IBM Data Science Capstone

Predicting Car Accident Severity

Peio Alcorta

1. Introduction

- In this project, we will be working on a case study which is to predict the severity of an accident.

- A car accident occurs every 4 minutes and on average and a person dies due to a car accident almost every 20 hours in the state of Washington, while lethal crashes went from 508 in 2016 to 525 in 2017, resulting in the death of 555 people. Our purpose is to develop an algorithm which given the weather and the road conditions, among some other features, predicts the probability of getting into a car accident and how severe it would be so that it encourages the driver to drive more carefully or even change the travel if possible.

2. Data

2.1 Data Understanding

- The dataset used for this project is based on car accidents which have taken place within the city of Seattle, Washington from the year 2004 to 2020. This data is regarding car accidents the severity of each car accidents along with the time and conditions under which each accident occurred. The data set used for this project can be downloaded from:

- https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv

- The Metadata of the data can be found here:

- https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Metadata.pdf

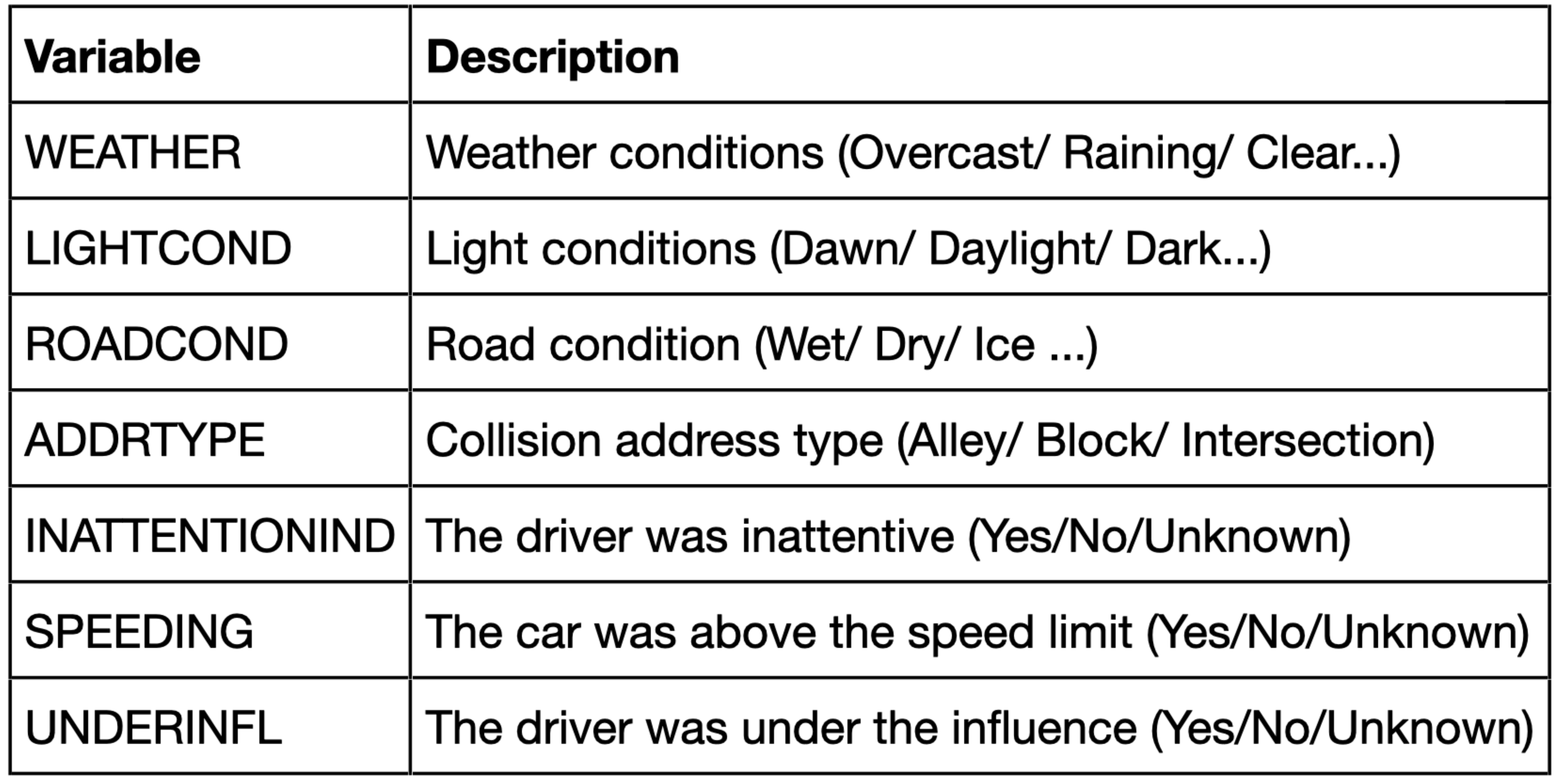
- The original dataset contains 194,673 rows, one for each traffic accident, and 38 columns, which encode several aspects of each accident, such as, location, severity, weather condition…

- The aim of this project is to develop a Machine Learning algorithm that predicts the severity of an accident. For that purpose, we will use the variable SEVERITYCODE which encodes the severity of the accident by giving value 1 if the accident entailed property damage only, and 2 if it entailed physical or health damage too.

