Seattle's Collision Severity of Traffic Accident

1. Business Understanding

In early 2020, World Health Organization stated that every year the lives of approximately 1.35 million people are cut short as a result of a road traffic crash. Between 20 and 50 million more people suffer non-fatal injuries, with many incurring a disability as a result of their injury.

Road traffic injuries cause considerable economic losses to individuals, their families, and to nations as a whole. These losses arise from the cost of treatment as well as lost productivity for those killed or disabled by their injuries, and for family members who need to take time off work or school to care for the injured. Road traffic crashes cost most countries 3% of their gross domestic product.

In order to reduce traffic accident in Seatlle, Government is going to apply new approach of early-alert system that could prevent accidents. Such an approach takes into account people's vulnerability to injuries in road traffic crashes. The cornerstones of this approach by predicting severity collition type as a result of Clssification of Machine Learning algorithm with condition given such as vehicle speed, weather, road condition, visibility, number of pedestrian and vehicle.

2. Data Understanding

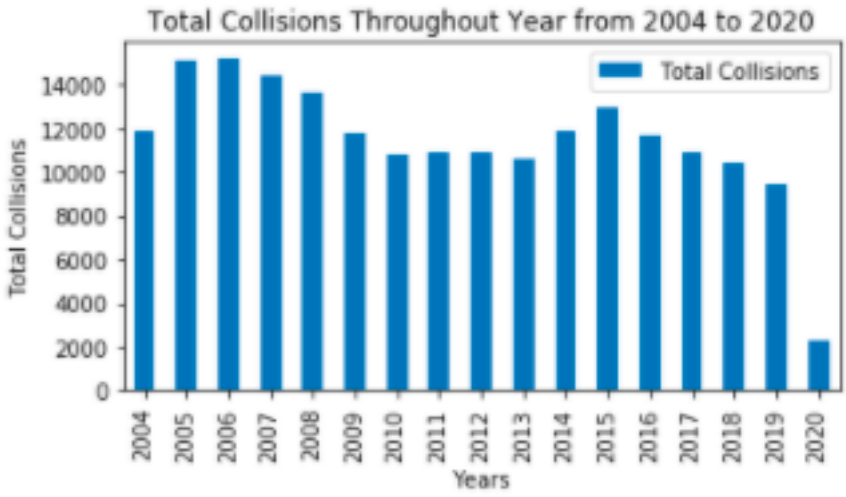
The data was gathered from the Seattle Department of Transportation (SDOT) Traffic Management Division from 2004 to 2020 in csv format. It is consisted of 37 independent variables and 194,673 rows. The dependent variable which becomes target, “SEVERITYCODE”, contains numbers that correspond to different levels of severity caused by an accident from 0 to 3, as following :

0: Little to no Probability (Clear Conditions)
1: Very Low Probability — Chance or Property Damage
2: Low Probability — Chance of Injury
2b: Mild Probability — Chance of Serious Injury
3: High Probability — Chance of Fatality

Particularly in the data given, severity type will only be for low severity collision : 1 (property damage only) and 2 (injury collision without fatality accident).

In this section of Data Understanding, we'd like to know the trend of collision accident in Seattle, skewness of the dataset, the correlation of each attributes in the data (of which later the most predictable variable of accident will be chosen to train the data) and to know how correlate weather, road condition and visibility combination to collisions, and statistical significant of attributes.

a. Trend of Collisions in Seattle



From figure above we can say, the trend of collisions occured has been declining from 2005 to 2013, but then rise up in 2014 to 2015 and starting to gradually declining in 2016 to 2019.

