Predict if a driver will file an insurance claim next year

Final Project for ISE 5103

Svetlana Baca	**svetlana.baca@ou.edu**
Erdoo Segher	**erdoo.segher@ou.edu**
Jishan Luo	**jishanluo@ou.edu**

1 Executive Summary

Using predictive modeling in determining the probability of a driver will initiate an auto insurance claim next year isn't a magic wand but it is a valuable tool. This project is based on a 2017 Kaggle data analysis competition to predict if a driver will initiate an auto insurance in the next year. The data were collected from Kaggle's Safe Driver Prediction competition sponsored by Porto Seguero, a Brazilian auto insurance company (link is https://www.kaggle.com/c/porto-seguro-safe-driver-prediction). The competition was based on the data in the past 20 years of the driver initiated the claim. The goal of this project is to explore new, more powerful methods to predict the probability of a driver will initiate an auto insurance claim in the next year. For purposes of this project, building a prediction model to help Porto Seguro, make accurate predictions to reduce the cost of insurance for good drivers and raise the price for bad ones.

The data set used for this project was provided for the competition and was composed of 595,212 observations and 59 variables; while the test data set was composed of 892,816 observations and 58 variables. 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 signify whether or not a claim was filed for that policy holder.

Three models were used to predict the possibility of a driver will initiate an auto insurance in the next year. These models were: GLM, Bagging, and svm model.
Based on the results and analysis from models we chose, we would like to recommend applying Logistic model — glm model, to predict the probability of a driver will initiate an insurance in next year.

2 Problem Description

2.1. Background

Making accurate prediction for an auto insurance company is significant. Building a good prediction model will allow auto insurance companies to further tailor their prices, and would make auto insurance coverage more accessible to more drivers. A more accurate prediction model can help auto insurance companies to identify the probabilities of a driver will initiate a claim in the next year. Prediction the probabilities needs us to classify the importance of each variable, and pick up the variables which exerts big impact on the probabilities of initiating the claim in the next year. This report will focus on the regression aspect of the probability prediction. With more accurate model to make the prediction, the auto insurance companies can get fast and more accurate results that allow good driver to reduce their cost of insurance.

2.2. Kaggle Competition

This report is based around 2017 kaggle competition to predict a driver will initiate an auto insurance in the next year. The data set is from of the records in the past 20 years from Porto Seguro— one of Brazil's largest auto and homeowner insurance companies, the competition asks participants to build model to predict the probability of a driver will initiate the insurance in the next year. The competitors were evaluated using Normalized Gini Coefficient. The competition provides data of Porto Seguro in the past 20 years.

2.3. Problem Definition

The goal of this report is to build a model which will increase prediction accuracy. Due to the large number of features relative to number of observations, feature selection is a very important step in this problem and is believed to have a high impact on prediction accuracy. The goal of the project is to determine if different features impact accuracy when used to build controlled models. To give the project perspective, the winner of the kaggle competition's solution method was analyzed and used as a benchmark throughout the rest of the report to compare models and features. This project will provide key insight to the winner's success, and create a foundation for approaching similar data analysis problems.

2.4. Data Exploration

The data comes from Kaggle form of one training and test file. Each row corresponds to a specific
policy holder and the columns describe their features. The target variable indicates whether this
policy holder made an insurance claim in the past. For example, the value of 0 means that this
client did not make an insurance claim, while the value of 1 means that the client made the claim.
In the train and test data, the features that belong to similar groupings are tagged as such in the features' 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 signify whether or not a claim was filed for that policy holder. The value of target is set to 0 when the client will not initiate an insurance; the value of target is set to 1 means the client will initiate an auto insurance. There are 595,212 observations and 59 variables in train data set; there are 892,816 observations and 58 variables in test data set.

In the dataset with 59 columns each row corresponds to a policy holder, and the target column reflects that a claim was filed (1) or not (0). Features that belong to similar groupings are tagged as such in the feature names. For example:

-"ind" is related to individual or driver,
-"reg" is related to quality of life in a certain region,
-"car" is related to car itself,
-"calc" is an calculated feature.

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. We will need to convert these to a missing value representation.

2.5. Features

It is essential to understand the dataset before we create models for probability prediction. The 59 variables (including id and target) in the train dataset can be classified in four types—binary features, categorical features, integer features, and float features. The target variable belongs to binary variable (value of target is 0 or 1). Values of -1 from the dataset indicate that the feature was missing from the observation. The data description mentions that the names of the features indicate whether they are binary (bin) or categorical (cat) variables. Everything else is continuous or ordinal. We have already been told by Adriano Moala that 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 a calculated feature. In this project, we will treat these properties as groups.

