**FINAL REPORT**

**SEATTLE ROAD ACCIDENT SEVERITY PREDICTION**

***IBM Data Science Capstone***

**Tanuj Agrawal**

September, 2020

**1. INTRODUCTION/BUSINESS PROBLEM**

**1.1 Problem Statement**

Say you are driving to another city for work or to visit some friends. It is rainy and windy, and on the way, you come across a terrible traffic jam on the other side of the highway. Long lines of cars barely moving. As you keep driving, police car start appearing from afar shutting down the highway. Oh, it is an accident and there's a helicopter transporting the ones involved in the crash to the nearest hospital. They must be in critical condition for all of this to be happening. Now, wouldn't it be great if there is something in place that could warn you, given the weather and the road conditions about the possibility of you getting into a car accident and how severe it would be, so that you would drive more carefully or even change your travel if you are able to. This is exactly what we will try to address through this project and come up with an effective model that caters to this particular need.

**1.2 Target Audience**

Who will be benefitted from this particular solution? Residents of Seattle can use this model which could aid them in deciding whether or not they should travel along the route under pre-existing weather and road conditions. Police Department can also use this particular model to predict problem areas and have an immediate response team ready at these locations to lessen the damages to life and property. Few visualizations and data for the streets and blocks with maximum number of mishaps due to various uncontrollable forces during road travel namely weather, road condition and light conditions will also be made available to the users. Another far-fetched use could be Google Maps using the data to improve time estimations and suggest faster alternative routes for such trips. This can be done in addition to its report a/an jam/accident feature. Now users can report the existing weather and road conditions, and google maps can alert other uses about the risks of travelling along this route based on these aforementioned parameters.

**2. DATA**

The data that we will use to create the model, is the *Road Accident Severity Data* from the *Seattle State Department of Transport* from [here](https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv). It is a csv file that contains 194673 entries and 36 attributes. Many of these 36 attributes are not really helpful and we will be cleaning the unnecessary data in the next stages. This file contains data about human factor’s play in these accidents and the details of damage for each instance along with dates and other ‘Yes’ and ‘No’ questions regarding whether the driver was under influence of alcohol or not, or whether the accident involved speeding or not ,or whether the pedestrian was granted the right of way to name some. The file also contains data pertaining to number of pedestrians, number of vehicles, number of bicycles and number of total people involved in each accident. This dataset has an attribute that classifies the accident into 5 different categories labelled as a number from 0 to 3. SEVERITYCODE is the attribute of interest here. A code that corresponds to the severity of the collision:

• 3- Fatality

• 2b- Serious Injury

• 2- Injury

• 1- Property Damage

• 0- Unknown

We will extract data for weather, road and light conditions, and the time of day and location into a new data-frame which will be used for our model. The columns for the same would be ROADCOND, WEATHER, LIGHTCOND, LOCATION, X, Y, INCDTTM and INCDATE. We will clean this data, remove NaN values and categories that have a very small sample size in order to balance the data. Once data extraction and cleaning are completed we will move ahead with visualizations and modelling using either KNN, Decision Tree or Logistic Regression.

**2.1 Data Extraction**

The required columns out of the 38 attributes are ROADCOND, WEATHER, LIGHTCOND, LOCATION, X, Y, INCDTTM, INCDATE and SEVERITYCODE. These will be the main attributes that we will be using throughout the project.

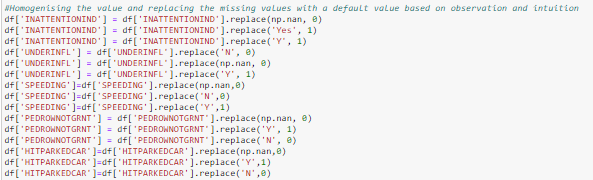
For initial stages I considered keeping PERSONCOUNT, PEDCOUNT, PEDCYLCOUNT, VEHCOUNT, INATTENTIONIND, UNDERINFL, PEDROWNOTGRNT, SPEEDING and HITPARKEDCAR. The metadata for the same can be found [here](https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Metadata.pdf). We will try to clean these extra attributes and see if it can be used in the model. (These attributes involve number of vehicles etc. involved and human factors they might not be a right fit for our model)



