# DATA 624: Project 2

DATA 624 - Predictive Analytics

Group 2

**Group Members:**

*Vinicio Haro*

*Sang Yoon (Andy) Hwang*

*Julian McEachern*

*Jeremy O'Brien*

*Bethany Poulin*

*10 December 2019*

# Contents

File for final submission of Project 2.

# 1 Executive Summary

Because it is to central to the design of a product's drinking experience, pH is a key performance indicator in the beverage manufacturing process and is tested for and tracked diligently, as the final pH is dependent on and vulnerable to even slight changes in production methods.

Having monitored and recorded these production variables, as well as the final pH, we have the opportunity to improve production outcomes by more closely controlling pH in our beverages with predictive modeling with the potential to catch and correct variations in process which negative impact our taget pH.

[CONTENT EDITORS: ADD IN SUMMARY OF CONCLUSION AND ANY SUPPORTING INSIGHT FROM LAST SECTION]

# 2 Approach

After thorough examination, we approached this task by splitting the provided data into training and test sets. We evaluated several models on this split and found that **what-ever-worked-best** method yielded the best results.

Each group member worked individually to create their own solution. We built our final submission by collaboratively evaluating and combining each others' approaches.

[CONTENT EDITORS: ADD IN FOLLOWING LAST: Our introduction should further outline individual responsibilities. For example, **so-and-so** was responsible for **xyz task**.]

[FORMAT EDITORS: UPDATE BELOW] For replication and grading purposes, we made our code avaliable in the appendix section. This code, along with the provided data, score-set results, and individual contributions, can also be accessed through our group github repository:

- Pretend I'm a working link to R Source Code
- Pretend I'm a working link to Provided Data
- Pretend I'm a working link to Excel Results
- Pretend I'm a working link to Individual Work

# 3 Data Exploration

Preparing the data was the most discussed and influential part of our modeling process. It was clear from early on that in order to build a useful model with such a narrow range of expected pH values, how we groomed our data and the decisions we made would likely be as or more influential than the model we ultimately chose.

The beverage dataset includes 2,571 cases, 32 predictor variables, and a single response variable. One of these predictor variables (Brand Code) is categorical with four levels - A through D; for the purpose of our analysis we interpreted these to represent four distinct beverage brands.

While we found missing observations in both response and predictor variables, in our assessment the extent of NAs did not suggest a systemic issue in measurement or recording that imputing values could not remedy. For context: - The response variable (PH) is missing a total of four observations (< 1%). - Most (30) predictor variables are missing at least one observation, but only eleven are missing more than 1% of total cases and only three are missing more than 2% of total cases. These are: 1. MFR (continuous, 8.2%) 2. BrandCode (categorical, 4.7%) 3. and FillerSpeed (continuous, 2.2%)

[CONTENT EDITORS: DO WE STILL WANT TO CREATE MISSING DATA TABLE? IF SO, MISSINGDATA OBJECT NEEDS TO

BE REBUILT IN MODEL_PREP.R]

```
## TO DO
Tbl_Top_MissingData <- MissingData %>% top_n(11, n) %>% column_to_rownames("predictor") %>%
 kable(caption = "Variables with Highest Frequency of NA Values",
  booktabs = T, digits = 1) %>% kable_styling() %>%
 row_spec()

Tbl_Top_MissingData
```

