] "3 minutter 50 sekunder"         "230"
##  [10,] "3 minutter 4 sekunder"          "184"
##  [11,] "1 minutt"                       "60"
##  [12,] "1 minutt 21 sekunder"           "81"
##  [13,] "1 minutt 5 sekunder"            "65"
##  [14,] "2 minutter 17 sekunder"         "137"
##  [15,] "2 minutter"                     "120"
##  [16,] "1 minutt 42 sekunder"           "102"
##  [17,] "1 minutt 17 sekunder"           "77"
##  [18,] "1 minutt 46 sekunder"           "106"
##  [19,] "3 minutter 57 sekunder"         "237"
##  [20,] "2 minutter 7 sekunder"          "127"
##  [21,] "57 sekunder"                    "57"
##  [22,] "2 minutter 17 sekunder"         "137"
##  [23,] "1 minutt 27 sekunder"           "87"
##  [24,] "6 minutter 53 sekunder"         "413"
##  [25,] "28 minutter 51 sekunder"        "1731"
##  [26,] "1 minutt 50 sekunder"           "110"
##  [27,] "2 minutter 41 sekunder"         "161"
##  [28,] "1 minutt 57 sekunder"           "117"
##  [29,] "1 minutt 54 sekunder"           "114"
##  [30,] "2 minutter 30 sekunder"         "150"
##  [31,] "1 minutt 46 sekunder"           "106"
##  [32,] "2 minutter 38 sekunder"         "158"
##  [33,] "59 sekunder"                    "59"
##  [34,] "2 minutter 59 sekunder"        "179"
##  [35,] "1 minutt 28 sekunder"          "88"
##  [36,] "1 minutt 41 sekunder"          "101"
##  [37,] "1 minutt 58 sekunder"          "118"
##  [38,] "2 minutter 24 sekunder"        "144"
##  [39,] "2 minutter 25 sekunder"        "145"
##  [40,] "2 minutter 26 sekunder"        "146"
##  [41,] "2 minutter 19 sekunder"        "139"
##  [42,] "3 minutter 50 sekunder"        "230"
##  [43,] "2 minutter 31 sekunder"        "151"
##  [44,] "2 minutter 37 sekunder"        "157"
##  [45,] "2 minutter 49 sekunder"        "169"
##  [46,] "4 minutter 48 sekunder"        "288"
##  [47,] "3 minutter 38 sekunder"        "218"
##  [48,] "6 minutter 7 sekunder"         "367"
##  [49,] "1 minutt 45 sekunder"          "105"
##  [50,] "1 minutt 36 sekunder"          "96"
##  [51,] "1 minutt 48 sekunder"          "108"
##  [52,] "2 minutter 9 sekunder"         "129"
##  [53,] "2 minutter 17 sekunder"        "137"
##  [54,] "3 minutter 1 sekund"           "181"
##  [55,] "38 minutter 42 sekunder"       "2322"
##  [56,] "1 minutt 40 sekunder"          "100"
##  [57,] "2 minutter 12 sekunder"        "132"
##  [58,] "1 minutt 36 sekunder"          "96"
##  [59,] "2 minutter 25 sekunder"        "145"
##  [60,] "1 minutt 27 sekunder"          "87"
##  [61,] "1 minutt 45 sekunder"          "105"
##  [62,] "2 minutter 15 sekunder"        "135"
##  [63,] "1 minutt 11 sekunder"          "71"
##  [64,] "1 minutt 25 sekunder"          "85"
##  [65,] "2 minutter 47 sekunder"        "167"
##  [66,] "1 minutt 26 sekunder"          "86"
##  [67,] "2 minutter 24 sekunder"        "144"
##  [68,] "1 minutt 9 sekunder"           "69"
##  [69,] "2 minutter 13 sekunder"        "133"
##  [70,] "1 minutt 31 sekunder"          "91"
##  [71,] "2 minutter 5 sekunder"         "125"
##  [72,] "2 minutter 13 sekunder"        "133"
##  [73,] "5 minutter 50 sekunder"        "350"
##  [74,] "1 minutt 42 sekunder"          "102"
##  [75,] "2 minutter 9 sekunder"         "129"
##  [76,] "1 minutt 32 sekunder"          "92"
##  [77,] "54 sekunder"                   "54"
##  [78,] "1 minutt 7 sekunder"           "67"
##  [79,] "3 minutter 22 sekunder"        "202"
##  [80,] "1 minutt 24 sekunder"          "84"
##  [81,] "12 minutter 27 sekunder"       "747"
##  [82,] "5 minutter 40 sekunder"        "340"
##  [83,] "1 minutt 41 sekunder"          "101"
##  [84,] "2 minutter 1 sekund"           "121"
##  [85,] "1 minutt 40 sekunder"          "100"
##  [86,] "4 minutter 50 sekunder"        "290"
##  [87,] "55 sekunder"                   "55"
##  [88,] "1 minutt 35 sekunder"          "95"
##  [89,] "1 minutt 46 sekunder"          "106"
##  [90,] "3 minutter 47 sekunder"        "227"
##  [91,] "1 minutt 22 sekunder"          "82"
##  [92,] "5 minutter 24 sekunder"        "324"
##  [93,] "7 minutter 50 sekunder"        "470"
##  [94,] "2 minutter 4 sekunder"         "124"
##  [95,] "2 minutter 27 sekunder"        "147"
##  [96,] "6 minutter 34 sekunder"        "394"
##  [97,] "1 minutt 48 sekunder"          "108"
##  [98,] "5 minutter 46 sekunder"        "346"
##  [99,] "2 minutter"                    "120"
## [100,] "1 minutt 45 sekunder"          "105"
## [101,] "2 minutter 9 sekunder"         "129"
## [102,] "3 minutter 51 sekunder"        "231"
## [103,] "1 time 16 minutter 9 sekunder" "4569"
## [104,] "1 minutt 59 sekunder"          "119"
## [105,] "1 minutt 12 sekunder"          "72"
## [106,] "3 minutter 19 sekunder"        "199"
## [107,] "5 minutter 26 sekunder"        "326"
## [108,] "3 minutter 14 sekunder"        "194"
## [109,] "1 minutt 6 sekunder"           "66"
## [110,] "54 sekunder"                   "54"
## [111,] "4 minutter 1 sekund"           "241"
## [112,] "3 minutter 19 sekunder"        "199"
## [113,] "58 sekunder"                   "58"
## [114,] "4 minutter 18 sekunder"        "258"
## [115,] "28 minutter 53 sekunder"       "1733"
## [116,] "1 minutt 33 sekunder"          "93"
## [117,] "3 minutter 33 sekunder"        "213"
## [118,] "1 minutt 25 sekunder"          "85"
## [119,] "37 sekunder"                   "37"
## [120,] "2 minutter"                    "120"
## [121,] "6 minutter 32 sekunder"        "392"
## [122,] "3 minutter"                    "180"
## [123,] "2 minutter 28 sekunder"        "148"
## [124,] "59 sekunder"                   "59"
## [125,] "2 minutter 27 sekunder"        "147"
## [126,] "1 minutt 37 sekunder"          "97"
## [127,] "5 minutter 44 sekunder"        "344"
## [128,] "37 minutter 8 sekunder"        "2228"
## [129,] "1 minutt 55 sekunder"          "115"
## [130,] "1 minutt 18 sekunder"          "78"
## [131,] "1 minutt 10 sekunder"          "70"
## [132,] "2 minutter 3 sekunder"         "123"
## [133,] "1 minutt 57 sekunder"          "117"
## [134,] "3 minutter 26 sekunder"        "206"
```

