Multiple Linear Regression and Binary Logistic Regression

Data 621: Homework 4 - Group 4

Sachid Deshmukh

Michael Yampol

Vishal Arora

Ann Liu-Ferrara

Nov. 10, 2019

1. Data Exploration

Let's load training dataset and preview.

	IND… <int>	TARGET_FLAG <int>	TARGET_AMT <dbl>	KIDSDRIV <int>	… <int>	HOMEKI… <int>	… <int>	INCOME <chr>	PARENT1 <chr>
1	1	0	0	0	60	0	11	$67,349	No
2	2	0	0	0	43	0	11	$91,449	No
3	4	0	0	0	35	1	10	$16,039	No
4	5	0	0	0	51	0	14	*NA*	No
5	6	0	0	0	50	0	*NA*	$114,986	No
6	7	1	2946	0	34	1	12	$125,301	Yes

6 rows | 1-10 of 27 columns

```
## [1] "Number of columns =  26"
```

```
## [1] "Number of rows =  8161"
```

As we can see in the preview, training data has 26 columns and 8161 rows

Let's analyze datatypes of each column.

```
## 'data.frame':    8161 obs. of  26 variables:
##  $ INDEX      : int  1 2 4 5 6 7 8 11 12 13 ...
##  $ TARGET_FLAG: int  0 0 0 0 0 1 0 1 1 0 ...
##  $ TARGET_AMT : num  0 0 0 0 0 ...
##  $ KIDSDRIV   : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ AGE        : int  60 43 35 51 50 34 54 37 34 50 ...
##  $ HOMEKIDS   : int  0 0 1 0 0 1 0 2 0 0 ...
##  $ YOJ        : int  11 11 10 14 NA 12 NA NA 10 7 ...
##  $ INCOME     : chr  "$67,349" "$91,449" "$16,039" NA ...
##  $ PARENT1    : chr  "No" "No" "No" "No" ...
##  $ HOME_VAL   : chr  "$0" "$257,252" "$124,191" "$306,251" ...
##  $ MSTATUS    : chr  "z_No" "z_No" "Yes" "Yes" ...
##  $ SEX        : chr  "M" "M" "z_F" "M" ...
##  $ EDUCATION  : chr  "PhD" "z_High School" "z_High School" "<High School" ...
##  $ JOB        : chr  "Professional" "z_Blue Collar" "Clerical" "z_Blue Collar" ...
##  $ TRAVTIME   : int  14 22 5 32 36 46 33 44 34 48 ...
##  $ CAR_USE    : chr  "Private" "Commercial" "Private" "Private" ...
##  $ BLUEBOOK   : chr  "$14,230" "$14,940" "$4,010" "$15,440" ...
##  $ TIF        : int  11 1 4 7 1 1 1 1 1 7 ...
##  $ CAR_TYPE   : chr  "Minivan" "Minivan" "z_SUV" "Minivan" ...
##  $ RED_CAR    : chr  "yes" "yes" "no" "yes" ...
##  $ OLDCLAIM   : chr  "$4,461" "$0" "$38,690" "$0" ...
##  $ CLM_FREQ   : int  2 0 2 0 2 0 0 1 0 0 ...
##  $ REVOKED    : chr  "No" "No" "No" "No" ...
##  $ MVR_PTS    : int  3 0 3 0 3 0 0 10 0 1 ...
##  $ CAR_AGE    : int  18 1 10 6 17 7 1 7 1 17 ...
##  $ URBANICITY : chr  "Highly Urban/ Urban" "Highly Urban/ Urban" "Highly Urban/ Urban" "Highl
y Urban/ Urban" ...
```

As we can see Income, Parent1, Home_val, MStatus, Sex, Education, Job, Car_Use, BlueBook, Car_Type, Red_Car, OldClaim, Revoked and UrbaniCity variables are qualitative. Linear Regression and Logistic regression algorithms works best with numerical variables. We need to transform these variagles making them quantitative so that we can use them for model training

Let's check if any variables have missing values. Values which are NULL or NA.

Check missing values

```
## [1] "Number of columns with missing values =  6"
```

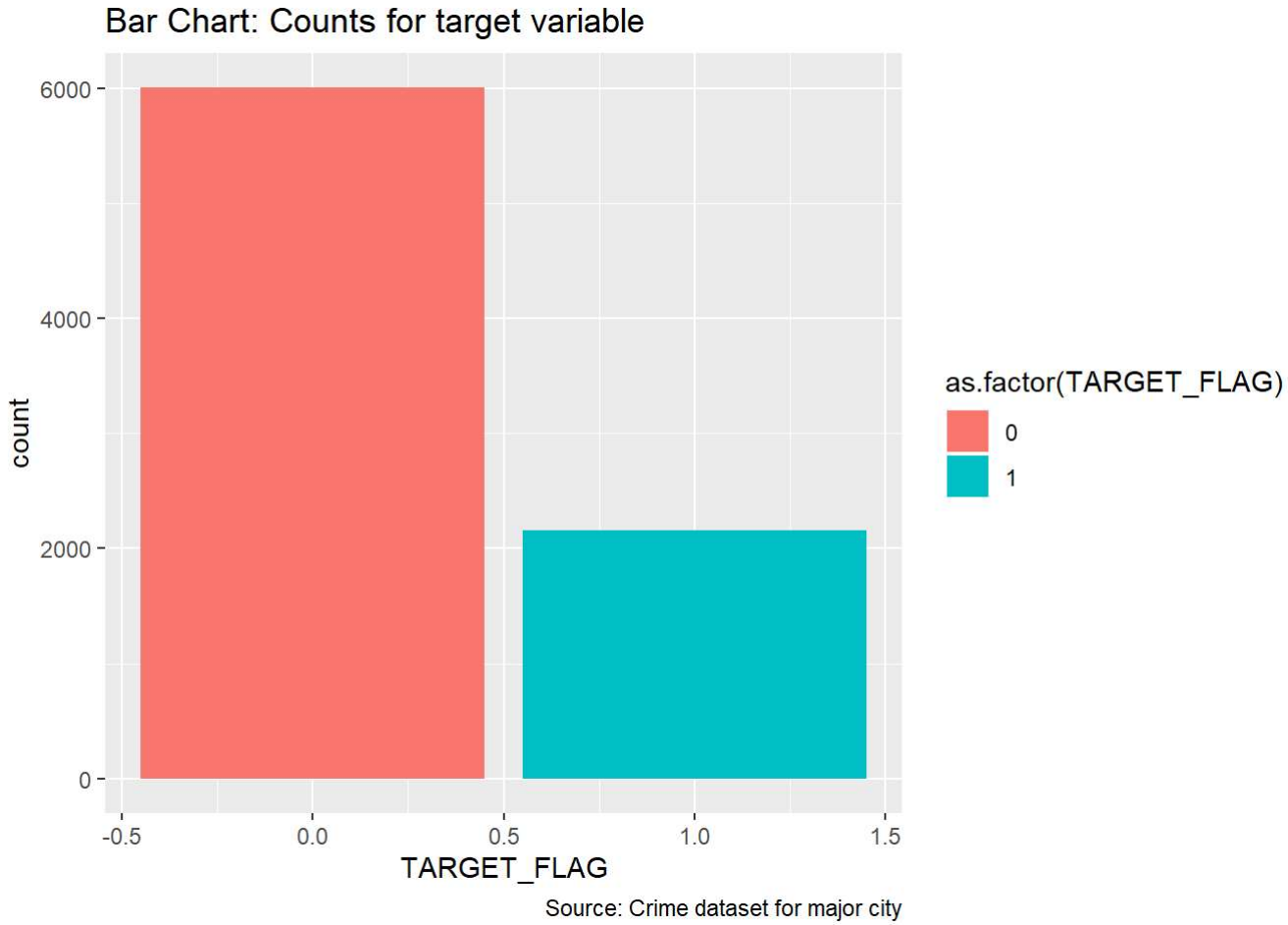
```
## [1] "Names of columns with missing values =  AGE, YOJ, INCOME, HOME_VAL, JOB, CAR_AGE"
```

We can see 6 variables namely Age, Yoj, Income, Home_VAL, Job and Car_Age have missing values. We will impute values for these variables for better model accuracy.

