




BOAZ kaggle

Porto Seguro's Safe Driver Prediction Part 1 - EDA

BOAZ MLDL

김아영 / 성민석 / 남궁찬

Porto Seguro's Safe Driver Prediction

 Featured Prediction Competition

Porto Seguro's Safe Driver Prediction

Predict if a driver will file an insurance claim next year.

\$25,000

Prize Money



Porto Seguro · 5,169 teams · a year ago

Index

1. Defining the Problem Statement
2. Evaluation
3. Data / File Description
4. Train / Test Data
5. Descriptive Statistics
6. Handling Imbalanced Classes
7. Data Quality Check
8. Data Visualization

Defining the Problem Statement

Defining the Problem Statement

Nothing ruins the thrill of buying a brand new car more quickly than seeing your new insurance bill. The sting's even more painful when you know you're a good driver. It doesn't seem fair that you have to pay so much if you've been cautious on the road for years.

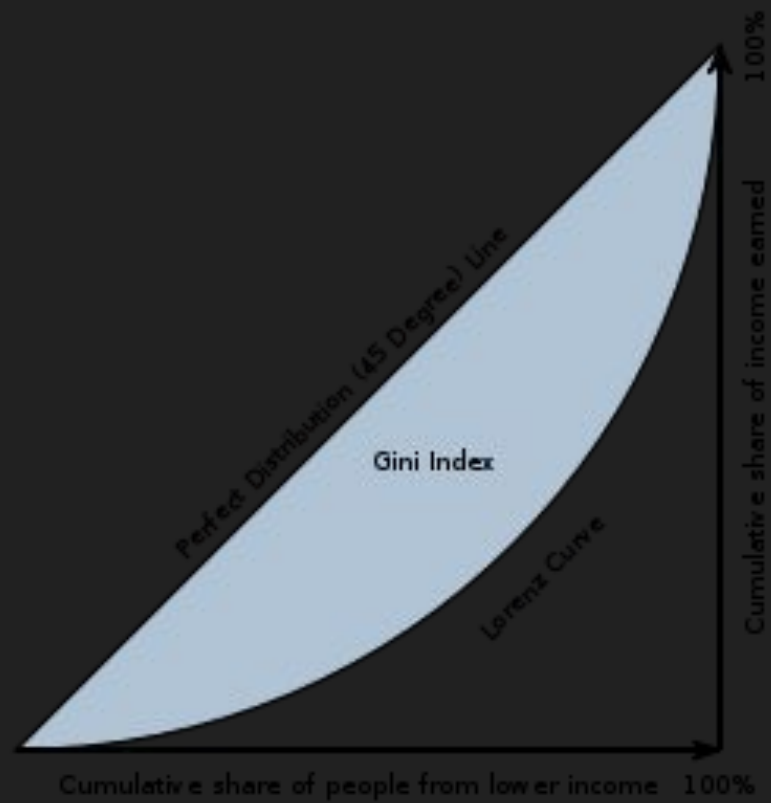
Porto Seguro, one of Brazil's largest auto and homeowner insurance companies, completely agrees. Inaccuracies in car insurance company's claim predictions raise the cost of insurance for good drivers and reduce the price for bad ones.

Defining the Problem Statement

In this competition, you're challenged to build a model that predicts the probability that a driver will initiate an auto insurance claim in the next year. While Porto Seguro has used machine learning for the past 20 years, they're looking to Kaggle's machine learning community to explore new, more powerful methods. A more accurate prediction will allow them to further tailor their prices, and hopefully make auto insurance coverage more accessible to more drivers.

Evaluation

Gini



Normalized Gini Coefficient

Submissions are evaluated using the **Normalized Gini Coefficient**.

During scoring, observations are **sorted from the largest to the smallest** predictions. Predictions are only used for ordering observations; therefore, the relative magnitude of the predictions are not used during scoring. The scoring algorithm then compares the cumulative proportion of positive class observations to a theoretical uniform proportion.

Gini Coefficient

The **Gini Coefficient** ranges from approximately 0 for random guessing, to approximately 0.5 for a perfect score. The theoretical maximum for the discrete calculation is $(1 - \text{frac_pos}) / 2$.

The Normalized Gini Coefficient adjusts the score by the theoretical maximum so that the maximum score is 1.

The code to calculate Normalized Gini Coefficient in a number of different languages can be found in this forum thread.

Data / File Description

Data Description

In this competition, you will predict the probability that an auto insurance policy holder files a claim.

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** column signifies whether or not a claim was filed for that policy holder.

File Description

- train.csv : contains the training data, where each row corresponds to a policy holder, and the target columns signifies that a claim was filed.
- test.csv : contains the test data.
- sample_submission.csv is submission file showing the correct format.

Train / Test Data

Shape of Train / Test Data

```
1 train.head()
```

	id	target	ps_ind_01	ps_ind_02_cat	ps_ind_03	ps_ind_04_cat	ps_ind_05_cat	ps_ind_06_bin	ps_ind_07_bin	ps_ind_08_bin	ps_ind_09_bin	ps_ind_10_bin	ps_ind_11_bin
0	7	0	2	2	5	1	0	0	1	0	0	0	0
1	9	0	1	1	7	0	0	0	0	1	0	0	0
2	13	0	5	4	9	1	0	0	0	1	0	0	0
3	16	0	0	1	2	0	0	1	0	0	0	0	0
4	17	0	0	2	0	1	0	1	0	0	0	0	0

```
[ ] 1 train.shape
```

```
➜ (595212, 59)
```

```
[ ] 1 test.shape
```

```
➜ (892816, 58)
```

Encryption

```
1 train.head()
```

	id	target	ps_ind_01	ps_ind_02_cat	ps_ind_03	ps_ind_04_cat	ps_ind_05_cat	ps_ind_06_bin	ps_ind_07_bin	ps_ind_08_bin	ps_ind_09_bin	ps_ind_10_bin	ps_ind_11_bin
0	7	0	2	2	5	1	0	0	1	0	0	0	0
1	9	0	1	1	7	0	0	0	0	1	0	0	0
2	13	0	5	4	9	1	0	0	0	1	0	0	0
3	16	0	0	1	2	0	0	1	0	0	0	0	0
4	17	0	0	2	0	1	0	1	0	0	0	0	0

Encryption

```
[ ] 1 train.shape
```

➞ (595212, 59)

```
[ ] 1 test.shape
```

➞ (892816, 58)

