PROJECT 2: PH PREDICTION

GROUP 5:

MIA SIRACUSA

JOHN SUH

LIDIIA TRONINA

HENRY VASQUEZ

AARON ZALKI

CONTENTS

PROJECT SUMMARY

For the project we were given a dataset from our beverage manufacturing company containing 2,571 records. The records contained 33 columns/variables including pH that were recorded during the beverage batch process. Our goal was to create a model to best predict pH in our beverages. The project is important because pH is a measure of the acidity/alkalinity which effects the overall taste of our drinks. The pH in our drinks must conform in a critical range in order to brew quality drinks. In our project, we found:

- We tested 6 different models including Multiple Linear Regression, Support Vector Machine, k-Nearest Neighbors, Random Forest, Ranger and Extreme Gradient Boosting
- Random forest models (Random Forest, Ranger and Extreme Gradient Boosting) performed the best in predicting pH based on RMSE, R-squared and MAE metrics
- The time to train each model varied with the k-Nearest Neighbor model having the fasted time and the Random Forest model having the slowest time
- The variable importance measure showed that Mnf Flow was consistently the most used variable in all models
    - Other consistent variables include Usage Count, Bowl Setpoint and Pressure Vacuum
- The model we recommend using in predicting pH is the Ranger model
    - The model represents 70% of the variance in the data while having minimal error

After building our final model we loaded the evaluation dataset from our company GitHub site and ran our model on the new data. The predictions were then saved in an excel sheet and sent off to our Product department to score the results.

INTRODUCTION

The beverage manufacturing company uses a special recipe to create its cola flavored soda. In order to keep up with the high demand for the delicious drink we sell, we automate this process through machinery that allows us to create large quantities of our delicious soda to meet the high demand. In doing so, the machines automatically record several data measurements taken through the entire process and store the data for each drink created. By doing so, we are able to understand the exact conditions of each drink that is being made.

The machinery records 33 different measurements during the process, the most important being the potential for hydrogen (pH). The pH plays a role in the acidity/alkalinity of our cola, and therefore affects the way our drinks will taste. If our drinks are too low or too high in pH then our taste will be off, leading to dissatisfied customers.

Quality control is key, therefore our objective is to understand how the variables affect the pH of our cola in order to consistently brew great tasting beverages.

DATA OVERVIEW

The data set being used to perform this analysis includes a model set of 2,561 records and a test set of 267 records. The data consists of 33 different variables including pH level, the key performance indicator (KPI). See the list of variables below:

Brand Code	Filler Level
Carb Volume	Filler Speed
Fill Ounces	Temperature
PC Volume	Usage cont
Carb Pressure	Carb Flow
Carb Temp	Density
PSC	MFR
PSC Fill	Balling
PSC CO2	Pressure Vacuum
Mnf Flow	Oxygen Filler
Carb Pressure1	Bowl Setpoint
Fill Pressure	Pressure Setpoint
Hyd Pressure1	Air Pressurer
Hyd Pressure2	Alch Rel
Hyd Pressure3	Carb Rel
Hyd Pressure4	Balling Lvl

These variables were first analyzed to handle missing valued and excessive 0's as well as create new variables where seen necessary. The only categorical variable included is Brand Code, while the rest are numeric type. For more information regarding the meaning of the variables, please refer to product for actual definitions.

DATA PREPROCESSING

In order to perform the analysis effectively the correct tools need to be used. These tools include the programming language as well as the libraries that have the proper functions for the job. Furthermore, the training and evaluation data were uploaded as csv files (easier to work with) to the company GitHub site for easier reproducibility. For this project, the R programming language was utilized alongside with several packages listed below with their purpose:

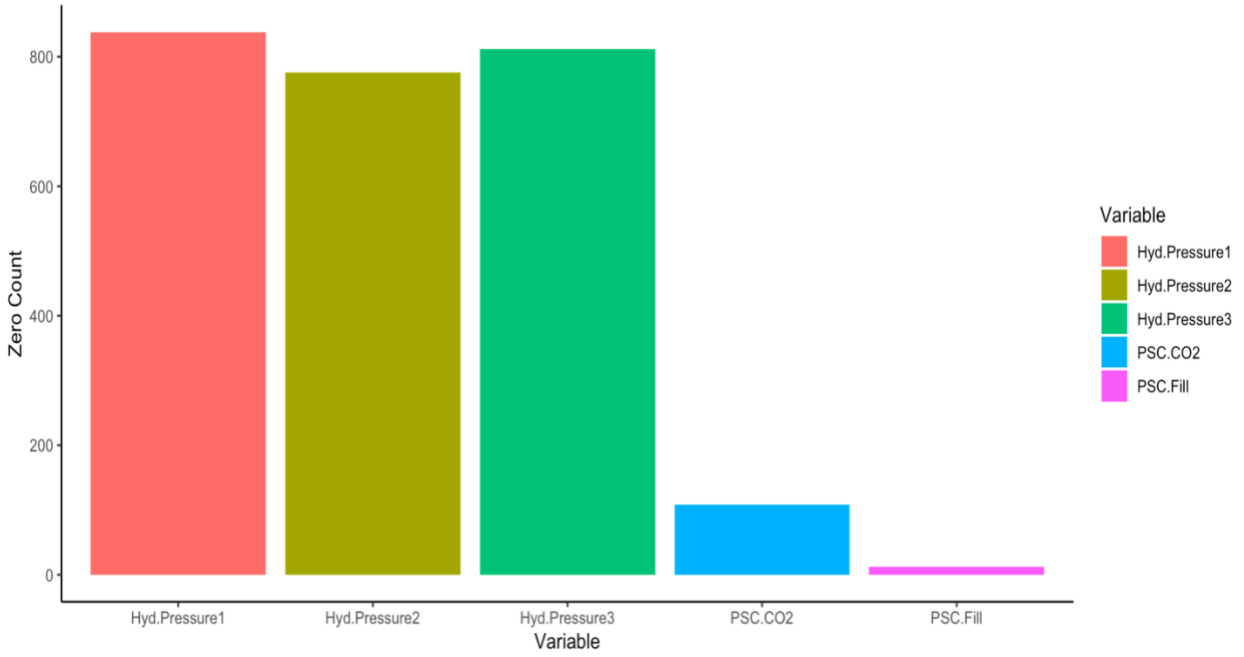
MISSING VALUES - NAS

In order to build a quality predictive model, missing values need to be handled appropriately. A brief look at the data showed that 4 records had missing values for pH levels. Because pH is the variable of interest those records were removed from the data since they will not improve the model's predictions. Furthermore, all but one of the variables are integers or numbers, that variable being brand code which is split between 4 different codes (A, B, C and D) and 1 blank code. Luckily, most of the variables have less than 1% missing values, with the exception of the 10 listed below. Most variables have a relatively low percentage of missing values. Variables that follow a normal distribution can be simply replaced with average value of that variable, whereas variables that have a skewed distribution or multimodal distribution may need a different approach while testing the model on test data. For the purpose of the model building, rows with missing values will be excluded.

Variable	Percent of NAs
MFR	8.10
Filler.Speed	2.10
PC.Volume	1.52
PSC.CO2	1.52
Fill.Ounces	1.48
PSC	1.29
Carb.Pressure1	1.25
Hyd.Pressure4	1.09
Carb.Pressure	1.05
Carb.Temp	1.01

