# Semester Project

William Huibregtse[1], Joshua Baker[1], Chris East[1]

**Abstract**
Include abstract here – A summary of your work

**Keywords**
Keyword1 — Synergy — Keyword3

[1] *Computer Science, School of Informatics , Computing and Engineering, Indiana University, Bloomington, IN, USA*

## Contents



## 1. Problem and Data Description

First we want to get a general idea of our data set and get a deeper understanding of the underlying structure.
There are 59 named features or variables for our data set.
With 892816 observations for training and 595212 for test
There are no duplicate observations.
Features that belong to similar groupings are given certain feature names.
- Ind: related to individual or driver
- Reg: related to geographical region
- Car: related to car being insured
- Calc: are calculated features done by Proto themselves
Postfix descriptors describes the features data type.
- Bin: Binary (1 or 0)
- Cat: Categorical *Note: the dataset has the categorical data already convert into factors and then integers
- All other variables are either integer or numeric
As stated the Data Types are numeric and integer, with integer being the predominant type 49 to 10.
Missing values are represented by -1.
In total, there are 13 variables with missing values.
There is Target feature which denotes the binary classification for that observation. This feature is the feature we are trying to learn/predict for the test data.
There is an ID feature which is an anonymized identities of insured drivers.

Porto Seguro's Safe Driver Prediction has 59 variables and 1.3 million observations, which qualifies as a good candidate for reducing overall dimensions of the data to significantly increase the speed of analysis techniques at the cost of more upfront data processing.

There are only 21694 cases of classification 1, which is 3.64 percent of the observations in the training data set, showing significant skew in the expected class towards a "0" prediction.

## 2. Data Preprocessing & Exploratory Data Analysis

### 2.1 Feature Engineering

As even missing data can be significant, a new feature was added to the data set. This feature was the count of missing values for each entry before these missing values were processed. This technique allows the retention of the potentially useful information provided by the missing values.

### 2.2 Handling Missing Values

**Overview of features with missing entries**

An important aspect of data preprocessing is handling missing data. In total, there are 13 features that have at least 1 missing value. For both the training (*Table*1.) and testing (*Table*2.) data the individual features are broken down into: number of missing entries, and percentage of total entries that are missing. As the tables show, both training and training have extremely close value for all the features. The next step is comparing the distribution of features with missing entries between training and testing.
Taking a look at the graphs located in the *Appendix* we see that all the features have incredible close distribution between Training and Testing, for the exception of $ps-car-12$. However, this feature has 1 missing entry in Training and 0 in Testing. Therefore, the difference in this feature should be ignored. The importance of both distribution and number of

missing entries being extremely close is that the methodology developed to handle NA's in Training is applicable for Testing.

**Table 1.** Train features with missing values

| Feature | Train | % Missing |
|---|---|---|
| $ps-ind-02-cat$ | 216 | 0.036 |
| $ps-ind-04-cat$ | 83 | 0.014 |
| $ps-ind-05-cat$ | 5809 | 0.98 |
| $ps-reg-03$ | 107772 | 18.11 |
| $ps-car-01-cat$ | 107 | 0.018 |
| $ps-car-02-cat$ | 5 | 0.001 |
| $ps-car-03-cat$ | 411231 | 69.09 |
| $ps-car-05-cat$ | 266551 | 44.78 |
| $ps-car-07-cat$ | 11489 | 1.93 |
| $ps-car-09-cat$ | 569 | 0.10 |
| $ps-car-11$ | 5 | 0.001 |
| $ps-car-12$ | 1 | 0.0002 |
| $ps-car-14$ | 42620 | 7.160474 |

**Table 2.** Test features with missing values

| Feature | Test | % Missing |
|---|---|---|
| $ps-ind-02-cat$ | 307 | 0.034 |
| $ps-ind-04-cat$ | 145 | 0.016 |
| $ps-ind-05-cat$ | 8710 | 0.97 |
| $ps-reg-03$ | 161684 | 18.11 |
| $ps-car-01-cat$ | 160 | 0.018 |
| $ps-car-02-cat$ | 5 | 0.001 |
| $ps-car-03-cat$ | 616911 | 69.10 |
| $ps-car-05-cat$ | 400359 | 44.84 |
| $ps-car-07-cat$ | 17331 | 1.94 |
| $ps-car-09-cat$ | 877 | 0.10 |
| $ps-car-11$ | 1 | 0.0001 |
| $ps-car-12$ | 0 | 0 |
| $ps-car-14$ | 63805 | 7.15 |

**Handling missing entries**

The features with missing entries are a mixture of categorical with binary or several factors, and then continuous values between 0 and 1. This mixture requires a different approach for each feature.