## Response Variable

[CONTENT EDITORS: DO WE STILL WANT TO CREATE VARIABLE HISTOGRAMS?]

```
## TO DO (remove echo/eval when ready)

Plt_pH1 <- StudentData %>% select(PH) %>% mutate(type = "pH") %>%
 ggplot(aes(PH)) + geom_histogram(aes(y = ..density..),
 bins = 40, fill = "#57A0D3", alpha = 0.65) + geom_density(alpha = 0.2,
 color = "#000000", size = 0.65) + scale_x_continuous() +
 scale_y_continuous(limits = c(0, 3.5)) + labs(x = "",
 y = "") + theme_bw() + theme(axis.title.x = element_blank(),
 axis.ticks.length.x = unit(0, "cm")) + facet_wrap(~type)

Plt_pH2 <- StudentData %>% select(PH) %>% mutate(type = "pH") %>%
 ggplot(aes(PH, "")) + geom_boxplot(fill = "#57A0D3",
 outlier.colour = "#4682B4", alpha = 0.65) + theme_bw() +
 theme(legend.title = element_blank(), strip.background = element_blank(),
  strip.text.x = element_blank(), axis.title.x = element_blank(),
  axis.ticks.length.x = unit(0, "cm")) + labs(x = "",
 y = "") + scale_y_discrete(labels = "pH") + scale_x_continuous() +
 facet_wrap(~type, nrow = 1, strip.position = "top")

Plt_pH3 <- StudentData %>% filter(!is.na(`Brand Code`)) %>%
 ggplot(aes(PH, "")) + geom_boxplot(aes(fill = `Brand Code`),
 outlier.colour = "#4682B4", alpha = 0.3) + scale_fill_manual() +
 theme_bw() + theme(legend.position = "none", strip.background = element_blank(),
 strip.text.x = element_blank(), axis.title.x = element_blank(),
 axis.text.y = element_blank()) + labs(x = "", y = "") +
 scale_x_continuous() + facet_wrap(~`Brand Code`, nrow = 1,
 strip.position = "top", scales = "fixed")

Plt_pH4 <- StudentData %>% select(PH, `Brand Code`) %>% filter(!is.na(`Brand Code`)) %>%
 ggplot(aes(PH)) + geom_histogram(aes(y = ..density..,
 fill = `Brand Code`), bins = 20, alpha = 0.65) + geom_density(alpha = 0.2,
 color = "#000000", size = 0.65) + scale_fill_manual() +
 scale_x_continuous() + scale_y_continuous(limits = c(0,
 3.5)) + labs(x = "", y = "") + facet_wrap(~`Brand Code`,
 nrow = 1) + theme_bw() + theme(axis.title.x = element_blank(),
 axis.ticks.length.x = unit(0, "cm"), axis.text.y = element_blank(),
 legend.position = "none")
```

```
Plt_pH_lay <- rbind(c(1, 2, 2), c(3, 4, 4))

grid.arrange(Plt_pH1, Plt_pH4, Plt_pH2, Plt_pH3, layout_matrix = Plt_pH_lay,
 heights = c(2, 1), padding = unit(0, "cm"))
```

The response variable PH is a logarithmically scaled measure of how acidic or basic a water-based solution is (https://en.wikipedia.org/wiki/PH). It ranges from 0 (acidic) and to 14 (alkaline); 7 is neutral (e.g. room temperature water).

In aggregate, PH distribution is approximately normal and centered around 8.546 (i.e. slightly base), with some negative skew / outliers. When evaluated by BrandCode: - A (293 observations) appears to be multimodal and have the most outliers, with a mean slightly lower than the aggregate (8.495) - B (1293 observations) appears to be bimodal with a number of outliers, as well as a mean nearest the aggregate (8.562) - C (304 observations) appears to be bimodal and is the most acid (8.419) - D (615 observations) is the most normal distribution and also has the highest alkalinity (8.603)

## Predictor Variables

We examined the density of our variables to visualize the distribution of the predictors. Many of these variables contain outliers and present with a skewed distribution. The outliers fall outside the red-line boundaries, and highlight which predictors have heavier tails.

The density plots also contain an overlay of the only categorical indicator, `BrandCode`. This view shows us that some variables, including `AlchRel`, `CarbRel`, `CarbVolume`, `HydPressure4`, and `Tempature`, are strongly influenced by brand type.

[CONTENT EDITORS: DO WE STILL WANT TO CREATE THESE TABLES? IF SO, OUTLIER_WITH OBJECT NEEDS TO BE REBUILT IN MODEL_PREP.R]

```
## TO DO (remove echo/eval when ready)

Plt_Outlier1 <- ggplot(outlier_with, aes(value)) + geom_density(aes(fill = `Brand Code`),
 color = "#999999", alpha = 0.3) + labs(title = "Density Distribution of Numeric Predictor Variables",
 subtitle = "With Outliers", x = "", y = "") + geom_vline(data = outlier_with,
 mapping = aes(xintercept = outlier_lower), color = "#ff8080") +
 geom_vline(data = outlier_with, mapping = aes(xintercept = outlier_upper),
  color = "#ff8080") + facet_wrap(~key, scales = "free",
 nrow = 3) + theme_bw() + theme(axis.text.y = element_blank(),
 axis.title.x = element_blank(), axis.text.x = element_blank(),
 legend.position = "none") + scale_fill_manual()

Plt_Outlier2 <- ggplot(outlier_wo, aes(value)) + geom_density(aes(fill = `Brand Code`),
 color = "#999999", alpha = 0.3) + labs(subtitle = "Without Outliers",
 x = "", y = "") + facet_wrap(~key, scales = "free", nrow = 1) +
 theme_bw() + theme(axis.text.y = element_blank(), axis.title.x = element_blank(),
 axis.text.x = element_blank(), legend.position = "bottom",
 legend.key.size = unit(0.4, "cm")) + scale_fill_manual()

grid.arrange(Plt_Outlier1, Plt_Outlier2, nrow = 2, heights = c(3,
 2))
```

As no predictor variable shows a particularly pronounced monotonic linear relationship with response, a non-linear approach to modeling seems warranted.