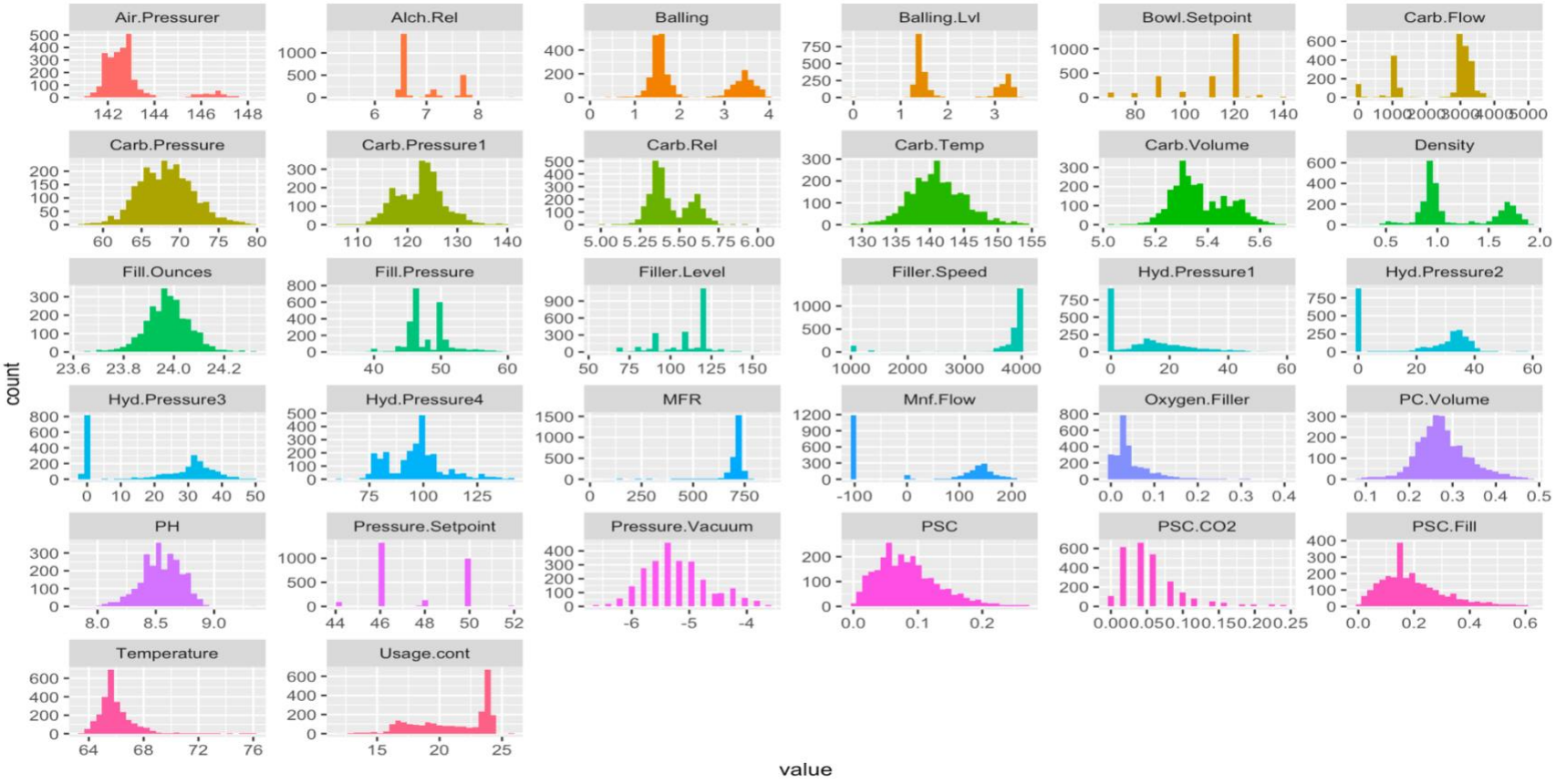
Missing Values – o's

Zeros can also represent missing values. In this case we'll explore the variables to see the frequency of zeros. Below, the chart shows the top 5 variables by frequency of zeros. Three of them have roughly 800 zeros, where as PSC CO2 has about 100 and PSC fill has about 10. It is interesting to note that there are 4 different Hyd Pressure variables and only number 4 has no zeros present. Rather than exclude the 3 Hyd variables below, the creation of a new dummy variable representing the presence of any of the 3 other Hyd pressure variables will be created to see how it affects pH levels. The variable HydPressureRecorded123 will equal 1 if Hyd Pressure 1,2 or 3 are not equal to 0 or NA. The variables Hyd Pressure 1, Hyd Pressure 2 and Hyd Pressure 3 will be dropped. The remaining variables below don't appear to have a significant number of zeros and can be considered normal.

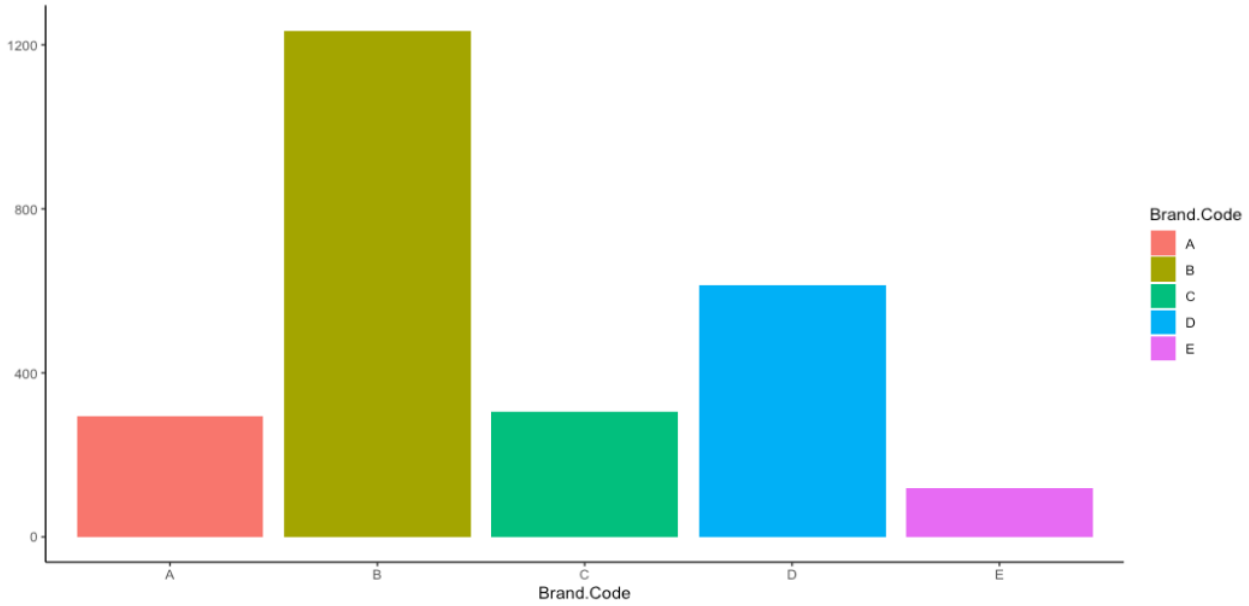


VARIABLE DISTRIBUTION AND FREQUENCY

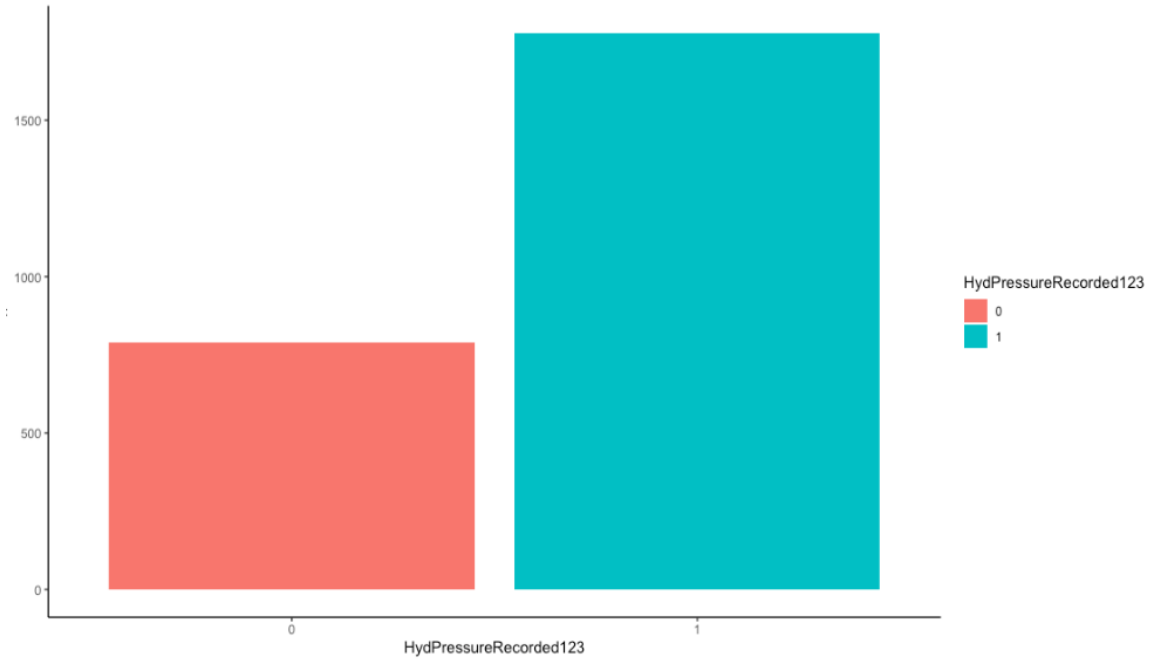
Looking at the chart on the next page, variables that have the normal distribution are those with a bell-shaped curve such as carb pressure, carb temp, fill ounces, PC volume and pH. It's also noticeable the variables of Hyd Pressure previously looked at and their large count of zeros in comparison to other values.


