PREDICTING VEHICLE ACCIDENT SEVERITY IN SEATTLE, WASHINGTON

IBM Python Applied Data Science Capstone

The goal of this project is to use vehicle collision data recorded in the city of Seattle to predict whether an accident would cause property damage or personal injury.

By Mohit Mhapuskar
mmhapusk@stevens.edu
linkedin.com/in/mmhapusk
github.com/mmhapusk

TABLE OF CONTENTS

Introduction: Business Understanding……………………………….3

- **Background……………………………………………………………….……3**
- **Solution…………………………………………………………………………3**

The Data………………………………………………………………………4

- **Data Acquisition…………………………………………………………4**
- **Data Cleaning and Preprocessing………………………………………..4**
- **Feature Selection.………………………………………………………..8**

Methodology…………………………………………………………….9

Analysis……………………………………………………………………..10

- **Exploratory Data Analysis…………………………………………..10**
- **Data Modeling………………………………………………………..19**
- **Model Evaluation……………………………………………………22**

Results and Recommendations…………………………………26

Conclusion…………………………………………………………………..28

Introduction: Business Understanding

Background

As we all know, Seattle is a major seaport city on the West Coast of the United States in the northwestern state of Washington. Seattle is the largest city in both the state of Washington and the Pacific Northwest region of North America. According to U.S. Census data released in 2019, the Seattle metropolitan area's population stands at 3.98 million, making it the 15th-largest in the United States. As of July 2016, Seattle was the fastest-growing major U.S. city, with a 3.1% annual growth rate.

Like any other fast-growing city in America, an increase in development coincides with an increase in population which in turn leads to more and more automobiles traversing through the city. As of 2016, the number of vehicles in Seattle hit a new high of 446,000. More automobiles mean a higher number of vehicle accidents take place. In 2019 alone, there were 10,315 total car crashes within the city, with about 2,612 of those crashes resulting in injuries and 22 proving fatal.

Solution

As you might have guessed then, our goal for this project is to analyze vehicle accident data from the city of Seattle and apply Machine Learning techniques to determine which attributes and conditions (for example, road condition and weather) can lead to an accident that just causes property damage or can lead to something more serious like a personal injury.

The findings and results of this project can then be used by the concerned stakeholders such as the Seattle Department of Transportation (SDOT) or the Seattle Police Department (SPD) to take improved measures in order to reduce the number of fatal accidents as well as total accidents overall.

The Data

1. Data Acquisition

The dataset that we will be using for this project is a collection of records of vehicle collisions that have taken place in Seattle city proper between the years 2004-present. The data has been recorded by the Seattle Police Department (SPD) and compiled together by the Seattle Department of Transportation (SDOT). The dataset itself can be found here, while the metadata for the dataset can be found here. The dataset contains a total of 194,673 records and 37 columns. The dataset will undergo extensive cleaning and engineering before it can be used for training our models.

The target variable we will be trying to predict is Accident Severity (represented in the dataset as **SEVERITYCODE**), which is a binary variable that classifies whether a particular accident involved Property damage only or Injury to a person as well.

2. Data Cleaning and Preprocessing

The dataset was relatively raw and had to undergo a lot of cleaning, preprocessing and transformation before it could be used for the initial exploratory data analysis and eventually for modeling purposes.

2.1 Dropping Features

Firstly, there was the issue of irrelevant features. As mentioned before, our dataset consisted of about 37 columns, including our target variable. A lot of these attributes are either redundant or irrelevant in terms of modeling purposes and do not contribute towards predicting our target variable. Hence, we dropped these unnecessary variables.

Here is a list of variables we dropped from our dataset:

1. **KEYS (OBJECTID, COLDETKEY, REPORTNO, INTKEY, SEGLANEKEY, CROSSWALKKEY, EXCEPTRSNCODE, EXCEPTRSNDESC, SDOTCOLNUM) :** These columns are keys or

numbers that have been assigned by SDOT to uniquely identify certain records, incidents and locations. This data is irrelevant and not useful for modeling. We will only keep INCKEY as our unique identifier for our records.

2. **STATUS :** Irrelevant variable.
3. **INCDATE:** As the date of the incident is already present in INCDTTM, this is a redundant variable and therefore safe to drop.
4. **LOCATION:** The street addresses of the accidents. As we already have Latitude and Longitude co-ordinates which we can use to plot on a map, this variable is redundant and therefore can be dropped.
5. **ST_COLDESC, ST_COLCODE:** There are two collision code sets in the data to identify collisions, ST codes and SDOT codes. Since we are using the SDOT codes to identify the types of collisions in our EDA, we can drop the ST codes.
6. **VEHCOUNT, PERSONCOUNT, PEDCYLCOUNT, PEDCOUNT, PEDROWNOTGRNT, HITPARKEDCAR:** These attributes are descriptive statistics recorded after the accident has already taken place. Since we are more interested in the features which describe the conditions preceding the accident, these attributes are not that useful for modeling or EDA.

2.2 Data Transformations

Secondly, there was the issue of transforming the attributes into the correct formatting to make it easier for analysis. Our target variable of **SEVERITYCODE** is a binary class categorical variable that was initially encoded as ‘1’ (Property Damage Only) and ‘2’ (Personal Injury). This was changed to ‘0’ (Property Damage Only) and ‘1’ (Personal Injury), also known as one-hot encoding. This was done primarily because one-hot encoding is easier to interpret for the Machine Learning algorithms in Python’s scikit-learn library. In a similar manner we performed one-hot encoding for the following variables:

1. **UNDERINFL:** This variable denotes whether the driver was under the influence of alcohol or drugs. Initially encoded as ‘Y’ (positive) or a missing value (negative). One-hot encoded to ‘0’ (negative) and ‘1’ (positive).

2. **INATTENTIONIND:** Variable denotes whether the driver was inattentive or not. Initially encoded as ‘Y’ (inattentive) or a missing value (attentive). One-hot encoded to ‘0’ (attentive) and ‘1’ (inattentive).

3. **SPEEDING:** Variable denotes whether the driver was over the speed limit or not. Initially encoded as ‘Y’ (over speeding) or a missing value (not speeding). One-hot encoded to ‘0’ (not speeding) and ‘1’ (over speeding).

The variables **ROADCOND** (Road Condition), **LIGHTCOND** (Lighting Conditions) and **WEATHER** had a lot of values which were simply labelled ‘Unknown’. This is as good as a missing value and was therefore replaced by a blank value, to be dealt with later when we began dealing with missing values.

The variable **INCDTTM** (Incident Date Time) contained a timestamp for each accident, consisting of Year, Month and Day of Week. We would use the to_datetime() function in the pandas library to extract the year, month and day from the timestamp into separate columns. This would make it easier for us to analyze the data during EDA.

The variables **X** and **Y** which represent our geospatial coordinates were also renamed as **LONGITUDE** and **LATITUDE** respectively for our convenience.

2.3 Dealing with Missing Values

Thirdly, there was the issue of missing values in our dataset. Generally there are three ways to deal with missing values in data: (i) Drop the rows entirely, (ii) Substitute those values on the basis of other values in that column or (iii) Leave them as is (generally not recommended). After compiling a list of missing values for each attribute in our dataset, this is how we dealt with each of them.

1. **LONGITUDE, LATITUDE:** There were 5334 missing values for longitude and latitude. However, since this is geospatial data, substituting the missing values is overly complex and would have required the use of some sort of geography-based libraries. Since it was a negligible portion of the data, it simply made more sense to drop these rows.

2. **JUNCTIONTYPE:** There were 4199 missing values for junction type. Since this attribute is also location-based, it would not make logical sense to substitute the data. Like before, these rows also represented a miniscule portion of the data, so we just dropped these rows.

3. **ROADCOND, LIGHTCOND, WEATHER:** There were roughly 18000 missing values for these 3 attributes. Since these 3 are multi-class categorical variables, it made sense to replace the missing values on the basis of frequency of occurrence of the classes (provided there was a clear winner in each case). For Road condition, the most occurring class was ‘Dry’ and therefore the missing values were replaced with class ‘Dry’. In a similar manner, the missing values in Light Condition were replaced with class ‘Daylight’ and the missing values in Weather were replaced by class ‘Clear’.

