MP.R

US16120

Wed Sep 26 18:39:43 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

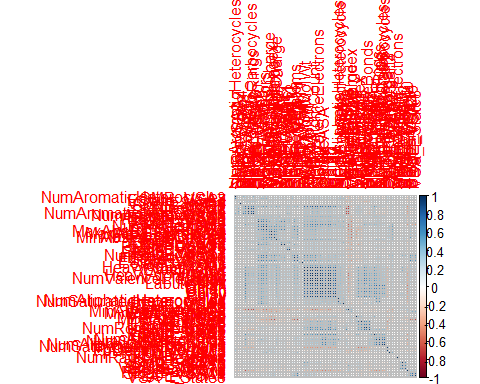
##   
## Attaching package: 'caret'

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

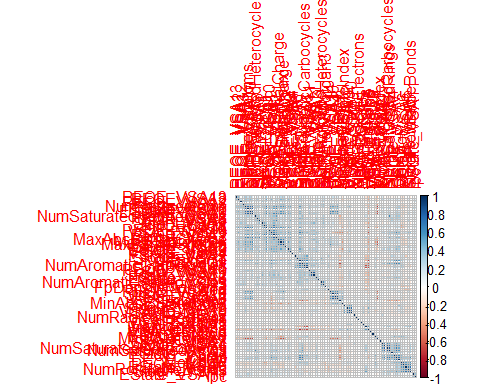
library(corrplot)

## corrplot 0.84 loaded

library(ggplot2)  
library(ggthemes)  
  
# read data  
  
## training data  
train <-  
 read.csv('cache/TR\_MP\_6486\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(MP, everything()) %>%  
 na.omit()  
  
X\_train <- train %>%  
 select(-MP)  
y\_train <- train %>%  
 select(MP) %>%  
 data.frame()  
  
## test data  
test <-  
 read.csv('cache/TST\_MP\_2167\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(MP, everything()) %>%  
 na.omit()  
  
X\_test <- test %>%  
 select(-MP)  
y\_test <- test %>%  
 select(MP) %>%  
 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(MP ~ .,  
 data = trainSet,  
 method = 'lm',  
 trControl = fitControl)

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

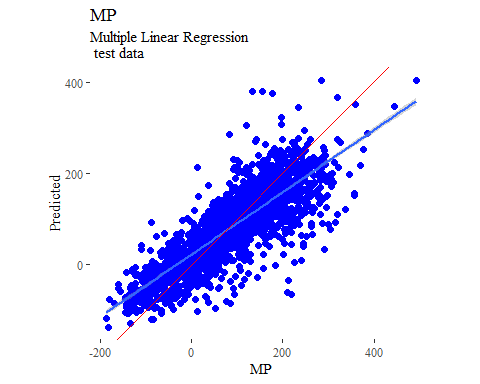
y\_predict <- predict(mlr, newdata = X\_testTransformed) %>%  
 data.frame()

## 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 ~ MP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -237.283 -24.886 -0.232 27.047 265.198   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 22.810138 1.307485 17.45 <2e-16 \*\*\*  
## MP 0.682564 0.009958 68.54 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 47.12 on 2160 degrees of freedom  
## Multiple R-squared: 0.685, Adjusted R-squared: 0.6849   
## F-statistic: 4698 on 1 and 2160 DF, p-value: < 2.2e-16

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



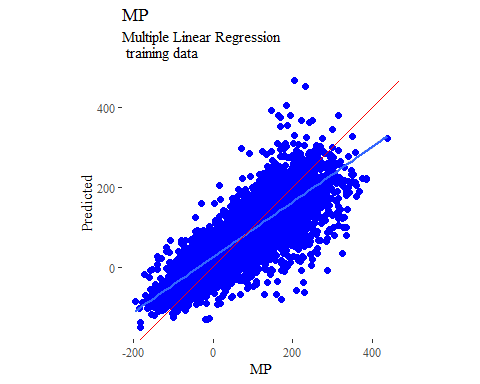
y\_predict <- predict(mlr, newdata = X\_trainTransformed) %>%  
 data.frame()

## 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 ~ MP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -247.791 -24.669 0.551 25.588 304.238   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 24.434051 0.726710 33.62 <2e-16 \*\*\*  
## MP 0.693066 0.005738 120.78 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 45.44 on 6461 degrees of freedom  
## Multiple R-squared: 0.6931, Adjusted R-squared: 0.693   
## F-statistic: 1.459e+04 on 1 and 6461 DF, p-value: < 2.2e-16