[FIGURE SUCH AND SUCH] helps to further visualize the effect `BrandCode` has on our predictor and `pH` values. For example, `AlchRel` shows distinct `BrandCode` groupings. Other variables, such as `PSC02`, `BowlSetpoint`, `MinFlow`, and `PressureSetup` show unique features likely related to system processes.

[CONTENT EDITORS: DO WE STILL WANT TO CREATE THESE TABLES? IF SO, OUTLIER_WITH OBJECT NEEDS TO BE REBUILT IN MODEL_PREP.R]

```
## TO DO (remove echo/eval when ready)
Plt_Scatter1 <- outlier_with %>% mutate(PH = as.numeric(as.character(PH))) %>%
 ggplot(aes(x = value, y = PH)) + geom_jitter(aes(color = `Brand Code`),
 alpha = 0.15) + stat_smooth(color = "#000000", method = "loess") +
 labs(title = "pH~Predictor Scatterplots", subtitle = "Without Outliers",
  x = "", y = "") + facet_wrap(~key, scales = "free_x",
 nrow = 3) + theme_bw() + theme(axis.title.x = element_blank(),
 axis.text.x = element_blank(), legend.position = "none",
 legend.key.size = unit(0.4, "cm")) + scale_color_manual(values = c("#999999",
 "#95C8D8", "#008081", "#034D92"))


Plt_Scatter2 <- outlier_wo %>% mutate(PH = as.numeric(as.character(PH))) %>%
 ggplot(aes(x = value, y = PH)) + geom_jitter(aes(color = `Brand Code`),
 alpha = 0.15) + stat_smooth(color = "#000000", method = "loess") +
 labs(subtitle = "Without Outliers", x = "", y = "") +
 facet_wrap(~key, scales = "free_x", nrow = 1) + theme_bw() +
 theme(axis.title.x = element_blank(), axis.text.x = element_blank(),
  legend.position = "bottom", legend.key.size = unit(0.4,
   "cm")) + scale_color_manual(values = c("#999999",
 "#95C8D8", "#008081", "#034D92"))

grid.arrange(Plt_Scatter1, Plt_Scatter2, nrow = 2, heights = c(3,
 2))
```

Collinearity measures between numeric predictors indicate that several of these variables are heavily related, with correlation values exceeding $\pm 0.7$.

```
## TO DO (remove echo/eval when ready)

Plt_Corr <- StudentData %>% select_if(is.numeric) %>% select(-PH) %>%
 ggcorr(method = "pairwise.complete.obs", geom = "tile",
  label = F, hjust = 0.95, layout.exp = 7, label_round = 1,
  low = "#95C8D8", mid = "grey90", high = "#034D92") +
 theme_bw() + theme(legend.key.size = unit(0.6, "cm"),
 legend.justification = c(0.05, 0.98), legend.position = c(0.05,
  0.98)) + labs(title = "Predictor Variable Correlation Matrix")

g = ggplot_build(Plt_Corr)
g$data[[2]]$size = 2

grid::grid.draw(ggplot_gtable(g))
```

# 4 Data Preparation

In our exploration, we detected missing data, extreme outliers, and multicollinearity. We kept these factors in mind and applied strategic transformations when preparing our models to evaluate their performance with and without normalization changes.

**Train/Test Splits:**

Prior to all pre-processing, we divided the production dataset using an 80/20 split to create a train and test sets.

[BETHANY / JULIANN: PLEASE CONFIRM THIS IS STILL ACCURATE:

All models incorporated k-folds cross-validation set at 10 folds to protect against overfitting the data. We set up unique model tuning grids to find the optimal parameters for each regression type to ensure the highest accuracy within our predictions.

For both KNN and SVM models a grid of seeds was created from our original seed to ensure that out repeated cross validation would be repeateable. The same seeds were used in both the SVM and KNN.

]

**Data Imputation:**

Missing values are imputed using the `caret` package so that the same range of imputed values could be applied to the test and validation sets without confounding our training data and a bagging algorithm was used to impute all continuous variables.

Because we were convinced that the ’brand variable `BrandCode` may be one of the strongest predictors of pH, after much discussion, we decided not to impute the Brand Code variable, so that each of the observations with a known brand would be more accurately described by the other variables relative to pH.

