# Homework 1

Pattern Mining and Social Network Analysis

BOUYSSOU Gatien , de POURTALES Caroline, LAMBA Ankit

16 octobre, 2020

## Contents

# Classification

## Overall

Classification algorithms have categorical responses. In classification we build a function f(X) that takes a vector of input variables X and predicts its class membership, such that Y in C.

## Possibilities of models

There are classifiers as logistic regression, Decision tree, Perceptron / Neural networks, K-nearest-neighbors, linear and quadratic logistic regression, Bayes …

## Some indicators

### Sensitivity and recall

The sensitivity (also named recall) is the percentage of true defaulters that are identified (True positive tests). For example, probability of predicting disease given true state is disease.

$$sensitivity = recall = \frac{TruePositiveTests}{PositivePopulation}$$

### Specificity

The specificity is the percentage of non-defaulters that are correctly identified (True negative tests). 1 - specificity is the Type 1 error, it is the false positive rate. For example, probability of predicting non-disease given true state is non- disease.

$$specificity = \frac{TrueNegativeTests}{NegativePopulation}$$

### Precision

The precision is the proportion of true positive tests among the positive tests.

$$precision = \frac{TruePositiveTests}{PositiveTests}$$

### F-Mesure

The traditional F measure is calculated as follows:

$$F_Measure = \frac{(2 * Precision * Recall)}{(Precision + Recall)}$$

### Rand index

The rand index is a mesure of similarity between two partitions from a single set.

Given two partitions $\pi_1$ and $\pi_2$ in E :

- a, the number of elements in $\pi_1$ and $\pi_2$
- b, the number of elements in $\pi_1$ and not in $\pi_2$
- c, the number of elements in $\pi_2$ and not in $\pi_1$
- d, the number of elements not in both $\pi_1$ and $\pi_2$

| | in $\pi_2$ | not in $\pi_2$ |
|---|---|---|
| in $\pi_1$ | a | b |
| not in $\pi_1$ | c | d |

$$RI(\pi_1, \pi_2) = \frac{a + d}{a + b + c + d}$$

## Criterions for best model

How do we determine which model is best? Various statistics can be used to judge the quality of a model. \ These include Mallow's $C_p$, Akaike information criterion (AIC), Bayesian information criterion (BIC), and adjusted $R^2$.

### Mallow's Cp

TO DO

If there are d predictors :

$$C_p = \frac{RSS + 2d\hat{\sigma}^2}{n}$$

### AIC : Akaike information criterion

TO DO

$$AIC = \frac{RSS + 2d\hat{\sigma}^2}{n\hat{\sigma}^2}$$

### BIC : Bayesian information criterion

TO DO

$$BIC = \frac{RSS + log(n)d\hat{\sigma}^2}{n}$$

**Adjusted R statistic**

TO DO

$$AdustedR^2 = 1 - \frac{\frac{RSS}{n-d-1}}{\frac{TSS}{n-1}}$$

## Logistic Regression

### How it works

In logistic regression, for covariates (X_1 , . . . , X_p ), we want to estimate $p_i = P_r(Y_i = 1|X_1, ..., X_p)$

$$p_i = \frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}+...}}{1 + e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}+...}}$$

To come back to linear regression we define the logistic function as follow.

$$logit(p_i) = log(\frac{p_i}{1-p_i}) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + ...$$

We can define the odds :

$$\frac{odds(Y_i = 1|X1 = x_{i1} + 1)}{odds(Y_i = 1|X1 = x_{i1})} = e^{\beta_1}$$

### Which indicator for validity ?

We use Maximum Likehood :

$$L(\beta) = \Pi_{i=1}^{n} p_i^{y_i} * (1 - p_i)^{y_i}$$

The goal is to maximise it by adjusting $\beta$ vector.

### An example in R

We use a dataset from the Wimbledon tennis tournament for Women in 2013. We will predict the result for player 1 (win=1 or loose=0) based on the number of aces won by each player and the number of unforced errors commited by both players. The data set is a subset of a data set from https://archive.ics.uci.edu/ml/datasets/Tennis+Major+Tournament+Match+Statistics.

```
##          Player1      Player2 Result ACE.1 UFE.1 ACE.2 UFE.2
## 1      M.Koehler   V.Azarenka      0     2    18     3    14
## 2     E.Baltacha   F.Pennetta      0     0    10     4    14
## 3      S-W.Hsieh      T.Maria      1     1    13     2    29
## 4       A.Cornet       V.King      1     4    30     0    45
## 5   Y.Putintseva   K.Flipkens      0     2    28     6    19
## 6 A.Tomljanovic B.Jovanovski      0     6    42    11    40
```

```
# reduction to two variables
tennis$ACEdiff = tennis$ACE.1 - tennis$ACE.2
tennis$UFEdiff = tennis$UFE.1 - tennis$UFE.2
head(tennis)
```

```
##         Player1      Player2 Result ACE.1 UFE.1 ACE.2 UFE.2 ACEdiff UFEdiff
## 1     M.Koehler   V.Azarenka      0     2    18     3    14      -1       4
## 2    E.Baltacha   F.Pennetta      0     0    10     4    14      -4      -4
## 3      S-W.Hsieh      T.Maria      1     1    13     2    29      -1     -16
## 4       A.Cornet       V.King      1     4    30     0    45       4     -15
## 5  Y.Putintseva   K.Flipkens      0     2    28     6    19      -4       9
## 6 A.Tomljanovic B.Jovanovski      0     6    42    11    40      -5       2
```