2.6. Data Preparation

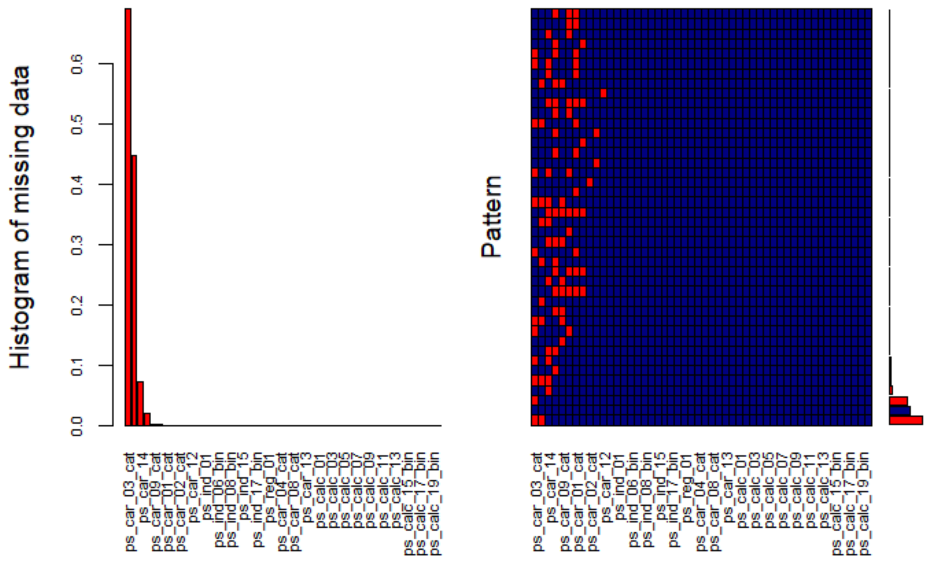
First, we used functions View and head to explore the data visually and manually. We then used nearZeroVar function to identify the columns with low variations because, being static, they wouldn't contribute much to the results. This generated a list of 9 attributes, which we removed.

Columns with near zero variance:

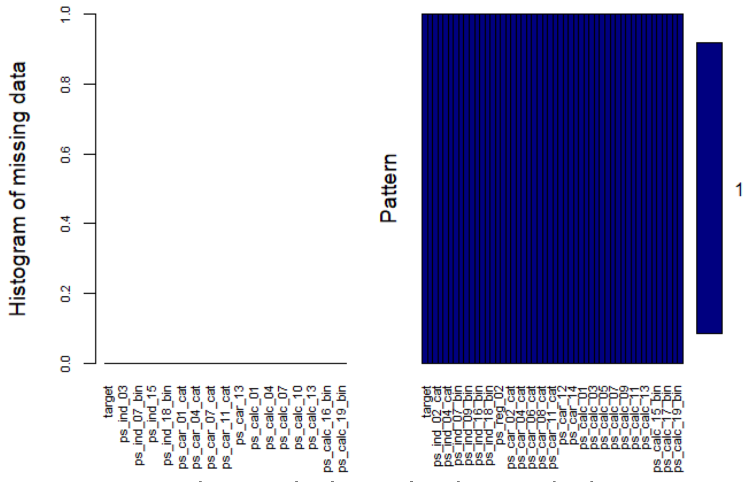
	freqRatio	percentUnique	zeroVar	nzv
<dbl>	<dbl>	<lgl>	<lgl>	
target	26.4367	0.00034	FALSE	TRUE
ps_ind_05_cat	25.5546	0.00134	FALSE	TRUE
ps_ind_10_bin	2680.14	0.00034	FALSE	TRUE
ps_ind_11_bin	590.074	0.00034	FALSE	TRUE
ps_ind_12_bin	104.947	0.00034	FALSE	TRUE

ps_ind_13_bin	1054.34	0.00034	FALSE	TRUE
ps_ind_14	107.158	0.00084	FALSE	TRUE
ps_reg_03	162.307	0.84222	FALSE	TRUE
ps_car_10_cat	121.511	0.0005	FALSE	TRUE

We then identified all the missing values and imputed them with the mean values of their respective attributes.