2. Data

2.2 Data Preparation

- Apart from the severity code, a total of 7 variables (columns) are selected from the dataset. We restrict our attention to just those variables which we believe will have predictive power on the severity of the accident. These variables comprise both external conditions (weather, road, lighting, and address type), as well as human conditions (speed, drugs, and inattention):

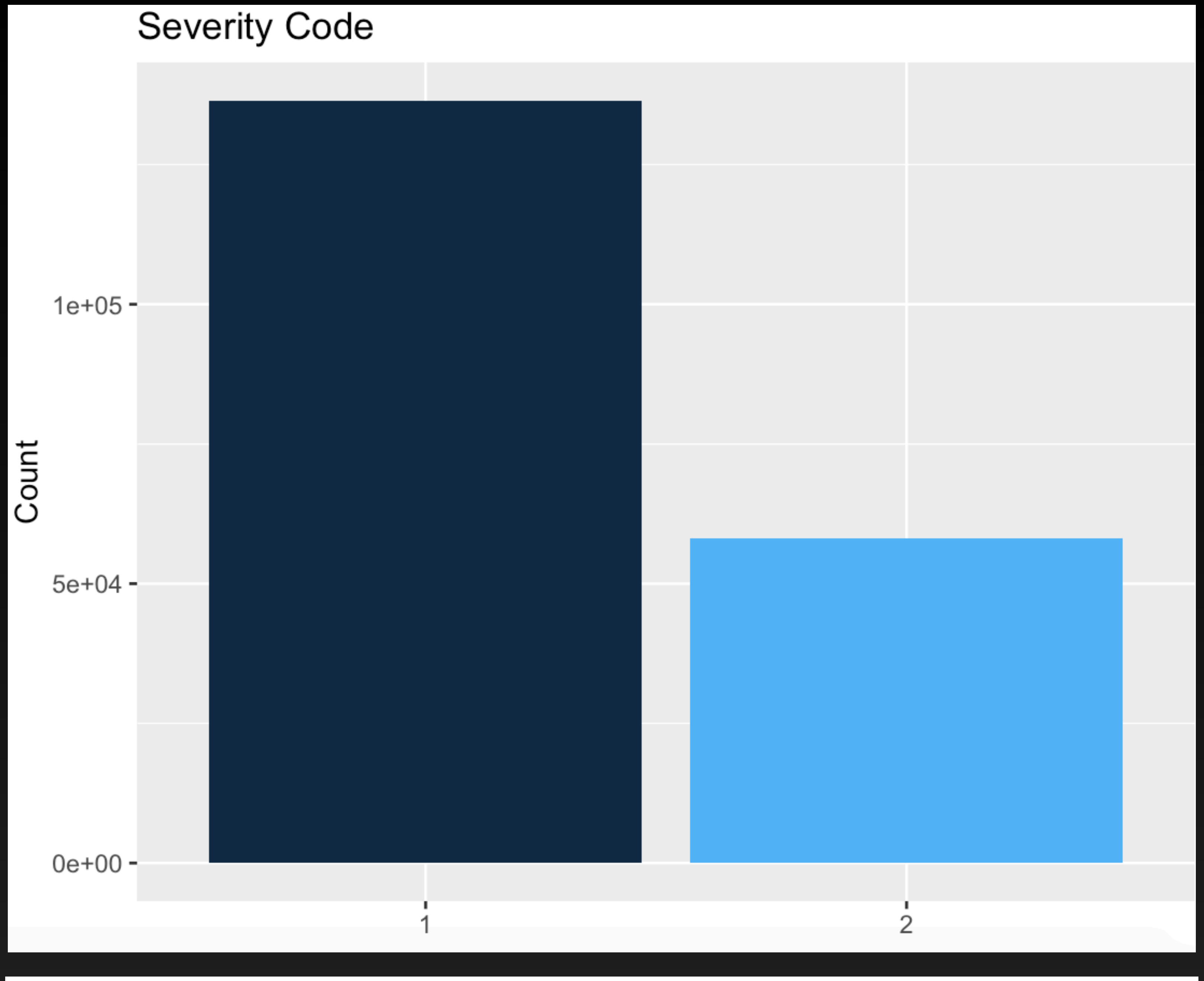


Variable	Description
WEATHER	Weather conditions (Overcast/ Raining/ Clear...)
LIGHTCOND	Light conditions (Dawn/ Daylight/ Dark...)
ROADCOND	Road condition (Wet/ Dry/ Ice ...)
ADDRTYPE	Collision address type (Alley/ Block/ Intersection)
INATTENTIONIND	The driver was inattentive (Yes/No/Unknown)
SPEEDING	The car was above the speed limit (Yes/No/Unknown)
UNDERINFL	The driver was under the influence (Yes/No/Unknown)

