Super Soaker Prototype Failure Predictions

Reese Sumrall

2025-02-15

## Project Preparations

Load Libraries for R

Load data from the training and testing files. Combine data to get total picture of “missingness”.

train\_initial = read\_csv("train.csv")

## Rows: 26570 Columns: 26  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (4): product\_code, attribute\_0, attribute\_1, failure  
## dbl (22): id, loading, attribute\_2, attribute\_3, measurement\_0, measurement\_...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

test\_initial = read\_csv("test.csv")

## Rows: 20775 Columns: 25  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (3): product\_code, attribute\_0, attribute\_1  
## dbl (22): id, loading, attribute\_2, attribute\_3, measurement\_0, measurement\_...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

combined\_initial = bind\_rows(train\_initial,test\_initial)  
count(combined\_initial)

## # A tibble: 1 × 1  
## n  
## <int>  
## 1 47345

skim(combined\_initial)

Data summary

|  |  |
| --- | --- |
| Name | combined\_initial |
| Number of rows | 47345 |
| Number of columns | 26 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 4 |
| numeric | 22 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- |
| product\_code | 0 | 1.00 | 1 | 1 | 0 | 9 | 0 |
| attribute\_0 | 0 | 1.00 | 10 | 10 | 0 | 2 | 0 |
| attribute\_1 | 0 | 1.00 | 10 | 10 | 0 | 4 | 0 |
| failure | 20775 | 0.56 | 2 | 3 | 0 | 2 | 0 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id | 0 | 1.00 | 23672.00 | 13667.47 | 0.00 | 11836.00 | 23672.00 | 35508.00 | 47344.00 | ▇▇▇▇▇ |
| loading | 473 | 0.99 | 127.74 | 39.08 | 33.16 | 99.78 | 122.26 | 149.01 | 385.86 | ▃▇▂▁▁ |
| attribute\_2 | 0 | 1.00 | 7.18 | 1.48 | 5.00 | 6.00 | 7.00 | 9.00 | 9.00 | ▃▇▂▂▇ |
| attribute\_3 | 0 | 1.00 | 6.78 | 1.75 | 4.00 | 5.00 | 7.00 | 8.00 | 9.00 | ▇▂▂▆▅ |
| measurement\_0 | 0 | 1.00 | 7.43 | 4.19 | 0.00 | 4.00 | 7.00 | 10.00 | 30.00 | ▇▇▂▁▁ |
| measurement\_1 | 0 | 1.00 | 8.55 | 4.27 | 0.00 | 5.00 | 8.00 | 11.00 | 33.00 | ▅▇▂▁▁ |
| measurement\_2 | 0 | 1.00 | 6.20 | 3.55 | 0.00 | 4.00 | 6.00 | 8.00 | 28.00 | ▇▇▁▁▁ |
| measurement\_3 | 710 | 0.99 | 17.79 | 1.00 | 13.56 | 17.12 | 17.79 | 18.47 | 21.50 | ▁▂▇▅▁ |
| measurement\_4 | 947 | 0.98 | 11.73 | 1.00 | 7.38 | 11.05 | 11.73 | 12.41 | 16.48 | ▁▃▇▂▁ |
| measurement\_5 | 1184 | 0.97 | 17.13 | 1.00 | 12.07 | 16.45 | 17.13 | 17.81 | 21.68 | ▁▂▇▃▁ |
| measurement\_6 | 1420 | 0.97 | 17.51 | 1.00 | 12.71 | 16.84 | 17.51 | 18.19 | 21.54 | ▁▂▇▅▁ |
| measurement\_7 | 1657 | 0.97 | 11.71 | 1.00 | 7.85 | 11.04 | 11.71 | 12.39 | 15.83 | ▁▃▇▂▁ |
| measurement\_8 | 1894 | 0.96 | 19.03 | 1.01 | 14.88 | 18.35 | 19.03 | 19.71 | 23.81 | ▁▃▇▂▁ |
| measurement\_9 | 2131 | 0.95 | 11.43 | 1.00 | 7.54 | 10.75 | 11.42 | 12.10 | 15.41 | ▁▃▇▃▁ |
| measurement\_10 | 2367 | 0.95 | 16.12 | 1.48 | 9.17 | 15.16 | 16.12 | 17.08 | 23.35 | ▁▂▇▂▁ |
| measurement\_11 | 2604 | 0.94 | 19.03 | 1.56 | 12.46 | 17.96 | 19.05 | 20.11 | 25.64 | ▁▃▇▂▁ |
| measurement\_12 | 2841 | 0.94 | 11.80 | 1.44 | 5.17 | 10.86 | 11.82 | 12.75 | 18.96 | ▁▂▇▂▁ |
| measurement\_13 | 3077 | 0.94 | 15.69 | 1.25 | 9.21 | 14.88 | 15.67 | 16.47 | 22.71 | ▁▂▇▁▁ |
| measurement\_14 | 3314 | 0.93 | 16.08 | 1.46 | 8.41 | 15.14 | 16.08 | 17.05 | 23.14 | ▁▁▇▂▁ |
| measurement\_15 | 3551 | 0.92 | 15.05 | 1.55 | 8.42 | 14.01 | 15.01 | 16.06 | 22.10 | ▁▃▇▂▁ |
| measurement\_16 | 3788 | 0.92 | 16.54 | 1.68 | 9.70 | 15.37 | 16.56 | 17.70 | 24.09 | ▁▃▇▂▁ |
| measurement\_17 | 4024 | 0.92 | 701.32 | 126.38 | 1.67 | 618.90 | 701.22 | 784.37 | 1312.79 | ▁▁▇▃▁ |

Remove ID column from training data and make new tibbles

train = train\_initial %>% select(-id)  
test = test\_initial

Structure and summary of training and testing sets.

str(train)

