Predicting Car Accident Severity

Luigi De Marco

1 Introduction

While car accidents are typically considered to be random events, there are a large number of factors that can influence the severity of an accident. The severity of an accident can be classified according to what type of damage occurred as well as whether any injuries or fatalities were involved. The factors that can influence the severity of an accident can include, for example, weather, visibility, or whether excessive speed was involved. As such, it stands to reason that a predictive model can be used to describe the correlations between various factors and the likelihood of accidents of a given severity occurring.

The development of an accurate model to predict accident severity has significant implications for commuters and city traffic planners alike. If it is well understood which factors increase the probability of severe accidents occurring, a warning system may be put in place to alert commuters to drive particularly carefully or change travel plans. Likewise, if certain conditions are likely to cause severe accidents, city planners may shut roads and divert traffic while those conditions prevail.

Here, we build a machine learning model to predict the severity of car accidents based on a variety of factors. In what follows, we will describe the data set we will use to build such a model.

2 Data

2.1 Introduction to the Data Set

To build a model to predict the severity of accidents, we use a dataset provided by the Seattle Department of Transportation (SDOT). The raw data may be accessed online at https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv.

The data contains 194,673 records of accidents in the Seattle area from 2004 to present. Each record corresponds to an accident of a given severity. The severity of the accident is labeled according to whether property damage occurred or whether an injury was incurred. We note that this is an imbalanced data set, with property damage incidents occurring roughly three times more frequently than injury incidents.

In addition to the severity label, each record contains 37 additional attributes describing the conditions under which the accident occurred. These include numerical attributes, such as the

number of people involved in a collision and the number of vehicles involved, as well as categorical attributes, such as the weather conditions when the incident occurred and whether or not a parked car was hit. A full description of the attributes can be found online at https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Metadata.pdf.

	SEVERITYCODE	INCDTTM	ADDRTYPE	WEATHER	ROADCOND	LIGHTCOND
0	2	3/27/2013 2:54:00 PM	Intersection	Overcast	Wet	Daylight
1	1	12/20/2006 6:55:00 PM	Block	Raining	Wet	Dark - Street Lights On
2	1	11/18/2004 10:20:00 AM	Block	Overcast	Dry	Daylight
3	1	3/29/2013 9:26:00 AM	Block	Clear	Dry	Daylight
4	2	1/28/2004 8:04:00 AM	Intersection	Raining	Wet	Daylight

The above table shows an example of the raw data that will be used to build a model. Each incident is labeled by a severity code ('SEVERITYCODE') that indicates whether only property damage occurred (label 1) or whether injury was involved (label 2). In addition, each record contains attributes describing the conditions of the incident. In the above example, these attributes include: the date and time of the incident ('INCDTTM'), the location type at which the incident occurred ('ADDRTYPE'), the weather conditions ('WEATHER'), the road conditions ('ROADCOND'), and the light conditions ('LIGHTCOND'). Intuitively, we expect these to be good predictors of accident severity. In the next section, we will analyze the features more carefully and determine which may give rise to the best model.

2.2 Feature Selection

Since the goal is to build a warning system for commuters and city planners, it is necessary to choose features that are predictive. That is, we must use features that a user could feed into the model *before* a drive starts. So while the number of vehicles involved in a collision might be a good predictor of accident severity, this is not something that could be useful in a warning system. Instead, we will focus on features such as weather conditions, road conditions, and light conditions to make predictions before severe accidents occur.

Within our data set, there are a number of such features that can be used, and these are summarized in Table 1. With the exception of ADDRTYPE, the features listed in the table can all known before a driver departs. These can all be easily determined based on, for example, the time and date and local weather reports. Even though ADDRTYPE is not a predictive variable in the sense that it's not possible to know beforehand where a collision will occur, it can be useful to help drivers plan their route. For example, if severe collisions are more likely to occur at intersections, a driver could plan their route to avoid heavily busy intersections.

