DATA EXPLORATION AND DATA PREPARATION

Originally, this data set includes 194673 records and 38 variables, which contains categorical variables, such as ADDRTYOE, and numerical variables, such as ROADCOND. There are some problems in this data set which need to be fixed. These problems are: missing values, duplicate variables in different formats, conflict formats in the same variable and imbalanced data. Besides, to apply the data to the models, other steps like data transformation and resampling methods are also necessary. The details of each problem and the corresponding way to fix the problem are followed.

1) ***Removing Variables with a Large Number of Missing Value***

   As shown in ***Appendix 1***, there are 19 variables (out of 38) with missing values. There are 7 variables have a large number of missing values (at least 40%) out of the total 194673 records. These variables are: INTKEY, EXCEPTRSNCODE, EXCEPTRSNDESC, INATTENTIONIND, PEDROWNOTGRNT, PEEDING, and SDOTCOLNUM. Since variables with a large number of missing values can't provide us enough information, they will be removed from the data for the further use. As a result, all the remaining variables have missing values less than 3.5%.

2) ***Excluding Variables with Same Values in Different Formats***

   After reviewing the data set, there are 5 pairs of variables with the same values but in different format: SEVERITYCODE & SEVERITYDESC, INCDATE & INCDTTM, SDOT_COLCODE & SDOT_COLDESC, and ST_COLCODE & ST_COLDESC. Hence, only one variable of each pair is kept in the data set.

Besides, SEVERITYCODE has existed for twice in this data set, so one of them will be removed.

3) *Deleting Unnecessary Variables*

Since the value of some variables are not relevant to the value of the severity that we want to predict, five variables are excluded: X, Y, LOCATION, SDOT_COLCODE, INCDATE and ST_COLCODE. Other variables, such as the dummy variable created for ADDRTYOE=Alley, are deleted based on t-tests, which will be explained later in the data analysis part.

4) *Change Values to the Same Format*

As shown below, the values of UNDERINFL used two formats: 1/0 and Y/N. According to the understanding of data, these two formats are indicating the same values. These values will be changed to use the same format - 1 for Y and 0 for N.

UNDERINFL	
0	80394
1	3995
N	100274
Y	5126

Picture 1 - Conflict Formats for Variable UNDERINFL

5) *Data Transformation*

To apply the values from the data to the model, there are some necessary transformation steps needed to be done before training the models. Transforming Binary Variable to 1/0 for variables - SEVERITYCODE, STATUS, and HITPARKEDCAR, is necessary since models like logistic regression can only compute numeric values. Similarly, categorical variables - ADDRTYPE, COLLISIONTYPE, JUNCTIONTYPE, WEATHER, ROADCOND, LIGHTCOND, that have more than two categories should be transformed to

dummy variables from n categories to n dummy variables.

6) *Removing Missing Values*

As the result shown in the ***Appendix 1***, there are some missing values would still exist in the data set even we have removed those variables with a large proportion of missing data. This would impact the result that should be removed as well.

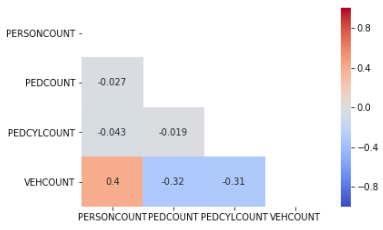
7) *Imbalanced Data*

In this dataset, there are 132630 records with severity code as 1 while there are only 57159 records with severity code as 2, about 30% of the total number. When the distribution of examples across the known classes is biased, it would be an unequal distribution of classes, which would impact the performance of the model. To keep as much information as possible, upsampling is used for logistic regression and decision tree. After upsampling, there are 265260 records used to train the models. Due to computability, downsampling is used for k-nearest neighbors. After downsampling, there are 114318 records used to train the models.

DATA ANALYSIS

1) *Correlation for Numeric Data*

As we can see from the correlation heat map below, the correlations of the numeric variables are very low, which means, there is no collinearity problem here.

