Historical MAL dataset cleanup

Edward Yu

December 07, 2018

Contents

Introduction

From Marek:

If you are interested I have a small project for R, which is very useful. It has to do with history records of MAL. Here is some basic info:

- Goal: Clean up and consolidate dataset to enable easy searching of past melt records
- Tasks:
  - Mostly working with strings removing duplicates (E. Yu, YU, Yu Edward . . .)
  - Removing empty records
  - Missing data
  - Multiple variables in one column

There might be other things to do but I have not spent much time looking at the dataset. We could also pull some basic stats on usage, costs, repeats etc. I don't know your skill level, but it is relatively simple project and I am estimating it would take me about 8 hrs of work. Actual coding, if you know what to use, could be done in less than 1 hour but that requires proficiency in typing and in R.

I just noticed that sand for this year should have all been W410, excel incremented the name by 1 each time. I think I might be adding information about individual tests from this year incrementally as it comes in and since it is only several rows, perhaps you can delete the entire set of rows from this year, if that makes things easier on your end.

Load & peak data

Import data

```
x <- read_csv("data/History.csv")
glimpse(x)
```

```
## Observations: 3,629
## Variables: 19
## $ Request              <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13...
## $ ID                   <dbl> NA, NA, NA, NA, 12201, 12194, NA, NA, NA,...
## $ `Date  Poured`       <chr> "1/5/1999", "1/6/1999", "1/7/1999", "1/8/...
## $ `Date Received`      <chr> "1/4/1999", "1/4/1999", "1/4/1999", "1/4/...
## $ `Date Completed`     <chr> "1/13/1999", "1/13/1999", "1/13/1999", "1...
## $ `Requested by`       <chr> "18", "CLINGERMAN,M.", "CLINGERMAN, M.", ...
## $ `Customer Name`      <chr> "TS&D", "TS&D", "TS&D", "TS&D", "BRILLION...
## $ `Product Tested`     <chr> "ISOCURE", "ISOCURE", "ISOCURE", "ISOCURE...
## $ `Casting Type`       <chr> "STEPCONE", "STEPCONE", "EROSION WEDGE", ...
## $ `Number of castings` <dbl> 8, 8, 8, 8, 3, 1, 8, 10, 8, 4, 10, 8, 2, ...
## $ Alloy                <chr> "GRAY IRON", "GRAY IRON", "GRAY IRON", "G...
## $ lbs                  <dbl> 250, 250, 600, 600, 90, 90, 160, 30, 20, ...
## $ `Sand type`          <chr> "TECHNISAND 1L-5W", "TECHNISAND 1L-5W", "...
## $ `Amount used`        <dbl> 840, 840, 1680, 1680, 270, 210, 640, 240,...
## $ `Total hours`        <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
## $ `Total Cost`         <dbl> 1300, 1300, 2210, 2210, 862, 715, 2080, 8...
## $ `Furnace Cycle`      <chr> "W68", "W69", "W70, W71", "W72, W73", "W7...
## $ `Notes ML`           <chr> "TEST NEW BASE RESIN WITH STEPCONE CASTIN...
## $ `Special Projects`   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
```

Check levels

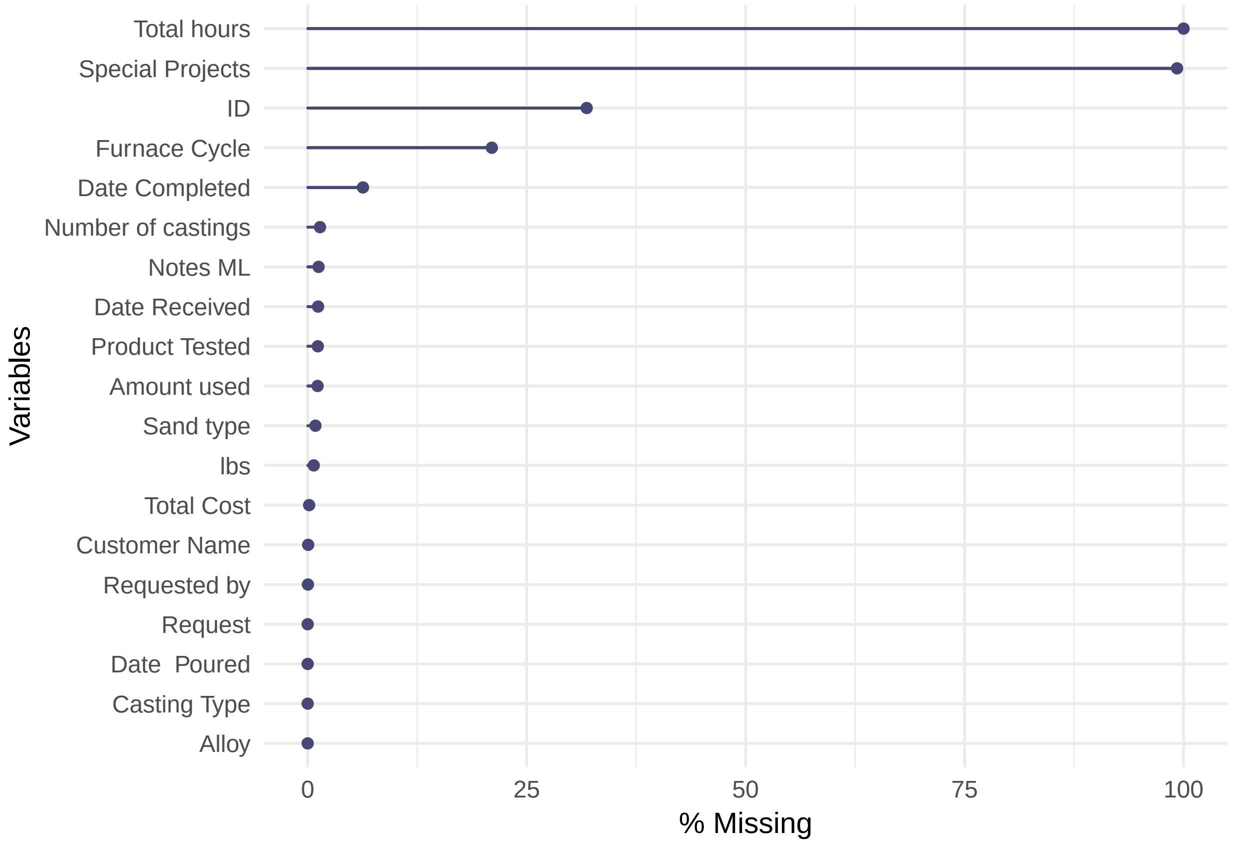
```
# create function for later use
get_levels <- function(df, col){
  x.levels <- cbind(colnames(df),
                    (as.data.frame(sapply(df,function(x) length(unique(x)))))
  )
  colnames(x.levels) <- c("var","levels")
  row.names(x.levels) <- NULL
  levels <- x.levels[order(-x.levels[,2]),]
  return(levels[col,])
}
get_levels(x)
```

	var	levels
1	Request	3628
18	Notes ML	2902
3	Date Poured	2746
5	Date Completed	1989

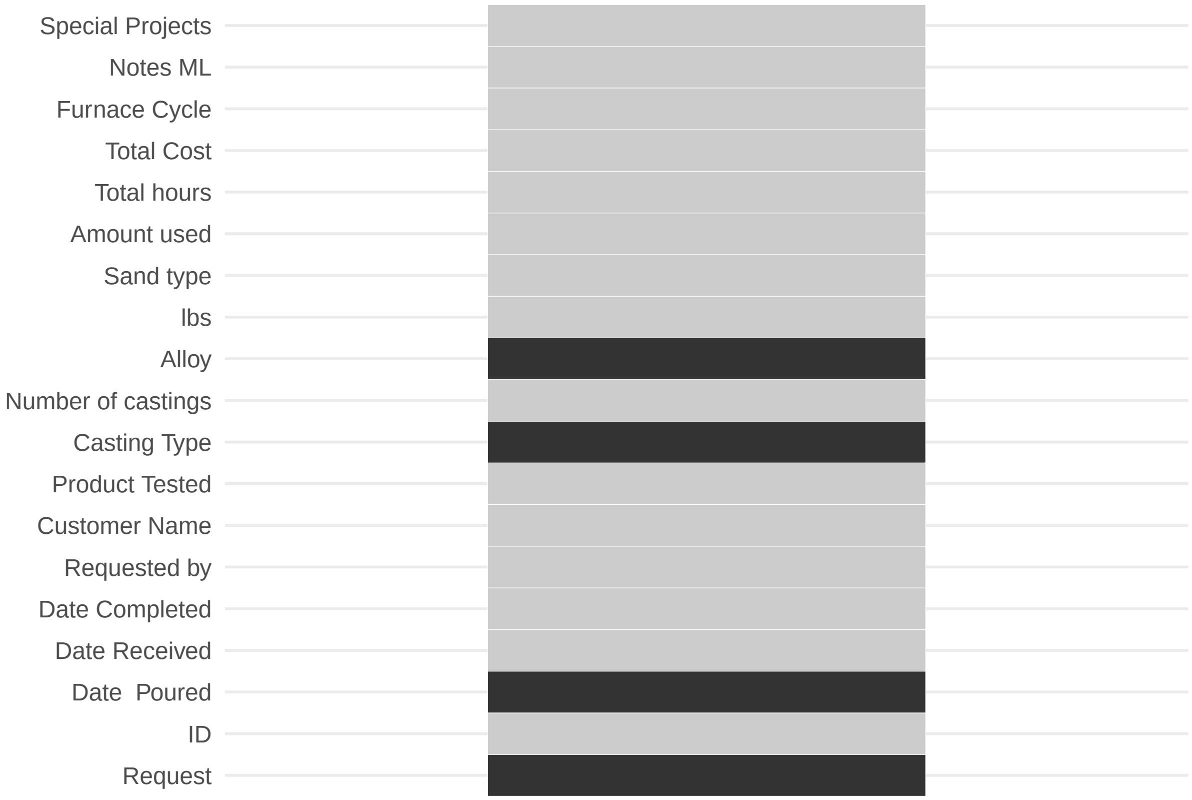
	var	levels
4	Date Received	1962
17	Furnace Cycle	1718
2	ID	1623
7	Customer Name	373
9	Casting Type	273
6	Requested by	259
16	Total Cost	211
8	Product Tested	175
14	Amount used	125
13	Sand type	113
12	lbs	87
11	Alloy	60
10	Number of castings	42
19	Special Projects	19
15	Total hours	1

Peak missing values

```r
gg_miss_var(x, show_pct = T)
```



```
gg_miss_which(x)
```



Outline of actions to take

Rename variables to be all lowercase with no spaces. Seems the most important variables are casting type and alloy type, as these are the only with zero missing values.

- **Request**: should have 3,629 levels
- **ID**: not utilized in recent pours, delete
- **Date received**: convert to date format, fill missing values, for some reason there are less dates received than dates completed
- **Date poured**: convert to date format, fill missing values
- **Date completed**: convert to date format, fill missing values, perhaps create new column calculating days to complete from date received/completed
- **Notes ML**: NA
- **Special projects**: most values are missing, unsure of importance of this field, should likely merge with comments or remove entirely
- **Requested by**: fill missing values, will require some renaming/matching
- **Customer name**: fill missing values, will require some renaming/matching
- **Product tested**: fill missing values, will require some renaming/matching
- **Alloy**: NA

- **Casting type**: NA
- **Number of castings**: some NA values, fill in with rounded averages
- **lbs**: lbs of metal used, could be calculated based on values, fill missing values
- **Sand type**: fill missing values, will require some renaming/matching
- **Amount used**: sand? unsure what amount this is talking about
- **Furnace cycle**: need to come up with new way to ID new lining and cycles
- **Total hours**: many NA values, should be calculated automatically based on number of castings, casting type, etc
- **Total cost**: fill missing values, perhaps determine how it is calculated to automate the calculation

We have many missing datapoints, fields that aren't inuitive, some useless fields, fields that need added, etc. We'll start with the most simple and move on.

Cleaning

Rename columns

Convert column names to lower case, replace spaces with periods.

```
names <- tolower(colnames(x))    # convert to lowercase
names <- gsub("  ", " ", names)  # remove double spaces
names <- gsub(" ", "\\.", names) # replace space with .
names[c(12,14)] <- c("alloy.lbs", "sand.lbs")
colnames(x) <- names
colnames(x)
```

```
##  [1] "request"           "id"                "date.poured"
##  [4] "date.received"     "date.completed"    "requested.by"
##  [7] "customer.name"     "product.tested"    "casting.type"
## [10] "number.of.castings" "alloy"            "alloy.lbs"
## [13] "sand.type"         "sand.lbs"          "total.hours"
## [16] "total.cost"        "furnace.cycle"     "notes.ml"
## [19] "special.projects"
```

$request

There is a duplicate entry somewhere based on number of unique levels versus number of rows.

```
which(duplicated(x$request)==TRUE)
```

```
## [1] 3611
```

```
as.data.frame(t(x[3609:3611,]))
```

	V1	V2	V3
request	3609	3610	3610
id	NA	NA	NA
date.poured	9/13/2017	unknown	6/28/2018
date.received	NA	NA	NA
date.completed	NA	NA	NA
requested.by	VIVAS	unknown	unknown
customer.name	ASK	unknown	ASK

	V1	V2	V3
product.tested	COATINGS	unknown	unknown
casting.type	STEP CONES	unknown	unknown
number.of.castings	NA	NA	NA
alloy	GRAY IRON	unknown	Aluminum
alloy.lbs	NA	NA	NA
sand.type	NA	unknown	W410
sand.lbs	NA	NA	NA
total.hours	NA	NA	NA
total.cost	0	NA	0
furnace.cycle	S1	S2	S3
notes.ml	NA	NA	NA
special.projects	NA	NA	NA

The first entry appears to have been made in error until we see the furnace cycle was incremented. Probably shouldn't remove, will simply re-assign all request variables to equal row numbers.

```
x <- x %>%
  mutate(request = seq(1:nrow(x)))
get_levels(x, 1)
```

var	levels
request	3629

$id

Delete useless column.

```
x <- x %>%
  select(-id)
```

Convert dates, add lead time

Convert char to date values.

```
x <- x %>%
  mutate(date.poured = as.Date(x$date.poured, "%m/%d/%Y")) %>%
  mutate(date.received = as.Date(x$date.received, "%m/%d/%Y")) %>%
  mutate(date.completed = as.Date(x$date.completed, "%m/%d/%Y"))

summary(x[c(3,2,4)])
```

```
##  date.received         date.poured         date.completed
##  Min.   :1990-10-20   Min.   :1995-08-04   Min.   :1999-01-13
##  1st Qu.:2001-07-27   1st Qu.:2001-08-08   1st Qu.:2001-10-31
##  Median :2005-06-08   Median :2005-07-11   Median :2005-09-23
##  Mean   :2006-02-25   Mean   :2006-03-03   Mean   :2006-04-11
##  3rd Qu.:2010-03-31   3rd Qu.:2010-05-18   3rd Qu.:2010-06-11
##  Max.   :2513-11-01   Max.   :2078-08-24   Max.   :2106-04-05
##  NA's   :43           NA's   :1            NA's   :229
```

With these dates we can now determine a few useful values:

- Preprocessing time: date poured - date recieved

- Postprocessing time: date complete - date poured
- Lead time: date complete - date received

Of course, we need to fix the erroneous entries that are pushing our `Max` values all the way up to the year 2513.