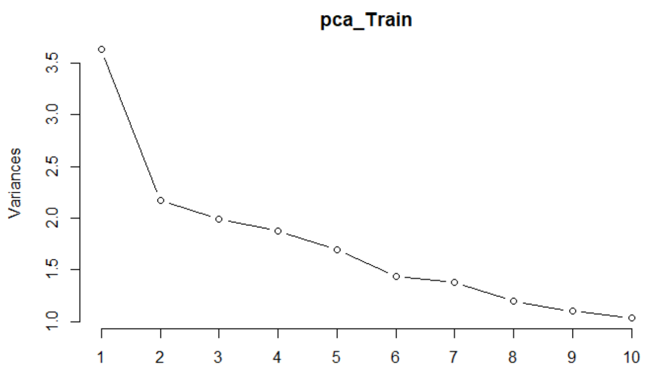


(Prior missing value imputation)

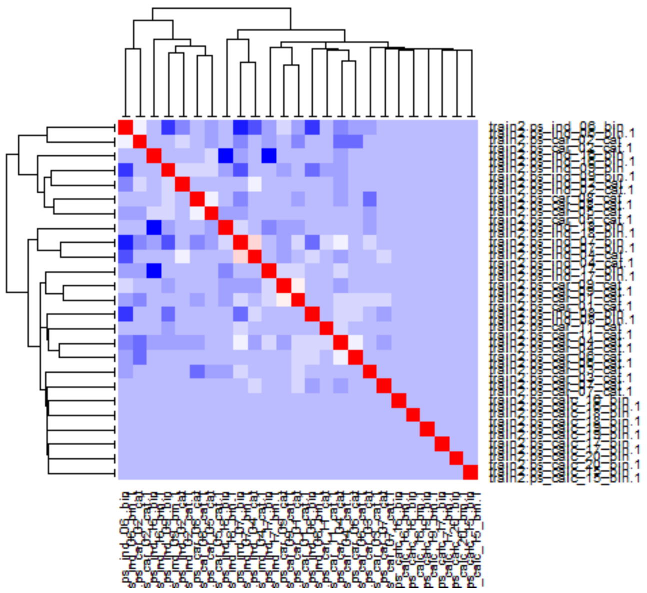


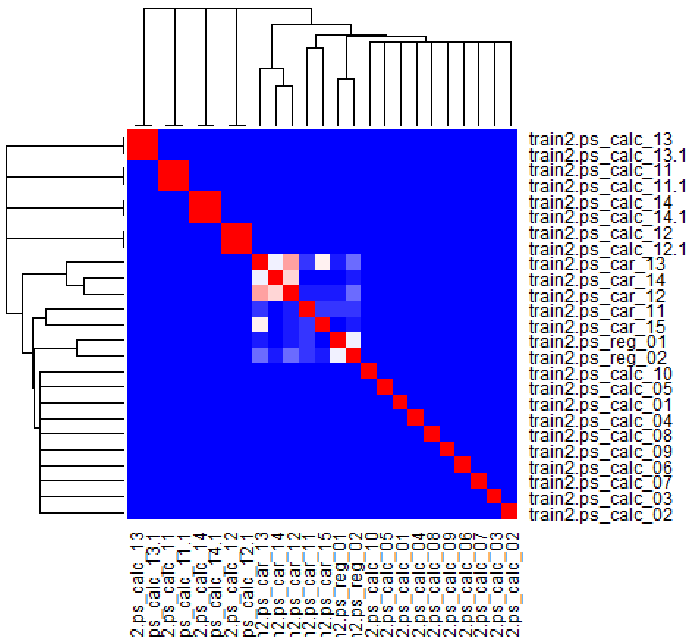
(Post Missing value imputation)

Since our data set has a lot of attributes, we ran the Principal Component Analysis to identify the most important attributes that we should focus on.



In order to get more information about our dataset we reviewed the correlations of categorical and non categorical variables.


(Categorical Variables)



(Non-Categorical Variable)

That concluded our data exploration and preprocessing.

2.7. Train and Test data sets

Since the test dataset did not have target values, we created the target column with NA values. The train data set had 595,212 observations and the test data set had 892,816 observations

3 Data analysis and Validation

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis and easily to implement. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

As a classification method, SVM is a global classification model that generates non-overlapping partitions and usually employs all attributes. The entity space is partitioned in a single pass, so that flat and linear partitions are generated.

Bootstrap Aggregation is a general procedure that can be used to reduce the variance for those algorithms that have high variance. Bagging is the application of the Bootstrap procedure to a high-variance machine learning algorithm, typically decision trees.