The SEASON feature was generated from the date-time stamp for each incident report, and it is

Feature	Description
WEATHER	Weather conditions when collision occurred
ROADCOND	Road conditions when collision occurred
LIGHTCOND	Light conditions when collision occurred
SEASON	Season during which collision occurred
ADDRTYPE	Location type where collision occurred

Table 1: Description of features used to build predictive model.

meant to be an indicator of the time of year and not the weather. Each season corresponds to a range of three months, with the first month being that in which the season starts. For example, the range March–May corresponds to spring.

We expected the selected features to be related, but not redundant. For instance, we know it will be far more likely for the road conditions to be wet during heavy rainfall, but it is not always the case that we have a perfect correlation. A collision which occurs immediately after heavy rainfall may be in clear weather with wet road conditions.

2.3 Data Cleaning

Before the data could be used to build a predictive model it is necessary to clean and process it.

Of the original 194,673 rows in the data set, 7,148 of those rows contained at least one NaN value for the selected features. Furthermore, 17,744 entries were labeled 'Unknown' for at least on of the features. Since these missing values are a small fraction of the total number of rows, they were simply discarded, leaving us with 169,781 data points.

The original data set has two unique entries for SEVERITYCODE, the target variable. A value of 1 corresponds to a collision which is not considered severe (property damage only), while a value of 2 corresponds to a severe collision (injury occurred). For convenience these were changed to be 0 and 1, respectively.

Many of the features had categorical values with few records. For example, the WEATHER feature has 32,976 data points corresponding to 'Raining', but only 49 entries corresponding to 'Blowing Sand/Dirt.' For these types of data points, we do not expect to have enough useful information to build a model. Therefore, any feature for which there were fewer than 1,000 counts of a certain value had the corresponding rows discarded.

We group categories within a particular feature that a strongly related. In particular, the LIGHTCOND feature had three different values which all essentially corresponded to dark conditions (Dark - Street Lights On, Dark - Street Lights Off, Dark - No Street Lights), with 95% of the entries being Dark - Street Lights On. These were all combined into a single category 'Dark.'

After the data was cleaned and processed, we were left with 111,013 negative entries (SEVERI-

TYCODE 0) and 54,721 positive entries (SEVERYITYCODE 1). This roughly 2:1 imbalance can make data visualization challenging, and can skew the final predictive model. Since we still have a large number of data points overall, we randomly choose 54,721 out of the 111,013 negative entries to yield a balanced data set. The final balanced data set contains 109,442 entries.

2.4 Data Visualization

Since the severity of car collisions is expected to have a large random component to it, it is useful to visualize data according to how a given feature predicts the deviation of accident severity from perfect randomness. That is, if severe collisions are represented by a 1 and non-severe collisions by a 0, a perfectly uncorrelated feature should give a mean value of 0.5. For convenience, we offset the mean value by 0.5 and multiply by 100 to give a correlation on a percentage scale. An uncorrelated variable has a value of 0.

We begin by looking at how individual features correlate with collision severity. Figure 1 shows how the season and light conditions affect the probability of a severe accident occurring. In the

