1. Introduction

1.1 Background

Accidents happen all the time. And with the increasing number of vehicles out there, there is also an increase in the number of car accidents. There are a lot of factors to consider that may contribute to the occurrence of the accident such as the conditions of the road due to the weather or traffic. The condition of the drive must also be considered, whether he/she is under the influence of alcohol or is simply not paying attention at the time. We will be using machine learning algorithms to determine which of these factors they should look out for so that they may take certain precautions before heading out in the road.

1.2 Problem

We will be looking at car accident data in finding out what should a driver consider when heading out in order to avoid an accident. We will be looking at the severity of the accident.

1.3 Interest

Aside from car driver's, other parties that might be interested in this study would include insurance companies and local government agencies such as police and traffic enforcement. The model developed in this study would be able to provide some insights to the target audience on how to reduce the number of the car accidents happening.

2. Data

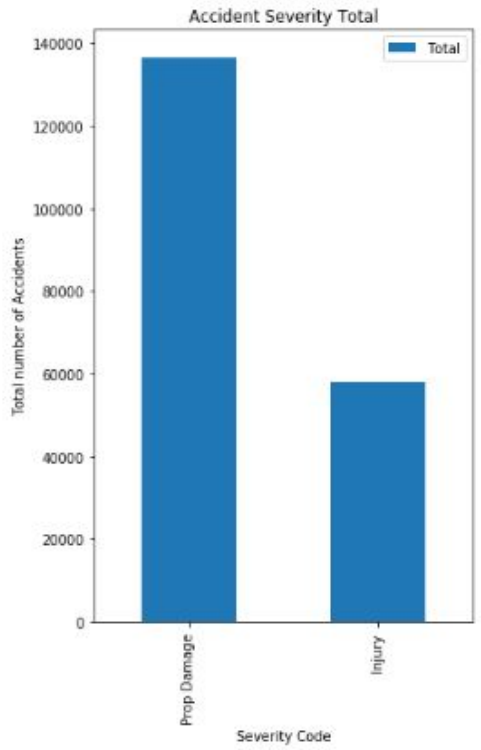
2.1. Data description

The data to be used for modelling will be taken from the Seattle area through car accident information gathered from 2004 to May 2020 by the Seattle Department of Transportation (SDOT). The data which is available in the SDOT website, will comprise 37 columns containing variables on car accidents such as its severity, street, place of accident, weather among others.

The dependent variable that we will be using for this study is the SEVERITYCODE which has measurements on the degree on the accident as seen below based on information:

0: Unknown
1: Property Damage
2: Injury
2b: serious injury
3: fatality

Despite the above, the data given on the SEVERITYCODE shows only ‘1’ or ‘2’.



Of the several variables available in the dataset we will be only using the ones we find useful which are:

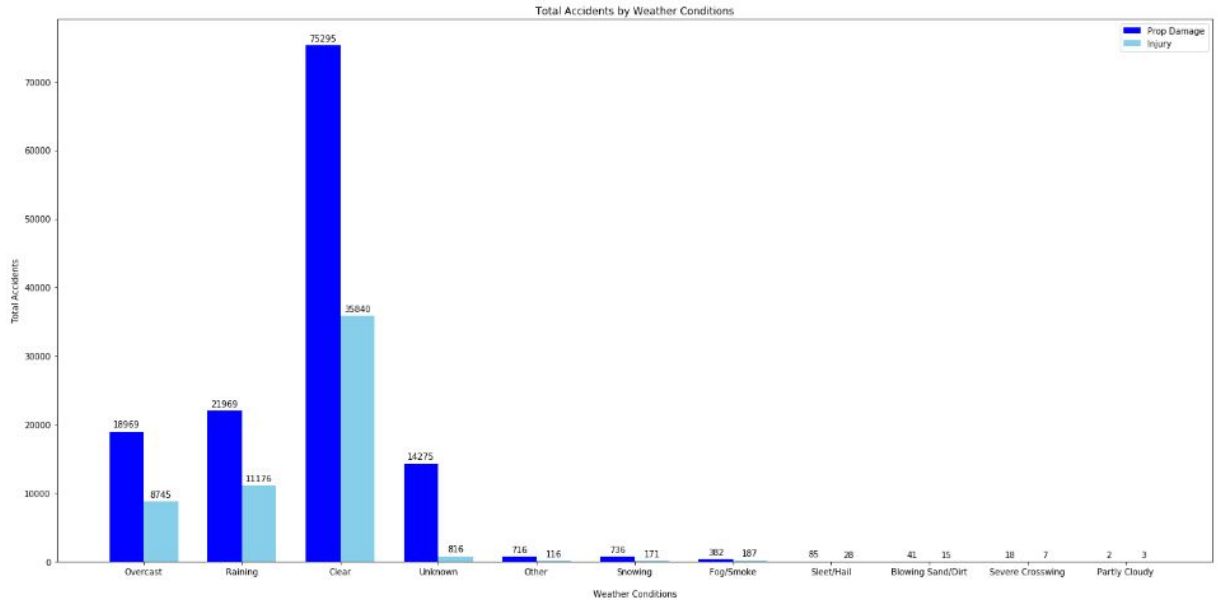
SEVERITYCODE
COLLISIONTYPE
UNDERINFL
INATTENTIONIND
WEATHER
ROADCOND

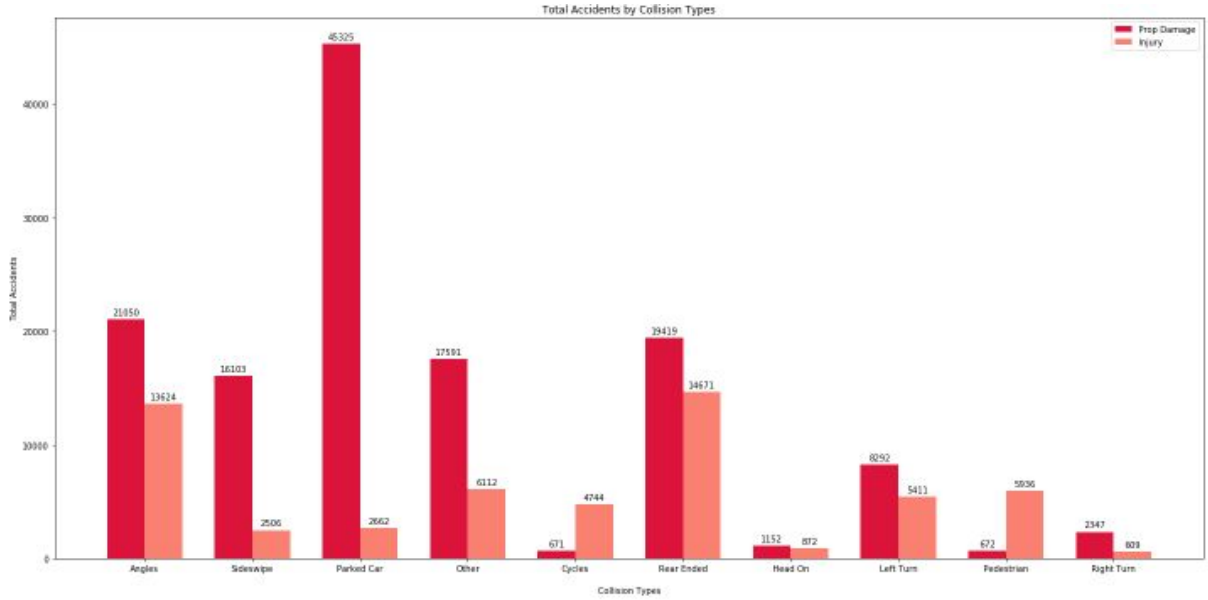
LIGHTCOND
SPEEDING

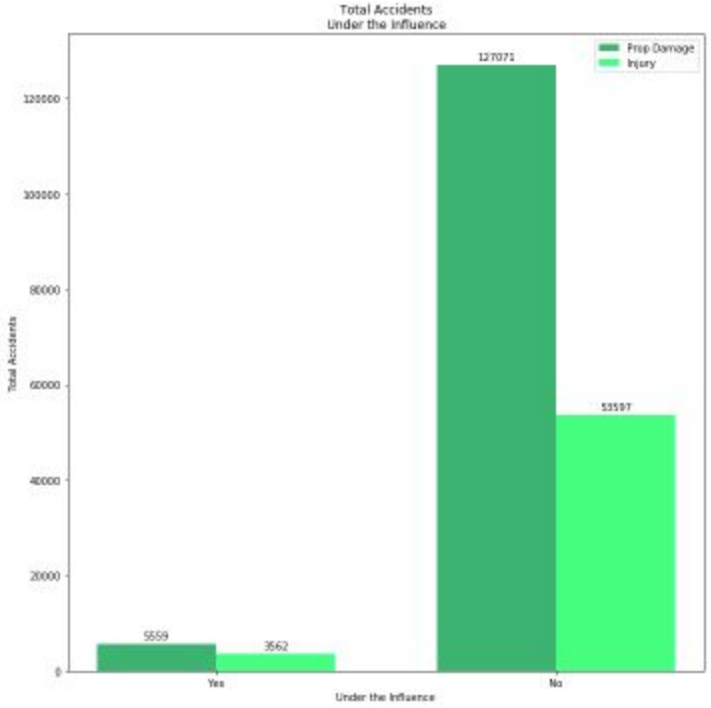
However, because of a lot of missing values, we had to remove INATTENTIONIND and SPEEDING as well. Please see below:

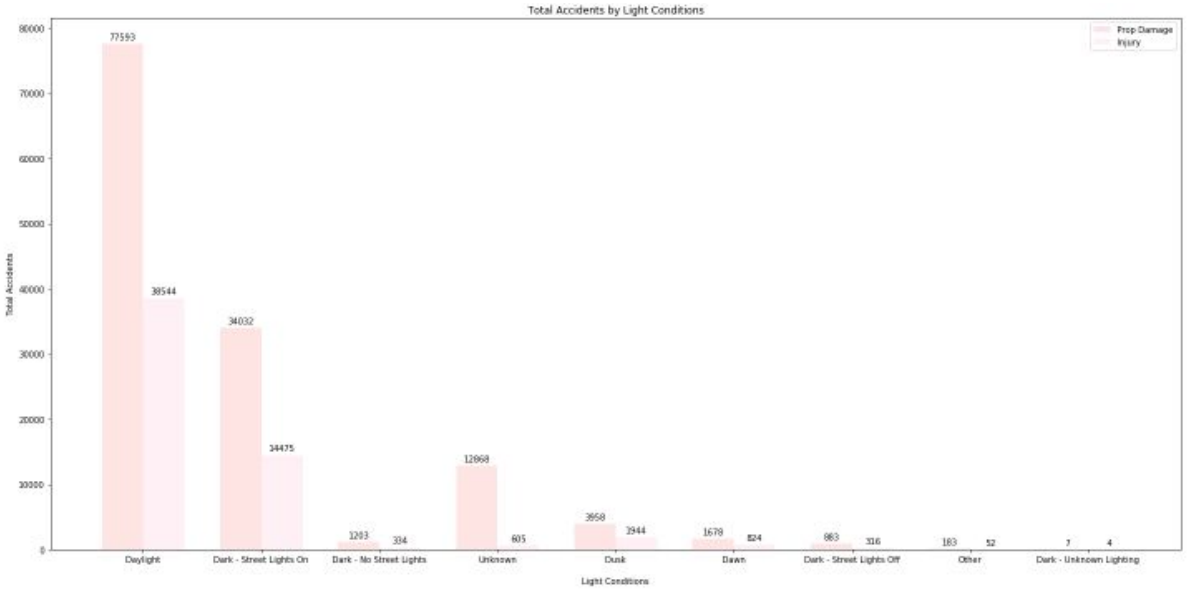
```
In [10]: inatt = df['INATTENTIONIND'].isna().sum() / df.shape[0] * 100
         inatt = round(inatt, 2)
         print ('In the INATTENTIONIND attribute,',inatt,'% of the values are unknown.')

         In the INATTENTIONIND attribute, 84.69 % of the values are unknown.

In [11]: speeding = df['SPEEDING'].isna().sum() / df.shape[0] * 100
         speeding = round(speeding, 2)
         print ('In the SPEEDING attribute,',speeding,'% of the values are unknown.')

         In the SPEEDING attribute, 95.21 % of the values are unknown.
```

Please see below the individual relationship between the SEVERITYCODE and several variables in the study:

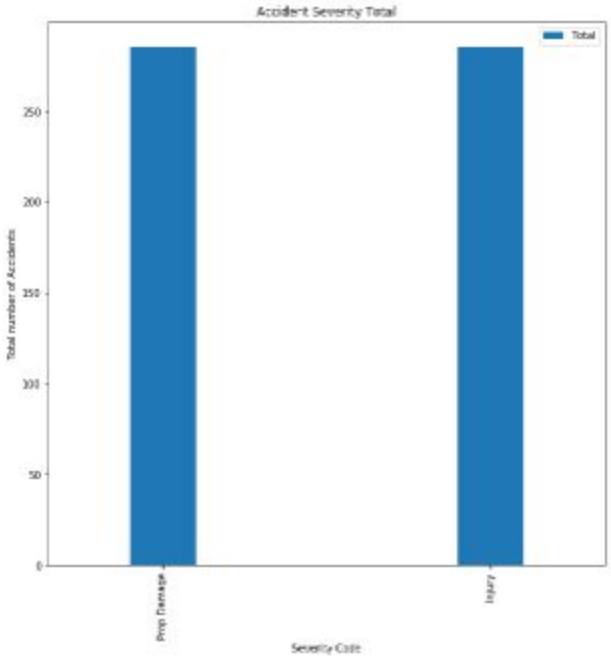



Total Accidents by Collision Types
Prop Damage
Injury
Total Accidents
Collision Types
Angles
Sideswipe
Parked Car
Other
Cycles
Rear Ended
Head On
Left Turn
Pedestrian
Right Turn
21050
13624
16103
2506
46325
2662
17591
6112
671
4744
19419
14671
1152
872
8292
5411
672
5936
2347
809


Total Accidents
Under the Influence
Prop Damage
Injury
Total Accidents
Under the Influence
Yes
No
5559
3562
127071
53597



Based on the charts above, we can see that there is a disbalance because of the large difference in the SEVERITYCODE. Because of this, we will balance out the number of accidents per severity. Please see below:



The values in the dataset contain both numbers and words. But in order for us to be able to process our model, they should be in numbers. With respect to that, we will apply the One Hot Encoding method to turn some values into 0's and 1's. Please see below:

```
In [30]: df_test = df_tr[['SEVERITYCODE','COLLISIONTYPE','UNDERINFL','WEATHER','ROADCOND','LIGHTCOND']]
         df_test = pd.concat([df_test,pd.get_dummies(df_tr['COLLISIONTYPE'])], axis=1)
         df_test.drop(['COLLISIONTYPE'], axis = 1,inplace=True)
         df_test = pd.concat([df_test,pd.get_dummies(df_tr['WEATHER'])], axis=1)
         df_test.drop(['WEATHER'], axis = 1,inplace=True)
         df_test = pd.concat([df_test,pd.get_dummies(df_tr['ROADCOND'])], axis=1)
         df_test.drop(['ROADCOND'], axis = 1,inplace=True)
         df_test = pd.concat([df_test,pd.get_dummies(df_tr['LIGHTCOND'])], axis=1)
         df_test.drop(['LIGHTCOND'], axis = 1,inplace=True)

In [31]: df_test.head()
```

Out[31]:

	SEVERITYCODE	UNDERINFL	Angles	CT_Other	Cycles	Head On	Left Turn	Parked Car	Pedestrian	Rear Ended	...	Standing Water	Wet	Dark - No Street Lights	Dark - Street Lights Off	Dark - Street Lights On	Dawn	Daylight	Dusk	LC_Other	LC_Unknown
11657	1	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	1	0	0	0
20012	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	0
66649	1	0	0	1	0	0	0	0	0	0	...	0	1	0	0	0	0	1	0	0	0
120972	1	0	0	0	0	1	0	0	0	0	...	0	1	0	0	0	0	1	0	0	0
143764	1	0	0	1	0	0	0	0	0	0	...	0	0	0	0	1	0	0	0	0	0