Type of Features

```
[ ] 1 train.info()
```

```
↳ ps_ind_05_cat    595212 non-null int64
   ps_ind_06_bin    595212 non-null int64
   ps_ind_07_bin    595212 non-null int64
   ps_ind_08_bin    595212 non-null int64
   ps_ind_09_bin    595212 non-null int64
   ps_ind_10_bin    595212 non-null int64
   ps_ind_11_bin    595212 non-null int64
   ps_ind_12_bin    595212 non-null int64
   ps_ind_13_bin    595212 non-null int64
   ps_ind_14        595212 non-null int64
   ps_ind_15        595212 non-null int64
   ps_ind_16_bin    595212 non-null int64
   ps_ind_17_bin    595212 non-null int64
   ps_ind_18_bin    595212 non-null int64
   ps_reg_01        595212 non-null float64
   ps_reg_02        595212 non-null float64
   ps_reg_03        595212 non-null float64
   ps_car_01_cat    595212 non-null int64
   ps_car_02_cat    595212 non-null int64
   ps_car_03_cat    595212 non-null int64
   ps_car_04_cat    595212 non-null int64
   ps_car_05_cat    595212 non-null int64
   ps_car_06_cat    595212 non-null int64
   ps_car_07_cat    595212 non-null int64
   ps_car_08_cat    595212 non-null int64
```

```
ps_car_09_cat    595212 non-null int64
ps_car_10_cat    595212 non-null int64
ps_car_11_cat    595212 non-null int64
ps_car_11        595212 non-null int64
ps_car_12        595212 non-null float64
ps_car_13        595212 non-null float64
ps_car_14        595212 non-null float64
ps_car_15        595212 non-null float64
ps_calc_01       595212 non-null float64
ps_calc_02       595212 non-null float64
ps_calc_03       595212 non-null float64
ps_calc_04       595212 non-null int64
ps_calc_05       595212 non-null int64
ps_calc_06       595212 non-null int64
ps_calc_07       595212 non-null int64
ps_calc_08       595212 non-null int64
ps_calc_09       595212 non-null int64
ps_calc_10       595212 non-null int64
ps_calc_11       595212 non-null int64
ps_calc_12       595212 non-null int64
ps_calc_13       595212 non-null int64
ps_calc_14       595212 non-null int64
ps_calc_15_bin   595212 non-null int64
ps_calc_16_bin   595212 non-null int64
ps_calc_17_bin   595212 non-null int64
```

Metadata

id	id	nominal	False	int64
target	target	binary	True	int64
ps_ind_01	input	ordinal	True	int64
ps_ind_02_cat	input	nominal	True	int64
ps_ind_03	input	ordinal	True	int64
ps_ind_04_cat	input	nominal	True	int64
ps_ind_05_cat	input	nominal	True	int64
ps_ind_06_bin	input	binary	True	int64
ps_ind_07_bin	input	binary	True	int64
ps_ind_08_bin	input	binary	True	int64
ps_ind_09_bin	input	binary	True	int64
ps_ind_10_bin	input	binary	True	int64
ps_ind_11_bin	input	binary	True	int64
ps_ind_12_bin	input	binary	True	int64
ps_ind_13_bin	input	binary	True	int64
ps_ind_14	input	ordinal	True	int64
ps_ind_15	input	ordinal	True	int64
ps_ind_16_bin	input	binary	True	int64
ps_ind_17_bin	input	binary	True	int64
ps_ind_18_bin	input	binary	True	int64
ps_reg_01	input	interval	True	float64
ps_reg_02	input	interval	True	float64
ps_reg_03	input	interval	True	float64
ps_car_01_cat	input	nominal	True	int64
ps_car_02_cat	input	nominal	True	int64
ps_car_03_cat	input	nominal	True	int64

ps_car_03_cat	input	nominal	True	int64
ps_car_04_cat	input	nominal	True	int64
ps_car_05_cat	input	nominal	True	int64
ps_car_06_cat	input	nominal	True	int64
ps_car_07_cat	input	nominal	True	int64
ps_car_08_cat	input	nominal	True	int64
ps_car_09_cat	input	nominal	True	int64
ps_car_10_cat	input	nominal	True	int64
ps_car_11_cat	input	nominal	True	int64
ps_car_11	input	ordinal	True	int64
ps_car_12	input	interval	True	float64
ps_car_13	input	interval	True	float64
ps_car_14	input	interval	True	float64
ps_car_15	input	interval	True	float64
ps_calc_01	input	interval	True	float64
ps_calc_02	input	interval	True	float64
ps_calc_03	input	interval	True	float64
ps_calc_04	input	ordinal	True	int64
ps_calc_05	input	ordinal	True	int64
ps_calc_06	input	ordinal	True	int64
ps_calc_07	input	ordinal	True	int64
ps_calc_08	input	ordinal	True	int64
ps_calc_09	input	ordinal	True	int64
ps_calc_10	input	ordinal	True	int64
ps_calc_11	input	ordinal	True	int64
ps_calc_12	input	ordinal	True	int64
ps_calc_13	input	ordinal	True	int64

ps_calc_14	input	ordinal	True	int64
ps_calc_15_bin	input	binary	True	int64
ps_calc_16_bin	input	binary	True	int64
ps_calc_17_bin	input	binary	True	int64
ps_calc_18_bin	input	binary	True	int64
ps_calc_19_bin	input	binary	True	int64
ps_calc_20_bin	input	binary	True	int64

Metadata

	role	level	count
0	id	nominal	1
1	input	binary	17
2	input	interval	10
3	input	nominal	14
4	input	ordinal	16
5	target	binary	1

Metadata

	role	level	count
0	id	nominal	1
1	input	binary	17
2	input	interval	10
3	input	nominal	14
4	input	ordinal	16
5	target	binary	1

```
['ps_ind_02_cat',  
'ps_ind_04_cat',  
'ps_ind_05_cat',  
'ps_car_01_cat',  
'ps_car_02_cat',  
'ps_car_03_cat',  
'ps_car_04_cat',  
'ps_car_05_cat',  
'ps_car_06_cat',  
'ps_car_07_cat',  
'ps_car_08_cat',  
'ps_car_09_cat',  
'ps_car_10_cat',  
'ps_car_11_cat']
```

```
['ps_ind_06_bin',  
'ps_ind_07_bin',  
'ps_ind_08_bin',  
'ps_ind_09_bin',  
'ps_ind_10_bin',  
'ps_ind_11_bin',  
'ps_ind_12_bin',  
'ps_ind_13_bin',  
'ps_ind_16_bin',  
'ps_ind_17_bin',  
'ps_ind_18_bin',  
'ps_calc_15_bin',  
'ps_calc_16_bin',  
'ps_calc_17_bin',  
'ps_calc_18_bin',  
'ps_calc_19_bin',  
'ps_calc_20_bin']
```

```
['id',  
'target',  
'ps_ind_01',  
'ps_ind_03',  
'ps_ind_14',  
'ps_ind_15',  
'ps_reg_01',  
'ps_reg_02',  
'ps_reg_03',  
'ps_car_11',  
'ps_car_12',  
'ps_car_13',  
'ps_car_14',  
'ps_car_15',  
'ps_calc_01',  
'ps_calc_02',  
'ps_calc_03',  
'ps_calc_04',  
'ps_calc_05',  
'ps_calc_06',  
'ps_calc_07',  
'ps_calc_08',  
'ps_calc_09',  
'ps_calc_10',  
'ps_calc_11',  
'ps_calc_12',  
'ps_calc_13',  
'ps_calc_14']
```