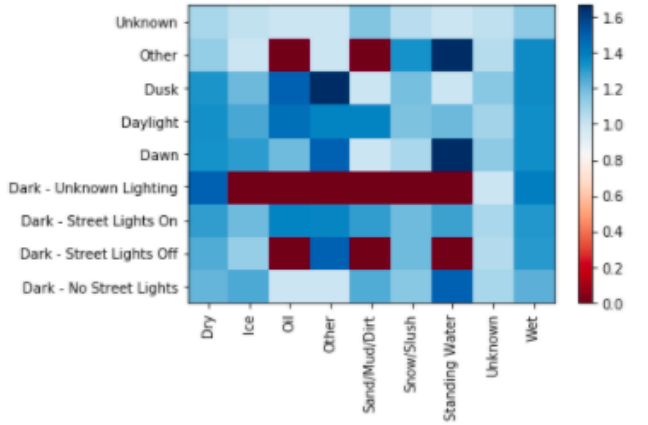
b. **Skewness of Dataset**

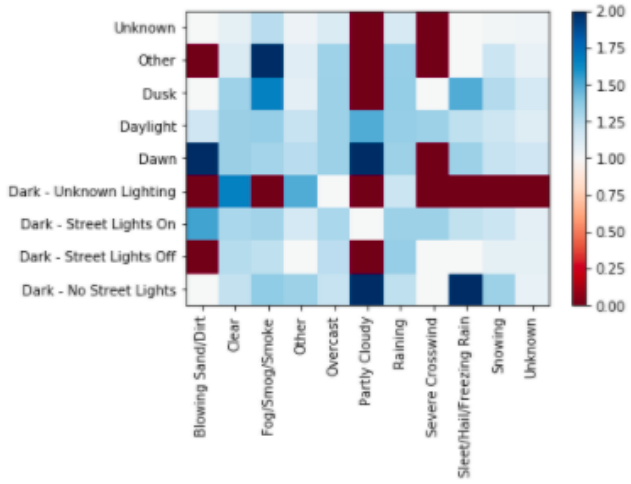
Another information can be obtained from the data is skewness, which this element takes an important role for accuracy especially in Classification. The data supposed to be balanced before we input the train dataset to classification machine learning algorithm. We need to check whether the data is balance or otherwise.

Now we know the dataset are unbalanced and need rebalancing later in pre-processing.

c. Correlation Attributes to Collision Severity

Before doing any data wrenching, we can exploit the data to know how correlate other variables in object type to collision severity.





From those 2 figures of data visualisation that tells combination from visibility, road condition and weather, we can easily know in which particular condition if they are met are probable to have more severe collision. For instance, collision become more severe when the lighting road is dark (no street lights) and the weather is sleed/hail/freezing rain.
As conclusion we can say that Lighting Condition, Road Condition and Weather are also becoming the most predictable variable to severe collision.

d. Statistical Significant through P-Value

The P-value is the probability value that the correlation between these two variables is statistically significant. Normally, we choose a significance level of 0.05, which means that we are 95% confident that the correlation between the variables is significant.</p>

By convention, when the

- p-value is < 0.001: there is strong evidence that the correlation is significant.
- the p-value is < 0.05: there is moderate evidence that the correlation is significant.
- the p-value is < 0.1: there is weak evidence that the correlation is significant.
- the p-value is > 0.1: there is no evidence that the correlation is significant.

```
from scipy import stats
pearson_coef, p_value = stats.pearsonr(df['PEDCYLCOUNT'], df['SEVERITYCODE'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is 0.21421818271573084  with a P-value of P = 0.0
```

```
pearson_coef, p_value = stats.pearsonr(df['PEDCOUNT'], df['SEVERITYCODE'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is 0.24633815772799883  with a P-value of P = 0.0
```

Thus we have strong evidence that the hypothesis of choosing attribute PEDCYLCOUNT and PEDCOUNT are significant to Severity Type occured. This means that bicycle and pedestrian becoming predictor to occuring collision. As an addition, we could not involve speeding in predictor variable since scarcity of data, that the missing data more than 90% out of whole dataset.

