HL.R

US16120

Wed Sep 26 15:56:59 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

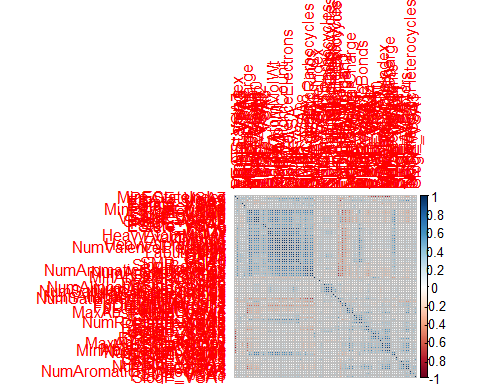
##   
## Attaching package: 'caret'

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

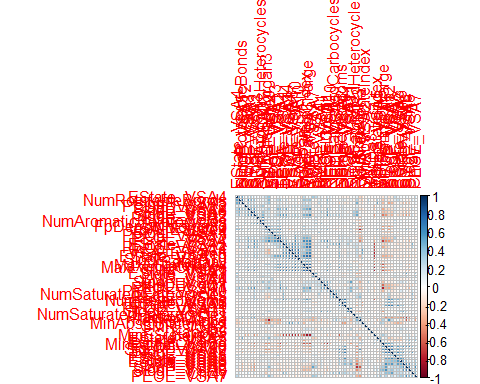
library(corrplot)

## corrplot 0.84 loaded

library(ggplot2)  
library(ggthemes)  
  
# read data  
  
## training data  
train <-  
 read.csv('cache/TR\_HL\_441\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(HL, everything()) %>%  
 na.omit()  
  
X\_train <- train %>%  
 select(-HL)  
y\_train <- train %>%  
 select(HL) %>%  
 data.frame()  
  
## test data  
test <-  
 read.csv('cache/TST\_HL\_150\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(HL, everything()) %>%  
 na.omit()  
  
X\_test <- test %>%  
 select(-HL)  
y\_test <- test %>%  
 select(HL) %>%  
 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(HL ~ .,  
 data = trainSet,  
 method = 'lm',  
 trControl = fitControl)

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

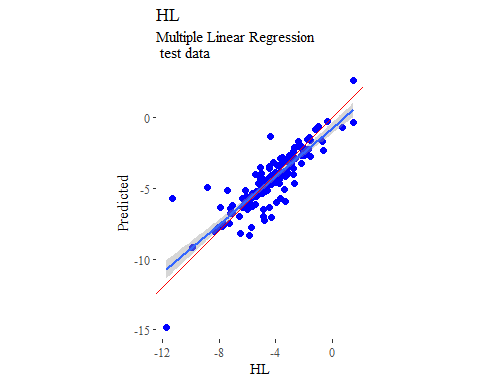
y\_predict <- predict(mlr, newdata = X\_testTransformed) %>%  
 data.frame()

## 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 ~ HL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.0680 -0.3521 0.0220 0.4259 4.6692   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.74809 0.20001 -3.74 0.000263 \*\*\*  
## HL 0.85066 0.04161 20.44 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.042 on 147 degrees of freedom  
## Multiple R-squared: 0.7398, Adjusted R-squared: 0.738   
## F-statistic: 417.9 on 1 and 147 DF, p-value: < 2.2e-16

p <-  
 ggplot(data2plot, aes(HL, 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 = 'HL',  
 subtitle = 'Multiple Linear Regression\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p



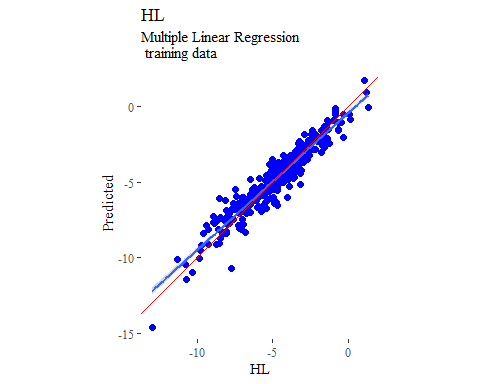
y\_predict <- predict(mlr, newdata = X\_trainTransformed) %>%  
 data.frame()

## 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 ~ HL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.2511 -0.3176 0.0051 0.3776 2.0776   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.42953 0.07051 -6.091 2.46e-09 \*\*\*  
## HL 0.90695 0.01390 65.264 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.6135 on 437 degrees of freedom  
## Multiple R-squared: 0.9069, Adjusted R-squared: 0.9067   
## F-statistic: 4259 on 1 and 437 DF, p-value: < 2.2e-16

