BP.R

US16120

Wed Sep 26 14:43:57 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

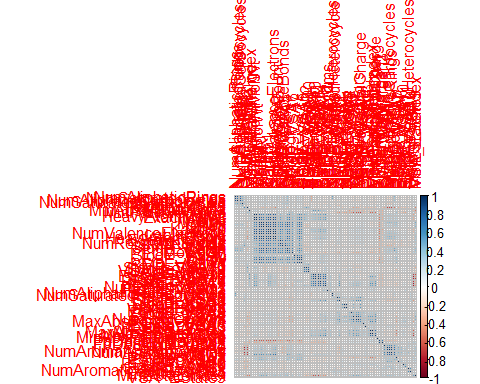
##   
## Attaching package: 'caret'

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

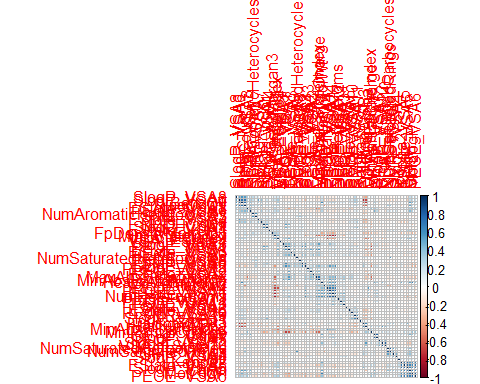
library(corrplot)

## corrplot 0.84 loaded

library(ggplot2)  
library(ggthemes)  
  
# read data  
  
## training data  
train <-  
 read.csv('cache/TR\_BP\_4077\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(BP, everything()) %>%  
 na.omit()  
  
X\_train <- train %>%  
 select(-BP)  
y\_train <- train %>%  
 select(BP) %>%  
 data.frame()  
  
## test data  
test <-  
 read.csv('cache/TST\_BP\_1358\_descrs.csv',  
 header = TRUE,  
 stringsAsFactors = FALSE) %>%  
 select(-X,-CAS,-ROMol,-SMILES,-ID) %>%  
 select(BP, everything()) %>%  
 na.omit()  
  
X\_test <- test %>%  
 select(-BP)  
y\_test <- test %>%  
 select(BP) %>%  
 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(BP ~ .,  
 data = trainSet,  
 method = 'lm',  
 trControl = fitControl)

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

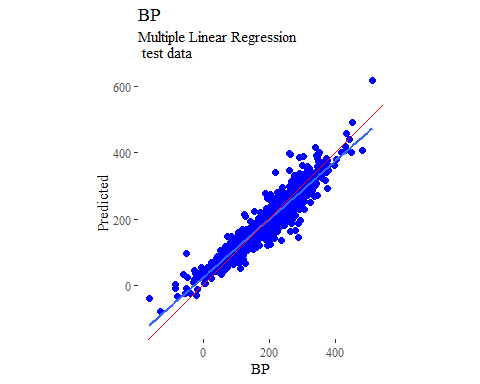
y\_predict <- predict(mlr, newdata = X\_testTransformed) %>%  
 data.frame()

## 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 ~ BP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -132.414 -12.536 -0.187 12.471 146.693   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 22.183004 1.666098 13.31 <2e-16 \*\*\*  
## BP 0.883083 0.008101 109.01 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 25.17 on 1351 degrees of freedom  
## Multiple R-squared: 0.8979, Adjusted R-squared: 0.8978   
## F-statistic: 1.188e+04 on 1 and 1351 DF, p-value: < 2.2e-16

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



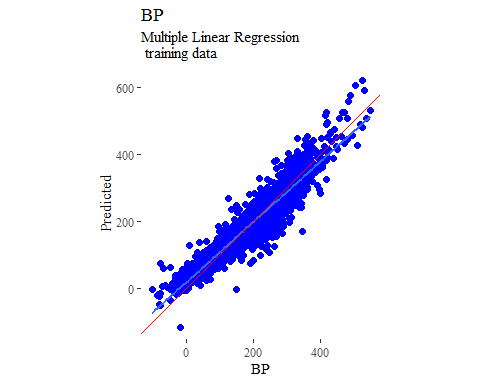
y\_predict <- predict(mlr, newdata = X\_trainTransformed) %>%  
 data.frame()

## 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 ~ BP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -162.062 -12.041 -0.099 11.649 136.658   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 18.744198 0.973748 19.25 <2e-16 \*\*\*  
## BP 0.900994 0.004687 192.22 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 25.55 on 4060 degrees of freedom  
## Multiple R-squared: 0.901, Adjusted R-squared: 0.901   
## F-statistic: 3.695e+04 on 1 and 4060 DF, p-value: < 2.2e-16

