Alexander Booth

Predict 411, Section 58

Professor Ott

November 5, 2017

Unit 02: Insurance Logistic Regression Project

Introduction:

In this assignment, I will build the components for a probability/severity model by constructing a logistic regression to estimate the probability that a person will crash their car, and a model to estimate the cost in the event of a crash. I will only use the variables given to me, or variables that I derive from the data provided. This data set contains approximately 8000 records. Each record represents a customer at an auto insurance company and has two target variables. The first target variable, TARGET_FLAG, is a 1 or a 0. A “1” means that the person was in a car crash. A zero means that the person was not in a car crash. The second target variable is TARGET_AMT. This value is zero if the person did not crash their car. But if they did crash their car, this number will be a value greater than zero.

I plan to use logistic regression to predict the probability that a person will get into a crash. I will use Ordinary Least Squares (OLS) to predict how much a person would have to pay if they got into a crash. I will construct multiple models and select a single model based on both model comparison criteria, such as AIC, BIC, and Adjusted R^2, as well as interpretability. The data manipulation steps, as well as the parameters from the model, will comprise a strategy for predicting wins on a test set.

I am provided with two data sets, one for training the model, the other to feed into the predicting strategy. As I move further into examining the data, I will keep an eye out for outliers since there might be cases when people pay an unusually high sum of money after getting into a crash. The hope is that through exploratory data analysis I can become intimate with the data, and use some of the techniques learned to create a set of variables that will perform well when moving into model selection.

Exploratory Data Analysis:

My plan to learn and explore the data set will consist of the following steps:

- Examine the data, as well as the data dictionary. Fix any errors (e.g. label-switching)
- Consider the arithmetic relationship between variables (e.g. combining or decomposing), as well as contextual relationships between variables (e.g. variables that represent conceptually opposite measurements)
- Understand what data is missing
- Understand the initial relationship variables have with our dependent variable
- Impute variables that have missing data
- Examine variable distributions and consider further imputation or indicator coding
- Re-examine relationship between re-expressed/imputed independent variables and dependent variable
- Begin model construction

There are two data sets provided, one is the training data set comprised of 8161 observations. The other is a testing data set comprised of 2141 observations. I will begin my exploratory data analysis (EDA) by examining the variables provided to us in the data dictionary.

Table 1: Data Dictionary with Data Definition

VARIABLE NAME	DEFINITION
INDEX	Identification Variable (do not use)
TARGET_FLAG	Was Car in a crash? 1=YES 0=NO
TARGET_AMT	If car was in a crash, what was the cost
AGE	Age of Driver
BLUEBOOK	Value of Vehicle
CAR_AGE	Vehicle Age
CAR_TYPE	Type of Car
CAR_USE	Vehicle Use
CLM_FREQ	#Claims(Past 5 Years)
EDUCATION	Max Education Level
HOMEKIDS	#Children @Home
HOME_VAL	Home Value
INCOME	Income
JOB	Job Category
KIDSDRIV	#Driving Children
MSTATUS	Marital Status
MVR_PTS	Motor Vehicle Record Points
OLDCLAIM	Total Claims(Past 5 Years)
PARENT1	Single Parent
RED_CAR	A Red Car
REVOKED	License Revoked (Past 7 Years)
SEX	Gender
TIF	Time in Force
TRAVTIME	Distance to Work
URBANICITY	Home/Work Area
YOJ	Years on Job

Table 2: Data Dictionary with Proposed Theoretical Effect

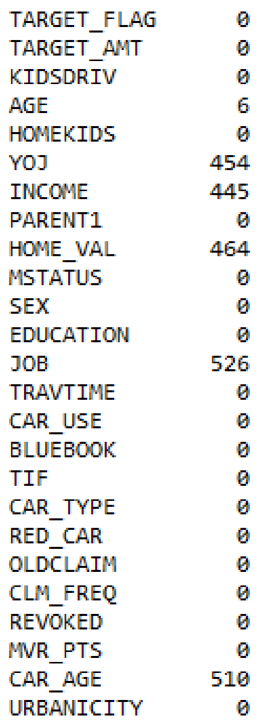
VARIABLE NAME	THEORETICAL EFFECT
INDEX	None
TARGET_FLAG	None
TARGET_AMT	None
AGE	Very young people tend to be risky. Maybe very old people also.
BLUEBOOK	Unknown effect on probability of collision, but probably effect the payout if there is a crash
CAR_AGE	Unknown effect on probability of collision, but probably effect the payout if there is a crash
CAR_TYPE	Unknown effect on probability of collision, but probably effect the payout if there is a crash
CAR_USE	Commercial vehicles are driven more, so might increase probability of collision
CLM_FREQ	The more claims you filed in the past, the more you are likely to file in the future
EDUCATION	Unknown effect, but in theory more educated people tend to drive more safely
HOMEKIDS	Unknown effect
HOME_VAL	In theory, home owners tend to drive more responsibly
INCOME	In theory, rich people tend to get into fewer crashes
JOB	In theory, white collar jobs tend to be safer
KIDSDRIV	When teenagers drive your car, you are more likely to get into crashes
MSTATUS	In theory, married people drive more safely
MVR_PTS	If you get lots of traffic tickets, you tend to get into more crashes
OLDCLAIM	If your total payout over the past five years was high, this suggests future payouts will be high
PARENT1	Unknown effect
RED_CAR	Urban legend says that red cars (especially red sports cars) are more risky. Is that true?
REVOKED	If your license was revoked in the past 7 years, you probably are a more risky driver.
SEX	Urban legend says that women have less crashes then men. Is that true?
TIF	People who have been customers for a long time are usually more safe.
TRAVTIME	Long drives to work usually suggest greater risk
URBANICITY	Unknown
YOJ	People who stay at a job for a long time are usually more safe

These theoretical effects are good to have loaded into memory as I begin initial examination of the data set. I want to pay attention to variables equally at the onset. However, if I am observing something such as a low correlation between our dependent variable and a variable that I think theoretically should be indicative, I may consider what manipulations of the variable are available to examine further.

First, it is apparent that this data has been prepared for analysis since all of the variable labels are consistent. Additionally, the variables are a mixture of continuous and categorical. The categorical data ranges from labels, such as EDUCATION, to numeric categories, such as how many kids one has at home (HOMEKIDS). It is imperative to remain diligent to potential relationships between variables that can be explored, where one variable can help to provide context for examining another variable. One example of these presumed relationships is between HOME_VAL and INCOME. I would expect that people who have a large income also have a home with a large value. Another example is between HOMEKIDS and KIDSDRIV. I would expect families with more children to have more teens driving.

First I will examine the training data for missing values.

Table 3: Missing Values



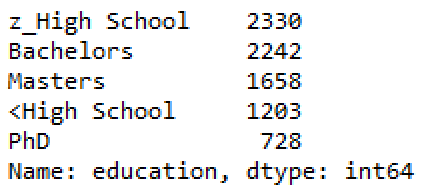
```
TARGET_FLAG      0
TARGET_AMT       0
KIDSDRIV         0
AGE              6
HOMEKIDS         0
YOJ            454
INCOME         445
PARENT1          0
HOME_VAL       464
MSTATUS          0
SEX              0
EDUCATION        0
JOB            526
TRAVTIME         0
CAR_USE          0
BLUEBOOK         0
TIF              0
CAR_TYPE         0
RED_CAR          0
OLDCLAIM         0
CLM_FREQ         0
REVOKED          0
MVR_PTS          0
CAR_AGE        510
URBANICITY       0
```

Six of the variables have missing data. We will create two new variables in this process; one with the IMP_* prefix for the imputed variable, leaving the original variable untouched, and one with the m_* prefix as an indicator variable for the imputed variable. Sometimes, the fact that a variable was missing can actually be predictive. This means the indicator variables might be entered into the predictive model. One of the variables with missing data is categorical, with labels. Instead of imputing the data missing in Jobs with an existing label, I instead will create a new category called “No Job Information Available”. For the other five numeric variables, I will impute each with their median. The median, unlike the mean, is less susceptible to extreme values.

Next, after imputing these values, I will examine the distribution of each continuous variable via a histogram and boxplot for extreme values, outliers, or wrongly inserted data. For the categorical variables, I will obtain counts and a cross-tabulation table to determine any skew. The visual examination of the variables as histograms and boxplots show a few variables differing mildly from expected normal shapes. I will attach all of these graphics in the Appendix.

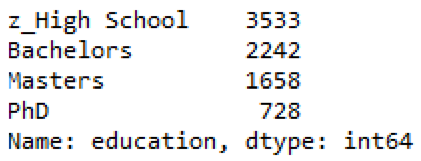
For the categorical variables, most of the results were not surprising. For instance, most people lived in urban centers, had not had their license revoked or filed any claims. Most had few, if any, children at home and few teens driving. Consequently, few people had actually crashed their vehicle. One variable stood out in error. EDUCATION consisted of two labels for High School, “z_High School” and “<High School”. I combined them both into “z_High School”. Furthermore, I created two new variables from JOB and EDUCATION. I created a variable called WHITE_COLLAR. If a person’s job was as a Doctor, Manager, or Lawyer, their value would be 1. Anything else was 0. I also created a variable called UNIVERSITY. If a person’s education level was Bachelor’s, Master’s, or PhD, then this value would be 1. Anything else was 0. These variables should amplify the effects of higher education and white collar jobs.

Table 4: Education



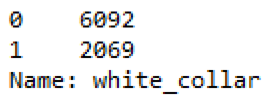
```
z_High School    2330
Bachelors        2242
Masters          1658
<High School     1203
PhD               728
Name: education, dtype: int64
```

Table 4.1: Education (Fixed)



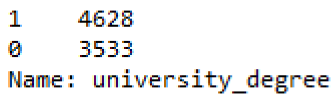
```
z_High School    3533
Bachelors        2242
Masters          1658
PhD               728
Name: education, dtype: int64
```

Table 5: White Collar Jobs



```
0    6092
1    2069
Name: white_collar
```

Table 6: University Degree



```
1    4628
0    3533
Name: university_degree
```

Upon analysis of the continuous variables, I noticed that some of these variables have an alarming shape to their distribution, with indication that there are some extreme values. I am looking for values that appear to be so extreme that I should simply imputate them. With this line of reasoning I found several unreasonable values skewing the data. The following variables were highly skewed to the right with unreasonably long tails: TARGET_AMT, INCOME, HOME_VAL, TRAVTIME, and BLUEBOOK. As I suspected TARGET_AMT had a value of over $100,000 which was the largest outlier. TRAVTIME had a reported commute time of over two hours and twenty minutes, which could be a mistake, but is otherwise skewing the data. The distributions for HOME_VAL and INCOME do look strikingly similar. Likewise, they both have extreme values in the right tails. There are a few income’s reported to be over $350,000 and a few home values to be over $700,000. Finally, BLUEBOOK had a few valuations of the vehicle as being well over $60,000, skewing the data right. I suspect that BLUEBOOK may also have a correlation with INCOME and HOME_VAL, as people with a high income might be more likely to buy expensive cars.

Finally, I analyzed the OLDCLAIM variable. This variable had a quite interesting distribution. Not only had most people not filed any old claims, but the distribution was bimodal. A separate population seemed to have filed old claims greater than $12,000. I created three dummy variables from this, the first being if one had filed an old claim or not, the second being if one had filed an old claim between $1 and $12,000, and the third being if one had filed old claims greater than $12,000. These value counts can also be found in the appendix.

Figure 1: Histogram of TARGET_AMT

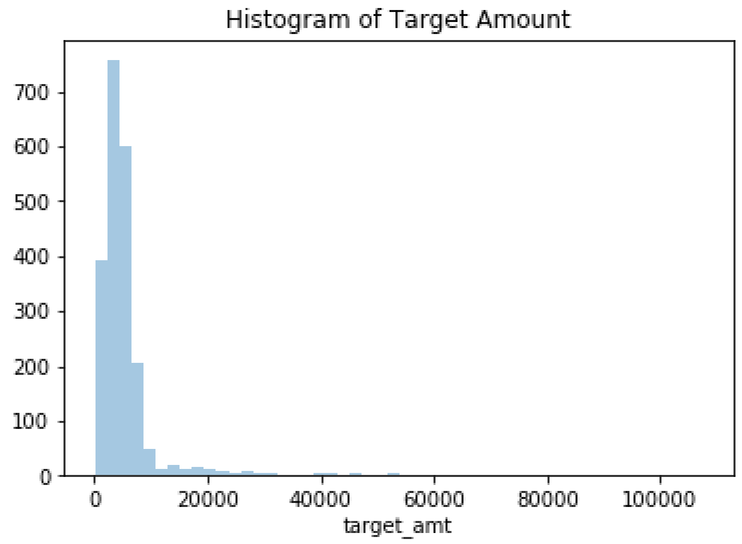


Figure 2: Histogram of TRAVTIME

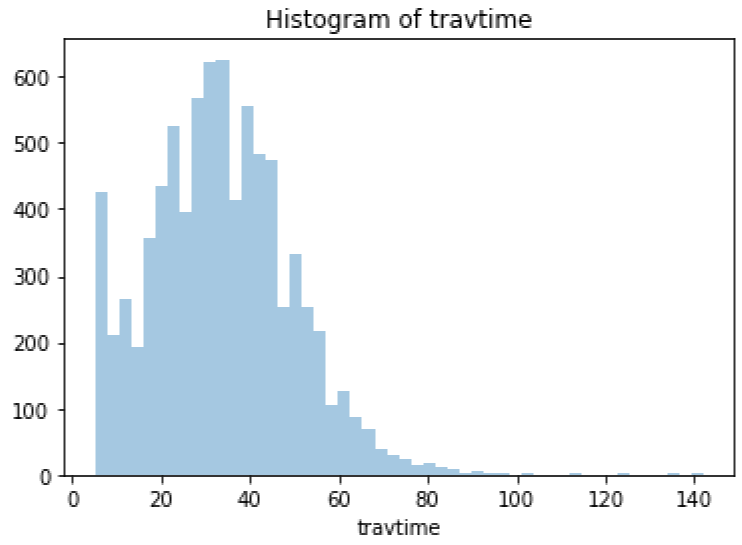


Figure 3: Histogram of INCOME

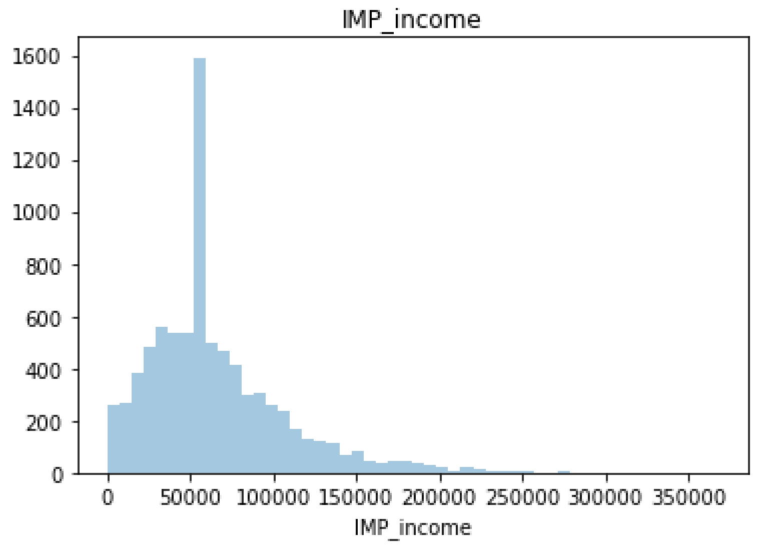


Figure 4: Histogram of HOME_VAL

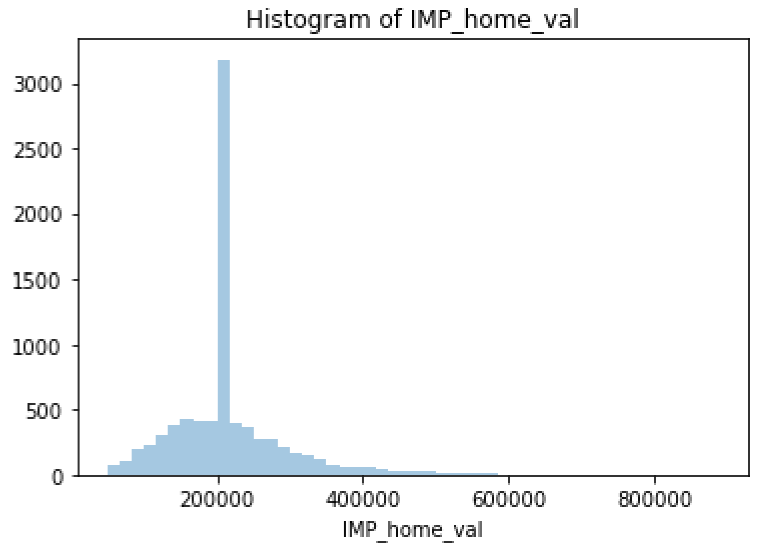


Figure 5: Histogram of BLUEBOOK

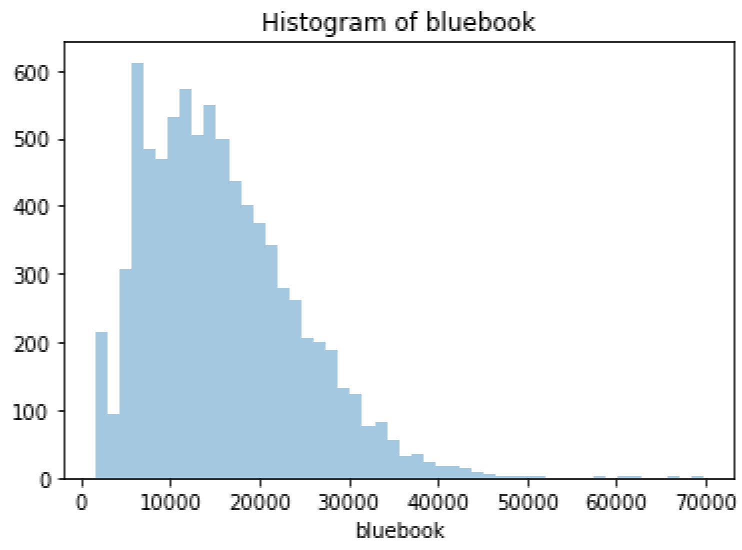
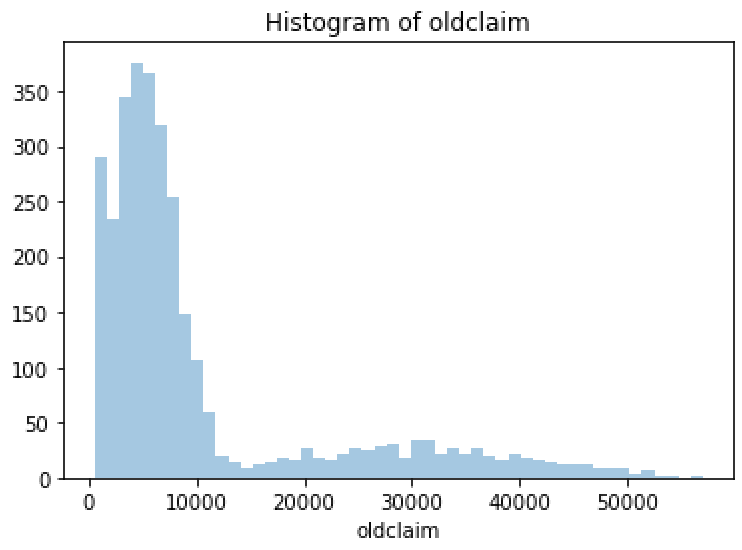