4. **COLLISIONTYPE:** There were 4757 missing values for collision type. Since this is also a multi-class categorical variable, it made sense to replace the missing values on the basis of frequency of occurrence of each of the classes. However, there was no clear winner in this case, with the frequency of classes being fairly distributed. Since there already existed a collision class labeled ‘Other’, we substituted the missing values with ‘Other’.

3. Feature Selection

After data preprocessing, this was the final list of features that were selected with which we could begin EDA.

Feature	Data Type	Description
Year	Date, Integer	Year of accident
Month	Date, Integer	Month of accident
Day of Week	Date, Integer	Day of accident
ADDRTYPE	Multi-class Object, String	Type of Address (Block, Intersection)
JUNCTIONTYPE	Multi-class Object, String	Type of Junction (Mid-Block, Driveway junction etc.)
LONGITUDE	Geodata, Float	Longitudinal Coordinates of Accident
LATITUDE	Geodata, Float	Latitudinal Coordinates of Accident
COLLISIONTYPE	Multi-class Object, String	Type of Collision (Angle, Rear Ended etc.)
WEATHER	Multi-class Object, String	Weather (Clear, Raining etc.)
ROADCOND	Multi-class Object, String	Road Condition(Dry, Wet etc.)
LIGHTCOND	Multi-class Object, String	Lighting Condition (Daylight, Dark etc.)
UNDERINFL	Binary-class Object, Integer	Whether driver was under the influence(0,1)
SPEEDING	Binary-class Object, Integer	Whether driver was over the speed limit (0,1)
INATTENTIONIND	Binary-class Object, Integer	Whether driver was inattentive (0,1)
SEVERITYCODE	Binary-class Object, Integer, Target Variable	Whether accident caused only property damage or Injury (0,1)

We selected these 15 attributes from our initial 37 to proceed forward with our Exploratory Data Analysis for now. We applied some further transformations and reductions down the line during the Data modeling phase to make the data suitable for training and testing purposes.

Methodology

Now that we have preprocessed our data, we can begin our analysis. There are three major steps that will be involved in our analysis:

1. **Exploratory Data Analysis (EDA):** We started off by carrying out some EDA to further understand our data and try to find some useful correlations between our independent variables and our target variable of Accident Severity. We will create some visualizations of our data such as charts and map visualizations.

2. **Data Modeling:** After EDA, we begin building our Predictive models by training our data to predict our target variable of Accident Severity. Since our target variable is a categorical variable, we will be applying classification algorithms to train our models. More specifically, we will be applying three Machine Learning algorithms:

    - **Decision Trees**
    - **K-Nearest-Neighbor**
    - **Logistic Regression**

We will discuss these models a bit more in detail during our Analysis phase.

3. **Model Evaluation:** Finally, we will evaluate each model based on its accuracy in predicting the target variable and recommend the best one going forward. We will be using 3 metrics to determine the accuracy of the models:

    - **F-1 Score**
    - **Jaccard's Similarity Index**
    - **Log Loss (Logistic Regression Only)**

With that done, let us move forward onto our analysis.

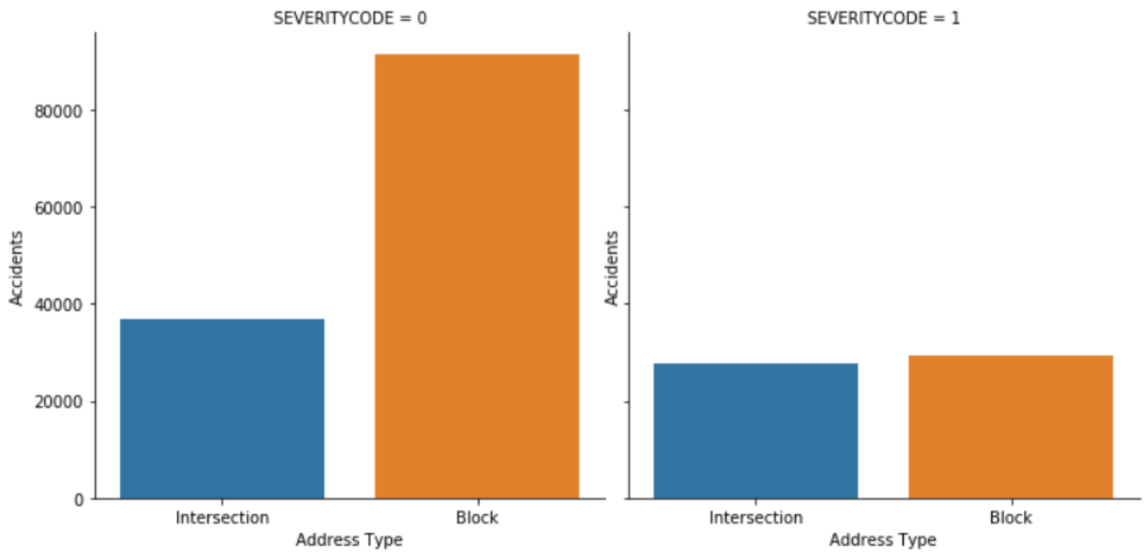
Analysis

1. Exploratory Data Analysis (EDA)

In this phase, we will create some visualizations to understand our data more and understand correlations between our independent variables and our target variable of accident severity. Since most of our independent variables within the feature set are categorical variables, we can use Seaborn's catplot() function to perform most of the plotting.

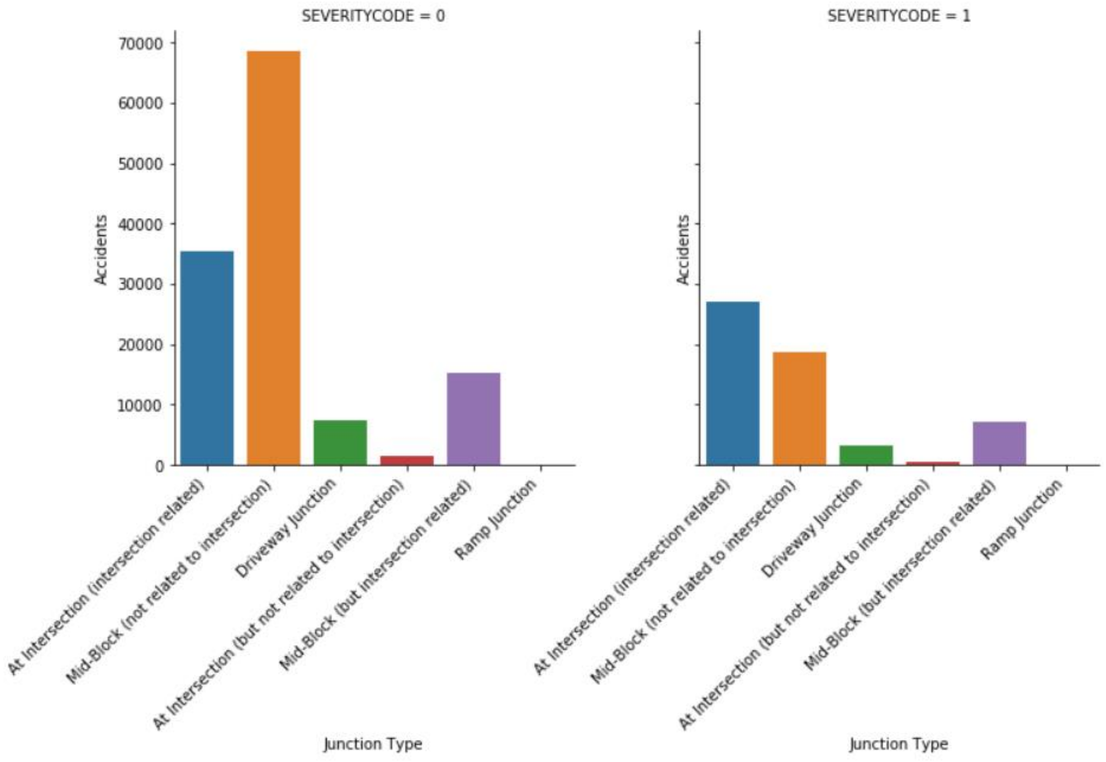
Let us start off with a catplot of the two address types where the accidents have taken place, and then split it based on whether there was Property Damage only or Injury to a person also occurred.

