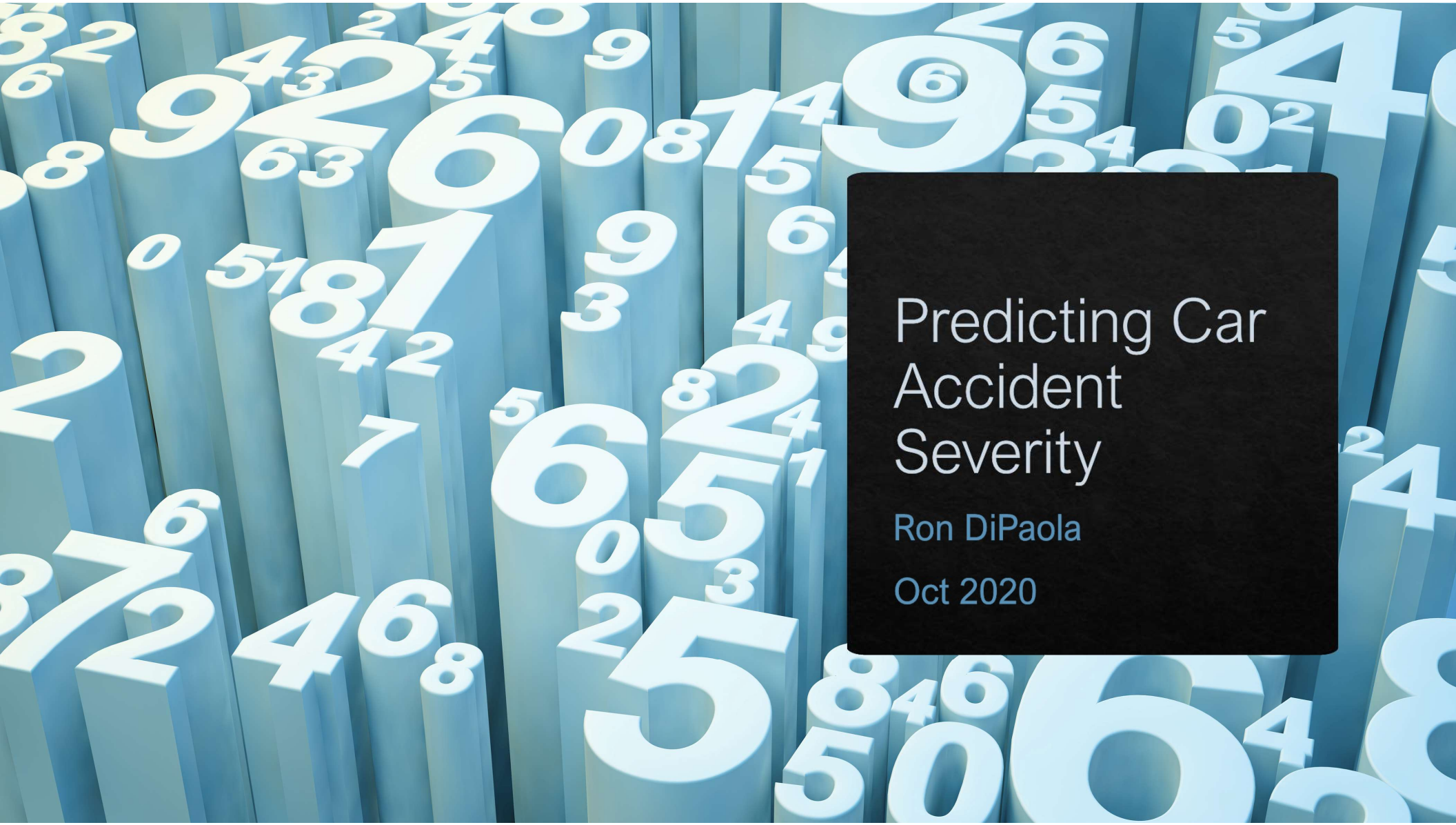


Predicting Car Accident Severity

Ron DiPaola

Oct 2020

Predicting Car Accident Severity Helps Us All

- 36k fatal crashes in 2018
- 2.7 million people injured in 2018
- Leading cause of death for youth (16-20) & young adults (21-24) in 2018
- Costs are also economic and societal exceeding $836 billion US in 2010
- Helping reduce the number of accidents and being able to respond to accidents can help bring these numbers down
- Predicting the severity of accidents can alert stakeholders conditions are present which are highly associated with collisions

