EDA

2022-12-13

Contents

Libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## 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
```

```
library(ggthemes)
library(VIM)
```

```
## Loading required package: colorspace
## Loading required package: grid
## VIM is ready to use.
##
## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues
##
## Attaching package: 'VIM'
##
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
library(dplyr)
library(readr)
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
```

```
library(tibble)
library(tidyr)
library(stringr)
library(forcats)
library(rlang)
```

```
##
## Attaching package: 'rlang'
##
## The following object is masked from 'package:data.table':
##
##     :=
##
## The following objects are masked from 'package:purrr':
##
##     %@%, as_function, flatten, flatten_chr, flatten_dbl, flatten_int,
##     flatten_lgl, flatten_raw, invoke, splice
```

Data Import

```
dtrain <- read_csv('/Users/xinyu/Downloads/Data/train.csv')
```

```
## Rows: 595212 Columns: 59
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (59): id, target, ps_ind_01, ps_ind_02_cat, ps_ind_03, ps_ind_04_cat, ps...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Overview of Data

```
## [1] "Training data size in RAM:"
```

```
## 268 Mb
```

```
glimpse(dtrain)
```

```
## Rows: 595,212
## Columns: 59
## $ id             <dbl> 7, 9, 13, 16, 17, 19, 20, 22, 26, 28, 34, 35, 36, 43, 4~
## $ target         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_ind_01      <dbl> 2, 1, 5, 0, 0, 5, 2, 5, 5, 1, 5, 2, 2, 1, 5, 5, 1, 5, 5~
## $ ps_ind_02_cat  <dbl> 2, 1, 4, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1~
## $ ps_ind_03      <dbl> 5, 7, 9, 2, 0, 4, 3, 4, 3, 2, 2, 3, 1, 3, 11, 3, 1, 6, ~
## $ ps_ind_04_cat  <dbl> 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0~
## $ ps_ind_05_cat  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0~
## $ ps_ind_06_bin  <dbl> 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1~
## $ ps_ind_07_bin  <dbl> 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0~
## $ ps_ind_08_bin  <dbl> 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0~
## $ ps_ind_09_bin  <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0~
## $ ps_ind_10_bin  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_ind_11_bin  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_ind_12_bin  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_ind_13_bin  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_ind_14      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