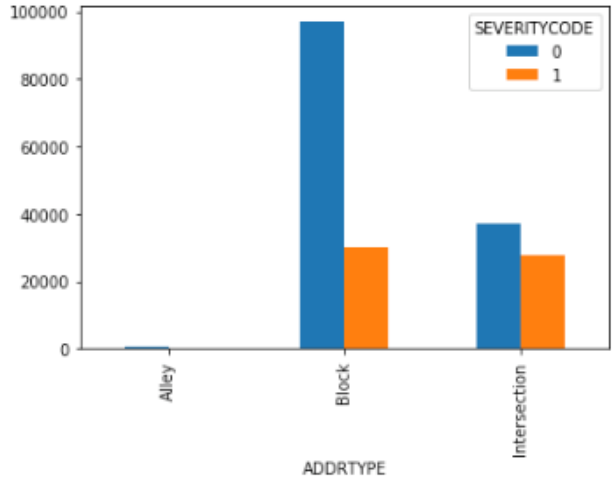


Picture 2 - Correlation Heat Map

2) *Bar Charts and ANOVA Tests for Categorical Data*

As shown in the bar chart and the result of the ANOVA tests, we can find that, in terms of severity, the impact of different address types has a significant difference.

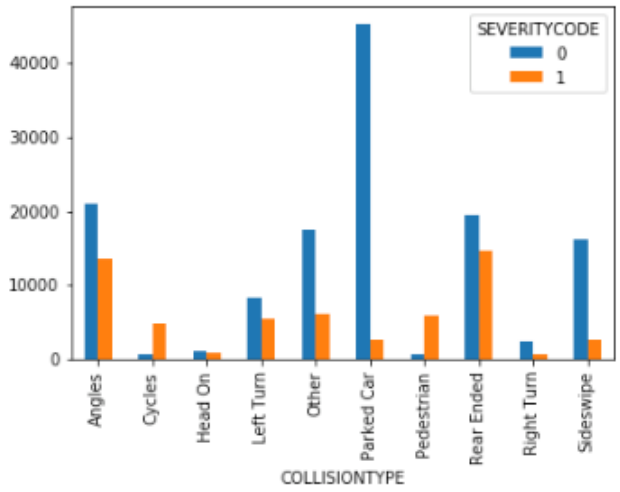
```
F_onewayResult(statistic=3530.8078377494403, pvalue=0.0)
```



Picture 3 - Number of Records of Severity Code 1&2 by ADDTYPE

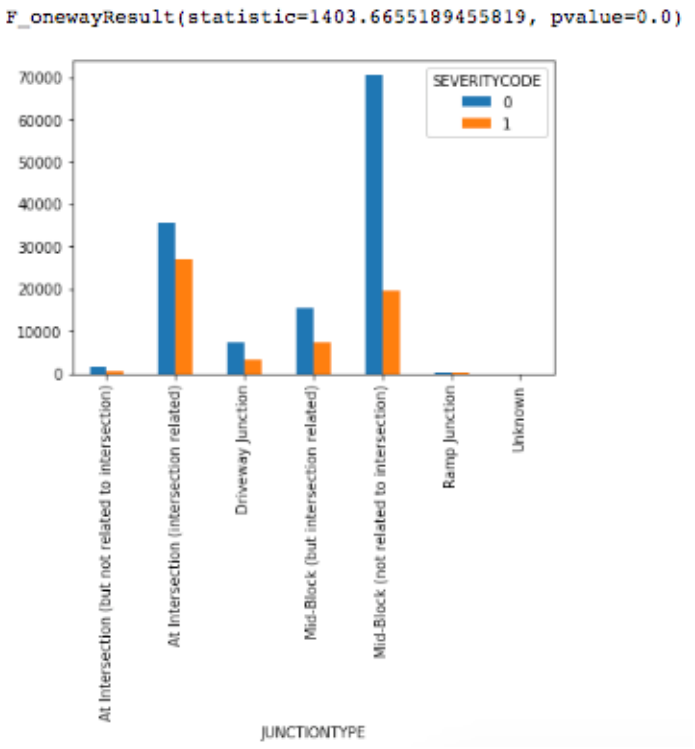
As shown in the bar chart and the result of the ANOVA tests, we can find that, in terms of severity, the impact of different collision types has a significant difference.

```
F_onewayResult(statistic=5369.001306448202, pvalue=0.0)
```