Fix far future dates

We manually fix the handful of dates with typos.

```
wrong.dates <- x %>%
  filter(date.received  > "2020-01-01" |
         date.poured    > "2020-01-01" |
         date.completed > "2020-01-01")
as.data.frame(wrong.dates)[,c(1,3,2,4)]
```

request	date.received	date.poured	date.completed
1103	2002-04-12	2020-04-30	2002-05-03
1198	2002-07-02	2002-08-02	2020-08-22
1889	2021-11-17	2005-11-21	2005-11-23
2735	2010-06-04	2020-06-10	2010-06-11
2740	2021-06-15	2010-06-21	2010-06-22
3106	2021-03-20	2012-03-21	2012-03-22
3149	2023-06-19	2012-06-26	2012-06-27
3341	2513-11-01	2013-11-22	2013-12-02
3582	2016-03-16	2016-04-01	2106-04-05
3606	NA	2078-08-24	NA

```
# manually fix
x$date.poured[1103]     <- as.Date("2002-04-30")
x$date.completed[1198] <- as.Date("2002-08-22")
x$date.received[1889]  <- as.Date("2005-11-17")
x$date.poured[2735]     <- as.Date("2002-06-10")
x$date.received[2740]  <- as.Date("2010-06-15")
x$date.received[3106]  <- as.Date("2012-03-20")
x$date.received[3149]  <- as.Date("2012-06-19")
x$date.received[3341]  <- as.Date("2013-11-01")
x$date.completed[3582] <- as.Date("2016-04-05")
x$date.poured[3606]     <- as.Date("2017-08-24")

# dates now seem to be in a normal range
summary(x[c(3,2,4)])
```

```
##  date.received         date.poured          date.completed
##  Min.   :1990-10-20   Min.   :1995-08-04   Min.   :1999-01-13
##  1st Qu.:2001-07-27   1st Qu.:2001-08-08   1st Qu.:2001-10-31
##  Median :2005-06-08   Median :2005-07-07   Median :2005-09-20
##  Mean   :2005-12-31   Mean   :2006-02-21   Mean   :2006-03-30
##  3rd Qu.:2010-03-30   3rd Qu.:2010-05-17   3rd Qu.:2010-06-10
##  Max.   :2016-04-25   Max.   :2018-11-28   Max.   :2016-05-02
##  NA's   :43           NA's   :1            NA's   :229
```

Calculate lead times

Now that are values are all within somewhat normal ranges, detecting further errors will require calculating the differences in dates. For example if `date.received` has a later date than `date.completed` we will see a negative value in our new `lead.time` variable.

Using the function on our current data shows negative values in all new variables as well as some unrealistically large `Max` values.

```
## create function so that results of editing can be seen quickly
calc_lead <- function(){
  preprocessing.time  <- as.numeric(x$date.poured-x$date.received)
  postprocessing.time <- as.numeric(x$date.completed-x$date.poured)
  lead.time           <- preprocessing.time + postprocessing.time
  x.temp <- as_tibble(cbind(x,
                            preprocessing.time,
                            postprocessing.time,
                            lead.time))
  return(x.temp)
}

summary(calc_lead()[c(19:21)])
```

```
##  preprocessing.time postprocessing.time   lead.time
##  Min.   :-3288.00   Min.   :-4014.000   Min.   :-4011.000
##  1st Qu.:    3.00   1st Qu.:    1.000   1st Qu.:    5.000
##  Median :    5.00   Median :    2.000   Median :    8.000
##  Mean   :    5.74   Mean   :    3.641   Mean   :    8.176
##  3rd Qu.:    8.00   3rd Qu.:    5.000   3rd Qu.:   14.000
##  Max.   : 3293.00   Max.   : 3288.000   Max.   : 2804.000
##  NA's   :43         NA's   :229         NA's   :232
```

Fix large processing values

We filter for values larger than 400 and find quite a few entries have simple typos. We correct the handful of errors by hand.

```
## fix large values
wrong.dates <- calc_lead() %>%
  filter(preprocessing.time > 400 |
           postprocessing.time > 400)
wrong.dates[c(1,3,2,4)]
```

request	date.received	date.poured	date.completed
310	1990-10-20	1999-10-21	NA
930	2001-08-24	2010-08-30	2001-09-04
1091	2002-04-09	2001-04-22	2002-07-12
1616	2000-04-13	2004-04-21	2004-04-26
1668	2004-08-04	1995-08-04	2004-08-04
1877	2005-10-31	2003-11-01	2005-11-14
2229	2000-01-07	2007-09-11	2007-09-11
2735	2010-06-04	2002-06-10	2010-06-11
3133	2012-05-16	2010-05-17	2012-05-18

```
# manually fix
x$date.received[310]  <- as.Date("1999-10-20")
x$date.poured[930]    <- as.Date("2001-08-30")
x$date.poured[1091]   <- as.Date("2002-04-22")
x$date.received[1616] <- as.Date("2004-04-13")
x$date.poured[1668]   <- as.Date("2004-08-04")
x$date.poured[1877]   <- as.Date("2005-11-01")
x$date.received[2229] <- as.Date("2007-01-07")
x$date.poured[2735]   <- as.Date("2010-06-10")
x$date.poured[3133]   <- as.Date("2012-05-17")

# max values look better now
summary(calc_lead()[c(19:21)])
```

```
##  preprocessing.time  postprocessing.time   lead.time
##  Min.   :-3283.000   Min.   :-4014.000   Min.   :-4011.000
##  1st Qu.:    3.000   1st Qu.:    1.000   1st Qu.:    5.000
##  Median :    5.000   Median :    2.000   Median :    8.000
##  Mean   :    5.028   Mean   :    2.244   Mean   :    6.993
##  3rd Qu.:    8.000   3rd Qu.:    5.000   3rd Qu.:   14.000
##  Max.   :  374.000   Max.   :  383.000   Max.   :  434.000
##  NA's   :43          NA's   :229         NA's   :232
```

Fix negative processing values

This occurs when dates are not in proper chronology: date.received < date.poured < date.completed. We can fix this by filtering for dates that do not meet this criteria and adjusting them based on available dates and median values for preprocessing/postprocessing/lead times.

In this case we have 244 rows of incorrectly ordered data, definitely not going to do this manually. This time we'll impute the missing data by taking the median values of the correct data. First part of the code fixes NA values.

```
# fix negatives
# must follow: received < poured < completed
wrong.dates2 <- calc_lead() %>%
  filter(preprocessing.time < 0 | postprocessing.time < 0)
wrong.dates2[c(1,3,2,4)]
```

request	date.received	date.poured	date.completed
13	1999-01-15	1999-01-19	1999-01-17
40	1999-02-27	1999-02-23	1999-02-24
41	1999-02-27	1999-02-24	1999-02-25
191	1999-07-06	1999-07-08	1999-06-29
314	1999-10-26	1999-10-27	1999-10-26
398	2000-12-22	2000-01-26	NA
424	2000-03-10	2000-02-16	2000-02-24
448	2000-03-06	2000-03-10	2000-03-06
457	2000-03-17	2000-03-20	2000-03-18
458	2000-03-16	2000-03-21	2000-03-17
483	2000-04-11	2000-04-12	2000-04-11
492	2000-04-19	2000-04-20	2000-04-19
609	2000-08-11	2000-08-16	2000-08-11
673	2000-11-09	2000-11-14	2000-11-13
735	2001-01-26	2000-01-31	2001-01-31

request	date.received	date.poured	date.completed
764	2001-03-23	2001-02-28	NA
870	2010-07-07	2001-07-11	2001-07-12
929	2010-08-22	2001-08-29	2001-08-30
953	2001-09-28	2001-10-01	2001-09-28
972	2001-10-18	2001-10-25	2001-10-19
1027	2002-01-22	2002-01-29	2002-01-23
1029	2002-01-24	2002-01-30	2002-01-24
1155	2002-06-11	2003-06-20	2002-06-27
1164	2002-06-20	2002-06-27	2002-06-20
1212	2002-08-20	2001-08-22	2002-09-09
1287	2002-12-18	2002-12-19	2002-01-03
1288	2002-12-18	2002-12-19	2002-12-18
1289	2002-12-18	2002-12-01	2003-01-02
1307	2003-01-16	2003-01-27	2003-01-03
1556	2004-01-09	2004-01-15	2003-01-19
1600	2004-03-07	2004-03-17	2004-03-10
1610	2004-04-12	2004-04-14	2004-04-13
1619	2004-04-19	2004-04-23	2004-02-28
1628	2004-05-10	2004-05-11	2004-05-10
1708	2004-10-22	2004-10-27	2004-10-25
1709	2004-10-22	2004-10-27	2004-10-25
1741	2005-11-18	2005-01-20	2005-01-24
1821	2005-07-12	2005-07-15	2003-07-18
2025	2006-07-20	2006-07-25	2006-07-21
2237	2007-09-14	2007-09-26	2006-09-27
2283	2008-02-21	2008-01-23	2008-01-24
2285	2008-01-23	2008-01-28	2008-01-25
2304	2008-03-29	2008-03-03	NA
2351	2008-05-30	2008-05-13	2008-05-14
2408	2008-09-05	2008-09-09	2000-09-08
2496	2009-02-13	2009-02-17	2009-02-10
2547	2009-06-03	2009-06-09	2009-05-10
2583	2009-08-17	2009-08-31	2009-08-03
2662	2010-01-26	2010-02-02	2010-01-03
2812	2010-10-22	2010-10-26	2010-10-17
2962	2011-08-09	2011-08-11	2011-06-12
3058	2012-01-03	2012-01-06	2001-01-09
3064	2011-12-21	2011-01-16	2012-01-17
3065	2012-01-12	2012-11-17	2012-01-18
3068	2012-01-12	2012-01-19	2012-01-13
3070	2012-01-19	2012-01-24	2012-01-15
3072	2012-02-25	2012-01-26	2012-01-26
3135	2012-05-27	2012-05-22	2012-05-24
3136	2012-05-27	2012-05-23	2012-05-24
3157	2012-07-10	2012-07-23	2012-07-18
3159	2012-07-27	2012-07-26	2012-07-27
3160	2012-07-27	2012-07-26	2012-07-27
3192	2012-10-11	2012-10-15	2011-10-16
3197	2012-12-25	2012-10-31	2012-11-01
3210	2012-12-05	2012-12-07	2012-12-06
3237	2013-03-18	2013-03-04	2013-03-05
3259	2013-05-28	2013-05-21	2013-05-23

request	date.received	date.poured	date.completed
3260	2013-06-10	2013-05-22	2013-05-23
3261	2013-05-29	2013-05-22	2013-05-24
3263	2013-05-29	2013-05-24	2013-05-29
3265	2013-06-04	2013-06-03	2013-06-04
3266	2013-06-04	2013-06-03	2013-06-04
3268	2013-06-13	2013-06-05	2013-06-11
3270	2013-06-14	2013-06-07	2013-06-18
3271	2013-06-12	2013-06-10	2013-06-11
3272	2013-06-14	2013-06-10	2013-06-20
3273	2013-06-27	2013-06-12	2013-06-18
3275	2013-06-27	2013-06-19	2013-06-20
3276	2013-07-03	2013-06-19	2013-06-20
3277	2013-07-03	2013-06-24	2013-06-25
3278	2013-07-03	2013-06-24	2013-06-25
3279	2013-06-27	2013-06-26	2013-06-27
3280	2013-07-03	2013-06-26	2013-06-28
3284	2013-07-19	2013-07-12	2013-07-15
3285	2013-08-01	2013-07-16	2013-07-17
3286	2013-08-06	2013-07-18	2013-07-19
3288	2013-07-31	2013-07-22	2013-07-23
3289	2013-07-24	2013-07-22	2013-07-26
3291	2013-07-31	2013-07-25	2013-07-26
3292	2013-08-08	2013-08-01	2013-08-19
3293	2013-08-09	2013-08-01	2013-08-05
3295	2013-08-19	2013-08-08	2013-08-09
3296	2013-08-19	2013-08-06	2013-08-07
3298	2013-08-20	2013-08-14	2013-08-15
3301	2013-09-11	2013-08-22	2013-08-23
3303	2013-09-11	2013-08-28	2013-08-29
3304	2013-09-13	2013-09-04	2013-09-05
3305	2013-09-11	2013-08-30	2013-09-03
3306	2013-09-06	2013-08-30	2013-09-03
3307	2013-10-03	2013-09-18	2013-09-19
3308	2013-09-12	2013-09-06	2013-09-06
3310	2013-09-16	2013-09-13	2013-09-16
3314	2013-10-08	2013-09-23	2013-09-24
3323	2013-10-30	2013-10-11	2013-10-14
3325	2013-10-30	2013-10-11	2013-10-14
3326	2013-11-04	2013-10-15	2013-10-16
3327	2013-11-04	2013-10-17	2013-10-18
3328	2013-11-04	2013-10-18	2013-10-22
3329	2013-11-04	2013-10-23	2013-10-24
3330	2013-11-08	2013-10-29	2013-10-29
3331	2013-11-08	2013-10-30	2013-10-30
3333	2013-11-20	2013-11-01	2013-11-04
3334	2013-11-25	2013-11-07	2013-11-07
3336	2013-11-27	2013-11-14	2013-11-15
3337	2013-11-27	2013-11-14	2013-11-15
3340	2013-11-22	2013-11-19	2013-11-22
3342	2013-12-02	2013-11-22	2013-12-03
3345	2013-11-08	2013-11-05	2013-11-08
3347	2013-12-20	2013-12-16	2013-12-18

