KM.R

US16120

Wed Sep 26 16:26:06 2018

library(rcdk)

## Loading required package: rcdklibs

## Loading required package: rJava

library(tidyverse)

## -- Attaching packages ------------------------------------------------------ tidyverse 1.2.1 --

## v ggplot2 3.0.0 v purrr 0.2.5  
## v tibble 1.4.2 v dplyr 0.7.6  
## v tidyr 0.8.1 v stringr 1.3.1  
## v readr 1.1.1 v forcats 0.3.0

## -- Conflicts --------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::matches() masks rcdk::matches()

library(magrittr)

##   
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':  
##   
## set\_names

## The following object is masked from 'package:tidyr':  
##   
## extract

library(purrr)  
library(stringr)  
library(caret)

## Loading required package: lattice

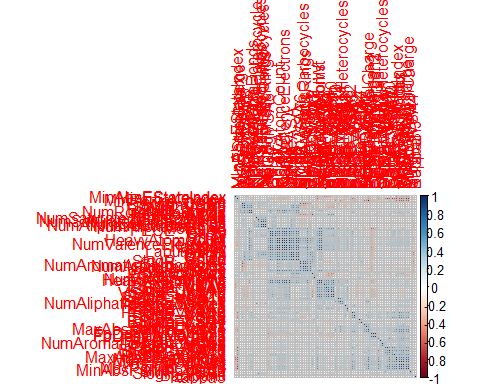
##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

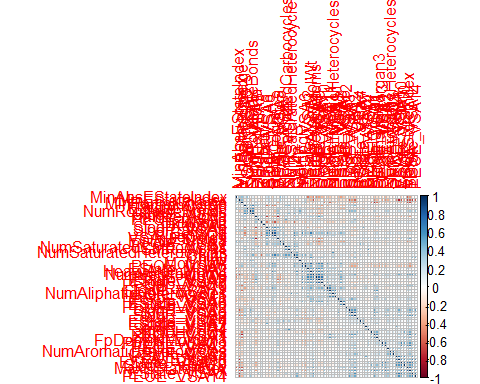
library(corrplot)

## corrplot 0.84 loaded

library(ggplot2)  
library(ggthemes)  
  
# read data  
  
## training data  
train <-  
 read.csv('cache/TR\_KM\_405\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(LogKmHL, everything()) %>%  
 na.omit()  
  
X\_train <- train %>%  
 select(-LogKmHL)  
y\_train <- train %>%  
 select(LogKmHL) %>%  
 data.frame()  
  
## test data  
test <-  
 read.csv('cache/TST\_KM\_136\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(LogKmHL, everything()) %>%  
 na.omit()  
  
X\_test <- test %>%  
 select(-LogKmHL)  
y\_test <- test %>%  
 select(LogKmHL) %>%  
 data.frame()  
  
# curate data  
  
## near-zero variance descriptors  
  
nzv <- nearZeroVar(X\_train, freqCut = 100/0)  
X\_train <- X\_train[ , -nzv]  
### and  
X\_test <- X\_test[ , -nzv]  
  
## highly correlated descriptors  
  
correlations <- cor(X\_train)  
corrplot::corrplot(correlations, order = 'hclust')



highCorr <- findCorrelation(correlations, cutoff = 0.85)  
X\_train <- X\_train[ , -highCorr]  
### and  
X\_test <- X\_test[ , -highCorr]  
  
correlations <- cor(X\_train)  
corrplot::corrplot(correlations, order = 'hclust')



## linear combinations  
  
comboInfo <- findLinearCombos(X\_train) # returns NULL  
# X\_train <- X\_train[ , -comboInfo$remove]  
# ### and  
# X\_test <- X\_test[ , -nzv]  
  
## center & scale descriptors  
  
preProcValues <- preProcess(X\_train, method = c("center", "scale"))  
  
X\_trainTransformed <- predict(preProcValues, X\_train)  
### and  
X\_testTransformed <- predict(preProcValues, X\_test)  
  
### PCA  
  
# pca <- preProcess(X\_trainTransformed, method = c('pca'))  
# X\_train\_pca <- predict(pca, X\_trainTransformed)  
# X\_test\_pca <- predict(pca, X\_testTransformed)  
#   
# train\_pca <- X\_train\_pca %>%  
# select(PC1, PC2) %>%  
# mutate(dataset = 'train')  
# test\_pca <- X\_test\_pca %>%  
# select(PC1, PC2) %>%  
# mutate(dataset = 'test')  
# pcaPts <- rbind(train\_pca, test\_pca)  
#   
# p <-  
# ggplot(pcaPts, aes(PC1, PC2)) +  
# geom\_point(aes(colour = factor(dataset), shape = factor(dataset))) +  
# ggthemes::theme\_tufte()  
# p  
  
