Cleaning, Preprocessing and Exploring DATA for Safe driver prediction
CS685A: DATA MINING Project Report

Jayant Gangwar
180327, AE
jayantg@iitk.ac.in

Parth Somani
180503, CSE
psomani@iitk.ac.in

Shivam Tulsyan
180723, EE
shivtuls@iitk.ac.in

1. INTRODUCTION

This project aims at getting a good insight in the data for the Safe Driver Prediction. Besides that, it focuses mainly on preparing your data for modeling. The project can be classified into these following main sections:

- Visual inspection of data
- Defining the metadata
- Descriptive statistics
- Handling imbalanced classes
- Checking Data quality
- Exploratory data visualization
- Feature engineering
- Feature selection
- Feature scaling

2. DATA DESCRIPTION

The following is an excerpt of the data description:

- Features belonging to similar groupings are tagged accordingly in the feature titles, i.e., `ind, reg, car, calc...`
- This titles include a postfix either `bin` to indicate binary features and `cat` to indicate categorical features.
- Features without any postfix are either ordinal or continuous.
- Entries missing in the original observation are indicated by -1.
- The target column signifies if that policy holder filed a claim or not.

The training-data set has 59 features (including target) and 5,95,212 rows in total. The number of features in the test set is 58, which is correct as we have to predict the remaining column, i.e., target. The data has 10 float columns and 49 integer columns. No null data found as the missing values are replaced by -1. Moreover, the data did not feature any duplicate rows.

3. DATA MANAGEMENT AND ANALYSIS

For better data management, the features are categorized on multiple grounds. This helps to select specific variables for tasks like analysis, visualization, modeling, ...
Following are the grounds and categories used

- `role`: input, ID, target
- `level`: nominal, interval, ordinal, binary
- `keep`: True, False
- `data_type`: integer, float

3.1 Analysing different categories by level

1. `Interval features`: range is comparatively small, seems like some transformation is already applied on the data beforehand.
   - `reg features`: only ps_reg_03 has missing values, the range differs across variables and hence needed to be scaled.
   - `car features`: ps_car_12 ans ps_car_15 have missing values, the range differs again and need scaling.
   - `cal features`: no missing values, all features seems to have similar distributions, could be some ratio as the maximum is `0.9`.
2. `Ordinal features`
   - `ps_car_11`: only feature with missing values.
   - Need scaling to deal features with different ranges.
3. `Binary features`
   - From the means of the feature vector, we can conclude that the values are zero in most cases.
   - A priori in the train data is `3.645%`, which is highly imbalanced.

3.2 Handling Imbalanced Classes

As the proportion of records with target=1 is far less than target=0. This can lead to a model that has great accuracy but does have any added value in practice. Two possible general strategies to deal with this problem are:

- oversampling records with target=1
- under-sampling records with target=0

As we have a rather large training set, we can go for under-sampling. After this handling the dataset has target=0 9times, it has target=1.

3.3 Handling Data Quality

Following are the measures taken to improve data quality and handle missing values.

- ps_reg_03 (continuous) has missing values for 18% of all records. Replacing these by respective `mean` of the feature column.
- ps_car_12 (continuous) has only 1 records with missing value. Replacing these by respective `mean` of the feature column.
- ps_car_14 (continuous) has missing values for 7% of all records. Replacing these by respective `mean` of the feature column.
- ps_car_11 (ordinal) has only 5 records with missing values. Replacing these by respective `mode` of the feature column.
- ps_car_03_cat and ps_car_05_cat have a large proportion of records with missing values. Removing these features from the dataset altogether.
- For the other categorical features with missing values, we can leave the missing value -1 as such.

3.4 Exploratory Data Visualization

After looking into the categorical features and the proportion of customers with target = 1, we can see from the features with missing values, it is a good idea to keep the missing values as a separate category value, instead of replacing them by the mode for instance. The customers with a missing value appear to have a much higher (in some cases much lower) probability to ask for an insurance claim.
For checking the correlations between interval features, heatmap is a good way to visualize the correlation between features Following features are found to have strong correlations:

- ps_reg_02 and ps_reg_03 (0.7)
- ps_car_12 and ps_car13 (0.67)
- ps_car_12 and ps_car14 (0.58)
- ps_car_13 and ps_car15 (0.67)

