Capstone Project - Car accident severity (Week 2)

1. Introduction:

When there is an accident happening on the road, a terrible traffic jam may be caused, which influences other people's plans. It would be great if there is something in place that could warn people, given the weather and the road conditions about the possibility of people getting into a car accident and how severe it would be. So people would drive more carefully or even change their travels if they are able to. In this project, we will be working on a case study which is to predict the severity of an accident. We will try to use the accident data in Seattle City that include many influence factors of collisions for the prediction of the severity of an accident.

1.1. Problems:

To improve the safely driving level of all the car users in Seattle City, we will try to find the answers to the following problems:

Q1) What are the main factors that influence the severity of an accident? How to list and visualize them?

Q2) How to use basic machine learning technologies to build a model to predict the severity of an accident?

2. Data Information

All collisions are provided by SPD and recorded by Traffic Records. This includes all types of collisions happening in Seattle City, which was updated weekly. Collisions will display at the intersection or mid-block of a segment.

Timeframe: 2004 to Present.

Contact Person: SDOT GIS Analyst

Contact Organization: SDOT Traffic Management Division, Traffic Records Group

Contact email: DOT_IT_GIS@seattle.gov

2.1. Data discussion:

We have the following discussion about the data:

1. The dataset includes 37 attributes. Some of them are much related to the severity of an accident, such as weather condition, car speeding and light conditions. Note that not all the attributes are useful.
2. There are many empty inputs in the accident data. For instance, the column "ROADCOND" includes 5012 empty inputs. There also exist both numerical and categorical types of data, which leads to the utilization of feature engineering.
3. The dataset has labels so that supervised machine learning technologies are suitable to predict the severity of an accident; nevertheless, the dataset is not label balanced. We should balance the data firstly before we create a model.

3. Methodology

In this part, we mainly utilize logistic regression to classify the data for two groups according to different severity levels. The reason why we choose this method is that logistic regression has the advantage to classify data, especially for data belonging to two categories. Compared with SVM, logistic regression is much faster.

Firstly, let us do some Exploratory Data Analysis. The part of data can be shown as follows:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	REPORTNO	STATUS	ADDRTYPE	INTKEY	...	ROADCOND	LIGHTCOND	PEDROWNOTGRNT
0	2	-122.323148	47.703140	1	1307	1307	3502005	Matched	Intersection	37475.0	...	Wet	Daylight	NaN
1	1	-122.347294	47.647172	2	52200	52200	2607959	Matched	Block	NaN	...	Wet	Dark - Street Lights On	NaN
2	1	-122.334540	47.607871	3	26700	26700	1482393	Matched	Block	NaN	...	Dry	Daylight	NaN
3	1	-122.334803	47.604803	4	1144	1144	3503937	Matched	Block	NaN	...	Dry	Daylight	NaN
4	2	-122.306426	47.545739	5	17700	17700	1807429	Matched	Intersection	34387.0	...	Wet	Daylight	NaN

5 rows × 38 columns

We can clearly see the dataset includes 38 columns (37 attributes). Note that some columns are categorical data, which leads to the need of feature engineering. We can also use the 'describe' method to show some features of the data, which can be seen hereafter:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	INTKEY	SEVERITYCODE.1	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	SDOT_COLCODE	SDOTCOLNUM	SEGLANEKEY	CROSSWALKKEY
count	194673.000000	189339.000000	189339.000000	194673.000000	194673.000000	194673.000000	65070.000000	194673.000000	194673.000000	194673.000000	194673.000000	194673.000000	194673.000000	1.149360e+05	194673.000000	1.946730e+05
mean	1.298901	-122.330518	47.619543	108479.364930	141091.456350	141298.811381	37558.450576	1.298901	2.444427	0.037139	0.028391	1.920780	13.867768	7.972521e+06	269.401114	9.782452e+03
std	0.457778	0.029976	0.056157	62649.722558	86634.402737	86986.542110	51745.990273	0.457778	1.345929	0.198150	0.167413	0.631047	6.868755	2.553533e+06	3315.776055	7.226926e+04
min	1.000000	-122.419091	47.495573	1.000000	1001.000000	1001.000000	23807.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.007024e+06	0.000000	0.000000e+00
25%	1.000000	-122.348673	47.575956	54267.000000	70383.000000	70383.000000	28667.000000	1.000000	2.000000	0.000000	0.000000	2.000000	11.000000	6.040015e+06	0.000000	0.000000e+00
50%	1.000000	-122.330224	47.615369	106912.000000	123363.000000	123363.000000	29973.000000	1.000000	2.000000	0.000000	0.000000	2.000000	13.000000	8.023022e+06	0.000000	0.000000e+00
75%	2.000000	-122.311937	47.663664	162272.000000	203319.000000	203459.000000	33973.000000	2.000000	3.000000	0.000000	0.000000	2.000000	14.000000	1.015501e+07	0.000000	0.000000e+00
max	2.000000	-122.238949	47.734142	219547.000000	331454.000000	332954.000000	757580.000000	2.000000	81.000000	6.000000	2.000000	12.000000	69.000000	1.307202e+07	525241.000000	5.239700e+06