Instead the missing labels were replaces with Unknown and the variables were converted to dummies of 0 and 1 to ensure that all modeling methods would be able to consider Brand Code.

Test data is imputed with the same model, with that target variable `PH` removed from the set.

**Pre-Processing:**

Most of the models concidered in our modeling process require scaling and centering, so we included this in our prepartions. Although, only one variable showed near-zero variance, Hyde.Pressure_1 we opted to remove it from all models during preprocessing and likewise applied Box-Cox conversions to the data to compensate for andy skews and non-normal modaliteis in the variables which might confound our models. Again, the preprocessing model was saved so that the test and validation sets could be consistenly transformed using caret’s predict method.

# 5 Modeling

We assessed the effectiveness of more than ten different non-linear regression models in our exploratory process. We settled on four models that exhibited the most favorable test metrics, tuned those models, and then chose the best performing model of that set to use in our final analyses (all performance results from the other five are included in [TABLE BLAH BLAH] in [APPENDIX BLAH BLAH]).

[BETHANY: INSERT SIDE-BY-SIDE TRAINING / TESTING METRICS FOR PREPRCESSED MODELS (SET 2) HERE].

- Model 1: Support Vector Machines Regression
- Model 2: Cubist Tree Regression

- Model 3: Multivariate Adaptive Regression Splines Regression
- Model 4: Random Forest Regression

# 6 Model Performance

- Set1 = Caret: bagImputed; no additional pre-processing

- Set2 = Caret: bagImputed; PreP `method=c('center', 'scale', 'nzv', 'BoxCox')`

**Train Performance:**

Table 6.1: Train1 Performance

| MAPE | RMSE | RSquared | MAE | Method |
|---|---|---|---|---|
| 1.0948 | 0.5916 | 0.6757 | 0.4319 | rf |
| 1.1629 | 0.5500 | 0.6979 | 0.3938 | cubist |
| 1.2664 | 0.6985 | 0.5165 | 0.5037 | svmRadial |
| 1.5170 | 0.6921 | 0.5261 | 0.5153 | earth |

Table 6.2: Train2 Performance

| MAPE | RMSE | RSquared | MAE | Method |
|---|---|---|---|---|
| 0.0081 | 0.0965 | 0.6904 | 0.0688 | cubist |
| 0.0088 | 0.1030 | 0.6735 | 0.0751 | rf |
| 0.0104 | 0.1224 | 0.5042 | 0.0883 | svmRadial |
| 0.0112 | 0.1265 | 0.4690 | 0.0954 | earth |

**Test Accuracy:**

Table 6.3: Test1 Performance

| MAPE | RMSE | Rsquared | MAE | Method |
|---|---|---|---|---|
| 0.4930 | 0.2234 | 0.9539 | 0.1578 | cubist |
| 0.5351 | 0.3265 | 0.9568 | 0.2466 | rf |
| 0.9987 | 0.6076 | 0.6385 | 0.4101 | svmRadial |
| 1.6688 | 0.7260 | 0.5226 | 0.5269 | earth |

Table 6.4: Test2 Performance

| MAPE | RMSE | Rsquared | MAE | Method |
|---|---|---|---|---|
| 0.0029 | 0.0367 | 0.9586 | 0.0250 | cubist |
| 0.0037 | 0.0436 | 0.9596 | 0.0311 | rf |
| 0.0084 | 0.1064 | 0.6307 | 0.0717 | svmRadial |
| 0.0107 | 0.1193 | 0.5257 | 0.0908 | earth |

[BETHANY: EACH OF US SHOULD WRITE BULLETS WITH REASONS TO CHOOSE THIS MODEL BY SAT EVENING 12/7 - BETHANY WILL FLESH OUT]

## Model Selection Considerations

[BETHANY: NEED TO PICK OUR FIRST CHOICE MODEL, THINK IS SHOULD BE VARIMP-ABLE SO WE CAN USE THAT IN INTERPRETATION / CONCLUSIONS]

## Model 1: Support Vector Machines (SVM) Regression

[BETHANY TO WORDSMITH RATIONALE FOR USING SVM MODEL]

The support vector machine, although less efficient than the k-nearest neighbor to train, provided robust final model using a radial kernel with a cost of 10, passed as the tune length settling on $\sigma = 0.020$ and $cost = 8$ returning a $RMSE = 0.1127$

```
# remove echo/eval later [BETHANY: ADD IN VARIMP /
# PERFORMANCE CHART FOR SVM AS ALIGNED WITH GROUP]
```

