Final Report

Background & Goal

Alchohols have different structures, and are used frequently in hygiene products and cosmetics. Since some of them are harmful for people, successful detection of different alchohol types is important and necessary.

In this study, five different types of alcohol are detected by five quartz crystal microbalance (QCM) sensors with different structures. The main goal of the study is to use measurement of the QCM sensors to construct a data-driven model to classify five types of alchohol. Moreover, we compare the accuracy of five QCM sensors and determine the best one.

Data Description

Introduction of data source

Five different QCM gas sensors are used, and five different gas measurements (1-octanol, 1-propanol, 2-butanol, 2- propanol and 1-isobutanol) are conducted in each of these sensors. A QCM is an electromechanical oscillator that contains a thin slice of quartz crystal with chemical receptive material placed on its surface. The measurements of frequency reduction in the oscillation are our samples. There are two different channels in these QCM sensors. One of these channel includes molecularly imprinted polymers (MIP), and the other includes nanoparticles (NP). Diverse QCM sensor structures are obtained using different MIP and NP ratios. For each type of alchohol gas, we mix it and air with different ratio, ranging from 0.799/0.201 to 0.400/0.6000. The data is available from UCI repository (https://archive.ics.uci.edu/ml/datasets/Alcohol+QCM+Sensor+Dataset#).

Data description

For each QCM detector, we have ten measurements of frequency reduction and and a five-dimension one-hot vector to show the type of achohol. The number of observations is 25 for each QCM detector. Now we briefly show several data from QCM10 dataset.

```
>>head(QCM10)
      `0.799_0.201` `0.799_0.201_1` `0.700_0.300` `0.700_0.300_1` `0.600_0.400` `0.600_0.400_1`
          <dbl>            <dbl>          <dbl>            <dbl>          <dbl>            <dbl>
1         -12.0            -11.0          -19.1            -17.3          -33.1            -28.4
2         -12.2            -11.3          -22.3            -20.0          -39.8            -33.6
3         -12.6            -11.7          -26.7            -23.3          -46.5            -38.7
4         -13.8            -12.8          -30.6            -26.2          -52.3            -43.0
5         -15.7            -13.9          -34.5            -28.6          -57.4            -46.3
6         -58.4            -38.7          -83.6            -57.3         -110.             -76.8


      `0.501_0.499` `0.501_0.499_1` `0.400_0.600` `0.400_0.600_1` `1-Octanol` `1-Propanol`
          <dbl>            <dbl>          <dbl>            <dbl>          <dbl>        <dbl>
1         -48.8            -40.8          -62.5            -50.8           1            0
2         -56.9            -46.8          -73.3            -59.0           1            0
```

```
3         -66.0          -53.5         -84.5          -67.2          1           0
4         -73.8          -59.2         -94.4          -74.4          1           0
5         -80.4          -63.5        -103.           -80.2          1           0
6        -134.           -96.1        -171.          -124.           0           1

        `2-Butanol` `2-propanol` `1-isobutanol`
           <dbl>        <dbl>          <dbl>
1            0            0              0
2            0            0              0
3            0            0              0
4            0            0              0
5            0            0              0
6            0            0              0
```

Explorative data analysis

Data preprocessing

Convert five dataset into the form of dataframe and transform the one-hot vector into label. Also some other work such as renaming columns and variable.

```r
library(dplyr)
library(readr)
library(stringr)
first_category_name = list.files("/Users/wenhongwei/Desktop/textbook/QCM_Sensor_Alcohol_Dataset")
dir = paste("/Users/wenhongwei/Desktop/textbook/QCM_Sensor_Alcohol_Dataset/",first_category_name,sep=""
for(i in 1:length(dir)){
  index = str_sub(first_category_name[i],1,-5)
  data_index = data.frame(read_csv(dir[i]))
  #print(names(data_index))
  names(data_index) = c("ch1p0.8", "ch2p0.8", "ch1p0.7","ch2p0.7","ch1p0.6","ch2p0.6","ch1p0.5","ch2p0.
  data_index = mutate(data_index,labels = Oct1*1+Pro1*2+But2*3+pro2*4+iso1*5) %>% select(-(Oct1:iso1))
  assign(index, data_index)
  #print(head(get(index)))
}
```

```r
head(QCM10)
```

```
##   ch1p0.8 ch2p0.8 ch1p0.7 ch2p0.7 ch1p0.6 ch2p0.6 ch1p0.5 ch2p0.5 ch1p0.4
## 1  -11.98  -10.99  -19.12  -17.28  -33.13  -28.45  -48.83  -40.77  -62.49
## 2  -12.15  -11.33  -22.33  -19.95  -39.82  -33.64  -56.90  -46.77  -73.32
## 3  -12.58  -11.74  -26.67  -23.34  -46.48  -38.69  -65.95  -53.46  -84.53
## 4  -13.79  -12.82  -30.56  -26.18  -52.30  -42.98  -73.81  -59.19  -94.41
## 5  -15.73  -13.87  -34.54  -28.65  -57.44  -46.26  -80.37  -63.49 -102.94
## 6  -58.41  -38.66  -83.58  -57.33 -110.24  -76.80 -133.77  -96.13 -171.05
##   ch2p0.4 labels
## 1  -50.82      1
## 2  -58.96      1
## 3  -67.21      1
## 4  -74.40      1
## 5  -80.25      1
## 6 -124.15      2
```

