Predicting Severity of Car Crashes

Balwinder Khakh

October 10, 2020

1.Introduction

1.1 Background

According to the National Safety Council (NSC) website, the estimated annual deaths due to car accidents was 38,800, with 4.4 million being injured enough to warrant medical attention. Considering the scale of this problem, various state and municipal agencies collect large amounts of data on these accidents. For example, the city of Seattle in the state of Washington has open, available data on various crashes that have occurred over the years. Using machine learning algorithms it is possible to analyze this data and find key common factors that can predict how severe a potential accident can be. From there, it can be possible to take preventative measures to minimize damage.

1.2 Problem

Using the data from Seattle, is it possible to predict crash severity based on universal factors not unique to the city specifically? For example, would it be possible to look at factors such as weather or road conditions, that can be found in any city, and build predictive models around them? This scope can allow for more universal solutions to improve traffic safety. This is the aim of the project.

1.3 Interest

This analysis can prove valuable not only to government agencies and the average driver, but to insurance companies, automobile manufacturers and any company with vested interest in transportation and logistics.

2. Data Acquisition and Cleaning

2.1 Data Sources

The data used in this project is hosted on https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv. This data was recorded by the Seattle Police Department from 2004 to the latest of May, 2020. It is a csv file containing 38 columns and 194,673 rows. It contains various information such as weather, date, location of each accident, etc.

SEVERITYCODE	int64
X	float64
Y	float64
OBJECTID	int64
INCKEY	int64
COLDETKEY	int64
REPORTNO	object
STATUS	object
ADDRTYPE	object
INTKEY	float64
LOCATION	object
EXCEPTRSNCODE	object
EXCEPTRSNDESC	object
SEVERITYCODE.1	int64
SEVERITYDESC	object
COLLISIONTYPE	object
PERSONCOUNT	int64
PEDCOUNT	int64
PEDCYLCOUNT	int64
VEHCOUNT	int64
INCDATE	object
INCDTTM	object
JUNCTIONTYPE	object
SDOT_COLCODE	int64
SDOT_COLDESC	object
INATTENTIONIND	object
UNDERINFL	object
WEATHER	object
ROADCOND	object
LIGHTCOND	object
PEDROWNOTGRNT	object
SDOTCOLNUM	float64
SPEEDING	object

ST_COLCODE	object
ST_COLDESC	object
SEGLANEKEY	int64
CROSSWALKKEY	int64
HITPARKEDCAR	object
dtype: object	

2.2 Feature Selection

Considering the number of columns, the selection of the data features to be used becomes key. As stated in our Problem, the goal is to use more of the universal factors in this analysis. Data specific to the incident, such as whether a parked car was hit (HITPARKEDCAR) or detail information such as where the incident occurred (LOCATION) will be dropped. This leaves us with a few selected features.

```
SEVERITYCODE       0
SEVERITYDESC       0
INCDATE            0
INCDTTM            0
WEATHER         5081
ROADCOND        5012
LIGHTCOND       5170
dtype: int64
```

Illustration 1: Count of nullvalues for the selected data features

SEVERITYCODE is either 1 or 2, which described by SEVERITYDESC as being either a property only collision (1) or one that resulted in an injury (2). INCDATE/INCDTTM is incident date and time. WEATHER indicates the weather conditions at the time of the incident with 11 conditions recorded. ROADCOND is condition of the road with 9 conditions and similarly, LIGHTCOND is the light conditions with 9 as well.

2.3 Data Cleaning

As indicated in Illustration 1, the condition features have thousands of null values in the csv, with the highest being 5170. Considering the total of 194,673, this is 2% reduction of total data. The decision was made to simply drop these values out of the

data frame. Before being ran through the machine algorithms, these variables had to be label encoded with numbers as well. This was done using the LabelEncoder from the Sklearn package.

The values of the INCDATE and INCDTTM were not used directly. Values were converted to datetime format for Python from which the weekday and hour of the incident was obtained and put into separate columns. This was to see if the days of the week and hour of the day had significance, since these can be universal factors as well to traffic safety.

3.Exploratory Data Analysis

3.1 Target Variable

The target variable that was used in the analysis was SEVERITYCODE. With a value of either 1 or 2, there was no need to further encode this variable. The conditions variables can be compared with how many counts of incidents fall into these 2 codes.

