We were still only managing to explain 37% of the data's variance with our model. Given how 'dirty' the target variable 'time\_delayed' is (because it is human reported and of dubious precision), we decided that perhaps we should rephrase the question in order to get a more sensible answer. When a bus breakdown is called into operations, perhaps the question to ask is: “Will this delay exceed twenty minutes?”

**Classification – more than 20 minutes?**

We could choose some other time, but for simplicity, twenty minutes is probably the breaking point of human patience – be they, passengers or providers. This division also breaks the dataset approximately in half, so we don't have to deal with imbalance. As a side note, I have also run the problem for a thirty minute or more delay, breaking the data into ¾ and ¼; and the solution only improves.

We begin by setting up the data that we need, and the caret control objects:

in\_csv <- "../output/intermediate/ii\_times.csv"

ii\_times <- read\_csv(in\_csv) %>%

filter(reported\_before\_resolved == 1) %>%

select(-reported\_before\_resolved) %>%

mutate(

time\_delayed = cut(time\_delayed, breaks = c(0, 22, 10000), labels = FALSE)

) %>%

mutate(time\_delayed = factor(time\_delayed, labels = c("t0.22","t22.")))

halfFold <- createFolds(ii\_times$Busbreakdown\_ID, k = 5, list=TRUE)

ii\_small <- ii\_times[halfFold[[2]],]

ii\_test <-ii\_times[halfFold[[3]],]

ii\_train <- ii\_times %>%

anti\_join(ii\_test, by = "Busbreakdown\_ID")

rctrl\_manual <- trainControl(method = "none", returnResamp = "all", classProbs = TRUE, summaryFunction = twoClassSummary)

rctrl\_repcv <- trainControl(method = "repeatedcv", number = 2, repeats = 5, returnResamp = "all", classProbs = TRUE, summaryFunction = twoClassSummary)

Then we can use cross-fold validation on a reduced data set to quickly get a sense of which parameters we should use. Note that we deliberately choose 'rules', not 'tree' here, because we want a human-readable set of if/then conditions. Unfortunately, we will end up using 15 trials, which means our rule set is large. I have commented its output here, but it can be read this way, or with 'summary(model)'.

#---cubist

c50\_grd<-expand.grid(

.winnow = FALSE,

.trials=c(15,30,50),

.model="rules"

)

set.seed(849)

c50\_cv <- train(time\_delayed ~ . -Busbreakdown\_ID,

data = ii\_small,

method = "C5.0",

metric="ROC",

na.action = na.pass,

trControl = rctrl\_repcv,

tuneGrid = c50\_grd

)

c50\_cv

*## C5.0*

*##*

*## 40665 samples*

*## 46 predictor*

*## 2 classes: 't0.22', 't22.'*

*##*

*## No pre-processing*

*## Resampling: Cross-Validated (2 fold, repeated 5 times)*

*## Summary of sample sizes: 20332, 20333, 20332, 20333, 20332, 20333, ...*

*## Resampling results across tuning parameters:*

*##*

*## trials ROC Sens Spec*

*## 15 0.8261069 0.8028816 0.6830990*

*## 30 0.8258429 0.8030490 0.6855583*

*## 50 0.8258429 0.8030490 0.6855583*

*##*

*## Tuning parameter 'model' was held constant at a value of rules*

*##*

*## Tuning parameter 'winnow' was held constant at a value of FALSE*

*## ROC was used to select the optimal model using the largest value.*

*## The final values used for the model were trials = 15, model = rules*

*## and winnow = FALSE.*

#writeLines(c50\_cv$finalModel$rules)

#We can output the rule set as a block of text like this, but we have suppressed this output because with 50 trials, it is a very, very long list.