## tibble [26,570 × 25] (S3: tbl\_df/tbl/data.frame)  
## $ product\_code : chr [1:26570] "A" "A" "A" "A" ...  
## $ loading : num [1:26570] 80.1 84.9 82.4 101.1 188.1 ...  
## $ attribute\_0 : chr [1:26570] "material\_7" "material\_7" "material\_7" "material\_7" ...  
## $ attribute\_1 : chr [1:26570] "material\_8" "material\_8" "material\_8" "material\_8" ...  
## $ attribute\_2 : num [1:26570] 9 9 9 9 9 9 9 9 9 9 ...  
## $ attribute\_3 : num [1:26570] 5 5 5 5 5 5 5 5 5 5 ...  
## $ measurement\_0 : num [1:26570] 7 14 12 13 9 11 12 4 9 10 ...  
## $ measurement\_1 : num [1:26570] 8 3 1 2 2 4 2 8 6 4 ...  
## $ measurement\_2 : num [1:26570] 4 3 5 6 8 0 4 8 5 7 ...  
## $ measurement\_3 : num [1:26570] 18 18.2 18.1 17.3 19.3 ...  
## $ measurement\_4 : num [1:26570] 12.5 11.5 11.7 11.2 12.9 ...  
## $ measurement\_5 : num [1:26570] 15.7 17.7 16.7 18.6 17 ...  
## $ measurement\_6 : num [1:26570] 19.3 17.9 18.2 18.3 15.7 ...  
## $ measurement\_7 : num [1:26570] 11.7 12.7 12.7 12.6 11.3 ...  
## $ measurement\_8 : num [1:26570] 20.2 17.9 18.3 19.1 18.1 ...  
## $ measurement\_9 : num [1:26570] 10.7 12.4 12.7 12.5 10.3 ...  
## $ measurement\_10: num [1:26570] 15.9 17.9 15.6 16.3 17.1 ...  
## $ measurement\_11: num [1:26570] 17.6 17.9 NA 18.4 19.9 ...  
## $ measurement\_12: num [1:26570] 15.2 11.8 13.8 10 12.4 ...  
## $ measurement\_13: num [1:26570] 15 14.7 16.7 15.2 16.2 ...  
## $ measurement\_14: num [1:26570] NA 15.4 18.6 15.6 12.8 ...  
## $ measurement\_15: num [1:26570] 13 14.4 14.1 16.2 13.2 ...  
## $ measurement\_16: num [1:26570] 14.7 15.6 17.9 17.2 16.4 ...  
## $ measurement\_17: num [1:26570] 764 682 663 826 580 ...  
## $ failure : chr [1:26570] "No" "No" "No" "No" ...

summary(train)

## product\_code loading attribute\_0 attribute\_1   
## Length:26570 Min. : 33.16 Length:26570 Length:26570   
## Class :character 1st Qu.: 99.99 Class :character Class :character   
## Mode :character Median :122.39 Mode :character Mode :character   
## Mean :127.83   
## 3rd Qu.:149.15   
## Max. :385.86   
## NA's :250   
## attribute\_2 attribute\_3 measurement\_0 measurement\_1   
## Min. :5.000 Min. :5.00 Min. : 0.000 Min. : 0.000   
## 1st Qu.:6.000 1st Qu.:6.00 1st Qu.: 4.000 1st Qu.: 5.000   
## Median :6.000 Median :8.00 Median : 7.000 Median : 8.000   
## Mean :6.754 Mean :7.24 Mean : 7.416 Mean : 8.233   
## 3rd Qu.:8.000 3rd Qu.:8.00 3rd Qu.:10.000 3rd Qu.:11.000   
## Max. :9.000 Max. :9.00 Max. :29.000 Max. :29.000   
##   
## measurement\_2 measurement\_3 measurement\_4 measurement\_5   
## Min. : 0.000 Min. :13.97 Min. : 8.008 Min. :12.07   
## 1st Qu.: 4.000 1st Qu.:17.12 1st Qu.:11.051 1st Qu.:16.44   
## Median : 6.000 Median :17.79 Median :11.733 Median :17.13   
## Mean : 6.257 Mean :17.79 Mean :11.732 Mean :17.13   
## 3rd Qu.: 8.000 3rd Qu.:18.47 3rd Qu.:12.410 3rd Qu.:17.80   
## Max. :24.000 Max. :21.50 Max. :16.484 Max. :21.43   
## NA's :381 NA's :538 NA's :676   
## measurement\_6 measurement\_7 measurement\_8 measurement\_9   
## Min. :12.71 Min. : 7.968 Min. :15.22 Min. : 7.537   
## 1st Qu.:16.84 1st Qu.:11.045 1st Qu.:18.34 1st Qu.:10.757   
## Median :17.52 Median :11.712 Median :19.02 Median :11.430   
## Mean :17.51 Mean :11.717 Mean :19.02 Mean :11.431   
## 3rd Qu.:18.18 3rd Qu.:12.391 3rd Qu.:19.71 3rd Qu.:12.102   
## Max. :21.54 Max. :15.419 Max. :23.81 Max. :15.412   
## NA's :796 NA's :937 NA's :1048 NA's :1227   
## measurement\_10 measurement\_11 measurement\_12 measurement\_13   
## Min. : 9.323 Min. :12.46 Min. : 5.167 Min. :10.89   
## 1st Qu.:15.209 1st Qu.:18.17 1st Qu.:10.703 1st Qu.:14.89   
## Median :16.127 Median :19.21 Median :11.717 Median :15.63   
## Mean :16.118 Mean :19.17 Mean :11.703 Mean :15.65   
## 3rd Qu.:17.025 3rd Qu.:20.21 3rd Qu.:12.709 3rd Qu.:16.37   
## Max. :22.479 Max. :25.64 Max. :17.663 Max. :22.71   
## NA's :1300 NA's :1468 NA's :1601 NA's :1774   
## measurement\_14 measurement\_15 measurement\_16 measurement\_17   
## Min. : 9.14 Min. : 9.104 Min. : 9.701 Min. : 196.8   
## 1st Qu.:15.06 1st Qu.:13.957 1st Qu.:15.268 1st Qu.: 619.0   
## Median :16.04 Median :14.969 Median :16.436 Median : 701.0   
## Mean :16.05 Mean :14.996 Mean :16.461 Mean : 701.3   
## 3rd Qu.:17.08 3rd Qu.:16.018 3rd Qu.:17.628 3rd Qu.: 784.1   
## Max. :22.30 Max. :21.626 Max. :24.094 Max. :1312.8   
## NA's :1874 NA's :2009 NA's :2110 NA's :2284   
## failure   
## Length:26570   
## Class :character   
## Mode :character   
##   
##   
##   
##

str(test)

