Final Project

Hector Santana, Harris Dupre, Christopher Ayre

5/9/2022

Our initial step is to import the relevant libraries containing the requisite models of this analysis.

```
library(mlbench)
library(randomForest)
library(caret)
library(party)
library(Cubist)
library(dplyr)
library(rpart.plot)
library(kernlab)
library(earth)
library(nnet)
library(DataExplorer)
library(RANN)
library(corrplot)
pacman::p_load(tidyverse,janitor,DataExplorer,knitr,arsenal,kableExtra,car,geoR,caret,
               psych,gridExtra,DMwR2,lmtest,pscl,MKmisc,ROCR,survey,stats,rstatix,Rcpp,
               corrplot,forecast,cowplot)

library(mice)
library(RColorBrewer)
library(VIM)
library(openxlsx)
```

To create ease around the data importation process we converted the excel files into CSV and stored them in a GitHub repository.

```
studenttrainingdata = read.csv('https://raw.githubusercontent.com/manonfire86/Data624FinalProject/main/S

studenttestdata = read.csv('https://raw.githubusercontent.com/manonfire86/Data624FinalProject/main/Stude
```

The below exploratory analysis indicates that missing values are fairly low. For our model to be robust and dynamic we will impute missing data, filter out correlations, near zero variables, scale the data, and center the data in our preprocess methodology.This will remove collinearity, remove skew, normalize distributions, scale the data, and remove non significant variables.

```
str(studenttrainingdata)
```

```
## 'data.frame':    2571 obs. of  33 variables:
##  $ ï..Brand.Code    : chr  "B" "A" "B" "A" ...
```

```
##  $ Carb.Volume      : num  5.34 5.43 5.29 5.44 5.49 ...
##  $ Fill.Ounces      : num  24 24 24.1 24 24.3 ...
##  $ PC.Volume        : num  0.263 0.239 0.263 0.293 0.111 ...
##  $ Carb.Pressure    : num  68.2 68.4 70.8 63 67.2 66.6 64.2 67.6 64.2 72 ...
##  $ Carb.Temp        : num  141 140 145 133 137 ...
##  $ PSC              : num  0.104 0.124 0.09 NA 0.026 0.09 0.128 0.154 0.132 0.014 ...
##  $ PSC.Fill         : num  0.26 0.22 0.34 0.42 0.16 0.24 0.4 0.34 0.12 0.24 ...
##  $ PSC.CO2          : num  0.04 0.04 0.16 0.04 0.12 0.04 0.04 0.04 0.14 0.06 ...
##  $ Mnf.Flow         : num  -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 ...
##  $ Carb.Pressure1   : num  119 122 120 115 118 ...
##  $ Fill.Pressure    : num  46 46 46 46.4 45.8 45.6 51.8 46.8 46 45.2 ...
##  $ Hyd.Pressure1    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Hyd.Pressure2    : num  NA NA NA 0 0 0 0 0 0 0 ...
##  $ Hyd.Pressure3    : num  NA NA NA 0 0 0 0 0 0 0 ...
##  $ Hyd.Pressure4    : int  118 106 82 92 92 116 124 132 90 108 ...
##  $ Filler.Level     : num  121 119 120 118 119 ...
##  $ Filler.Speed     : int  4002 3986 4020 4012 4010 4014 NA 1004 4014 4028 ...
##  $ Temperature      : num  66 67.6 67 65.6 65.6 66.2 65.8 65.2 65.4 66.6 ...
##  $ Usage.cont       : num  16.2 19.9 17.8 17.4 17.7 ...
##  $ Carb.Flow        : int  2932 3144 2914 3062 3054 2948 30 684 2902 3038 ...
##  $ Density          : num  0.88 0.92 1.58 1.54 1.54 1.52 0.84 0.84 0.9 0.9 ...
##  $ MFR              : num  725 727 735 731 723 ...
##  $ Balling          : num  1.4 1.5 3.14 3.04 3.04 ...
##  $ Pressure.Vacuum  : num  -4 -4 -3.8 -4.4 -4.4 -4.4 -4.4 -4.4 -4.4 -4.4 ...
##  $ PH               : num  8.36 8.26 8.94 8.24 8.26 8.32 8.4 8.38 8.38 8.5 ...
##  $ Oxygen.Filler    : num  0.022 0.026 0.024 0.03 0.03 0.024 0.066 0.046 0.064 0.022 ...
##  $ Bowl.Setpoint    : int  120 120 120 120 120 120 120 120 120 120 ...
##  $ Pressure.Setpoint: num  46.4 46.8 46.6 46 46 46 46 46 46 46 ...
##  $ Air.Pressurer    : num  143 143 142 146 146 ...
##  $ Alch.Rel         : num  6.58 6.56 7.66 7.14 7.14 7.16 6.54 6.52 6.52 6.54 ...
##  $ Carb.Rel         : num  5.32 5.3 5.84 5.42 5.44 5.44 5.38 5.34 5.34 5.34 ...
##  $ Balling.Lvl      : num  1.48 1.56 3.28 3.04 3.04 3.02 1.44 1.44 1.44 1.38 ...
```

```
summary(studenttrainingdata)
```

```
##  ï..Brand.Code       Carb.Volume     Fill.Ounces      PC.Volume
##  Length:2571        Min.   :5.040   Min.   :23.63   Min.   :0.07933
##  Class :character   1st Qu.:5.293   1st Qu.:23.92   1st Qu.:0.23917
##  Mode  :character   Median :5.347   Median :23.97   Median :0.27133
##                     Mean   :5.370   Mean   :23.97   Mean   :0.27712
##                     3rd Qu.:5.453   3rd Qu.:24.03   3rd Qu.:0.31200
##                     Max.   :5.700   Max.   :24.32   Max.   :0.47800
##                     NA's   :10      NA's   :38      NA's   :39
##  Carb.Pressure     Carb.Temp          PSC             PSC.Fill
##  Min.   :57.00   Min.   :128.6   Min.   :0.00200   Min.   :0.0000
##  1st Qu.:65.60   1st Qu.:138.4   1st Qu.:0.04800   1st Qu.:0.1000
##  Median :68.20   Median :140.8   Median :0.07600   Median :0.1800
##  Mean   :68.19   Mean   :141.1   Mean   :0.08457   Mean   :0.1954
##  3rd Qu.:70.60   3rd Qu.:143.8   3rd Qu.:0.11200   3rd Qu.:0.2600
##  Max.   :79.40   Max.   :154.0   Max.   :0.27000   Max.   :0.6200
##  NA's   :27      NA's   :26      NA's   :33        NA's   :23
##     PSC.CO2           Mnf.Flow       Carb.Pressure1  Fill.Pressure
##  Min.   :0.00000   Min.   :-100.20   Min.   :105.6   Min.   :34.60
##  1st Qu.:0.02000   1st Qu.:-100.00   1st Qu.:119.0   1st Qu.:46.00
```

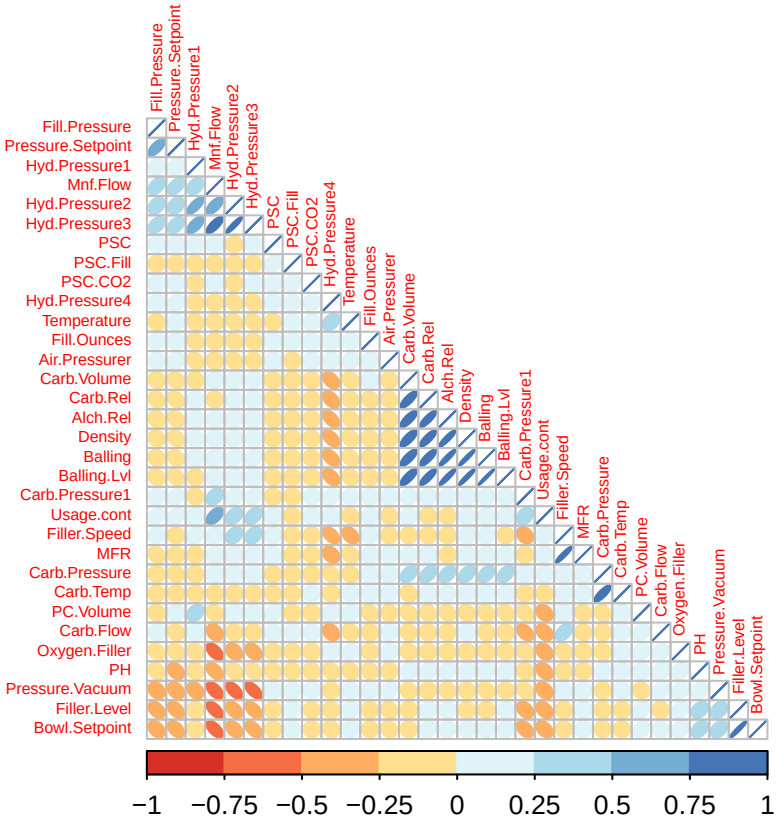
```
##  Median :0.04000   Median :  65.20   Median :123.2   Median :46.40
##  Mean   :0.05641   Mean   :  24.57   Mean   :122.6   Mean   :47.92
##  3rd Qu.:0.08000   3rd Qu.: 140.80   3rd Qu.:125.4   3rd Qu.:50.00
##  Max.   :0.24000   Max.   : 229.40   Max.   :140.2   Max.   :60.40
##  NA's   :39        NA's   :2         NA's   :32      NA's   :22
##  Hyd.Pressure1   Hyd.Pressure2   Hyd.Pressure3   Hyd.Pressure4
##  Min.   :-0.80   Min.   : 0.00   Min.   :-1.20   Min.   : 52.00
##  1st Qu.: 0.00   1st Qu.: 0.00   1st Qu.: 0.00   1st Qu.: 86.00
##  Median :11.40   Median :28.60   Median :27.60   Median : 96.00
##  Mean   :12.44   Mean   :20.96   Mean   :20.46   Mean   : 96.29
##  3rd Qu.:20.20   3rd Qu.:34.60   3rd Qu.:33.40   3rd Qu.:102.00
##  Max.   :58.00   Max.   :59.40   Max.   :50.00   Max.   :142.00
##  NA's   :11      NA's   :15      NA's   :15      NA's   :30
##   Filler.Level    Filler.Speed   Temperature      Usage.cont      Carb.Flow
##  Min.   : 55.8   Min.   : 998   Min.   :63.60   Min.   :12.08   Min.   :  26
##  1st Qu.: 98.3   1st Qu.:3888   1st Qu.:65.20   1st Qu.:18.36   1st Qu.:1144
##  Median :118.4   Median :3982   Median :65.60   Median :21.79   Median :3028
##  Mean   :109.3   Mean   :3687   Mean   :65.97   Mean   :20.99   Mean   :2468
##  3rd Qu.:120.0   3rd Qu.:3998   3rd Qu.:66.40   3rd Qu.:23.75   3rd Qu.:3186
##  Max.   :161.2   Max.   :4030   Max.   :76.20   Max.   :25.90   Max.   :5104
##  NA's   :20      NA's   :57     NA's   :14      NA's   :5       NA's   :2
##     Density           MFR           Balling       Pressure.Vacuum
##  Min.   :0.240   Min.   : 31.4   Min.   :-0.170   Min.   :-6.600
##  1st Qu.:0.900   1st Qu.:706.3   1st Qu.: 1.496   1st Qu.:-5.600
##  Median :0.980   Median :724.0   Median : 1.648   Median :-5.400
##  Mean   :1.174   Mean   :704.0   Mean   : 2.198   Mean   :-5.216
##  3rd Qu.:1.620   3rd Qu.:731.0   3rd Qu.: 3.292   3rd Qu.:-5.000
##  Max.   :1.920   Max.   :868.6   Max.   : 4.012   Max.   :-3.600
##  NA's   :1       NA's   :212     NA's   :1
##        PH        Oxygen.Filler     Bowl.Setpoint   Pressure.Setpoint
##  Min.   :7.880   Min.   :0.00240   Min.   : 70.0   Min.   :44.00
##  1st Qu.:8.440   1st Qu.:0.02200   1st Qu.:100.0   1st Qu.:46.00
##  Median :8.540   Median :0.03340   Median :120.0   Median :46.00
##  Mean   :8.546   Mean   :0.04684   Mean   :109.3   Mean   :47.62
##  3rd Qu.:8.680   3rd Qu.:0.06000   3rd Qu.:120.0   3rd Qu.:50.00
##  Max.   :9.360   Max.   :0.40000   Max.   :140.0   Max.   :52.00
##  NA's   :4       NA's   :12        NA's   :2       NA's   :12
##  Air.Pressurer      Alch.Rel        Carb.Rel      Balling.Lvl
##  Min.   :140.8   Min.   :5.280   Min.   :4.960   Min.   :0.00
##  1st Qu.:142.2   1st Qu.:6.540   1st Qu.:5.340   1st Qu.:1.38
##  Median :142.6   Median :6.560   Median :5.400   Median :1.48
##  Mean   :142.8   Mean   :6.897   Mean   :5.437   Mean   :2.05
##  3rd Qu.:143.0   3rd Qu.:7.240   3rd Qu.:5.540   3rd Qu.:3.14
##  Max.   :148.2   Max.   :8.620   Max.   :6.060   Max.   :3.66
##                  NA's   :9       NA's   :10      NA's   :1
```

```
dim(studenttrainingdata)
```

```
## [1] 2571   33
```