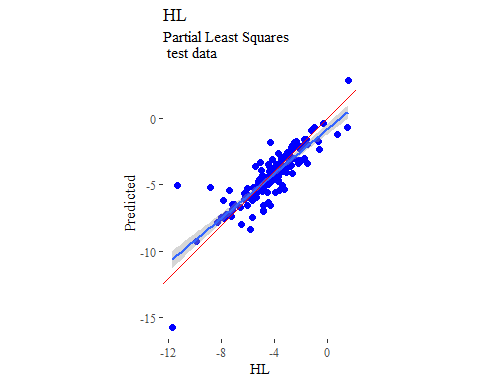
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'HL',  
 subtitle = 'Multiple Linear Regression\n training data') +  
 ggthemes::theme\_tufte()  
p 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



## partial least squares  
  
plsModel <- train(  
 HL ~ .,  
 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 ~ HL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -5.0847 -0.4192 0.0640 0.4674 5.2285   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.77558 0.20447 -3.793 0.000217 \*\*\*  
## HL 0.83958 0.04254 19.736 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.066 on 147 degrees of freedom  
## Multiple R-squared: 0.726, Adjusted R-squared: 0.7241   
## F-statistic: 389.5 on 1 and 147 DF, p-value: < 2.2e-16

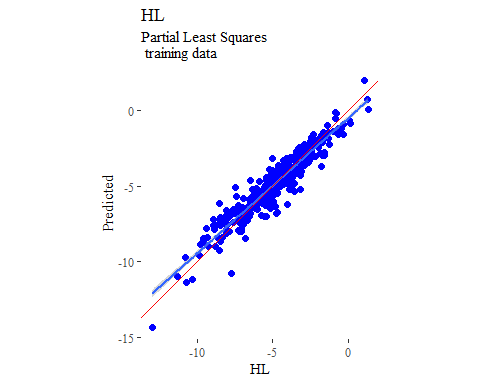
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'HL',  
 subtitle = 'Partial Least Squares\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.3346 -0.3637 0.0377 0.3974 2.0965   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.48603 0.07450 -6.524 1.9e-10 \*\*\*  
## HL 0.89471 0.01468 60.938 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.6482 on 437 degrees of freedom  
## Multiple R-squared: 0.8947, Adjusted R-squared: 0.8945   
## F-statistic: 3713 on 1 and 437 DF, p-value: < 2.2e-16

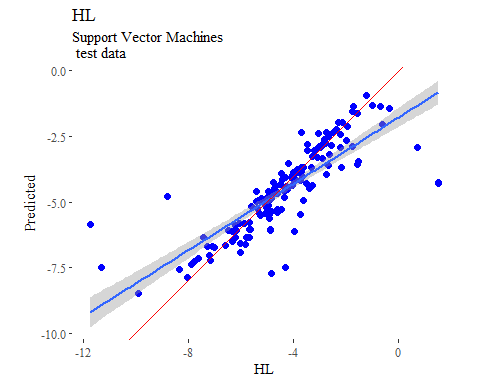
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'HL',  
 subtitle = 'Partial Least Squares\n training data') +  
 ggthemes::theme\_tufte()  
p <- p +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



## support vector machines  
  
svmModel <- train(  
 HL ~ .,  
 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 ~ HL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.4566 -0.5109 0.0560 0.4466 3.3500   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.77700 0.17910 -9.922 <2e-16 \*\*\*  
## HL 0.63158 0.03726 16.950 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9334 on 147 degrees of freedom  
## Multiple R-squared: 0.6615, Adjusted R-squared: 0.6592   
## F-statistic: 287.3 on 1 and 147 DF, p-value: < 2.2e-16

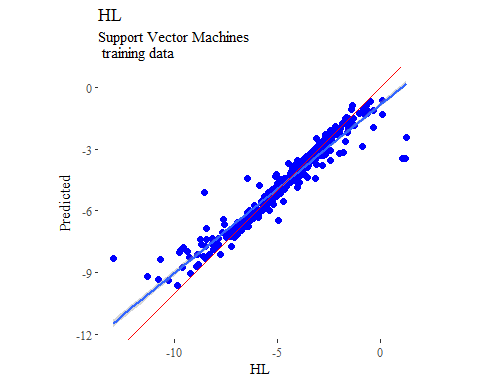
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'HL',  
 subtitle = 'Support Vector Machines\n test data') +  
 ggthemes::theme\_tufte()  
p <- p +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