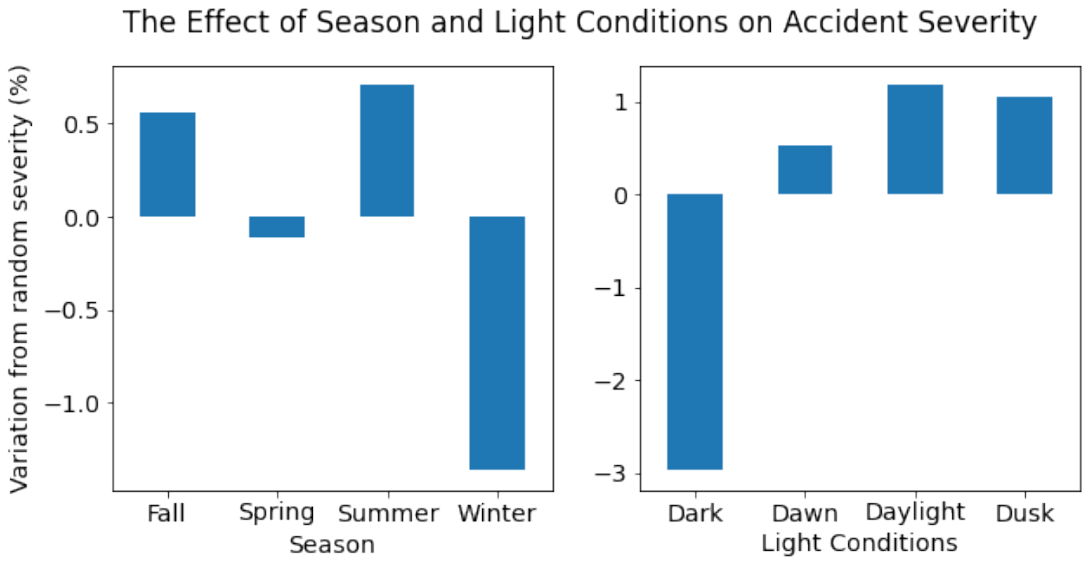

Figure 1: Effect of season and light conditions on accident severity.

left-hand panel, we see that the largest effect of season is that the probability of severe collisions occurring is the smallest in the winter, while it is slightly more likely to have sever accidents occurring in the fall and summer. On the other hand, we see that spring does not have a large effect on the occurrence of severe collisions.

This observation may seem counter-intuitive as the poorer weather in winter may be expected to give rise to more severe collisions than in summer, for example. However, this does not account for the fact that poorer winter weather also encourages drivers to be more prudent, reducing the risk of a severe collision.

A similar effect is observed when we consider light conditions (right panel). We see that the probability of severe collisions is reduced by nearly 3% in dark conditions, while we see a greater propensity (between 0.5 and 1%) for severe accidents in other light conditions. We again attribute this to the greater precautions taken by drivers in dark conditions.

Visualization is more powerful if we consider not only how collision severity is predicted by a single feature, but also by correlations between features. To that end, we generate a pivot table using the season and light condition features to create a 2D visualization of the data. Figure 3 shows these correlations. In looking at Fig. 3, we see stronger effects emerge than in the single

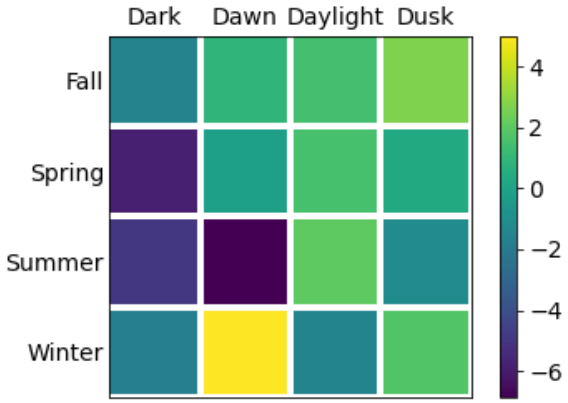

Figure 2: Correlation between season and light conditions on accident severity.

feature case. For example, we see that collisions which occur at dawn in summer are nearly 6% less likely to be severe. This is impressive considering that each of those individual factors taken independently were not seen to be a good predictor of accident severity in Fig. 1. Likewise, we see that the majority of severe accidents occur at dawn in the winter. It is these correlations between features in predicting accident severity that will allow us to build an accurate model.

Initially, we intended to use ADDRTYPE as an additional factor that can alert drivers whether or not to avoid intersections, for example. However, upon inspection of the data, it became clear that the strong correlation of ADDRTYPE with SEVERITYCODE would limit any model's predictive utility, as importance would be disproportionally placed on ADDRTYPE. This is evident in the figure below, where we see that it is 10% more likely for an accident to be severe at an intersection compared to 7.5% less likely if it occurred on a block.