Let's check if the training data is class imbalance. A Dataset is called class imbalance when there are very few obesrvations corresponding to minority class. This is very

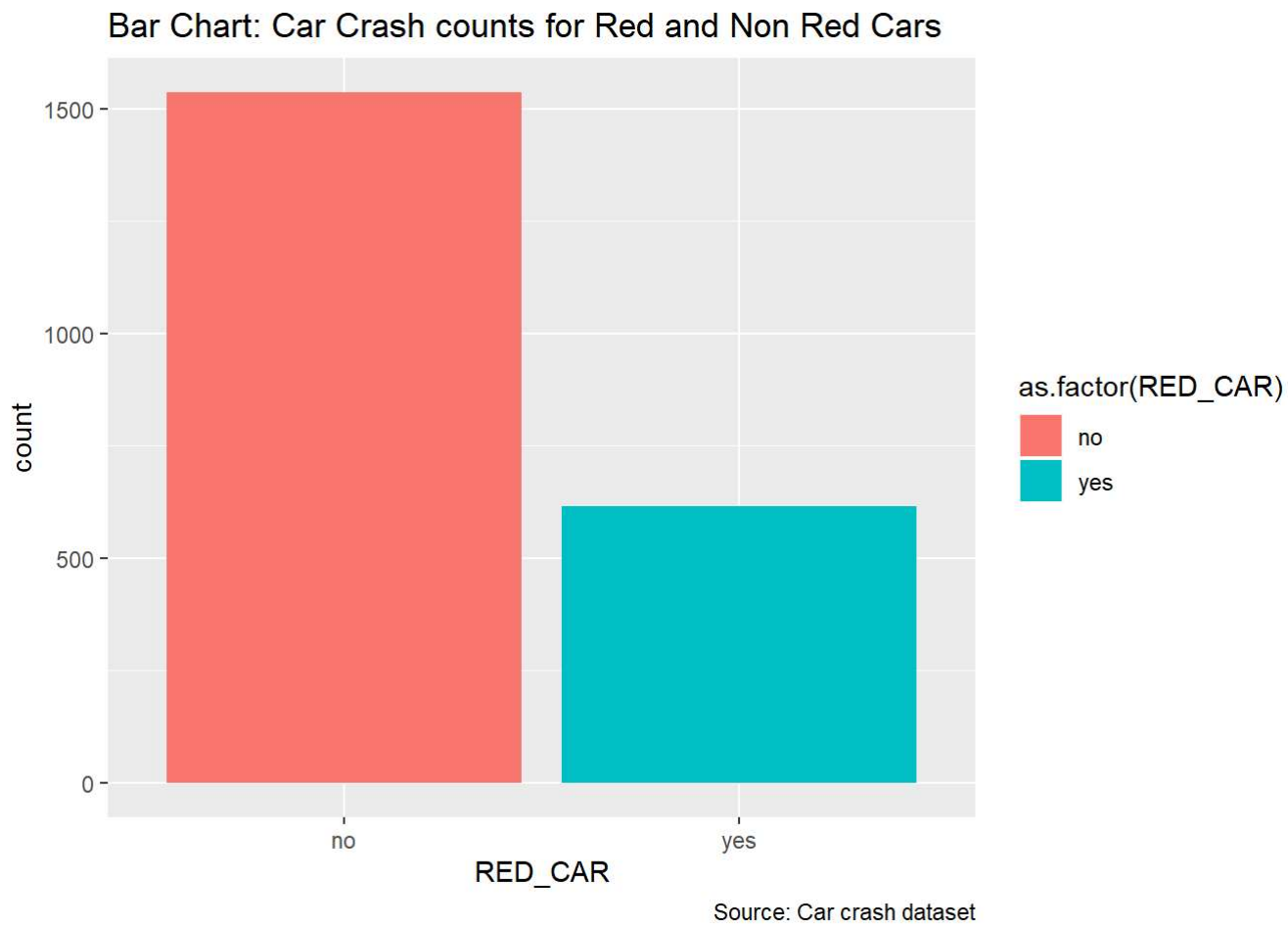
important in deciding model evaluation metrics.

Check class Imbalance



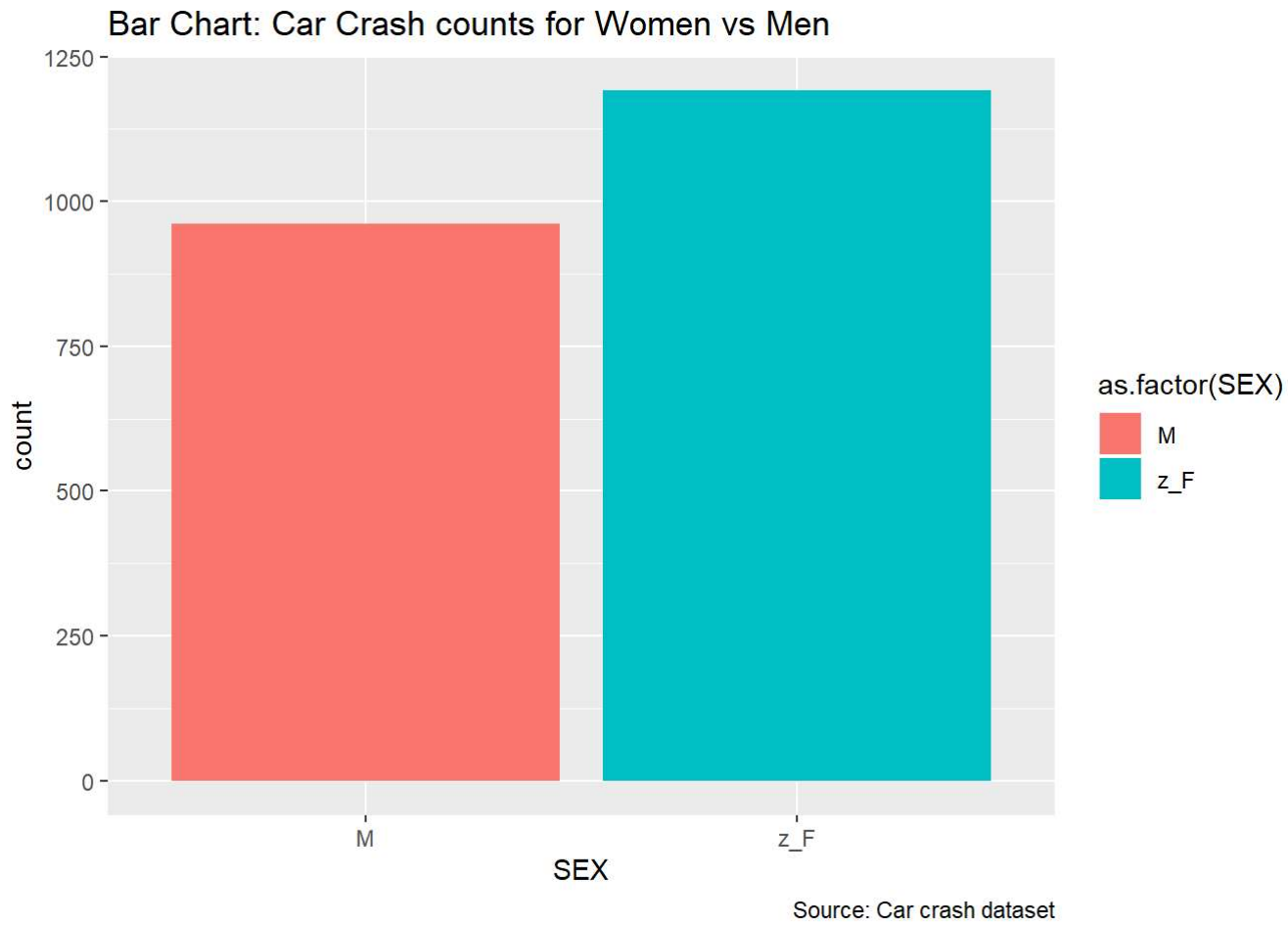
Above barchart indicates that training dataset is class imbalance. There are fewer observation of car crashes compared to observations with no car crash. This makes the dataset class imbalance. For logistic regression we can't rely on model metrics like Accuracy due to this. Since the dataset is class imbalance, we will give more importance to Precision, Recall and ROC AUC for evaluating logistic regression model.