```
## $ ps_ind_15      <dbl> 11, 3, 12, 8, 9, 6, 8, 13, 6, 4, 3, 9, 10, 12, 10, 5, 9~
## $ ps_ind_16_bin  <dbl> 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1~
## $ ps_ind_17_bin  <dbl> 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_ind_18_bin  <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0~
## $ ps_reg_01      <dbl> 0.7, 0.8, 0.0, 0.9, 0.7, 0.9, 0.6, 0.7, 0.9, 0.9, 0.5, ~
## $ ps_reg_02      <dbl> 0.2, 0.4, 0.0, 0.2, 0.6, 1.8, 0.1, 0.4, 0.7, 1.4, 0.4, ~
## $ ps_reg_03      <dbl> 0.7180703, 0.7660777, -1.0000000, 0.5809475, 0.8407586,~
## $ ps_car_01_cat  <dbl> 10, 11, 7, 7, 11, 10, 6, 11, 10, 11, 11, 11, 6, 9, 11, ~
## $ ps_car_02_cat  <dbl> 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1~
## $ ps_car_03_cat  <dbl> -1, -1, -1, 0, -1, -1, -1, 0, -1, 0, -1, -1, -1, 0, -1,~
## $ ps_car_04_cat  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 8, 0, 0, 0, 0, 9, 1, 0, 0~
## $ ps_car_05_cat  <dbl> 1, -1, -1, 1, -1, 0, 1, 0, 1, 0, -1, -1, -1, 1, -1, 1, ~
## $ ps_car_06_cat  <dbl> 4, 11, 14, 11, 14, 14, 11, 11, 14, 14, 13, 11, 11, 6, 1~
## $ ps_car_07_cat  <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ ps_car_08_cat  <dbl> 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1~
## $ ps_car_09_cat  <dbl> 0, 2, 2, 3, 2, 0, 0, 2, 0, 2, 2, 0, 2, 2, 2, 0, 0, 2, 2~
## $ ps_car_10_cat  <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ ps_car_11_cat  <dbl> 12, 19, 60, 104, 82, 104, 99, 30, 68, 104, 20, 36, 101,~
## $ ps_car_11      <dbl> 2, 3, 1, 1, 3, 2, 2, 3, 3, 2, 3, 3, 3, 3, 1, 2, 3, 2, 3~
## $ ps_car_12      <dbl> 0.4000000, 0.3162278, 0.3162278, 0.3741657, 0.3160696, ~
## $ ps_car_13      <dbl> 0.8836789, 0.6188165, 0.6415857, 0.5429488, 0.5658315, ~
## $ ps_car_14      <dbl> 0.3708099, 0.3887158, 0.3472751, 0.2949576, 0.3651027, ~
## $ ps_car_15      <dbl> 3.605551, 2.449490, 3.316625, 2.000000, 2.000000, 3.000~
## $ ps_calc_01     <dbl> 0.6, 0.3, 0.5, 0.6, 0.4, 0.7, 0.2, 0.1, 0.9, 0.7, 0.8, ~
## $ ps_calc_02     <dbl> 0.5, 0.1, 0.7, 0.9, 0.6, 0.8, 0.6, 0.5, 0.8, 0.8, 0.1, ~
## $ ps_calc_03     <dbl> 0.2, 0.3, 0.1, 0.1, 0.0, 0.4, 0.5, 0.1, 0.6, 0.8, 0.0, ~
## $ ps_calc_04     <dbl> 3, 2, 2, 2, 2, 3, 2, 1, 3, 2, 2, 2, 4, 2, 3, 2, 2, 1, 3~
## $ ps_calc_05     <dbl> 1, 1, 2, 4, 2, 1, 2, 2, 1, 2, 3, 2, 1, 1, 1, 1, 1, 3, 1~
## $ ps_calc_06     <dbl> 10, 9, 9, 7, 6, 8, 8, 7, 7, 8, 8, 8, 8, 10, 8, 9, 10, 8~
## $ ps_calc_07     <dbl> 1, 5, 1, 1, 3, 2, 1, 1, 3, 2, 2, 2, 4, 1, 2, 5, 6, 2, 3~
## $ ps_calc_08     <dbl> 10, 8, 8, 8, 10, 11, 8, 6, 9, 9, 9, 10, 11, 8, 6, 10, 7~
## $ ps_calc_09     <dbl> 1, 1, 2, 4, 2, 3, 3, 1, 4, 1, 4, 1, 1, 3, 3, 2, 3, 1, 2~
## $ ps_calc_10     <dbl> 5, 7, 7, 2, 12, 8, 10, 13, 11, 11, 7, 8, 9, 8, 12, 13, ~
## $ ps_calc_11     <dbl> 9, 3, 4, 2, 3, 4, 3, 7, 4, 3, 6, 9, 6, 2, 4, 5, 3, 9, 3~
## $ ps_calc_12     <dbl> 1, 1, 2, 