Project 2

Draft

Group 2

12/10/2019

Getting Started

Overview

Include details on our process in creating this document.

Dependencies

```
# Predicitve Modeling
libraries("AppliedPredictiveModeling", "mice", "caret", "tidyverse",
    "impute", "pls", "caTools", "mlbench", "car", "olsrr", "neuralnet",
    "nnet", "earth")
# Formatting Libraries
libraries("default", "knitr", "kableExtra", "readxl", "sqldf",
    "sjPlot", "sjmisc", "sjlabelled", "MASS")
# Plotting Libraries
libraries("ggplot2", "grid", "ggfortify", "DataExplorer", "ggcorrplot")
```

Project 2

\begin{question}{Project #2 - Requirement}You are given a simple data set from a beverage manufacturing company. It consists of 2,571 rows/cases of data and 33 columns / variables. Your goal is to use this data to predict PH (a column in the set). PH is a measure of acidity/alkalinity, it must conform in a critical range and therefore it is important to understand its influence and predict its values. This is production data. PH is a KPI, Key Performance Indicator.

You are also given a scoring set (267 cases). All variables other than the dependent or target. You will use this data to score your model with your best predictions. :\end{question}

```
df <- read_excel("C:/Users/traveler/Documents/GitHub/CUNY_DATA_624/Project_Two/data/StudentData.xlsx")
# df <-
# read_excel('~/GitHub/CUNY_DATA_624/Project_Two/data/StudentData.xlsx')
df_eval <- read_excel("C:/Users/traveler/Documents/GitHub/CUNY_DATA_624/Project_Two/data/StudentEvaluat
# df_eval <-
# read_excel('~/GitHub/CUNY_DATA_624/Project_Two/data/StudentEvaluation.xlsx')
dict <- read_excel("C:/Users/traveler/Documents/GitHub/CUNY_DATA_624/Project_Two/data/DataDictionary.xls
# dict <-
# read_excel('~/GitHub/CUNY_DATA_624/Project_Two/data/DataDictionary.xlsx')
# remove space in-between variable names
colnames(df) <- gsub(" ", "", colnames(df))
```

Introduction

The goal of this project is to predict `PH`, a measure of acidity/alkalinity, using train data set from a beverage company which consists of 2571 rows of data and 33 variables. After creating models based on training data, we will test on scoring set of 267 rows with 32 variables (excluding target variable which is `PH` in our training set)

As a group project, each member of the group is responsible for creating their own models of choice. For instance, my own selections were `Linear model` and `NNet`. However, the choice of models can be altered after careful review of data exploration - it may require different type of model in case data suffers from outliers or any other data related issues.

Explaining why some necessary steps were applied before modeling and model A was preferred to Model B is often a topic in academic papers which is a meaningful topic that helps audience learn the concept of bagged regression and least square method better.

The final version of report will contain all of our approaches with results of `MAPE` for each model with detailed explanation of why/what/how each model of choice was chosen.

NNet, linear model, and one of your choice

Data Exploration

Data dictionary

The table below describes the variables in the train data set.

```
kable(dict, caption = "Data dictionary", booktabs = T) %>% kable_styling() %>%
    row_spec()
```

Table 1: Data dictionary

Name	Type	MD code	Length	Measurement Type	Excluded
Test Time	Double	-9999	8	m/d/yy h:mm Am/Pm	Auto
Brand Code	Double	-9999	8	General	Categorical
Carb Volume	Double	-9999	8	General	Auto
Fill Ounces	Double	-9999	8	General	Auto
PC Volume	Double	-9999	8	General	Auto
Carb Pressure	Double	-9999	8	General	Auto
Carb Temp	Double	-9999	8	General	Auto
PSC	Double	-9999	8	General	Auto
PSC Fill	Double	-9999	8	General	Auto
PSC CO2	Double	-9999	8	General	Auto
Mnf Flow	Double	-9999	8	General	Auto
Carb Pressure1	Double	-9999	8	General	Auto
Fill Pressure	Double	-9999	8	General	Auto
Hyd Pressure1	Double	-9999	8	General	Auto
Hyd Pressure2	Double	-9999	8	General	Auto
Hyd Pressure3	Double	-9999	8	General	Auto
Hyd Pressure4	Double	-9999	8	General	Auto
Filler Level	Double	-9999	8	General	Auto
Filler Speed	Double	-9999	8	General	Auto
Temperature	Double	-9999	8	General	Auto
Usage cont	Double	-9999	8	General	Auto
Carb Flow	Double	-9999	8	General	Auto
Density	Double	-9999	8	General	Auto
MFR	Double	-9999	8	General	Auto
Balling	Double	-9999	8	General	Auto
Pressure Vacuum	Double	-9999	8	General	Auto
PH	Double	-9999	8	General	Auto
Oxygen Filler	Double	-9999	8	General	Auto
Bowl Setpoint	Double	-9999	8	General	Auto
Pressure Setpoint	Double	-9999	8	General	Auto
Air Pressurer	Double	-9999	8	General	Auto
Alch Rel	Double	-9999	8	General	Auto
Carb Rel	Double	-9999	8	General	Auto
Balling Lvl	Double	-9999	8	General	Auto
sample	Double	-999999998	8	General	Auto

