ACCIDENT SEVERITY PREDICTION

19.09.2020

— PRIYANKA JAIN

Problem Statement

To reduce the number of vehicle collisions in a community, we have to predict the severity of an accident given variable features like weather, road and visibility conditions.

The analysis is to predict the severity of accident occurrence using data provided with features like vehicle count ,weather conditions and road conditions in order to inform the public over probability of occurrence of severity of an accident on roads to take well informed measures beforehand. This analysis would help the department of transportation to improve upon the ways and minimize accidents on the roads for the purpose of increasing the quality of life for the residents, increasing transparency, accountability and comparability, promoting economic development and research, and improving internal performance management.

Data Understanding

I have used the data provided in the course.The dataset is available as comma-separated values (CSV) files. The data is also available from RESTful API services in formats such as GeoJSON.Our predictor or target variable will be "SEVERITYCODE" which is used to measure the severity of an accident from 1 to 2 in the dataset . Attributes used to weigh the severity of an accident are "WEATHER","ROADCOND", "LIGHTCOND" and "VEHCOUNT". The data set consists of 37 features from which weather, road condition, light condition and vehicle count were selected based on correlation analysis of each feature to the target variable "SEVERITYCODE".

Data Preparation

The data first needs to be converted to the desired data frame which includes all the meaningful features and target set. Also, the type of some of the features are object which need to be converted to desired types for analysis. I have used label encoding to convert the features to our desired data type.The initial data table has features as shown in *table 1*.

:

	SEVERITYCODE	COLLISIONTYPE	VEHCOUNT	WEATHER	ROADCOND	LIGHTCOND
0	2	Angles	2	Overcast	Wet	Daylight
1	1	Sideswipe	2	Raining	Wet	Dark - Street Lights On
2	1	Parked Car	3	Overcast	Dry	Daylight
3	1	Other	3	Clear	Dry	Daylight
4	2	Angles	2	Raining	Wet	Daylight
...	...	...	...	...	...	...
194668	2	Head On	2	Clear	Dry	Daylight
194669	1	Rear Ended	2	Raining	Wet	Daylight
194670	2	Left Turn	2	Clear	Dry	Daylight
194671	2	Cycles	1	Clear	Dry	Dusk
194672	1	Rear Ended	2	Clear	Wet	Daylight

194673 rows × 6 columns

Table 1

Here the target variable is “SEVERITYCODE” which is integer type and vehicle count is integer type and the rest of the features are object type as shown Fig 2.

```
2]: data.dtypes

ut[42]: SEVERITYCODE      int64
        COLLISIONTYPE    object
        VEHCOUNT          int64
        WEATHER          object
        ROADCOND         object
        LIGHTCOND        object
        dtype: object
```

Fig 2

First I have checked for all the null values in the data set and dropped all the rows that contain a minimum of one null value. As the original data set consists of 194673 rows shown in *Table 1*, after dropping the number of rows reduces to 189316 shown in *Fig 3*.

```
]:  new_data = data.dropna(axis = 0, how ='any')
```

```
]:  m=new_data.isnull()
    for col in m.columns.values.tolist():
        print(m[col].value_counts())
```

```
False    189316
Name: SEVERITYCODE, dtype: int64
False    189316
Name: COLLISIONTYPE, dtype: int64
False    189316
Name: VEHCOUNT, dtype: int64
False    189316
Name: WEATHER, dtype: int64
False    189316
Name: ROADCOND, dtype: int64
False    189316
Name: LIGHTCOND, dtype: int64
```

Fig 3

Each of the features is described with its unique no of counts and the totals counts as in fig 4.

```
new_data.describe(include=['object'])
```

[9]:

	COLLISIONTYPE	WEATHER	ROADCOND	LIGHTCOND
count	189316	189316	189316	189316
unique	10	11	9	9
top	Parked Car	Clear	Dry	Daylight
freq	47815	111002	124294	116064

Fig 4

Data Visualization

To visualize our data we select all features individually and plot bar graph and histogram to see the variability of counts.