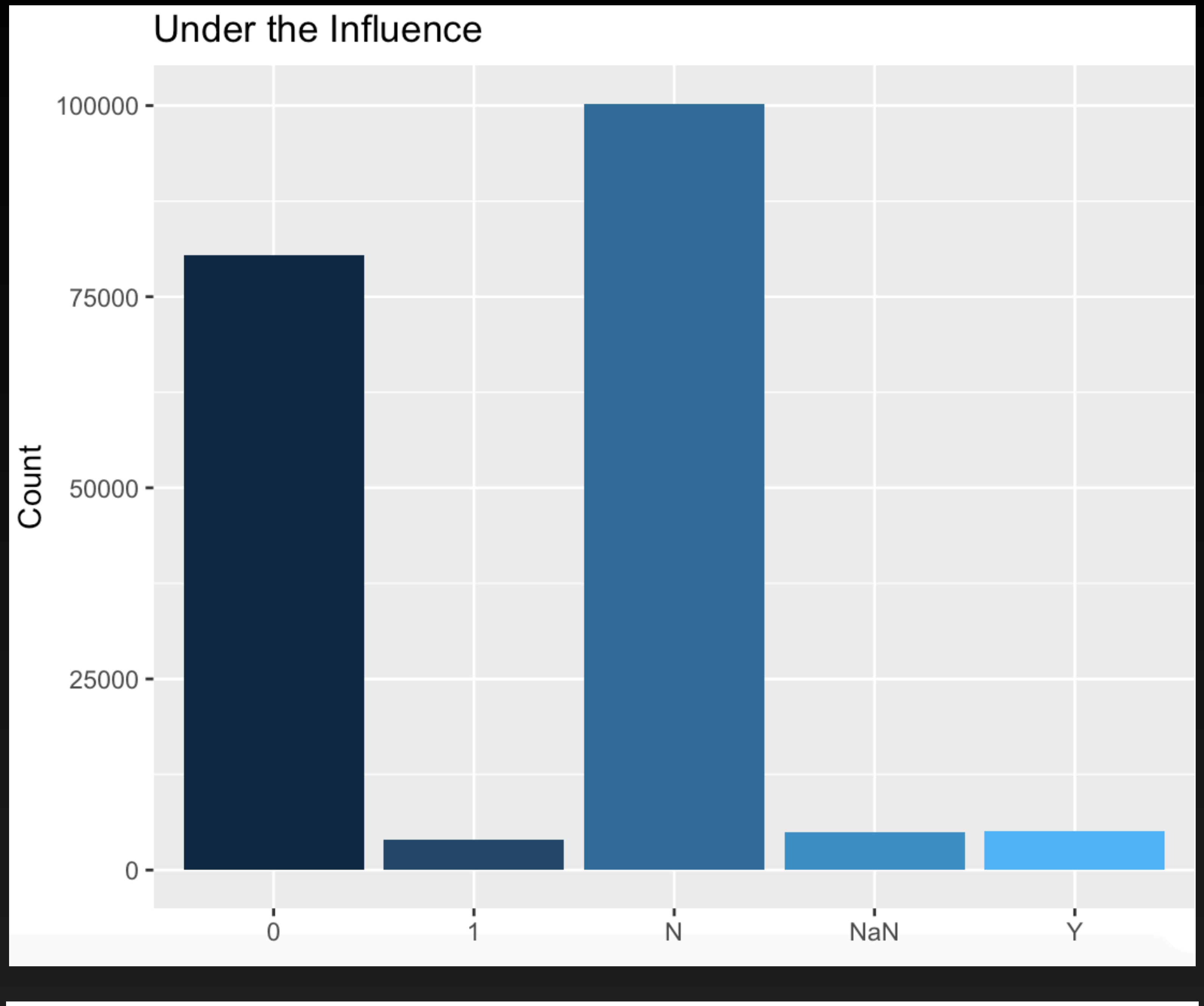
2. Data

2.2 Data Preparation

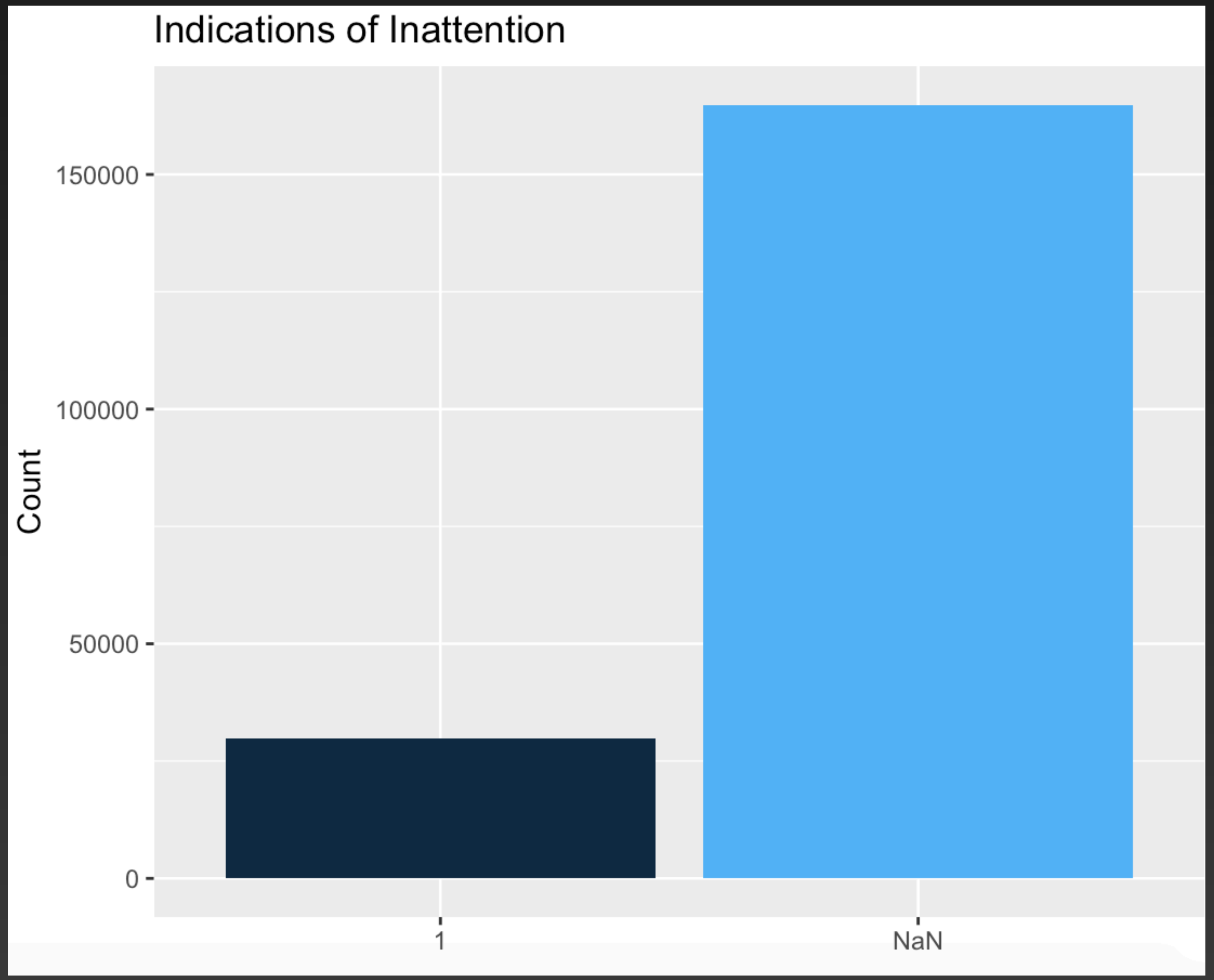
- Our data contains 194,673 accidents (rows) and 38 features or variables (columns). The variable SEVERITYCODE is encoded by giving value 1 if the accident entailed property damage only, and 2 if it entailed physical or health damage too. The variable UNDERINFL, is originally encoded as Y or 1 if there is evidence that the driver was under the influence, and as N or 0 if not. Similarly, the variable SPEEDING is encoded as Y if there is evidence that the driver was driving above the speed limit, and as missing value if there is a lack of evidence. The variable INATTENTIONIND, is encoded as Y if there is evidence that the driver was not paying attention while driving, and as a missing value if there is no such evidence. We show how each of the variables is distributed.