## spc\_tbl\_ [20,775 × 25] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ id : num [1:20775] 26570 26571 26572 26573 26574 ...  
## $ product\_code : chr [1:20775] "F" "F" "F" "F" ...  
## $ loading : num [1:20775] 120 114 112 113 208 ...  
## $ attribute\_0 : chr [1:20775] "material\_5" "material\_5" "material\_5" "material\_5" ...  
## $ attribute\_1 : chr [1:20775] "material\_6" "material\_6" "material\_6" "material\_6" ...  
## $ attribute\_2 : num [1:20775] 6 6 6 6 6 6 6 6 6 6 ...  
## $ attribute\_3 : num [1:20775] 4 4 4 4 4 4 4 4 4 4 ...  
## $ measurement\_0 : num [1:20775] 6 11 8 8 14 10 6 16 7 14 ...  
## $ measurement\_1 : num [1:20775] 9 8 12 11 16 11 18 7 9 15 ...  
## $ measurement\_2 : num [1:20775] 6 0 4 10 8 7 11 4 7 7 ...  
## $ measurement\_3 : num [1:20775] 19.3 17.9 18.5 16.5 17.8 ...  
## $ measurement\_4 : num [1:20775] 10.2 11.9 10.5 10.9 12.7 ...  
## $ measurement\_5 : num [1:20775] 17.5 17.2 16.6 15.3 17.7 ...  
## $ measurement\_6 : num [1:20775] 18.2 16 18.2 18.6 15.8 ...  
## $ measurement\_7 : num [1:20775] 11.6 11.2 12.1 11.3 13.4 ...  
## $ measurement\_8 : num [1:20775] 18.7 19.4 17.8 18.9 19.1 ...  
## $ measurement\_9 : num [1:20775] 10.8 12 11.7 11.8 12.4 ...  
## $ measurement\_10: num [1:20775] 15.9 14 17 18.2 14.6 ...  
## $ measurement\_11: num [1:20775] 18.1 NA 18.1 16.2 17.8 ...  
## $ measurement\_12: num [1:20775] 13.8 12.5 10.9 10.9 11.9 ...  
## $ measurement\_13: num [1:20775] 13.7 17.5 13.4 15.5 16.1 ...  
## $ measurement\_14: num [1:20775] 16.8 16.7 15.7 15.7 16.2 ...  
## $ measurement\_15: num [1:20775] 13.7 14.8 17.1 12.6 13.3 ...  
## $ measurement\_16: num [1:20775] 17.7 14.1 16 16.1 17.1 ...  
## $ measurement\_17: num [1:20775] 635 537 659 594 801 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. id = col\_double(),  
## .. product\_code = col\_character(),  
## .. loading = col\_double(),  
## .. attribute\_0 = col\_character(),  
## .. attribute\_1 = col\_character(),  
## .. attribute\_2 = col\_double(),  
## .. attribute\_3 = col\_double(),  
## .. measurement\_0 = col\_double(),  
## .. measurement\_1 = col\_double(),  
## .. measurement\_2 = col\_double(),  
## .. measurement\_3 = col\_double(),  
## .. measurement\_4 = col\_double(),  
## .. measurement\_5 = col\_double(),  
## .. measurement\_6 = col\_double(),  
## .. measurement\_7 = col\_double(),  
## .. measurement\_8 = col\_double(),  
## .. measurement\_9 = col\_double(),  
## .. measurement\_10 = col\_double(),  
## .. measurement\_11 = col\_double(),  
## .. measurement\_12 = col\_double(),  
## .. measurement\_13 = col\_double(),  
## .. measurement\_14 = col\_double(),  
## .. measurement\_15 = col\_double(),  
## .. measurement\_16 = col\_double(),  
## .. measurement\_17 = col\_double()  
## .. )  
## - attr(\*, "problems")=<externalptr>

summary(test)

## id product\_code loading attribute\_0   
## Min. :26570 Length:20775 Min. : 37.70 Length:20775   
## 1st Qu.:31764 Class :character 1st Qu.: 99.47 Class :character   
## Median :36957 Mode :character Median :122.11 Mode :character   
## Mean :36957 Mean :127.63   
## 3rd Qu.:42151 3rd Qu.:148.84   
## Max. :47344 Max. :385.57   
## NA's :223   
## attribute\_1 attribute\_2 attribute\_3 measurement\_0   
## Length:20775 Min. :6.000 Min. :4.000 Min. : 0.000   
## Class :character 1st Qu.:6.000 1st Qu.:4.000 1st Qu.: 4.000   
## Mode :character Median :7.000 Median :5.000 Median : 7.000   
## Mean :7.734 Mean :6.197 Mean : 7.454   
## 3rd Qu.:9.000 3rd Qu.:7.000 3rd Qu.:10.000   
## Max. :9.000 Max. :9.000 Max. :30.000   
##   
## measurement\_1 measurement\_2 measurement\_3 measurement\_4   
## Min. : 0.000 Min. : 0.000 Min. :13.56 Min. : 7.384   
## 1st Qu.: 6.000 1st Qu.: 3.000 1st Qu.:17.12 1st Qu.:11.048   
## Median : 9.000 Median : 6.000 Median :17.79 Median :11.729   
## Mean : 8.962 Mean : 6.127 Mean :17.79 Mean :11.727   
## 3rd Qu.:12.000 3rd Qu.: 8.000 3rd Qu.:18.48 3rd Qu.:12.411   
## Max. :33.000 Max. :28.000 Max. :21.39 Max. :15.623   
## NA's :329 NA's :409   
## measurement\_5 measurement\_6 measurement\_7 measurement\_8   
## Min. :12.21 Min. :13.54 Min. : 7.853 Min. :14.88   
## 1st Qu.:16.46 1st Qu.:16.85 1st Qu.:11.035 1st Qu.:18.35   
## Median :17.13 Median :17.51 Median :11.704 Median :19.04   
## Mean :17.14 Mean :17.52 Mean :11.711 Mean :19.03   
## 3rd Qu.:17.82 3rd Qu.:18.20 3rd Qu.:12.385 3rd Qu.:19.71   
## Max. :21.68 Max. :21.18 Max. :15.828 Max. :23.09   
## NA's :508 NA's :624 NA's :720 NA's :846   
## measurement\_9 measurement\_10 measurement\_11 measurement\_12   
## Min. : 7.578 Min. : 9.167 Min. :13.13 Min. : 6.116   
## 1st Qu.:10.744 1st Qu.:15.095 1st Qu.:17.71 1st Qu.:11.069   
## Median :11.414 Median :16.110 Median :18.81 Median :11.941   
## Mean :11.418 Mean :16.124 Mean :18.85 Mean :11.914   
## 3rd Qu.:12.093 3rd Qu.:17.156 3rd Qu.:19.97 3rd Qu.:12.791   
## Max. :15.091 Max. :23.354 Max. :24.95 Max. :18.962   
## NA's :904 NA's :1067 NA's :1136 NA's :1240   
## measurement\_13 measurement\_14 measurement\_15 measurement\_16   
## Min. : 9.209 Min. : 8.415 Min. : 8.417 Min. :10.16   
## 1st Qu.:14.871 1st Qu.:15.238 1st Qu.:14.082 1st Qu.:15.51   
## Median :15.734 Median :16.119 Median :15.062 Median :16.71   
## Mean :15.736 Mean :16.124 Mean :15.116 Mean :16.64   
## 3rd Qu.:16.605 3rd Qu.:17.002 3rd Qu.:16.107 3rd Qu.:17.78   
## Max. :21.677 Max. :23.140 Max. :22.097 Max. :22.27   
## NA's :1303 NA's :1440 NA's :1542 NA's :1678   
## measurement\_17   
## Min. : 1.671   
## 1st Qu.: 618.723   
## Median : 701.379   
## Mean : 701.390   
## 3rd Qu.: 784.872   
## Max. :1242.786   
## NA's :1740

Looking at the summary and structure, NA’s are present in many columns. This indicates missing values. Measurements 3 - 17 and the loading variable contain NA’s.