request	date.received	date.poured	date.completed
3348	2013-12-25	2013-12-20	2013-12-26
3349	2013-12-25	2013-12-11	2013-12-12
3350	2013-12-12	2013-12-11	2013-12-13
3351	2013-12-12	2013-12-12	2012-12-13
3353	2014-01-08	2013-12-20	2013-12-26
3354	2014-01-10	2014-01-08	2014-01-14
3359	2014-01-27	2014-01-14	2014-01-15
3361	2014-01-29	2014-01-21	2014-01-23
3362	2014-01-29	2014-01-22	2014-01-23
3363	2014-01-31	2014-01-22	2014-01-23
3364	2014-02-12	2014-01-28	2014-01-31
3366	2014-01-31	2014-01-22	2014-02-05
3368	2014-02-10	2014-02-07	2014-02-07
3369	2014-02-26	2014-02-11	2014-02-12
3370	2014-02-25	2014-02-17	2014-02-19
3371	2014-02-25	2014-02-17	2014-02-19
3372	2014-02-20	2014-02-18	2014-02-22
3373	2014-02-21	2014-02-20	2014-02-24
3374	2014-02-26	2014-02-25	2014-02-27
3375	2014-03-14	2014-02-27	2014-02-28
3376	2014-03-07	2014-02-26	2014-02-28
3378	2014-03-07	2014-03-05	2014-03-06
3383	2014-03-21	2014-03-14	2014-03-18
3385	2014-04-08	2014-03-24	2014-03-25
3386	2014-04-08	2014-03-24	2014-03-25
3387	2014-04-08	2014-03-31	2014-03-31
3388	2014-04-01	2014-03-26	2014-03-26
3391	2014-04-18	2014-04-09	2014-04-15
3393	2014-04-30	2014-04-25	2014-04-30
3394	2014-05-02	2014-04-25	2014-05-06
3398	2014-05-14	2014-05-02	2014-05-05
3402	2014-05-21	2014-05-16	2014-05-16
3403	2014-05-23	2014-05-20	2014-05-22
3404	2014-05-21	2014-05-16	2014-05-16
3406	2014-06-02	2014-05-23	2014-05-27
3407	2014-06-03	2014-05-29	2014-05-30
3408	2014-06-16	2014-05-29	2014-05-29
3409	2014-06-11	2014-06-03	2014-06-04
3411	2014-06-17	2014-06-04	2014-06-06
3413	2014-06-16	2014-06-12	2014-06-12
3414	2014-06-30	2014-06-17	2014-06-18
3415	2014-07-18	2014-06-20	2014-06-23
3416	2014-07-18	2014-06-24	2014-06-25
3417	2014-07-18	2014-06-26	2014-06-27
3420	2014-07-10	2014-07-01	2014-07-02
3421	2014-07-10	2014-07-07	2014-07-08
3422	2014-07-11	2014-07-09	2014-07-10
3424	2014-07-18	2014-07-11	2014-07-14
3425	2014-07-28	2014-07-24	2014-07-25
3426	2014-07-23	2014-07-18	2014-07-21
3428	2014-08-05	2014-08-04	2014-08-05
3429	2014-08-08	2014-07-31	2014-07-31

request	date.received	date.poured	date.completed
3430	2014-08-01	2014-07-31	2014-07-31
3433	2014-08-14	2014-08-12	2014-08-13
3435	2014-08-28	2014-08-27	2014-08-27
3436	2014-09-19	2014-09-04	2014-09-05
3437	2014-09-26	2014-09-08	2014-09-10
3438	2014-09-15	2014-09-11	2014-09-12
3440	2014-10-15	2014-09-23	2014-09-23
3441	2014-10-15	2014-09-25	2014-09-29
3442	2014-10-10	2014-09-30	2014-09-30
3443	2014-10-10	2014-09-30	2014-09-30
3445	2014-10-10	2014-10-08	2014-10-09
3446	2014-10-10	2014-10-09	2014-10-10
3447	2014-10-17	2014-10-16	2014-10-17
3448	2014-10-22	2014-10-14	2014-10-15
3449	2014-10-22	2014-10-15	2014-10-15
3450	2014-10-24	2014-10-23	2014-10-23
3451	2014-10-31	2014-10-28	2014-10-28
3452	2014-10-31	2014-10-30	2014-10-30
3453	2014-11-15	2014-11-11	2014-11-11
3454	2014-11-21	2014-11-14	2014-11-14
3455	2014-11-15	2014-11-14	2014-11-17
3456	2014-12-09	2014-12-03	2014-12-05
3457	2014-12-05	2014-11-26	2014-11-26
3459	2014-12-12	2014-12-09	2014-12-10
3461	2015-01-06	2014-12-18	2015-01-08
3463	2015-01-19	2015-01-14	2015-01-19
3464	2015-02-03	2015-01-20	2015-01-21
3465	2015-01-27	2015-01-21	2015-01-22
3466	2015-02-09	2015-01-27	2015-02-06
3467	2015-02-06	2015-02-04	2015-02-05
3473	2015-03-05	2015-02-23	2015-02-25
3474	2015-03-02	2015-02-19	2015-02-25
3476	2015-03-20	2015-03-11	2015-03-18
3477	2015-03-15	2015-03-03	NA
3478	2015-03-15	2015-03-06	2015-03-17
3479	2015-03-15	2015-03-06	2015-03-05
3481	2015-04-08	2015-03-24	2015-03-27
3482	2015-04-09	2015-03-30	2015-03-31
3485	2015-04-27	2015-04-15	2015-04-16
3486	2015-04-20	2015-04-09	2015-04-17
3487	2015-04-20	2015-04-10	2015-04-17
3488	2015-05-04	2015-04-23	2015-04-24
3489	2015-04-23	2015-04-17	2015-04-23
3490	2015-05-01	2015-04-28	2015-04-28
3491	2015-05-05	2015-04-29	2015-04-30
3492	2015-05-04	2015-05-01	2015-05-04
3494	2015-05-14	2015-05-12	2015-05-20
3497	2015-05-26	2015-05-20	2015-05-21
3498	2015-05-26	2015-05-12	2015-05-15
3499	2015-05-20	2015-05-14	2015-05-15
3503	2015-05-22	2015-05-21	2015-05-22
3504	2015-06-05	2015-05-27	2015-05-28

request	date.received	date.poured	date.completed
3505	2015-06-10	2015-06-03	2015-06-06
3506	2015-06-10	2015-06-03	2015-06-06
3507	2015-06-12	2015-06-05	2015-06-12
3508	2015-06-10	2015-06-09	2015-06-10
3509	2015-06-26	2015-06-18	2015-06-19
3510	2015-07-06	2015-06-17	2015-06-18
3511	2015-07-06	2015-06-19	2015-06-22
3512	2015-06-18	2015-06-17	2015-06-18
3517	2015-07-15	2015-07-07	2015-07-08
3525	2015-08-06	2015-07-14	2015-07-28
3528	2015-07-31	2015-07-30	2015-07-31
3533	2015-08-07	2015-08-05	2015-08-05
3534	2015-08-26	2015-08-18	2015-08-21
3535	2015-08-26	2015-08-18	2015-08-20
3536	2015-08-26	2015-08-25	2015-08-27
3537	2015-09-02	2015-08-20	2015-08-20
3543	2015-10-08	2015-09-17	2015-09-18
3545	2015-10-05	2015-10-02	2015-10-05
3546	2015-10-26	2015-10-08	2015-10-13
3554	2015-12-04	2015-12-02	2015-12-03
3573	2016-02-24	2016-02-25	2016-02-24

```
# antijoin the incorrect data
x.anti <- anti_join(calc_lead(), wrong.dates2, c("request"))

# record median values for imputation
summary(x.anti[c(19:21)])
```

```
##  preprocessing.time postprocessing.time   lead.time
##  Min.   :  0.000    Min.   :  0.000     Min.   :  0.00
##  1st Qu.:  3.000    1st Qu.:  1.000     1st Qu.:  6.00
##  Median :  6.000    Median :  3.000     Median :  9.00
##  Mean   :  8.141    Mean   :  5.218     Mean   : 13.11
##  3rd Qu.:  8.000    3rd Qu.:  6.000     3rd Qu.: 14.00
##  Max.   :372.000    Max.   :308.000     Max.   :434.00
##  NA's   :43         NA's   :225         NA's   :228
```

```
# preprocessing.time postprocessing.time   lead.time
# Median :  6.000    Median :  3.000     Median :  9.00

# have NA values in date.completed
summary(wrong.dates2[c(3,2,4)])
```

```
##  date.received         date.poured          date.completed
##  Min.   :1999-01-15   Min.   :1999-01-19   Min.   :1999-01-17
##  1st Qu.:2012-07-27   1st Qu.:2012-09-24   1st Qu.:2012-10-07
##  Median :2013-12-25   Median :2013-12-18   Median :2013-12-26
##  Mean   :2012-03-08   Mean   :2012-01-30   Mean   :2012-01-29
##  3rd Qu.:2014-10-15   3rd Qu.:2014-10-10   3rd Qu.:2014-10-11
##  Max.   :2016-02-24   Max.   :2016-02-25   Max.   :2016-02-24
##                                            NA's   :4
```

```
# if NA change completed date to received + 9
for (i in 1:dim(wrong.dates2)[1]){
  if (is.na(wrong.dates2$date.completed[[i]]) == TRUE){
    wrong.dates2$date.completed[[i]] <- wrong.dates2$date.received[[i]] + 9
  }
}

# no more NA's
summary(wrong.dates2[c(3,2,4)])
```

```
##  date.received         date.poured          date.completed
##  Min.   :1999-01-15   Min.   :1999-01-19   Min.   :1999-01-17
##  1st Qu.:2012-07-27   1st Qu.:2012-09-24   1st Qu.:2012-07-24
##  Median :2013-12-25   Median :2013-12-18   Median :2013-12-22
##  Mean   :2012-03-08   Mean   :2012-01-30   Mean   :2011-12-27
##  3rd Qu.:2014-10-15   3rd Qu.:2014-10-10   3rd Qu.:2014-10-11
##  Max.   :2016-02-24   Max.   :2016-02-25   Max.   :2016-02-24
```

```
#
# now we fix errors in chronology causing negative time calculations
summary(wrong.dates2[c(19:21)])
```

```
##  preprocessing.time postprocessing.time   lead.time
##  Min.   :-3283.00   Min.   :-4014.00    Min.   :-4011.00
##  1st Qu.:  -13.00   1st Qu.:    0.00    1st Qu.:  -12.00
##  Median :   -7.00   Median :    1.00    Median :   -5.00
##  Mean   :  -37.62   Mean   :  -36.91    Mean   :  -73.52
##  3rd Qu.:   -1.00   3rd Qu.:    3.00    3rd Qu.:    0.00
##  Max.   :  374.00   Max.   :  383.00    Max.   :   27.00
##                     NA's   :4           NA's   :4
```

```
# start with received coming before poured
# wrong.dates2$date.received > wrong.dates2$date.poured
wrong.dates2[c(2,3,6,7),c(3,2,4)]
```

date.received	date.poured	date.completed
1999-02-27	1999-02-23	1999-02-24
1999-02-27	1999-02-24	1999-02-25
2000-12-22	2000-01-26	2000-12-31
2000-03-10	2000-02-16	2000-02-24

```
# then completed coming before poured
# wrong.dates2$date.poured > wrong.dates2$date.completed
# wrong.dates2$date.completed < wrong.dates2$date.poured
wrong.dates2[c(1,4,5,8),c(3,2,4)]
```

date.received	date.poured	date.completed
1999-01-15	1999-01-19	1999-01-17
1999-07-06	1999-07-08	1999-06-29
1999-10-26	1999-10-27	1999-10-26
2000-03-06	2000-03-10	2000-03-06

```
for (i in 1:dim(wrong.dates2)[1]){
  # preprocessing time = poured - received; median = 6
  if (wrong.dates2$date.received[[i]] > wrong.dates2$date.poured[[i]]){
    wrong.dates2$date.received[[i]] <- wrong.dates2$date.poured[[i]] - 6
  }
  # postprocessing time = completed - poured; median = 3
  if (wrong.dates2$date.completed[[i]] < wrong.dates2$date.poured[[i]]){
    wrong.dates2$date.completed[[i]] <- wrong.dates2$date.poured[[i]] + 3
  }
}

# confirm chronology
# wrong.dates2$date.received <= wrong.dates2$date.poured
# wrong.dates2$date.poured <= wrong.dates2$date.completed

# now that `wrong.dates2` has corrected values, merge with original df
counter=1
for (i in 1:nrow(x)){
  if (counter == nrow(wrong.dates)+1){break}
  if (x$request[[i]] == wrong.dates$request[[counter]]){
    x[i,c(2,3,4)] <- wrong.dates[counter,c(2,3,4)]
    counter=counter+1
  }
}

# date summary looks okay now... except for NA's
summary(calc_lead()[c(19:21)])
```

```
##  preprocessing.time postprocessing.time   lead.time
##  Min.   :-3288.00   Min.   :-4014.000   Min.   :-4011.000
##  1st Qu.:    3.00   1st Qu.:    1.000   1st Qu.:    5.000
##  Median :    5.00   Median :    2.000   Median :    8.000
##  Mean   :    5.74   Mean   :    3.641   Mean   :    8.176
##  3rd Qu.:    8.00   3rd Qu.:    5.000   3rd Qu.:   14.000
##  Max.   : 3293.00   Max.   : 3288.000   Max.   : 2804.000
##  NA's   :43         NA's   :229         NA's   :232
```

Fix NA values in dates

Our calculations have NA values which means our dates must have NA's. We'll check which date columns contain NA's and in a similar fashion to above will impute appropriate dates based on the calculated median values above.