Figure 6: Histogram of OLDCLAIM greater than 0



From our studies, I know that we have several options to fix this data:

- Truncating at a specific value (or set of values, e.g. ranges)
- log transformations
- Standardization (e.g. Z-Score)
- Binning (e.g. Buckets, Quantiles)

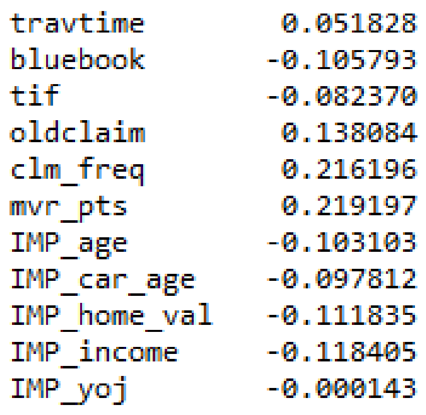
- Combining above techniques (e.g. log followed by binning)

Since I am committed to using the logistic and OLS regression techniques as our models, I cannot simply choose to alter the modeling technique in the face of the extreme values. Even though this is an assignment, there are many industries that have regulatory limitations that would prevent one from pursuing a different model out of convenience. I chose to use the truncating strategy at this time based off of quantiles. For the variables listed above, if any values exceeded the 99th percentile, then they were replaced with the value of the 99th percentile. Likewise for values less than the 1th percentile. Finally, with missing values imputed and outliers mostly fixed, it was important to remember to do these same actions with the test data. As such, I imputed missing data with the medians from the training data and truncating the variables using the original 99th and 1th percentiles of the same variables from the training set.

Model Creation (Logistic)

Before moving onto to creating models, I first looked at the correlations of all of our variables with a heatmap and a pair plot. Initially, I examined the simple correlation between the continuous variables and TARGET_FLAG:

Table 7: Correlation between continuous variables and TARGET_FLAG



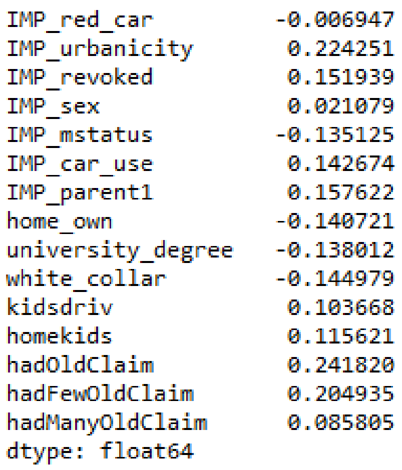
```
travtime          0.051828
bluebook         -0.105793
tif              -0.082370
oldclaim          0.138084
clm_freq          0.216196
mvr_pts           0.219197
IMP_age          -0.103103
IMP_car_age      -0.097812
IMP_home_val     -0.111835
IMP_income       -0.118405
IMP_yoj          -0.000143
```

:

From this I can see that there are some variables that I would automatically take forward into model construction, notably anything whose absolute value exceeds that of a 0.01 threshold. It is interesting to see that travel time has such a low correlation. I would expect that the concept of exposure to operation of their vehicle would significantly increase the likelihood of a crash.

Further, I created dummy variables for all of the binary categorical variables. I also created a HOME_OWN variable that was 0 if the HOME_VAL was 0 or missing, and 1 otherwise. I then looked at the correlations between the binary categorical variables and the TARGET_FLAG:

Table 8: Correlation between categorical variables and TARGET_FLAG



```
IMP_red_car          -0.006947
IMP_urbanicity        0.224251
IMP_revoked           0.151939
IMP_sex               0.021079
IMP_mstatus          -0.135125
IMP_car_use           0.142674
IMP_parent1           0.157622
home_own             -0.140721
university_degree    -0.138012
white_collar         -0.144979
kidsdriv              0.103668
homekids              0.115621
hadOldClaim           0.241820
hadFewOldClaim        0.204935
hadManyOldClaim       0.085805
dtype: float64
```

Interestingly, it looks like most of these correlations adhere to the theoretical effects. For instance, having a white collar job, a university degree, owning a home, and being married all reduce your likelihood of getting into a crash. Likewise, being a single parent, living in the city, driving a commercial vehicle, and having your license revoked all increase your likelihood of getting into a crash. I can dismiss gender and owning a red car as having any influence on this model. Finally, to examine all the variables identified as potentially influencing this model, I created a heatmap and pairplot to depict their relationships.

Figure 7: Correlation Heatmap

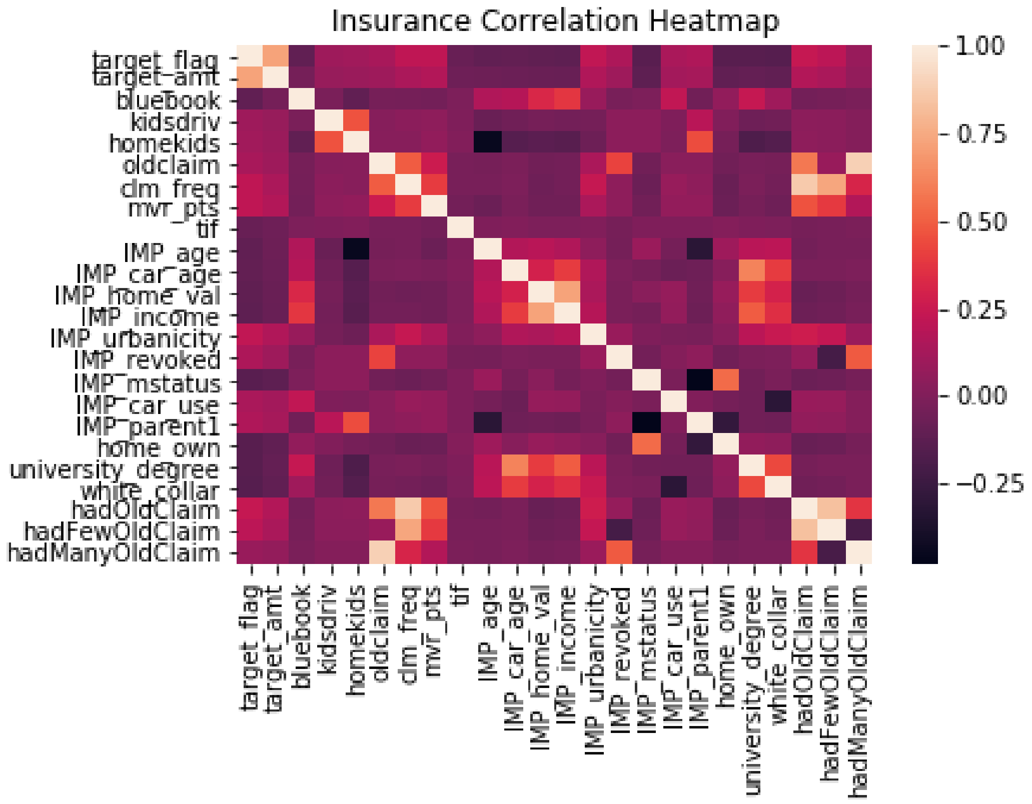
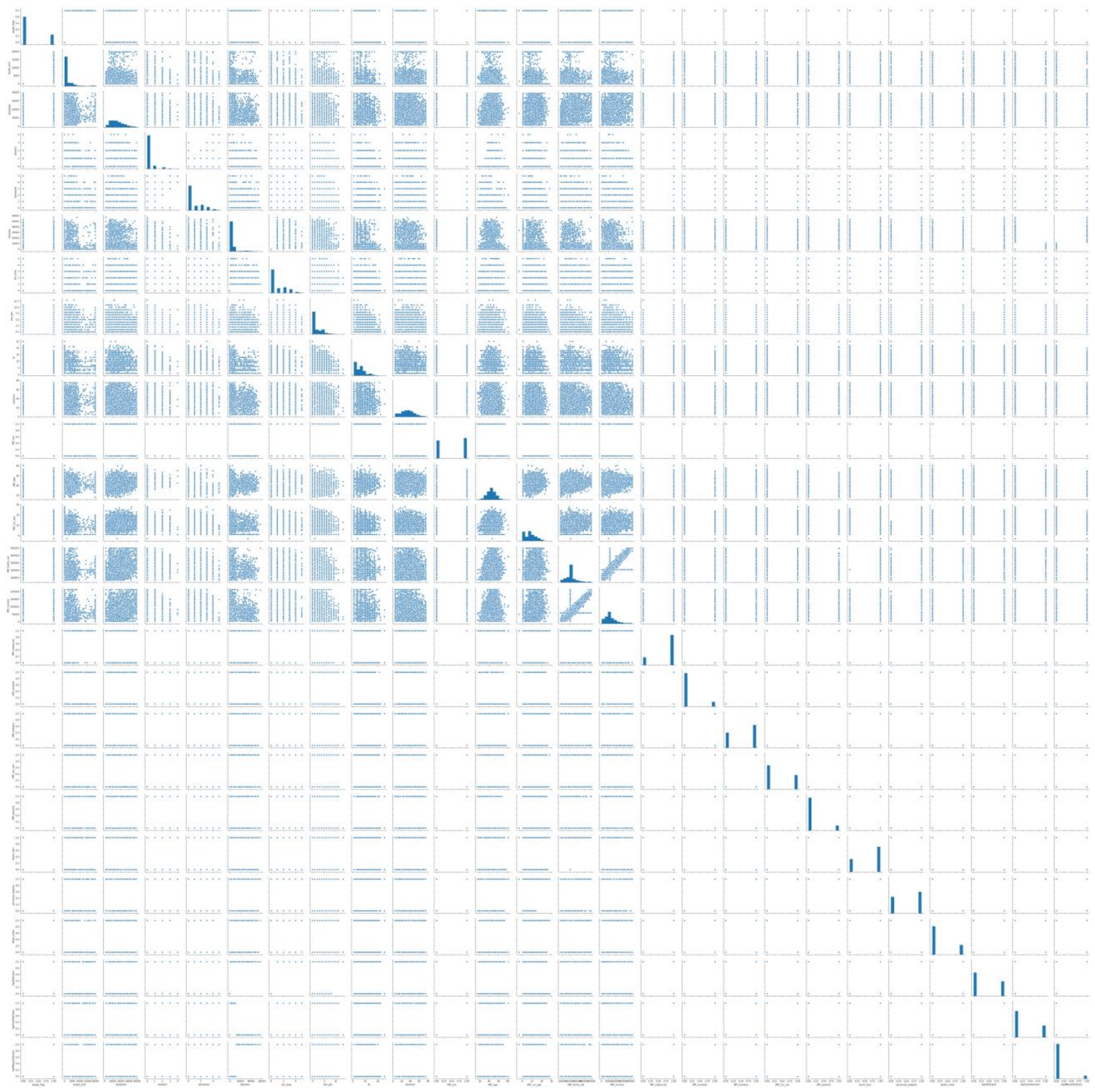


Figure 8: Correlation Pairplot

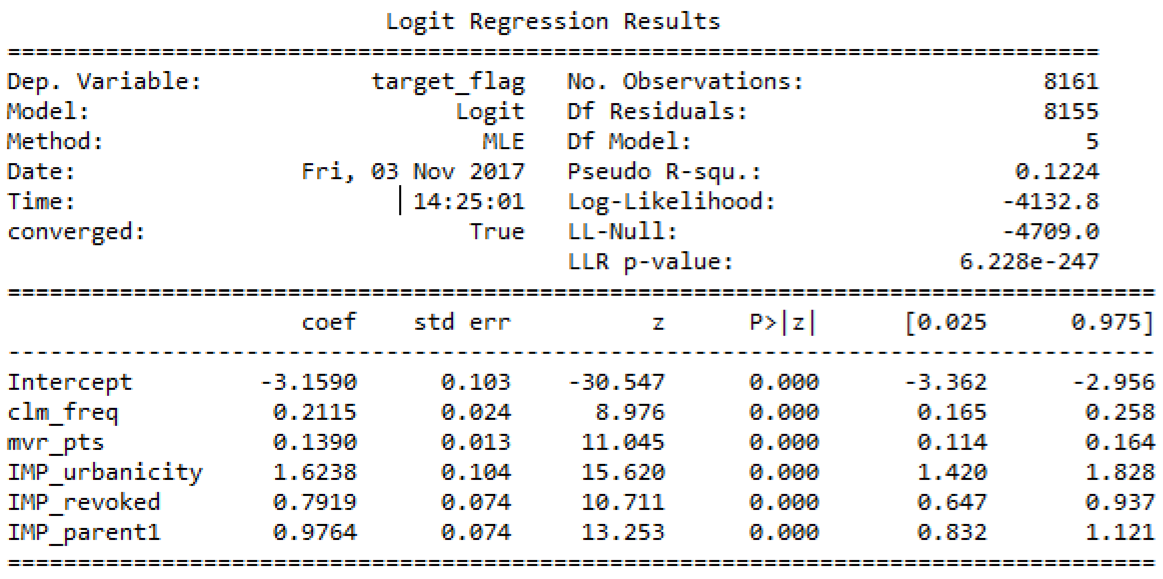


There are some interesting correlations here. First, income, home value, and bluebook all seem to have a positive relationship with each other, as postulated earlier. Further, having a university degree correlates with white collar jobs, higher income and home value, and older cars. Being married correlates positively with owning your own home. Having your license revoked correlates with how many old claims have been filed. How many kids one has at home correlates strongly with how many teens one has driving. Based off of this analysis, I need to be careful of the effects that multi-correlation can have on this model.

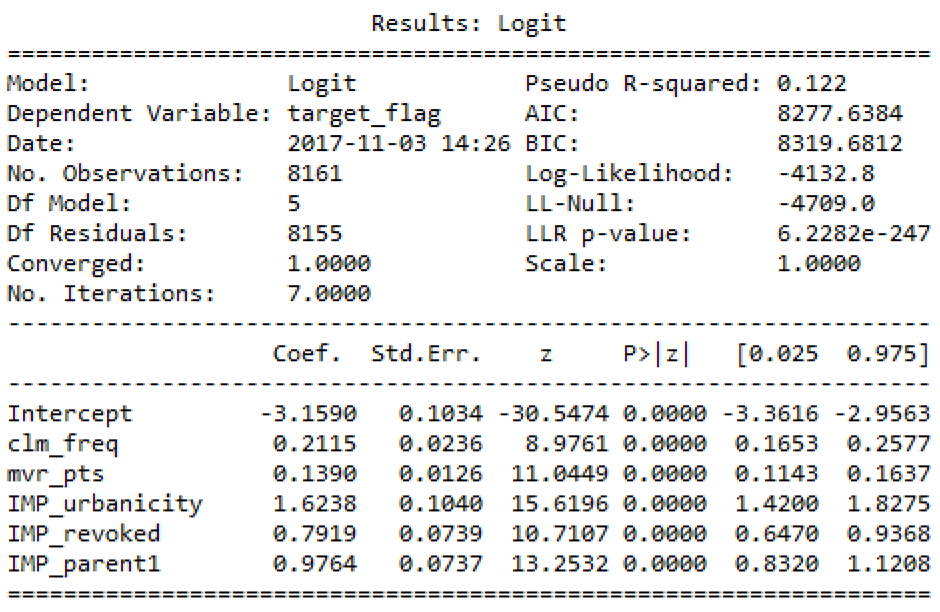
For the next step of model creation, I took almost every variable. I had a very liberal correlation cutoff, as I did not want to remove a potential variable that had a low correlation but would be significant in the model. I can always remove variables once I see that they are not significant. I decided to judge my potential models on AIC, BIC, ROC, Pseudo-R^squared, and complexity.