Descriptive Statistics

Descriptive Statistics - Interval Variables

	ps_reg_01	ps_reg_02	ps_reg_03	ps_car_12	ps_car_13	ps_car_14	ps_car_15	ps_calc_01	ps_calc_02	ps_calc_03
count	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000
mean	0.610991	0.439184	0.551102	0.379945	0.813265	0.276256	3.065899	0.449756	0.449589	0.449849
std	0.287643	0.404264	0.793506	0.058327	0.224588	0.357154	0.731366	0.287198	0.286893	0.287153
min	0.000000	0.000000	-1.000000	-1.000000	0.250619	-1.000000	0.000000	0.000000	0.000000	0.000000
25%	0.400000	0.200000	0.525000	0.316228	0.670867	0.333167	2.828427	0.200000	0.200000	0.200000
50%	0.700000	0.300000	0.720677	0.374166	0.765811	0.368782	3.316625	0.500000	0.400000	0.500000
75%	0.900000	0.600000	1.000000	0.400000	0.906190	0.396485	3.605551	0.700000	0.700000	0.700000
max	0.900000	1.800000	4.037945	1.264911	3.720626	0.636396	3.741657	0.900000	0.900000	0.900000

Descriptive Statistics - Interval Variables

	ps_reg_01	ps_reg_02	ps_reg_03	ps_car_12	ps_car_13	ps_car_14	ps_car_15	ps_calc_01	ps_calc_02	ps_calc_03
count	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000
mean	0.610991	0.439184	0.551102	0.379945	0.813265	0.276256	3.065899	0.449756	0.449589	0.449849
std	0.287643	0.404264	0.793506	0.058327	0.224588	0.357154	0.731366	0.287198	0.286893	0.287153
min	0.000000	0.000000	-1.000000	-1.000000	0.250619	-1.000000	0.000000	0.000000	0.000000	0.000000
25%	0.400000	0.200000	0.525000	0.310028	0.670867	0.303167	2.828427	0.200000	0.200000	0.200000
50%	0.700000	0.300000	0.720677	0.374166	0.765811	0.368782	3.316625	0.500000	0.400000	0.500000
75%	0.900000	0.600000	1.000000	0.400000	0.906190	0.396485	3.605551	0.700000	0.700000	0.700000
max	0.900000	1.800000	4.037945	1.264911	3.720626	0.636396	3.741657	0.900000	0.900000	0.900000

Missing Values



Descriptive Statistics - Ordinal Variables

	ps_ind_01	ps_ind_03	ps_ind_14	ps_ind_15	ps_car_11	ps_calc_04	ps_calc_05	ps_calc_06	ps_calc_07	ps_calc_08	ps_calc_09	ps_calc_10	ps_calc_11	ps_calc_12	ps_calc_13	ps_calc_14
count	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000
mean	1.900378	4.423318	0.012451	7.299922	2.346072	2.372081	1.885886	7.689445	3.005823	9.225904	2.339034	8.433590	5.441382	1.441918	2.872288	7.539026
std	1.983789	2.699902	0.127545	3.546042	0.832548	1.117219	1.134927	1.334312	1.414564	1.459672	1.246949	2.904597	2.332871	1.202963	1.694887	2.746652
min	0.000000	0.000000	0.000000	0.000000	-1.000000	0.000000	0.000000	0.000000	0.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	0.000000	5.000000	2.000000	2.000000	1.000000	7.000000	2.000000	8.000000	1.000000	6.000000	4.000000	1.000000	2.000000	6.000000
50%	1.000000	4.000000	0.000000	7.000000	3.000000	2.000000	2.000000	8.000000	3.000000	9.000000	2.000000	8.000000	5.000000	1.000000	3.000000	7.000000
75%	3.000000	6.000000	0.000000	10.000000	3.000000	3.000000	3.000000	9.000000	4.000000	10.000000	3.000000	10.000000	7.000000	2.000000	4.000000	9.000000
max	7.000000	11.000000	4.000000	13.000000	3.000000	5.000000	6.000000	10.000000	9.000000	12.000000	7.000000	25.000000	19.000000	10.000000	13.000000	23.000000

Descriptive Statistics - Ordinal Variables

	ps_ind_01	ps_ind_03	ps_ind_14	ps_ind_15	ps_car_11	ps_calc_04	ps_calc_05	ps_calc_06	ps_calc_07	ps_calc_08	ps_calc_09	ps_calc_10	ps_calc_11	ps_calc_12	ps_calc_13	ps_calc_14
count	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000
mean	1.900378	4.423318	0.012451	7.299922	2.346072	2.372081	1.885886	7.689445	3.005823	9.225904	2.339034	8.433590	5.441382	1.441918	2.872288	7.539026
std	1.983789	2.699902	0.127545	3.546042	0.832548	1.117219	1.134927	1.334312	1.414564	1.459672	1.246949	2.904597	2.332871	1.202963	1.694887	2.746652
min	0.000000	0.000000	0.000000	0.000000	-1.000000	0.000000	0.000000	0.000000	0.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	0.000000	5.000000	2.000000	2.000000	1.000000	7.000000	2.000000	8.000000	1.000000	6.000000	4.000000	1.000000	2.000000	6.000000
50%	1.000000	4.000000	0.000000	7.000000	3.000000	2.000000	2.000000	8.000000	3.000000	9.000000	2.000000	8.000000	5.000000	1.000000	3.000000	7.000000
75%	3.000000	6.000000	0.000000	10.000000	3.000000	2.000000	3.000000	9.000000	4.000000	10.000000	3.000000	10.000000	7.000000	2.000000	4.000000	9.000000
max	7.000000	11.000000	4.000000	13.000000	3.000000	5.000000	6.000000	10.000000	9.000000	12.000000	7.000000	25.000000	19.000000	10.000000	13.000000	23.000000

Missing Values

Descriptive Statistics - Binary Variables

	target	ps_ind_06_bin	ps_ind_07_bin	ps_ind_08_bin	ps_ind_09_bin	ps_ind_10_bin	ps_ind_11_bin	ps_ind_12_bin	ps_ind_13_bin	ps_ind_16_bin	ps_ind_17_bin	ps_ind_18_bin
count	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000
mean	0.036448	0.393742	0.257033	0.163921	0.185304	0.000373	0.001692	0.009439	0.000948	0.660823	0.121081	0.153446
std	0.187401	0.488579	0.436998	0.370205	0.388544	0.019309	0.041097	0.096693	0.030768	0.473430	0.326222	0.360417
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
75%	0.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