Also, it appears as though some of the columns have different values in test vs train sets. Check those variable values in the training vs testing sets.

# Product Code   
unique(train$product\_code)

## [1] "A" "B" "C" "D" "E"

unique(test$product\_code)

## [1] "F" "G" "H" "I"

# attribute\_0  
unique(train$attribute\_0)

## [1] "material\_7" "material\_5"

unique(test$attribute\_0)

## [1] "material\_5" "material\_7"

# attribute\_1  
unique(train$attribute\_1)

## [1] "material\_8" "material\_5" "material\_6"

unique(test$attribute\_1)

## [1] "material\_6" "material\_7" "material\_5"

# attribute\_2  
unique(train$attribute\_2)

## [1] 9 8 5 6

unique(test$attribute\_2)

## [1] 6 9 7

# attribute\_3  
unique(train$attribute\_3)

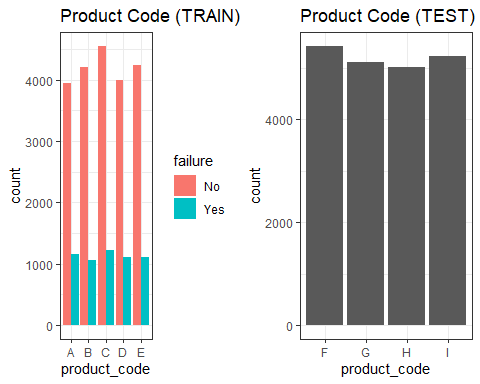
## [1] 5 8 6 9

unique(test$attribute\_3)

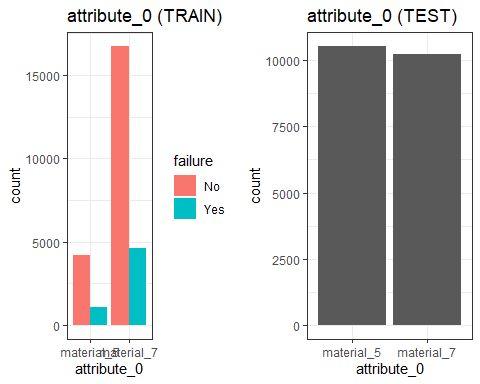
## [1] 4 7 9 5

Unique values for product\_code, attribute\_3, attribute\_2, and attribute\_1 variables exist between test and train sets. This may skew the results. We could remove those variables, but for now they can remain and we can try to plan around them. Doing some feature engineering can allow us to make changes to these settings. It was mentioned in the project notes that the attributes are specific to the product code so with some feature engineering we might be able to train the model to handle different attribute and product code values in the testing set.

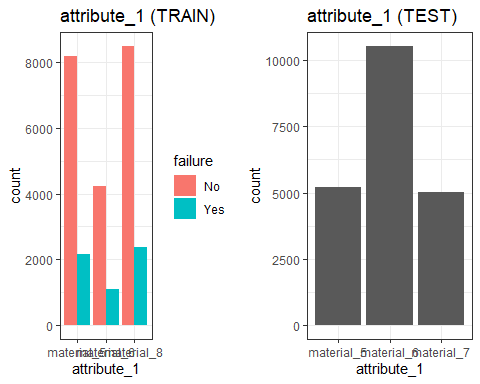
# train = train %>% select(-product\_code, -attribute\_3, -attribute\_2, -attribute\_1) #in case we want to just remove these later  
# test = test %>% select(-product\_code, -attribute\_3, -attribute\_2, -attribute\_1) #in case we want to just remove these later  
  
a1 = ggplot(train, aes(x = product\_code, fill = failure)) +  
 geom\_bar(position="dodge") +  
 ggtitle("Product Code (TRAIN)") +  
 theme\_bw()   
  
a2 = ggplot(test, aes(x = product\_code)) +  
 geom\_bar() +  
 ggtitle("Product Code (TEST)") +  
 theme\_bw()   
  
grid.arrange(a1,a2, ncol = 2)



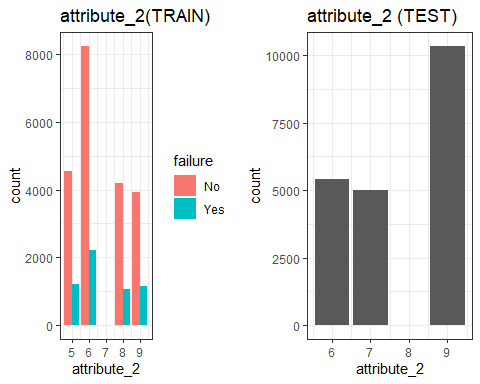
a3 = ggplot(train, aes(x = attribute\_0, fill = failure)) +  
 geom\_bar(position="dodge") +  
 ggtitle("attribute\_0 (TRAIN)") +  
 theme\_bw()   
  
a4 = ggplot(test, aes(x = attribute\_0)) +  
 geom\_bar() +  
 ggtitle("attribute\_0 (TEST)") +  
 theme\_bw()   
  
grid.arrange(a3,a4, ncol = 2)



a5 = ggplot(train, aes(x = attribute\_1, fill = failure)) +  
 geom\_bar(position="dodge") +  
 ggtitle("attribute\_1 (TRAIN)") +  
 theme\_bw()   
  
a6 = ggplot(test, aes(x = attribute\_1)) +  
 geom\_bar() +  
 ggtitle("attribute\_1 (TEST)") +  
 theme\_bw()   
  
grid.arrange(a5,a6, ncol = 2)

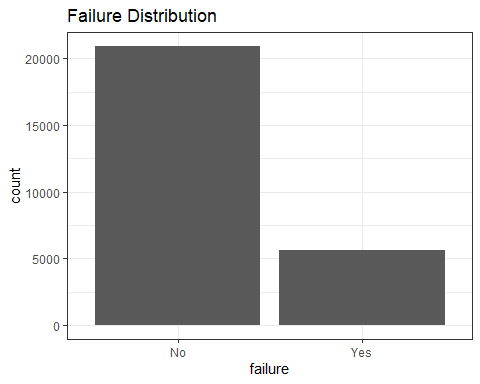


a7 = ggplot(train, aes(x = attribute\_2, fill = failure)) +  
 geom\_bar(position="dodge") +  
 ggtitle("attribute\_2(TRAIN)") +  
 theme\_bw()   
  
a8 = ggplot(test, aes(x = attribute\_2)) +  
 geom\_bar() +  
 ggtitle("attribute\_2 (TEST)") +  
 theme\_bw()   
  
grid.arrange(a7,a8, ncol = 2)



Check for imbalance.

ggplot(train, aes(x = failure)) +  
 geom\_bar(position="dodge") +  
 ggtitle("Failure Distribution") +  
 theme\_bw()



There is some imbalance in the dataset for the response variable, failure.