Based on the `summary` output it looks like `date.poured` has only a single `NA` value. If we fill this date in by hand we can just calculate the other variables based on the `date.poured` value and our previously calculated median values of processing times.

```
# check NAs
summary(x)[,c(3,2,4)]
```

```
##  date.received         date.poured          date.completed
##  Min.   :1990-10-20   Min.   :1995-08-04   Min.   :1999-01-13
##  1st Qu.:2001-07-27   1st Qu.:2001-08-08   1st Qu.:2001-10-31
##  Median :2005-06-08   Median :2005-07-07   Median :2005-09-20
##  Mean   :2005-12-31   Mean   :2006-02-21   Mean   :2006-03-30
##  3rd Qu.:2010-03-30   3rd Qu.:2010-05-17   3rd Qu.:2010-06-10
##  Max.   :2016-04-25   Max.   :2018-11-28   Max.   :2016-05-02
##  NA's   :43           NA's   :1            NA's   :229
```

```
# fill NA values using date.poured to calculate received and completed
# guess on single NA
x %>% filter(is.na(date.poured))
```

request	date.poured	date.received	date.completed	requested.by	customer.name	product.tested	casting.type	n
3610	NA	NA	NA	unknown	unknown	unknown	unknown	

```
(x[3608:3612,])
```

request	date.poured	date.received	date.completed	requested.by	customer.name	product.tested	casting.type
3608	2017-09-08	NA	NA	NOCERA	ASK	NA	EROSION WEI
3609	2017-09-13	NA	NA	VIVAS	ASK	COATINGS	STEP CONES
3610	NA	NA	NA	unknown	unknown	unknown	unknown
3611	2018-06-28	NA	NA	unknown	ASK	unknown	unknown
3612	2018-06-28	NA	NA	Edward Yu	ASK	SLEEVES	Shrink Cubes

```
(x$date.poured[3611] - x$date.poured[3609]) / 2 # 144 days between dates
```

```
## Time difference of 144 days
```

```
# just assign the middle date
x$date.poured[3610] <- x$date.poured[3609] - 72

# now date.poured has no NA's and we can extrapolate from this
summary(x)[,c(3,2,4)]
```

```
##  date.received         date.poured          date.completed
##  Min.   :1990-10-20   Min.   :1995-08-04   Min.   :1999-01-13
##  1st Qu.:2001-07-27   1st Qu.:2001-08-08   1st Qu.:2001-10-31
##  Median :2005-06-08   Median :2005-07-08   Median :2005-09-20
##  Mean   :2005-12-31   Mean   :2006-02-22   Mean   :2006-03-30
##  3rd Qu.:2010-03-30   3rd Qu.:2010-05-17   3rd Qu.:2010-06-10
##  Max.   :2016-04-25   Max.   :2018-11-28   Max.   :2016-05-02
##  NA's   :43                                NA's   :229
```

```
# convert all NA date.received to date.poured-6
x.rec <- x %>%
  filter(is.na(date.received)) %>%
  mutate(date.received = date.poured - 6)

# merge back into original df
counter=1
for (i in 1:nrow(x)){
  if (x$request[[i]] == x.rec$request[[counter]]){
    x[i,c(2,3,4)] <- x.rec[counter,c(2,3,4)]
    counter=counter+1
  }
}
```

```
# convert all NA date.completed to date.poured+6
x.com <- x %>%
  filter(is.na(date.completed)) %>%
  mutate(date.completed = date.poured + 3)

# merge back into original df
counter=1
for (i in 1:nrow(x)){
  if (x$request[[i]] == x.com$request[[counter]]){
    x[i,c(2,3,4)] <- x.com[counter,c(2,3,4)]
    counter=counter+1
  }
}

# NOW we have zero NA values
summary(calc_lead()[c(19:21)])
```

```
##  preprocessing.time  postprocessing.time   lead.time
##  Min.   :-3288.000   Min.   :-4014.000   Min.   :-4011.000
##  1st Qu.:    3.000   1st Qu.:    1.000   1st Qu.:    5.000
##  Median :    5.000   Median :    3.000   Median :    8.000
##  Mean   :    5.743   Mean   :    3.601   Mean   :    9.344
##  3rd Qu.:    8.000   3rd Qu.:    5.000   3rd Qu.:   14.000
##  Max.   : 3293.000   Max.   : 3288.000   Max.   : 3291.000
```

```
# assign our new values to the df
x <- calc_lead()
```

$special.projects

Seems to be a redundant column when the $notes column would suffice. Check if values are stored in the column and concatenate them with the notes column.

```
# list non-NA values in special projects
x$special.projects[!is.na(x$special.projects)]
```

```
##  [1] "RESIN"
##  [2] "RESIN"
##  [3] "RESIN"
##  [4] "RESIN"
##  [5] "RESIN"
##  [6] "RESIN"
##  [7] "G"
##  [8] "RAW 3114."
##  [9] "ROTORS FOR GILSON"
## [10] "ROTORS FOR R.SHOWMAN"
## [11] "ROTORS FOR R. SHOWMAN-ADD. HEAD HEIGHT"
## [12] "ROTOR D.O.E. R. SHOWMAN"
## [13] "ROTOR D.O.E. R. SHOWMAN"
## [14] "COPPER STEPCONES-FALCON FOUNDRY"
## [15] "COPPER STEPCONES-FALCON FOUNDRY"
## [16] "COPPER STEPCONES-FALCON FOUNDRY"
## [17] "HEAT EXCHANGER"
## [18] "Belt Buckels"
## [19] "BELT BUCKELS"
## [20] "ADDITIONAL METAL FOR 300 LBS."
## [21] "\\"
## [22] "'"
## [23] "'"
## [24] "SEE IF .5MM SILICA BEADS HAVE A BENEFIT"
## [25] "COATINGS AND ADDITIVES, PRODUCT SUPPORT"
## [26] "Coatings and Additives, Product Support"
## [27] "double height risers for penetration"
```

```
# find rownums of non-NA vlaues
spec.rows <- which(!is.na(x$special.projects)==T)

# check notes.ml of same rownums
x$notes.ml[spec.rows]
```

```
##  [1] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB-EXP. BASE"
##  [2] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB"
##  [3] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB BASE"
##  [4] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB BASE"
##  [5] "COMPARE EROSION(REVERSE SPRUE) OF 450WB/850WB  W/EXP.BASE"
##  [6] "COMPARE EROSION(REVERSE SPRUE) OF 450WB/850WB EXP. BASE"
##  [7] "COMPARE SHAKEOUT OF EXISTING BINDER SYSTEMS FOR SALES MEETIN"
##  [8] "EVALUATE ISOSET BINDERS ON CUSTOMERS SAND TO REPLACE INSTAD"
##  [9] "MAKE AND SHIP ROTORS TO SHIN-KEN FOR D. GILSON"
## [10] "EVALUATE LARGER (50%) ROTOR CORE FOR VEINING"
## [11] "DOUBLE STACK ROTOR & ADD HEAD HEIGHT FOR VEINING."
## [12] "ROTOR D.O.E. TO EVALUATE VEINING, PENETRATION, &  SUR.FINISH"
## [13] "ROTOR D.O.E. TO EVALUATE VEINING, PENETRATION, & SUR. FINISH"
## [14] "FIND ONE BINDER SYSTEM TO WORK IN A COPPER BASE ALLOY"
## [15] "FIND ONE BINDER SYSTEM TO WORK IN A COPPER BASE ALLOY"
## [16] "FIND ONE BINDER SYSTEM TO WORK IN A COPPER BASE ALLOY"
## [17] "REPLACE HEAT EXCHANGER ON POWER TRACK"
## [18] "TEST DEFINITION OF BELT BUCKLE PATTERN"
## [19] "POUR BELT BUCKLES FOR SEMINAR GIFTS"
## [20] "DETERMINE AN ADDITIVE FOR USE WITH WARM BOX RESIN"
## [21] "INVESTIGATE NEW ISOCOTE SGW 32 VARIAITION-STEPCONE CST."
## [22] "EVALUATE DILATION, PENETRATION, VEINING, AND SURFACE FINISH"
## [23] "TEST MIRATEC TS 505 AND FORMULATION VARIATIONS"
## [24] "DETERMINE AFFECRS THAT ACTIVE CARBON PLAYS AS AN ADDITIVE"
## [25] "TEST MODIFICATIONS MADE TO MIRATEC 508"
## [26] "Test modifications made to MIRATEC MB 508"
## [27] NA
```

```
# concatenate the columns
x[spec.rows,] <- x[spec.rows,] %>%
  mutate(notes.ml = paste(notes.ml, special.projects, sep="--"))

# confirm
x$notes.ml[spec.rows]
```

```
##  [1] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB-EXP. BASE--RESIN"
##  [2] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB--RESIN"
##  [3] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB BASE--RESIN"
##  [4] "DETERMINE EROSION (REVERSE SPRUE) OF 450WB/850WB BASE--RESIN"
##  [5] "COMPARE EROSION(REVERSE SPRUE) OF 450WB/850WB  W/EXP.BASE--RESIN"
##  [6] "COMPARE EROSION(REVERSE SPRUE) OF 450WB/850WB EXP. BASE--RESIN"
##  [7] "COMPARE SHAKEOUT OF EXISTING BINDER SYSTEMS FOR SALES MEETIN--G"
##  [8] "EVALUATE ISOSET BINDERS ON CUSTOMERS SAND TO REPLACE INSTAD--RAW 3114."
##  [9] "MAKE AND SHIP ROTORS TO SHIN-KEN FOR D. GILSON--ROTORS FOR GILSON"
## [10] "EVALUATE LARGER (50%) ROTOR CORE FOR VEINING--ROTORS FOR R.SHOWMAN"
## [11] "DOUBLE STACK ROTOR & ADD HEAD HEIGHT FOR VEINING.--ROTORS FOR R. SHOWMAN-ADD. HEAD HEIGHT"
## [12] "ROTOR D.O.E. TO EVALUATE VEINING, PENETRATION, &  SUR.FINISH--ROTOR D.O.E. R. SHOWMAN"
## [13] "ROTOR D.O.E. TO EVALUATE VEINING, PENETRATION, & SUR. FINISH--ROTOR D.O.E. R. SHOWMAN"
## [14] "FIND ONE BINDER SYSTEM TO WORK IN A COPPER BASE ALLOY--COPPER STEPCONES-FALCON FOUNDRY"
## [15] "FIND ONE BINDER SYSTEM TO WORK IN A COPPER BASE ALLOY--COPPER STEPCONES-FALCON FOUNDRY"
## [16] "FIND ONE BINDER SYSTEM TO WORK IN A COPPER BASE ALLOY--COPPER STEPCONES-FALCON FOUNDRY"
## [17] "REPLACE HEAT EXCHANGER ON POWER TRACK--HEAT EXCHANGER"
## [18] "TEST DEFINITION OF BELT BUCKLE PATTERN--Belt Buckels"
## [19] "POUR BELT BUCKLES FOR SEMINAR GIFTS--BELT BUCKELS"
## [20] "DETERMINE AN ADDITIVE FOR USE WITH WARM BOX RESIN--ADDITIONAL METAL FOR 300 LBS."
## [21] "INVESTIGATE NEW ISOCOTE SGW 32 VARIAITION-STEPCONE CST.--\\"
## [22] "EVALUATE DILATION, PENETRATION, VEINING, AND SURFACE FINISH--'"
## [23] "TEST MIRATEC TS 505 AND FORMULATION VARIATIONS--'"
## [24] "DETERMINE AFFECRS THAT ACTIVE CARBON PLAYS AS AN ADDITIVE--SEE IF .5MM SILICA BEADS HAVE A BEN
## [25] "TEST MODIFICATIONS MADE TO MIRATEC 508--COATINGS AND ADDITIVES, PRODUCT SUPPORT"
## [26] "Test modifications made to MIRATEC MB 508--Coatings and Additives, Product Support"
## [27] "NA--double height risers for penetration"
```

$requested.by

Remove duplicate and misspelled names.

```r
# remove double spaces, commas, periods, caps, generate soundex
x <- x %>%
  mutate(requested.by = str_replace_all(requested.by, '\\  ', '')) %>%
  mutate(requested.by = str_replace_all(requested.by, '\\,', '')) %>%
  mutate(requested.by = str_replace_all(requested.by, '\\.', '')) %>%
  mutate(requested.by = str_to_lower(requested.by)) %>%
  mutate(sound = soundex(requested.by,clean=F))

# list unique sounds
unique(x$sound)
```

```
##   [1] ""     "C452" "B431" "A621" "T632" "F236" "S616" "S245" "S556" "S123"
##  [11] "A351" "G416" "C155" "M325" "H562" "W425" "L526" "F200" "M215" "C520"
##  [21] "I645" "S363" "G436" "H536" "W412" "D533" "S526" "S432" "B255" "K340"
##  [31] "C642" "T620" "K651" "H325" "L320" "H155" "F635" "H245" "C610" "G425"
##  [41] "H613" "L325" "S550" "D351" "A136" "L510" "N532" "N500" "T142" "R242"
##  [51] "S536" "S552" "F300" "W325" "T520" "H242" "D342" "S453" "J242" "A551"
##  [61] "M260" "H635" "M434" "A313" "K626" "M522" "M210" "D254" "W452" "A652"
##  [71] "C525" "M323" "M362" "C623" "W532" "H620" "D253" "G615" "W520" "Y552"
##  [81] "M610" "A161" "O416" "C600" "F535" "S632" "D525" "Y550" "S532" "R524"
##  [91] "T612" "C515" "J523" "D521" "D452" "C100" "L200" "Y625" "B522" "S530"
## [101] "A163" "J500" "P535" "C416" "C262" "H236" "A132" "A363" "C462" "N265"
## [111] "H655" "A353" "Y000" "M351" "W523" "B625" "A536" "H632" "H323" "K620"
## [121] "A624" "A431" "V121" NA     "X520" "P120" "E255" "V120" "N260" "K600"
## [131] "U525" "E363" "P411" "M363"
```

```r
# find problem rows: 1,540,3484
x %>%
  filter(sound == "" | is.na(sound))
```

request	date.poured	date.received	date.completed	requested.by	customer.name	product.tested	casting.type
1	1999-01-05	1999-01-04	1999-01-13	18	TS&D	ISOCURE	STEPCONE
540	2000-06-12	2000-06-08	2000-06-13	0000	CATERPILLAR	ISOSET	WARPAGE B
3484	2015-04-02	2015-04-02	2015-04-02	NA	ASK	FILTERS	LAUNDER

```r
# check surrounding rows
x[c(1:3,539:542,3483:3485),c(1,5,22)]
```

request	requested.by	sound
1	18	
2	clingermanm	C452
3	clingerman m	C452
539	showmanr	S556
540	0000	
541	fechter r	F236
542	skoglund m	S245
3483	clifford s	C416
3484	NA	NA
3485	yu e	Y000

```r
# replace NA/number values with next name in line
x$requested.by[c(1,540,3484)] <- x$requested.by[c(2,541,3485)]

# unique names and sounds
length(unique(x$requested.by)) # 204 unique names
```

```
## [1] 204
```

```r
length(unique(x$sound))         # 132 unique sounds
```

```
## [1] 134
```

We can see that we have quite a few unique names with less unique sounds. This might be because some names are misspelled and the misspellings don't change the sounds of the names. To address this we'll loop through each unique name, then take the sound of that name, grouped with all other names that have the same sound. Using this subset that all shares the same sound, we can sort the names in descending order, choosing the most popular and replacing all names by this most popular one. We'll see this reduces the amount of unique names fom 204 to 132: the same value of unique sounds.

```r
unique.names <- unique(x$sound)
# some names are misspelled but have the same sound
# replace any same-sounding with top used name
# replace unique name with most popular unique name filtered by sound
for (i in 1:length(unique.names)){
  # find most popular name of same sounding names
  replacement.name <- x %>%
    filter(sound == unique.names[[i]]) %>%
    group_by(requested.by) %>%
    summarise(count=n()) %>%
    arrange(desc(count))
  replacement.name <- replacement.name[[1]][1]
  # if unique.name == requestor$sound, replace with replacement.name
  x$requested.by[x$sound == unique.names[[i]]] <-
```

```
    replacement.name
}

# 129 unique names now
length(unique(x$requested.by))
```

```
## [1] 133
```

