PREDICTING THE SEVERITY OF ROAD COLLISIONS IN SEATTLE

Thang Le Dinh

October 2020

INTRODUCTION

Background

- Seattle is the largest city in both the state of Washington and the Pacific Northwest region of North America, being the fastest-growing major U.S. city, with a 3.1% annual growth rate.

Problem

- One of the most important challenges for the government of Seattle is to prevent car collisions.

Motivation

- This project aims at predicting the severity of car collisions for the Seattle Police Department (SPD) based on different factors, such as weather, road and light conditions as well as the number of peoples and vehicles involved.

DATA UNDERSTANDING

Data cleaning

- The dataset is downloaded and read into a data frame using a Jupyter Notebook.
- This dataset is cleaned up to remove columns that are not informative to us for visualization.
- Columns and rows, where at least one element is missing, have been removed.

Clean up the dataset to remove columns that are not informative to us for visualization

```
# Remove columns, which are not informative

collision_df.drop(["X", "Y", "INCKEY","COLDETKEY","REPORTNO","STATUS","INTKEY","LOCATION","EXCEPTRSNCODE","EXCEPTRSNDESC",
                   "SEVERITYCODE.1", "SEVERITYDESC", "COLLISIONTYPE", "UNDERINFL","PEDCOUNT","PEDCYLCOUNT","INCDATE","INCDTTM",
                   "JUNCTIONTYPE","SDOT_COLCODE","SDOT_COLDESC", "PEDROWNOTGRNT","SDOTCOLNUM","ST_COLCODE","ST_COLDESC",
                   "SEGLANEKEY","CROSSWALKKEY","HITPARKEDCAR"] ,
                    axis=1 , inplace=True)
collision_df.head()
```

	SEVERITYCODE	OBJECTID	ADDRTYPE	PERSONCOUNT	VEHCOUNT	INATTENTIONIND	WEATHER	ROADCOND	LIGHTCOND	SPEEDING
0	2	1	Intersection	2	2	NaN	Overcast	Wet	Daylight	NaN
1	1	2	Block	2	2	NaN	Raining	Wet	Dark - Street Lights On	NaN
2	1	3	Block	4	3	NaN	Overcast	Dry	Daylight	NaN
3	1	4	Block	3	3	NaN	Clear	Dry	Daylight	NaN
4	2	5	Intersection	2	2	NaN	Raining	Wet	Daylight	NaN

DATA UNDERSTANDING (CONT.)

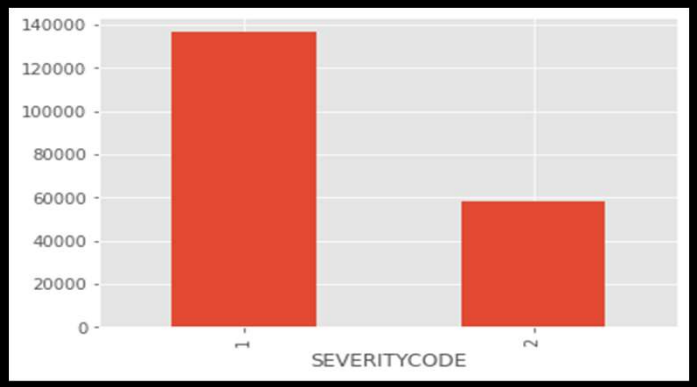
Feature selection

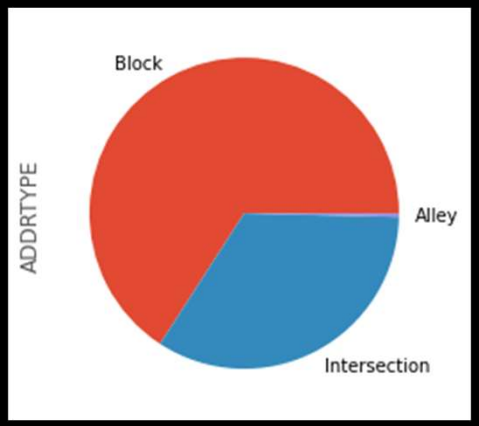
- Key attributes were selected to determine certain patterns and correlations and then used for training a machine-learning model to predict the probability and severity of collisions.

