MDS556 Final - Techical Report

# Purpose

As much of this term has been spent dicussing classification methods that are used for prediction, it seems apropos to apply some of the models discussed to an actual business problem at my company (hereafter called Company A): will this product be manufactured in time?

This seems like a simple question at first glance. It would be possible to do things like determine the mean likelihood of on-time completion and apply this to future productions. However, given the multiple product types, work centers, and changes over time, it suddenly becomes more difficult to determine. As such, this paper will focus on the feature engineering and model selection for classifying producion orders (smallest unit of manufactured product at Company A) into groups of “On-time” and “Past Due.”

If the model performs well, then additional categories may be defined and tested as well, to give a more detailed prediction of completion.

# Background

Currently at Company A all manufactured items (from components used in other products to the final products themselves) are called production orders. These orders contain information such as the final item, the quantity, descriptions, and many other categorical variables (too many to dive into in this paper). These production orders also contain relationships to routing lines (the steps undertaken to build the item) as well as the componentry and sales lines. These production orders are the primary method of production as Company A only outsources minimal amounts of assembly work.

One of missions of our production team is to manage the level loading and output of the independent work centers. This requires a certain level of certainty about the ability of the cell to complete the work at hand. Great care has been taken to develop tools for this level loading, and it would appear on the surface to have helped produce more materials on-time. As part of this project, we will be analyzing this assumption and use regression to see if the on-time completion has improved over time. If this is the case, then it would make sense to rerun this classifier at intervals to improve the model performance over time. Part of the reason to create such a classifier is to integrate the prediction into the level loading tools such that the individual loading the work centers can begin to look at the likelihood of on-time completion and come up with methods to improve such metrics.

# Data

## Extraction

Data is drawn directly out of the production system and as such the method of extraction will be partially obscurred for security. The data was extracted using SQL and has had some aggregations and joins already applied in order to simplify later steps.

The leading table is that of the Production Order lines and two other related tables have been joined onto this (the Production Order Components and the Production Order Routing Lines). These two sources provide some additional detail that is not stored on the production order itself. The production orders have been filtered already to only include those which were due in the past 2 years, as well as removal of phantom items (such as hardware kits, which have no assembly time and are a unique aspect of our ERP system).

Data is then stored as a .xlsx file and is exported for use here. The raw data is roughly 14MB, consisting of 141,710 records with 23 fields.

library("tidyverse")

## -- Attaching packages -------------------------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.2.1 v purrr 0.3.2  
## v tibble 2.1.3 v dplyr 0.8.3  
## v tidyr 1.0.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.4.0

## -- Conflicts ----------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

#Fields that were interpreted wrong were defined manually  
RawData <- readxl::read\_xlsx('E:/Shared drives/Sale/Escuela/Elmhurst\_MSDS/MDS\_556\_analytical\_methods/ClassifierRaw.xlsx', col\_types = c("guess","guess","guess","guess","text","guess","guess","guess","guess","guess","guess","text","guess","guess","guess","guess","guess","guess","guess","date","guess","guess","guess"))

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T82 / R82C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T37226 / R37226C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T37227 / R37227C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T49562 / R49562C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T49563 / R49563C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T62631 / R62631C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T75115 / R75115C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T75116 / R75116C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting numeric in V77347 / R77347C22: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T78429 / R78429C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T134937 / R134937C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T134938 / R134938C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T134939 / R134939C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T134940 / R134940C20: got 'NULL'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i =  
## sheet, : Expecting date in T134941 / R134941C20: got 'NULL'

head(RawData)

## # A tibble: 6 x 23  
## `Prod\_ Order No~ `Line No\_` `Item No\_` `Sales Order No~ Segment  
## <chr> <dbl> <chr> <dbl> <chr>   
## 1 M6039184 10000 80620324 NA <NA>   
## 2 M6043550 10000 E0041086 NA <NA>   
## 3 M6043909 10000 1115145 1170509 5   
## 4 M6044245 10000 1199474 1170509 4   
## 5 M6044246 10000 1199474 1170509 4   
## 6 M6044247 10000 1199474 1170509 4   
## # ... with 18 more variables: `Variant Code` <chr>, Description <chr>,  
## # `Item Category Code` <chr>, Department <chr>, `Final Output  
## # Location` <chr>, `Inventory Posting Group` <chr>, `Planner  
## # Code` <chr>, `Job Color` <chr>, `Finished Quantity` <dbl>, `Ending  
## # Date` <dttm>, `Ending Date Modifier` <dbl>, `Modified Ending  
## # Date` <dttm>, `Due Date` <dttm>, Destination <chr>, `Finish  
## # Date` <dttm>, `Count of Components` <dbl>, `Sum of Component  
## # Quantity` <dbl>, `Routing Steps` <dbl>

## Transformation

One of the first changes that must take place is the calculation of the “On-Time” label for each record. Two fields will actually be made, one will be the number of workdays past due (negative indicates early completion), and the second will be a binary label of TRUE = “Past Due” and FALSE = “On-Time.” The dates used for this calculation are labeled “Modified Ending Date” and the “Finished Date.”