We will be using the function accuracy from the R programming language as our basis. The function is as below:

```
accuracy(f, x, test = NULL, d = NULL, D = NULL, ...)
```

We also tried to use Mean Absolute Error (MAE) to measure the actual and forecasted values and found the average. The formula is as below

$$\frac{1}{n}\sum_{n}^{t=1}|A_t - F_t|$$

Root Mean Squared Error (RMSE), Mean Absolute Scaled Error (MASE), and Mean Absolute Percentage Error (MAPE) were used to evaluate our model accuracy.

$\sqrt{\frac{\sum \lvert A_t - F_t \rvert^2}{n}}$
$\left\{\frac{1}{n}\sum\frac{\lvert A_t - F_t\rvert}{\lvert A_t\rvert}\right\} * 100$
$\frac{MAE}{MAE_{NaiveModel}}$

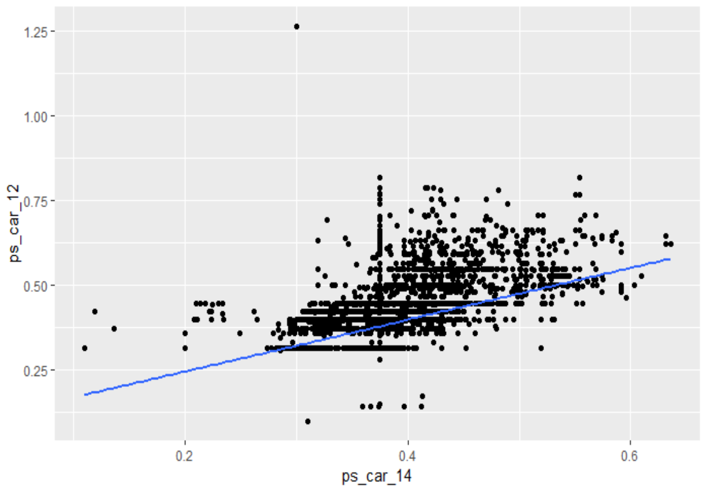
In this project, we applied confusion matrix to see the training result accuracy. At the same time,

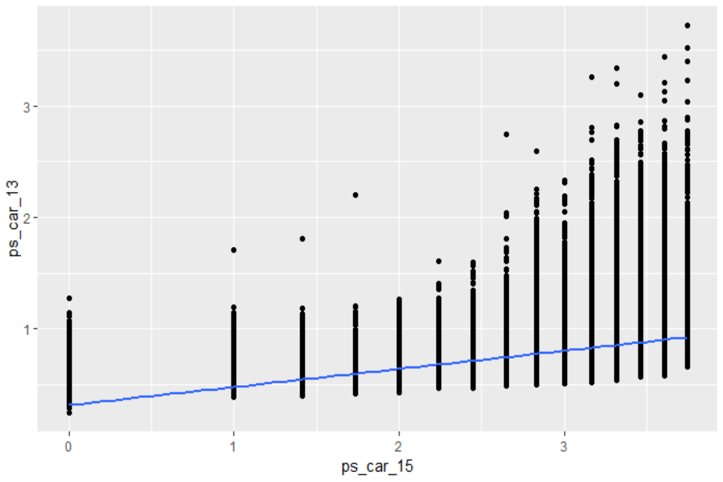
3.1 GLM Model—logistic regression model

Plot data that correlates

```
ggplot(train2, aes(x=ps_car_14, y=ps_car_12)) +
geom_point()+
geom_smooth(method=lm)
```

```
ggplot(train2, aes(x=ps_car_15, y=ps_car_13)) +
 geom_point()+
 geom_smooth(method=lm)
```