```r
d$Svartid = ftimes

# Free text answers
```

```
write.csv(d[d[, "spmTekst"] != "", c("NR", "spmTekst")], "./data/freeTxt.csv", row.names = FALSE)
```

Make response

As mentioned, the responses are `spm3` and `spm4`.

```
# head(d) variates = names(d) yt = d$spmTriv ye = d$spmEff
locs = unique(d$spmHvor)
locs
```

```
##  [1] "elD"    "elB"    "kjel"   "elE"    "real"   "sent2"  "verk"   "bygg"  
##  [9] "sent1"  "hand"   "tapir"  "berg"   "gamEl"  "hoved"  "varme"  "gruve" 
## [17] "gamFys" "itSyd"  "kjemi1" "kjemi2" "elA"    "kjemi5"
```

```
N = length(d[, 1])
ancPercentage = length(d$spmTiltak_1[d$spmTiltak_1 == "mNc"])/N
ancPercentage
```

```
## [1] 0.7686567
```

Response (empirical mean and sd)

```
buildMeans = data.frame(matrix(ncol = length(locs), nrow = 2, dimnames = list(c("trivsel",
    "effektivitet"), locs)))
buildSd = data.frame(matrix(ncol = length(locs), nrow = 2, dimnames = list(c("trivsel",
    "effektivitet"), locs)))
for (loc in locs) {
    # browser()
    buildMeans[loc] = c(mean(d$spmTriv[d$spmHvor == loc]), mean(d$spmEff[d$spmHvor ==
        loc]))
    buildSd[loc] = c(sd(d$spmTriv[d$spmHvor == loc]), sd(d$spmEff[d$spmHvor == loc]))
}
buildMeans[1:5]
```

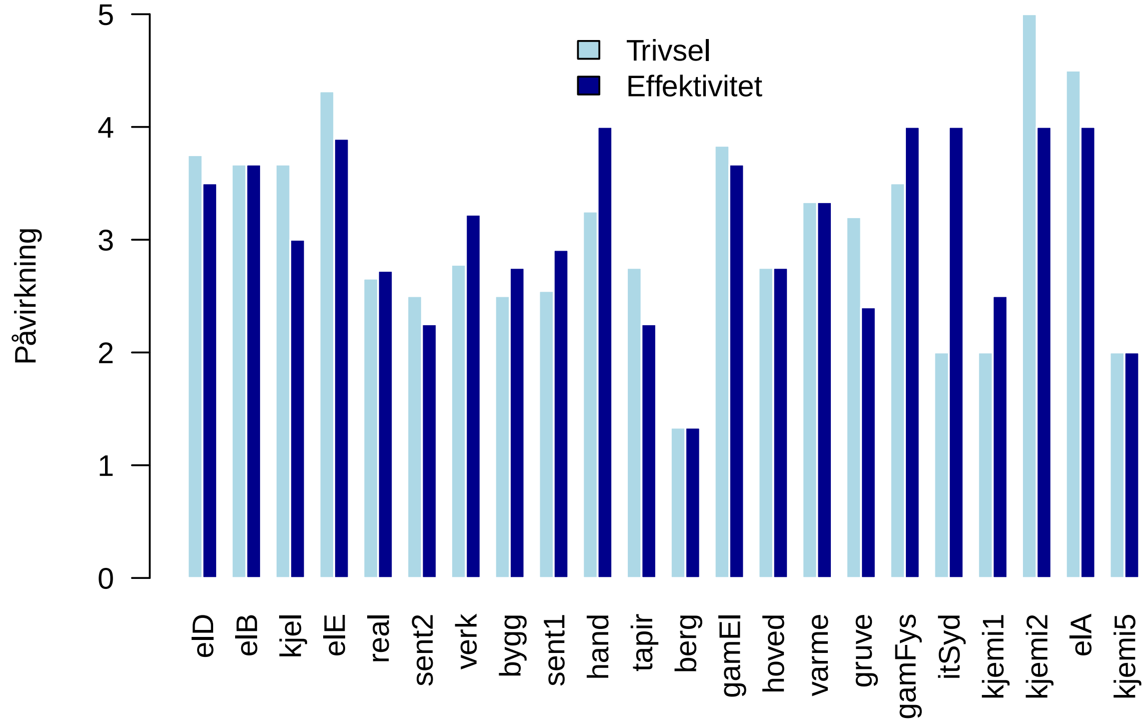
```
##                 elD      elB     kjel      elE     real
## trivsel      3.75 3.666667 3.666667 4.315789 2.655172
## effektivitet 3.50 3.666667 3.000000 3.894737 2.724138
```

