R challenge B

github link: https://github.com/WardPlessers/R-programming-Challenge-2

Task 1B - Predicting house prices in Ames, Iowa

Step 1

For this task we're going to choose the ML technique: random forest. This technique works with decision trees. Normal decision trees work like this: you have one big tree, and you do an iteration in order to make the best dececion tree possible. Your variables are the amount of branches the tree has. In the first iteration the tree will look at all the variables, and decide wich variable it should use to predict the target variable as well as possible. It will take the best variable and this will be the first branch. For the next iteration it will do the same with the remaining variables, untill all variables are in the tree. After this you will have one decision tree to predict target variables. Now we are going to use a random forest, this means we're also gonna work with trees, but in a quite different way. We will have more than one tree. We choose the amount of trees at the start of our process. Every tree we have, will just take a number of random variables and decide wich one of these is the best, using a sample of the training data (just like the old decision tree, only now the tree choses from a smaller number of variables). The tree will do this a couple of times untill it has enough branches (chosen by the user) So when a testset goes trough the random forest, all the trees will decide on one possible value for the target value. The average of this value will be the answer of the forest.

Step 2

```
data<-read.csv("train.csv",header=TRUE) #we read the data into R
#we want the data without the forst column of identities
training<-data[,c(2:81)]
#we are going to use the econometrics technique: random forest
#we are going to eliminate the NA values, we are going to start with eliminating the variables with
#a very high number of NA values
summary(training) #we are going to eliminate variables with a lot of NA values, lets take 150 or
```

```
##    MSSubClass       MSZoning     LotFrontage        LotArea
##  Min.   : 20.0   C (all):  10   Min.   : 21.00   Min.   :  1300
##  1st Qu.: 20.0   FV     :  65   1st Qu.: 59.00   1st Qu.:  7554
##  Median : 50.0   RH     :  16   Median : 69.00   Median :  9478
##  Mean   : 56.9   RL     :1151   Mean   : 70.05   Mean   : 10517
##  3rd Qu.: 70.0   RM     : 218   3rd Qu.: 80.00   3rd Qu.: 11602
##  Max.   :190.0                   Max.   :313.00   Max.   :215245
##                                  NA's   :259
##   Street      Alley      LotShape  LandContour  Utilities
##  Grvl:   6   Grvl:  50   IR1:484   Bnk:  63    AllPub:1459
##  Pave:1454   Pave:  41   IR2: 41   HLS:  50    NoSeWa:   1
##              NA's:1369   IR3: 10   Low:  36
##                          Reg:925   Lvl:1311
##
##
##
##    LotConfig    LandSlope   Neighborhood   Condition1     Condition2
##  Corner : 263   Gtl:1382   NAmes  :225   Norm   :1260   Norm   :1445
##  CulDSac:  94   Mod:  65   CollgCr:150   Feedr  :  81   Feedr  :   6
##  FR2    :  47   Sev:  13   OldTown:113   Artery :  48   Artery :   2
##  FR3    :   4              Edwards:100   RRAn   :  26   PosN   :   2
##  Inside :1052              Somerst: 86   PosN   :  19   RRNn   :   2
##                            Gilbert: 79   RRAe   :  11   PosA   :   1
```

```
##                           (Other):707   (Other):  15   (Other):   2
##    BldgType      HouseStyle   OverallQual      OverallCond
##  1Fam  :1220   1Story :726   Min.   : 1.000   Min.   :1.000
##  2fmCon:  31   2Story :445   1st Qu.: 5.000   1st Qu.:5.000
##  Duplex:  52   1.5Fin :154   Median : 6.000   Median :5.000
##  Twnhs :  43   SLvl   : 65   Mean   : 6.099   Mean   :5.575
##  TwnhsE: 114   SFoyer : 37   3rd Qu.: 7.000   3rd Qu.:6.000
##                1.5Unf : 14   Max.   :10.000   Max.   :9.000
##                (Other): 19
##    YearBuilt     YearRemodAdd    RoofStyle       RoofMatl     Exterior1st
##  Min.   :1872   Min.   :1950   Flat   :  13   CompShg:1434   VinylSd:515
##  1st Qu.:1954   1st Qu.:1967   Gable  :1141   Tar&Grv:  11   HdBoard:222
##  Median :1973   Median :1994   Gambrel:  11   WdShngl:   6   MetalSd:220
##  Mean   :1971   Mean   :1985   Hip    : 286   WdShake:   5   Wd Sdng:206
##  3rd Qu.:2000   3rd Qu.:2004   Mansard:   7   ClyTile:   1   Plywood:108
##  Max.   :2010   Max.   :2010   Shed   :   2   Membran:   1   CemntBd: 61
##                                               (Other):   2   (Other):128
##   Exterior2nd    MasVnrType    MasVnrArea     ExterQual ExterCond
##  VinylSd:504   BrkCmn : 15   Min.   :   0.0   Ex: 52    Ex:   3
##  MetalSd:214   BrkFace:445   1st Qu.:   0.0   Fa: 14    Fa:  28
##  HdBoard:207   None   :864   Median :   0.0   Gd:488    Gd: 146
##  Wd Sdng:197   Stone  :128   Mean   : 103.7   TA:906    Po:   1
##  Plywood:142   NA's   :  8   3rd Qu.: 166.0             TA:1282
##  CmentBd: 60                 Max.   :1600.0
##  (Other):136                 NA's   :8
##   Foundation  BsmtQual   BsmtCond    BsmtExposure BsmtFinType1
##  BrkTil:146   Ex  :121   Fa  :  45   Av  :221     ALQ :220
##  CBlock:634   Fa  : 35   Gd  :  65   Gd  :134     BLQ :148
##  PConc :647   Gd  :618   Po  :   2   Mn  :114     GLQ :418
##  Slab  : 24   TA  :649   TA  :1311   No  :953     LwQ : 74
##  Stone :  6   NA's: 37   NA's:  37   NA's: 38     Rec :133
##  Wood  :  3                                      Unf :430
##                                                  NA's: 37
##    BsmtFinSF1     BsmtFinType2   BsmtFinSF2        BsmtUnfSF
##  Min.   :   0.0   ALQ :  19    Min.   :   0.00   Min.   :   0.0
##  1st Qu.:   0.0   BLQ :  33    1st Qu.:   0.00   1st Qu.: 223.0
##  Median : 383.5   GLQ :  14    Median :   0.00   Median : 477.5
##  Mean   : 443.6   LwQ :  46    Mean   :  46.55   Mean   : 567.2
##  3rd Qu.: 712.2   Rec :  54    3rd Qu.:   0.00   3rd Qu.: 808.0
##  Max.   :5644.0   Unf :1256    Max.   :1474.00   Max.   :2336.0
##                   NA's:  38
##   TotalBsmtSF      Heating     HeatingQC CentralAir Electrical
##  Min.   :   0.0   Floor:   1   Ex:741    N:  95     FuseA:  94
##  1st Qu.: 795.8   GasA :1428   Fa: 49    Y:1365     FuseF:  27
##  Median : 991.5   GasW :  18   Gd:241               FuseP:   3
##  Mean   :1057.4   Grav :   7   Po:  1               Mix  :   1
##  3rd Qu.:1298.2   OthW :   2   TA:428               SBrkr:1334
##  Max.   :6110.0   Wall :   4                        NA's :   1
##
##    X1stFlrSF      X2ndFlrSF     LowQualFinSF       GrLivArea
##  Min.   : 334   Min.   :   0   Min.   :  0.000   Min.   : 334
##  1st Qu.: 882   1st Qu.:   0   1st Qu.:  0.000   1st Qu.:1130
##  Median :1087   Median :   0   Median :  0.000   Median :1464
##  Mean   :1163   Mean   : 347   Mean   :  5.845   Mean   :1515
```

```
##  3rd Qu.:1391   3rd Qu.: 728   3rd Qu.:  0.000   3rd Qu.:1777
##  Max.   :4692   Max.   :2065   Max.   :572.000   Max.   :5642
##
##   BsmtFullBath     BsmtHalfBath        FullBath        HalfBath
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000
##  Median :0.0000   Median :0.00000   Median :2.000   Median :0.0000
##  Mean   :0.4253   Mean   :0.05753   Mean   :1.565   Mean   :0.3829
##  3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :3.0000   Max.   :2.00000   Max.   :3.000   Max.   :2.0000
##
##   BedroomAbvGr    KitchenAbvGr   KitchenQual  TotRmsAbvGrd    Functional
##  Min.   :0.000   Min.   :0.000   Ex:100       Min.   : 2.000   Maj1:  14
##  1st Qu.:2.000   1st Qu.:1.000   Fa: 39       1st Qu.: 5.000   Maj2:   5
##  Median :3.000   Median :1.000   Gd:586       Median : 6.000   Min1:  31
##  Mean   :2.866   Mean   :1.047   TA:735       Mean   : 6.518   Min2:  34
##  3rd Qu.:3.000   3rd Qu.:1.000                3rd Qu.: 7.000   Mod :  15
##  Max.   :8.000   Max.   :3.000                Max.   :14.000   Sev :   1
##                                                                Typ :1360
##    Fireplaces    FireplaceQu   GarageType   GarageYrBlt   GarageFinish
##  Min.   :0.000   Ex  : 24    2Types :  6   Min.   :1900   Fin :352
##  1st Qu.:0.000   Fa  : 33    Attchd :870   1st Qu.:1961   RFn :422
##  Median :1.000   Gd  :380    Basment: 19   Median :1980   Unf :605
##  Mean   :0.613   Po  : 20    BuiltIn: 88   Mean   :1979   NA's: 81
##  3rd Qu.:1.000   TA  :313    CarPort:  9   3rd Qu.:2002
##  Max.   :3.000   NA's:690    Detchd :387   Max.   :2010
##                              NA's   : 81   NA's   :81
##    GarageCars      GarageArea     GarageQual  GarageCond  PavedDrive
##  Min.   :0.000   Min.   :   0.0   Ex  :   3   Ex  :   2   N:  90
##  1st Qu.:1.000   1st Qu.: 334.5   Fa  :  48   Fa  :  35   P:  30
##  Median :2.000   Median : 480.0   Gd  :  14   Gd  :   9   Y:1340
##  Mean   :1.767   Mean   : 473.0   Po  :   3   Po  :   7
##  3rd Qu.:2.000   3rd Qu.: 576.0   TA  :1311   TA  :1326
##  Max.   :4.000   Max.   :1418.0   NA's:  81   NA's:  81
##
##    WoodDeckSF      OpenPorchSF     EnclosedPorch      X3SsnPorch
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00
##  Median :  0.00   Median : 25.00   Median :  0.00   Median :  0.00
##  Mean   : 94.24   Mean   : 46.66   Mean   : 21.95   Mean   :  3.41
##  3rd Qu.:168.00   3rd Qu.: 68.00   3rd Qu.:  0.00   3rd Qu.:  0.00
##  Max.   :857.00   Max.   :547.00   Max.   :552.00   Max.   :508.00
##
##   ScreenPorch        PoolArea        PoolQC       Fence      MiscFeature
##  Min.   :  0.00   Min.   :  0.000   Ex  :   2   GdPrv:  59   Gar2:   2
##  1st Qu.:  0.00   1st Qu.:  0.000   Fa  :   2   GdWo :  54   Othr:   2
##  Median :  0.00   Median :  0.000   Gd  :   3   MnPrv: 157   Shed:  49
##  Mean   : 15.06   Mean   :  2.759   NA's:1453   MnWw :  11   TenC:   1
##  3rd Qu.:  0.00   3rd Qu.:  0.000               NA's :1179   NA's:1406
##  Max.   :480.00   Max.   :738.000
##
##     MiscVal             MoSold           YrSold        SaleType
##  Min.   :    0.00   Min.   : 1.000   Min.   :2006   WD     :1267
##  1st Qu.:    0.00   1st Qu.: 5.000   1st Qu.:2007   New    : 122
```

```
##  Median :    0.00   Median : 6.000   Median :2008   COD    :  43
##  Mean   :   43.49   Mean   : 6.322   Mean   :2008   ConLD  :   9
##  3rd Qu.:    0.00   3rd Qu.: 8.000   3rd Qu.:2009   ConLI  :   5
##  Max.   :15500.00   Max.   :12.000   Max.   :2010   ConLw  :   5
##                                                     (Other):   9
##  SaleCondition    SalePrice
##  Abnorml: 101   Min.   : 34900
##  AdjLand:   4   1st Qu.:129975
##  Alloca :  12   Median :163000
##  Family :  20   Mean   :180921
##  Normal :1198   3rd Qu.:214000
##  Partial: 125   Max.   :755000
##
```

```
#more. We will eliminate: LotFrontage(c3) , Alley(c6), FireplaceQu(c57),PoolQC(c72), Fence(c73) and
#MiscFeature(c74)
training<-training[,c(-3,-6,-57,-72,-73,-74)]
#we still have some NA values, we are going to erase all the observations with an NA value
training<-na.omit(training)#now our training set has no more NA values
#we also have to convert al character variables to factor variables, but there are none

install.packages("randomForest",repos="https://github.com/WardPlessers/R-programming-Challenge-2")
#we're #going to install the package, randomforest
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
pricemodel<-randomForest(SalePrice~.,data=training, ntree= 400, nodesize=6) #we make our model, we
#will take the number of trees as 400 and the amont of branches as 6
```

