08 Feature Engineering

Fabian Blasch

06/22/2022

1 Load Data

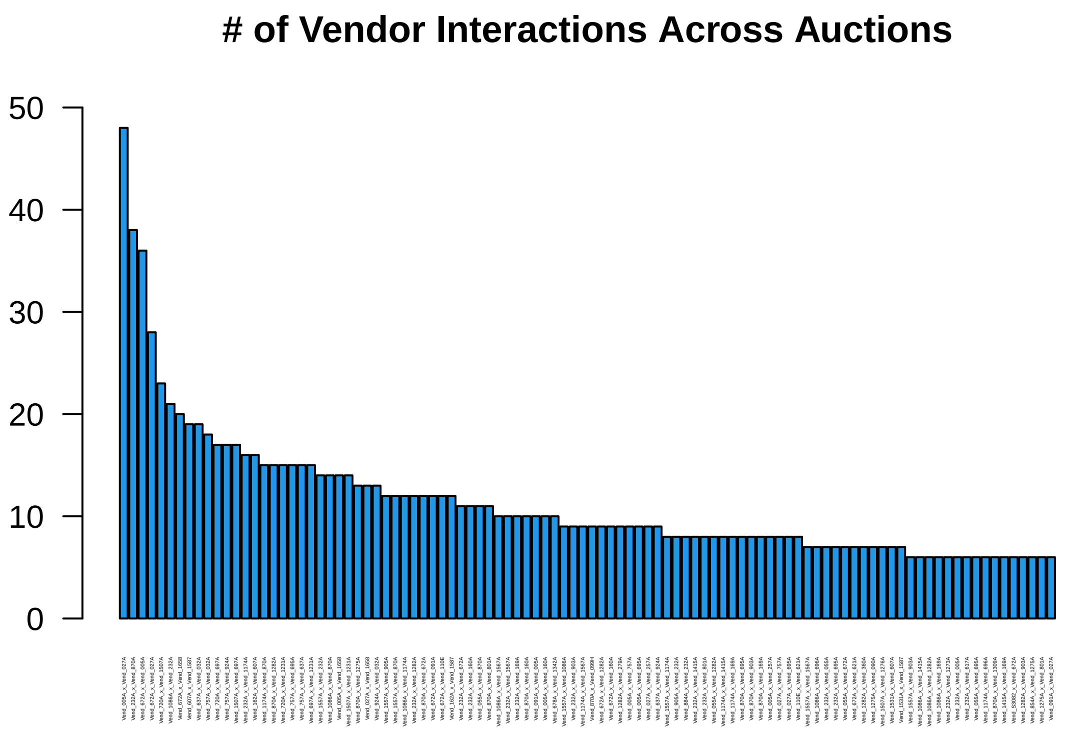
```r
# source AUX
source("./../Misc/Auxilliary.R")

# packages
get.package(c("ggplot2", "patchwork", "modeldata", "recipes", "stringr"))

# load data
dat_bids <- readRDS("./../../Data/Bid Tab RDS/Bids_df.RDS")
dat_aucs <- readRDS("./../../Data/Bid Tab RDS/Aucs_df.RDS")
```

2 Bidder Interactions

Given that some bidders may interact when finding prices it makes sense to create a dummy that represents bidder groups. Accordingly, we may first identify the column via a prefix *Vend_* to then find all the pairwise interactions utilizing the package *recipes*.

```r
# change vendor ID names to subset easier with starts_with
indlv <- which(names(dat_aucs) == "969A")
names(dat_aucs)[11:indlv] <- paste0("Vend_", names(dat_aucs)[11:indlv])

# consolidate vendors that only appear combined
auc_cols_log <- names(dat_aucs) |> stringr::str_detect("Vend_")
convend <- remove_dups_keep_name(dat_aucs[, auc_cols_log])

# remove
dat_aucs[, auc_cols_log] <- NULL

# add corrected factors
dat_aucs <- cbind(dat_aucs, convend)

# now use recipes to create vendor interactions
recip <- recipe(Winning_Bid ~., data = dat_aucs)

# interactions
```

```r
recip |> step_interact(terms = ~ starts_with("Vend"):starts_with("Vend")) |>
         prep(dat_aucs) |>
         bake(dat_aucs) -> dat_aucs_vend_int

# sapply over cols remove all columns that contain only 0, i.e., all interactions
# of firms that never bid in the same auction
sapply(dat_aucs_vend_int, \(x){

  # check if amount of occurrences is larger than
  (sum(abs(x)) > 0) |> tryCatch(error = \(e) TRUE) # catch all cases where summing
                                                    # makes no sense and return TRUE

}) -> tmp

# use to subset our data set
dat_aucs_vend_int <- dat_aucs_vend_int[, tmp]


# display which bidders interact most
ind <- str_detect(names(dat_aucs_vend_int), "_x_")
sm <- sapply(dat_aucs_vend_int[, ind], sum) |> sort(decreasing = TRUE)
barplot(sm[1:100], ylim = c(0, 50), las = 2, cex.names = 0.2, col = 4,
        main = "# of Vendor Interactions Across Auctions")
```