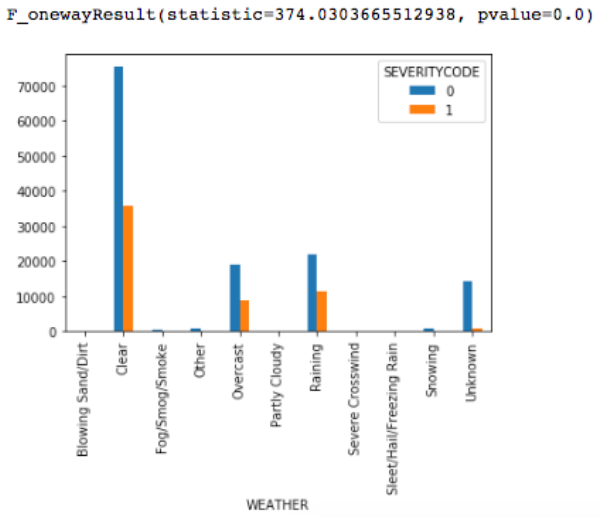
Picture 4 - Number of Records of Severity Code 1&2 by COLLISIONTYOE

As shown in the bar chart and the result of the ANOVA tests, we can find that, the impact of different categories of junction has a significant difference.



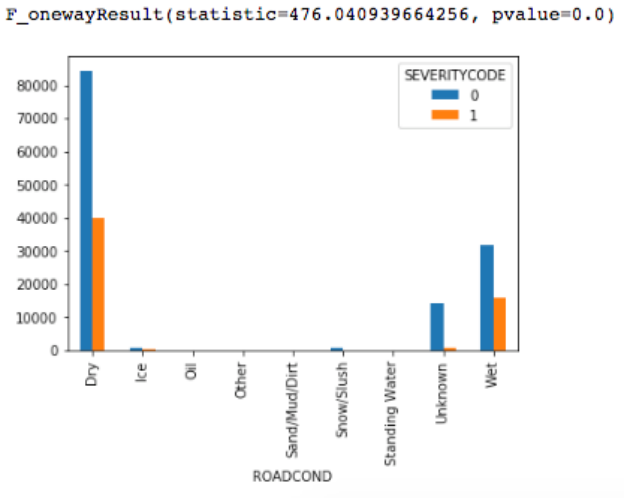
Picture 5 - Number of Records of Severity Code 1&2 by JUNCTIONTYPE

As shown in the bar chart and the result of the ANOVA tests, we can find that, in terms of severity, the impact of different weather types has a significant difference.



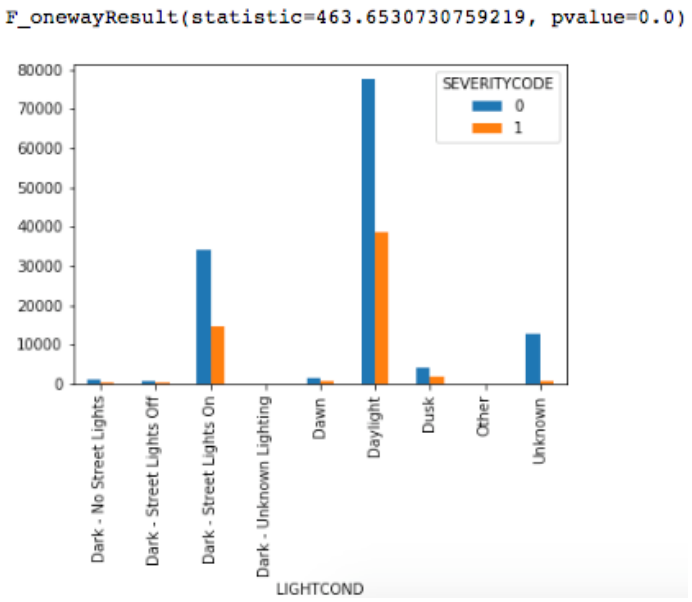
Picture 6 - Number of Records of Severity Code 1&2 by WEATHER

As shown in the bar chart and the result of the ANOVA tests, we can find that, in terms of severity, the impact of different conditions of the road during the collision has a significant difference.