Before we begin to try models against our data, we need to get an understanding of what our data looks like. We want to know what variables contain missing data. We also want to see the distributions of our target variable in addition to predictors. We need to establish a direction when it comes to pre-processing methods such as imputation, variable removal, and transformations. As a first step, we want to see the percentage of missing data per variable.

```
# code
round((apply(df, 2, function(col) sum(is.na(col))/length(col))) *
    100, 2) %>% kable(caption = "Distribution of Missing data in Beverage Manufacturing Company Data") %
    kable_styling(full_width = F, position = "center") %>% row_spec()
```

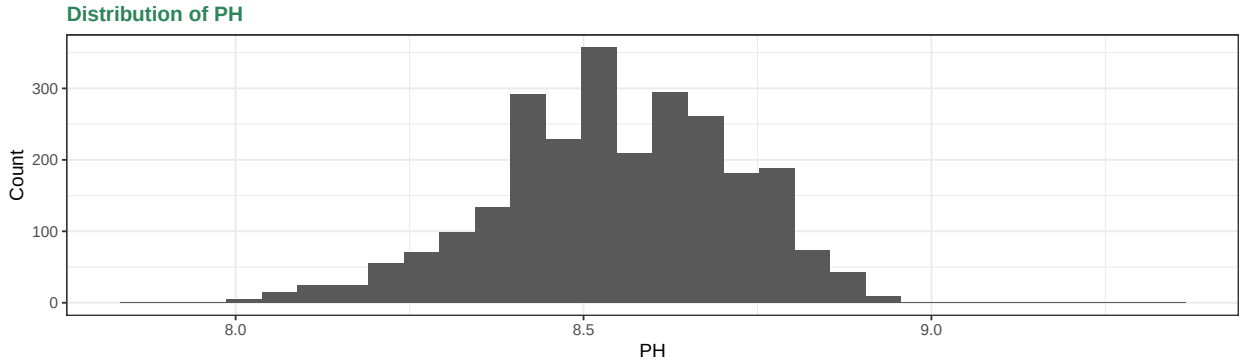
Table 2: Distribution of Missing data in Beverage Manufacturing Company Data

	x
BrandCode	4.67
CarbVolume	0.39
FillOunces	1.48
PCVolume	1.52
CarbPressure	1.05
CarbTemp	1.01
PSC	1.28
PSCFill	0.89
PSCCO2	1.52
MnfFlow	0.08
CarbPressure1	1.24
FillPressure	0.86
HydPressure1	0.43
HydPressure2	0.58
HydPressure3	0.58
HydPressure4	1.17
FillerLevel	0.78
FillerSpeed	2.22
Temperature	0.54
Usagecont	0.19
CarbFlow	0.08
Density	0.04
MFR	8.25
Balling	0.04
PressureVacuum	0.00
PH	0.16
OxygenFiller	0.47
BowlSetpoint	0.08
PressureSetpoint	0.47
AirPressurer	0.00
AlchRel	0.35
CarbRel	0.39
BallingLvl	0.04

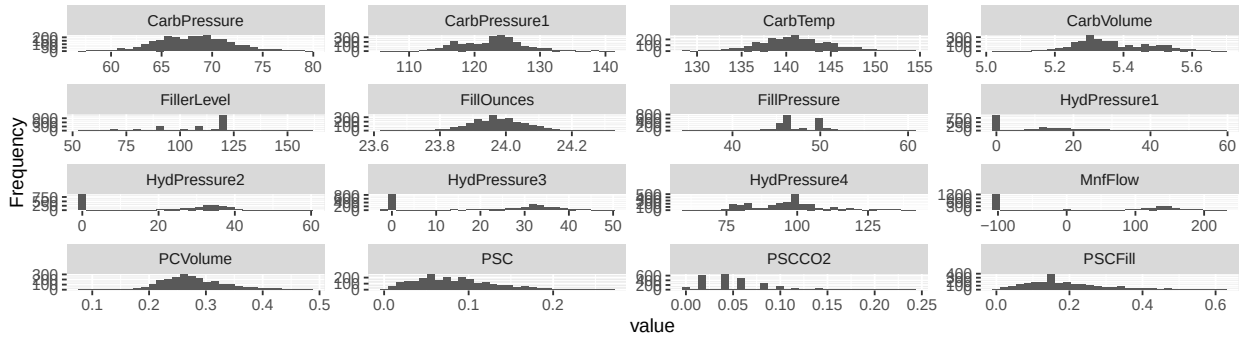
MFR is missing 8.25% of its data followed by BrandCode,which is missing a little under 5% of its data. Brandcode isa categorical variable. The rest of our predictors are missing less than 3% of their data if not missing any at all. We can deal with these variables using a method of imputation. From our previous experience, we have used MICE succesfully as an imputation method. We have previously found that there