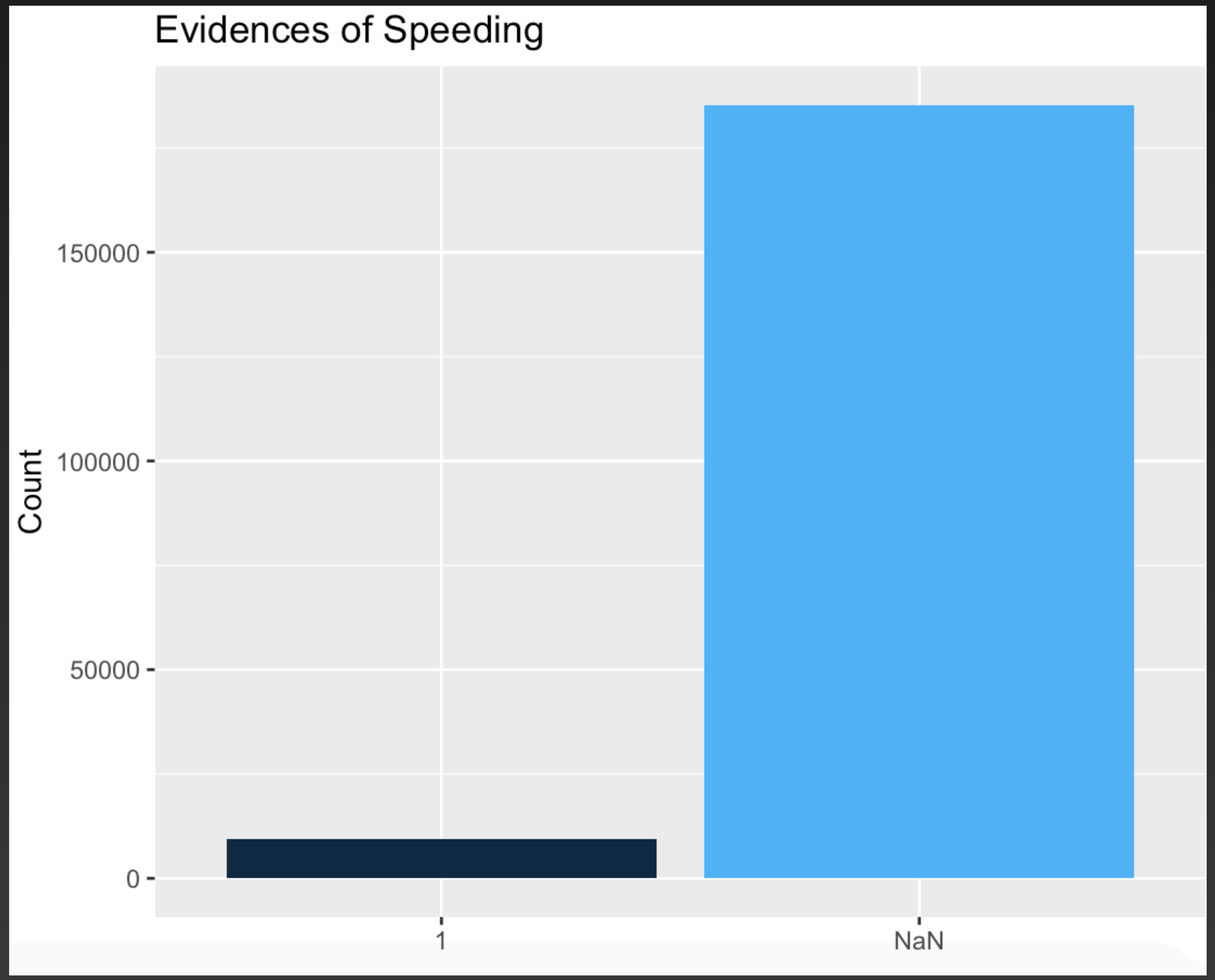
Severity Code



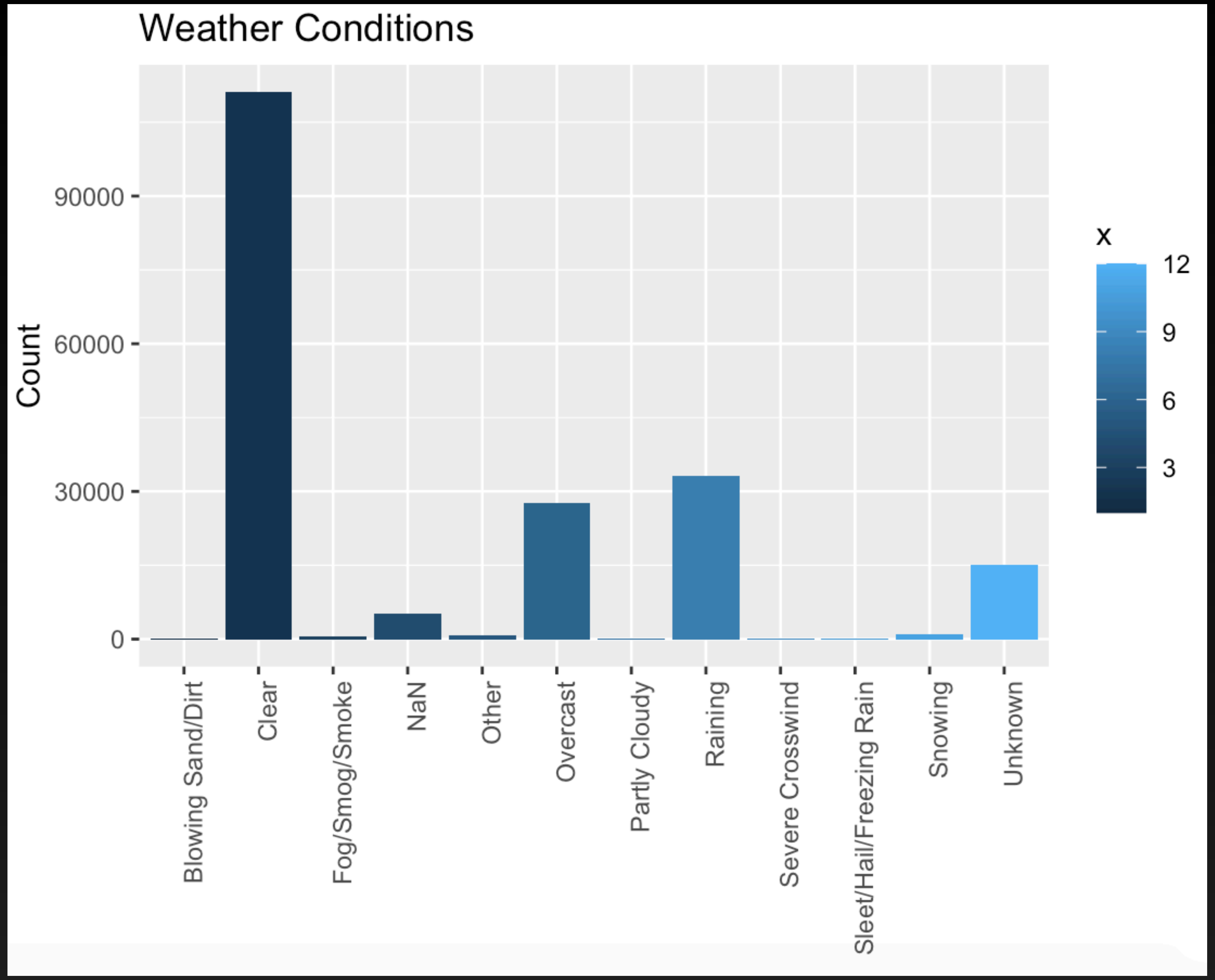
Under the Influence



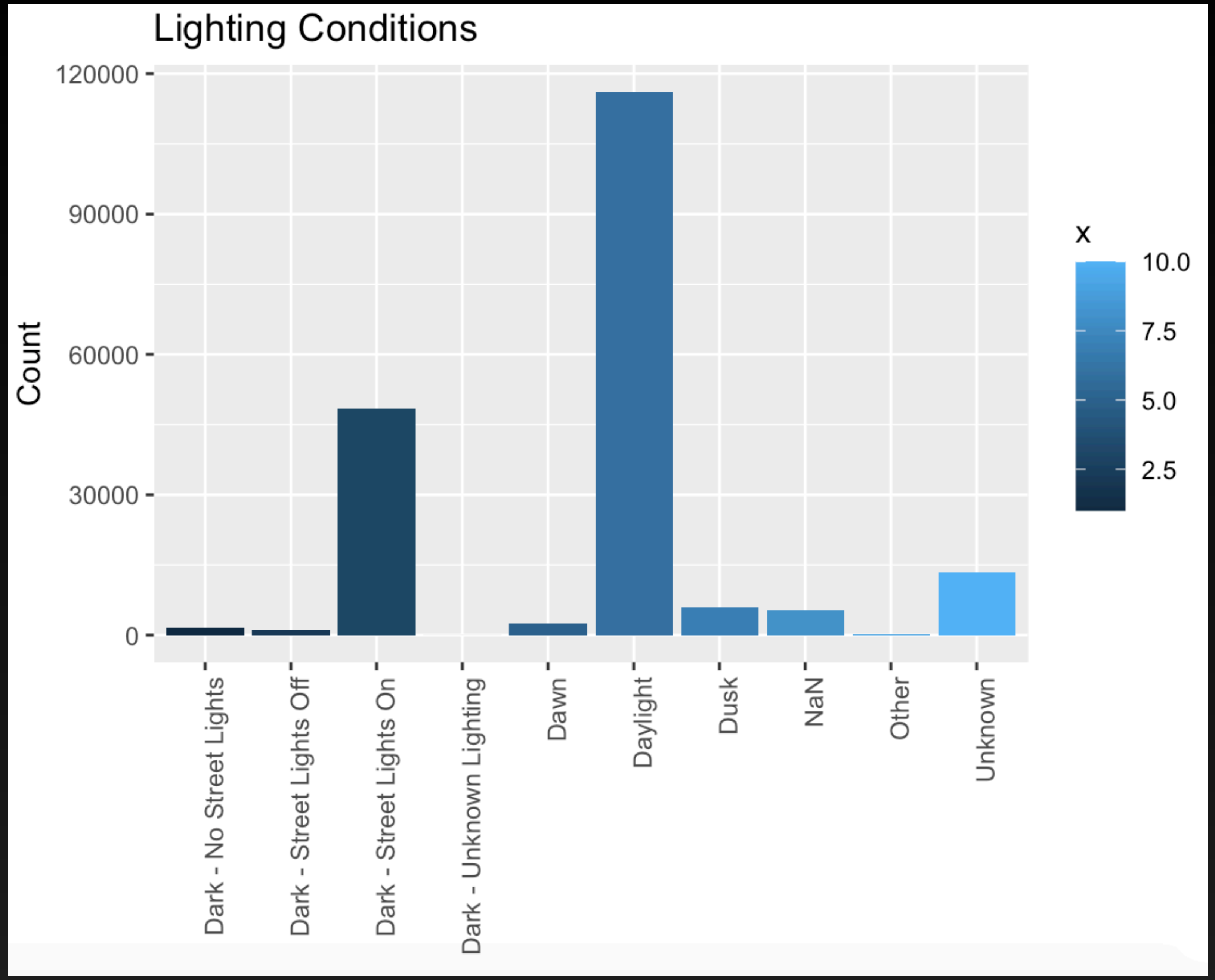
Indications of Inattention