The modified ending date is used instead of the due date because this is the date in which assembly has level loaded the items. If the production order has been modified, then the “Ending Date Modifier” will be something other than zero, indicating level loading tools were used to better prepare the work centers. As this is the date in which they have committed to build and complete the production order, it is used to measure on-time performance.

The default method of removing weekends from the *bizdays* package will be used as the calendar. There may be a slight change from Company A’s calendar due to holidays, but they will be insignificant at worst and the existing metrics for on-time completion are similarly computed.

library("bizdays")

##   
## Attaching package: 'bizdays'

## The following object is masked from 'package:stats':  
##   
## offset

#Only removing weekends  
WeekDaysCal <- create.calendar(name='WeekDays', weekdays=c('sunday', 'saturday'))  
  
#Add DaysPastDue to original data  
RawData$DaysPastDue = ifelse(bizdays(RawData$`Modified Ending Date`,RawData$`Finish Date`, WeekDaysCal) >= 0,  
 bizdays(RawData$`Modified Ending Date`,RawData$`Finish Date`, WeekDaysCal),  
 - bizdays(RawData$`Finish Date`,RawData$`Modified Ending Date`, WeekDaysCal)  
 )  
  
#Add PastDue to modified data  
RawData$PastDue = ifelse(RawData$DaysPastDue > 0,  
 TRUE,  
 FALSE)  
  
PastDueView <- RawData[c(1:10), c(1,17,20,24,25)]  
  
PastDueView

## # A tibble: 10 x 5  
## `Prod\_ Order No~ `Modified Ending D~ `Finish Date` DaysPastDue  
## <chr> <dttm> <dttm> <dbl>  
## 1 M6039184 2017-12-11 00:00:00 2017-12-18 00:00:00 5  
## 2 M6043550 2017-12-19 00:00:00 2017-11-29 00:00:00 -14  
## 3 M6043909 2018-01-03 00:00:00 2018-01-09 00:00:00 4  
## 4 M6044245 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 5 M6044246 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 6 M6044247 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 7 M6044248 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 8 M6044249 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 9 M6044250 2018-01-02 00:00:00 2018-01-05 00:00:00 3  
## 10 M6044251 2018-01-02 00:00:00 2018-01-05 00:00:00 3  
## # ... with 1 more variable: PastDue <lgl>

In addition to these steps, another field will be added to categorize the number of days past due. This is for creating a more granular classifier that will define how late or early, rather than just past due or on-time.

RawData$PastDueCategory = ifelse(RawData$DaysPastDue >= 4,  
 "4 or greater",  
 ifelse(RawData$DaysPastDue <= 3 & RawData$DaysPastDue >= 1,  
 "1 to 3",  
 ifelse(RawData$DaysPastDue == 0,  
 "0",  
 ifelse(RawData$DaysPastDue <= -1 & RawData$DaysPastDue >= -3,  
 "-3 to -1",  
 ifelse(RawData$DaysPastDue <= -4,  
 "-4 or less",  
 "N/A"  
 )  
 )  
 )  
 )  
 )  
  
PastDueView2 <- RawData[c(1:10), c(1,17,20,24,25,26)]  
  
PastDueView2

## # A tibble: 10 x 6  
## `Prod\_ Order No~ `Modified Ending D~ `Finish Date` DaysPastDue  
## <chr> <dttm> <dttm> <dbl>  
## 1 M6039184 2017-12-11 00:00:00 2017-12-18 00:00:00 5  
## 2 M6043550 2017-12-19 00:00:00 2017-11-29 00:00:00 -14  
## 3 M6043909 2018-01-03 00:00:00 2018-01-09 00:00:00 4  
## 4 M6044245 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 5 M6044246 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 6 M6044247 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 7 M6044248 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 8 M6044249 2018-01-02 00:00:00 2018-01-03 00:00:00 1  
## 9 M6044250 2018-01-02 00:00:00 2018-01-05 00:00:00 3  
## 10 M6044251 2018-01-02 00:00:00 2018-01-05 00:00:00 3  
## # ... with 2 more variables: PastDue <lgl>, PastDueCategory <chr>