does not seem to be major changes in the summary statistics across different imputation methods such as KNN or mean/mode. We also see that we are missing a small amount of data in the target variable (PH). PH is our ground truth, therefore it is better to remove rows where ground truth is missing without major data loss. We next examine the distribution of our response and predictor variables.

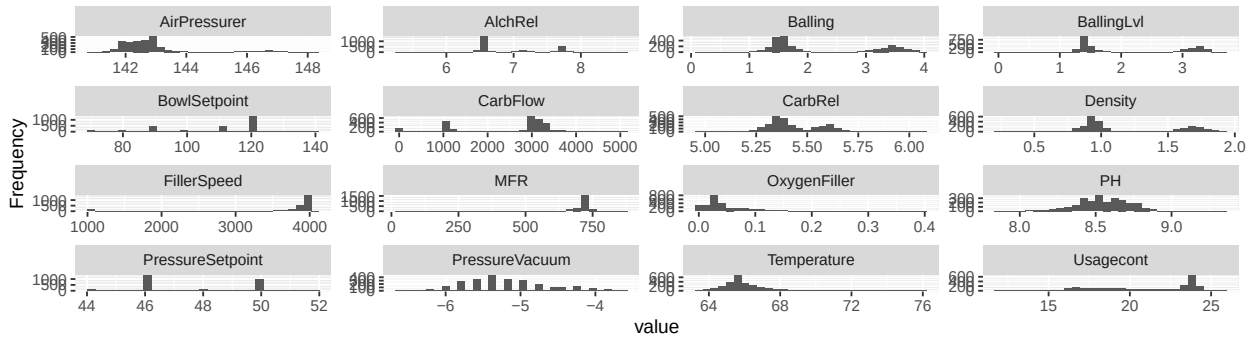
```
# drop rows where PH is missing
df <- sqldf("select
*
from df
where PH is not null
      ")


df %>% ggplot(aes(PH)) + geom_histogram(bins = 30) + theme_bw() +
    theme(legend.position = "center") + labs(y = "Count", title = "Distribution of PH")
```




```
plot_histogram(df)
```



Page 1



Page 2

We observe evidence of outliers in addition to skew such as in the varibales Temperature. Based on the information we have gathered from our EDA, we can move into the data processing phase where we will address our findings. We will not know the true effect of outliers until we build and diagnose our models. We aniticipate linear model to perform worse than NNET.

Data Preperation

We will use the MICE method to impute missing variables. We will also evaluate the which predictors have near zero variance. This will help us identify pedictors that need to be dropped from the data frame all together.We will hold off on removing predictors that have strong pairwise correlation until after we see how it affects our models. Andy has written code that nicely completes the data processing steps.

```
# set seed for split to allow for reproducibility
set.seed(58677)
# use mice w/ default settings to impute missing data
miceImput <- mice(df, printFlag = FALSE)

# add imputed data to original data set
df_mice <- mice::complete(miceImput)
df_mice$BrandCode[is.na(df_mice$BrandCode)] <- "B"

zero_cols <- nearZeroVar(df_mice)
df_final <- df_mice[, -zero_cols]  # drop these zero variance columns
df_final$BrandCode <- as.factor(df_final$BrandCode)

# convert categorical factor into numeric
M <- df_final
must_convert <- sapply(M, is.factor)  # logical vector telling if a variable needs to be displayed as n
BrandNumeric <- sapply(M[, must_convert], unclass)  # data.frame of all categorical variables now displ
df_final2 <- cbind(M[, !must_convert], BrandNumeric)  # complete data.frame with all variables put toge

# split data train/test
training <- df_final$PH %>% createDataPartition(p = 0.8, list = FALSE)
df_train <- df_final[training, ]
df_test <- df_final[-training, ]
```

Modeling

Linear Models

Linear Model: The first model we will be building is the classical linear model. The linear model represents the taret variable as a function of its predictor variables. Beta represents the linear parameter estimate and epsilon represents the error term. $y = \beta_0 + \sum \beta_i X_i + \epsilon_i$

There are some assumptions that need to be followed regarding linear regression. The first is that the residuals need to follow a near normal distribution and the second is that variance has to be constant. We have some diagnostic tests that can help us determine if these assumptions are satisfied or not. These assumptions need to be checked before we even consider taking into account how the model fits.

