Project :

Predicting Accident Severity Level based on Data Science Application on Car Collisions Data for City of Seattle.

Author: Andre K.

Date: September 2020

DISCLAIMER : This project is conducted for the purpose of completing capstone project for IBM Data Science Professional certification. Although the project makes use of publicly available car accident data in Seattle, there is no relation of this project to the city of Seattle and result of this project, as a whole or part, should not be used outside of the purpose of completing the above-mentioned capstone project.

I. INTRODUCTION

The authority of Seattle has been able to collect nearly 200,000 cases of car collisions over the span of 15+ years (2004-2020), which should serve as a great resource to understand why collisions happen, and most importantly, how can we prevent collisions to happen, in particular for the case of severity level 2, which is collision that cause injuries or fatalities. This project aims to understanding the attributes of collisions that can be drawn from raw data, to name a few: road condition, weather, and then applies scientific data analysis mechanism (i.e. machine learning) to model how those attributes contribute to causing collisions level 1 and 2. The result of the model is tested using appropriate methodology to ensure accuracy. Using the accident prediction model built from this project, based on the attributes being observed at any given time, traffic authorities can immediately take appropriate measures to prevent collisions.

II. DATA EXPLORATORY AND ANALYSIS

This project utilizes the car collisions data made available by city of Seattle. The data spans over period from January 2004 to May 2020, which contains nearly 200,000 of car collisions data. The target variable (also known as “label”) of the data is “accident_severity_level”, which indicates whether the accident being observed is limited to property damage (level 1), or involves people injury or fatality (level 2). Other than the target variable, There are 37 attributes (columns), as summarized in following table:

Column Name	Description
SEVERITYCODE	The target variable, or label, where code 1 indicates property damage and code 2 indicates injury/fatality
X	Latitude coordinate of location
Y	Longitude coordinate of location
OBJECTID	Case Unique Identifier
INCKEY	Incident Key
COLDETKEY	Secondary Key
REPORTNO	Report Number
STATUS	Matched/Unmatched (for official reference purpose)
ADDRTYPE	Address Type (Alley, Block, Intersection)
INTKEY	Intersection Key (Applicable when Address Type is Intersection)
LOCATION	Address where collision is located
EXCEPTRSNCODE	Exception Code when location information is missing
EXCEPTRSNDESC	Exception Code description
SEVERITYCODE	Copy of Severity Code
SEVERITYDESC	Severity Code description
COLLISIONTYPE	Type of collision
PERSONCOUNT	Number of person involved
PEDCOUNT	Number of pedestrian involved
PEDCYLCOUNT	Number of cyclist involved
VEHCOUNT	Number of vehicle involved
INCDATE	Incident Date
INCDTTM	Incident Date and Time
JUNCTIONTYPE	Type of junction where collision happened
SDOT_COLCODE	Code use by SDOT (Seattle Department of Transportation)
SDOT_COLDESC	SDOT Code description
INATTENTIONIND	"Y" if collision is caused by Inattention
UNDERINFL	"Y" if collision is found related to drug or alcoholic influence
WEATHER	Weather condition at the time of collision
ROADCOND	Road condition at the time of collision
LIGHTCOND	Light condition at the time of collision
PEDROWNOTGRNT	"Y" if pedestrian right of way is not granted
SDOTCOLNUM	Collision number assigned by SDOT
SPEEDING	"Y" if speeding was involved
ST_COLCODE	Code used by State
ST_COLDESC	Description of code used by State
SEGLANEKEY	Key to indicate lane segment
CROSSWALKKEY	Key to indicate crosswalk
HITPARKEDCAR	"Y" if the collision involves hitting parked car

For data analysis purpose, we can drop some of the columns that are clearly not related to the attributes of collision, such as IDs. For the remaining data, we can further provide explanations on whether they can potentially be important attributes for collisions. Some of the attributes can also be grouped into same category.