For the feature “VEHCOUNT” histogram is plotted showing the number of counts with respect to number of vehicles as shown in *Fig 5*.

```
bins=np.arange(new_data['VEHCOUNT'].min(),8,1)
plt.hist(new_data['VEHCOUNT'],bins=bins)
plt.xlabel('no of vehichles')
plt.ylabel('no of accidents')
```

```
]: Text(0, 0.5, 'no of accidents')
```

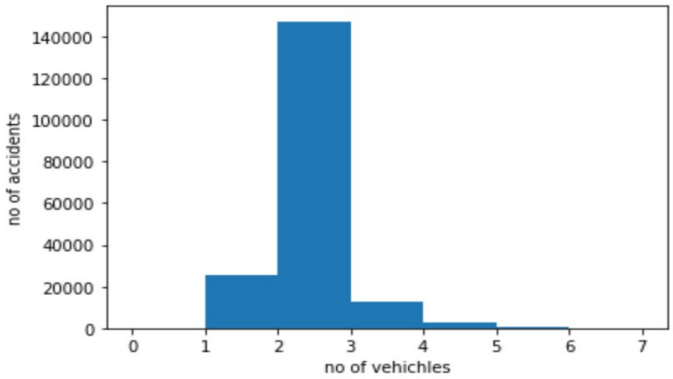

Fig 5

For the feature “ROADCOND” , a bar graph is plotted showing the number of counts with respect to number of vehicles as shown in *Fig 6*.

```
group_rc = new_data['ROADCOND'].unique()
data=new_data['ROADCOND'].value_counts()
plt.bar(x = group_rc,height = data,color = 'g')
plt.xlabel('road condition')
plt.ylabel('no of accidents')
plt.xticks(rotation = 90)
```

```
]: ([0, 1, 2, 3, 4, 5, 6, 7, 8], <a list of 9 Text xticklabel objects>)
```

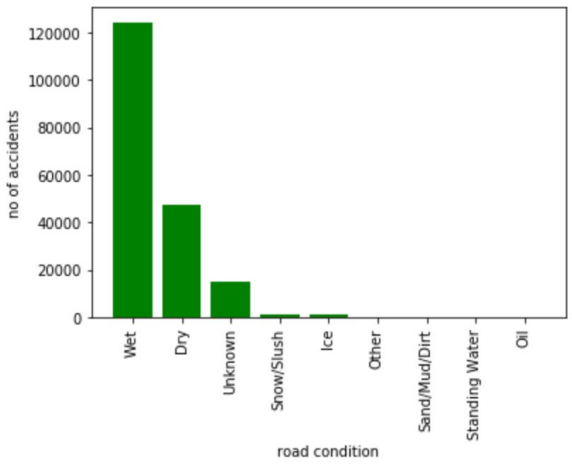

Fig 6

For the feature “WEATHER” , a bar graph is plotted showing the number of counts with respect to number of vehicles as shown in *Fig 7*.

```
group_w = new_data['WEATHER'].unique()
data_2=new_data['WEATHER'].value_counts()
plt.bar(x = group_w,height = data_2,color = 'b')
plt.xlabel('weather condition')
plt.ylabel('no of accidents')
plt.xticks(rotation = 90)
```

```
13]: ([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10], <a list of 11 Text xticklabel objects
```

