Predicting the probability of physical or property damage during a road trip.

By: Darnel Lloyd

1. Introduction

1.1 Background

It is estimated that each year, over 1.5 million people lose their life due to road fatalities. These stem from many factors such as driving under the influence, vehicle failure, reckless driving, road conditions, freak accidents etc. During adverse weather conditions, there is a known increase in the probability of getting into a fatal accident, however, during good weather conditions the same probability is not directly correlated.

1.2 Aim

Given the historical data provided on road accidents in Seattle, we seek to predict the probability of an accident resulting in either physical or property damage based on different road and weather conditions.

1.2 Interest

This information will be directly beneficial to the average road user or road users who travel during adverse weather conditions such as during heavy rainfall or snow. By being able to predict the likelihood of an accident during different weather, given the road condition, users will have a better understanding of the rick factors involved in their road journey which could save their life.

2.Data accusation and cleaning

2.1 Source

For this project, historical data containing vehicle collisions from the year 2004 to present was used. The data was collected in Seattle, USA.

2.2 Cleaning and Feature selection

2.2.1 Feature selection

The raw data set contained 194673 rows and 38 columns, some of which contained null values. Due the computational constraints, a subset of this data set containing 1000 entry rows were taken. Various features where present in the data set (figure. 1).

```
df.columns
```

```
Index(['SEVERITYCODE', 'X', 'Y', 'OBJECTID', 'INCKEY', 'COLDETKEY', 'REPORTNO',
       'STATUS', 'ADDRTYPE', 'INTKEY', 'LOCATION', 'EXCEPTRSNCODE',
       'EXCEPTRSNDESC', 'SEVERITYCODE.1', 'SEVERITYDESC', 'COLLISIONTYPE',
       'PERSONCOUNT', 'PEDCOUNT', 'PEDCYLCOUNT', 'VEHCOUNT', 'INCDATE',
       'INCDTTM', 'JUNCTIONTYPE', 'SDOT_COLCODE', 'SDOT_COLDESC',
       'INATTENTIONIND', 'UNDERINFL', 'WEATHER', 'ROADCOND', 'LIGHTCOND',
       'PEDROWNOTGRNT', 'SDOTCOLNUM', 'SPEEDING', 'ST_COLCODE', 'ST_COLDESC',
       'SEGLANEKEY', 'CROSSWALKKEY', 'HITPARKEDCAR'],
      dtype='object')
```

Figure 1.

Giving the aim of the research, the features *'SEVERITYCODE','LIGHTCOND', 'WEATHER', 'ROADCOND'* were selected as appropriate parameters for building the model. These were chosen because SeverityCode provided a numerical entry (1 or 2) for the outcome of the collision. Lightcond provided data on different lighting condition on the road. Weather is self-explanatory and gave a breakdown of the weather condition. Roadcond provided data on the physical condition of the road at the time of the collision. All of which were directly relevant to the aim of the research.

2.2.2 Cleaning

With the exception of SEVERITYCODE, *'LIGHTCOND', 'WEATHER', 'ROADCOND'* were categorical entries, which needed to be converted into binary entries (figure 2). From each of the chose categories, the field title “unknown” and “other” were dropped. Seeing that these fields provided no direct/translatable data, they were not relevant to this research.

LIGHTCOND	WEATHER	ROADCOND
Daylight	Clear	Dry
Dark- Street Lights On	Raining	Wet
Unknown	Overcast	Unknown
Dusk	Unknown	Snow/Slush
Dawn	Fog/Smog/Smoke	
Dark – Street Lights Off	Other	
Dark – No Street Lights	Snowing	
Other	Sleet/Hail/Freezing Rain	

Figure 2. Feature columns and entries.

After removing these entries, our final data set contained all the relevant columns, also with usable data types (figure 3). The data set was now clean and ready to be analysed.

```
dfx2.dtypes
```

```
SEVERITYCODE                int64
Clear                       uint8
Fog/Smog/Smoke              uint8
Overcast                    uint8
Raining                     uint8
Sleet/Hail/Freezing_Rain    uint8
Snowing                     uint8
Dry                         uint8
Ice                         uint8
Snow/Slush                  uint8
Wet                         uint8
Dark - No Street Lights     uint8
Dark - Street Lights Off    uint8
Dark - Street Lights On     uint8
Dawn                        uint8
Daylight                    uint8
Dusk                        uint8
dtype: object
```

Figure 3. Feature columns and data type

3 Data Analysis

3.1 Target Variable

As mention initially, our target variable was “SEVERITYCODE” where an outcome of 1 indicated property damage and 2 indicated physical injury. By examining our data set, we can see that this field was well represented (Figure 3).