Data exploring

Different Ratios

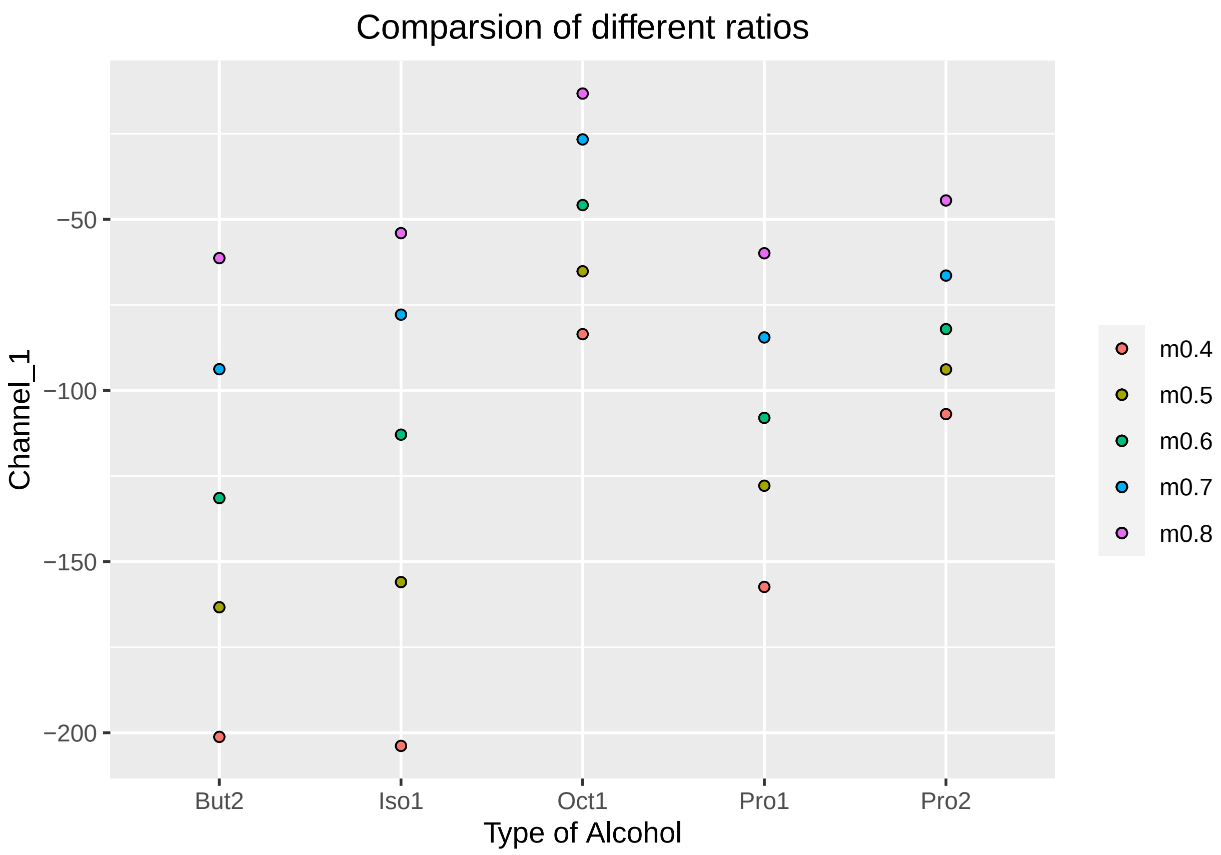
Based on measurement of QCM10, we summarize the average frequency reduction of channel 1 for five types of alchohol with different gas ratio.

```
diffratio = group_by(QCM10,labels) %>% summarize(m0.8=mean(ch1p0.8), m0.7=mean(ch1p0.7), m0.6=mean(ch1p
diffratio["labels"] = c("Oct1","Pro1","But2","Pro2","Iso1")
diffratio
```

```
## # A tibble: 5 x 6
##   labels  m0.8  m0.7   m0.6   m0.5   m0.4
##   <chr>  <dbl> <dbl>  <dbl>  <dbl>  <dbl>
## 1 Oct1   -13.2 -26.6  -45.8  -65.2  -83.5
## 2 Pro1   -59.9 -84.5 -108.  -128.  -157.
## 3 But2   -61.3 -93.8 -131.  -163.  -201.
## 4 Pro2   -44.5 -66.4  -82.1  -93.8 -107.
## 5 Iso1   -54.0 -77.8 -113.  -156.  -204.
```

Figure measurement of channel 1 in QCM10 :

```
library("ggplot2")
ggplot(diffratio)+
  geom_point(aes(x=labels,y=m0.8,fill="m0.8"),size=1.5,shape=21,color="black")+
  geom_point(aes(x=labels,y=m0.7,fill="m0.7"),size=1.5,shape=21)+
  geom_point(aes(x=labels,y=m0.6,fill="m0.6"),size=1.5,shape=21)+
  geom_point(aes(x=labels,y=m0.5,fill="m0.5"),size=1.5,shape=21)+
  geom_point(aes(x=labels,y=m0.4,fill="m0.4"),size=1.5,shape=21)+
  labs(x="Type of Alcohol",y="Channel_1",title= "Comparsion of different ratios",fill="")+
  theme(plot.title = element_text(hjust = 0.5))
```



For each type of alcohol, different ratio corresponds to different levels of measurement. However two types of alcohol But2 and Iso1 have similar measurement for different alcohol/air ratio ranging from 0.4 to 0.8, which is difficult for differentiate them only according to channel 1.

Different Channels

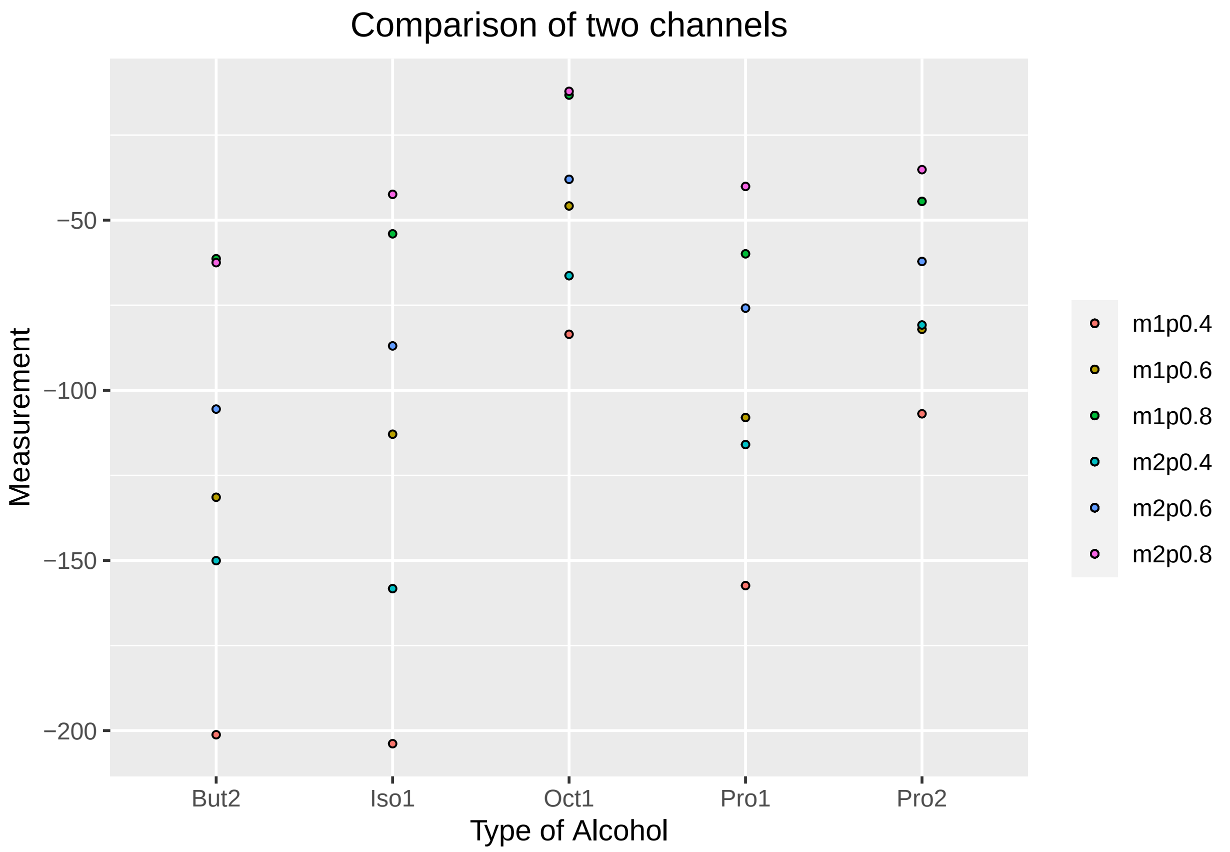
Under the condition of same detector, we summarize the average frequency reduction for both two channel.

```r
diffchannel = group_by(QCM10,labels) %>% summarize(m1p0.8=mean(ch1p0.8), m2p0.8=mean(ch2p0.8), m1p0.6=me
diffchannel
```

```
## # A tibble: 5 x 7
##   labels m1p0.8 m2p0.8 m1p0.6 m2p0.6 m1p0.4 m2p0.4
##    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1      1  -13.2  -12.2  -45.8  -38.0  -83.5  -66.3
## 2      2  -59.9  -40.1 -108.   -75.9 -157.  -116.
## 3      3  -61.3  -62.5 -131.  -106.  -201.  -150.
## 4      4  -44.5  -35.2  -82.1  -62.1 -107.   -80.8
## 5      5  -54.0  -42.4 -113.   -87.0 -204.  -158.
```