My first model consisted of solely the top five variables correlating the most with TARGET_FLAG. These variables were claim frequency, MVR points, urban or city, revoked, and single parent. I did not include the old claim buckets. Although they had a relatively high correlation, they also multi-correlate with claim frequency. Additionally, the five variables selected here were included in the data set and needed very little imputation. The results are below:

Table 9: Model 1 Summary

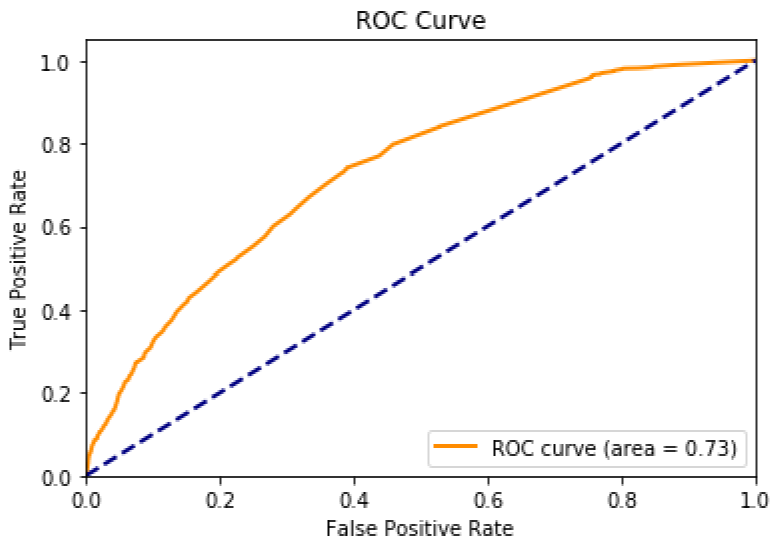


```
                           Logit Regression Results
==============================================================================
Dep. Variable:            target_flag   No. Observations:                 8161
Model:                          Logit   Df Residuals:                     8155
Method:                           MLE   Df Model:                            5
Date:                Fri, 03 Nov 2017   Pseudo R-squ.:                  0.1224
Time:                        14:25:01   Log-Likelihood:                -4132.8
converged:                       True   LL-Null:                       -4709.0
                                        LLR p-value:                 6.228e-247
==================================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------
Intercept         -3.1590      0.103    -30.547      0.000      -3.362      -2.956
clm_freq           0.2115      0.024      8.976      0.000       0.165       0.258
mvr_pts            0.1390      0.013     11.045      0.000       0.114       0.164
IMP_urbanicity     1.6238      0.104     15.620      0.000       1.420       1.828
IMP_revoked        0.7919      0.074     10.711      0.000       0.647       0.937
IMP_parent1        0.9764      0.074     13.253      0.000       0.832       1.121
==================================================================================
```



```
                         Results: Logit
================================================================
Model:              Logit            Pseudo R-squared: 0.122
Dependent Variable: target_flag      AIC:              8277.6384
Date:               2017-11-03 14:26 BIC:              8319.6812
No. Observations:   8161             Log-Likelihood:   -4132.8
Df Model:           5                LL-Null:          -4709.0
Df Residuals:       8155             LLR p-value:      6.2282e-247
Converged:          1.0000           Scale:            1.0000
No. Iterations:     7.0000
----------------------------------------------------------------
                  Coef.  Std.Err.    z     P>|z|   [0.025  0.975]
----------------------------------------------------------------
Intercept        -3.1590   0.1034 -30.5474 0.0000 -3.3616 -2.9563
clm_freq          0.2115   0.0236   8.9761 0.0000  0.1653  0.2577
mvr_pts           0.1390   0.0126  11.0449 0.0000  0.1143  0.1637
IMP_urbanicity    1.6238   0.1040  15.6196 0.0000  1.4200  1.8275
IMP_revoked       0.7919   0.0739  10.7107 0.0000  0.6470  0.9368
IMP_parent1       0.9764   0.0737  13.2532 0.0000  0.8320  1.1208
================================================================
```

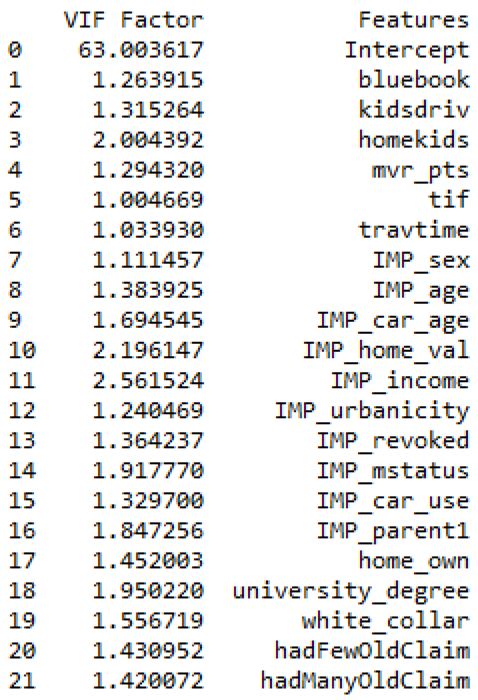
Figure 9: Model 1 ROC Curve



This model is actually not too bad for a first pass! All of the included variables are statistically significant. The coefficient values match the expected theoretical effects, as all five should increase one's probability of getting into a crash. The pseudo-R squared is fairly low and the AIC and BIC values are fairly high. Additionally, while a ROC curve area of .73 is not awful, I think that I can do better.

For the second model I created, I decided to eliminate multicollinearity as defined by the Variance Inflation Factor or VIF. After creating a feature auto-selection function, I set the VIF limit to be three. The following features were then used in the second model:

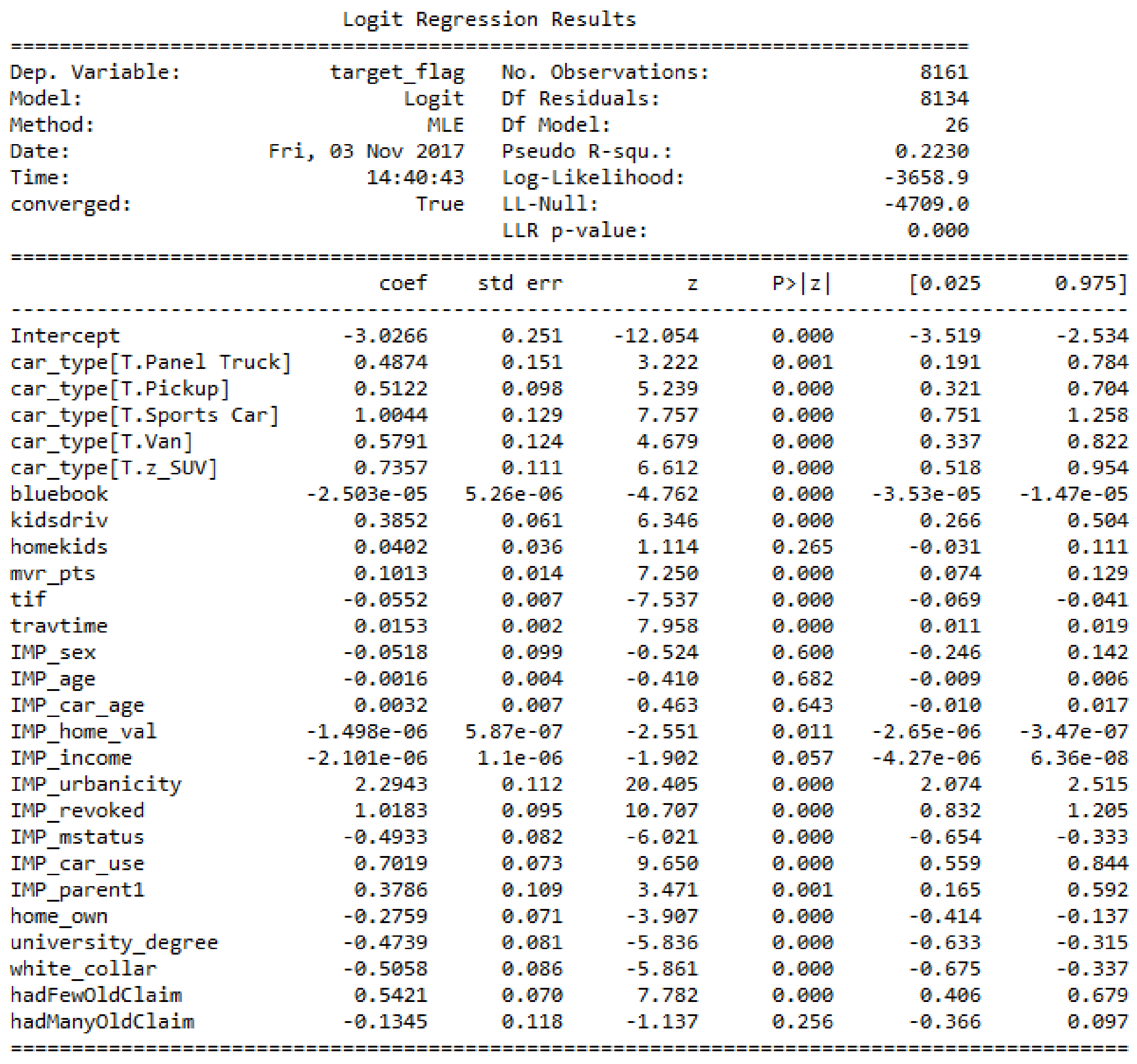
Table 10: Features with VIF less than 3



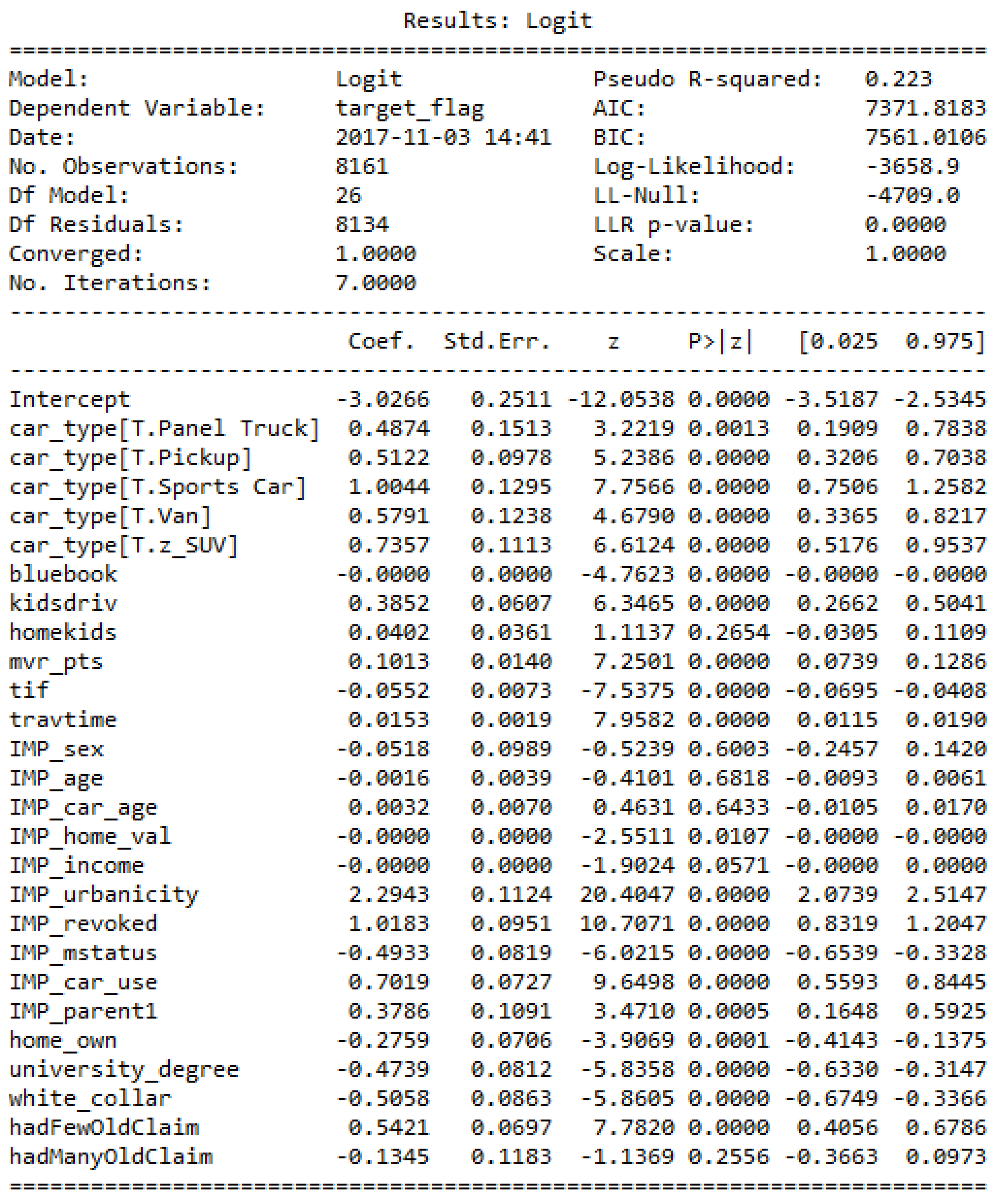
```
    VIF Factor          Features
0    63.003617         Intercept
1     1.263915          bluebook
2     1.315264          kidsdriv
3     2.004392          homekids
4     1.294320           mvr_pts
5     1.004669               tif
6     1.033930          travtime
7     1.111457           IMP_sex
8     1.383925           IMP_age
9     1.694545       IMP_car_age
10    2.196147      IMP_home_val
11    2.561524        IMP_income
12    1.240469    IMP_urbanicity
13    1.364237       IMP_revoked
14    1.917770       IMP_mstatus
15    1.329700       IMP_car_use
16    1.847256       IMP_parent1
17    1.452003          home_own
18    1.950220 university_degree
19    1.556719      white_collar
20    1.430952    hadFewOldClaim
21    1.420072   hadManyOldClaim
```

It turns out that multi-collinearity is not as big as a problem as I initially thought in this dataset. The algorithm dropped old claims, claim frequency, and the “had an old claim” flag variable. The binary variables of hadFewOldClaim and hadManyOldClaim are better predictors. The output from a model created from these remaining features, plus car_type, are described below.

Table 11: Model 2 Summary

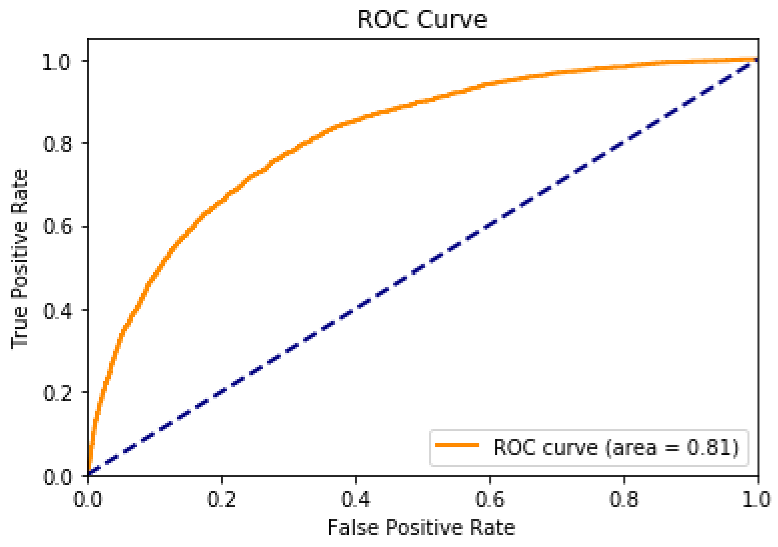


```
                           Logit Regression Results
==============================================================================
Dep. Variable:            target_flag   No. Observations:                 8161
Model:                          Logit   Df Residuals:                     8134
Method:                           MLE   Df Model:                           26
Date:                Fri, 03 Nov 2017   Pseudo R-squ.:                  0.2230
Time:                        14:40:43   Log-Likelihood:                -3658.9
converged:                       True   LL-Null:                       -4709.0
                                        LLR p-value:                     0.000
===========================================================================================
                              coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------------
Intercept                  -3.0266      0.251    -12.054      0.000      -3.519      -2.534
car_type[T.Panel Truck]     0.4874      0.151      3.222      0.001       0.191       0.784
car_type[T.Pickup]          0.5122      0.098      5.239      0.000       0.321       0.704
car_type[T.Sports Car]      1.0044      0.129      7.757      0.000       0.751       1.258
car_type[T.Van]             0.5791      0.124      4.679      0.000       0.337       0.822
car_type[T.z_SUV]           0.7357      0.111      6.612      0.000       0.518       0.954
bluebook                -2.503e-05   5.26e-06     -4.762      0.000   -3.53e-05   -1.47e-05
kidsdriv                    0.3852      0.061      6.346      0.000       0.266       0.504
homekids                    0.0402      0.036      1.114      0.265      -0.031       0.111
mvr_pts                     0.1013      0.014      7.250      0.000       0.074       0.129
tif                        -0.0552      0.007     -7.537      0.000      -0.069      -0.041
travtime                    0.0153      0.002      7.958      0.000       0.011       0.019
IMP_sex                    -0.0518      0.099     -0.524      0.600      -0.246       0.142
IMP_age                    -0.0016      0.004     -0.410      0.682      -0.009       0.006
IMP_car_age                 0.0032      0.007      0.463      0.643      -0.010       0.017
IMP_home_val            -1.498e-06   5.87e-07     -2.551      0.011   -2.65e-06   -3.47e-07
IMP_income              -2.101e-06    1.1e-06     -1.902      0.057   -4.27e-06    6.36e-08
IMP_urbanicity              2.2943      0.112     20.405      0.000       2.074       2.515
IMP_revoked                 1.0183      0.095     10.707      0.000       0.832       1.205
IMP_mstatus                -0.4933      0.082     -6.021      0.000      -0.654      -0.333
IMP_car_use                 0.7019      0.073      9.650      0.000       0.559       0.844
IMP_parent1                 0.3786      0.109      3.471      0.001       0.165       0.592
home_own                   -0.2759      0.071     -3.907      0.000      -0.414      -0.137
university_degree          -0.4739      0.081     -5.836      0.000      -0.633      -0.315
white_collar               -0.5058      0.086     -5.861      0.000      -0.675      -0.337
hadFewOldClaim              0.5421      0.070      7.782      0.000       0.406       0.679
hadManyOldClaim            -0.1345      0.118     -1.137      0.256      -0.366       0.097
===========================================================================================
```



