PROJECT 2: PREDICTING PH

DATA 624 - Predictive Analytics

Group 2

Group Members:

Juliann McEachern

10 December 2019

Contents


```
library(tidyverse); library(readxl); library(psych); library(ggplot2); library(mice); library(xtable); l
```

Introduction

This project is designed to evaluate production data from a beverage manufacturing company. Our assignment is to predict PH, a Key Performance Indicator (KPI), with a high degree of accuracy through predictive modeling. 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. Our introduction should further outline individual responsibilities. For example, **so-and-so** was responsible for **xyz task**.

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

1 Data Exploration

The beverage manufacturing production dataset contained 33 columns/variables and 2,571 rows/cases. In our initial review, we found that the response variable, PH, had four missing observations.

We also identified that 94% of the predictor variables had missing data points. Despite this high occurance, the NA values in the majority of these predictors accounted for less than 1% of the total observations. Only eleven variables were missing more than 1% of data.

```
Tbl_Top_MissingData
```

Table 1.1: Variables with Highest Frequency of NA Values

	MFR	BrandCode	FillerSpeed	PCVolume	PSCCO2	FillOunces	PSC	CarbPressure1	HydPressure4	CarbPressure	CarbTemp
n	208.0	120.0	54.0	39.0	39.0	38.0	33.0	32.0	28.0	27.0	26
%	8.1	4.7	2.1	1.5	1.5	1.5	1.3	1.2	1.1	1.1	1

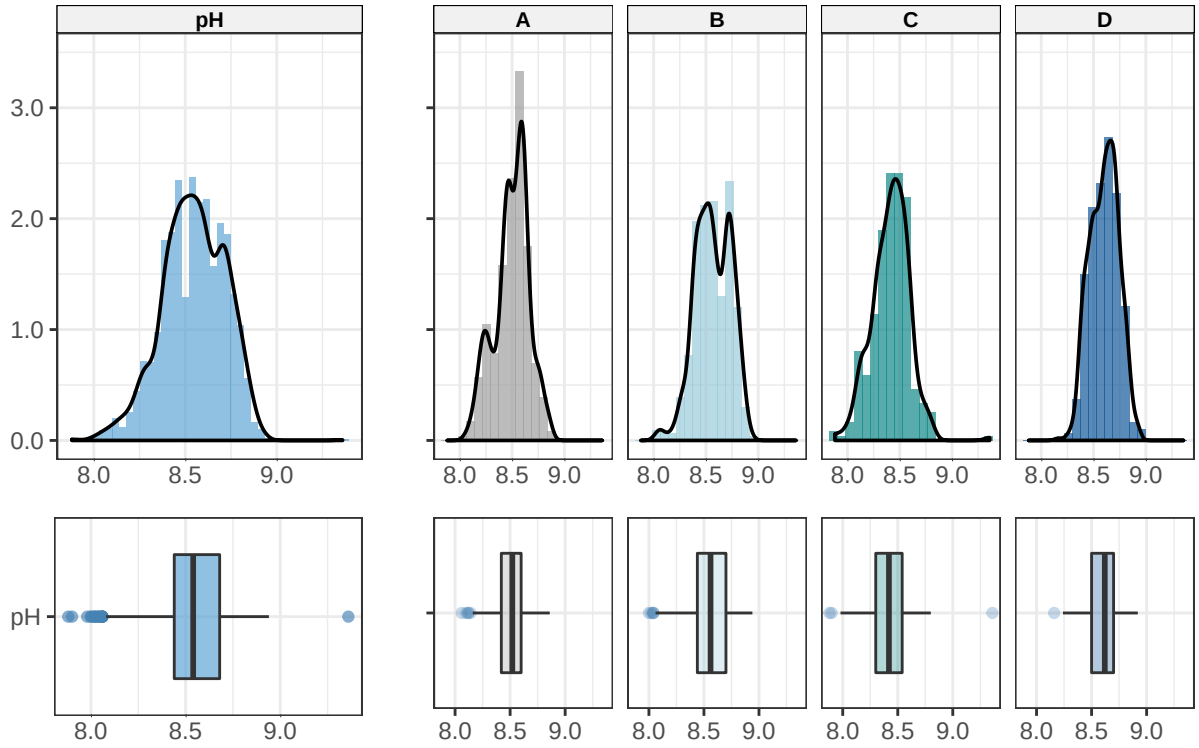
Response Variable

```
grid.arrange(Plt_pH1, Plt_pH4, Plt_pH2, Plt_pH3, layout_matrix = Plt_pH_lay, heights=c(2,1), padding=uni
```

Understanding the influence pH has on our predictors is key to building an accurate predictive model. pH is a measure of acidity/alkalinity that must conform in a critical range. The value of pH ranges from 0 to 14, where 0 is acidic, 7 is neutral, and 14 is basic.