Figure measurement of two channels in QCM10 :

```r
diffchannel["labels"] = c("Oct1","Pro1","But2","Pro2","Iso1")
ggplot(diffchannel)+
  geom_point(aes(x=labels,y=m1p0.8,fill="m1p0.8"),size=1,shape=21,color="black")+
  geom_point(aes(x=labels,y=m2p0.8,fill="m2p0.8"),size=1,shape=21)+
  geom_point(aes(x=labels,y=m1p0.6,fill="m1p0.6"),size=1,shape=21)+
  geom_point(aes(x=labels,y=m2p0.6,fill="m2p0.6"),size=1,shape=21)+
  geom_point(aes(x=labels,y=m1p0.4,fill="m1p0.4"),size=1,shape=21)+
  geom_point(aes(x=labels,y=m2p0.4,fill="m2p0.4"),size=1,shape=21)+
  labs(x="Type of Alcohol",y="Measurement",title= "Comparison of two channels",fill="")+
  theme(plot.title = element_text(hjust = 0.5))
```



It seems that the trend of measurement of two channel is similar. Moreover, we find the difference between But2 and Iso1 in channel 2 is significant compared with channel 1, which helps to classify them by data both from two channels.

Different QCM Detectors

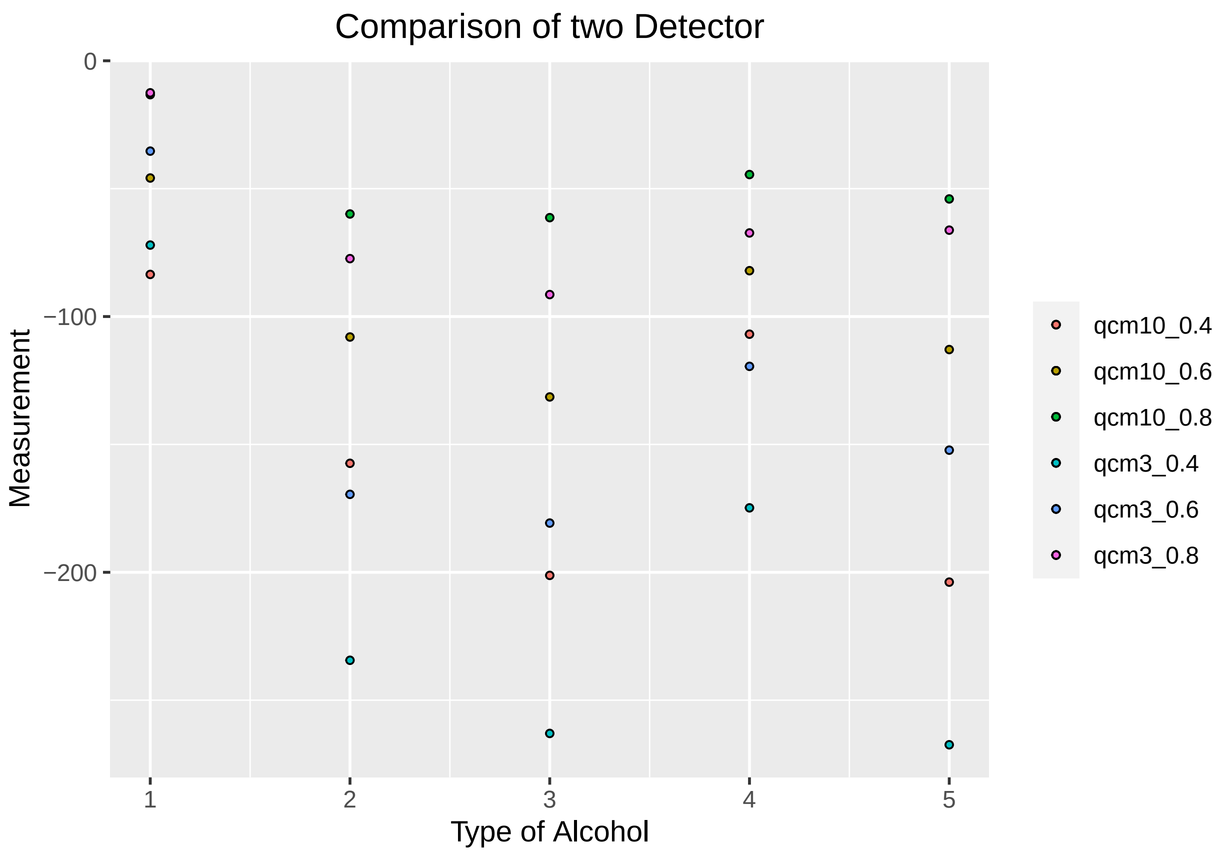
Next we compare the result of different QCM detector with the same channel and same alchohol.

```
qcm10ch1 = group_by(QCM10,labels) %>% summarize(m0.8=mean(ch1p0.8), m0.6=mean(ch1p0.6), m0.4=mean(ch1p0
qcm3ch1 = group_by(QCM3,labels) %>% summarize(m0.8=mean(ch1p0.8), m0.6=mean(ch1p0.6), m0.4=mean(ch1p0.4
qcm10_3ch1 = mutate(data.frame(qcm10ch1), qcm3m0.8 = data.frame(qcm3ch1)[,2], qcm3m0.6 = data.frame(qcm3
qcm10_3ch1
```

```
##   labels    m0.8     m0.6     m0.4 qcm3m0.8 qcm3m0.6 qcm3m0.4
## 1      1 -13.246  -45.834  -83.538  -12.520  -35.324  -72.060
## 2      2 -59.898 -108.002 -157.390  -77.352 -169.522 -234.402
## 3      3 -61.330 -131.434 -201.210  -91.412 -180.750 -263.004
## 4      4 -44.482  -82.072 -106.902  -67.316 -119.478 -174.800
## 5      5 -54.020 -112.906 -203.842  -66.224 -152.242 -267.434
```