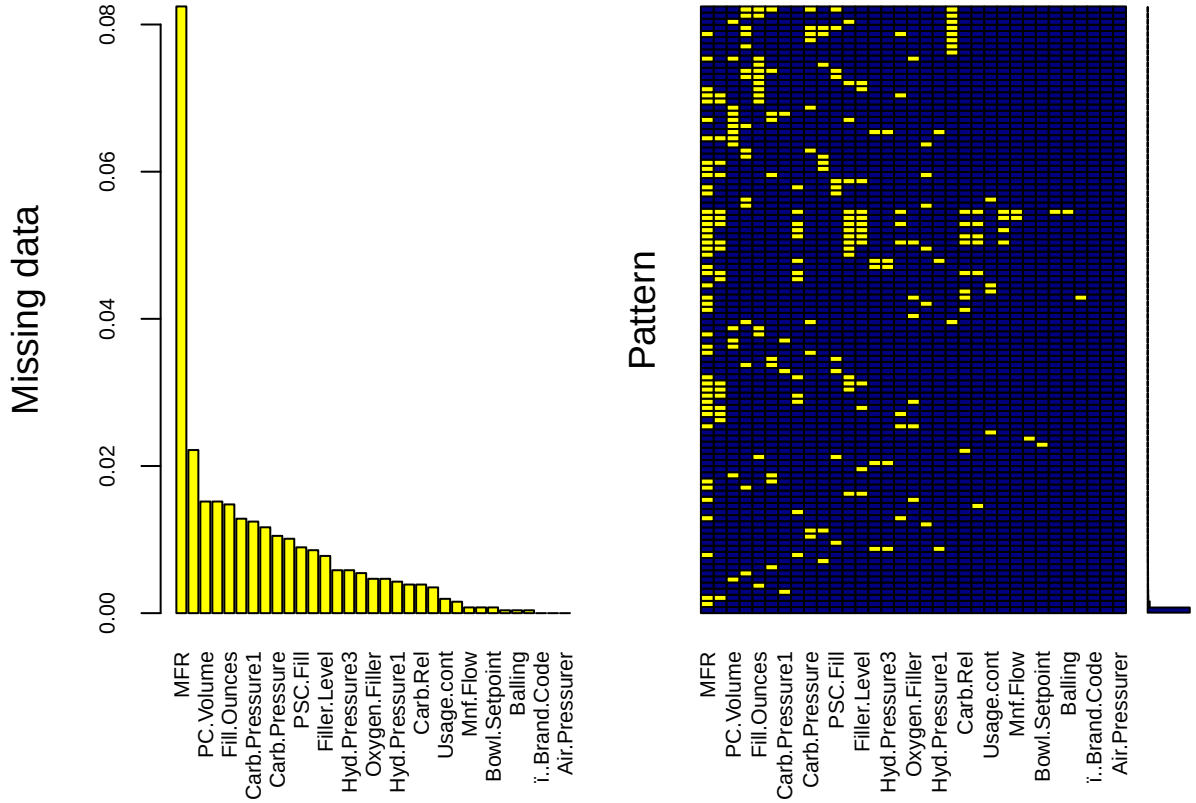
```
df_features <- studenttrainingdata %>%
  dplyr::select(-c(`ï..Brand.Code`))
```

```
correlation = cor(df_features, use = 'pairwise.complete.obs')
corrplot(correlation, 'ellipse', type = 'lower', order = 'hclust',
         col=brewer.pal(n=8, name="RdYlBu"), tl.cex=0.5)
```




```
mice_plot <- aggr(studenttrainingdata, col=c('navyblue','yellow'),
                  numbers=TRUE, sortVars=TRUE,
                  labels=names(studenttrainingdata), cex.axis=.7,
                  gap=3, ylab=c("Missing data","Pattern"))
```

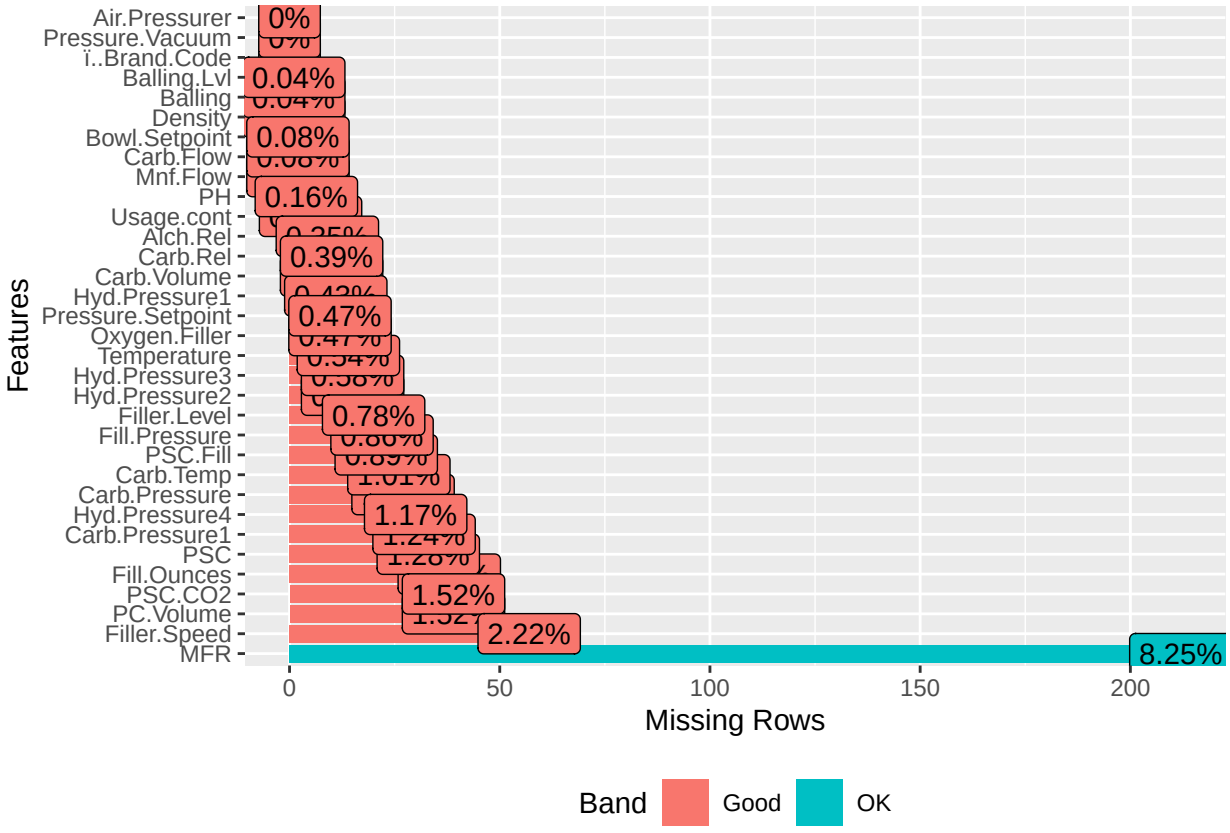
```
## Warning in plot.aggr(res, ...): not enough vertical space to display frequencies
## (too many combinations)
```