Though we're in a better place, we still see mispelled names in our data. Not much choice but to manually sift through and decide which names should be replaced by what. After manual replacement, our total unique names dips again to 106 from 129.

```
# but we see mispellings such as adamotvits or lowek
unique(x$requested.by)
```

```
##   [1] "clingermanm"     "clingerman m"    "belt p"
##   [4] "archibald j"     "twardowska h"    "fechter r"
##   [7] "shriver r"       "skoglund m"      "showman r"
##  [10] "szpak t"         "adamovits m"     "gilbreath t"
##  [13] "chapman c"       "madigan j"       "henry c"
##  [16] "wilson s"        "langer h"        "fox j"
##  [19] "moosavian t"     "chen j"          "ireland e"
##  [22] "sturtz g"        "gualtiere d"     "hendershot g"
##  [25] "wolfgram t"      "dando t"         "singh r"
##  [28] "schultz b"       "buchanan c"      "kathy lowe"
##  [31] "carlson g"       "torres h"        "kreinbrink j"
##  [34] "hutchings d"     "lute c"          "hoffman m"
##  [37] "fredendall a"    "hysell m"        "carr b"
##  [40] "gilson d"        "horvath l"       "lute/showman"
##  [43] "showman"         "dudenhofer r"    "aufderheide r"
##  [46] "lamb b"          "neu m-dgh"       "neu m"
##  [49] "toplikar e"      "rigel j"         "schneider j"
##  [52] "showman j"       "fitt w"          "woodson w"
##  [55] "thomas k"        "hysell g"        "dietl j"
##  [58] "skolund m"       "jigel j"         "amamovits m"
##  [61] "maser r"         "hartman m"       "melt lab"
##  [64] "adaovits m"      "kroker j"        "muniza j"
##  [67] "massey b"        "daigneault b"    "williams r"
##  [70] "armstrong s"     "chen jenny"      "matthews t"
##  [73] "matthers r"      "crockett l"      "wandtke g"
##  [76] "herry c"         "desmit d"        "gerry fountaine"
##  [79] "wang x"          "yeomans n"       "meyer f"
##  [82] "auferfheide r"   "oliver t"        "carr"
##  [85] "fountain g"      "swartzlander m"  "duncan f"
##  [88] "yeoman n"        "sandstrom r"     "rangel a"
##  [91] "trevisan s"      "champman c"      "johnston s"
##  [94] "duanca f"        "delong t"        "chew b"
##  [97] "lowe k"          "yirgoyen d"      "bangcuyo c"
## [100] "sun d"           "auferheide r"    "jain n"
## [103] "pinto m"         "clifford s"      "cecere j"
## [106] "hector r"        "aufd/showman"    "audderheide r"
## [109] "clark k"         "nocera m"        "harmon s"
## [112] "adamotvits m"    "yu e"            "m adamovits"
## [115] "wang sturtz"     "beers m"         "andrews r"
## [118] "hoertz c"        "hoodstack"       "kar s"
## [121] "archlbald j"     "altepeter m"     "vivas p"
## [124] "x wang"          "p vivas"         "esenwein e"
## [127] "vivas"           "nocera"          "kar"
## [130] "unknown"         "edward yu"       "paula vivas"
## [133] "matt hartman"
```

```
# not many options but to skim thru manually
name.levels <- as.data.frame(table(x$requested.by))
name.levels
```

Var1	Freq
adamotvits m	1
adamovits m	167
adaovits m	1
altepeter m	4
amamovits m	1
andrews r	5
archibald j	51
archlbald j	1
armstrong s	10
audderheide r	1
aufd/showman	1
aufderheide r	230
auferfheide r	1
auferheide r	4
bangcuyo c	31
beers m	7
belt p	1
buchanan c	3
carlson g	20
carr	1
carr b	71
cecere j	1
champman c	2
chapman c	14
chen j	243
chen jenny	2
chew b	1
clark k	1
clifford s	64
clingerman m	20
clingermanm	2
crockett l	1
daigneault b	2
dando t	27
delong t	1
desmit d	9
dietl j	14
duanca f	1
dudenhofer r	1
duncan f	267
edward yu	7
esenwein e	1
fechter r	71

Var1	Freq
fitt w	5
fountain g	4
fox j	97
fredendall a	3
gerry fountaine	1
gilbreath t	3
gilson d	3
gualtiere d	3
harmon s	50
hartman m	86
hector r	4
hendershot g	53
henry c	131
herry c	1
hoertz c	1
hoffman m	16
hoodstack	2
horvath l	38
hutchings d	11
hysell g	1
hysell m	3
ireland e	17
jain n	15
jigel j	1
johnston s	8
kar	5
kar s	1
kathy lowe	1
kreinbrink j	4
kroker j	5
lamb b	2
langer h	3
lowe k	39
lute c	60
lute/showman	1
m adamovits	3
madigan j	9
maser r	5
massey b	4
matt hartman	2
matthers r	1
matthews t	2
melt lab	2
meyer f	5
moosavian t	14
muniza j	28
neu m	5
neu m-dgh	1
nocera	5
nocera m	1
oliver t	4
p vivas	1

Var1	Freq
paula vivas	9
pinto m	39
rangel a	18
rigel j	100
sandstrom r	6
schneider j	1
schultz b	13
showman	1
showman j	3
showman r	448
shriver r	249
singh r	4
skoglund m	49
skolund m	1
sturtz g	67
sun d	4
swartzlander m	3
szpak t	14
thomas k	8
toplikar e	31
torres h	5
trevisan s	45
twardowska h	29
unknown	2
vivas	10
vivas p	26
wandtke g	1
wang sturtz	1
wang x	147
williams r	1
wilson s	10
wolfgram t	12
woodson w	1
x wang	1
yeoman n	1
yeomans n	16
yirgoyen d	2
yu e	117

```
x <- x %>%
  mutate(requested.by.tf = requested.by) %>%
  mutate(requested.by.tf = ifelse(grepl('vits',requested.by), "mark adamovits", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('bald',requested.by), "jim archibald", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('heid',requested.by), "ron aufderheide", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('carr',requested.by), "ben carr", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('yu',requested.by), "edward yu", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('tain',requested.by), "gerry fountaine", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('herr',requested.by), "henry c", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('henr',requested.by), "henry c", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('igel',requested.by), "judy rigel", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('lowe',requested.by), "kathy lowe", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('vivas',requested.by), "paula vivas", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('lund',requested.by), "m skoglund", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('showm',requested.by), "ralph showman", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('wang',requested.by), "xianping wang", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('hart',requested.by), "matt hartman", requested.by.tf)) %>%
  mutate(requested.by.tf = ifelse(grepl('yeom',requested.by), "n yeomans", requested.by.tf)) %>%
  mutate(requested.by = requested.by.tf) %>%
  select(-requested.by.tf)

# check length again
length(unique(x$requested.by)) # 106
```

```
## [1] 107
```

```
# seems to be good enough
unique(x$requested.by)
```

```
##   [1] "clingermanm"     "clingerman m"   "belt p"
##   [4] "jim archibald"   "twardowska h"   "fechter r"
##   [7] "shriver r"       "m skoglund"     "ralph showman"
##  [10] "szpak t"         "mark adamovits" "gilbreath t"
##  [13] "chapman c"       "madigan j"      "henry c"
##  [16] "wilson s"        "langer h"       "fox j"
##  [19] "moosavian t"     "chen j"         "ireland e"
##  [22] "sturtz g"        "gualtiere d"    "hendershot g"
##  [25] "wolfgram t"      "dando t"        "singh r"
##  [28] "schultz b"       "buchanan c"     "kathy lowe"
##  [31] "carlson g"       "torres h"       "kreinbrink j"
##  [34] "hutchings d"     "lute c"         "hoffman m"
##  [37] "fredendall a"    "hysell m"       "ben carr"
##  [40] "gilson d"        "horvath l"      "dudenhofer r"
##  [43] "ron aufderheide" "lamb b"         "neu m-dgh"
##  [46] "neu m"           "toplikar e"     "judy rigel"
##  [49] "schneider j"     "fitt w"         "woodson w"
##  [52] "thomas k"        "hysell g"       "dietl j"
##  [55] "maser r"         "matt hartman"   "melt lab"
##  [58] "kroker j"        "muniza j"       "massey b"
##  [61] "daigneault b"    "williams r"     "armstrong s"
##  [64] "chen jenny"      "matthews t"     "matthers r"
##  [67] "crockett l"      "wandtke g"      "desmit d"
##  [70] "gerry fountaine" "xianping wang"  "n yeomans"
##  [73] "meyer f"         "oliver t"       "swartzlander m"
##  [76] "duncan f"        "sandstrom r"    "rangel a"
##  [79] "trevisan s"      "champman c"     "johnston s"
##  [82] "duanca f"        "delong t"       "chew b"
##  [85] "yirgoyen d"      "bangcuyo c"     "sun d"
##  [88] "jain n"          "pinto m"        "clifford s"
##  [91] "cecere j"        "hector r"       "clark k"
##  [94] "nocera m"        "harmon s"       "edward yu"
##  [97] "beers m"         "andrews r"      "hoertz c"
## [100] "hoodstack"       "kar s"          "altepeter m"
## [103] "paula vivas"     "esenwein e"     "nocera"
## [106] "kar"             "unknown"
```

$customer.name

```
# only 2 missing customer names, replace with ASK
x %>%
  filter(is.na(x$customer.name)==TRUE)
```

request	date.poured	date.received	date.completed	requested.by	customer.name	product.tested	casting.type
3366	2014-01-22	2014-01-31	2014-02-05	ralph showman	NA	SLEEVE	Riser
3377	2014-02-25	2014-02-25	2014-02-25	ralph showman	NA	SLEEVE	Riser

```
x$customer.name[c(3366,3377)] <- "ASK"
```

$alloy

We perform pretty much the same actions as we did above, with `requested.by`.

```
unique(x$alloy)
```

```
##  [1] "GRAY IRON"            "L C STEEL"            "319 Al"
##  [4] "DUCTILE IRON"         "319 AL"               "STEEL"
##  [7] "LC STEEL"             "Al"                   "L.C. STEEL"
## [10] "WHITE IRON"           "GRAY IRPN"            "GRAY IRON /DUCTILE I"
## [13] "O"                    "NONE"                 "GRAY IRON/DUCTILE IR"
## [16] "319B Al"              "S.S."                 "319  Al"
## [19] "GRAYIRON"             "DUCILE IRON"          "CG"
## [22] "GRAY IRON,C.G."       "C.G."                 "DUCTILE IRON,GRAY IR"
## [25] "356 Al"               "440 S.S."             "440 S.S."
## [28] "COPPER"               "BRASS"                "C.G.I."
## [31] "D.I."                 "CUSTOMER STEEL"       "319Al"
## [34] "CGI"                  "FISHER STEEL"         "STEEL (FISHER)"
## [37] "GRAY IRON,D.I."       "GARY IRON"            "FRAY IRON"
## [40] "SiMo"                 "DUCITLE IRON"         "L.C STEEL"
## [43] "L.C  STEEL"           "GRAY  IRON"           "L.C.STEEL"
## [46] "ALUMINUM"             "L. C. STEEL"          "\""
## [49] "SiMo DUCTILE"         "GI/DI"                "STAINLESS STEEL"
## [52] "L.C. Steel"           "836 RED BRASS"        "Aluminum"
## [55] "ALUMINIUM"            "STEEL/GRAY IRON"      "DUCTILE"
## [58] "Gray Iron"            "unknown"              "Steel"
```

```
length(unique(x$alloy)) # 60
```

```
## [1] 60
```

```
# convert case, remove punctuations
x <- x %>%
  mutate(alloy = str_replace_all(alloy, '\\  ', '')) %>%
  mutate(alloy = str_replace_all(alloy, '\\,', '')) %>%
  mutate(alloy = str_replace_all(alloy, '\\.', '')) %>%
  mutate(alloy = str_to_lower(   alloy))

length(unique(x$alloy)) # 47
```

```
## [1] 47
```

```r
x <- x %>%
  mutate(alloy.new = alloy) %>%
  mutate(alloy.new = str_replace_all(alloy.new, "[:punct:]","none")) %>%
  mutate(alloy.new = ifelse(grepl('al',alloy), "aluminum", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('di',alloy),      "ductile iron", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('ductile',alloy), "ductile iron", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('le iron',alloy), "ductile iron", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('gray',alloy),   "grey iron", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('y iron',alloy), "grey iron", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('cg',alloy), "cgi", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('brass',alloy), "bras", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('s steel',alloy), "stainless", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('44',alloy),      "stainless", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('ss',alloy),      "stainless", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('teel',alloy), "lc steel", alloy.new)) %>%
  mutate(alloy.new = ifelse(grepl('bras',alloy), "brass", alloy.new)) %>%
  mutate(alloy.new = ifelse(alloy.new == "0" |
                              alloy.new == "none" |
                              alloy.new == "unknown", NA, alloy.new)) %>%
  mutate(alloy = alloy.new) %>%
  select(-alloy.new)

# confirm
unique(x$alloy)
```

```
##  [1] "grey iron"    "lc steel"     "aluminum"     "ductile iron"
##  [5] "white iron"   NA             "stainless"    "cgi"
##  [9] "copper"       "brass"        "simo"
```

```r
length(unique(x$alloy)) # 11
```

```
## [1] 11
```

$furnace.cycle

This datapoint kept track of how many uses each furnace lining accumulated. Instead of using continuing with this way of measuring where we increment each number, we'll split the measure into two columns: one representing the furnace liner, the other representing how many pours it lasted.