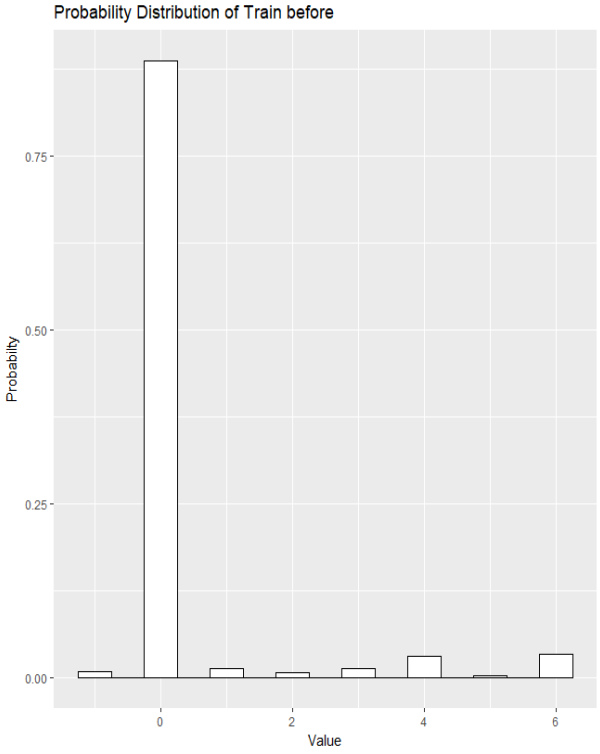
*Features with high percentage of missing values:*

Two main features stick out: $ps-car-03-cat$ and $ps-car-05-cat$ . With 69% and 44.8% missing entries respectively. Both features are binary categorical variables describing the car that's being insured. Given the fact this is a binary variable that describes the car and that there's an overwhelming percentage of missing values we thought it might be best to delete these features. As a quick check before we remove them, we examine the percentage of missing values with the target value of 1 vs the number of total claims (target value = 1). Surprisingly, $ps-car-03-cat$ feature with 69.1% of it's entries missing with the target 1 accounts for 62% of the total claims! Same goes for $ps-car-05-cat$ feature with 44.8% of it's entries missing with the target 1 accounts for 39% of the total claims! Here it seems that the value missing can be an important feature in predicting claims. Therefore, we replace -1 with 2 and set NA's to their own category. Changing the binary value to an ordinal to preserve the no response characteristic of our data. $Note$ : 2 was decided vs -1 to remove the negative value *Other Categorical Features with low percentage of missing values:*

All of these features are categorical, with low % of missing entries $< 2\%$ and mainly very low % missing entry claim vs total claim. Therefore, we can treat these features as discrete random variables. Which allows us to replace these missing values from a probability density function model after the non-missing entries in each feature. The important goal here is to not alter the distribution of the non-missing values.

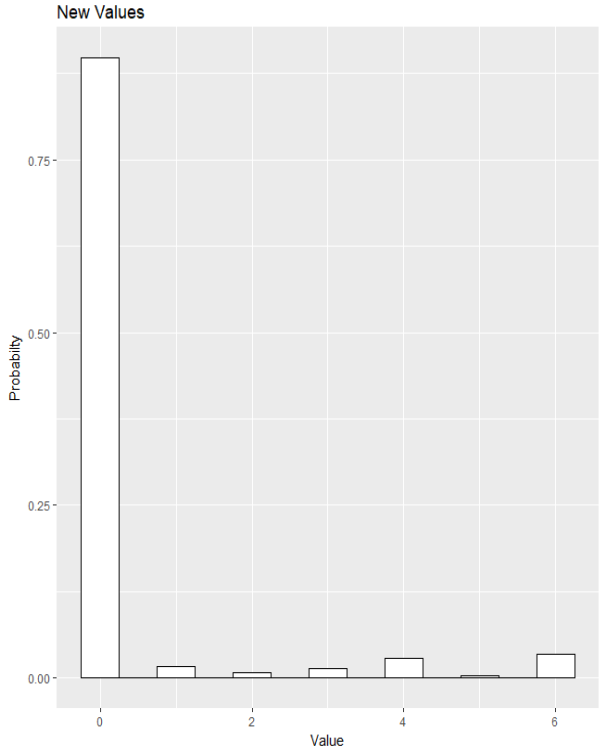
**Example:** ps-ind-05-cat Below is an example of the effects of using a PDF based on the probabilities of the non-missing entries to draw replacement values.