Air.Pressurer
Alch.Rel
Balling
Balling.Lvl
Bowl.Setpoint
Carb.Flow
Carb.Pressure
Carb.Pressure1
Carb.Rel
Carb.Temp
Carb.Volume
Density
Fill.Ounces
Fill.Pressure
Filler.Level
Filler.Speed
Hyd.Pressure1
Hyd.Pressure2
Hyd.Pressure3
Hyd.Pressure4
MFR
Mnf.Flow
Oxygen.Filler
PC.Volume
PH
Pressure.Setpoint
Pressure.Vacuum
PSC
PSC.CO2
PSC.Fill
Temperature
Usage.cont
count
value

There is only one variable that is a categorical type and that is the brand code and the other is a created variable HydPressureRecorded123. Brand code is broken out by 5 different categories, one of which is “E” that replaced blank or missing values. The most frequent brand code is “B.” The chart below shows the frequency of each Brand Code.

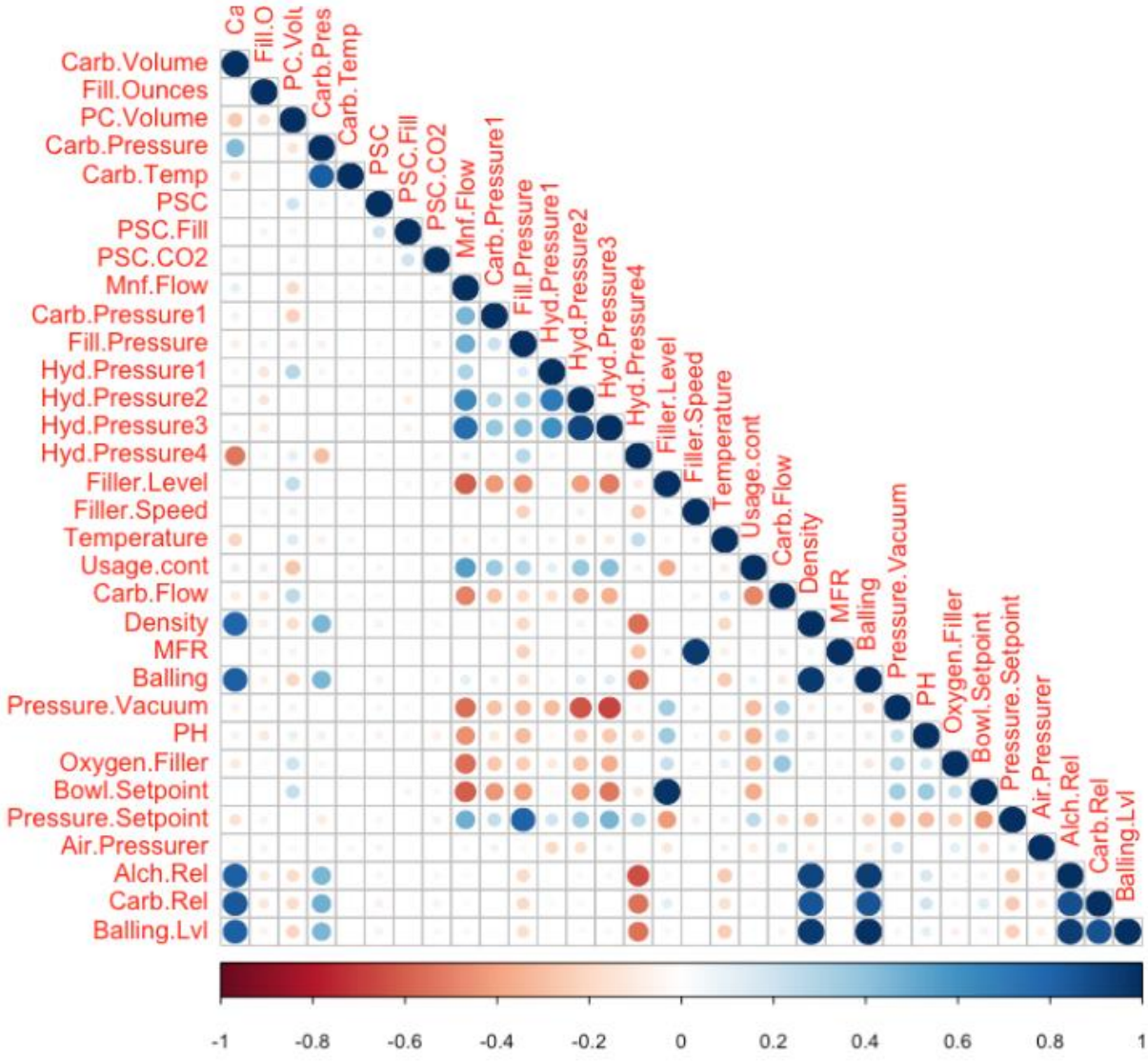


Also included is the frequency for the newly calculated field, HydPressureRecorded123. Majority records have at least a value for Hyd Pressure 1, 2 or 3.



VARIABLE CORRELATION

When creating models, it is important to know the relationship between the independent variables and the dependent variable (pH). Variables that have a strong correlation to the target variable (pH), whether it be positive or negative help predict its value. Variables that have a weak correlation or no correlation (correlation coefficients between -0.3 and 0.3) with the target variable will not be helpful in predicting pH. Looking at the correlation matrix below, the correlation between all variables are shown where blue means positively correlated and red means negatively correlated. Variables that have a positive correlation with pH are filler level and bowl set point. Variables with a negative correlation to pH are Mnf Flow, fill pressure, usage count and pressure setpoint.



BUILDING THE MODELS

PREPARATION

Before beginning the modeling, NAs were replaced with mean values for each numeric variable. Since it was determined that zeros were only a major issue in the Hyd Pressure variables 1, 2 and 3, these values were not replaced with the variable mean and instead the new variable HydPressure123 was created. The process to do these steps was saved as a function to handle the introduction of new data for future predictions.

The data was then split 80:20 for the training set and test set. The training set has a size of 2,053 and the test set has a size of 514. Several models were chosen to fit the data and the complexities of the models varied from multiple linear regression to random forest models. Overall, 6 different modeling methods were used to estimate pH in our beverages. The models include 1 linear model (Multiple Linear Regression), 2 non-linear model (Support Vector Machine and k-Nearest Neighbors) and 3 tree models (Random Forest, Ranger and Extreme Gradient Boosting).

