2020

ML Model building (Classification)

This project is about building a classification model in order to predict the severity of the vehicle accidents/collisions in the state of Seattle, Washington, USA.

Published by Raman G
Capstone Project
9/28/2020

TABLE OF CONTENTS

Your security on the road depends on many factors. Seat belts, road conditions, visibility, and perhaps most alarmingly, other drivers. Unintentional injury is the leading cause of death for people under fifty in America, and over 37,000 people die yearly in motor vehicle incidents.

1. INTRODUCTION | Business Understanding

Seattle Department of Transportation (SDOT) develops, maintains, and operates a transportation system that enhances the quality of life, environment and economy of Seattle, as well as making sure people get around the city safely. Moving goods in, out of, and around Seattle is key to the regional economy and everyday life.

1.1 Problem Statement

Weather condition, visibility, roads condition, and other reasons are major factors in road accidents in the city, resulting in the following direct (or indirect) consequences:

a) fatalities
b) property damage
c) traffic delays
d) other indirect consequences (not covered under the current scope of this project.)

The scope of this project is limited to the dataset that contains car accidents that occurred in the city of Seattle, Washington, USA, between the years Jan 2004 and May, 2020

1.2 Audience/Stakeholders

The following are the major stakeholders that are directly/indirectly impacted:

- Drivers of vehicles that are involved in the accident
- Other passengers who are traveling in the vehicle that was involved in the accident
- Insurance companies (life and non-life)
- Seattle Police Department (SPD)
- Accident Traffic Records Department
- Other parties (owners of public/private property)
- People living in Seattle

These stakeholders will have to pay close attention to the problem described above. Therefore, in the larger interest of all the stakeholders, it is highly recommended to build and use a predictive collision model.

1.3 Objective

The objective of this project is to build a robust predictive collision model that will help to predict severity of accidents and to reduce the frequency of car collisions in the city of Seattle. The model should predict the severity of an accident given the conditions: weather, road, and visibility, location or a combination of these conditions.

Based on the prediction, the stakeholders and/or designated authorities can take appropriate measures/actions to mitigate the impending dangers or risks.

2. DATA

2.1 Data understanding

- **Data Source:** Historical data about motor vehicles collision (accidents) has been collected from the Seattle Police Department (SPD) and Accident Traffic Records Department, from the year 2004 till May 2020. The dataset is available on IBM cloud as a CSV file. To be able to build a good machine learning model, you need a rich dataset, and our dataset is rich, as it contains many observations *(194,674 rows)* and various attributes *(37 columns).*

  However, for reasons of computational speed and convenience, a sample size has been selected from the original dataset. In other words, from the original dataset, 1,000 rows have been selected for a training dataset and about 500 rows for testing *(out of sample)* dataset.

  This is a supervised machine learning model, therefore we need labeled data to train and validate the model. In this section, we will use exploratory data analysis and visualization techniques to study the key attributes that will be used to build the machine learning model that can predict the severity of the vehicle collision.

- **Label/Target** The target variable for the dataset is **severity desription** *(dependent variable **y**)*, which describes the **accident**. Incidents are recorded with a code called SEVERITYCODE which classifies the accients into two types:

  1. Property Damage Only Collision
  2. Injury Collision

  SEVERITYDESC variable will be used as our *dependent variable Y*. This is an 'object' datatype , and the remaining columns *(independent variables **X**)* are also of the datatype: object.

  The data shows that there is an imbalance of class labels – while the type 1 label has about 709 observations, type 2 label has less than half of type 1: only 291. Therefore we should balance the dataset otherwise we will create a biased machine learning model.

2.2 Data Preparation/Data Wrangling

Data wrangling is the process of converting data from initial format to a format that may be better for analysis. Therefore, to get the data set ready for analysis, the following tasks will be carried out as a part of data preparation. We will cover the data cleansing and formatting the features, and these are the steps for working with the missing data.

1. **Identify missing data** : Replacing NaN / Blanks
2. **Deal with missing data** using **One-Hot-Coding** : Changing column data types to numerical datatypes
3. **Correct Data format**

2.3 Feature selection

In total, there are 38 attributes (columns or features or independent variables). Some attributes have missing data, and there are numerical and categorical types of data in the original dataset. We'll retain only those that are useful to our analysis and model building, and drop the rest. Therefore, the **relevant attributes** are chosen from the dataset, and they're listed below. We need to do feature-engineering on the data in order to improve the **features** predictability of our model.

