Abel Fernandez

Car Accident Severity Report¶

1. Introduction / Business Problem

When talking about most of health issues or affections, it is common to find out that most of the causes for health affections to the human health is by its own hand. In other words, most of the health threats are cause by the own creation of the human.

When talking about the progresses and evolution, car has been an incredible change, allowing mobility, agility, and creating communication from distant places. Also, due to the global expansion and peak evolution of human population, the accidents caused or occurred, are becoming on spotlight of most countries health policies, especially when most of them point to be linked to lack of attention when driving.

Some studies point out impacts other than the health impact, having an economic toll of almost $1 trillion USD. This estimation is subject to economic valuation and other impacts estimation, that may vary from year to year, but overall, they provide and estimate of the importance of this issue.

When reviewing the numbers it is important to gather data, and start questioning the different factors that may be linked to this impacts, economic and health matters.

This document aims to review the gathered data from different car accidents in order to understand the factors affecting or influencing the frequency and severity of car accidents, so it may be plausible to implement health policies for the mitigation of the problem.

Interest

Apart from the explained above, that could be qualified as common interest, it is a personal interest to research on identifying factors due to a motivation on identifying a common cause that is destroying or causing pain to not only individuals, but complete families. I was car crashed because on daylight, in a small neighborhood, and not especially bad road conditions, so doing this report, may show what other factors where affecting, or could have affected. If not it will always remain as an outlier from the most logic explanation.

2. DATA REVIEW

The data to be used is the one proposed with 252,861 car accidents with the categorization of severity, location and other factors that will guide us for the severity identification of each car accident. Hence dependent variable will be SEVERITYCODE.

This data was obtaind by IBM data science course (“Applied Data science Capstone”), hence no other double checks or validation of the origin or verification of the source were performed.

Data Treatment

The data obtained it can be identified that after some queries being run, there are quite a lot of data that is not complete, hence it should be treated careful to avoid 50% or NaN values among the data that is plan to identify some patterns or relationships.

Additionally, we have the problem of the data being categorical, hence we do have two options, use them with a categorical values or if we think that are not to be important for our model drop them.

Looking at the correlation among variables, to identify the simple correlation and color map of each one of them we can see how they are linked, and luckily not many high correlations from this scale are identified.

We first start by importing Data and analyzing it:

```
In [34]: #Importing Dataset
         filepath=r'C:\Users\abel.fernandez.alva1\Desktop\car_severity.csv'
         df=pd.read_csv(filepath,low_memory=False)
         df.head
```

```
Out[34]: <bound method NDFrame.head of         SEVERITYCODE           X          Y  OBJECTID  INCKEY  COLDETKEY  \
         0                2 -122.323148  47.703140         1    1307       1307
         1                1 -122.347294  47.647172         2   52200      52200
         2                1 -122.334540  47.607871         3   26700      26700
         3                1 -122.334803  47.604803         4    1144       1144
         4                2 -122.306426  47.545739         5   17700      17700
         5                1 -122.387598  47.690575         6  320840     322340
         6                1 -122.338485  47.618534         7   83300      83300
         7                2 -122.320780  47.614076         9  330897     332397
```

We are going to review the missing data and try to avoid using data with too many NaN values:

```
In [36]: #Unique values per column
         df.nunique()
```

```
Out[36]: SEVERITYCODE          2
         X                 23563
         Y                 23839
         OBJECTID         194673
         INCKEY           194673
         COLDETKEY        194673
         REPORTNO         194670
         STATUS                2
         ADDRTYPE              3
         INTKEY             7614
         LOCATION          24102
         EXCEPTRSNCODE         2
         EXCEPTRSNDESC         1
         SEVERITYCODE.1        2
         SEVERITYDESC          2
         COLLISIONTYPE        10
         PERSONCOUNT          47
         PEDCOUNT              7
         PEDCYLCOUNT           3
         VEHCOUNT             13
         INCDATE            5985
         INCDTTM          162058
         JUNCTIONTYPE          7
```

Then, by using the correlation map, we can get the whole picture of it:

```
In [39]: corr = df.corr()
         corr.style.background_gradient(cmap='coolwarm')
```

Out[39]:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	INTKEY	SEVERITYCODE.1	PERSONCOUNT	PEDCOUNT
SEVERITYCODE	1	0.0103089	0.0177369	0.020131	0.0220653	0.0220786	0.00655253	1	0.130949	0.246338
X	0.0103089	1	-0.160262	0.00995556	0.0103088	0.0102999	0.120754	0.0103089	0.0128873	0.0113045
Y	0.0177369	-0.160262	1	-0.023848	-0.0273964	-0.0274151	-0.114935	0.0177369	-0.0138497	0.0101776
OBJECTID	0.020131	0.00995556	-0.023848	1	0.946383	0.945837	0.0469293	0.020131	-0.0623333	0.0246043
INCKEY	0.0220653	0.0103088	-0.0273964	0.946383	1	0.999996	0.0485245	0.0220653	-0.0615004	0.024918
COLDETKEY	0.0220786	0.0102999	-0.0274151	0.945837	0.999996	1	0.0484989	0.0220786	-0.0614031	0.0249142
INTKEY	0.00655253	0.120754	-0.114935	0.0469293	0.0485245	0.0484989	1	0.00655253	0.00188597	-0.00478439
SEVERITYCODE.1	1	0.0103089	0.0177369	0.020131	0.0220653	0.0220786	0.00655253	1	0.130949	0.246338
PERSONCOUNT	0.130949	0.0128873	-0.0138497	-0.0623333	-0.0615004	-0.0614031	0.00188597	0.130949	1	-0.023464
PEDCOUNT	0.246338	0.0113045	0.0101776	0.0246043	0.024918	0.0249142	-0.00478439	0.246338	-0.023464	1
PEDCYLCOUNT	0.214218	-0.0017524	0.0263036	0.0344322	0.0313416	0.031296	0.000530993	0.214218	-0.0388092	-0.0169203