Attributes	Description	Data type
OBJECTID	ESRI unique identifier	int64
SEVERITYCODE	A code that corresponds to the severity of the collision	int64
ADDRTYPE	Collision address type	object
PERSONCOUNT	The total number of people involved in the collision	int64
VEHCOUNT	The number of vehicles involved in the collision	int64
INATTENTIONIND	Whether or not collision was due to inattention	object
WEATHER	A description of the weather conditions during the time of the collision	object
ROADCOND	The condition of the road during the collision	object
LIGHTCOND	The light conditions during the collision	object
SPEEDING	Whether or not speeding was a factor in the collision	object

EXPLORATORY DATA ANALYSIS

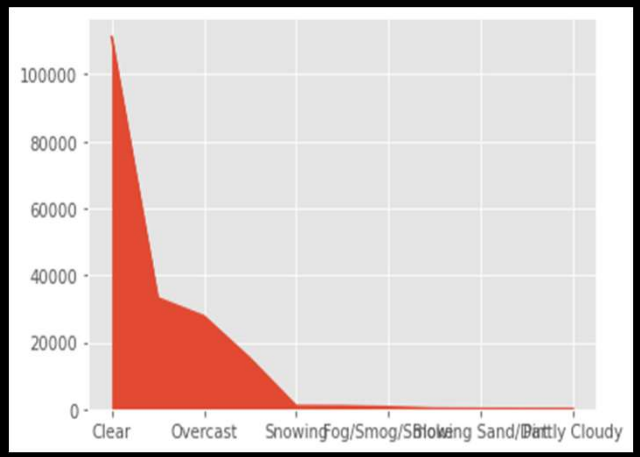
- The dataset includes different categorical variables, those data can be explored to determine the relationship with the severity of collisions.
- Firstly, we begin with Severity Code and Address type, Weather, Road condition, and Light condition.

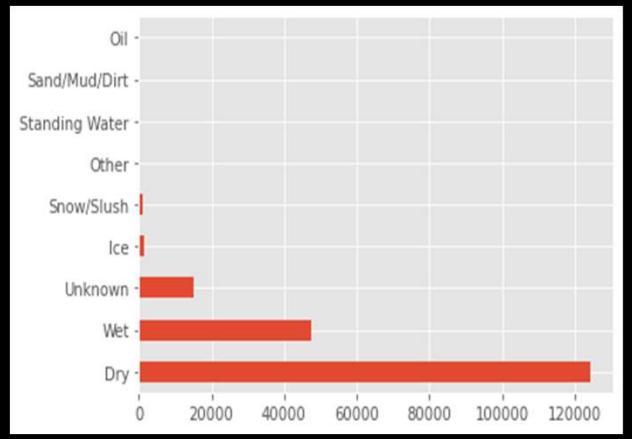


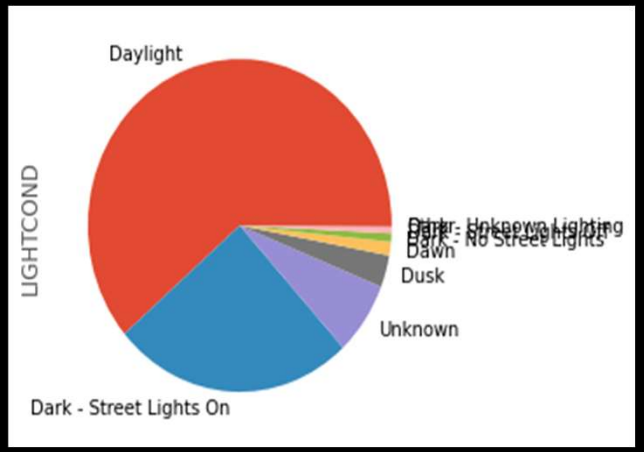


EXPLORATORY DATA ANALYSIS (CONT.)

- Thus, we continue with Weather, Road condition, and Light condition.

