ML: Daphnia

. . . narrative ground truth . . .

2019-09-27

```r
path_to_data <- here('data', 'Khan_Chemosphere_229_8.xlsx')

df <-
  readxl::read_xlsx(path_to_data, sheet = 'Daphnia Dataset')  %>%
  data.frame() %>%
  janitor::clean_names()

names(df)
```

```
## [1] "id_biocides"           "cas_number"            "canonical_smiles"
## [4] "x4"                    "exp_log_ec50_mmol_l"  "geom_mean_value_mmol"
## [7] "mol_l"                 "p_ec50_mol_l_daphnia"
```

```r
df <- df %>%
  select(cas_number, canonical_smiles, p_ec50_mol_l_daphnia) %>%
  rename('CAS' = 'cas_number',
         'SMILES' = 'canonical_smiles',
         'pEC50' = 'p_ec50_mol_l_daphnia')

head(df)
```

```
##           CAS
## 1    112-53-8
## 2     94-75-7
## 3  57960-19-7
## 4 101007-06-1
## 5 348635-87-0
## 6 120162-55-2
##                                                               SMILES
## 1                                                       OCCCCCCCCCCCC
## 2                                                  O=C(O)COc1ccc(cc1Cl)Cl
## 3                              O=C(OC=1C(=O)c2ccccc2(C(=O)C=1CCCCCCCCCCCC))C
## 4 N#CC(OC(=O)C1C(C=CC(=O)OC(C(F)(F)F)C(F)(F)F)C1(C)(C))c3cccc(Oc2ccccc2)c3
## 5                        O=S(=O)(c1ncn(n1)S(=O)(=O)N(C)C)n3c2cc(F)ccc2c(c3C)Br
## 6                         O=C(Nc1nc(OC)cc(n1)OC)NS(=O)(=O)c2c(cnn2C)c3nnn(n3)C
##        pEC50
## 1  2.765249
## 2  3.216718
## 3  7.993900
## 4 10.391160
## 5  7.100510
## 6  3.050345
```

```
import numpy as np
import pandas as pd
from rdkit import Chem
from rdkit.Chem import Descriptors
from rdkit.ML.Descriptors import MoleculeDescriptors
from scipy import stats