## Model 2: Cubist Tree Regression

[JEREMY: CONDENSING BELOW WITH RATIONALE FOR USING MODEL IN BULLET FORM BY EVENING SAT 12/7 - BETHANY WILL WORDSMITH]

[BETHANY: PLEASE IGNORE BELOW DESCRIPTION UNTIL SAT EVENING

For a continuous response variable, a rule-based Cubist model functions like a piecewise linear model: each rule is a conjunction of conditions associated with a linear expression, and those rules can overlap with each other. Adding to the interpretive complexity of those rules, Cubist models can also integrate an instance-based, nearest-neighbor approach that performs a composite prediction based on actual values of neighbors, predicted values of neighbors, and predicted values of observations of interest.

Accordingly, hyper-parameters for Cubist models include: - The number of rule-based models, or committees - these issue separate predictions that are averaged (5 recommended to balance computational cost with ensemble benefits) - The number of neighbors over which to predict response values based on similar training observations

Based on cross-validation and a grid search across hyper-parameters, we found the best RMSE performance with an instance-based model that factoring in many neighbors built on non-pre-processed training data.

References: Background: https://www.rulequest.com/cubist-win.html Overview: https://static1.squarespace.com/static/51156277e4b0b8b2ffe11c00/t/56e3056a3c44d8779a61988a/1457718645593/cubist_BRUG.pdf Mechanics: http://ftp.uni-bayreuth.de/math/statlib/R/CRAN/doc/vignettes/caret/caretTrain.pdf]

```
# remove echo/eval later [JEREMY: ADD IN VARIMP /
# PERFORMANCE CHART FOR SVM AS ALIGNED WITH GROUP]
```

## Model 3: Multivariate Adaptive Regression Splines (MARS) Regression

[VINICIO / JULIANN: PLEASE ADD CONCISE BULLETS FOR USING MARS MODEL BY EVENING SAT 12/7 - BETHANY WILL WORDSMITH]

MARS modeling was selected to assess the non-linear features in our data. This method uses a weighted sum to models non-linearities and interactions between variables. The model assesses cut-points between features that create the smallest error and prunes insignificant points to improve model accuracy.

Our RMSE Cross Validation plots show us that pre-processing transformations did not have improve the MARS model. The model performed best on our training data when no transformations were applied.

```
# remove echo/eval later [VINICIO / JULIANN: ADD IN
# VARIMP / PERFORMANCE CHART FOR SVM AS ALIGNED WITH
# GROUP]
```

### Model 4: Random Forest Regression

[ANDY: PLEASE ADD CONCISE BULLETS WITH RATIONALE FOR USING RF MODEL BY EVENING SAT 12/7 - BETHANY WILL WORDSMITH]

The optimal parameters for model was mtry = 31 and ntree = 2500. MAPE is **r s$MAPE** where as top 3 important predictors are `MnfFlow`, `BrandCode` and `PressureVacuum` for %incMSE and `MnfFlow`, `BrandCode` and `OxygenFiller` for IncNodePurity. Unlike `PLS`, `Random Forest` can produce 2 different variable importance plots.

The first graph shows how much MSE would increase if a variable is assigned with values by random permutation. The second plot is based on `node purity` which is measured by the difference between RSS before and after the split on that variable (`Gini Index`). In short, each graph shows how much MSE or Impurity increases when each variable is randomly permuted.

```
# remove echo/eval later [ANDY: ADD IN VARIMP /
# PERFORMANCE CHART FOR SVM AS ALIGNED WITH GROUP]
```

# 7 Interpretation

[BETHANY MAKING MAGIC HAPPEN WITH APPROPRIATE VARIMP GRAPH ONCE FINAL MODEL SELECTED]

# 8 Conclusion

[BETHANY CREATING NEXT STEPS FOR PRODUCTION PROCESS BASED ON FINAL MODEL]

# Appendix

#### Code

#### Data Dictionary

#### Exploratory Plots and List Models

# 9 Citations

Shelton, Robert B. “PH Values Of Common Drinks.” Robert B. Shelton, DDS MAGD Dentist Longview Texas, 2019, www.sheltondentistry.com/patient-information/ph-values-common-drinks/.

Cubist Model Background: https://www.rulequest.com/cubist-win.html Cubist Model Overview: https://static1.squarespace.com/static/51156277e4b0b8b2ffe11c00/t/56e3056a3c44d8779a61988a/1457718645593/cubist_BRUG.pdf Cubist Model Mechanics: http://ftp.uni-bayreuth.de/math/statlib/R/CRAN/doc/vignettes/caret/caretTrain.pdf]