2, 1, 2, 0, 1, 2, 5, 3, 2, 3, 0, 1, 2, 3, 1, 1~
## $ ps_calc_13     <dbl> 5, 1, 7, 4, 1, 0, 0, 3, 1, 0, 3, 1, 3, 4, 3, 6, 1, 3, 6~
## $ ps_calc_14     <dbl> 8, 9, 7, 9, 3, 9, 10, 6, 5, 6, 6, 10, 8, 3, 9, 7, 8, 9,~
## $ ps_calc_15_bin <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ps_calc_16_bin <dbl> 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0~
## $ ps_calc_17_bin <dbl> 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0~
## $ ps_calc_18_bin <dbl> 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1~
## $ ps_calc_19_bin <dbl> 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0~
## $ ps_calc_20_bin <dbl> 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0~
```

There are just under 600,000 observations and 59 columns including the `id` and `target` column. The `target` column is 1 if the customer filed a claim and 0 if not.

We find:

- There are lots of features here. In total, our *training* data has 59 variables, including *id* and *target*. In some of them we already see a number of NAs.
- Based on he data description, the names of the features indicate whether they are binary (*bin*) or categorical (*cat*) variables. Everything else is continuous or ordinal.

- A few groups are defined, and features that belong to these groups include paterns in the name (ind, reg, car, calc). The names of the variables indicate certain properties: "Ind" is related to individual or driver, "reg" is related to region, "car" is related to car itself and "calc" is an calculated feature.
- Note, that there is a *ps_car_11* as well as a *ps_car_11_cat*. This is the only occasion where the numbering per group is neither consecutive nor unique. Probably a typo in the script that created the variable names.
- The value that is subject of prediction is in the "target" column. This one indicates whether or not a claim was filed for that insured person. "id" is a data input ordinal number.
- A missing value is indicated by -1.
- The features are anonymised.

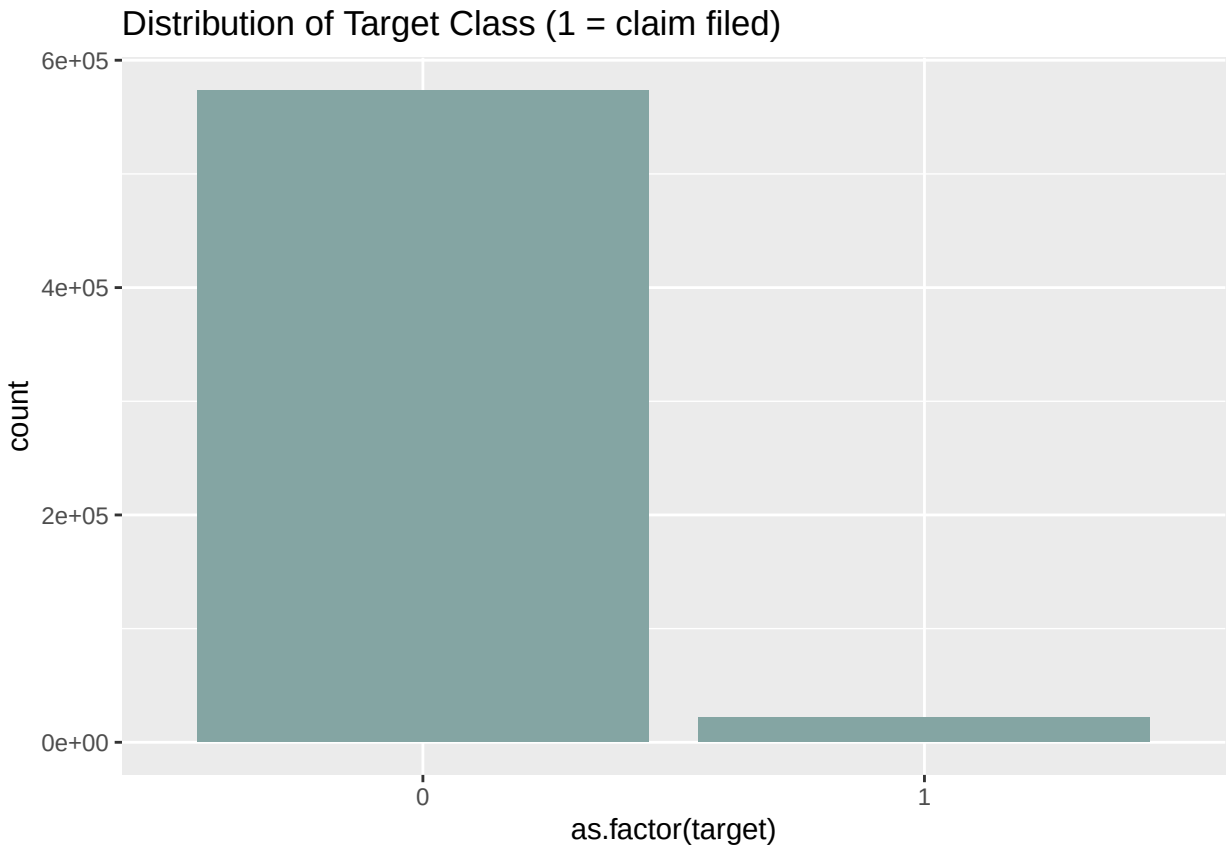
From the competitions page:

"In the train and test data, features that belong to similar groupings are tagged as such in the feature names (e.g., ind, reg, car, calc). In addition, feature names include the postfix bin to indicate binary features and cat to indicate categorical features. Features without these designations are either continuous or ordinal. Values of -1 indicate that the feature was missing from the observation. The target columns signifies whether or not a claim was filed for that policy holder"

Target Feature Analysis

First let's look at the target variable, which is the label we want to predict. We found earlier that there were no missing values for the target. We want to find out whether a claim has been filed ("1") or not ("0"): How many positives are there?