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.5972 -0.2303 0.0191 0.2332 3.1355   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.83723 0.05866 -14.27 <2e-16 \*\*\*  
## HL 0.81784 0.01156 70.75 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5103 on 437 degrees of freedom  
## Multiple R-squared: 0.9197, Adjusted R-squared: 0.9195   
## F-statistic: 5005 on 1 and 437 DF, p-value: < 2.2e-16

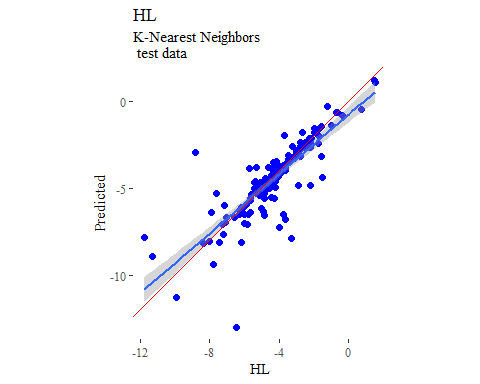
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'HL',  
 subtitle = 'Support Vector Machines\n training data') +  
 ggthemes::theme\_tufte()  
p <- p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



## k-nearest neighbors  
  
knnModel <- train(  
 HL ~ .,  
 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 ~ HL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -6.6745 -0.2158 0.1426 0.4644 5.4064   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.75598 0.22675 -3.334 0.00108 \*\*   
## HL 0.85710 0.04718 18.168 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.182 on 147 degrees of freedom  
## Multiple R-squared: 0.6919, Adjusted R-squared: 0.6898   
## F-statistic: 330.1 on 1 and 147 DF, p-value: < 2.2e-16

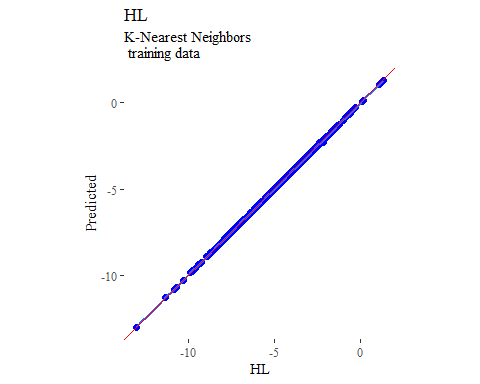
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'HL',  
 subtitle = 'K-Nearest Neighbors\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.108648 -0.000014 0.000001 0.000016 0.108704   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -5.595e-05 8.451e-04 -0.066 0.947   
## HL 1.000e+00 1.665e-04 6004.344 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.007352 on 437 degrees of freedom  
## Multiple R-squared: 1, Adjusted R-squared: 1   
## F-statistic: 3.605e+07 on 1 and 437 DF, p-value: < 2.2e-16

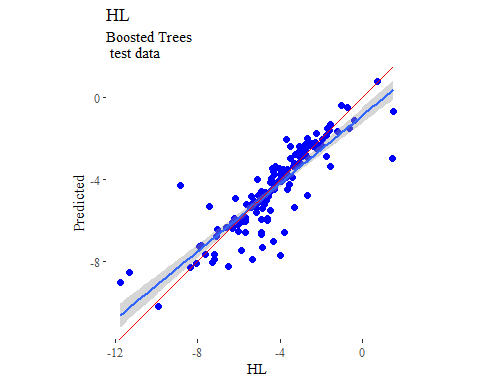
p <-  
 ggplot(data2plot, aes(HL, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'HL',  
 subtitle = 'K-Nearest Neighbors\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.4872 -0.2878 0.1157 0.4585 3.8619   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.90017 0.18484 -4.87 2.85e-06 \*\*\*  
## HL 0.82521 0.03846 21.46 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9633 on 147 degrees of freedom  
## Multiple R-squared: 0.758, Adjusted R-squared: 0.7564   
## F-statistic: 460.5 on 1 and 147 DF, p-value: < 2.2e-16

p <-  
 ggplot(data2plot, aes(HL, 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 = 'HL',  
 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 ~ HL, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ HL, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.22757 -0.04193 -0.00145 0.04218 0.18331   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.019534 0.007499 -2.605 0.00951 \*\*   
## HL 0.995585 0.001478 673.663 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.06524 on 437 degrees of freedom  
## Multiple R-squared: 0.999, Adjusted R-squared: 0.999   
## F-statistic: 4.538e+05 on 1 and 437 DF, p-value: < 2.2e-16

p <-  
 ggplot(data2plot, aes(HL, 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 = 'HL',  
 subtitle = 'Boosted Trees\n training data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept = 0,  
 slope = 1,  
 colour = 'red')  
p