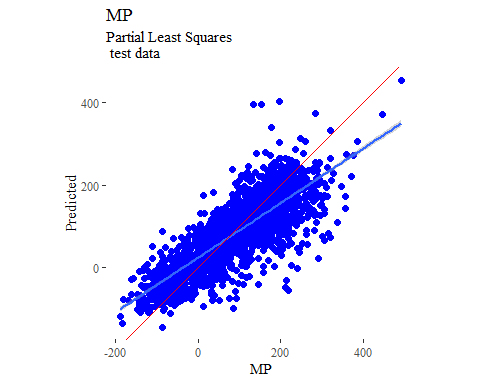
p <-  
 ggplot(data2plot, aes(MP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'MP',  
 subtitle = 'Multiple Linear Regression\n training data') +  
 ggthemes::theme\_tufte()  
p 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



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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -223.057 -25.984 0.465 28.235 281.630   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 24.26621 1.32302 18.34 <2e-16 \*\*\*  
## MP 0.66096 0.01008 65.59 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 47.68 on 2160 degrees of freedom  
## Multiple R-squared: 0.6658, Adjusted R-squared: 0.6656   
## F-statistic: 4302 on 1 and 2160 DF, p-value: < 2.2e-16

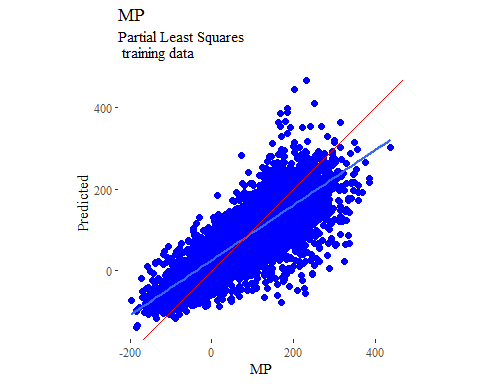
p <-  
 ggplot(data2plot, aes(MP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'MP',  
 subtitle = 'Partial Least Squares\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -233.993 -25.453 0.724 27.796 285.194   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 25.956505 0.738602 35.14 <2e-16 \*\*\*  
## MP 0.673941 0.005832 115.56 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 46.18 on 6461 degrees of freedom  
## Multiple R-squared: 0.6739, Adjusted R-squared: 0.6739   
## F-statistic: 1.335e+04 on 1 and 6461 DF, p-value: < 2.2e-16

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



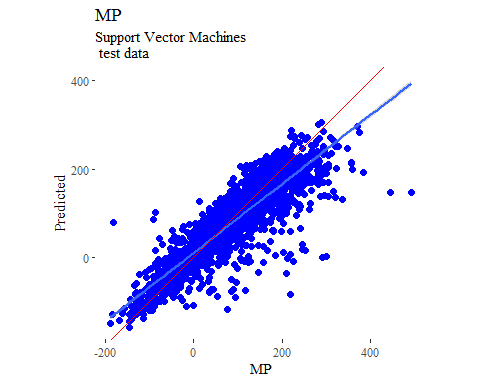
## support vector machines  
  
svmModel <- train(  
 MP ~ .,  
 data = trainSet,  
 method = 'svmRadial',  
 # tuneLength = 14,  
 trControl = fitControl  
)

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -264.230 -21.657 1.391 24.500 210.444   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.393030 1.154808 9.00 <2e-16 \*\*\*  
## MP 0.779212 0.008796 88.59 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 41.62 on 2160 degrees of freedom  
## Multiple R-squared: 0.7842, Adjusted R-squared: 0.7841   
## F-statistic: 7848 on 1 and 2160 DF, p-value: < 2.2e-16

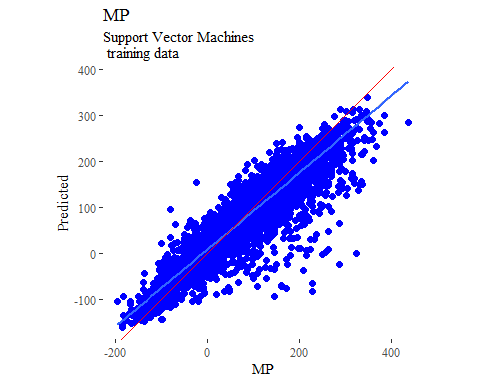
p <-  
 ggplot(data2plot, aes(MP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'MP',  
 subtitle = 'Support Vector Machines\n test data') +  
 ggthemes::theme\_tufte()  
p <- p +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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -282.815 -14.524 3.824 19.135 166.381   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 8.673430 0.534371 16.23 <2e-16 \*\*\*  
## MP 0.835679 0.004219 198.06 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 33.41 on 6461 degrees of freedom  
## Multiple R-squared: 0.8586, Adjusted R-squared: 0.8586   
## F-statistic: 3.923e+04 on 1 and 6461 DF, p-value: < 2.2e-16