Step 3

```
data2<-read.csv("test.csv")
#we want the data without the forst column of identities
testset<-data[,c(2:81)]
testset<-testset[,c(-3,-6,-57,-72,-73,-74)]#we want the testset to have the same variables as the
#training set
testset<-na.omit(testset) #we dont want observations with NA values
#we also have to convert al character variables to factor variables, but there are none

p<-predict(pricemodel,testset)
sumerrorforest<-sum((p-testset$SalePrice)^2)#we're gonna look for the sum squared error between the
#predicted value by the model, and the actual value of the test set
meansqerrorforest<-sqrt(sumerrorforest/length(testset$SalePrice)) #this is the MSE of the predicions
#and the real value


#now we're gonna use a normal lineair regression model and afterwards compare the two methods
lm1<-lm(SalePrice~.,data=training)

p2<-predict(lm1,testset)

sumerrorlm<-sum((testset$SalePrice-p2)^2)
meansqerrorlm<-sqrt(sumerrorlm/length(testset$SalePrice)) #this is again the MSE of the predicted
```

```
#values and the real value

#now to compare the two models
diff<-meansqerrorforest-meansqerrorlm
diff #This number is negative, this means that the error in the lineair model is a bigger than the one
```

```
## [1] -9232.59
```

```
#from the random forest model, thus we can conclude that the random forest model gives a better
#predicition.
```