This is the histogram of the values for ps-ind-05-cat, clearly there are a few -1 values. It also shows the general distribution of the other categorical values for this feature.

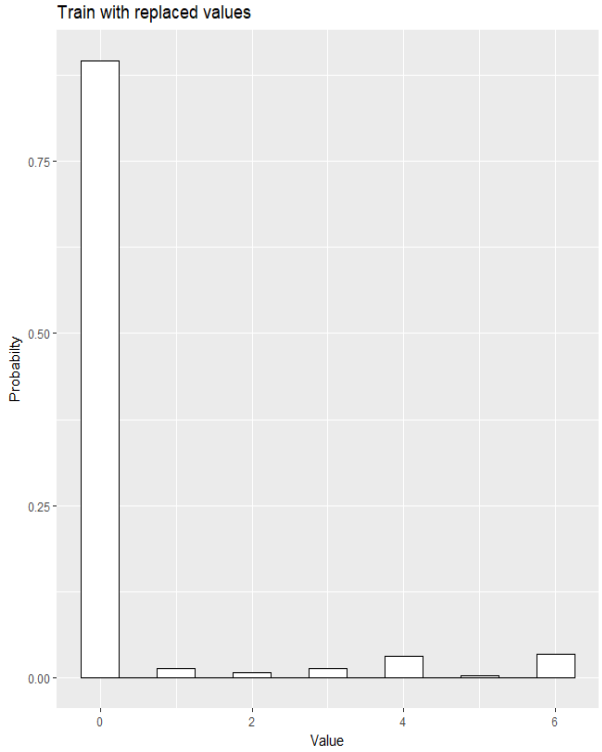



This is a plot of the new values generate to replace the missing entries based on the distribution of the non-missing entries for

the feature. This plot clearly shows that the distribution of the non-missing entries is retained in these new values.




This is the plot of the values after replacing. We see that there are no longer -1 entries, and that the overall distribution has changed very little.

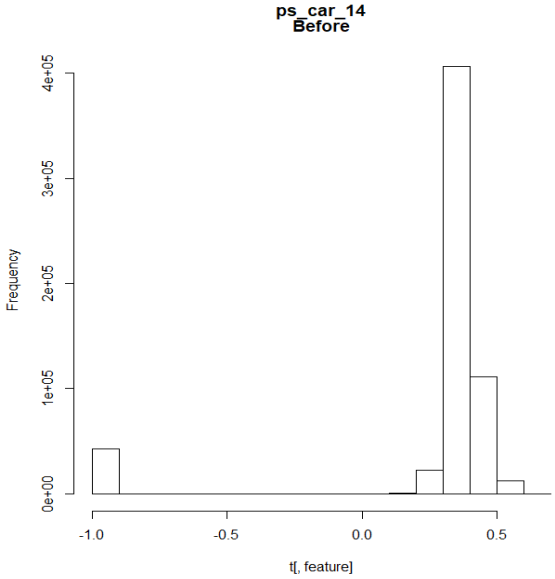



This method is applied to 6 categorical features: ps-car-01-cat, ps-car-07-cat, ps-car-09-cat, ps-ind-02-cat, ps-ind-04-cat, and ps-ind-05-cat

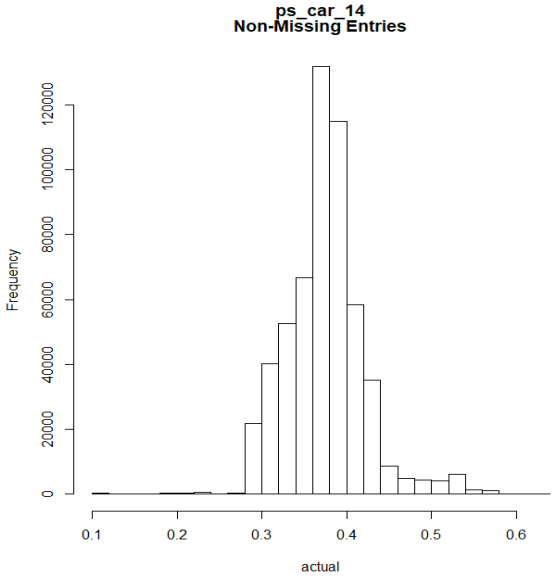
*Features with continuous distributions between 0 and 1:*

There is a single continuous feature: ps-car-14. This feature has a low number of missing entries (7.2%) and a low % of missing that are a claim vs total claim (7.9%), making this a good candidate to replace missing entries with values drawn from a Continuous Random Variable.
Here the feature's non-missing entries has a log-normal distribution. Therefore, we'll use *rlnorm* to approximate the distribution when drawing replacement values.