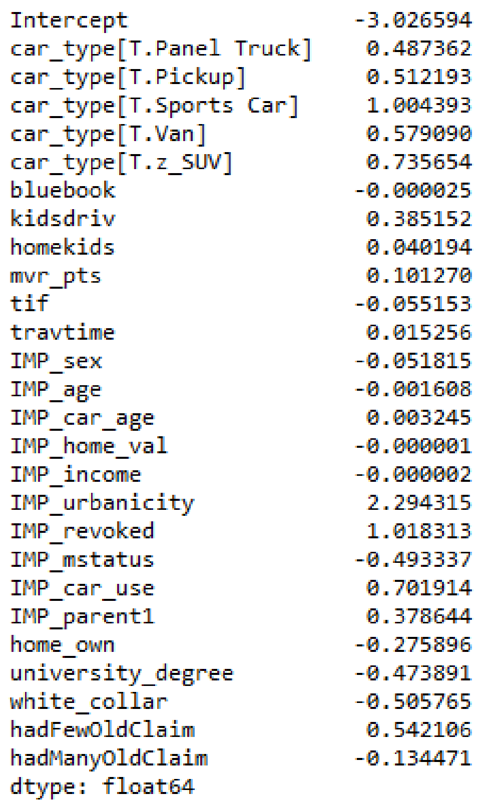
```
                         Results: Logit
=================================================================
Model:              Logit            Pseudo R-squared: 0.223
Dependent Variable: target_flag      AIC:              7371.8183
Date:               2017-11-03 14:41 BIC:              7561.0106
No. Observations:   8161             Log-Likelihood:   -3658.9
Df Model:           26               LL-Null:          -4709.0
Df Residuals:       8134             LLR p-value:      0.0000
Converged:          1.0000           Scale:            1.0000
No. Iterations:     7.0000
-----------------------------------------------------------------
                     Coef.  Std.Err.    z     P>|z|   [0.025  0.975]
-----------------------------------------------------------------
Intercept           -3.0266   0.2511 -12.0538 0.0000 -3.5187 -2.5345
car_type[T.Panel Truck] 0.4874   0.1513   3.2219 0.0013  0.1909  0.7838
car_type[T.Pickup]   0.5122   0.0978   5.2386 0.0000  0.3206  0.7038
car_type[T.Sports Car] 1.0044   0.1295   7.7566 0.0000  0.7506  1.2582
car_type[T.Van]      0.5791   0.1238   4.6790 0.0000  0.3365  0.8217
car_type[T.z_SUV]    0.7357   0.1113   6.6124 0.0000  0.5176  0.9537
bluebook            -0.0000   0.0000  -4.7623 0.0000 -0.0000 -0.0000
kidsdriv             0.3852   0.0607   6.3465 0.0000  0.2662  0.5041
homekids             0.0402   0.0361   1.1137 0.2654 -0.0305  0.1109
mvr_pts              0.1013   0.0140   7.2501 0.0000  0.0739  0.1286
tif                 -0.0552   0.0073  -7.5375 0.0000 -0.0695 -0.0408
travtime             0.0153   0.0019   7.9582 0.0000  0.0115  0.0190
IMP_sex             -0.0518   0.0989  -0.5239 0.6003 -0.2457  0.1420
IMP_age             -0.0016   0.0039  -0.4101 0.6818 -0.0093  0.0061
IMP_car_age          0.0032   0.0070   0.4631 0.6433 -0.0105  0.0170
IMP_home_val        -0.0000   0.0000  -2.5511 0.0107 -0.0000 -0.0000
IMP_income          -0.0000   0.0000  -1.9024 0.0571 -0.0000  0.0000
IMP_urbanicity       2.2943   0.1124  20.4047 0.0000  2.0739  2.5147
IMP_revoked          1.0183   0.0951  10.7071 0.0000  0.8319  1.2047
IMP_mstatus         -0.4933   0.0819  -6.0215 0.0000 -0.6539 -0.3328
IMP_car_use          0.7019   0.0727   9.6498 0.0000  0.5593  0.8445
IMP_parent1          0.3786   0.1091   3.4710 0.0005  0.1648  0.5925
home_own            -0.2759   0.0706  -3.9069 0.0001 -0.4143 -0.1375
university_degree   -0.4739   0.0812  -5.8358 0.0000 -0.6330 -0.3147
white_collar        -0.5058   0.0863  -5.8605 0.0000 -0.6749 -0.3366
hadFewOldClaim       0.5421   0.0697   7.7820 0.0000  0.4056  0.6786
hadManyOldClaim     -0.1345   0.1183  -1.1369 0.2556 -0.3663  0.0973
=================================================================
```

Figure 10: Model 2 ROC Curve



While more complex, this model has a better AIC, BIC, pseudo-R squared, and ROC curve area than the first model. The AIC and BIC plummeted down 7,371 and 7,561 respectively, the pseudo R squared increased to .2230 and the ROC curve area jumped to .8127. However, there are some notable flaws in this model. A few of the included independent variables have coefficients that are not statistically significantly different than zero. These should be removed. Further, I noticed that a couple of the variables had a coefficient of zero, but were statistically significant. To explore further, I printed out the parameters.

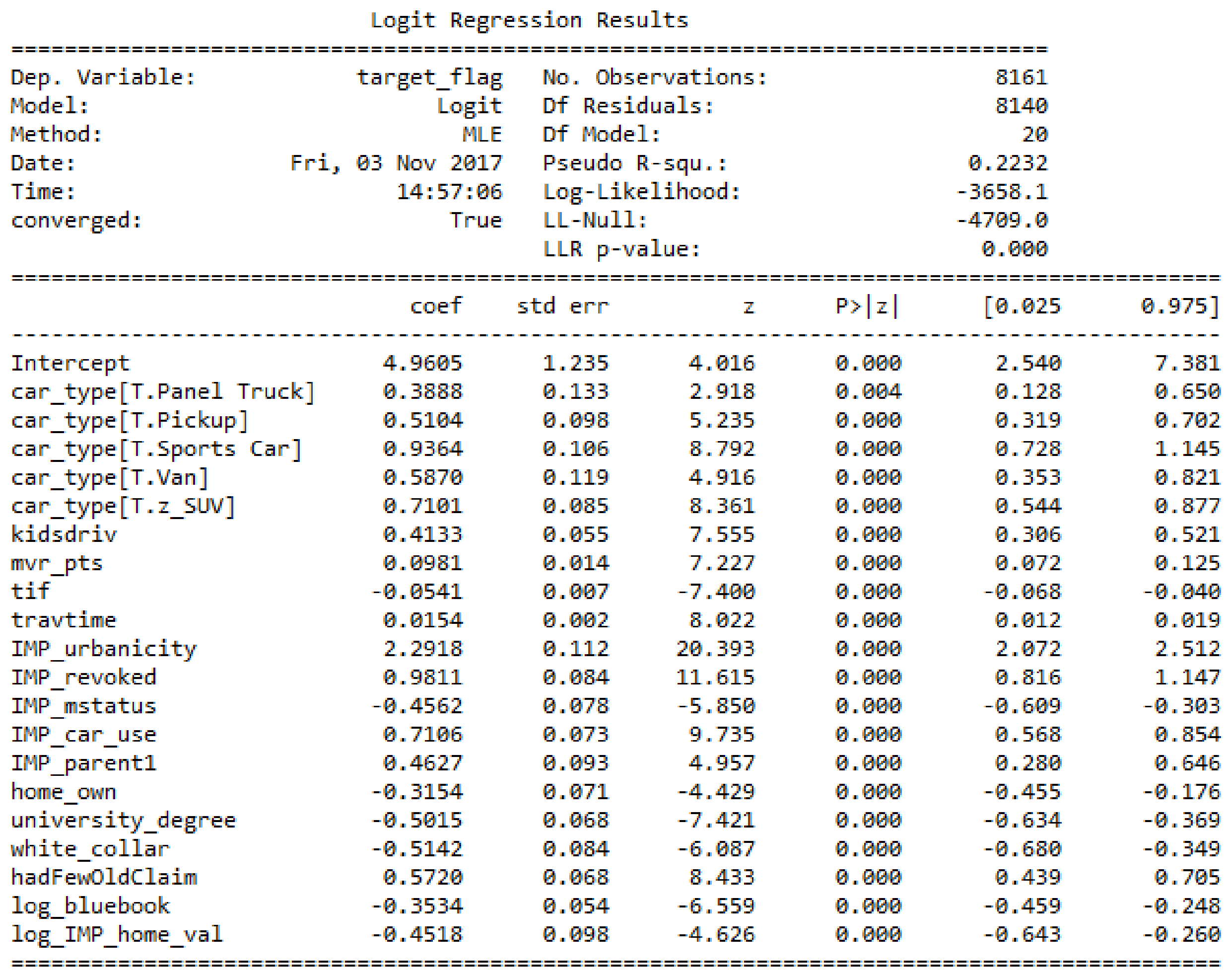
Table 12: Model 2 Parameter Coefficients



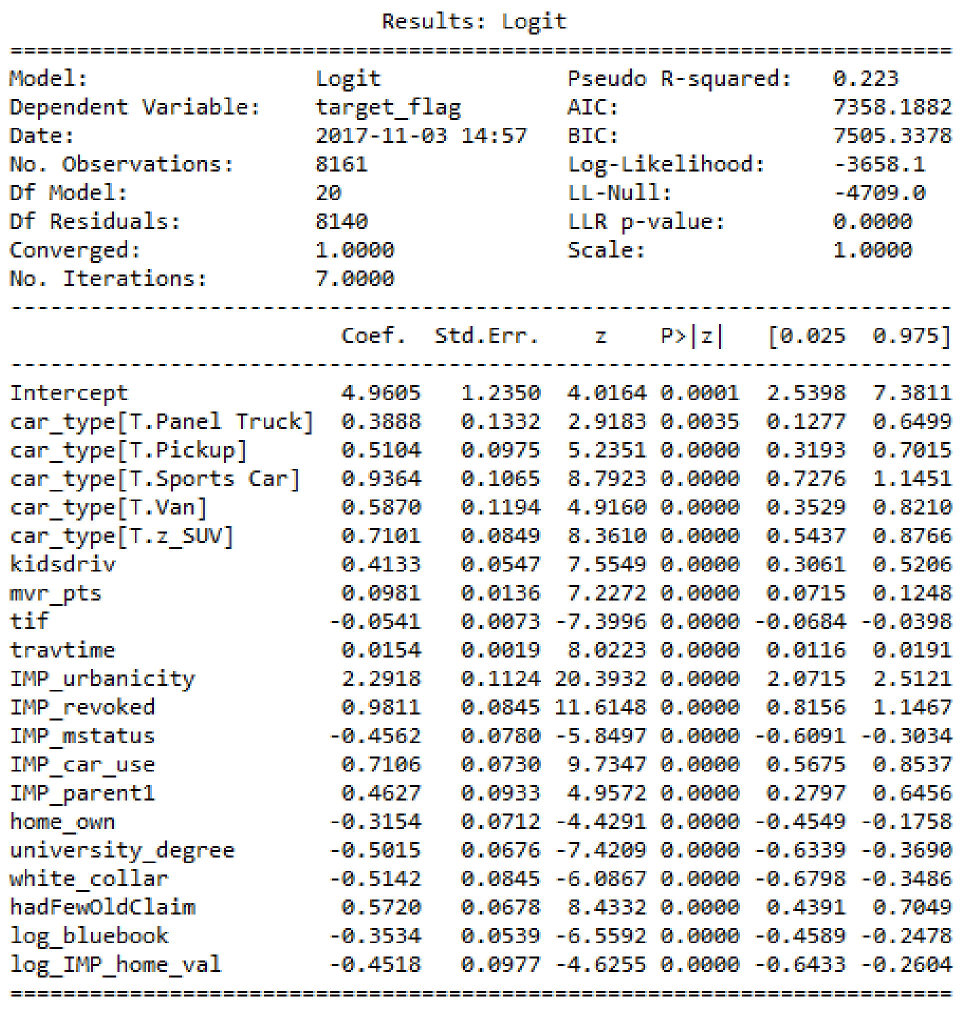
```
Intercept                 -3.026594
car_type[T.Panel Truck]    0.487362
car_type[T.Pickup]         0.512193
car_type[T.Sports Car]     1.004393
car_type[T.Van]            0.579090
car_type[T.z_SUV]          0.735654
bluebook                  -0.000025
kidsdriv                   0.385152
homekids                   0.040194
mvr_pts                    0.101270
tif                       -0.055153
travtime                   0.015256
IMP_sex                   -0.051815
IMP_age                   -0.001608
IMP_car_age                0.003245
IMP_home_val              -0.000001
IMP_income                -0.000002
IMP_urbanicity             2.294315
IMP_revoked                1.018313
IMP_mstatus               -0.493337
IMP_car_use                0.701914
IMP_parent1                0.378644
home_own                  -0.275896
university_degree         -0.473891
white_collar              -0.505765
hadFewOldClaim             0.542106
hadManyOldClaim           -0.134471
dtype: float64
```

It turned out that the coefficients were different than zero, they were just very small. This was because the values of income and home value were so large, that only a tiny number needed to be used in the model. To fix these problems, I decided to remove any variable whose coefficient was not statistically significantly different than zero, at the alpha = .05 level. Further, for the variables whose coefficients were very close to zero, I transformed them with a log transform. Since income was not statistically significant, this only included home value and bluebook. My third model consisted of these changes.

Table 13: Model 3 Summary

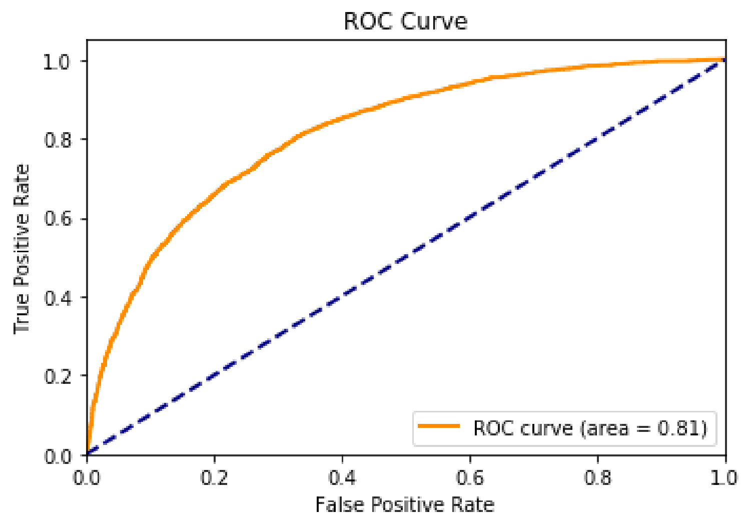


```
                           Logit Regression Results
==============================================================================
Dep. Variable:            target_flag   No. Observations:                 8161
Model:                          Logit   Df Residuals:                     8140
Method:                           MLE   Df Model:                           20
Date:                Fri, 03 Nov 2017   Pseudo R-squ.:                  0.2232
Time:                        14:57:06   Log-Likelihood:                -3658.1
converged:                       True   LL-Null:                       -4709.0
                                        LLR p-value:                     0.000
============================================================================================
                               coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------------
Intercept                    4.9605      1.235      4.016      0.000       2.540       7.381
car_type[T.Panel Truck]      0.3888      0.133      2.918      0.004       0.128       0.650
car_type[T.Pickup]           0.5104      0.098      5.235      0.000       0.319       0.702
car_type[T.Sports Car]       0.9364      0.106      8.792      0.000       0.728       1.145
car_type[T.Van]              0.5870      0.119      4.916      0.000       0.353       0.821
car_type[T.z_SUV]            0.7101      0.085      8.361      0.000       0.544       0.877
kidsdriv                     0.4133      0.055      7.555      0.000       0.306       0.521
mvr_pts                      0.0981      0.014      7.227      0.000       0.072       0.125
tif                         -0.0541      0.007     -7.400      0.000      -0.068      -0.040
travtime                     0.0154      0.002      8.022      0.000       0.012       0.019
IMP_urbanicity               2.2918      0.112     20.393      0.000       2.072       2.512
IMP_revoked                  0.9811      0.084     11.615      0.000       0.816       1.147
IMP_mstatus                 -0.4562      0.078     -5.850      0.000      -0.609      -0.303
IMP_car_use                  0.7106      0.073      9.735      0.000       0.568       0.854
IMP_parent1                  0.4627      0.093      4.957      0.000       0.280       0.646
home_own                    -0.3154      0.071     -4.429      0.000      -0.455      -0.176
university_degree           -0.5015      0.068     -7.421      0.000      -0.634      -0.369
white_collar                -0.5142      0.084     -6.087      0.000      -0.680      -0.349
hadFewOldClaim               0.5720      0.068      8.433      0.000       0.439       0.705
log_bluebook                -0.3534      0.054     -6.559      0.000      -0.459      -0.248
log_IMP_home_val            -0.4518      0.098     -4.626      0.000      -0.643      -0.260
============================================================================================
```



```
                        Results: Logit
=================================================================
Model:               Logit            Pseudo R-squared:  0.223
Dependent Variable:  target_flag      AIC:               7358.1882
Date:                2017-11-03 14:57 BIC:               7505.3378
No. Observations:    8161             Log-Likelihood:    -3658.1
Df Model:            20               LL-Null:           -4709.0
Df Residuals:        8140             LLR p-value:       0.0000
Converged:           1.0000           Scale:             1.0000
No. Iterations:      7.0000
-----------------------------------------------------------------
                       Coef.  Std.Err.    z    P>|z|  [0.025 0.975]
-----------------------------------------------------------------
Intercept              4.9605  1.2350  4.0164 0.0001  2.5398  7.3811
car_type[T.Panel Truck] 0.3888  0.1332  2.9183 0.0035  0.1277  0.6499
car_type[T.Pickup]     0.5104  0.0975  5.2351 0.0000  0.3193  0.7015
car_type[T.Sports Car] 0.9364  0.1065  8.7923 0.0000  0.7276  1.1451
car_type[T.Van]        0.5870  0.1194  4.9160 0.0000  0.3529  0.8210
car_type[T.z_SUV]      0.7101  0.0849  8.3610 0.0000  0.5437  0.8766
kidsdriv               0.4133  0.0547  7.5549 0.0000  0.3061  0.5206
mvr_pts                0.0981  0.0136  7.2272 0.0000  0.0715  0.1248
tif                   -0.0541  0.0073 -7.3996 0.0000 -0.0684 -0.0398
travtime               0.0154  0.0019  8.0223 0.0000  0.0116  0.0191
IMP_urbanicity         2.2918  0.1124 20.3932 0.0000  2.0715  2.5121
IMP_revoked            0.9811  0.0845 11.6148 0.0000  0.8156  1.1467
IMP_mstatus           -0.4562  0.0780 -5.8497 0.0000 -0.6091 -0.3034
IMP_car_use            0.7106  0.0730  9.7347 0.0000  0.5675  0.8537
IMP_parent1            0.4627  0.0933  4.9572 0.0000  0.2797  0.6456
home_own              -0.3154  0.0712 -4.4291 0.0000 -0.4549 -0.1758
university_degree     -0.5015  0.0676 -7.4209 0.0000 -0.6339 -0.3690
white_collar          -0.5142  0.0845 -6.0867 0.0000 -0.6798 -0.3486
hadFewOldClaim         0.5720  0.0678  8.4332 0.0000  0.4391  0.7049
log_bluebook          -0.3534  0.0539 -6.5592 0.0000 -0.4589 -0.2478
log_IMP_home_val      -0.4518  0.0977 -4.6255 0.0000 -0.6433 -0.2604
=================================================================
```

Figure 11: Model 3 ROC Curve



This model is by far the best. The pseudo R-squared increased slightly to .2232 and the ROC area curve increased slightly to .8128. The AIC and BIC decreased immensely though, down to 7,358 and 7,505 respectively. Furthermore, not only does each variable have a statistically significant coefficient, the signs of each variable correspond to the theoretical effect. This means there is minimal multi-collinearity and overfitting issues.