```
buildSd[1:5]
```

```
##              elD       elB     kjel       elE     real
## trivsel      0.5 0.8164966 1.751190 0.8200699 1.316811
## effektivitet 1.0 1.2110601 1.414214 1.1002392 1.306483
```

```
# Write to file
# means
write.table(t(buildMeans["trivsel", ]), file = paste(pathData, "/trivselPerBygning.dat",
    sep = ""), row.names = T, sep = ",", quote = F)
write.table(t(buildMeans["effektivitet", ]), file = paste(pathData, "/effektivitetPerBygning.dat",
    sep = ""), row.names = T, sep = ",", quote = F)
# Standard deviations (sd)
write.table(t(buildSd["trivsel", ]), file = paste(pathData, "/trivselPerBygningSD.dat",
    sep = ""), row.names = T, sep = ",", quote = F)
write.table(t(buildSd["effektivitet", ]), file = paste(pathData, "/effektivitetPerBygningSD.dat",
    sep = ""), row.names = T, sep = ",", quote = F)
```

```
# æ is \u00E6
# ø is \u00F8
# å is \u00E5
barplot(
  ((as.matrix(buildMeans))),
  col = c("lightblue", 'darkblue'),
  border = "white",
  # main="Trivsel",
  ylab="P\u00E5virkning",
  beside = T,
  las=2,
  ylim = c(0,5)
  # space=0.1
)
legend(
  "top",
  legend = c("Trivsel", "Effektivitet"),
  fill = c("lightblue", 'darkblue'), bty = 'n')
```



Make building data

```
buildings = split(d, f = d$spmHvor)
for (build in buildings) {
    print(mean(build$spmTriv))
}
```

```
## [1] 1.333333
## [1] 2.5
## [1] 4.5
## [1] 3.666667
## [1] 3.75
## [1] 4.315789
## [1] 3.833333
## [1] 3.5
## [1] 3.2
## [1] 3.25
## [1] 2.75
## [1] 2
## [1] 3.666667
## [1] 2
## [1] 5
## [1] 2
## [1] 2.655172
## [1] 2.545455
## [1] 2.5
## [1] 2.75
## [1] 3.333333
## [1] 2.777778
```

GLMM fit on errytin’

```
set.seed(420)
fitAll = glmm(spmTriv ~ 0 + Svartid, varcomps.names = c(""), data = d, family.glmm = Gaussian)
dSummary = summary(d)
head(d)
```

GLM fit on Svartid

```
# Fit
fitTidTriv = glm(factor(spmTriv, seq(1, 5, 1)) ~ Svartid, data = d, family = "binomial")
fitTidEff = glm(factor(spmEff, seq(1, 5, 1)) ~ Svartid, data = d, family = "binomial")

summary(fitTidTriv)$coefficients
```

```
##                  Estimate   Std. Error   z value     Pr(>|z|)
## (Intercept)  1.9382384576 0.2764729103  7.010591 2.373138e-12
## Svartid     -0.0004750205 0.0003684438 -1.289262 1.973071e-01
```

```
summary(fitTidEff)$coefficients
```

```
##                  Estimate   Std. Error    z value     Pr(>|z|)
## (Intercept)  1.8286887821 0.2685787185  6.8087628 9.844164e-12
## Svartid     -0.0003192636 0.0003758653 -0.8494097 3.956534e-01
```

```
summary(fitTidEff)
```

```
##
## Call:
## glm(formula = factor(spmEff, seq(1, 5, 1)) ~ Svartid, family = "binomial",
##     data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9809   0.5518   0.5556   0.5606   0.7622
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.8286888  0.2685787   6.809 9.84e-12 ***
## Svartid     -0.0003193  0.0003759  -0.849    0.396
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 112.94  on 133  degrees of freedom
## Residual deviance: 112.30  on 132  degrees of freedom
## AIC: 116.3
##
## Number of Fisher Scoring iterations: 4
```

GLM fit on noise types