Task 2B - Overfitting in Machine Learning

Step 1

```
#creating the data for this excercise:
set.seed(1200)
ns <- 150
e <- rnorm(n=ns , mean = 0, sd = 1)
x <- rnorm(n=ns, mean = 0, sd = 1)
y <- x^3+e
df <- data.frame(y,x)

trainingset=df[1:120,]
testset=df[121:150,] #we take 120 observations as traing set, and 30 as testset

install.packages("np",repos="https://github.com/WardPlessers/R-programming-Challenge-2") #we install
#np #to make use of the function npreg
library(tidyverse) #we load tidyverse in order to use ggplot
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## combine(): dplyr, randomForest
## filter():  dplyr, stats
## lag():     dplyr, stats
## margin():  ggplot2, randomForest
```

```
library(np)
```

```
## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-4)
## [vignette("np_faq",package="np") provides answers to frequently asked questions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]
```

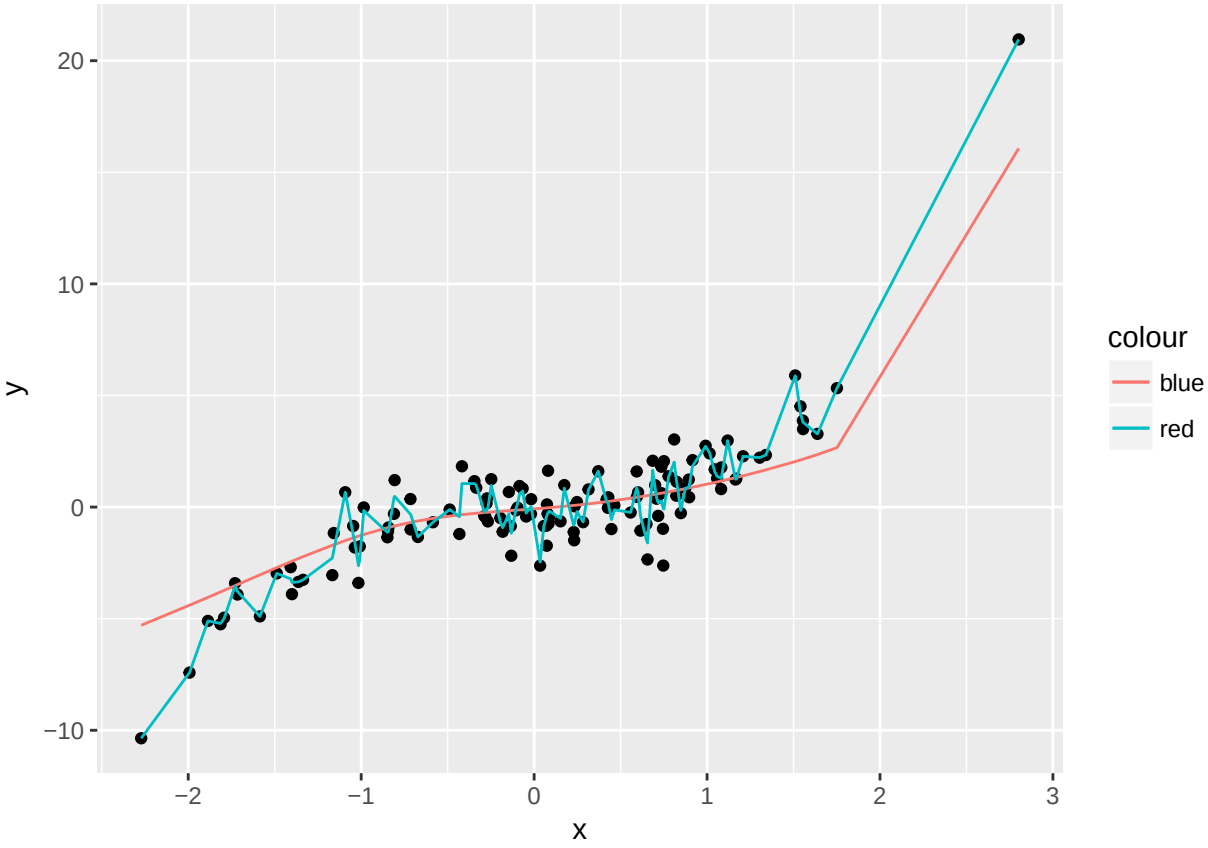
```
ll.fit.lowflex=npreg(trainingset, formula = y ~ x, method = "ll", bws = 0.5) #making the ll.fit.lowflex
#model
```

Step 2

```
ll.fit.highflex=npreg(trainingset, formula = y ~ x, method = "ll", bws=0.01) #making the #ll.fit.highfl
#model
```

Step 3

```
ggplot(data = trainingset) + geom_point(aes(x = x, y = y)) +
  geom_line(aes(x = x, y = ll.fit.lowflex$mean, color = "blue")) +
  geom_line(aes(x = x, y = ll.fit.highflex$mean, color = "red")) #we plot the traing data, the
```