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

lm1 <- lm(PH ~ ., data = df_train)

# tab_model(lm1)

# summary(lm1)

# plot(lm1)
```

Lets examine some metrics against this model. The first is to compute VIF numbers, also known as variance inflation numbers. The VIF numbers will also provide direction into which variables to remove from future iterations of the linear model. We typically remove predictors with a VIF bigger than 4. In this case, we

remove CarbPressurem CarbTemp, Balling, AlchRel, and BallingLvl. https://www.researchgate.net/post/Multicollinearity_issues_is_a_value_less_than_10_acceptable_for_VIF

```
vif(lm1) %>% kable(caption = "Linear Model 1 Variance Inflation Numbers") %>%
    kable_styling(full_width = F, position = "center") %>% row_spec()
```

Table 3: Linear Model 1 Variance Inflation Numbers

	GVIF	Df	GVIF^(1/(2*Df))
BrandCode	53.571866	3	1.941584
CarbVolume	13.634820	1	3.692536
FillOunces	1.160774	1	1.077392
PCVolume	1.502752	1	1.225868
CarbPressure	32.477299	1	5.698886
CarbTemp	26.671687	1	5.164464
PSC	1.153714	1	1.074111
PSCFill	1.100332	1	1.048967
PSCCO2	1.071535	1	1.035150
MnfFlow	4.389162	1	2.095033
CarbPressure1	1.572528	1	1.254005
FillPressure	2.074542	1	1.440327
HydPressure2	9.197523	1	3.032742
HydPressure3	12.507241	1	3.536558
HydPressure4	2.650557	1	1.628053
FillerLevel	11.055076	1	3.324918
FillerSpeed	11.544613	1	3.397737
Temperature	1.400337	1	1.183358
Usagecont	1.703794	1	1.305295
CarbFlow	2.356651	1	1.535139
Density	16.299332	1	4.037243
MFR	9.832086	1	3.135616
Balling	71.779454	1	8.472276
PressureVacuum	2.685955	1	1.638888
OxygenFiller	1.522703	1	1.233979
BowlSetpoint	11.584124	1	3.403546
PressureSetpoint	2.268053	1	1.506006
AirPressurer	1.184406	1	1.088304
AlchRel	23.237818	1	4.820562
CarbRel	5.258863	1	2.293221
BallingLvl	54.441289	1	7.378434

We also want to see if outliers had any significant effect on our model by performing a significance test using Bonferroni statistics. According to our test, the low p value suggests that outliers are not significant and we should not go through the effort to treat outliers.

```
outlierTest(lm1)
```

```
     rstudent unadjusted p-value Bonferroni p
1093  5.85526         5.5454e-09   1.1396e-05
```

We fit a new LM with features showing high VIF removed.

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

```
lm2 <- lm(PH ~ . - BallingLvl - AlchRel - Balling - CarbTemp -
    CarbPressure, data = df_train)

# tab_model(lm2)
```

The adjusted r squared without the high VIF numbers remains unchanged,hence we were able to simplify our model without a change in data variability capture. We can furthur remove predictors by calculating variable importance.

```
lm2_imp <- varImp(lm2, scale = TRUE)

lm2_imp %>% kable(caption = "Linear Model 2 Variable Importance") %>%
    kable_styling() %>% row_spec()
```

Table 4: Linear Model 2 Variable Importance

	Overall
BrandCodeB	0.6751499
BrandCodeC	5.8589596
BrandCodeD	6.5503953
CarbVolume	2.2600273
FillOunces	2.1716942
PCVolume	1.5224876
PSC	1.5731511
PSCFill	1.2302135
PSCCO2	2.7008473
MnfFlow	13.9429622
CarbPressure1	7.5918084
FillPressure	2.0097608
HydPressure2	1.4157779
HydPressure3	4.6336202
HydPressure4	0.7423533
FillerLevel	1.9066804
FillerSpeed	2.6517876
Temperature	6.2404708
Usagecont	5.7947060
CarbFlow	1.6203759
Density	3.5540669
MFR	3.5678464
PressureVacuum	0.1724955
OxygenFiller	3.8880210
BowlSetpoint	4.7217460
PressureSetpoint	4.3666264
AirPressurer	0.8192357
CarbRel	2.8073009