3 Remove Identical Factors

```r
# relocate
dat_aucs_vend_int <- relocate(dat_aucs_vend_int, Winning_Bid, .after = Eng_Est)

# consolidate identical columns
vend_cols_log <- names(dat_aucs_vend_int) |> stringr::str_detect("Vend_")
interact_cols <- names(dat_aucs_vend_int) |> stringr::str_detect("_x_")
descr_words <- which(!vend_cols_log)[-c(1:10)]
Vend_not_int <- vend_cols_log & (!interact_cols)

stringr::str_detect(names(dat_aucs_vend_int), "Winning") |> sum()
```

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

```r
# list for removal
lst_rem <- list(dat_aucs_vend_int[, descr_words],
                dat_aucs_vend_int[, interact_cols])

# check if all went well
do.call(cbind, lapply(lst_rem, \(x){

  # consolidate
  tmp <- remove_dups_keep_name(x)
  dups <- x[, duplicated(as.list(x))]

  # print
  cat(names(tmp)[stringr::str_detect(names(tmp), "&")], "\n",
      names(dups), "\n\n\n")

  # return
  return(tmp)

})) -> new_dum
```

```
## strip&rumbl rap&rip lift&emuls cantilev&monotub cellular&modem teller&woodland erect&power&undergrou
##  rumbl rip emuls modem monotub woodland power retrofit underground
##
##
## Vend_1557A_x_Vend_102A&Vend_102A_x_Vend_905A Vend_1557A_x_Vend_1554A&Vend_905A_x_Vend_1554A Vend_155
##  Vend_1557A_x_Vend_S3082 Vend_1557A_x_Vend_617A Vend_1557A_x_Vend_866A Vend_1557A_x_Vend_1350A Vend_
```

```r
# create new df
dat_aucs_vend_int <- cbind(dat_aucs_vend_int[, 1:10], dat_aucs_vend_int[, Vend_not_int], new_dum)
```

4 Dimension reduction

4.1 Contract Description

5 Sum of Auctions Won

Since the Data does not contain any firm specific information and estimating cost functions is not feasible on firm level with the small data set at hand, we may sum up the cumulative volume of contracts won by all firms that participate in an auction. Given that firms have no other advantage than their const structure this should account for potential difference in cost structure among participating companies.

```
# find sum of auctions won grouped by year and Vendor ID
```

6 Train and Test

```
# splits
{set.seed(33)
ind <- sample(1:nrow(dat_aucs_vend_int), replace = FALSE,
              size = floor(nrow(dat_aucs_vend_int) * 0.2))
}

# train and test
dat_aucs_vend_split <- list("Train" = dat_aucs_vend_int[!(1:nrow(dat_aucs_vend_int) %in% ind), ],
                            "Test" = dat_aucs_vend_int[ind , ])

# write split
# saveRDS(dat_aucs_vend_split, "./../../Data/Bid Tab RDS/Aucs_df_feateng_split.RDS")
# saveRDS(dat_aucs_vend_int, "./../../Data/Bid Tab RDS/Aucs_df_feateng.RDS")

# write train and test separately for RF training in python
# saveRDS(dat_aucs_vend_split[["Train"]],
#         "./../../Data/Bid Tab RDS/Colab Transfer/Aucs_df_feateng_train.RDS")
# saveRDS(dat_aucs_vend_split[["Test"]],
#         "./../../Data/Bid Tab RDS/Colab Transfer/Aucs_df_feateng_test.RDS")
```