Model Selection

Below is a table comparing the selected evaluation metrics for each model:

Table 14: Model Comparison

	Model 1	**Model 2**	**Model 3**
Pseudo R^2	.1224	.2230	.2232
AIC	8,277	7,371	7,358
BIC	8,319	7,561	7,505
ROC	.7323	.8127	.8128

I chose Model 3 to implement due its relative success across all metrics when compared to its competitors. Further, each variable coefficient matches with theoretical behavior. All variables with a negative sign indicate a potential decrease in crashing risk, while a plus sign indicates a variable of increased risk. The entire model equation is written out below:

P_TARGET_FLAG = 4.96 + .41 * KIDSDRIV + .098 * MVR_POINTS + .0154 * TRAVTIME + 2.292 * URBANCITY + .98 * REVOKED + .71 * CARUSE + .462 * PARENT1 + .572 * HADFEWOLDCLAIMS + .39 * CAR_TYPE [Panel Truck] + .51 * CAR_TYPE [Pickup] + .94 * CAR_TYPE [Sports Car] + .587 * CAR_TYPE [Van] + .71 * CAR_TYPE [SUV] - .054 * TIF - .456 * MSTATUS - .315 * HOME_OWNERSHIP - .501 * UNIVERSITY_DEGREE - .514 * WHITE_COLLAR_JOB - .353 * LOG_BLUEBOOK - .452 * LOG_HOME_VALUE

Model Explanation (Crash Probability Logistic Regression):

For any missing data, the median should be used. For any missing job data, a level for “No Data Provided” should be used. Jobs should then be split into an indicator as to whether it is a white collar job, e.g. manager, doctor, or lawyer. Furthermore, an indicator should be established indicating whether a person achieved any higher level degree or not. KIDSDRIV is the number of kids a person has driving. MVR_POINTS are the number of motor vehicle record points a person has. TRAVTIME is the number of minutes a person’s commute takes. URBANCITY is a 1 if a person lives in an urban area, else 0. REVOKED is a 1 if a person’s license has been revoked or not. CARUSE is 1 is a person’s vehicle is used for commercial purposes, else 0. PARENT1 is 1 if a person is a single parent, else 0. HADFEWCLAIMS is 1 if a person had previously filed between $1 and $12,000 worth of claims.

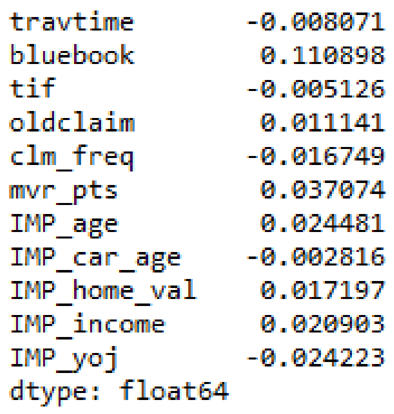
For whichever car type a person has, the corresponding variable should be 1, with the others as 0. The TIF is the time in force, measured in years. MSTATUS should be 1 if a person is married, else 0. HOME_OWNERSHIP is 1 if a person owns a house, else 0. UNIVERSITY_DEGREE should be 1 if a person has achieved a higher level degree, else 0. WHITE_COLLAR_JOB should be if a person has a white collar job, as defined above, else 0. LOG_BLUEBOOK is the natural logarithm of a person’s vehicular bluebook value. Finally, LOG_HOME_VALUE is the natural logarithm of a person’s home value. If a person does not own a home, this value is 0.

Model Construction and Selection (OLS Regression to Estimate Cost)

Given that a person got into a crash, how much should we expect them to pay? This target amount is zero if the person did not crash their car. But if they did crash their car, this number will be a value greater than zero. In this part of the assignment I will come up with a model to predict what it will cost if a person does crash their car.

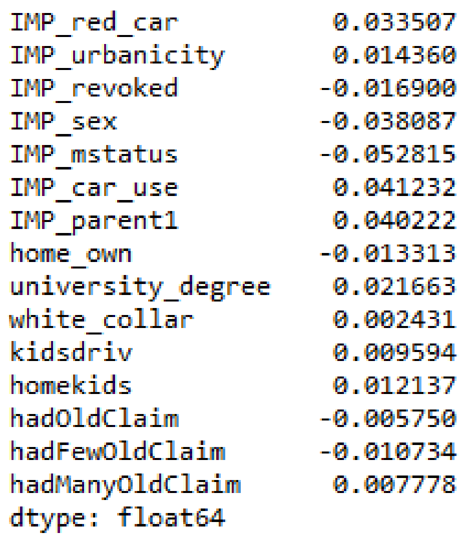
As this model is not a major part of the assignment, I will create a simple OLS regression model. First, I checked the correlation with each variable against the target cost, given that a person got into a crash.

Table 15: Correlations of Continuous Variables with Target Cost



```
travtime       -0.008071
bluebook        0.110898
tif            -0.005126
oldclaim        0.011141
clm_freq       -0.016749
mvr_pts         0.037074
IMP_age         0.024481
IMP_car_age    -0.002816
IMP_home_val    0.017197
IMP_income      0.020903
IMP_yoj        -0.024223
dtype: float64
```

Table 16: Correlations of Categorical Variables with Target Cost



```
IMP_red_car           0.033507
IMP_urbanicity        0.014360
IMP_revoked          -0.016900
IMP_sex              -0.038087
IMP_mstatus          -0.052815
IMP_car_use           0.041232
IMP_parent1           0.040222
home_own             -0.013313
university_degree     0.021663
white_collar          0.002431
kidsdriv              0.009594
homekids              0.012137
hadOldClaim          -0.005750
hadFewOldClaim       -0.010734
hadManyOldClaim       0.007778
dtype: float64
```

While none of the variables correlate to closely with the target cost, I chose the three variables that correlated the most. Those variables were BLUEBOOK, IMP_MSTATUS, and MVR_POINTS. Since bluebook had to be log transformed in the last model, I similarly used a log transform on that variable here. The output from this model is below.

Table 17: Cost Model Summary

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             target_amt   R-squared:                       0.019
Model:                            OLS   Adj. R-squared:                  0.018
Method:                 Least Squares   F-statistic:                     14.07
Date:                Fri, 03 Nov 2017   Prob (F-statistic):           4.41e-09
Time:                        17:42:03   Log-Likelihood:                -20940.
No. Observations:                2153   AIC:                         4.189e+04
Df Residuals:                    2149   BIC:                         4.191e+04
Df Model:                           3
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept    -2113.3893   1258.219     -1.680      0.093   -4580.844     354.065
log_bluebook   764.3108    132.754      5.757      0.000     503.971    1024.650
IMP_mstatus   -402.4122    174.962     -2.300      0.022    -745.524     -59.300
mvr_pts         63.9457     33.920      1.885      0.060      -2.573     130.465
==============================================================================
Omnibus:                      979.439   Durbin-Watson:                   2.011
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4316.894
Skew:                           2.230   Prob(JB):                         0.00
Kurtosis:                       8.313   Cond. No.                         142.
==============================================================================
```

Figure 12: Residuals vs Fitted Values

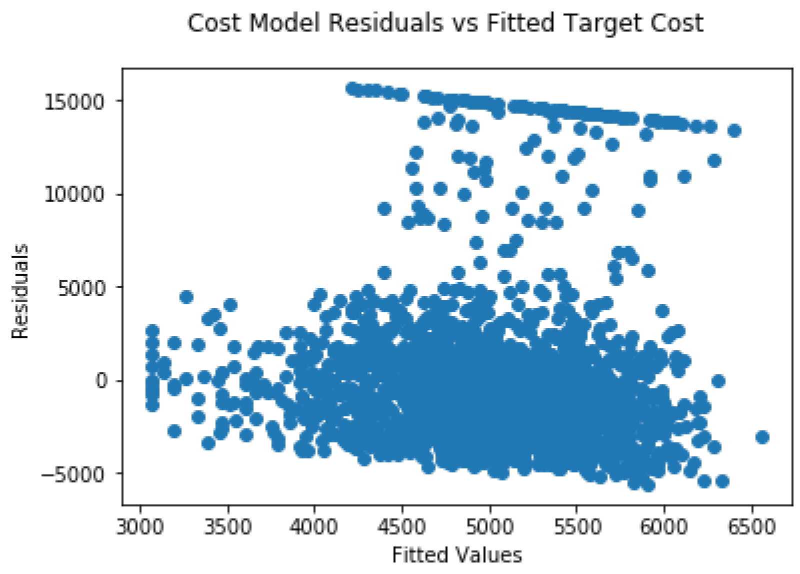
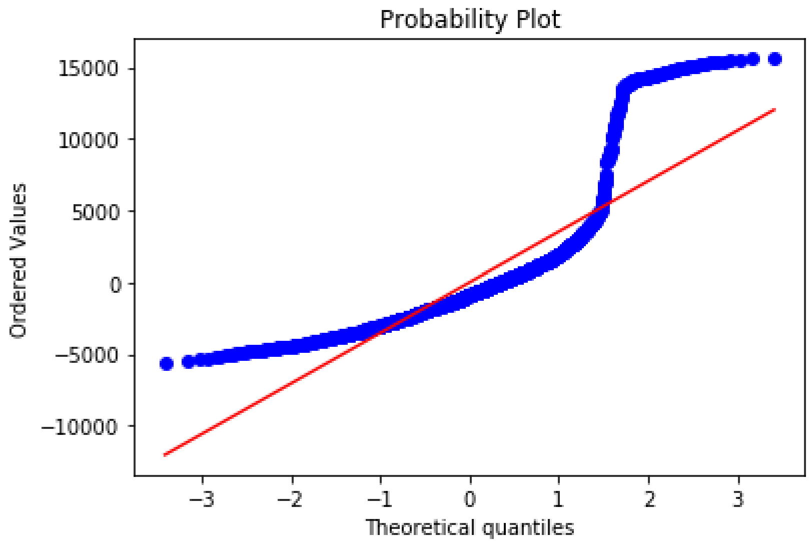


Figure 13: QQ-Plot



This model is not great. The Adjusted R-Squared is quite low, the AIC and BIC are quite high, and the QQ-Plot shows that the residuals do not follow a normal distribution at all. The mean absolute error is enormous at 2,641. While I expect there are better models out there, and despite the poor performance, I think it is best to choose a simple model for implementation in the deployment phase.

Model Explanation (Cost OLS Regression):

The entire model equation is written below:

P_TARGET_AMT = -2113 + 764.31 * LOG_BLUEBOOK + 63.9 * MVR_POINTS - 402.4 IMP_MSTATUS.

LOG_BLUEBOOK is the natural logarithm of a person’s vehicular bluebook value. MVR_POINTS are the number of motor vehicle record points a person has. Finally, MSTATUS should be 1 if a person is married, else 0. Interestingly, the coefficients adhere to theoretical behaviors. The more expensive the car and the more motor vehicle points on a person’s record, the greater their cost. However, being married seems to lower their total cost by over $400.

Conclusion:

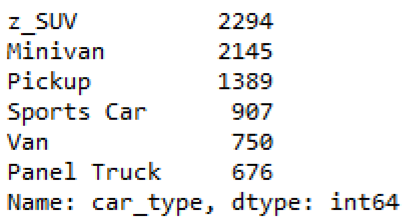
The construction of logistics models requires a decent amount of initial data manipulation. Within the data set I was given were few variables that had already been prepared slightly to assist in this process. This meant that there was still a lot of iterative work to examine variables that would be acceptable to take forward into the model. In the case of using VIF and correlation with the dependent variable, I noticed that almost every single variable qualified to be taken forward. However, once in the model, I removed quite a few that failed to be statistically significant. As time goes on and we continue to utilize this modeling methodology it will be interesting to examine how early observation of other statistics will be indicative of need for incorporation into the model, and likely more importantly how the variable needs to be transformed. I did not explore exhaustive transformation of variables that were incorporated into our model and it likely will impact the overall performance of the models constructed. However, the variables I did transform improved the model eventually selected. The logistics model is more naturally interpretable than previous models I have constructed with the OLS method.

While the model selected is quite complex, simpler models suffered from poor predictability. I valued the ability to more accurately predict a crash, than eliminate variables to reduce dimensionality. I am quite proud of the eventual model. The variables included make intuitive sense with their respective impact, and again it is performing well on Kaggle. I am excited to perform more research to determine if Actuaries predict risk using this same kind of techniques. While I did not even comprehensively exhaust all of the modeling options at my disposal, I feel that management can still use this solution to achieve a competitive advantage when assigning insurance rates and risks to customers. Finally, having a red car really has no impact on crashing or the cost one would have to pay. Well, unless it is a Ferrari.

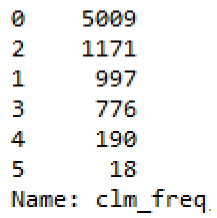
Appendix

Exploratory Data Analysis Graphics

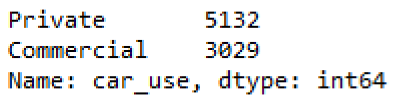
Categorical Tables



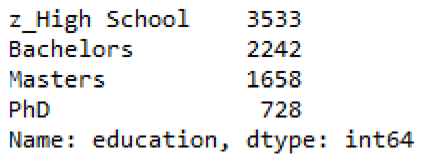
```
z_SUV          2294
Minivan        2145
Pickup         1389
Sports Car      907
Van             750
Panel Truck     676
Name: car_type, dtype: int64
```



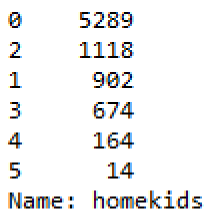
```
0    5009
2    1171
1     997
3     776
4     190
5      18
Name: clm_freq
```



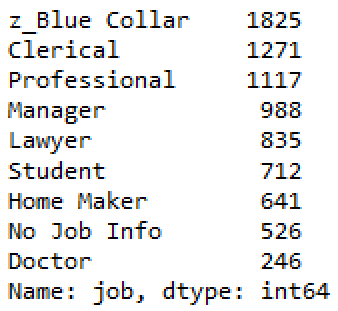
```
Private       5132
Commercial    3029
Name: car_use, dtype: int64
```



```
z_High School    3533
Bachelors        2242
Masters          1658
PhD               728
Name: education, dtype: int64
```



```
0    5289
2    1118
1     902
3     674
4     164
5      14
Name: homekids
```



```
z_Blue Collar    1825
Clerical         1271
Professional     1117
Manager           988
Lawyer            835
Student           712
Home Maker        641
No Job Info       526
Doctor            246
Name: job, dtype: int64
```