First, we assign `NA` to all furnace values with alloys of aluminum as aluminum uses a different furnace. We then create a few new columns, the first of which is `furnace` and will represent the furnace lining being used; `furnace cycle` will increment with each use of the `furnace`; and `furnace.name` will be a more endearing name given to the furnace.

```r
# since aluminum uses a different furnace, change all to NA
x$furnace.cycle[x$alloy=="aluminum"] <- NA

# test df
# select first letter to call furnace
xx <- x %>%
  select(request, furnace.cycle, alloy) %>%
  filter(alloy != "aluminum") %>%
  mutate(furnace = str_sub(furnace.cycle,1,1)) %>%
  mutate(furnace = str_to_lower(furnace)) %>%
  mutate(cycle = NA) %>%
  mutate(furnace.name = NA)
```

```r
# some NA values
xx[is.na(xx$furnace.cycle),]
```

request	furnace.cycle	alloy	furnace	cycle	furnace.name
835	NA	grey iron	NA	NA	NA
836	NA	ductile iron	NA	NA	NA
1074	NA	lc steel	NA	NA	NA
1162	NA	grey iron	NA	NA	NA
1163	NA	grey iron	NA	NA	NA
1422	NA	grey iron	NA	NA	NA
1647	NA	grey iron	NA	NA	NA
3287	NA	grey iron	NA	NA	NA
3289	NA	grey iron	NA	NA	NA
3366	NA	lc steel	NA	NA	NA
3377	NA	lc steel	NA	NA	NA

```r
# if NA, pull value above
for (i in 1:nrow(xx)){
  if (is.na(xx$furnace[[i]])){
    xx$furnace[[i]] <- xx$furnace[[i-1]]
  }
}

# zeros confused with letter o
xx[1543:1576,]
```

request	furnace.cycle	alloy	furnace	cycle	furnace.name
2186	N69	grey iron	n	NA	NA
2187	01	grey iron	0	NA	NA
2188	02	cgi	0	NA	NA
2189	03	lc steel	0	NA	NA
2190	04	grey iron	0	NA	NA
2191	05	grey iron	0	NA	NA
2192	06	grey iron	0	NA	NA
2193	06	grey iron	0	NA	NA
2194	07	grey iron	0	NA	NA
2195	08	grey iron	0	NA	NA
2197	09	grey iron	0	NA	NA
2198	010	lc steel	0	NA	NA
2199	011	grey iron	0	NA	NA
2201	O12	grey iron	o	NA	NA
2202	013	grey iron	0	NA	NA
2203	014	grey iron	0	NA	NA
2204	014	grey iron	0	NA	NA
2205	015	grey iron	0	NA	NA
2206	016	lc steel	0	NA	NA
2208	017	ductile iron	0	NA	NA
2209	018	cgi	0	NA	NA
2210	019	grey iron	0	NA	NA
2211	020	grey iron	0	NA	NA
2212	021	grey iron	0	NA	NA

request	furnace.cycle	alloy	furnace	cycle	furnace.name
2213	022	lc steel	0	NA	NA
2214	022	lc steel	0	NA	NA
2215	023	lc steel	0	NA	NA
2216	024	grey iron	0	NA	NA
2217	025	grey iron	0	NA	NA
2218	026	grey iron	0	NA	NA
2219	027	grey iron	0	NA	NA
2220	027	grey iron	0	NA	NA
2221	028	grey iron	0	NA	NA
2222	P1	grey iron	p	NA	NA

```
# replace zeros with o's
xx[xx$furnace==0,][4] <- "o"

# M between L's
xx[2679:2684,]
```

request	furnace.cycle	alloy	furnace	cycle	furnace.name
3466	L-25	lc steel	l	NA	NA
3467	M1, M3	ductile iron	m	NA	NA
3468	L-26, M2	grey iron	l	NA	NA
3469	M6	grey iron	m	NA	NA
3470	M4, M5	ductile iron	m	NA	NA
3471	M6	grey iron	m	NA	NA

```
xx[xx$request==3468,][4] <- "m"

# O between N's
xx[2725:2729,]
```

request	furnace.cycle	alloy	furnace	cycle	furnace.name
3516	N16	lc steel	n	NA	NA
3517	O1	grey iron	o	NA	NA
3518	N17	lc steel	n	NA	NA
3519	O5	lc steel	o	NA	NA
3520	O4	grey iron	o	NA	NA

```
# switch place
which(xx$request==3517) # 2726
```

```
## [1] 2726
```

```
xx[2726,][1] <- 3518
xx[2727,][1] <- 3517
# rearrange rows
xx <- xx %>%
  arrange(request)

# increment furnace cycle if furnace before = furnace current
# if not, assign value = 1
cycle.counter=1
for (i in 2:nrow(xx)){
  # first row = 1
  xx$cycle[[1]] <- 1
  # vars
  before =  i-1
  current = i
  # current != before, start counter over
  if (xx$furnace[[current]] != xx$furnace[[before]]){
    cycle.counter=1
    xx$cycle[[current]] <- cycle.counter
  }
  if (xx$furnace[[current]] == xx$furnace[[before]]){
    cycle.counter=cycle.counter+1
    xx$cycle[[current]] <- cycle.counter
  }
}

# load names to assign to furnaces, shuffle them
data("common_names")
names <- common_names[1:length(common_names)]
set.seed(1111)
names <- sample(names)

# assign names instead of letters to each furnace
name.counter = 0
for (i in 1:nrow(xx)){
  if (xx$cycle[[i]] == 1){
    name.counter=name.counter+1
    xx$furnace.name[[i]] <- names[[name.counter]]
  }
  if (xx$cycle[[i]] != 1){
    xx$furnace.name[[i]] <- names[[name.counter]]
  }
}

# rejoin data
x <- full_join(x,xx)
```

$casting.type

There are quite a few different kinds of castings. I've manually gone thru and renamed a few, it seems an improvement.

```
# way too many unique
unique(x$casting.type)
```

```
##   [1] "STEPCONE"                "EROSION WEDGE"
##   [3] "PENETRATION"             "SHRINKAGE CUBE"
##   [5] "SHAKEOUT TREE"           "AFS MUGS"
##   [7] "WARPAGE BLOCKS"          "CUBE/SLEEVE"
##   [9] "IMPELLAR"                "EROEION WEDGE"
##  [11] "SLEEVE"                  "STEPCONE-GRAPHITE"
##  [13] "IMPELLER"               "WARPAGE BLOCK"
##  [15] "GEAR"                   "GRAPHITE STEPCONE"
##  [17] "SLEEVE MODULUS"         "SHAKEOUT TREES"
##  [19] "MODULUS EXT."           "SOOT PLATE"
##  [21] "MYSTERY"                "CUBES & MOD. EXT."
##  [23] "SHAKEOUT TREE 2\""      "WARPAGE BLOCK LRG"
##  [25] "UNSUPPORTED SLEEVE"     "SHAKEOUT"
##  [27] "UN-SUPP, SLEEVE"        "PEPETRATION"
##  [29] "SHRINKAGE CUBER"        "IMPELLER CASTING"
##  [31] "TEST BAR"               "DILATION"
##  [33] "GEAR MOLD"              "EROSON WEDGE"
##  [35] "PIG"                    "CHILL WEDGE"
##  [37] "FLUIDITY SPIRAL"        "FLUIDITY TREE"
##  [39] "STEPSONE"               "FLOW PLATE"
##  [41] "U.S. SLEEVE"            "EROSION WEDGE BASE"
##  [43] "FLOW TREE"              "OIL GALLERY"
##  [45] "LARGE GEAR BOX"         "POURING CUP"
##  [47] "REDFORD PLATE"          "EROSION WEDGGE"
##  [49] "MANIFOLD"               "OIL GALLEY"
##  [51] "7 INCH SHAKEOUT"        "NONE"
##  [53] "SPIRAL"                 "FLOW PATTERN"
##  [55] "CUSTOMER"               "EXACTCAST PLAQUE"
##  [57] "SHAPE TEST"             "PLATE"
##  [59] "WEDGE"                  "PLAQUE"
##  [61] "DETAIL PLAQUE"          "STEP BLOCK"
##  [63] "MODULUS"                "REFINER PLATE"
##  [65] "BISHOP"                 "SEMI-PERM"
##  [67] "UNSUPPORTED RISERS"     "CAROUSEL-TEMP."
##  [69] "BUCKLE"                 "R.R. WHEEL"
##  [71] "BLOCK"                  "EROISON WEDGE"
##  [73] "EROSION WEDGE TREE"     "SLEEVE-SUPPORTED"
##  [75] "DOG-BONE"               "MANDREL"
##  [77] "EXPERIMENTAL"           "SOOT PLATE INSERTS"
##  [79] "EXPERIMENTAL-NEMAK"     "HOODSTACK"
##  [81] "PENETRATIONS"           "PIPE"
##  [83] "FRYING PAN"             "SLEEVE FILTER"
##  [85] "THIN WALL"              "POURING CUP FILTER"
##  [87] "GM BLOCK"               "ROTOR"
##  [89] "END CAP"                "PENETRATION RISER"
##  [91] "BELT BUCKLES"           "BELT BUCKLE"
##  [93] "FLUIDITY FILTER"        "PENTRATION"
##  [95] "EROSION WEDGES"         "TEST-CUSTOMER"
##  [97] "PLAQUES"                "SHRINKAGE CUBES"
##  [99] "PIG TEST COUPONS"       "PIG,WARPAGE BLOCK"
## [101] "SAND MAGAZINE"          "SMALL STEP BLOCK"
## [103] "TENSILE BARS"           "STEPBLOCK"
## [105] "4 X 8 PLATE"            "BRACKET"
## [107] "BRACKETS"               "POKER CHIP/B.B."
## [109] "PENETRATION,PIG"        "FILTER CAVITY"
## [111] "PENETRATION+PIG"        "EROSION  WEDGE"
## [113] "SLEEVES"                "FIAT HEAD"
## [115] "TENSILE SHAKEOUT"       "TENSILES"
## [117] "PROTOTYPE"              "SOOT PLATE, PIG"
## [119] "PIG-FILTERED"           "REFINER TEST PLATE"
## [121] "FILTER TEST"            "FILTER-PIG"
## [123] "ROTORS"                 "SHRINK CUBE"
## [125] "SHINKAGE CUBE"          "EROSION WEDGE,PIG"
## [127] "GEAR, PIG"              "MOD. PIG"
## [129] "TENSILE,CHILL,BUT."     "THIN-WALL"
## [131] "IMPELLER-PIG"           "MOD. SOOT PLATE"
## [133] "BUCKLES"                "FILTER"
## [135] "STEP BLOCK-SMALL"       "SCAB BLOCK"
## [137] "CLAMP"                  "GATOR CORE CASTING"
## [139] "GATOR CORE"             "GATOR"
## [141] "STEP BLOCK SINT."       "ANCHOR"
## [143] "PROPELLER"              "SLEEVE/PIG"
## [145] "POKER CHIP"             "CHESS PIECES"
## [147] "DUDE"                   "STE[PCONE"
## [149] "GRAVE MARKER"           "TEST BARS"
## [151] "SHAKEOUT-SPM"           "TEST CASTING-CGI"
## [153] "SHRICK CUBE"            "SHAKEOUT S.P.M."
## [155] "PEN/DURAMETAL"          "SHRIINKAGE CUBE"
## [157] "STEPBLOCKS"             "WARAPGE BLOCK"
## [159] "GM BLOCKS"              "OSU CASTING"
## [161] "WARPAGE CASTING"        "SMALL STEPCONE"
## [163] "STEPCONE GRAPHITE"      "IIMPELLER"
## [165] "PENTERATION"            "SOOTPLATE"
## [167] "RISER SLEEVE"           "HELICOPTER"
## [169] "5\" SHRINK CUBE"        "3.5\" SHRINK CUBE"
## [171] "FILTER TEST-PIG"        "SLEEVE CYLINDER"
## [173] "SLEEVE-PIG"             "FILTER POURING CUP"
## [175] "FILTERS"                "FILTER TESTS"
## [177] "SHRIKAGE CUBE"          "SHRINKABE CUBE"
## [179] "SHRINKEAGE CUBE"        "GM HEAD TEST"
## [181] "ASK SYMBOL"             "SLEEVES-SMOKE"
## [183] "ASK BALL, PIG"          "PIG-M.L. CALIB."
## [185] "WAPPAGE BLOCK"          "FILTER MOLD"
## [187] "PENETRATION-SLEEVE"     "SHRINK CUBES"
## [189] "SHRINKAGE  CUBE"        "PENETRTATION"
## [191] "PIG-FILTER TEST"        "STEPCONES"
## [193] "SMALL PIG"              "SMALL PIG MOLD"
## [195] "POUR CUP & PIGS"        "SLEEVES IN DRY SAN"
## [197] "IMPELLERS"              "WARPAGE BARS"
## [199] "PIG MOLDS"              "STEP CONES"
## [201] "GEAR MOLDS"             "BRAKE ROTORS"
## [203] "RISERS"                 "WARPAGE BAR"
## [205] "SAMPLES"                "WEDGES/STEP CONES"
## [207] "STEP CONE"              "STEP-CONES"
## [209] "4\" SHRINK CUBES"       "3\" SHRINK CUBES"
## [211] "SC / PENETRATIONS"      "ER WEDGE/ STEP-CON"
## [213] "CHILL WEDGE/COUPON"     "PENETRATION/STEP-"
## [215] "SPM"                    "Riser"
## [217] "WARPAGE BLOCKS/BIO"     "wedge/pene/so tree"
## [219] "SPM/WARPAGE BLOCKS"     "SHAKE-OUT TREE"
## [221] "PIGS"                   "Bio-Spheres"
## [223] "PENE/STEP/EROSION"      "EROSION WEDGE(RS)"
## [225] "PENE/STEP CONE"         "STEP CONE/EROSION"
## [227] "SHRINK CUBE/IMPELL"     "\""
## [229] "BRAKE ROTOR"           "EROSION WEDGES/SC"
## [231] "SHAKE OUT TREE"        "EROSION WEDGE/STEP"
## [233] "MODIFIED RISER"        "inverted sleeves"
## [235] "INVERTED SLEEVES"      "STEP-CONE"
## [237] "EROSION/STEPCONE"      "MTI castings"
## [239] "MTI Castings"          "MTI Casting"
## [241] "PENETRATION/STEP-C"    "MTI CASTING"
## [243] "LAUNDER"               "SC/PENE/HALF PIGS"
## [245] "S. CUBE/PENETRATIO"    "S. CUBE/PIG MOLD"
## [247] "PIG MOLD"              "MTI MOLD"
## [249] "SHRINK CUBE 5\""       "SHRINK PLATE"
## [251] "STEP BLOCK/SPIRAL"     "STEP CONE/PENE"
## [253] "EROSION WEDGE/PENE"    "EROSION/PENE"
## [255] "STEP-CONE/PENETRAT"    "PENE/STEP-CONE"
## [257] "PENE/BRAKE"            "PENE/STEPCONE"
## [259] "Step cones"            "PENE/SHRINK CUBE"
## [261] "CAT BLOCK"             "IRREGULAR GEAR"
## [263] "DBL PENETRATIONS"      "WARPAGE"
## [265] "GRAPHITE MOLDS"        "unknown"
## [267] "Shrink Cubes"          "Penetrations"
## [269] "Erosion wedges"        "Stepcones + Investment"
## [271] "3\" Shrink cubes"      "Pen + Ero"
## [273] "Shakeout trees"        NA
```

```
length(unique(x$casting.type)) # 274
```

```
## [1] 274
```

```
# remove double spaces, commas, periods
x <- x %>%
  mutate(casting.type1 = str_replace_all(casting.type, '\\  ', ' ')) %>%
  mutate(casting.type1 = str_replace_all(casting.type, '\\,', ' ')) %>%
  mutate(casting.type1 = str_replace_all(casting.type, '\\.', ' ')) %>%
  mutate(casting.type1 = str_to_lower(   casting.type)) %>%
  mutate(casting.type = casting.type1) %>%
  select(-casting.type1)

# unique(x$casting.type)
length(unique(x$casting.type)) # 264
```

```
## [1] 265
```

```
x <- x %>%
  mutate(casting.type1 = casting.type) %>%
  mutate(casting.type1 = ifelse(grepl('cube',casting.type), "shrink cube", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('ero',casting.type), "erosion wedge", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('sleeve',casting.type), "sleeves", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('shake',casting.type), "shakeout tree", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('page',casting.type), "warpage block", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('ration',casting.type), "penetration", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('pene',casting.type), "penetration", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('graphit',casting.type), "graphite step", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('cone',casting.type), "stepcone", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('steps',casting.type), "stepcone", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('graphite',casting.type), "graphite stepcones", casting.type1)) %
  mutate(casting.type1 = ifelse(grepl('fluid',casting.type), "fluidity spiral", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('buck',casting.type), "belt buckles", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('gator',casting.type), "gator", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('soot p',casting.type), "sootplate", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('gear',casting.type), "gear mold", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('rotor',casting.type), "brake rotor", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('pig',casting.type), "pigs", casting.type1)) %>%
  mutate(casting.type1 = ifelse(grepl('impel',casting.type), "di impeller", casting.type1)) %>%
  mutate(casting.type = casting.type1) %>%
  select(-casting.type1)

# slightly better, not perfect
unique(x$casting.type)
```

```
##   [1] "stepcone"           "erosion wedge"      "penetration"
##   [4] "shrink cube"        "shakeout tree"      "afs mugs"
##   [7] "warpage block"      "sleeves"            "di impeller"
##  [10] "graphite stepcones" "gear mold"          "modulus ext."
##  [13] "sootplate"          "mystery"            "test bar"
##  [16] "dilation"           "pigs"               "chill wedge"
##  [19] "fluidity spiral"    "flow plate"         "flow tree"
##  [22] "oil gallery"        "pouring cup"        "redford plate"
##  [25] "manifold"           "oil galley"         "none"
##  [28] "spiral"             "flow pattern"       "customer"
##  [31] "exactcast plaque"   "shape test"         "plate"
##  [34] "wedge"              "plaque"             "detail plaque"
##  [37] "step block"         "modulus"            "refiner plate"
##  [40] "bishop"             "semi-perm"          "unsupported risers"
##  [43] "carousel-temp."     "belt buckles"       "r.r. wheel"
##  [46] "block"              "dog-bone"           "mandrel"
##  [49] "experimental"       "experimental-nemak" "hoodstack"
##  [52] "pipe"               "frying pan"         "thin wall"
##  [55] "pouring cup filter" "gm block"           "brake rotor"
##  [58] "end cap"            "test-customer"      "plaques"
##  [61] "sand magazine"      "small step block"   "tensile bars"
##  [64] "stepblock"          "4 x 8 plate"        "bracket"
##  [67] "brackets"           "poker chip/b.b."    "filter cavity"
##  [70] "fiat head"          "tensiles"           "prototype"
##  [73] "refiner test plate" "filter test"        "tensile,chill,but."
##  [76] "thin-wall"          "filter"             "step block-small"
##  [79] "scab block"         "clamp"              "gator"
##  [82] "step block sint."   "anchor"             "propeller"
##  [85] "poker chip"         "chess pieces"       "dude"
##  [88] "grave marker"       "test bars"          "test casting-cgi"
##  [91] "pen/durametal"      "stepblocks"         "warapge block"
##  [94] "gm blocks"          "osu casting"        "helicopter"
##  [97] "filter pouring cup" "filters"            "filter tests"
## [100] "gm head test"       "ask symbol"         "filter mold"
## [103] "risers"             "samples"            "er wedge/ step-con"
## [106] "chill wedge/coupon" "spm"                "riser"
## [109] "bio-spheres"        "\""                 "modified riser"
## [112] "mti castings"       "mti casting"        "launder"
## [115] "mti mold"           "shrink plate"       "step block/spiral"
## [118] "cat block"          "unknown"            NA
```

```r
length(unique(x$casting.type)) # 120
```

```
## [1] 120
```