As a reminder, for SEVERITYCODE, 0 = Property Damage Only and 1 = Injury to Person.



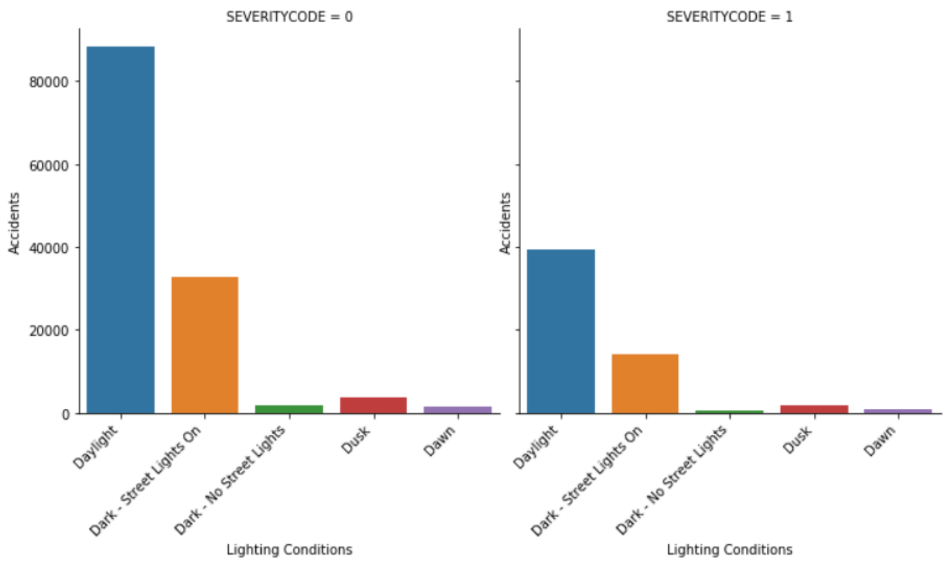
From the above distribution its immediately apparent that property damage was more prevalent at a block rather than an intersection, whereas accidents where injuries took place seem to be more evenly distributed across blocks and intersections. This makes logical sense considering that there is a lot more happening at an intersection, with pedestrian crossings, oncoming vehicles, signals and stop signs, which increase the likelihood that an accident at an intersection would involve injury.

Let us dive a bit deeper into this by plotting JUNCTIONTYPE next.



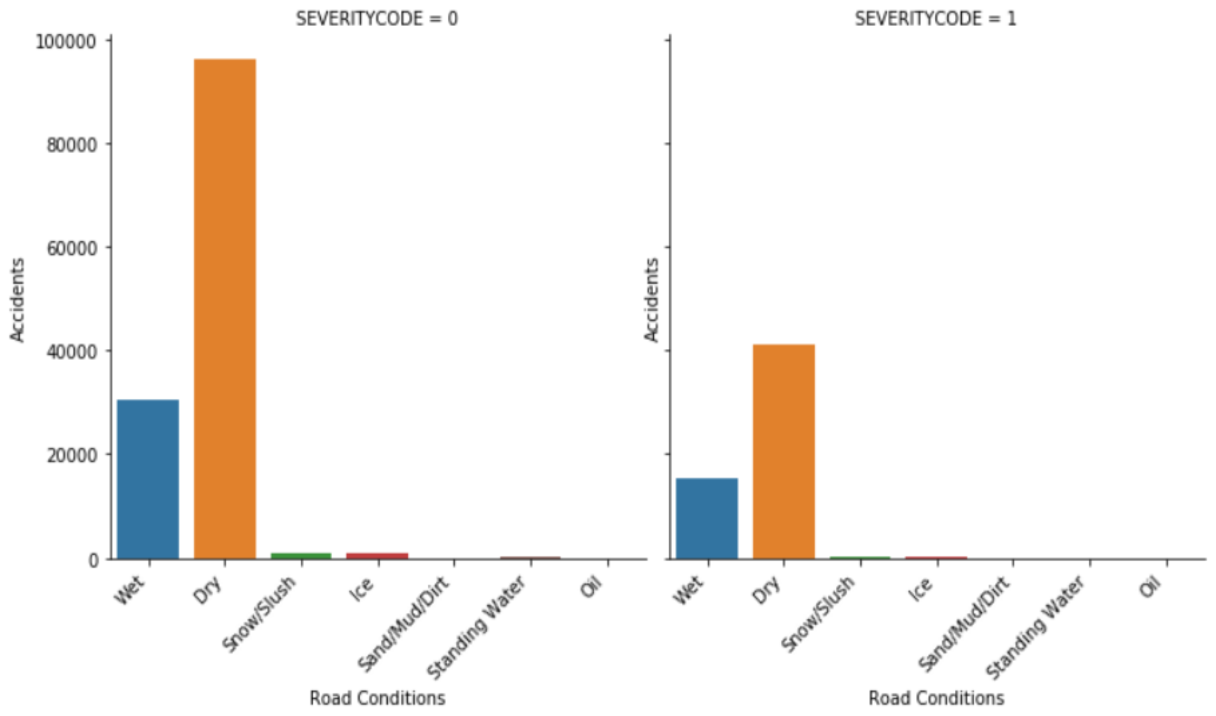
This plot does seem to add more evidence to our earlier observation about property damage taking place on blocks and injurious damage happening more on intersections. This means that the above two variables are likely good indicators by themselves for classifying accident severity.

Let us plot the Lighting conditions, LIGHTCOND next.



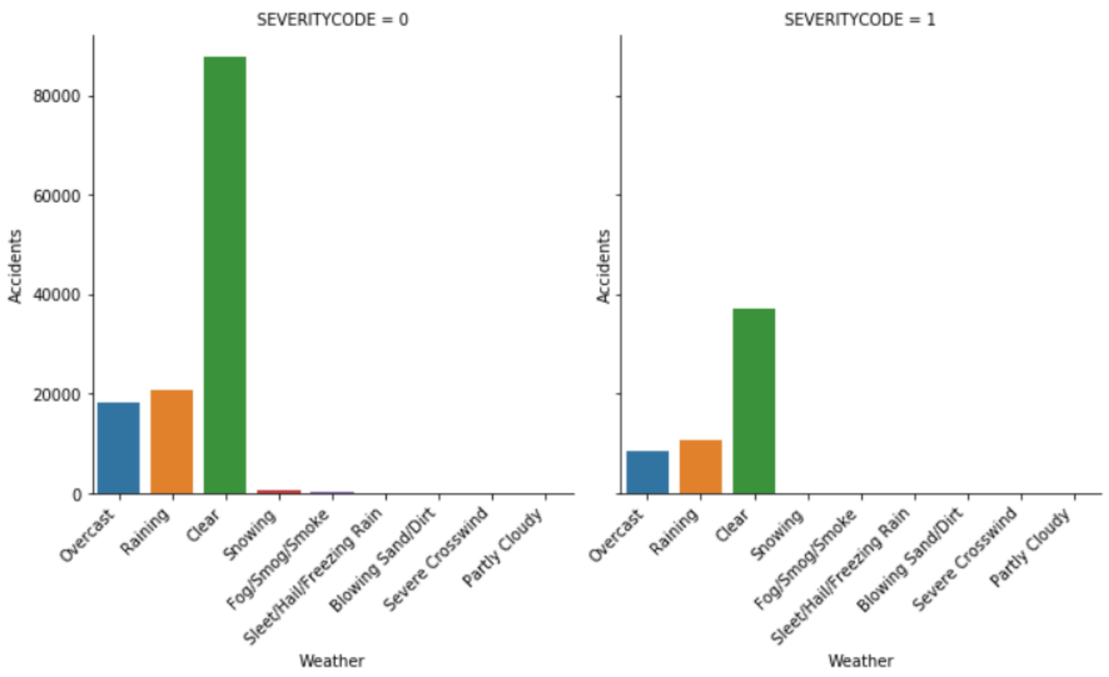
It seems that most accidents, regardless of property damage or injury took place in Daylight. Again, this makes logical sense, considering that most commuters are driving during the daytime. On its own, this variable may not be that useful in predicting accident severity, but in combination with other variables it may prove useful.