Figure measurement of two detectors:

```
ggplot(qcm10_3ch1)+
  geom_point(aes(x=labels,y=m0.8,fill="qcm10_0.8"),size=1,shape=21,color="black")+
  geom_point(aes(x=labels,y=m0.6,fill="qcm10_0.6"),size=1,shape=21)+
  geom_point(aes(x=labels,y=m0.4,fill="qcm10_0.4"),size=1,shape=21)+
  geom_point(aes(x=labels,y=qcm3m0.8,fill="qcm3_0.8"),size=1,shape=21)+
  geom_point(aes(x=labels,y=qcm3m0.6,fill="qcm3_0.6"),size=1,shape=21)+
  geom_point(aes(x=labels,y=qcm3m0.4,fill="qcm3_0.4"),size=1,shape=21)+
  labs(x="Type of Alcohol",y="Measurement",title= "Comparison of two Detector",fill="")+
  theme(plot.title = element_text(hjust = 0.5))
```



The trends between QCM10 and QCM3 detectors are similar. Distinct measurement between different types of alcohol gives detectors chance to classify correctly.

Models

Each QCM itself has one model

For every QCM detector, we fit a random forest model with its dataset. Each dataset is splitted to training and test dataset with 0/7/0.3 ratio. We repeat the process 10 times.

```
library(randomForest)
acc_vec = rep(0,5)
repeat_times = 10
acc_vec_all = matrix(rep(0,repeat_times*5),ncol=5)
for(j in 1:repeat_times){
  for(i in 1:length(dir)){
    index = str_sub(first_category_name[i],1,-5)
    data_index = get(index)
    data_index$labels = as.factor(data_index$labels)
    set.seed(12)
    train_sub = sample(nrow(data_index),7/10*nrow(data_index))
    train_data = data_index[train_sub,]
    test_data = data_index[-train_sub,]
    model <- randomForest(labels~., data=train_data, importance=TRUE,proximity=TRUE)
    #print(model$importance)
    #varImpPlot(model, main = "variable importance")
    test_hat_label <- predict(model,newdata=test_data)
    table(test_data$labels,test_hat_label,dnn=c("Truth","Predict"))
    acc = sum(test_data$labels==test_hat_label)/nrow(test_data)
    acc_vec[i] = acc
    }
  acc_vec_all[j,1:5] = acc_vec
}
acc_vec_all
```

```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]    1    1    1    1    1
##  [2,]    1    1    1    1    1
##  [3,]    1    1    1    1    1
##  [4,]    1    1    1    1    1
##  [5,]    1    1    1    1    1
##  [6,]    1    1    1    1    1
##  [7,]    1    1    1    1    1
##  [8,]    1    1    1    1    1
##  [9,]    1    1    1    1    1
## [10,]    1    1    1    1    1
```

```
sapply(data.frame(acc_vec_all), mean, na.rm = T)
```

```
## X1 X2 X3 X4 X5
##  1  1  1  1  1
```

```
sapply(data.frame(acc_vec_all), var, na.rm = T)
```

```
## X1 X2 X3 X4 X5
##  0  0  0  0  0
```

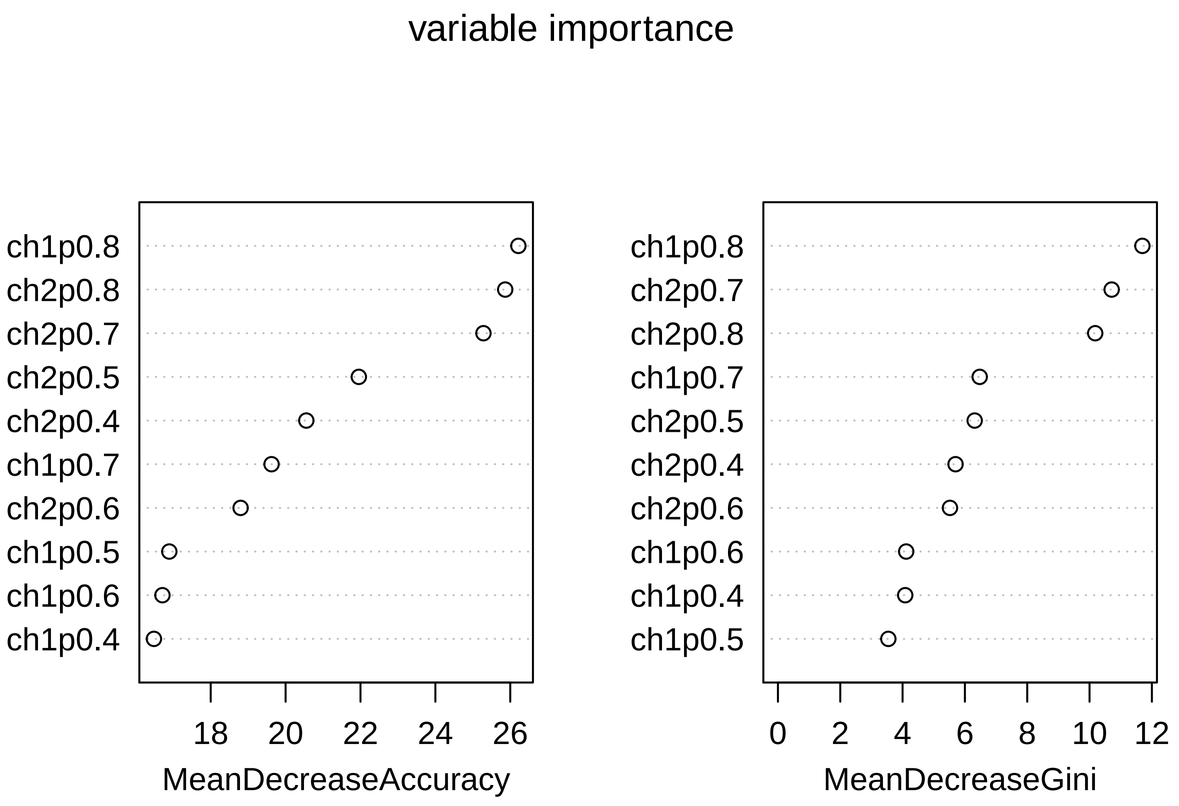
Each column represents the classfication accuracy of one detector on test set. Three QCM detectors(QCM10, QCM12, QCM3) achieve perfect prediction by its own random forest model. The results appear to be satisfying, but we want to want a big model to process data aggregated from five dataset because five models is not convenient in reality. The problem makes sense because maybe new data is from other QCM detector, not any of the five detectors we studied here.