# models  
  
fitControl <- trainControl(## 10-fold CV  
 method = "repeatedcv",  
 repeats = 5)  
  
set.seed(350)  
  
## multiple linear regression  
  
trainSet <- cbind(y\_train, X\_trainTransformed)  
  
mlr <- train(LogKmHL ~ .,  
 data = trainSet,  
 method = 'lm',  
 trControl = fitControl)

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient  
## fit may be misleading

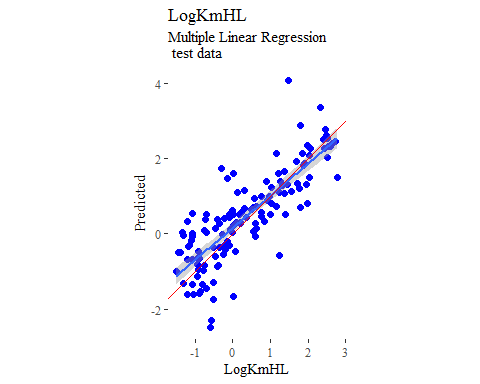
y\_predict <- predict(mlr, newdata = X\_testTransformed) %>%  
 data.frame()

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient  
## fit may be misleading

colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_test, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.12807 -0.39653 -0.00157 0.41365 2.65965   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.15646 0.06433 2.432 0.0163 \*   
## LogKmHL 0.85330 0.05119 16.669 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.7096 on 133 degrees of freedom  
## Multiple R-squared: 0.6763, Adjusted R-squared: 0.6738   
## F-statistic: 277.8 on 1 and 133 DF, p-value: < 2.2e-16

p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Multiple Linear Regression\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



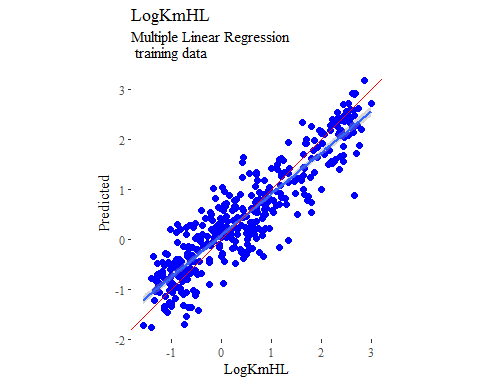
y\_predict <- predict(mlr, newdata = X\_trainTransformed) %>%  
 data.frame()

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient  
## fit may be misleading

colnames(y\_predict) <- c('Predicted')  
  
mlrPR <- postResample(pred = y\_predict, obs = X\_trainTransformed)  
rmse\_train = c(mlrPR[1])  
r2\_train = c(mlrPR[2])  
  
data2plot <- cbind(y\_train, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.40564 -0.30544 0.02621 0.29720 1.20041   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.08091 0.02394 3.379 0.000798 \*\*\*  
## LogKmHL 0.82880 0.01879 44.116 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4469 on 402 degrees of freedom  
## Multiple R-squared: 0.8288, Adjusted R-squared: 0.8284   
## F-statistic: 1946 on 1 and 402 DF, p-value: < 2.2e-16

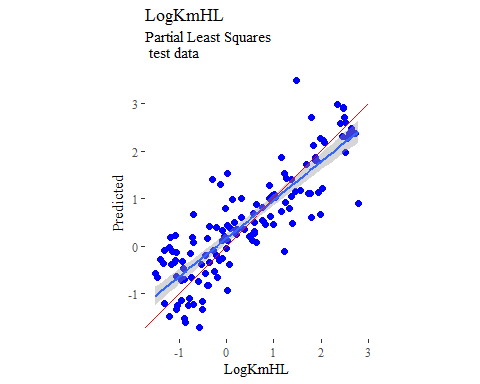
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Multiple Linear Regression\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



## partial least squares  
  
plsModel <- train(  
 LogKmHL ~ .,  
 data = trainSet,  
 method = 'pls',  
 tuneLength = 20,  
 trControl = fitControl  
)  
  
y\_predict <- predict(plsModel, newdata = X\_testTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_test, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.51777 -0.37969 0.03008 0.39578 2.14092   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.16068 0.05508 2.917 0.00415 \*\*   
## LogKmHL 0.81020 0.04383 18.483 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.6076 on 133 degrees of freedom  
## Multiple R-squared: 0.7198, Adjusted R-squared: 0.7177   
## F-statistic: 341.6 on 1 and 133 DF, p-value: < 2.2e-16

