VP.R

US16120

Wed Sep 26 21:59:07 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

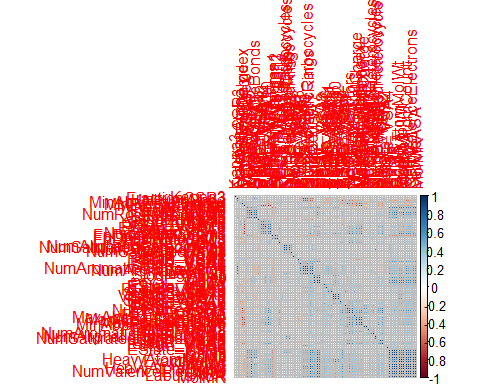
##   
## Attaching package: 'caret'

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

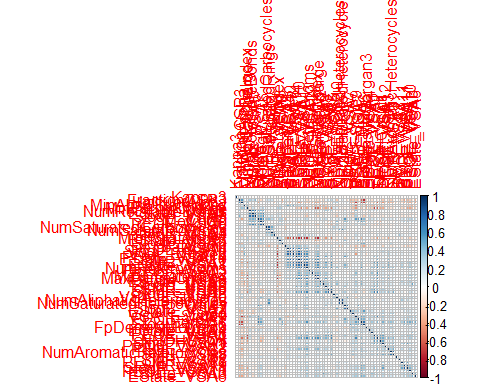
library(corrplot)

## corrplot 0.84 loaded

library(ggplot2)  
library(ggthemes)  
  
# read data  
  
## training data  
train <-  
 read.csv('cache/TR\_VP\_2034\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(LogVP, everything()) %>%  
 na.omit()  
  
X\_train <- train %>%  
 select(-LogVP)  
y\_train <- train %>%  
 select(LogVP) %>%  
 data.frame()  
  
## test data  
test <-  
 read.csv('cache/TST\_VP\_679\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(LogVP, everything()) %>%  
 na.omit()  
  
X\_test <- test %>%  
 select(-LogVP)  
y\_test <- test %>%  
 select(LogVP) %>%  
 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(LogVP ~ .,  
 data = trainSet,  
 method = 'lm',  
 trControl = fitControl)

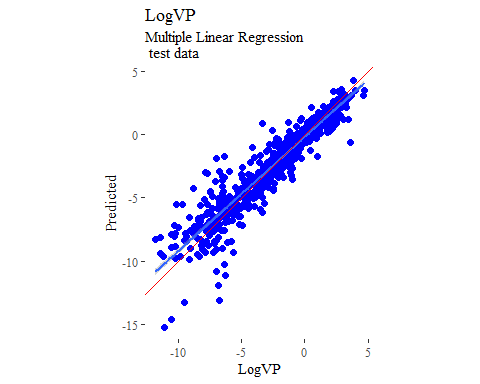
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient  
## fit may be misleading  
  
y\_predict <- predict(mlr, newdata = X\_testTransformed) %>%  
 data.frame()

## 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 ~ LogVP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -6.7767 -0.4560 0.0772 0.4914 4.5895   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.18899 0.05238 -3.608 0.000331 \*\*\*  
## LogVP 0.90328 0.01262 71.559 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.167 on 675 degrees of freedom  
## Multiple R-squared: 0.8835, Adjusted R-squared: 0.8834   
## F-statistic: 5121 on 1 and 675 DF, p-value: < 2.2e-16

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



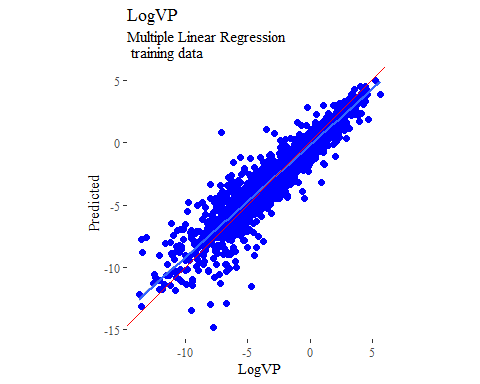
y\_predict <- predict(mlr, newdata = X\_trainTransformed) %>%  
 data.frame()

## 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 ~ LogVP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.5951 -0.4558 0.0891 0.4867 7.4389   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.19906 0.02723 -7.31 3.84e-13 \*\*\*  
## LogVP 0.90073 0.00665 135.45 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.068 on 2022 degrees of freedom  
## Multiple R-squared: 0.9007, Adjusted R-squared: 0.9007   
## F-statistic: 1.835e+04 on 1 and 2022 DF, p-value: < 2.2e-16