Note that the above figure only includes the numerical data and some columns are useless for our prediction model.

We also tried to do a correlation analysis and the results can be seen as follows:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	INTKEY	SEVERITYCODE.1	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	S
SEVERITYCODE	1.000000	0.010309	0.017737	0.020131	0.022065	0.022079	0.006553	1.000000	0.130949	0.246338	0.214218	-0.054686	
X	0.010309	1.000000	-0.160262	0.009956	0.010309	0.010300	0.120754	0.010309	0.012887	0.011304	-0.001752	-0.012168	
Y	0.017737	-0.160262	1.000000	-0.023848	-0.027396	-0.027415	-0.114935	0.017737	-0.013850	0.010178	0.026304	0.017058	
OBJECTID	0.020131	0.009956	-0.023848	1.000000	0.946383	0.945837	0.046929	0.020131	-0.062333	0.024604	0.034432	-0.094280	
INCKEY	0.022065	0.010309	-0.027396	0.946383	1.000000	0.999996	0.048524	0.022065	-0.061500	0.024918	0.031342	-0.107528	
COLDETKEY	0.022079	0.010300	-0.027415	0.945837	0.999996	1.000000	0.048499	0.022079	-0.061403	0.024914	0.031296	-0.107598	
INTKEY	0.006553	0.120754	-0.114935	0.046929	0.048524	0.048499	1.000000	0.006553	0.001886	-0.004784	0.000531	-0.012929	
SEVERITYCODE.1	1.000000	0.010309	0.017737	0.020131	0.022065	0.022079	0.006553	1.000000	0.130949	0.246338	0.214218	-0.054686	
PERSONCOUNT	0.130949	0.012887	-0.013850	-0.062333	-0.061500	-0.061403	0.001886	0.130949	1.000000	-0.023464	-0.038809	0.380523	
PEDCOUNT	0.246338	0.011304	0.010178	0.024604	0.024918	0.024914	-0.004784	0.246338	-0.023464	1.000000	-0.016920	-0.261285	
PEDCYLCOUNT	0.214218	-0.001752	0.026304	0.034432	0.031342	0.031296	0.000531	0.214218	-0.038809	-0.016920	1.000000	-0.253773	
VEHCOUNT	-0.054686	-0.012168	0.017058	-0.094280	-0.107528	-0.107598	-0.012929	-0.054686	0.380523	-0.261285	-0.253773	1.000000	
SDOT_COLCODE	0.188905	0.010904	-0.019694	-0.037094	-0.027617	-0.027461	0.007114	0.188905	-0.128960	0.260393	0.382521	-0.365814	
SDOTCOLNUM	0.004226	-0.001016	-0.006958	0.969276	0.990571	0.990571	0.032604	0.004226	0.011784	0.021461	0.034593	-0.023813	

Regarding the feature engineering part, we selected 12 columns of data as model variables, which can be found hereafter: X, Y, ADDRTYPE, PERSONCOUNT, VEHCOUNT, PEDCOUNT, PEDCYLCOUNT, WEATHER, ROADCOND, LIGHTCOND, SPEEDING, PEDROWNOTGRNT. Note that the chosen of variables is based on life experience as well as the result of the correlation analysis.