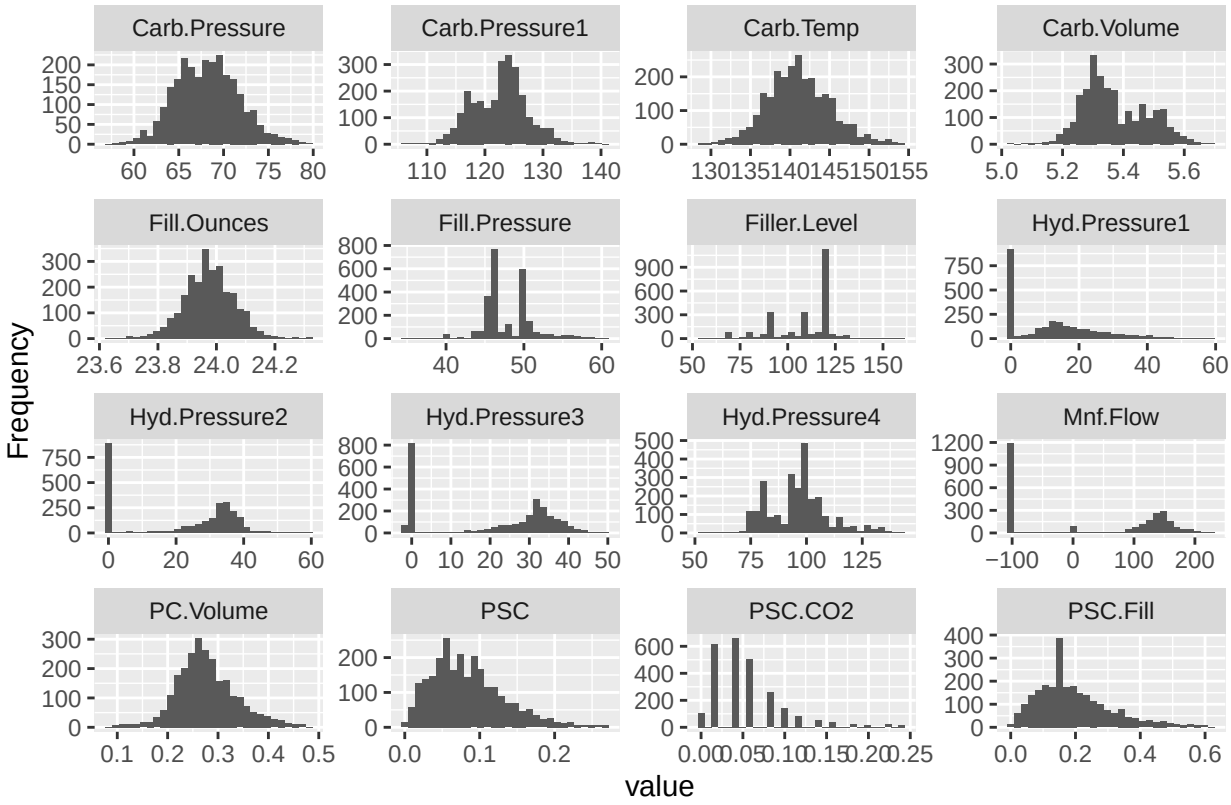

```
##
##  Variables sorted by number of missings:
##            Variable        Count
##                 MFR 0.0824581875
##        Filler.Speed 0.0221703617
##           PC.Volume 0.0151691949
##             PSC.CO2 0.0151691949
##          Fill.Ounces 0.0147802412
##                 PSC 0.0128354726
##     Carb.Pressure1 0.0124465189
##      Hyd.Pressure4 0.0116686114
##      Carb.Pressure 0.0105017503
##          Carb.Temp 0.0101127966
##           PSC.Fill 0.0089459354
##      Fill.Pressure 0.0085569817
##        Filler.Level 0.0077790743
##      Hyd.Pressure2 0.0058343057
##      Hyd.Pressure3 0.0058343057
##        Temperature 0.0054453520
##      Oxygen.Filler 0.0046674446
##  Pressure.Setpoint 0.0046674446
##      Hyd.Pressure1 0.0042784909
##         Carb.Volume 0.0038895371
##            Carb.Rel 0.0038895371
##            Alch.Rel 0.0035005834
##         Usage.cont 0.0019447686
```

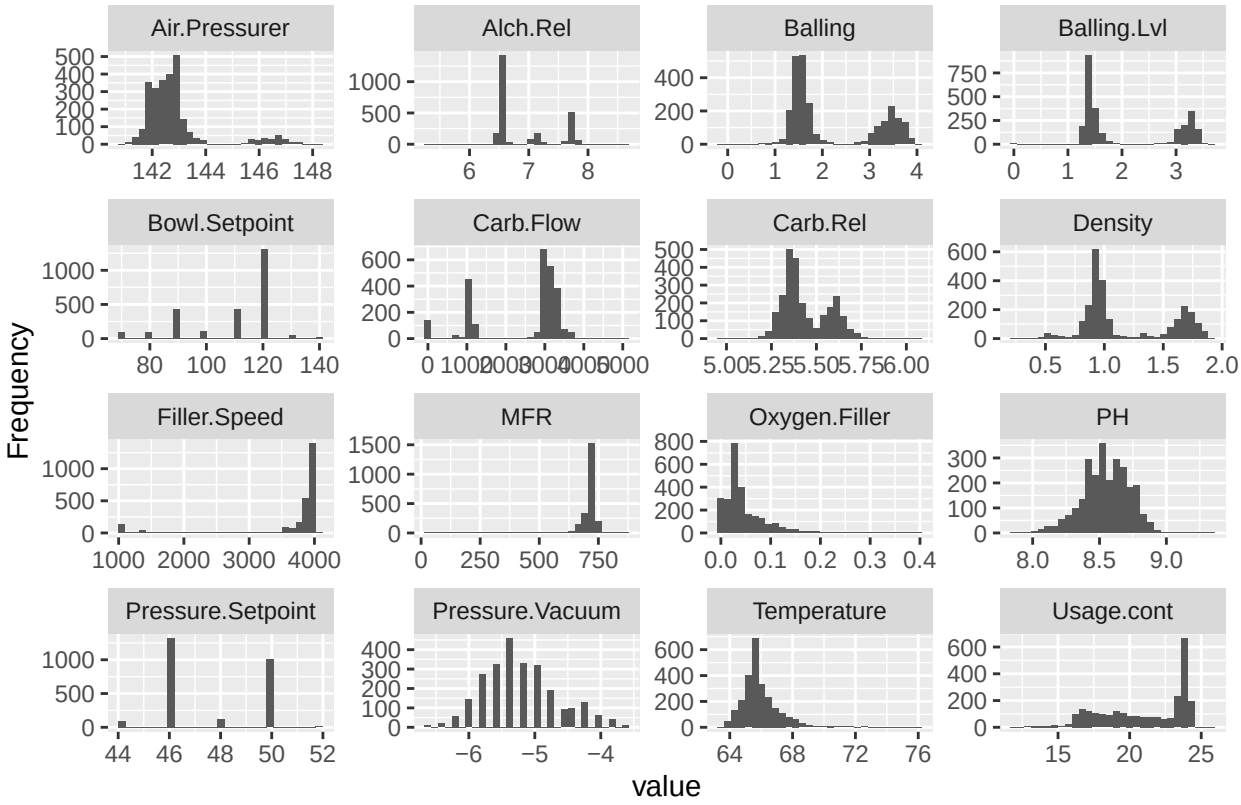
```
##                 PH 0.0015558149
##           Mnf.Flow 0.0007779074
##          Carb.Flow 0.0007779074
##      Bowl.Setpoint 0.0007779074
##            Density 0.0003889537
##            Balling 0.0003889537
##        Balling.Lvl 0.0003889537
##      ï..Brand.Code 0.0000000000
##    Pressure.Vacuum 0.0000000000
##      Air.Pressurer 0.0000000000
```

```
plot_missing(studenttrainingdata)
```