Let's check if Red color contributes to more car crashes



Above bar chart doesn't show significant evidence that Red cars are more accident prone compare to non red cars

Let's check if women are safe driver compared to men



Above bar chart shows that women have more car crashes compared to men. We can discard the fact that in general women are safer driver than men

2. Data Preparation

Encode Parent1 variable. No = 0 and Yes = 1

```
train.df$PARENT1 = ifelse(train.df$PARENT1 == 'No', 0, 1)
train.df$PARENT1 = as.numeric(train.df$PARENT1)
```

Encode MStatus variable. No = 0 and Yes = 1

```
train.df$MSTATUS = ifelse(train.df$MSTATUS == 'z_No', 0, 1)
train.df$MSTATUS = as.numeric(train.df$MSTATUS)
```

Encode Sex variable. Male = 0 and Female = 1

```
train.df$SEX = ifelse(train.df$SEX == 'M', 0, 1)
train.df$SEX = as.numeric(train.df$SEX)
```

Encode Education variable.

```
train.df$EDUCATION = as.numeric(factor(train.df$EDUCATION, order = TRUE, levels = c("<High Schoo
l", "z_High School", "Bachelors", "Masters", "PhD")))
```

Encode Job variable.

```
train.df$JOB = as.numeric(factor(train.df$JOB, order = TRUE, levels = c("Student", "Home Maker",
"z_Blue Collar", "Clerical", "Professional", 'Manager', 'Lawyer', 'Doctor')))
```

Encode Car_Use variable. Private = 0 and Commercial = 1

```
train.df$CAR_USE = ifelse(train.df$CAR_USE == "Private", 0, 1)
train.df$CAR_USE  = as.numeric(train.df$CAR_USE)
```

Encode Job variable.

```
train.df$CAR_TYPE = as.numeric(factor(train.df$CAR_TYPE, order = TRUE, levels = c("Minivan", "z_
SUV", "Van", "Pickup", "Panel Truck", 'Sports Car')))
```

Encode Red_car variable. No = 0 and Yes = 1

```
train.df$RED_CAR = ifelse(train.df$RED_CAR == "no", 0, 1)
train.df$RED_CAR  = as.numeric(train.df$RED_CAR)
```

Encode Revoked variable. No = 0 and Yes = 1

```
train.df$REVOKED = ifelse(train.df$REVOKED == "No", 0, 1)
train.df$REVOKED  = as.numeric(train.df$REVOKED)
```

Encode Urban city variable. Rural = 0 and Urban = 1

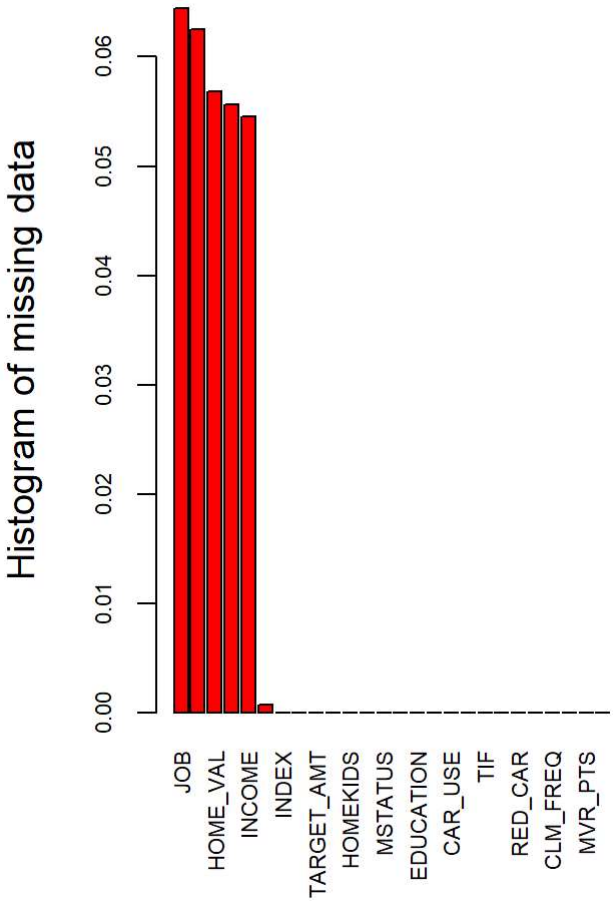
```
train.df$URBANICITY = ifelse(train.df$URBANICITY == "z_Highly Rural/ Rural", 0, 1)
train.df$URBANICITY  = as.numeric(train.df$URBANICITY)
```

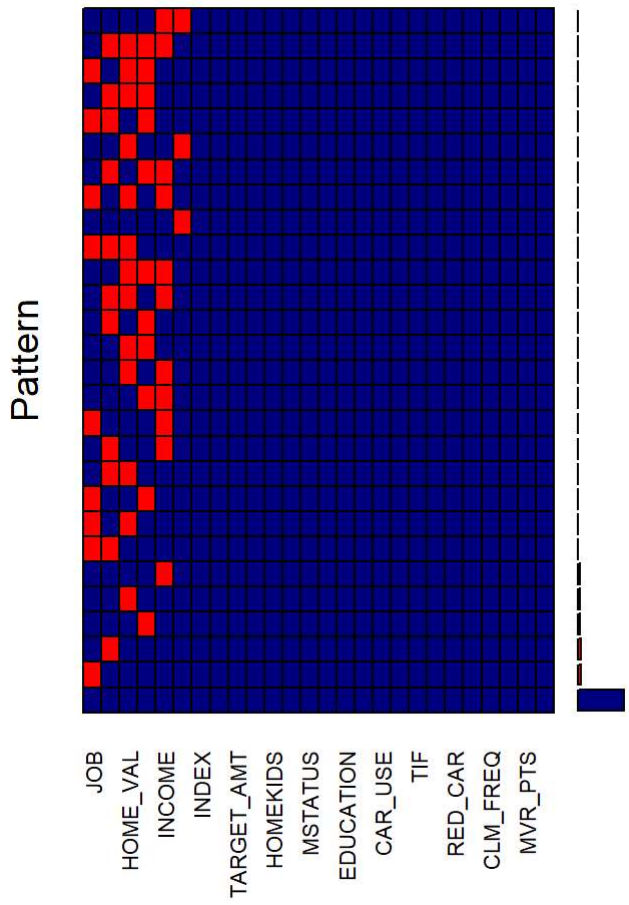
Convert Income, Home_val, BlueBook, oldClaim Variable to quantitative variable

Let’s do data imputation for missing columns

Which columns are mssing and what is a missing pattern. Let’s leverage VIM package to get this information

```
## Warning in plot.aggr(res, ...): not enough horizontal space to display
## frequencies
```


Histogram of missing data
0.00
0.01
0.02
0.03
0.04
0.05
0.06
JOB
HOME_VAL
INCOME
INDEX
TARGET_AMT
HOMEKIDS
MSTATUS
EDUCATION
CAR_USE
TIF
RED_CAR
CLM_FREQ
MVR_PTS


Pattern
JOB
HOME_VAL
INCOME
INDEX
TARGET_AMT
HOMEKIDS
MSTATUS
EDUCATION
CAR_USE
TIF
RED_CAR
CLM_FREQ
MVR_PTS

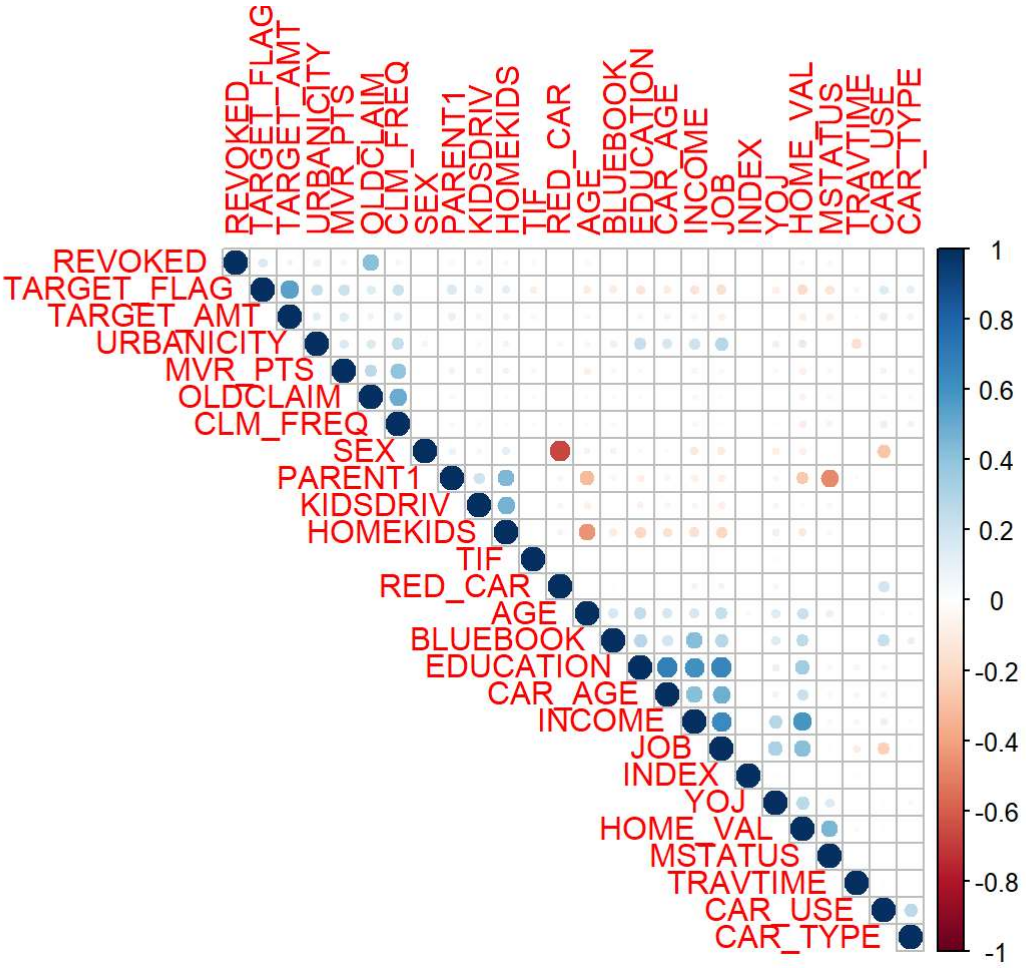
```
##
##  Variables sorted by number of missings:
##     Variable       Count
##          JOB 0.064452886
##      CAR_AGE 0.062492342
##     HOME_VAL 0.056855777
##          YOJ 0.055630437
##       INCOME 0.054527631
##          AGE 0.000735204
##        INDEX 0.000000000
##  TARGET_FLAG 0.000000000
##   TARGET_AMT 0.000000000
##     KIDSDRIV 0.000000000
##     HOMEKIDS 0.000000000
##      PARENT1 0.000000000
##      MSTATUS 0.000000000
##          SEX 0.000000000
##    EDUCATION 0.000000000
##     TRAVTIME 0.000000000
##      CAR_USE 0.000000000
##     BLUEBOOK 0.000000000
##          TIF 0.000000000
##     CAR_TYPE 0.000000000
##      RED_CAR 0.000000000
##     OLDCLAIM 0.000000000
##     CLM_FREQ 0.000000000
##      REVOKED 0.000000000
##      MVR_PTS 0.000000000
##   URBANICITY 0.000000000
```