Now we will also remove any row which contains a value of N/A in the “Finish Date” or “Destination” columns, remove the columns “Department” and “Line No” as they are the same for all records, and filter for only a select group of “Destination” categories (work centers that are level loaded by the business).

unique(RawData$Destination)

## [1] "109" "901" NA "939"   
## [5] "ABENCHASSY" "BBENCHASSY" "926" "ASY-150-B"   
## [9] "ASY-150-A" "184" "103" "CBENCHASSY"   
## [13] "301" "102DB" "370" "110"   
## [17] "250S" "ASY-150-C" "998" "906"   
## [21] "186" "335" "175" "120"   
## [25] "101A" "907" "105" "249"   
## [29] "940" "LINEASSY" "934" "908"   
## [33] "999" "305" "182" "935"   
## [37] "370RF" "399" "370B" "182B"   
## [41] "943" "947" "ASY-NBC-165" "ASY-MOUNT"   
## [45] "ASY-CZB-150" "NULL" "ASY-SPECIALS" "ASY-ITR-130"   
## [49] "ASY-ITR-130A" "948"

WorkCenters <- c("ASY-NBC-165","ASY-MOUNT","ASY-CZB-150","ASY-SPECIALS","ASY-ITR-130","ASY-ITR-130A","370RF")  
  
ModData <- RawData %>% select(-c(Department,`Line No\_`,`Variant Code`)) %>% filter(RawData$Destination %in% WorkCenters) %>% drop\_na(`Finish Date`,Destination)

# Exploration

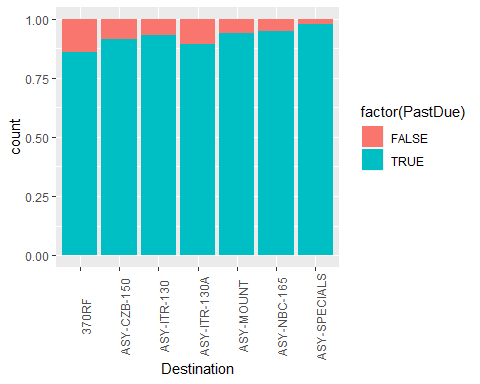
Before beginning the modelling procedures it is prudent to see which features in this set may have useful information and which ones might be able to be removed. In addition, we will look to see if there are any other insights that may be useful for a business user (such as the proportion of past due work separated by the work center).

## Visualization

First we will look to see how the past due value relates to some of the categories by plotting a couple of the main features from knowledge of the business which would make the most sense.

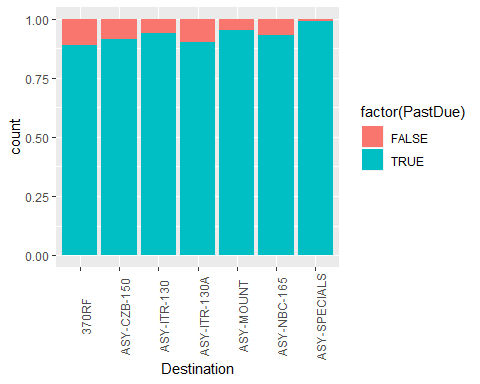
### Past Due Proportion by Destination (Work Center)

PDbyDest <- ggplot(data = ModData, aes(x = Destination, fill = factor(PastDue))) +   
 geom\_bar(position = "fill") +   
 theme(axis.text.x = element\_text(angle = 90))  
  
PDbyDest



The first piece of learning to be had here, is that all the new work centers are quite poor at producing materials in a timely manner. Now Company A has recently transitioned to a new facility and all the work centers have been drastically altered, so in the time range we are looking at it would make sense that the performance is quite poor. Perhaps a shortening of time would improve such metrics. We will shorten the period to only include months after the initial move and setup phase.

ModDataShort <- ModData %>% filter(ModData$`Modified Ending Date` > as.Date("2019-05-01"))  
  
PDbyDest2 <- ggplot(data = ModDataShort, aes(x = Destination, fill = factor(PastDue))) +   
 geom\_bar(position = "fill") +   
 theme(axis.text.x = element\_text(angle = 90))  
  
PDbyDest2

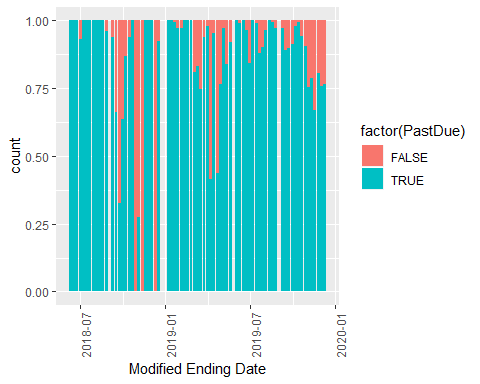


Although we have shortened the time frame, the work centers have not really improved in their performance. As such we will use the whole time period instead of the shortened period.