$sand.type

Not the most important variable, will change a few of the obvious errors.

```r
unique(x$sand.type)
```

```
##   [1] "TECHNISAND 1L-5W"     NA                     "NONE"                
##   [4] "UNIMIN F-60"          "WEDRON 540"           "CUSTOMER"            
##   [7] "TECHNISAND 1L-=5W"    "TECHNIAND 1L-5W"      "WEDRON RECLAIM"      
##  [10] "1L-5W/SGT"            "1L-5W+SGT"            "TECNNISAND 1L-5W"    
##  [13] "RECLAIM/WEDRON 540"   "WEDRON RECLAIM/540"   "WEDRON REC"          
##  [16] "SEMI-PERM MOLD"       "GREENSAND"            "WEDRON 530"          
##  [19] "TECHNISAND 1L-5W/SGT" "OKLAHOMA 90"          "TECHNISAND 1LK-5W"   
##  [22] "CUSTOMER RECLAIM"     "1L-5W/J1"             "1L-5W/SGT/J1"        
##  [25] "WEDRON 520"           "WEDRON 520/ZIRCON"    "TECHNISAND 1L05W"    
##  [28] "CUSTOMERS"            "WEDRON 510"           "ZIRCON RECLAIM"      
##  [31] "NUGENT W-3"           "WEXFORD 450H"         "TECHNISAND/J1"       
##  [34] "NUGENT 630/GREENSAND" "ZIRCON RECLIAM"       "TECHNISAND1L-5W"     
##  [37] "NUGENT 480"           "UNKNOWN"              "TECHNISAND/SGT"      
##  [40] "RECLAIM/WEDRN 540"    "TECHNISAND 1L-6W"     "OK 80/1L-5W"         
##  [43] "RECLAIMED ZIRCON/ZIR" "W-540,OK90,ZIRCON,1L" "W-540,OK90,ZIRCON"   
##  [46] "OK90,G220,1L-5W"      "1L-5W"                "OK-90/1L-5W"         
##  [49] "1L-5W/OK-90"          "1L-5W/0K-90"          "TECHNISAND 1L-51"    
##  [52] "1L-5W/OK 90"          "OGELBAY"              "WEDRON 320"          
##  [55] "1L-5W/OK90"           "TECHNISAND  1L-5W"    "DUR. RECLIAM/1L-5W"  
##  [58] "BADGER 5574"          "DUR.RECLAIM/1L-5W"    "DUR.RECLALIM/1L-5W"  
##  [61] "RECLAIM/1L-5W"        "NUGENT 510"           "OK/90,1L-5W"         
##  [64] "GELHAR M-50"          "TECHISAND 1L-5W"      "VEIGA"               
##  [67] "VEIGA/AL-5W"          "OK 90/1L-5W"          "540/520"             
##  [70] "SLEEVE 220"           "1L-5W/OK 90"          "OK 90/EXACTHERM"     
##  [73] "TECHNIASAND 1L-5W"    "W540,CHROMITE,EX."    "TECHNISAND 1l-5W"    
##  [76] "ZIRCON,W540,1L-5W"    "MANELY 1L-5W"         "ZIRCON, W540"        
##  [79] "MANLEY 1L-5W"         "WERON 540/500W"       "WEDRON 460"          
##  [82] "WEDRON 410"           "WEDORN 410"           "WERON 410"           
##  [85] "CUSTOMER,410"         "CUSTOMER R."          "410/540"             
##  [88] "WEDORON 410"          "WEDRON410"            "SPM"                 
##  [91] "\""                  "GRAPHITE MOLD"        "WEDRON  410"         
##  [94] "W410"                 "unknown"              "W411"                
##  [97] "W412"                 "W413"                 "W414"                
## [100] "W415"                 "W416"                 "W417"                
## [103] "W418"                 "W419"                 "W420"                
## [106] "W421"                 "W422"                 "W423"                
## [109] "W424"                 "W425"                 "W426"                
## [112] "W427"                 "W428"
```

```r
length(unique(x$sand.type)) # 113
```

```
## [1] 113
```

```r
x <- x %>%
  mutate(sand.type1 = str_replace_all(sand.type, '\\ ', ' ')) %>%
  mutate(sand.type1 = str_replace_all(sand.type, '\\,', ' ')) %>%
  mutate(sand.type1 = str_replace_all(sand.type, '\\.', ' ')) %>%
  mutate(sand.type1 = str_to_lower(   sand.type)) %>%
  mutate(sand.type = sand.type1) %>%
  select(-sand.type1)

# unique(x$sand.type)
length(unique(x$sand.type)) # 111
```

```
## [1] 111
```

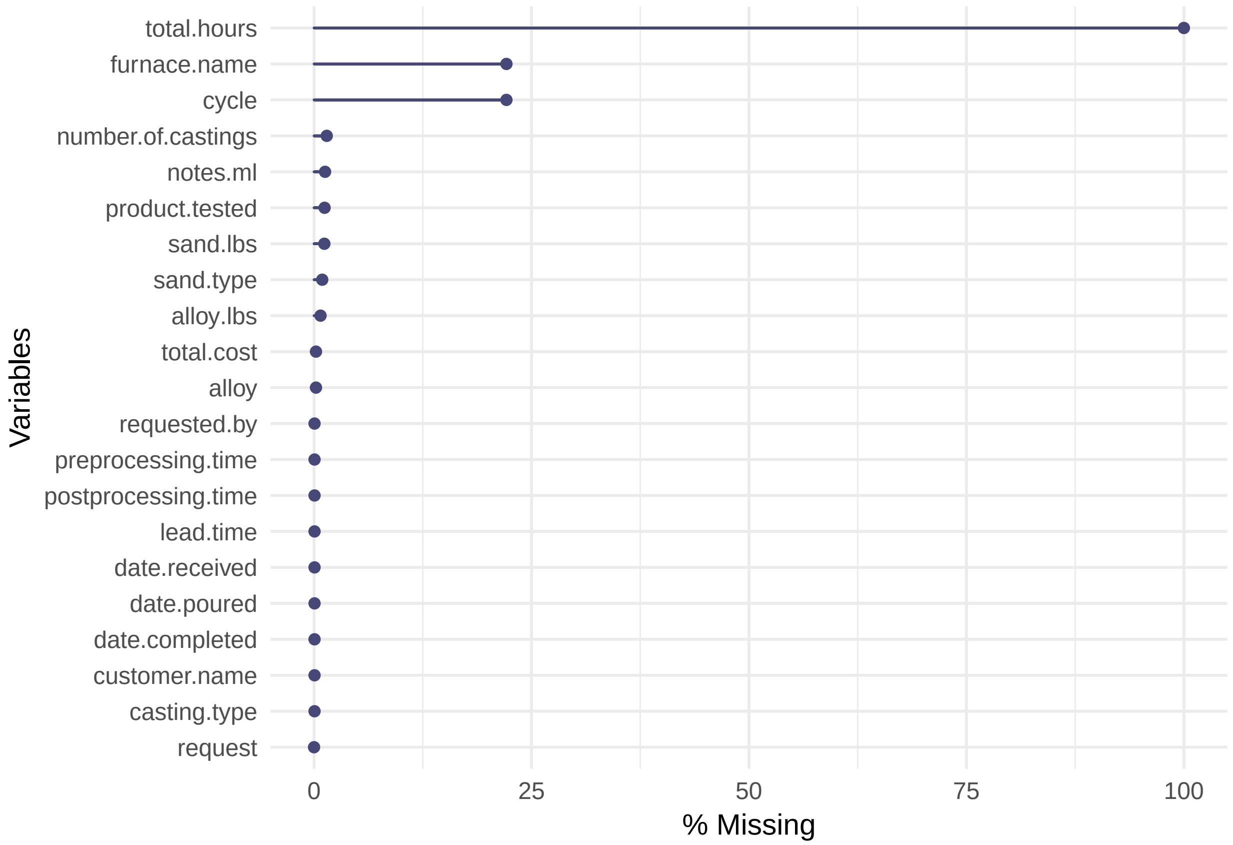
```
x <- x %>%
  mutate(sand.type1 = sand.type) %>%
  mutate(sand.type1 = ifelse(grepl('w41',sand.type), "w410", sand.type1)) %>%
  mutate(sand.type = sand.type1) %>%
  select(-sand.type1)

# unique(x$sand.type)
length(unique(x$sand.type)) # 102
```

```
## [1] 102
```

Finalize, reorder variables

We've done enough cleaning for some analysis, will reorder some variables for more clear presentation and change some column classes. The final dataframe will be renamed `y` instead of `x` and exported to a new file..

```
################################
y <- x %>%
  select(request,
         date.received,
         date.poured,
         date.completed,
         requested.by,
         customer.name,
         product.tested,
         casting.type,
         number.of.castings,
         alloy,
         alloy.lbs,
         sand.type,
         sand.lbs,
         total.hours,
         total.cost,
         preprocessing.time,
         postprocessing.time,
         lead.time,
         furnace.name,
         cycle,
         notes.ml) %>%
  mutate(requested.by=as.factor(requested.by)) %>%
  mutate(customer.name=as.factor(customer.name)) %>%
  mutate(product.tested=as.factor(product.tested)) %>%
  mutate(casting.type=as.factor(casting.type)) %>%
  mutate(alloy=as.factor(alloy)) %>%
  mutate(sand.type=as.factor(sand.type)) %>%
  mutate(furnace.name=as.factor(furnace.name))

gg_miss_var(y, show_pct = T)
```



```
glimpse(y)
```

```
## Observations: 3,631
## Variables: 21
## $ request             <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...
## $ date.received       <date> 1999-01-04, 1999-01-04, 1999-01-04, 1999-...
## $ date.poured         <date> 1999-01-05, 1999-01-06, 1999-01-07, 1999-...
## $ date.completed      <date> 1999-01-13, 1999-01-13, 1999-01-13, 1999-...
## $ requested.by        <fct> clingermanm, clingerman m, clingerman m, c...
## $ customer.name       <fct> TS&D, TS&D, TS&D, TS&D, BRILLION, K O STEE...
## $ product.tested      <fct> ISOCURE, ISOCURE, ISOCURE, ISOCURE, ISOCUR...
## $ casting.type        <fct> stepcone, stepcone, erosion wedge, erosion...
## $ number.of.castings  <dbl> 8, 8, 8, 8, 3, 1, 8, 10, 8, 4, 10, 8, 2, 1...
## $ alloy               <fct> grey iron, grey iron, grey iron, grey iron...
## $ alloy.lbs           <dbl> 250, 250, 600, 600, 90, 90, 160, 30, 20, 1...
## $ sand.type           <fct> technisand 11-5w, technisand 11-5w, techni...
## $ sand.lbs            <dbl> 840, 840, 1680, 1680, 270, 210, 640, 240, ...
## $ total.hours         <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ total.cost          <dbl> 1300, 1300, 2210, 2210, 862, 715, 2080, 84...
## $ preprocessing.time  <dbl> 1, 2, 3, 4, 5, 2, 1, 3, 2, 7, 1, 1, 4, 1, ...
## $ postprocessing.time <dbl> 8, 7, 6, 5, 1, 2, 8, 1, 1, 4, 1, 3, -2, 5,...
## $ lead.time           <dbl> 9, 9, 9, 9, 6, 4, 9, 4, 3, 11, 2, 4, 2, 6,...
## $ furnace.name        <fct> regenia, regenia, regenia, regenia, regeni...
## $ cycle               <dbl> 1, 2, 3, 4, 5, 6, 7, NA, NA, NA, NA, NA, N...
## $ notes.ml            <chr> "TEST NEW BASE RESIN WITH STEPCONE CASTING...
```

```
# export to xls
write.xlsx(y, file=paste0(getwd(),"/data/cleanedMAL.xlsx"), sheetName="Sheet1",
           col.names=TRUE, row.names=TRUE, append=FALSE)
```