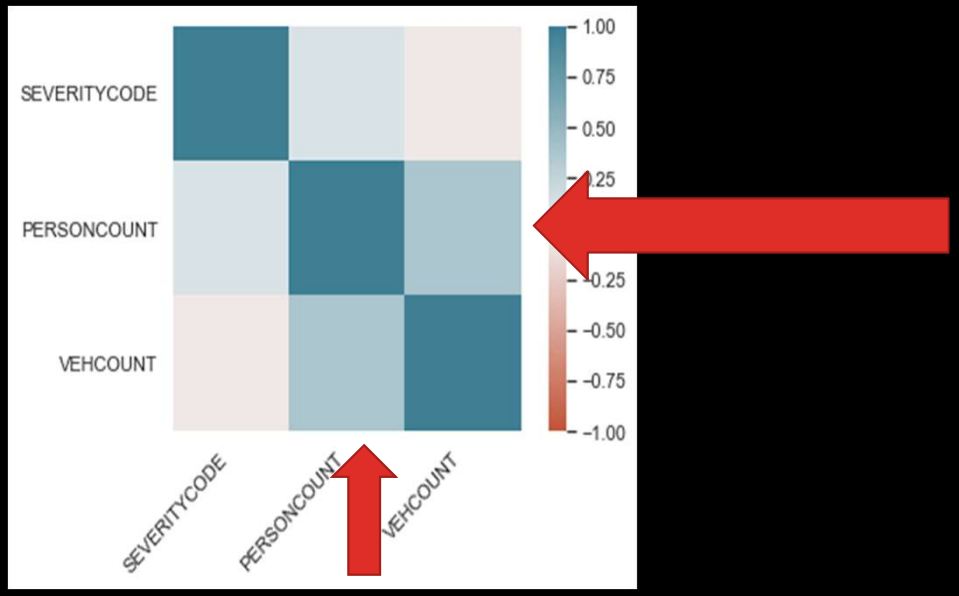






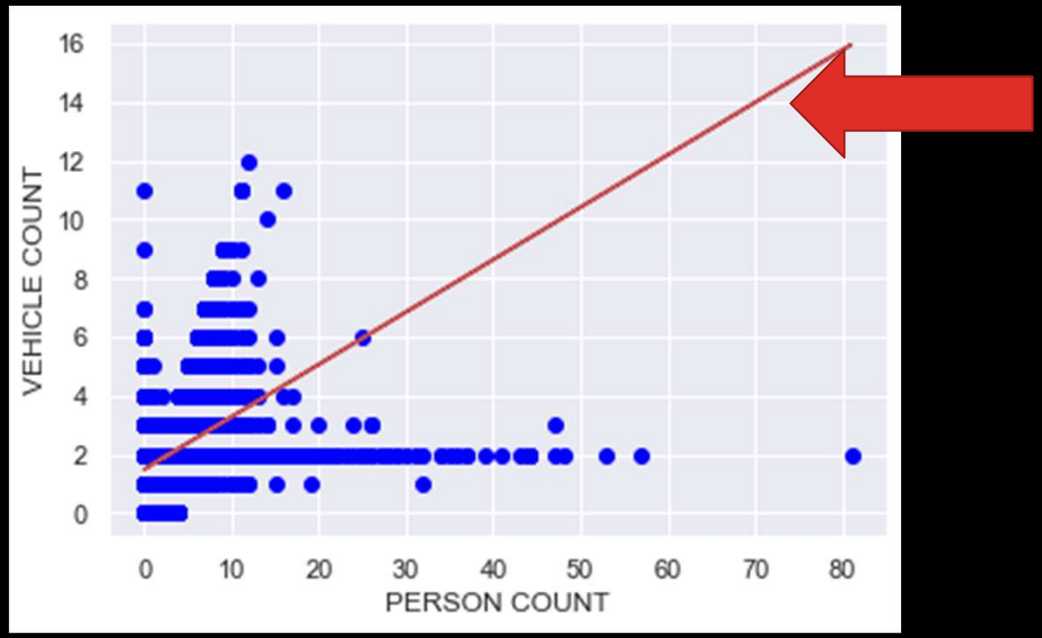
EXAMINING THE POTENTIAL RELATIONSHIPS WITH SCATTER PLOTS

- Let us focus on the numerical variables of the dataset: Severity code, Person count and Vehicle count.
- A heat map is used to visualize the relationships depicted by colour



