Intel Case Study

Kushal Kapadia

I'm so excited for this Case study! Here's a little summary of my thought process about the problem:

So, here is a manufacturing quality commonality problem where commonality is a characteristic of bad lots. Analyzing commonalities would help us find where the defect/problem/trouble lies. In this case, looking at the figure, it is crystal clear that there are multiple material suppliers which are termed as lots (collection of units) and they are made to pass through the assembly and testing stations where these stations would either add the unit or test it. But what's intriguing here is that the testing station has found some irregularities regarding the failure of the units. Hence to find an answer to why this occurs, here's my approach:

Loading the data:

```
unit_level = read.csv("F:\\Unit_Level.csv")
lot_level = read.csv("F:\\Lot_Level.csv")
library(ggplot2) #data visualization library

#Combining both the dataframes for data visualization
full = merge(unit_level,lot_level)
```

Let's try to find the total number of failed units.

```
sum(full$RESPONSE_FLAG == 1)

## [1] 1889

sum(full$RESPONSE_FLAG == 0)

## [1] 275074
```

This explains that out of the total of 276963 units, 275074 units had a tag of pass and 1889 units failed.

Also, let's find the total number of lots and then total number of unique lots that have failed units in them.

```
#Total number of unique lots
length(unique(full$LOT_ID))

## [1] 267

#Total number of unique lots with failed units
length(unique(full$LOT_ID[full$RESPONSE_FLAG==1]))
```

```
## [1] 73
```

```
#Total number of unique processing dates with failed units
length(unique(full$UNIT_PROCESS_DATE[full$RESPONSE_FLAG==1]))
```

```
## [1] 73
```

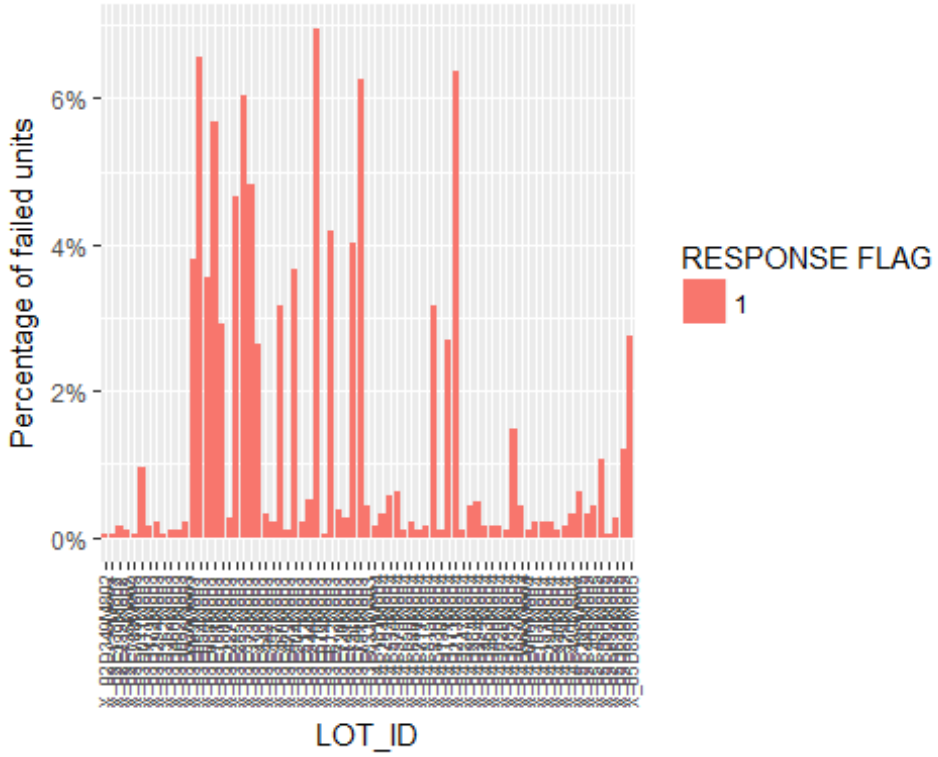
So, out of 267 unique lot_id, we have narrowed it down to 73 that have failed units in them.