```
ggplot(data = dtrain, aes(x = as.factor(target))) +
    geom_bar(fill = '#84a5a3') +
    labs(title = 'Distribution of Target Class (1 = claim filed)')
```



We find there are much more 0's than 1's:

- Most cases have no filed claim:

```
tab <- table(dtrain$target)
print(tab)
```

```
##
##      0      1
## 573518  21694
```

```
print(paste0(round((tab[2] / tab[1]) * 100, 2), "%", " of customers in the train set filed a claim."))
```

```
## [1] "3.78% of customers in the train set filed a claim."
```

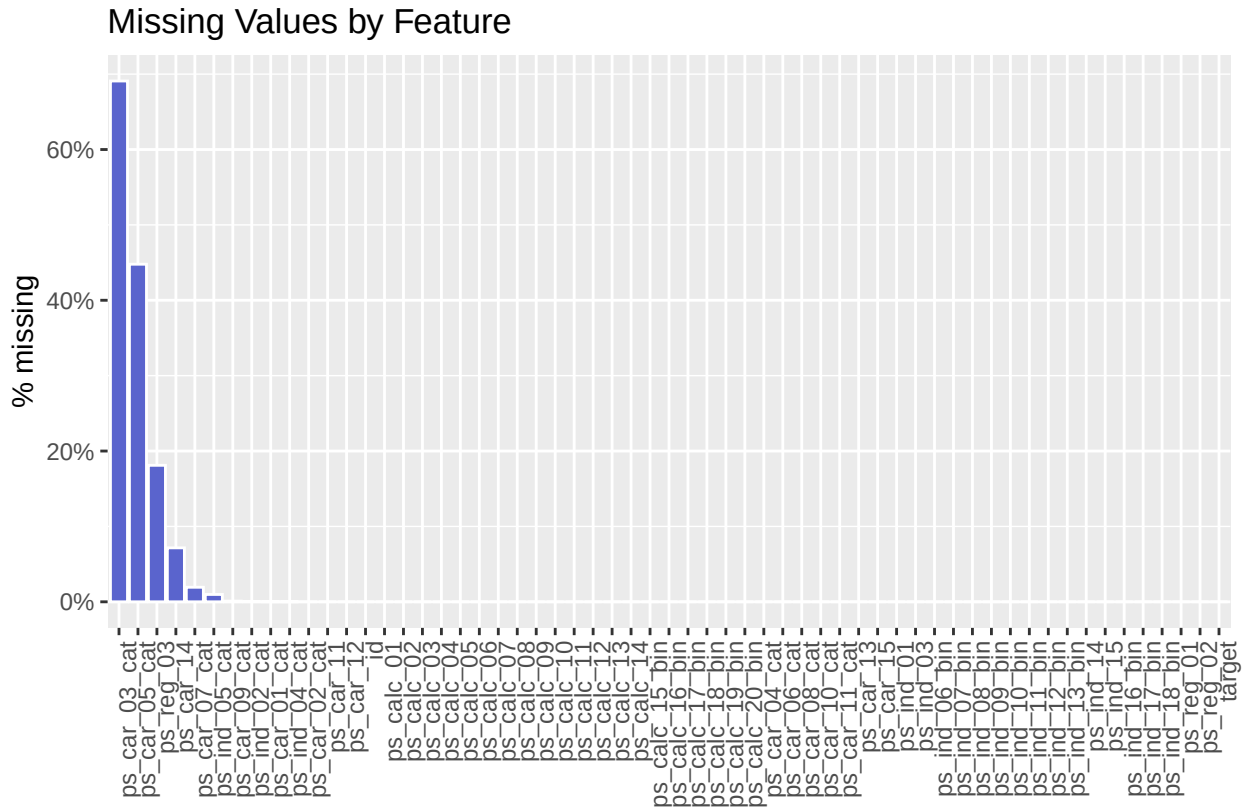
With less than 4% of policy holders filing a claim, so the problem is heavily imbalanced.

Feature Exploration

Missing Data

Now we want to learn the percentage of NA that each variable has and understand the distribution of these NA.