Figure 1.1 shows that our response distribution follows a somewhat normal pattern and is centered around 8.5. The histogram for pH is bimodal in the aggregate, but varies by brand. The boxplot view allows us to better visualize the effect outliers have on the skewness within our target variable.

Fig. 1.1: Distribution of Response Variable: pH



Brand A has a negatively skewed, multimodal distribution, which could be suggestive of several distinct underlying response patterns or a higher degree of variation in pH response for this brand. The density plot and histogram for Brand B show two bimodal peaks with a slight positive skew. These peaks indicate that this brand has two distinct response values that occur more frequently. The distribution for Brand C and D are both more normal, with a slight negative skew. Brand D has the highest median pH value and Brand C has the lowest. Brand C also appears to have the largest spread of pH values.

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.

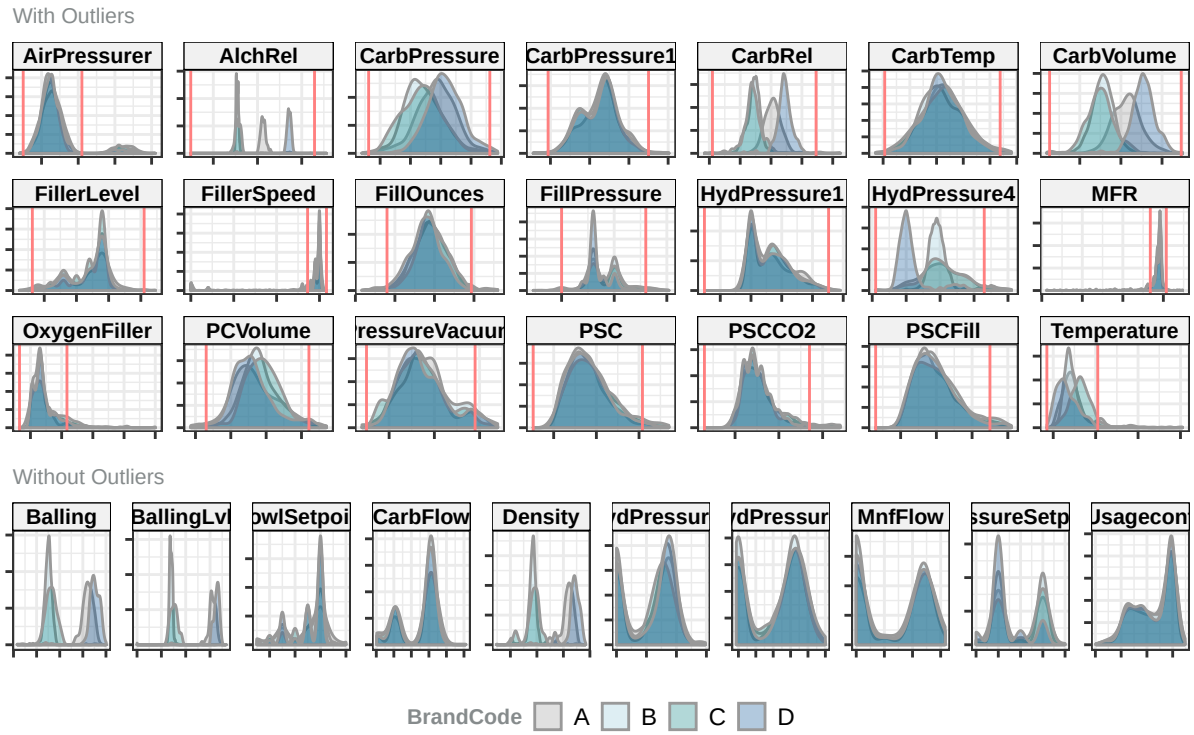
The density plots also contain an overlay of the only categorical indicator, `BrandCode`. This view shows us that some variables, including `AlchRel`, `CarbRel`, `CarbVolume`, `HydPressure4`, and `Tempature`, are strongly influenced by brand type.

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

We also looked at the relationship of our predictors against the response variable below. There are a few predictors that have a weak, linear association with our response variable. However, most of the indicators show no strong patterns. Given these trends, we do not expect linear modeling to provide optimal predictions for pH.