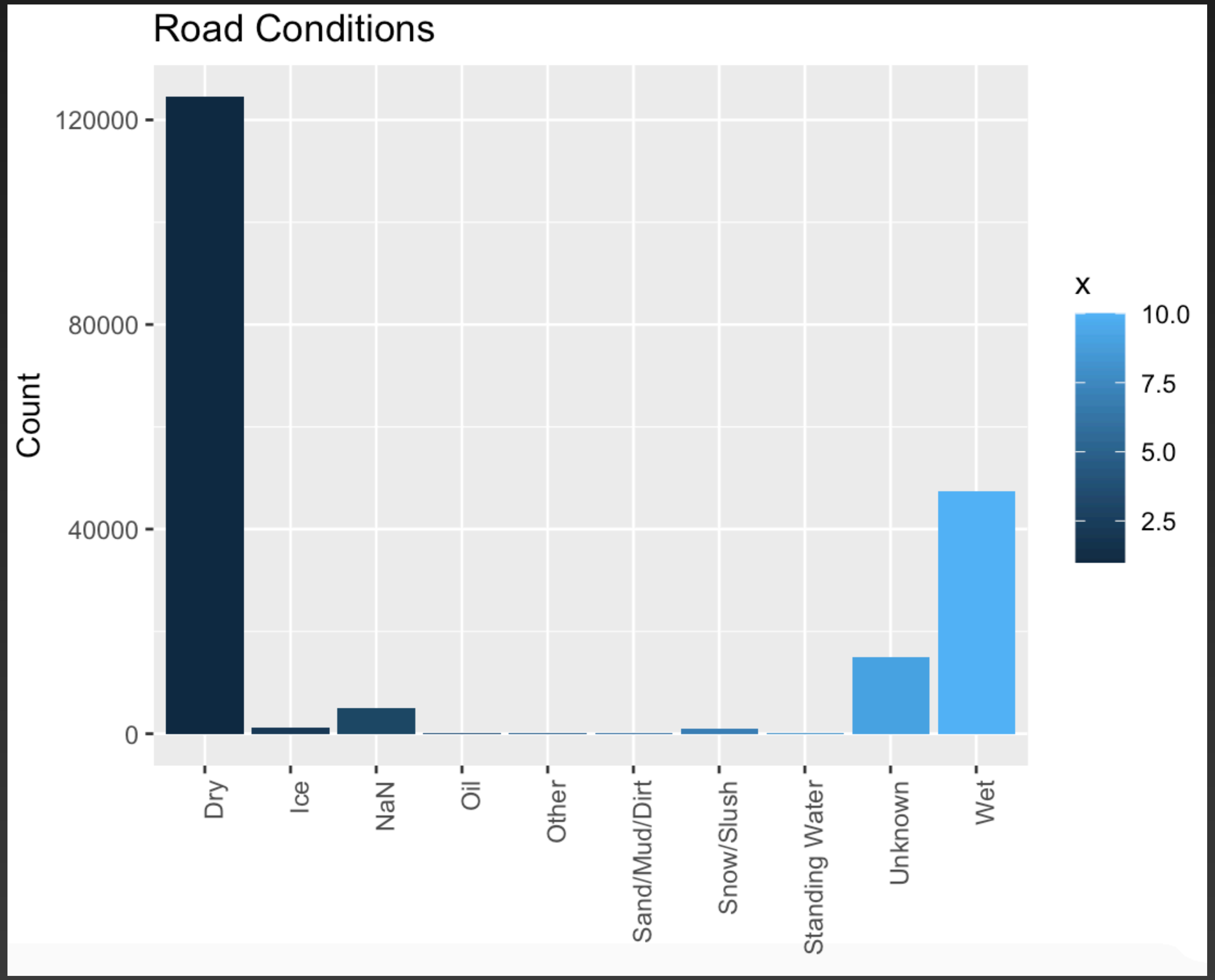
Evidences of Speeding



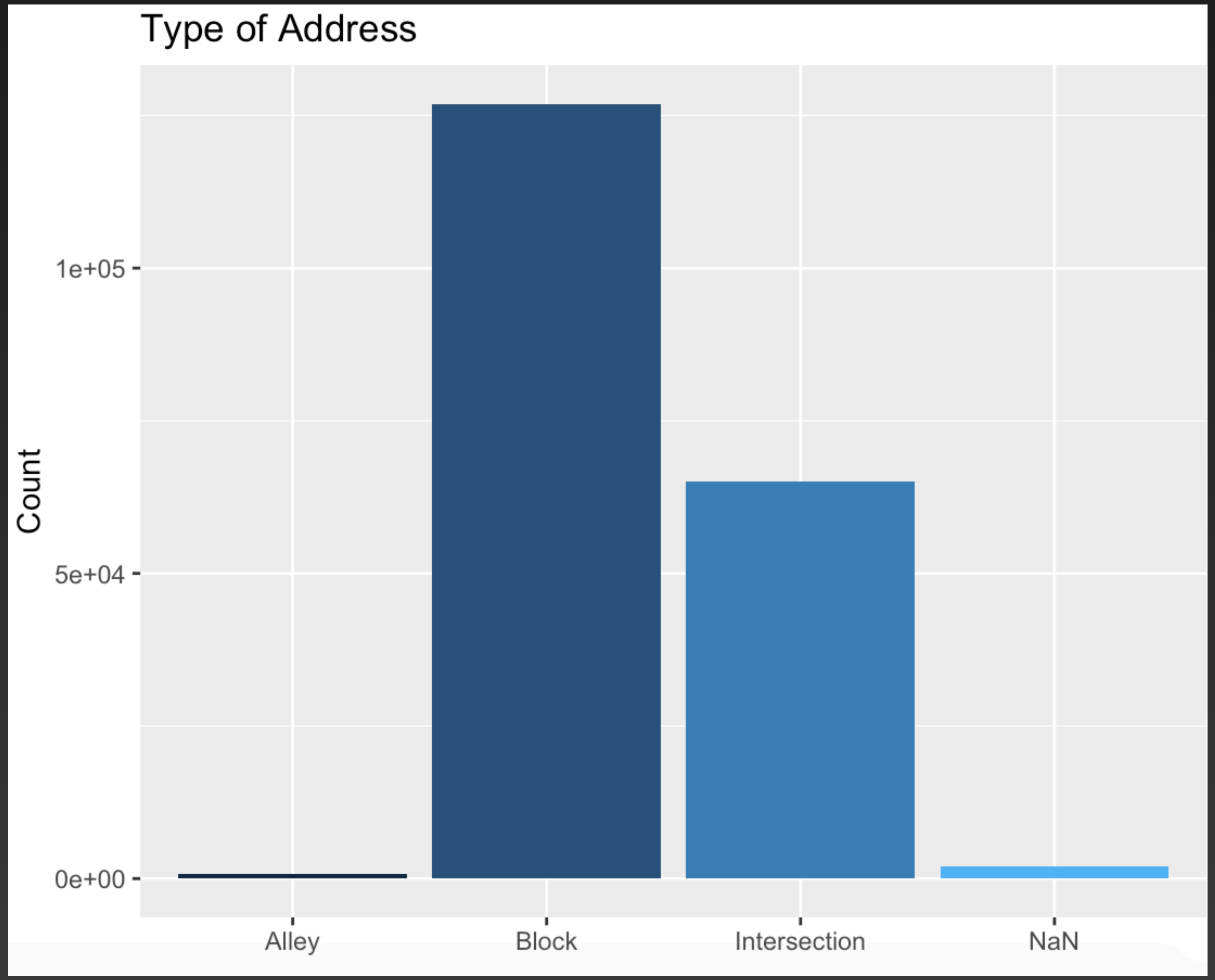
Weather Conditions



Lighting Conditions



Road Conditions



Type of Address

3. Methodology