Check for missing data. Graph and display data for missing data by number of rows missing data per variable.

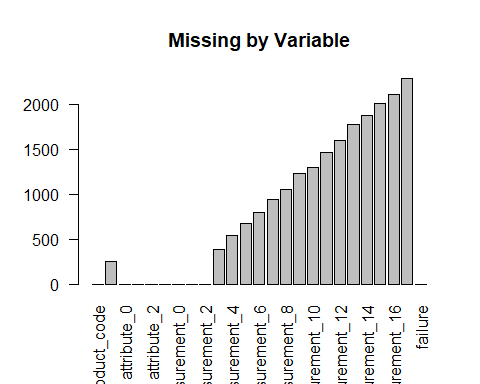
missing\_train = data.frame(sapply(train, function(x) sum(is.na(x))))  
missing\_test = data.frame(sapply(test, function(x) sum(is.na(x))))  
  
# missing values in train and test sets   
missing\_train

## sapply.train..function.x..sum.is.na.x...  
## product\_code 0  
## loading 250  
## attribute\_0 0  
## attribute\_1 0  
## attribute\_2 0  
## attribute\_3 0  
## measurement\_0 0  
## measurement\_1 0  
## measurement\_2 0  
## measurement\_3 381  
## measurement\_4 538  
## measurement\_5 676  
## measurement\_6 796  
## measurement\_7 937  
## measurement\_8 1048  
## measurement\_9 1227  
## measurement\_10 1300  
## measurement\_11 1468  
## measurement\_12 1601  
## measurement\_13 1774  
## measurement\_14 1874  
## measurement\_15 2009  
## measurement\_16 2110  
## measurement\_17 2284  
## failure 0

missing\_test

## sapply.test..function.x..sum.is.na.x...  
## id 0  
## product\_code 0  
## loading 223  
## attribute\_0 0  
## attribute\_1 0  
## attribute\_2 0  
## attribute\_3 0  
## measurement\_0 0  
## measurement\_1 0  
## measurement\_2 0  
## measurement\_3 329  
## measurement\_4 409  
## measurement\_5 508  
## measurement\_6 624  
## measurement\_7 720  
## measurement\_8 846  
## measurement\_9 904  
## measurement\_10 1067  
## measurement\_11 1136  
## measurement\_12 1240  
## measurement\_13 1303  
## measurement\_14 1440  
## measurement\_15 1542  
## measurement\_16 1678  
## measurement\_17 1740

# Graph  
missing\_graph <- colSums(is.na(train))  
barplot(missing\_graph, names.arg = names(missing\_graph), las = 2, main = 'Missing by Variable')



Get the percent of rows that have missing data

combined = combined\_initial %>% select(-failure)  
  
num\_missing\_rows = sum(rowSums(is.na(combined)) > 0)  
variable1 = num\_missing\_rows / count(combined)  
variable1

## n  
## 1 0.5401204

Roughly *54%* of the data rows contain at least one missing variable value.

Check to see if missing data for single variables affects failure rate. Create new variables that will show if a row is missing a value for each of the continuous variables.

train = train %>% mutate(loading\_missing = ifelse(is.na(loading), "Yes", "No")) %>%  
 mutate(m3\_missing = ifelse(is.na(measurement\_3), "Yes", "No")) %>%  
 mutate(m4\_missing = ifelse(is.na(measurement\_4), "Yes", "No")) %>%  
 mutate(m5\_missing = ifelse(is.na(measurement\_5), "Yes", "No"))   
  
test = test %>% mutate(loading\_missing = ifelse(is.na(loading), "Yes", "No")) %>%  
 mutate(m3\_missing = ifelse(is.na(measurement\_3), "Yes", "No")) %>%  
 mutate(m4\_missing = ifelse(is.na(measurement\_4), "Yes", "No")) %>%  
 mutate(m5\_missing = ifelse(is.na(measurement\_5), "Yes", "No"))

“Missingness” for the other variables was found to be not significant so they were removed and the four above remain to proceed further.

Convert all character variables to factor.

train = train %>% mutate\_if(is.character,as\_factor)   
test = test %>% mutate\_if(is.character,as\_factor)  
  
train = train %>% mutate(attribute\_0 = as\_factor(attribute\_0)) %>%  
 mutate(attribute\_1 = as\_factor(attribute\_1))   
  
test = test %>% mutate(attribute\_0 = as\_factor(attribute\_0)) %>%  
 mutate(attribute\_1 = as\_factor(attribute\_1))

Split train and test data by product code for grouped imputation.

unique\_products <- unique(train$product\_code)  
print(unique\_products)

## [1] A B C D E  
## Levels: A B C D E

df\_pcA <- train %>% filter(product\_code == 'A')  
df\_pcB <- train %>% filter(product\_code == 'B')  
df\_pcC <- train %>% filter(product\_code == 'C')  
df\_pcD <- train %>% filter(product\_code == 'D')  
df\_pcE <- train %>% filter(product\_code == 'E')  
  
unique\_products2 <- unique(test$product\_code)  
print(unique\_products2)

## [1] F G H I  
## Levels: F G H I

df\_pcF\_t <- filter(test, product\_code == 'F')  
df\_pcG\_t <- filter(test, product\_code == 'G')  
df\_pcH\_t <- filter(test, product\_code == 'H')  
df\_pcI\_t <- filter(test, product\_code == 'I')

imputation for missing data on both training and testing sets. Use predictive mean matching and perform the imputation by group of product code. This was done so that the average values of the variables to be imputed will be taken from other tests for that same product code.

set.seed(1234)  
  