df = r.df
df.head()
```

```
##            CAS                                               SMILES      pEC50
## 0     112-53-8                                        OCCCCCCCCCCCC   2.765249
## 1      94-75-7                                 O=C(O)COc1ccc(cc1Cl)Cl   3.216718
## 2   57960-19-7      O=C(OC=1C(=O)c2ccccc2(C(=O)C=1CCCCCCCCCCCC))C   7.993900
## 3  101007-06-1  N#CC(OC(=O)C1C(C=CC(=O)OC(C(F)(F)F)C(F)(F)F)C1...  10.391160
## 4  348635-87-0  O=S(=O)(c1ncn(n1)S(=O)(=O)N(C)C)n3c2cc(F)ccc2c...   7.100510
```

```
nms = [x[0] for x in Descriptors._descList]
calc = MoleculeDescriptors.MolecularDescriptorCalculator(nms)
#for i in range(5):
for i in range(len(df)):
    try:
        descrs = calc.CalcDescriptors(Chem.MolFromSmiles(df.iloc[i, 1]))
        for x in range(len(descrs)):
            df.at[i, str(nms[x])] = descrs[x]
    except:
        for x in range(len(descrs)):
            df.at[i, str(nms[x])] = 'NaN'

df = df.replace([np.inf, -np.inf], np.nan)
df = df.dropna()
df = df.reset_index(drop=True)

df.head()
```

```
##            CAS  ... fr_urea
## 0     112-53-8  ...     0.0
## 1      94-75-7  ...     0.0
## 2   57960-19-7  ...     0.0
## 3  101007-06-1  ...     0.0
## 4  348635-87-0  ...     0.0
##
## [5 rows x 203 columns]
```

```
df.shape
```

```
## (132, 203)
```

```
df <- py$df
dim(df)
```

```
## [1] 132 203
```

```
in_train <-
  createDataPartition(df$pEC50
                      , p = 0.8
                      , list = FALSE)
train <- df[in_train,] %>%
  mutate(set = 'train') %>%
  data.frame()
test <- df[-in_train,] %>%
  mutate(set = 'test') %>%
  data.frame()
```

```
X_train <- train %>%