Let us plot the Road conditions, ROADCOND next.



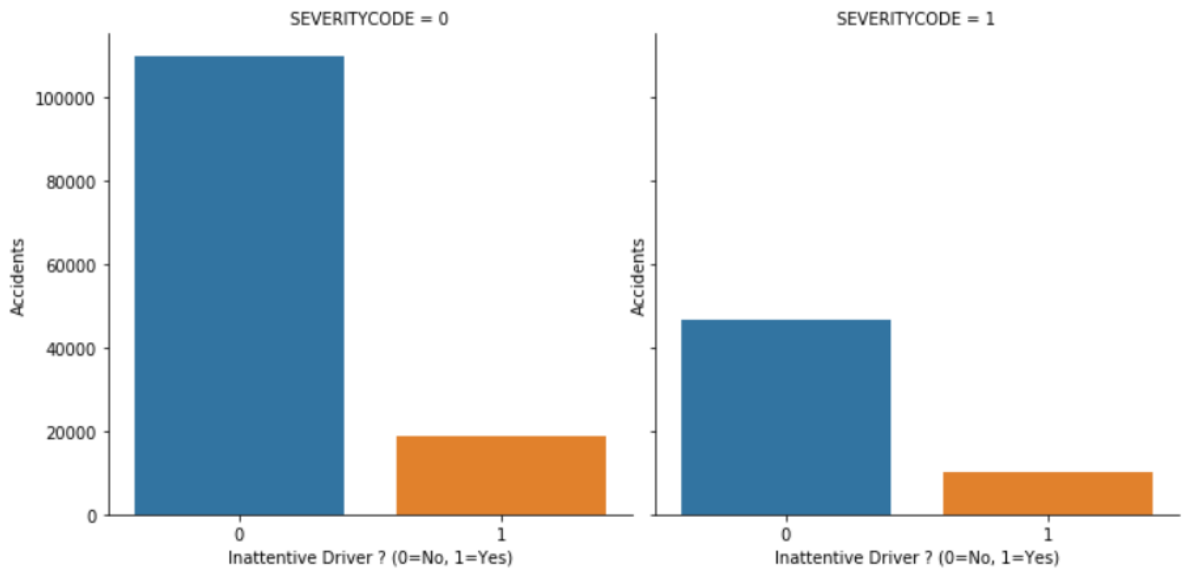
Most of the accidents took place when the road was dry, indicating that the road condition on its own, may not play a huge role in an accident. However, it may play a bigger role in combination with other variables.

Let us plot WEATHER next.



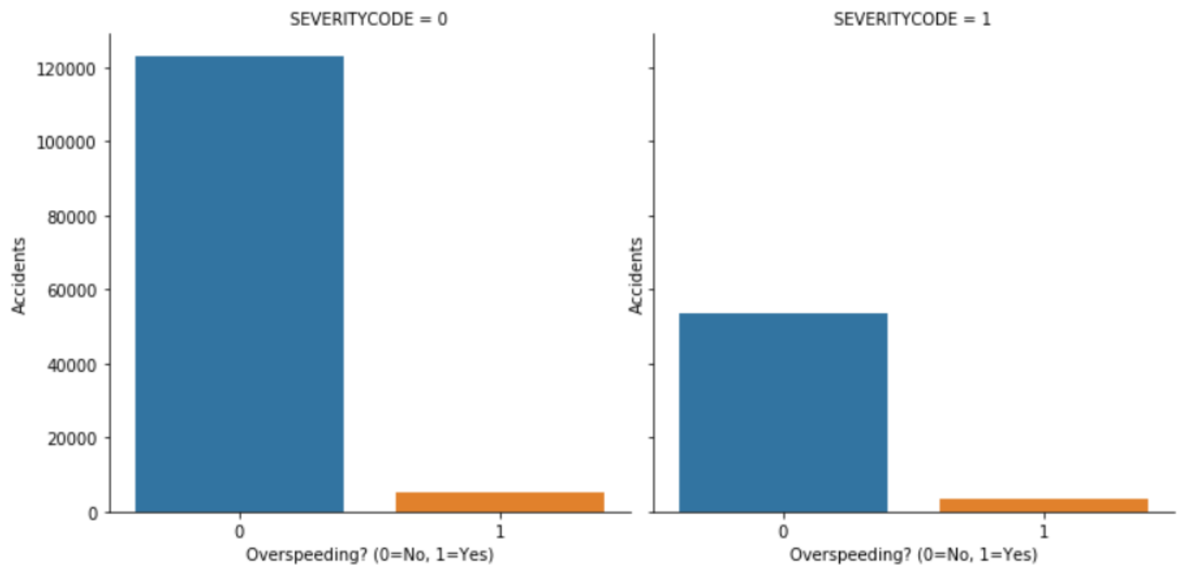
Most of the accidents took place in Clear weather. Although for accidents where injury was involved, non-dry weather conditions seem slightly more prevalent.

Let us plot whether a driver was inattentive, INATTENTIONIND next.



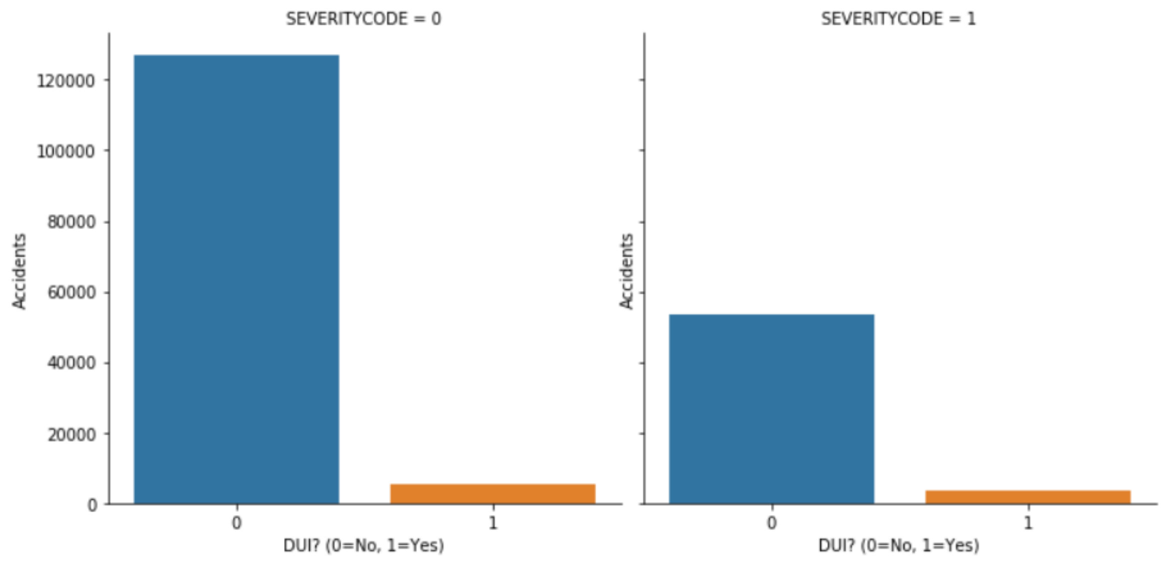
From the distributions, it seems that driver inattention seems to be more prevalent in accidents where injury was involved and less prevalent when only property was damaged.