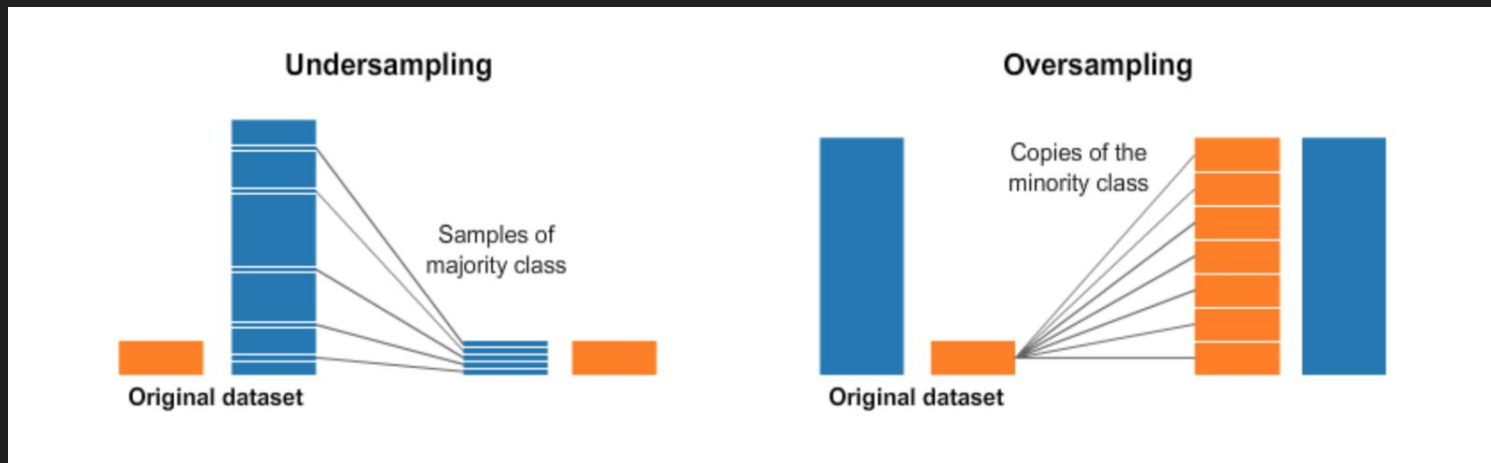
ps_calc_15_bin	ps_calc_16_bin	ps_calc_17_bin	ps_calc_18_bin	ps_calc_19_bin	ps_calc_20_bin
595212.000000	595212.000000	595212.000000	595212.000000	595212.000000	595212.000000
0.122427	0.627840	0.554182	0.287182	0.349024	0.153318
0.327779	0.483381	0.497056	0.452447	0.476662	0.360295
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
0.000000	1.000000	1.000000	0.000000	0.000000	0.000000
0.000000	1.000000	1.000000	1.000000	1.000000	0.000000
1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

strongly imbalanced

Handling Imbalanced Classes

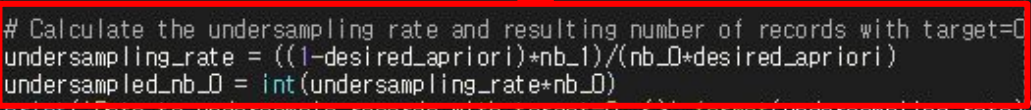
Handling Imbalanced Classes

- Undersampling records with target = 0
- Oversampling records with target = 1



Handling Imbalanced Classes

```
1 desired_apriori=0.10
2
3 # Get the indices per target value
4 idx_0 = train[train.target == 0].index
5 idx_1 = train[train.target == 1].index
6
7 # Get original number of records per target value
8 nb_0 = len(train.loc[idx_0])
9 nb_1 = len(train.loc[idx_1])
10
11 # Calculate the undersampling rate and resulting number of records with target=0
12 undersampling_rate = ((1-desired_apriori)*nb_1)/(nb_0*desired_apriori)
13 undersampled_nb_0 = int(undersampling_rate*nb_0)
14 print('Rate to undersample records with target=0: {}'.format(undersampling_rate))
15 print('Number of records with target=0 after undersampling: {}'.format(undersampled_nb_0))
16
17 # Randomly select records with target=0 to get at the desired a priori
18 undersampled_idx = shuffle(idx_0, random_state=37, n_samples=undersampled_nb_0)
19
20 # Construct list with remaining indices
21 idx_list = list(undersampled_idx) + list(idx_1)
22
23 # Return undersample data frame
24 train = train.loc[idx_list].reset_index(drop=True)
```



undersampling

```
Rate to undersample records with target=0: 0.34043569687437886
Number of records with target=0 after undersampling: 195246
```

Data Quality Check

Data Quality Check - Missing Values

	column	count	ratio
3	ps_ind_02_cat	216	0.036290
5	ps_ind_04_cat	83	0.013945
6	ps_ind_05_cat	5809	0.975955
22	ps_reg_03	107772	18.106490
23	ps_car_01_cat	107	0.017977
24	ps_car_02_cat	5	0.000840
25	ps_car_03_cat	411231	69.089837
27	ps_car_05_cat	266551	44.782531
29	ps_car_07_cat	11489	1.930237
31	ps_car_09_cat	569	0.095596
34	ps_car_11	5	0.000840
35	ps_car_12	1	0.000168
37	ps_car_14	42620	7.160474

Data Quality Check - Missing Values

```
1 vars_with_missing = []
2
3 for f in train.columns:
4     missings = train[train[f] == -1][f].count()
5     if missings > 0:
6         vars_with_missing.append(f)
7         missings_perc = missings/train.shape[0]
8
9     print('Variable {} has {} records ( {:.2%}) with missing values'.format(f, missings, missings_perc))
10
11 print('In total, there are {} variables with missing values'.format(len(vars_with_missing)))
```

```
Variable ps_ind_02_cat has 103 records (0.05%) with missing values
Variable ps_ind_04_cat has 51 records (0.02%) with missing values
Variable ps_ind_05_cat has 2256 records (1.04%) with missing values
Variable ps_reg_03 has 38580 records (17.78%) with missing values
Variable ps_car_01_cat has 62 records (0.03%) with missing values
Variable ps_car_02_cat has 2 records (0.00%) with missing values
Variable ps_car_03_cat has 148367 records (68.39%) with missing values
Variable ps_car_05_cat has 96026 records (44.26%) with missing values
Variable ps_car_07_cat has 4431 records (2.04%) with missing values
Variable ps_car_09_cat has 230 records (0.11%) with missing values
Variable ps_car_11 has 1 records (0.00%) with missing values
Variable ps_car_14 has 15726 records (7.25%) with missing values
In total, there are 12 variables with missing values
```