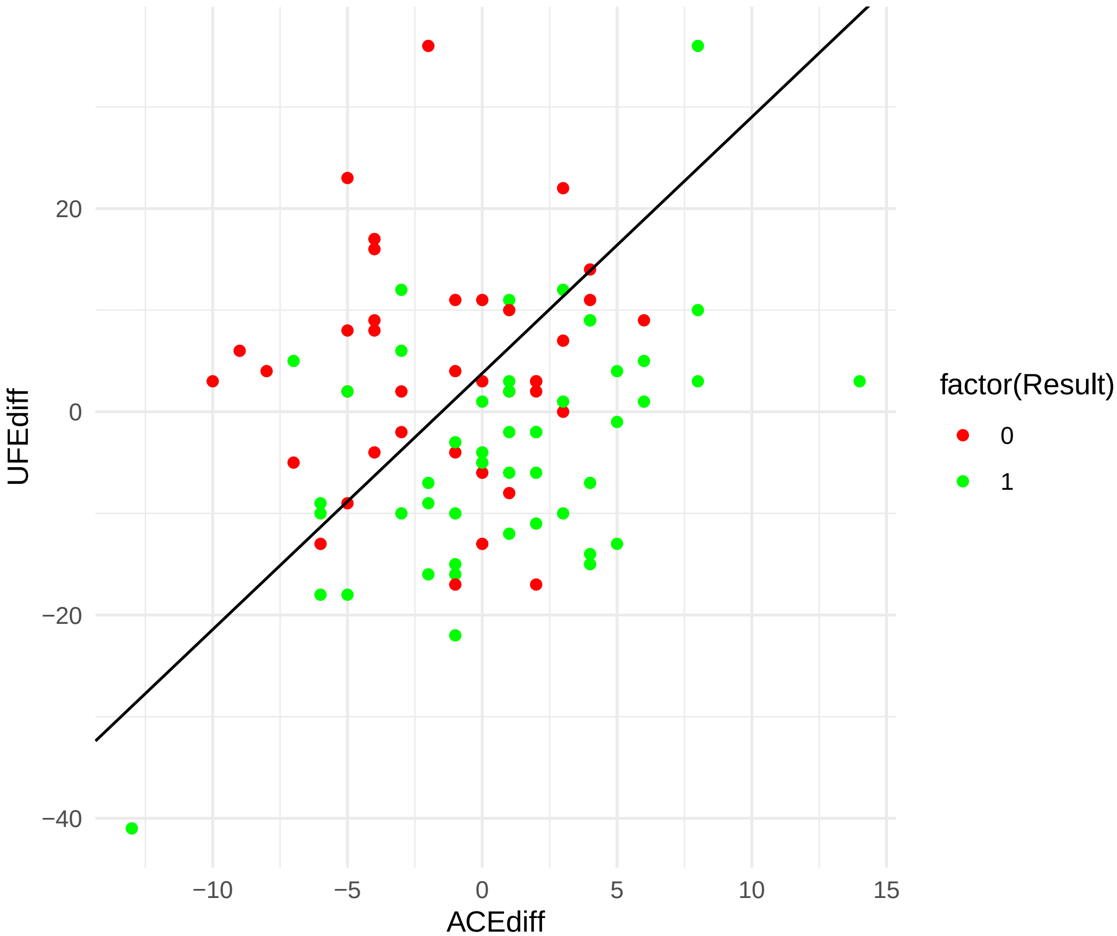
```r
tennisTest = tennis[-train, ]
tennisTrain = tennis[train, ]
r.tennis2 = glm(Result ~ ACEdiff + UFEdiff, data = tennisTrain, family = "binomial")
summary(r.tennis2)
```

```
##
## Call:
## glm(formula = Result ~ ACEdiff + UFEdiff, family = "binomial",
##     data = tennisTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1204  -0.9994   0.5662   0.8918   1.8714
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.31318    0.24439   1.281  0.20004
## ACEdiff      0.20856    0.06575   3.172  0.00151 **
## UFEdiff     -0.08272    0.02454  -3.371  0.00075 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 120.352  on 87  degrees of freedom
## Residual deviance:  99.102  on 85  degrees of freedom
## AIC: 105.1
##
## Number of Fisher Scoring iterations: 4
```