This view helps us 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.

Fig. 1.2: Box-Plot Distribution of Numeric Predictor Variables




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

Lastly, we examined collinearity measures between our numeric predictors and found that 10 of these variables were heavily related, with correlation values exceeding ± 0.75. Linear modeling requires predictor variables to be independent of one another. This final evaluation confirms that standard linear modeling should not be used as it would require removing and transforming a significant proportion of our predictors, which would potentially overfit our data and lead to unreliable predictions.

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

2 Data Preparation

In our exploration, we detected missing data, extreme outliers, and multicollinearity. We selected our modeling methods keeping these factors in mind. Our approach included the application strategic transformations to evaluate several types of non-linear and tree-based modeling.

We divided the production dataset using an 80/20 split to create a train and test set. 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.

Data Imputation

We choose to drop the complete cases of all pH observations with null data in the target as they accounted for such a small proportion (< 0.002%) of the observations. We compared this approach with other types of imputation and found that dropping

Fig. 1.3: pH~Predictor Scatterplots

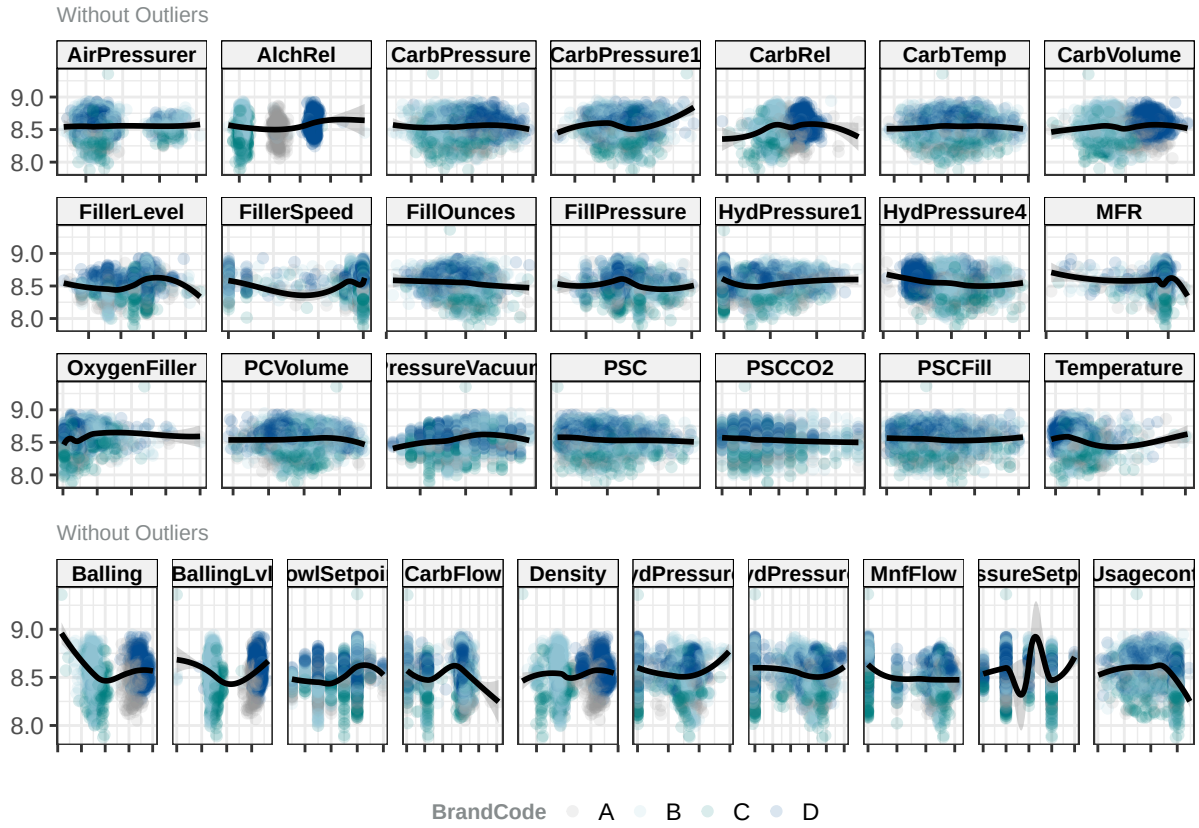
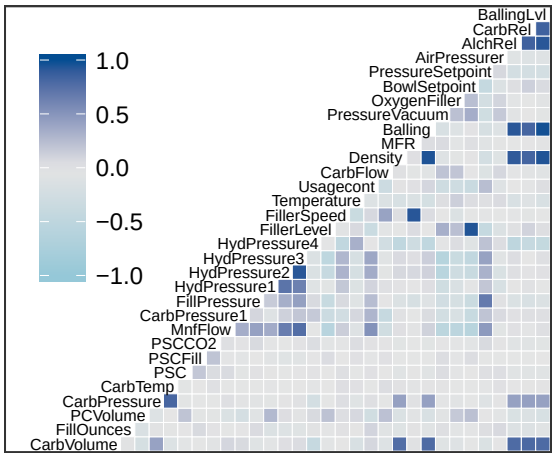