### Past Due by Date

The above conclusion is also supported if we plot the past due proportion over a time span (only looking at Mondays in order to clean up the plot). An additional weekday field may also assist in the model building process if we base it off of the completion date, so another field will also be added here.

ModData$wd = weekdays(ModData$`Modified Ending Date`)  
  
ModData$wdFinish = weekdays(ModData$`Finish Date`)  
  
MDMon <- ModData %>% filter(wd == "Monday")  
  
PDbyDate <- ggplot(data = MDMon, aes(x = `Modified Ending Date`, fill = factor(PastDue))) +   
 geom\_bar(position = "fill") +   
 theme(axis.text.x = element\_text(angle = 90))  
  
PDbyDate



# Correlations

This data set contains many categorical fields, which do not allow for the easy creation of correlation matrices without a little bit of modification. In order to create a matrix, “One Hot” encoding will be applied to the ModData set to convert features with multiple factor levels into multiple features all with a binary value.

First we will look at the structure of the data frame, then change the necessary features to a type of factor. We will then reexamine and remove unnecessary features that will overly complicate the model (such as the description field, which is almost as unique as the item number and their heavy covariance negates its use). After this we will apply One Hot encoding using the *mltools* package.

library("mltools")

##   
## Attaching package: 'mltools'

## The following object is masked from 'package:tidyr':  
##   
## replace\_na

library("data.table")

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

str(ModData)

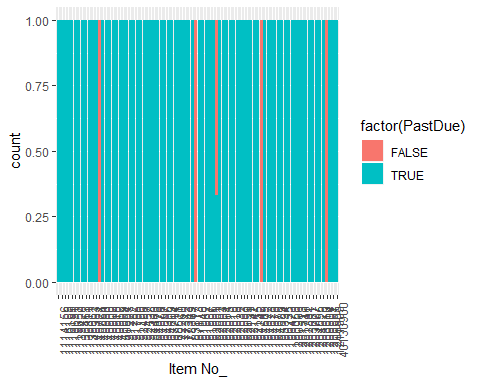
## Classes 'tbl\_df', 'tbl' and 'data.frame': 48503 obs. of 25 variables:  
## $ Prod\_ Order No\_ : chr "M6122174" "M6142336" "M6142337" "M6142338" ...  
## $ Item No\_ : chr "QS1191944" "1191947" "1191947" "1191973" ...  
## $ Sales Order No\_ : num NA 1177605 1177605 1178000 1177605 ...  
## $ Segment : chr NA "1" "1" "1" ...  
## $ Description : chr "SUPPORT,ACC2-RF-13BF SAMPLE" "SUPPORT,ACC2-RF-28BF" "SUPPORT,ACC2-RF-28BF" "SUPPORT,ACC2-RF-16BF" ...  
## $ Item Category Code : chr "MI" "PRODUCT" "PRODUCT" "PRODUCT" ...  
## $ Final Output Location : chr "A023E" "SHIP1210" "SHIP1210" "SHIP1210" ...  
## $ Inventory Posting Group : chr "INV" "PRODUCT" "PRODUCT" "PRODUCT" ...  
## $ Planner Code : chr "B10" "5000" "5000" "5000" ...  
## $ Job Color : chr NA "ER-3-PWD" "ER-3-PWD" "ER10-PWD" ...  
## $ Finished Quantity : num 20 2 2 3 1 2 1 11 4 8 ...  
## $ Ending Date : POSIXct, format: "2018-04-05" "2018-06-08" ...  
## $ Ending Date Modifier : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Modified Ending Date : POSIXct, format: "2018-04-05" "2018-06-08" ...  
## $ Due Date : POSIXct, format: "2018-04-09" "2018-06-12" ...  
## $ Destination : chr "370RF" "370RF" "370RF" "370RF" ...  
## $ Finish Date : POSIXct, format: "2018-05-09" "2018-06-12" ...  
## $ Count of Components : num 10 11 11 11 11 11 11 11 11 11 ...  
## $ Sum of Component Quantity: num 48 41 41 41 41 41 43 43 43 43 ...  
## $ Routing Steps : num 2 4 4 4 4 4 4 4 4 4 ...  
## $ DaysPastDue : num 24 2 2 1 2 2 2 4 2 2 ...  
## $ PastDue : logi TRUE TRUE TRUE TRUE TRUE TRUE ...  
## $ PastDueCategory : chr "4 or greater" "1 to 3" "1 to 3" "1 to 3" ...  
## $ wd : chr "Thursday" "Friday" "Friday" "Thursday" ...  
## $ wdFinish : chr "Wednesday" "Tuesday" "Tuesday" "Friday" ...