  select(-CAS, -SMILES, -pEC50, -set) %>%
  data.frame()
dim(X_train)
```

```
## [1] 108 200
```

```
X_test <- test %>%
  select(-CAS, -SMILES, -pEC50, -set) %>%
  data.frame()
dim(X_test)
```

```
## [1]  24 200
```

```
y_train <- train %>%
  select(pEC50) %>%
  data.frame()
colnames(y_train) <- c('Observed')
dim(y_train)
```

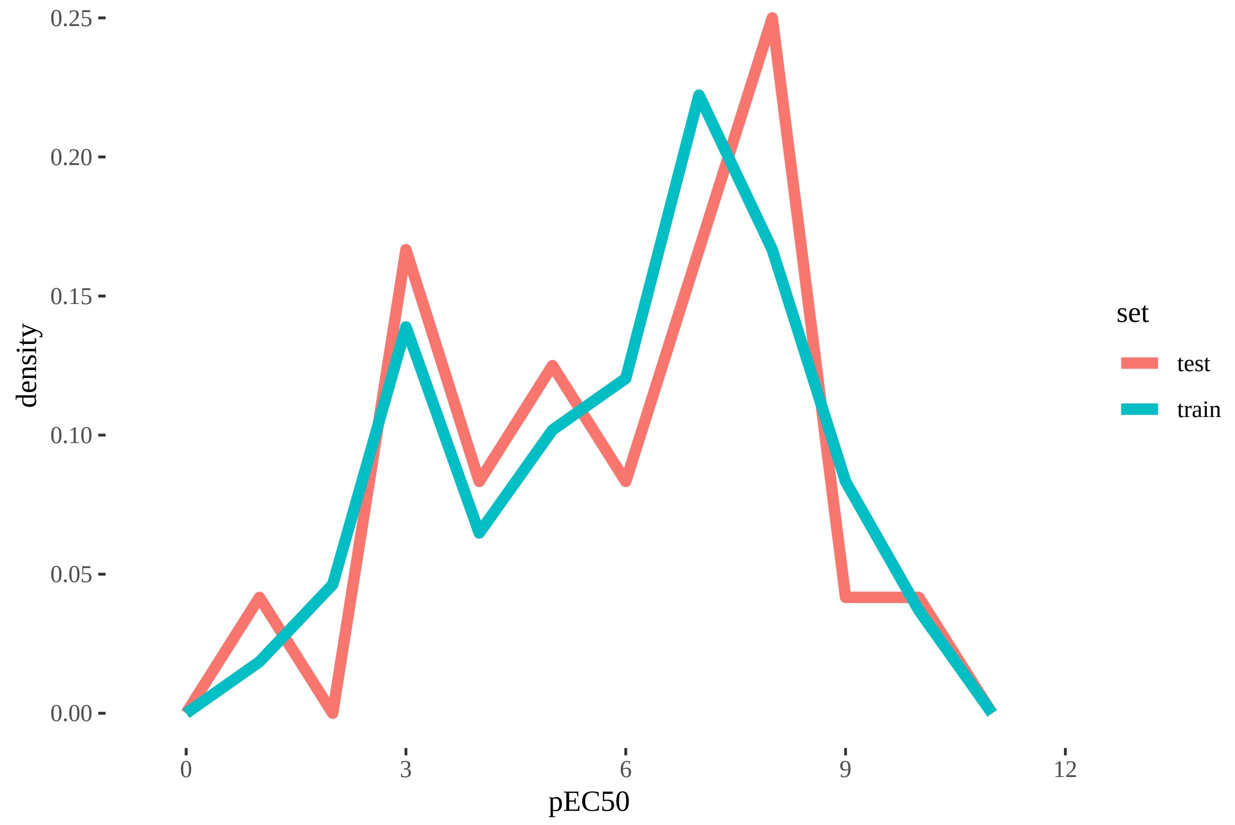
```
## [1] 108   1
```

```
y_test <- test %>%
  select(pEC50) %>%
  data.frame()
dim(y_test)
```

```
## [1] 24  1
```

```
colnames(y_test) <- c('Observed')
```

```
data2plot <- rbind(train, test) %>%
  data.frame()
daphnia_train_test <-
  ggplot(data2plot, aes(pEC50, stat(density), colour = set)) +
  geom_freqpoly(binwidth = 1.0, size = 2) +
  ggthemes::theme_tufte()
daphnia_train_test
```



```
ggsave('daphnia_train_test.png', daphnia_train_test, width = 4.0, height = 2.5, units = 'in')
```

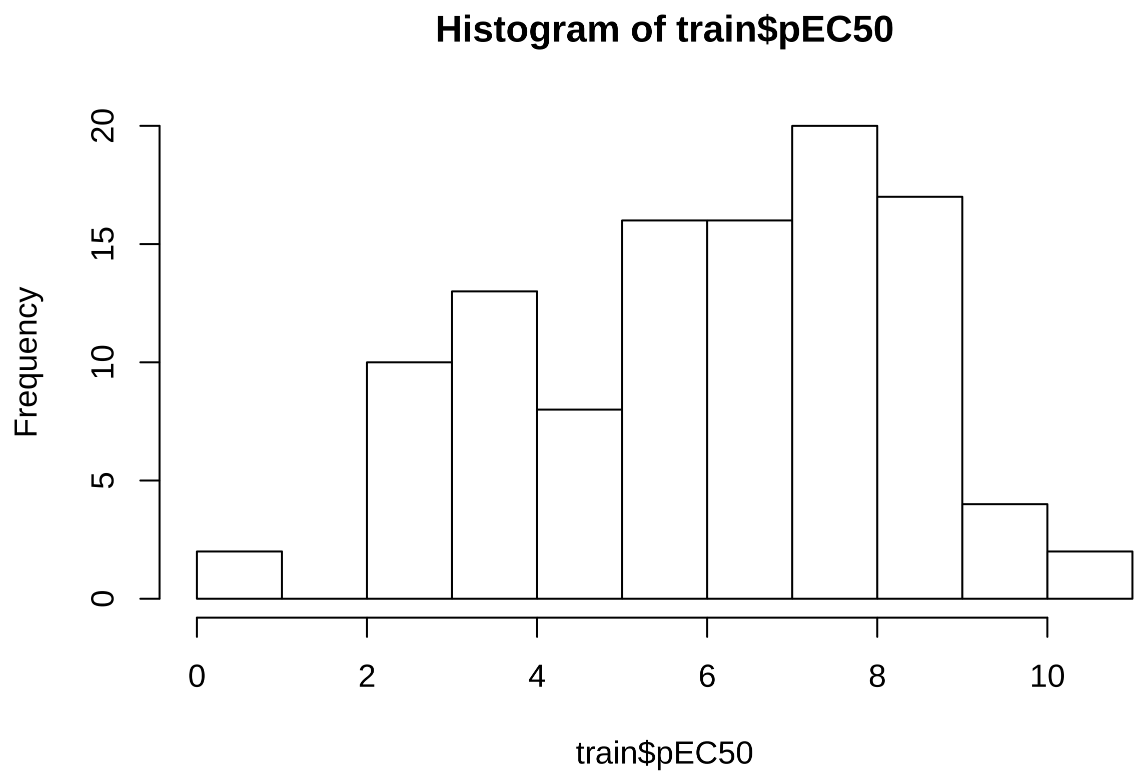
```
dim(train)
```

```
## [1] 108 204
```

```
summary(train$pEC50)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.7746  4.1041  6.5552  6.0352  7.6900 10.3912
```

```
hist(train$pEC50)
```



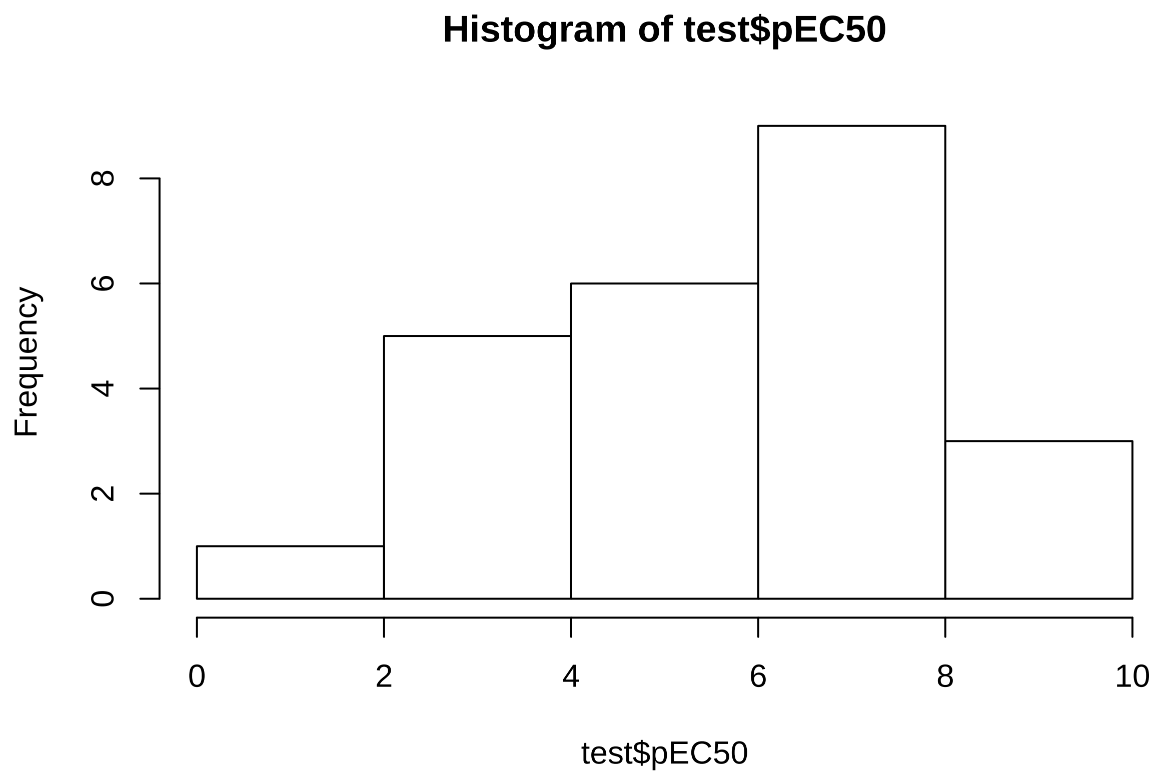
```r
dim(test)
```

```
## [1]  24 204
```

```r
summary(test$pEC50)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.6582  4.1246  6.3694  5.9015  7.6793  9.6029
```

```r
hist(test$pEC50)
```



```
nzv <- nearZeroVar(X_train, freqCut = 100 / 0)
names(X_train[ , nzv])
```

```
