Seattle City Car Accident Case Study

1. Business Understanding

A rational person is always trying to avoid car collisions, one of the ways to do so is to pay more attention while driving. However, human is not a machine, we cannot expect people always be on their guard. So, for the sake of efficiency, our question is if there is a possible method that can tell people when to pay more attention on driving or maybe stop from driving.

By my intuition, horrible weather or road conditions might lead to higher risk of car collisions. Based on this assumption, the main purpose of this project is to alert drivers to be more careful under some certain weather and road conditions. To achieve this purpose, similar studies suggested that we should build a machine learning model to predict the severity of an accident given related attributes.

2. Data Understanding

This project is based on the data set provided by Seattle Police Department and recorded by Traffic Records of the City of Seattle City. The purpose of analyzing this data set is to find the conditions that will lead to car collisions (84 types of collisions).

The data set consist 38 columns and 194,673 rows,

The dependent variable is "SEVERITYCODE" and for simplicity we use "WEATHER"," ROADCOND" and "LIGHTCOND" as independent variables.

Before starting data analysis, we need to pre-process the data set due to the existence of null values in some records and imbalance of "SEVERITYCODE".

2.1. Dependent variable "SEVERITYCODE"

The "SEVERITYCODE" stated the severity of the collision into the following 5 baskets using corresponded codes in the data set:

I. Fatality-3
II. Serious injury-2b
III. Injury-2
IV. Prop damage-1
V. Unknown-0

After using value_counts(), I find there is only level 1 prop damage and level 2 injury in our data set.

2.2. Possible independent variables

Within the 38 columns:

1. 2 of them are dependent variable "SEVERITYCODE"
2. 2 of them are coordinates
3. 2 of them are time
4. 29 of them are either code or unrelated information
5. The rest 3 of them are useful in this study

After examining all variables, I find that the useful independent variables are: WEATHER, ROADCOND and LIGHTCOND.

2.3. Data Pre-processing (using python)

2.3.1. Missing Data

By using isnull().sum(), I find that my independent variables have null values.
By using value_counts(), I find there is an 'Unknown' value which should be treated as missing value.
So, I changed 'Unknown' to null and calculated the percentage of null values to decide what should I do with null values.

	null count	null percentage
SEVERITYCODE	0	0.000000
WEATHER	20172	0.103620
ROADCOND	20090	0.103199
LIGHTCOND	18643	0.095766

Since WEATHER, ROADCOND and LIGHTCOND are all categorical data and the null percentages are lower than 20%, I decide to use the mode to replace the null values.
Also, I deleted columns with value 'Others' due to it is ambiguous and its low percentage.

2.3.2. Balancing Data

After dealing with the missing data, I find that label is imbalanced.

```
#Imbalanced Data
dfn['SEVERITYCODE'].value_counts()
```

```
1    135525
2     57981
Name: SEVERITYCODE, dtype: int64
```

So, I used resample() function for down-sampling to fix this.

```
#create new dataset, Balanced df
bdf=[dfn_majority_down, dfn_minority]
bdf=pd.concat(bdf)
bdf['SEVERITYCODE'].value_counts()
```

```
2    57981
1    57981
Name: SEVERITYCODE, dtype: int64
```

2.3.3. Turning categorical variables into quantitative variables

By using pd.get_dummies() we get our new numerical variables. There are 24 columns in our new dataframe.

```
#new dataframe
print(bdf.columns)
print(bdf.shape)
```

```
Index(['SEVERITYCODE', 'Blowing Sand/Dirt', 'Clear', 'Fog/Smog/Smoke',
       'Overcast', 'Partly Cloudy', 'Raining', 'Severe Crosswind',
       'Sleet/Hail/Freezing Rain', 'Snowing', 'Dry', 'Ice', 'Oil',
       'Sand/Mud/Dirt', 'Snow/Slush', 'Standing Water', 'Wet',
       'Dark - No Street Lights', 'Dark - Street Lights Off',
       'Dark - Street Lights On', 'Dark - Unknown Lighting', 'Dawn',
       'Daylight', 'Dusk'],
      dtype='object')
(115962, 24)
```

