Predicting The Severity of Car Accidents in Seattle

Table of contents

Introduction: Business Problem

Car accidents around the world occur very frequently, many of them are due to road conditions and the environment. The idea is to have a tool that, based on the conditions of the cart, the weather and the drivers; can give to know the possibility of having an accident as well as its severity, seeking to change your travel plan or if not possible, be more attentive and careful. For this objective, we are going to work with the data of the city of Seattle, which will allow us to put together a model that will give us the best results. This data does not disclose information on past accidents based on road conditions, weather, light, driver attention, drug or alcohol use, and speed. At the end, a report will be presented in which the model that meets the proposed objective will be presented, in addition to considering results based on the possibility of inappropriate behaviors (lack of attention, going too fast or consuming drugs or alcohol) on the part of the drivers, this last point seeks to alert drivers to their behavior as well as that of others. This tool is not only aimed at private drivers or those who work on road trips, but also at police and state personnel in charge of road infrastructure, as it can help prevent accidents either by hiring personnel for traffic controls or making changes to the streets such as adding speed bumps or traffic signals.

Data

The dataset to use is about collisions recorded in the city of Seattle. Its attributes are as follows:

Attribute	Data type, length	Description
OBJECTID	ObjectID	ESRI unique identifier
SHAPE	Geometry	ESRI geometry field
INCKEY	Long	A unique key for the incident
COLDETKEY	Long	Secondary key for the incident
ADDRTYPE	Text, 12	Collision address type: • Alley • Block • Intersection
INTKEY	Double	Key that corresponds to the intersection associated with a collision
LOCATION	Text, 255	Description of the general location of the collision
EXCEPTRSNCODE	Text, 10	
EXCEPTRSNDESC	Text, 300	
SEVERITYCODE	Text, 100	A code that corresponds to the severity of the collision: • 3—fatality • 2b—serious injury • 2—injury • 1—prop damage • 0—unknown
SEVERITYDESC	Text	A detailed description of the severity of the collision
COLLISIONTYPE	Text, 300	Collision type
PERSONCOUNT	Double	The total number of people involved in the collision
PEDCOUNT	Double	The number of pedestrians involved in the collision. This is entered by the state.
PEDCYLCOUNT	Double	The number of bicycles involved in the collision. This is entered by the state.
VEHCOUNT	Double	The number of vehicles involved in the collision. This is entered by the state.
INJURIES	Double	The number of total injuries in the collision. This is entered by the state.
SERIOUSINJURIES	Double	The number of serious injuries in the collision. This is entered by the state.
FATALITIES	Double	The number of fatalities in the collision. This is entered by the state.
INCDATE	Date	The date of the incident.
INCDTTM	Text, 30	The date and time of the incident.
JUNCTIONTYPE	Text, 300	Category of junction at which collision took place
SDOT_COLCODE	Text, 10	A code given to the collision by SDOT.
SDOT_COLDESC	Text, 300	A description of the collision corresponding to the collision code.
INATTENTIONIND	Text, 1	Whether or not collision was due to inattention. (Y/N)
UNDERINFL	Text, 10	Whether or not a driver involved was under the influence of drugs or alcohol.
WEATHER	Text, 300	A description of the weather conditions during the time of the collision.
ROADCOND	Text, 300	The condition of the road during the collision.
LIGHTCOND	Text, 300	The light conditions during the collision.
PEDROWNOTGRNT	Text, 1	Whether or not the pedestrian right of way was not granted. (Y/N)
SDOTCOLNUM	Text, 10	A number given to the collision by SDOT.
SPEEDING	Text, 1	Whether or not speeding was a factor in the collision. (Y/N)
ST_COLCODE	Text, 10	A code provided by the state that describes the collision. For more information about these codes, please see the State Collision Code Dictionary.
ST_COLDESC	Text, 300	A description that corresponds to the state’s coding designation.
SEGLANEKEY	Long	A key for the lane segment in which the collision occurred.
CROSSWALKKEY	Long	A key for the crosswalk at which the collision occurred.
HITPARKEDCAR	Text, 1	Whether or not the collision involved hitting a parked car. (Y/N)