FacCol <- c("Item No\_","Sales Order No\_","Segment","Item Category Code","Final Output Location","Inventory Posting Group","Planner Code","Job Color","Finished Quantity","Ending Date Modifier","Destination","Count of Components","Sum of Component Quantity","Routing Steps","DaysPastDue","PastDue","PastDueCategory","wd","wdFinish")  
  
#Commented out, see following section  
# ModFac <- mutate\_at(ModData,vars(FacCol), as.factor)  
#   
# ModFac <- ModFac %>% select(-c(`Prod\_ Order No\_`,Description,`Ending Date`,`Due Date`))  
#   
# str(ModFac)  
#   
# ModFacDT <- as.data.table(ModFac)  
#   
# ModHot <- one\_hot(ModFacDT, cols = "auto")  
#   
# ModHot

After all the necessary conversions we end up with a table that contains the same number of records (48,503), but with drastically more features (7812) which are predominantly due to the “Item No\_” field from the raw data. Due to the radical size increase, the original code was altered and has been simplified to remove the “Item No\_” after a brief examination of the item number impact on past due work.

## Past Due by Item Investigation

PDItemSample <- ModData[sample(nrow(ModData),100),]  
  
PDbyItem <- ggplot(data = PDItemSample, aes(x = `Item No\_`, fill = factor(PastDue))) +   
 geom\_bar(position = "fill") +   
 theme(axis.text.x = element\_text(angle = 90))  
  
PDbyItem



Here we can see that the item number does have an effect on the past due, but since the majority of items show that the majority of the time they are past due, it seems reasonable to remove this factor. For this same reason, as well as the fact that the sales order is a not reused after job completion, the “Sales Order No\_” has also been removed, but the segment has been retained (this value can repeat in other orders and thus may provide predictive valuye).

## New Encoding

ModFac <- mutate\_at(ModData,vars(FacCol), as.factor)  
  
ModFac <- ModFac %>% select(-c(`Prod\_ Order No\_`,Description,`Ending Date`,`Due Date`,`Item No\_`,`Sales Order No\_`,`Modified Ending Date`,`Finish Date`))  
  
str(ModFac)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 48503 obs. of 17 variables:  
## $ Segment : Factor w/ 137 levels "0.1","1","1.1000000000000001",..: NA 2 2 2 2 2 2 2 2 2 ...  
## $ Item Category Code : Factor w/ 2 levels "MI","PRODUCT": 1 2 2 2 2 2 2 2 2 2 ...  
## $ Final Output Location : Factor w/ 7 levels "1300-IMP1","1300TEMP",..: 3 6 6 6 6 6 6 6 6 6 ...  
## $ Inventory Posting Group : Factor w/ 2 levels "INV","PRODUCT": 1 2 2 2 2 2 2 2 2 2 ...  
## $ Planner Code : Factor w/ 4 levels "4000","4100",..: 4 3 3 3 3 3 3 3 3 3 ...  
## $ Job Color : Factor w/ 61 levels "ER-1-PWD","ER-2-PWD",..: NA 3 3 4 3 3 3 3 3 3 ...  
## $ Finished Quantity : Factor w/ 58 levels "1","2","3","4",..: 20 2 2 3 1 2 1 11 4 8 ...  
## $ Ending Date Modifier : Factor w/ 4 levels "-1","0","1","2": 2 2 2 2 2 2 2 2 2 2 ...  
## $ Destination : Factor w/ 7 levels "370RF","ASY-CZB-150",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Count of Components : Factor w/ 159 levels "1","2","3","4",..: 10 11 11 11 11 11 11 11 11 11 ...  
## $ Sum of Component Quantity: Factor w/ 556 levels "0","1","2","4",..: 69 56 56 56 56 56 62 62 62 62 ...  
## $ Routing Steps : Factor w/ 12 levels "1","2","3","4",..: 2 4 4 4 4 4 4 4 4 4 ...  
## $ DaysPastDue : Factor w/ 38 levels "-9","-6","-5",..: 32 10 10 9 10 10 10 12 10 10 ...  
## $ PastDue : Factor w/ 2 levels "FALSE","TRUE": 2 2 2 2 2 2 2 2 2 2 ...  
## $ PastDueCategory : Factor w/ 5 levels "-3 to -1","-4 or less",..: 5 4 4 4 4 4 4 5 4 4 ...  
## $ wd : Factor w/ 5 levels "Friday","Monday",..: 3 1 1 3 1 1 1 1 1 1 ...  
## $ wdFinish : Factor w/ 7 levels "Friday","Monday",..: 7 6 6 1 6 6 6 5 6 6 ...