**too many missing values
→ Drop**

Data Quality Check - Missing Values

```
1 vars_with_missing = []
2
3 for f in train.columns:
4     missings = train[train[f] == -1][f].count()
5     if missings > 0:
6         vars_with_missing.append(f)
7         missings_perc = missings/train.shape[0]
8
9     print('Variable {} has {} records ( {:.2%}) with missing values'.format(f, missings, missings_perc))
10
11 print('In total, there are {} variables with missing values'.format(len(vars_with_missing)))
```

```
Variable ps_ind_02_cat has 103 records (0.05%) with missing values
Variable ps_ind_04_cat has 51 records (0.02%) with missing values
Variable ps_ind_05_cat has 2256 records (1.04%) with missing values
Variable ps_reg_03 has 39590 records (17.78%) with missing values
Variable ps_car_01_cat has 62 records (0.03%) with missing values
Variable ps_car_02_cat has 2 records (0.00%) with missing values
Variable ps_car_03_cat has 148367 records (68.39%) with missing values
Variable ps_car_05_cat has 96026 records (44.26%) with missing values
Variable ps_car_07_cat has 4431 records (2.04%) with missing values
Variable ps_car_09_cat has 230 records (0.11%) with missing values
Variable ps_car_11 has 1 records (0.00%) with missing values
Variable ps_car_14 has 15726 records (7.25%) with missing values
In total, there are 12 variables with missing values
```

categorical variables
→ leave the missing value

Data Quality Check - Missing Values

```
1 vars_with_missing = []
2
3 for f in train.columns:
4     missings = train[train[f] == -1][f].count()
5     if missings > 0:
6         vars_with_missing.append(f)
7         missings_perc = missings/train.shape[0]
8
9     print('Variable {} has {} records ( {:.2%}) with missing values'.format(f, missings, missings_perc))
10
11 print('In total, there are {} variables with missing values'.format(len(vars_with_missing)))
```

Variable ps_ind_02_cat has 103 records (0.05%) with missing values
Variable ps_ind_04_cat has 51 records (0.02%) with missing values
Variable ps_ind_05_cat has 2256 records (1.04%) with missing values
Variable ps_reg_03 has 38580 records (17.78%) with missing values
Variable ps_car_01_cat has 62 records (0.03%) with missing values
Variable ps_car_02_cat has 2 records (0.00%) with missing values
Variable ps_car_03_cat has 148367 records (68.39%) with missing values
Variable ps_car_05_cat has 96026 records (44.26%) with missing values
Variable ps_car_07_cat has 4431 records (2.04%) with missing values
Variable ps_car_09_cat has 230 records (0.11%) with missing values
Variable ps_car_11 has 1 records (0.00%) with missing values
Variable ps_car_14 has 15726 records (7.25%) with missing values
In total, there are 12 variables with missing values

ps_reg_03

ps_car_12

ps_car_14

**→ continuous variable
replace by mean**

Data Quality Check - Missing Values

```
1 vars_with_missing = []
2
3 for f in train.columns:
4     missings = train[train[f] == -1][f].count()
5     if missings > 0:
6         vars_with_missing.append(f)
7         missings_perc = missings/train.shape[0]
8
9     print('Variable {} has {} records ({:.2%}) with missing values'.format(f, missings, missings_perc))
10
11 print('In total, there are {} variables with missing values'.format(len(vars_with_missing)))
```

```
Variable ps_ind_02_cat has 103 records (0.05%) with missing values
Variable ps_ind_04_cat has 51 records (0.02%) with missing values
Variable ps_ind_05_cat has 2256 records (1.04%) with missing values
Variable ps_reg_03 has 38580 records (17.78%) with missing values
Variable ps_car_01_cat has 62 records (0.03%) with missing values
Variable ps_car_02_cat has 2 records (0.00%) with missing values
Variable ps_car_03_cat has 148367 records (68.39%) with missing values
Variable ps_car_05_cat has 96026 records (44.26%) with missing values
Variable ps_car_07_cat has 4431 records (2.04%) with missing values
Variable ps_car_09_cat has 230 records (0.11%) with missing values
Variable ps_car_11 has 1 records (0.00%) with missing values
Variable ps_car_14 has 1576 records (7.25%) with missing values
In total, there are 12 variables with missing values
```

ordinal variables
→ **replace by the mode**

Data Quality Check - Categorical Variables

```
1 v = meta[(meta.level == 'nominal') & (meta.keep)].index
2
3 for f in v:
4     dist_values = train[f].value_counts().shape[0]
5     print('Variable {} has {} distinct values'.format(f, dist_values))
```

```
Variable ps_ind_02_cat has 5 distinct values
Variable ps_ind_04_cat has 3 distinct values
Variable ps_ind_05_cat has 8 distinct values
Variable ps_car_01_cat has 13 distinct values
Variable ps_car_02_cat has 3 distinct values
Variable ps_car_04_cat has 10 distinct values
Variable ps_car_06_cat has 18 distinct values
Variable ps_car_07_cat has 3 distinct values
Variable ps_car_08_cat has 2 distinct values
Variable ps_car_09_cat has 6 distinct values
Variable ps_car_10_cat has 3 distinct values
Variable ps_car_11_cat has 104 distinct values
```