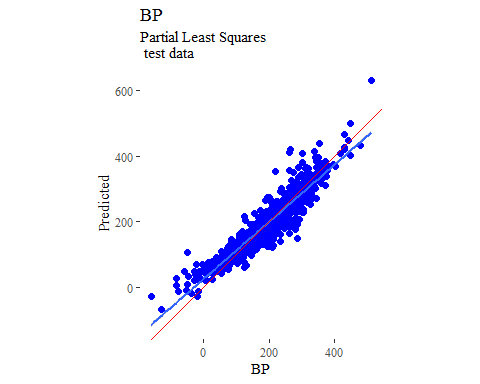
p <-  
 ggplot(data2plot, aes(BP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'BP',  
 subtitle = 'Multiple Linear Regression\n training data') +  
 ggthemes::theme\_tufte()  
p 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



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

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -127.794 -13.425 -0.633 12.140 166.323   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 24.388656 1.740113 14.02 <2e-16 \*\*\*  
## BP 0.873670 0.008461 103.26 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 26.29 on 1351 degrees of freedom  
## Multiple R-squared: 0.8875, Adjusted R-squared: 0.8875   
## F-statistic: 1.066e+04 on 1 and 1351 DF, p-value: < 2.2e-16

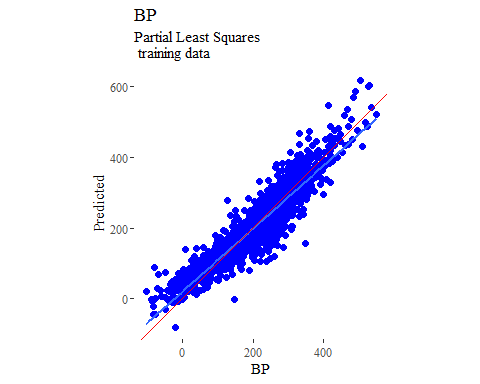
p <-  
 ggplot(data2plot, aes(BP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'BP',  
 subtitle = 'Partial Least Squares\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -174.49 -13.47 -0.57 12.22 158.92   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 20.422002 1.011384 20.19 <2e-16 \*\*\*  
## BP 0.892132 0.004869 183.24 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 26.53 on 4060 degrees of freedom  
## Multiple R-squared: 0.8921, Adjusted R-squared: 0.8921   
## F-statistic: 3.358e+04 on 1 and 4060 DF, p-value: < 2.2e-16

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



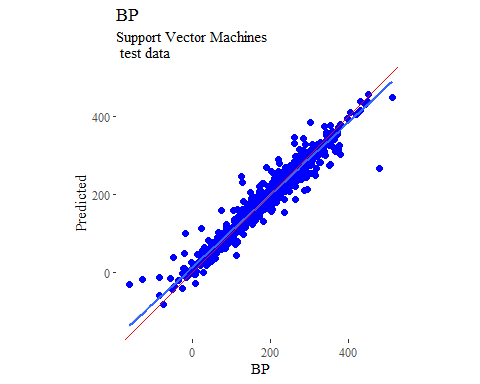
## support vector machines  
  
svmModel <- train(  
 BP ~ .,  
 data = trainSet,  
 method = 'svmRadial',  
 # tuneLength = 14,  
 trControl = fitControl  
)

## 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 ~ BP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -194.501 -7.900 -0.416 7.235 115.913   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 14.262654 1.208559 11.8 <2e-16 \*\*\*  
## BP 0.929447 0.005876 158.2 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 18.26 on 1351 degrees of freedom  
## Multiple R-squared: 0.9488, Adjusted R-squared: 0.9487   
## F-statistic: 2.502e+04 on 1 and 1351 DF, p-value: < 2.2e-16

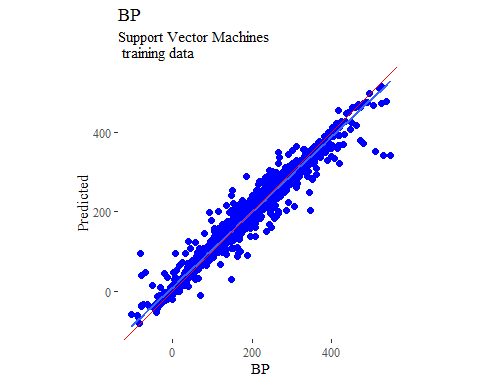
p <-  
 ggplot(data2plot, aes(BP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'BP',  
 subtitle = 'Support Vector Machines\n test data') +  
 ggthemes::theme\_tufte()  
p <- p +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

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -187.641 -5.871 -0.011 6.244 162.900   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.229446 0.572210 17.88 <2e-16 \*\*\*  
## BP 0.945770 0.002754 343.36 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 15.01 on 4060 degrees of freedom  
## Multiple R-squared: 0.9667, Adjusted R-squared: 0.9667   
## F-statistic: 1.179e+05 on 1 and 4060 DF, p-value: < 2.2e-16