ModFacDT <- as.data.table(ModFac)  
  
ModHot <- one\_hot(ModFacDT, cols = "auto")

After this new encoding we have reduced the features to 1069, which is much more manageable than the 7812 that we saw previously.

## Correlation Matrix

Unfortunately, the volume of features does mean that using a visualization for the correlation matrix is not really practical, as the pixels become incredibly small and meaningful correlations hard to determine clearly. We will remove the non-factor variable in order to demonstrate the issue and allow the use of a correlation matrix. This is one of the heaviest computations thus far due to the number of features found in the set. Unfortunately, again we run into an error though, as the correlation matrix has many N/A values, thus keeping us from plotting.

library("Hmisc")

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':  
##   
## src, summarize

## The following objects are masked from 'package:base':  
##   
## format.pval, units

CorelMat <- rcorr(as.matrix(ModHot))

## Warning in sqrt(1 - h \* h): NaNs produced

CorHeat <- heatmap(CorelMat, na.rm = FALSE)

## Error in heatmap(CorelMat, na.rm = FALSE): 'x' must be a numeric matrix

Even though we cannot plot, we can extract the correlation coefficients and see which ones have the greatest relationships.

RMat <- as.matrix(CorelMat$r)  
  
RVals <- data.frame(row=rownames(RMat)[row(RMat)[upper.tri(RMat)]],   
 col=colnames(RMat)[col(RMat)[upper.tri(RMat)]],   
 corrnum=RMat[upper.tri(RMat)])  
  
TopRVals <- RVals %>% filter(col %in% c("PastDue\_TRUE","PastDue\_FALSE"))  
  
TopRVals$AbsCorr = abs(TopRVals$corrnum)  
  
TopRVals <- TopRVals[order(-TopRVals$AbsCorr),]  
  
head(TopRVals)

## row col corrnum AbsCorr  
## 2095 PastDue\_FALSE PastDue\_TRUE -1.0000000 1.0000000  
## 2064 DaysPastDue\_0 PastDue\_TRUE -0.7847488 0.7847488  
## 1017 DaysPastDue\_0 PastDue\_FALSE 0.7847488 0.7847488  
## 2063 DaysPastDue\_-1 PastDue\_TRUE -0.4947048 0.4947048  
## 1016 DaysPastDue\_-1 PastDue\_FALSE 0.4947048 0.4947048  
## 2062 DaysPastDue\_-2 PastDue\_TRUE -0.2639319 0.2639319

We can see here that clearly the days past due are correlated with the Past Due field, but if we look further we actually see a couple interesting fields like “Destination\_370RF” and “Ending Date Modifier\_2” which also have correlations, even if they are not very strong.

# Modeling

Now that certain features have been explored, a couple models will be tested to see which performs best for classifying past due production orders.

## Data Prep

First we will split the data into training and test sets. We will use 3 different sets in order to see which performs best (Only factor data, One Hot encoded data, or the full data set (without the couple columns remove early on)). The sets are *ModHot* for the encoded, *ModFac* for the same data as found in *ModHot* just without the encoding, as well as the normal *ModData* set (including the columns that are removed from the others).

library("caTools")  
library("randomForest")

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

set.seed(794613)  
  
ModHot <- ModHot %>% drop\_na(PastDue\_TRUE)  
ModFac <- ModFac %>% drop\_na(PastDue)  
ModData <- ModData %>% drop\_na(PastDue)  
  
HotSample = sample.split(ModHot$Segment\_0.1, SplitRatio = 0.80)  
HotTrain = subset(ModHot, HotSample == TRUE)  
HotTest = subset(ModHot, HotSample == FALSE)  
  
FacSample = sample.split(ModFac$Segment, SplitRatio = 0.80)  
FacTrain = subset(ModFac, FacSample == TRUE)  
FacTest = subset(ModFac, FacSample == FALSE)  
  
FullSample = sample.split(ModData$Segment, SplitRatio = 0.80)  
FullTrain = subset(ModData, FullSample == TRUE)  
FullTest = subset(ModData, FullSample == FALSE)

## Random Forest

The first model will be that of a Random Forest, primarily to test to efficacy of using the One Hot encoding vs. the data set that is not encoded, to see if perhaps one should be pursued over the other.