Data Quality Check - Categorical Variables

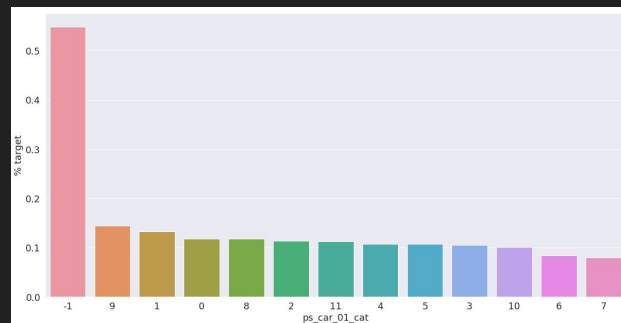
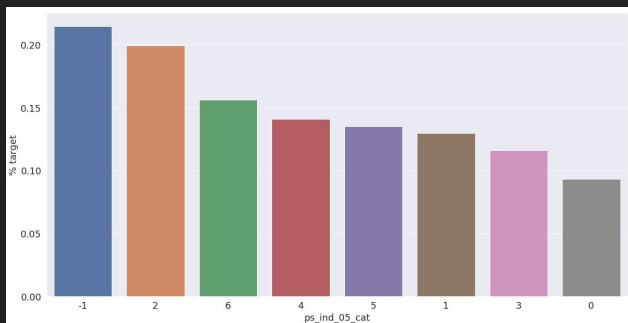
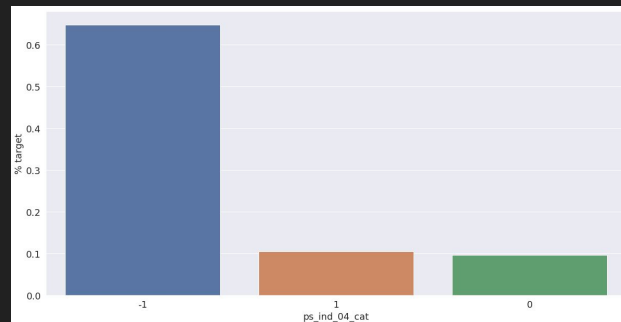
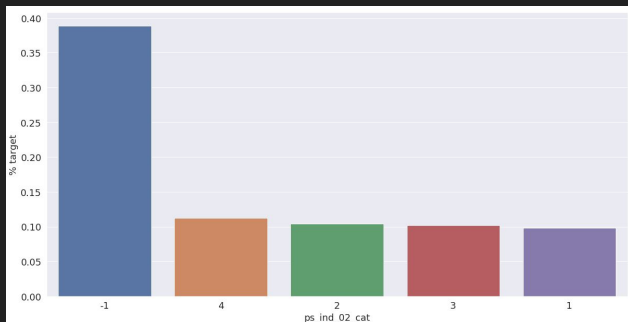
```
1 v = meta[(meta.level == 'nominal') & (meta.keep)].index
2
3 for f in v:
4     dist_values = train[f].value_counts().shape[0]
5     print('Variable {} has {} distinct values'.format(f, dist_values))
```

```
Variable ps_ind_02_cat has 5 distinct values
Variable ps_ind_04_cat has 3 distinct values
Variable ps_ind_05_cat has 8 distinct values
Variable ps_car_01_cat has 13 distinct values
Variable ps_car_02_cat has 3 distinct values
Variable ps_car_04_cat has 10 distinct values
Variable ps_car_06_cat has 18 distinct values
Variable ps_car_07_cat has 3 distinct values
Variable ps_car_08_cat has 2 distinct values
Variable ps_car_09_cat has 6 distinct values
Variable ps_car_10_cat has 3 distinct values
Variable ps_car_11_cat has 104 distinct values
```

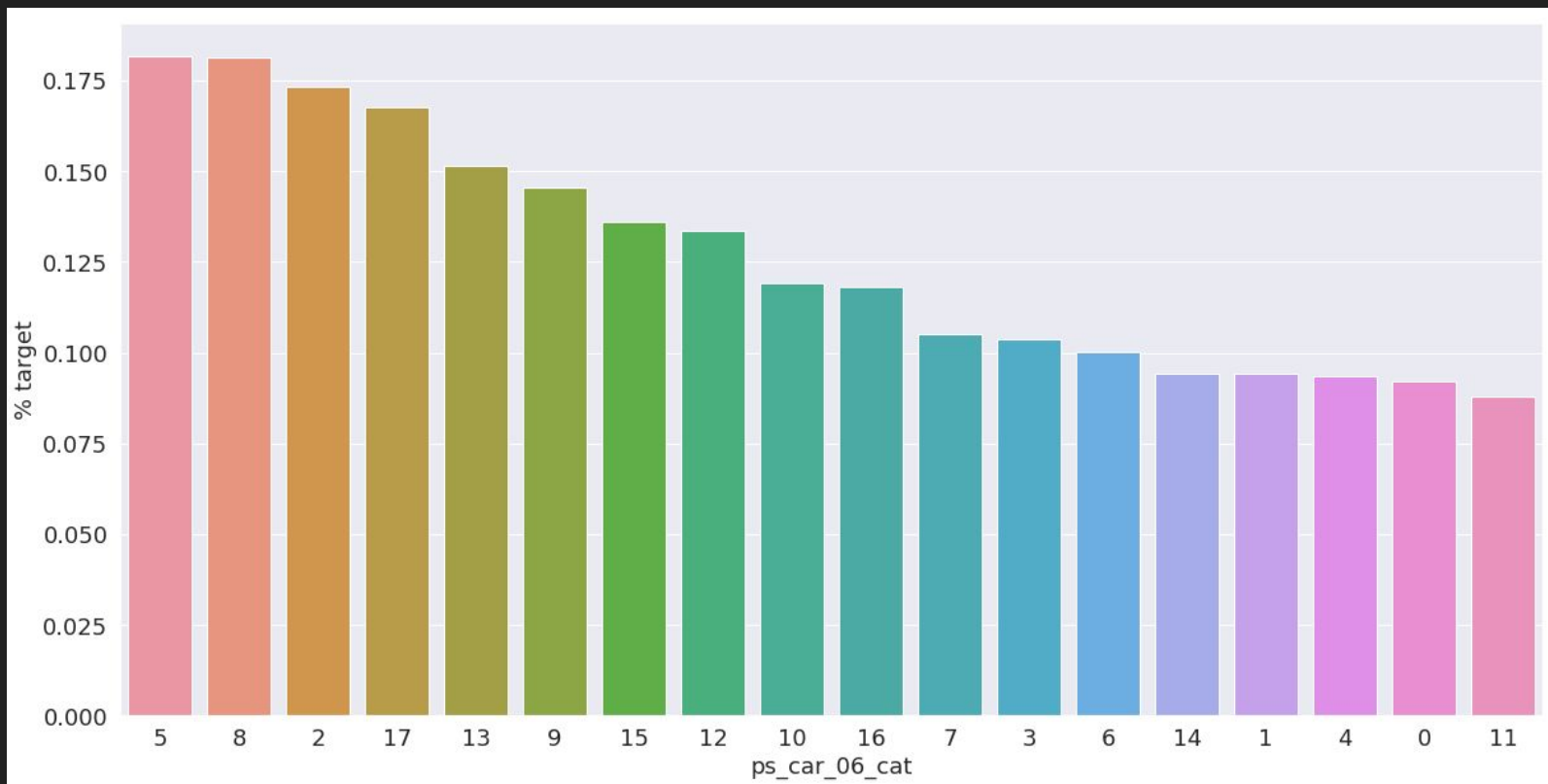
**many distinct values
→ still reasonable**