##  [1] "NumRadicalElectrons" "SMR_VSA8"            "SlogP_VSA9"
##  [4] "EState_VSA11"        "VSA_EState1"         "VSA_EState2"
##  [7] "VSA_EState3"         "VSA_EState4"         "VSA_EState5"
## [10] "VSA_EState6"         "VSA_EState7"         "fr_HOCCN"
## [13] "fr_N_O"              "fr_Ndealkylation2"   "fr_SH"
## [16] "fr_amidine"          "fr_azide"            "fr_azo"
## [19] "fr_barbitur"         "fr_benzodiazepine"   "fr_diazo"
## [22] "fr_dihydropyridine"  "fr_furan"            "fr_guanido"
## [25] "fr_hdrzone"          "fr_isocyan"          "fr_lactam"
## [28] "fr_nitroso"          "fr_phos_acid"        "fr_phos_ester"
## [31] "fr_piperdine"        "fr_piperzine"        "fr_prisulfonamd"
## [34] "fr_thiophene"
```

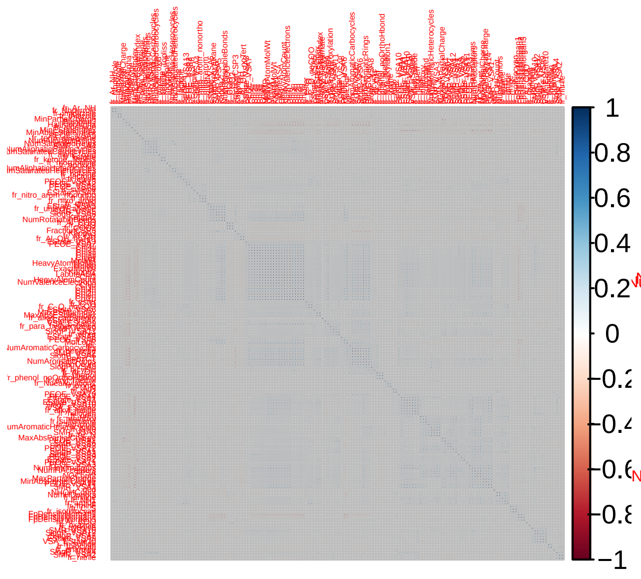
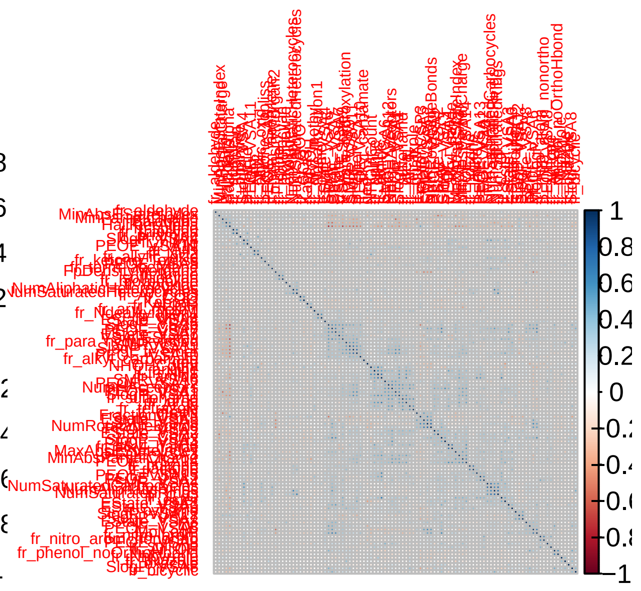
```
X_train_nzv <- X_train[ , -nzv]
X_test_nzv <- X_test[ , -nzv]
```

```
par(mfrow=c(1,2))

correlations <- cor(X_train_nzv)
corrplot(correlations, order = "hclust", tl.cex = 0.25)
```

```r
highCorr <- findCorrelation(correlations, cutoff = 0.8)
names(X_train_nzv[ , highCorr])
```

```
##  [1] "ExactMolWt"                "NumValenceElectrons"
##  [3] "MaxAbsPartialCharge"       "FpDensityMorgan3"
##  [5] "Chi0"                      "Chi0n"
##  [7] "Chi0v"                     "Chi1"
##  [9] "Kappa1"                    "LabuteASA"
## [11] "SMR_VSA1"                  "SlogP_VSA2"
## [13] "SlogP_VSA5"                "EState_VSA1"
## [15] "EState_VSA10"              "VSA_EState10"
## [17] "HeavyAtomCount"            "NOCount"
## [19] "NumAromaticCarbocycles"    "NumAromaticRings"
## [21] "NumHDonors"                "NumHeteroatoms"
## [23] "RingCount"                 "MolMR"
## [25] "fr_Al_OH_noTert"           "fr_COO2"
## [27] "fr_C_O_noCOO"              "fr_NHO"
## [29] "fr_benzene"                "fr_halogen"
## [31] "fr_unbrch_alkane"          "MaxEStateIndex"
## [33] "MolWt"                     "MaxPartialCharge"
## [35] "FpDensityMorgan1"          "HeavyAtomMolWt"
## [37] "Chi1n"                     "Chi1v"
## [39] "Chi2n"                     "Chi2v"
## [41] "Chi3n"                     "Chi3v"
## [43] "BertzCT"                   "SMR_VSA10"
## [45] "SMR_VSA7"                  "EState_VSA9"
## [47] "MinEStateIndex"            "SlogP_VSA10"
## [49] "TPSA"                      "Kappa2"
## [51] "NumAliphaticCarbocycles"   "SMR_VSA3"
## [53] "NumAromaticHeterocycles"   "fr_Al_COO"
## [55] "fr_Ar_NH"                  "VSA_EState8"
## [57] "fr_C_S"                    "fr_ketone"
## [59] "SMR_VSA2"                  "fr_nitro"
## [61] "fr_phenol"                 "fr_methoxy"
## [63] "fr_imidazole"
```

```r
X_train_curated <- X_train_nzv[ , -highCorr]
X_test_curated <- X_test_nzv[ , -highCorr]

correlations <- cor(X_train_curated)
corrplot(correlations, order = "hclust", tl.cex = 0.5)
```

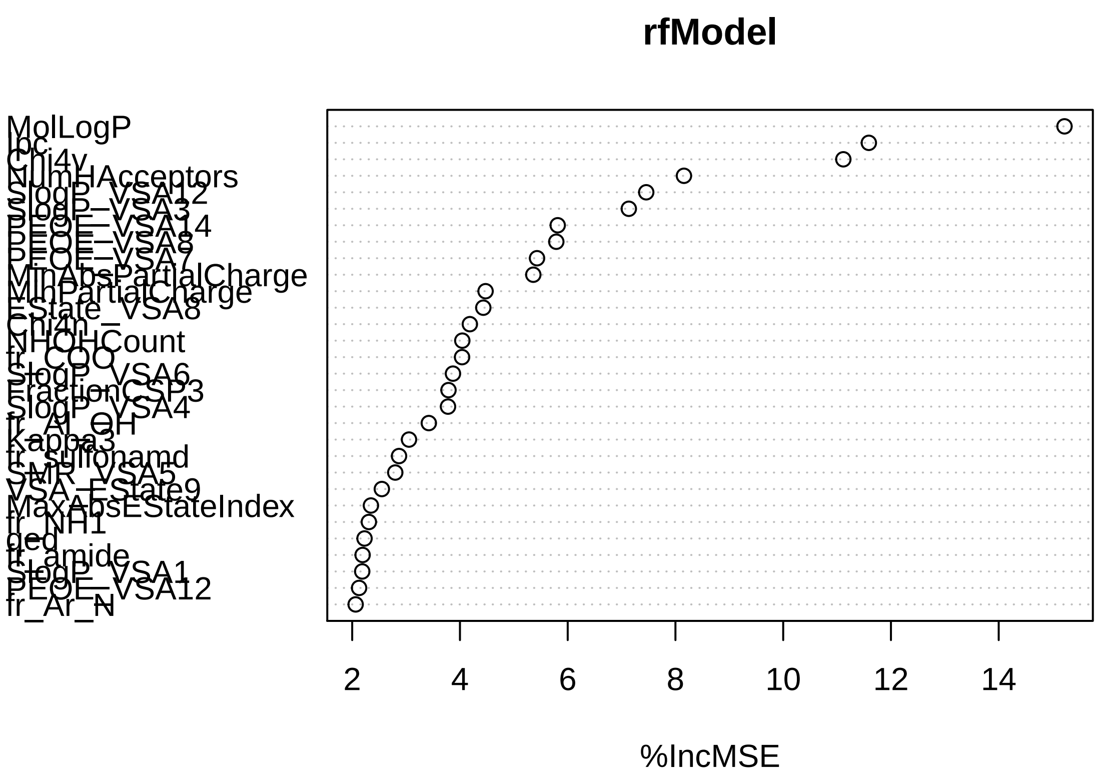
```r
names(X_train_curated)
```

```
##  [1] "MaxAbsEStateIndex"       "MinAbsEStateIndex"
##  [3] "qed"                     "MinPartialCharge"
##  [5] "MinAbsPartialCharge"     "FpDensityMorgan2"
##  [7] "BalabanJ"                "Chi4n"
##  [9] "Chi4v"                   "HallKierAlpha"
## [11] "Ipc"                     "Kappa3"
## [13] "PEOE_VSA1"               "PEOE_VSA10"
## [15] "PEOE_VSA11"              "PEOE_VSA12"
## [17] "PEOE_VSA13"              "PEOE_VSA14"
## [19] "PEOE_VSA2"               "PEOE_VSA3"
## [21] "PEOE_VSA4"               "PEOE_VSA5"
## [23] "PEOE_VSA6"               "PEOE_VSA7"
## [25] "PEOE_VSA8"               "PEOE_VSA9"
## [27] "SMR_VSA4"                "SMR_VSA5"
## [29] "SMR_VSA6"                "SMR_VSA9"
## [31] "SlogP_VSA1"              "SlogP_VSA11"
## [33] "SlogP_VSA12"             "SlogP_VSA3"
## [35] "SlogP_VSA4"              "SlogP_VSA6"
## [37] "SlogP_VSA7"              "SlogP_VSA8"
## [39] "EState_VSA2"             "EState_VSA3"
## [41] "EState_VSA4"             "EState_VSA5"
## [43] "EState_VSA6"             "EState_VSA7"
## [45] "EState_VSA8"             "VSA_EState9"
##  [47] "FractionCSP3"              "NHOHCount"
##  [49] "NumAliphaticHeterocycles" "NumAliphaticRings"
##  [51] "NumHAcceptors"             "NumRotatableBonds"
##  [53] "NumSaturatedCarbocycles"  "NumSaturatedHeterocycles"
##  [55] "NumSaturatedRings"         "MolLogP"
##  [57] "fr_Al_OH"                  "fr_ArN"
##  [59] "fr_Ar_COO"                 "fr_Ar_N"
##  [61] "fr_Ar_OH"                  "fr_COO"
##  [63] "fr_C_O"                    "fr_Imine"
##  [65] "fr_NH1"                    "fr_NH2"
##  [67] "fr_Ndealkylation1"         "fr_Nhpyrrole"
##  [69] "fr_aldehyde"               "fr_alkyl_carbamate"
##  [71] "fr_alkyl_halide"           "fr_allylic_oxid"
##  [73] "fr_amide"                  "fr_aniline"
##  [75] "fr_aryl_methyl"            "fr_bicyclic"
##  [77] "fr_epoxide"                "fr_ester"
##  [79] "fr_ether"                  "fr_hdrzine"
##  [81] "fr_imide"                  "fr_isothiocyan"
##  [83] "fr_ketone_Topliss"         "fr_lactone"
##  [85] "fr_morpholine"             "fr_nitrile"
##  [87] "fr_nitro_arom"             "fr_nitro_arom_nonortho"
##  [89] "fr_oxazole"                "fr_oxime"
##  [91] "fr_para_hydroxylation"     "fr_phenol_noOrthoHbond"
##  [93] "fr_priamide"               "fr_pyridine"
##  [95] "fr_quatN"                  "fr_sulfide"
##  [97] "fr_sulfonamd"              "fr_sulfone"
##  [99] "fr_term_acetylene"         "fr_tetrazole"
## [101] "fr_thiazole"               "fr_thiocyan"
## [103] "fr_urea"
```

```r
## Random Forest

trainSet <- cbind(y_train, X_train_curated) %>%
  rename(pEC50 = Observed)
testSet <- cbind(y_test, X_test_curated) %>%
  rename(pEC50 = Observed)
rfModel <- randomForest(
  pEC50 ~ .,
  data = trainSet,
  importance = TRUE,
  ntrees = 1000
)
print(rfModel)
```

```
##
## Call:
##  randomForest(formula = pEC50 ~ ., data = trainSet, importance = TRUE,      ntrees = 1000)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 34
##
##           Mean of squared residuals: 2.136553
##                     % Var explained: 56.64
```

```r
varImpPlot(rfModel, type = 1)
```

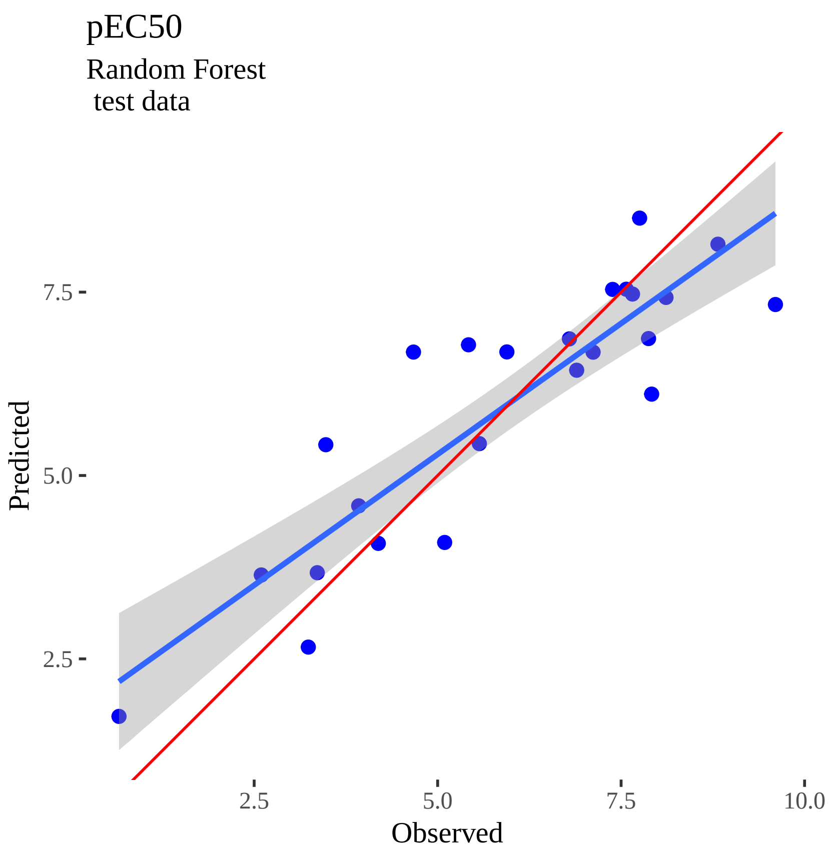


```r
y_predict <- predict(rfModel, newdata = X_test_curated) %>%
  data.frame()
colnames(y_predict) <- c('Predicted')

data2plot <- cbind(y_test, y_predict)

summary(lm(Predicted ~ Observed, data = data2plot))
```

```
##
## Call:
## lm(formula = Predicted ~ Observed, data = data2plot)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.37037 -0.47318 -0.00835  0.44915  1.62907
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.72008    0.49898   3.447   0.0023 **
## Observed     0.71377    0.07919   9.013 7.72e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.8565 on 22 degrees of freedom
## Multiple R-squared:  0.7869, Adjusted R-squared:  0.7772
## F-statistic: 81.24 on 1 and 22 DF,  p-value: 7.718e-09
```