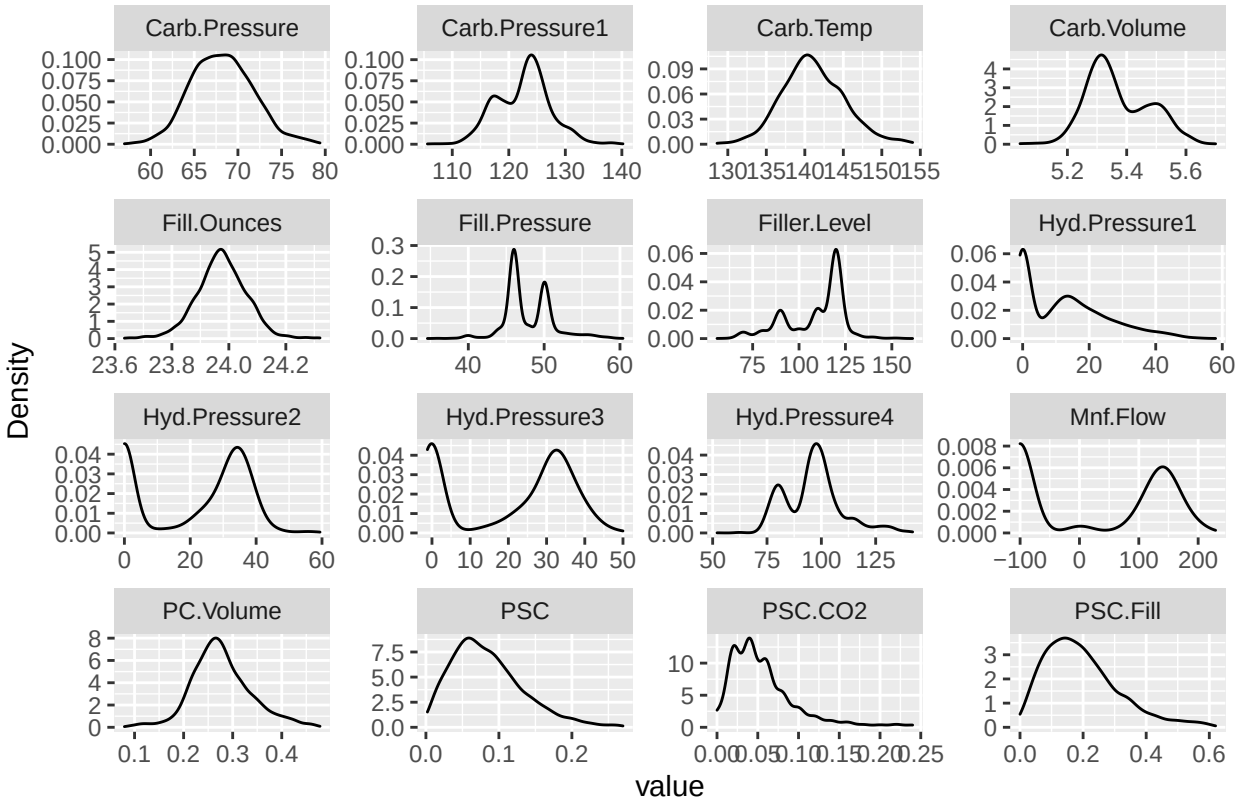


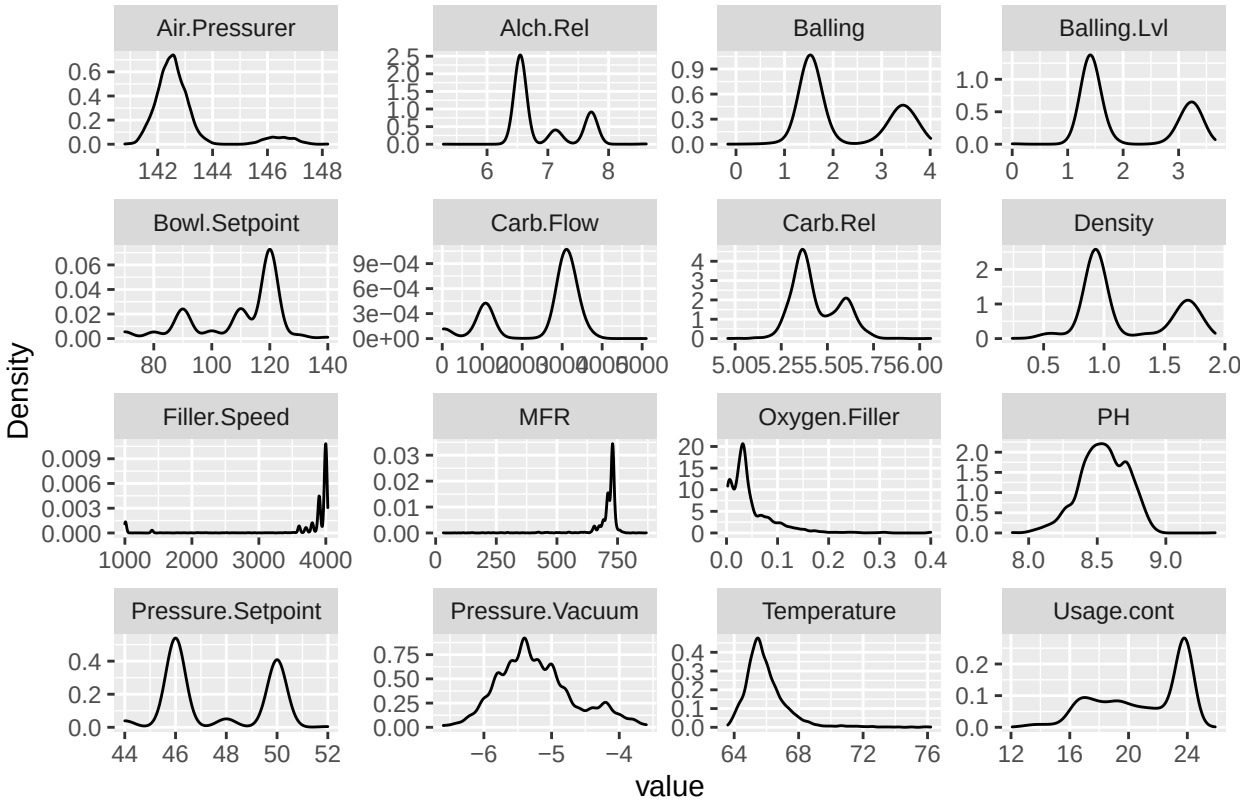

```
plot_histogram(studenttrainingdata)
```


Carb.Pressure
Carb.Pressure1
Carb.Temp
Carb.Volume
Fill.Ounces
Fill.Pressure
Filler.Level
Hyd.Pressure1
Hyd.Pressure2
Hyd.Pressure3
Hyd.Pressure4
Mnf.Flow
PC.Volume
PSC
PSC.CO2
PSC.Fill
Frequency
value



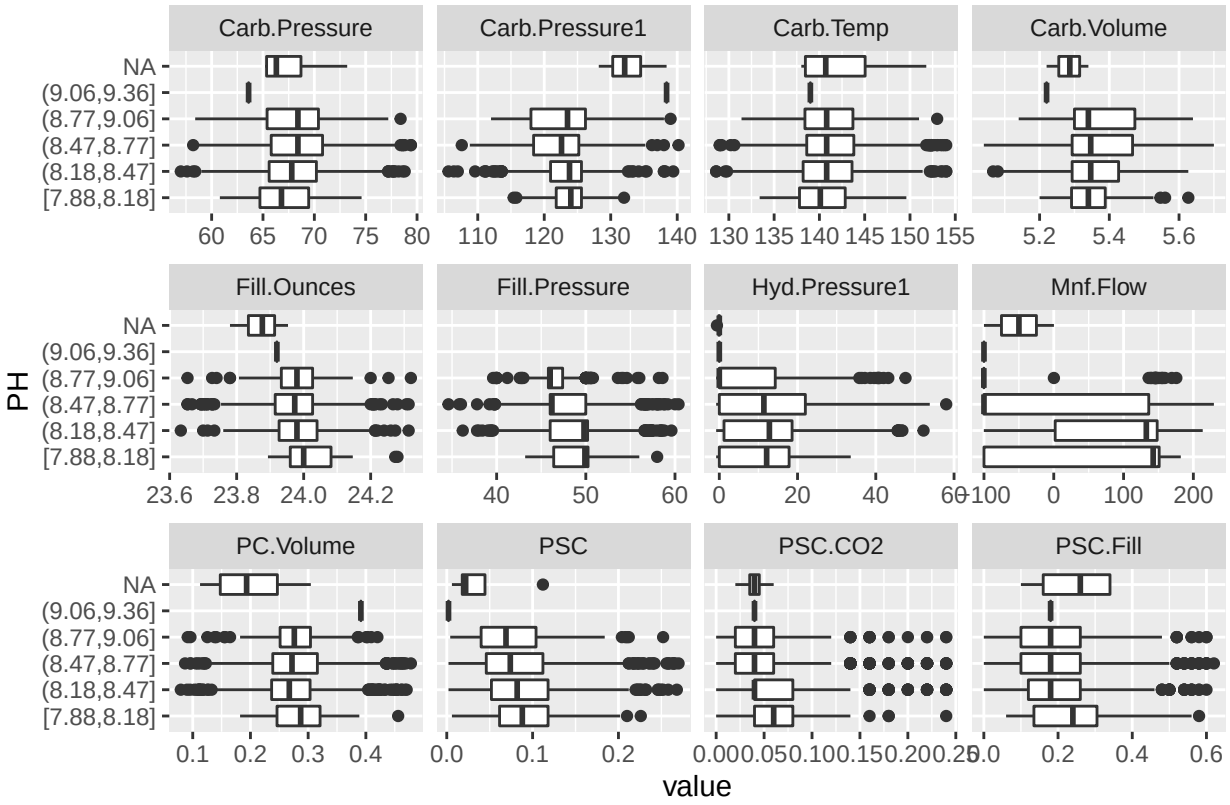

```
plot_density(studenttrainingdata)
```


Carb.Pressure
Carb.Pressure1
Carb.Temp
Carb.Volume
Fill.Ounces
Fill.Pressure
Filler.Level
Hyd.Pressure1
Hyd.Pressure2
Hyd.Pressure3
Hyd.Pressure4
Mnf.Flow
PC.Volume
PSC
PSC.CO2
PSC.Fill
Density
value

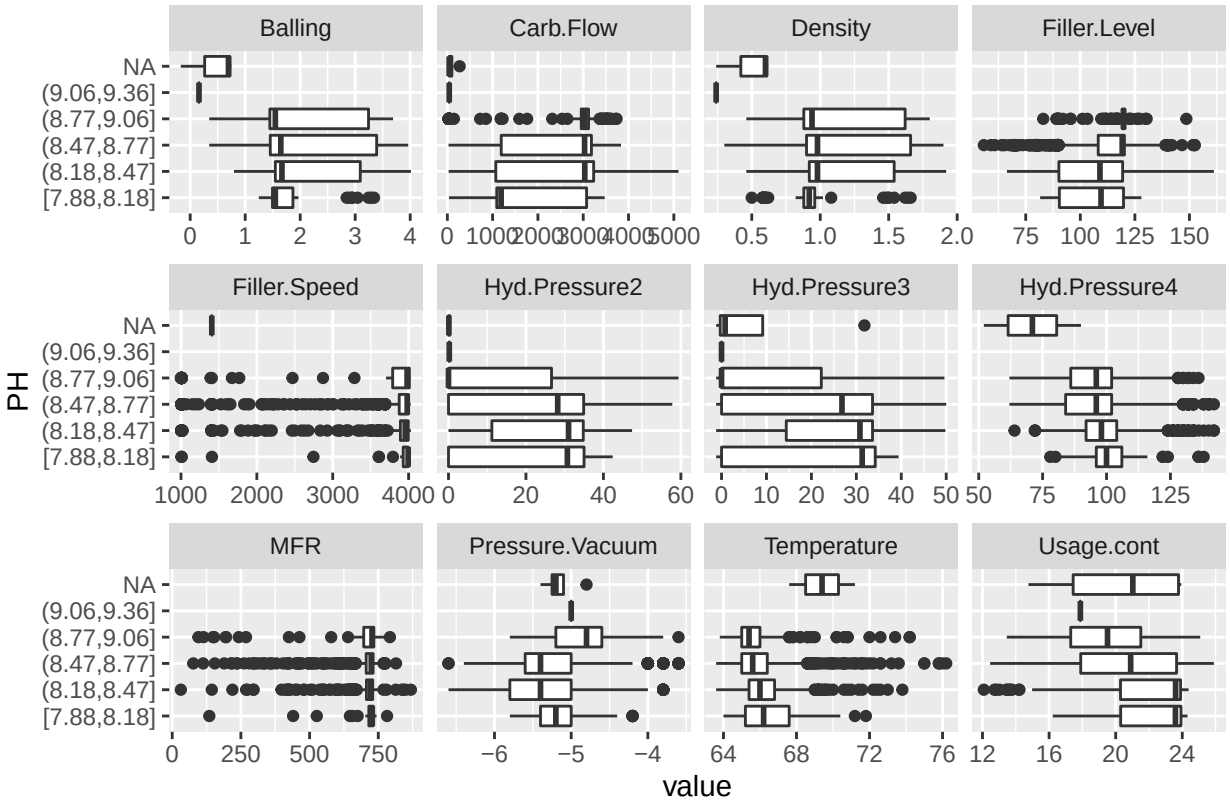



```
plot_boxplot(
  data = studenttrainingdata,
  by = "PH")
```