```
factors = c("spmTriv", "spmEff", "byggestoy1", "personstoy1", "trafikk1", "vifte1",
    "annet1", "byggestoy2", "personstoy2", "trafikk2", "vifte2", "annet2")
dFactored = d
# factorize = function()
for (f in factors) {
    dFactored[f, ] = factor(d[f, ], seq(1, 5, 1), ordered = T)
}

fitTrivSource = glm(factor(spmTriv, seq(1, 5, 1), ordered = T) ~ byggestoy1 + personstoy1 +
    trafikk1 + vifte1 + annet1, data = dFactored, family = "binomial")
summary(fitTrivSource)
```

```
##
## Call:
## glm(formula = factor(spmTriv, seq(1, 5, 1), ordered = T) ~ byggestoy1 +
##     personstoy1 + trafikk1 + vifte1 + annet1, family = "binomial",
##     data = dFactored)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.58146   0.00000   0.00004   0.21799   1.78586
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.310e+00  7.180e-01  -1.824  0.06808 .
## byggestoy12  -5.326e-01  9.382e-01  -0.568  0.57021
## byggestoy13   2.046e+00  1.356e+00   1.509  0.13119
## byggestoy14   1.879e+01  5.093e+03   0.004  0.99706
## byggestoy15   3.307e+00  1.228e+00   2.694  0.00707 **
## personstoy12  2.050e+00  9.444e-01   2.171  0.02993 *
## personstoy13  1.704e+00  9.764e-01   1.745  0.08103 .
## personstoy14  3.424e+00  1.359e+00   2.520  0.01174 *
## personstoy15  2.111e+01  6.189e+03   0.003  0.99728
## trafikk12    -1.124e+00  1.643e+00  -0.684  0.49402
## trafikk13     6.090e+00  1.063e+05   0.000  0.99995
## trafikk14     2.228e+01  1.612e+04   0.001  0.99890
## trafikk15    -3.049e+01  1.108e+05   0.000  0.99978
## vifte12      -5.778e-02  9.986e-01  -0.058  0.95386
## vifte13       1.815e+01  5.972e+03   0.003  0.99758
## vifte14       1.673e+01  8.444e+03   0.002  0.99842
## vifte15       2.065e+01  9.031e+03   0.002  0.99818
## annet12       2.058e+01  4.893e+03   0.004  0.99664
## annet13       7.770e-01  1.759e+00   0.442  0.65876
## annet14      -1.713e+01  2.172e+04  -0.001  0.99937
## annet15       1.080e+01  1.067e+05   0.000  0.99992
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 109.398  on 133  degrees of freedom
## Residual deviance:  52.957  on 113  degrees of freedom
##   (12 observations deleted due to missingness)
## AIC: 94.957
##
## Number of Fisher Scoring iterations: 20
```

```
cov(d$spmTriv, d$byggestoy1)
```

```
## [1] 1.035911
```

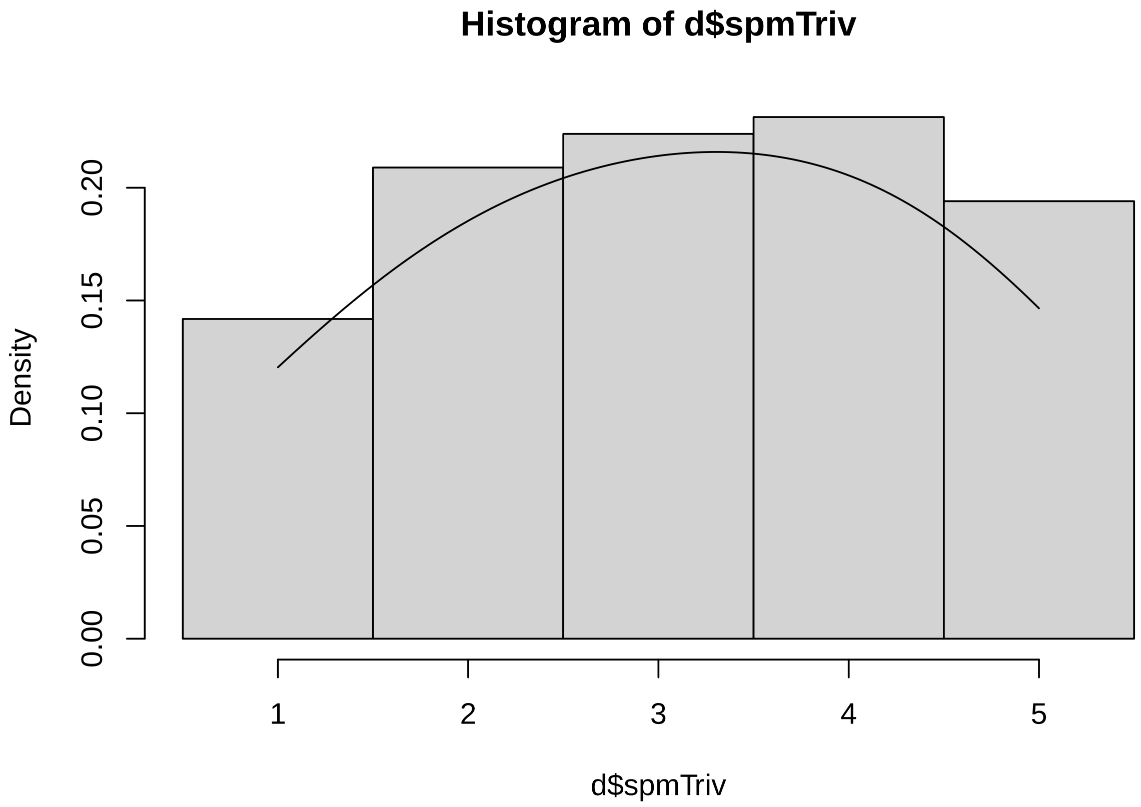
```
cov(d$spmEff, d$byggestoy2)
```