From the above missing values pattern we can see that most of the obervation don't have missing values. Non missing values are shown in blue. This is a good news and we can assert good quality of data in this case.

Let's use MICE package to imput missing values

```
##
##  iter imp variable
##   1   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   1   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   2   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   2   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   3   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   3   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   4   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   4   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   5   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   5   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   6   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   6   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   7   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   7   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   8   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   8   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   9   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   9   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   10   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   10   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
```

The variables which are highly correlated carry similar information and can affect model accuracy. Highly correlated variables also impacts estimation of model coefficients. Let's figure out which variables in the training datasets are highly correlated to each other



From the above correlation graph we can see that target variable TARGET_FLAG is highly correlated with following variables

- **REVOKED: License Revoked**
- **MVR_PTS: Motor Vehicle Record Points**
- **OLD_CLAIM: Total Claims**
- **CLAIM_FREQ: Claim Frequency**
- **URBANICITY: Home/Work Area**
- **JOB: Job**
- **INCOME: Income**
- **HOME_VAL: Home Value**
- **MSTATUS: Maritial Status**
- **CAR_USE: Use of car**

In addition to above variables, following variables are important for Target_Amt output variable

- **BLUEBOOK: Value of vehicle**
- **CAR_AGE: Age of car**
- **CAR_TYPE: TType of car**

Add interaction terms to our dataset

```
train.df$JOB_EDU =  round(log(train.df$JOB * train.df$EDUCATION))
train.df$HOME_INCOME = log(1+ train.df$HOME_VAL) * log(1 + train.df$INCOME)
train.df$MVR_PTS_Trans = round(log(1+ train.df$MVR_PTS))
train.df$AGE_SEX =log(1 + train.df$AGE) * (1+train.df$SEX)
```

3. Build Models - Logistic Regression

Model fitting and evaluation

For model evaluation we will use train test split technique. Since goal of the exercise is to predict car crashes, we will build high recall model

High Recall Model

High recall model focuses on identifying maximum possible positive instance. In this case it means we are optimizing our model to identify as much potential car crash target as possible. Note that sometimes this can come at the cost of precision where we might get high number of false positives

Model Evaluation Metrics

We will use following metrics for model evaluation and comparison

- **ROC AUC** : AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is
- **Model Accuracy** : Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition: Accuracy = (TP + TN)/(TP + FP + TN + FN)
- **Model Recall** : Recall is a metrics that focuses on how many true positives are identified from total positive observations in the data set. Formally Recall has following definition: Recall = TP/TP + FN
- **Model Precision** : Model precision is a metric that focuses on how many observations are truly positive out of totally identified positives. Formally Precision has following definition : Precision = TP/TP + FP

Where

- **TP** : Stands for True Positives
- **TN** : Stands for True Negatives
- **FP** : Stands for False Positives
- **FN** : Stands for False Negatives

Now we are clear on our model fitting and evaluation method (Train test split) and also have model evaluation metrics (Recall) which we will use to compare the model effectiveness

we are all set to build different models and access it's performance

We will build three different models and compare them using above mentioned model metrics

1. Let's build model with important predictors as per above analysis

- **REVOKED: License Revoked**
- **MVR_PTS: Motor Vehicle Record Points**
- **OLD_CLAIM: Total Claims**
- **CLAIM_FREQ: Claim Frequency**
- **URBANICITY: Home/Work Area**
- **JOB: Job**
- **INCOME: Income**
- **HOME_VAL: Home Value**
- **MSTATUS: Maritial Status**
- **CAR_USE: Use of car**
- **CAR_TYPE: Type of Car**

1. Model with selected important variables

```
model.metrix = model.fit.evaluate("target ~ INCOME + HOME_VAL + MSTATUS  + JOB + CAR_USE + CAR_T
YPE + OLDCLAIM + CLM_FREQ + REVOKED + MVR_PTS + URBANICITY", train.data, test.data)
print.model.matrix("Base Model", model.metrix)
```

```
## [1] "Printing Metrix for model:  Base Model"
## [1] "AUC : 0.781038017616685"
## [1] "Accuracy : 0.768068599428338"
## [1] "Recall : 0.287650602409639"
## [1] "Precision : 0.667832167832168"
```

2. Model with all the predictor with transformed variables

We will add new variables to model

```
model.metrix = model.fit.evaluate("target ~ KIDSDRIV + AGE + HOMEKIDS + YOJ + INCOME + PARENT1 +
HOME_VAL + MSTATUS+ SEX + EDUCATION + JOB + TRAVTIME + CAR_USE + BLUEBOOK  + TIF + CAR_TYPE + RE
D_CAR + OLDCLAIM + CLM_FREQ + REVOKED + MVR_PTS + CAR_AGE + URBANICITY + JOB_EDU + MVR_PTS_Trans
+ HOME_INCOME + AGE_SEX", train.data, test.data)
print.model.matrix("Interaction Term", model.metrix)
```

```
## [1] "Printing Metrix for model:  Interaction Term"
## [1] "AUC : 0.794767304512163"
## [1] "Accuracy : 0.775826868109432"
## [1] "Recall : 0.337349397590361"
## [1] "Precision : 0.672672672672673"
```

3. Model with balanced training set

Since we know that data is class imbalance, we will use ROSE package and leverage sythetic data generation technique to balance the training data.

```
train.data.balanced <- ROSE(target ~ ., data = train.data, seed = 1)$data
model.metrix = model.fit.evaluate("target ~ KIDSDRIV + AGE + HOMEKIDS + YOJ + INCOME + PARENT1 +
HOME_VAL + MSTATUS+ SEX + EDUCATION + JOB + TRAVTIME + CAR_USE + BLUEBOOK  + TIF + CAR_TYPE + RE
D_CAR + OLDCLAIM + CLM_FREQ + REVOKED + MVR_PTS + CAR_AGE + URBANICITY + JOB_EDU + MVR_PTS_Trans
+ HOME_INCOME + AGE_SEX", train.data.balanced, test.data)
print.model.matrix("Class Balanced", model.metrix)
```

```
## [1] "Printing Metrix for model:  Class Balanced"
## [1] "AUC : 0.790119300732339"
## [1] "Accuracy : 0.746835443037975"
## [1] "Recall : 0.635542168674699"
## [1] "Precision : 0.5275"
```

From the above results we can see that model which uses all the variables along with newly added interaction term and which has class balance data performs better. We will use this model for prediction