Picture 7 - Number of Records of Severity Code 1&2 by ROADCOND

As shown in the bar chart and the result of the ANOVA tests, we can find that, in terms of severity, the impact of different conditions of the light during the collision has a significant difference.



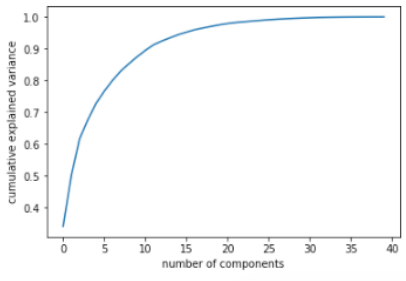
Picture 8 - Number of Records of Severity Code 1&2 by LIGHTCOND

3) *T-test*

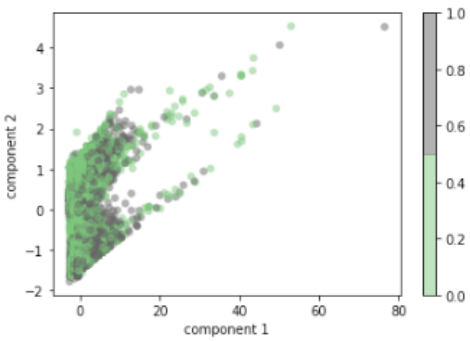
As an example of the result of t-test shown in ***Appendix 2***, we can notice that no matter which status it is recorded, statistically speaking, there is no significant difference on the value of severity code between the two groups of status. Similarity, we can notice that other variables, for example, left turn collision type, ramp junction and dawn weather.

4) *Principal Component Analysis*

As the result shown in ***Appendix 3***, there are three principal components that have a relatively high explained variance, which are chosen to use when building the models. However, we can also notice that, as shown in the plot below, with only 3 principal components, the cumulative explained variance is relatively low - around 70% of total. This shortage is also reflected in ***Picture 10***.



Picture 9 - Number of Principal Components vs. Cumulative Explained Variance



Picture 10 - Number of Principal Components vs. Cumulative Explained Variance

APPENDICES

Appendix 1 - Missing Values (Part)

```
df.isnull().sum()
```

```
SEVERITYCODE           0
X                   5334
Y                   5334
OBJECTID               0
INCKEY                 0
COLDETKEY              0
REPORTNO               0
STATUS                 0
ADDRTYPE            1926
INTKEY            129603
LOCATION            2677
EXCEPTRSNCODE     109862
EXCEPTRSNDESC     189035
SEVERITYCODE.1         0
SEVERITYDESC           0
COLLISIONTYPE       4904
PERSONCOUNT            0
```

Appendix 2 - Result of T-test

```
results = rp.ttest(group1= df_trans['SEVERITYCODE'][df_trans.iloc[:,1] == 1], group1_name= "1",
                   group2= df_trans['SEVERITYCODE'][df_trans.iloc[:,1] == 0], group2_name= "0")
results[1].iloc[3,1]
```

```
/opt/conda/envs/Python36/lib/python3.6/site-packages/scipy/stats/_distn_infrastructure.py:1920:
multiply
  lower_bound = self.a * scale + loc
/opt/conda/envs/Python36/lib/python3.6/site-packages/scipy/stats/_distn_infrastructure.py:1921:
multiply
  upper_bound = self.b * scale + loc
```

```
0.2555
```

Appendix 3 - Result of Principal Component Analysis

```
pca = PCA(n_components=3)
pca.fit(df_trans[df_trans.columns[1:42]])
print(pca.explained_variance_)
```

```
[1.91661579 0.89834652 0.64487807]
```

```
projected = pca.fit_transform(df_trans[df_trans.columns[1:42]])
print(df_trans[df_trans.columns[1:42]].shape)
print(projected.shape)
```

```
(189789, 40)
(189789, 3)
```