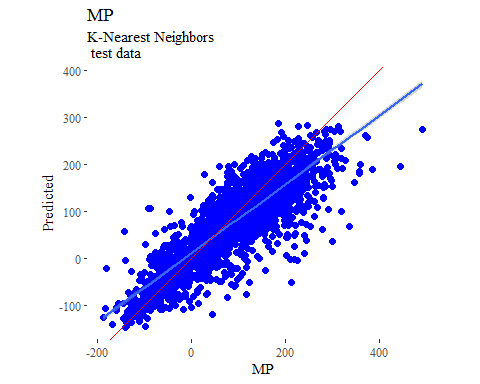
p <-  
 ggplot(data2plot, aes(MP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'MP',  
 subtitle = 'Support Vector Machines\n training data') +  
 ggthemes::theme\_tufte()  
p <- p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



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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -218.510 -26.394 0.306 30.463 161.928   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.814327 1.273870 8.489 <2e-16 \*\*\*  
## MP 0.732924 0.009702 75.540 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 45.91 on 2160 degrees of freedom  
## Multiple R-squared: 0.7254, Adjusted R-squared: 0.7253   
## F-statistic: 5706 on 1 and 2160 DF, p-value: < 2.2e-16

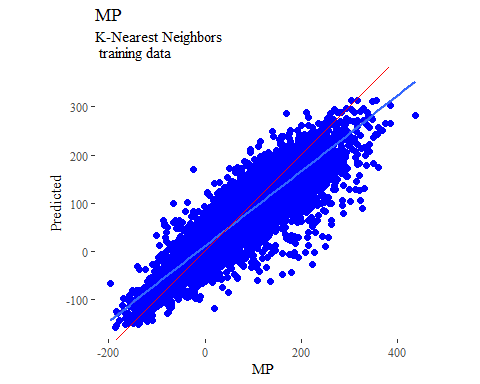
p <-  
 ggplot(data2plot, aes(MP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'MP',  
 subtitle = 'K-Nearest Neighbors\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -211.405 -22.829 0.315 24.462 178.483   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.238868 0.619645 16.52 <2e-16 \*\*\*  
## MP 0.782210 0.004893 159.88 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 38.74 on 6461 degrees of freedom  
## Multiple R-squared: 0.7982, Adjusted R-squared: 0.7982   
## F-statistic: 2.556e+04 on 1 and 6461 DF, p-value: < 2.2e-16

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



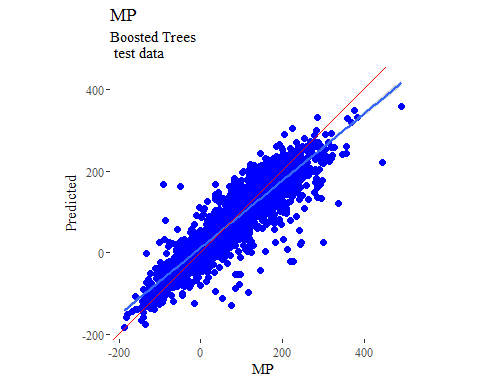
## boosted trees  
  
gbmGrid <- expand.grid(  
 .interaction.depth = seq(1, 7, by = 2),  
 .n.trees = seq(100, 1000, by = 50),  
 .shrinkage = c(0.001, 0.1),  
 .n.minobsinnode = 3  
)  
  
treeModel <- train(  
 MP ~ .,  
 data = trainSet,  
 method = 'gbm',  
 tuneGrid = gbmGrid,  
 verbose = FALSE  
)

## Warning in (function (x, y, offset = NULL, misc = NULL, distribution =  
## "bernoulli", : variable 69: VSA\_EState1 has no variation.

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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -234.763 -21.520 0.474 24.255 229.900   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 12.475078 1.118703 11.15 <2e-16 \*\*\*  
## MP 0.819472 0.008521 96.17 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 40.31 on 2160 degrees of freedom  
## Multiple R-squared: 0.8107, Adjusted R-squared: 0.8106   
## F-statistic: 9250 on 1 and 2160 DF, p-value: < 2.2e-16

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

##   
## Call:  
## lm(formula = Predicted ~ MP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -160.497 -13.416 -0.079 13.767 144.472   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.829071 0.349053 19.57 <2e-16 \*\*\*  
## MP 0.914304 0.002756 331.74 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 21.82 on 6461 degrees of freedom  
## Multiple R-squared: 0.9445, Adjusted R-squared: 0.9445   
## F-statistic: 1.101e+05 on 1 and 6461 DF, p-value: < 2.2e-16

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