With the model, we can draw the slope which indicates the category of a point.

```r
#We calculate the slope
glm.b = -r.tennis2$coefficients[2]/r.tennis2$coefficients[3]
glm.a = -r.tennis2$coefficients[1]/r.tennis2$coefficients[3]

ggplot() + geom_point(aes(ACEdiff, UFEdiff, color = factor(Result)), data = tennisTrain, ) + scale_colo
  geom_abline(slope = glm.b, intercept = glm.a) +
  theme_minimal()
```




We can write :

$$logit(p_i) = log(\frac{p_i}{1-p_i}) = 0,31318 + 0,20856 * ACEDiff - 0,08272 * UFEDiff$$

We can observe AIC = 105.1

The confusion matrix is :

```
##
## glm.Result_pred  0  1
##               0 15  5
##               1  2  8
```

The accuracy rate is $\frac{17+25}{13+25+4+17} = 0.71$.

The sensitivity is the percentage of true output giving Player1-winner among the population of true Player1-winner :

```
## [1] 0.6153846
```

The specificity is the percentage of true output giving Player2-winner (= Player1-looser) among the population of true Player2-winner:

```
## [1] 0.8823529
```

The precision is the percentage of true output giving Player1-winner among all the outputs giving Player1-winner (even if not winner) :

```
## [1] 0.8
```

So the F_Mesure is :

```
## [1] 0.6956522
```

#### Same example in python with scikit learn

### Decisions trees

#### An example in R : Decision trees and Random Forest

```r
MSE <- rep(NA,25)
deg = 1:25
for (d in deg) {
  modelD <-  randomForest(Result ~ ACE.1 + ACE.2 + UFE.1 + UFE.2, tennisTrain, mtry = 6, ntree = 500, no
  yRandomForest = predict(modelD, newdata = tennisTest)
  MSE[d] = mean((yRandomForest - tennisTest[,3])^2)
}

#The model with the smallest MSE has 14 nodesizes
which.min(MSE)
```

```
## [1] 13
```

```r
#The best model is
modelD <-  randomForest(Result ~ ACE.1 + ACE.2 + UFE.1 + UFE.2, tennisTrain, mtry = 6, ntree = 500, node

#the MSE of random forest
MSE[which.min(MSE)]
```

```
## [1] 0.2102826
```

#### Same example in python with scikit learn

TO DO

## Regression

### Supervised learning

TO DO

### Possibilities of models

TO DO

### The accuracy of a model

TO DO RMSE, MAE, MAPE,R2

Root Mean Squared Error (RMSE), which measures the average prediction error made by the model in predicting the outcome for an observation. That is, the average difference between the observed known outcome values and the values predicted by the model. The lower the RMSE, the better the model.

Mean Absolute Error (MAE), an alternative to the RMSE that is less sensitive to outliers. It corresponds to the average absolute difference between observed and predicted outcomes. The lower the MAE, the better the model

**The Mean Squarred error**

The MSE mesures the mean accuracy of the predicted responses values for given observations. There are two MSE : the train MSE and the test MSE. \ The train MSE is use to fit a model while training. \ The test MSE is use to choose between models already trained. \

Let's define the mean squared error or MSE.

$$MSE = \frac{1}{n}\sum_i (y_i - \hat{f}(x_i))^2$$

Then the expected test MSE refers to the average test MSE that we would obtain if we repeatedly estimated f using a large number of training sets, and tested each at $x_0$. So that the expected test MSE is :

$$E(y_0 - \hat{f}(x_0))^2$$

$$E(y_0 - \hat{f}(x_0))^2 = Var(\hat{f}(x_0)) + (f(x_0) - E(\hat{f}(x_0)))^2 + Var(\varepsilon)$$

$Var(\varepsilon)$ represents the irreductible error. This term can not be reduced regardless how well our statstical model fits the data.

$(f(x_0) - E(\hat{f}(x_0))^2 = [Bias(\hat{f}(x_0))]^2$ is the squared Bias and refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model. If the bias is low the model gives a prediction which is close to the true value.

$Var(\hat{f}(x_0))$ is the Variance of the prediction at $\hat{f}(x_0)$ and refers to the amount by which $\hat{f}$ would change if we estimated it using a different training data set. If the variance is high, there is a large uncertainty associated with the prediction.

**RSS : residual sum of squares**

We define the residual sum of squares (RSS) as :

$$RSS = \Sigma(y_i - \hat{y}_i)^2$$

We want to minimize the RSS.

**RSE : residual standard error**

TO DO

$$RSE = \sqrt{\frac{1}{n-2}RSS}$$

**R statistic**

TO DO

$$R^2 = 1 - \frac{RSS}{TSS}$$

$$TSS = \Sigma(y_i - \bar{y}_i)^2$$

is the total sum of squares. TSS measures the total variance in the response Y.

TSS − RSS measures the amount of variability in the response that is explained.