For the desired objective we use the following attributes:

- ADDRTYPE
- SEVERITYCODE
- INATTENTIONIND
- UNDERINFL
- WEATHER
- ROADCOND
- LIGHTCOND
- SPEEDING

Let's download the dataset and load data from CSV file:

Out[3]:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	REPORTNO	STATUS	ADDRTYPE	INTKEY	...	ROADCOND	LIGHTCOND	PEDROWNOTGRNT	SDOTCOLNUM	SPEEDING	ST_COLCODE	ST_
0	2	-122.323148	47.703140	1	1307	1307	3502005	Matched	Intersection	37475.0	...	Wet	Daylight	NaN	NaN	NaN	10	
1	1	-122.347294	47.647172	2	52200	52200	2607959	Matched	Block	NaN	...	Wet	Dark - Street Lights On	NaN	6354039.0	NaN	11	dire goi
2	1	-122.334540	47.607871	3	26700	26700	1482393	Matched	Block	NaN	...	Dry	Daylight	NaN	4323031.0	NaN	32	C
3	1	-122.334803	47.604803	4	1144	1144	3503937	Matched	Block	NaN	...	Dry	Daylight	NaN	NaN	NaN	23	c
4	2	-122.306426	47.545739	5	17700	17700	1807429	Matched	Intersection	34387.0	...	Wet	Daylight	NaN	4028032.0	NaN	10	

5 rows × 38 columns

Let's reduce the dateset to the identified attributes:

Out[5]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING	SEVERITYCODE
0	Intersection	NaN	N	Overcast	Wet	Daylight	NaN	2
1	Block	NaN	0	Raining	Wet	Dark - Street Lights On	NaN	1
2	Block	NaN	0	Overcast	Dry	Daylight	NaN	1
3	Block	NaN	N	Clear	Dry	Daylight	NaN	1
4	Intersection	NaN	0	Raining	Wet	Daylight	NaN	2

In [6]:
```
Feature.shape
```

Out[6]:
```
(194673, 8)
```

Let's see how many nulls there are:

Out[7]:
```
ADDRTYPE           1926
INATTENTIONIND    164868
UNDERINFL           4884
WEATHER             5081
ROADCOND            5012
LIGHTCOND           5170
SPEEDING          185340
SEVERITYCODE           0
dtype: int64
```

Let's delete the rows with null values:

Out[8]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING	SEVERITYCODE
1320	Block	Y	N	Clear	Dry	Daylight	Y	2
1572	Intersection	Y	N	Clear	Dry	Daylight	Y	1
2918	Block	Y	N	Clear	Dry	Daylight	Y	2
3045	Block	Y	N	Snowing	Snow/Slush	Daylight	Y	2
3499	Block	Y	0	Overcast	Dry	Dark - Street Lights On	Y	1

Let's see how many collision address type in our data set:

Out[9]:
```
Block           526
Intersection    161
Alley             1
Name: ADDRTYPE, dtype: int64
```

Let's see how many inattention accidents are in our data set:

Out[10]:
```
Y    688
Name: INATTENTIONIND, dtype: int64
```

Let's see how many substance-related accidents are in our data set:

Out[11]:
```
N    449
0    181
Y     41
1     17
Name: UNDERINFL, dtype: int64
```

Let's replace the 0's with 'N' and 1's with 'Y' for the substance-related accident values:

Out[12]:
```
N    630
Y     58
Name: UNDERINFL, dtype: int64
```

Let's see how many of each weather is in our data set:

Out[13]:
```
Clear                       343
Raining                     222
Overcast                    102
Snowing                       7
Unknown                       6
Fog/Smog/Smoke                5
Other                         2
Sleet/Hail/Freezing Rain      1
Name: WEATHER, dtype: int64
```

Let's see how many road conditions are in our data set:

Out[14]:
```
Dry               368
Wet               298
Ice                 7
Snow/Slush          7
Unknown             5
Standing Water      3
Name: ROADCOND, dtype: int64
```

Let's see how many light conditions there are in our data set.