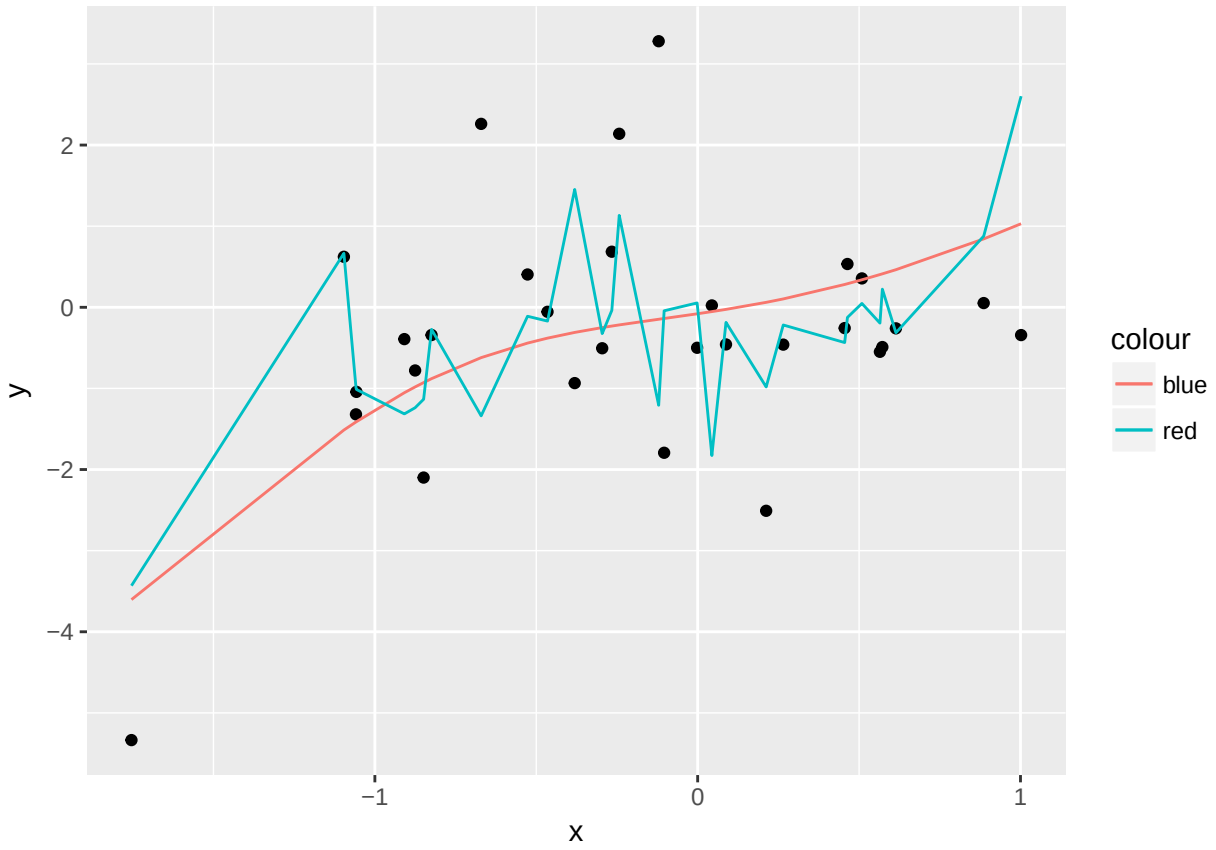

```
#overfitting and the underfitting #model
```

Step 4

The highflexmodel is an example of overfitting, you make your model really specific for your prediction of the training data. When you use it on other data to test your model the performance will not be that great because you're predictions will use a lot of coincidences of the training data. The lowflex model is an example of underfitting. You don't make your predicitions specific enough. Bacause of this, when you use your model on the test data, the results will not be that good. The highflex model is more variable than the lowflex model.(as the highflex model is a lot more specific and takes all the variations in acoount, while the lowflex model doesn't variate that much) The bias of the highflex model is also lower than the bias of the lowflex model. (bias indicates the variance of the model itself, so how much the model fluctuates around its mean)

Step 5

```
ggplot(data = testset) + geom_point(aes(x = x, y = y)) +
  geom_line(aes(x = x, y = predict(ll.fit.lowflex, newdata=testset), color = "blue")) +
  geom_line(aes(x = x, y = predict(ll.fit.highflex, newdata=testset), color = "red"))
```




```
#the predictions of the highflex model are a lot more variable than the ones of the low flex model
#the bias of the least biased model (highflex model) has now become a lot higer than the bias of the
#lowflex model.
```

Step 6

```
bandwidth=c(seq(0.01,0.5,0.001))
```

Step 7

```
linearmodelsTrain=list(rep(0, length(bandwidth)))
for(i in 1:length(bandwidth)){
  linearmodelsTrain[[i]]=npreg(trainingset, formula = y ~ x, method="ll", bws=bandwidth[i])
}
```