$R^2$ measures the proportion of variability in Y that can be explained using X.

### F statistic

TO - DO

## Simple Linear Regression

### Definition

TO DO

DEFINITION

WHICH INDICATORS CAN WE USE

Simple linear regression lives up to its name: it is a very straightforward approach for predicting a quantitative response Y on the basis of a single predictor variable X. It assumes that there is approximately a linear relationship between X and Y. Mathematically, we can write this linear relationship as

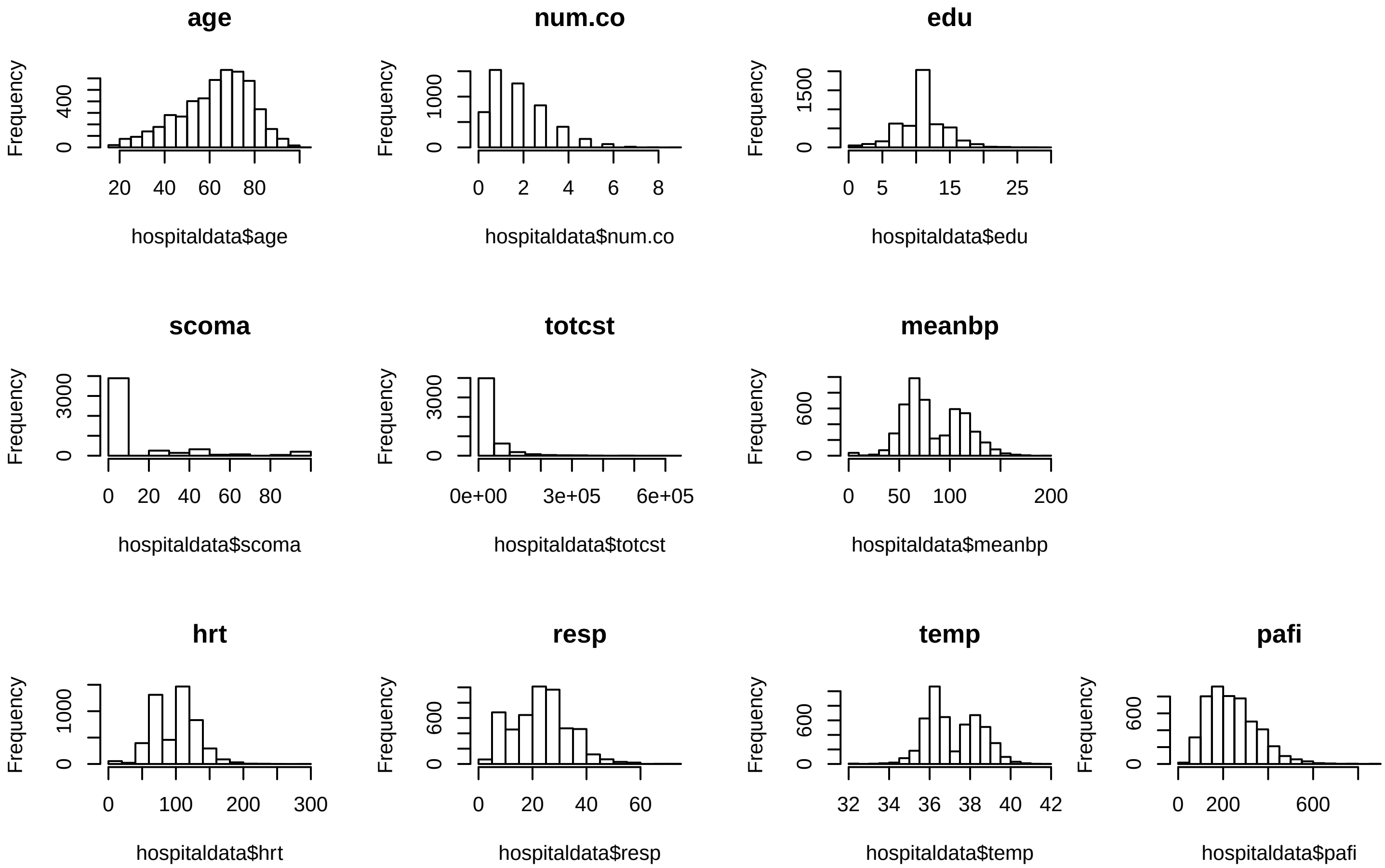
$$Y \approx \beta_0 + \beta_1 * X$$

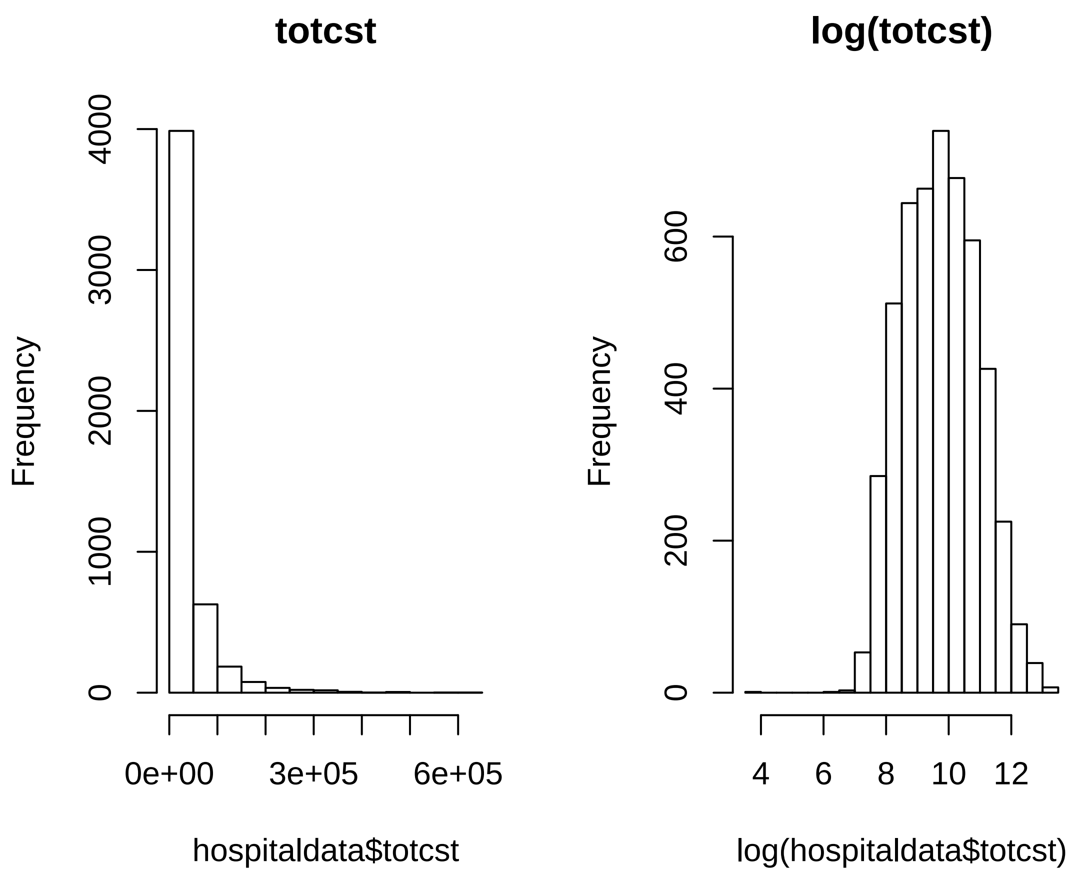
### An example in R

The next dataset (source F. E. Harrell, Regression Modeling Strategies) contains the total hospital costs of 9105 patients with certain diseases in American hospitals between 1989 and 1991. The different variables are :

```
##     age           dzgroup num.co edu     income scoma totcst  race meanbp hrt
## 1 62.85       Lung Cancer      0  11   $11-$25k     0     NA other     97  69
## 2 60.34         Cirrhosis      2  12   $11-$25k    44     NA white     43 112
## 3 52.75         Cirrhosis      2  12 under $11k     0     NA white     70  88
## 4 42.38       Lung Cancer      2  11 under $11k     0     NA white     75  88
## 5 79.88 ARF/MOSF w/Sepsis      1  NA               26     NA white     59 112
## 6 93.02              Coma      1  14               55     NA white    110 101
##   resp  temp   pafi
## 1   22 36.00 388.00
## 2   34 34.59  98.00
## 3   28 37.40 231.66
## 4   32 35.00     NA
## 5   20 37.90 173.31
## 6   44 38.40 266.63
```