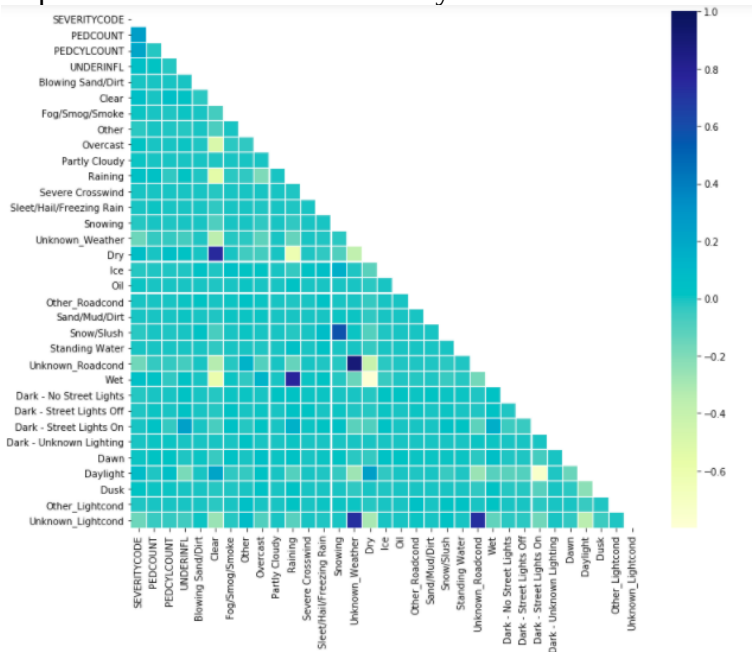
Should be noted that we could not applying directly Pearson Correlation and calculating P-Value for object type, such as UNDERINFL, WEATHER, ROADCOND, LIGHTCOND, then we need to do pre-processing on get dummies for those attributes later in Pre-Processing, then we could figure out this statistical significant evidence.

3. Data Preparation

There are numerous steps have been taken in Data Preparation, they are :

a.. Drop unused attributes
b. Delete rows for majority missing data
c. Turn Categorical Variable (object type) into Quantitative Variables, especially on variables : WEATHER, ROADCOND, LIGHTCOND, UNDERINFL.

As final result of these 3 data pre-processing, all variables are set into quantitative variable, hence we can see all predictors correlation to severity collision with a data visualisation below:



GREAT! Finally we have strong evidence of attributes which become the strongest predictors amongst other variable:

1. PEDCOUNT : The total number of pedestrian in the collision
2. PEDCYLCOUNT : The number of bicycles involved in the collision
3. UNDERINFL : Drugs influenced of driver
4. WEATHER
5. ROADCOND : Road Condition
6. LIGHTCOND : Visibility (Road Lighting)

d. Balancing dataset through under sampling

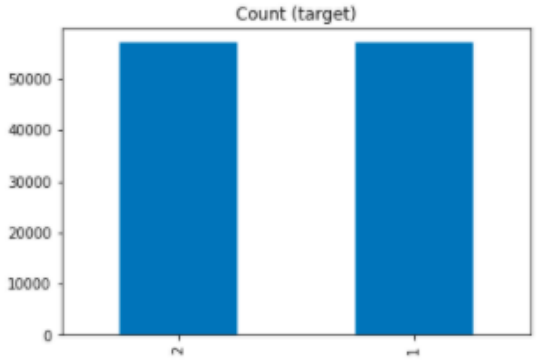
```
df_class_1_under = df_class_1.sample(count_class_2)
df_test_under = pd.concat([df_class_1_under, df_class_2], axis=0)

print('Random under-sampling:')
print(df_test_under['SEVERITYCODE'].value_counts())

df_test_under['SEVERITYCODE'].value_counts().plot(kind='bar', title='Count (target)')
```

```
Random under-sampling:
2    57052
1    57052
Name: SEVERITYCODE, dtype: int64

<matplotlib.axes._subplots.AxesSubplot at 0x1a234b5d10>
```



Now we have balanced dataset!

e. Shuffling The Dataset

Considering biased are introduced after under sampling, that the first 50% of whole dataset are having same target which is '1'. Hence to train dataset better, would be great if we can do shuffling before splitting the dataset into train and test data.

f. Normalization

After initializing X as predictor and y as target, we can finally normalize the X value before splitting the data into train and test set.