```
0    7180
1     636
2     279
3      62
4       4
Name: kidsdriv
```

```
Yes     4894
z_No    3267
Name: mstatus
```

```
no      5783
yes     2378
Name: red_car, dtype: int64
```

```
No      7161
Yes     1000
Name: revoked, dtype: int64
```

```
0    6008
1    2153
Name: target_flag
```

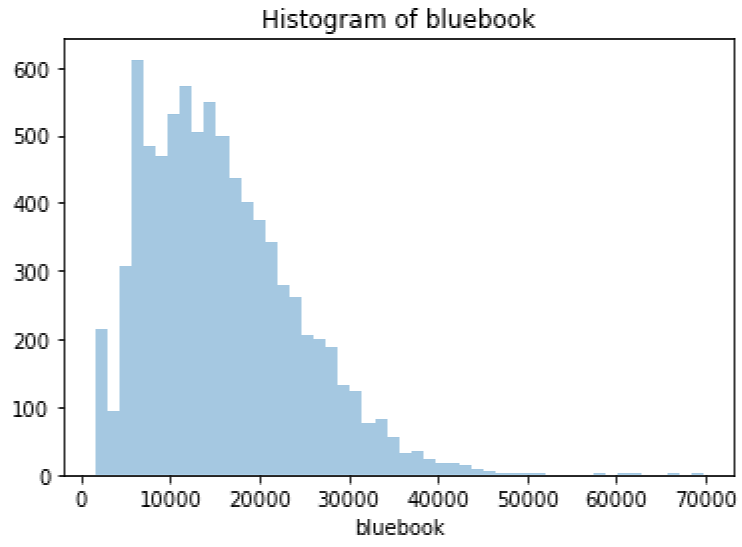
```
z_F    4375
M      3786
Name: sex, dtype: int64
```

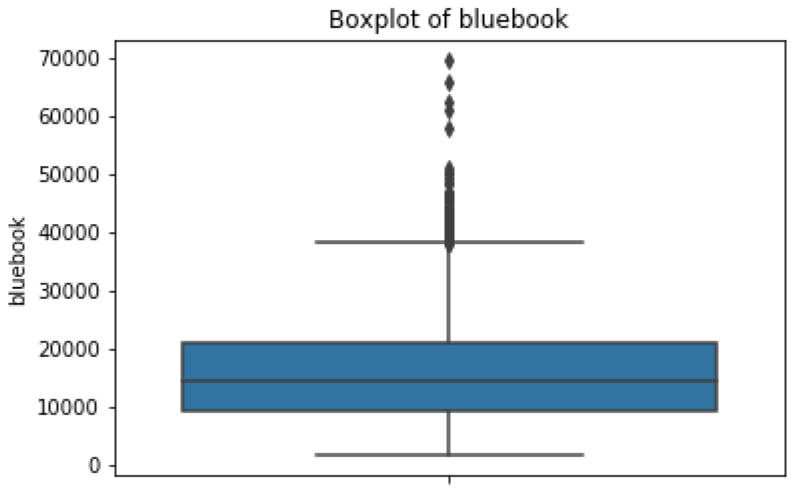
```
1    4628
0    3533
Name: university_degree
```

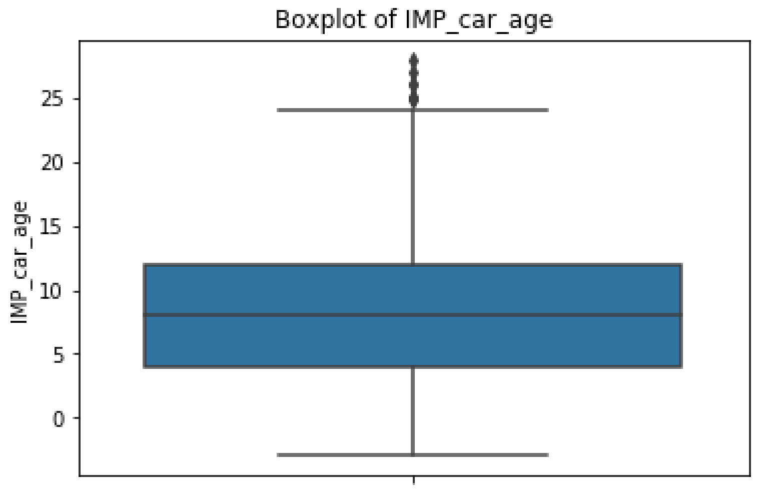
```
Highly Urban/ Urban      6492
z_Highly Rural/ Rural    1669
Name: urbanicity, dtype: int64
```

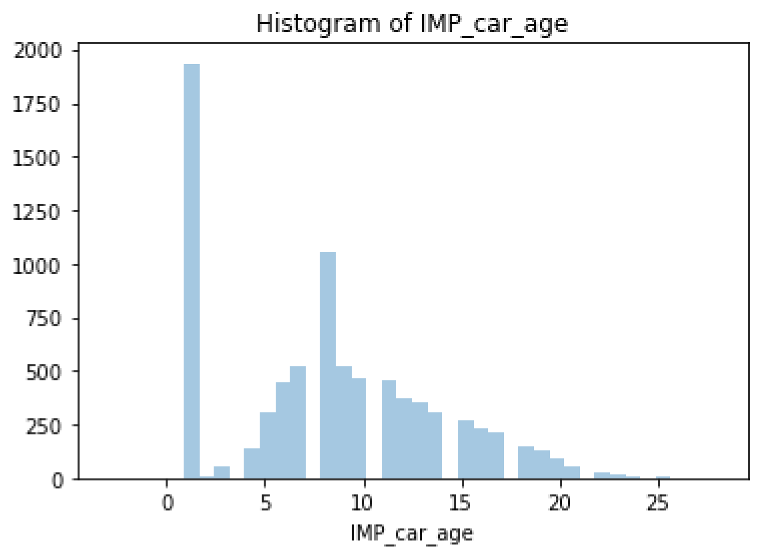
```
0    6092
1    2069
Name: white_collar
```

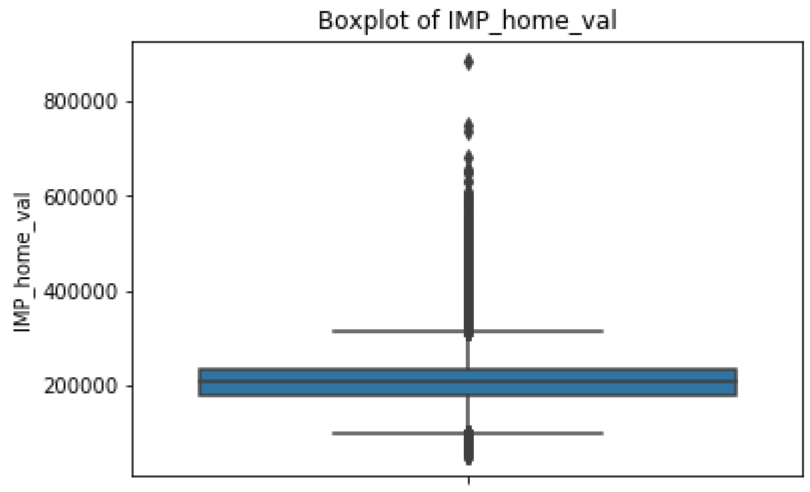
Continuous Variable Graphics

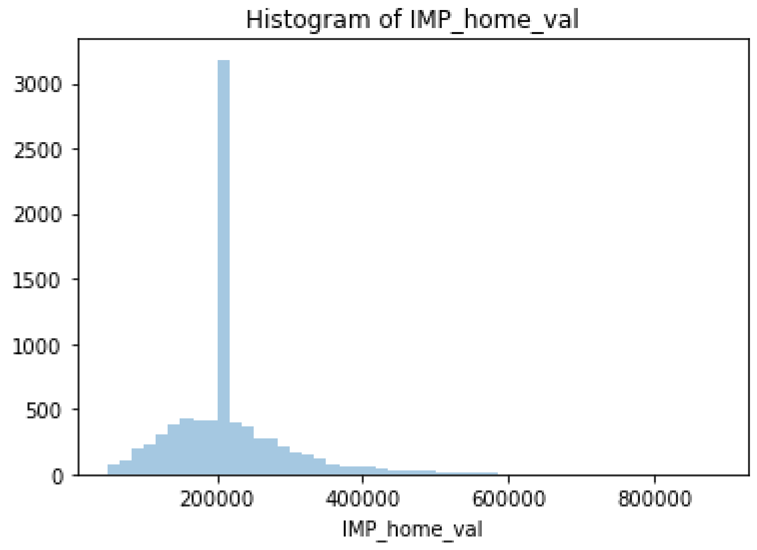


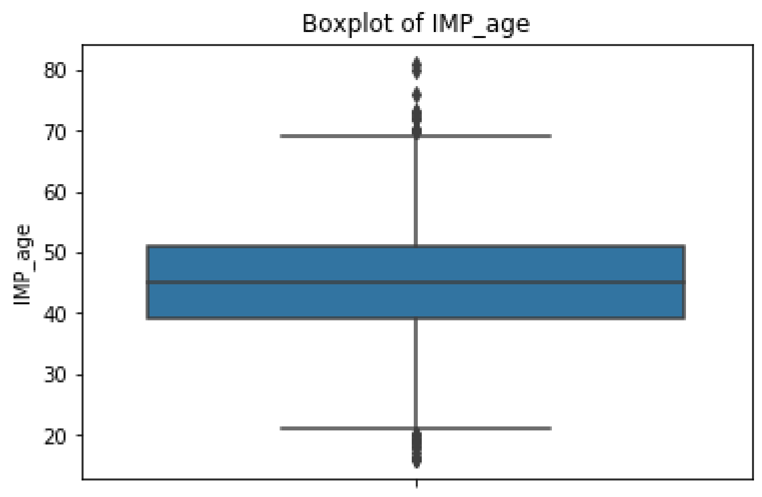


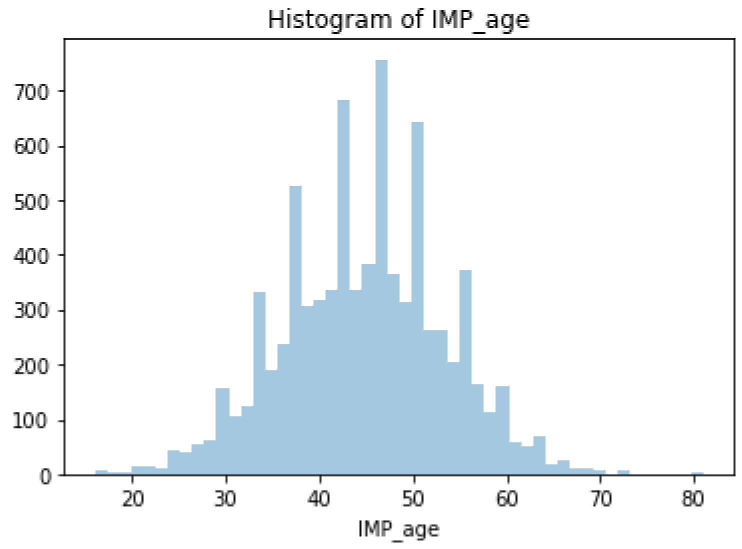


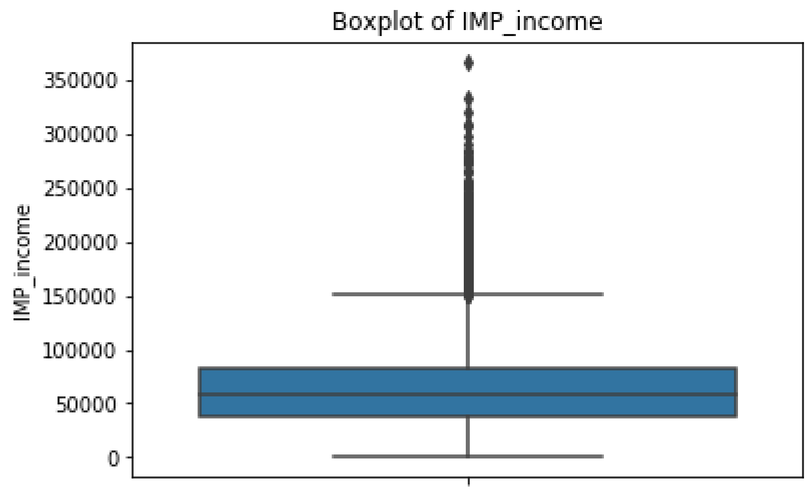

Histogram of IMP_car_age
2000
1750
1500
1250
1000
750
500
250
0
0
5
10
15
20
25
IMP_car_age

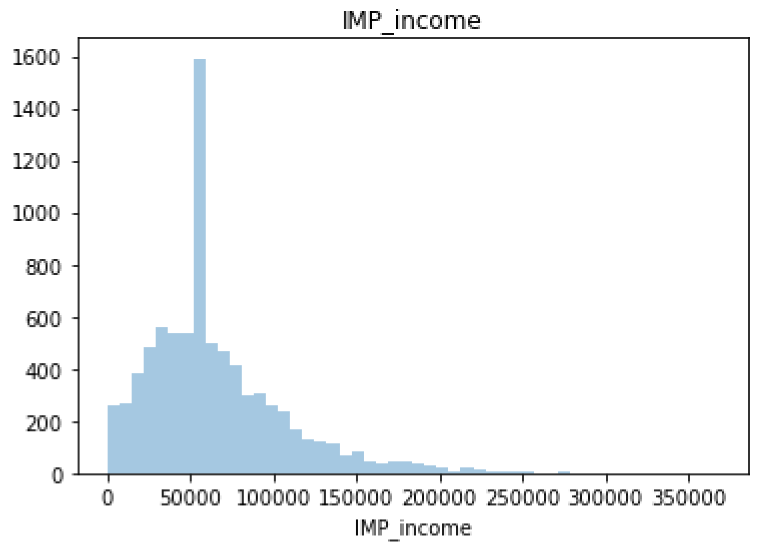

Boxplot of IMP_home_val
800000
600000
400000
200000
IMP_home_val

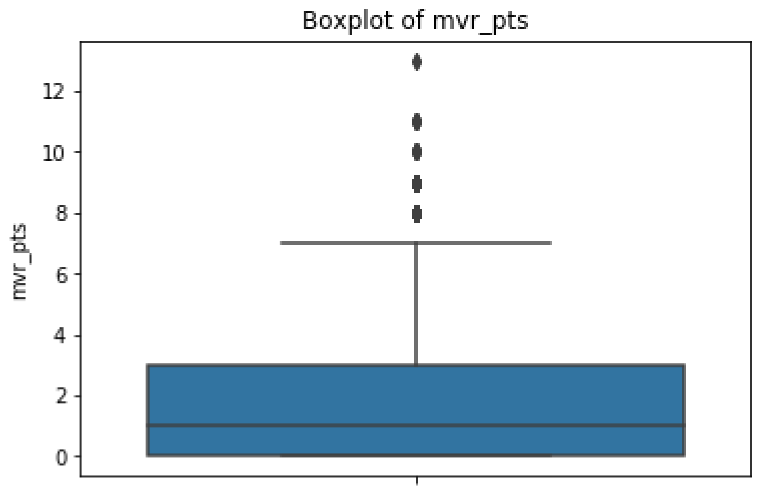

Histogram of IMP_home_val
3000
2500
2000
1500
1000
500
0
200000
400000
600000
800000
IMP_home_val

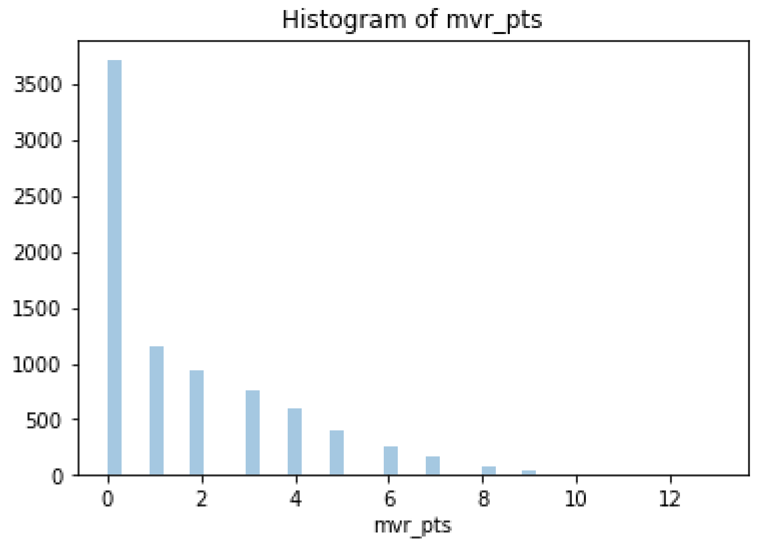

Boxplot of IMP_age
IMP_age
80
70
60
50
40
30
20

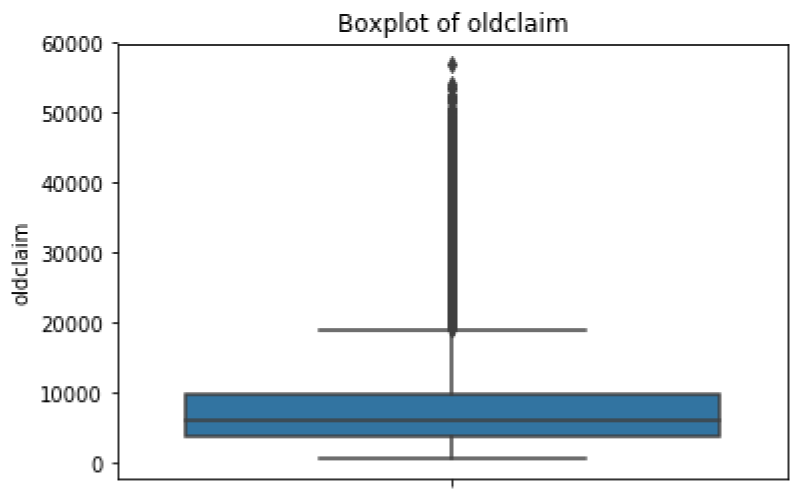

Histogram of IMP_age
700
600
500
400
300
200
100
0
20
30
40
50
60
70
80
IMP_age

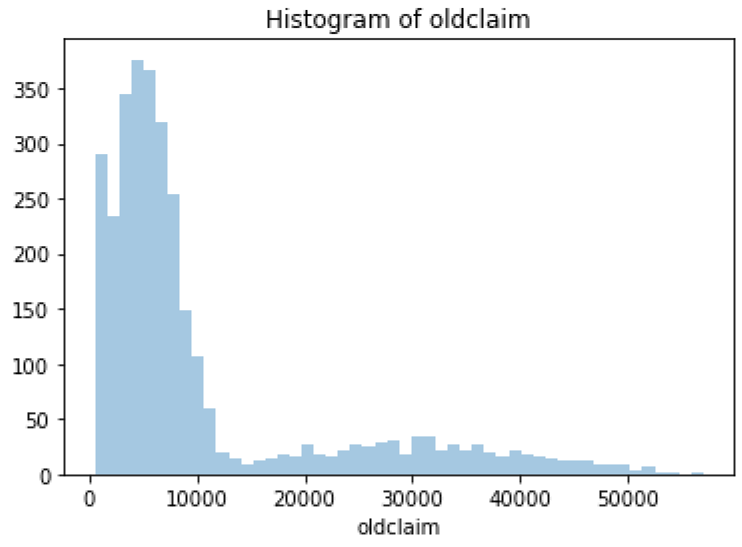

Boxplot of IMP_income
IMP_income
350000
300000
250000
200000
150000
100000
50000
0

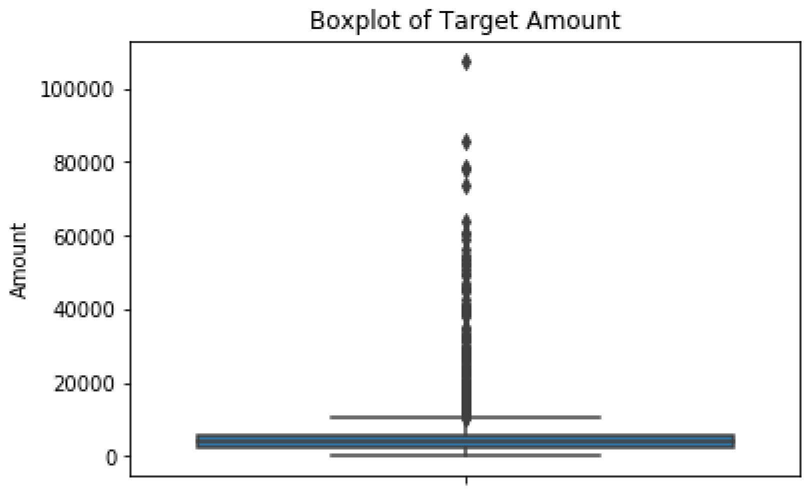

IMP_income
1600
1400
1200
1000
800
600
400
200
0
0
50000
100000
150000
200000
250000
300000
350000
IMP_income

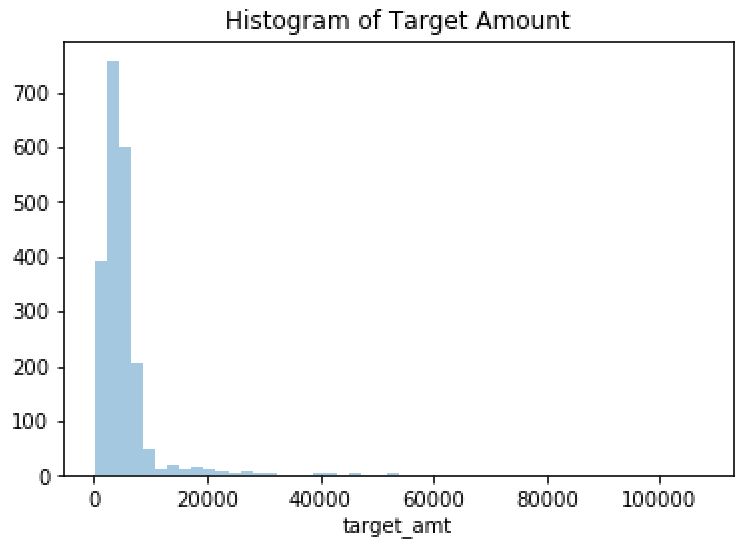

Boxplot of mvr_pts
12
10
8
6
4
2
0
mvr_pts

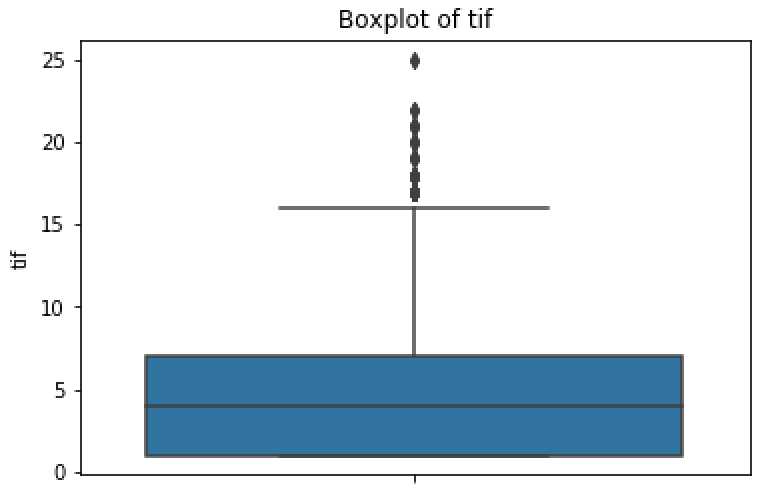

Histogram of mvr_pts
3500
3000
2500
2000
1500
1000
500
0
0
2
4
6
8
10
12
mvr_pts

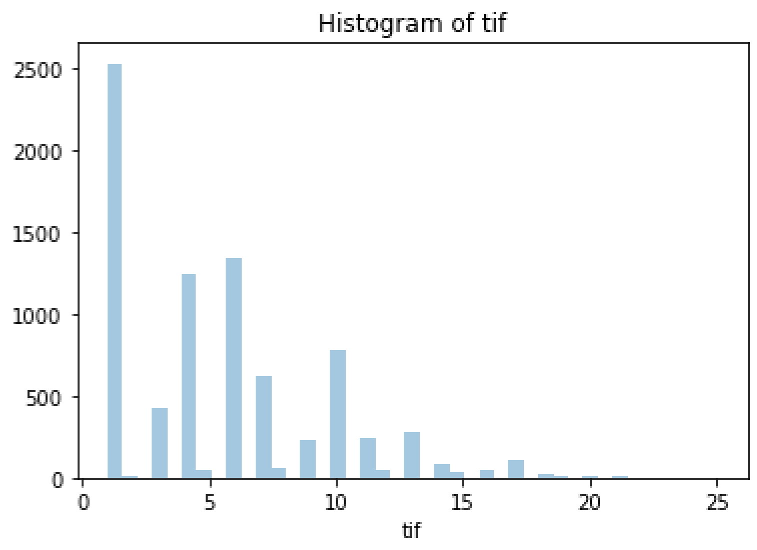

Boxplot of oldclaim
oldclaim
60000
50000
40000
30000
20000
10000
0

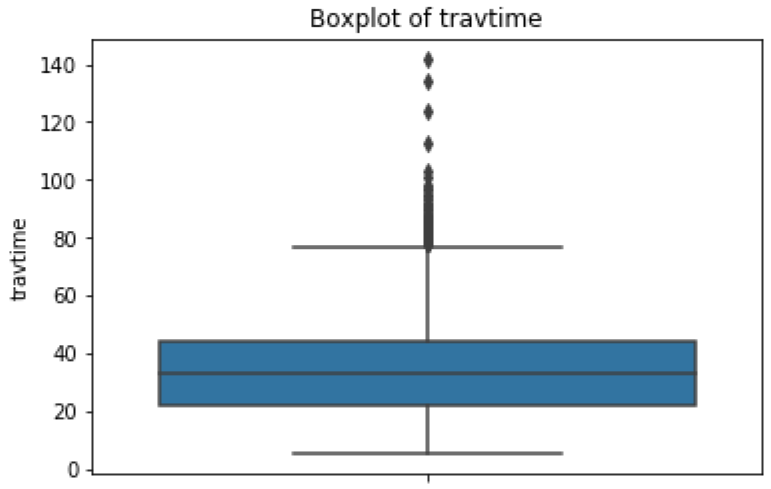

Histogram of oldclaim
350
300
250
200
150
100
50
0
0
10000
20000
30000
40000
50000
oldclaim

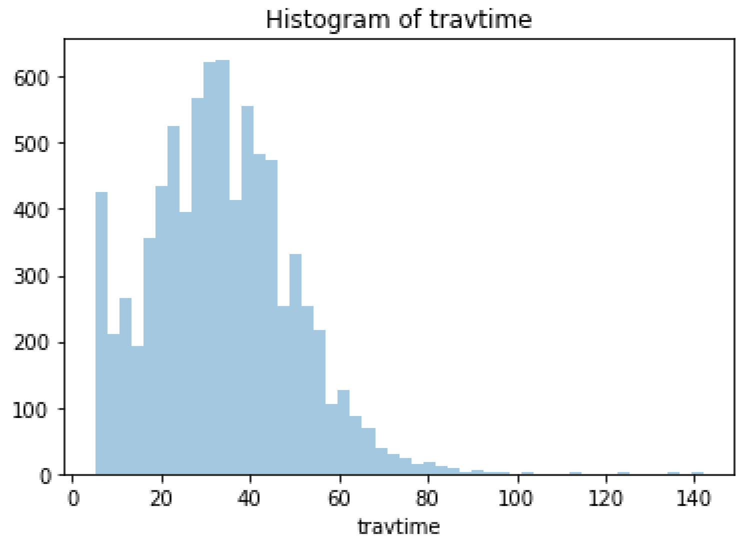

Boxplot of Target Amount
Amount
100000
80000
60000
40000
20000
0

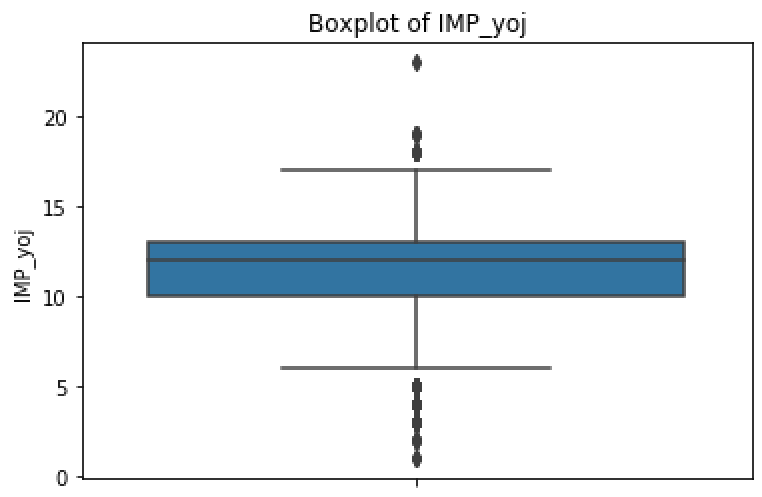

Histogram of Target Amount
700
600
500
400
300
200
100
0
0
20000
40000
60000
80000
100000
target_amt

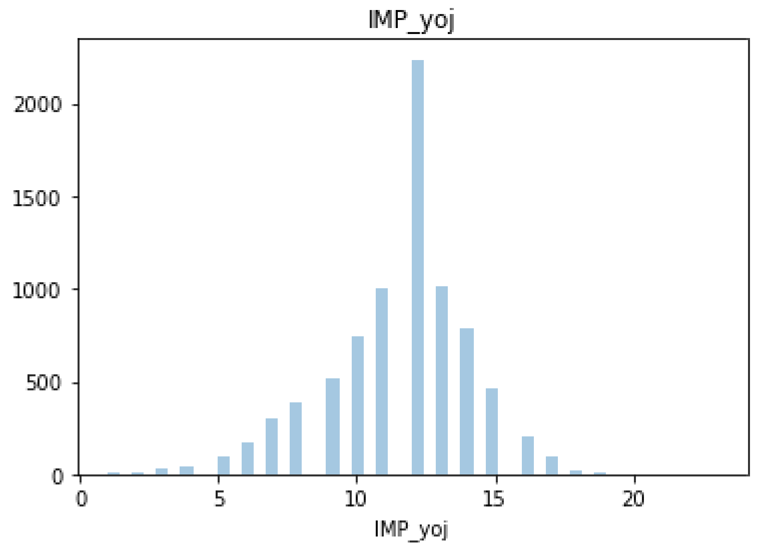

Boxplot of tif
25
20
15
10
5
0
tif


Histogram of tif
2500
2000
1500
1000
500
0
0
5
10
15
20
25
tif


Boxplot of travtime
140
120
100
80
60
40
20
0
travtime


Histogram of travtime
600
500
400
300
200
100
0
0
20
40
60
80
100
120
140
travtime


Boxplot of IMP_yoj
IMP_yoj
0
5
10
15
20


IMP_yoj
0
500
1000
1500
2000
0
5
10
15
20
IMP_yoj

Code:

```
# -*- coding: utf-8 -*-
"""
Created on Sun Oct 29 17:43:43 2017

@author: Alexander
"""

# Imports
# prepare for Python version 3x features and functions
from __future__ import division, print_function

import os
import pandas as pd
import numpy as np  # arrays and math functions
import seaborn as sns
import matplotlib.pyplot as plt
from re import sub
from decimal import Decimal
import statsmodels.formula.api as smf  # R-like model specification
from statsmodels.stats.outliers_influence import variance_inflation_factor
from patsy import dmatrices
from sklearn import metrics
import scipy.stats as stats

#Set Directory
os.chdir(r"C:\Users\Alexander\Documents\Northwestern\Fall 2017\411\Unit 02\Project 2")

#Read in the auto ins dataset
train = pd.read_csv('train_auto.csv')
test = pd.read_csv('test_auto.csv')

#A good step to take is to convert all variable names to lower case
train.columns = [s.lower() for s in train.columns]
test.columns = [s.lower() for s in train.columns]

#get columns for reference
cols = list(train.columns.values)
cols.sort()

print('')
print('----- Summary of Input Data -----')
print('')

# show the object is a DataFrame
print('Object type: ', type(train))

# show the object's shape
print('Object shape: ', train.shape)

# show number of observations in the DataFrame
print('Number of observations: ', len(train))

# show variable names
print('Variable names: ', train.columns)

# show head
print(train.head(5))

# show descriptive statistics
```

```
print(train.describe())

#Na Analysis
naColsCount = train.isnull().sum()
naCols = train.isnull().any()

#Note the missing values are recorded as NaN, we need to replace these with something
#(median? or your choice)
train1 = train.copy()

#One of the columns with NaN is categorical. Replace with "No Job Info"
train1.job = train.job.fillna("No Job Info")
naCols = train1.isnull().any()

#there are dollar signs and commas in some of these variables
#convert to int
for indx in cols:
    if train[indx].astype(str).str.contains("\$").any() == True:
        tempVals = []
        for money in train[indx].values:
            if not pd.isnull(money):
                value = int(Decimal(sub(r'[^\d\-.]', '', money)))
                tempVals.append(value)
            else:
                tempVals.append(np.nan)
        train1[indx] = tempVals

#Create home own variable. 1 if home val is greater than 0 and not NaN
train1["home_own"] = np.where(train1["home_val"].fillna(0) > 0, 1, 0)

#start by converting all NaN values to 0
train1=train1.fillna(0)

#convert all NaN values to median
for indx in cols:
    if naCols[indx] == True:
        newName = "IMP_" + indx
        newInd = "m_" + indx
        m = np.median(train1[indx][train1[indx] > 0])
        train1[newName] = train1[indx].replace({0: m}).astype(int)
        train1[newInd] = (train1[indx] == 0).astype(int)

#delete variables that had missing variable imputed
df = train1.copy()
del(df["index"])
del(df["yoj"])
del(df["income"])
del(df["home_val"])
del(df["age"])
del(df["car_age"])
##################################################################################
dfCols = df.columns.values

#Gets SIGNIFICANT outliers
def get_outliers(pdSer):
    """Get outliers from a pandas series
    """
    quants = pdSer.quantile([.25, .75])
    iqr = quants[.75] - quants[.25]

    lower = quants[.25] - 3 * iqr
    upper = quants[.75] + 3 * iqr
    return [value for value in pdSer if value < lower or value > upper]


#EDA
print(df.describe())

contVars = ['target_amt', 'target_flag', 'travtime', 'bluebook', 'tif', 'oldclaim',
            'clm_freq', 'mvr_pts', 'IMP_age', 'IMP_car_age','IMP_home_val',
            'IMP_income', 'IMP_yoj']
#EDA
#Targets
#counts of Flag
df.target_flag.value_counts()

#Plots of Amnt
ax = sns.boxplot(df[df.target_amt > 0].target_amt, orient='v')
ax.set_title("Boxplot of Target Amount")
ax.set_ylabel("Amount")
plt.show()

ax = sns.distplot(df[df.target_amt > 0].target_amt, kde=False)
ax.set_title("Histogram of Target Amount")
plt.show()

#Get outliers
print(get_outliers(df.target_amt))
#Couple of outliers here

#Categorical
#counts of kidsdriv
df.kidsdriv.value_counts()

#counts of homekids
df.homekids.value_counts()

#counts of parent1
df.parent1.value_counts()

#counts of mstatus
df.mstatus.value_counts()

#counts of sex
df.sex.value_counts()

#counts of education
df.education.value_counts()
#need to fix high School
df.loc[df.education == "<High School", 'education'] = "z_High School"
df.education.value_counts()

#counts of job
df.job.value_counts()

#counts of car_use
df.car_use.value_counts()

#counts of car_type
df.car_type.value_counts()

#counts of red_car
```

```
df.red_car.value_counts()

#counts of revoked
df.revoked.value_counts()

#counts of urbanicity
df.urbanicity.value_counts()

#counts of clm_freq
df.clm_freq.value_counts()

df["white_collar"] = np.where(df["job"].isin(["Doctor", "Manager",
       "Lawyer"]), 1, 0)
df["university_degree"] = np.where(df["education"].isin(["Bachelors",
       "Masters", "PhD"]), 1, 0)

df.white_collar.value_counts()

df.university_degree.value_counts()

df.home_own.value_counts()

#Continuous
#Plots of travtime
ax = sns.boxplot(df.travtime, orient='v')
ax.set_title("Boxplot of travtime")
ax.set_ylabel("travtime")
plt.show()

ax = sns.distplot(df.travtime, kde=False)
ax.set_title("Histogram of travtime")
plt.show()

#Get outliers
print(get_outliers(df.travtime))
#couple of outliers

#Plots of bluebook
ax = sns.boxplot(df.bluebook, orient='v')
ax.set_title("Boxplot of bluebook")
ax.set_ylabel("bluebook")
plt.show()

ax = sns.distplot(df.bluebook, kde=False)
ax.set_title("Histogram of bluebook")
plt.show()

#Get outliers
print(get_outliers(df.bluebook))
#couple of outliers

#Plots of tif
ax = sns.boxplot(df.tif, orient='v')
ax.set_title("Boxplot of tif")
ax.set_ylabel("tif")
plt.show()

ax = sns.distplot(df.tif, kde=False)
ax.set_title("Histogram of tif")
plt.show()
```

```
#Get outliers
print(get_outliers(df.tif))

#Plots of oldclaim
ax = sns.boxplot(df[df.oldclaim > 0].oldclaim, orient='v')
ax.set_title("Boxplot of oldclaim")
ax.set_ylabel("oldclaim")
plt.show()

ax = sns.distplot(df[df.oldclaim > 0].oldclaim, kde=False)
ax.set_title("Histogram of oldclaim")
plt.show()

#Get outliers
print(get_outliers(df[df.oldclaim > 0].oldclaim))

df["hadOldClaim"] = np.where(df["oldclaim"] > 0, 1, 0)
df["hadFewOldClaim"] = np.where((df["oldclaim"] > 0) & (df["oldclaim"] <= 12000), 1, 0)
df["hadManyOldClaim"] = np.where(df["oldclaim"] > 12000, 1, 0)

df.hadOldClaim.value_counts()
df.hadFewOldClaim.value_counts()
df.hadManyOldClaim.value_counts()