### One Hot Encoded Data

HotForest <- randomForest(PastDue\_TRUE ~ ., data = HotTrain, ntree = 100, na.action = na.roughfix)

## Error in eval(predvars, data, env): object 'Item Category Code\_MI' not found

Our first test with the One Hot data was quite disappointing as the input had some N/A values and I do not feel comfortable trying to impute them, thus we have to try the other sets. Evaluation was set to false as the computation time is non-trivial and only returns an error.

### Only Factor Data

FactorForest <- randomForest(PastDue ~ ., FacTrain, ntree = 100, na.action = na.roughfix)

## Error in eval(predvars, data, env): object 'Item Category Code' not found

Applying the roughfix method to impute the values we see that we still receive an error, but this time related to the replacement value. To be entirely honest, the error and traceback do not make much sense as I am not sure how to interpret the function calls. This is also disappointing and leads us to the last test.

### All Data

FullForest <- randomForest(PastDue ~ ., FacTrain, ntree = 100, na.action = na.roughfix)

## Error in eval(predvars, data, env): object 'Item Category Code' not found

Here we get the same error! This must indicate that the data set is missing too many values. However, if we use *na.omit* to try and remove the records with issues, we end up with no data at all. As such, I returned to the very top of the analysis, and removed the “Variant Code” feature. This allows for more data, but now the ’Item Category Code" has taken the place of the N/A issue, and all models run into the same problem. This field is fully filled with values and is in the data sets, so the error messages are effectively impossible to interpret and correct for me.

HotOmit <- na.omit(ModHot)  
nrow(HotOmit)

## [1] 48303

FacOmit <- na.omit(ModFac)  
nrow(FacOmit)

## [1] 48303

FullOmit <- na.omit(ModData)  
nrow(FullOmit)

## [1] 48302

str(ModFac)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 48503 obs. of 17 variables:  
## $ Segment : Factor w/ 137 levels "0.1","1","1.1000000000000001",..: NA 2 2 2 2 2 2 2 2 2 ...  
## $ Item Category Code : Factor w/ 2 levels "MI","PRODUCT": 1 2 2 2 2 2 2 2 2 2 ...  
## $ Final Output Location : Factor w/ 7 levels "1300-IMP1","1300TEMP",..: 3 6 6 6 6 6 6 6 6 6 ...  
## $ Inventory Posting Group : Factor w/ 2 levels "INV","PRODUCT": 1 2 2 2 2 2 2 2 2 2 ...  
## $ Planner Code : Factor w/ 4 levels "4000","4100",..: 4 3 3 3 3 3 3 3 3 3 ...  
## $ Job Color : Factor w/ 61 levels "ER-1-PWD","ER-2-PWD",..: NA 3 3 4 3 3 3 3 3 3 ...  
## $ Finished Quantity : Factor w/ 58 levels "1","2","3","4",..: 20 2 2 3 1 2 1 11 4 8 ...  
## $ Ending Date Modifier : Factor w/ 4 levels "-1","0","1","2": 2 2 2 2 2 2 2 2 2 2 ...  
## $ Destination : Factor w/ 7 levels "370RF","ASY-CZB-150",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Count of Components : Factor w/ 159 levels "1","2","3","4",..: 10 11 11 11 11 11 11 11 11 11 ...  
## $ Sum of Component Quantity: Factor w/ 556 levels "0","1","2","4",..: 69 56 56 56 56 56 62 62 62 62 ...  
## $ Routing Steps : Factor w/ 12 levels "1","2","3","4",..: 2 4 4 4 4 4 4 4 4 4 ...  
## $ DaysPastDue : Factor w/ 38 levels "-9","-6","-5",..: 32 10 10 9 10 10 10 12 10 10 ...  
## $ PastDue : Factor w/ 2 levels "FALSE","TRUE": 2 2 2 2 2 2 2 2 2 2 ...  
## $ PastDueCategory : Factor w/ 5 levels "-3 to -1","-4 or less",..: 5 4 4 4 4 4 4 5 4 4 ...  
## $ wd : Factor w/ 5 levels "Friday","Monday",..: 3 1 1 3 1 1 1 1 1 1 ...  
## $ wdFinish : Factor w/ 7 levels "Friday","Monday",..: 7 6 6 1 6 6 6 5 6 6 ...