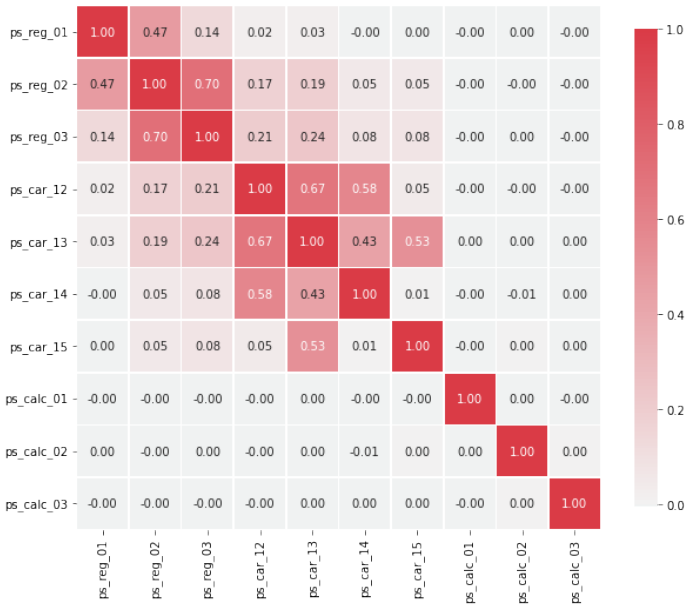

Figure 1: Heatmap of interval features

We could use a pairplot to visualize the relationship between the features. But because the heatmap already showed the limited number of correlated variables, we'll look at each of the highly correlated variables separately.