#Plots of clm_freq
ax = sns.distplot(df.clm_freq, kde=False)
ax.set_title("Histogram of clm_freq")
plt.show()


#Plots of mvr_pts
ax = sns.boxplot(df.mvr_pts, orient='v')
ax.set_title("Boxplot of mvr_pts")
ax.set_ylabel("mvr_pts")
plt.show()

ax = sns.distplot(df.mvr_pts, kde=False)
ax.set_title("Histogram of mvr_pts")
plt.show()

#Get outliers
print(get_outliers(df.mvr_pts))

#Plots of IMP_age
ax = sns.boxplot(df.IMP_age, orient='v')
ax.set_title("Boxplot of IMP_age")
ax.set_ylabel("IMP_age")
plt.show()

ax = sns.distplot(df.IMP_age, kde=False)
ax.set_title("Histogram of IMP_age")
plt.show()

#Get outliers
print(get_outliers(df.IMP_age))

#Plots of IMP_car_age
ax = sns.boxplot(df.IMP_car_age, orient='v')
ax.set_title("Boxplot of IMP_car_age")
ax.set_ylabel("IMP_car_age")
plt.show()
```

```
ax = sns.distplot(df.IMP_car_age, kde=False)
ax.set_title("Histogram of IMP_car_age")
plt.show()

#Get outliers
print(get_outliers(df.IMP_car_age))

#Plots of IMP_home_val
ax = sns.boxplot(df.IMP_home_val, orient='v')
ax.set_title("Boxplot of IMP_home_val")
ax.set_ylabel("IMP_home_val")
plt.show()

ax = sns.distplot(df.IMP_home_val, kde=False)
ax.set_title("Histogram of IMP_home_val")
plt.show()

#Get outliers
print(get_outliers(df.IMP_home_val))
#Fix outliers

#Plots of IMP_income
ax = sns.boxplot(df.IMP_income, orient='v')
ax.set_title("Boxplot of IMP_income")
ax.set_ylabel("IMP_income")
plt.show()

ax = sns.distplot(df.IMP_income, kde=False)
ax.set_title("IMP_income")
plt.show()

#Get outliers
print(get_outliers(df.IMP_income))
#fix outliers

#Plots of IMP_yoj
ax = sns.boxplot(df.IMP_yoj, orient='v')
ax.set_title("Boxplot of IMP_yoj")
ax.set_ylabel("IMP_yoj")
plt.show()

ax = sns.distplot(df.IMP_yoj, kde=False)
ax.set_title("IMP_yoj")
plt.show()

#Get outliers
print(get_outliers(df.IMP_yoj))

##########################################################
def fixOutliers(onePdSer, otherPdSer, q1, q2):
    newpdSer = onePdSer.copy()
    for val in range(0, len(onePdSer)):
        if onePdSer[val] > otherPdSer.quantile(q1):
            newpdSer[val] = otherPdSer.quantile(q1)

        if onePdSer[val] < otherPdSer.quantile(q2):
            newpdSer[val] = otherPdSer.quantile(q2)

    return newpdSer
```

```
#truncate outliers
df.target_amt = fixOutliers(df.target_amt, df.target_amt, .99, .01)
df.IMP_income = fixOutliers(df.IMP_income, df.IMP_income, .99, .01)
df.IMP_home_val = fixOutliers(df.IMP_home_val, df.IMP_home_val, .99, .01)
df.bluebook = fixOutliers(df.bluebook, df.bluebook, .99, .01)
df.travtime = fixOutliers(df.travtime, df.travtime, .99, .01)

#Get dummies
cols_to_dum = [ 'IMP_red_car', 'IMP_urbanicity', 'IMP_revoked',
                 'IMP_sex','IMP_mstatus','IMP_car_use', 'IMP_parent1',
                 "home_own", "university_degree", "white_collar",
             'kidsdriv', 'homekids', "hadOldClaim", "hadFewOldClaim", "hadManyOldClaim"]

df["IMP_mstatus"] = np.where(df["mstatus"].isin(["Yes"]), 1, 0)
df["IMP_parent1"] = np.where(df["parent1"].isin(["Yes"]), 1, 0)
df["IMP_sex"] = np.where(df["sex"].isin(["z_F"]), 1, 0)
df["IMP_revoked"] = np.where(df["revoked"].isin(["Yes"]), 1, 0)
df["IMP_red_car"] = np.where(df["red_car"].isin(["yes"]), 1, 0)
df["IMP_urbanicity"] = np.where(df["urbanicity"].isin(["Highly Urban/ Urban"]), 1, 0)
df["IMP_car_use"] = np.where(df["car_use"].isin(["Commercial"]), 1, 0)

#continuous vars correlation
df[contVars].corrwith(df.target_flag)

#categorical vars correlation
df[cols_to_dum].corrwith(df.target_flag)

#only drop yoj and red_car due to essentially 0 correlation
potentCols = ["target_flag", "target_amt",
 'bluebook',
 'kidsdriv',
 'homekids',
 'oldclaim',
 'clm_freq',
 'mvr_pts',
 'tif',
 'travtime',
 'IMP_sex',
 'IMP_age',
 'IMP_car_age',
 'IMP_home_val',
 'IMP_income', 'IMP_urbanicity', 'IMP_revoked',
 'IMP_mstatus','IMP_car_use', 'IMP_parent1',
 "home_own", "university_degree", "white_collar", "car_type",
 "hadOldClaim", "hadFewOldClaim", "hadManyOldClaim"]

df_sub = df[potentCols]

#Collinearity Check
corrs = df_sub.corr()
print(corrs)

#Get heatmap
ax = plt.axes()
sns.heatmap(corrs,
            xticklabels=corrs.columns.values,
            yticklabels=corrs.columns.values, ax = ax)

ax.set_title('Insurance Correlation Heatmap')
plt.show()
```

```
#sns.pairplot(df_sub)

####################################################
features = "+".join(df_sub.columns[2:27])
logit1 = smf.logit('target_flag ~ ' + features, data=df_sub).fit()
logit1.summary()
logit1.summary2()

features2 =  "clm_freq+mvr_pts"
logit12 = smf.logit('target_flag ~ ' + features2, data=df_sub).fit()
logit12.summary()
logit12.summary2()

features3 =  "clm_freq+mvr_pts+IMP_urbanicity"
logit3 = smf.logit('target_flag ~ ' + features3, data=df_sub).fit()
logit3.summary()
logit3.summary2()

######REGRESSION ONE######################
features4 =  "clm_freq+mvr_pts+IMP_urbanicity+IMP_revoked+IMP_parent1"
logit4 = smf.logit('target_flag ~ ' + features4, data=df_sub).fit()
logit4.summary()
logit4.summary2()

preds = logit4.predict()
fpr, tpr, _ = metrics.roc_curve(df_sub['target_flag'], preds)

# calculate AUC and create ROC curve
roc_auc = metrics.auc(fpr,tpr)
print(roc_auc)

plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()

############################################################
#Gets the columns without collinearity
def getBestCols_LOGIT(inpDF, inpYString, thresh = 3):
    tempList = list(inpDF.columns.values)
    tempList.remove(inpYString)
    inpFeatures = "+".join(tempList)
    y, X = dmatrices(inpYString + " ~ " + inpFeatures, inpDF, return_type='dataframe')
    # For each X, calculate VIF and save in dataframe
    vif = pd.DataFrame()
    vif["VIF Factor"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
    vif["Features"] = X.columns
    #check if collinearity is below thresh limit
    isDone = all(n < thresh for n in vif[vif.Features != "Intercept"]["VIF Factor"])
    if isDone:
        return vif.Features
    else:
        #Get the Features with the top 2 VIF
        vif_sub = vif[vif.Features != "Intercept"]
        colToTest1 = vif_sub[vif_sub["VIF Factor"] == max(vif_sub["VIF 
Factor"])]["Features"].values[0]
        vif_sub2 = vif_sub[vif_sub.Features != colToTest1]
        colToTest2 = vif_sub2[vif_sub2["VIF Factor"] == max(vif_sub2["VIF 
Factor"])]["Features"].values[0]

        #Drop one and create model
        testDF1 = inpDF.drop(colToTest1, 1)
        tempList1 = list(testDF1.columns.values)
        tempList1.remove(inpYString)
        testFeatures1 = "+".join(tempList1)
        firstModel = smf.logit(inpYString + " ~ " + testFeatures1, data=testDF1).fit()

        #Drop other and create model
        testDF2 = inpDF.drop(colToTest2, 1)
        tempList2 = list(testDF2.columns.values)
        tempList2.remove(inpYString)
        testFeatures2 = "+".join(tempList2)
        secondModel = smf.logit(inpYString + " ~ " + testFeatures2, data=testDF2).fit()

        #Pick better model and recurse
        if firstModel.prsquared > secondModel.prsquared:
            return getBestCols_LOGIT(testDF1, inpYString)
        elif secondModel.prsquared > firstModel.prsquared:
            return getBestCols_LOGIT(testDF2, inpYString)
        else:
            return getBestCols_LOGIT(testDF1, inpYString)

##################################
df_sub2 = df_sub.copy()
del(df_sub2["target_amt"])
del(df_sub2["car_type"])

theBestCols = getBestCols_LOGIT(df_sub2, "target_flag").values
index = np.argwhere(theBestCols=="Intercept")
theBestCols = np.delete(theBestCols, index)
theBestCols = np.append(theBestCols, "target_flag")
df_sub_VIF = df_sub2[theBestCols]

#Double check the results of the above function
tempList = list(df_sub_VIF.columns.values)
tempList.remove("target_flag")
testFeatures = "+".join(tempList)
y, X = dmatrices('target_flag ~' + testFeatures, df_sub_VIF, return_type='dataframe')
# For each X, calculate VIF and save in dataframe
vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif["Features"] = X.columns

print(vif)

#Looks good!
##################################
df_sub_VIF["car_type"] = df_sub["car_type"]
tempList = list(df_sub_VIF.columns.values)
tempList.remove("target_flag")
testFeatures = "+".join(tempList)

###########REFGRESSION 2###########################333#
logit5 = smf.logit('target_flag ~ ' + testFeatures, data=df_sub_VIF).fit()
```

```
logit5.summary()
logit5.summary2()

preds = logit5.predict()
fpr, tpr, _ = metrics.roc_curve(df_sub['target_flag'], preds)

# calculate AUC and create ROC curve
roc_auc = metrics.auc(fpr,tpr)
print(roc_auc)

plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()

logit5.params

###########################REGRESSION 3#########################3
df_sub3 = df_sub_VIF.copy()
del(df_sub3["IMP_sex"])
del(df_sub3["IMP_age"])
del(df_sub3["IMP_car_age"])
del(df_sub3["IMP_income"])
del(df_sub3["homekids"])
del(df_sub3["hadManyOldClaim"])

tempList = list(df_sub3.columns.values)
tempList.remove("target_flag")
testFeatures = "+".join(tempList)
logit6 = smf.logit('target_flag ~ ' + testFeatures, data=df_sub3).fit()
logit6.summary()
logit6.summary2()

logit6.params

preds = logit6.predict()
fpr, tpr, _ = metrics.roc_curve(df_sub['target_flag'], preds)

# calculate AUC and create ROC curve
roc_auc = metrics.auc(fpr,tpr)
print(roc_auc)

plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()

#The coefficients for bluebook, home_val, and oldclaim are tiny

##############################REGRESSION 4###############
df_sub3["log_bluebook"] = np.log(df_sub3.bluebook)
df_sub3["log_IMP_home_val"] = np.log(df_sub3.IMP_home_val)
#df_sub3["log_IMP_income"] = np.log(df_sub3.IMP_income)

tempList = list(df_sub3.columns.values)
tempList.remove("target_flag")
tempList.remove("bluebook")
tempList.remove("IMP_home_val")

testFeatures = "+".join(tempList)
logit7 = smf.logit('target_flag ~ ' + testFeatures, data=df_sub3).fit()
logit7.summary()
logit7.summary2()

preds = logit7.predict()

fpr, tpr, _ = metrics.roc_curve(df_sub3['target_flag'], preds)

# calculate AUC and create ROC curve
roc_auc = metrics.auc(fpr,tpr)
print(roc_auc)

plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()

#Finally for the Hosmer-Lemeshow Lack of Fit
pred_train = preds
pred_out = df_sub3.loc[:,['index','target_flag']]
pred_out['p_target_flag'] = pred_train[: ]
pred_out.head
#sort by pred
result1 = pred_out.sort_values(by=('p_target_flag'))
print (result1)

#rank by decile
result2 = pd.qcut(result1['p_target_flag'], 10, labels=[1,2,3,4,5,6,7,8,9,10])
print (result2)
d1 = {'g' : result2}
df3 = pd.DataFrame(data=d1)
print (df3)
result3 = pd.concat([result1, df3], axis=1, join_axes=[result1.index])
print (result3)

sums = result3.groupby('g')
sums1 = sums.aggregate(np.sum)
print (sums1)
```

```
#plot target_flag vs pred
def scat(dataframe,var1,var2,var3):
    dataframe[var2].plot()
    dataframe[var3].plot()
    plt.title('Hosmer-Lemeshow lack of fit trng data')
    plt.xlabel(var1)
    plt.ylabel('Sum by Group')

scat(sums1, 'g', 'target_flag', 'p_target_flag')

#########################################################

df2 = df[df.target_flag == 1]

#continuous vars correlation
df2[contVars].corrwith(df2.target_amt)

#categorical vars correlation
df2[cols_to_dum].corrwith(df2.target_amt)

df2["log_bluebook"] = np.log(df2.bluebook)

features2 = "log_bluebook+IMP_mstatus+mvr_pts"
result2 = smf.ols('target_amt ~' + features2, data=df2).fit()
print(result2.summary())

y = df2.target_amt

#Get predictions and residuals
x2_predictions = result2.predict()
x2_residuals = y - x2_predictions

#Plots
#Residuals vs Fitted
fig = plt.figure()
plt.scatter(x=x2_predictions, y= x2_residuals)
fig.suptitle('Cost Model Residuals vs Fitted Target Cost')
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.show()

#QQPlot
stats.probplot(x2_residuals, dist="norm", plot=plt)
plt.show()

print(metrics.mean_absolute_error(y, x2_predictions))


###########################################################################
#Do all of this to the test set
test1 = test.copy()
#del(test1["target_amt"])
#del(test1["target_flag"])
testCols = list(test1.columns.values)

#One of the columns with NaN is categorical. Replace with "No Job Info"
test1.job = test1.job.fillna("No Job Info")
naColsTest = test1.isnull().any()

#there are dollar signs and commas in some of these variables
```

```
#convert to int
for indx in testCols:
    if test1[indx].astype(str).str.contains("\$").any() == True:
        tempVals = []
        for money in test1[indx].values:
            if not pd.isnull(money):
                value = int(Decimal(sub(r'[^\d\-.]', '', money)))
                tempVals.append(value)
            else:
                tempVals.append(np.nan)
        test1[indx] = tempVals

#Create home own variable. 1 if home val is greater than 0 and not NaN
test1["home_own"] = np.where(test1["home_val"].fillna(0) > 0, 1, 0)

#start by converting all NaN values to 0
test1=test1.fillna(0)

#convert all NaN values to median
for indx in testCols:
    if naColsTest[indx] == True:
        newName = "IMP_" + indx
        newInd = "m_" + indx
        m = np.median(train1[indx][train1[indx] > 0])
        test1[newName] = test1[indx].replace({0: m}).astype(int)
        test1[newInd] = (test1[indx] == 0).astype(int)

#need to fix high School
test1.loc[test1.education == "<High School", 'education'] = "z_High School"
test1.education.value_counts()

#truncate outliers
test1.IMP_income = fixOutliers(test1.IMP_income, train1.IMP_income, .99, .01)
test1.IMP_home_val = fixOutliers(test1.IMP_home_val, train1.IMP_home_val, .99, .01)
test1.bluebook = fixOutliers(test1.bluebook, df.bluebook, .99, .01)
test1.travtime = fixOutliers(test1.travtime, df.travtime, .99, .01)

test1["white_collar"] = np.where(test1["job"].isin(["Doctor", "Manager",
       "Lawyer"]), 1, 0)
test1["university_degree"] = np.where(test1["education"].isin(["Bachelors",
       "Masters", "PhD"]), 1, 0)

#Get new variables
test1["hadOldClaim"] = np.where(test1["oldclaim"] > 0, 1, 0)
test1["hadFewOldClaim"] = np.where((test1["oldclaim"] > 0) & (test1["oldclaim"] <= 12000), 1,
0)
test1["hadManyOldClaim"] = np.where(test1["oldclaim"] > 12000, 1, 0)

test1["IMP_mstatus"] = np.where(test1["mstatus"].isin(["Yes"]), 1, 0)
test1["IMP_parent1"] = np.where(test1["parent1"].isin(["Yes"]), 1, 0)
test1["IMP_sex"] = np.where(test1["sex"].isin(["z_F"]), 1, 0)
test1["IMP_revoked"] = np.where(test1["revoked"].isin(["Yes"]), 1, 0)
test1["IMP_red_car"] = np.where(test1["red_car"].isin(["yes"]), 1, 0)
test1["IMP_urbanicity"] = np.where(test1["urbanicity"].isin(["Highly Urban/ Urban"]), 1, 0)
test1["IMP_car_use"] = np.where(test1["car_use"].isin(["Commercial"]), 1, 0)

test1["log_bluebook"] = np.log(test1.bluebook)
test1["log_IMP_home_val"] = np.log(test1.IMP_home_val)

#use this model to predict for the test data
preds = logit7.predict(test1)
```

```
preds.head
pred_out = test1.loc[:,['index','target_flag']]
pred_out['p_target_flag'] = preds[: ]
pred_out.head

preds2 = result2.predict(test1)
pred_out['p_target_amt'] = preds2[: ]
pred_out.head
#watch your record count should be 2141
your_model = pred_out.loc[:,['index','p_target_flag','p_target_amt']]
your_model.head
your_model.to_csv('booth_alexander_hw02_predictions_final.csv')
```