# Train   
train\_impute\_A <- mice(df\_pcA, m=5, method='pmm', printFlag=FALSE)  
train\_impute\_B <- mice(df\_pcB, m=5, method='pmm', printFlag=FALSE)  
train\_impute\_C <- mice(df\_pcC, m=5, method='pmm', printFlag=FALSE)  
train\_impute\_D <- mice(df\_pcD, m=5, method='pmm', printFlag=FALSE)  
train\_impute\_E <- mice(df\_pcE, m=5, method='pmm', printFlag=FALSE)  
  
# Test  
test\_impute\_F <- mice(df\_pcF\_t, m=5, method='pmm', printFlag=FALSE)  
test\_impute\_G <- mice(df\_pcG\_t, m=5, method='pmm', printFlag=FALSE)  
test\_impute\_H <- mice(df\_pcH\_t, m=5, method='pmm', printFlag=FALSE)  
test\_impute\_I <- mice(df\_pcI\_t, m=5, method='pmm', printFlag=FALSE)

Recombine split sets

train = bind\_rows(complete(train\_impute\_A),complete(train\_impute\_B),complete(train\_impute\_C) ,complete(train\_impute\_D) ,complete(train\_impute\_E) )  
test = bind\_rows(complete(test\_impute\_F),complete(test\_impute\_G),complete(test\_impute\_H) ,complete(test\_impute\_I) )

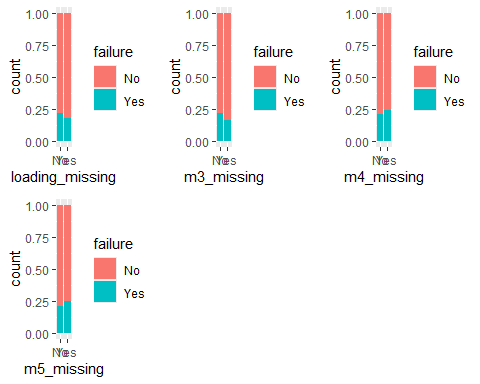
### Continue with EDA

Now we’ll split the training data set into a train and test set. A 70/30 split between train/test sets, respectively.

set.seed(1234)   
kid\_split = initial\_split(train, prop = 0.7, strata = failure)   
model\_train = training(kid\_split)  
model\_test = testing(kid\_split)

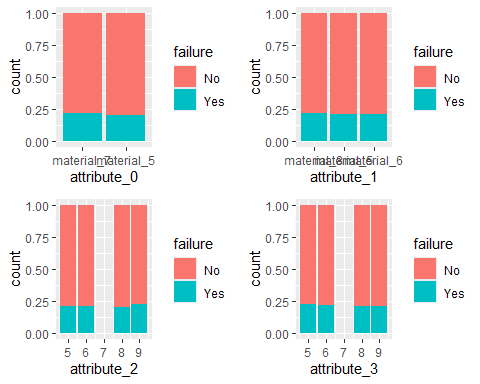
visualize the missing value variables for correlation with failure rate

p1 = ggplot(model\_train, aes(x = loading\_missing, fill = failure)) + geom\_bar(position = "fill")  
p2 = ggplot(model\_train, aes(x = m3\_missing, fill = failure)) + geom\_bar(position = "fill")  
p3 = ggplot(model\_train, aes(x = m4\_missing, fill = failure)) + geom\_bar(position = "fill")  
p4 = ggplot(model\_train, aes(x = m5\_missing, fill = failure)) + geom\_bar(position = "fill")  
grid.arrange(p1,p2,p3,p4,ncol=3)



Visualization for factors/characters

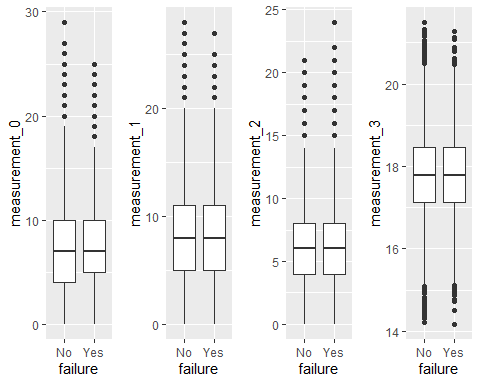
c1 = ggplot(model\_train, aes(x = attribute\_0, fill = failure)) + geom\_bar(position = "fill")  
c2 = ggplot(model\_train, aes(x = attribute\_1, fill = failure)) + geom\_bar(position = "fill")  
c3 = ggplot(model\_train, aes(x = attribute\_2, fill = failure)) + geom\_bar(position = "fill")  
c4 = ggplot(model\_train, aes(x = attribute\_3, fill = failure)) + geom\_bar(position = "fill")  
grid.arrange(c1,c2,c3,c4)



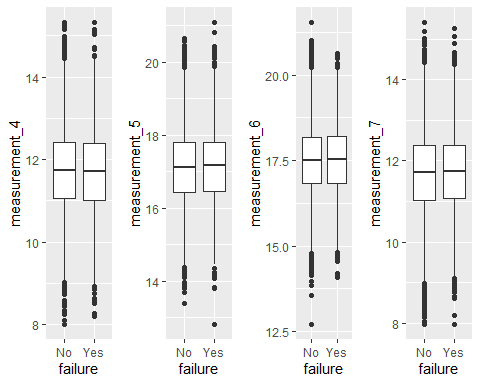
Visualization for numeric variables vs failure rate

n1 = ggplot(model\_train, aes(x = failure, y = measurement\_0)) + geom\_boxplot()  
n2 = ggplot(model\_train, aes(x = failure, y = measurement\_1)) + geom\_boxplot()  
n3 = ggplot(model\_train, aes(x = failure, y = measurement\_2)) + geom\_boxplot()  
n4 = ggplot(model\_train, aes(x = failure, y = measurement\_3)) + geom\_boxplot()  
n5 = ggplot(model\_train, aes(x = failure, y = measurement\_4)) + geom\_boxplot()  
n6 = ggplot(model\_train, aes(x = failure, y = measurement\_5)) + geom\_boxplot()  
n7 = ggplot(model\_train, aes(x = failure, y = measurement\_6)) + geom\_boxplot()  
n8 = ggplot(model\_train, aes(x = failure, y = measurement\_7)) + geom\_boxplot()  
n9 = ggplot(model\_train, aes(x = failure, y = measurement\_8)) + geom\_boxplot()  
n10 = ggplot(model\_train, aes(x = failure, y = measurement\_9)) + geom\_boxplot()  
n11 = ggplot(model\_train, aes(x = failure, y = measurement\_10)) + geom\_boxplot()  
n12 = ggplot(model\_train, aes(x = failure, y = measurement\_11)) + geom\_boxplot()  
n13 = ggplot(model\_train, aes(x = failure, y = measurement\_12)) + geom\_boxplot()  
n14 = ggplot(model\_train, aes(x = failure, y = measurement\_13)) + geom\_boxplot()  
n15 = ggplot(model\_train, aes(x = failure, y = measurement\_14)) + geom\_boxplot()  
n16 = ggplot(model\_train, aes(x = failure, y = measurement\_15)) + geom\_boxplot()  
n17 = ggplot(model\_train, aes(x = failure, y = measurement\_16)) + geom\_boxplot()  
n18 = ggplot(model\_train, aes(x = failure, y = measurement\_17)) + geom\_boxplot()  
n19 = ggplot(model\_train, aes(x = failure, y = loading)) + geom\_boxplot()