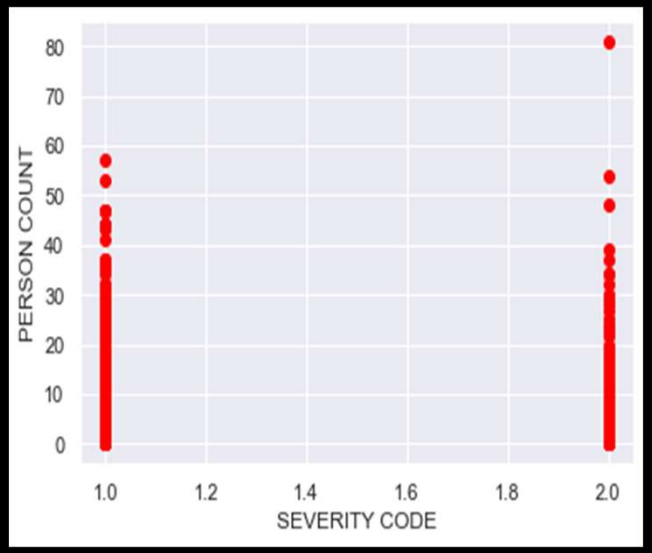
EXAMINING THE POTENTIAL RELATIONSHIPS WITH SCATTER PLOTS (CONT.)

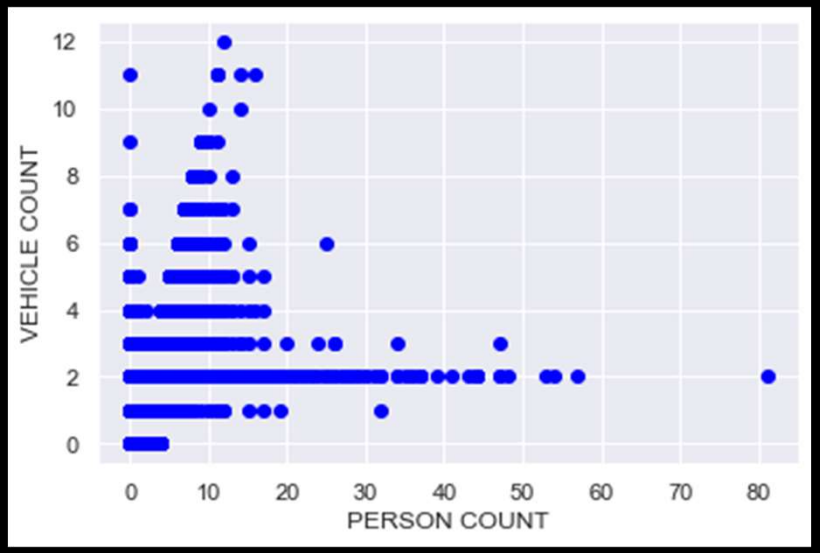
- We can then plot the fit line over the data:



EXAMINING THE POTENTIAL RELATIONSHIPS WITH SCATTER PLOTS

- The simple scatter plots have created for exploring about these two potential relationships.





MACHINE LEARNING MODELING

Creating train and test dataset

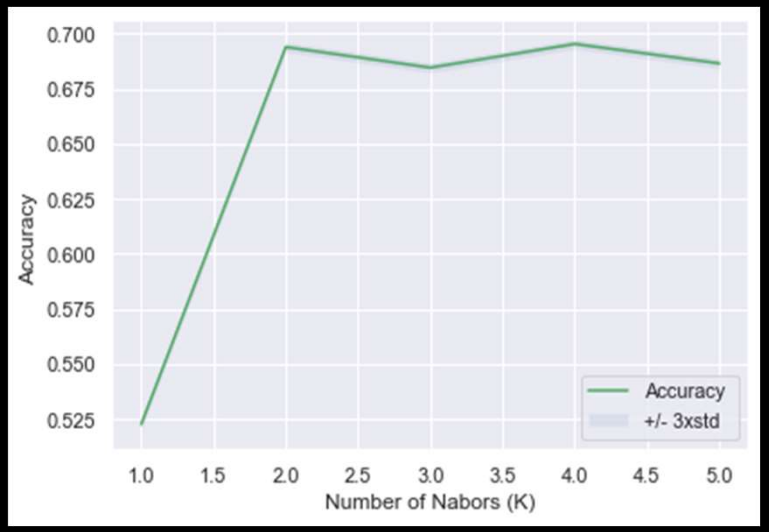
- We will split the dataset into train and test sets again, 80% of the entire data for training, and the 20% for testing.

```
: # Test/Train split
  from sklearn.model_selection import train_test_split
  X_train, X_test, y_train, y_test = train_test_split(X_df, y_df, test_size=0.2, random_state=4)
  y_train = y_train.ravel()
  print ('Train set:', X_train.shape,  y_train.shape)
  print ('Test set:', X_test.shape,  y_test.shape)

Train set: (155738, 6) (155738,)
Test set: (38935, 6) (38935, 1)
```

K NEAREST NEIGHBORS ANALYSIS

- To choose right value for K, we choose k =1 to calculate the accuracy of prediction using all samples in the test set and then increase the k to find which k is the best for the model.