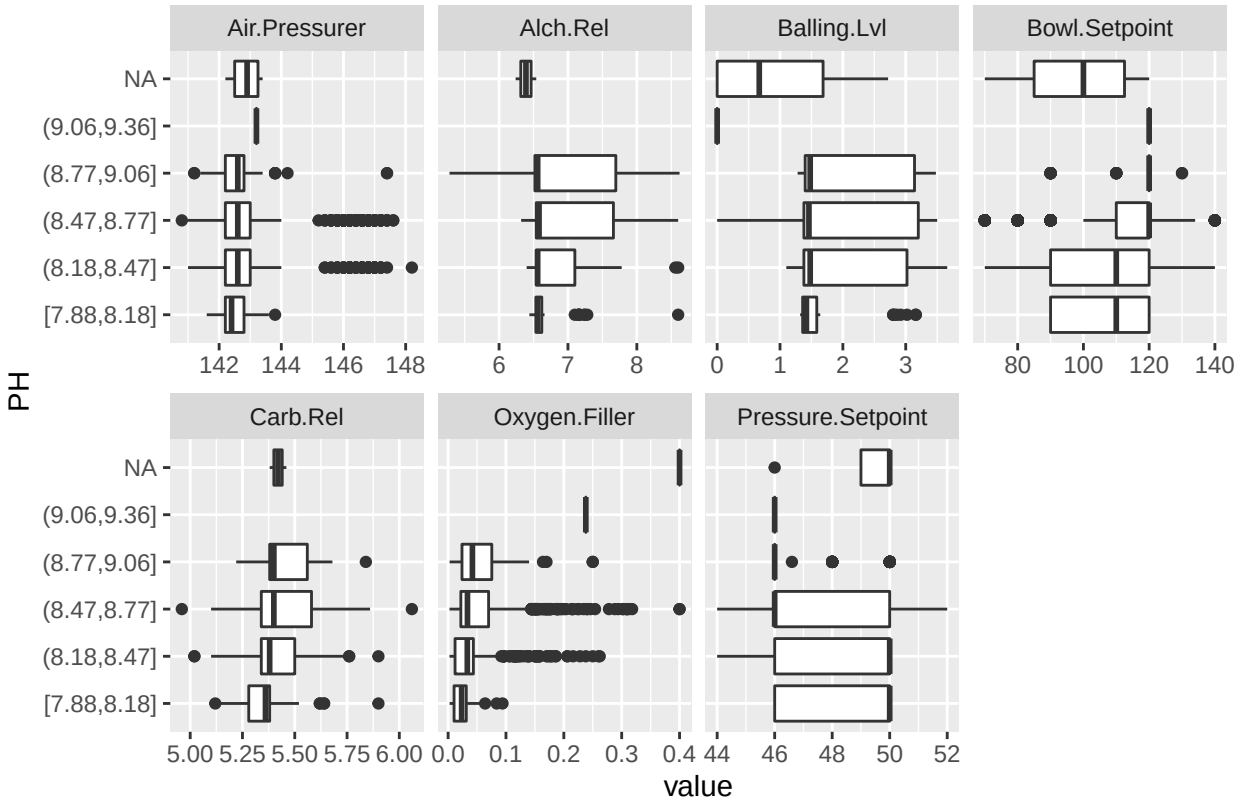
```
## Warning: Removed 302 rows containing non-finite values (stat_boxplot).
```




```
## Warning: Removed 372 rows containing non-finite values (stat_boxplot).
```




```
## Warning: Removed 46 rows containing non-finite values (stat_boxplot).
```




```
set.seed(100)

studentrainingdata <- studenttrainingdata[complete.cases(studenttrainingdata$PH),]
preprocess_data_model = preProcess(studenttrainingdata, c("center", "scale", "knnImpute", "corr", "nzv"
new_dataset = predict(preprocess_data_model,studenttrainingdata)
```

We will now split our data into a training set and validation set to analyze model outputs. Given the evaluation set we will need to verify our model analytics using RMSE, R^2, and MAE prior to running our predictions. Obtaining a model with a combination of the lowest RMSE, highest R^2, and lowest MAE will be optimal.

```
training_partition = createDataPartition(new_dataset$PH,p=.8,list=FALSE)

training_df = new_dataset[training_partition,]
validation_df = new_dataset[-training_partition,]
```

We will now build models using various linear, nonlinear, and tree based approaches.

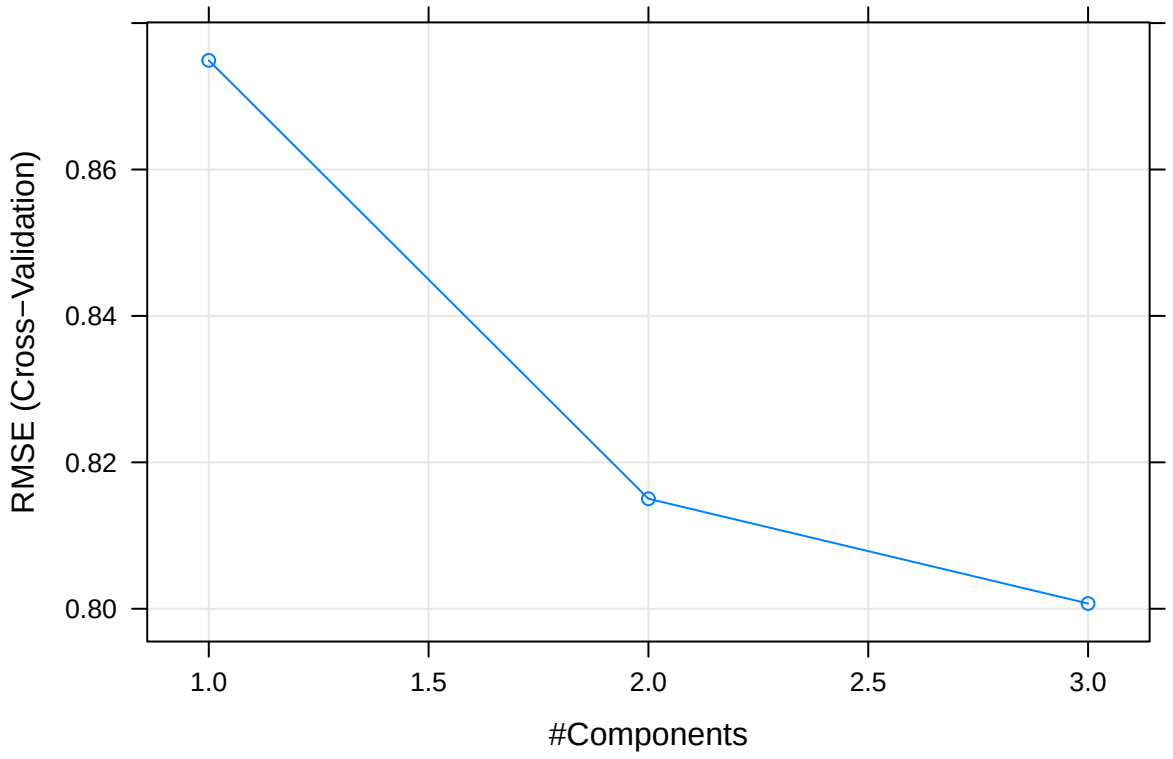
Linear Models

Ordinary Least Regression

```
olrmod = train(PH~.,data = training_df,method='lm',trControl=trainControl('cv',number=10))

olrpred = predict(olrmod,validation_df)
olrpred_results = postResample(pred = olrpred, obs = validation_df$PH)
```

Partial Least Squares

```
pls_mod = train(PH~.,data = training_df,method='pls',trControl=trainControl('cv',number=10),center=T,tu

plot(pls_mod)
```

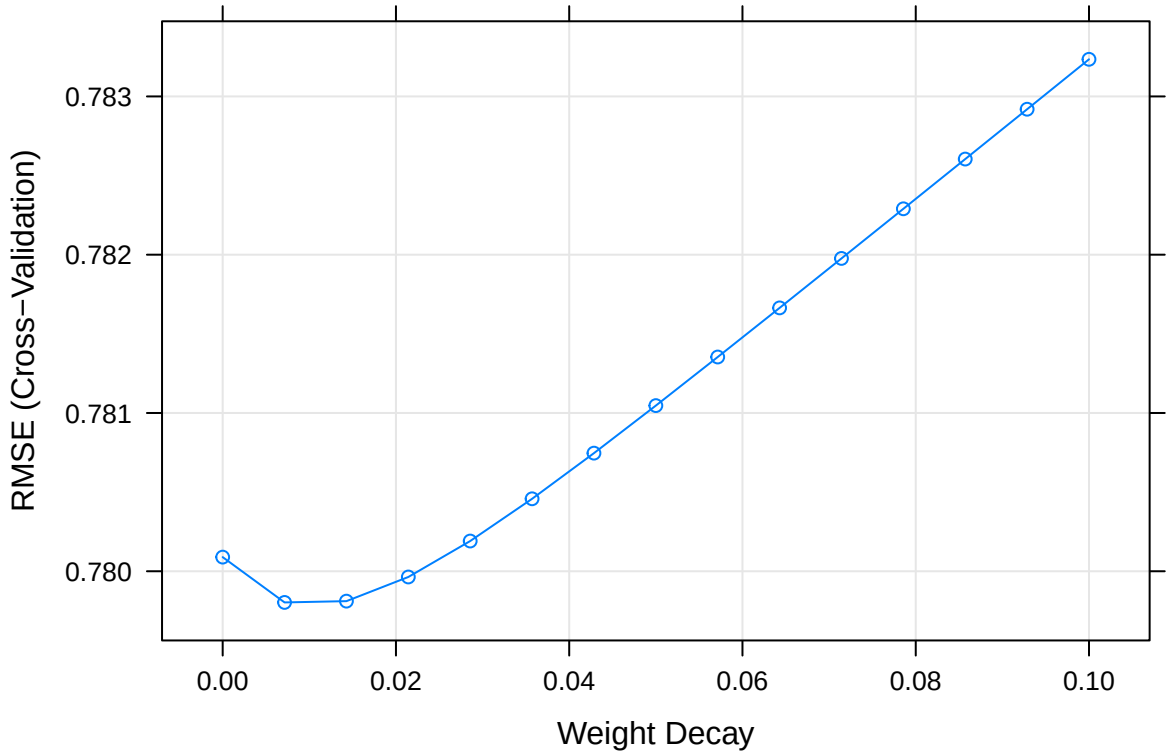



```
plspred = predict(pls_mod,validation_df)
plspred_results = postResample(pred = plspred, obs = validation_df$PH)
```

Ridge Regression

```
set.seed(100)

rrfit = train(PH~.,data=training_df,method='ridge',tuneGrid=data.frame(.lambda=seq(0,.1,length=15)),trC
```

```
plot(rrfit)
```




```
rrpred = predict(rrfit,validation_df)
rrpred_results = postResample(pred = rrpred, obs = validation_df$PH)
```

Non Linear Models

Neural Networks

```
set.seed(100)
nnetmod = train(PH~.,data=training_df,
                  method = "avNNet",
                  preProc = c("center", "scale"),
                  tuneGrid = expand.grid( .decay = c(0, 0.01, .1), .size = c(1:10), .bag= F ),
                  trControl = trainControl(method = "cv",number = 10),
                  linout = T,
                  trace= F,
                  MaxNWts = 5 * (ncol(training_df) + 1) + 5 + 1,
                  maxit = 500)

nnetpred = predict(nnetmod,validation_df)
nnetpred_results = postResample(pred = nnetpred, obs = validation_df$PH)
```

MARS

```
set.seed(100)
marsmod = train(PH~.,data=training_df,method='earth',trControl=trainControl(method='cv'))

marspred = predict(marsmod,validation_df)
marspred_results = postResample(pred = marspred, obs = validation_df$PH)
```

SVM

```
set.seed(100)
svmmod = train(PH~., data = training_df,
               method= "svmLinear",
               trControl = trainControl(method = "repeatedcv",number = 10, repeats=3),
               tuneLength = 10)

svmpred = predict(svmmod,validation_df)
svmpred_results = postResample(pred = svmpred, obs = validation_df$PH)
```