The system is relatively insensitive to the number of trials, so we can use 15. We run on our training and test sets to derive our true test ROC, and have a look at how our predicted probabilities correlate with the predictors, compared to the actual values.

c50\_grd<-expand.grid(

.winnow = FALSE,

.trials=15,

.model="rules"

)

c50\_man <- train(time\_delayed ~ . -Busbreakdown\_ID,

data = ii\_train,

method = "C5.0",

metric="ROC",

na.action = na.pass,

trControl = rctrl\_manual,

tuneGrid = c50\_grd

)

predictions <- predict(c50\_man, newdata = ii\_test, type = "prob")

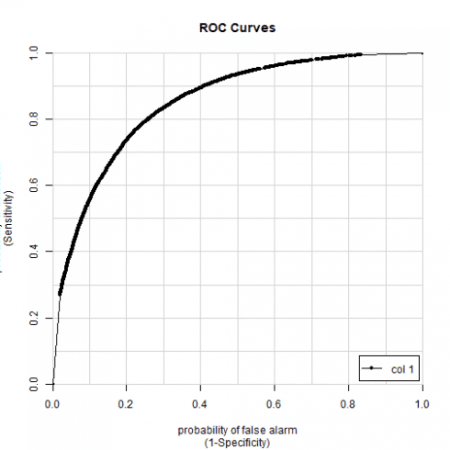
ii\_withpred <- ii\_test %>%

cbind(predictions %>% tbl\_df())

colAUC(ii\_withpred[["t0.22"]], (ii\_withpred[["time\_delayed"]] =="t0.22") \* 1L, plotROC=TRUE)

*## [,1]*

*## 0 vs. 1 0.8486122*

[](https://i0.wp.com/datascienceplus.com/wp-content/uploads/2018/11/unnamed-chunk-3-1-15.png?ssl=1)

#cubist: 0.847

corrr\_analysis <- ii\_withpred %>%

select(-Busbreakdown\_ID, -t0.22, -t22.) %>%

mutate(

time\_delayed = (time\_delayed == "t22.")\*1L

) %>%

correlate() %>%

focus(time\_delayed) %>%

rename(feature = rowname) %>%

arrange(desc(abs(time\_delayed))) %>%

mutate(feature = as\_factor(feature))

corrr\_analysis %>% print(n=61)