Model with aggregated dataset

For building one model to predict data from five QCM dataset, we simply bind five QCM dataset together by column and obtain a large dataset. Then the large dataset are split with 0/7/0.3 ratio into training and test dataset. We fit one random forest model with the training set and predict the test data.

```
QCM_all = rbind(QCM3,QCM6,QCM7,QCM10,QCM12)
QCM_all$labels = as.factor(QCM_all$labels)
repeat_times = 10
acc_vec_all = rep(0,repeat_times)
```

```r
for(j in 1:repeat_times){
  train_sub = sample(nrow(QCM_all),7/10*nrow(QCM_all))
  train_data = QCM_all[train_sub,]
  test_data = QCM_all[-train_sub,]
  model <- randomForest(labels~., data=train_data, importance=TRUE,proximity=TRUE)
  #print(model$importance)
  #varImpPlot(model, main = "variable importance")
  test_hat_label <- predict(model,newdata=test_data)
  table(test_data$labels,test_hat_label,dnn=c("Truth","Predict"))
  acc_vec_all[j] = sum(test_data$labels==test_hat_label)/nrow(test_data)
}
varImpPlot(model, main = "variable importance")
```



```r
print(mean(acc_vec_all))
```

```
## [1] 0.9736842
```

```r
print(var(acc_vec_all))
```

```
## [1] 0.001231148
```

The result of naive concatenated dataset is not satisfying. In addition, the information of five QCM detectors is not included in data. Then we explore the reason why we cannot perfectly predict the class of alcohol.

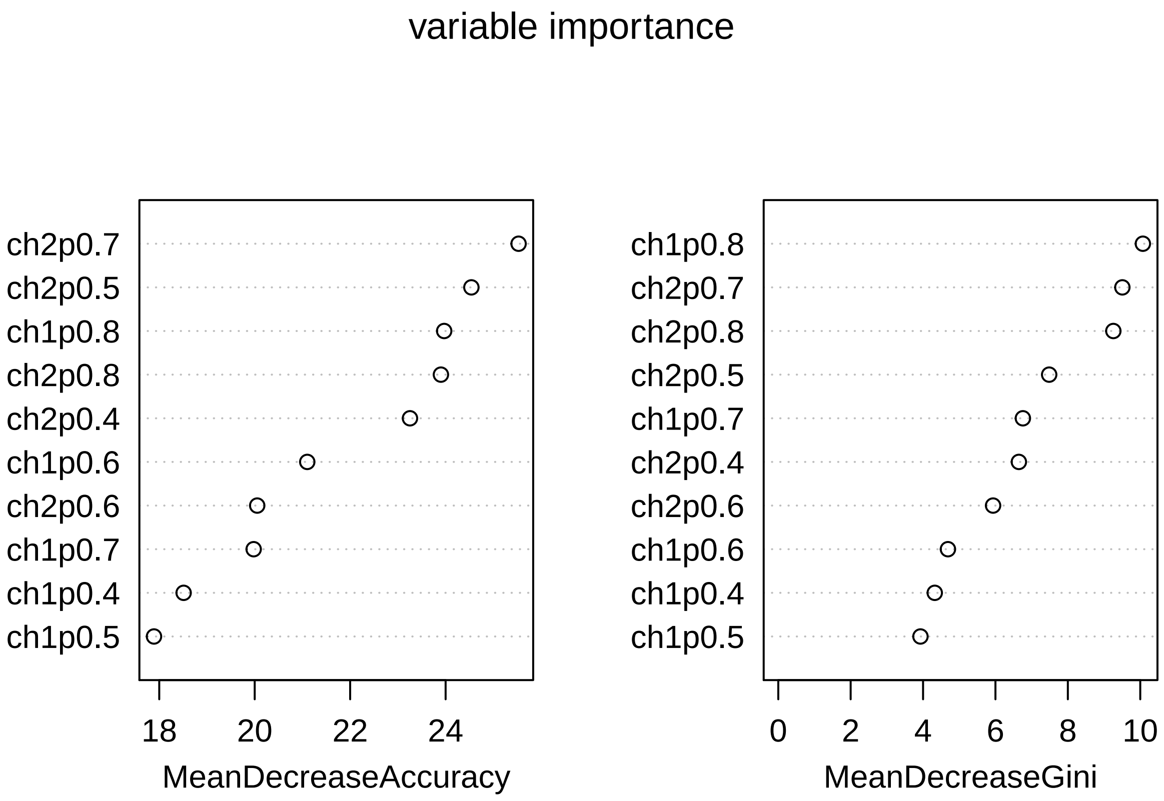
Why Misclassify?

Because we adopt the random forest algorithm, the predicted label depends on the majority votes of labels of the points which resides in the same leaf cell. Because the nearest neighbor of a point are the most possible ones residing in the same leaf cell as it, we explore labels of the nearest neighbors of misclassified point. All Misclassification Examples in ten repeated experients:

```
set.seed(1233)
train_sub = sample(nrow(QCM_all),7/10*nrow(QCM_all))
train_data = QCM_all[train_sub,]
test_data = QCM_all[-train_sub,]
model <- randomForest(labels~., data=train_data, importance=TRUE,proximity=TRUE)
#print(model$importance)
test_hat_label <- predict(model,newdata=test_data)
table(test_data$labels,test_hat_label,dnn=c("Truth","Predict"))
```

```
##      Predict
## Truth 1 2 3 4 5
##     1 9 0 0 0 0
##     2 0 7 0 0 1
##     3 0 0 4 0 0
##     4 0 0 0 8 0
##     5 0 1 0 0 8
```

```
acc = sum(test_data$labels==test_hat_label)/nrow(test_data)
varImpPlot(model, main = "variable importance")
```



```
print(acc)
```

```
## [1] 0.9473684
```

```
false_points = test_data[test_data$labels!=test_hat_label,]
dist_vec = rep(0,nrow(train_data))
for(i in 1:nrow(false_points)){
    print("Misclassified Point:")
    print(false_points[i,])
    print("Predicted Label:")
    print(predict(model,false_points[i,])[1])
    for (k in 1:nrow(train_data)){
        dist_vec[k] = sum((false_points[i,1:10]-train_data[k,1:10])^2)
    }
    print("Labels of ten nearest neighbors of the misclassified point:")
    print(train_data$labels[order(dist_vec)[1:10]])
    print("Indices of ten nearest neighbors of the misclassified point:")
    print(rownames(train_data[order(dist_vec)[1:10],]))
}
```