```
data.frame(feature = names(dtrain),
           per_miss = map_dbl(dtrain, function(x) { sum(x == - 1) / length(x) })) %>%
    ggplot(aes(x = reorder(feature, -per_miss), y = per_miss)) +
    geom_bar(stat = 'identity', color = 'white', fill = '#5a64cd') +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    labs(x = '', y = '% missing', title = 'Missing Values by Feature') +
    scale_y_continuous(labels = scales::percent)
```



Most variables have no missing data. `ps_car_03_cat` has almost 70% of its values missing. `ps_car_05_cat` has about 45% missing. `ps_reg_03` has around 18% missing. The remaining three have only a small percent missing data.

The following command gives the sum of missing values in the whole training data frame column wise:

```
train <- as.tibble(fread('/Users/xinyu/Downloads/Data/train.csv', na.strings=c("-1","-1.0")))
```

```
## Warning: ‘as.tibble()‘ was deprecated in tibble 2.0.0.
## i Please use ‘as_tibble()‘ instead.
## i The signature and semantics have changed, see ‘?as_tibble‘.
```

```
colSums(is.na.data.frame(train))
```

```
##             id         target      ps_ind_01  ps_ind_02_cat      ps_ind_03
##              0              0              0            216              0
##  ps_ind_04_cat  ps_ind_05_cat  ps_ind_06_bin  ps_ind_07_bin  ps_ind_08_bin
```

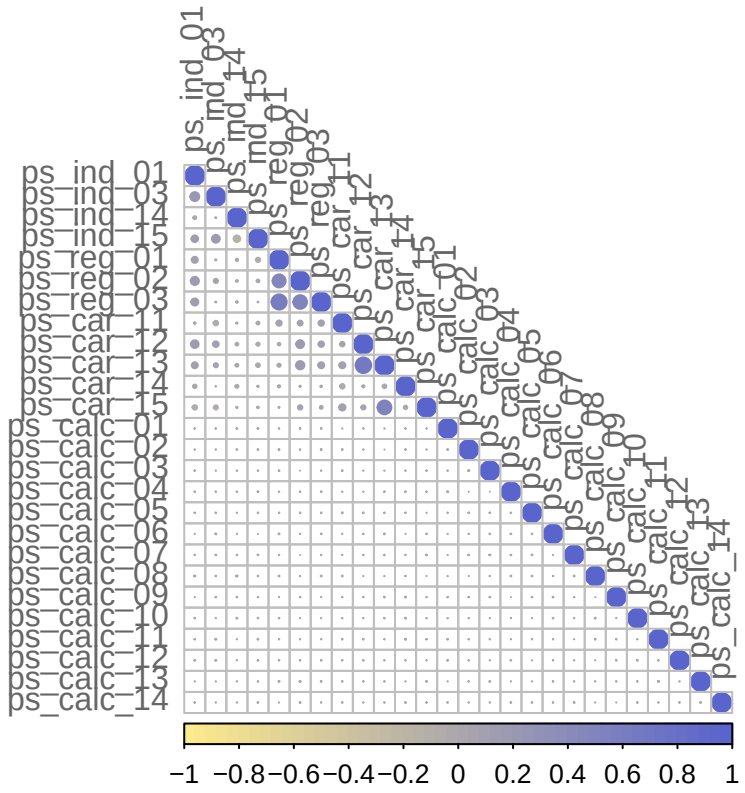
```
##             83           5809              0              0              0
##  ps_ind_09_bin  ps_ind_10_bin  ps_ind_11_bin  ps_ind_12_bin  ps_ind_13_bin
##              0              0              0              0              0
##      ps_ind_14      ps_ind_15  ps_ind_16_bin  ps_ind_17_bin  ps_ind_18_bin
##              0              0              0              0              0
##      ps_reg_01      ps_reg_02      ps_reg_03  ps_car_01_cat  ps_car_02_cat
##              0              0         107772            107              5
##  ps_car_03_cat  ps_car_04_cat  ps_car_05_cat  ps_car_06_cat  ps_car_07_cat
##         411231              0         266551              0          11489
##  ps_car_08_cat  ps_car_09_cat  ps_car_10_cat  ps_car_11_cat      ps_car_11
##              0            569              0              0              5
##      ps_car_12      ps_car_13      ps_car_14      ps_car_15     ps_calc_01
##              1              0          42620              0              0
##     ps_calc_02     ps_calc_03     ps_calc_04     ps_calc_05     ps_calc_06
##              0              0              0              0              0
##     ps_calc_07     ps_calc_08     ps_calc_09     ps_calc_10     ps_calc_11
##              0              0              0              0              0
##     ps_calc_12     ps_calc_13     ps_calc_14 ps_calc_15_bin ps_calc_16_bin
##              0              0              0              0              0
## ps_calc_17_bin ps_calc_18_bin ps_calc_19_bin ps_calc_20_bin
##              0              0              0              0
```

We can see that `ps_car_03_cat` has the largest amount of missing values, which is 411231. And `ps_car_05_cat` has the second largest number of missing values, which is 266551. `ps_reg_03` has 107772 missing values.

Correlations

We begin with a correlation matrix plot as a first comprehensive overview of our multi-parameter space.