Fig. 1.4: Predictor Variable Correlation Matrix



variables, in this instance, provided a slight boost within our accuracy measures.

For our predictor variables, we applied a Multiple Imputation by Chained Equations (MICE) algorthim to predict the missing data using sequential regression. This method filled in all incomplete cases, including `BrandCode`, our one unordered categorical variable.

Pre-Processing

Decision models trees are robust against these identifed issues. Thus, requiring minimal data tranformation to properly evaluate our data. Our non-linear models required more attention, so we incorporated variable centering and scaling into the pre-processing to maximize their performance.

We tested the effect of box-cox transformations on our non-linear and tree-based models. The box-cox method changes the distribution shape of our predictor variables, which normalizes the scale that they are evaluated on. We found this transformation improve our modeling outcomes in some instances.

3 Modeling

For our non-linear approach, we selected Support Vector Machines (SVM), Multivariate Adaptive Regression Splines (MARS), and Elastic Net (eNet) models. We compared these methods to tree-based regression using gradient boosted models (GBM).

SVM

For SVM, we choose to work with a non-linear, radial kernel because many of our data's features did not appear to be linearly separable. SVM models work well in maintain a large amount of features and can make distinctions between class differences.

Our RMSE Cross Validation plots show us that bost models performed similiary, but the second model performed slightly better. We choose this model, which contained box-cox, centering, and spread transformations, to be our preferred SVM model.

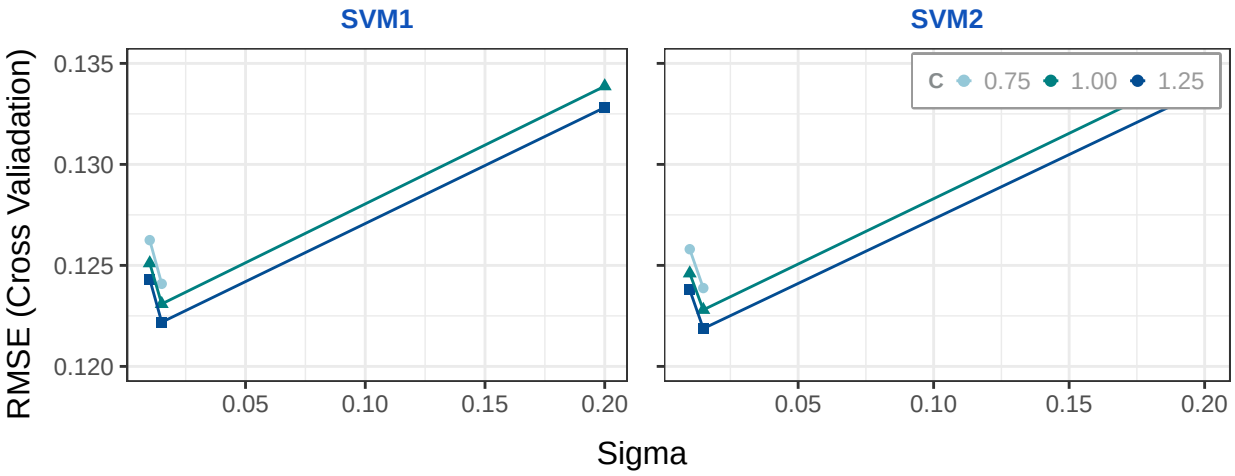
```
grid.arrange(svm1_plot, svm2_plot, nrow=1, left=textGrob("RMSE (Cross Valiadation)", rot=90), bottom = t
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_path).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_path).
```