Data Acquisition and Cleaning

- The Seattle Department of Transportation (SDOT) collects and publishes accident data through their Open Data Portal (https://data.seattle.gov/), available to all.
- For this project the data was provided by Coursera
- Data runs from Jan 2014 through May 2020 and has 194k rows and 37 features
- Attributes with missing, “unknown” or “other” values were dropped
- Cleaned data contains 8 features

Features Used in Modeling

Field	Description
WEATHER	A description of the weather conditions during the time of the collision.
ROADCOND	The condition of the road during the collision.
LIGHTCOND	The light conditions during the collision
ADDRTYPE	Collision address type: • Alley • Block • Intersection
COLLISIONTYPE	Collision type
JUNCTIONTYPE	Category of junction at which collision took place
PEDCOUNT	The number of pedestrians involved in the collision.
PEDCYLCOUNT	The number of bicycles involved in the collision.

Balancing the Data

The data set also had a bias towards one type of accident (Severity 1), so we resampled to reduce the bias giving us an equal amount of severity code records to run through our model.

```
# rebalancing data, see counts of target variables
df_new['SEVERITYCODE'].value_counts()
```

```
1    95921
2    49448
Name: SEVERITYCODE, dtype: int64
```

```
# resample the dataframe to be balanced
from sklearn.utils import resample
```

```
sev_one_df = df_new[df_new.SEVERITYCODE==1]
sev_two_df = df_new[df_new.SEVERITYCODE==2]

balanced_df_sev_one = resample(sev_one_df,replace=False,n_samples=49448,random_state=20)

balanced_df = pd.concat([balanced_df_sev_one,sev_two_df])
```

```
balanced_df.SEVERITYCODE.value_counts()
```

```
2    49448
1    49448
Name: SEVERITYCODE, dtype: int64
```

Test/Train Split

- The cleaned and balanced dataset was split into testing and training subsets (containing 30% and 70% of the samples) using the scikit learn “train_test_split” method.

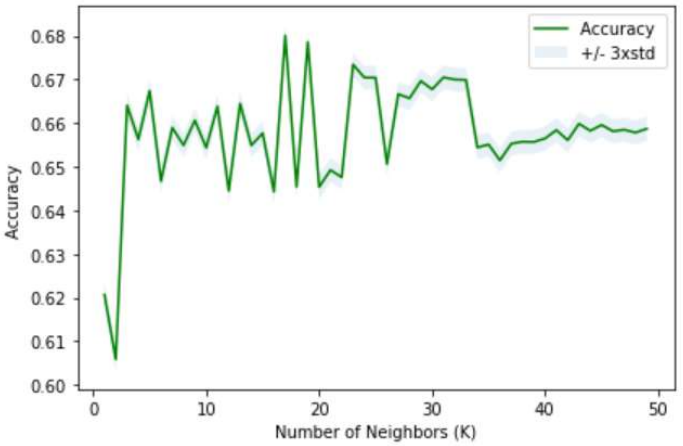
```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=4)
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_test.shape,  Y_test.shape)
```

```
Train set: (69263, 7) (69263,)
Test set: (29685, 7) (29685,)
```

- The data is separated to train the models based the training data and then take the model after it has been trained and run it against test data to see how it performed. Each of the models will use the same training and testing sets so we can evaluate their accuracy.

Machine Learning: K-Nearest Neighbors

- The K-Nearest Neighbors algorithm is a classification algorithm which takes a bunch of labelled points and uses them to learn how to label other points. This algorithm classifies cases based on their similarity to other cases. In K-Nearest Neighbors, data points that are near each other are said to be neighbors. K-Nearest Neighbors is based on this paradigm.




```
print( "The best accuracy was with", mean_acc.max(), "with k=", mean_acc.argmax()+1)
```

```
The best accuracy was with 0.6801038120597256 with k= 17
```

Machine Learning: Support Vector Machine

- A Support Vector Machine (SVM) is a supervised algorithm that can classify cases by finding a separator. SVM works by first mapping data to a high dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. Then, a separator is estimated for the data.

```
# Support Vector Machine
from sklearn import svm
SVM_model = svm.SVC(kernel='linear')
SVM_model.fit(X_train, Y_train)
```

```
SVC(kernel='linear')
```

```
SVM_yhat = SVM_model.predict(X_test)
```