K NEAREST NEIGHBORS ANALYSIS (CONT.)

- Continuing with K=4, we can train the data and evaluate the K Nearest Neighbours (KNN) using Jaccard score and F1 score

```
# Best accuracy was with k = 4
k = 4
knn = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train)

knn_y_pred = knn.predict(X_test)
knn_y_pred[0:5]
```

```
# KNN Evaluation
from sklearn.metrics import jaccard_score
from sklearn.metrics import f1_score

knn_js = jaccard_score(y_test, knn_y_pred)
knn_f1 = f1_score(y_test, knn_y_pred, average='macro')
print ('KNN - Jaccard_score:', knn_js)
print ('KNN - F1 score:', knn_f1)
```

```
KNN - Jaccard_score: 0.6927081984198938
KNN - F1 score: 0.4355822353605406
```

DECISION TREE ANALYSIS

- We will first create an instance of the decision tree, fit the data , make some predictions. Thus, we check the accuracy of our model

```
# Decision Tree
from sklearn.tree import DecisionTreeClassifier
collisionTree = DecisionTreeClassifier(criterion="entropy", max_depth = 4)

collisionTree.fit(X_train,y_train)
predTree = collisionTree.predict(X_test)
```

```
: # Decision Tree Evaluation
  tree_js = jaccard_score(y_test, predTree)
  tree_f1 = f1_score(y_test, predTree, average='macro')
  print ('Decision tree - Jaccard_score:', tree_js)
  print ('Decision tree - F1 score:', tree_f1)

Decision tree - Jaccard_score: 0.7043715004880053
Decision tree - F1 score: 0.41336032198083383
```

LOGISTIC REGRESSION ANALYSIS

- We fit the model with the train set, predict using the test set and try log loss for evaluation. Next, we try Jaccard index for accuracy evaluation and calculate the F1 scores for each label based on the precision and recall of that label.

```
: # Regression
  from sklearn.metrics import log_loss
  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_pred = LR.predict(X_test)
  LR_prob = LR.predict_proba(X_test)
  reg_ll = log_loss(y_test, LR_prob)
  print ('Regression - Log loss:', reg_ll)
```

```
Regression - Log loss: 0.5986679016584695
```

```
: # Regression evaluation
  reg_js = jaccard_score(y_test, LR_pred)
  reg_f1 = f1_score(y_test, LR_pred, average='macro')
  print ('Regression - Jaccard_score:', reg_js)
  print ('Regression - F1 score:', reg_f1)
```

```
Regression - Jaccard_score: 0.704227884516593
Regression - F1 score: 0.41348440648219276
```

DISCUSSION

Model evaluation

- Firstly, the Jaccard score is used for accuracy evaluation. The best model in this case is Decision tree (highest Jaccard score = 0.704371500488005 3).
- Secondly, the F1 score is the harmonic average of the precision and recall. The best model based on this score is the K Nearest Neighbours (highest F1 score: 0.435582235360540 6)

Model	K Nearest Neighbours	Decision Tree	Regression
Jaccard score	0.6927081984198938	0.7043715004880053	0.704227884516593
F1 score	0.4355822353605406	0.41336032198083383	0.4134844064821 9276
Log loss	N/A	N/A	0.5986679016584695

FINDINGS

- The majority of collisions causes injury and vehicle damage
- Most of the collision happened at certain specific address types: block and intersection.
- Most of the collisions can be happened in a good driving condition such as vision clear, day light and dry weather.
- When the collusions involved several vehicles, it may involve several persons.
- Decision tree model could be a good model to build a decision process to warn drivers the possibility of collisions.
- The K Nearest Neighbors model could be a good model to determine the patterns of collision to build an alert system.
- The Regression model could be an appropriate model to predict the severity of collisions in certain specific cases.

RECOMMENDATIONS

- **For the Public Development Authority of Seattle**
  - Based on the patterns of vehicle collisions, the safety signs should be installed at and the lighting / road conditions should be improved the corresponding locations.

- **For software developers:**
  - There is a need to develop a mobile application to display the potentiality and severity of vehicle collisions based on the patterns and predictions based on their locations. This application can integrate with GPS-based services (such as Google maps) to warn drivers at these locations or to suggest an alternative route if required.