4. Model coefficient analysis

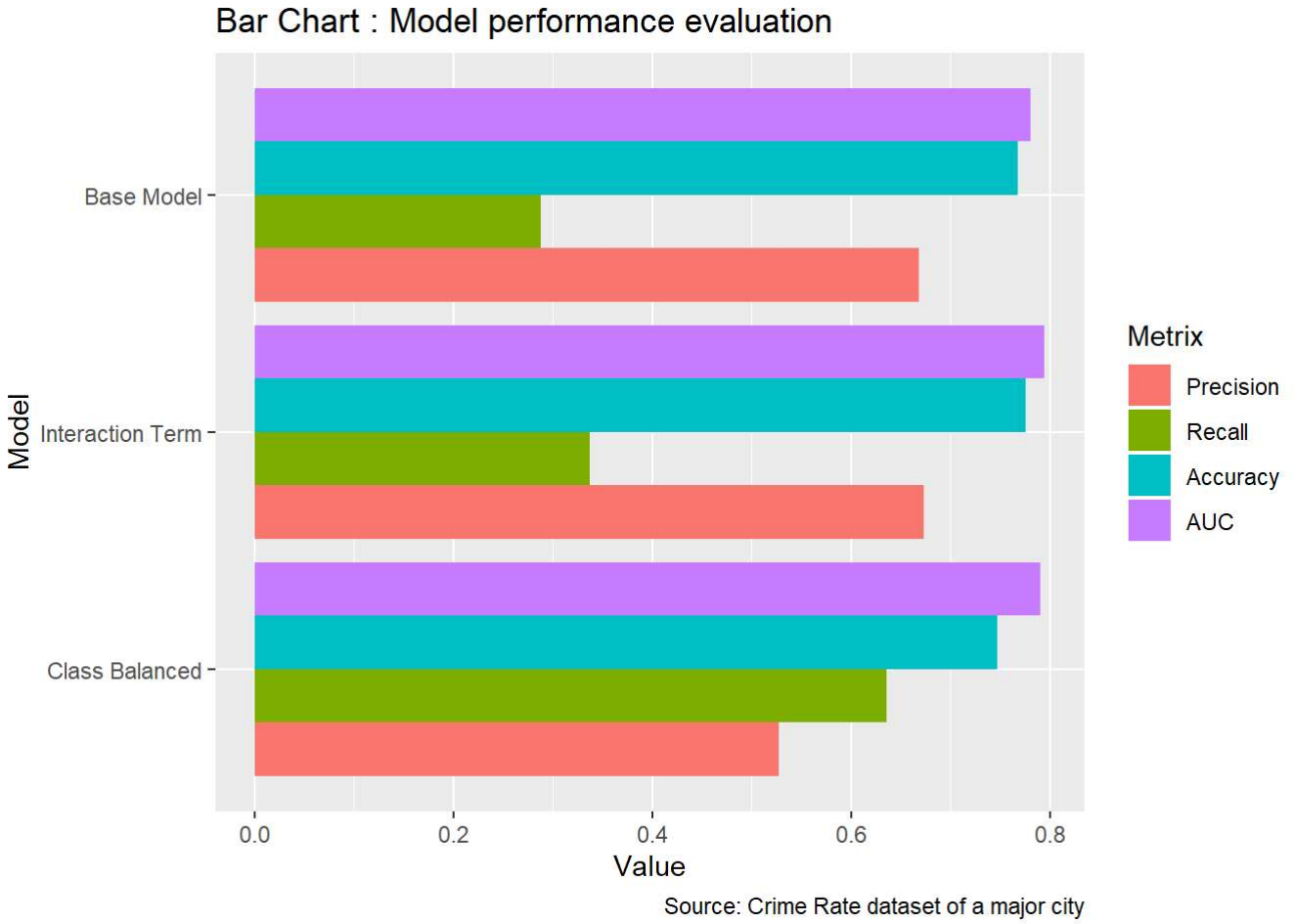
```
fit = glm(formula = "target ~ KIDSDRIV + AGE + HOMEKIDS + YOJ + INCOME + PARENT1 + HOME_VAL + MS
TATUS+ SEX + EDUCATION + JOB + TRAVTIME + CAR_USE + BLUEBOOK  + TIF + CAR_TYPE + RED_CAR + OLDCL
AIM + CLM_FREQ + REVOKED + MVR_PTS + CAR_AGE + URBANICITY + JOB_EDU + MVR_PTS_Trans + HOME_INCOM
E + AGE_SEX", data = train.df.class)

summary(fit)
```

```
##
## Call:
## glm(formula = "target ~ KIDSDRIV + AGE + HOMEKIDS + YOJ + INCOME + PARENT1 + HOME_VAL + MSTAT
US+ SEX + EDUCATION + JOB + TRAVTIME + CAR_USE + BLUEBOOK  + TIF + CAR_TYPE + RED_CAR + OLDCLAIM
+ CLM_FREQ + REVOKED + MVR_PTS + CAR_AGE + URBANICITY + JOB_EDU + MVR_PTS_Trans + HOME_INCOME +
AGE_SEX",
##     data = train.df.class)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9871  -0.2812  -0.1128   0.2871   1.1862
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.192e-01  9.737e-02   2.251 0.024417 *
## KIDSDRIV       6.135e-02  9.760e-03   6.286 3.42e-10 ***
## AGE            1.039e-03  1.501e-03   0.692 0.488812
## HOMEKIDS       5.686e-03  5.618e-03   1.012 0.311566
## YOJ            2.048e-04  1.284e-03   0.160 0.873257
## INCOME        -7.131e-07  1.705e-07  -4.182 2.91e-05 ***
## PARENT1        6.852e-02  1.735e-02   3.950 7.88e-05 ***
## HOME_VAL       1.524e-07  9.602e-08   1.587 0.112449
## MSTATUS       -6.807e-02  1.280e-02  -5.319 1.07e-07 ***
## SEX            1.856e-01  1.522e-01   1.219 0.222801
## EDUCATION     -3.203e-02  8.504e-03  -3.766 0.000167 ***
## JOB           -1.491e-02  5.132e-03  -2.906 0.003674 **
## TRAVTIME       2.008e-03  2.763e-04   7.267 4.00e-13 ***
## CAR_USE        1.188e-01  1.127e-02  10.545  < 2e-16 ***
## BLUEBOOK      -4.058e-06  5.880e-07  -6.901 5.54e-12 ***
## TIF           -7.678e-03  1.044e-03  -7.356 2.07e-13 ***
## CAR_TYPE       1.722e-02  2.707e-03   6.362 2.10e-10 ***
## RED_CAR       -9.730e-03  1.278e-02  -0.762 0.446373
## OLDCLAIM      -2.398e-06  6.373e-07  -3.762 0.000170 ***
## CLM_FREQ       3.378e-02  4.717e-03   7.162 8.65e-13 ***
## REVOKED        1.592e-01  1.486e-02  10.712  < 2e-16 ***
## MVR_PTS        3.603e-02  4.883e-03   7.378 1.77e-13 ***
## CAR_AGE       -8.086e-04  1.058e-03  -0.764 0.444658
## URBANICITY     2.823e-01  1.178e-02  23.958  < 2e-16 ***
## JOB_EDU        2.232e-02  1.370e-02   1.629 0.103341
## MVR_PTS_Trans -4.242e-02  1.352e-02  -3.137 0.001713 **
## HOME_INCOME   -6.314e-04  1.770e-04  -3.568 0.000362 ***
## AGE_SEX       -4.151e-02  3.990e-02  -1.040 0.298173
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.1519754)
##
##     Null deviance: 1585  on 8160  degrees of freedom
## Residual deviance: 1236  on 8133  degrees of freedom
## AIC: 7814.2
##
## Number of Fisher Scoring iterations: 2
```

Following conclusions can be drawn from model coefficients

- **KidsDriv: As number of kids driver increses log odds of car crash also increases**
- **Age: Not very significant in predicting car crash**
- **HomeKids: Not very significant in predicting car crash**
- **YOJ: Not very significant in predicting car crash**
- **Income: As Income increases log odds of car crash decreases**
- **Home_Val: Not very important for predicting car crash**
- **MStatus: If you are married, it decreases the log odds of car crash**
- **Education: As education increases log odds of car crash decreases**
- **Jobs: Higher the job level less likely the log odds of car crash**
- **Travtime: Longer the travel time increases the log odds of car crash**
- **Car Use: Commercial cars have more risk compared to private cars**
- **BlueBook: As the cost of cars increases log odds of car crash decreases**
- **TIF: Longer the people are in force less risky it becomes**
- **Red_Car: As expected a car being red doesn't contribute to car crash**
- **Clm_Freq, Revoked and Mvr_Pts: As expected all these variables increases the risk of car crash**
- **Urbanicity : Interestigly Work area is more prone to car crashes than home area**

From the above analysis it is proven that model-3 which uses all the variables along with newly added interaction term and which has class balance data performs better. We will use this model for prediction. Below bar chart shows different models and helps us to compare them on the basis of model metrics