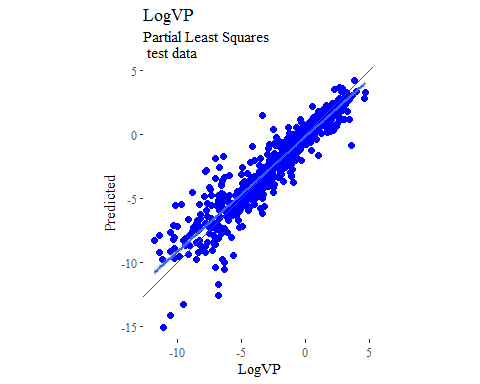
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogVP',  
 subtitle = 'Multiple Linear Regression\n training data') +  
 ggthemes::theme\_tufte()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



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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -6.3162 -0.4859 0.0905 0.4879 4.6674   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.19367 0.05232 -3.702 0.000232 \*\*\*  
## LogVP 0.89740 0.01261 71.177 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.166 on 675 degrees of freedom  
## Multiple R-squared: 0.8824, Adjusted R-squared: 0.8823   
## F-statistic: 5066 on 1 and 675 DF, p-value: < 2.2e-16

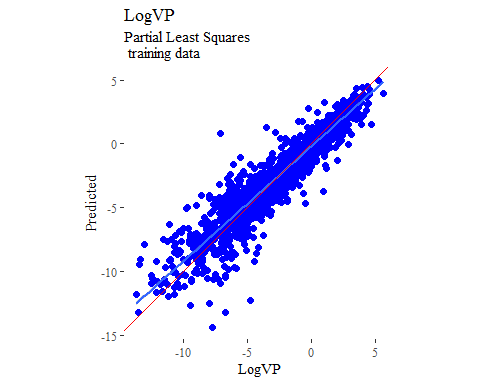
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogVP',  
 subtitle = 'Partial Least Squares\n test data') +  
 ggthemes::theme\_tufte()  
p <-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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.7732 -0.4670 0.0855 0.5065 7.4380   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.204653 0.027570 -7.423 1.68e-13 \*\*\*  
## LogVP 0.897948 0.006732 133.384 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.081 on 2022 degrees of freedom  
## Multiple R-squared: 0.8979, Adjusted R-squared: 0.8979   
## F-statistic: 1.779e+04 on 1 and 2022 DF, p-value: < 2.2e-16

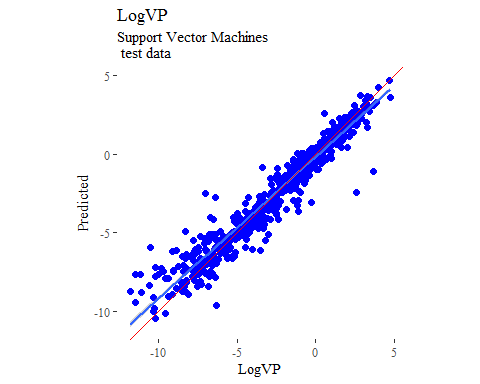
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogVP',  
 subtitle = 'Partial Least Squares\n training data') +  
 ggthemes::theme\_tufte()  
p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



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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.5877 -0.4161 0.0690 0.4129 4.0174   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.163346 0.036843 -4.434 1.08e-05 \*\*\*  
## LogVP 0.902093 0.008879 101.604 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8211 on 675 degrees of freedom  
## Multiple R-squared: 0.9386, Adjusted R-squared: 0.9385   
## F-statistic: 1.032e+04 on 1 and 675 DF, p-value: < 2.2e-16

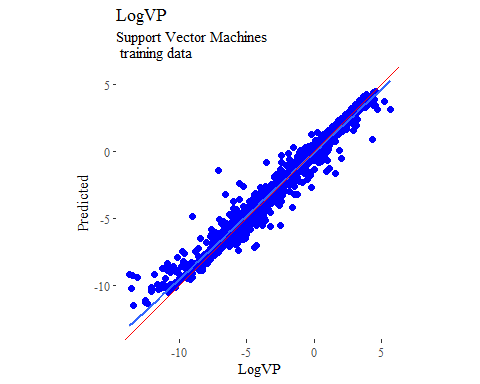
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogVP',  
 subtitle = 'Support Vector Machines\n test data') +  
 ggthemes::theme\_tufte()  
p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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.1012 -0.2771 0.0138 0.2717 5.4050   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.101353 0.014411 -7.033 2.75e-12 \*\*\*  
## LogVP 0.941017 0.003519 267.425 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5653 on 2022 degrees of freedom  
## Multiple R-squared: 0.9725, Adjusted R-squared: 0.9725   
## F-statistic: 7.152e+04 on 1 and 2022 DF, p-value: < 2.2e-16