KNN

```
set.seed(100)
knnmod = train(PH~., data = training_df,
               method= "knn",
               trControl = trainControl(method = "repeatedcv",number = 10, repeats=3),
               tuneLength = 10)

knnpred = predict(knnmod,validation_df)
knnpred_results = postResample(pred = knnpred, obs = validation_df$PH)
```

Trees

Random Forest

```
randomforest_model = train(PH~.,data=training_df,method='rf',
                       preProcess = c('center', 'scale'),trControl=trainControl('cv'))

randomforest_pred = predict(randomforest_model, validation_df)
randomforest_results = postResample(pred = randomforest_pred, obs = validation_df$PH)
```

Boosted Trees

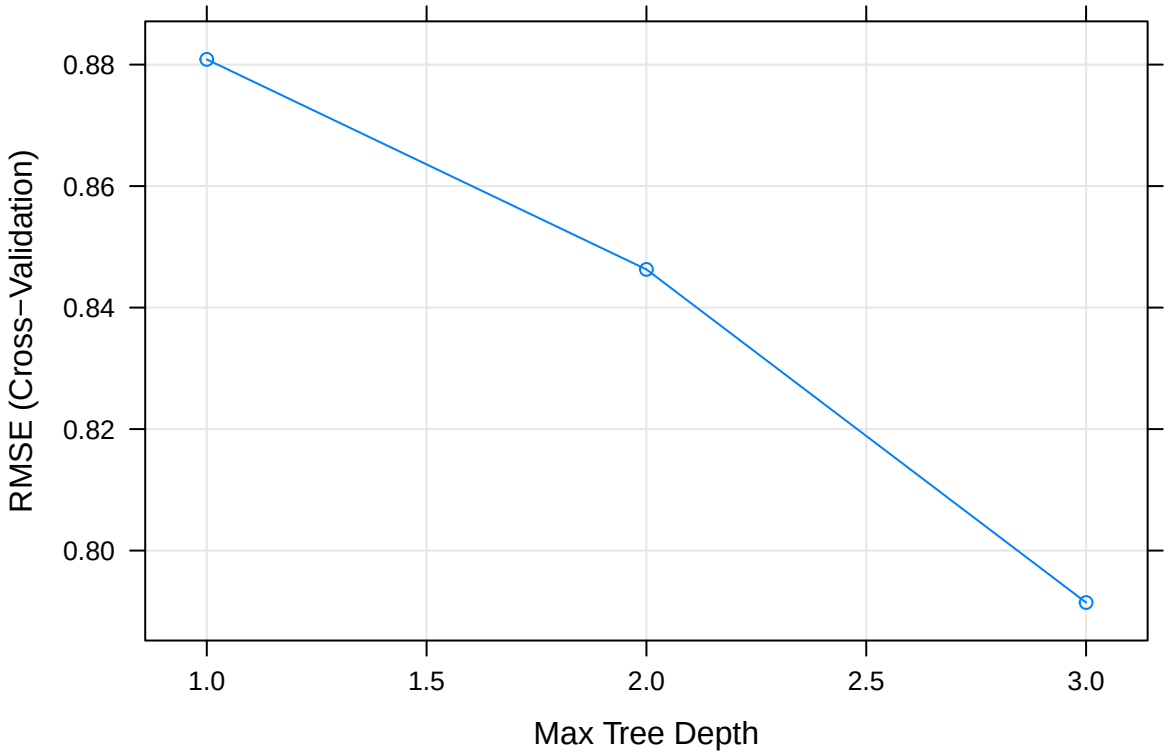
```
gbm_model = train(PH~.,data=training_df,method='gbm',tuneGrid=expand.grid(.interaction.depth = seq(1, 7
                        .n.trees = seq(100, 1000, by = 100),
                        .shrinkage = c(0.01, 0.1),
                        .n.minobsinnode = 8),
                       trControl=trainControl('cv'))

gbm_pred = predict(gbm_model, validation_df)
gbm_results = postResample(pred = gbm_pred, obs = validation_df$PH)
```

Cubist

```
cub_model = train(PH~.,data=training_df,method='cubist',
                       preProcess = c('center', 'scale'),trControl=trainControl('cv'))

cub_pred = predict(cub_model, validation_df)
cub_results = postResample(pred = cub_pred, obs = validation_df$PH)
```

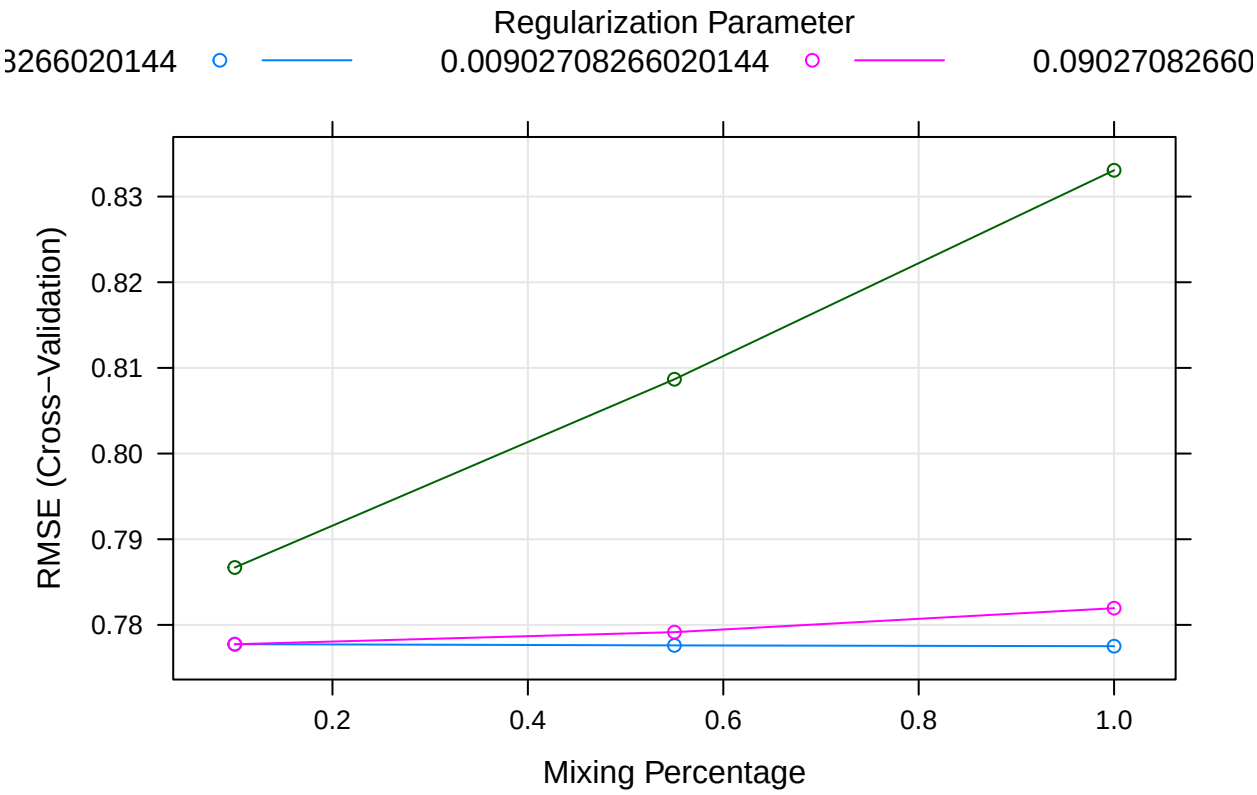
Single Tree

```
rp_model = train(PH~.,data=training_df,method='rpart2',
                       preProcess = c('center', 'scale'),trControl=trainControl('cv'))

rp_pred = predict(rp_model, validation_df)
rp_results = postResample(pred = rp_pred, obs = validation_df$PH)

plot(rp_model)
```



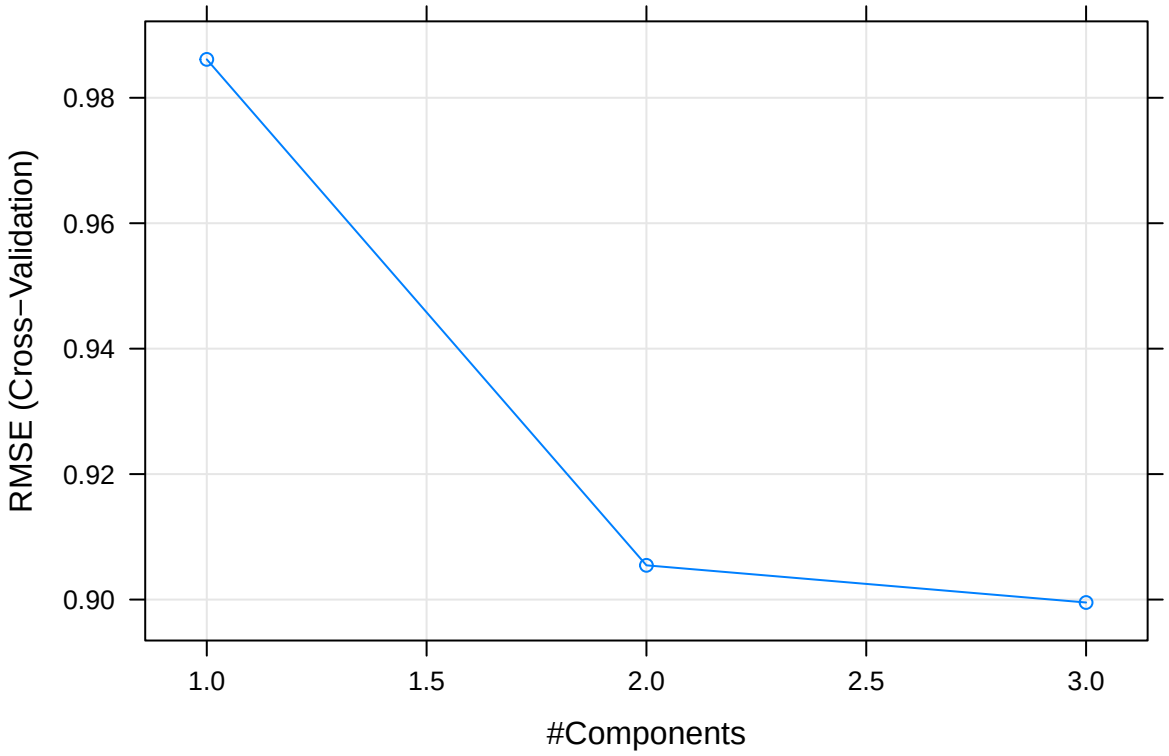
GLMNET

```
glm_model = train(PH~.,data=training_df,method='glmnet',
                        preProcess = c('center', 'scale'),trControl=trainControl('cv'))

glm_pred = predict(glm_model, validation_df)
glm_results = postResample(pred = glm_pred, obs = validation_df$PH)


plot(glm_model)
```