2.3.4. Data Normalization

I split the data into training set (70%) and test set (30%) and normalized training set.

```
# normalization
s = StandardScaler()
X_train_scale = s.fit_transform(X_train)
X_test_scale = s.transform(X_test)
X_train_scale
```

```
array([[-0.01530107,  0.70421448, -0.05535852, ..., -0.11796075,
         0.66479142, -0.18107437],
       [-0.01530107,  0.70421448, -0.05535852, ..., -0.11796075,
         0.66479142, -0.18107437],
       [-0.01530107,  0.70421448, -0.05535852, ..., -0.11796075,
         0.66479142, -0.18107437],
       ...,
       [-0.01530107, -1.42002193, -0.05535852, ..., -0.11796075,
        -1.50423121, -0.18107437],
       [-0.01530107,  0.70421448, -0.05535852, ..., -0.11796075,
         0.66479142, -0.18107437],
       [-0.01530107, -1.42002193, -0.05535852, ..., -0.11796075,
         0.66479142, -0.18107437]])
```

3. Data analysis using Machine Learning

In this project, I will use three machine learning models with sklearn and evaluate it with Jaccard Score, F1 Score and Accuracy.

3.1. KNN

```
# Using KNN
knn = KNeighborsClassifier(n_neighbors=5).fit(X_train_scale,y_train)
knn
ky_hat = knn.predict(X_test_scale)
print(classification_report(y_test,ky_hat))
```

	precision	recall	f1-score	support
1	0.51	0.24	0.32	17471
2	0.50	0.77	0.61	17318
accuracy			0.50	34789
macro avg	0.51	0.50	0.47	34789
weighted avg	0.51	0.50	0.46	34789

3.2. Decision Tree

```
# Using Decision Tree
decisiontree = DecisionTreeClassifier(criterion="entropy").fit(X_train_scale,y_train)
dy_hat = decisiontree.predict(X_test_scale)
print(classification_report(y_test,dy_hat))
```

	precision	recall	f1-score	support
1	0.51	0.66	0.58	17471
2	0.52	0.37	0.44	17318
accuracy			0.52	34789
macro avg	0.52	0.52	0.51	34789
weighted avg	0.52	0.52	0.51	34789

3.3. Logistic Regression

```
# Using Logistic Regression
logr = LogisticRegression(C=1, solver='liblinear').fit(X_train_scale,y_train)
ly_hat = logr.predict(X_test_scale)
print(classification_report(y_test,dy_hat))
```

	precision	recall	f1-score	support
1	0.51	0.66	0.58	17471
2	0.52	0.37	0.44	17318
accuracy			0.52	34789
macro avg	0.52	0.52	0.51	34789
weighted avg	0.52	0.52	0.51	34789

3.4. Jaccard Score

```
# Result Evaluation for KNN
jaccard_score(y_test, ky_hat)
```

```
0.19284813967601885
```

```
# Result Evaluation for Decision Tree
jaccard_score(y_test, dy_hat)
```

```
0.4049478706485245
```

```
# Result Evaluation for Logistic Regression
jaccard_score(y_test, ly_hat)
```

```
0.40503869121232466
```

4. Evaluation
   The following table concluded F1 Score based on macro average, Accuracy and Jaccard Score:

ML Model	F1 Score(macro)	Accuracy	Jaccard Score
KNN	0.46	0.50	0.19
Decision Tree	0.51	0.52	0.40
Logistic Regression	0.51	0.52	0.41

   Based on the above table, Logistic Regression appeared to be the best model for predicting car accident severity level.

5. Conclusion
   In this study, I analyzed the relationship between weather, road condition, light condition and severity of accident. I used classification models to predict the likelihood of an accident. These models can be helpful to drivers to understand that weather, road condition and light condition may have an impact on their safety.