As for 'NaN' values, we utilized the following code to remove them:

```
#r emove rows containing NaN
new = new.dropna(axis=0, how='any')
new.head(20)
```

As for the categorical data, we utilized the method "pd.get_dummies" to generate numerical variables. The detailed process can be found in the following code:

```
dummy_variable_1 = pd.get_dummies(new["ADDRTYPE"])

dummy_variable_2 = pd.get_dummies(new["WEATHER"])

dummy_variable_3 = pd.get_dummies(new["ROADCOND"])

dummy_variable_4 = pd.get_dummies(new["LIGHTCOND"])

dummy_variable_5 = pd.get_dummies(new["SPEEDING"])

dummy_variable_6 = pd.get_dummies(new["PEDROWNOTGRNT"])
```

```
# merge data frame "df" and "dummy_variable_1"
new_final = pd.concat([new['X'],new['Y'],new['PERSONCOUNT'], new['VEHCOUNT'],new['PEDCOUNT'], new['PEDCYLCOUNT'],dummy_variable_1, dummy_variable_2, dummy_variable_3, dummy_variable_4, dummy_variable_5, dummy_variable_6, n
# new_final=new
new_final.head()
```

	X	Y	PERSONCOUNT	VEHCOUNT	PEDCOUNT	SDOT_COLCODE	CROSSWALKKEY	PEDCYLCOUNT	SEGLANEKEY	Block	...	Dawn	Daylight	Dusk	Other	Unknown	NO	Y	NO	Y	SEVERITYCODE
0	-122.323148	47.703140	2	2	0	11	0	0	0	0	...	0	1	0	0	0	1	0	1	0	2
1	-122.347294	47.647172	2	2	0	16	0	0	0	1	...	0	0	0	0	0	1	0	1	0	1
2	-122.334540	47.607871	4	3	0	14	0	0	0	1	...	0	1	0	0	0	1	0	1	0	1
3	-122.334803	47.604803	3	3	0	11	0	0	0	1	...	0	1	0	0	0	1	0	1	0	1
4	-122.306426	47.545739	2	2	0	11	0	0	0	0	...	0	1	0	0	0	1	0	1	0	2

5 rows × 45 columns

Another important issue is that the data is unbalanced. To fix this outstanding problem, we utilized the following code to create balanced dataset:

```
from imblearn.over_sampling import RandomOverSampler

from imblearn.under_sampling import RandomUnderSampler

rus=RandomUnderSampler(random_state=0,replacement=True)

X = np.asarray(new_final.iloc[:,0:-1])
y = np.asarray(new_final['SEVERITYCODE'])

X_r, y_r = rus.fit_sample(X, y)
```

Then, the data normalization part can be seen as follows:

```
from sklearn import preprocessing
X_r = preprocessing.StandardScaler().fit(X_r).transform(X_r)
X_r[0:5]
```

Last but not least, the logistic model can be seen as follows:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X_r, y_r, test_size=0.2, random_state=4)
print ('Train set:', X_train.shape,  y_train.shape)
print ('Test set:', X_test.shape,  y_test.shape)
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
LR = LogisticRegression(C=0.01, solver='liblinear').fit(X_train,y_train)
LR
```

4. Results

We utilized ‘f1_score’ to evaluate the accuracy of our model. The code can be seen as follows:

```
from sklearn.metrics import f1_score
yhat = LR.predict(X_test)
yhat
f1_score(y_test, yhat, average='weighted')
```

```
Out[111]: 0.6633856705446096
```

The final accuracy is 66.34%, which is actually not a very good result. However, according to our testing and our understanding of the dataset, we believe the accuracy is acceptable for this certain situation.

5. Discussion section

According to the results, we have the following recommendations:

The dataset is not very suitable for accurate prediction, because of many columns of categorical data and large amount of useless data. According to the results of correlation analysis, it is east to find there are not many columns relating to the severity of accidents. We believe more related data need to be collected for accurate prediction.

Normal machine learning techniques are not very suitable to this problem, even though the final results are acceptable. We have tried several methods to solve this problem such as SVM and decision tree; nevertheless, the results are basically same. We believe that neural networks can be utilized to deal with this problem and obtain higher accuracy level.

6. Conclusions

Through this capstone project, we have reviewed how to pre-processing data, how to do feature engineering and how to build a prediction model using basic machine learning techniques. In this case, we utilized logistic regression method to build a model to predict the severity of an accident via 12 variables and the final result is acceptable. Additionally, we also find the stage of data-processing is very important for the whole case study. Accurate data selection also directly determine the accuracy of our final model. In the future, we may try to utilize neural networks (even deep neural networks) to solve this problem. We believe the prediction accuracy will be better.