GLM Model:

```
Call:
glm(formula = target ~ ., family = "binomial", data = train1)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-1.2361  -0.2945  -0.2534  -0.2198   3.0539  

Coefficients: (1 not defined because of singularities)
                                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)                    -3.813e+00  2.383e-01 -15.999  < 2e-16 ***
ps_ind_01                       1.225e-02  4.000e-03   3.063 0.002189 ** 
ps_ind_02_cat1.35979905747266   4.422e-01  3.672e-01   1.204 0.228496    
ps_ind_02_cat2                  1.309e-01  1.928e-02   6.788 1.14e-11 ***
ps_ind_02_cat3                 -4.504e-02  3.320e-02  -1.356 0.174959    
ps_ind_02_cat4                  6.343e-02  4.971e-02   1.276 0.201947    
ps_ind_03                       2.907e-02  3.088e-03   9.414  < 2e-16 ***
ps_ind_04_cat0.416991946283915  1.575e+00  7.444e-01   2.116 0.034326 *  
…
ps_calc_08                     -4.031e-03  4.754e-03  -0.848 0.396503    
ps_calc_09                      2.094e-03  5.569e-03   0.376 0.706933    
ps_calc_10                      1.764e-03  2.390e-03   0.738 0.460376    
ps_calc_11                      1.093e-03  2.977e-03   0.367 0.713578    
ps_calc_12                     -4.599e-03  5.793e-03  -0.794 0.427336    
ps_calc_13                     -1.202e-03  4.103e-03  -0.293 0.769639    
ps_calc_14                      2.767e-03  2.526e-03   1.096 0.273203    
ps_calc_15_bin1                -5.937e-03  2.126e-02  -0.279 0.779986    
ps_calc_16_bin1                 7.110e-03  1.439e-02   0.494 0.621172    
ps_calc_17_bin1                -1.511e-03  1.398e-02  -0.108 0.913940    
ps_calc_18_bin1                 5.411e-03  1.534e-02   0.353 0.724240    
ps_calc_19_bin1                -1.828e-02  1.462e-02  -1.251 0.211082    
ps_calc_20_bin1                -1.616e-02  1.938e-02  -0.834 0.404282    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 186283  on 595211  degrees of freedom
Residual deviance: 181958  on 595016  degrees of freedom
AIC: 182350

Number of Fisher Scoring iterations: 8
```

ReRun GLM model without variables that are statistically insignificant:

```
Call:
glm(formula = target ~ ., family = "binomial", data = train1_NonSig)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
```

```
-0.5915  -0.2932  -0.2569  -0.2298   2.9556  

Coefficients:
                Estimate Std. Error z value Pr(>|z|)    
(Intercept)    -4.247473   0.061862 -68.660  < 2e-16 ***
ps_ind_01       0.021095   0.003686   5.723 1.04e-08 ***
ps_ind_03       0.010739   0.002674   4.015 5.93e-05 ***
ps_ind_07_bin1  0.333920   0.017148  19.473  < 2e-16 ***
ps_ind_08_bin1  0.260863   0.020151  12.945  < 2e-16 ***
ps_ind_16_bin1 -0.069444   0.029328  -2.368 0.017894 *  
ps_ind_17_bin1  0.393197   0.032501  12.098  < 2e-16 ***
ps_ind_18_bin1  0.075552   0.032843   2.300 0.021424 *  
ps_reg_01       0.288846   0.029264   9.870  < 2e-16 ***
ps_reg_02       0.229459   0.018551  12.369  < 2e-16 ***
ps_car_04_cat1  0.094753   0.028403   3.336 0.000850 ***
ps_car_04_cat2  0.361317   0.040991   8.815  < 2e-16 ***
ps_car_04_cat3  0.152836   0.179573   0.851 0.394709    
ps_car_04_cat4 -0.544738   0.452893  -1.203 0.229055    
ps_car_04_cat5  0.384916   0.168412   2.286 0.022280 *  
ps_car_04_cat6  0.594154   0.110484   5.378 7.54e-08 ***
ps_car_04_cat7  1.028105   0.293104   3.508 0.000452 ***
ps_car_04_cat8  0.204640   0.033550   6.100 1.06e-09 ***
ps_car_04_cat9  0.315505   0.032498   9.708  < 2e-16 ***
ps_car_08_cat1 -0.119534   0.019030  -6.281 3.36e-10 ***
ps_car_11       0.012632   0.010662   1.185 0.236085    
ps_car_15       0.142173   0.012330  11.530  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 186283  on 595211  degrees of freedom
Residual deviance: 183394  on 595190  degrees of freedom
AIC: 183438

Number of Fisher Scoring iterations: 6
```