3.1 Data preparation and visualization

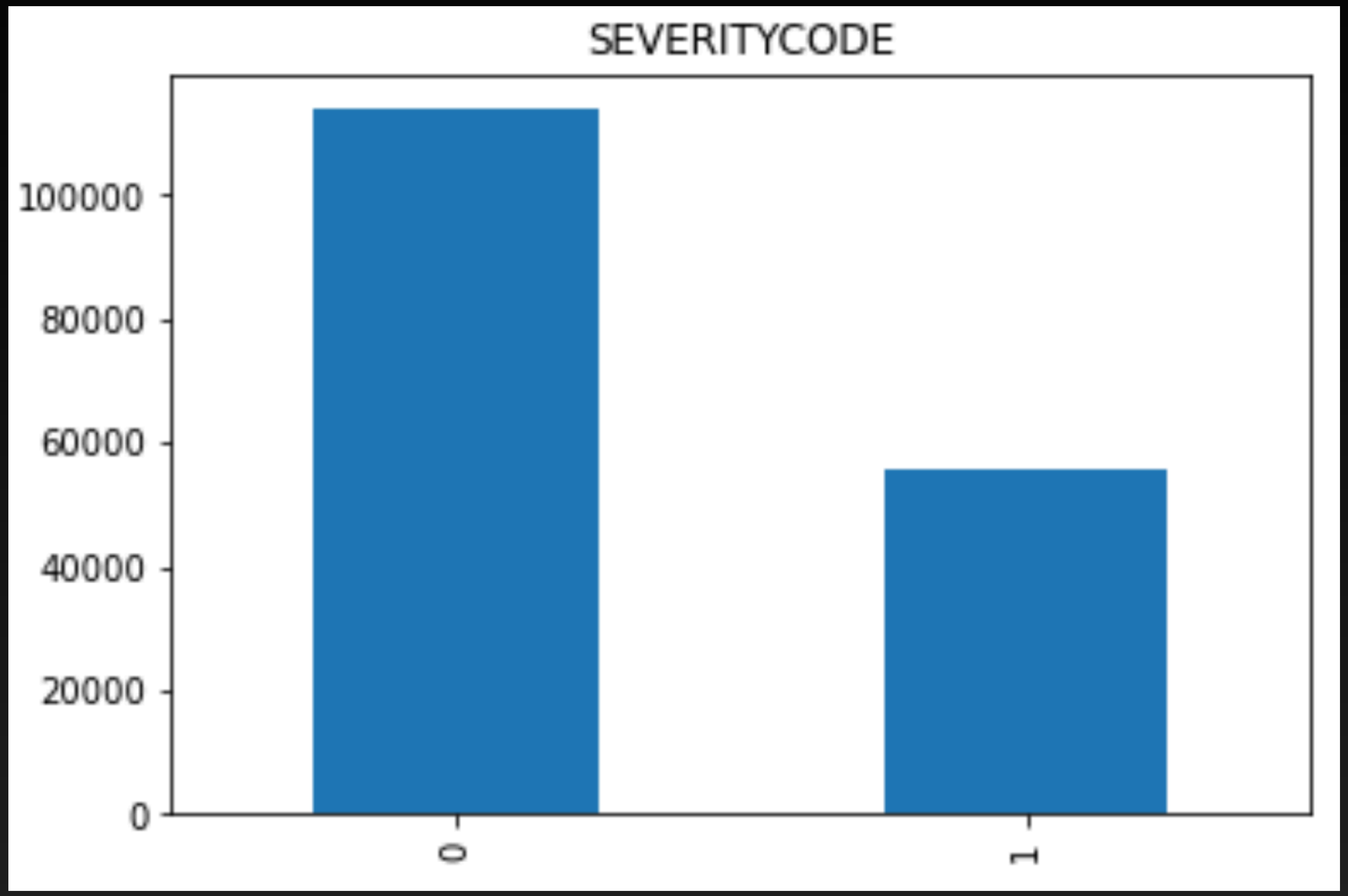
- Originally, as we already mentioned, the variable SEVERITYCODE is encoded by giving value 1 if the accident entailed property damage only, and 2 if it entailed physical or health damage too. For simplicity, we redefine it giving value 0 for no physical damage, and 1 if there are physical damages.