Now, it seems that there are too many letters in the UNIT_PROCESS_DATE. Let's also take out the substring from the variable. I'm doing this for the ease in data visualization

```
#Extracting only the date from the UNIT_PROCESS_DATE variable and adding it
as a column in the the 'full' dataframe
full_date = substring(full$UNIT_PROCESS_DATE, 1, 10)
full$date = full_date
```

Now that we're ready to visualize, I want to first see the percentage of failed units without the processing dates just to get the idea of which lost has most defective units.

```
library(scales)
ggplot(full[full$RESPONSE_FLAG==1,], aes(x = LOT_ID, fill =
factor(RESPONSE_FLAG))) +
  geom_bar(aes(y = (..count..)/sum(..count..))) +
  scale_y_continuous(labels = percent) +
    theme(axis.text.x = element_text(angle = 90, hjust = 0.3, vjust = 0.4,
size = 7)) +
  xlab("LOT_ID") +
  ylab("Percentage of failed units") +
  labs(fill="RESPONSE FLAG")
```



From this plot, the highest unit failure is roughly around 7% and it occurs in the lot_id 'X_03E576M803'.

```
#To be precise
sum(full$LOT_ID == 'X_03E576M803' & full$RESPONSE_FLAG == 1)
```

```
## [1] 131
```

```
sum(full$LOT_ID == 'X_03E576M803')
```

```
## [1] 1477
```

```
(131/1477)*100
```

```
## [1] 8.86933
```

```
paste('Precisely, the lot_id X_03E576M803 has', (131/1889)*100, '% failed
units out of 1889')
```

```
## [1] "Precisely, the lot_id X_03E576M803 has 6.9348861831657 % failed units
out of 1889"
```

Now, I'm interested in adding dates to the plot and knowing on what exact dates do this failure occurs. That would help us get the whole time frame of failing of units and also the date where the highest failure of 6.9% occurs in lot_id 'X_03E576M803'.

```
library(tidyverse)
```

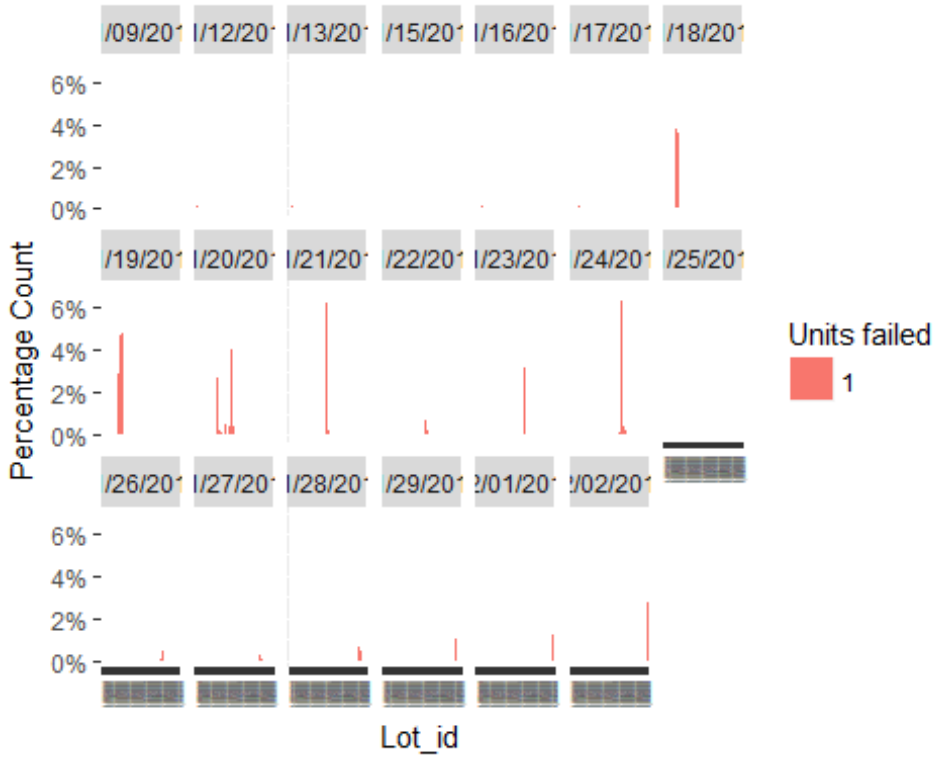
```
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
```

```
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr

## Conflicts with tidy packages ---------------------------------------------
-

## col_factor(): readr, scales
## discard():    purrr, scales
## filter():     dplyr, stats
## lag():        dplyr, stats
```

```
  ggplot(full[full$RESPONSE_FLAG == 1,], aes(x = LOT_ID, date, fill =
factor(RESPONSE_FLAG))) +
  geom_bar(aes(y = (..count..)/sum(..count..)), position = 'dodge') +
  scale_y_continuous(labels = percent) +
  facet_wrap(~date, ncol = 7) +
  theme(axis.text.x = element_text(angle = 90, hjust = 0.3, vjust = 0.4, size
= 1)) +
  xlab("Lot_id") +
  ylab("Percentage Count") +
  labs(fill="Units failed")
```