4. Build Models - Multiple Linear Regression

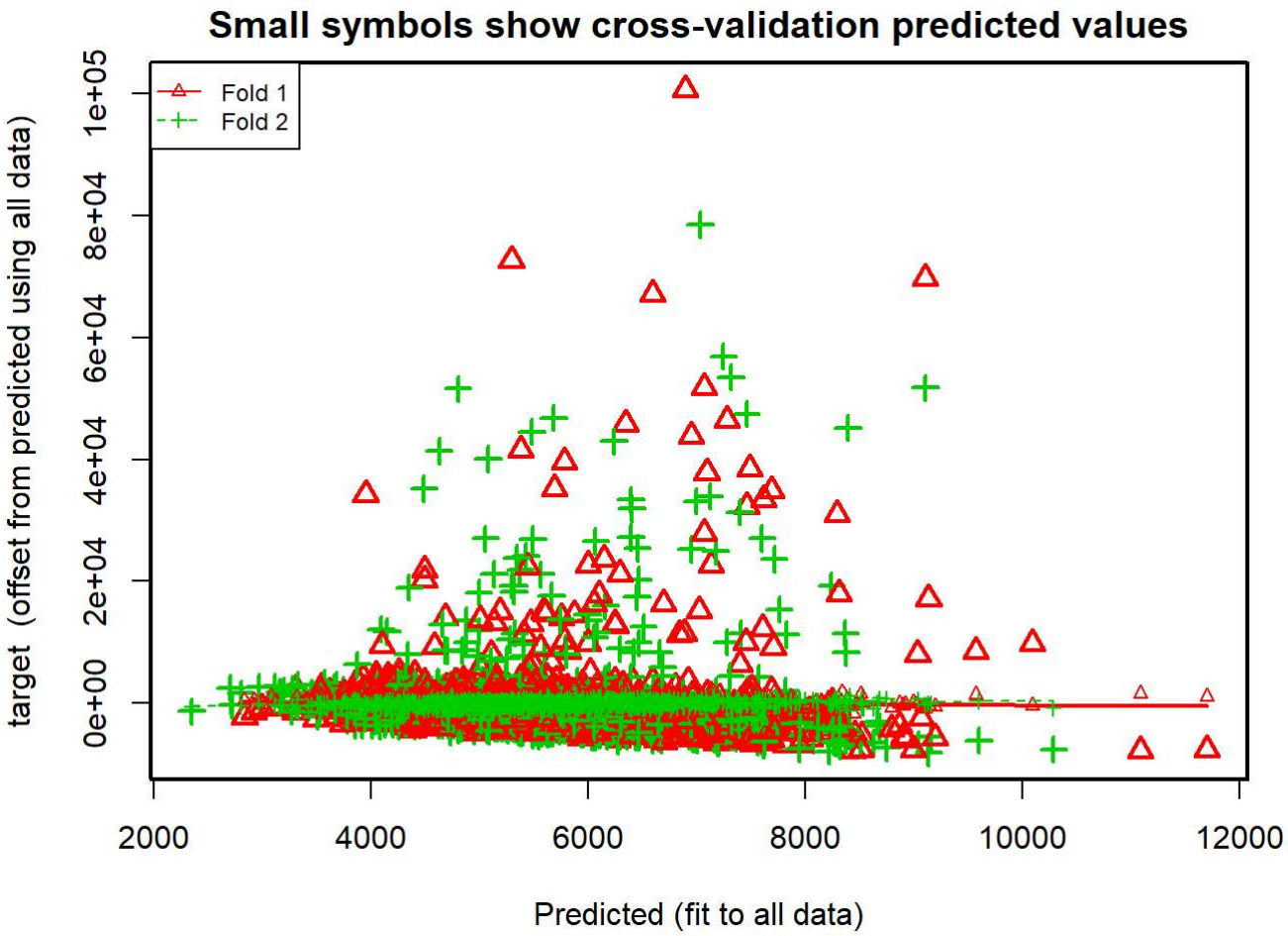
For Linear regression model we will use cross validation technique. We will use Root Mean Square as a model evaluation metric. Model with lower RMSE will be the best model.

RMSE = sqrt(sum(Actual Value - Predicted Value)^2)

1. Model with selected important variables

```
model.metrix = lm.cv("target ~ INCOME + HOME_VAL + MSTATUS  + JOB + CAR_USE + CAR_TYPE + OLDCLAIM + CLM_FREQ + REVOKED + MVR_PTS + URBANICITY + BLUEBOOK + CAR_AGE", train.df.reg)
```

```
## Warning in CVlm(data = input.data, form.lm = formula(form), m = 2, plotit = "Residual", :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```



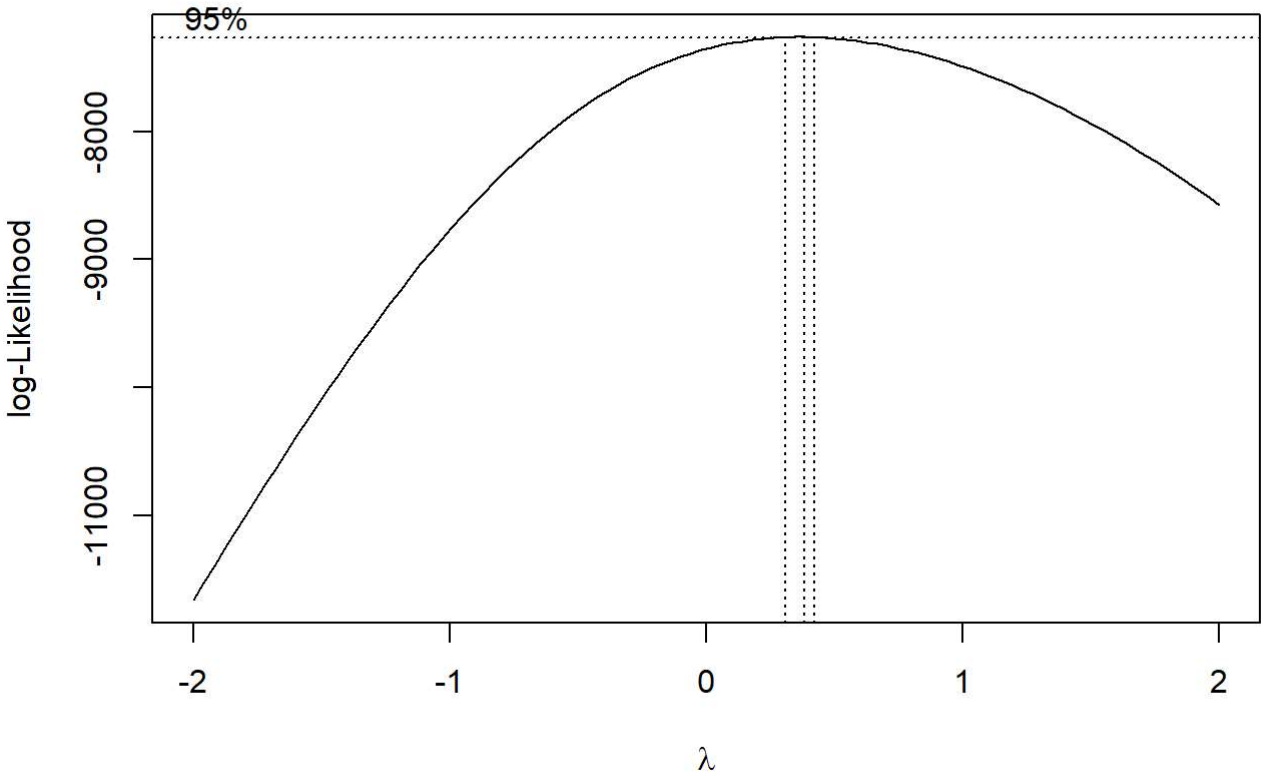