Category	Column Name	Potential use as attributes
Target Variable	SEVERITYCODE	The target variable, or label, where code 1 indicates property damage and code 2 indicates injury/fatality
Location	X	Coordinate information can point out to location where collision is more frequent. Having said that, given the proximity of location, it can be hard to use coordinate to accurately allocate which location within city of Seattle that collision happens.
	Y	
	ADDRTYPE	Address Type (Alley, Block, Intersection). We can presume that collision is likely to happen at intersection, which we will study further in our analysis.
	LOCATION	Address where collision is located can be important attribute to indicate location where collision frequently happen, however given the large amount of different address, it can be hard to use address as useful attribute in our analysis
	JUNCTIONTYPE	Type of junction where collision happened can serve as important information, in particular if address type "intersection" is used as decision tree parameter
	SEGLANEKEY	Key to indicate lane segment. To use location information as attribute, this can be more practical to use than address.
	CROSSWALKKEY	Key to indicate crosswalk. To use location information as attribute, this can be more practical to use than address.
Type	COLLISIONTYPE	These columns indicate type of collision, which can be useful attribute related to collision and its severity. Further analysis is needed to potentially reduce the columns as the information can be redundant. For analysis, we can also remove the column that contains description, as for data processing purpose we only need the codes. For example: [Collision Type] : "Parked Car" is redundant with column [HITPARKEDCAR] so we can drop [HITPARKEDCAR] column. Similarly, [Collision Type] : "Angles" is redundant with [SDOT_COLCODE]: "11", [SDOT_COLDESC]: "MOTOR VEHICLE STRUCK MOTOR VEHICLE, FRONT END AT ANGLE", as well as with [ST_COLCODE]: "10", [ST_COLDESC]: "Entering at angle". Presumably, other than [COLLISIONTYPE], we can drop other type related columns.
	SDOT_COLCODE	
	SDOT_COLDESC	
	ST_COLCODE	
	ST_COLDESC	
	HITPARKEDCAR	
Number involved	PERSONCOUNT	This information is important attribute as more number involves in collision, the more likeliness it will cause injury or fatility. Although we potentially can reduce the attributes by only keeping [PERSONCOUNT] and drop the other columns, pedestrians and cyclists are more prone to injury so our data analysis need to study if we need to keep each of these columns as attributes related to severity level.
	PEDCOUNT	
	PEDCYLCOUNT	
	VEHCOUNT	
Date / Time	INCDATE	Dates can provide information on holiday period or weekend where traffic behavior is different than weekday. Time is also important indication as we can analyze how the trafic condition at rush-hours, morning, afternoon, evening are related to collisions.
	INCDTTM	
Violation	PEDROWNOTGRNT	"Y" if pedestrian right of way is not granted
	SPEEDING	"Y" if speeding was involved
	INATTENTIONIND	"Y" if collision is caused by Inattention

	UNDERINFL	"Y" if collision is found related to drug or alcoholic influence
Environment condition	WEATHER	Weather condition (raining, snowing, severe crosswinds etc.) is likely important attribute to cause collision.
	ROADCOND	Road condition (dry, ice, wet, oil, sand etc.) is likely important attribute to cause collision.
	LIGHTCOND	Light condition (daylight, dawn, dark etc.) is likely important attribute to cause collision.

Having been able to identify and reduce the dimension of potential attributes to use in data analysis, next step is to examine quality and availability for each attributes, which include following aspects:

a. Unbalance (Skewness) of target variable.
Data of target variable is skewed to collision severity level 1, which means there are significantly more data for severity level 1, compared to level 2. By performing a value count to "SEVERITYCODE" columns, we can see there are 136,485 of Severity Code 1 data, which is almost triple of 58,188 data for Severity Code 2. Having said that, in this analysis, it is reasonable to assume that severity code level 2 occur much less frequently than level 1, therefore there is no data reduction procedure applied further to balance the data. This will also ensure we can train the model more accurately by utilizing more data.

```
In [6]: collision_data['SEVERITYCODE'].value_counts()

Out[6]: 1    136485
        2     58188
        Name: SEVERITYCODE, dtype: int64
```

b. Missing data.
If the attributes that we select do not have sufficient data, it maybe better to remove the attribute as it will impair predictability of the model to be built.
Following columns are considered important attributes to the model, therefore further process is needed either to replace or drop the rows.