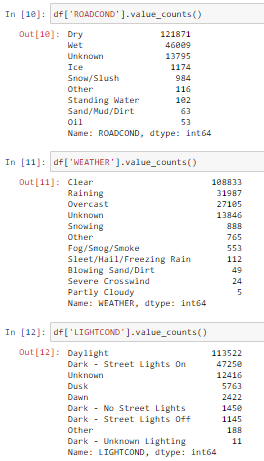
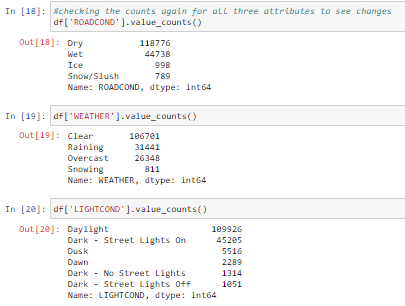
Dropping NaN values from primary attributes before moving to the next step. (We use the dropna function) Next step is data cleaning where modifications for missing values will be done and modifying contents of certain attributes will take place.

**2.2 Data Cleaning**

Starting with the extra attributes and homogenizing the data i.e. converting Yes/Y to 1, No/N to 0 and NaN values to 0 as most of the rows are 0 (average of entire column)

****

Next we need to clean the important attributes. Starting by getting value counts for each attribute. It is observed that there are few categories that have a count very less compared to the categories with maximum instances. To maintain uniformity in data categories we shall cap the number of occurrence to 700 and remove the rest categories. Also ‘unknown’ category within these attributes don’t help our model either. So we shall drop the entries that contain these values. (This cleaning is carried out on ROADCOND, WEATHER and LIGHTCOND)

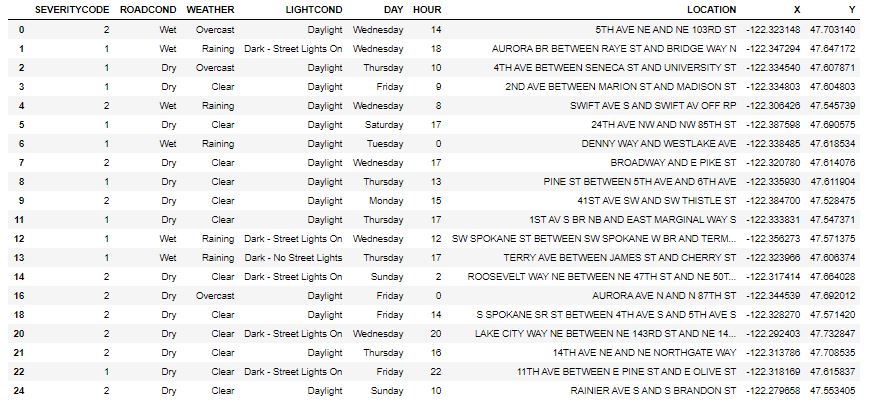
 

*Before cleaning After Cleaning*

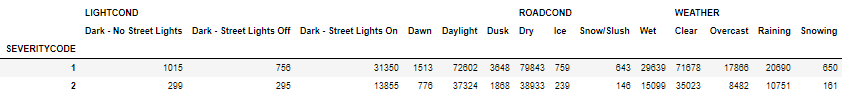
****

Creating new columns named day and hour from INCDATE and INCDTMM respectively using dt.day\_name () and dt.hour functions. These attributes will be used for visualizations and modelling as this data makes more sense than YYYY-MM-DD HH-MM-SS-TTTT format.

Once all the required data attributes are created and available, next step is to create a separate dataframe df\_new using copy () function. The new dataframe has the following attributes: SEVERITYCODE, X, Y, ROADCOND, WEATHER, LIGHTCOND, DAY, HOUR and LOCATION.



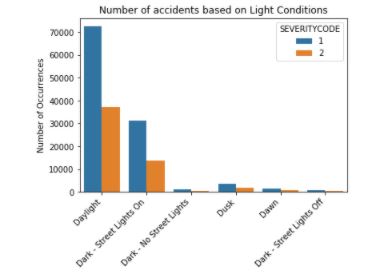
***Figure 1.*** *df\_new dataframe after dropping the unnecessary columns and removing the NaN values from all the primary attributes*



***Figure 2.*** *df\_t dataframe illustrates the value counts of each type of road, weather and light conditions where the attributes are stacked using stack()*