```
# Get features names that are not binary or categorical
cont_vars <- names(dtrain)[!grepl("_cat|_bin", names(dtrain))]

corrplot(cor(dtrain[, cont_vars][3:length(cont_vars)]),
         type = 'lower',
         col = colorRampPalette(c('#feeb8c', '#5a64cd'))(50),
         tl.col = 'grey40',
         mar = c(0,0,1,0),
         title = 'Correlation Matrix of Continuous Features')
```



The group of `ind` variables show some correlation amongst themselves as well as with some of the `reg` and `car` features. The `calc` features do not appear to be correlated with anything, including the targets. So we would decide to drop off the `calc` variables.

Feature Importance

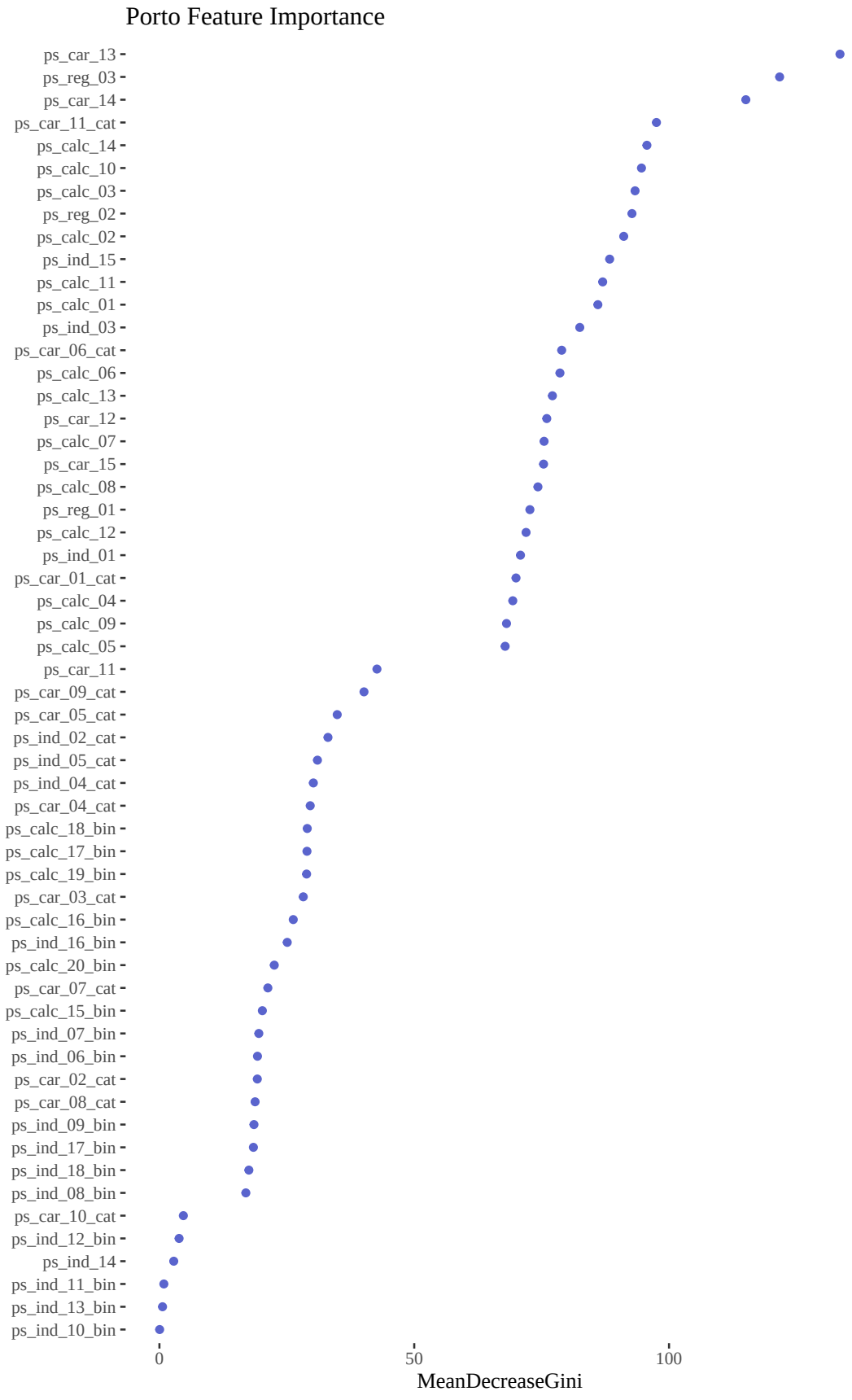
We will check which features are the most important for our model. Here we run a very quick random forest model on the raw training data so that we can look at feature importance.

We first run the random forest then we can extract the feature importances and plot them.