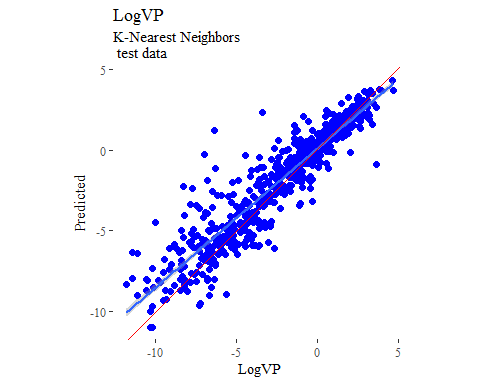
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogVP',  
 subtitle = 'Support Vector Machines\n training data') +  
 ggthemes::theme\_tufte()  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



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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.1741 -0.6096 -0.0224 0.5136 6.5919   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.08479 0.05388 1.574 0.116   
## LogVP 0.85913 0.01298 66.164 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.201 on 675 degrees of freedom  
## Multiple R-squared: 0.8664, Adjusted R-squared: 0.8662   
## F-statistic: 4378 on 1 and 675 DF, p-value: < 2.2e-16

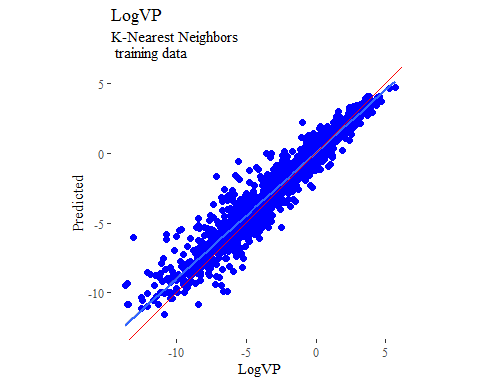
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'LogVP',  
 subtitle = 'K-Nearest Neighbors\n test data') +  
 ggthemes::theme\_tufte()  
p <- p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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.1600 -0.3891 -0.0050 0.3362 5.7431   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.027754 0.020726 1.339 0.181   
## LogVP 0.903660 0.005061 178.558 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.813 on 2022 degrees of freedom  
## Multiple R-squared: 0.9404, Adjusted R-squared: 0.9403   
## F-statistic: 3.188e+04 on 1 and 2022 DF, p-value: < 2.2e-16

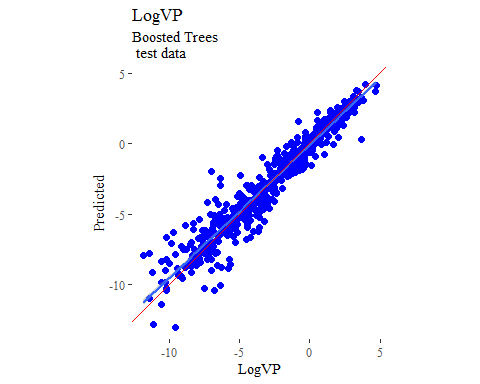
p <-  
 ggplot(data2plot, aes(LogVP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'LogVP',  
 subtitle = 'K-Nearest Neighbors\n training data') +  
 ggthemes::theme\_tufte()  
p <-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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.9678 -0.3687 0.0722 0.3308 4.7247   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.088907 0.040600 -2.19 0.0289 \*   
## LogVP 0.946999 0.009784 96.79 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9048 on 675 degrees of freedom  
## Multiple R-squared: 0.9328, Adjusted R-squared: 0.9327   
## F-statistic: 9368 on 1 and 675 DF, p-value: < 2.2e-16

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

##   
## Call:  
## lm(formula = Predicted ~ LogVP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.67278 -0.11043 0.00163 0.11028 2.96820   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.012598 0.005086 -2.477 0.0133 \*   
## LogVP 0.993772 0.001242 800.207 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1995 on 2022 degrees of freedom  
## Multiple R-squared: 0.9969, Adjusted R-squared: 0.9969   
## F-statistic: 6.403e+05 on 1 and 2022 DF, p-value: < 2.2e-16

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