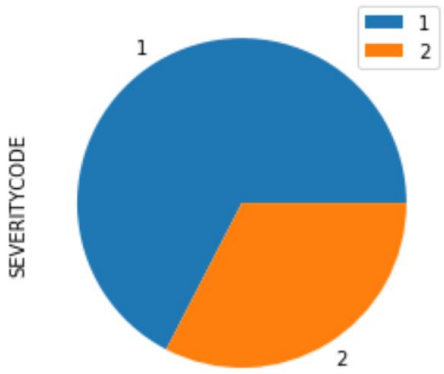
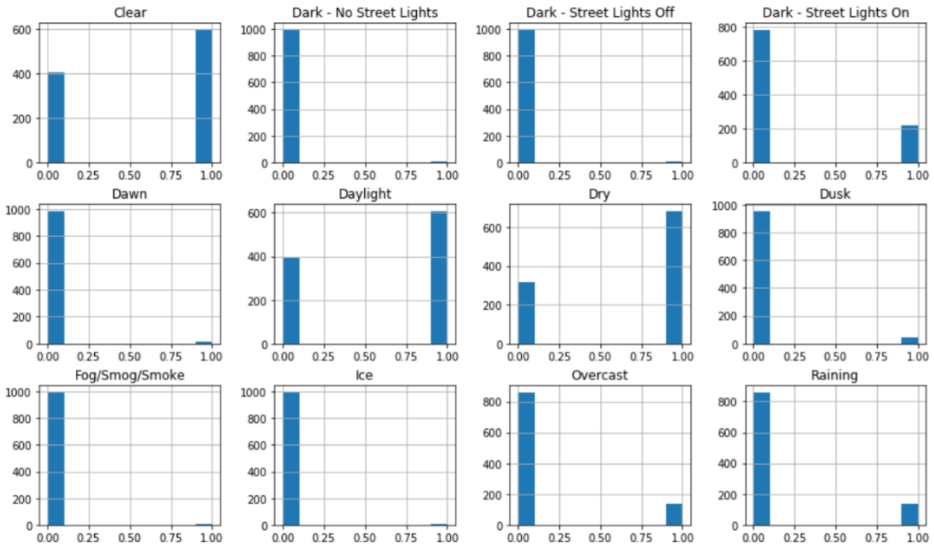


Figure 4. Breakdown of the field SEVERITYCODE

There were no huge disparity between the outcome which, if present, might have not been good when building a predictive model. The model would not have had sufficient data to build knowledge of outcome number 2.

3.2 Feature sets analysis.

Next, we examined our feature sets to have an idea of the spread our or data. This gave us a broad understand of the total number of features as well as an idea as to which feature might have had the largest weighting on our model. Furthermore, we can also identify possible outliers in the data set.



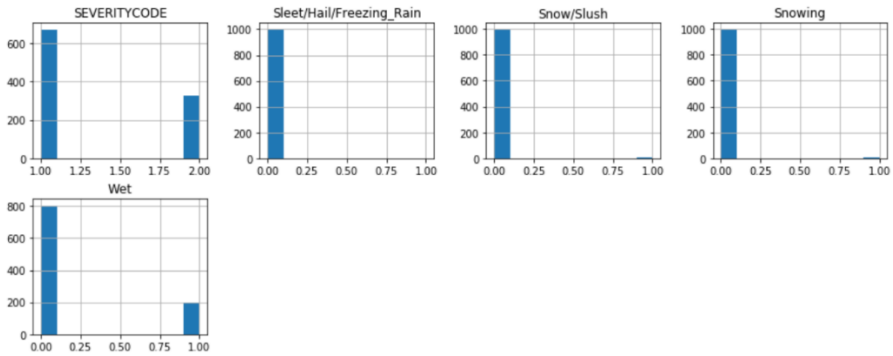


Figure 5. Bar graph representations of feature sets.

The above bar graphs show the composition of each feature set where 0 represent the number of no entries and 1 represent the number of entries recorded for that specific field. From a general overview we observe that there are more 0 recorded entries. Furthermore, “Clear”, “Day light” and “Dry” presented the most recorded entries in our data set while “Sleet/Hail/Freezing_Hail” had the least recorded entries.

3.3 Correlation

Another way to decide whether or not the chosen features had an impact on the target variable was to calculate the correlation among these sets. We used the corr_matrix function to achieve this.

```
corr_matrix = dfx2.corr()
```

```
corr_matrix['SEVERITYCODE'].sort_values(ascending=False)
```

```
SEVERITYCODE                1.000000
Clear                       0.124793
Dry                         0.111603
Daylight                    0.106961
Dusk                        0.048430
Dark - No Street Lights     0.023524
Fog/Smog/Smoke              0.023524
Dawn                       -0.010241
Ice                        -0.010275
Wet                        -0.019223
Raining                    -0.020436
Sleet/Hail/Freezing_Rain   -0.022004
Overcast                   -0.024667
Dark - Street Lights Off   -0.026409
Snow/Slush                 -0.038150
Snowing                    -0.038150
Dark - Street Lights On    -0.046853
Name: SEVERITYCODE, dtype: float64
```

Figure 6. Correlation matrix

It can be noted that the feature sets “Clear” and “Dry” were the most positively correlated to the SEVERITYCODE, while “Dawn” is the most negatively corelated. There were also no NAN features which indicated that they all shared some form of corelation to SEVERITYCODE.