We would like to build models that help us to understand which predictors are mostly driving the total cost.


age
Frequency
hospitaldata$age
num.co
hospitaldata$num.co
edu
hospitaldata$edu
scoma
hospitaldata$scoma
totcst
hospitaldata$totcst
meanbp
hospitaldata$meanbp
hrt
hospitaldata$hrt
resp
hospitaldata$resp
temp
hospitaldata$temp
pafi
hospitaldata$pafi




Looking at the distribution of the cost we see we should apply a log transformation for a better distribution. Moreover it seems that only age and disease have an impact.

```r
set.seed(12345)
train.proportion = 0.7
train.ind = sample(1:nrow(hospitaldata), train.proportion* nrow(hospitaldata))
hospitaldata.train = hospitaldata[train.ind, ]
hospitaldata.test = hospitaldata[-train.ind, ]

fit = lm(log(totcst)~ age + temp + edu + resp + num.co + as.factor(dzgroup), data = hospitaldata.train)
summary(fit)
```

```
##
## Call:
## lm(formula = log(totcst) ~ age + temp + edu + resp + num.co +
##     as.factor(dzgroup), data = hospitaldata.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0582 -0.6588 -0.0389  0.6184  3.4372
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       7.848451   0.483978  16.217  < 2e-16 ***
## age                              -0.006611   0.001043  -6.341 2.58e-10 ***
## temp                              0.075107   0.012571   5.975 2.54e-09 ***
## edu                               0.026643   0.004646   5.734 1.06e-08 ***
```