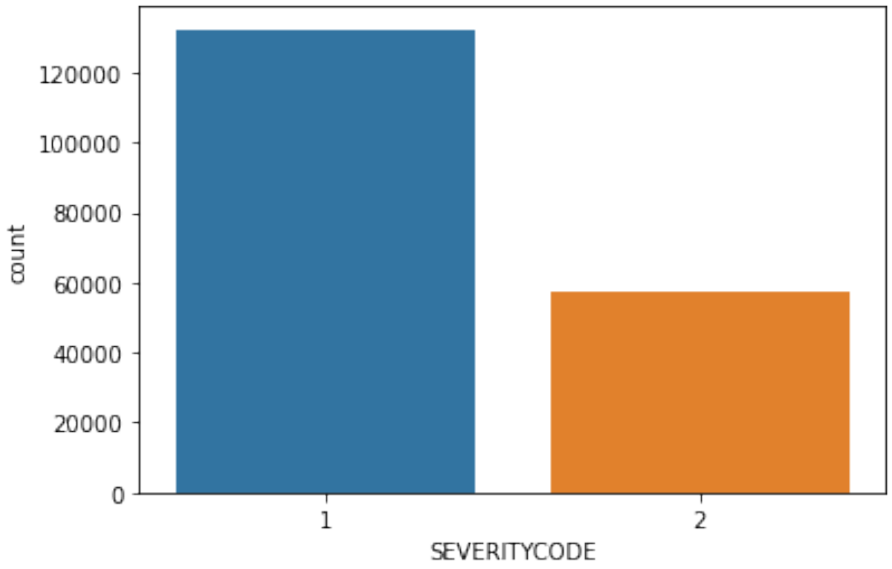

Illustration 2: Total counts of crashes categorized by severity

As seen in Illustration 2, the overall data has more incidents that resulted in just property damage instead of any injury. This is a factor to note when working with the proportionality of the data.

3.2 Relationships Between Weather, Road and Light Conditions

In the data exploration before using the machine algorithms, count plots using the Seaborn package were produced to see if any remarkable trends were visible.