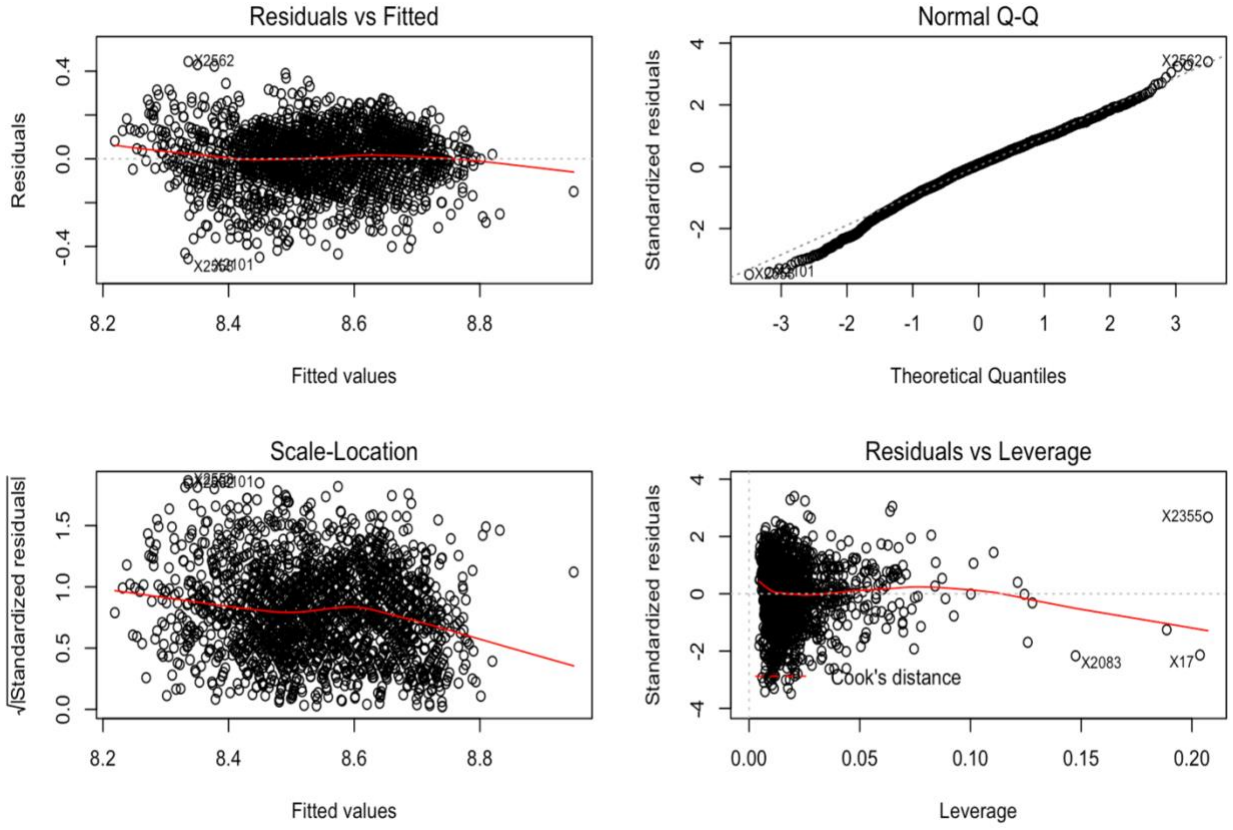
MULTIPLE LINEAR REGRESSION

The first model we used for predicting pH levels was a simple Multiple Linear Regression model. Since NA values for the variables were replaced by means, the only other transformations left were handling skewed data. To handle the skewed data, we used a preprocess Box Cox transformation to handle skewed variables. We then built the model using all of the variables, including the created variable HydPressure123. See the final Multiple Linear Regression model below:

```
Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-0.45553 -0.08153  0.00953  0.08715  0.44411

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              4.952e+03  3.923e+03   1.262 0.207073
Brand.CodeB              1.123e-01  2.445e-02   4.592 4.67e-06 ***
Brand.CodeC             -2.926e-02  2.436e-02  -1.201 0.229861
Brand.CodeD              4.953e-02  1.640e-02   3.020 0.002563 **
Brand.CodeE              5.281e-02  2.712e-02   1.947 0.051618 .
Carb.Volume             -6.829e+00  1.020e+01  -0.670 0.503148
Fill.Ounces             -3.290e-03  1.510e-03  -2.179 0.029481 *
PC.Volume               -7.728e-02  4.053e-02  -1.907 0.056664 .
Carb.Pressure           -2.398e-01  6.102e-01  -0.393 0.694389
Carb.Temp                3.438e+02  4.829e+02   0.712 0.476658
PSC                     -2.641e-02  1.827e-02  -1.445 0.148515
PSC.Fill                -4.315e-02  2.586e-02  -1.668 0.095377 .
PSC.CO2                 -1.128e-01  6.909e-02  -1.633 0.102728
Mnf.Flow                -7.275e-04  5.416e-05 -13.431  < 2e-16 ***
Carb.Pressure1           2.798e-02  3.293e-03   8.497  < 2e-16 ***
Fill.Pressure            1.207e+01  4.559e+00   2.647 0.008175 **
Hyd.Pressure1           -3.132e-05  4.213e-04  -0.074 0.940739
Hyd.Pressure2           -1.421e-03  6.217e-04  -2.285 0.022411 *
Hyd.Pressure3            2.981e-03  6.492e-04   4.592 4.66e-06 ***
Hyd.Pressure4           -1.248e-01  2.108e-01  -0.592 0.553957
Filler.Level            -7.110e-06  5.591e-06  -1.272 0.203660
Filler.Speed            -4.228e-11  2.488e-09  -0.017 0.986446
Temperature             -5.448e+03  8.122e+02  -6.708 2.56e-11 ***
Usage.cont              -3.712e-04  6.303e-05  -5.890 4.52e-09 ***
Carb.Flow                1.414e-06  4.277e-07   3.306 0.000963 ***
Density                 -2.990e-02  2.903e-02  -1.030 0.303118
MFR                      1.011e-07  1.048e-07   0.965 0.334735
Balling                 -2.175e-01  3.937e-02  -5.524 3.75e-08 ***
Pressure.Vacuum         -2.082e-02  7.860e-03  -2.649 0.008132 **
Oxygen.Filler           -5.268e-02  1.156e-02  -4.558 5.48e-06 ***
Bowl.Setpoint            2.841e-05  6.033e-06   4.708 2.67e-06 ***
Pressure.Setpoint       -1.065e+03  2.350e+02  -4.533 6.17e-06 ***
Air.Pressurer           -3.868e+03  7.857e+03  -0.492 0.622586
Alch.Rel                 1.562e+01  8.582e+00   1.820 0.068892 .
Carb.Rel                 4.868e+00  8.180e+00   0.595 0.551880
Balling.Lvl              9.020e-02  1.895e-02   4.759 2.08e-06 ***
HydPressureRecorded1231  2.519e-02  1.718e-02   1.466 0.142778
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.132 on 2016 degrees of freedom
Multiple R-squared:  0.4248,    Adjusted R-squared:  0.4145
F-statistic: 41.35 on 36 and 2016 DF,  p-value: < 2.2e-16
```

Overall, the model recorded an adjusted R-squared value of 0.4145 meaning that the model explains roughly 41% of the variance in the data. The adjusted R-squared takes into consideration the impact of adding new variables whereas the regular R-squared does not. The estimate

column shows the coefficients fitted for each variable of the model. Coefficients estimate how the prediction of pH will increase or decrease for every increase in unit of the corresponding variable. We also tested the assumptions for linear regression in this model, see below:



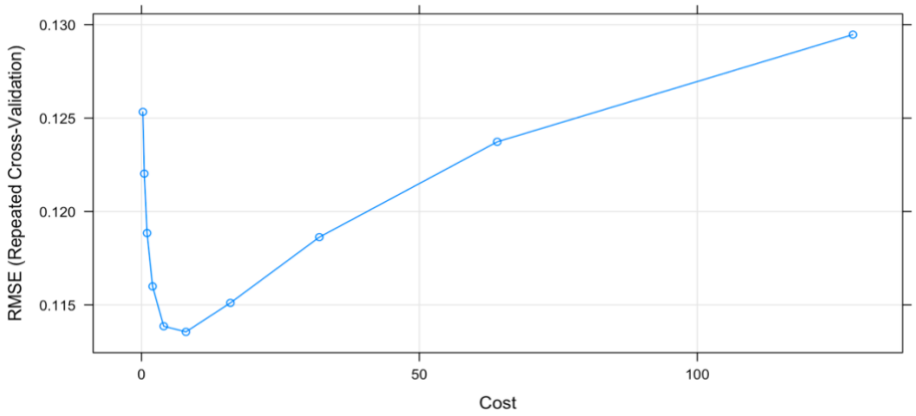
In the Residuals vs Fitted plot, the residuals are evenly spread across the horizontal line at zero and there is no indication of a non-linear pattern present. In the Q-Q plot, the residuals follow a straight line showing a normal distribution. The Scale-Location plot shows equal variance. In the leverage plot there does not appear to be any extreme cases that we need to worry about. Overall, we concluded that the model is a good fit

SUPPORT VECTOR MACHINE-RADIAL

The second model method we used is a non-linear Support Vector Machine (SVM) classifier with a radial kernel function. For this project we used the caret package on all of the models, which automatically chooses the best parameters that result in the highest R-squared value. We also added some preprocessing parameters which normalizes the variables making their scale comparable See below for the final SVM model:

```
Support Vector Machines with Radial Basis Function Kernel

2053 samples
  33 predictor

Pre-processing: centered (36), scaled (36)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 1847, 1849, 1848, 1849, 1847, 1846, ...
Resampling results across tuning parameters:

  C       RMSE       Rsquared   MAE
    0.25  0.1253314  0.4821863  0.09425464
    0.50  0.1220268  0.5061482  0.09136571
    1.00  0.1188438  0.5298340  0.08883888
    2.00  0.1159916  0.5509624  0.08668065
    4.00  0.1138589  0.5672383  0.08510688
    8.00  0.1135566  0.5716828  0.08480987
   16.00  0.1151081  0.5653459  0.08573549
   32.00  0.1186258  0.5493984  0.08823466
   64.00  0.1237304  0.5256535  0.09198362
  128.00  0.1294660  0.5009720  0.09621228

Tuning parameter 'sigma' was held constant at a value of 0.01971668
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were sigma = 0.01971668 and C = 8.
```

The best SVM model that was tuned had parameters of C (cost) equal to 8 and sigma equal to 0.01971668 and an R-squared value of 0.5716828. The SVM model explain roughly 57% of the variance of the data, which is over 33% higher than the Multiple Linear Regression model. Also reported in the SVM model summary above is Mean Absolute Error (MAE). MAE is similar to RMSE in that it tells us how accurate the model is and how big of an error we can expect on average. See below for a RMSE vs Cost plot of the SVM model:



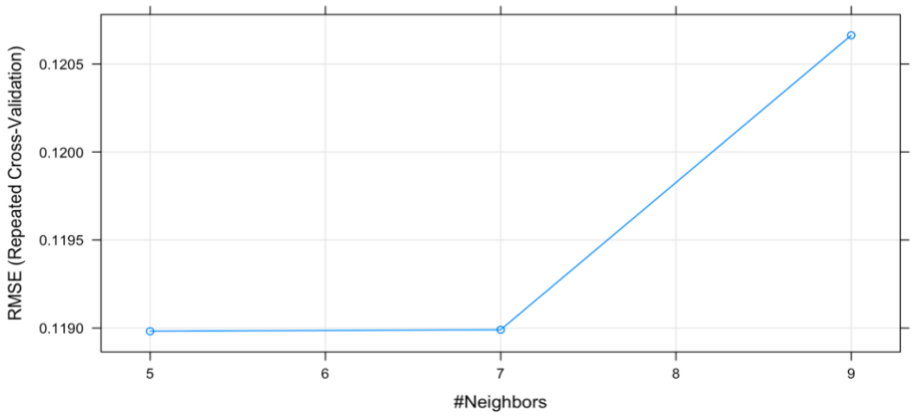
The plot shows the change of RMSE during the tuning of the model parameter cost (C), which determines misclassifications. A higher C value will classify more points correctly but will result in a more complex model. Increasing C may result in less error for an SVM model however the issue of overfitting becomes a problem. RMSE means root mean squared error and is a way of measuring the error of the predicted values against the actual values. A lower RMSE is better for the model and according to the tuned model the parameter C equal to 8 is optimal. Looking at the chart above, at cost equal to 8 the RMSE value is the lowest.

