# Capstone Project - Car accident severity

## 1. Introduction and Business Problem

Car accidents can vary in severity. Emergency services such as the police, fire brigade or paramedics are often called to deal with car accidents.

Knowing the severity of an accident before they arrive can help these emergency services plan ahead to prepare for what vehicles and equipment they need to send and how they might start tackling the problem when they arrive. This in turn can lead to better outcomes from the accident for all involved.

The severity of accidents can depend on a number of factors, many of which can easily be identified from the accident scene by those present.

Our aim is to build a supervised machine learning model that can use the inputs of these factors from an initial call or report of an accident and use it to predict the severity of the accident. This can then be used to inform the emergency services before they arrive at the scene.

In this study, we will focus on building a model for the emergency services in the Seattle area of the United States.

## 2. Data

Our data set contains accident data recorded in Traffic Records for the city of Seattle as collected by the SDOT and SPD. There are over 194,000 records for accidents from 2004 to present, each with up to 37 different attributes set and an indication of the severity of the accident.

We will use this data set to identify the keep attributes and then to train and test our model in order to predict the severity of an accident.

### 2.1 Attributes

Our data contains attributes covering a range of different areas:

* time/date - such the time of the accident (INCDTTM) and the date of the accident (INCDATE)
* location - such as associated intersection (INTKEY), junction type (JUNCTIONTYPE), address type (ADDRTYPE), a description of the location (LOCATION) and crosswalk id (CROSSWALKKEY)
* involvement - such as the number of people involved (PERSONCOUNT), the number of pedestrians involved (PEDCOUNT), the number of bicycles involved (PEDCYLCOUNT) and the number of vehicles involved (VEHCOUNT)
* conditions - such as weather (WEATHER), road conditions (ROADCOND), light conditions (LIGHTCOND)
* collision details - such as collision description (COLLISIONTYPE, ST\_COLCODE, ST\_COLDESC, SDOT\_COLCODE, SDOT\_COLDESC), lane segment involved (SEGLANEKEY), if speeding was a factor (SPEEDING), if a parked car was involved (HITPARKEDCAR), information on the pedestrian right of way (PEDROWNOTGRNT), whether the driver was under the influence or not (UNDERINFL), whether the accident was due to inattention (INATTENTIONIND)
* identification - unique IDs given by various organisations involved in collecting the data, such as OBJECTID, INCKEY, COLDETKEY, REPORTNO and SDOTCOLNUM

### 2.2 Identifying the relevant attributes

Not all of these attributes will be relevant for our model. For example, UNDERINFL - attribute identifying whether someone was under the influence of drugs or alcohol, is unlikely to be known about in advance, therefore, we should not include it in our model. In the same way, information on the pedestrian right of way (PEDROWNOTGRNT), knowing if speeding (SPEEDING) or if driver inattention (INATTENTIONIND) were factors may also not be known in advance.

The codes given by the SDOT and SPD are quite detailed. Examples for SDOT\_COLDESC include:

* MOTOR VEHICLE STRUCK MOTOR VEHICLE, FRONT END AT ANGLE
* DRIVERLESS VEHICLE STRUCK MOTOR VEHICLE REAR END

While examples for ST\_COLDESC include:

* From opposite direction - one left turn - one straight
* From same direction - both going straight - both moving - rear-end

It is unlikely sufficient information will be obtained from initial call outs to identify the correct code and description here and therefore these should not be used in our model in their current format.

However, COLLISIONTYPE appears to give a more usable attribute containing similar data, but in a format which is more likely to be obtainable from the initial report. Examples of COLLISIONTYPE values include:

* Rear Ended
* Angles
* Parked Car

The data contained in the LOCATION attribute can give very specific addresses. This data will be hard to analyse and group in a machine learning model. We will likely be unable to use it unless we can discover a way to split or group it into a usable format.

The various unique IDs will also not form part of our model.

We will therefore attempt to use the following attributes in our model, all of which should normally be easily identifiable or estimated for a collision at the time it is first reporting to the emergency services:

* time of the accident (INCDTTM)
* the date of the accident (INCDATE)
* intersection (INTKEY)
* junction type (JUNCTIONTYPE)
* address type (ADDRTYPE)
* crosswalk identifier (CROSSWALKKEY)
* lane segment involved (SEGLANEKEY)
* the number of people involved (PERSONCOUNT)
* the number of pedestrians involved (PEDCOUNT)
* the number of bicycles involved (PEDCYLCOUNT)
* the number of vehicles involved (VEHCOUNT)
* weather (WEATHER)
* road conditions (ROADCOND)
* light conditions (LIGHTCOND)
* collision type (COLLISIONTYPE)
* if a parked car was involved (HITPARKEDCAR)

## 3. Methodology

### **3.1 Initial data pre-processing**

In this section, we look at converting our data in to usable, numeric formats.