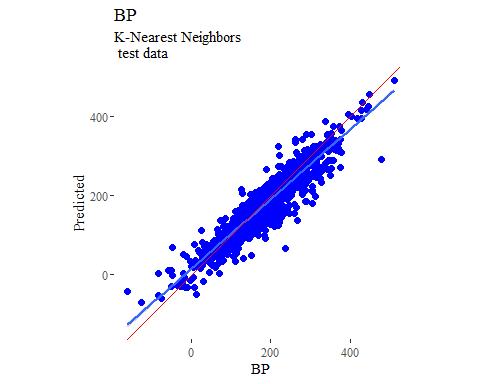
p <-  
 ggplot(data2plot, aes(BP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method='lm') +  
 labs(title = 'BP',  
 subtitle = 'Support Vector Machines\n training data') +  
 ggthemes::theme\_tufte()  
p <- p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



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

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -156.560 -12.730 2.592 15.913 114.295   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 16.1178 1.7893 9.008 <2e-16 \*\*\*  
## BP 0.8784 0.0087 100.965 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 27.03 on 1351 degrees of freedom  
## Multiple R-squared: 0.883, Adjusted R-squared: 0.8829   
## F-statistic: 1.019e+04 on 1 and 1351 DF, p-value: < 2.2e-16

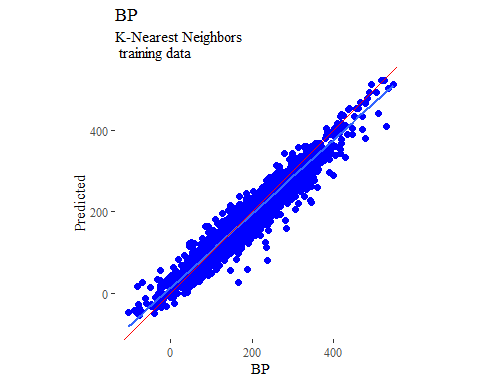
p <-  
 ggplot(data2plot, aes(BP, Predicted)) +  
 geom\_point(colour = "blue", size = 2) +  
 coord\_equal() +  
 # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +  
 geom\_smooth(method = 'lm') +  
 labs(title = 'BP',  
 subtitle = 'K-Nearest Neighbors\n test data') +  
 ggthemes::theme\_tufte()  
p <- p + geom\_abline(intercept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

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -149.590 -9.893 1.905 11.330 78.416   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 12.811690 0.750293 17.08 <2e-16 \*\*\*  
## BP 0.909433 0.003612 251.80 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 19.68 on 4060 degrees of freedom  
## Multiple R-squared: 0.9398, Adjusted R-squared: 0.9398   
## F-statistic: 6.34e+04 on 1 and 4060 DF, p-value: < 2.2e-16

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



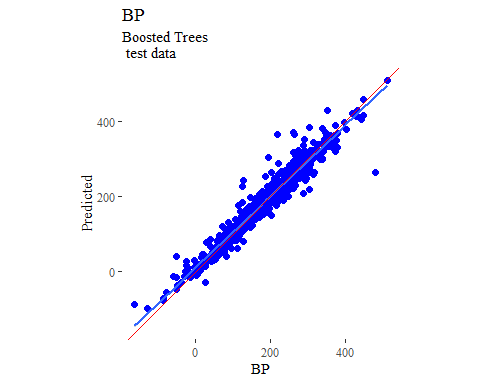
## boosted trees  
  
gbmGrid <- expand.grid(  
 .interaction.depth = seq(1, 7, by = 2),  
 .n.trees = seq(100, 1000, by = 50),  
 .shrinkage = c(0.001, 0.1),  
 .n.minobsinnode = 3  
)  
  
treeModel <- train(  
 BP ~ .,  
 data = trainSet,  
 method = 'gbm',  
 tuneGrid = gbmGrid,  
 verbose = FALSE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

## Warning in (function (x, y, offset = NULL, misc = NULL, distribution =  
## "bernoulli", : variable 64: VSA\_EState4 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 ~ BP, data = data2plot))

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -201.123 -7.967 -1.033 6.637 146.623   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.302903 1.174479 8.772 <2e-16 \*\*\*  
## BP 0.951150 0.005711 166.556 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 17.74 on 1351 degrees of freedom  
## Multiple R-squared: 0.9536, Adjusted R-squared: 0.9535   
## F-statistic: 2.774e+04 on 1 and 1351 DF, p-value: < 2.2e-16

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

##   
## Call:  
## lm(formula = Predicted ~ BP, data = data2plot)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -26.4498 -3.5648 -0.0827 3.4805 23.8880   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.66852 0.20974 7.955 2.3e-15 \*\*\*  
## BP 0.99120 0.00101 981.757 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5.502 on 4060 degrees of freedom  
## Multiple R-squared: 0.9958, Adjusted R-squared: 0.9958   
## F-statistic: 9.638e+05 on 1 and 4060 DF, p-value: < 2.2e-16

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

