CAR ACCIDENT SEVERITY REPORT

Coursera Capstone Project

Sep' 20

CONTENT

1. Data set
2. Data preparation
3. Unbalanced classes
4. Data normalization
5. Build a model
6. Results and Evaluation

What	All type of collisions
Who	SDOT Traffic Management Division
When	Data from 2004 till now
Why	To develop an algorithm to reduce collisions

DATA SET SUMMARY

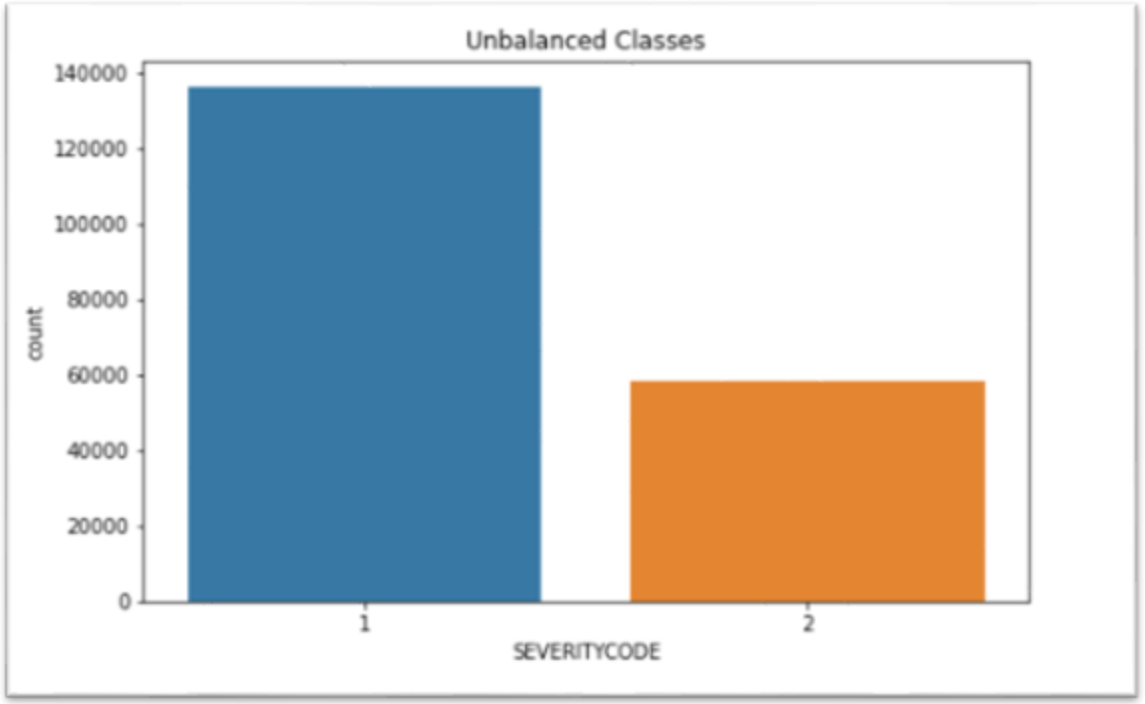
1. Remove unnecessary data

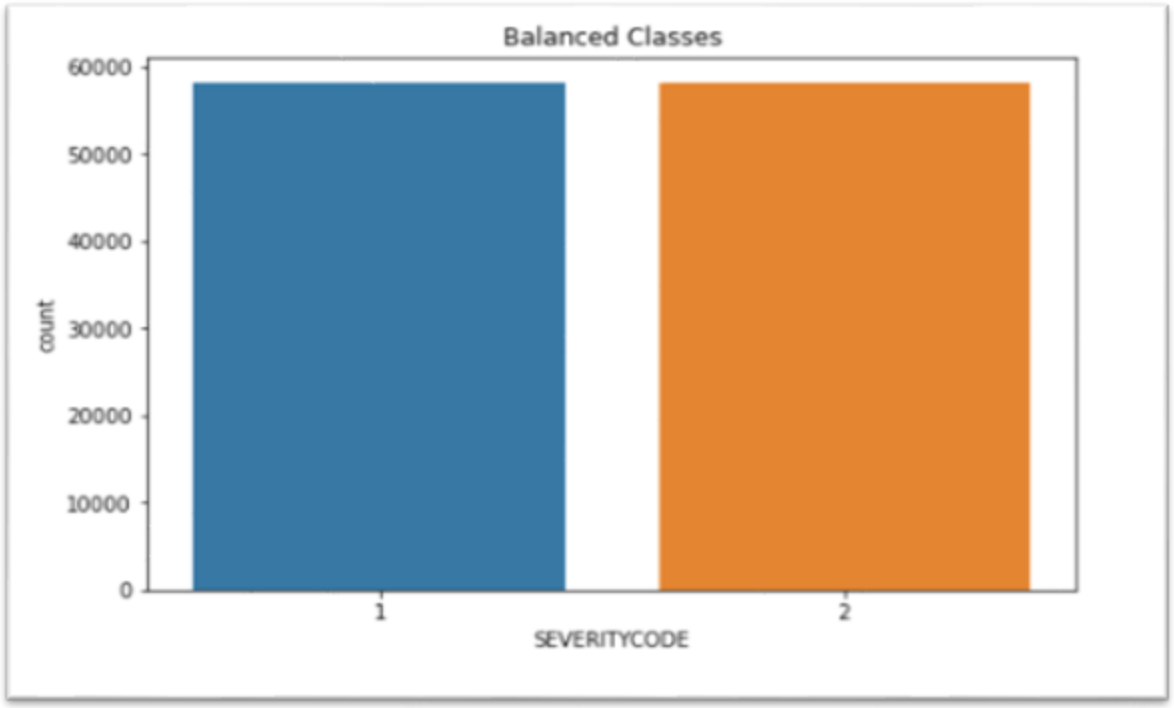
	SEVERITYCODE	WEATHER	ROADCOND	LIGHTCOND
0	2	Overcast	Wet	Daylight
1	1	Raining	Wet	Dark - Street Lights On
2	1	Overcast	Dry	Daylight
3	1	Clear	Dry	Daylight
4	2	Raining	Wet	Daylight

	SEVERITYCODE	WEATHER	ROADCOND	LIGHTCOND	WEATHER_CAT	ROADCOND_CAT	LIGHTCOND_CAT
0	2	Overcast	Wet	Daylight	4	8	5
1	1	Raining	Wet	Dark - Street Lights On	6	8	2
2	1	Overcast	Dry	Daylight	4	0	5
3	1	Clear	Dry	Daylight	1	0	5
4	2	Raining	Wet	Daylight	6	8	5

2. Convert categorical data to numerical

DATA SET PREPARATION





UNBALANCED CLASSES

DATA NORMALIZATION

Define X and Y:

```
X = df[['WEATHER_CAT', 'ROADCOND_CAT','LIGHTCOND_CAT']] .values
X[0:5]
```

```
array([[6, 8, 5],
       [4, 0, 2],
       [1, 0, 5],
       [1, 0, 2],
       [1, 0, 5]], dtype=int8)
```

```
y = df['SEVERITYCODE'].values
y[0:5]
```

```
array([2, 2, 2, 2, 2])
```

Normalize dataset:

```
X = preprocessing.StandardScaler().fit(X).transform(X.astype(float))
X[0:5]
```

Train and Test Split (70% of train, 30% of test):

```
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: (81463, 3) (81463,)
Test set: (34913, 3) (34913,)