4. Methodology

Classification method used are KNN, Logistic Regression, SVM Classification

KNN CLASSIFICATION

```
from sklearn.neighbors import KNeighborsClassifier

k = 17
#Train model
KNN_model = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train)
KNN_model

#predict
yhat = KNN_model.predict(X_test)
yhat[0:5]
```

```
/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:5: DataConversionWarning: A column-vector y was pass
ed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  """
```

```
array([2, 2, 2, 1, 2])
```

LOGISTIC REGRESSION

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

LR_yhat_prob = LR.predict_proba(X_test)
```

```
/opt/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y
was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

SVM Classification

```
from sklearn import svm
SVM_model = svm.SVC()
SVM_model.fit(X_train, y_train)
yhat_SVM = SVM_model.predict(X_test)
yhat_SVM
```

```
/opt/anaconda3/lib/python3.7/site-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y
was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
array([2, 1, 1, ..., 1, 2, 1])
```

5. Model Evaluation

To evaluate models used, here we have measurements of accuracy, Jaccard index and F1-Score to know how well machine learning model works.

```
print("Train set Accuracy: ", metrics.accuracy_score(y_train, KNN_model.predict(X_train)))
print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat))

print("KNN Jaccard index: %.2f" % jaccard_similarity_score(y_test, yhat))
print("KNN F1-score: %.2f" % f1_score(y_test, yhat, average='weighted') )
```

```
Train set Accuracy:  0.5840079751980106
Test set Accuracy:  0.5819639805442356
KNN Jaccard index: 0.58
KNN F1-score: 0.58
```

```
/opt/anaconda3/lib/python3.7/site-packages/sklearn/metrics/_classification.py:664: FutureWarning: jaccard_similarity_
score has been deprecated and replaced with jaccard_score. It will be removed in version 0.23. This implementation ha
s surprising behavior for binary and multiclass classification tasks.
  FutureWarning)
```

```
print("Train set Accuracy: ", metrics.accuracy_score(y_train, LR.predict(X_train)))
print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat_LR))

print("LR Jaccard index: %.2f" % jaccard_similarity_score(y_test, yhat_LR))
print("LR F1-score: %.2f" % f1_score(y_test, yhat_LR, average='weighted') )
print("LR LogLoss: %.2f" % log_loss(y_test, LR_yhat_prob))
```

```
Train set Accuracy:  0.5998159569689866
Test set Accuracy:  0.6029972393847772
LR Jaccard index: 0.60
LR F1-score: 0.58
LR LogLoss: 0.62
```

```
/opt/anaconda3/lib/python3.7/site-packages/sklearn/metrics/_classification.py:664: FutureWarning: jaccard_similarity_
score has been deprecated and replaced with jaccard_score. It will be removed in version 0.23. This implementation ha
s surprising behavior for binary and multiclass classification tasks.
  FutureWarning)
```

```
print("Train set Accuracy: ", metrics.accuracy_score(y_train, SVM_model.predict(X_train)))
print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat_SVM))

print("SVM Jaccard index: %.2f" % jaccard_similarity_score(y_test, yhat_SVM))
print("SVM F1-score: %.2f" % f1_score(y_test, yhat_SVM, average='weighted') )
```

```
Train set Accuracy:  0.6018754861255655
Test set Accuracy:  0.6020770343105035
SVM Jaccard index: 0.60
SVM F1-score: 0.58
```

6. Result

Final result of 3 (three) different Classification of Machine Learning Algorithm used, we have model measurements as can be seen below :

Algorithm	Accuracy	Jaccard	F1 Score	LogLoss
KNN	0.58	0.58	0.58	N/A
Logistic Regression	0.60	0.60	0.58	0.62
SVM	0.60	0.60	0.58	N/A

Logistic Regression considered to have the highest accuracy since it is suited best for binary classification, while SVM also took same accuracy, Jaccard Index and F1 Score with Logistic Regression considering it uses a technique called the kernel trick to transform data and then based on these transformations it finds an optimal boundary between the possible outputs, thus it can capture much more complex relationships between datapoints without performing difficult transformations.