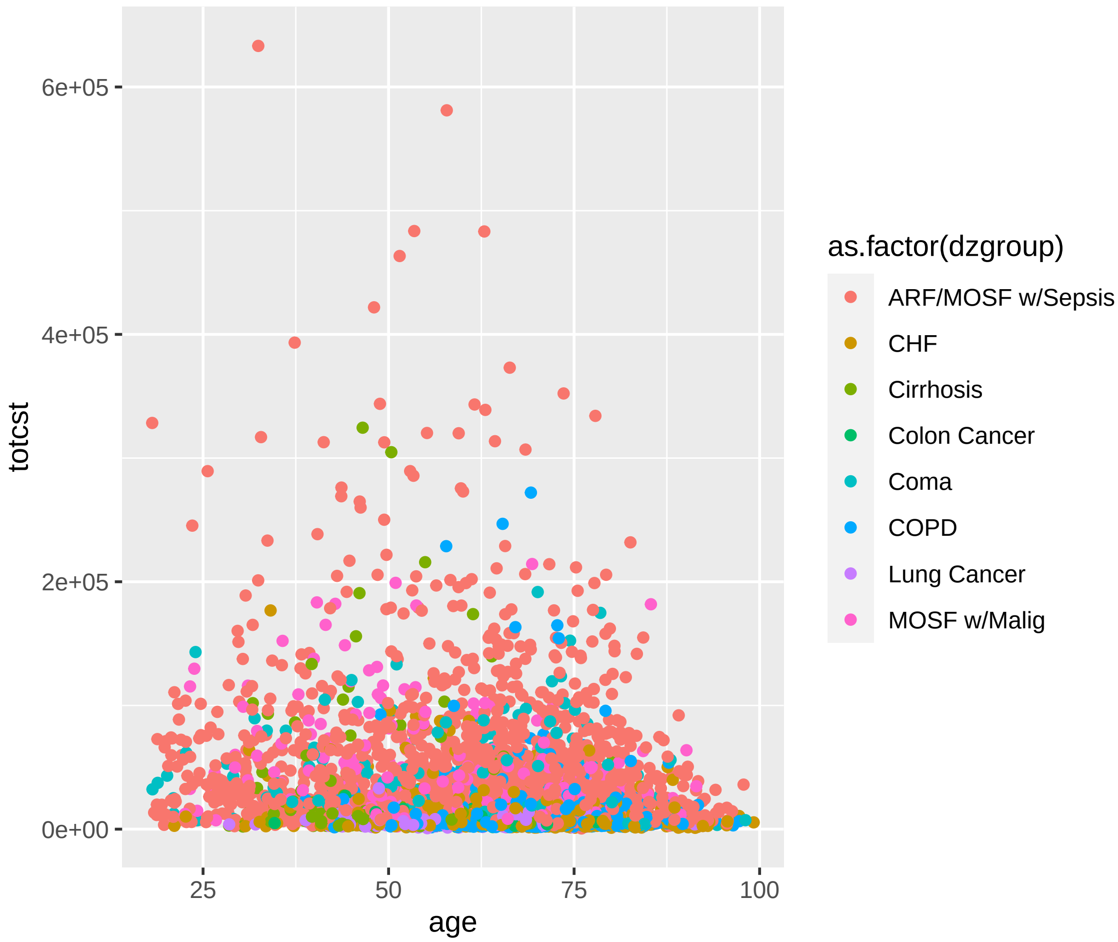
```
## resp                         -0.004025   0.001541  -2.613 0.009026 **
## num.co                       -0.043125   0.012922  -3.337 0.000855 ***
## as.factor(dzgroup)CHF        -1.405058   0.052299 -26.866  < 2e-16 ***
## as.factor(dzgroup)Cirrhosis  -0.923605   0.077611 -11.900  < 2e-16 ***
## as.factor(dzgroup)Colon Cancer -1.458633  0.100672 -14.489  < 2e-16 ***
## as.factor(dzgroup)Coma       -0.452564   0.067303  -6.724 2.06e-11 ***
## as.factor(dzgroup)COPD       -1.242045   0.052407 -23.700  < 2e-16 ***
## as.factor(dzgroup)Lung Cancer  -1.688832   0.064437 -26.209  < 2e-16 ***
## as.factor(dzgroup)MOSF w/Malig -0.256303   0.060087  -4.266 2.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9297 on 3459 degrees of freedom
## Multiple R-squared:  0.3846, Adjusted R-squared:  0.3825
## F-statistic: 180.2 on 12 and 3459 DF,  p-value: < 2.2e-16
```