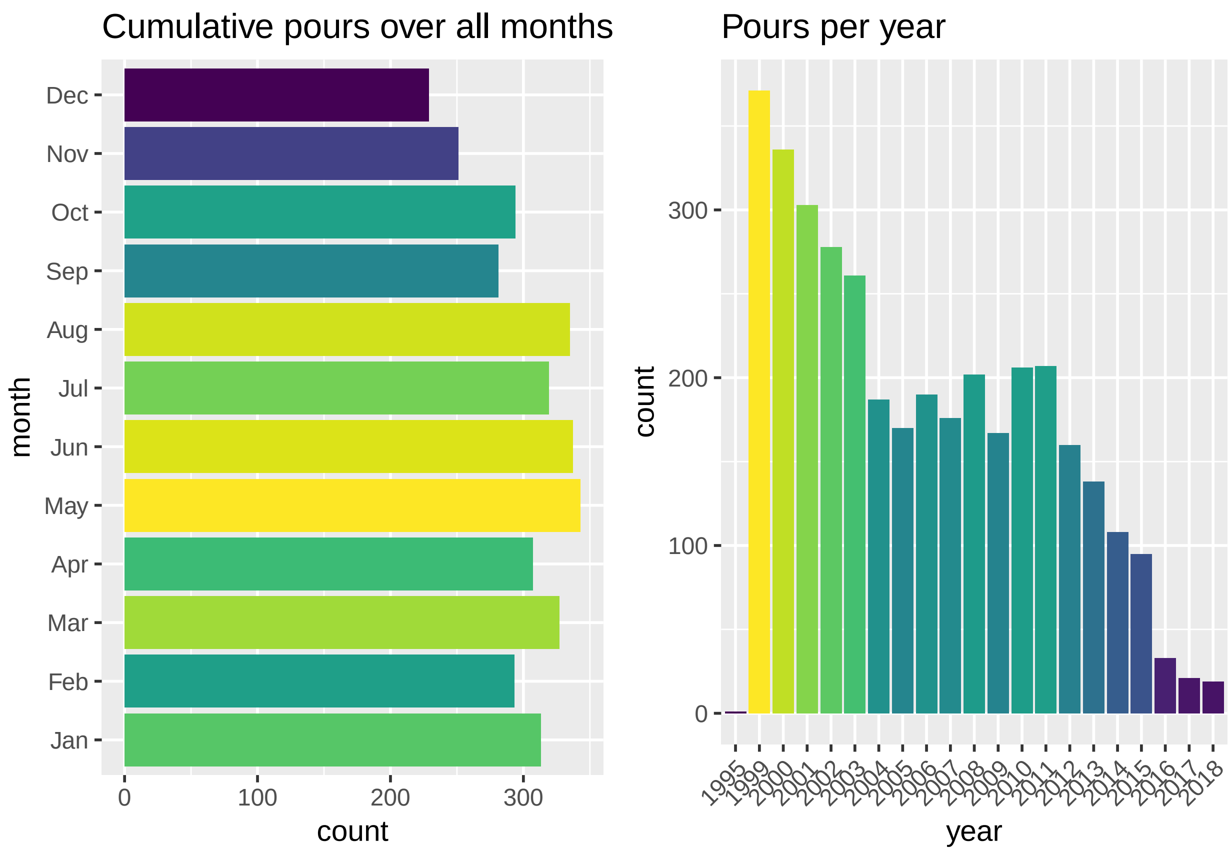
Analysis

Now we need to figure out what to do with the data. First we can try some simple EDA with what variables we have, then get into analysis more focused on furnace life.

EDA

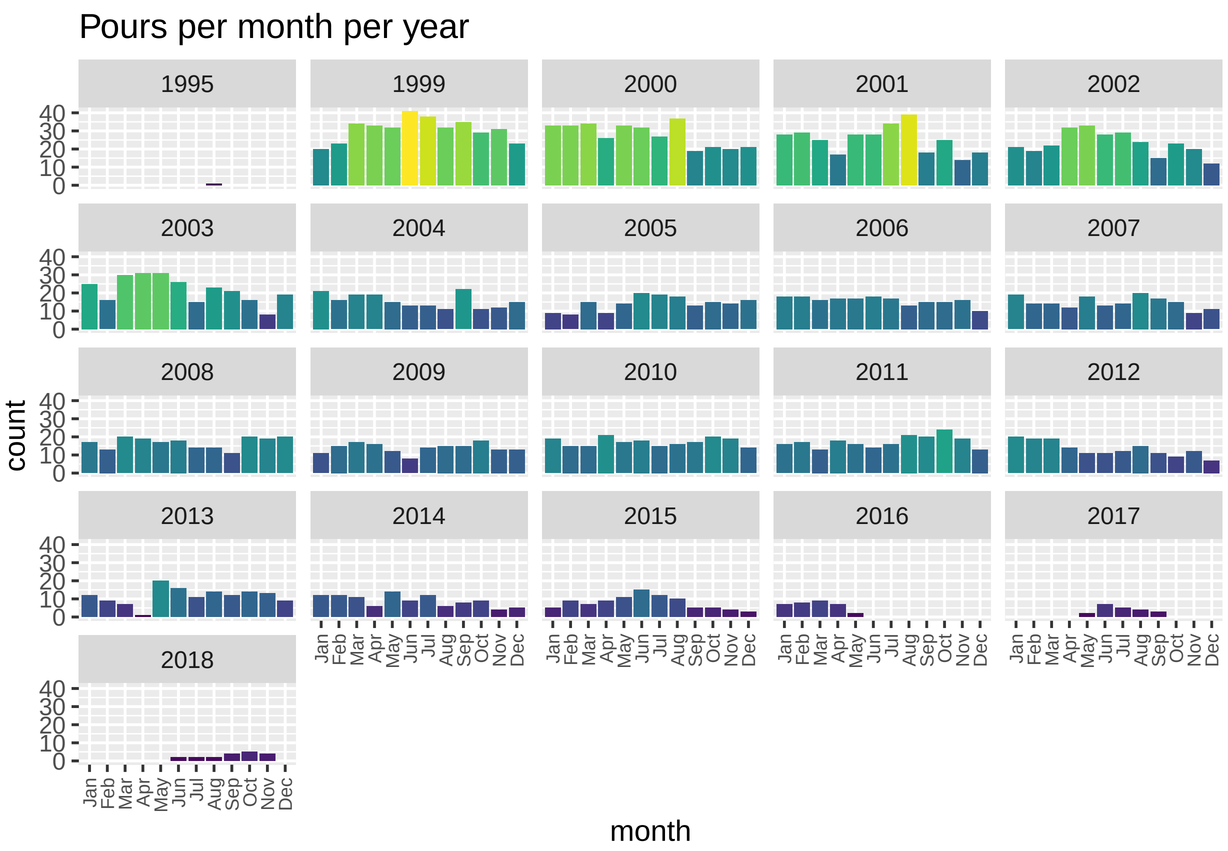
A couple basic plots show our busiest months occur between August and May, and that pour frequency has reduced dramatically since 1999, or even 2015 for that matter.

```
# Histogram of pours per month
g1 <- y %>%
  mutate(month=as.factor(substring(months.Date(x$date.poured),1,3))) %>%
  ggplot(aes(x=month,fill=..count..))+
  geom_histogram(stat="count")+
  scale_x_discrete(limits=c("Jan","Feb","Mar","Apr","May","Jun",
                            "Jul","Aug","Sep","Oct","Nov","Dec"))+
  ggtitle("Cumulative pours over all months")+
  scale_fill_viridis()+
  theme(legend.position = "none")+
  coord_flip()

# Histogram of pours per year
g2 <- y %>%
  mutate(year=as.factor(substring(x$date.poured,1,4))) %>%
  filter(!is.na(year)) %>%
  ggplot(aes(x=year,fill=..count..))+
  geom_histogram(stat="count")+
  ggtitle("Pours per year")+
  scale_fill_viridis()+
  theme(legend.position = "none")+
  theme(axis.text.x = element_text(angle=45,hjust=1))

grid.arrange(g1,g2,ncol=2)
```



Faceting number of pours over the years gives a little more insight as to when pours were occuring.

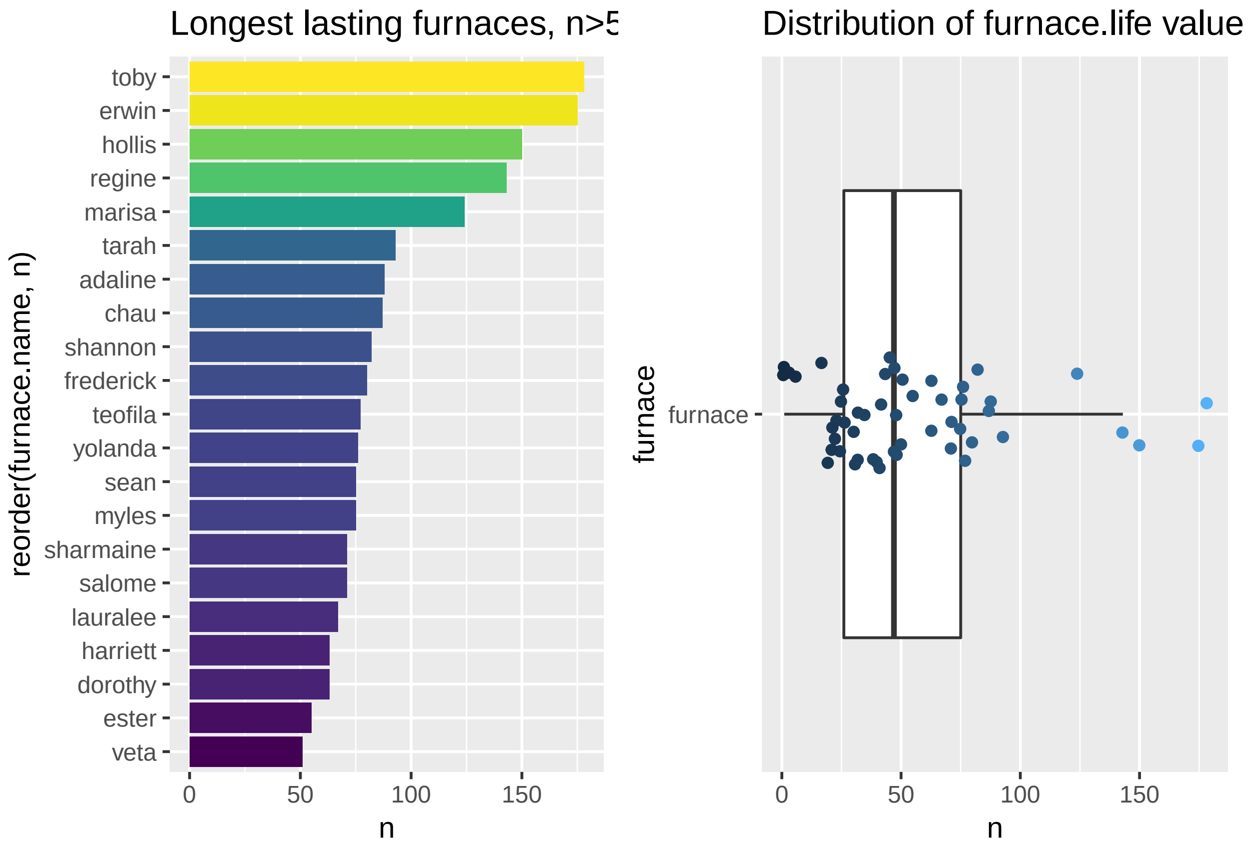
```
# Pours per month faceted by year
y %>%
  mutate(month=as.factor(substring(months.Date(x$date.poured),1,3))) %>%
  mutate(year=as.factor(substring(x$date.poured,1,4))) %>%
  filter(!is.na(year)) %>%
  ggplot(aes(x=month,fill=..count..))+
  geom_histogram(stat="count")+
  scale_x_discrete(limits=c("Jan","Feb","Mar","Apr","May","Jun",
                            "Jul","Aug","Sep","Oct","Nov","Dec"))+
  ggtitle("Pours per month per year")+
  scale_fill_viridis()+
  theme(legend.position = "none")+
  theme(axis.text.x = element_text(angle=90,hjust=1,vjust=0.5,size=7))+
  facet_wrap(year~.)
```



Furnace life

Why are some furnaces lasting longer than others? Dramatically so in some cases? Plotting the longest lasting furnaces (`n > 50`) shows the longest lasting furnace is `toby`, which lasted 178 days. These extremely high values seem like outliers based on experience and when we plot the values using a boxplot, our plot confirms they are outliers.

```
# barplot of longest lasting furnaces
p1 <- y %>%
  filter(!is.na(furnace.name)) %>%
  mutate(furnace.name=as.factor(furnace.name)) %>%
  count(furnace.name) %>%
  # arrange(desc(n)) %>%
  filter(n>50) %>%
  ggplot(aes(x=reorder(furnace.name,n),y=n,fill=n))+
  geom_bar(stat="identity")+
  coord_flip()+
  scale_fill_viridis()+
  theme(legend.position = "none")+
  ggtitle("Longest lasting furnaces, n>50")

# boxplot of furnace life
p2 <- y %>%
  filter(!is.na(furnace.name)) %>%
  mutate(furnace.name=as.factor(furnace.name)) %>%
  count(furnace.name) %>%
  select(-furnace.name) %>%
  mutate(furnace = as.factor("furnace")) %>%
  ggplot(aes(y=n,x=furnace))+
  geom_boxplot(outlier.shape = NA,
               position=position_dodge(width=.9))+
  geom_jitter(aes(color=n),width=.1)+
  coord_flip()+
  theme(legend.position = "none")+
  ggtitle("Distribution of furnace.life values")

grid.arrange(p1,p2,nrow=1)
```


Longest lasting furnaces, n>5
reorder(furnace.name, n)
toby
erwin
hollis
regine
marisa
tarah
adaline
chau
shannon
frederick
teofila
yolanda
sean
myles
sharmaine
salome
lauralee
harriett
dorothy
ester
veta
n
Distribution of furnace.life value
furnace
furnace
n