Out[15]:
```
Daylight                    390
Dark - Street Lights On     246
Dawn                        18
Dusk                        16
Dark - No Street Lights      8
Unknown                      4
Other                        3
Dark - Street Lights Off     3
Name: LIGHTCOND, dtype: int64
```

Let's see how many recorded speed conditions in our data set

Out[16]:
```
Y    688
Name: SPEEDING, dtype: int64
```

Let's see how many of each severity code is in our data set:

Out[17]:
```
1    406
2    282
Name: SEVERITYCODE, dtype: int64
```

Let's validate if there are duplicates:

Out[18]:
```
True     565
False    123
dtype: int64
```

Let's delete duplicate rows:

Out[19]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING	SEVERITYCODE
1320	Block	Y	N	Clear	Dry	Daylight	Y	2
1572	Intersection	Y	N	Clear	Dry	Daylight	Y	1
3045	Block	Y	N	Snowing	Snow/Slush	Daylight	Y	2
3499	Block	Y	N	Overcast	Dry	Dark - Street Lights On	Y	1
3517	Block	Y	N	Raining	Wet	Dark - Street Lights On	Y	2

Out[20]:
```
(123, 8)
```

Let's validate if there are severity codes with the same values:

Out[21]:
```
False    87
True     36
dtype: int64
```

In the original dataset you can see:

- That many of the attributes considered have not been filled in or have been filled with values other than those expected.
- That the INATTENTIONIND and SPEEDING attributes have only been filled in a few cases.
- That the UNDERINFL attribute has been filled in with different values for the same meaning, for example with '1' or 'Y' to indicate if there were substance use.
- That only 2 of the 5 codes of the description of the dataset attributes have been considered.
- Most of the rows in the data set is duplicated.
- As there are severity codes with equal values, we are going to use only those with the highest severity, since the objective is better suited to our objective.

Since the objective is to give the probability of a car accident occurring, the null values have been completed, the values that have the same meaning have been corrected to be able to consider all the alternatives presented by the data set. For the normalization, the value 'N' will be considered for the INATTENTIONIND and SPEEDING attributes, in addition to considering the 3 missing codes of the SEVERITYCODE attribute, this as part of the reuse process of the model to be generated.

Methodology

To generate the model that meets the objective, we will follow the following steps:

1. We will replace the textual values with numeric values, giving the value of '0' to the value of 'Unknown' and removing duplicate rows.
2. We are going to normalize the data and separate the data set into two groups, one for training and one for testing.
3. We are going to test the following classifiers:
   - K Nearest Neighbor (KNN)
   - Decision Tree
   - Support Vector Machine
   - Logistic Regression

Out[22]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING	SEVERITYCODE
1320	1	2	1	1	1	1	2	2
1572	2	2	1	1	1	1	2	1
3045	1	2	1	4	4	1	2	2
3499	1	2	1	3	1	2	2	1
3517	1	2	1	2	2	2	2	2

To eliminate duplicates, we are going to sort the rows in descending order by the severity code and eliminate the last duplicates considering only the columns:

- ADDRTYPE
- INATTENTIONIND
- UNDERINFL
- WEATHER
- ROADCOND
- LIGHTCOND
- SPEEDING'

Out[23]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING	SEVERITYCODE
1320	1	2	1	1	1	1	2	2
153252	1	2	1	2	2	4	2	2
39747	1	2	1	3	2	3	2	2
41260	2	2	1	2	2	2	2	2
41547	1	2	1	0	1	1	2	2

Out[24]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING	SEVERITYCODE
1320	1	2	1	1	1	1	2	2
153252	1	2	1	2	2	4	2	2
39747	1	2	1	3	2	3	2	2
41260	2	2	1	2	2	2	2	2
41547	1	2	1	0	1	1	2	2

Out[25]: (87, 8)