Data Visualization

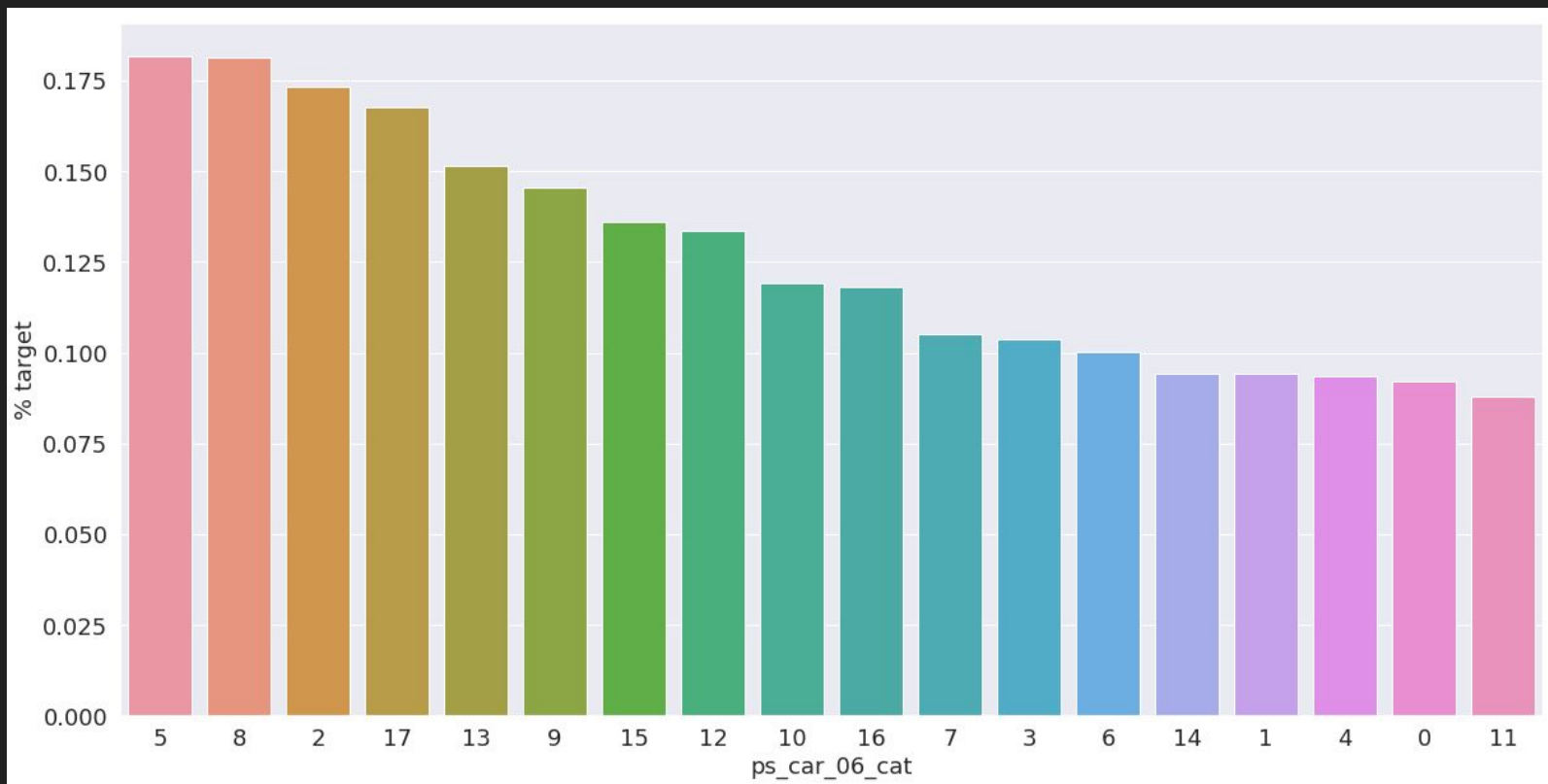
Data Visualization - Categorical



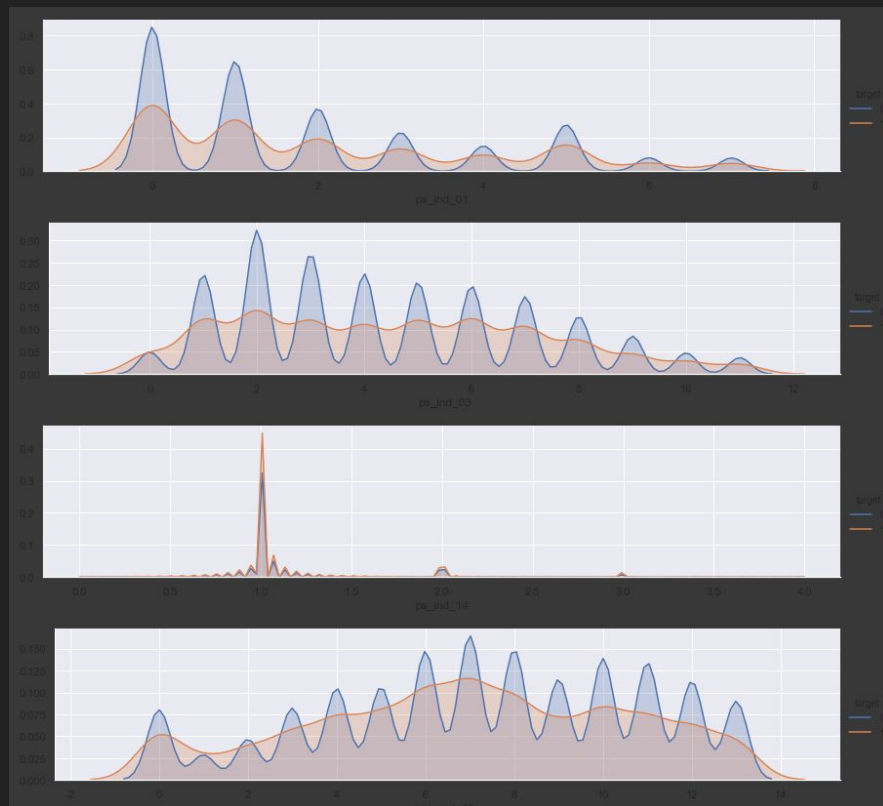
Data Visualization - Categorical



Data Visualization - Categorical

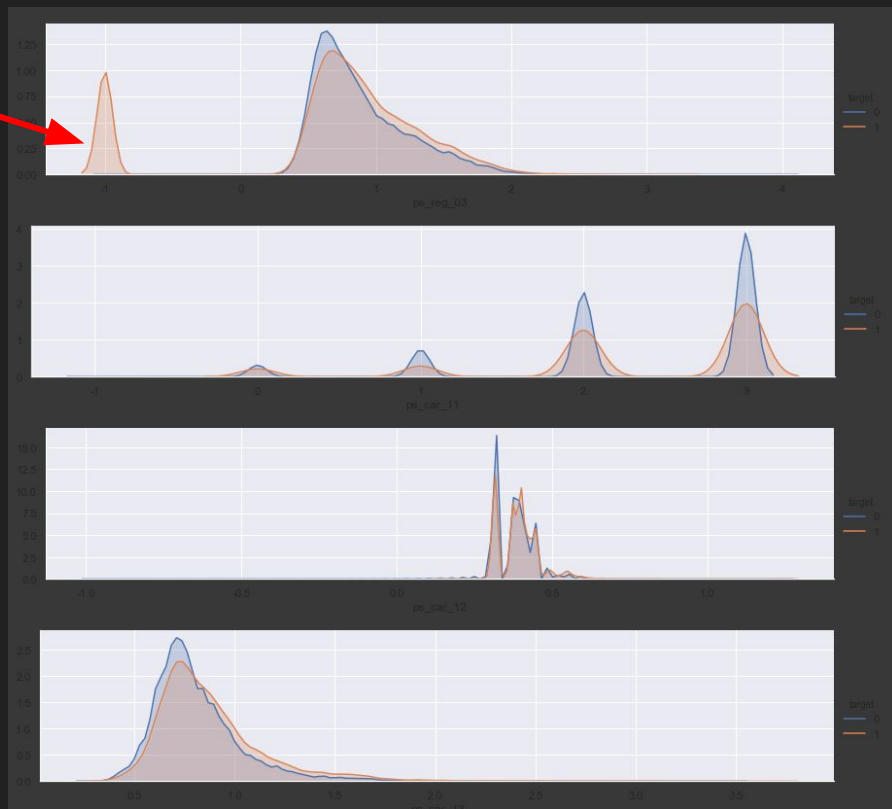


Data Visualization - Interval



Data Visualization - Interval

Missing Values

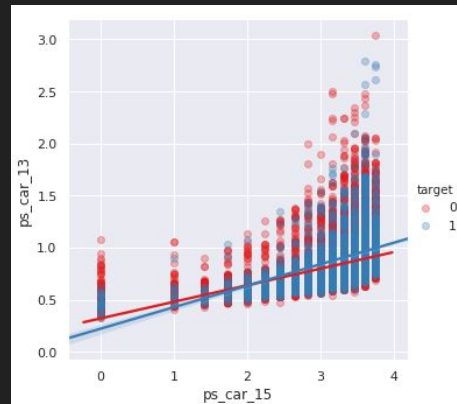
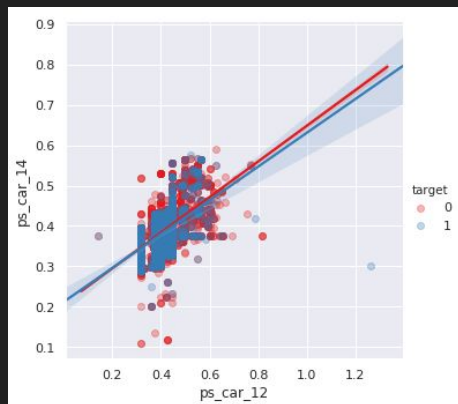
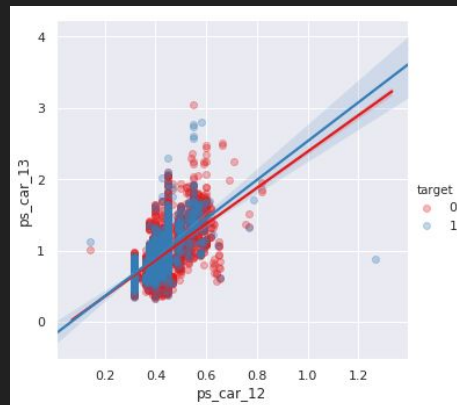
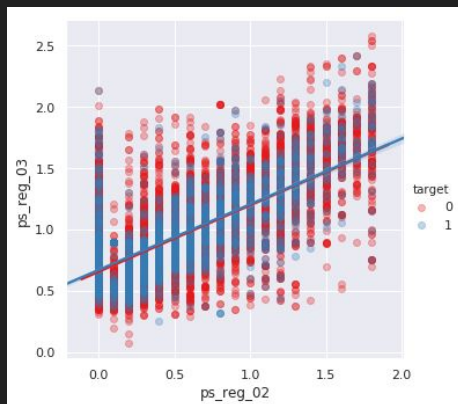


Data Visualization - Interval

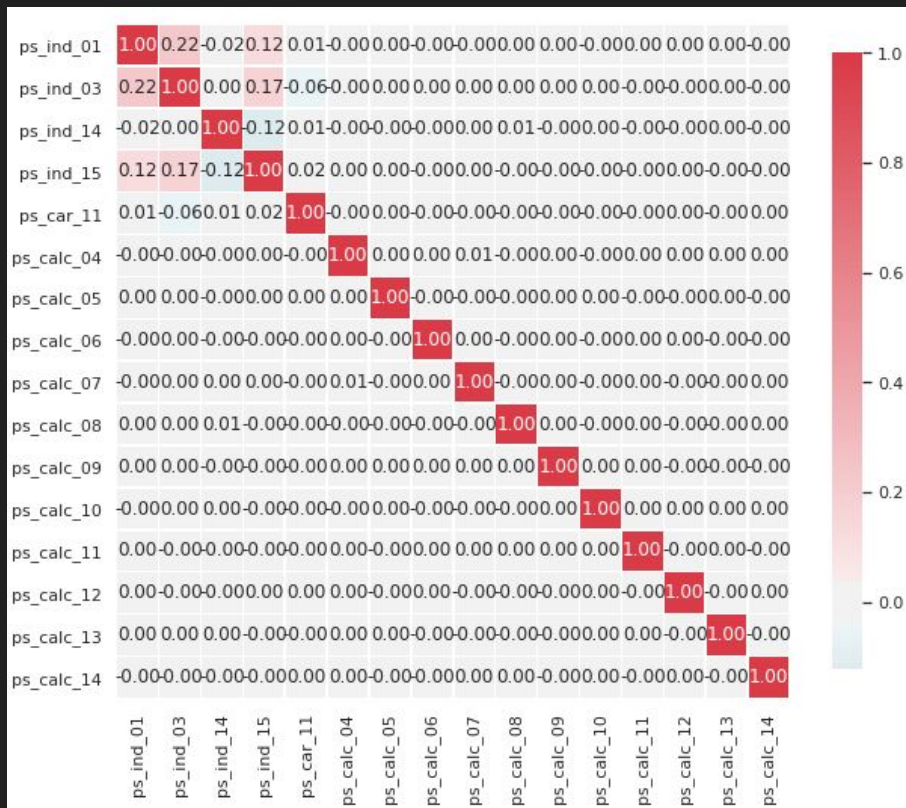


Strong Correlation

Data Visualization - Interval



Data Visualization - Ordinal



QnA

Reference

Reference

- <https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/overview>
- <https://www.kaggle.com/c/porto-seguro-safe-driver-prediction>
- https://cdn-images-1.medium.com/max/1600/1*P93SeDGPgw0MhwvCcvVcXA.png
- https://upload.wikimedia.org/wikipedia/commons/thumb/5/5b/Economics_Gini_coefficient.svg/280px-Economics_Gini_coefficient.svg.png
- <https://m.blog.naver.com/PostView.nhn?blogId=ssdyka&logNo=221284738829&proxyReferer=https%3A%2F%2Fwww.google.com%2F>
-

Thanks