7. Discussion

There are numerous fascinating insights have been obtained after wrenching the data:

1. Not only weather, road condition, and lighting condition becoming predictor to severe collision, but also drug influence of driver (took 4% of whole low severity collision cases), existence of pedestrian and bicyles on the road (which took 8% of whole low severity cases), this is evidenced by calculation of Pearson Correlation and Statistical Significant of p-value of those variables to collision type.

2. As mentioned by World Heatlh Organization, vehicle's speed considered as a key risk factor in road traffic injuries. An increase in average speed of 1 km/h typically results in a 3% higher risk of a crash involving injury, with a 4–5% increase for crashes that result in fatalities. It is unfortunate the dataset could not reveal vehicle's speed with considerable number of cases due to its data scarcity (more than 90% speeding data out of whole dataset are missing). We suggest to the Government to enhance the dataset quantity especially with vehicle speeding, by improving data retrieval method of vehicle speed (one of example using computer vision technology to capture real time speed).

3. Shuffling is needed in data preparation, as an effect of data biased introduced after balancing the dataset. Shuffle will ensure that each data point creates an "independent" change on the model, without being biased by the same points before them. Data has shown that doing shuffling after applying the most predictor variables into machine learning could increase 0.6% accuracy.

8. Conclusion

To tackle case of Collision Severity in Seattle, dataset utilized has been provided by Seattle Department of Transportation (SDOT) Traffic Management Division from 2004 to 2020 in csv format. From the dataset, there are 37 variables or attributes lies on 194.673 cases that can be used to do further analysis on predicting the severity of collision occurs by using Classification algoritms of Machine Learning.

Should be noted that the dataset only covered a low severity collision : property damage only and injury collision (non-fatality accident). Another point regarding dataset was the vehicle speeding'data given cannot be utilized optimally due to its scarcity (more than 90% out of whole dataset are missing). It's so unfortunate, since speed become one of key predictor in severity collision.

Before doing further data wrenching, we found the trend of collisions accured has been declining from 2005 to 2013, but then rise up in 2014 to 2015 and starting to gradually declining in 2016 to 2019.
Besides, from correlation amongst object we found that when some certain condition are met could increase likelihood of severity collision. For an instance, collision become more severe when the lighting road is dark (no street lights) and the weather is sleet/hail/freezing rain.

There are numerous steps have been taken in Data Preparation, they are :
1. Drop unused attributes
2. Delete rows for majority missing data
3. Turn Categorical Variable (object type) into Quantitative Variables
4. Balancing through Under Sampling
5. Shuffling The Dataset
6. Normalization

On top of that, we highlighting that the predictors of severity collisions are not only weather, road condition, and lighting condition, but also drug influence of driver (took 4% of whole low severity collision cases), existence of pedestrian and bicyles on the road (which took 8% approximately of whole low severity cases), this is evidenced by calculation of Pearson Correlation and Statistical Significant of p-value.

Three methods are utilized in this Classification Machine Learning work, there are K-Neural Network (KNN), Logistic Regression (Log Reg) and Support Vector Machine (SVM).
As a result, Logistic Regression & Support Vector Machine took the highest accuracy, 60%, which slightly different than the KNN method. One most possible cause is Log Reg suited best for binary classification, while SVM uses a technique called the kernel trick to transform data and then based on these transformations it finds an optimal boundary between the possible outputs, thus it can capture much more complex relationships between datapoints without performing difficult transformations.