K-NEAREST NEIGHBORS

The k-Nearest Neighbors (kNN) model predicts the values of new data by using feature similarity. It is important to note that kNN models do not take categorical variables, therefore they will be excluded. We use the caret package to train a model using the knn method and the automatic parameter tuning results in the model below:

```
k-Nearest Neighbors

2053 samples
  33 predictor

Pre-processing: centered (36), scaled (36)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 1847, 1848, 1847, 1847, 1848, 1848, ...
Resampling results across tuning parameters:

  k  RMSE       Rsquared   MAE
  5  0.1198263  0.5230672  0.09012394
  7  0.1190908  0.5272403  0.09022070
  9  0.1204967  0.5160022  0.09168791

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was k = 7.
```

The final kNN model has k = 7 and an R-squared value of 0.5272403. Overall, the kNN model represents about 53% of the variance in the data. The R-squared value is less than the SVM model but is still significantly higher than the Multiple Linear Regression model. Also, the MAE reported for the best model is higher than our SVM model meaning that estimates from the kNN model will have a larger error on average. It is important to note that the MAE for the optimal model is actually higher than it is for the model with k equal to 5. Below we have the chart showing RMSE vs # of neighbors:



The RMSE for seems about the same for 5 and 7 neighbors where the difference is minute (about than 0.0007). It appears either 5 or 7 neighbors could be used for the kNN model based on the RMSE, the R-squared and the MAE.

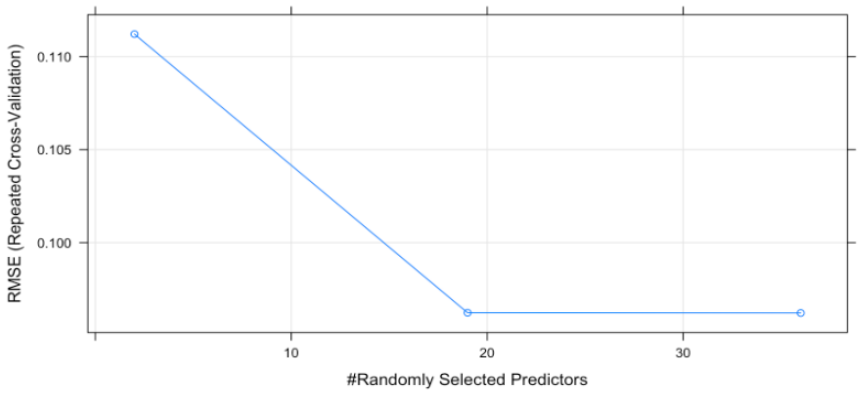
RANDOM FOREST

The last 3 models we used to estimate pH were tree-based models, the first being a Random Forest model. A Radom Forest model works by creating multiple decision trees during the training process and outputs the mean prediction of the individual trees. It is important to note that these models can be a bit more time consuming than the previous models we used and may require more computing power.

The Random Forest model is tuned through the mtry variable, which represents the number of randomly selected predictors at each cut in the tree. Overall, there was a very small difference in RMSE, R-squared and MAE between a mtry value of 19 and 36. See the Random Forest model summary below:

```
Random Forest

2053 samples
  33 predictor

Pre-processing: centered (36), scaled (36)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 1847, 1847, 1849, 1848, 1847, 1848, ...
Resampling results across tuning parameters:

  mtry  RMSE        Rsquared   MAE
   2    0.11120562  0.6359926  0.08547354
  19    0.09623161  0.7033716  0.07164128
  36    0.09621852  0.6953974  0.07077459

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 36.
```

The optimal model chosen for the Random Forest method has a RMSE value of 0.09621852 an R-squared of 0.6953974 and MAE of 0.07077459. This is the best model we have created based on RMSE, R-squared and MAE. In the chart below we can verify that the number of randomly selected predictors equal to 36 results in the lowest RMSE.

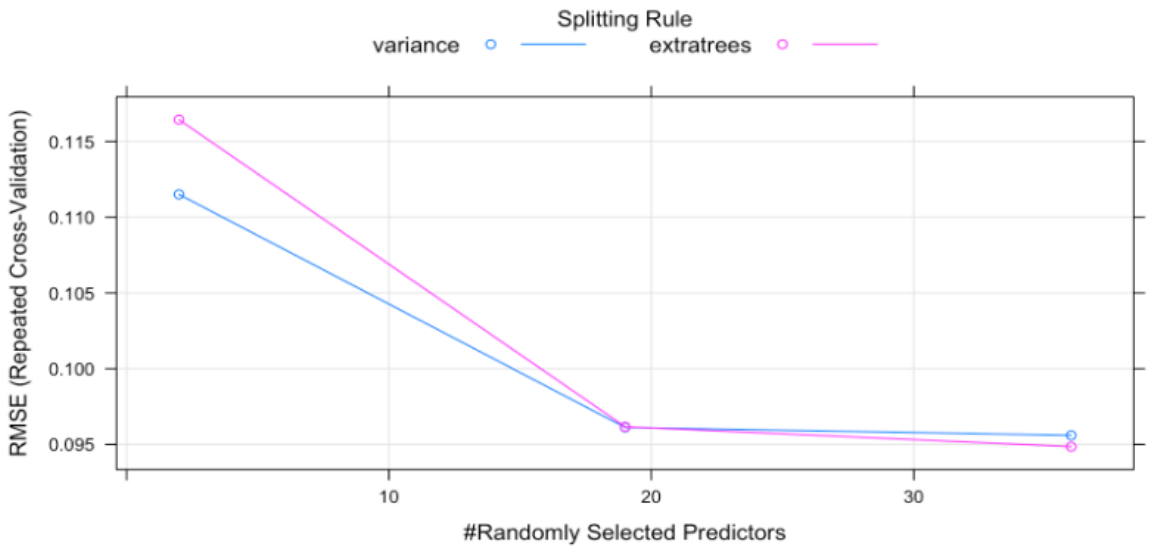


RANGER

The ranger method is a Random Forest model that is suited for high dimensional data. Using this method while modeling is supposed to speed up the modeling process. Using the ranger method, we reproduced similar and slightly better accuracy from a Random Forest model. See below:

```
Random Forest

2053 samples
  33 predictor

Pre-processing: centered (36), scaled (36)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 1847, 1848, 1848, 1848, 1847, 1849, ...
Resampling results across tuning parameters:

  mtry  splitrule   RMSE        Rsquared   MAE
   2    variance    0.11150644  0.6349789  0.08564695
   2    extratrees  0.11643670  0.5964600  0.09058291
  19    variance    0.09612357  0.7052532  0.07145675
  19    extratrees  0.09616677  0.7022624  0.07141264
  36    variance    0.09560067  0.7008276  0.07027878
  36    extratrees  0.09485469  0.7070962  0.07018635

Tuning parameter 'min.node.size' was held constant at a value of 5
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were mtry = 36, splitrule = extratrees and min.node.size = 5.
```