To build a predictive model, then, we do not include ADDRTYPE as a feature. We use the four features described in Table 1.

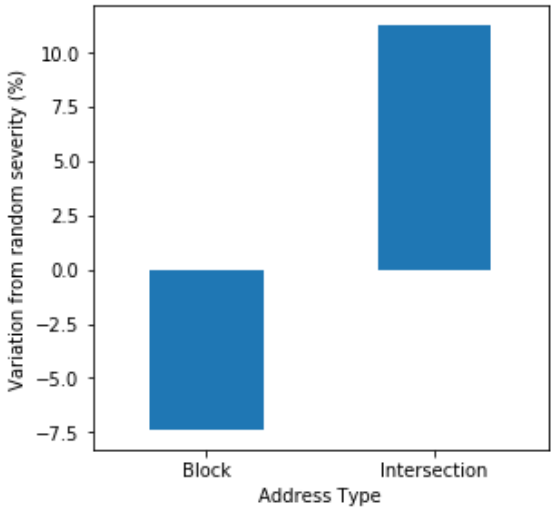

Figure 3: Correlation between address type and accident severity.

3 Model Development and Results

3.1 Classification Models and Data Sets

Since this is a supervised classification problem, there are a number of common classification algorithms we may use. For this work, we will use the K-nearest neighbors (KNN) algoritm, logistic regression, state-vector machine (SVM), and decision tree.

In order to develop the models, the data set is split into three separate data sets. 1) A training set for training the model, 2) a cross-validation (CV) set to tune hyperparameters (e.g. the value of K in KNN), and 3) the test set to evaluate the model's accuracy and ability to generalize. The use of a separate CV set is important to ensure that the accuracy of the model is not biased by the fact that the test set was used to select hyperparameters. The data was randomly split in a 3:1:1 ratio for the training, CV, and test sets, respectively.

3.2 Hyperparameter tuning

For each classifier, we train several different models with different hyperparameters. The accuracy of the model is determined by calculating the Jaccard index, defined as

$$J = \frac{M_{11}}{M_{11} + M_{10} + M_{01}}, \tag{1}$$

where M_{ij} is the number of times the model predicted i when the true value was j. That is, for example, M_{11} is the number of true positives and M_{10} is the number of false positives.

3.2.1 KNN

For the KNN algorithm, we must select the value of K, which is the number of neighbors considered when classifying a particular example. The plot below shows the Jaccard index on the CV set as a function of K.

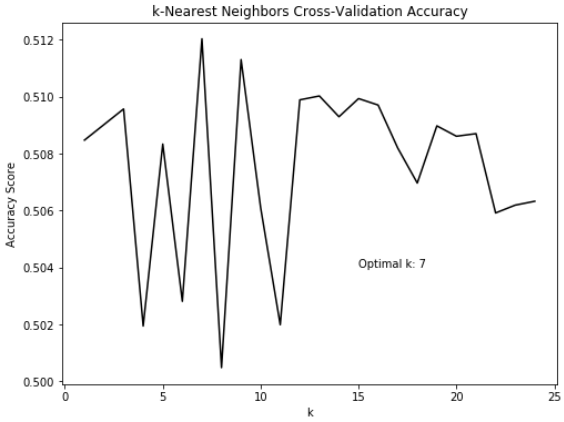

Figure 4: Accuracy of the KNN model on the CV data.

We see that over the range of K values tried, we do not see a large variation in the accuracy of the model. The accuracy shows a small peak for $K = 7$ at a value of 0.512, which shows that the number of true positives is larger than the sum of false positives and false negatives. Although $K = 7$ is the optimum, the plot shows that a few values of K perform roughly as well in the range 2-10.