From this plot, as there are 73 lot_id's, they are not perfectly visible but if we zoom, we could see them. However, from the plot, we can see that the highest failure percentage occurs on **01/27/2018** as many of the lot_id's are showing good failure percentage with the lot_id 'X_03E576M803' having the highest percent of failed units included. Also, if we

include every failed unit, then the whole failure process time interval is from **01/09/2018 - 02/02/2018**. That answers the first question.

Question 2

Feature Selection Problem:

Here, to sum the question up, we're interested in finding the association of 13 parameters in the unit_level with the response flag. Here, one thing to note is that our response variable (dependent) is categorical and rest all the 13 parameters are continuous. Thus, if we wanted to check the association in between 13 parameters, then we could have just chose the correlation test and calculated the pearson coefficient. But here, as we have to find the association of continuous with categorical variable, we're gonna build a ***logistic regression** model on each of the 13 parameters and check the p-value to conclude the significance. Lesser the p-value, more will be the association.

Glancing through the full dataframe, we can see that there are couple missing values in all the 13 parameters. Let's see the number of missing values we've got in each of them seperately to take care of every short detail.

```
sum(is.na(full$PARAMETER1))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER2))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER3))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER4))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER5))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER6))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER7))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER8))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER9))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER10))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER11))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER12))
```

```
## [1] 6572
```

```
sum(is.na(full$PARAMETER13))
```

```
## [1] 6572
```

Thus, we can see that we have 6572 missing values in each of the 13 parameters. We could replace it with either mean, median or mode. I'm gonna replace all the missing values with mean here and then move on to make the model.

```
full$PARAMETER1[is.na(full$PARAMETER1)] = mean(full$PARAMETER1, na.rm = T)
full$PARAMETER2[is.na(full$PARAMETER2)] = mean(full$PARAMETER2, na.rm = T)
full$PARAMETER3[is.na(full$PARAMETER3)] = mean(full$PARAMETER3, na.rm = T)
full$PARAMETER4[is.na(full$PARAMETER4)] = mean(full$PARAMETER4, na.rm = T)
full$PARAMETER5[is.na(full$PARAMETER5)] = mean(full$PARAMETER5, na.rm = T)
full$PARAMETER6[is.na(full$PARAMETER6)] = mean(full$PARAMETER6, na.rm = T)
full$PARAMETER7[is.na(full$PARAMETER7)] = mean(full$PARAMETER7, na.rm = T)
full$PARAMETER8[is.na(full$PARAMETER8)] = mean(full$PARAMETER8, na.rm = T)
full$PARAMETER9[is.na(full$PARAMETER9)] = mean(full$PARAMETER9, na.rm = T)
full$PARAMETER10[is.na(full$PARAMETER10)] = mean(full$PARAMETER10, na.rm = T)
full$PARAMETER11[is.na(full$PARAMETER11)] = mean(full$PARAMETER11, na.rm = T)
full$PARAMETER12[is.na(full$PARAMETER12)] = mean(full$PARAMETER12, na.rm = T)
full$PARAMETER13[is.na(full$PARAMETER13)] = mean(full$PARAMETER13, na.rm = T)
```

Now that all the missing values have been replaced, we're in a good shape to fit a logistic regression model now. As the data is kind of huge, I'd fit the model individually for all the 13 parameters and check the p-value simultaneously.