```
print.rmse(model.metrix$output)
```

```
## [1] "Root Mean Square =  355315.14804385"
```

From the above residual plot we can see that residuals are y axis imbalance and heterogenious.

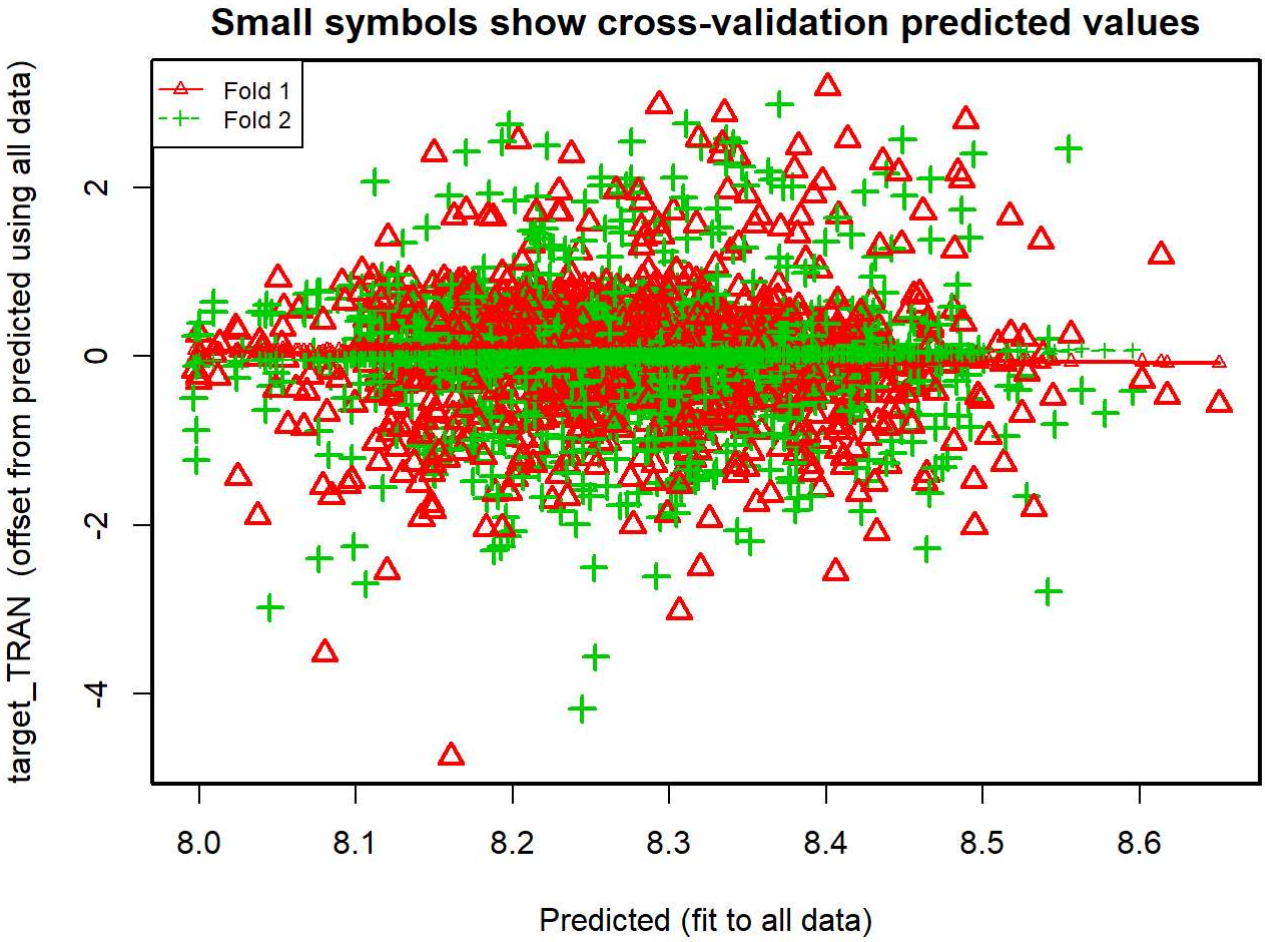
2. Model with BoxCox transformation along with logarithmic transformation of output variable. We are also selecting fewer variables as per important regression coefficients

```
bc <- boxcox(train.df.reg$BLUEBOOK ~ log(train.df.reg$target))
```




```
lambda <- bc$x[which.max(bc$y)]

train.df.reg$target_TRAN = log(train.df.reg$target)
train.df.reg$BLUEBOOK_TRAN = (train.df.reg$BLUEBOOK^lambda -1)/lambda

model.metrix = lm.cv("target_TRAN ~ BLUEBOOK_TRAN + MVR_PTS + CAR_AGE + CAR_TYPE", train.df.reg)
```

```
## Warning in CVlm(data = input.data, form.lm = formula(form), m = 2, plotit = "Residual", :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
model.metrix$output$Predicted = exp(model.metrix$output$Predicted)
print.rmse(model.metrix$output)
```

```
## [1] "Root Mean Square =  366286.69929848"
```

Above residual plot looks better. It is balanced on both the axis and homogenious

We can see that model with box cox transformation along with target variable logarithmic transformation gives us higher RMSE comapared to model with important variable. However diffrence in RMSE is not that big and in general it is alway best practice to select model with lower variables toavoid overfitting. For this reason we will choose second model (Model with box cox transformation with fewer variables) for prediction

3. Regression model coefficient analysis

```
fit = lm(target_TRAN ~ BLUEBOOK_TRAN + MVR_PTS + CAR_AGE + CAR_TYPE + REVOKED + SEX, train.df.re
g)
summary(fit)
```

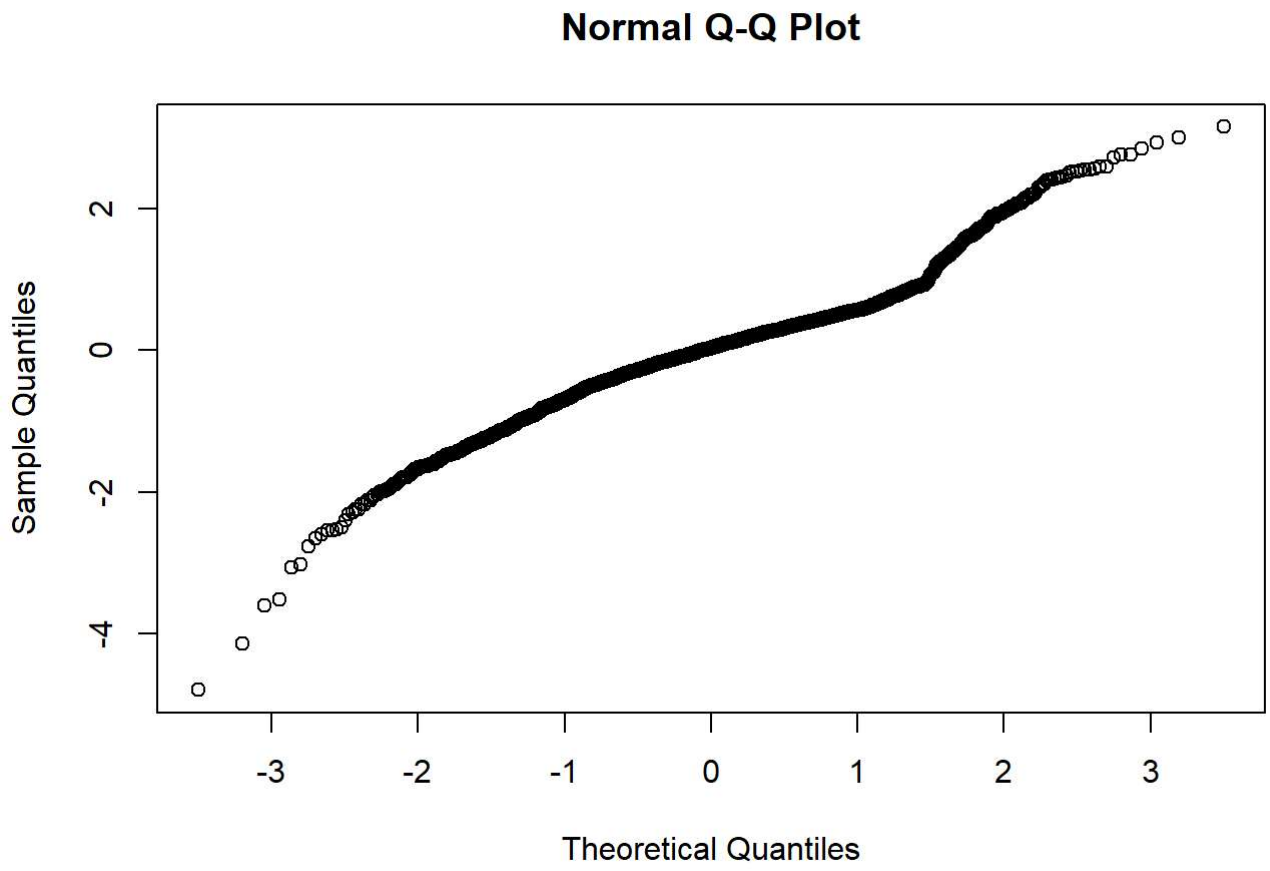
```
##
## Call:
## lm(formula = target_TRAN ~ BLUEBOOK_TRAN + MVR_PTS + CAR_AGE +
##     CAR_TYPE + REVOKED + SEX, data = train.df.reg)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7888 -0.4018  0.0342  0.4030  3.1519
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    7.865e+00  8.550e-02  91.983  < 2e-16 ***
## BLUEBOOK_TRAN  4.232e-03  7.686e-04   5.506  4.1e-08 ***
## MVR_PTS        1.395e-02  6.753e-03   2.066   0.0389 *
## CAR_AGE        3.301e-05  3.216e-03   0.010   0.9918
## CAR_TYPE       2.532e-03  1.054e-02   0.240   0.8102
## REVOKED       -2.926e-02  4.300e-02  -0.681   0.4962
## SEX           -5.424e-02  3.511e-02  -1.545   0.1226
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8062 on 2146 degrees of freedom
## Multiple R-squared:  0.01838,    Adjusted R-squared:  0.01564
## F-statistic: 6.699 on 6 and 2146 DF,  p-value: 4.857e-07
```

Interestingly we can see that only two variables are statistically significant and contributing towards target amount prediction.

- **BLUEBOOK : Vlaue of car is very important factor in determining claim amount. This perfectly makes sense**
- **MVR_PTS: Number of traffic tickets is next important variable. Note that coefficient is positive which indicates pay out amount increases with number of ticket violation. This is counter intuitive. Ideally more the traffic violations lesser should be the payout. However we will use this model since this is giving us lower RMSE indicating better model fit**

3. Regression model redicual analysis