```
# plot(lm2_imp)
```

BrandCode, PSCCO2, FillerSpeed, MFR,and PressureVacuum are flagged as the least important variables. We will remove these from the overall linear model and then compare how each of the three linear models does against each other.

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

lm3 <- lm(PH ~ . - BallingLvl - AlchRel - Balling - BrandCode -
    CarbTemp - CarbPressure - PSCCO2 - FillerSpeed - MFR - PressureVacuum,
    data = df_train)

tab_model(lm1, lm2, lm3)
```

PH

PH

PH

Predictors

Estimates

CI

p

Estimates

CI

p

Estimates

CI

p

(Intercept)

11.36

9.01 – 13.71

<0.001

11.32

9.42 – 13.22

<0.001

11.75

9.76 – 13.74

<0.001

BrandCode [B]

0.07

0.02 – 0.12

0.005

0.01

-0.02 – 0.05

0.500

BrandCode [C]

-0.07

-0.11 – -0.02

0.007

-0.12

-0.16 – -0.08

<0.001

BrandCode [D]

0.04

0.00 – 0.08

0.033

0.09

0.06 – 0.11

<0.001

CarbVolume

-0.15

-0.35 – 0.05

0.134

-0.12

-0.23 – -0.02

0.024

0.02

-0.09 – 0.13

0.733

FillOunces

-0.08

-0.15 – -0.02

0.017

-0.08

-0.15 – -0.01

0.030

-0.13

-0.20 – -0.06

0.001

PCVolume

-0.08

-0.20 – 0.03

0.145

-0.09

-0.20 – 0.03

0.128

-0.08

-0.20 – 0.04

0.193

CarbPressure

0.00

-0.00 – 0.01

0.364

CarbTemp

-0.00

-0.01 – 0.00

0.495

PSC

-0.11

-0.24 – 0.01

0.082

-0.10

-0.23 – 0.03

0.116

-0.13

-0.26 – 0.01

0.065

PSCFill

-0.03

-0.08 – 0.02

0.213

-0.03

-0.08 – 0.02

0.219

-0.06

-0.11 – -0.00

0.035

PSCCO2

-0.17

-0.31 – -0.03

0.015

-0.19

-0.33 – -0.05

0.007

MnfFlow

-0.00

-0.00 – -0.00

<0.001

-0.00

-0.00 – -0.00

<0.001

-0.00

-0.00 – -0.00

<0.001

CarbPressure1

0.01

0.00 – 0.01

<0.001

0.01

0.00 – 0.01

<0.001

0.01

0.00 – 0.01

<0.001

FillPressure

0.00

-0.00 – 0.00

0.091

0.00

0.00 – 0.01

0.045

0.00

-0.00 – 0.01

0.068

HydPressure2

-0.00

-0.00 – 0.00

0.163

-0.00

-0.00 – 0.00

0.157

-0.00

-0.00 – 0.00

0.076

HydPressure3

0.00

0.00 – 0.00

<0.001

0.00

0.00 – 0.00

<0.001

0.00

0.00 – 0.00

<0.001

HydPressure4

0.00

-0.00 – 0.00

0.673

0.00

-0.00 – 0.00

0.458

-0.00

-0.00 – 0.00

0.190

FillerLevel

-0.00

-0.00 – 0.00

0.058

-0.00

-0.00 – 0.00