3.2.2 Logistic Regression

For logistic regression, the hyperparameter we tune is the inverse of the regularization strength, C. If C is very large, then there is little regularization, implying that the model is susceptible to a large variance. If C is very small, this corresponds to a large regularization, suggesting that the model is susceptible to a large bias.

In the plot above, we see that the model becomes susceptible to a large bias for inverse regularization parameters smaller than 10^{-4}. Similarly, accuracy suffers if $C > 10^{-3}$. We therefore select an optimum value of $C = 1 \times 10^{-3}$, which is likely to produces the most accurate model with little bias or variance. The accuracy for this model is 0.518.

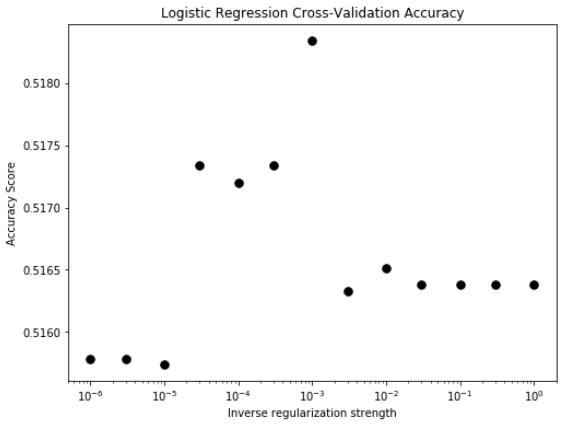

Figure 5: Accuracy of the logistic regression model on the CV data.

3.2.3 State-Vector Machine

For the state vector machine, the hyperparameter is the selection of the kernel function to be used. We generate models using the four basis functions: linear, polynomial, radial basis function (RBF), and sigmoid. We find that the polynomial kernel gives the best accuracy of 0.518, which is only slightly better than the RBF performance.

3.2.4 Decision Tree

For the decision tree model, the hyperparameter of interest is the number of number of layers in the model. We similarly benchmark the model against the cross-validation data, and we find an optimum depth of three layers, which gives an accuracy of 0.518. Adding additional layers reduces the accuracy slightly.

3.3 Results

We begin by summarizing the results of the models, using the hyperparameter values which gave the best accuracy on the cross-validation set. Table 2 shows the Jaccard index for each of the models on the training, CV, and test sets. For each of the data sets, the model with the best performance is highlighted in bold.

From Table 2, it is evident that the SVM is the model which performs best. SVM yielded the best accuracy for all three data sets. However, we also see that Logistic regression and Decision tree models had similar performance. In particular, the accuracy on the test set for all three of these models is 0.52 to within 0.1%. On the other hand, the KNN model had the worst performance of any model, giving rise to significantly worse accuracy for all three data sets.

	Training	CV	Test
K-Nearest Neighbors	0.512	0.512	0.510
Logistic Regression	0.520	**0.518**	**0.520**
State Vector Machine	**0.523**	**0.518**	**0.520**
Decision Tree	0.522	**0.518**	**0.520**

Table 2: Summary of the performance of the best model within each class characterized by the Jaccard index.

In order to better understand these results, it is instructive to look not only at the aggregated accuracy score, but also at the confusion matrix and corresponding F1 score as well as the precision and recall. These results are shown in Table 3, for each model's performance on the test data. In the

	TP	TN	FP	FN	Precision	Recall	F1 Score
K-Nearest Neighbors	6454	**4720**	**6191**	4524	0.510	0.588	0.546
Logistic Regression	7300	4084	6827	3678	0.516	0.665	0.581
State Vector Machine	6987	4385	6526	3991	**0.517**	0.636	0.570
Decision Tree	**7614**	3771	7140	**3364**	0.516	**0.694**	**0.591**

Table 3: Detailed summary of the performance of the best model within each class.

table TP, TN, FP, and FN stand for true positives, true negatives, false positives, and false negatives, respectively. The precision, recall, and F1 score are all defined in the usual way.