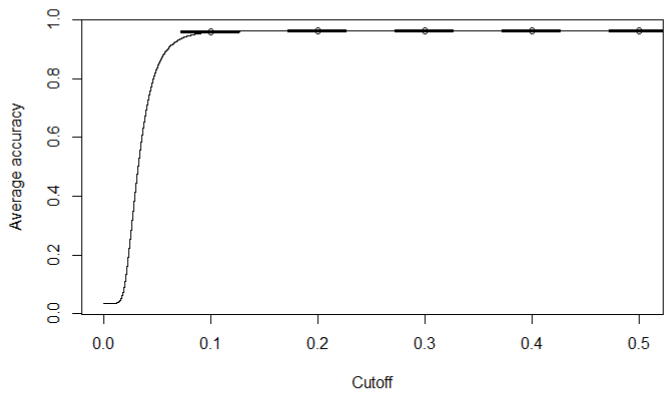
Interpretation of logistic coefficients
For a one-unit change in **ps_ind_01**, the log odds of filling an insurance claim increases by `0.0136186` .
For a one unit increase in **ps_ind_03**, the log odds of filling an insurance claim decreases by `0.010739`
For a one unit increase in `ps_ind_07_bin1` the log odds of filling an insurance claim increases by `0.333920`

```
# or the confustion matrix function and assessment metrics we have seen before
confusionMatrix(as.factor(fit2$fitted.values>0.5), as.factor(fit2$y==1))
Confusion Matrix and Statistics

          Reference
Prediction  FALSE   TRUE
     FALSE 573518  21694
     TRUE       0      0
                                          
               Accuracy : 0.9636          
                 95% CI : (0.9631, 0.964) 
    No Information Rate : 0.9636          
    P-Value [Acc > NIR] : 0.5018          
                                          
                  Kappa : 0               
 Mcnemar's Test P-Value : <2e-16          
                                          
            Sensitivity : 1.0000          
            Specificity : 0.0000          
         Pos Pred Value : 0.9636          
         Neg Pred Value :    NaN          
             Prevalence : 0.9636          
         Detection Rate : 0.9636          
   Detection Prevalence : 1.0000          
      Balanced Accuracy : 0.5000          
```

```
       'Positive' Class : FALSE
```

Prediction of Train dataset Interpretation:
According to the prediction model, 57351 individuals where predicted to not file a claim next year which is in accordance with the actual model. 21694 individuals where predicted to file an insurance in the following year, which is not in accordance with model

```
# can also plot accuracy by average cutoff level
perf <- performance(pred, "acc")
plot(perf, avg= "vertical",
     spread.estimate="boxplot",
     show.spread.at= seq(0.1, 0.9, by=0.1))
```



```
#residuals
pearsonRes <-residuals(fit2,type="pearson")
devianceRes <-residuals(fit2,type="deviance")
rawRes <-residuals(fit2,type="response")
studentDevRes<-rstudent(fit2)
fv<-fitted(fit2)

#dataframe of target variable, predicted variable and residuals
predVals <- data.frame(trueVal=train1_NonSig$target, predClass=train1_NonSig$pred, predProb=fv,
rawRes, pearsonRes, devianceRes, studentDevRes)

tail(predVals)
```

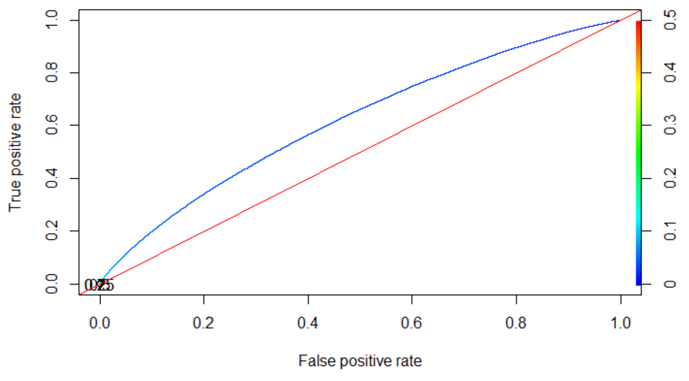
	trueVal	predClass	predProb	rawRes	pearsonRes	devianceRes	studDevRes
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
595207	0	0	0.03106659	-0.03106659	-0.1790605	-0.2512345	-0.2512374
595208	0	0	0.02561907	-0.02561907	-0.1621501	-0.2278287	-0.2278293
595209	0	0	0.03839176	-0.03839176	-0.1998112	-0.2798147	-0.2798164
595210	0	0	0.02370759	-0.02370759	-0.155831	-0.2190577	-0.2190587
595211	0	0	0.03921609	-0.03921609	-0.2020316	-0.2828631	-0.2828646
595212	0	0	0.02662284	-0.02662284	-0.1653814	-0.2323086	-0.2323097