Attribute / Column (Feature)	Data type, length	Description of attribute/column
X	Float64	Latitude
Y	Float64	Longitude
WEATHER	Text / Object	Description of weather conditions at the time of collision *(overcast; raining; clear)*
ROADCOND	Text / Object	Condition of the road during collision *(wet; dry; unknown)*
LIGHT COND	Text / Object	Description of the general location of the collision *(daylight; dusk; dawn) (dark-street lights on; dark-street lights off; dark- no street lights)*

2.4 Missing Data: The following table shows the chosen variables that are found to have missing values or values like 'unknown'. These values have been replaced with appropriate values, and reasons have been provided for understanding. Missing values (blanks) and 'unknown' have been found in the following categorical variables:

Variable	Missing Values	Replaced with	Comments / Reasons
X (Latitude)	BLANKS	Mean: -122.33	Choose arithmetic mean
Y (Longitude)	BLANKS	Mean: 47.61	Choose arithmetic mean
ADDRTYPE	BLANKS	'Intersection'	Mode: **Intersection**
WEATHER	BLANKS	'Clear'	Mode: **Clear**
ROADCOND	BLANKS	'Dry'	Mode: **Dry**
LIGHTCOND	BLANKS	'Daylight'	Mode: **Daylight**

Now that we have cleaned the data, we need to do some feature engineering. This involves transforming the values in the dataset into numeric values, utilizing one-hot-encoding method that machine learning algorithms can use.

2.5 Formatting

The Incident Date has been formatted to read as YYY-MM-DD. From this date, we can extract month, date, and the day of the week which then can be used for our analysis purposes.

3. METHODOLOGY

We have used classification in this project/model building to predict a categorical variable, which is the **severity** of the collision / accident. As we know that in machine learning, classification is a supervised learning approach which can be thought of as a means of categorizing or classifying some unknown items into a discrete set of classes. In other worlds, classification attempts to learn the relationship between a set of feature variables and a target variable of interest. The **target attribute in classification is a categorical variable** with discrete values. In our case, it is 'Property Damage Only Collision' and 'Injury Collsion.'

Python Notebook was used for data pre-processing, and the notebook has been published on the Github. Classifiers are used to build the model and relevant modules/packages have been used in the process of model building.

- **Pandas** library has been used for importing, manipulating and analyzing data.
- **NumPy, SciPy, Matplotib** – these three packages which are built on top of Python, are good assets to work with real-world problems.
- **SciKit Learn** has been used for tasks that included pre-processing, feature selection/extraction, train-test-splitting, defining the algorithms, fitting our model, tuning parameters, prediction, evaluation. SciKit Learn can split arrays or matrices into random train and test subsets in one line of code, and then we can set up our algorithm. The most important point to remember is that the entire process of a machine learning task can be done simply in a few lines of code using SciKit Learn.

3.1 Exploratory data analysis (EDA)

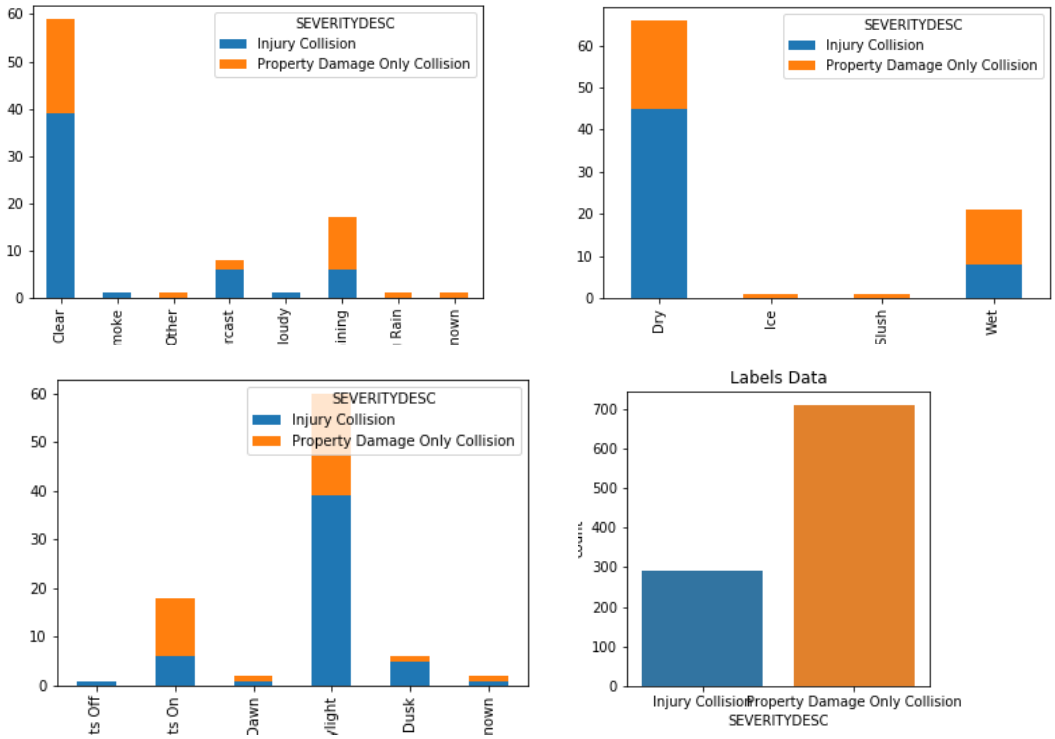
When we begin to analyze data, it's important to first explore your data before you spend time building complicated models. In this section, we will be asking the fundamental question:

What are the characteristics that have the most important impact on the 'severity' of vehicle collision?