Step 8

```
MSETrain=c(rep(0,length(bandwidth)))
for(i in 1:length(bandwidth)){
  MSETrain[i]=linearmodelsTrain[[i]]$MSE
}
```

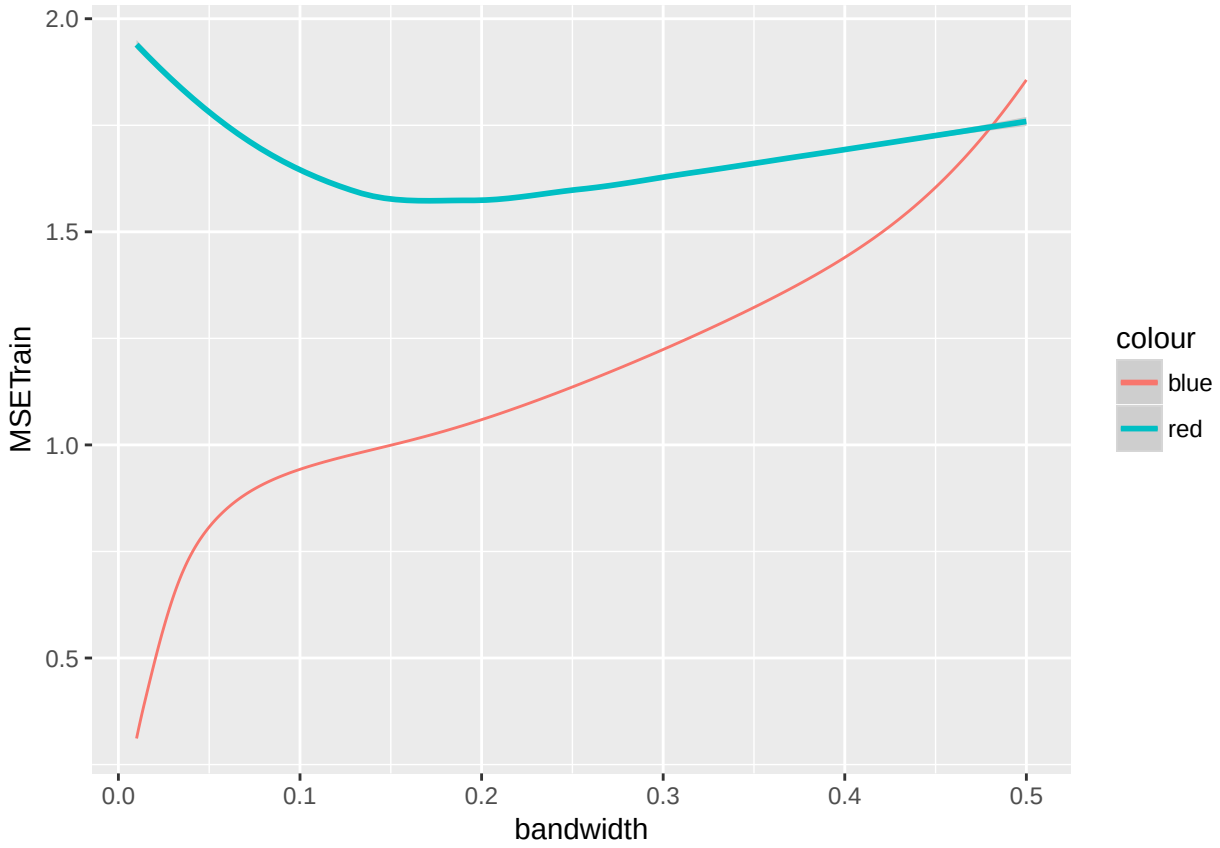
Step 9

```
MSETest <- c(rep(0,length(bandwidth)))
for(i in 1:length(bandwidth)){
  MSETest[i] <- mean((testset[,1] - predict(linearmodelsTrain[[i]], newdata = testset))^2)
}
```

Step 10

```r
MSEdata<-data.frame(MSETrain,MSETest,bandwidth)
ggplot(data=MSEdata) +
  geom_line(mapping=aes(x=bandwidth, y=MSETrain, color="blue")) +
  geom_smooth(mapping=aes(x=bandwidth, y=MSETest, color="red"))
```

```
## `geom_smooth()` using method = 'loess'
```




```r
#as expected, when we have a higher bandwidth, the MSE on the training data will become bigger and
#bigger. This is because you will have more and more underfitting and your model will be less
#precise. On the test tdata on the other hand we can observe antoher phenomenon. As we increase the
#bandwidth, the mse of the test data will first deccrease and afterwards increase. The decrease is
#because you have #less and les overfitting, the increase is when you have more and more underfitting.
#Thus in the minimum of this function you will have your optimal bandwidth to have the best possible
#model.
```

Task 3B - Privacy regulation compliance in France

Step 1

```r
CNIL <- read.csv("OpenCNIL_Organismes_avec_CIL_VD_20171204.csv", header=TRUE, sep = ";") #we read
#the data from the CNIL file
```

