glmnet

Alina

25 11 2020

```
library(glmnet)
library(clusterSim)
setwd("~/GitHub/fallstudien_2_projekt_1/datasets")
data <- read.csv("dataset_ul.csv", header = TRUE, sep = ",")


#data <- data[c(data$provider == "o2", data$scenario == "highway"),]
```

Data

read the table and select the covariates for the model

```
data <- subset(data, select = c(scenario,
                                provider,
                                velocity_mps,
                                acceleration_mpss,
                                rsrp_dbm,
                                rsrq_db,
                                rssnr_db,
                                cqi,
                                ta,
                                payload_mb,
                                f_mhz,
                                throughput_mbits))
```

now eliminate the rows whith NA's init

```
data <- data[complete.cases(data),]
```

seperate the full dataset in train and test data, hereby 75% of the data get to be the training set and the rest will be the test set

```
#set.seed(101)
sample <- sample.int(n = nrow(data), size = floor(.80*nrow(data)), replace = F)
train <- data[sample, ]
test  <- data[-sample, ]
```

in our case there are two variables that do not contain integers but factors, therefore we have to endcode them, so that the model can handle them here one-hot encoding is used

```
X <- makeX(train, test = test)

train <- X[["x"]]
test <- X[["xtest"]]
```

scale the variables for feature importance, whereby the test set has to be scaled with mean and standard deviation from train set

```
scaled.train <- scale(train)
scaled.test <- scale(test, center=attr(scaled.train, "scaled:center"),
                              scale=attr(scaled.train, "scaled:scale"))
```

Modelfitting

fit the glmnet model with cross validation for the penalty parameter lambda the parameter alpha for the elastic net model has to be set by user

```
fit.cv.scaled <- cv.glmnet(subset(scaled.train, select = -throughput_mbits),
                    subset(scaled.train, select = throughput_mbits),
                    type.measure = "mae", nfolds = 20, alpha = 1)

fit.cv <- cv.glmnet(subset(train, select = -throughput_mbits),
                    subset(train, select = throughput_mbits),
                    type.measure = "mae", nfolds = 20, alpha = 1)
```

Prediction

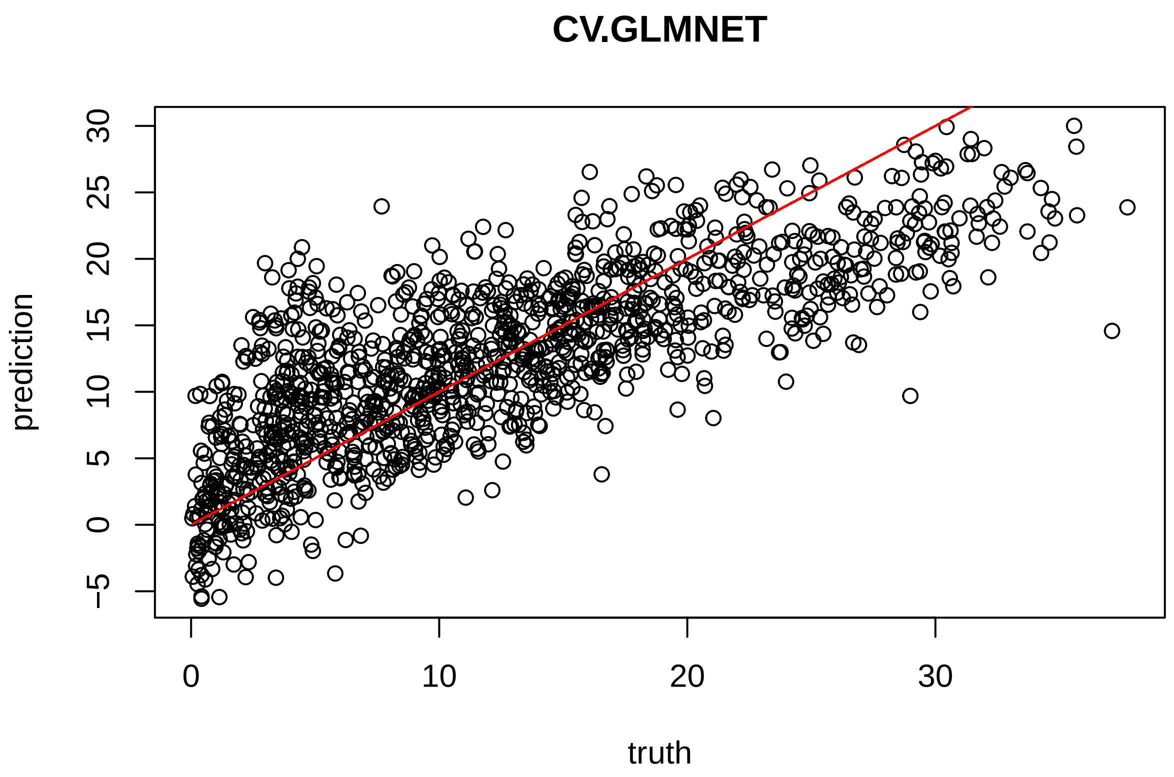
from the fitted cv.glmnet model we now generate the predictions with the covariates from the test set, we hereby use the penalty parameter lambda that generates the lowest error in de cv process

```
pred.cv <- predict(object = fit.cv, newx = subset(test, select = -throughput_mbits),
                   s = "lambda.min", type = "response")
```

Results

plot the predictions from the cv glmnet model against the thruth values from our test set

```
plot(subset(test, select = throughput_mbits), pred.cv, main = "CV.GLMNET",
     xlab = "truth", ylab = "prediction")
lines(subset(test, select = throughput_mbits), subset(test, select = throughput_mbits), col = "red")
```



we want to know the modelrating, therefore we calculate the R-squared, MSE and MAE

```
#calculate R-Squared R^2
yq <- mean(subset(test, select = throughput_mbits))
R2 <- sum((pred.cv-yq)^2)/sum((subset(test, select = throughput_mbits)-yq)^2)
R2
```

```
## [1] 0.6322935
```

```
#calculate MSE
n <- 1/length(pred.cv)
mse <- n*sum((subset(test, select = throughput_mbits)-pred.cv)^2)
mse
```

```
## [1] 28.86704
```

```
#calculate MAE
mae <- n*sum(abs(subset(test, select = throughput_mbits)-pred.cv))
mae
```

```
## [1] 4.204399
```

Feature Importance

compare the absolute coefficients of the model, the larger the value the more information does the corresponding covariate brings

```
coef <- abs(coef(fit.cv.scaled))
coef
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                              1
## (Intercept)       1.603993e-16
## scenariocampus    .
## scenariohighway   .
## scenariosuburban  2.907616e-03
## scenariourban     .
## providertmobile   2.464666e-01
## providervodafone  .
## velocity_mps      .
## acceleration_mpss .
## rsrp_dbm          2.291826e-01
## rsrq_db           1.358380e-01
## rssnr_db          6.863047e-02
## cqi               9.530920e-04
## ta                4.980962e-02
## payload_mb        3.528549e-01
## f_mhz             1.895650e-01
```

```
order(coef)
```

```
## <sparse>[ <logic> ] : .M.sub.i.logical() maybe inefficient
```

```
##  [1]  2  3  5  7  8  9  1 13  4 14 12 11 16 10  6 15
```

single models

run the model with the following data

```
#data <- data[data$provider == "vodafone",] R2 ~ 64
#data <- data[data$provider == "o2",] R2 ~ 54
#data <- data[data$provider == "tmobile",] R2 ~ 53
```