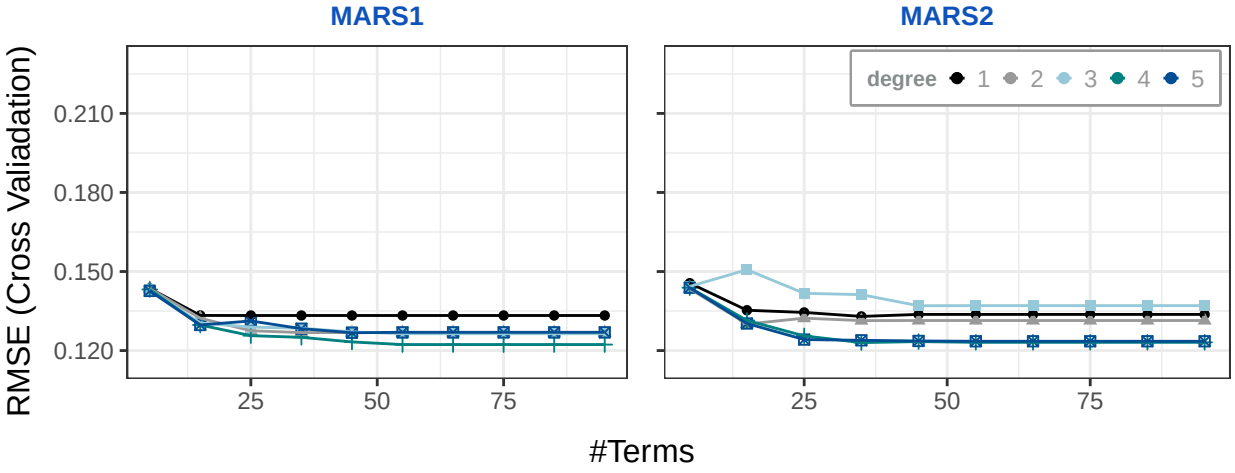


MARS

MARS modeling was also selected to assess the non-linear features in our data. This method uses a weighted sum to models nonlinearities 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 the best tune for both MARS were very similiar. The second model, with box-cox transformations, performed the most consistently, thus we selected this as our preferred MARS model.

```
grid.arrange(mars1_plot, mars2_plot, nrow=1, left=textGrob("RMSE (Cross Valiadation)", rot=90), bottom =
```

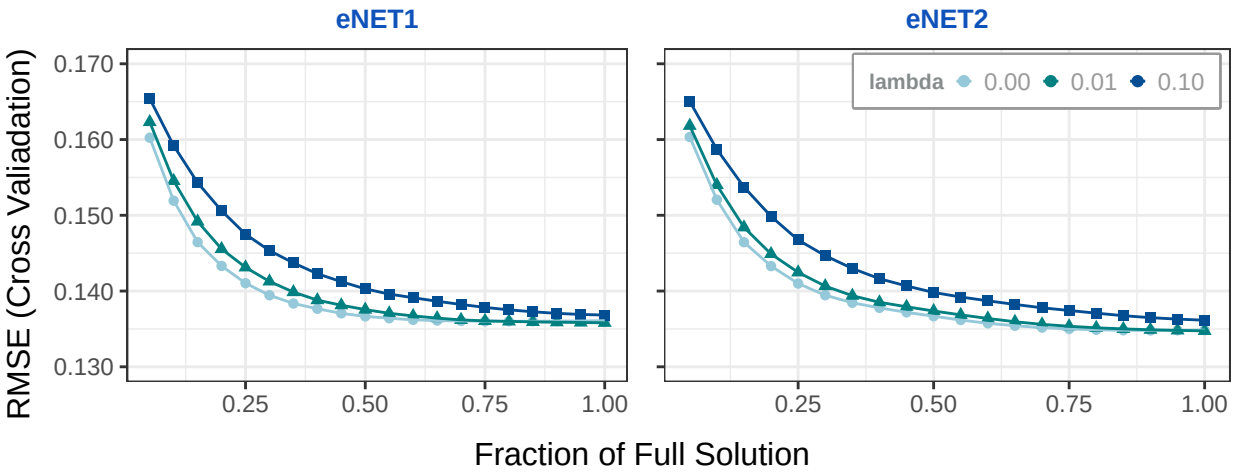


eNET

The Elasticnet model was used as it can handle a large number of predictor variables. It combines ridge and lasso regression techniques to reduce the size of coefficients.