We first looked at extracting data from the date and time fields. Note that all the information included in INCDATE is duplicated in INCDATTM, so we can drop INCDATE column. We can also convert the date column (INCDTTM) to the correct format, while also extracting hour, the day of the week and month from it for new columns. Finally, we can drop the INCDTTM column, as it is no longer needed now we have extracted the useful data from it.

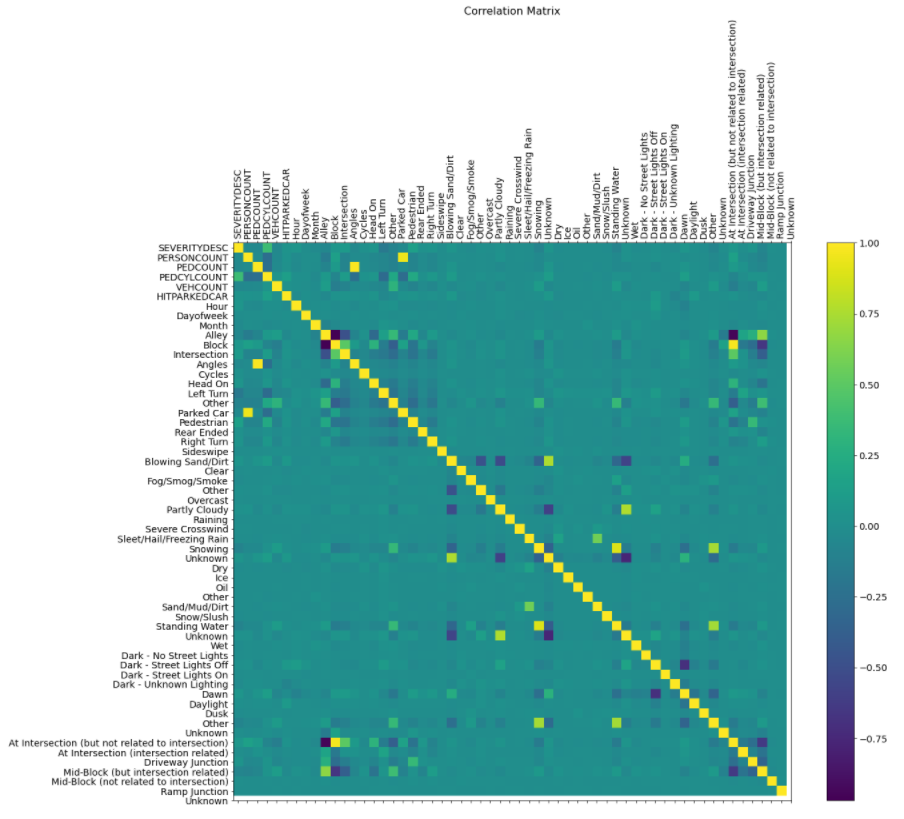
We used one hot encoding to convert categorical columns into several numeric columns. This was used on ADDRTYPE, COLLISIONTYPE, WEATHER, ROADCOND, LIGHTCOND and JUNCTIONTYPE. The original columns were then dropped from our data frame.

Finally, we convert our last non-numeric field, HITPARKEDCAR, into a binary numeric column, as it only contains two values. – Yes and No. Again, the original column was then dropped.

### 3.2 Reducing features

We next looked at reducing features, as our data now had 58 different columns.

We plotted a correlation matrix and identified those factors closely correlated with each other (either negatively, or positively).



The following are all highly correlated:

* Angles, PEDCOUNT
* Parked car, PERSONCOUNT
* Block, Alley
* At intersection (but not related to intersection), Alley
* At intersection (but not related to intersection), Block
* Mid-Block (but intersection related), At intersection (but not related to intersection)
* Unknown (weather), Unknown (road conditions)
* Blowing sand /dirt, Unknown (weather)
* Blowing sand /dirt, Unknown (road conditions)
* Unknown (weather), Partially cloudy
* Unknown (road conditions), Partially cloudy
* Snowing, Standing water
* Sleet/hail/freezing rain, Sand/mud/dirt
* Other (time of day), Standing water
* Dawn, dark (street lights off)

We then attempted to eliminate as many features as possible, based on the correlated features.

This resulted in dropping the following fields, leaving us with 47 columns remaining.

* Alley
* Angles
* Parked car
* Mid-Block (but intersection related)
* Unknown (weather)
* Unknown (road conditions)
* Sand/mud/dirt
* Standing water
* dark (street lights off)

### 3.3 Preparing for a machine learning model

We then split our data in to training data sets (X\_train and y\_train) and test data sets (X\_test and y\_test) in order to build machine learning models to classify collision types.