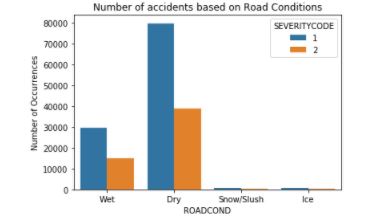
**3. METHODOLOGIES**

**3.1 Exploratory Data Analysis**

****

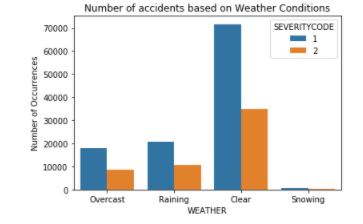
***Figure 3****. Number of accidents for various light conditions*

I used the seaborn class to create this visualization of df\_new dataframe which shows the distribution of accidents along with the severity index classifications, based on different light conditions. Daylight causes the most number of accidents and the occurrence of injuries is greater than property damage.

****

***Figure 4.*** *Number of accidents for various road conditions*

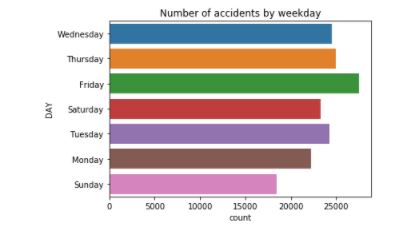
I used the seaborn class to create this visualization of df\_new dataframe which shows the distribution of accidents along with the severity index classifications, based on different road conditions. Dry conditions causes the most number of accidents and the occurrence of injuries is greater than property damage.

****

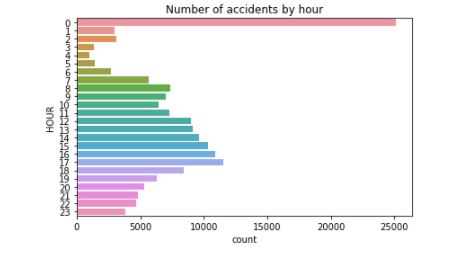
***Figure 5.*** *Number of accidents for various weather conditions*

I used the seaborn class to create this visualization of df\_new dataframe which shows the distribution of accidents along with the severity index classifications, based on different road conditions. Clear conditions causes the most number of accidents and the occurrence of injuries is greater than property damage.

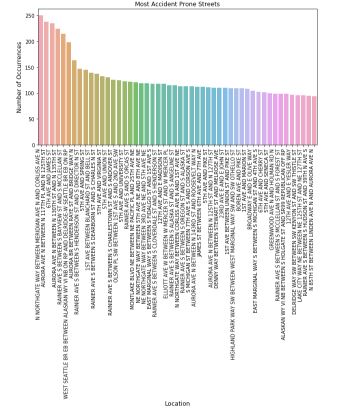
The next two visualizations for accidents occurring on a particular weekday and at a particular hour of the day, which is carried out using seaborn countplot (). It can be seen that 0000 hours have the most number of accidents. Similarly Friday witnessed the maximum number of incidents. Although the distribution for weekday is not as dramatic as that for accident by hour. The barplot () function is used to plot the top 50 most accident prone streets/blocks in Seattle.

****

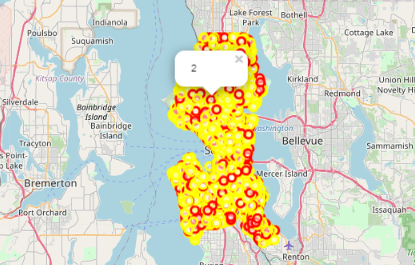
***Figure 6.*** *Number of accidents by weekdays*

****

***Figure 7.*** *Number of accidents by hour*

****

***Figure 8.*** *Number of accidents for top 50 locations*



***Figure 9.*** *Plot of all the accidents on the map along with markers for severity index*

Using Folium and Features a map visualization is created to plot all the accidents in various colors and a pop up marker to indicate the severity of each incident. This plot shows the distribution of accidents and further modifications can be made to the parameters. For example, all the incidents in the past year can be plotted and a grouping can be done for accidents occurring in a particular region.

**3.2 Predictive Modelling**

To apply supervised machine learning algorithm for classification all the data must be numerical. So, in order to convert the categorical attributes to numerical values I implemented label encoding which labels each category within the attribute with a number. Next we will drop the categorical attributes as we have already created the new attributes from label encoding and these old columns are not required for modelling. This encoding and further cleaning gives us a table ready to be used for predictive modelling.