Our RMSE Cross Validation plots show us that the best tune for both eNET were also similiar, with eNET1 performing slightly better. This model center and scaled the data but did not apply a box-cox transformations.

```
grid.arrange(enet1_plot, enet2_plot, nrow=1, left=textGrob("RMSE (Cross Valiadation)", rot=90), bottom =
```



GBM

```
grid.arrange(gbm1_plot,gbm2_plot, nrow=1)
```

4 Regression Analysis

This section will discuss the highlights and draw back of our tested models. I am withholding content until we review and merge.

Accuracy

MAPE not currently working for selected SVM/MARS train model. NaN is indicative of a zero in the calculations. Will address in final merge. The SVM model achieved the highest accuracy measures for non-linear modeling and the GBM model outperformed all of those other attempts.

```
Tbl_Accuracy
```

Table 4.1: Accuracy Measures

	SVM_Train	SVM_Test	MARS_Train	MARS_Test	eNET_Train	eNET_Test	GBM_PERF_TST	gbm1_Train
RMSE	0.12187	0.11057	0.12303	0.11669	0.13581	0.12581	0.10157	0.10622
Rsquared	0.51456	0.55908	0.50555	0.52490	0.39269	0.42688	0.62633	0.62788
MAE	0.08861	0.08142	0.09286	0.08964	0.10506	0.09740	0.07589	0.07894
MAPE	0.01110	0.00958	0.01142	0.01053	0.01298	0.01061	0.01021	0.00891