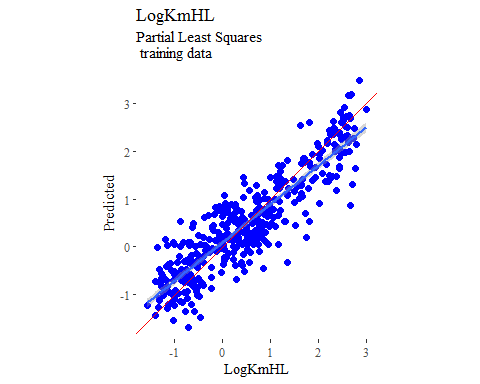
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Partial Least Squares\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



y\_predict <- predict(plsModel, newdata = X\_trainTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_train, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.49022 -0.32833 0.01099 0.33237 1.15310   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.09366 0.02534 3.697 0.000249 \*\*\*  
## LogKmHL 0.80181 0.01988 40.329 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.473 on 402 degrees of freedom  
## Multiple R-squared: 0.8018, Adjusted R-squared: 0.8013   
## F-statistic: 1626 on 1 and 402 DF, p-value: < 2.2e-16

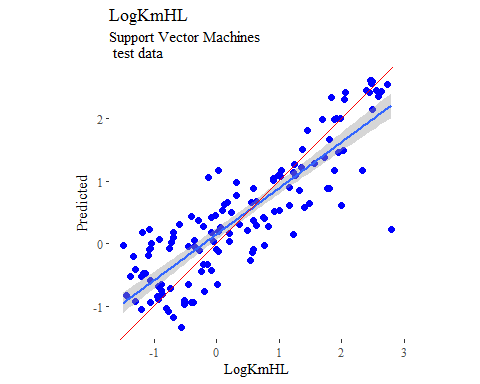
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Partial Least Squares\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



## support vector machines  
  
svmModel <- train(  
 LogKmHL ~ .,  
 data = trainSet,  
 method = 'svmRadial',  
 # tuneLength = 14,  
 trControl = fitControl  
)  
  
y\_predict <- predict(svmModel, newdata = X\_testTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_test, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.97825 -0.33100 0.06966 0.38924 1.00028   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.15324 0.04695 3.264 0.0014 \*\*   
## LogKmHL 0.73683 0.03736 19.723 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5179 on 133 degrees of freedom  
## Multiple R-squared: 0.7452, Adjusted R-squared: 0.7433   
## F-statistic: 389 on 1 and 133 DF, p-value: < 2.2e-16

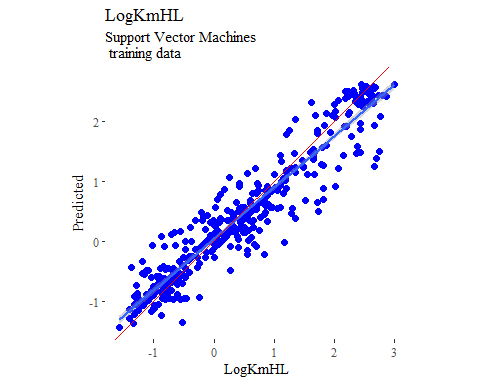
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Support Vector Machines\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



y\_predict <- predict(svmModel, newdata = X\_trainTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_train, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.07359 -0.14868 -0.01303 0.17944 0.89247   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.04297 0.01811 2.372 0.0181 \*   
## LogKmHL 0.85393 0.01421 60.088 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3381 on 402 degrees of freedom  
## Multiple R-squared: 0.8998, Adjusted R-squared: 0.8996   
## F-statistic: 3611 on 1 and 402 DF, p-value: < 2.2e-16

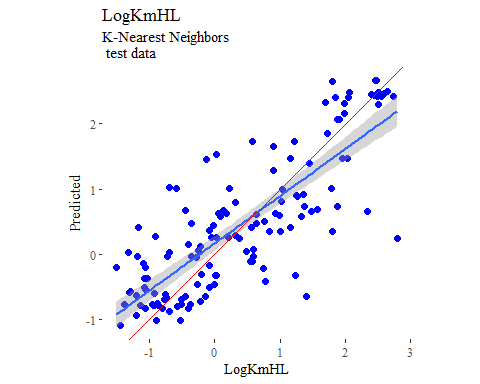
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Support Vector Machines\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