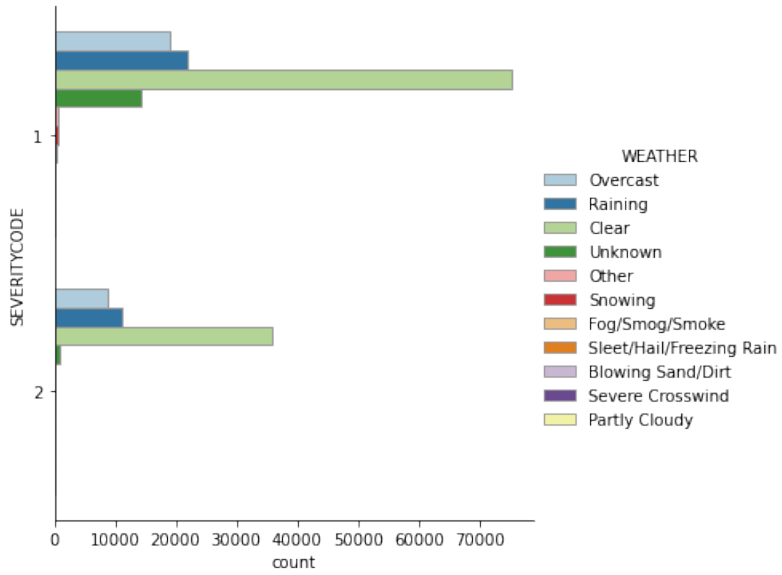

Illustration 3: Count of crash severity by weather conditions

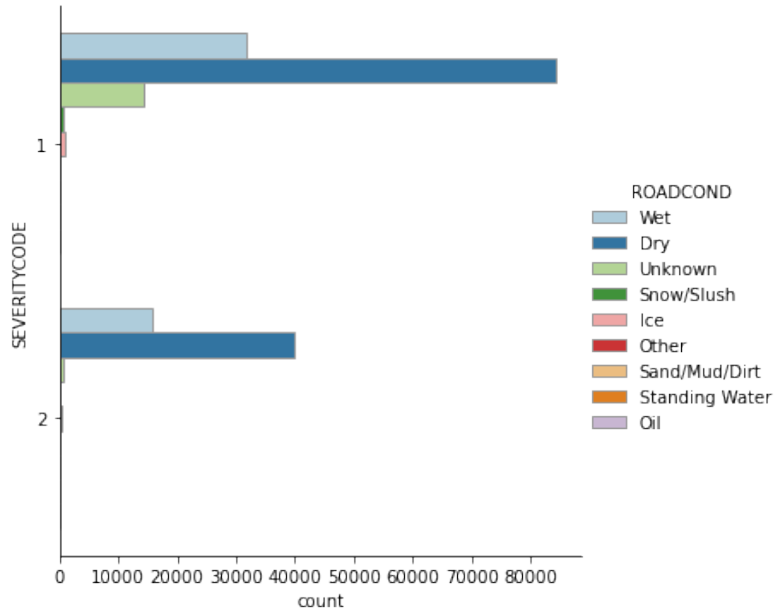

Illustration 4: Count of crash severity by road conditions

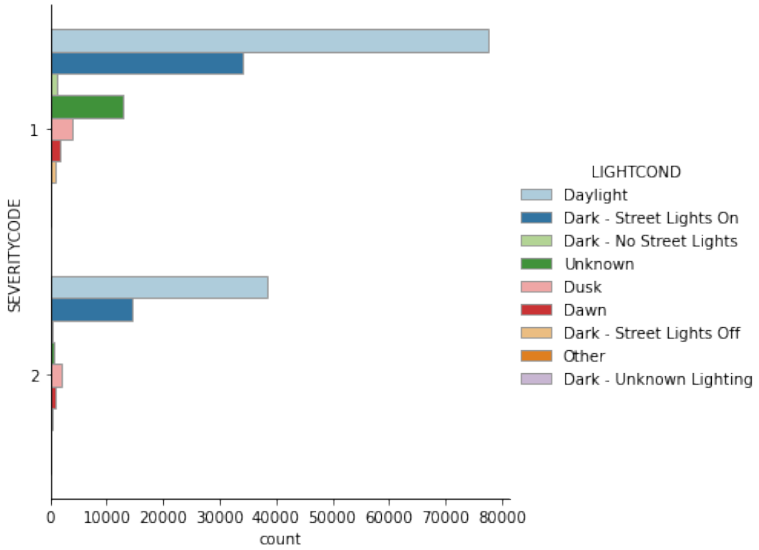

Illustration 5: Count of crash severity by light conditions

In Illustration 3, the overall largest category of crashes was property damage only, under clear conditions. Proportionally, clear skies was the majority condition for both severity codes. Similarly in Illustrations 4 and 5, this is the same pattern for dry roads and daylight. So the bulk of overall incidents occurred in dry roads in the daytime with clear conditions and were mostly property damage only. The crashes involving injuries follow the same trends. In all three of these conditions, the impact of extreme conditions appears very minimal.

3.3 Relationships Between Day of the Week and Hour

After formatting the incident time data, a count analysis was done for the days of the week and hour of the day.

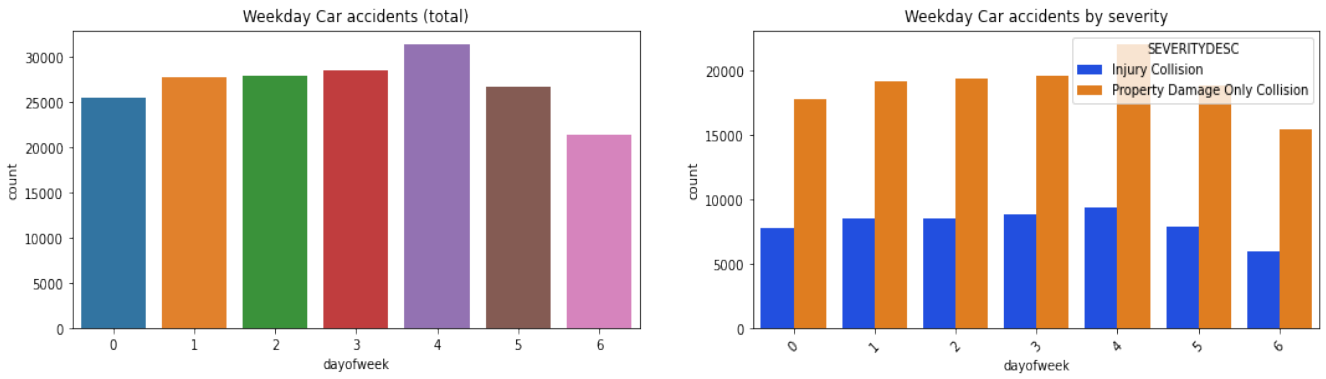

Illustration 6: Count plot of total crashes on each day of the week and by severity