Column Name	Number of Missing Data (out of 194673 total data)
SEVERITYCODE	0
SEVERITYDESC	0
ADDRTYPE	1926
INTKEY	65070
LOCATION	2677
EXCEPTRSNCODE	109862
EXCEPTRSNDESC	189035
COLLISIONTYPE	4904
PERSONCOUNT	0
PEDCOUNT	0
PEDCYLCOUNT	0

VEHCOUNT	0
INCDATE	0
INCDTTM	0
JUNCTIONTYPE	6329
SDOT_COLCODE	0
SDOT_COLDESC	0
INATTENTIONIND	164868
UNDERINFL	189789
WEATHER	5081
ROADCOND	5012
LIGHTCOND	5170
PEDROWNOTGRNT	190006
SDOTCOLNUM	79737
SPEEDING	185340
ST_COLCODE	18
ST_COLDESC	4904
SEGLANEKEY	0
CROSSWALKKEY	0
HITPARKEDCAR	0

c. Data conversion.
   In order to be processed further for analysis. Attribute which has object type need to be converted to integer. For example: [COLLISIONTYPE] has description such as "Parked Car", "Rear Ended", which need to be converted into [1,2,…].
   Similarly, attribute with "Y" value, such as [SPEEDING] need to be converted into binary [0,1] for data processing and modelling purpose.

Understanding of the data as described above will pave a strong foundation to move into next section about methodology.

III. METHODOLOGY

III.a. Feature selection

In the data analysis section above, we have seen that the attributes that we collected can be categorized into: location of collisions, type of collisions, number of persons involved, date/time, violations involved and environment conditions. In this section, we will perform further analysis and decide on features that can be used for collisions prediction model.

- **Location.**
  Location contains information of coordinate of location (X: longitude, Y: latitude), Address Type, Location/Address, Junction Type, lane segment key and crosswalk key. Using the X, Y coordinates, we can plot location of collisions to the map of Seattle as illustrated in following figure. This figure plot the location of severity code level 2 (causing injury) in Seattle. This can serve as valuable information for the traffic authorities to analyse spots where collisions happen more frequently. However, for the scope of this project, we want to understand the collisions trend in Seattle as overall, not location specific. Therefore we will not use the X, Y coordinate as feature for our model.

Figure 1. Mat plot of severity code 2 (causing injury) collisions location in Seattle

Address Type contains three values: “Block”, “Alleys” and “Intersection”, which counts as follows for each severity code level 1 and 2. As we can see from table below, Address Type shows relation with number of collisions and severity code. Therefore we will choose [ADDRTYPE] column as one feature for the model. However, as Address

Type is in form of object, we need to convert the values into numeric, i.e.: “Alley”=1, “Block”=2 and “Intersection”=3.

ADDRTYPE	SEVERITYCODE	
Alley	1	669
	2	82
Block	1	96830
	2	30096
Intersection	1	37251
	2	27819

Junction Type contains a number of values that further classify intersections. However, the top two entries of this column are “not related to intersection” and “intersection related”, which is somehow duplicate with Address Type. Therefore, the column [JUNCTIONTYPE] is not selected as feature for the model.

Location/Address, lane segment key and crosswalk key provide specific location of the collision, similar with X and Y coordinates. As the scope of this project is Seattle in general, without classifying it into locations, these columns are not used as feature for the model.

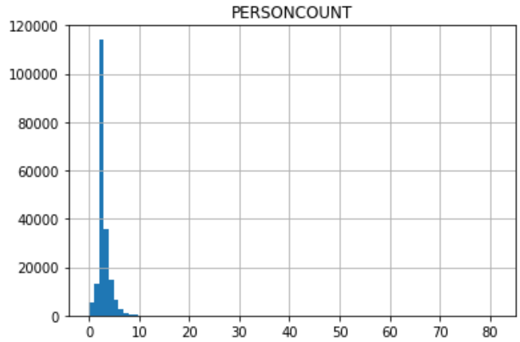
- **Collision Type.**

There are several columns that are related with the type of collision, namely: Collision Type, SDOT code, SDOT description, ST code, ST description and Hit Parked Car. These columns explain the types of collisions which then can explain how various types of collisions cause difference in severity code. However, these columns are redundant, therefore Collision Type is chosen as feature as the entries are relatively in small numbers and easier to interpret. Following table shows the number of collisions and its severity based on collision type. As with Address Type, for Collision Type we also should convert the type of column from object type of values into number.