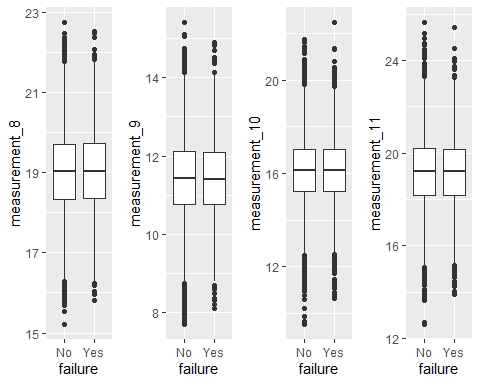
grid.arrange(n1,n2,n3,n4, ncol = 4)



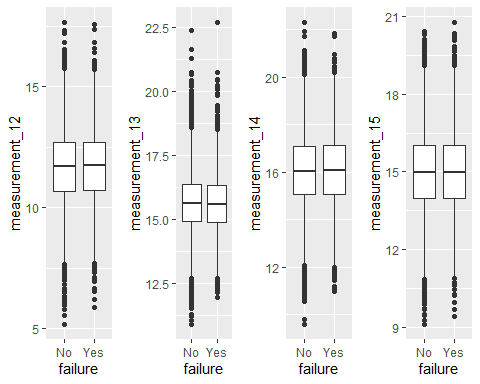
grid.arrange(n5,n6,n7,n8, ncol = 4)



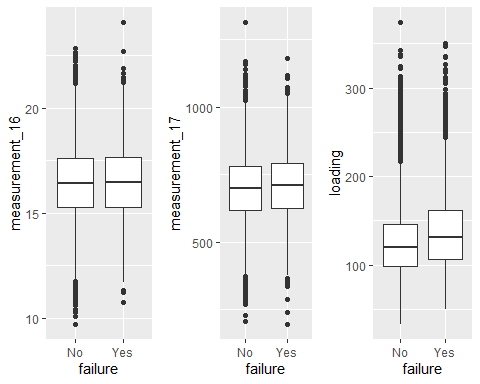
grid.arrange(n9,n10,n11,n12, ncol = 4)



grid.arrange(n13,n14,n15,n16, ncol = 4)



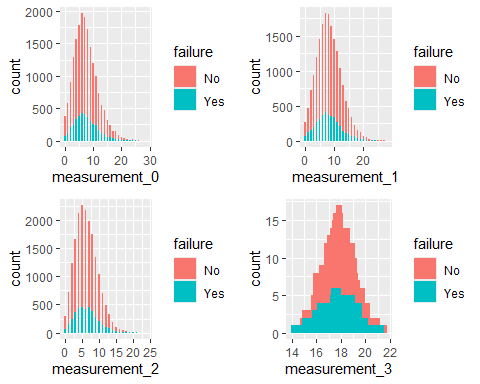
grid.arrange(n17,n18,n19, ncol = 3)



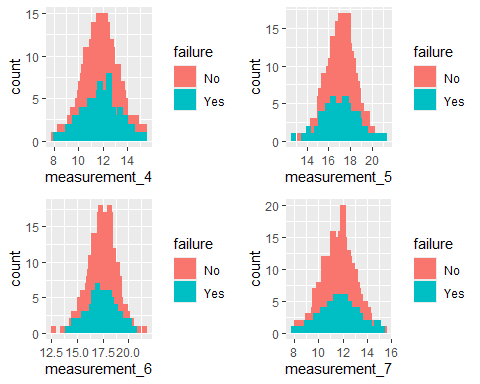
Look at numeric variable distributions grouped by failure/success.

e1 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_0,fill=failure) )  
e2 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_1,fill=failure) )  
e3 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_2,fill=failure) )  
e4 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_3,fill=failure) )  
e5 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_4,fill=failure) )  
e6 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_5,fill=failure) )  
e7 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_6,fill=failure) )  
e8 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_7,fill=failure) )  
e9 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_8,fill=failure) )  
e10 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_9,fill=failure) )  
e11 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_10,fill=failure) )  
e12 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_11,fill=failure) )  
e13 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_12,fill=failure) )  
e14 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_13,fill=failure) )  
e15 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_14,fill=failure) )  
e16 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_15,fill=failure) )  
e17 = ggplot() + geom\_bar(data = model\_train, width = 0.5, aes(x = measurement\_16,fill=failure) )  
e18 = ggplot() + geom\_bar(data = model\_train, width = 20, aes(x = measurement\_17,fill=failure) )  
e19 = ggplot() + geom\_bar(data = model\_train, width = 10, aes(x = loading,fill=failure) )

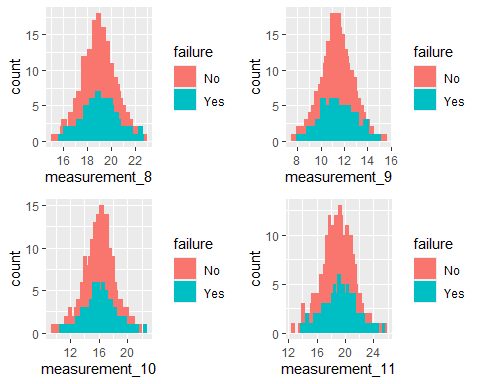
grid.arrange(e1,e2,e3,e4, ncol = 2)



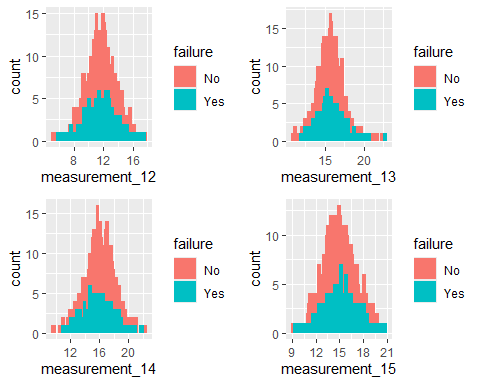
grid.arrange(e5,e6,e7,e8, ncol = 2)