We can that just age and dzgroup seem to have an impact on totcst.

```
fit = lm(log(totcst)~ age + as.factor(dzgroup) , data = hospitaldata.train)
summary(fit)
```

```
##
## Call:
## lm(formula = log(totcst) ~ age + as.factor(dzgroup), data = hospitaldata.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9555 -0.6766 -0.0325  0.6116  3.5094
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    10.885440   0.068097 159.853  < 2e-16 ***
## age                            -0.007856   0.001043  -7.529 6.49e-14 ***
## as.factor(dzgroup)CHF          -1.527689   0.048791 -31.311  < 2e-16 ***
## as.factor(dzgroup)Cirrhosis    -0.998127   0.076748 -13.005  < 2e-16 ***
## as.factor(dzgroup)Colon Cancer -1.423058   0.101698 -13.993  < 2e-16 ***
## as.factor(dzgroup)Coma         -0.425403   0.067871  -6.268 4.11e-10 ***
## as.factor(dzgroup)COPD         -1.337596   0.051472 -25.987  < 2e-16 ***
## as.factor(dzgroup)Lung Cancer  -1.714271   0.065095 -26.335  < 2e-16 ***
## as.factor(dzgroup)MOSF w/Malig -0.241001   0.060559  -3.980 7.04e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9413 on 3463 degrees of freedom
## Multiple R-squared:  0.3685, Adjusted R-squared:  0.367
## F-statistic: 252.6 on 8 and 3463 DF,  p-value: < 2.2e-16
```

```
ggplot() + geom_point(aes(age, totcst, color = as.factor(dzgroup) ), data = hospitaldata.train)
```




We can write :

$$log(totcost) = 8.0823597 - 0.0069950 * age + x_{ij} * \beta_j$$

where $x_{ij}$ is 1 if patient i has disease j and $\beta_j$ is the coefficient matchinf the disease in the previous tab.

We can calculate the MSE on the test set to evaluate the simple linear regression model.

```
predictions <- fit %>% predict(hospitaldata.test)
mse = mean((predictions - log(hospitaldata.test$totcst))^2)
mse
```

```
## [1] 0.9017872
```

#### Same example in python with scikit learn

## Multiple linear regression

### Definition

TO DO

DEFINITION

WHICH INDICATORS ?

### An example in R

We use the same example than for simple linear regression.

```
fit_multiple = lm(log(totcst)~age*as.factor(dzgroup), data = hospitaldata.train)
summary(fit_multiple)
```

```
##
## Call:
## lm(formula = log(totcst) ~ age * as.factor(dzgroup), data = hospitaldata.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9919 -0.6711 -0.0403  0.6162  3.5138
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         10.737765   0.087214 123.120  < 2e-16 ***
## age                                 -0.005434   0.001374  -3.955 7.82e-05 ***
## as.factor(dzgroup)CHF               -1.223927   0.222589  -5.499 4.11e-08 ***
## as.factor(dzgroup)Cirrhosis         -0.581175   0.309688  -1.877  0.06065 .
## as.factor(dzgroup)Colon Cancer      -1.264904   0.649568  -1.947  0.05158 .
## as.factor(dzgroup)Coma               0.368400   0.265318   1.389  0.16507
## as.factor(dzgroup)COPD              -1.465964   0.309251  -4.740 2.22e-06 ***
## as.factor(dzgroup)Lung Cancer       -2.050794   0.358454  -5.721 1.15e-08 ***
## as.factor(dzgroup)MOSF w/Malig       0.401009   0.240319   1.669  0.09528 .
## age:as.factor(dzgroup)CHF           -0.004751   0.003292  -1.443  0.14906
## age:as.factor(dzgroup)Cirrhosis     -0.007435   0.005539  -1.342  0.17959
## age:as.factor(dzgroup)Colon Cancer -0.002584   0.009925  -0.260  0.79461
## age:as.factor(dzgroup)Coma          -0.012590   0.004055  -3.104  0.00192 **
## age:as.factor(dzgroup)COPD           0.001504   0.004393   0.342  0.73212
## age:as.factor(dzgroup)Lung Cancer    0.005387   0.005691   0.947  0.34389
## age:as.factor(dzgroup)MOSF w/Malig -0.010661   0.003868  -2.756  0.00589 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9396 on 3456 degrees of freedom
## Multiple R-squared:  0.372,  Adjusted R-squared:  0.3693
## F-statistic: 136.5 on 15 and 3456 DF,  p-value: < 2.2e-16
```

We can calculate the MSE on the test set to evaluate the multiple linear regression model.

```
predictions <- fit_multiple %>% predict(hospitaldata.test)
mse = mean((predictions - log(hospitaldata.test$totcst))^2)
mse
```

```
## [1] 0.8979065
```

The MSE-test for multiple linear regression is worst than for simple linear regression.

Simple linear regression is the best model so far for this problem.