Fig. 3.1: GBM RMSE Cross-Validated Profile

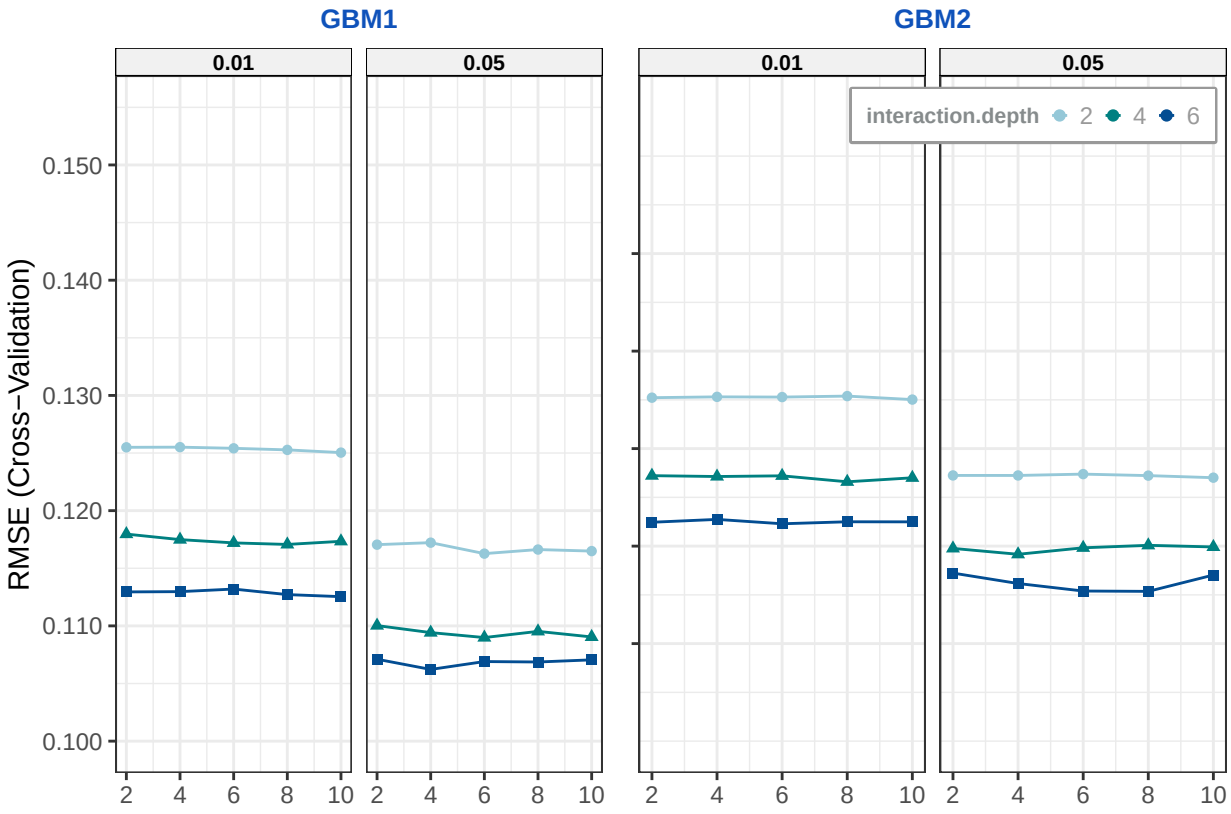
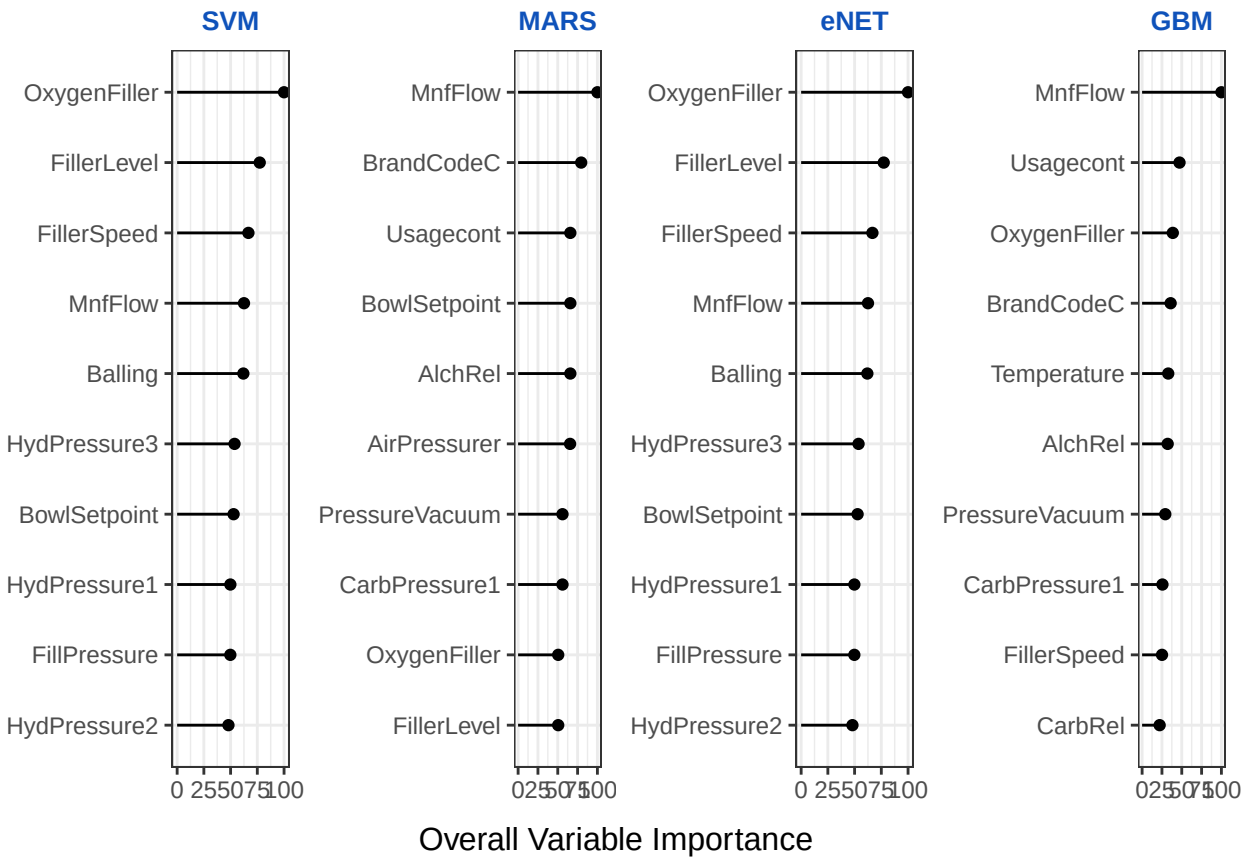


Fig. 4.1: Top 10 Important Variables by Model



Variable Importance

This section will discuss the trends in variable features in our selected models.

```
grid.arrange(Plt_SVM_VarImp, Plt_MARS_VarImp, Plt_eNET_VarImp, Plt_GBM_VarImp, nrow=1, bottom = textGrob
```

5 Conclusion

I will save sprusing up this section once everyones models are live and selected for final analysis.