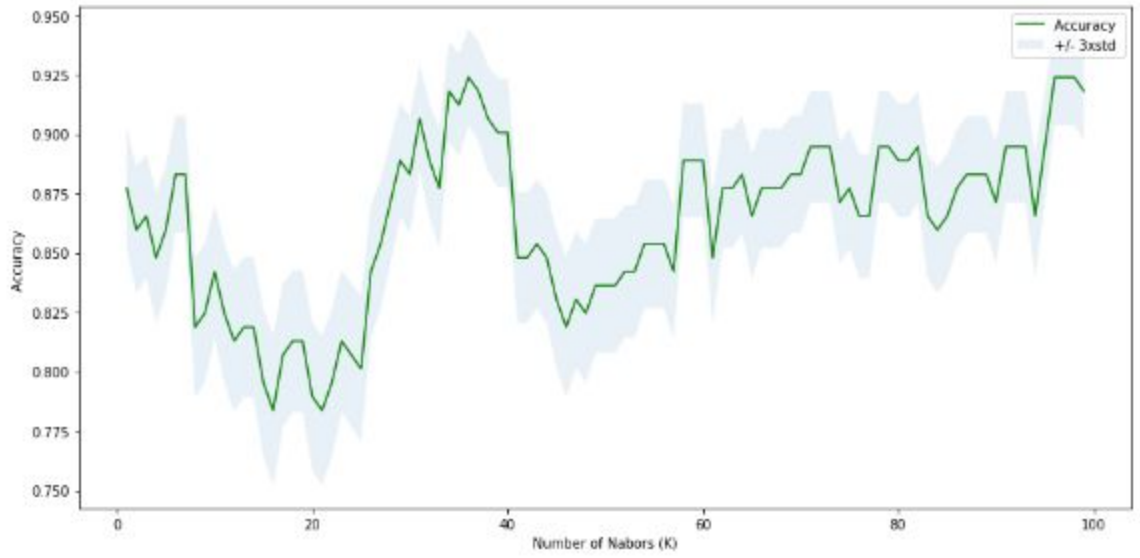
5 rows × 34 columns

3. Modelling

In this section, the models we will be using are the below:

K-Nearest Neighbor (KNN)

KNN will be used to categorize the severity of an outcome based on other outcomes with the nearest data points at k distance. For this study, the KNN is most accurate when k = 36.




```
Best Accuracy: 0.9239766081871345 , K = 36
```

Decision Tree

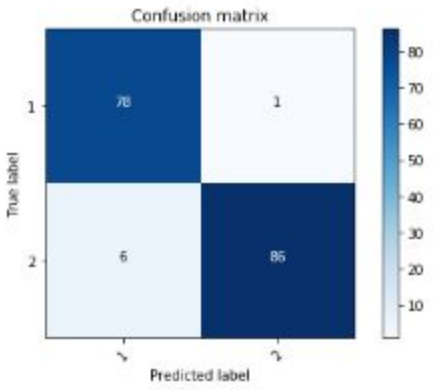
A Decision Tree builds a classification model in the shape of a tree. It classifies the data into smaller subsets or decision nodes or leaf nodes. A decision node normally has two branches while a leaf node may be a decision node or classification. Unlike some models, decision trees can handle both categorical and numerical data.

```
In [75]: yhatDEC = Tree.predict(X)
         DTJaccard = jaccard_similarity_score(y, yhatDEC)
         DTF1 = f1_score(y, yhatDEC, average='weighted')
         print("Avg F1-Score: %.2f" % DTF1 )
         print("Decision Tree Jaccard Score: %.2f" % DTJaccard)

         Avg F1-Score: 1.00
         Decision Tree Jaccard Score: 1.00
```

Support Vector Machines(SVM)

We will also be constructing a model using the SVM method to categorize the possible severity code of an outcome into two-group classifications.

	precision	recall	f1-score	support
1	0.93	0.99	0.96	79
2	0.99	0.93	0.96	92
micro avg	0.96	0.96	0.96	171
macro avg	0.96	0.96	0.96	171
weighted avg	0.96	0.96	0.96	171



Logistic Regression

Logistic Regression is used to predict a binary outcome which can have only two results. Since we only have two possible outcomes in our Severity Code variable, we could expect that Logistic Regression would be a good modelling method given our data.

```
In [77]: yhatLOG = LogR.predict(X)
         yhatLOGproba = LogR.predict_proba(X)
         LogRJaccard = jaccard_similarity_score(y, yhatLOG)
         LogRF1 = f1_score(y, yhatLOG, average='weighted')
         Logloss = log_loss(y, yhatLOGproba)
         print("Log Loss: : %.2f" % Logloss)
         print("Avg F1-Score: %.4f" % LogRF1)
         print("LOG Jaccard Score: %.4f" % LogRJaccard)

         Log Loss: : 0.30
         Avg F1-Score: 1.0000
         LOG Jaccard Score: 1.0000
```

4. Results

After all the data preprocessing which we have made, which included, removing several variables, removing missing values and balancing out the SEVERITYCODE, we can say that the models we have developed are fairly accurate based on the 30% testing. Of all the models, the Logistic Regression and Decision Tree are the best predictive model given its Jaccard and F-1 scores of 1.0.

Model	**Jaccard Score**	**F-1**	**Logloss**
KNN	0.94	0.94	
Decision Tree	1	1	
SVM	0.98	0.98	
Logistic Regression	1	1	0.3

5. Future studies

The models we have developed can help others in determining what situations to consider in order to avoid having car accidents. This study could be further improved with additional data as well as the inclusions of other variables that might affect the severity of an accident. We may also take another study on how findings from our models would affect existing road policy.