Parameter-1:

```
model1 = glm(data = full, RESPONSE_FLAG~PARAMETER1, family = binomial(link =
'logit'))
summary(model1)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER1, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -0.1296  -0.1185  -0.1168  -0.1153   3.1842
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.12693    0.09248 -55.440   <2e-16 ***
## PARAMETER1   0.05088    0.03107   1.637    0.102
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22606  on 276961  degrees of freedom
## AIC: 22610
##
## Number of Fisher Scoring iterations: 7
```

This parameter is clearly not significant as the p-value is high. I came to this conclusion by comparing the p-value with the significance level of 0.05. If the p-value is less than the significance level, the parameter is significant.

Parameter 2:

```
model2 = glm(data = full, RESPONSE_FLAG~PARAMETER2, family = binomial(link =
'logit'))
summary(model2)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER2, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3446  -0.1248  -0.1151  -0.1061   3.3166
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.03416    0.08475  -47.60   <2e-16 ***
## PARAMETER2   0.16938    0.01503   11.27   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22494  on 276961  degrees of freedom
## AIC: 22498
```

```
##
## Number of Fisher Scoring iterations: 8
```

Parameter 2 is significant and has good probability of having correlation with the response flag.

Parameter 3:

```
model3 = glm(data = full, RESPONSE_FLAG~PARAMETER3, family = binomial(link =
'logit'))
summary(model3)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER3, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.1384  -0.1171  -0.1170  -0.1169   3.1658
##
## Coefficients:
##             Estimate Std. Error  z value Pr(>|z|)
## (Intercept) -4.98103    0.02308 -215.769   <2e-16 ***
## PARAMETER3  -0.20112    0.77595   -0.259    0.795
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22609  on 276961  degrees of freedom
## AIC: 22613
##
## Number of Fisher Scoring iterations: 7
```

Not significant.

Paramter 4:

```
model4 = glm(data = full, RESPONSE_FLAG~PARAMETER4, family = binomial(link =
'logit'))
summary(model4)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER4, family = binomial(link =
"logit"),
##     data = full)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2213  -0.1177  -0.1168  -0.1160   3.1908
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.981379   0.023093 -215.71   <2e-16 ***
## PARAMETER4   0.002020   0.001616    1.25    0.211
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22608  on 276961  degrees of freedom
## AIC: 22612
##
## Number of Fisher Scoring iterations: 7
```

Not significant.

Parameter 5:

```
model5 = glm(data = full, RESPONSE_FLAG~PARAMETER5, family = binomial(link =
'logit'))
summary(model5)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER5, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2147  -0.1181  -0.1170  -0.1158   3.2135
##
## Coefficients:
##              Estimate Std. Error  z value Pr(>|z|)
## (Intercept) -4.981691   0.023100 -215.658   <2e-16 ***
## PARAMETER5   0.003386   0.002124    1.594    0.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22607  on 276961  degrees of freedom
## AIC: 22611
##
## Number of Fisher Scoring iterations: 7
```

Not significant.

Parameter 6:

```
model6 = glm(data = full, RESPONSE_FLAG~PARAMETER6, family = binomial(link =
'logit'))
summary(model6)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER6, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7622  -0.1203  -0.1167  -0.1128   3.2976
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.4088683  0.2688793 -23.835  < 2e-16 ***
## PARAMETER6   0.0015230  0.0002846   5.352 8.72e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22586  on 276961  degrees of freedom
## AIC: 22590
##
## Number of Fisher Scoring iterations: 7
```

Good significance.

Parameter 7:

```
model7 = glm(data = full, RESPONSE_FLAG~PARAMETER7, family = binomial(link =
'logit'))
summary(model7)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER7, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.1240  -0.0894  -0.0511  -0.0287   5.0412
##
```

```
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.601e+01  1.647e-01  -97.19   <2e-16 ***
## PARAMETER7   1.352e-02  1.787e-04   75.65   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 14107  on 276961  degrees of freedom
## AIC: 14111
##
## Number of Fisher Scoring iterations: 9
```

Great significance.