COLLISIONTYPE	SEVERITYCODE	
Angles	1	21050
	2	13624
Cycles	1	671
	2	4744
Head On	1	1152
	2	872
Left Turn	1	8292
	2	5411
Other	1	17591
	2	6112
Parked Car	1	45325
	2	2662
Pedestrian	1	672
	2	5936
Rear Ended	1	19419
	2	14671
Right Turn	1	2347
	2	609
Sideswipe	1	16103
	2	2506

- **Number Involved**

  Number involved in accident columns consist of person counts, pedestrian counts, cyclist counts and vehicle counts. Given that pedestrian and cyclist have more exposure to injuries when involved in accident, we can increase accuracy by treating of these columns as a separate feature. However, to maintain generality of our model, we will only use the person counts data as feature for the model.



- **Date / Time**

  Date and time is useful to understand the pattern on relation between certain segment during the day, or during the week where collisions happen more frequently. From the Date feature, we can categorize whether the collision happen in weekday or weekend. Meanwhile for Time feature, we can categorize whether it is in rush-hours (7-9 AM and 4-6PM).

weekend_indi	SEVERITYCODE	
0	1	101291
	2	44038
1	1	35194
	2	14150

rushhour_indi	SEVERITYCODE	
0	1	109659
	2	44832
1	1	26826
	2	13356

- **Violations**

  There are 4 columns that can be attributed to violation of traffic rules, namely: Pedestrian rights are not granted, speeding, inattention driving and driving under influence.

PEDROWNOTGRNT	SEVERITYCODE	
Y	1	460
	2	4207

SPEEDING	SEVERITYCODE	
Y	1	5802
	2	3531

INATTENTIONIND	SEVERITYCODE	
Y	1	19408
	2	10397

UNDERINFL	SEVERITYCODE	
N	1	127071
	2	53597
Y	1	5559
	2	3562

  All four violations attributes will be used as features of the model. Note that these are all Y/N questions (blank values are regarded as N), in which need to be converted to binary (0,1) type for further processing.

- **Environment Condition**

  Last attributes category is for the columns that indicate environment/condition at the time collision happens. This consists of weather, road condition and light condition.

WEATHER	SEVERITYCODE	
Blowing Sand/Dirt	1	41
	2	15
Clear	1	75295
	2	35840
Fog/Smog/Smoke	1	382
	2	187
Other	1	716
	2	116
Overcast	1	18969
	2	8745
Partly Cloudy	1	2
	2	3
Raining	1	21969
	2	11176
Severe Crosswind	1	18
	2	7
Sleet/Hail/Freezing Rain	1	85
	2	28
Snowing	1	736
	2	171
Unknown	1	14275
	2	816

ROADCOND	SEVERITYCODE	
Dry	1	84446
	2	40064
Ice	1	936
	2	273
Oil	1	40
	2	24
Other	1	89
	2	43
Sand/Mud/Dirt	1	52
	2	23
Snow/Slush	1	837
	2	167
Standing Water	1	85
	2	30
Unknown	1	14329
	2	749
Wet	1	31719
	2	15755

LIGHTCOND	SEVERITYCODE	
Dark - No Street Lights	1	1203
	2	334
Dark - Street Lights Off	1	883
	2	316
Dark - Street Lights On	1	34032
	2	14475
Dark - Unknown Lighting	1	7
	2	4
Dawn	1	1678
	2	824
Daylight	1	77593
	2	38544
Dusk	1	3958
	2	1944
Other	1	183
	2	52
Unknown	1	12868
	2	605

As a summary, we will use following features for our model. Please also note in comments section for any necessary data conversion and normalization that is required before feeding it into the model.