str(ModData)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 48503 obs. of 25 variables:  
## $ Prod\_ Order No\_ : chr "M6122174" "M6142336" "M6142337" "M6142338" ...  
## $ Item No\_ : chr "QS1191944" "1191947" "1191947" "1191973" ...  
## $ Sales Order No\_ : num NA 1177605 1177605 1178000 1177605 ...  
## $ Segment : chr NA "1" "1" "1" ...  
## $ Description : chr "SUPPORT,ACC2-RF-13BF SAMPLE" "SUPPORT,ACC2-RF-28BF" "SUPPORT,ACC2-RF-28BF" "SUPPORT,ACC2-RF-16BF" ...  
## $ Item Category Code : chr "MI" "PRODUCT" "PRODUCT" "PRODUCT" ...  
## $ Final Output Location : chr "A023E" "SHIP1210" "SHIP1210" "SHIP1210" ...  
## $ Inventory Posting Group : chr "INV" "PRODUCT" "PRODUCT" "PRODUCT" ...  
## $ Planner Code : chr "B10" "5000" "5000" "5000" ...  
## $ Job Color : chr NA "ER-3-PWD" "ER-3-PWD" "ER10-PWD" ...  
## $ Finished Quantity : num 20 2 2 3 1 2 1 11 4 8 ...  
## $ Ending Date : POSIXct, format: "2018-04-05" "2018-06-08" ...  
## $ Ending Date Modifier : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Modified Ending Date : POSIXct, format: "2018-04-05" "2018-06-08" ...  
## $ Due Date : POSIXct, format: "2018-04-09" "2018-06-12" ...  
## $ Destination : chr "370RF" "370RF" "370RF" "370RF" ...  
## $ Finish Date : POSIXct, format: "2018-05-09" "2018-06-12" ...  
## $ Count of Components : num 10 11 11 11 11 11 11 11 11 11 ...  
## $ Sum of Component Quantity: num 48 41 41 41 41 41 43 43 43 43 ...  
## $ Routing Steps : num 2 4 4 4 4 4 4 4 4 4 ...  
## $ DaysPastDue : num 24 2 2 1 2 2 2 4 2 2 ...  
## $ PastDue : logi TRUE TRUE TRUE TRUE TRUE TRUE ...  
## $ PastDueCategory : chr "4 or greater" "1 to 3" "1 to 3" "1 to 3" ...  
## $ wd : chr "Thursday" "Friday" "Friday" "Thursday" ...  
## $ wdFinish : chr "Wednesday" "Tuesday" "Tuesday" "Friday" ...

Perhaps there is something that I am missing in this method, but I cannot determine the underlying issue, nor how to replace or remove enough of the N/A values in order to allow the algorithms to run, while still having the values for the process to run on top of (“Variant Code” removal apparently caused other issues).

## Deep Learning

Considering the inability of Random Forest models to process, Deep Learning became the next option I wished to test. In order to give it the best change possible (i.e. less potential mistakes on my end), the model was fed the *ModData* set. This set contains 48503 records with 25 variables, not including “Variant Code” or a couple others as seen above. The data was split above so we will be reusing the training and testing sets.

library("neuralnet")

##   
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':  
##   
## compute

n <- names(FullTrain)  
f <- as.formula(paste("PastDue ~", paste(n[!n %in% "PastDue"], collapse = " + ")))

## Error in parse(text = x, keep.source = FALSE): <text>:1:17: unexpected symbol  
## 1: PastDue ~ Prod\_ Order  
## ^

FullNeural <- neuralnet(f, data = FullTrain, hidden = c(10,5), linear.output = FALSE)

## Error in neuralnet(f, data = FullTrain, hidden = c(10, 5), linear.output = FALSE): object 'f' not found

Unfortunately we are foiled yet again. It appears that the formula can only be passed as a string and cannot use the “Variable ~.” method as we are used to. However, in creating the formula string we are unable to use the text fields found in the data frame.

# Conclusion

Considering I have been foiled at each turn, I think it is best to admit defeat on this data set and attempt another. Although the data is real, it was cleaned, and all instructions were followed for each function and package, it just appears that there are some fundamental flaws in the original data set that I am unable to see.

Although this conclusion is not the one I wished to reach, I think it is nevertheless valuable for several reasons:

1. The quality of data is of utmost importance. Null values and replacements are difficult to guage and must be handled correctly.
2. Some analysis will not produce actionable results. Just as in other endeavors, experience is the ultimate teacher, and sometimes failure is inevitable.
3. I need to learn much more about the data cleansing aspect of modeling as it appears to be the reason for this project’s difficulty.
4. No matter how good the model or the method, if the data is not ready, then you will learn nothing.

Unfortunately, after sinking around 25-30 hours trying to troubleshoot and resolve, it is time to call this analysis quits.