PCR

```
pcr_model = train(PH~.,data=training_df,method='pcr',
                        preProcess = c('center', 'scale'),trControl=trainControl('cv'))

pcr_pred = predict(pcr_model, validation_df)
pcr_results = postResample(pred = pcr_pred, obs = validation_df$PH)


plot(pcr_model)
```



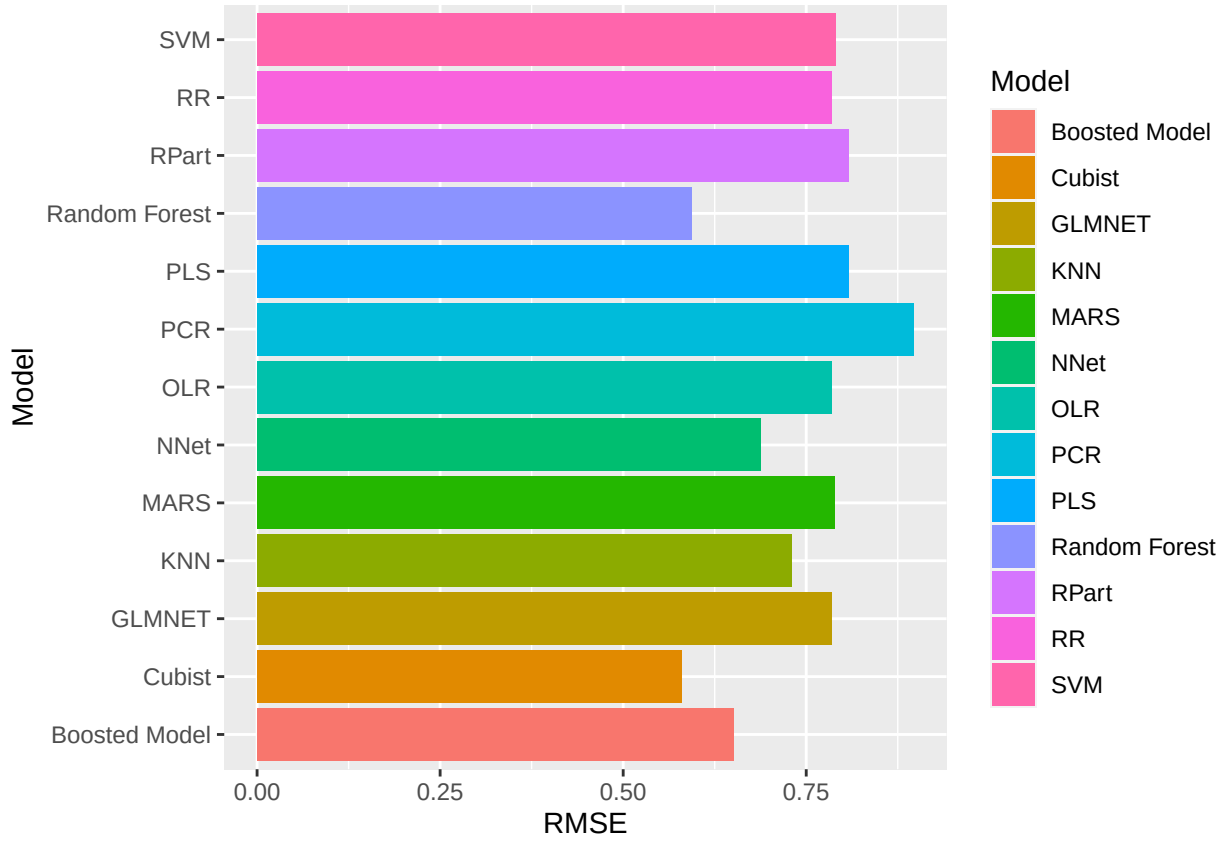
Evaluation

Aggregating the model results we see our cubist model performs the best among the 13 selected model types. The multi node linear model tree based methodology accounts best for the various intricacies of the data set. There is of course the risk of over fitting, however the cubist model we discerned to be best fit in this scenario.

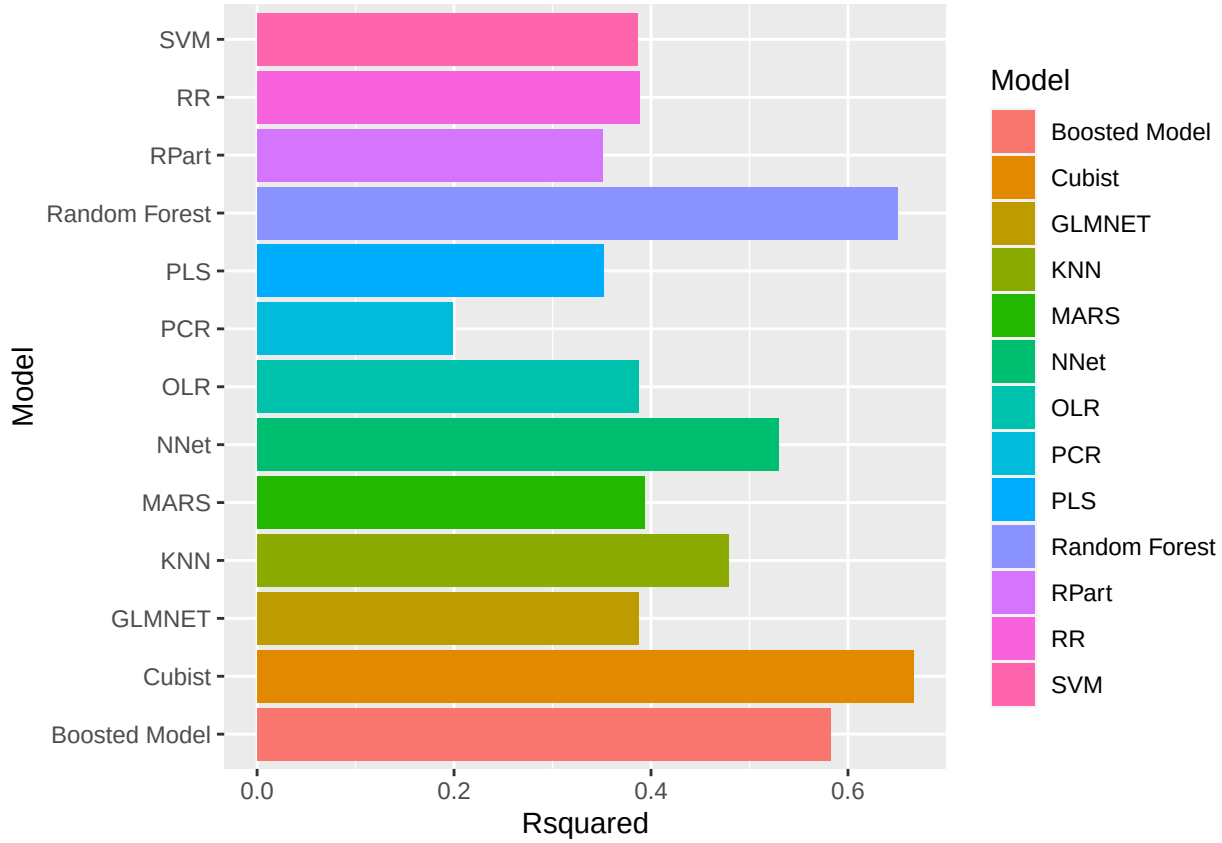
```
titles <- list("OLR","PLS","RR","NNet","MARS","SVM","KNN","Random Forest","Boosted Model","Cubist","RPa
results <- list(olrpred_results,plspred_results,rrpred_results,nnetpred_results,marspred_results,svmpre

df = NULL
for (x in 1:length(results)) {
  Model = titles[[x]]
  RMSE = unname(results[[x]]["RMSE"])
  Rsquared = unname(results[[x]]["Rsquared"])
  MAE = unname(results[[x]]["MAE"])

  df = rbind(df,data.frame(Model,RMSE,Rsquared,MAE))
}

knitr::kable(df[order(df$Rsquared,decreasing=TRUE),], digits=4, row.names=F)
```

Model	RMSE	Rsquared	MAE
Cubist	0.5796	0.6668	0.4223
Random Forest	0.5940	0.6508	0.4213
Boosted Model	0.6504	0.5822	0.4872
NNet	0.6872	0.5294	0.5075
KNN	0.7294	0.4790	0.5414
MARS	0.7886	0.3931	0.6009
RR	0.7845	0.3885	0.6027
GLMNET	0.7848	0.3878	0.6025
OLR	0.7852	0.3876	0.6025
SVM	0.7904	0.3862	0.5906
PLS	0.8074	0.3518	0.6176
RPart	0.8081	0.3507	0.6358
PCR	0.8969	0.1983	0.6952

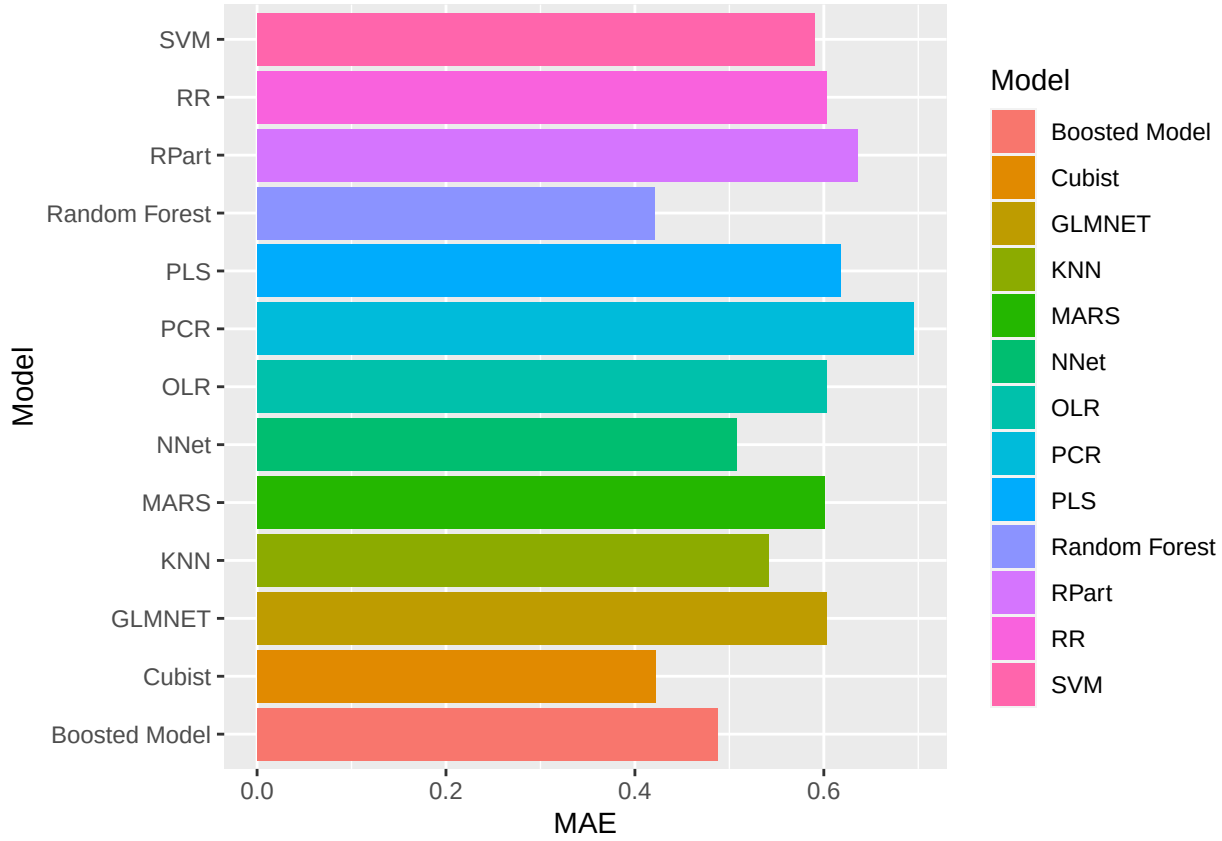
```
ggplot(data=df,aes(x=Model,y=RMSE)) +geom_bar(stat='identity',aes(fill=Model))+ coord_flip()
```




```
ggplot(data=df,aes(x=Model,y=Rsquared)) +geom_bar(stat='identity',aes(fill=Model))+ coord_flip()
```