Feature	Comment
ADDRTYPE	Values (“Alleys”, “Block”, “Intersection”) need to be converted into

	float 1,2,3 and then normalized.
COLLISIONTYPE	Values (“Angles”, “Cycles”, … “Sideswipe”) need to be converted into float 1,2,…,10 and then normalized.
PERSONCOUNT	Values need to be normalized.
INCDATE	Dates values will be categorized (binned) into weekend and weekdays, where we will have a binary column for weekend_indicator
INCDTTM	Hour values will be categorized (binned) into rush hours and not rush hours, where we will have a binary column for rushhour_indicator
PEDROWNOTGRNT	Values (Y/N) will be converted to binary. Blank is regarded as N.
SPEEDING	Values (Y/N) will be converted to binary. Blank is regarded as N.
INATTENTIONIND	Values (Y/N) will be converted to binary. Blank is regarded as N.
UNDERINFL	Values (Y/N) will be converted to binary. Blank and 0 is regarded as N. 1 is regarded as Y.
WEATHER	Values (“Clear”, “Raining”, … “Snowing”) need to be converted into float 1,2,…,11 and then normalized.
ROADCOND	Values (“Dry”, “Ice”, … “Wet”) need to be converted into float 1,2,…,9 and then normalized.
LIGHTCOND	Values (“Dawn”, “Daylight”,… “Dusk”) need to be converted into float 1,2,…,9 and then normalized.
SEVERITYCODE	This is the target variable. Values 1 and 2 will be converted into 0 and 1 for simplification and normalization purpose.

III.b. Model Development

The problem is categorized as classification problem, where from a set of data, we want to predict the outcome into category, in this case is severity level code 1 or 2. There are several machine learning model that we can utilize. Since this problem outcome only involves two category (code 1 or 2), we will use two methodologies : (1) Logistic Regression and (2) Decision Tree.

(1) Logistic Regression

Logistic Regression is a classification algorithm for categorical variables. In this case, we use independent variables, such as address type, collision type, day of the week, hours of a day, violations and environment conditions related features to predict the outcome of dependent variable, which is severity level code.

Logistic Regression is suitable to use, as the problem exhibit following conditions :

- Target variable is binary : in our case, Severity level code contains of 1 or 2, which can be transformed into binary (0/1) variable, where 0 means collision does not cause injury, and 1 means collision does cause injury.
- The problem can benefit from obtaining probabilistic result. While we would like to predict if a traffic accident will cause injury or not, we also will benefit by understanding the probability of injury to happen.

- The problem exhibits linear decision boundary. Features in the data show linear decision boundary that is suitable for logistic regression problem.
- We can explore and understand the impact of each feature to the severity level of accidents.

The implementation uses Python programming code and logistic regressions library on IBM Jupyter Notebook platform, which the codes are as follows. Firstly, the data is divided into training and testing set, where training set will use 80% of the cleaned version of the data, and testing set will use the remaining 20%. The model will use training data set to fit to logistic regression model. Testing data will be used to evaluate the model, which will be described in next section.

```
X = np.asarray(collision_features[['ADDRTYPE', 'COLLISIONTYPE', 'PERSONCOUNT', 'weekend_i
ndi', 'rushhour_indi', 'PEDROWNOTGRNT', 'SPEEDING','INATTENTIONIND','UNDERINFL','WEATHE
R','ROADCOND','LIGHTCOND']])
y = np.asarray(collision_features['Injury'])
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, y, 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)
```

```
Train set: (150360, 12) (150360,)
Test set: (37590, 12) (37590,)
```

```
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
```

(2) Decision Tree

Another model that we will utilize to create prediction model is decision tree model. Decision tree predict the outcome of the model by creating branches based on each feature. In this project, the tree will split the outcome by using hierarchical branch comprised of address type, collision type, weekend indicator, rush hour indicator, violation, and environment condition. The model will create the hierarchy iteratively by maximizing the information gain (or minimizing entropy) produced by the decision tree.