We see that that accuracy, as defined by the F1 score, follows the same trend as we saw when considering the Jaccard index. However, the F1 score does a better job differentiating the different models. While Logistic regression, SVM, and decision tree all had the same Jaccard index for the test set, we see that the large recall of the decision tree model makes it more accurate overall.

The decision tree model also does best at predicting true positives and has the lowest number of false negatives. We see that every model has a fairly low precision, which reflects that each model gives rise to a large number of false positives. We discuss this and its ramifications in the following section.

4 Discussion

As stated at the outset, the goal of this project is not only to come up with a predictive model, but to use that model to develop a warning system for commuters. This implies that there are considerations that must be made that go beyond finding the model with the best accuracy. In particular, it's important to have a model that is easy to use and intuitive. To that end, we will discuss each model and consider its strengths and weaknesses.

Before doing so, however, we consider the implications of incorrect predictions. We saw that while all the models have a reasonably low rate of false negatives, they all have a fairly high rate of

false positives (low precision). While greater accuracy in this regard is always desired, for this application, a large false positive rate is not terribly problematic. Since we are building a warning system, it pays off to err on the side of caution. While a false positive may prompt drivers to drive more carefully when unnecessary, there is no major cost associated with this. On the other hand, driving carelessly when conditions are prime for severe accidents could be costly in terms of injury or death. We will therefore value recall over precision.