Appendix

Summary Statistics

```
Tbl_summary_stats
```

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
BrandCode*	1	2447	2.5	1.0	2.0	2.5	0.0	1.0	4.0	3.0	0.4	-1.1	0.0
CarbVolume	2	2557	5.4	0.1	5.3	5.4	0.1	5.0	5.7	0.7	0.4	-0.5	0.0
FillOunces	3	2529	24.0	0.1	24.0	24.0	0.1	23.6	24.3	0.7	0.0	0.9	0.0
PCVolume	4	2528	0.3	0.1	0.3	0.3	0.1	0.1	0.5	0.4	0.3	0.7	0.0
CarbPressure	5	2540	68.2	3.5	68.2	68.1	3.6	57.0	79.4	22.4	0.2	0.0	0.1
CarbTemp	6	2541	141.1	4.0	140.8	141.0	3.9	128.6	154.0	25.4	0.2	0.2	0.1
PSC	7	2534	0.1	0.0	0.1	0.1	0.0	0.0	0.3	0.3	0.9	0.7	0.0
PSCFill	8	2544	0.2	0.1	0.2	0.2	0.1	0.0	0.6	0.6	0.9	0.8	0.0
PSCCO2	9	2528	0.1	0.0	0.0	0.0	0.0	0.0	0.2	0.2	1.7	3.7	0.0
MnfFlow	10	2567	24.6	119.5	70.2	21.1	161.6	-100.2	229.4	329.6	0.0	-1.9	2.4
CarbPressure1	11	2535	122.6	4.7	123.2	122.5	4.4	105.6	140.2	34.6	0.0	0.1	0.1
FillPressure	12	2549	47.9	3.2	46.4	47.7	2.4	34.6	60.4	25.8	0.5	1.4	0.1
HydPressure1	13	2556	12.5	12.4	11.4	10.9	16.9	-0.8	58.0	58.8	0.8	-0.1	0.2
HydPressure2	14	2552	21.0	16.4	28.6	21.1	13.3	0.0	59.4	59.4	-0.3	-1.6	0.3
HydPressure3	15	2552	20.5	16.0	27.6	20.5	13.8	-1.2	50.0	51.2	-0.3	-1.6	0.3
HydPressure4	16	2539	96.3	13.1	96.0	95.5	11.9	62.0	142.0	80.0	0.6	0.6	0.3
FillerLevel	17	2551	109.3	15.7	118.4	111.0	9.2	55.8	161.2	105.4	-0.8	0.0	0.3
FillerSpeed	18	2513	3688.1	769.6	3982.0	3920.2	47.4	998.0	4030.0	3032.0	-2.9	6.8	15.4
Temperature	19	2555	66.0	1.4	65.6	65.8	0.9	63.6	76.2	12.6	2.4	10.3	0.0
Usagecont	20	2562	21.0	3.0	21.8	21.3	3.2	12.1	25.9	13.8	-0.5	-1.0	0.1
CarbFlow	21	2565	2472.1	1070.4	3030.0	2604.2	323.2	26.0	5104.0	5078.0	-1.0	-0.6	21.1
Density	22	2567	1.2	0.4	1.0	1.2	0.1	0.2	1.9	1.7	0.5	-1.2	0.0
MFR	23	2359	704.0	73.9	724.0	718.2	15.4	31.4	868.6	837.2	-5.1	30.5	1.5
Balling	24	2567	2.2	0.9	1.6	2.1	0.4	0.2	4.0	3.9	0.6	-1.4	0.0
PressureVacuum	25	2567	-5.2	0.6	-5.4	-5.3	0.6	-6.6	-3.6	3.0	0.5	0.0	0.0
PH	26	2567	8.5	0.2	8.5	8.6	0.2	7.9	9.4	1.5	-0.3	0.1	0.0
OxygenFiller	27	2556	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.4	2.4	8.8	0.0
BowlSetpoint	28	2565	109.3	15.3	120.0	111.4	0.0	70.0	140.0	70.0	-1.0	-0.1	0.3
PressureSetpoint	29	2555	47.6	2.0	46.0	47.6	0.0	44.0	52.0	8.0	0.2	-1.6	0.0
AirPressurer	30	2567	142.8	1.2	142.6	142.6	0.6	140.8	148.2	7.4	2.3	4.7	0.0
AlchRel	31	2560	6.9	0.5	6.6	6.8	0.1	5.3	8.6	3.3	0.9	-0.9	0.0
CarbRel	32	2559	5.4	0.1	5.4	5.4	0.1	5.0	6.1	1.1	0.5	-0.3	0.0
BallingLvl	33	2566	2.1	0.9	1.5	2.0	0.2	0.0	3.7	3.7	0.6	-1.5	0.0

Code

```
#Final R Code Will Be Inserted Here
```