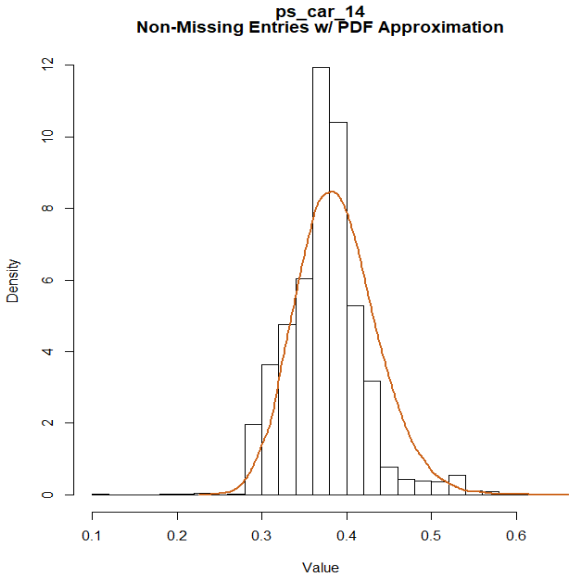
This is the histogram of the values for ps-car-14, clearly there are a few -1 values. It also shows the general distribution of the other continuous values for this feature.



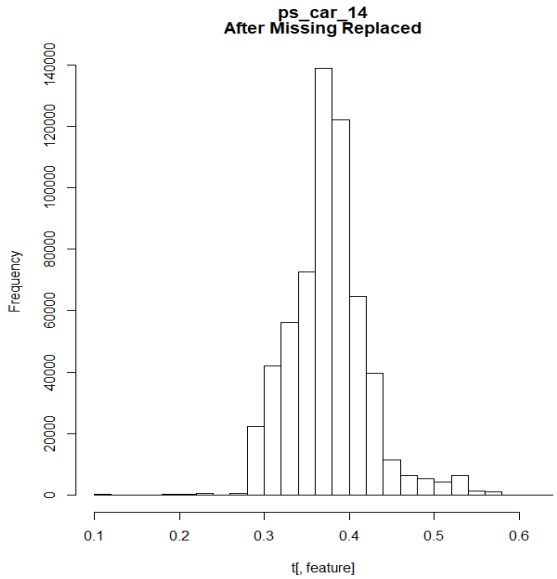
This is the histogram of the non-missing values for ps-car-14, showing the true distribution of the continuous values for this feature.




This is the histogram of the non-missing values plus the density curve for the log-normal pdf we generated from the distribution of the non-missing entries. As shown, the pdf has a good fit and allows for us to draw from it to replace missing entries for the ps-car-14 feature.




This is the final histogram of the feature with the missing entries replaced. We can see that the overall distribution is intact.




*Features with extremely few missing entries*

The rest of the features have an extreme small number of missing entries $< 5$. Therefore, we simply replace the missing entries with the mode of the feature. This is a more than adequate approach for so few observations in our data set (nearly 0%).

### 2.3 Exploratory Data Analysis

To reduce the complexity of future analysis techniques, we used the prcomp function from base R to perform PCA dimensionality reduction on our data. After the transformation, the first 16 principle components represented over 99 percent of the the data's variance. Using this as a reasonable cutoff, we proceeded forward using only these first 16 principle components.

Note: To perform this transformation correctly, the test and train data must be transformed together and split after the dimensions have been reduced.

## 3. Algorithm and Methodology

Once the data was processed, we used the Naive Bayes algorithm to form our model. We tested it with both the implementation from the e1071 package and the caret package. The caret package implementation ended up being more powerful since it allowed for easy cross validation. We used 10 folds, trained on our pca- reduced data set.

We wanted to test a second classifier, so we chose XGBoost, since it had been used successfully by a number of people previously. Despite trying a number of different tree

depths and iteration limits, we were unable to have it perform better than the Naive Bayes did.