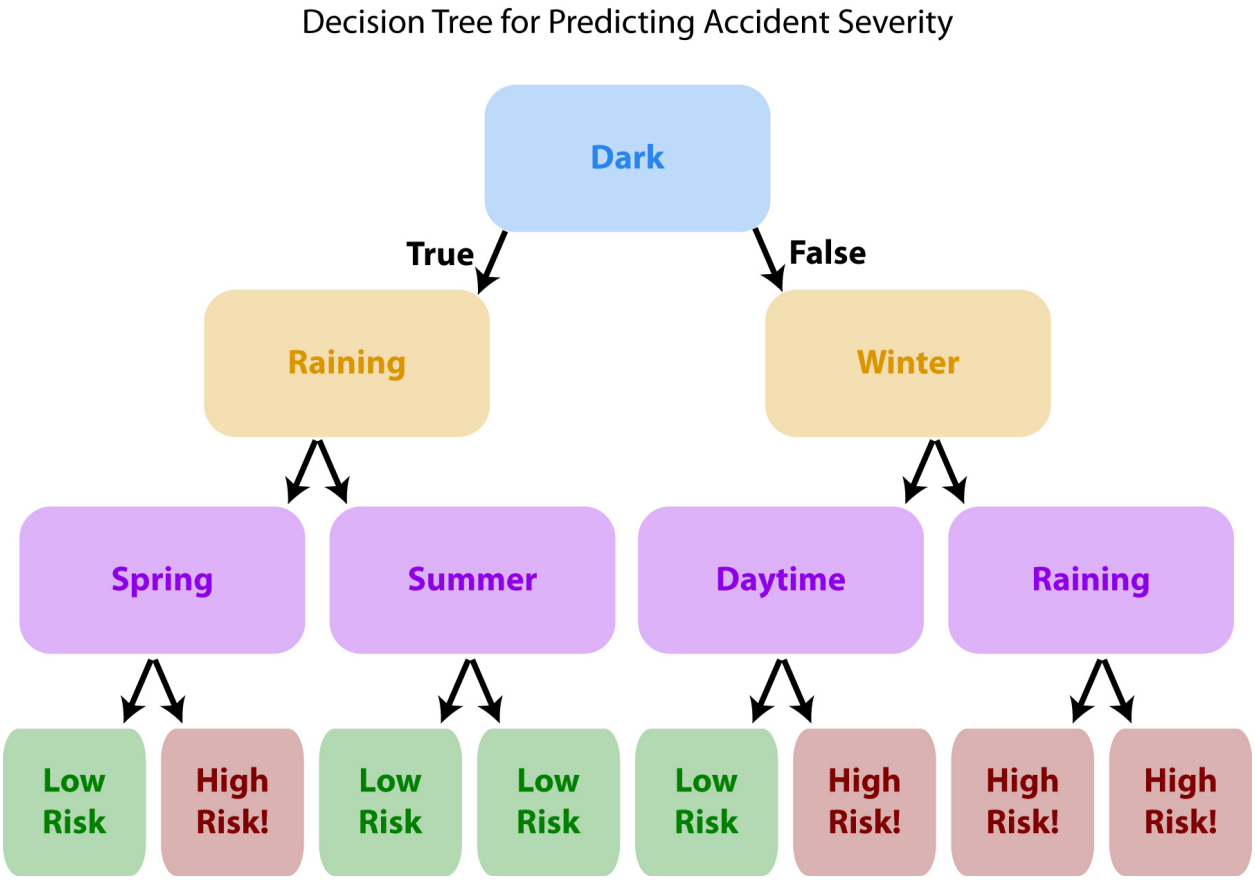

Figure 6: Decision tree model for predicting accident severity.

We found that the KNN model was the least accurate, as measured by both the Jaccard index and the F1 score. In particular, the KNN model has the lowest rate of true positives of the models tested. It is tempting to discard this model based on this alone, but KNN does have the advantage of providing some intuitive understand since it essentially compares cases to similar ones. However, another major issue is the fact that training KNN and using it is computationally slow since any new test point must be compared against the entire training data.

Logistic regression and SVM share many similar properties. While the former is more accurate based on the F1 score the two models show similar precision, and neither model is particularly intuitive. (Though the coefficients of the logistic regression model do provide a metric of the relative importance of each feature.) In addition, the SVM model is very slow to train and use, which is of course a large disadvantage. For this reason (and because of the better performance)

logistic regression is the superior of the two models.

Finally, we consider the decision tree model. This model boasts the best accuracy, as measured by the F1 score. While the precision of the decision tree model is similar to logistic regression and SVM, its recall is superior and this gives the larges F1 score of 0.591. In addition to being the most accurate model, the decision tree comes with the added benefit of being extremely simple and intuitive. In particular, with only three layers, the decision tree is an excellent tool for a warning system.

In Fig. 6 we show a visualization of the decision tree model, which a commuter could easily use to understand the risk of severe accidents. In this visualization, moving down and left corresponds to the condition being true, while moving down and right corresponds to the condition being false.

For example, consider that a driver must commute on a rainy winter day. We see that we move down the tree as Dark$\rightarrow$ False, Winter$\rightarrow$ True, Daytime$\rightarrow$ True, and we find that there is a low risk of severe accidents. However, if the daytime condition was false (i.e. it is dawn or dusk), then this would correspond to a situation in which the risk of severe accidents was high.

5 Conclusions and Outlook

In this work, we used data provided by the Seattle department of transportation to build a warning system that advises commuters of the possibility of sever accidents. After cleaning the data and selecting important predictive features, we employed a variety of supervised classification algorithms to make predictions on accident severity.

While every model had its own strength and weaknesses, we found that the decision tree model has the best performance in terms of accuracy, ease of use, and intuitiveness. The final decision tree model we used has a depth of 3. The precision and recall of this model on the test data were 0.516 and 0.694, respectively, yielding an F1 score of 0.591. Furthermore, by visualizing the decision tree we were able to construct a simple easy-to-use guide for predicting accident severity based on easily known variables. This guide is shown in Fig. 6.

The random nature of automobile collisions makes a high accuracy model difficult to develop. However, with more data types, we expect to build more accurate and more general models. For example, data which includes *how* severe the accident was, as oppose to a binary severity index, may be more useful. Likewise, this data set did not contain enough points considering snowy roads, which may be a powerful predictor.