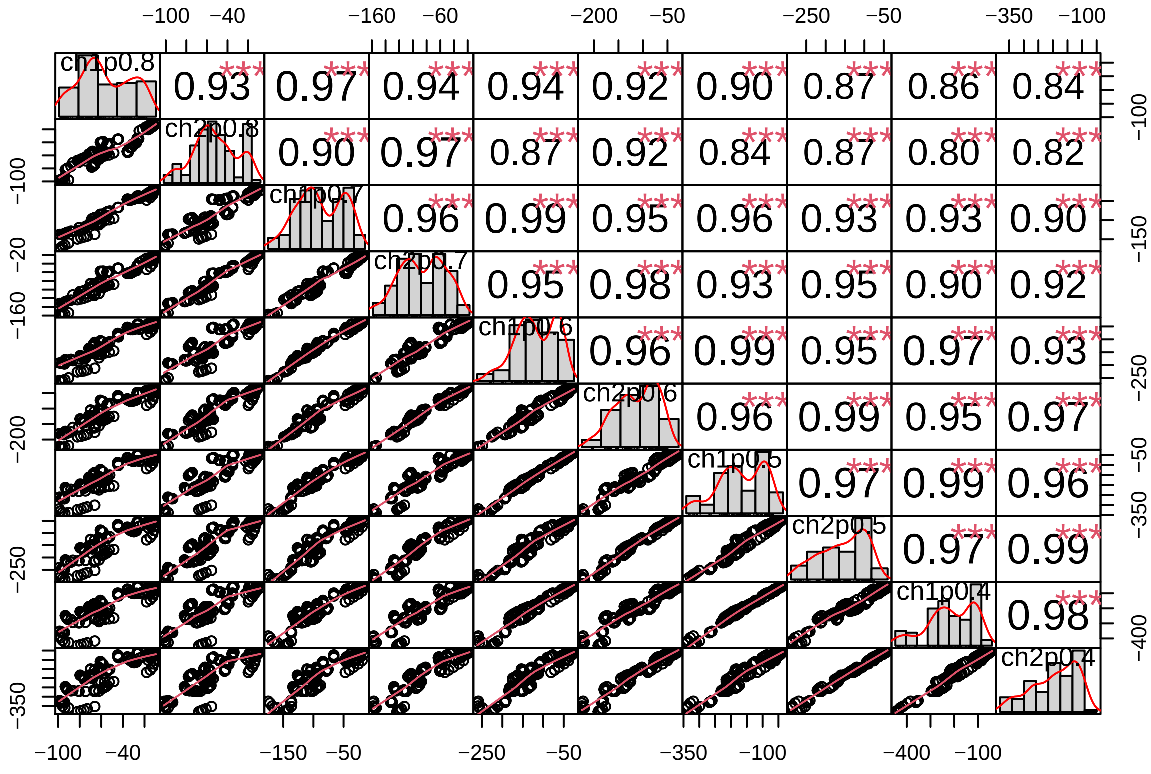
```
## [1] "Misclassified Point:"
##    ch1p0.8 ch2p0.8 ch1p0.7 ch2p0.7 ch1p0.6 ch2p0.6 ch1p0.5 ch2p0.5 ch1p0.4
## 71  -27.45  -40.42   -42.1  -56.14  -58.24  -72.12  -76.97  -90.61  -99.22
##    ch2p0.4 labels
## 71 -112.17      5
## [1] "Predicted Label:"
## 71
##  2
## Levels: 1 2 3 4 5
## [1] "Labels of ten nearest neighbors of the misclassified point:"
##  [1] 5 5 3 3 3 3 2 2 2 2
## Levels: 1 2 3 4 5
## [1] "Indices of ten nearest neighbors of the misclassified point:"
##  [1] "73" "74" "61" "62" "64" "65" "56" "57" "58" "59"
## [1] "Misclassified Point:"
##    ch1p0.8 ch2p0.8 ch1p0.7 ch2p0.7 ch1p0.6 ch2p0.6 ch1p0.5 ch2p0.5 ch1p0.4
## 81  -58.41  -38.66  -83.58  -57.33 -110.24   -76.8 -133.77  -96.13 -171.05
##    ch2p0.4 labels
## 81 -124.15      2
## [1] "Predicted Label:"
## 81
##  5
## Levels: 1 2 3 4 5
## [1] "Labels of ten nearest neighbors of the misclassified point:"
##  [1] 2 2 2 5 5 3 5 5 3 3
## Levels: 1 2 3 4 5
## [1] "Indices of ten nearest neighbors of the misclassified point:"
##  [1] "83"  "84"  "85"  "100" "97"  "90"  "98"  "99"  "89"  "86"
```

From the output when random seed for splitting training and test dataset is fixed, we find "71"th and "81"th points are misclassified. Although the nearest neighbor of the 71th point belongs to the same class as themselves(the first and second nearest neighbor of "71"th point are both belong to Class 5, the same as the "71"th point), a large proportion of nearest ten neighbors don't belong to the same class. Meanwhile, the ten points all have large possiblility to reside in the same leaf cell with the 71th point when the parameter "node size" is set by 5.

For improving the accuracy, according to the knowledge of random forest, the correlation between different decision trees is preferably low. When the size of samples is fixed, we hope that the correlation between feature space of different trees is low. Up to now, our features are only from measuring the alcohol/air gas mixed by different ratios. Apparently, these features are highly correlated showed in the following figure. Consequently, some variables uncorrelated to mixed gas are desirable, such as the QCM dectectors's chemistry component, structure, sentitivity…

Correlation Figure:

```
library(PerformanceAnalytics)
MM = subset(QCM_all,select=-labels)
chart.Correlation(MM,type = "upper", order = "hclust", tl.col = "black", pch=9)
```



Model with refined dataset

Fortunately, we find some information related to the ratio of component in two channals of different QCM detectors. One of the channels includes molecularly imprinted polymers (MIP), and the other includes nanoparticles (NP).