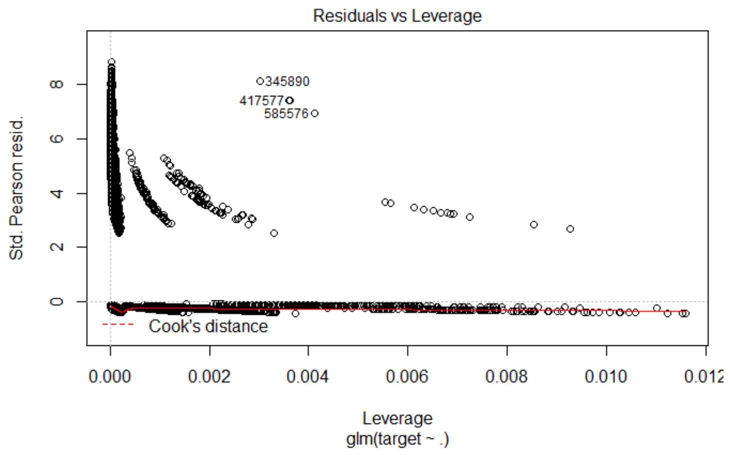
```
library(gmodels)
CrossTable(train1_NonSig$pred,train1_NonSig$target, chisq=T)
```

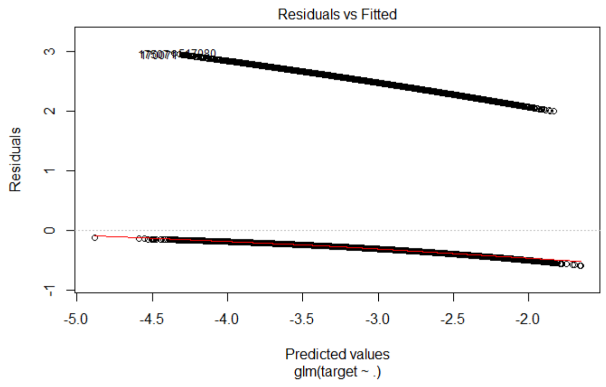
```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  595212
```

```
                    | train1_NonSig$target
train1_NonSig$pred |           0 |           1 |   Row Total |
--------------------|-------------|-------------|-------------|
                  0 |      573518 |       21694 |      595212 |
                    |       0.964 |       0.036 |             |
--------------------|-------------|-------------|-------------|
       Column Total |      573518 |       21694 |      595212 |
--------------------|-------------|-------------|-------------|
```

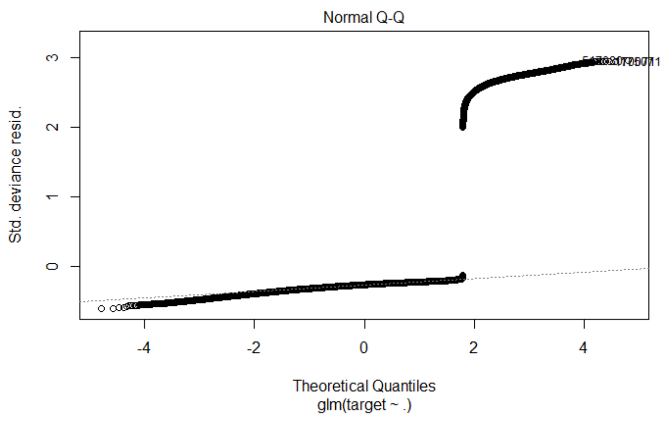
```
pred <- prediction(fit2$fitted, fit2$y)    #ROC curve for training data
perf <- performance(pred,"tpr","fpr")
plot(perf,colorize=TRUE, print.cutoffs.at = c(0.25,0.5,0.75));
abline(0, 1, col="red")
```







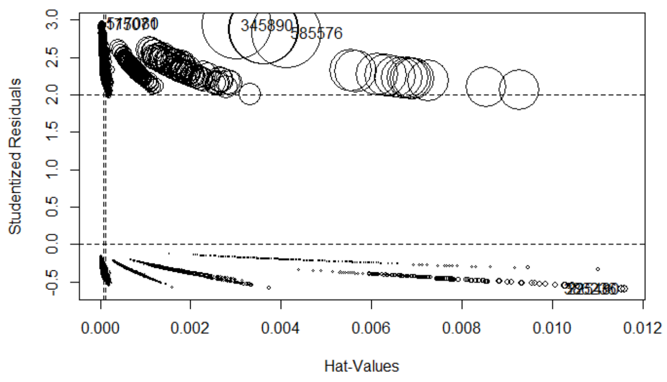
The plot of standardized residuals vs Leverage is shown above. Cook's distance is also indicated in the plot above. This plot is used to locate influential points and measures the impact of their presence or absence on the data.