Let us plot whether the driver was over the speed limit, SPEEDING next.



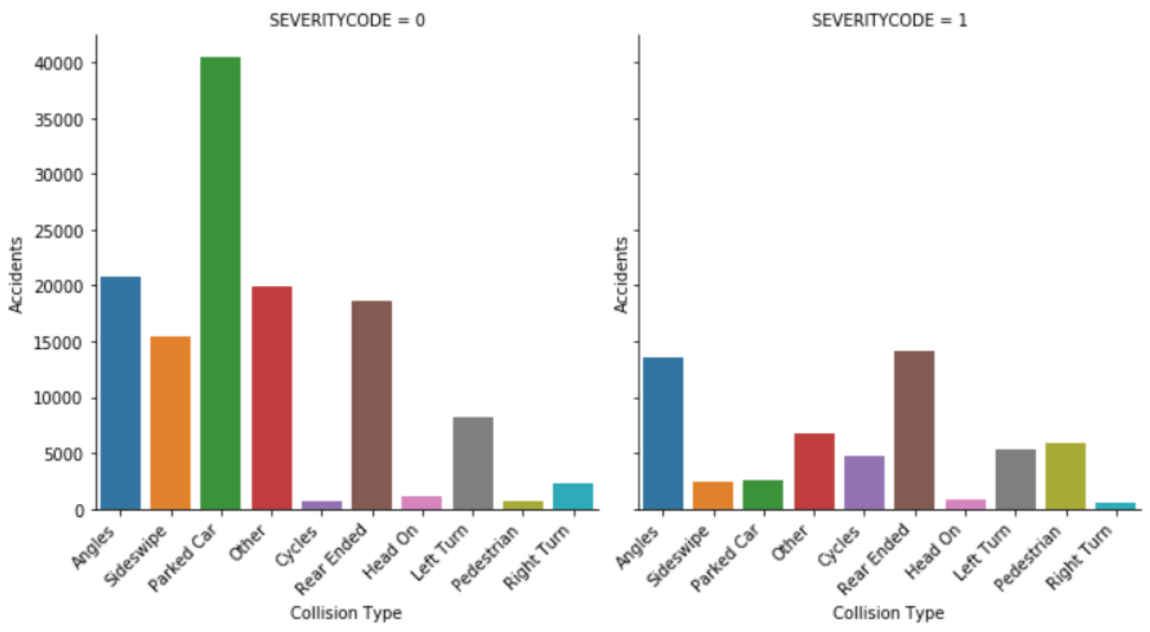
It seems that over speeding does not seem to be a major factor in accidents according to the data. Logically, this could be because the data is mainly from accidents in the area city proper, where roads are more crowded, and it is really difficult to drive fast.

Let us plot whether the driver was under the influence, UNDERINFL next.



Like over speeding before, DUI also does not seem to be a major indicator of an accident on its own.

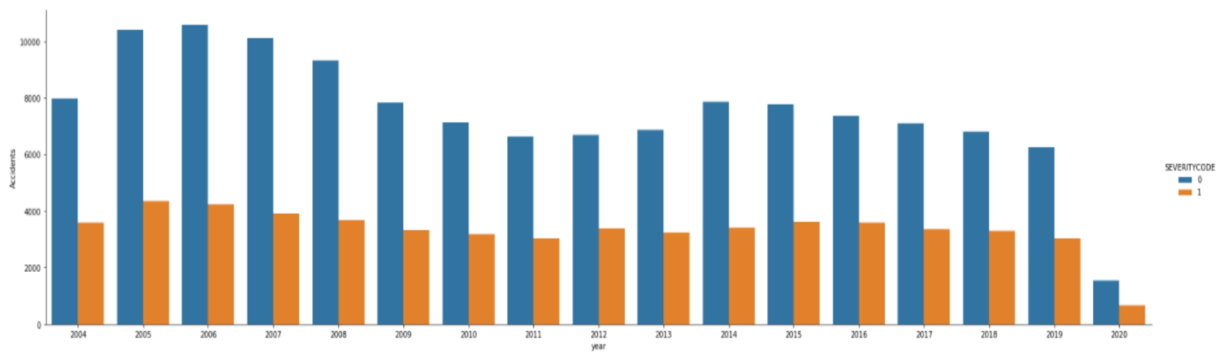
Let us plot COLLISIONTYPE next.



Immediately, collision type seems to be a much better classifier of the type of accident. The most common collision that involved property damage was collision with a parked car, followed by Angles. Logically, this makes sense since cars are parked all over city roads and blocks and drivers are more likely to hit a parked car in any accident involving property damage.

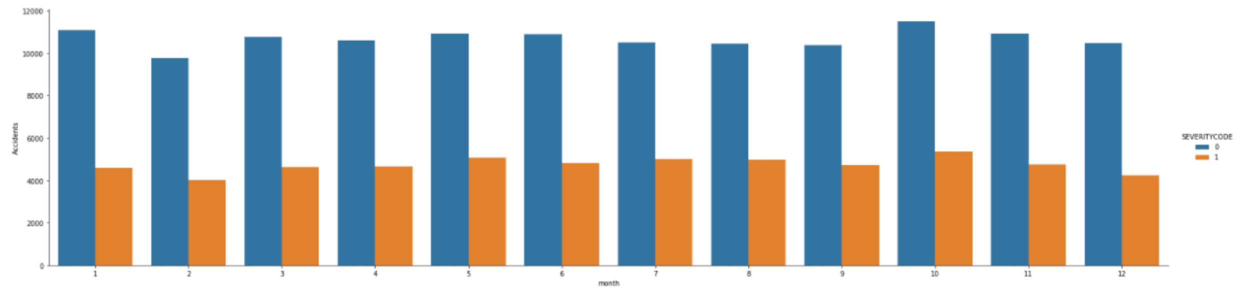
On the other hand, the most common collision where injury was involved seems to be Rear Ended, also followed by angles in second and pedestrian collisions in third. Rear ending on an open road is more likely to cause injury as compared to other collision types.

Let us plot the number of accidents distributed by year next.



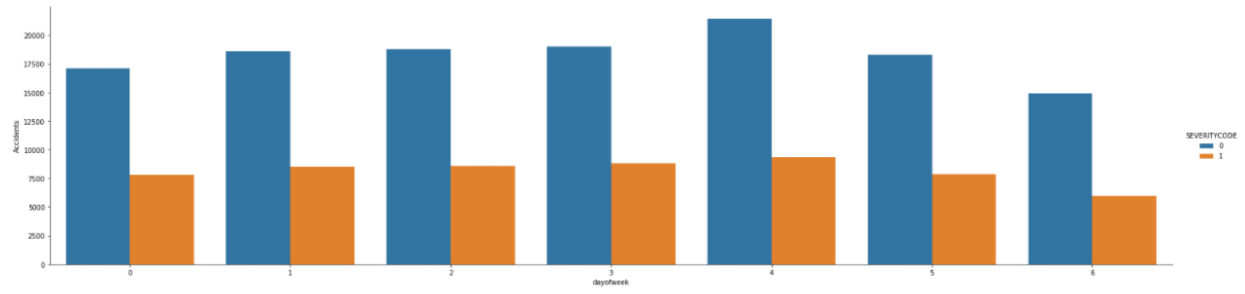
Seems like accidents peaked in 2005-2006 before following a downward trend, followed by another rise in 2014-2015. Since then there does seem to be downward trend again, this could be because of improvements in road conditions, sign and signal installations, better traffic management or rising use of public transport.

Let us plot the number of accidents distributed by month next.



Overall, there does not seem to be a significant pattern here with most of the accidents being distributed quite evenly.