Table of MIP and NP ratios used in QCM detectors:

Sensor name	MIP ratio	NP ratio
QCM3	1	1
QCM6	1	0
QCM7	1	0.5
QCM10	1	2
QCM12	0	1

Add new columns MIP_ratio and NP_ratio into five QCM dataframes:

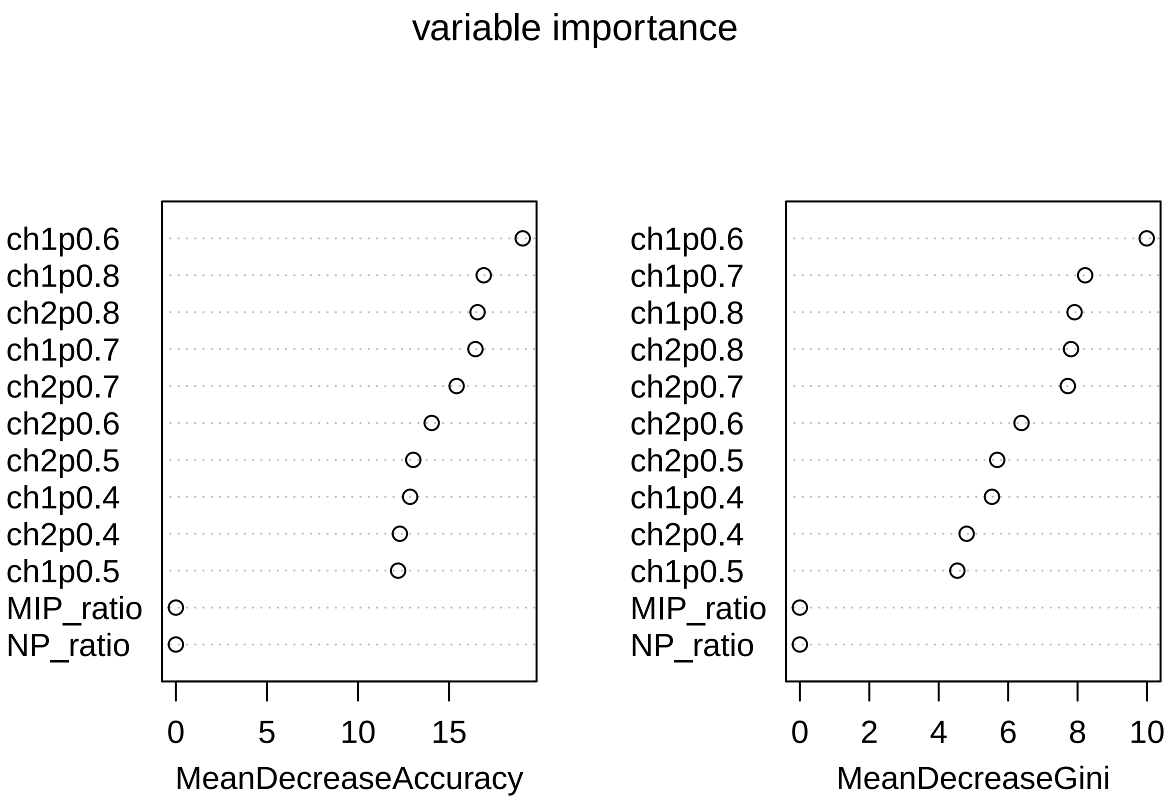
```
QCM3_more = mutate(QCM3,MIP_ratio = rep(1,25), NP_ratio=rep(1,25))
QCM6_more = mutate(QCM3,MIP_ratio = rep(1,25), NP_ratio=rep(0,25))
QCM7_more = mutate(QCM3,MIP_ratio = rep(1,25), NP_ratio=rep(0.5,25))
QCM10_more = mutate(QCM3,MIP_ratio = rep(1,25), NP_ratio=rep(2,25))
QCM12_more = mutate(QCM3,MIP_ratio = rep(0,25), NP_ratio=rep(1,25))
QCM_all_more = rbind(QCM3_more,QCM6_more,QCM7_more,QCM10_more,QCM12_more)
```

```r
QCM_all_more$labels = as.factor(QCM_all_more$labels)
head(QCM_all_more)
```

```
##   ch1p0.8 ch2p0.8 ch1p0.7 ch2p0.7 ch1p0.6 ch2p0.6 ch1p0.5 ch2p0.5 ch1p0.4
## 1  -10.06  -10.62  -14.43  -18.31  -24.64  -30.56  -38.62  -45.59  -54.89
## 2   -9.69  -10.86  -16.73  -21.75  -28.47  -35.83  -43.65  -52.43  -61.92
## 3  -12.07  -14.28  -21.54  -27.92  -35.19  -43.94  -52.04  -62.49  -71.97
## 4  -14.21  -17.41  -25.91  -33.36  -41.29  -51.27  -59.94  -71.55  -81.51
## 5  -16.57  -20.35  -29.97  -37.84  -47.03  -57.29  -67.13  -78.96  -90.01
## 6  -75.61  -64.10 -122.08 -102.17 -174.79 -145.50 -214.69 -177.30 -250.83
##   ch2p0.4 labels MIP_ratio NP_ratio
## 1  -62.28      1         1        1
## 2  -71.27      1         1        1
## 3  -83.10      1         1        1
## 4  -93.83      1         1        1
## 5 -102.65      1         1        1
## 6 -207.99      2         1        1
```