0.057

-0.00

-0.00 – 0.00

0.265

FillerSpeed

0.00

0.00 – 0.00

0.003

0.00

0.00 – 0.00

0.008

Temperature

-0.01

-0.02 – -0.01

<0.001

-0.02

-0.02 – -0.01

<0.001

-0.02

-0.02 – -0.01

<0.001

Usagecont

-0.01

-0.01 – -0.00

<0.001

-0.01

-0.01 – -0.00

<0.001

-0.01

-0.01 – -0.01

<0.001

CarbFlow

0.00

-0.00 – 0.00

0.169

0.00

-0.00 – 0.00

0.105

0.00

-0.00 – 0.00

0.333

Density

-0.11

-0.17 – -0.05

<0.001

-0.08

-0.13 – -0.04

<0.001

-0.07

-0.10 – -0.04

<0.001

MFR

-0.00

-0.00 – -0.00

0.001

-0.00

-0.00 – -0.00

<0.001

Balling

-0.10

-0.15 – -0.04

<0.001

PressureVacuum

-0.02

-0.04 – -0.00

0.017

-0.00

-0.02 – 0.01

0.863

OxygenFiller

-0.33

-0.49 – -0.18

<0.001

-0.31

-0.47 – -0.15

<0.001

-0.33

-0.50 – -0.17

<0.001

BowlSetpoint

0.00

0.00 – 0.00

<0.001

0.00

0.00 – 0.00

<0.001

0.00

0.00 – 0.00

0.001

PressureSetpoint

-0.01

-0.01 – -0.00

<0.001

-0.01

-0.01 – -0.01

<0.001

-0.01

-0.02 – -0.01

<0.001

AirPressurer

-0.00

-0.01 – 0.00

0.349

-0.00

-0.01 – 0.00

0.413

-0.00

-0.01 – 0.00

0.406

AlchRel

0.12

0.06 – 0.17

<0.001

CarbRel

0.03

-0.08 – 0.13

0.611

0.14

0.04 – 0.24

0.005

0.22

0.12 – 0.32

<0.001

BallingLvl

0.11

0.06 – 0.16

<0.001

Observations

2055

2055

2055

R2 / R2 adjusted

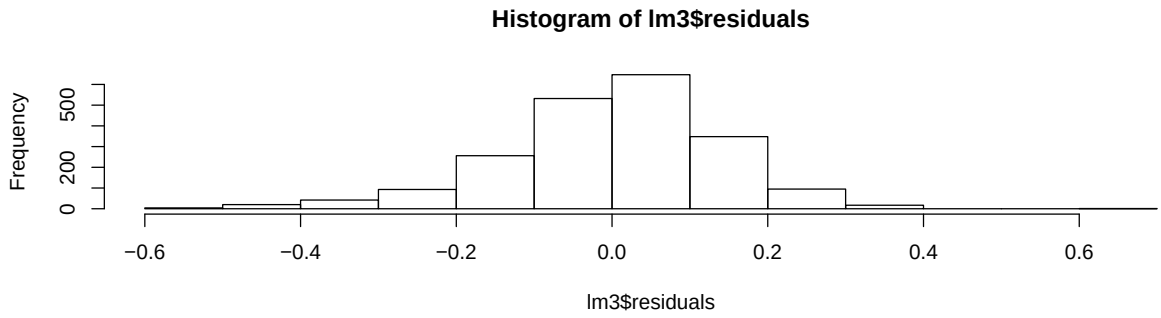
0.423 / 0.414

0.411 / 0.403

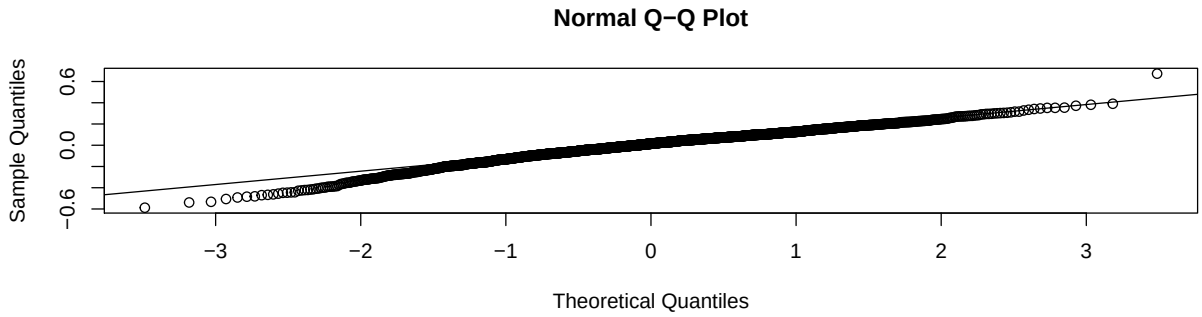
0.341 / 0.335

Across all three models, our adjusted R squared remains unchanged yet we were able to simplify the model. Using lm3, we will now check if the linear regression assumptions are satisfied.

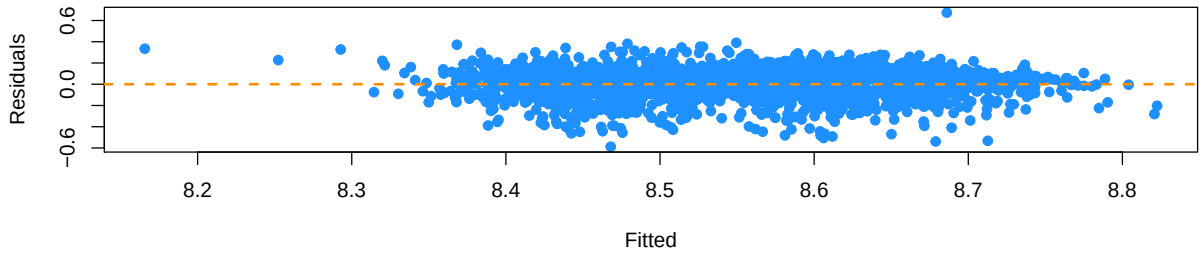
```
hist(lm3$residuals)
```




```
qqnorm(lm3$residuals)
qqline(lm3$residuals)
```




```
plot(fitted(lm3), resid(lm3), col = "dodgerblue", pch = 20, cex = 1.5,
    xlab = "Fitted", ylab = "Residuals")
abline(h = 0, lty = 2, col = "darkorange", lwd = 2)
```



Our final LM seems to have near normal residuals and constant variance as can be seen in the provided visualizations. We will stick to this model to compare performance vs our NNEt model when we move into the prediciton phase.