4. Data Modelling

The identified problem for this research was that of a classification type. We tired to predict whether out outcome would be either a 1 or 2. For this purpose K nearest Neighbour, Decision Tree, Logistic Regression and Support Vector Machine algorithms were used to develop our models. These models followed a sequence of FIT – PREDICT – TEST.

4.1 Model types

4.1.1 K Nearest Neighbour

A prediction model based on the KNN (K nearest neighbour) algorithm is developed from finding the best value for “K”. We identified this value, and for our model this value was 22.

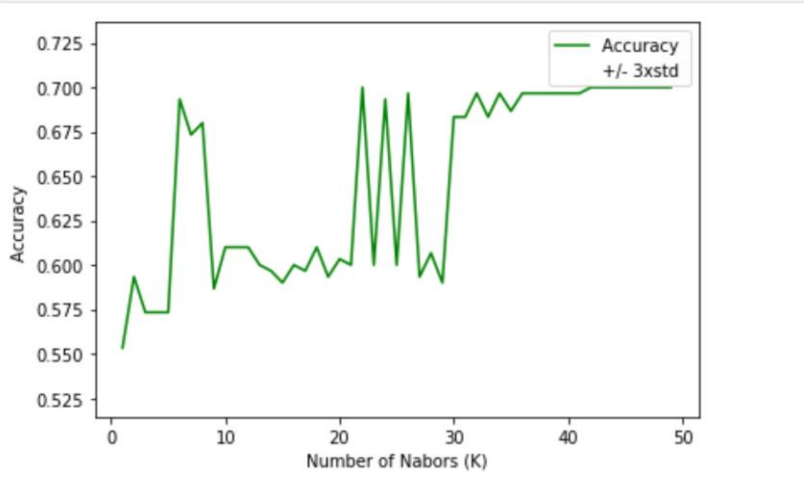


Figure 7. Finding the best value for K

Using this value, we then formulated the KNN model.

4.1.2 Logistic Regression

For the logistic regression model, a liblinear solver was used and a C parameter of 0.3.

```
LR = LogisticRegression(C=0.3, solver='liblinear').fit(X_train, Y_train)
yLR = LR.predict(X_test)
```

Figure 8. Logistic regression model

The C parameter represented the inverse of regularization strength in our model and at 0.3 it was an adequate fit. After running the mode, a confusion matrix was used to examine its performance.

```
confusion matrix for Logistic Regression:
[[200  88]
 [ 10   2]]
```

Figure 9. Confusion matrix for Logistic regression.

4.1.3 Decision Tree

When formulating a decision tree model, one of the key steps is to identify the depth of the decision tree. With the use of a function, looping through several given depths, the ideal one could have been identified.

```
def my_func():
    Ds = 6
    mean = np.zeros((Ds-1))

    for n in range(1,Ds):

        Dtree = DecisionTreeClassifier(criterion='entropy', max_depth = n)
        Dtree.fit(X_train, Y_train)
        predTree = Dtree.predict(X_test)
        mean[n-1] = metrics.accuracy_score(Y_test, predTree)
        mean.argmax()+1

        return print(" Best depth for decision tree is", mean.argmax()+1)
```

Figure 10. Maximum depth for decision tree.

For the purpose of this research, a depth of 1 was used.

4.1.4 Support Vector Machine (SVM)

For our SVM model, four different types of kernels were fitted to separate models and then tested. Rbf, linear, poly and sigmoid were the different kernel functions used. Each function achieved a different performance so the aim was to identify the one which would provide the highest accuracy for our model.

```
#building the model using different kernel functions

clf = svm.SVC(kernel='rbf')
clf1 = svm.SVC(kernel='linear')
clf2 = svm.SVC(kernel='poly', degree=6)
clf3 = svm.SVC(kernel='sigmoid')
```

Figure 11. SVM kernel functions

5. Accuracy scores

Algorithm	Accuracy Score
K Nearest Neighbour	0.7
Logistic Regression	0.673
Decision Tree	0.7
SVM - rbf	0.67
SVM - Linear	0.67
SVM - poly	0.66
SVM - sigmoid	0.697

Figure 11. Accuracy scores for various models.

From the table above, we note that the highest test set accuracy was achieved using K nearest neighbour and also using a decision tree. Additionally, the support vector machine using a sigmoid function also produced roughly the same accuracy at 0.697. On the other hand, the worst results were obtain using support vector machine with a poly function.

6. Conclusion

The aim of this study was to find the best model to predict the categorical outcome of whether or not a person is more likely to end up with property of physical damage as a result of a vehicular accident. As was observed in figure 2, the majority of accidents only resulted in property damage, so our model is expected to reflect this phenomenon. In order to most accurately predict the outcome of the research, either the K Nearest Neighbour or the Decision Tree model can be used.

It can be noted, that with further fine tunning of the parameters and feature sets, I might be possible to achieve even better results with either of these two chosen models.