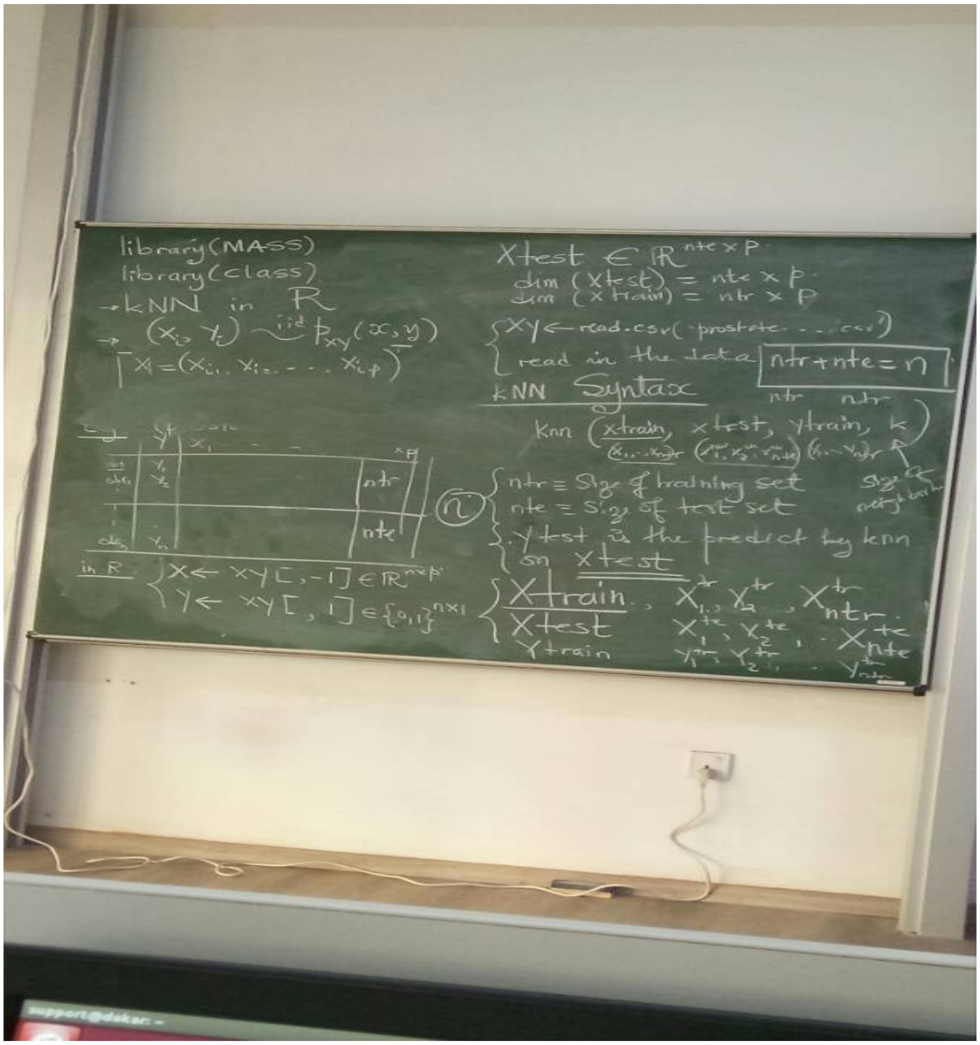
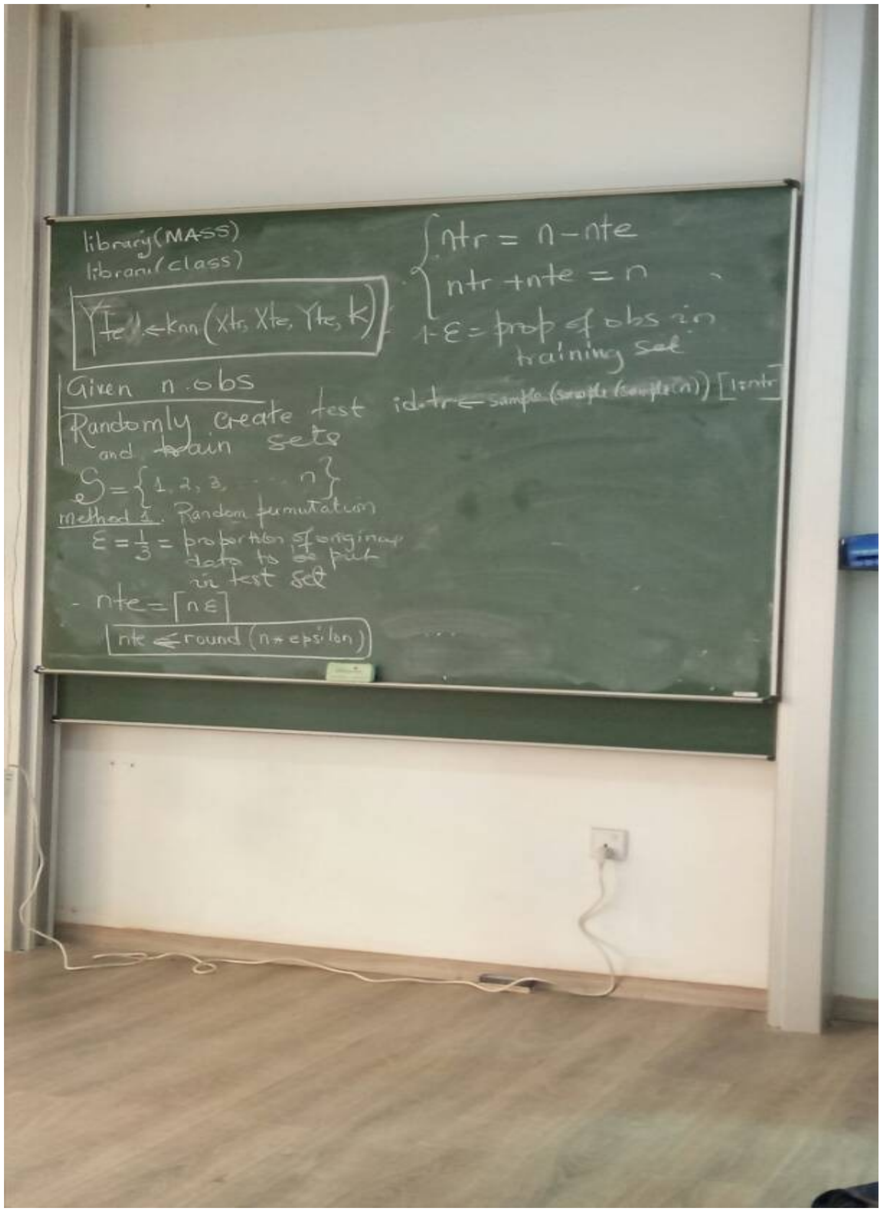
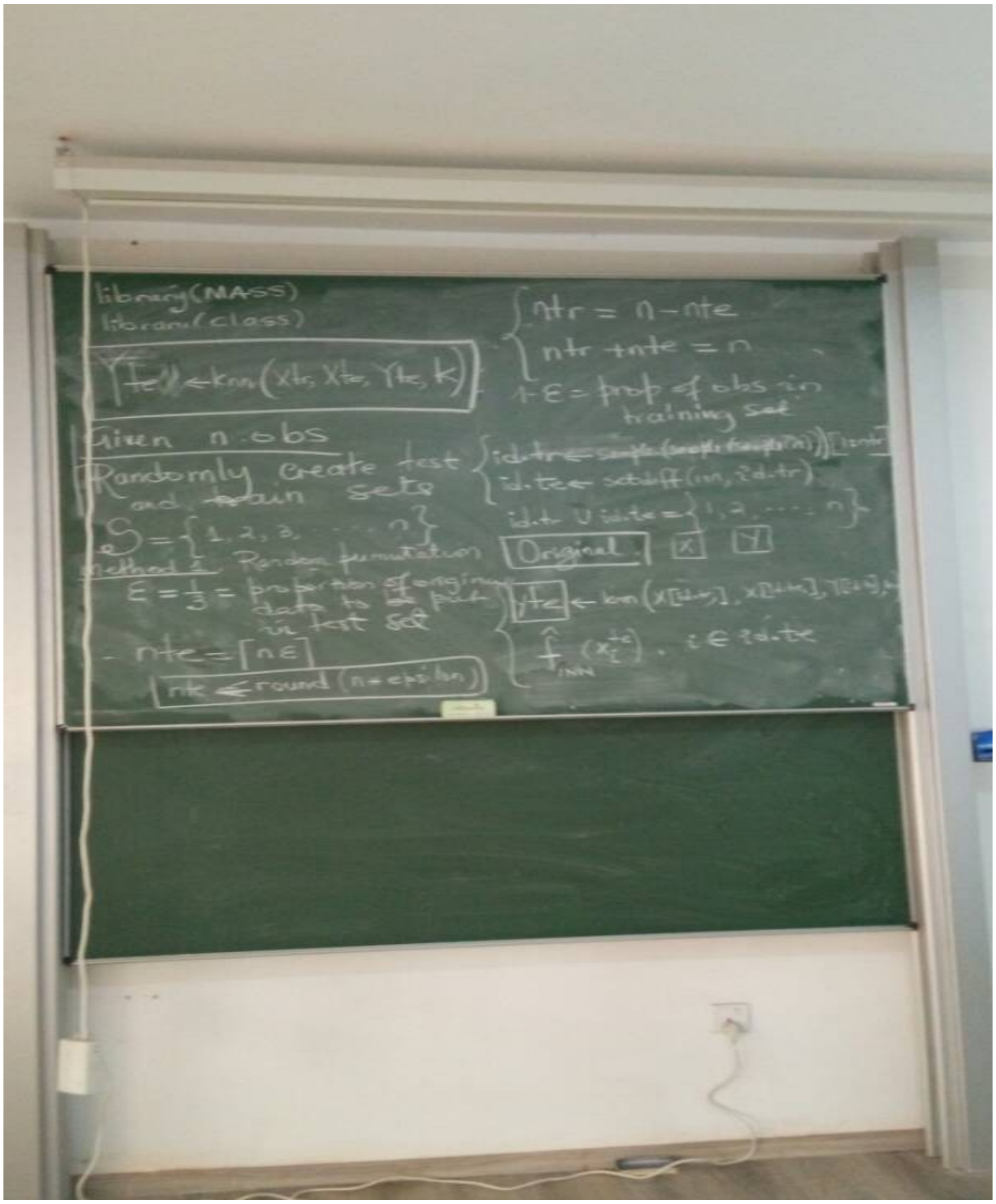
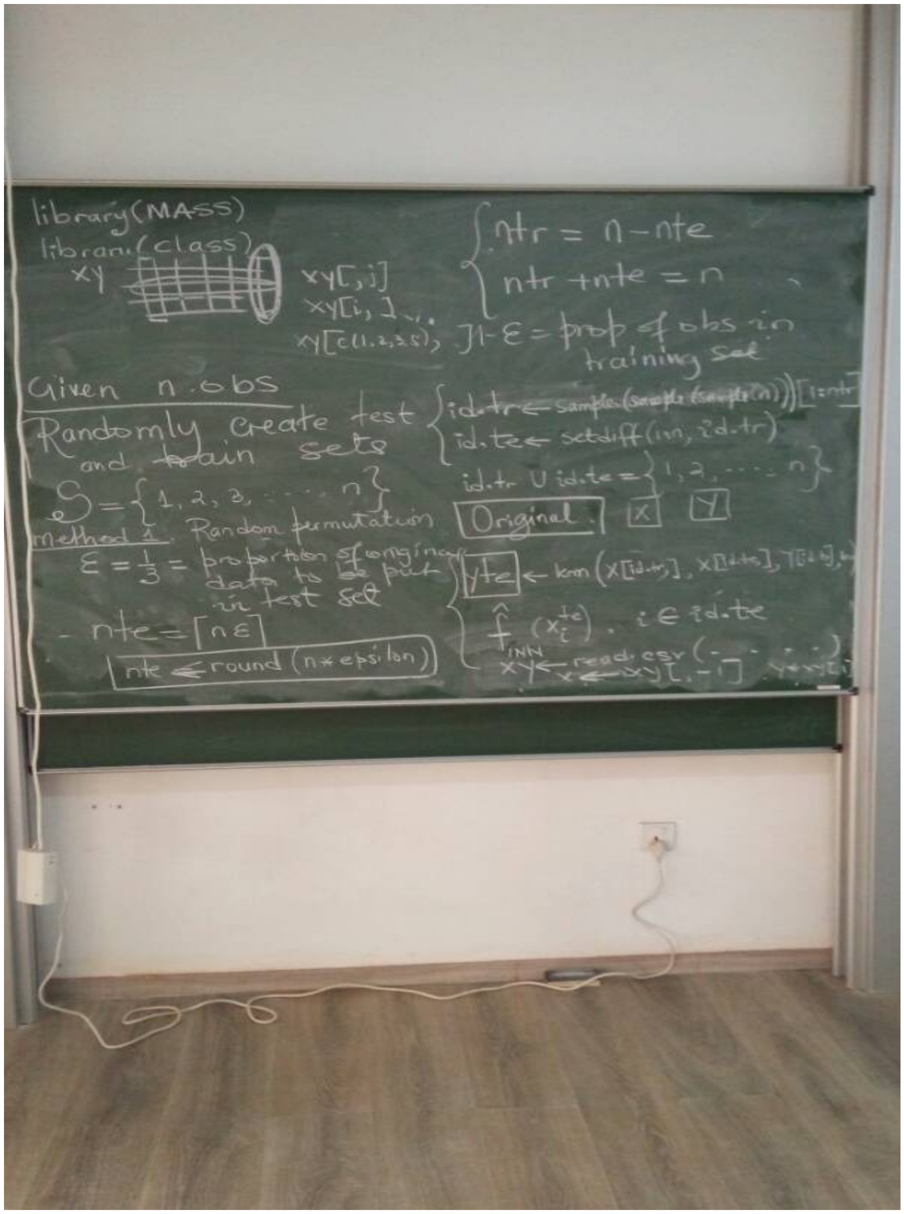
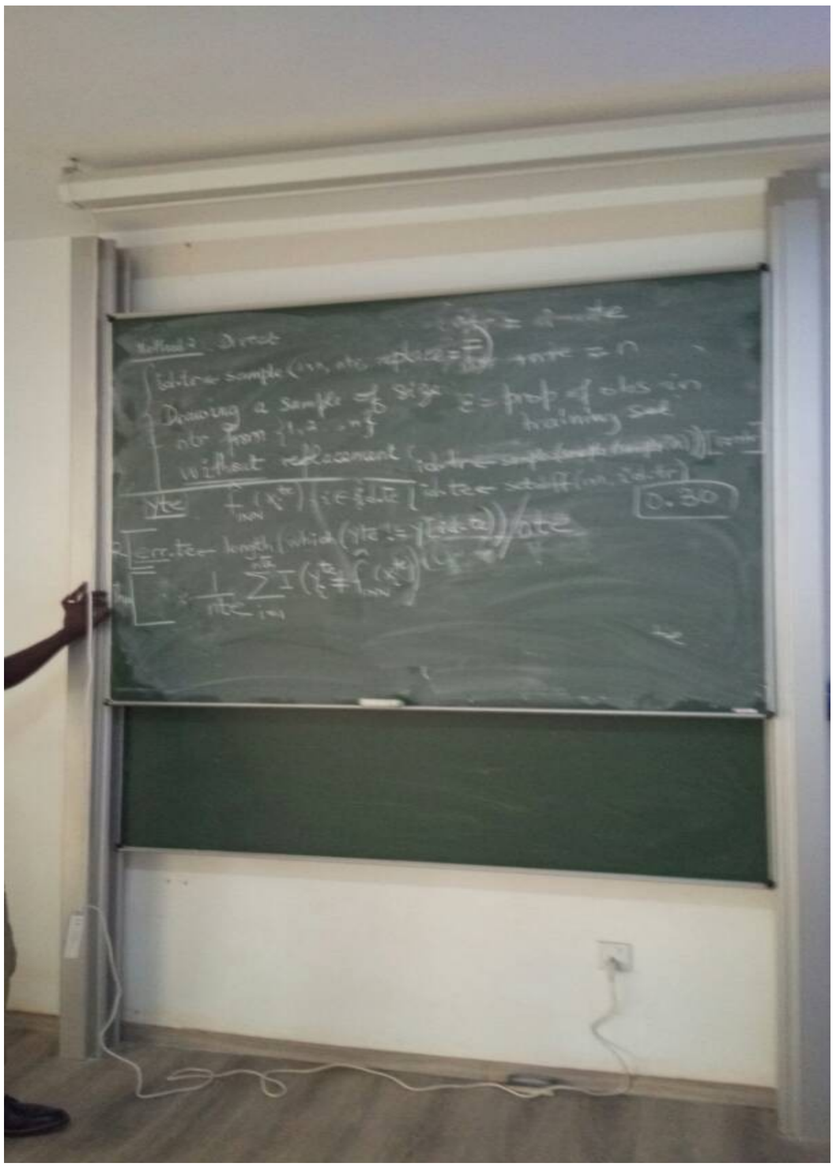

library(MASS)
library(class)
kNN in R
(Xi, Yi) ~iid pXY(x, y)
Xi = (Xi1, Xi2, ..., Xip)
in R
X ← xy[, -1] ∈ R^(n×p)
Y ← xy[, 1] ∈ {0,1}^(n×1)
Xtest ∈ R^(nte×p)
dim (Xtest) = nte × p
dim (Xtrain) = ntr × p