Because the data set contain numerical values of different types (some were 1s and 0s, others 1 to 12, others 0 to 23) we needed to standardise and normalise our training set, to ensure no factor over-influenced the machine learning mode. This created set X\_train\_n – our normalised training set.

### 3.4 Machine learning models.

We first attempted to use the knn algorithm to build our model for various k to identify the best k.

We then use decision trees to build our model, trying different depths to pick the ideal depth.

Finally, we used logistic regression to build a model.

In all cases, we used accuracy scores, F1 scores and in the case of logistic regression, log loss to measure the accuracy of our models. We use the test data sets to measure these.

We also plotted confusion matrices where possible.

## 4. Results

4.1 KNN

Our attempts to build a knn model ran in to difficulty over processing time required. We processed for k between 2 and 10.

However, when plotting the accuracy score, you can see for all k, the accuracy is reasonably similar and therefore would likely require significantly more processing for higher k to find a model which would be more accurate.

We therefore stopped using this model and moved on.

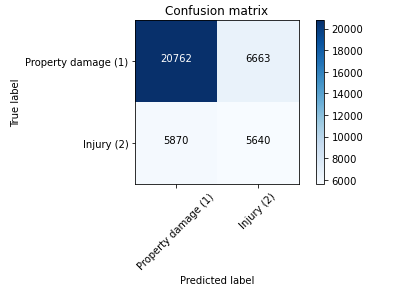
## 

4.2 Decision trees

By trying depths from 2 to 50 for our decision trees, we found our model was most accurate when the depth was 37 – when our decision tree went down to 37 levels on some branches.

* DecisionTrees's Accuracy: 0.6781045331963529
* DecisionTrees's F1 score: 0.6811032208372303

Plotting the confusion matrix for this gave us:



We see the following:

* Predict 'property damage' correctly: 75.7%
* Predict 'property damage' when it's an 'injury collision': 51.0%
* Predict 'injury' correctly: 49.0%
* Predict 'injury' when it's 'property damage': 24.3 %

## 4.3 Logistic regression

## Next, our logistic regression model was run and gave us:

* F1-score (lr): 0.7003
* Log loss: 0.805498175639136

Plotting the confusion matrix for this gave us:

## 

We see the following:

* Predict 'property damage' correctly: 96.5%
* Predict 'property damage' when it's an 'injury collision': 76.5 %
* Predict 'injury' correctly: 23.5 %
* Predict 'injury' when it's 'property damage': 3.5 %
* If a prediction of ‘property damage’ is given, it is correct 75% of the time.
* If a prediction of ‘injury’ is given, it is correct 74% of the time.

## 5. Discussion

We can see that of our two successfully built models that the logistic regression model was more accurate than the decision tree model, based on the F1 scores.

When plotting the confusion matrix, we can see that the logistic regression model correctly predicted a collision involving only property damage 96.5% of the time, only incorrectly suggesting it should have been an injury collision about 3.5% of the time. This is better than the decision tree model, where the prediction for property collisions was only correct 75.7% of the time.

However, we saw the decision tree model being more accurate in predicting the injury collisions, being correct 49% of the time, compared to just 23.5% of the time for the logistic regression model.

Considering the use of the models by the emergency services, in both cases, a majority of the most serious collisions involving injury would be mis-classified as just having property damage. This would mean the emergency services would be unprepared for what they find.

I would therefore suggest both the decision tree model and the logistic regression model are moved forward for refinement, as they show promise in identifying the severity of collisions, but need improvement in their accuracy of accurate predicting injury collisions before they can be considered useful.

One area to consider is the unequal representation of both types of collision in the data set. There are far more cases of property damage collisions, compared to injury collisions. Considering under-sampling or over-sampling to equalise the split of collision types in the training data set is likely one route to building a more accurate model in the future. Give the ratio between the two is not extreme, under-sampling is likely what is required.

An extension would also be to obtain extra data on more serious collisions – those including serious injury and fatalities. A model could then be built to identify the severity of injury. Given the lower frequency of these types of collisions, it is again likely some sort of over-sampling will be required.

## 6. Conclusion

Our initial aim was to build a model for predicting the severity of a collision to prepare the emergency services when on their way to the scene of the collision.

We were successfully able to build two machine learning models that could be used to predict the severity. Our logistic regression model had a 96.5% accuracy in predicting the lower severity ‘property damage’ collisions, which is great. However, both models had lower accuracy when predicting ‘injury’ collisions.

Our models would therefore not be suitable for use at the moment. But they do show promise and refinement, using under-sampling of the training data set, is likely to produce better results in future.