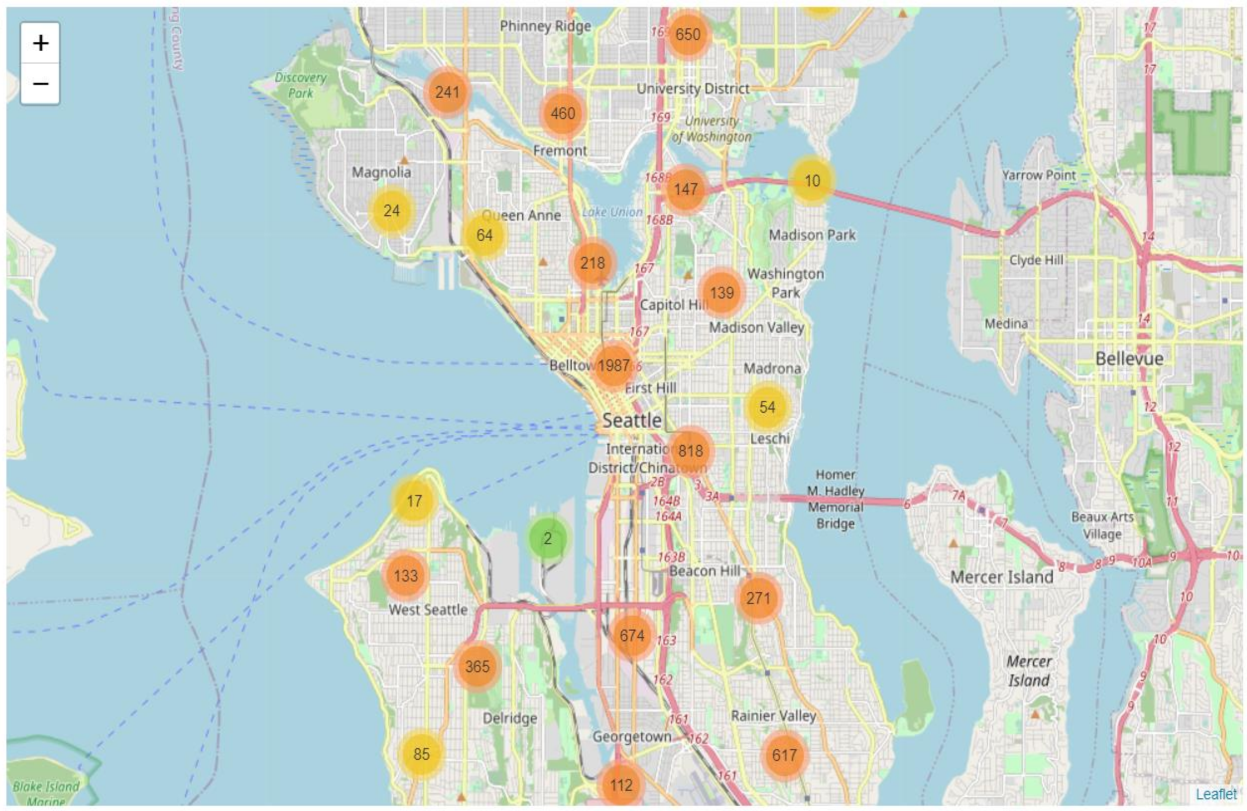
Let us plot the accidents distributed by day of the week.



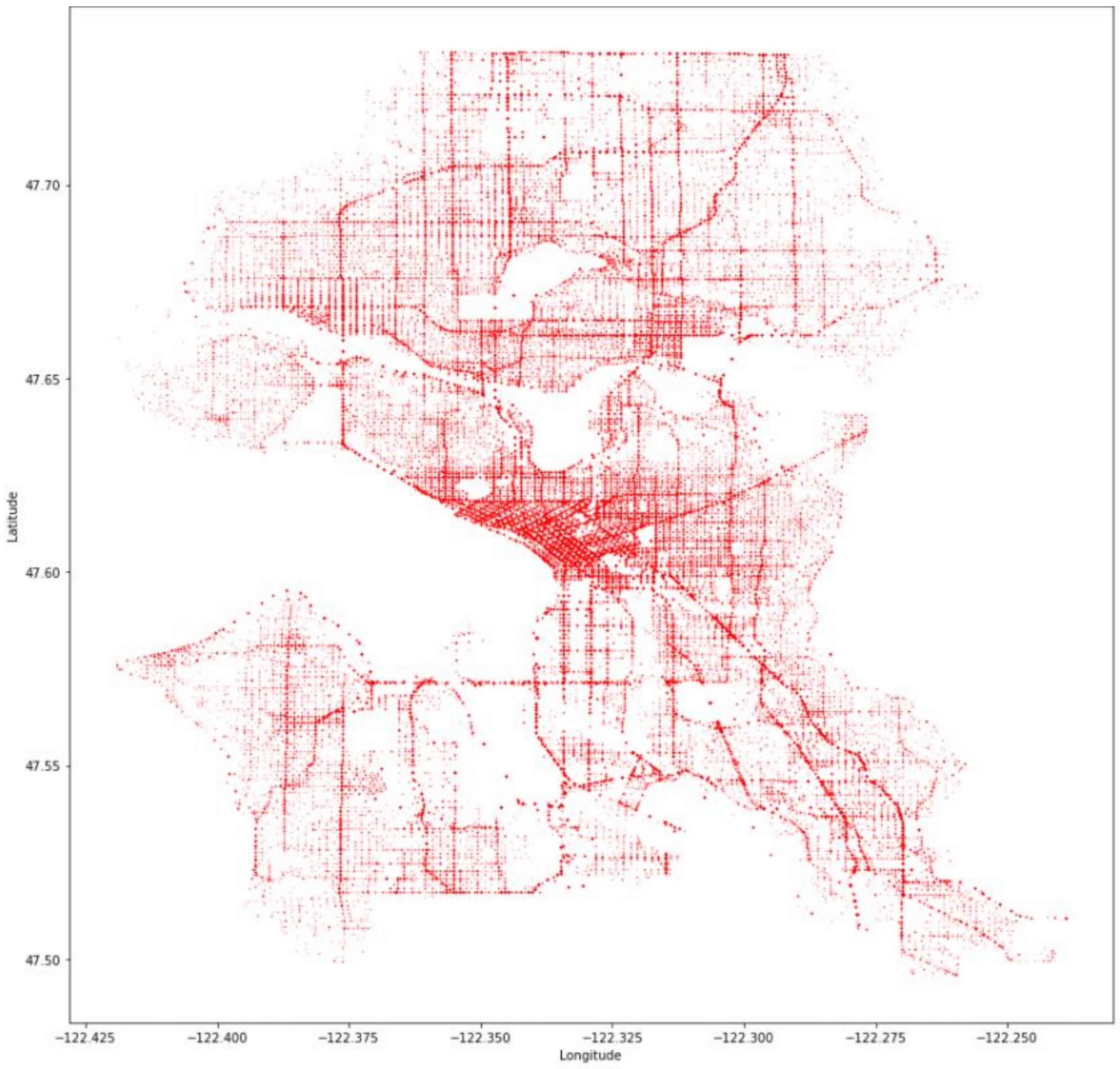
We can see here that the number of accidents tend to peak on a Thursday(Day 4) and are lowest on a Saturday(Day 6) and Sunday(Day 0). This makes logical sense considering that Thursday is middle of the week and therefore likely to experience the most traffic. Since most people do not work on weekends, the number of vehicles on the road will be much less, ergo lesser number of accidents.

Let us move on to some spatial analysis with some map visualizations.

We can visualize the clusters of accidents on a Map to gain a better idea where accidents seem to be occurring in Seattle. We can use the Folium maps library in Python to generate a map of Seattle and add the accidents onto the map using the Latitude and Longitude coordinates within the dataset. Since Folium maps are a bit more computationally expensive, we will only visualize accidents that took place in the year 2019.



It seems most of the accidents seem to be around the city center. This would make sense logically considering the city center is most likely to be densely populated with vehicles, buildings and pedestrians alike. We can confirm this by also creating a simple scatter plot of the accidents using latitude and longitude coordinates.



We can immediately see that the accidents become more and more denser as we move closer to the city center and sparser as we move away from it.

That concludes our EDA. Let us move on to building our Model.