Step 2

```r
CNIL$Department <- as.factor(substr(CNIL$Code_Postal, start = 1,stop = 2)) #we take the first 2
#numbers of the number because we know these indicate in wich department they are situated
CNIL_Siren_and_Department <- CNIL %>% count(Department,Siren) #we make a table of all the
#departments and their SIREN number
names(CNIL_Siren_and_Department) <- c("Department", "Siren", "Deleguates") #we give a name to the
```

```
#columns
CNIL_dep <- CNIL_Siren_and_Department %>% count(Department) #we make a table with the number of the
#department, and the amount of deleguates in this department
names(CNIL_dep) <- c("Department", "Deleguates") #we give the columns fitting names
CNIL_dep
```

```
## # A tibble: 96 x 2
##    Department Deleguates
##        <fctr>      <int>
##  1                     2
##  2         01         31
##  3         02         14
##  4         03          6
##  5         04         34
##  6         05         18
##  7         06         12
##  8         07         13
##  9         08         39
## 10         10         33
## # ... with 86 more rows
```

Step 3

```
#when the data is large, but managable, we use packages in R that make sure we don't have to store
#all the data in the memory all the time. Therefore we use the package datatable with its function
#fread
install.packages("data.table",repos="https://github.com/WardPlessers/R-programming-Challenge-2")
library(data.table)
```

```
##
## Attaching package: 'data.table'
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
#we make sure the data we need is stored in our local directory
#because the data is that big, we're only going to load a part of the variables, more specific
#EFENCENT (which indicates the size of the company (amount of employees)) and SIREN (because it's a
#common variable we use to merge the two tables)
system.time(fread("sirc-17804_9075_14209_201710_L_M_20171101_030132835.csv",
                  header = TRUE, sep=";",select = c("SIREN","EFENCENT"),na.omit))
```

```
##
Read 0.0% of 10831176 rows
Read 1.4% of 10831176 rows
Read 2.8% of 10831176 rows
Read 4.2% of 10831176 rows
Read 5.4% of 10831176 rows
Read 6.6% of 10831176 rows
Read 7.8% of 10831176 rows
Read 8.9% of 10831176 rows
Read 9.9% of 10831176 rows
```

```
Read 11.4% of 10831176 rows
Read 12.6% of 10831176 rows
Read 14.1% of 10831176 rows
Read 15.5% of 10831176 rows
Read 16.7% of 10831176 rows
Read 18.1% of 10831176 rows
Read 19.5% of 10831176 rows
Read 19.9% of 10831176 rows
Read 21.3% of 10831176 rows
Read 22.7% of 10831176 rows
Read 24.1% of 10831176 rows
Read 25.4% of 10831176 rows
Read 26.8% of 10831176 rows
Read 28.3% of 10831176 rows
Read 28.3% of 10831176 rows
Read 29.8% of 10831176 rows
Read 31.3% of 10831176 rows
Read 32.0% of 10831176 rows
Read 33.4% of 10831176 rows
Read 34.7% of 10831176 rows
Read 35.9% of 10831176 rows
Read 37.1% of 10831176 rows
Read 38.2% of 10831176 rows
Read 39.4% of 10831176 rows
Read 40.8% of 10831176 rows
Read 42.0% of 10831176 rows
Read 43.2% of 10831176 rows
Read 44.5% of 10831176 rows
Read 45.8% of 10831176 rows
Read 47.1% of 10831176 rows
Read 48.4% of 10831176 rows
Read 49.6% of 10831176 rows
Read 50.0% of 10831176 rows
Read 51.2% of 10831176 rows
Read 52.4% of 10831176 rows
Read 53.6% of 10831176 rows
Read 54.7% of 10831176 rows
Read 55.9% of 10831176 rows
Read 57.1% of 10831176 rows
Read 58.2% of 10831176 rows
Read 59.2% of 10831176 rows
Read 60.2% of 10831176 rows
Read 61.2% of 10831176 rows
Read 62.3% of 10831176 rows
Read 63.3% of 10831176 rows
Read 64.1% of 10831176 rows
Read 65.4% of 10831176 rows
Read 66.6% of 10831176 rows
Read 68.0% of 10831176 rows
Read 69.2% of 10831176 rows
Read 70.5% of 10831176 rows
Read 71.8% of 10831176 rows
Read 73.1% of 10831176 rows
Read 74.4% of 10831176 rows
```

```
Read 75.6% of 10831176 rows
Read 76.7% of 10831176 rows
Read 77.8% of 10831176 rows
Read 78.9% of 10831176 rows