## 4. Experiments and Results

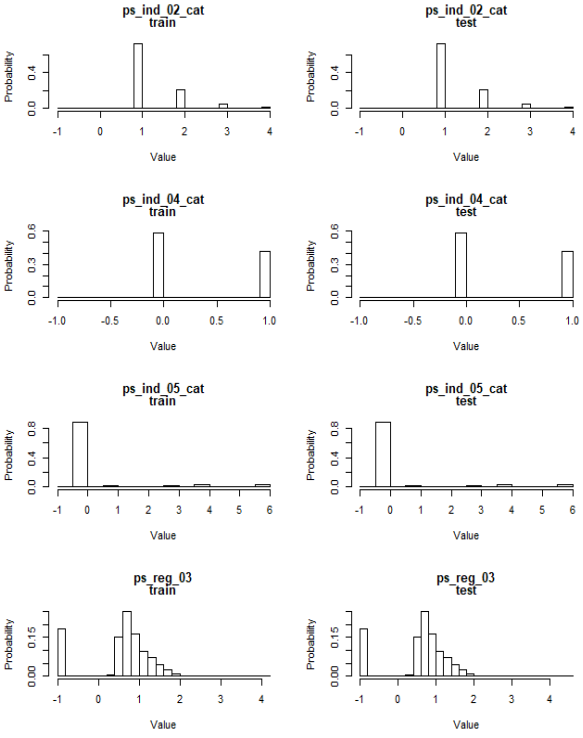
Since we had access to multiple submissions on Kaggle, we were able to use their fitness measurements to test our predictions. When we did PCA, we originally kept 95% of the variance, and used that to run Naive Bayes. When we submitted this, we only got a GINI score of .125, but when we increased the total variance to 99%, the score jumped up to .20566.

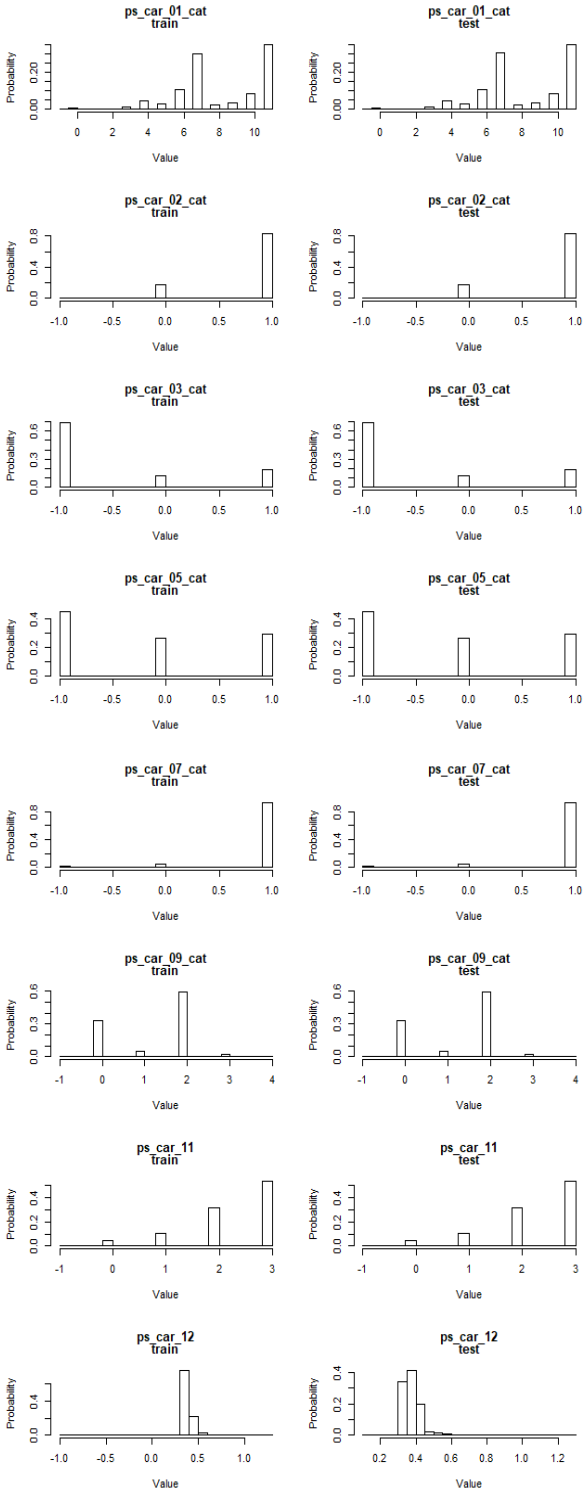
## 5. Summary and Conclusions

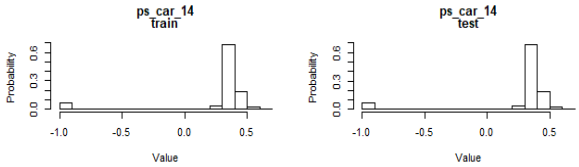
## 6. Appendix

### 6.1 Missing Values

Plotting the distribution of features with missing values: Training & Testing






## Acknowledgments