## k-nearest neighbors  
  
knnModel <- train(  
 LogKmHL ~ .,  
 data = trainSet,  
 method = 'knn',  
 tuneGrid = data.frame(.k = 1:20),  
 trControl = fitControl  
)  
  
y\_predict <- predict(knnModel, newdata = X\_testTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_test, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.9380 -0.4403 -0.0617 0.4235 1.3784   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.17133 0.05629 3.043 0.00282 \*\*   
## LogKmHL 0.72331 0.04480 16.146 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.621 on 133 degrees of freedom  
## Multiple R-squared: 0.6622, Adjusted R-squared: 0.6596   
## F-statistic: 260.7 on 1 and 133 DF, p-value: < 2.2e-16

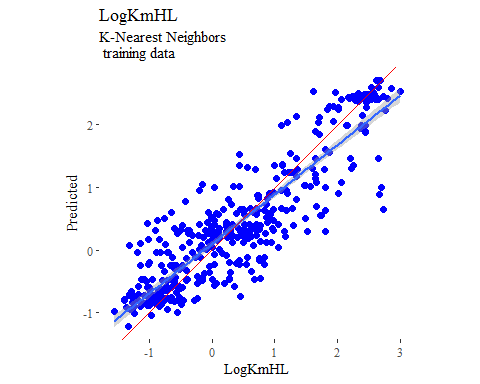
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'K-Nearest Neighbors\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



y\_predict <- predict(knnModel, newdata = X\_trainTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_train, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.61056 -0.30217 -0.00083 0.30896 1.15966   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.10199 0.02536 4.022 6.89e-05 \*\*\*  
## LogKmHL 0.78910 0.01990 39.659 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4734 on 402 degrees of freedom  
## Multiple R-squared: 0.7964, Adjusted R-squared: 0.7959   
## F-statistic: 1573 on 1 and 402 DF, p-value: < 2.2e-16

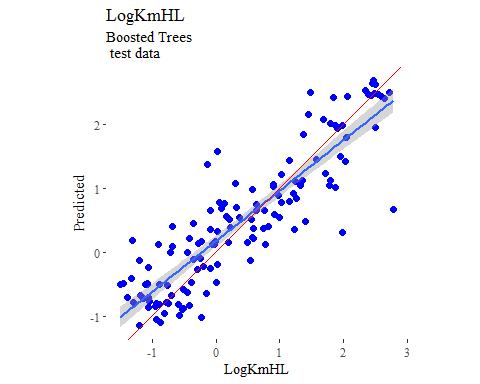
p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'K-Nearest Neighbors\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



## boosted trees  
  
gbmGrid <- expand.grid(  
 .interaction.depth = seq(1, 7, by = 2),  
 .n.trees = seq(100, 1000, by = 50),  
 .shrinkage = c(0.001, 0.1),  
 .n.minobsinnode = 3  
)  
  
treeModel <- train(  
 LogKmHL ~ .,  
 data = trainSet,  
 method = 'gbm',  
 tuneGrid = gbmGrid,  
 verbose = FALSE  
)  
  
y\_predict <- predict(treeModel, newdata = X\_testTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_test, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.70755 -0.34801 0.00644 0.34888 1.38716   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.17398 0.04492 3.873 0.000168 \*\*\*  
## LogKmHL 0.79272 0.03575 22.177 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4955 on 133 degrees of freedom  
## Multiple R-squared: 0.7871, Adjusted R-squared: 0.7855   
## F-statistic: 491.8 on 1 and 133 DF, p-value: < 2.2e-16

p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Boosted Trees\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



y\_predict <- predict(treeModel, newdata = X\_trainTransformed) %>%  
 data.frame()  
colnames(y\_predict) <- c('Predicted')  
  
data2plot <- cbind(y\_train, y\_predict)  
  
summary(lm(Predicted ~ LogKmHL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogKmHL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.49430 -0.13045 0.00488 0.12916 0.56033   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.029272 0.010315 2.838 0.00478 \*\*   
## LogKmHL 0.940697 0.008094 116.221 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1926 on 402 degrees of freedom  
## Multiple R-squared: 0.9711, Adjusted R-squared: 0.971   
## F-statistic: 1.351e+04 on 1 and 402 DF, p-value: < 2.2e-16

p <-  
 ggplot(data2plot, aes(LogKmHL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogKmHL',  
 subtitle = 'Boosted Trees\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p