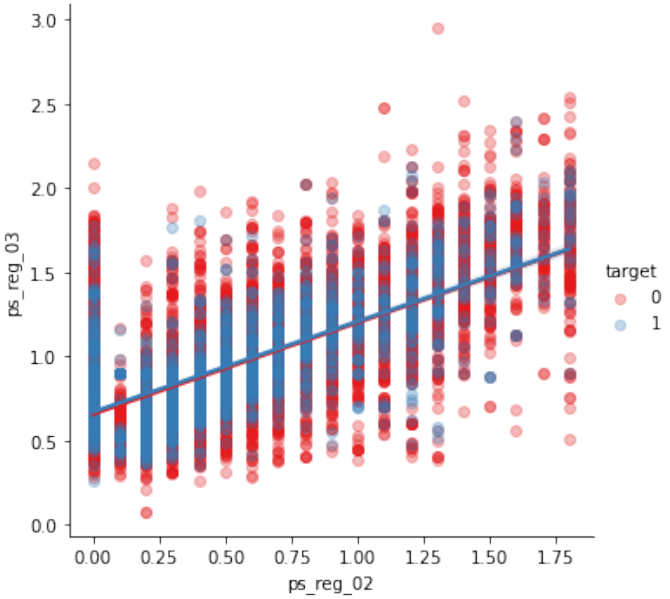

Figure 2: ps_reg_03 and ps_reg_02

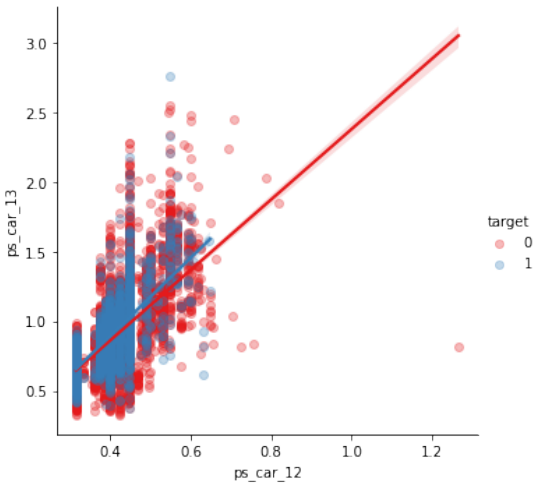

Figure 3: ps_car_13 and ps_car_12

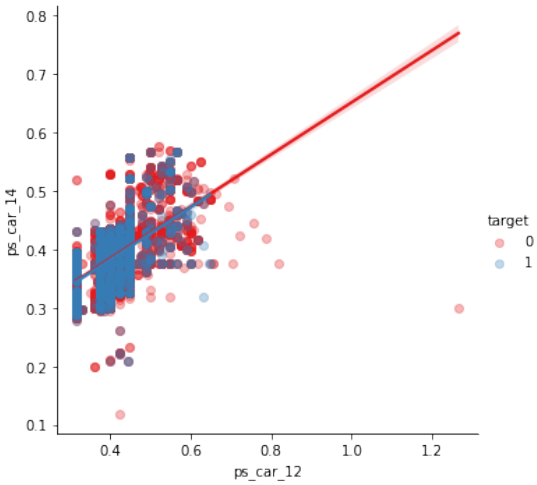

Figure 4: ps_car_14 and ps_car_12

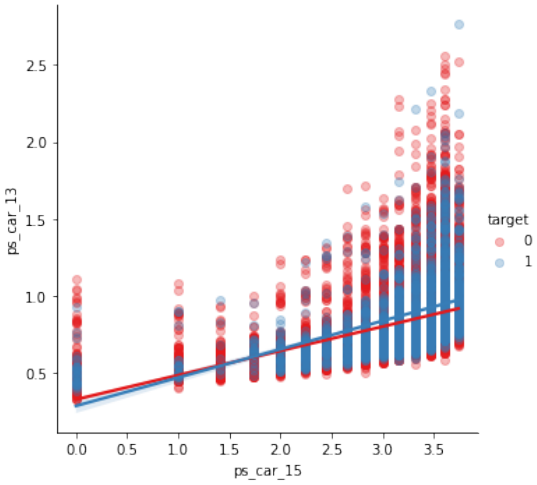

Figure 5: ps_car_13 and ps_car_15

As the regression line shows, there is a linear relationship between these variables. Thanks to the hue parameter we can see that the regression lines for target=0 and target=1 are the same.
For the ordinal variables we do not see many correlations. We could, on the other hand, look at how the distributions are when grouping by the target value.

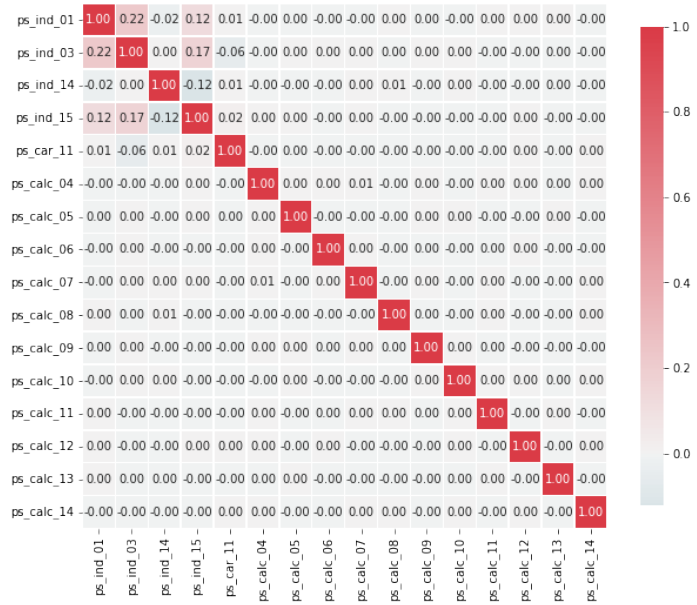

Figure 6: Heatmap of ordinal features

3.5 Feature-Selection

We can start with removing features with no or a very low variance. Using Sklearn library, which has a handy method to do that: `VarianceThreshold`. By default it removes features with zero variance. This will not be applicable for us as we saw there are no zero-variance variables in the previous steps. But if we would remove features with less than 1% variance, we would remove 31 variables. We would lose rather many variables if we would select based on variance. Because we do not have so many variables, we'll let the classifier chose. For data sets with many more variables this could reduce the processing time. Using Sklearn library which also comes with other feature selection methods. One of these methods is `SelectFromModel` in which we let another classifier select the best features and continue with these. Here we'll base, feature selection on the feature importances of a random forest. With Sklearn's `SelectFromModel` you can then specify how many variables you want to keep. You can set a threshold on the level of feature importance manually. But we'll simply select the top 50% best variables. With `SelectFromModel` we can specify which prefit classifier to use and what the threshold is for the feature importances. With the get_support method we can then limit the number of variables in the train data.

3.6 Feature-Scaling

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. We can apply standard scaling to the training data. Some classifiers perform better when this is done.
We used Feature Standardization method which makes the values of each feature in the data have zero-mean (when subtracting the mean in the numerator) and unit-variance.

$$x' = \frac{x - \mu}{\sigma}$$

4. CONCLUSION

After exploring the data thoroughly, it is evident that most of the collected big data has a lot of inconsistencies and biases. Without proper manipulation or scaling, the final model is bound to give results not up-to mark.