```
## [1] 1.2112
```

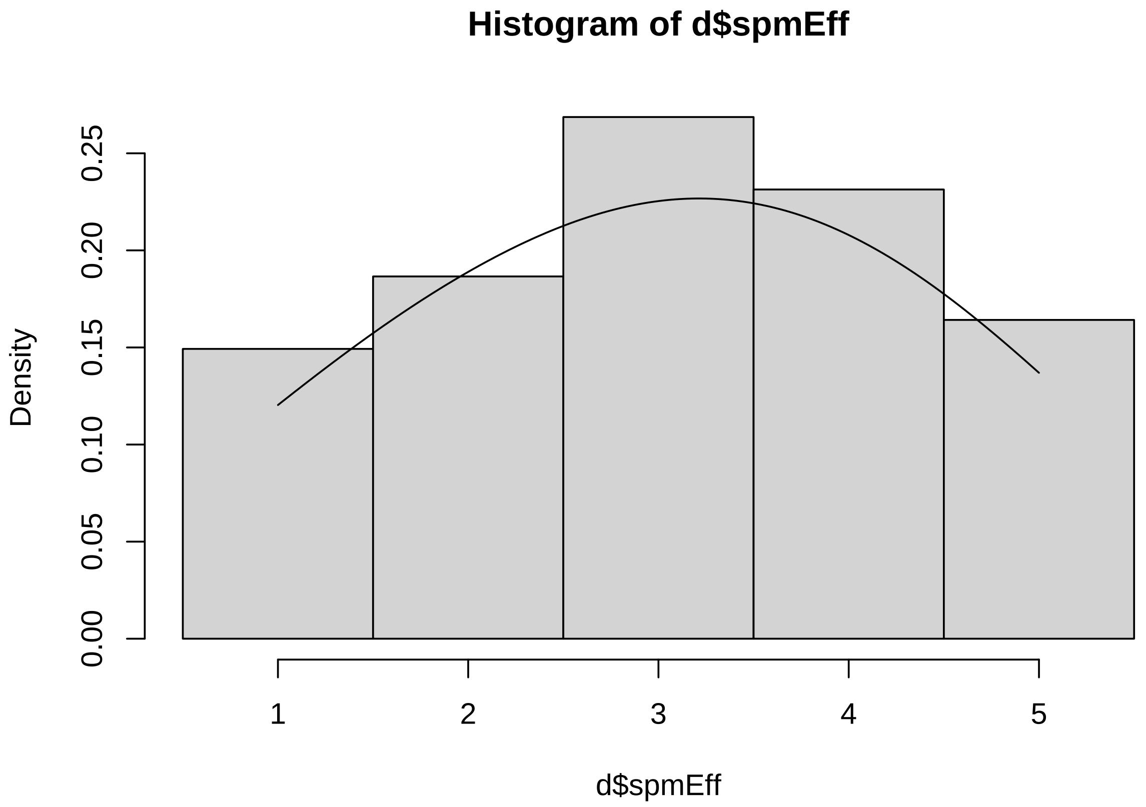
```
par(d)
```

Initial data observations

```
hist(d$spmTriv, breaks = seq(0.5, 5.5, 1), freq = F)
lines(density(d$spmTriv, bw = 1, from = 1, to = 5))
```



```
hist(d$spmEff, breaks = seq(0.5, 5.5, 1), freq = F)
lines(density(d$spmEff, bw = 1, from = 1, to = 5))
```



Linear models