Parameter 8:

```
model8 = glm(data = full, RESPONSE_FLAG~PARAMETER8, family = binomial(link =
'logit'))
summary(model8)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER8, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2082  -0.1269  -0.1167  -0.1062   3.6032
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.83891    0.09937  -38.63   <2e-16 ***
## PARAMETER8  57.72444    5.05374   11.42   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22464  on 276961  degrees of freedom
## AIC: 22468
##
## Number of Fisher Scoring iterations: 8
```

Great significance.

Parameter 9:

```
model9 = glm(data = full, RESPONSE_FLAG~PARAMETER9, family = binomial(link =
'logit'))
summary(model9)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER9, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7422  -0.1244  -0.1146  -0.1055   3.3340
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.8698320  0.0916796  -42.21   <2e-16 ***
## PARAMETER9   0.0034049  0.0002805   12.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22477  on 276961  degrees of freedom
## AIC: 22481
##
## Number of Fisher Scoring iterations: 8
```

Great significance.

Parameter 10:

```
model10 = glm(data = full, RESPONSE_FLAG~PARAMETER10, family = binomial(link
= 'logit'))
summary(model10)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER10, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9503  -0.1236  -0.1118  -0.1032   3.3191
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.9757705  0.1418167  -49.19   <2e-16 ***
## PARAMETER10 -0.0034450  0.0002361  -14.59   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22427  on 276961  degrees of freedom
## AIC: 22431
##
## Number of Fisher Scoring iterations: 8
```

Great significance.

Parameter 11:

```
model11 = glm(data = full, RESPONSE_FLAG~PARAMETER11, family = binomial(link
= 'logit'))
summary(model11)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER11, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7732  -0.1222  -0.0896  -0.0643   4.0653
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 0.1644579  0.0945497   1.739    0.082 .
## PARAMETER11 0.0229762  0.0004586  50.105   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 20320  on 276961  degrees of freedom
## AIC: 20324
##
## Number of Fisher Scoring iterations: 8
```

The p-value here is lesser than the other parameters. Most associated with the response flag till now.

Parameter 12:

```
model12 = glm(data = full, RESPONSE_FLAG~PARAMETER12, family = binomial(link
= 'logit'))
summary(model12)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER12, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7881  -0.1207  -0.1148  -0.1094   3.2257
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.254726   0.040202 -130.71   <2e-16 ***
## PARAMETER12  0.013407   0.001534    8.74   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22542  on 276961  degrees of freedom
## AIC: 22546
##
## Number of Fisher Scoring iterations: 7
```

Great significance. Same as paramter 11.

Parameter 13:

```
model13 = glm(data = full, RESPONSE_FLAG~PARAMETER13, family = binomial(link
= 'logit'))
summary(model13)
```

```
##
## Call:
## glm(formula = RESPONSE_FLAG ~ PARAMETER13, family = binomial(link =
"logit"),
##     data = full)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.1250  -0.1194  -0.1171  -0.1147   3.4017
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.8153651  0.0701806 -68.614   <2e-16 ***
## PARAMETER13 -0.0018451  0.0007456  -2.475   0.0133 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22609  on 276962  degrees of freedom
## Residual deviance: 22603  on 276961  degrees of freedom
## AIC: 22607
##
## Number of Fisher Scoring iterations: 7
```

Not as much significant.

So, from our models, we have found that **parameters 2,6,7,8,9,10,11 and 12** are most associated with the response flag.

Question 3

Like question 2, question 3 is also a feature selection problem. And each of the variables have missing values. Also, all of them are categorical variables and the response flag is also a categorical variable. Now, to determine the relationship between two categorical variables, we can **Chi squared test**.

Let's glance through a couple variables and see if they are associated with the response flag or not:

1) UNIT_CARRIER_POS_X:

```
#Missing values first
sum(is.na(full$UNIT_CARRIER_POS_X))

## [1] 5236

table(full$UNIT_CARRIER_POS_X)

##
##     0     1     2     3     4
## 54319 54373 54373 54319 54343

#As '1' and '2' appear slightly more than the rest, I'm gonna assign 5236
missing values the value of '1'
full$UNIT_CARRIER_POS_X[is.na(full$UNIT_CARRIER_POS_X)] = 1

#Just loading it as a categorical variable
full$UNIT_CARRIER_POS_X = as.factor(full$UNIT_CARRIER_POS_X)

table1 = table(full$RESPONSE_FLAG, full$UNIT_CARRIER_POS_X)

chisq.test(table1)

##
##  Pearson's Chi-squared test
```

```
##
## data:  table1
## X-squared = 289.08, df = 4, p-value < 2.2e-16
```

Here, the p-value again is <0.05 proving that 'unit_carrier-pos_X' has high association with the response flag.