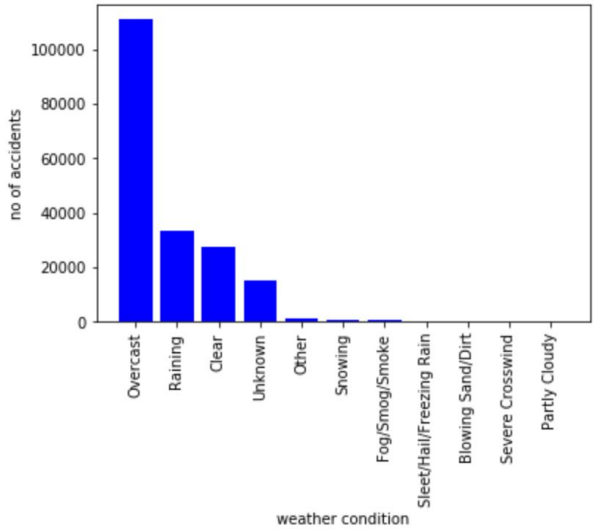

Fig 7

For the feature “COLLISIONTYPE” , a bar graph is plotted showing the number of counts with respect to number of vehicles as shown in *Fig 8*.

```
group_ct = new_data['COLLISIONTYPE'].unique()
group_ct
data_3 = new_data['COLLISIONTYPE'].value_counts()
plt.bar(x = group_ct,height = data_3, color = 'y')
plt.xlabel('weather condition')
plt.ylabel('no of accidents')
plt.xticks(rotation = 90)
```

```
14]: ([0, 1, 2, 3, 4, 5, 6, 7, 8, 9], <a list of 10 Text xticklabel objects>)
```

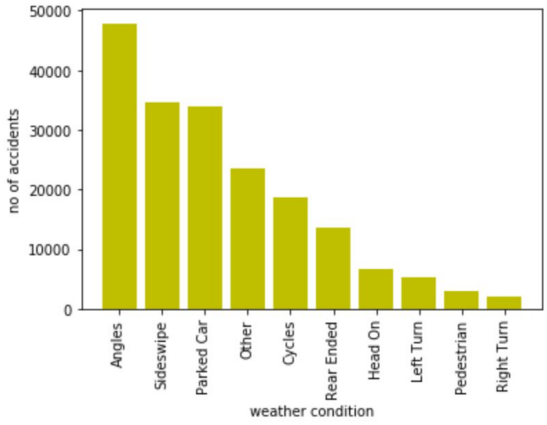

Fig 8

For the feature “LIGHTCOND” , a bar graph is plotted showing the number of counts with respect to number of vehicles as shown in *Fig 9*.

```
group_lt = new_data['LIGHTCOND'].unique()
group_lt
data_4 = new_data['LIGHTCOND'].value_counts()
plt.bar(x = group_lt,height = data_4, color = 'b')
plt.xlabel('light condition')
plt.ylabel('no of accidents')
plt.xticks(rotation = 90)
```

```
5]: ([0, 1, 2, 3, 4, 5, 6, 7, 8], <a list of 9 Text xticklabel objects>)
```