Then we differentiate the data sets for the dependant and independent variables and we categorize the data so we can identify those categorical:

```
In [43]: #Making a list of all categorical variables, so it can be easy to identify when working with them
         clmn = {"STATUS","ADDRTYPE","COLLISIONTYPE","JUNCTIONTYPE","UNDERINFL","WEATHER","ROADCOND","LIGHTCOND","HITPARKEDCAR"}

         #Converting them into dummy variables
         df = pd.get_dummies(data=df,columns=clmn,prefix=clmn)
```

```
In [46]: X = df.drop(["SEVERITYCODE"],axis=1)
         y = df["SEVERITYCODE"]
```

3. METHODOLOGY

With the Data ready, we can start sampling the data for its review using the following methods:

K-Nearest Neighbor (KNN)¶

From the data, it will help us predict the severity code depending on the data near it. This method is one of the first to implement due to the nature of its explanation, which may seem a basic procedure, but depending on the data sample it is effective.

Decision Tree¶

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

Logistic Regression¶

Due to the binarity of the nature of our variable SEVERITYCODE, the model will only predict one of those two classes. This makes our data binary, which is suitable to use with logistic regression.

For all of these methods, we had to go through a process of data cleansing and review of its distribution shown on the notebook. Once we have been able to review, we apply all the knowledge to implement on the coding performed. With it, it is time to review step by step which of the processes would be the most appropriate to the data set as it has been proven: binary dependant variable, with imbalance results and many categorical variables converted into a numerical set.

4. RESULTS

In order to review the results, a couple of coefficients are applied:

K-Nearest Neighbors

```
In [183]: # Jaccard Similarity Score
          jaccard_similarity_score(y_test, Kyhat)

          C:\Users\abel.fernandez.alva1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\metrics\classification.py:635: DeprecationWarning: jaccard_similarity_score has been deprecated and replaced with jaccard_score. It will be removed in version 0.23. This implementation has surprising behavior for binary and multiclass classification tasks.
            'and multiclass classification tasks.', DeprecationWarning)

Out[183]: 0.6985141368261681

In [184]: # F1-SCORE
          f1_score(y_test, Kyhat, average='macro')

Out[184]: 0.4112501166056863
```

Decision Tree

```
In [185]: # Jaccard Similarity Score
          jaccard_similarity_score(y_test, DTyhat)
```

```
C:\Users\abel.fernandez.alva1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\metrics\classification.py:635: Deprec
ationWarning: jaccard_similarity_score has been deprecated and replaced with jaccard_score. It will be removed in version 0.23.
This implementation has surprising behavior for binary and multiclass classification tasks.
  'and multiclass classification tasks.', DeprecationWarning)
```

```
Out[185]: 0.6985141368261681
```

```
In [186]: # F1-SCORE
          f1_score(y_test, DTyhat, average='macro')
```

```
Out[186]: 0.4112501166056863
```

Logistic Regression

```
In [187]: # Jaccard Similarity Score
          jaccard_similarity_score(y_test, LRyhat)
```

```
C:\Users\abel.fernandez.alva1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\metrics\classification.py:635: Deprec
ationWarning: jaccard_similarity_score has been deprecated and replaced with jaccard_score. It will be removed in version 0.23.
This implementation has surprising behavior for binary and multiclass classification tasks.
  'and multiclass classification tasks.', DeprecationWarning)
```

```
Out[187]: 0.6985141368261681
```

```
In [188]: # F1-SCORE
          f1_score(y_test, LRyhat, average='macro')
```

```
Out[188]: 0.4112501166056863
```

```
In [189]: # LOGLOSS
          yhat_prob = LR.predict_proba(X_test)
          log_loss(y_test, yhat_prob)
```

```
Out[189]: 0.5990960616849885
```

The results correspond to an iterative process of the train with 30% of the sample and 70% for the model, and corresponding to it the results that show. We can see that the scores are accepting to be shown.

5. DISCUSSION

As can be seen, data cleansing in order to work with it is critical. In a data sample where the majority was categorical and not simple to work with, it neede an important part to work with.

After that, and trying to balance the data for the SEVERITYCODE variable, which at the end is the variable that we are trying to predict (dependant), then we could work with the 3 models that have been the core of the study during these weeks: K-Nearest Neighbor, Decision Tree and Logistic Regression.

From all the models, due to the nature of the binary data of the SEVERITYCODE, logistic regression is the most appropriate to work with.

Evaluation metrics used to test the accuracy of our models were jaccard index, f-1 score and logloss for logistic regression. Choosing different k, max depth and hyperamater C values helped to improve our accuracy to be the best possible.

6. CONCLUSION

Based on historical data from weather conditions pointing to certain classes, we can conclude that particular weather conditions have a somewhat impact on whether or not travel could result in property damage (class 1) or injury (class 2).