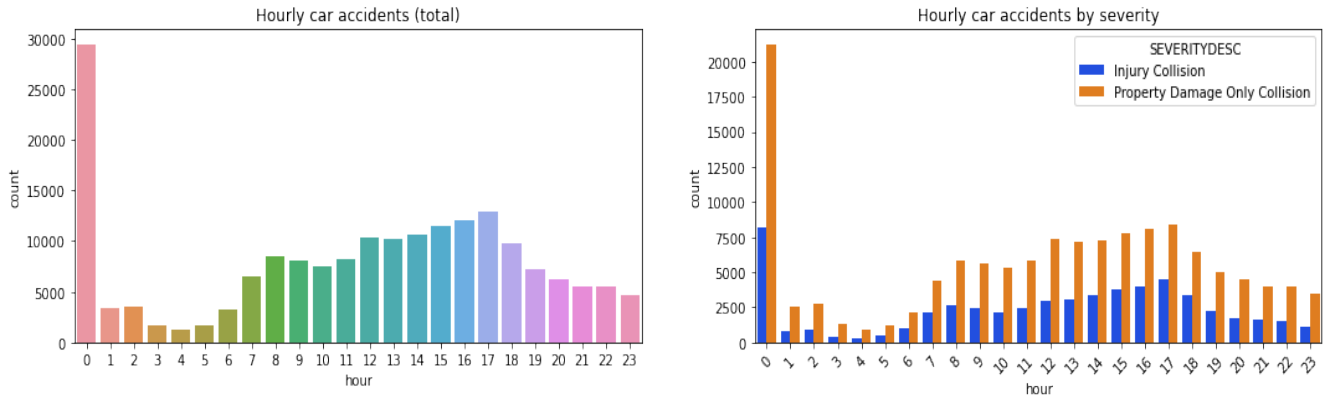

Illustration 7: Count plot of total crashes on each hour and by severity

In Illustration 6, the days of the week are encoded from 0-6, from Monday to Sunday. The amount of accidents seems to be similar every day with a slight increase on 4, which is Friday. In Illustration 7, the hours are set from 0-23, with is 12am to 11pm. There is a clear outlier in hour 0 which is midnight. It is the hour with the single highest amount of crashes compared to others. The majority of those crashes are property damage only but even the injury collisions are higher than every other hour.

4. Predictive Modeling

4.1 Models Used and Results

The nature of the data features used in this analysis is categorical and discrete. The goal is to predict whether how severe of a crash in the category of 1 or 2 will result from the selected features. As such, the algorithms selected were K Nearest Neighbor, Logistic Regression and Decision Tree.

```
[15]: from sklearn.metrics import jaccard_similarity_score
      from sklearn.metrics import f1_score
      from sklearn.metrics import log_loss
      from sklearn.metrics import precision_score

      from sklearn.neighbors import KNeighborsClassifier
      knn = KNeighborsClassifier(n_neighbors = 8).fit(X_train, y_train)
      knn

[15]: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                          metric_params=None, n_jobs=None, n_neighbors=8, p=2,
                          weights='uniform')
```

Illustration 8: Code for the KNN algorithm

```
[16]: from sklearn.linear_model import LogisticRegression
      from sklearn.metrics import confusion_matrix
      lr = LogisticRegression(C=0.0001, solver='liblinear')
      lr.fit(X_train, y_train)
      lr

[16]: LogisticRegression(C=0.0001, class_weight=None, dual=False,
                         fit_intercept=True, intercept_scaling=1, max_iter=100,
                         multi_class='warn', n_jobs=None, penalty='l2', random_state=None,
                         solver='liblinear', tol=0.0001, verbose=0, warm_start=False)
```

Illustration 9: Code for the Logistic Regression algorithm

```
[17]: from sklearn.tree import DecisionTreeClassifier

      tree = DecisionTreeClassifier(criterion="entropy", max_depth = 4)
      tree.fit(X_train, y_train)
      tree

[17]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=4,
                             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')
```

Illustration 10: Code for the Decision Tree algorithm

The illustrations shown above give the code and parameters of each algorithm. The data was split into train/test pairs and normalized before being run into the algorithms.

	Algorithm	Jaccard	Precision
0	KNN	0.68	0.6
1	Decision Tree	0.7	0.49
2	Logistic Regression	0.7	0.49

Illustration 11: Jaccard score and precision of algorithms

```
Train set KNN Accuracy:  0.6834421096314182
Test set KNN Accuracy:  0.6789725713883314
Train set Decission Tree Accuracy:  0.6990530803184064
Test set Decission Tree Accuracy:  0.6977923312559416
Train set Logistic regression Accuracy:  0.6990530803184064
Test set Logistic regression Accuracy:  0.6977923312559416
```

Illustration 12: Overall Accuracy of the algorithms

Illustration 11 gives the Jaccard score and precision of the models while Illustration 12 gives the overall accuracy. Both Decision Tree and Logistic Regression appear to be equally more accurate than KNN. Both models are about 70% accurate but the precision is lower than KNN.

5. Conclusions

The overall models resulting from this data seem to be adequate considering the nature of realistic data. The insight from the exploratory analysis reveals that, at least in the case of Seattle, the majority of accidents are property damage that occur under ideal conditions in the day time. The outlier to that trend is crashes occurring at midnight. An interpretation of this trend is that abnormal conditions are not the major cause of collisions, meaning drivers may be more alert at these times overall. Any solution should focus on what occurs during optimal normal driving conditions, where there are more likely to be more drivers on the road overall.

6. Future Directions

The techniques outlined and utilized in this report can be applied to other cities. Model accuracy would benefit from having more factors. Additionally, having an idea of

overall traffic might give more predictive value to how likely a crash might even occur, rather than just the severity of possible crashes. If the results found in this report apply to other cities, local government might want to focus more effort in accident reduction during peak or normal driving times.