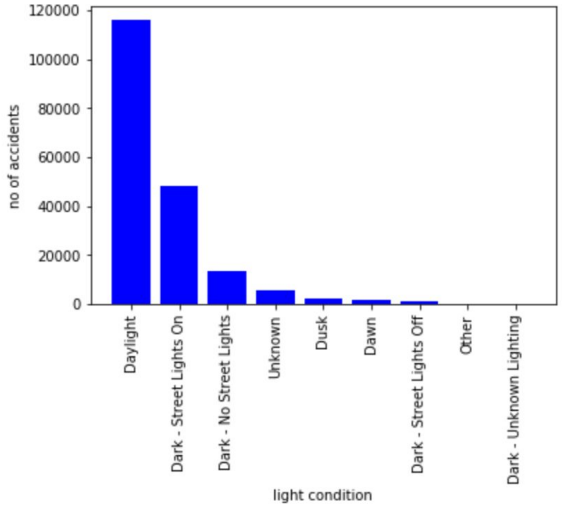

Fig 9

Now , I have converted all the object types to category type before changing data to numerical type using label encoding as in *Fig 10.*

```
vc=new_data['VEHCOUNT'].unique()
vc
```

```
array([ 2,  3,  1,  4,  0,  7,  5,  6,  8, 11,  9, 10, 12], dtype=int64)
```

```
wc=new_data['WEATHER'].unique()
wc
```

```
array(['Overcast', 'Raining', 'Clear', 'Unknown', 'Other', 'Snowing',
       'Fog/Smog/Smoke', 'Sleet/Hail/Freezing Rain', 'Blowing Sand/Dirt',
       'Severe Crosswind', 'Partly Cloudy'], dtype=object)
```

```
new_data['SEVERITYCODE'].unique()
```

```
array([2, 1], dtype=int64)
```

```
lc=new_data['LIGHTCOND'].unique()
lc
new_data["LIGHTCOND"] = new_data["LIGHTCOND"].astype('category')

rc=new_data['ROADCOND'].unique()
rc
new_data["ROADCOND"] = new_data["ROADCOND"].astype('category')

wc=new_data['WEATHER'].unique()
wc
new_data["WEATHER"] = new_data["WEATHER"].astype('category')

ct=new_data['COLLISIONTYPE'].unique()
ct
new_data["COLLISIONTYPE"] = new_data["COLLISIONTYPE"].astype('category')
```

Fig 10

Now converting all types to numerical types using label encoding as in *Fig 11.*”nWEATHER”, ”nROADCOND”, ”nLIGHTCOND” are now converted to integer type from the category type.

Label Encoding

```
data = new_data.drop(columns =['COLLISIONTYPE'])
data["nLIGHTCOND"] = data["LIGHTCOND"].cat.codes
data["nROADCOND"] = data["ROADCOND"].cat.codes
data["nWEATHER"] = data["WEATHER"].cat.codes
data["nWEATHER"] = data["WEATHER"].cat.codes
data.dtypes
```

```
SEVERITYCODE       int64
VEHCOUNT           int64
WEATHER         category
ROADCOND        category
LIGHTCOND       category
nLIGHTCOND          int8
nROADCOND           int8
nWEATHER            int8
dtype: object
```

Fig 11

The new data set consists of numerical values. Now the data is ready for building models and evaluation as in *Fig 12*.

	SEVERITYCODE	VEHCOUNT	nLIGHTCOND	nROADCOND	nWEATHER
0	2	2	5	8	4
1	1	2	2	8	6
2	1	3	5	0	4
3	1	3	5	0	1
4	2	2	5	8	6
...	...	...	...	...	...
194668	2	2	5	0	1
194669	1	2	5	8	6
194670	2	2	5	0	1
194671	2	1	6	0	1
194672	1	2	5	8	1

189316 rows × 5 columns

Fig 12

The heatmap shows the correlation of data among various variables and how they are correlated to other variables.This further helps in identifying features which are highly correlated to the target variable as in *Fig 13*.

```
sns.heatmap(data_new.corr(), cmap='YlGnBu_r')
plt.show()
```

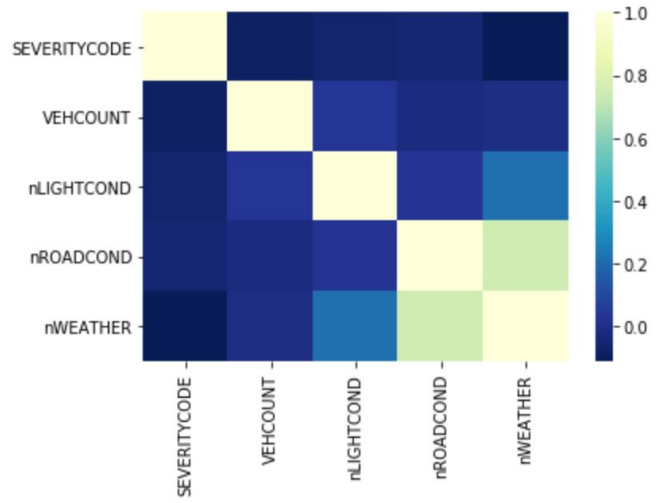

Fig 13