```
library(caret)

set.seed(12)
sample_index <- sample(c(TRUE, FALSE), size = nrow(dtrain), replace = TRUE, prob = c(0.1, 0.9))

x_train <- as.matrix(dtrain[sample_index, 3:59])
y_train <- as.factor(dtrain$target[sample_index])


rfmod <- train(x = x_train,
               y = y_train,
               method = 'rf',
               ntree = 20,
               trControl = trainControl(method = 'boot', number = 1))
```

this is a rough attempt. We have not converted the categorical features into factors because it would dramatically increase the run time of the model. And so the importance of the categorical features in the plot may not be accurately represented.

```
importance(rfmod$finalModel) %>%
    as.data.frame() %>%
    rownames_to_column(var = 'Feature') %>%
    ggplot(aes(x = reorder(Feature, MeanDecreaseGini), y = MeanDecreaseGini)) +
    geom_point(color = '#5a64cd') +
    coord_flip() +
    theme_tufte() +
    labs(x = '', title = 'Porto Feature Importance')
```


Porto Feature Importance
ps_car_13
ps_reg_03
ps_car_14
ps_car_11_cat
ps_calc_14
ps_calc_10
ps_calc_03
ps_reg_02
ps_calc_02
ps_ind_15
ps_calc_11
ps_calc_01
ps_ind_03
ps_car_06_cat
ps_calc_06
ps_calc_13
ps_car_12
ps_calc_07
ps_car_15
ps_calc_08
ps_reg_01
ps_calc_12
ps_ind_01
ps_car_01_cat
ps_calc_04
ps_calc_09
ps_calc_05
ps_car_11
ps_car_09_cat
ps_car_05_cat
ps_ind_02_cat
ps_ind_05_cat
ps_ind_04_cat
ps_car_04_cat
ps_calc_18_bin
ps_calc_17_bin
ps_calc_19_bin
ps_car_03_cat
ps_calc_16_bin
ps_ind_16_bin
ps_calc_20_bin
ps_car_07_cat
ps_calc_15_bin
ps_ind_07_bin
ps_ind_06_bin
ps_car_02_cat
ps_car_08_cat
ps_ind_09_bin
ps_ind_17_bin
ps_ind_18_bin
ps_ind_08_bin
ps_car_10_cat
ps_ind_12_bin
ps_ind_14
ps_ind_11_bin
ps_ind_13_bin
ps_ind_10_bin
0
50
100
MeanDecreaseGini