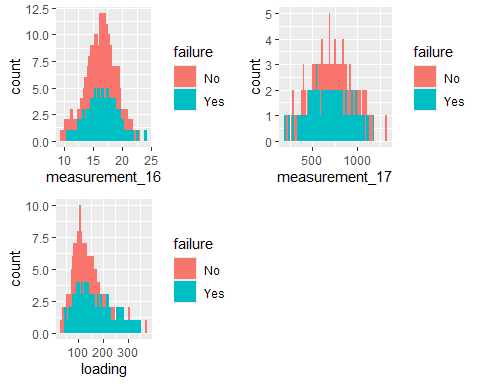
grid.arrange(e9,e10,e11,e12, ncol = 2)



grid.arrange(e13,e14,e15,e16, ncol = 2)



grid.arrange(e17,e18,e19, ncol = 2)



Notes:  
-Roughly *54%* of the data (train & test combined) is missing at least one variable value  
-Missing data increases as the measurement number increases and only the continuous variables contain missing values  
-Mostly, missing single values do not seem to have much effect on failure rate. The exceptions are slight variations in failure rate for the absence of values for measurement\_3, measurement\_4, measurement\_5 -The numeric measurements and loading variable follow mostly normal distributions, with some skewedness mainly for measurements 0,1,2 & loading

### Perform logistic regression to get baseline AOC performance and feature importance

model\_train = model\_train %>% select(-product\_code)

Build a model with all of the variables.

kid\_logm\_model =   
 logistic\_reg(mode = "classification") %>% #note the use of logistic\_reg and mode = "classification"  
 set\_engine("glm") #standard logistic regression engine is glm  
  
kid\_logm\_model\_recipe = recipe(failure ~., model\_train)  
  
logreg\_wf = workflow() %>%  
 add\_recipe(kid\_logm\_model\_recipe) %>%   
 add\_model(kid\_logm\_model)  
  
kid\_logm\_fit = fit(logreg\_wf, model\_train)

Develop predictions

predictions = predict(kid\_logm\_fit, model\_train, type="prob") #develop predicted probabilities  
head(predictions)

## # A tibble: 6 × 2  
## .pred\_No .pred\_Yes  
## <dbl> <dbl>  
## 1 0.833 0.167  
## 2 0.847 0.153  
## 3 0.797 0.203  
## 4 0.821 0.179  
## 5 0.819 0.181  
## 6 0.853 0.147

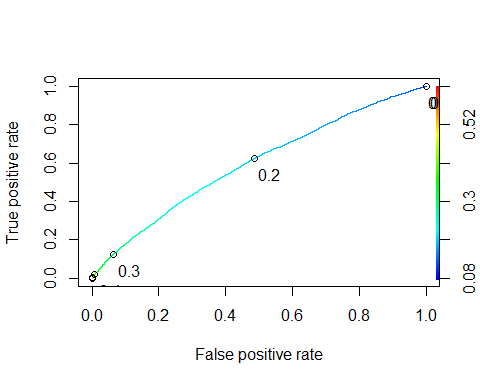
Let’s extract just the “Yes” prediction.

predictions = predict(kid\_logm\_fit, model\_train, type="prob")[2]  
head(predictions)

## # A tibble: 6 × 1  
## .pred\_Yes  
## <dbl>  
## 1 0.167  
## 2 0.153  
## 3 0.203  
## 4 0.179  
## 5 0.181  
## 6 0.147

Threshold selection

ROCRpred = prediction(predictions, model\_train$failure)   
ROCRperf = performance(ROCRpred, "tpr", "fpr")  
plot(ROCRperf, colorize=TRUE, print.cutoffs.at=seq(0,1,by=0.1), text.adj=c(-0.2,1.7))



as.numeric(performance(ROCRpred, "auc")@y.values)

## [1] 0.595597

#Determine threshold to balance sensitivity and specificity  
opt.cut = function(perf, pred){  
 cut.ind = mapply(FUN=function(x, y, p){  
 d = (x - 0)^2 + (y-1)^2  
 ind = which(d == min(d))  
 c(sensitivity = y[[ind]], specificity = 1-x[[ind]],   
 cutoff = p[[ind]])  
 }, perf@x.values, perf@y.values, pred@cutoffs)  
}  
print(opt.cut(ROCRperf, ROCRpred))

## [,1]  
## sensitivity 0.5612038  
## specificity 0.5771647  
## cutoff 0.2075912

Test thresholds to evaluate accuracy

#confusion matrix  
#The "No" and "Yes" represent the actual values  
#The "FALSE" and "TRUE" represent our predicted values  
t1 = table(model\_train$failure,predictions > 0.2075912)  
t1

##   
## FALSE TRUE  
## No 8452 6192  
## Yes 1735 2219

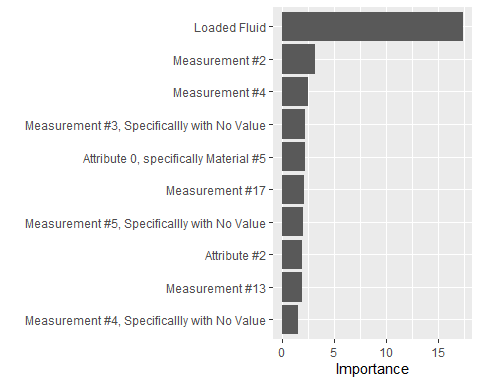
Calculate accuracy

(t1[1,1]+t1[2,2])/nrow(model\_train)

## [1] 0.5737714

Get variable importance from the logistic regression model.

labels = c(loading = 'Loaded Fluid', measurement\_2 = "Measurement #2", measurement\_4 = "Measurement #4", attribute\_0material\_5 = "Attribute 0, specifically Material #5",m3\_missingYes = "Measurement #3, Specificallly with No Value",m5\_missingYes = "Measurement #5, Specificallly with No Value", attribute\_2 = "Attribute #2",measurement\_13 = "Measurement #13",m4\_missingYes = "Measurement #4, Specificallly with No Value",measurement\_9 = "Measurement #9", measurement\_17 = "Measurement #17" )  
  
vip = vip(kid\_logm\_fit$fit$fit) + scale\_x\_discrete(labels = labels)   
  
vip



The *loading* variable is the most important variable. The other variables have very low importance. In the next phase, this model will be sent through the other modeling techniques to see if a better accuracy can be obtained. More feature engineering may be required in order to get more signal from the data.