*## # A tibble: 45 x 2*

*## feature time\_delayed*

*##*

*## 1 vehicle\_total\_with\_attendants 0.176*

*## 2 Has\_Contractor\_Notified\_Parents 0.171*

*## 3 Boro\_Bronx -0.165*

*## 4 drivers\_total\_attendant 0.163*

*## 5 vehicle\_total\_max\_riders -0.162*

*## 6 vehicle\_total\_reg\_seats -0.160*

*## 7 Boro\_Manhattan 0.151*

*## 8 service\_type\_d2d 0.149*

*## 9 Number\_Of\_Students\_On\_The\_Bus -0.149*

*## 10 Boro\_StatenIsland -0.140*

*## 11 vehicle\_total\_ambulatory\_seats -0.0972*

*## 12 rush\_min\_from\_peak 0.0954*

*## 13 drivers\_staff\_servSchool 0.0792*

*## 14 Reason\_MechanicalProblem 0.0729*

*## 15 Reason\_HeavyTraffic -0.0698*

*## 16 Boro\_Queens 0.0667*

*## 17 vehicle\_total\_with\_lifts 0.0654*

*## 18 rush\_within -0.0612*

*## 19 time\_am -0.0545*

*## 20 rush\_between 0.0538*

*## 21 Reason\_DelayedbySchool -0.0471*

*## 22 Reason\_FlatTire 0.0461*

*## 23 Have\_You\_Alerted\_OPT -0.0457*

*## 24 Reason\_Accident 0.0427*

*## 25 Boro\_Brooklyn 0.0421*

*## 26 School\_Age 0.0416*

*## 27 drivers\_numServ\_prek -0.0394*

*## 28 Boro\_NassauCounty 0.0389*

*## 29 drivers\_numServ\_school 0.0374*

*## 30 Reason\_WontStart 0.0371*

*## 31 Has\_Contractor\_Notified\_Schools 0.0346*

*## 32 drivers\_num\_servPreK -0.0315*

*## 33 drivers\_staff\_servPreK -0.0305*

*## 34 Reason\_WeatherConditions 0.0264*

*## 35 vehicle\_total\_disabled\_seats -0.0252*

*## 36 Boro\_NewJersey 0.0236*

*## 37 Reason\_ProblemRun 0.0193*

*## 38 rush\_outside 0.0164*

*## 39 Boro\_Connecticut -0.0110*

*## 40 Active\_Vehicles -0.0108*

*## 41 Boro\_Westchester -0.0103*

*## 42 Reason\_LatereturnfromFieldTrip 0.0102*

*## 43 Boro\_RocklandCounty 0.00316*

*## 44 drivers\_num\_servSchool 0.00243*

*## 45 drivers\_total\_driver -0.00199*

corrr\_pred <- ii\_withpred %>%

select(-Busbreakdown\_ID, -time\_delayed, -t0.22) %>%

correlate() %>%

focus(t22.) %>%

rename(feature = rowname) %>%

arrange(desc(abs(t22.))) %>%

mutate(feature = as\_factor(feature))

corrr\_full <- corrr\_analysis %>%

left\_join(corrr\_pred, by = "feature") %>%

mutate(

#pred\_rescale = t22.,

pred\_rescale = t22. \* median(abs(time\_delayed)) / median(abs(t22.)),

t22. = NULL

) %>%

gather(key = key, value = time\_delayed, -feature)

pal <- brewer.pal(8, "Dark2")

corrr\_full %>%

ggplot(aes(x = time\_delayed, y = fct\_reorder(feature, desc(time\_delayed)), color = key)) +

geom\_point() +

scale\_color\_manual(values = c(pal[2],pal[5])) +

# Positive Correlations - Contribute to churn

geom\_segment(aes(xend = 0, yend = feature, color = key),

data = corrr\_full %>% filter(time\_delayed > 0)) +

# Negative Correlations - Prevent churn

geom\_segment(aes(xend = 0, yend = feature, color = key),

data = corrr\_full %>% filter(time\_delayed < 0)) +

# Vertical lines

geom\_vline(xintercept = 0, color = plasma(3)[[2]], size = 1, linetype = 2) +

geom\_vline(xintercept = -0.25, color = plasma(3)[[2]], size = 1, linetype = 2) +

geom\_vline(xintercept = 0.25, color = plasma(3)[[2]], size = 1, linetype = 2) +

# Aesthetics

theme\_bw() +

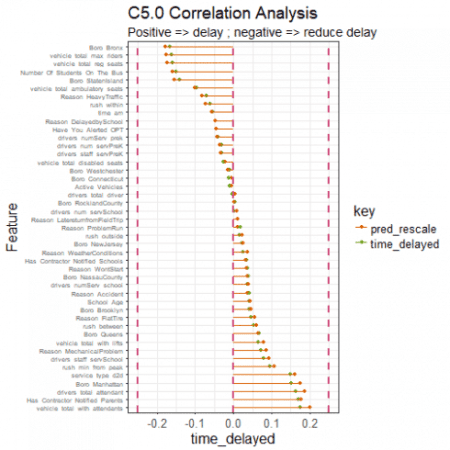
labs(title = "C5.0 Correlation Analysis",

subtitle = paste("Positive => delay",

"; negative => reduce delay"),

y = "Feature") +

theme(text=element\_text(family="Arial", size=16), axis.text.y = element\_text(size=7))

[](https://i1.wp.com/datascienceplus.com/wp-content/uploads/2018/11/unnamed-chunk-3-2-1.png?ssl=1)

Our ROC is 0.848, which we cannot complain about. Our predictions correlate with our predictors as expected. With a fairly limited set of predictors, and despite our very dirty target variable, we can be reasonably certain whether a delay will be greater or less than twenty minutes. But, of course, we should keep in mind that the equivalent regression is rather bad. We have had to choose our poison, and have made do.

I am leaving this data problem here. There is more than I could do to add predictors. One thing that I wanted to do at the outset of this problem was finding the geospatial data for the routes themselves, and see if I could add predictors based on neighborhood council fees, or house prices, or crime statistics, or the number of dogs / birds / trees, or whatever else seemed fun. But I have come far enough with this (for now) and will be moving on to something different. If you would like any of the ‘cleaner’ data, let me know and I’ll see what I can do.