```
lmTrivSource = lm(spmTriv ~ byggestoy1 + personstoy1 + trafikk1 + vifte1 + annet1,
    data = d)
summary(lmTrivSource)
```

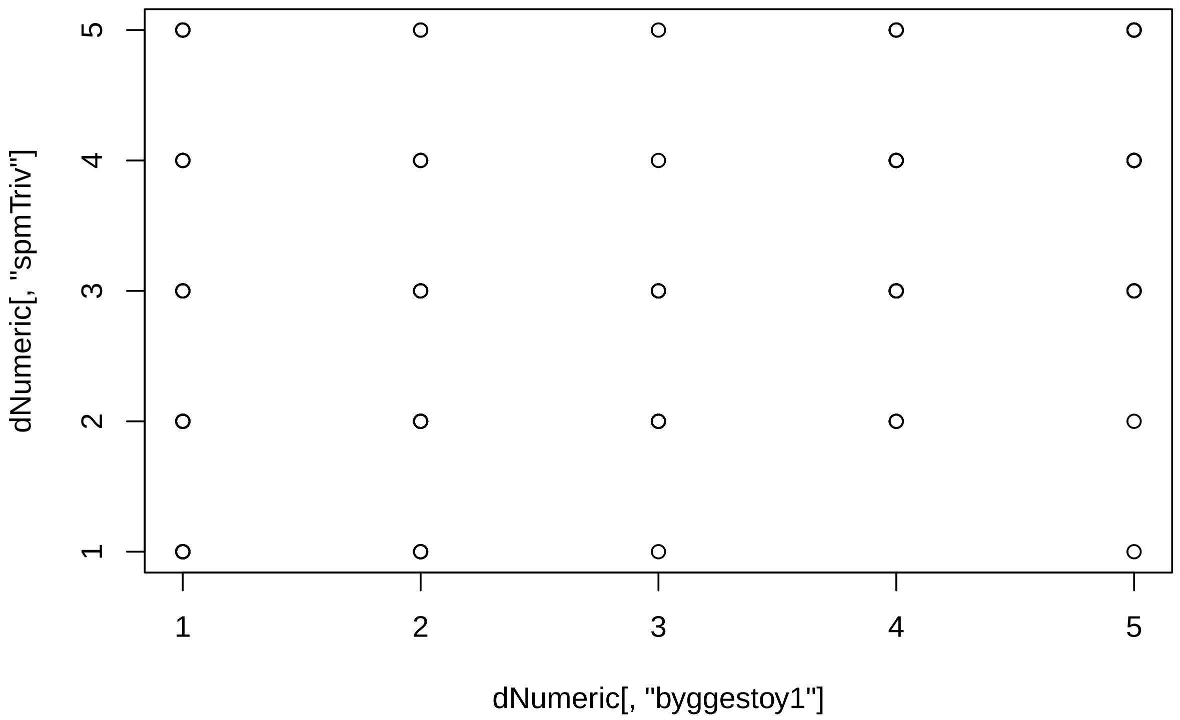
```
##
## Call:
## lm(formula = spmTriv ~ byggestoy1 + personstoy1 + trafikk1 +
##     vifte1 + annet1, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1671 -0.6546 -0.1026  0.5665  3.3842
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.68984    0.29306   2.354  0.02010 *
## byggestoy1   0.38252    0.05713   6.695 6.11e-10 ***
## personstoy1  0.32740    0.07142   4.584 1.07e-05 ***
## trafikk1    -0.16784    0.14300  -1.174  0.24270
## vifte1       0.23336    0.08733   2.672  0.00852 **
## annet1       0.15048    0.09952   1.512  0.13300
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.031 on 128 degrees of freedom
## Multiple R-squared:  0.4252, Adjusted R-squared:  0.4027
## F-statistic: 18.93 on 5 and 128 DF,  p-value: 4.625e-14
```

```
# pairs(d[,apply(d[1,],2, FUN=is.numeric)])
```

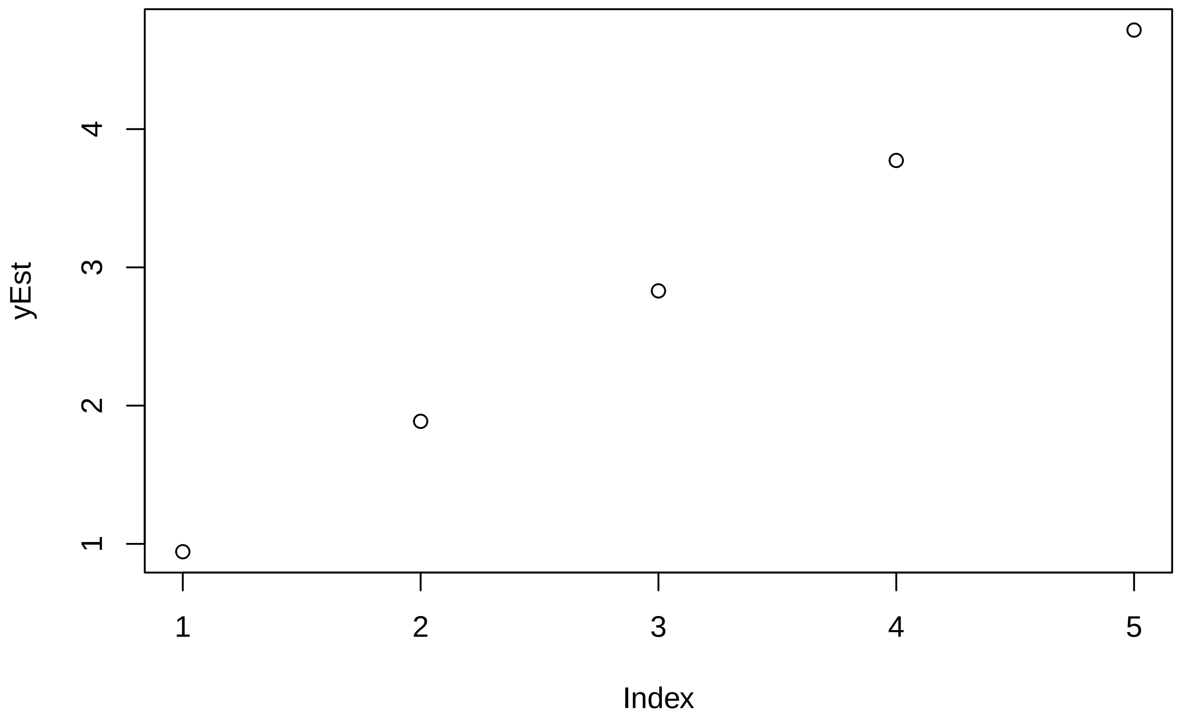
```
dNumeric = select_if(d, is.numeric)
head(dNumeric[, 2:7])
```