```

BUILD A MODEL

- K-Nearest Neighbor (KNN) KNN will help us predict the severity code of an outcome by finding the most similar to data point within k distance.

- Decision Tree A decision tree model gives us a layout of all possible outcomes so we can fully analyze the consequences of a decision. In context, the decision tree observes all possible outcomes of different weather conditions.

- Logistic Regression Because our dataset only provides us with two severity code outcomes, our model will only predict one of those two classes. This makes our data binary, which is perfect to use with logistic regression.

```
from sklearn.metrics import jaccard_similarity_score
jaccard_similarity_score(y_test, yhat)
```

```
0.5610517572251024
```

KNN

```
from sklearn.metrics import f1_score
f1_score(y_test, yhat, average = 'macro')
```

```
0.5364827523846807
```

```
from sklearn.metrics import jaccard_similarity_score
jaccard_similarity_score(y_test, predTree)
```

```
0.5605075473319394
```

Decision Tree

```
from sklearn.metrics import f1_score
f1_score(y_test, predTree, average = 'macro')
```

```
0.49779972176286846
```

```
from sklearn.metrics import jaccard_similarity_score
jaccard_similarity_score(y_test, yhat2)
```

```
0.526250966688626
```

```
from sklearn.metrics import f1_score
f1_score(y_test, yhat2, average = 'macro')
```

```
0.5118777863558591
```

Logistic Regression

```
from sklearn.metrics import log_loss
log_loss(y_test, yhat_prob)
```

```
0.684623485318135
```

RESULTS AND EVALUATION