2. Data Modeling

Data Preparation

Before we can begin modeling with our data, we have to process the data a bit further. Recall that we had originally selected 15 attributes to conduct our EDA. Out of those 15 however, year and month did not seem to show any major patterns toward predicting accident severity, so we dropped those variables in order to preserve computational resources. Day of week was the only time-based attribute selected as it did show to have an impact on accidents.

Creating Dummy Variables

Also, a lot of the independent variables in our feature set are multi-class categorical variables. Some examples of these variables along with their classes are:

- ROADCOND: Dry, Wet, Snow/Slush, Oil etc.
- WEATHER: Clear, Raining, Snowing, Overcast etc.
- LIGHTCOND: Daylight, Dusk, Dawn etc.

These variables cannot be interpreted by the scikit-learn library during modeling, hence they had to be transformed using one-hot encoding to create dummy variables with values of 0 or 1 to denote the occurrence of each class within the variable.

The other categorical variables such as SPEEDING or UNDERINFL are already binary-class variables and as such can easily be encoded as 0 and 1.

Creating an Unbiased Dataset

The observant among you would have noticed earlier that there is a severe class imbalance in our target variable of Accident Severity. This is because there are way more accidents that involve property damage, as compared to accidents that also result in personal injury. As a result the dataset is heavily biased towards class '0'. More specifically, class '0' has 128,221 cases as compared to only 56,919 cases in class '1'. We will have to correct this imbalance in the dataset before it can be used to train our models. This can be done by oversampling the cases belonging to class '1' in order to create an unbiased dataset.

After oversampling we have an equal number of cases for both class ‘0’ and class ‘1’.

Splitting Data into Training Set and Testing Set

As is customary with any machine learning project, we will split our data into two sets: Training Data Set and Testing Data Set. As the name implies, we will use the Training Set to build and train our model and then test its accuracy on the testing set. We will use 70% of the data for training the model and 30% of the data to perform our out of sample testing.

Models

We will be running 3 models on our dataset.

Decision Tree Model

Decision tree learning is one of the predictive modelling approaches used in statistics, data mining and machine learning. It uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

It is a supervised learning algorithm and utilizes recursive partitioning to split the data based on conditions in the features to classify the cases in each class. This is primarily based on reducing the entropy in each node as much as possible i.e. homogeneity of the cases. In most scenarios, the user must define how deep they want the tree to go.

K-Nearest Neighbor Model

K-Nearest Neighbors is another supervised learning algorithm, which is primarily based on the distance between individual data points. This distance metric can be either Euclidean or Minkowski distance. The 'K' represents the number of nearest data points that an individual case will be compared to, in order to receive a classification. For example, if we set k=4, and the four data points nearest to an individual case belong to class 0, then the algorithm will also classify that case as belonging to class 0. While we tried different values of k while running this algorithm, we ultimately went forward with k=4 for the model.

Logistic Regression Model

Unlike the typical Linear Regression which is used in modeling when predicting a continuous target variable, Logistic Regression is used when the target variable is a binary categorical variable. As its name implies, Logistic Regression uses a logistic function in estimating the parameters of a logistic model (a form of binary regression), where the two classes are some sort of positive-negative or in other words, '0' and '1'.

In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"), the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name.

After running these models, we proceed to model evaluation to see how well they perform in classifying our target variables and check their accuracy using a couple of metrics.

3. Model Evaluation

In this phase, we will evaluate our model accuracy by applying the models on the test data that we had created earlier, to observe out of sample accuracy. As we had mentioned in the methodology section, we will be using the following metrics to evaluate our machine learning models:

- **F-1 Score:** In statistical analysis of binary classification, the F1 score is a measure of a test's accuracy. It is calculated from the precision and recall of the test, where the precision is the number of correctly identified positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of correctly identified positive results divided by the number of all samples that should have been identified as positive. We will also compute confusion matrices for the same.

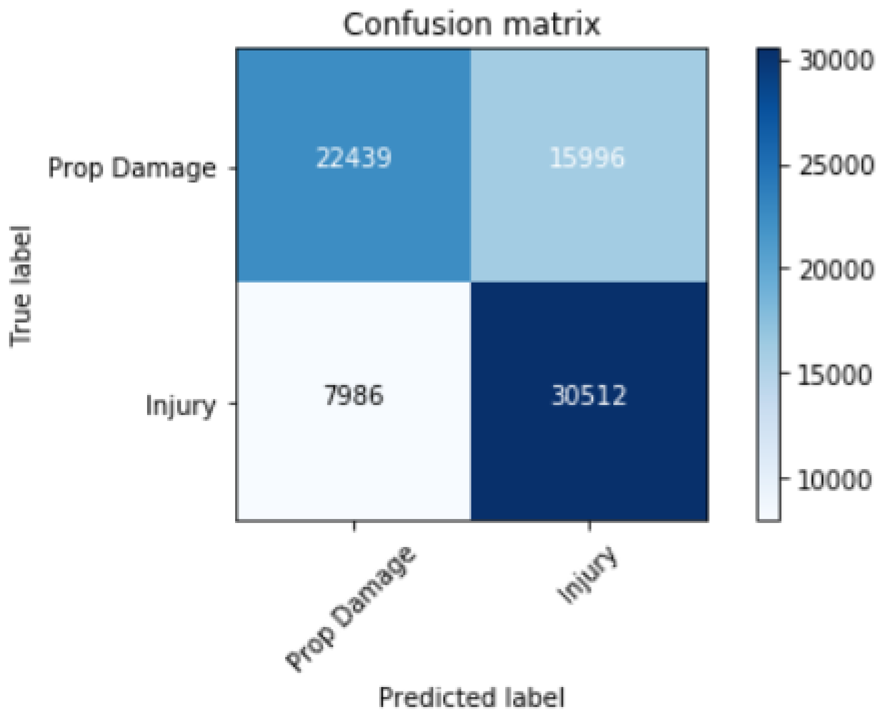
- **Jaccard's Similarity Index:** The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient, is a statistic used for gauging the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets and is defined as the size of the intersection divided by the size of the union of the sample sets.

- **Log Loss (Logistic Regression Only):** Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. So, predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value. A perfect model would have a log loss of 0.

With that done, let us proceed to evaluate our 3 models.

Decision Tree Model

Let us begin the evaluation by plotting the confusion matrix and generating the classification report to see the comparison of predicted labels versus actual labels.

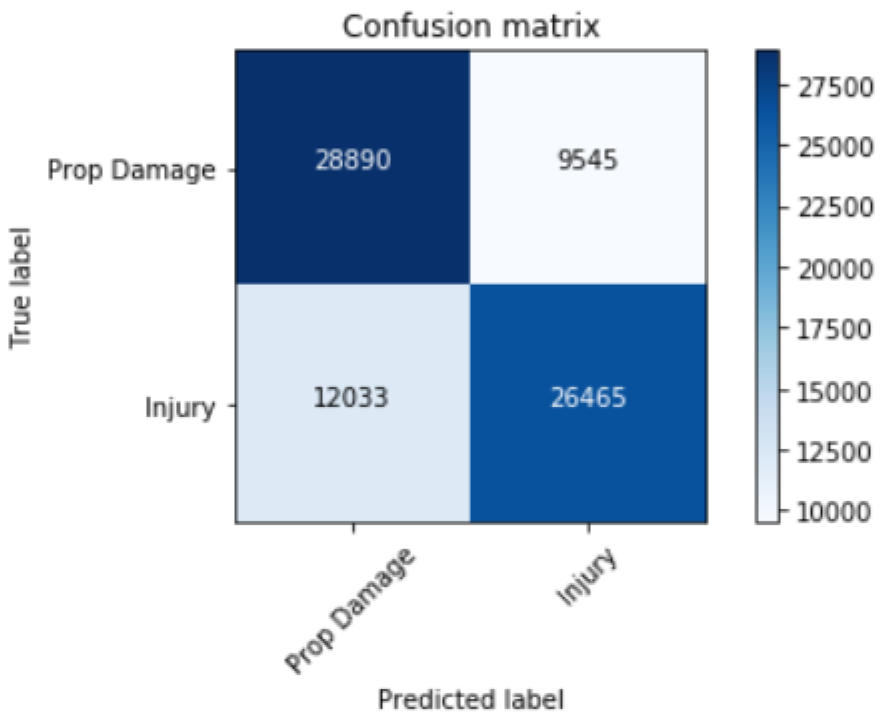


```
              precision    recall  f1-score   support

           0       0.74      0.58      0.65     38435
           1       0.66      0.79      0.72     38498

    accuracy                           0.69     76933
   macro avg       0.70      0.69      0.68     76933
weighted avg       0.70      0.69      0.68     76933
```

This model has relatively good precision and recall in classifying cases belonging to both classes. With a weighted f1-score of **0.68**, it is a fairly good model with a **68%** accuracy in classifying cases. This model also scores a **0.6882** on the Jaccard Similarity Index.