```
##   spmTriv byggestoy1 personstoy1 trafikk1 vifte1 annet1
## 1       4          4           2        1      2      5
## 2       3          4           1        1      3      1
## 3       1          1           2        1      1      1
## 4       3          1           2        1      1      1
## 5       4          5           2        1      1      1
## 6       3          1           4        1      3      1
```

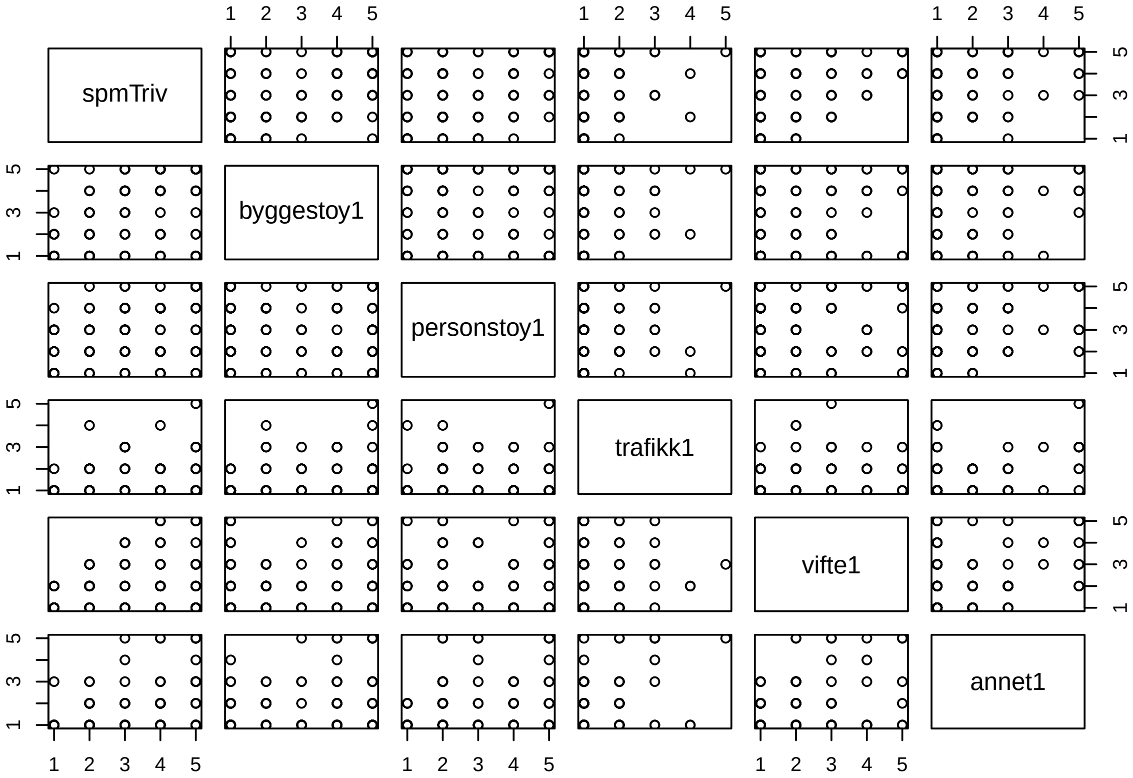
```
lmTrivSource = lm(spmTriv ~ ., data = dNumeric[2:7])
lmTrivCoefs = summary(lmTrivSource)$coefficients