Fixing the same random seed as the above section, we fit one random forest model to the refined data.

```r
set.seed(1233)
train_sub = sample(nrow(QCM_all_more),7/10*nrow(QCM_all_more))
train_data = QCM_all_more[train_sub,]
test_data = QCM_all_more[-train_sub,]
model <- randomForest(labels~., data=train_data, importance=TRUE,proximity=TRUE)
#print(model$importance)
varImpPlot(model, main = "variable importance")
```



```r
test_hat_label <- predict(model,newdata=test_data)
table(test_data$labels,test_hat_label,dnn=c("Truth","Predict"))
```

```
##       Predict
## Truth 1 2 3 4 5
##     1 9 0 0 0 0
##     2 0 8 0 0 0
##     3 0 0 4 0 0
##     4 0 0 0 8 0
##     5 0 0 0 0 9
```

```r
acc = sum(test_data$labels==test_hat_label)/nrow(test_data)
print(model$importance)
```

```
##                   1          2          3          4          5
## ch1p0.8   0.1200556 0.14201775 0.14382229 0.09540592 0.10754286
## ch2p0.8   0.1278389 0.15888680 0.07668211 0.14014673 0.11666667
## ch1p0.7   0.1473563 0.12357496 0.12771622 0.12648185 0.10453968
## ch2p0.7   0.1181743 0.08152338 0.15781638 0.08104925 0.12224863
## ch1p0.6   0.1677948 0.15356955 0.15023800 0.16031512 0.14603737
## ch2p0.6   0.1081651 0.07893593 0.14537565 0.06913311 0.10990411
## ch1p0.5   0.1261841 0.04580000 0.02139966 0.11276111 0.04996984
## ch2p0.5   0.1193690 0.08915678 0.03493057 0.11678588 0.10334524
## ch1p0.4   0.1413889 0.07331190 0.01755476 0.11736427 0.09218571
## ch2p0.4   0.1177706 0.06103355 0.04072339 0.06591854 0.09616825
## MIP_ratio 0.0000000 0.00000000 0.00000000 0.00000000 0.00000000
## NP_ratio  0.0000000 0.00000000 0.00000000 0.00000000 0.00000000
##           MeanDecreaseAccuracy MeanDecreaseGini
## ch1p0.8             0.12497488         7.913704
## ch2p0.8             0.12260545         7.810856
## ch1p0.7             0.12471008         8.218415
## ch2p0.7             0.11542058         7.719281
## ch1p0.6             0.15401129         9.992218
## ch2p0.6             0.10445777         6.385861
## ch1p0.5             0.07189118         4.536164
## ch2p0.5             0.09330005         5.685003
## ch1p0.4             0.08493448         5.533907
## ch2p0.4             0.07483003         4.804498
## MIP_ratio           0.00000000         0.000000
## NP_ratio            0.00000000         0.000000
```

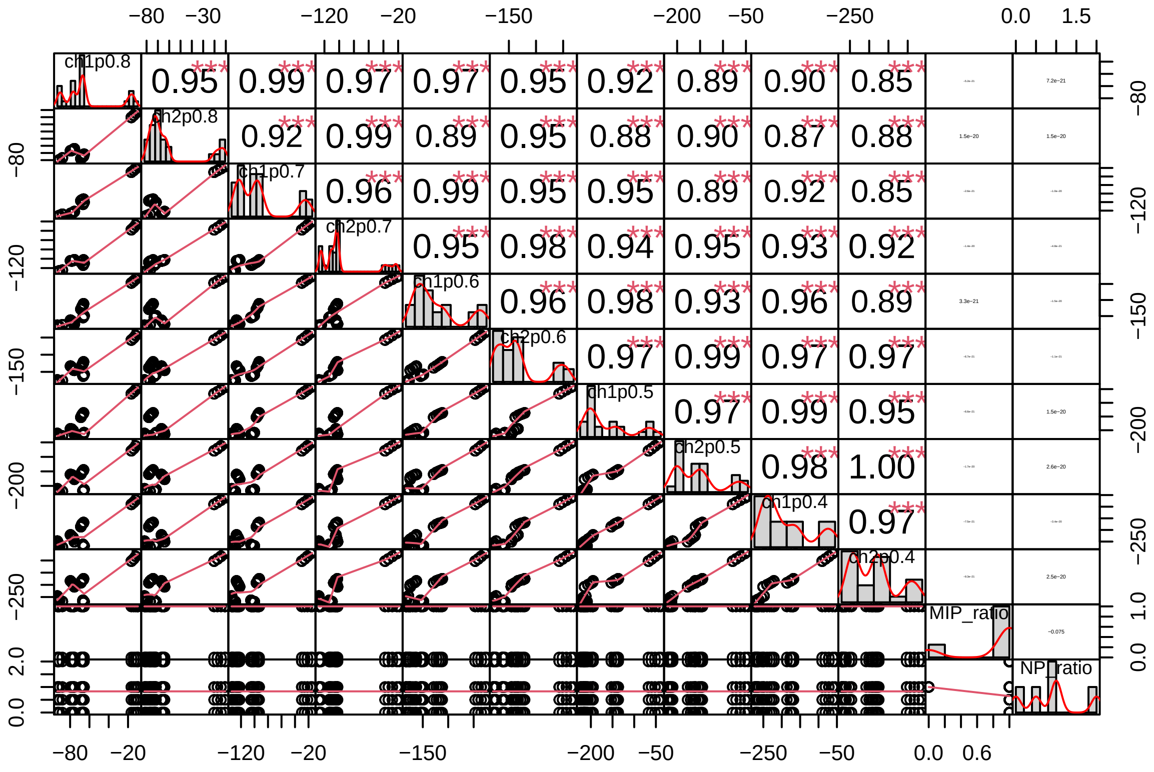
```r
print(acc)
```

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

Discussion about results

The accuracy are improved from 94.74% to 100%, a perfect result. However, from the above Variable Importance Figure, we find the variable importance index "MeanDecreaseAccuracy" and "MeanDecreaseGini" of features MIP_ratio and NP_ratio are both zero, which seems to indicate two variables are useless. But think carefully, this result is also reasonable. The type of detector should be independent of the type of alcohol you detect. Furthermore, this improvements on accuracy benefits from the the uncorrelation between MIP_ratio and NP_ratio and other ten measurements. The new features decrease the correlation between decision trees. The following correlation figure shows the correlation between MIP_ratio (NP_ratio) and other features are nearly zero.

```r
library(PerformanceAnalytics)
MM = subset(QCM_all_more,select=-labels)
chart.Correlation(MM,type = "upper", order = "hclust", tl.col = "black", pch=9)
```



Moreover, the model can be applied to data from new QCM detectors with different MIP and NP ratio, not merely limited to the five types of QCM detectors we studied here. At last, we repeat ten times to test the stability of prediction.

```r
repeat_times = 10
acc_vec_all = rep(0,repeat_times)
for(j in 1:repeat_times){
  train_sub = sample(nrow(QCM_all_more),7/10*nrow(QCM_all_more))
  train_data = QCM_all_more[train_sub,]
  test_data = QCM_all_more[-train_sub,]
  model <- randomForest(labels~., data=train_data, importance=TRUE,proximity=TRUE)
  #print(model$importance)
  #varImpPlot(model, main = "variable importance")
  test_hat_label <- predict(model,newdata=test_data)
  table(test_data$labels,test_hat_label,dnn=c("Truth","Predict"))
  acc_vec_all[j] = sum(test_data$labels==test_hat_label)/nrow(test_data)
}
print(mean(acc_vec_all))
```

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

```r
print(var(acc_vec_all))
```

```
## [1] 0
```

Consequently, we obtain the perfect prediction results by combining the random forest model and feature engineering with the help of knowledge of chemistry. Treat ratio of the components of QCM detectors as features is an innovative attempt that has not been done before.