Model 2: NNET

```
set.seed(58677)

# nn1<-nnet(PH~., data=df_train,size=2,linout=T)
```

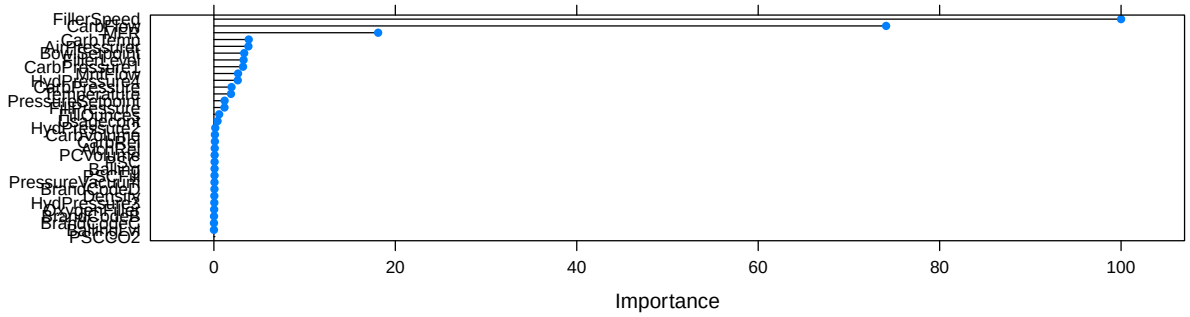
```r
# library(devtools)
# source_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1
nn_param <- expand.grid(.size = c(1:10), .decay = c(0, 0.01,
    0.1))

nn1 <- train(PH ~ ., data = df_train, method = "nnet", maxit = 1000,
    tuneGrid = nn_param, trace = F)

print(nn1)
```

```
Neural Network

2055 samples
  31 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 2055, 2055, 2055, 2055, 2055, 2055, ...
Resampling results across tuning parameters:

  size  decay  RMSE      Rsquared     MAE
   1    0.00   7.548907          NaN  7.546897
   1    0.01   7.548912  0.011144216  7.546902
   1    0.10   7.548941  0.003958879  7.546931
   2    0.00   7.548907          NaN  7.546897
   2    0.01   7.548910  0.001628368  7.546901
   2    0.10   7.548935  0.005394056  7.546926
   3    0.00   7.548907          NaN  7.546897
   3    0.01   7.548909  0.007667119  7.546900
   3    0.10   7.548925  0.008594149  7.546915
   4    0.00   7.548907          NaN  7.546897
   4    0.01   7.548909  0.004069893  7.546899
   4    0.10   7.548922  0.005327352  7.546912
   5    0.00   7.548907          NaN  7.546897
   5    0.01   7.548909  0.009582362  7.546899
   5    0.10   7.548919  0.006537257  7.546910
   6    0.00   7.548907          NaN  7.546897
   6    0.01   7.548908  0.006580751  7.546899
   6    0.10   7.548918  0.006370289  7.546908
   7    0.00   7.548907          NaN  7.546897
   7    0.01   7.548908  0.006409280  7.546899
   7    0.10   7.548916  0.006219754  7.546907
   8    0.00   7.548907          NaN  7.546897
   8    0.01   7.548908  0.003904651  7.546898
   8    0.10   7.548915  0.006191300  7.546906
   9    0.00   7.548907          NaN  7.546897
   9    0.01   7.548908  0.007235544  7.546898
   9    0.10   7.548915  0.005986394  7.546905
  10    0.00   7.548907          NaN  7.546897
  10    0.01   7.548908  0.006008687  7.546898
  10    0.10   7.548914  0.007989706  7.546905

RMSE was used to select the optimal model using the smallest value.
```

```
The final values used for the model were size = 1 and decay = 0.
```

What were the most imporant features in our nnet model?

```
nn1_imp <- varImp(nn1, scale = TRUE)

nn1_imp2 <- as.data.frame(as.matrix(nn1_imp$importance))

# nn1_imp2%>% kable(caption='Linear Model 2 Variable
# Importance') %>% kable_styling() %>% row_spec()

plot(nn1_imp)
```