lmTIntercept = lmTrivCoefs["(Intercept)", "Estimate"]
lmTBygg = lmTrivCoefs["byggestoy1", "Estimate"]
lmTPers = lmTrivCoefs["personstoy1", "Estimate"]
lmTVift = lmTrivCoefs["vifte1", "Estimate"]
yEst = lmTBygg * (1:5) + lmTPers * (1:5) + lmTVift * (1:5)
plot(dNumeric[, "byggestoy1"], dNumeric[, "spmTriv"], )
```



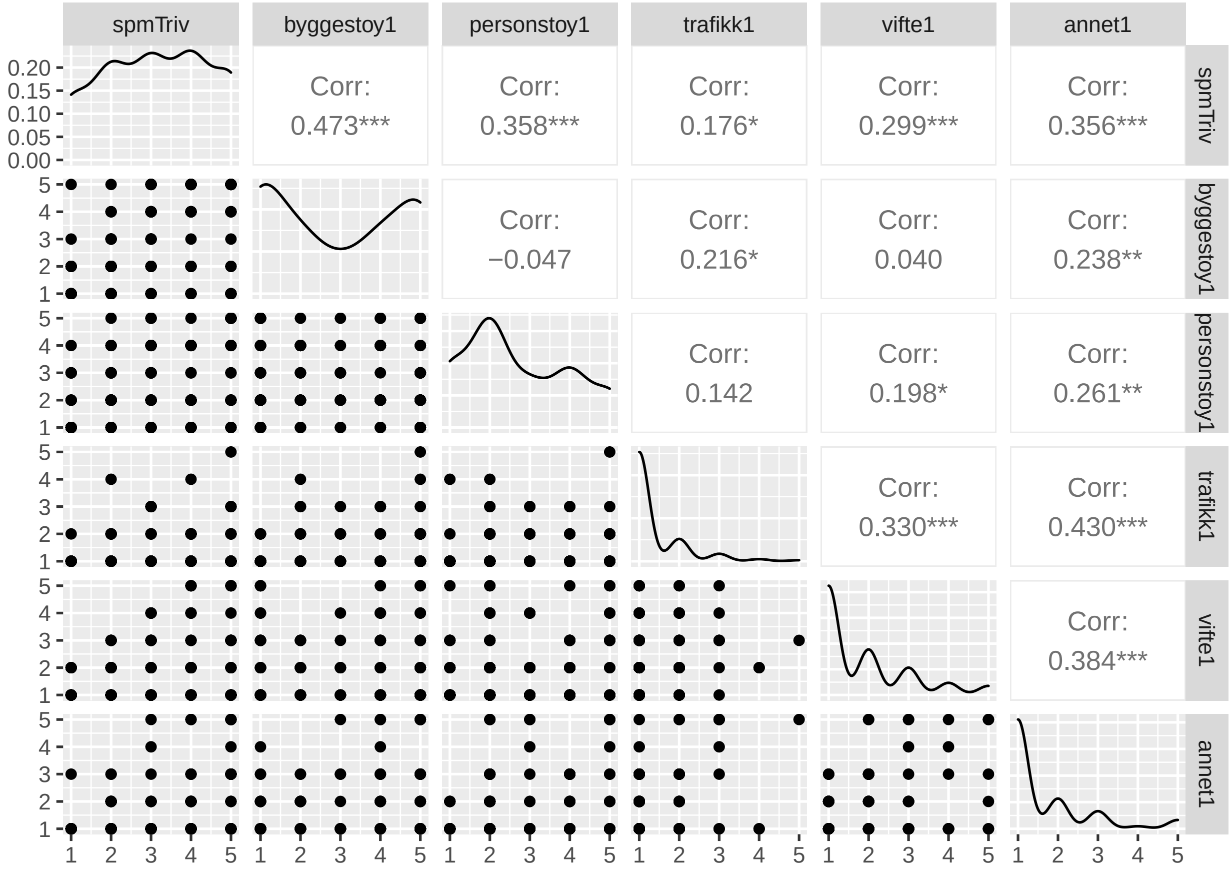
```
plot(yEst)
```



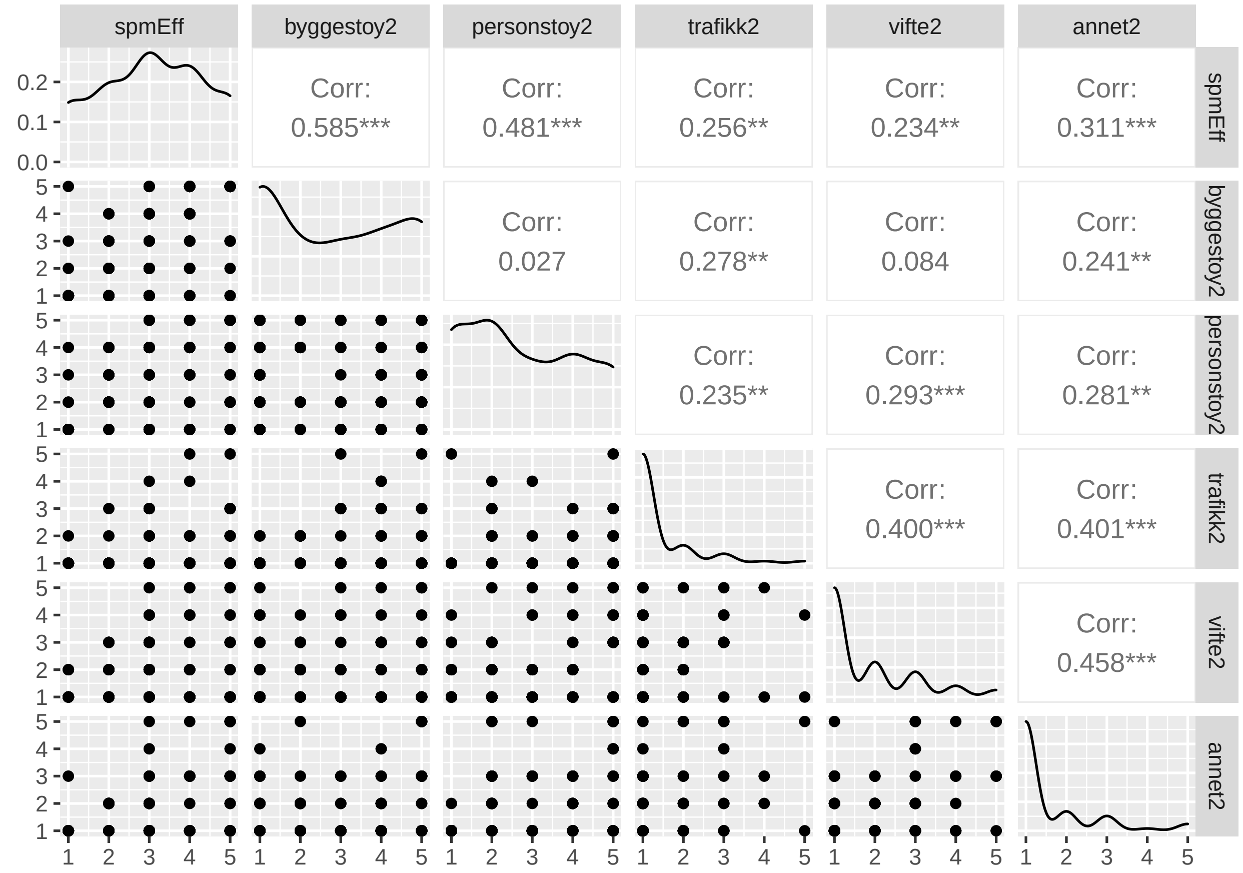
```
pairs(dNumeric[, 2:7])
```



```
library(GGally)
ggpairs(dNumeric[, 2:7])
```



```
ggpairs(dNumeric[, 8:13])
```



Building relations