2)UNIT_CARRIER_POS_Y:

```
#Missing values first
sum(is.na(full$UNIT_CARRIER_POS_Y))

## [1] 5236

table(full$UNIT_CARRIER_POS_Y)

##
##      0      1
## 135782 135945

#As '1' and '2' appear slightly more than the rest, I'm gonna assign 5236
missing values the value of '1'
full$UNIT_CARRIER_POS_Y[is.na(full$UNIT_CARRIER_POS_Y)] = 1

#Just loading it as a categorical variable
full$UNIT_CARRIER_POS_Y = as.factor(full$UNIT_CARRIER_POS_Y)

table2 = table(full$RESPONSE_FLAG, full$UNIT_CARRIER_POS_Y)

chisq.test(table2)

##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table2
## X-squared = 2.1045, df = 1, p-value = 0.1469
```

This at it seems is not significant since p-value here is >0.05.

Material 2 supplier have way too many missing values to handle. Let's move on to material supplier 2:

3) VD tool lane:

```
table(full$VD_TOOL_LANE)

##
##      .   BACK  FRONT
##   5236 135945 135782

full$VD_TOOL_LANE = as.factor(full$VD_TOOL_LANE)

table3 = table(full$RESPONSE_FLAG, full$VD_TOOL_LANE)
```

```
chisq.test(table3)

##
##  Pearson's Chi-squared test
##
## data:  table3
## X-squared = 37.08, df = 2, p-value = 8.876e-09
```

This is also highly significant thus proving the association with the response flag.

4) Material 2 supplier facility:

```
table(full$MATERIAL2_SUPPLIER_FACILITY)

##
##      .     SE     SW  Tech1  Tech3
##   5236  84621   5376   4365 177365

full$MATERIAL2_SUPPLIER_FACILITY =
as.factor(full$MATERIAL2_SUPPLIER_FACILITY)

table4 = table(full$RESPONSE_FLAG, full$MATERIAL2_SUPPLIER_FACILITY)

chisq.test(table4)

##
##  Pearson's Chi-squared test
##
## data:  table4
## X-squared = 2386.2, df = 4, p-value < 2.2e-16
```

Material 2 supplier facility is also greatly associated with response flag.

Here, I haven't tried all the variables apart from the 13 parameters but from the chi-squared test and on the basis of p-value, we can find the association between 2 categorical variables. **Here from what I have already tried, I have one example of a variable (Unit_carrier_pos_y) not associated with the response flag while the other 3 examples (unit_carrier_pos_X, Vd_tool_lane & material 2 supplier facility).**

Bonus Question

From an analysis and data collection perspective, we could try to address this topic on several fronts: i) We could design a more robust sub-system ii) We could try to develop more reliable testing stations iii) we could tighten up and develop a fool proof inspection process

We could start this process by first finding the factors/variables/predictors which potentially cause the problem and strongly influence the occurence of the issue. We could use **mutual information** to find the relationship between two variables. Mutual information sometimes works even better than PCA.

Now, the cardinal point here to develop an additional filter that would be able to identify the failed units. **This is where we can apply advanced machine learning techniques to model the inspection process of failure detection.**The machine learning algorithms can help to **predict** when a detection failure is most likely to occur based on a historical evidence and provide the manufacturer an opportunity to proactively improve quality control to prevent the defective product.

So, the main idea here is the **Predictive Analytics**.

How it would work in short:

We could build a model using historic data and train it using known cases, to be able to identify the defective units or not. **Then we could use this trained model on unknown cases.** We would also go through validating the model using the current data where the outcome is 'likelihood of failure'. Validating the model should give us everything from accuracy to true positives, false positives, true negatives and false negatives. This would actually act as a filter and help the manufacturer to accurately detect the failed units before it even occurs.