MARS regression

By experience, MARS has been one of the better performing models. We should not spend too much time on trying to pick the best nnet model and pick MARS as our third. We anticipate MARS to outperform LM and NNET.

```
# hyperparameter tuning for MARS
mars1 <- earth(PH ~ ., data = df_train)

print(mars1)
```

```
Selected 18 of 22 terms, and 11 of 33 predictors
Termination condition: RSq changed by less than 0.001 at 22 terms
Importance: MnfFlow, BrandCodeC, AlchRel, PressureVacuum, Balling, ...
Number of terms at each degree of interaction: 1 17 (additive model)
GCV 0.01677669    RSS 33.31172    GRSq 0.4390952    RSq 0.4575109
```

Baseline MARS model performed better than any of our NNET models or best linear models.

```
mars_imp <- varImp(mars1, scale = TRUE)

# mars_imp2<-as.data.frame(as.matrix(mars_imp$importance))

mars_imp %>% kable(caption = "Linear Model 2 Variable Importance") %>%
    kable_styling()  # %>% row_spec()
```

Table 5: Linear Model 2 Variable Importance

	Overall
MnfFlow	100.00000
BrandCodeC	69.99661
AlchRel	56.69627
PressureVacuum	52.13163
Balling	47.96692
Temperature	41.15045
Usagecont	36.99494
CarbPressure1	31.91182
BowlSetpoint	28.07472
Density	22.42714
HydPressure3	18.32855
BrandCode	0.00000
CarbVolume	0.00000
FillOunces	0.00000
PCVolume	0.00000
CarbPressure	0.00000
CarbTemp	0.00000
PSC	0.00000
PSCFill	0.00000
PSCCO2	0.00000
FillPressure	0.00000
HydPressure2	0.00000
HydPressure4	0.00000
FillerLevel	0.00000
FillerSpeed	0.00000
CarbFlow	0.00000
MFR	0.00000
OxygenFiller	0.00000
PressureSetpoint	0.00000
AirPressurer	0.00000
CarbRel	0.00000
BallingLvl	0.00000

```
# plot(mars_imp2)
```

```
# hyperparameter tuning for MARS
mars2 <- earth(PH ~ BrandCode + PressureVacuum + AlchRel + Balling +
    Temperature + Usagecont + CarbPressure1 + BowlSetpoint +
    HydPressure3 + Density, data = df_train)

print(mars2)
```

```
Selected 17 of 21 terms, and 9 of 12 predictors
Termination condition: Reached nk 25
Importance: Usagecont, BrandCodeC, BowlSetpoint, AlchRel, Balling, ...
Number of terms at each degree of interaction: 1 16 (additive model)
GCV 0.01781484    RSS 35.44315    GRSq 0.4043861    RSq 0.4228001
```

Evaluation

```
# Make predictions
p1 <- lm3 %>% predict(df_test)
p2 <- nn1 %>% predict(df_test)
p3 <- mars2 %>% predict(df_test)

# Model performance metrics
sum_t <- data.frame(
  MODEL = c('LinearModel',
            'NNET',
            'Mars'),
  RMSE = c(caret::RMSE(p1, df_test$PH),
           caret::RMSE(p2, df_test$PH),
           caret::RMSE(p3, df_test$PH)),

  Rsquare = c(caret::R2(p1, df_test$PH),
              caret::R2(p2, df_test$PH),
              caret::R2(p3, df_test$PH)),

  MAPE = c(Metrics::mape(p1, df_test$PH),
             Metrics::mape(p2, df_test$PH),
             Metrics::mape(p3, df_test$PH)))

sum_t%>%kable(caption="Evaluation Summary on test set", booktabs=T)%>%kable_styling()%>%row_spec()
```

Table 6: Evaluation Summary on test set

MODEL	RMSE	Rsquare	MAPE
LinearModel	0.1395271	0.3352301	0.0126972
NNET	7.5463897	NA	7.5444531
Mars	0.1332974	0.3923408	0.0121377

As predicted MARS is the best of our three models.

Prediction

```
# remove space in-between variable names
colnames(df_eval) <- gsub(" ", "", colnames(df_eval))
# remove column with zero-variance
set.seed(58677)
# use mice w/ default settings to impute missing data
miceImput2 <- mice(df_eval, printFlag = FALSE)
# add imputed data to original data set
df_mice2 <- mice::complete(miceImput2)
# table(df_eval$BrandCode, useNA = 'ifany')
df_mice2$BrandCode[is.na(df_mice2$BrandCode)] <- "B"
# table(df_mice$BrandCode, useNA = 'ifany') Look for any
# features with no variance: zero_cols <- nearZeroVar(
# df_mice2 )
df_final22 <- df_mice2[, -zero_cols]  # drop these zero variance columns
df_final22$BrandCode <- as.factor(df_final22$BrandCode)
```

```r
df_eval2 <- subset(df_eval, select = -PH)
pred_eval <- predict(mars2, subset(df_final22))
write.csv(pred_eval, "prediction.csv")
```