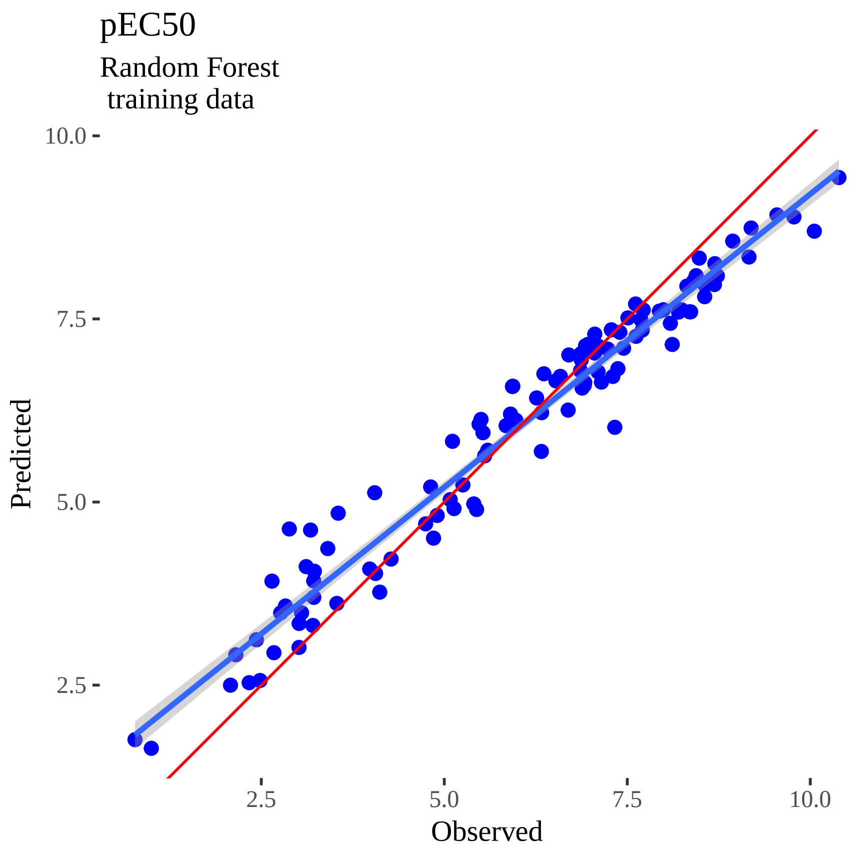
```
p <-
  ggplot(data2plot, aes(Observed, Predicted)) +
  geom_point(colour = "blue", size = 2) +
  coord_equal() +
  # xlim(c(0, 3.5)) + ylim(c(0, 3.5)) +
  geom_smooth(method = 'lm') +
  labs(title = 'pEC50',
       subtitle = 'Random Forest\n test data') +
  ggthemes::theme_tufte()
p <- p + geom_abline(intercept = 0,
                     slope = 1,
                     colour = 'red')
p
```



```
y_predict <- predict(rfModel, newdata = X_train_curated) %>%
  data.frame()
colnames(y_predict) <- c('Predicted')

data2plot <- cbind(y_train, y_predict)

summary(lm(Predicted ~ Observed, data = data2plot))
```

```
##
## Call:
## lm(formula = Predicted ~ Observed, data = data2plot)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.04680 -0.28097 -0.01612  0.24460  1.12457
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.20040    0.10569   11.36   <2e-16 ***
## Observed     0.80042    0.01644   48.70   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3791 on 106 degrees of freedom
## Multiple R-squared:  0.9572, Adjusted R-squared:  0.9568
## F-statistic:  2372 on 1 and 106 DF,  p-value: < 2.2e-16
```

```
p <-
  ggplot(data2plot, aes(Observed, 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 = 'pEC50',
       subtitle = 'Random Forest\n training data') +
  ggthemes::theme_tufte()
p <- p + geom_abline(intercept = 0,
                     slope = 1,
                     colour = 'red')
p
```


pEC50
Random Forest
training data
10.0
7.5
5.0
2.5
Predicted
2.5
5.0
7.5
10.0
Observed