The implementation uses Python programming code and decision tree library on IBM Jupyter Notebook platform. Similar to our approach in logistic regression model, the data is divided into training and testing set, where training set will use 80% of the cleaned version of the data, and testing set will use the remaining 20%. The model will use training data set to fit to decision tree model. Testing data will be used to evaluate the model, which will be described in next section. Below is the implementation code in Jupyter Notebook.

```
CollisionTree = DecisionTreeClassifier(criterion="entropy", max_depth = 8)
CollisionTree # it shows the default parameters
```

```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=8,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

```
CollisionTree.fit(X_train,y_train)
```

```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=8,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

```
predTree = CollisionTree.predict(X_test)
```

III.c. Model Testing

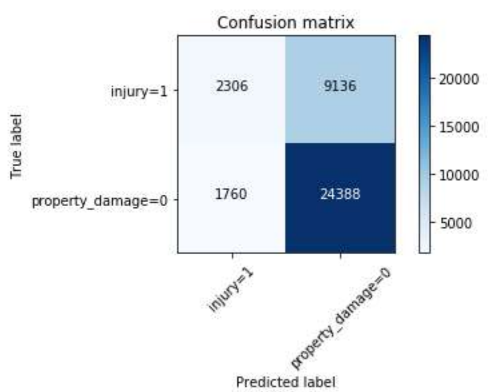
To test and evaluate our model, we use jaccard similarity score and confusion matrix for of logistic regression model, meanwhile decision tree model is evaluated using accuracy score.

For logistic regression model, the model obtained jaccard similarity score of 0.71.

```
from sklearn.metrics import jaccard_similarity_score
jaccard_similarity_score(y_test, yhat)
```

```
0.7101356743814844
```

Using f1 weighted average score that can be calculated confusion matrix, the resulting f1 score is 0.66.



```
print (classification_report(y_test, yhat))
```

```
              precision    recall  f1-score   support

           0       0.73      0.93      0.82     26148
           1       0.57      0.20      0.30     11442

   micro avg       0.71      0.71      0.71     37590
   macro avg       0.65      0.57      0.56     37590
weighted avg       0.68      0.71      0.66     37590
```

For the Decision Tree model, we can obtained the model accuracy score of 0.75.

```
predTree = CollisionTree.predict(X_test)
```

```
from sklearn import metrics
import matplotlib.pyplot as plt
print("DecisionTrees's Accuracy: ", metrics.accuracy_score(y_test, predTree))
```

```
DecisionTrees's Accuracy:  0.7502793296089385
```

IV.CONCLUSIONS AND FUTURE DEVELOPMENTS

By utilizing collisions data obtained from the city of Seattle, over past 15 years, we can build a prediction model that can predict whether a certain set of traffic conditions will likely cause injury or not. From the raw data, we can build model that utilize features of location type, collision type, number of persons involved, weekend/weekday indicator, rush hour indicator, whether there is violation of traffic rules and environment condition such as weather, road and light condition.

Two approaches were chosen to best model this problem, namely (1) Logistic Regression model and (2) Decision Tree model. With assistance of Python programming language and library on Jupyter Notebook platform, the model was build and tested. Evaluation of the model shows accuracy score around 70% for both models.

The resulting model will be an important guidance for the traffic authorities to take appropriate preventive as well as reactive measures against various traffic conditions, to prevent or minimize injury and potential fatalities.

Potential future enhancement of the model can be useful to be more specific in the prediction. For example, this project did not use location specific (latitude and longitude coordination) of collisions data, as the model is aiming to make prediction for city of Seattle in general. However, these data can be powerful to spot specific location where high level severity collision happens and make appropriate safety enhancements for the locations.

Other potential development is to utilize data of number of pedestrian and number of cyclist involved in the collisions. This project uses the overall number of persons involved in collision as feature, meanwhile it is reasonable to assume that pedestrian and cyclist are more prone to injury. Further use of these data can be helpful to tackle specific traffic policies toward pedestrian and cyclist.