Machine Learning: Decision Tree

- Decision tree models identify the key features on which the data can be partitioned (and the thresholds at which to partition the data) in the hope of arriving, after some iterations, at “leaves” which contain only accidents belonging to one target variable value.

```
#decision tree

from sklearn.tree import DecisionTreeClassifier
DT_model = DecisionTreeClassifier(criterion="entropy")#, max_depth = 8)
DT_model.fit(X_train,Y_train)
```

```
DecisionTreeClassifier(criterion='entropy')
```

```
DT_yhat = DT_model.predict(X_test)
```

Machine Learning: Logistic Regression

- Logistic regression is a statistical and machine learning technique for classifying records of a dataset based on the values of the input fields. Logistic regression is analogous to linear regression but tries to predict a categorical or discrete target field instead of a numeric one. In linear regression, we might try to predict a continuous value of variables such as the price of a house, blood pressure of a patient, or fuel consumption of a car. But in logistic regression, we predict a variable which is binary such as yes/no, true/false, successful or not successful.

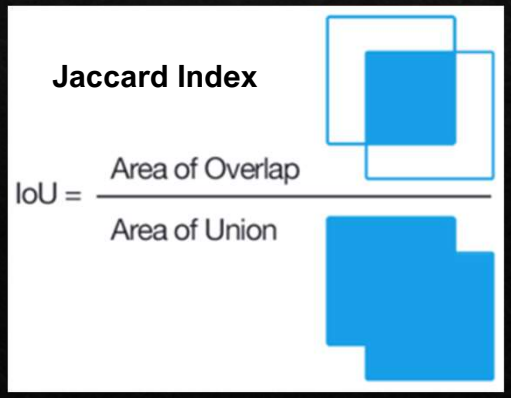
```
#Logistic Regression
from sklearn.linear_model import LogisticRegression
LR_model = LogisticRegression(C=0.01).fit(X_train,Y_train)
```

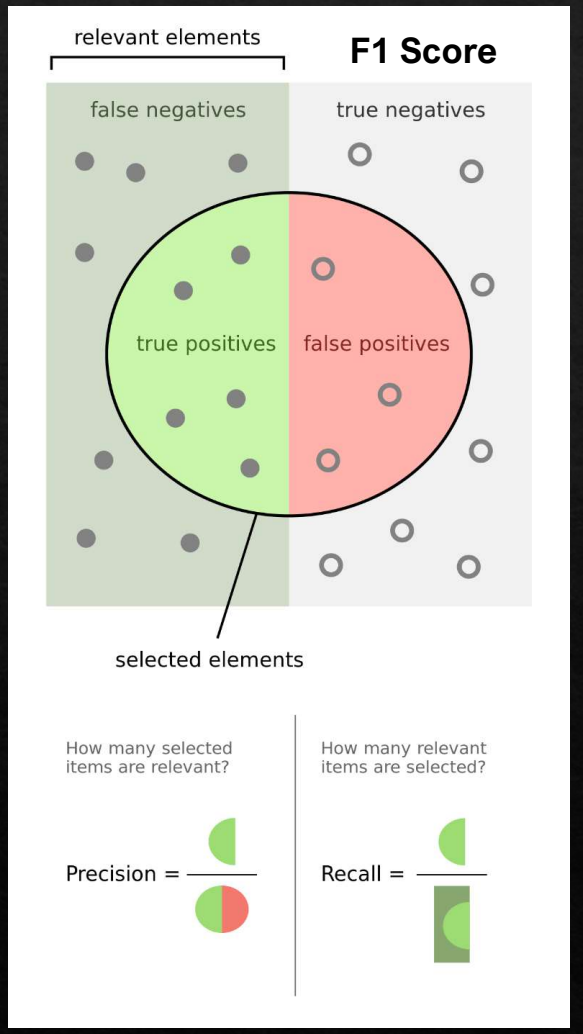
```
LR_yhat = LR_model.predict(X_test)
```

Results

A decision tree slightly outperforms K-Nearest Neighbor for the best model for predicting the severity of accidents. None of the models is particularly great, but additional data features could help bring the accuracy up and make the predictions better.

Model	F1 Score	Jaccard Index
K Nearest Neighbor	0.677	0.472
Decision Tree	0.682	0.461
SVM	0.628	0.470
Logistic Regression	0.627	0.470





Conclusion

- Severity of an accident can be predicted with medium accuracy ~70% of the time with the features used in this model.
- SDOT could use this model to warn stakeholders conditions favour severe accidents.
- SDOT can alter electronic speed limit signage, suggest an increased police presence in an area to help prevent accidents.
- SDOT can also warn first responders and medical facilities of the chances of an accident victim being in a critical state.
- More data and additional features can help improve the accuracy of the model, such as using sensors to measure traffic density and speed.