In a similar manner, let us evaluate the remaining models.

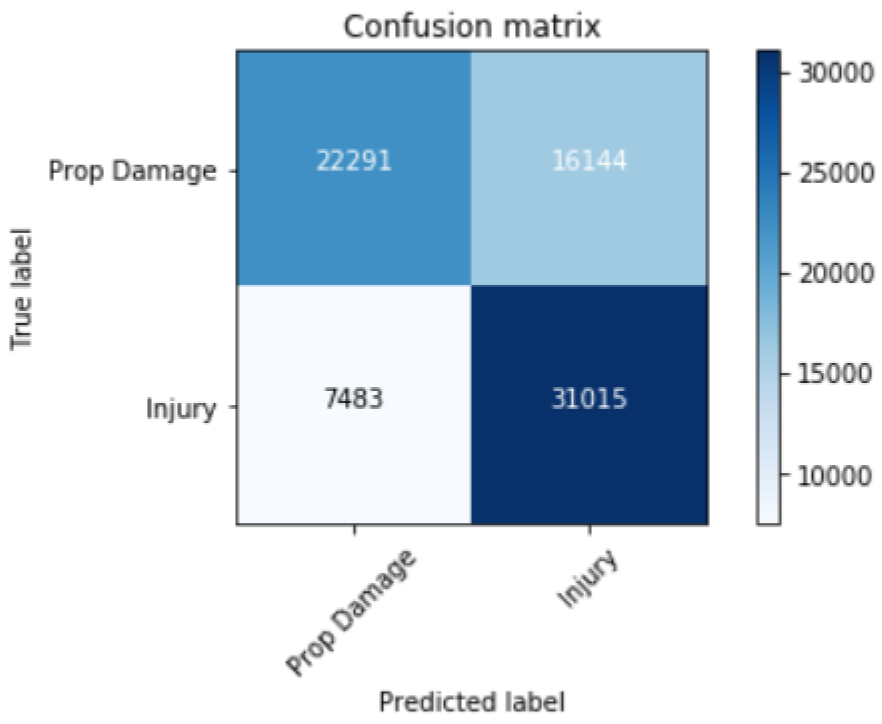
K-Nearest Neighbor Model



```
              precision    recall  f1-score   support

           0       0.71      0.75      0.73     38435
           1       0.73      0.69      0.71     38498

    accuracy                           0.72     76933
   macro avg       0.72      0.72      0.72     76933
weighted avg       0.72      0.72      0.72     76933
```

The KNN model seems to be a bit better around the board with balanced precision in classifying cases belonging to both classes and a good recall distribution as well. With a weighted f1-score of **0.72**, it also more accurate than the Decision Tree Model with a **72%** accuracy in classifying cases. This model also scores a **0.7195** on the Jaccard Similarity Index.

Logistic Regression Model



```
              precision    recall  f1-score   support

           0       0.75      0.58      0.65     38435
           1       0.66      0.81      0.72     38498

    accuracy                           0.69     76933
   macro avg       0.70      0.69      0.69     76933
weighted avg       0.70      0.69      0.69     76933
```

This model seems to perform remarkably similar to the Decision Tree Model in terms of precision and recall distribution. With a weighted f1-score of **0.69**, it is a fairly good model with a **69%** accuracy in classifying cases. This model also scores a **0.6928** on the Jaccard Similarity Index.

Since this is logistic regression model, we also calculate the log loss to see how far the predicted probabilities diverge from positive label or class ‘1’. This value ideally should be a lot lower. This model has a log loss of **0.5636**, indicating this is a fairly good model.

Results and Recommendations

Now that we have finished evaluating the models, let us compile the results of our evaluation into one table.

Algorithm	Jaccard	F1-score	LogLoss
KNN	0.7192	0.7195	NA
Decision Tree	0.6848	0.6882	NA
Logistic Regression	0.6889	0.6928	0.5636

On observation, the 3 models seem remarkably similar in terms of our evaluation metrics, with the KNN model having a slight edge. While all 3 models do perform fairly well, we get a clearer picture if we look at the precision and recall of the models in predicting the binary classes.

Algorithm	Class	Precision	Recall
KNN	0	0.71	0.75
	1	0.73	0.69
Decision Tree	0	0.74	0.58
	1	0.66	0.79
Logistic Regression	0	0.75	0.58
	1	0.66	0.81

We can immediately see that our KNN model seems to have relatively better precision in classifying cases where the accident severity is Injury or '1', as compared to the Decision Tree and Logistic Regression models, and the precision is a lot more evenly distributed between the binary classes. The recall distribution is fairly good across all models, so it is not a huge differentiator.

While the KNN model should be the go-to option for predicting accident severity here, our recommendation to the stakeholders would be to use the three models side-by-side together for each use case, comparing the results and selecting the one that is most appropriate for a given scenario.

From our overall analysis therefore, we have managed to glean the following insights:

- Accidents that involved property damage were more likely to happen on a block rather than an intersection. The distribution of accidents that involve injury however are pretty evenly distributed between intersections and blocks, implying that intersections are far more likely to have an accident where an injury is involved.
- This seems to be confirmed upon examining the types of junctions where the accidents took place. Highest number of property damage accidents were recorded at a mid-block, while injury accidents mostly took place at an intersection.
- The weather by itself does not seem to play a huge role in the severity of an accident, however it may be used in combination with other attributes in a model for predicting accident severity. The same applies to road condition and light condition.
- From the distributions, it does seem that driver inattentiveness seems to be more prevalent in accidents where injury was involved as compared to accidents where only property was damaged.
- There do not seem to be that many accidents where speeding or DUI was recorded.
- Upon examining collision types, we observed that collision with a parked car was the most common collision in accidents that involved property damage, whereas rear ending was most common collision type leading to injury.
- The overall number of accidents do seem to be on a somewhat downward trend since 2014.
- The number of accidents tend to peak on a Thursday and subside on the weekend, presumably when there are less vehicles on the road.
- From the map visualization, it does seem that most accidents tend to take place in the Belltown neighborhood, close to downtown Seattle.

By using a combination of the most relevant features, our data models can correctly predict whether an accident is likely to involve injury or property damage. The Seattle department of transportation or the Seattle police department can then input information into the model to determine whether injurious accidents are more likely to occur, given certain conditions. They can then take some preventive or corrective actions such as :

- Installing appropriate road signs where necessary.
- Issue accident warnings.
- Increase development in areas with poor road or light conditions.
- Increase the fines for DUI or Reckless Driving.
- Deploy more traffic police on days which generally see more accidents such as Thursday.
- Keep emergency crews on alert in certain locations.
- Install more traffic cams in densely populated areas.

Conclusion

We have analyzed the relationships between many attributes and how they can impact the severity of an accident. We performed analysis to see how each individual attribute such as road condition or address type can play a role in an accident and we even plotted maps to examine areas where accidents are more likely to occur. We built multiple data models that can use these attributes together to predict whether an accident could involve personal injury or not. The bigger goal from this project would be to use this information to reduce the number of injury-causing accidents taking place in the city of Seattle and eventually, reducing the number of accidents altogether.