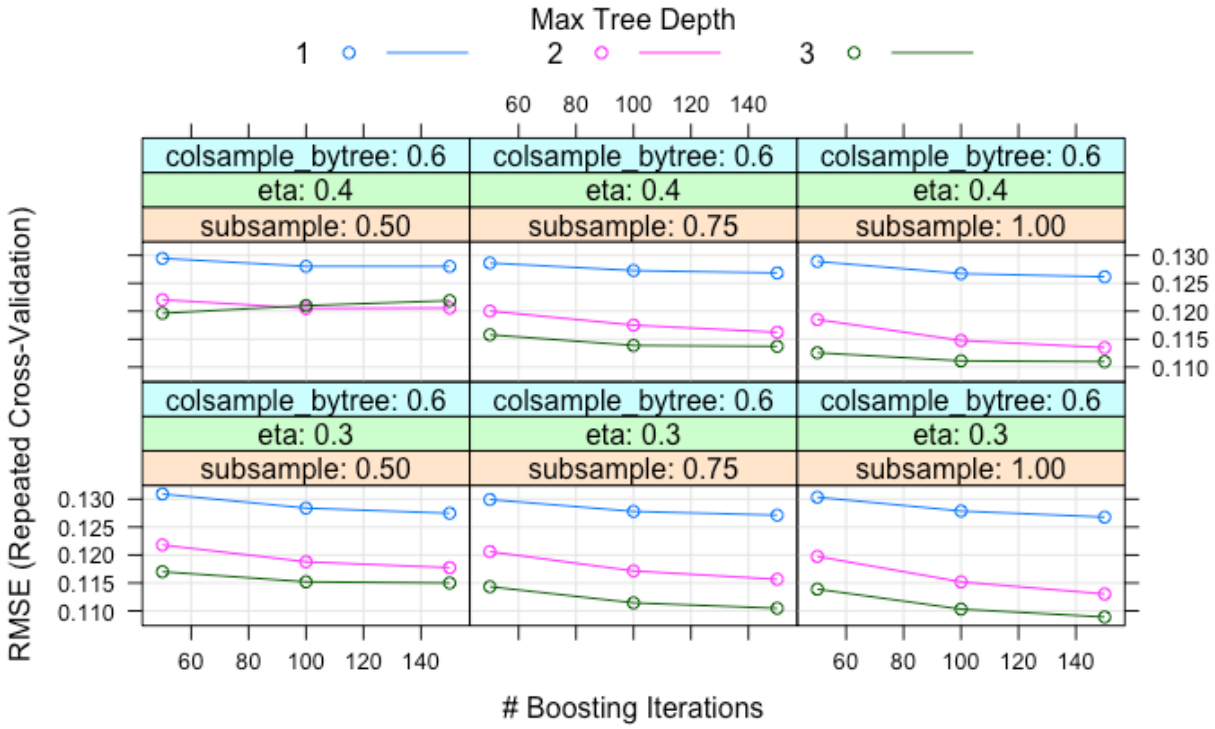
The tune parameters are mtry (same as the Random Forest model) and split rule, which is tuned between variance and extratrees. The final model resulted in a higher R-squared than our original Random Forest model and the highest R-squared among all of our models. The RMSE and MAE were also the lowest among all models reported at 0.09485469 and 0.07018635 respectively. In the chart below we see how the Ranger model's RMSE is affected by the increase of mtry between the splitting rules of variance and extratrees. It shows that mtry equal to 36 using the splitting rule of extratrees results in the lowest RMSE.



EXTREME GRADIENT BOOSTING-TREES

Extreme Gradient Boosting (XGB) is an efficient and scalable implementation of gradient boosting, which creates a prediction model from the collection of weak prediction models. While fitting the model, we noticed that it does process quite a bit of tuning for the parameters it takes. However, we concluded that the model did not perform better than our previous 2 tree models. Overall, the XGB model reported an R-squared of 0.6027581, the third highest. The model also had a MAE of 0.08290533, which is closed to our SVM model than our Random Forest models. The RMSE recorded was 0.1089021, which translates to third best.

The chart below shows the effects of tuning the parameters between different values of colsample_bytree, eta and subsample while increasing the number of iterations. The final model is in the bottom right chart with colsample_bytree equal to 0.6, eta equal to 0.3 and subsample equal to 1.0 and having a number of boosting iterations equal to 150.



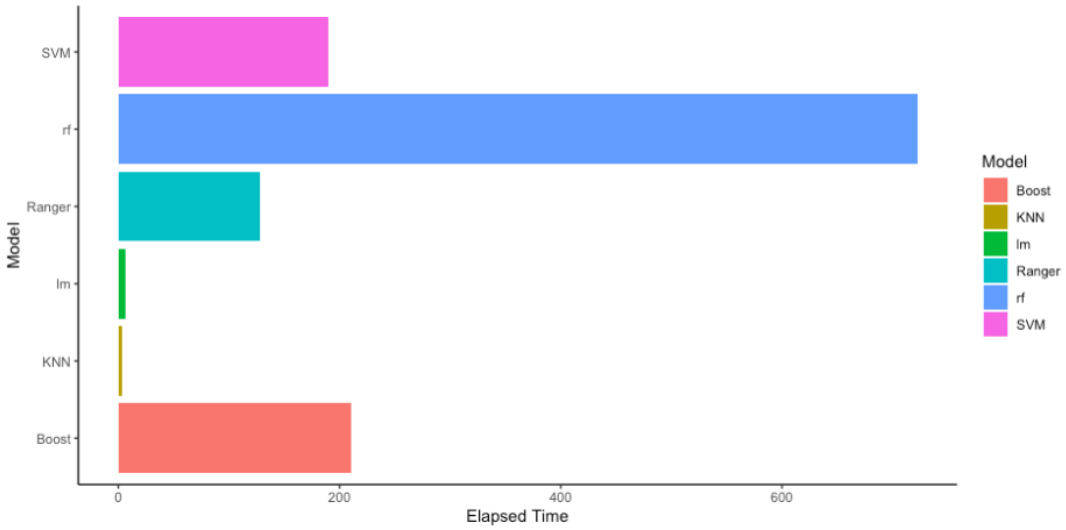
Model Evaluation

We ran 6 different models starting from a simple Multiple Linear Regression to a more complicated version of a Gradient Boost Algorithm. Not all models are created equal, and each model will have their strengths and weaknesses compared to others. To evaluate the models, we chose 3 measures to present. These measures include the time it took to train each model, the resulting RMSE, R-Squared and MAE the models recorded on the test data and the variable importance scoring.

Time to Train

Of the six models we ran, we expected the linear model to have the fastest time to train. We found that not to be the case and our kNN model was about twice as fast. We believe this was due to the Box Cox transformation we specified in the model to handle the skewness in some of the variables. It was also expected that the Random Forest models would have some of the longer time to train because they are more complex. Specifically, we expected the XGB model to have the longest run time, however the regular Random Forest model ended up being the longest to train.

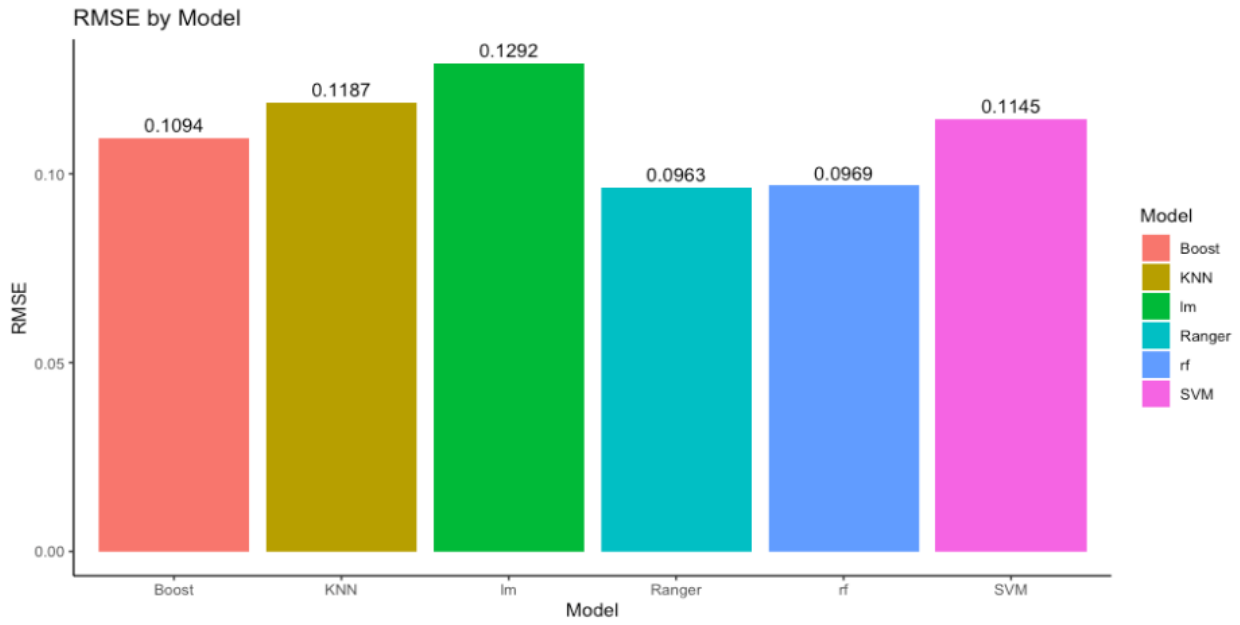
Overall, the random forest model took 3.5 times longer than the XGB model. The ranger model, which we used because of its efficiency proved to be surprisingly fast as it reduced the time to train a random forest model by over 80%. In the chart below we can see how the time to train differs among each model. Elapsed time is measured in seconds.