Lets defind feature sets, X:

Out[26]:

	ADDRTYPE	INATTENTIONIND	UNDERINFL	WEATHER	ROADCOND	LIGHTCOND	SPEEDING
1320	1	2	1	1	1	1	2
153252	1	2	1	2	2	4	2
39747	1	2	1	3	2	3	2
41260	2	2	1	2	2	2	2
41547	1	2	1	0	1	1	2

Our labels:

Out[27]:
```
array([2, 2, 2, 2, 2])
```

Let's normalize the data:

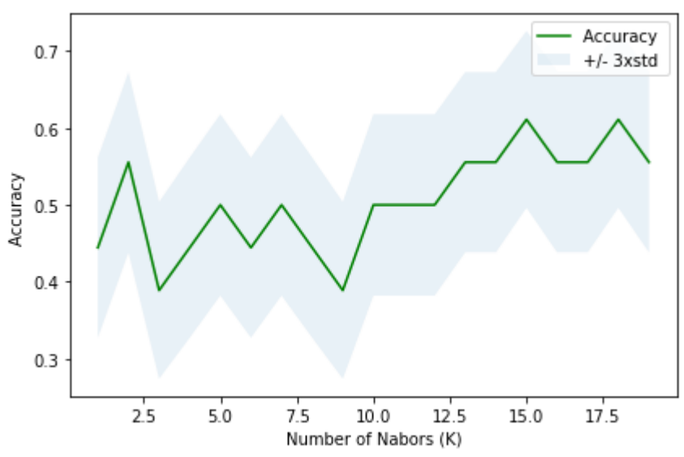
Out[28]:
```
array([[-0.76486616,  0.        , -0.51075392, -0.88087997, -0.76277007,
        -0.86371634,  0.        ],
       [-0.76486616,  0.        , -0.51075392, -0.19662499,  0.15891043,
         0.9690476 ,  0.        ],
       [-0.76486616,  0.        , -0.51075392,  0.48762998,  0.15891043,
         0.35812629,  0.        ],
       [ 1.19229137,  0.        , -0.51075392, -0.19662499,  0.15891043,
        -0.25279503,  0.        ],
       [-0.76486616,  0.        , -0.51075392, -1.56513495, -0.76277007,
        -0.86371634,  0.        ]])
```

Let's separate the data to train and test:

```
Train set: (69, 7) (69,)
Test set: (18, 7) (18,)
```

K Nearest Neighbor(KNN)

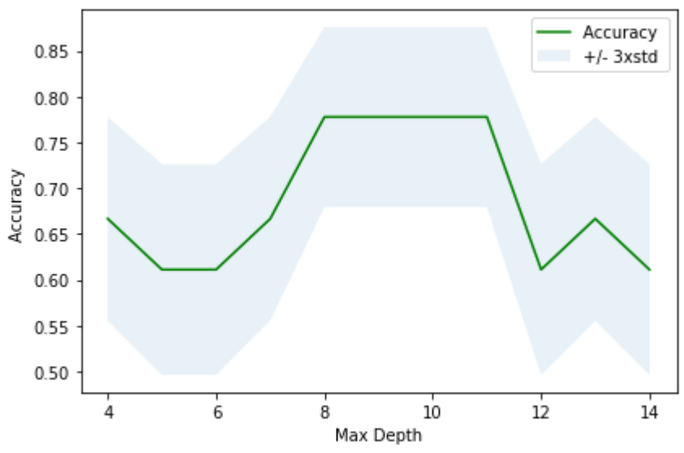
Let's start the KNN classifier by testing a number of neighbors ranging from 90 to 95:



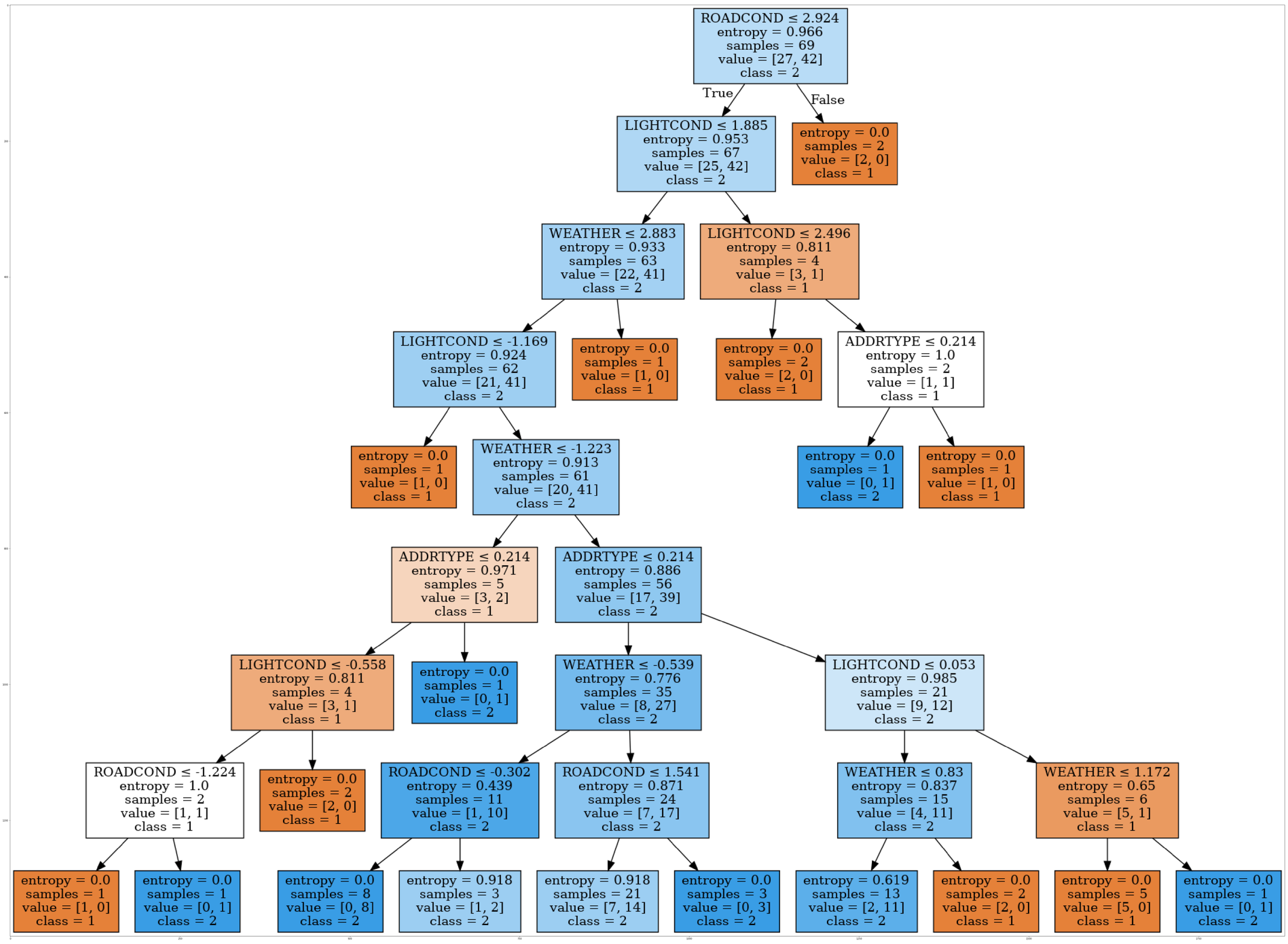

```
The best accuracy was with 0.6111111111111112 with k= 15
```

Decision Tree

Let's evaluate the decision tree classifier by testing with a depth of 4 to 10:




```
The best accuracy was with 0.7777777777777778 with max_depth= 8
```



Support Vector Machine

Let's evaluate the Support Vector Machine classifier usign the kernel RBF:

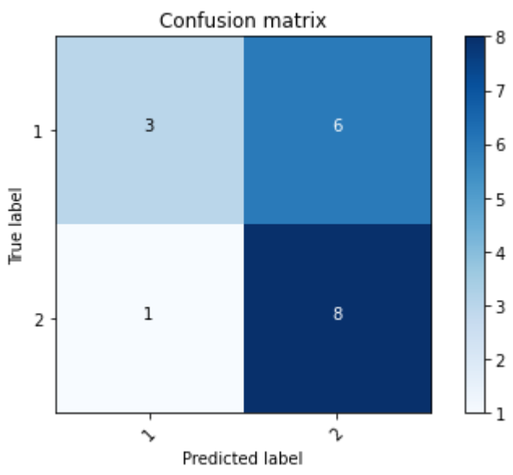
```
Avg F1-score: 0.5786
Jaccard score: 0.6111

              precision    recall  f1-score   support

           1       0.75      0.33      0.46         9
           2       0.57      0.89      0.70         9

   micro avg       0.61      0.61      0.61        18
   macro avg       0.66      0.61      0.58        18
weighted avg       0.66      0.61      0.58        18

Confusion matrix, without normalization
[[3 6]
 [1 8]]
```



Logistic Regression

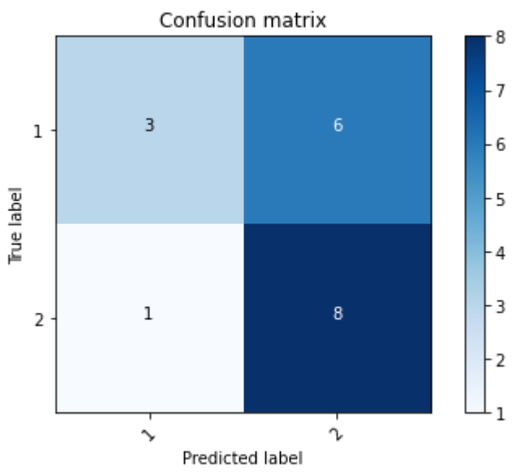
Let's evaluate the Logistic Regression classifier usign the solver saga:

```
                  precision    recall  f1-score   support

               1       0.75      0.33      0.46         9
               2       0.57      0.89      0.70         9

       micro avg       0.61      0.61      0.61        18
       macro avg       0.66      0.61      0.58        18
    weighted avg       0.66      0.61      0.58        18
```

```
Confusion matrix, without normalization
[[3 6]
 [1 8]]
```



Analysis

We are going to compare the results obtained from the 4 classifiers:

Out[40]:

	Algorithm	Jaccard	F1-score	LogLoss
0	KNN	0.555556	0.446154	
1	Decision Tree	0.722222	0.714286	
2	SVM	0.611111	0.578595	
3	LogisticRegression	0.500000	0.333333	0.706093

Results and Discussion

The dataset used contained many null values, as well as duplicates, having to be drastically reduced. Based on the results obtained, it can be observed that the Decision Tree classification algorithm is the one with the best results, allowing us to create a model that has acceptable precision despite the few data used. The Support Vector Machine classification algorithm is second despite showing the same values in the confusion matrix as the Logistic Regression classification algorithm- The K Nearest Neighbor ranking algorithm came in third place, but with very low results.

Conclusion

The data of the dataset used has not been recorded in an orderly and controlled manner, causing that there is not enough information to consider all circumstances. To solve this, it would be recommended to improve the registry of these and use the data from a longer period of time. According to the results obtained, we can see that the Decision Tree classification algorithm is the one with the best results, and given the nature of the data used, it can be seen that it best fits the objective to be achieved, since I will present it The more elements that affect the handling of a car, the greater the probability of having an accident. In production, the information provided to the model for the prediction should consider the limit values ('Y' and 'N') of the INATTENTIONIND, UNDERINFL and SPEEDING attributes, since these depend a lot on the behavior of the drivers at the time. . In addition, considering those of this may provide the user with a broader perspective of a possible accident and thus give him the opportunity to take action.