But before doing predictive modelling using various machine learning algorithms, data imbalance must be ruled out in order to create an unbiased model. We check the value counts for each of the two classification categories of accident severity 1 and 2 and observe that the severity index 1 has a sample size twice that of 2.

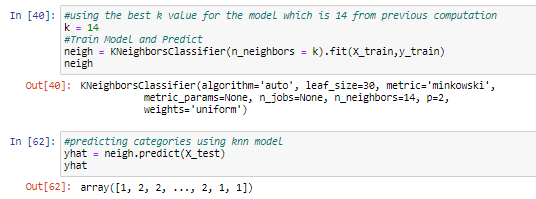
**** 

***Before resampling After resampling***

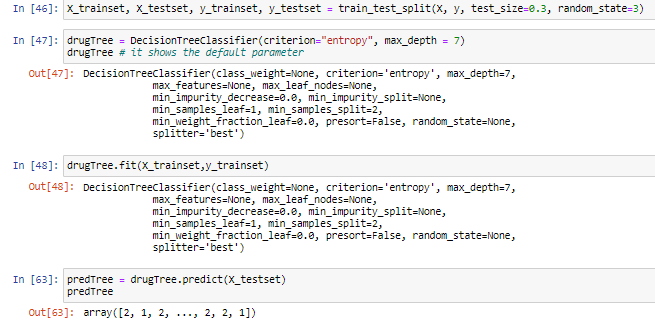
Down-sampling is the best way to proceed with balancing the dataset and this is achieved by importing resample from scikit learn library and writing few lines of code to down-sample the category with more number of instances such that both the categories now have equal number of records. This particular step ensures better balanced recall and precision for the model which otherwise could have been skewed.

Moving ahead with the modelling, I have used KNN, Decision Tree and Logistic Regression as the three machine learning algorithms. For each method used, evaluation metrics used are the Jaccard score and F1 score along with log loss for Logistic Regression. All the required libraries and packages are imported and after normalizing the data using StandardScalar(), this data is split into test and training set with 80% of the dataset being used to train the model.

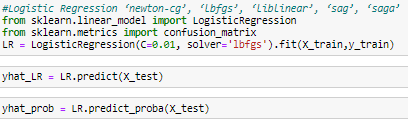
Starting with the KNN first, using a loop and calculating accuracy for each k value from 1 to 20, I obtained maximum accuracy for the test set for a k value of 14.



Next model is decision tree, which used a test size of 30% and random\_state of 3 to split the preprocessed data. The classifier is defined with a max depth 7, after calculating trail accuracies with other values ranging between 1 and 20. There is scope for refinement here but I decided to go ahead with 7 as the max depth and fit the classifier using the training set before predicting target attribute using the test set. F1 score and Jaccard score for this model is also evaluated.



Lastly using Logistic Regression to create a predictor model, I tried implementing various solvers like newton-cg, liblinear, saga but all the solvers had similar accuracy for a C value of 0.01. The probability for the predicted values is also calculated along with the metric evaluation.



**4. RESULTS AND EVALUATION**

It seems like decision tree would be the best machine learning algorithm for this particular model as indicated by the highest F1 score and Jaccard score of 0.5321. With further changes in max depth we could achieve a little more accuracy. But a score of 0.5321 shows a decent balance between recall and precision. It also indicates that road, weather and light conditions together with location and time of day can be used to predict severity of accident with an accuracy of 0.5321. Although, Logistic Regression can also be used as it would work well with binary classification which is the case with our dataset. (As it contains just two categories, injury and property damage)



**5. DISCUSSION**

Based on visualizations it seems as if dry, clear skies and daylight conditions have caused the most accidents but these conditions are normal conditions and we could expect more driver negligence or other factors unaccounted for to be the main culprit. Few more lines of code that check the status for all cases where all the three conditions occur together can be carried out. Since it doesn’t help the current model which is used to predict the severity based on just the uncontrollable natural conditions, I didn’t go ahead with implementing it. The X and Y coordinates reduced the F1 score when included in the model instead of label encoded LOCATION attribute, from 0.56 to 0.53. Analysis for the same could be carried out to understand effect that each attribute has separately on the model and how does all of these attributes considered together affect the model.

**6. CONCLUSION**

Based on the dataset provided and the prediction model created using decision tree, we can conclude that particular weather, road and light conditions have some level of impact on whether travel could result in property damage or an injury. If implemented properly can result in saving many lives and huge sums of money on a yearly basis for the city of Seattle.