RMSE, R-SQUARED AND MAE ON TEST DATA

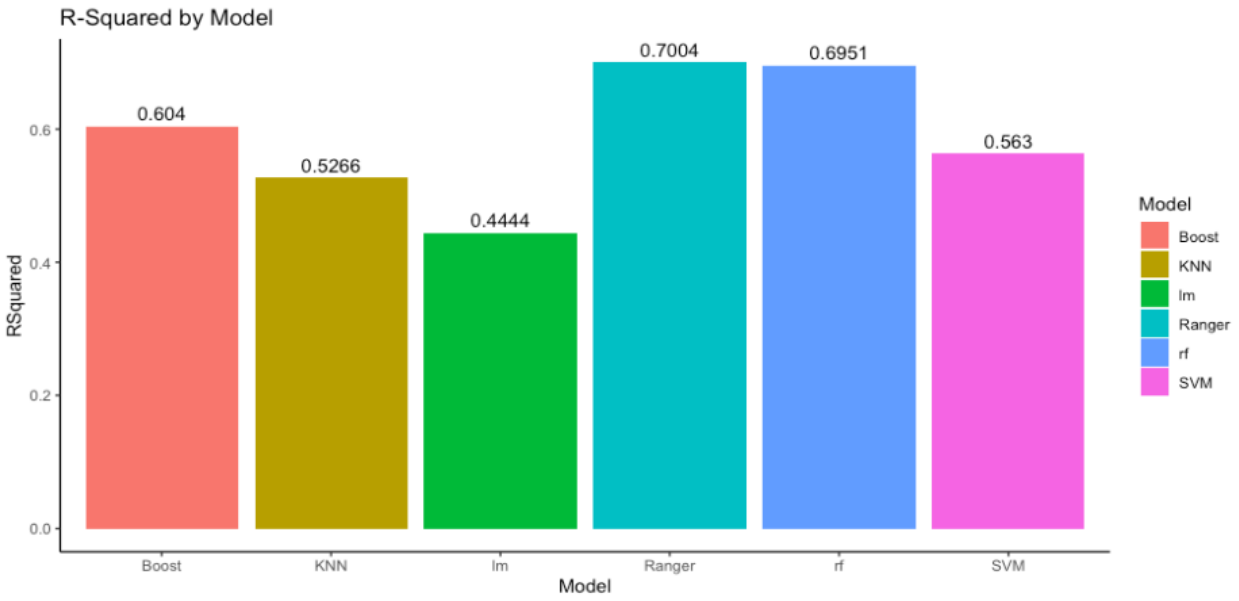
As we mentioned at the beginning of the Modeling section, the data was split 80:20 with 80% of the data being used to train the model and 20% of the data being left to test on. This left us with about 514 records to test the models on and the results below represent how each model did with new unseen data.

RMSE



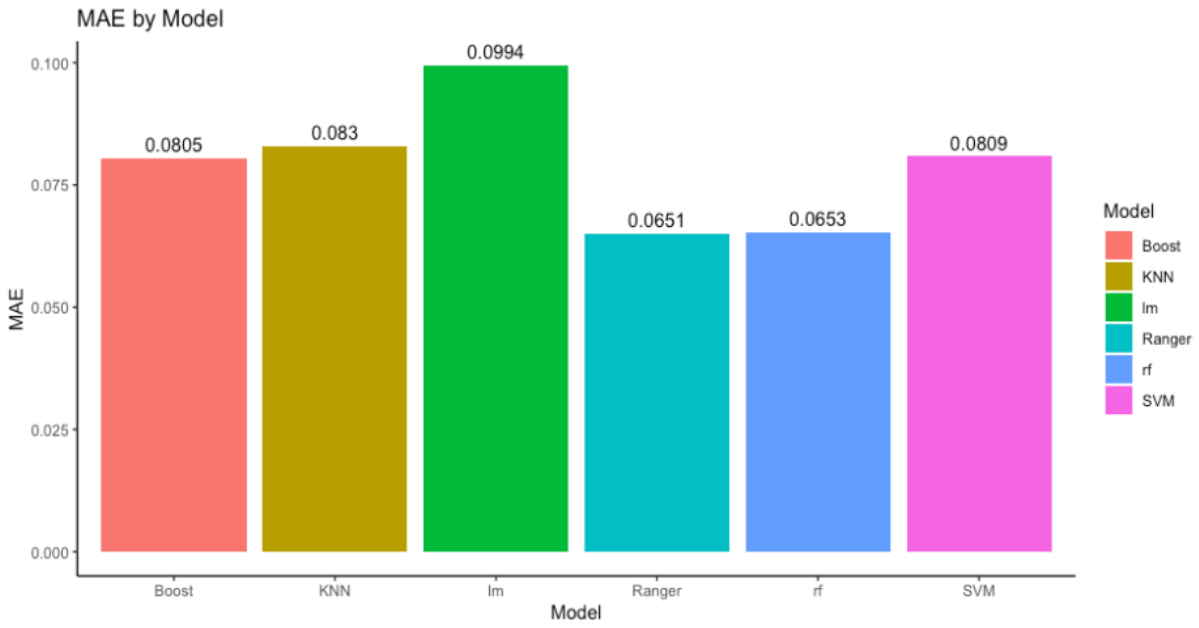
The Root Mean Squared Error (RMSE) measures error in the model predictions by taking the square root of the average squared error. The lower the RMSE the better the fit of the model. Displayed above is the RMSE calculated on the test data by each model. Our 2 random forest models recorded the lowest RMSE, followed by the Extreme Gradient Boost model and then the Support Vector Machine model.

R-Squared



R-squared represented how good the model fits the data overall. A higher R-squared value means a better model because it translates to how well the model represents the variance in our data. Our ranger model had the highest R-squared model, followed by the random forest model and then the XGB model. In conclusion, 70% of the variance in our data can be explained by the ranger model.