The variance-inflation and generalized variance-inflation factors for linear models test for multicollinearity.

```
                  GVIF Df GVIF^(1/(2*Df))
ps_ind_01     1.191340  1        1.091485
ps_ind_03     1.075325  1        1.036979
ps_ind_07_bin 1.352419  1        1.162936
ps_ind_08_bin 1.288919  1        1.135306
ps_ind_16_bin 4.303518  1        2.074492
ps_ind_17_bin 3.236279  1        1.798966
ps_ind_18_bin 3.040864  1        1.743807
ps_reg_01     1.372415  1        1.171501
ps_reg_02     1.400350  1        1.183364
ps_car_04_cat 2.094945  9        1.041941
ps_car_08_cat 1.227534  1        1.107941
ps_car_11     1.721308  1        1.311986
ps_car_15     1.271799  1        1.127741
```

```
influencePlot(fit2)
```



The above figure is a plot of studentized Residuals vs. Hat-Values. This plot provides the influential points of the data. Influential points have a great effect on the data and when they are the deleted the data becomes extremely skewed. The figure above shows that the influential points are a several rows including 345890 and 585572.

```
predTest<- predict(fit2, newdata = test1, type = "response")
head(predTest)
test1$target=predTest
head(test1)
write.csv(predTest, file = "logisticReg1.csv")
```

ps_car_10_cat	ps_car_11	ps_car_12	ps_car_13	ps_car_15	target
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
1	1	0.31623	0.6695564	3.464102	0.03563
1	1	0.31623	0.60632	2.828427	0.02906
1	3	0.4	0.8962387	3.316625	0.02648
1	2	0.37417	0.6521104	2.44949	0.01938
1	3	0.37417	0.8129144	3.316625	0.03246
1	2	0.31623	0.7509223	3.605551	0.03423

3.2.2 SVM model

SVMs are based on maximum margin linear discriminants, and are similar to probabilistic approaches, but do not consider the dependencies among attributes. Create a SVM model using the default parameters. We will also set the CV predictions and n folds to 5 to enable cross validation.

```
fit2 <- bagging(target ~ ., data = train, coob = T)
```

3.2.3 Bagging Model

Bagging regression trees with 25 bootstrap replications

Call: bagging.data.frame(formula = target ~ ., data = subtrain, coob = T)

Out-of-bag estimate of root mean squared error: 0.1911

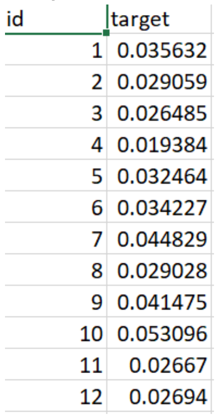
4 Results and validation of analysis

The results of analysis we discussed above are shown in Table1. The accuracy, RMSE, MAE, and other analysis results are shown in table. As we see from the table, the winner's selected features result in the highest accuracy of   96.36% using logistic regression model.

Model	Results Evaluation
Logistic Regression glm	Accuracy - 96.36%
SVM	The dataset was too large for us to properly run and generate formal results.
Bagging	RSME- 0.911

4.1. Results

GLM model gave us the accuracy of 96.36% for training data set. Here is the csv file we obtained by glm model. The possibilities are in the "target" column.



id	target
1	0.035632
2	0.029059
3	0.026485
4	0.019384
5	0.032464
6	0.034227
7	0.044829
8	0.029028
9	0.041475
10	0.053096
11	0.02667
12	0.02694

4.2. Validation of analysis

Though the ROC curve we can see that the performance of our model is good.

5 Conclusion

5.1. Conclusion

The goal of this project is to build a model which will give the auto insurance company more accurate probability of a driver who will initiate an insurance in the next year. For this project, we tested different kinds of models mentioned in the former part. We also wanted to apply XGBoost model which was quite popular with better accuracy result, however with the limitation of the personal laptop, we failed to do so.

In this project, it was observed that a glm model applied to all training data resulted in the highest accuracy. This must be validated with future work focused on a more varied data set with different native languages. Overall, the results of the project support the hypothesis that feature selection has a high impact on accuracy regardless of model chosen.

5.2. Issues

We met several issues throughout this project which had to be overcome. Our data set was a large one which was 110MB, so we fread option from package data.table to load big data set which is efficient. When we dummied the variables, the memory was not big enough to implement that task. We learned from google to apply ff package to fix this problem. The models we used for this project were not the most accurate ones. We wanted to test XGBoost model, but the program failed during the modeling.

We know that Neural Network usually give more robust and accurate models, but we did not learn these two models. At the same time, we can use different methods to measure the results and accuracy of our models.

5.3. Future Work

All the issues that were not addressed in the current project should be fixed. Even though accurate ranking was produced every time the code was run we did encounter unexplained variability with the accuracy rate results. It will be interesting to study more datasets like this projects to make predictions with limited variables.