We summarized the main characteristics of the data to gain better understanding of the dataset and uncover relationships between variables and then extract important variables. We have carried out a couple of useful exploratory data analysis techniques in order to answer this question.

3.2 Frequency Graphs

Weather	Road	Light
Clear	Dry	Daylight
Overcast	Ice	Dark - Street lights Off
Raining	Wet	Dark – Street lights On
Other	Unknown	Dark – No Street lights
Fog/mog/smoke	Snow/slush	Dusk
Snowing		Other
Unknown		Unknown
		Dawn



Descriptive statistics, which describe basic features of a dataset have been used to obtain a short summary about the sample and measures of the data; basic of grouping data has been done to see how this can help to transform our training dataset.

3.4 Machine Learning

We start building our machine learning model that **predicts** the accident **"severity."** Various algorithms and methods are selected here and applied to build the model - including supervised learning techniques. We chose to use classifiers like KNN, Decision tree, Logistic Regression. At this phase, stepping back to data preparation stage is often required.
We trained our model and then we calculated its accuracy using the test set. Basically, we compare the actual values in the test set with the values predicted by the model, to calculate the accuracy of the model. Evaluation metrics provide a key role in the development of a model, as they provide insight to areas that might require improvement.

There are three key model evaluation metrics we've used in our model building: **Jaccard index, F1-score, and Log Loss (Logistic Regression).** In logistic regression, the output can be the probability of a severity happening.

Algorithms / Classifiers used:

- K-Nearest Neighbor
- Decision Tree
- Support Vector Machine
- Logistic Regression

4. Results / Discussion

The model was evaluated using a test data set (out-of-sample) and accuracy of the model was reported, using different evaluation metrics. As we know, Scikit-learn has a metrics module that provides metrics that can be used for other purposes like when there is class imbalance etc. In scikit-learn, the default choice for classification is accuracy, which gives us the number of labels correctly classified. These are our model comparison and the result of evaluation. Our metrics are: Jaccard index, Precision, Recall, F1-score, Confusion Matrix and Log Loss accuracy.

KNN: The best accuracy with this classifier was with 0.675 i.e., **67% accuracy with k equals 11.** Model accuracy was plotted to see the visual.

Decision Tree: CollisionTree (an object) was constructed and printed to visually compare the prediction (label) to the actual values. It is attached under the section 'annexures.'

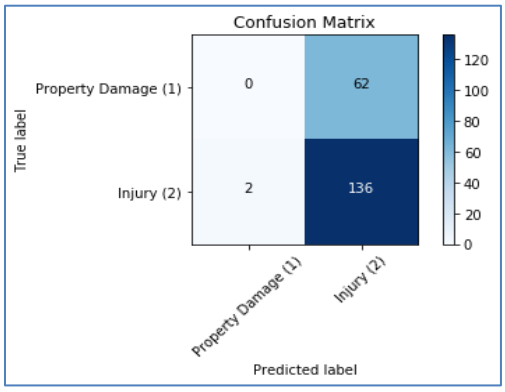
Support Vector Machine: Used the default RBF (Radial Basis Function) for this model for this classifier

Now, let's see the **F1-Score**. This is the harmonic average of the precision and recall, and it reaches its best at value 1 *(which represents a perfect precision and recall)*, and its worst at 0. In other words, the classifier with F1-score close to 1 is the ideal one. A **low F1 score** is an indication of both poor precision and poor recall.

Let's quickly summarize the **differences between** the **F1-score** and the **accuracy**: **Accuracy** is used when the True Positives and True negatives are more important while **F1-score** is used when the False Negatives and False Positives are crucial. **Accuracy** can be a useful measure if we have the same amount of samples per class but if we have an imbalanced set of samples **accuracy** isn't useful at all. Even more so, a test can have a high **accuracy** but actually perform worse than a test with a lower **accuracy**.