MAE

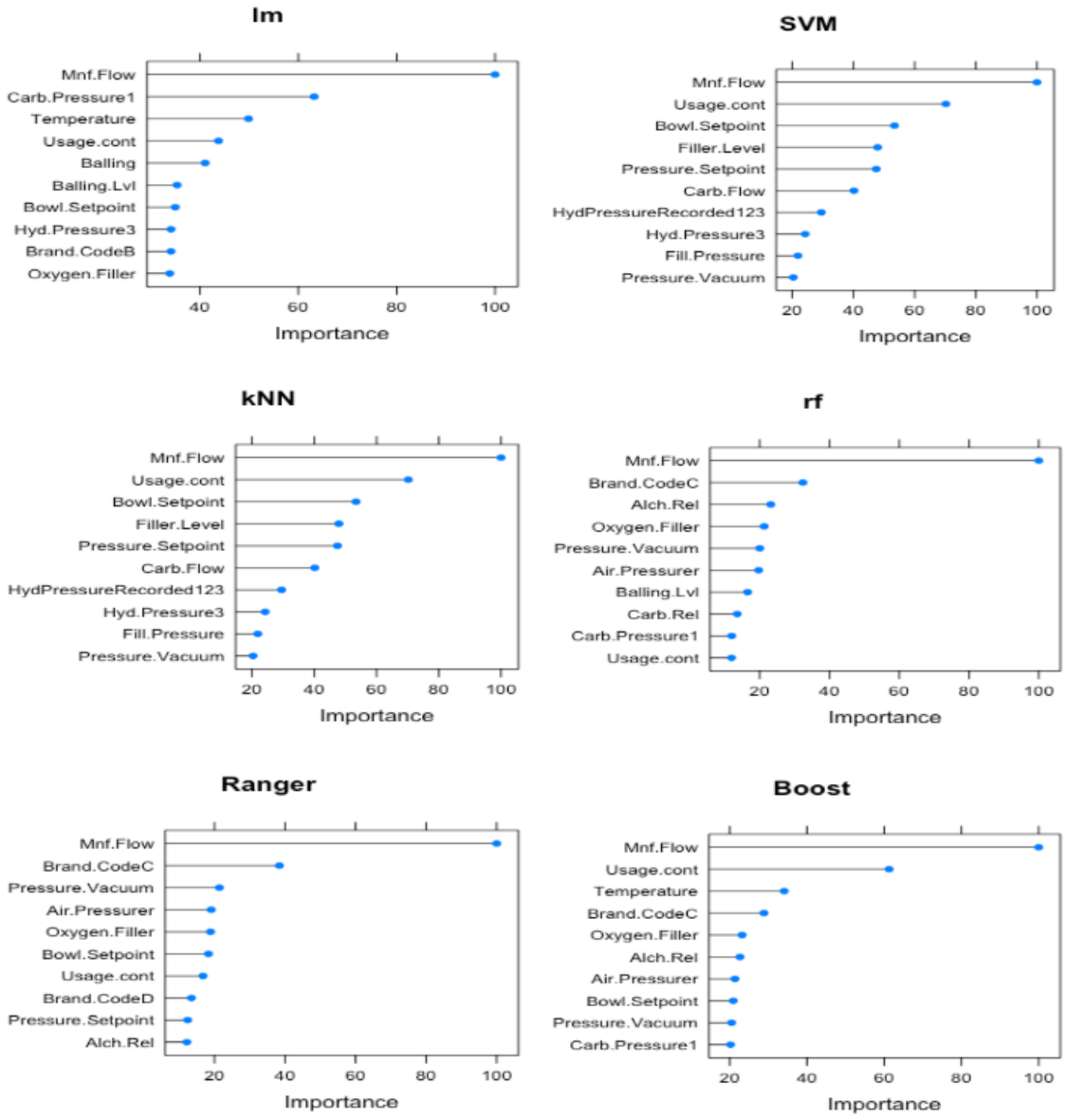


The final metric we recorded was the mean absolute error. This is calculated by average absolute error from all predictions for the model. The calculation is different from RMSE in which the errors are converted to absolute rather than squared. A lower MAE value translates to a lower error on average when the model predicts a value. Again, the ranger function performed the best in this metric by having the smallest MAE of 0.0651. This means that on average, every prediction the ranger model makes has an error of 0.0651 pH.

It's also important to mention that the random forest model had almost the same MAE as the ranger model. The MAE for the random forest model was only 0.0002 greater than the ranger model. The XGB, kNN and SVM models all had similar MAE compared to each other while the lm model had the greatest MAE of the group.

VARIABLE IMPORTANCE

The variable importance refers to how much the model uses a variable to make accurate predictions. Measuring the importance is rather simple thanks to the caret package used in creating the models. Each model already has importance calculated during the training process and therefore all that's left to do is compare. It's also important to note that variable importance is higher for variables that the model relies on more and lower for those that it doesn't. Depending on how we decide to proceed, variable importance can be used to remove variables from the models in order to simplify our models and handle larger datasets much fast. See the variable importance for the top 10 predictors in each chart below:



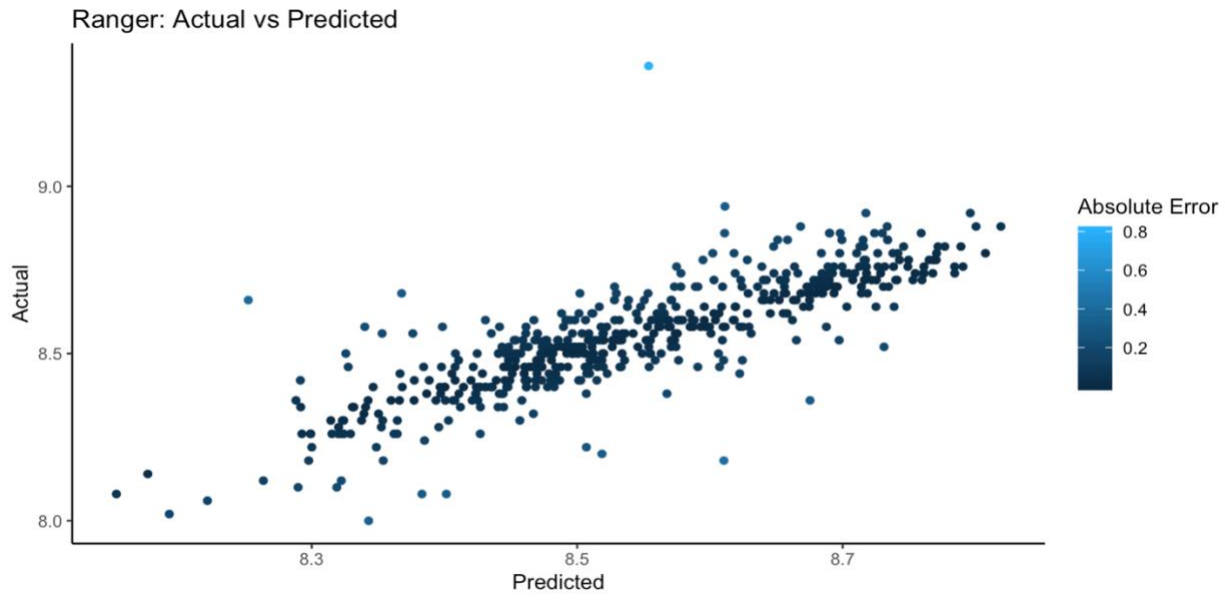
The most notable similarity among all models is that Mnf flow is the top and highest importance scoring predictor for all models. Also, the kNN and SVM models had the same variable importance scores for their variables. The linear, SVM and kNN models also have more variables with higher scored importance predictors meaning that these models rely on more variables to make their predictions than the random forest models. If we look at the random forest models, most of the variables are under an importance score of 20. The XGB model does tend to score its variables a bit higher but not nearly as high as the kNN, SVM or linear models.

The variables that were repeatedly shown having the highest variable importance in each model are Mnf Flow (6/6 models), Usage Count (6/6), Bowl Setpoint (5/6) and Pressure Vacuum (5/6).

CHOOSING THE BEST MODEL

Based on the previously discussed metrics, the ranger model performed the best overall by having the highest R-squared value and the lowest RMSE and MAE values. Although the standard random forest model was up to par with the R-squared and error rate of the ranger model, the ranger model was significantly faster during the training process. If we're looking to apply a model to a much larger dataset, we don't believe that the ordinary random forest model will be the best pick, especially if we add more variables. The ranger model works extremely fast and has slightly better results. Our linear model and kNN models did perform significantly faster than any of our models, however choosing those models would increase our RMSE by at least 23% and our MAE by at least 27%. Not only would error increase by choosing any other model but the R-squared value would drop significantly.

In the chart below we plotted the actual values vs the predicted values for the ranger model:



The graph was created by taking the predicted values of the ranger model and the actual values of the test dataset and plotting them. We then calculated the absolute error for every prediction and assigned each plot a color based on the absolution error. Plots with a lighter blue shade represent a higher absolute error whereas those points that have a darker blue or black shade are closer to the actual value. What we see is a linear relationship with a low absolute error between the ranger model's predicted values and the test data's actual values.

CONCLUSION

QUICK RECAP

Our recommendation is to proceed with the random forest ranger model to predict the pH levels of our beverages. The random forest model is a more complex model compared to some of the other models we decided to evaluate but the results are worthwhile:

- Works as a more efficient random forest model that can provide better results
- Time to train is reasonable but much faster than the original random forest model
- The model explained the most variance in the data (70%) than any other model
- Lowest RMSE and MAE meaning that errors are minimal compared to other models

We believe using the ranger model will help us in monitoring the pH levels of our drinks as we found that to be crucial to our signature taste. It is the most capable model and most accurate.

NEXT STEPS

We were provided with a testing dataset that includes 267 records with the pH values omitted. We loaded this data set and processed it with the same method used on the training set, including handling missing values and creating the new variable HydPressure123. After the preprocessing, we ran our ranger model to predict the pH values of the evaluation data. The results were then saved in an excel file and sent over to our product team to compare against the actual recorded pH values of the beverages and record the Mean Absolute Percent Error (MAPE).

APPENDIX

TOOLS AND PACKAGES

Library	Description
dplyr	Easy data manipulation
plyr	More data manipulation, sub setting, plotting
ggplot2	Data visualization and other plots
purrr	More data manipulation for model methods
tidyr	Data manipulation
Corrplot	Create correlation matrix among variables
MASS	Required for certain modeling methods
caret	Required to build machine learning models
e1071	Required for certain modeling methods
ranger	Required when building the ranger model
xgboost	Required when building the xbg model
knitr	Create rmd file
kableExtra	Styling tables

RESOURCES

- Training dataset:
  - https://github.com/hvasquez81/Data-624/blob/master/Project%202/StudentData%20-%20TO%20MODEL.csv
- Evaluation dataset:
  - https://github.com/hvasquez81/Data-624/blob/master/Project%202/StudentEvaluation-%20TO%20PREDICT.csv
- RMD file:
  - https://github.com/hvasquez81/Data-624/blob/master/Project%202/Project2.Rmd
- Predictions:
  - https://github.com/hvasquez81/Data-624/blob/master/Project%202/Group5_pH_Eval_Predictions.csv

REFERENCES

- Applied Predictive Modeling by Max Kuhn and Kjell Johnson
- Caret package: https://rdrr.io/cran/caret/man/models.html

- Caret modeling: http://www.rebeccabarter.com/blog/2017-11-17-caret_tutorial/
- Ranger method: https://cran.r-projecZt.org/web/packages/ranger/ranger.pdf
- XGB method: https://cran.r-project.org/web/packages/xgboost/vignettes/xgboost.pdf