xy ← read.csv("prostate.....csv")
read in the data
ntr + nte = n
kNN Syntax
knn (xtrain, xtest, ytrain, k)
ntr = Size of training set
nte = Size of test set
ytest is the predict by knn
on Xtest
Xtrain
Xtest
Ytrain


library(MASS)
library(class)
Yte <- knn(Xtr, Xte, Yte, k)
ntr = n - nte
ntr + nte = n
1-ε = prop of obs in training set
Given n obs
Randomly create test and train sets
idxtr <- sample(sample(sample(n)))[1:ntr]
S = {1, 2, 3, ... n}
method 1. Random permutation
ε = 1/3 = proportion of original data to be put in test set
nte = ⌈nε⌉
nte <- round(n*epsilon)


library(MASS)
library(class)
Given n obs
Randomly create test and train sets
Method 1: Random permutation
nte ← round(n*epsilon)
ntr = n − nte
ntr + nte = n
1 − ε = prop of obs in training set
id.te ← setdiff(1:n, id.tr)
Original
X
Y
i ∈ id.te



library(MASS)
library(class)
xy
xy[,j]
xy[i,]
xy[c(1,2,3,5),]
ntr = n-nte
ntr + nte = n
1-ε = prop of obs in training set
Given n.obs
Randomly create test and train sets
id.tr ← sample(sample(sample(n)))[1:ntr]
id.te ← setdiff(1:n, id.tr)
id.tr U id.te = {1, 2, ..., n}
S = {1, 2, 3, ... n}
Method 1. Random permutation
Original
X
Y
ε = 1/3 = proportion of original data to be put in test set
nte = ⌈nε⌉
nte ← round(n*epsilon)
yte ← knn(X[id.tr,], X[id.te,], Y[id.tr],
i ∈ id.te
xy ← read.csv(...)
x ← xy[,-1]


Drawing a sample of size
from {1,2 ..n}
without replacement
prop of obs in
training set
0.30

ROC
Prediction
Truth
TP
FN
FP
TN
= TPR
TP+FN
= True Positive Rate
Storing the confusion matrix
perfect binary classifier
FPR
Receiver Operating Characteristic (ROC)
binary classification
AUC
Area Under the Curve
Joel beats Fokoué