Confusion matrix was constructed to evaluate the accuracy of a classification. A good thing about the **confusion matrix** is that it shows the model's ability to correctly predict or separate the classes. As we see in our model, from a total dataset of 1000 rows/observations, 800 rows were split for training purpose, and the remaining 200 rows were split for test process (80% training and 20% for testing.) For these 200 rows, the classifier has predicted values correctly for the 2nd row. In other words, the classifier predicted accurately, the 'Injury Collision' class with 136 collisions, and missed out 2 that also belonged to 'Injury' class, and wrongly predicted it as 'Property Damage' class. Whereas for the 1st row, the classifier did not do a good job. Why? Because the classifier has predicted 0 as 'Property Damage' class, which is accurate, but the

model missed out 62 collisions that also belonged to this class, and predicted them as 'Injury Class'



Log Loss measures the performance of a classifier where the predicted output is a probability value between 0 and 1. We can calculate the log loss for each row using the log loss equation, which measures how far each prediction is, from the actual label.

5. CONCLUSION

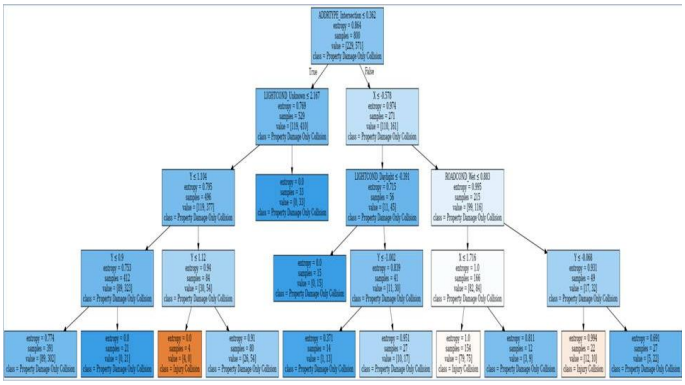
The purpose of this project was to predict the collision severity in the city of Seattle, Washington, USA, with an intent to aid various stakeholders in their respective areas. Using different classifiers, we have obtained the predictions and associated metrics to support the evaluation process. From the evaluation metrics report, we can see that the classifiers SVM and Logistic Regression are predicting 70% accuracy. Therefore the stakeholders can utilize these classifiers/models in their respective areas of operations, to be able to gain some tangible benefits. Needless to say, this model can be enhanced further, incorporating other category variables into the model to be able to make further analysis and to draw further relevant conclusions.

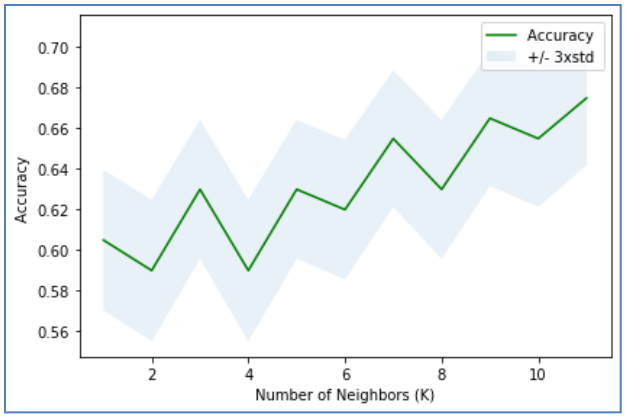
<u>References:</u>

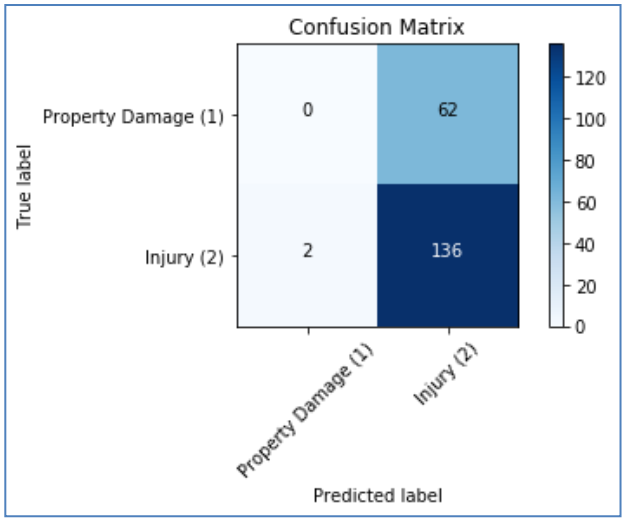
1. https://www.seattle.gov/transportation
2. https://pbpython.com
3. https://safesmartliving.com
4. https://www.cdc.gov
5. https://en.wikipedia.org/
6. https://www.safewise.com/

<u>Annexure:</u>

1. Decision Tree
2. KNN plot
3. Confusion Matrix
4. Evaluation Metrics Chart







EVALUATION METRICS CHART

Algorithm	Jaccard	F1_score	LogLoss
KNN	0.636542	0.589594	NA
Decision Tree	0.628684	0.623576	NA
SVM	0.703340	0.582466	NA
Logistic Regression	0.701375	0.588492	0.601364

End of report