```
qqnorm(fit$residuals)
```



From the residual normality plots we can see that residuals are normally distributed. This satisfies the normality assumption of residuals for multiple regression model.

5. Select Models

1. Read evaluation data and clean. Create required interaction terms

```
##
##  iter imp variable
##   1   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   1   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   2   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   2   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   3   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   3   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   4   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   4   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   5   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   5   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   6   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   6   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   7   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   7   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   8   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   8   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   9   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   9   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   10   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   10   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
```

```
## Warning: Number of logged events: 2
```

Logistic Regression

For Logistic regression we will use Model-3. Model-3 uses balanced dataset using Synthetic Data Genration technique (SMOTE) and also leverages other variable to determine risk of car crashes

```
model.formula = "target ~ KIDSDRIV + AGE + HOMEKIDS + YOJ + INCOME + PARENT1 + HOME_VAL + MSTATU
S+ SEX + EDUCATION + JOB + TRAVTIME + CAR_USE + BLUEBOOK  + TIF + CAR_TYPE + RED_CAR + OLDCLAIM
 + CLM_FREQ + REVOKED + MVR_PTS + CAR_AGE + URBANICITY + JOB_EDU + MVR_PTS_Trans + HOME_INCOME +
AGE_SEX"


model <- glm(formula = model.formula, family = "binomial", data = train.data.balanced)



predicted <- predict(model, newdata = testing,type="response")
lables = ifelse(predicted > 0.5, 1, 0)

testing$TARGET_FLAG_Proba = round(predicted,1)
testing$TARGET_FLAG = lables
testing = data.frame(testing)
#write.table(testing, "./Data/PredictedOutcome.csv", row.names = FALSE, sep=",")
```

Multiple Linear Regression

For multiple linear regression we will use the second model. Second model uses Box Cox transfomration and fewer variable. Even though RMSE is lower we are sticking to this model due to fewer variables and explainability

```
model.formula = "target_TRAN ~ BLUEBOOK_TRAN + MVR_PTS + CAR_AGE + CAR_TYPE"


model <- lm(formula = model.formula,  data = train.df.reg)



predicted <- predict(model, newdata = testing)
testing$TARGET_AMT = testing$TARGET_FLAG * exp(predicted)
testing = data.frame(testing)
write.table(testing, "./Data/PredictedOutcome.csv", row.names = FALSE, sep=",")
```

6. Appendix

Train Test Split Validation Code

```
model.fit.evaluate <- function(model.formula, train.data, test.data) {
  auclist = NULL
  accuracylist = NULL
  recalllist = NULL
  precisionlist = NULL
  k =1
  set.seed(123)

  training <-train.data
  testing.org<-test.data

  testing = testing.org[1:nrow(testing.org), names(testing.org)[names(testing.org) != 'target']]

  model <- glm(formula = model.formula,
               family = "binomial", data = training)
  predicted <- predict(model, newdata = testing,type="response")
  pred <- prediction(predicted, testing.org$target)
  ind = which.max(round(slot(pred, 'cutoffs')[[1]],1) == 0.5)
  perf <- performance(pred, measure = "tpr", x.measure = "fpr")

  auc.tmp <- performance(pred,"auc");
  auc <- as.numeric(auc.tmp@y.values)
  auclist[k] = auc

  acc.perf = performance(pred, measure = "acc")
  acc = slot(acc.perf, "y.values")[[1]][ind]
  accuracylist[k] = acc

  prec.perf = performance(pred, measure = "prec")
  prec = slot(prec.perf, "y.values")[[1]][ind]
  precisionlist[k] = prec

  recall.perf = performance(pred, measure = "tpr")
  recall = slot(recall.perf, "y.values")[[1]][ind]
  recalllist[k] = recall



  return(list("AUC" = mean(auclist), "Accuracy" = mean(accuracylist),  "Recall" = mean(recalllis
t), "Precision" = mean(precisionlist)))
}

df.metrix <<- NULL
print.model.matrix = function(model.name, matrixobj)
{

  print(paste("Printing Metrix for model: ", model.name))
  for(i in 1 : length(matrixobj))
  {
    df = data.frame("Model" = model.name, "Metrix"=names(matrixobj)[[i]], "Value" = matrixobj
[[i]])
    df.metrix <<- rbind(df, df.metrix)
    print(paste(names(matrixobj)[[i]], ":", matrixobj[[i]]))
```

```
  }

}
```

Linear Regression Cross Validation Code

```
lm.cv = function(form,input.data)
{
  out <- CVlm(data = input.data, form.lm = formula(form),m=2,plotit= "Residual", printit = FALSE
)
  cv.rmse <- sqrt(attr(out,"ms"))
  return(list('output' = out))
}
print.rmse = function(output.df)
{
  rss = sum((output.df$target - output.df$Predicted)^2)
  print(paste('Root Mean Square = ', sqrt(rss)))
}
```

Evaluation data cleanup code

```
testing = read.csv("./Data/insurance-evaluation-data.csv", stringsAsFactors = FALSE, na.strings=
c("NA","NaN", " ", ""))

testing$PARENT1 = ifelse(testing$PARENT1 == 'No', 0, 1)
testing$PARENT1 = as.numeric(testing$PARENT1)
testing$MSTATUS = ifelse(testing$MSTATUS == 'z_No', 0, 1)
testing$MSTATUS = as.numeric(testing$MSTATUS)
testing$SEX = ifelse(testing$SEX == 'M', 0, 1)
testing$SEX = as.numeric(testing$SEX)
testing$EDUCATION = as.numeric(factor(testing$EDUCATION, order = TRUE, levels = c("<High School"
, "z_High School", "Bachelors", "Masters", "PhD")))
testing$JOB = as.numeric(factor(testing$JOB, order = TRUE, levels = c("Student", "Home Maker",
"z_Blue Collar", "Clerical", "Professional", 'Manager', 'Lawyer', 'Doctor')))
testing$CAR_USE = ifelse(testing$CAR_USE == "Private", 0, 1)
testing$CAR_USE  = as.numeric(testing$CAR_USE)
testing$CAR_TYPE = as.numeric(factor(testing$CAR_TYPE, order = TRUE, levels = c("Minivan", "z_SU
V", "Van", "Pickup", "Panel Truck", 'Sports Car')))
testing$RED_CAR = ifelse(testing$RED_CAR == "no", 0, 1)
testing$RED_CAR  = as.numeric(testing$RED_CAR)
testing$REVOKED = ifelse(testing$REVOKED == "No", 0, 1)
testing$REVOKED  = as.numeric(testing$REVOKED)
testing$URBANICITY = ifelse(testing$URBANICITY == "z_Highly Rural/ Rural", 0, 1)
testing$URBANICITY  = as.numeric(testing$URBANICITY)

testing = apply(testing[], 2, function(x) ConvertQuatitative(x)) %>%data.frame()
comp.data <- mice(testing,m=2,maxit=10,meth='pmm',seed=500)
```

```
##
##  iter imp variable
##   1   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   1   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   2   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   2   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   3   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   3   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   4   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   4   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   5   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   5   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   6   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   6   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   7   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   7   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   8   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   8   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   9   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   9   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   10   1  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
##   10   2  AGE  YOJ  INCOME  HOME_VAL  JOB  CAR_AGE
```

```
## Warning: Number of logged events: 2
```

```
testing = complete(comp.data)
testing = testing[, !names(testing) %in% c('Index', 'TARGET_FLAG', 'TARGET_AMT')]

testing$JOB_EDU =  round(log(testing$JOB * testing$EDUCATION))
testing$HOME_INCOME = log(1+ testing$HOME_VAL) * log(1 + testing$INCOME)
testing$MVR_PTS_Trans = round(log(1+ testing$MVR_PTS))
testing$AGE_SEX =log(1 + testing$AGE) * (1+testing$SEX)
testing$BLUEBOOK_TRAN = (testing$BLUEBOOK^lambda -1)/lambda
```