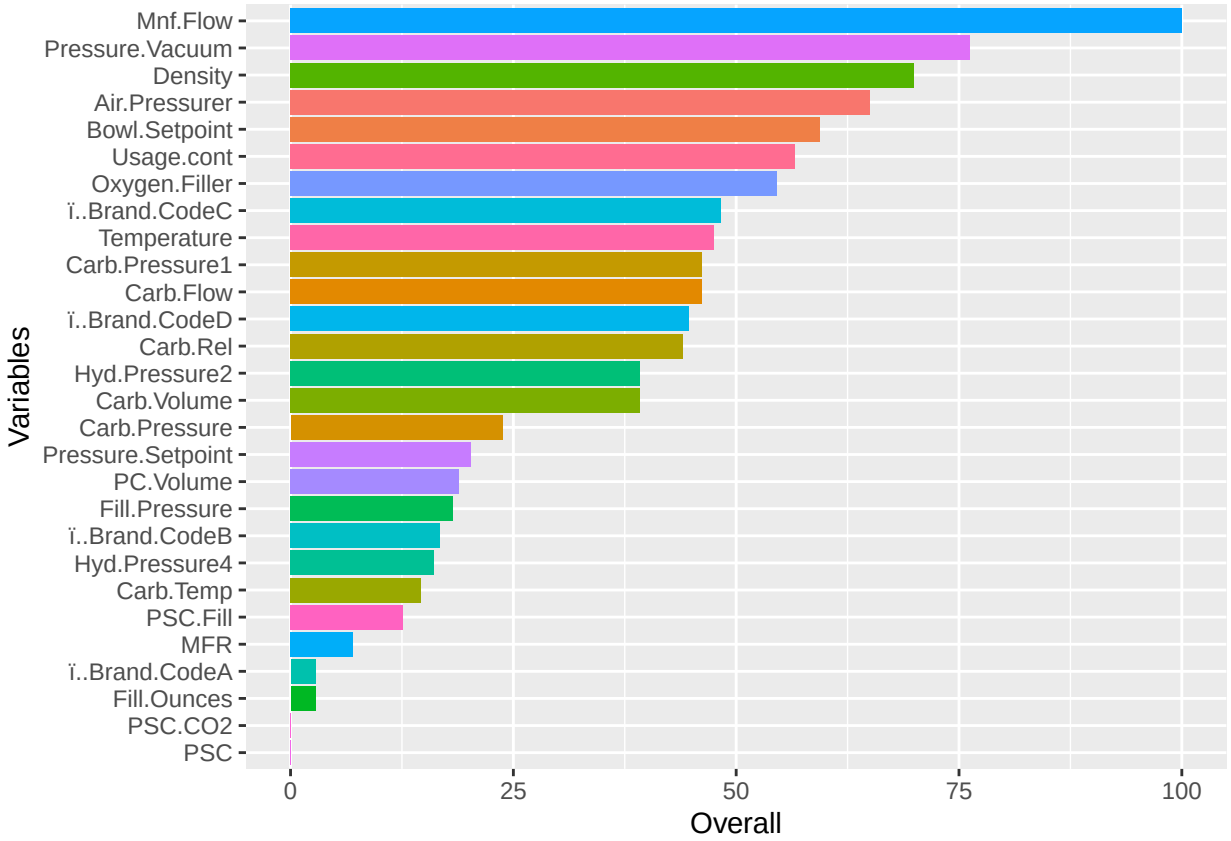

```
ggplot(data=df,aes(x=Model,y=MAE)) +geom_bar(stat='identity',aes(fill=Model))+ coord_flip()
```



Variable Importance

Using the cubist model we can discern how variables are weighted for predictor 'PH'. Mnf.Flow is the most significant variable in the model.

```
cub_var_imp = varImp(cub_model)

ggplot(data=cub_var_imp$importance,aes(x=reorder(rownames(cub_var_imp$importance), Overall),y=Overall))
```



Predictions

```
#studenteval_scaled = preProcess(studenttestdata, c("center", "scale","corr", "nzv"))
#studenteval_scaled_dataset = predict(studenteval_scaled,studenttestdata)
eval_cub_pred = predict(cub_model, studenttestdata)
scaled_ph = scale(studenttrainingdata$PH, center= TRUE, scale=TRUE)
scaledeval_predictions = eval_cub_pred * attr(scaled_ph, 'scaled:scale') +   attr(scaled_ph, 'scaled:ce
eval_df = data.frame(scaledeval_predictions)
#write.xlsx(eval_df, 'FinalProject_Predictions.xlsx')
eval_df
```

```
##     scaledeval_predictions
## 1                 8.635917
## 2                 8.735001
## 3                 8.737451
## 4                 8.637508
## 5                 8.645494
## 6                 8.735781
## 7                 8.723944
## 8                 8.723379
## 9                 8.348160
## 10                8.738002
## 11                8.727838
```

```
## 12              8.568844
## 13              8.727591
## 14              8.727823
## 15              8.750464
## 16              8.730200
## 17              8.341952
## 18              8.392659
## 19              8.576176
## 20              8.577821
## 21              8.573695
## 22              8.638799
## 23              8.647669
## 24              8.647156
## 25              8.567853
## 26              8.611181
## 27              8.437625
## 28              8.340116
## 29              8.707129
## 30              8.564879
## 31              8.563874
## 32              8.719247
## 33              8.621498
## 34              8.631879
## 35              8.730012
## 36              8.718423
## 37              8.565087
## 38              8.375747
## 39              8.427281
## 40              8.717648
## 41              8.725513
## 42              8.739523
## 43              8.720473
## 44              8.575594
## 45              8.742187
## 46              8.745922
## 47              8.725332
## 48              8.708360
## 49              8.710697
## 50              8.704257
## 51              8.709319
## 52              8.397345
## 53              8.361605
## 54              8.562620
## 55              8.560742
## 56              8.515244
## 57              8.515117
## 58              8.704040
## 59              8.475468
## 60              8.486117
## 61              8.621817
## 62              8.386630
## 63              8.444381
## 64              8.399710
## 65              8.472886
```

```
## 66              8.609892
## 67              8.546214
## 68              8.620893
## 69              8.788414
## 70              8.609068
## 71              8.611928
## 72              8.536005
## 73              8.472686
## 74              8.611984
## 75              8.618422
## 76              8.636980
## 77              8.636552
## 78              8.369666
## 79              8.328993
## 80              8.342468
## 81              8.602911
## 82              8.603445
## 83              8.493164
## 84              8.468000
## 85              8.608874
## 86              8.609155
## 87              9.181125
## 88              8.615038
## 89              8.617625
## 90              8.606761
## 91              8.606078
## 92              8.459819
## 93              8.459012
## 94              8.367888
## 95              8.371780
## 96              8.499194
## 97              8.499194
## 98              8.365715
## 99              8.372174
## 100             8.454131
## 101             8.455946
## 102             8.455195
## 103             8.453737
## 104             8.362333
## 105             8.362506
## 106             8.362110
## 107             8.452125
## 108             8.453901
## 109             8.453989
## 110             8.457717
## 111             8.374673
## 112             8.458561
## 113             8.386446
## 114             8.385490
## 115             8.385681
## 116             8.459448
## 117             8.459533
## 118             8.457453
## 119             8.459406
```

```
## 120              8.461402
## 121              8.461217
## 122              8.374438
## 123              8.499194
## 124              8.442812
## 125              8.499194
## 126              8.457124
## 127              8.457752
## 128              8.457264
## 129              8.457434
## 130              8.457087
## 131              8.459711
## 132              8.460958
## 133              8.461122
## 134              8.499194
## 135              8.459805
## 136              8.460014
## 137              8.380924
## 138              8.383102
## 139              8.331194
## 140              8.455692
## 141              8.366565
## 142              8.370239
## 143              8.459171
## 144              8.331194
## 145              8.378673
## 146              8.452763
## 147              8.369962
## 148              8.371750
## 149              8.371393
## 150              8.459524
## 151              8.423942
## 152              8.331194
## 153              8.353945
## 154              8.449094
## 155              8.499194
## 156              8.499194
## 157              8.366175
## 158              8.367188
## 159              8.257988
## 160              8.255486
## 161              8.254270
## 162              8.454047
## 163              8.367345
## 164              8.368510
## 165              8.454459
## 166              8.458114
## 167              8.499194
## 168              8.499194
## 169              8.453854
## 170              8.454039
## 171              8.455789
## 172              8.257440
## 173              8.499194
```

```
## 174              8.368085
## 175              8.459062
## 176              8.457748
## 177              8.457746
## 178              8.460351
## 179              8.461423
## 180              8.499194
## 181              8.499194
## 182              8.375185
## 183              8.457862
## 184              8.454635
## 185              8.455374
## 186              8.455436
## 187              8.459276
## 188              8.499194
## 189              8.373768
## 190              8.463685
## 191              8.463144
## 192              8.462815
## 193              8.499194
## 194              8.452421
## 195              8.461617
## 196              8.499194
## 197              8.499194
## 198              8.372388
## 199              8.499194
## 200              8.456342
## 201              8.453523
## 202              8.499194
## 203              8.499194
## 204              8.453058
## 205              8.499194
## 206              8.251489
## 207              8.254582
## 208              8.257047
```