Read 80.0% of 10831176 rows
Read 81.0% of 10831176 rows
Read 81.6% of 10831176 rows
Read 82.8% of 10831176 rows
Read 83.9% of 10831176 rows
Read 85.0% of 10831176 rows
Read 86.2% of 10831176 rows
Read 87.4% of 10831176 rows
Read 88.5% of 10831176 rows
Read 89.7% of 10831176 rows
Read 90.9% of 10831176 rows
Read 92.0% of 10831176 rows
Read 93.2% of 10831176 rows
Read 94.4% of 10831176 rows
Read 95.5% of 10831176 rows
Read 96.5% of 10831176 rows
Read 97.5% of 10831176 rows
Read 98.5% of 10831176 rows
Read 99.4% of 10831176 rows
Read 10831176 rows and 2 (of 100) columns from 8.068 GB file in 00:02:37
```

```
##    user  system elapsed
##  89.546  67.514 321.666
```

```
bigdata <- fread("sirc-17804_9075_14209_201710_L_M_20171101_030132835.csv",
                 header = TRUE, sep=";",select = c("SIREN","EFENCENT"),na.omit)
```

```
##
Read 0.0% of 10831176 rows
Read 1.6% of 10831176 rows
Read 3.2% of 10831176 rows
Read 4.8% of 10831176 rows
Read 6.3% of 10831176 rows
Read 7.7% of 10831176 rows
Read 9.1% of 10831176 rows
Read 10.4% of 10831176 rows
Read 11.6% of 10831176 rows
Read 12.8% of 10831176 rows
Read 14.0% of 10831176 rows
Read 15.1% of 10831176 rows
Read 16.2% of 10831176 rows
Read 17.4% of 10831176 rows
Read 18.5% of 10831176 rows
Read 19.6% of 10831176 rows
Read 20.7% of 10831176 rows
Read 21.9% of 10831176 rows
Read 23.0% of 10831176 rows
Read 24.1% of 10831176 rows
Read 25.3% of 10831176 rows
Read 26.5% of 10831176 rows
Read 27.7% of 10831176 rows
```

```
Read 28.8% of 10831176 rows
Read 29.9% of 10831176 rows
Read 31.1% of 10831176 rows
Read 32.2% of 10831176 rows
Read 33.3% of 10831176 rows
Read 34.5% of 10831176 rows
Read 35.7% of 10831176 rows
Read 36.9% of 10831176 rows
Read 38.0% of 10831176 rows
Read 39.2% of 10831176 rows
Read 40.6% of 10831176 rows
Read 41.8% of 10831176 rows
Read 43.0% of 10831176 rows
Read 44.1% of 10831176 rows
Read 45.2% of 10831176 rows
Read 46.3% of 10831176 rows
Read 47.5% of 10831176 rows
Read 48.6% of 10831176 rows
Read 49.7% of 10831176 rows
Read 50.8% of 10831176 rows
Read 51.9% of 10831176 rows
Read 53.0% of 10831176 rows
Read 54.2% of 10831176 rows
Read 55.3% of 10831176 rows
Read 56.6% of 10831176 rows
Read 57.7% of 10831176 rows
Read 58.9% of 10831176 rows
Read 60.1% of 10831176 rows
Read 61.2% of 10831176 rows
Read 62.3% of 10831176 rows
Read 63.6% of 10831176 rows
Read 64.8% of 10831176 rows
Read 66.0% of 10831176 rows
Read 67.2% of 10831176 rows
Read 68.4% of 10831176 rows
Read 69.6% of 10831176 rows
Read 70.8% of 10831176 rows
Read 72.0% of 10831176 rows
Read 73.1% of 10831176 rows
Read 74.2% of 10831176 rows
Read 75.3% of 10831176 rows
Read 76.4% of 10831176 rows
Read 77.5% of 10831176 rows
Read 78.0% of 10831176 rows
Read 79.3% of 10831176 rows
Read 80.5% of 10831176 rows
Read 81.7% of 10831176 rows
Read 82.8% of 10831176 rows
Read 83.9% of 10831176 rows
Read 85.1% of 10831176 rows
Read 86.3% of 10831176 rows
Read 87.4% of 10831176 rows
Read 88.5% of 10831176 rows
Read 89.6% of 10831176 rows
```

```
Read 90.8% of 10831176 rows
Read 91.8% of 10831176 rows
Read 92.8% of 10831176 rows
Read 93.7% of 10831176 rows
Read 94.7% of 10831176 rows
Read 95.7% of 10831176 rows
Read 96.9% of 10831176 rows
Read 98.1% of 10831176 rows
Read 99.3% of 10831176 rows
Read 10831176 rows and 2 (of 100) columns from 8.068 GB file in 00:02:35
```

```
bigdata$SIREN<-as.integer(bigdata$SIREN) #we have to make sure both Siren columns are of the same
#variable type. Therefore we convert the SIREN column of bigdata into an integer column

install.packages("dplyr",repos="https://github.com/WardPlessers/R-programming-Challenge-2")
library(dplyr)
datamerged<-right_join(bigdata,CNIL,by=c("SIREN"="Siren"))
```

Step 4

```
datamerged$EFENCENT<-as.integer(datamerged$EFENCENT)
ggplot(datamerged) + geom_histogram(aes(EFENCENT))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```