#### Same example in python with scikit learn

# Comparaison between R and sckit-learn in python

## On classification

### Logistic Regression

TO DO : comparaison between R and python

| | R | Scikit-learn |
|---|---|---|
| sensitivity | | |
| specificity | | |
| precision | | |
| f mesure | | |
| AIC | | |

### Decision trees

TO DO : comparaison between R and python

either knn, or decsion trees, or linear discriminant analysis or quadratic discriminant analysis

| | R | Scikit-learn |
|---|---|---|
| sensitivity | | |
| specificity | | |
| precision | | |
| f mesure | | |
| AIC | | |

## On Regression

### Simple Linear Regression

| | R | Scikit-learn |
|---|---|---|
| MSE | | |

### Multiple Linear Regression

| | R | Scikit-learn |
|---|---|---|
| MSE | | |

# Validation techniques

## Sampling

This consists in dividing the dataset into a training set and a test set.

## Cross validation

R2, RMSE and MAE are used to measure the regression model performance during cross-validation.

### Validation set approach

TO DO

```r
# Split the data into training and test set
set.seed(123)
training.samples <- log(hospitaldata$totcst) %>% createDataPartition(p = 0.8, list = FALSE)
hospitaldata.train2  <- hospitaldata[training.samples, ]
hospitaldata.test2 <- hospitaldata[-training.samples, ]
# Build the model
model <- lm(log(totcst) ~ age + as.factor(dzgroup), data = hospitaldata.train2)
print(model)
```

```
##
## Call:
## lm(formula = log(totcst) ~ age + as.factor(dzgroup), data = hospitaldata.train2)
##
## Coefficients:
##                    (Intercept)                                age
##                      10.912390                          -0.008337
##          as.factor(dzgroup)CHF     as.factor(dzgroup)Cirrhosis
##                      -1.530006                          -0.968137
## as.factor(dzgroup)Colon Cancer          as.factor(dzgroup)Coma
##                      -1.492058                          -0.397401
##         as.factor(dzgroup)COPD   as.factor(dzgroup)Lung Cancer
##                      -1.340868                          -1.735200
## as.factor(dzgroup)MOSF w/Malig
##                      -0.272309
```

```r
# Make predictions and compute the R2, RMSE and MAE
predictions <- model %>% predict(hospitaldata.test2)
data.frame( R2 = R2(predictions, log(hospitaldata.test2$totcst)),
            RMSE = RMSE(predictions, log(hospitaldata.test2$totcst)),
            MAE = MAE(predictions, log(hospitaldata.test2$totcst)))
```

```
##          R2      RMSE       MAE
## 1 0.3497368 0.9569546 0.7521536
```

#### Leave One out cross-validation

TO DO

This method works as follow:

Leave out one data point and build the model on the rest of the data set Test the model against the data point that is left out at step 1 and record the test error associated with the prediction Repeat the process for all data points Compute the overall prediction error by taking the average of all these test error estimates recorded at step 2.

```r
# Define training control
train.control <- trainControl(method = "LOOCV")
# Train the model
model <- train(log(totcst) ~ age + as.factor(dzgroup), data = hospitaldata, method = "lm",  trControl =
# Summarize the results
print(model)
```

```
## Linear Regression
##
## 4960 samples
##    2 predictor
##
```

```
## No pre-processing
## Resampling: Leave-One-Out Cross-Validation
## Summary of sample sizes: 4959, 4959, 4959, 4959, 4959, 4959, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   0.944522  0.3656192  0.7552076
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

#### k-Fold Cross-Validation

TO DO

We divide the set of data in k equals part and we use k-1 parts to train the model and 1 to test. We do do that k times in order to use each part as a test part.

Here are the steps :

1.Split the dataset into k equal partitions (or "folds")

2.For each fold

One fold is used as the testing set and the union of the other folds as the training set

Calculate testing accuracy for this fold :

$$\hat{f}_i = \frac{1}{K} \sum_{j \in N_0} (y_j)$$

$$MSE = \frac{k}{n} \sum_i I(y_i \neq \hat{y}_i)$$

3.Use the average testing accuracy as the estimate of out-of-sample accuracy :

We would use the cross-validation error :

$$CV_k = \frac{1}{k} \sum_i MSE_i$$

with $I(y_i \neq \hat{y}_i) = 1$ if $y_i \neq \hat{y}_i$, 0 else. So that we calculate the average of wrong predicted values.

```r
# Define training control
train.control <- trainControl(method = "cv", number = 10)
# Train the model
model <- train(log(totcst) ~ age + as.factor(dzgroup), data = hospitaldata, method = "lm",  trControl =
# Summarize the results
print(model)
```

```
## Linear Regression
##
## 4960 samples
##    2 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 4464, 4464, 4464, 4464, 4464, 4464, ...
## Resampling results:
##
```

```
##   RMSE       Rsquared   MAE
##   0.9442358  0.3678628  0.7551859
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```