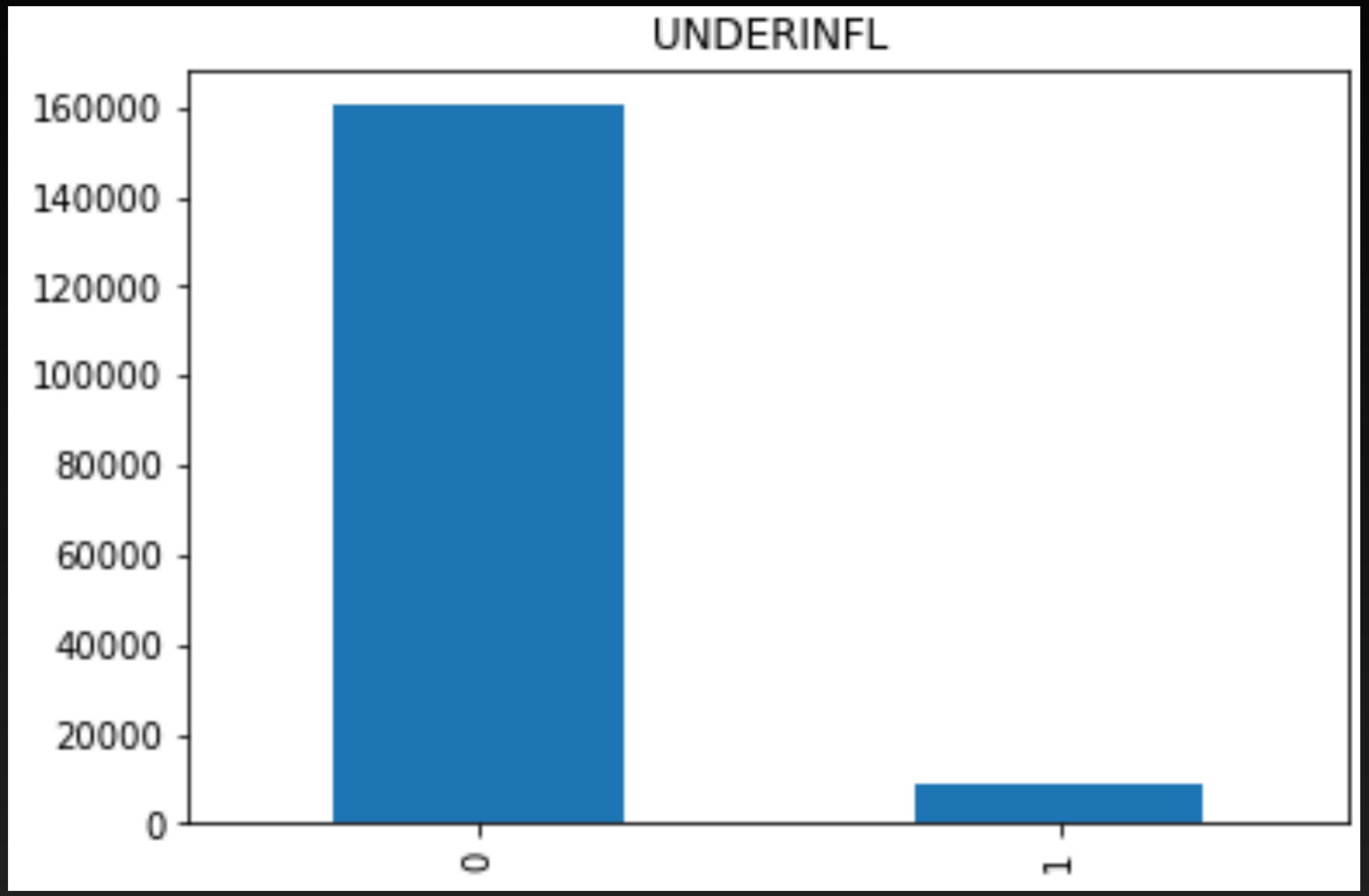
- The variable UNDERINFL, is originally encoded as Y or 1 if there is evidence that the driver was under the influence, and as N or 0 if not. We again redefine it as 0 (if not under the influence), and as 1 (if under the influence).

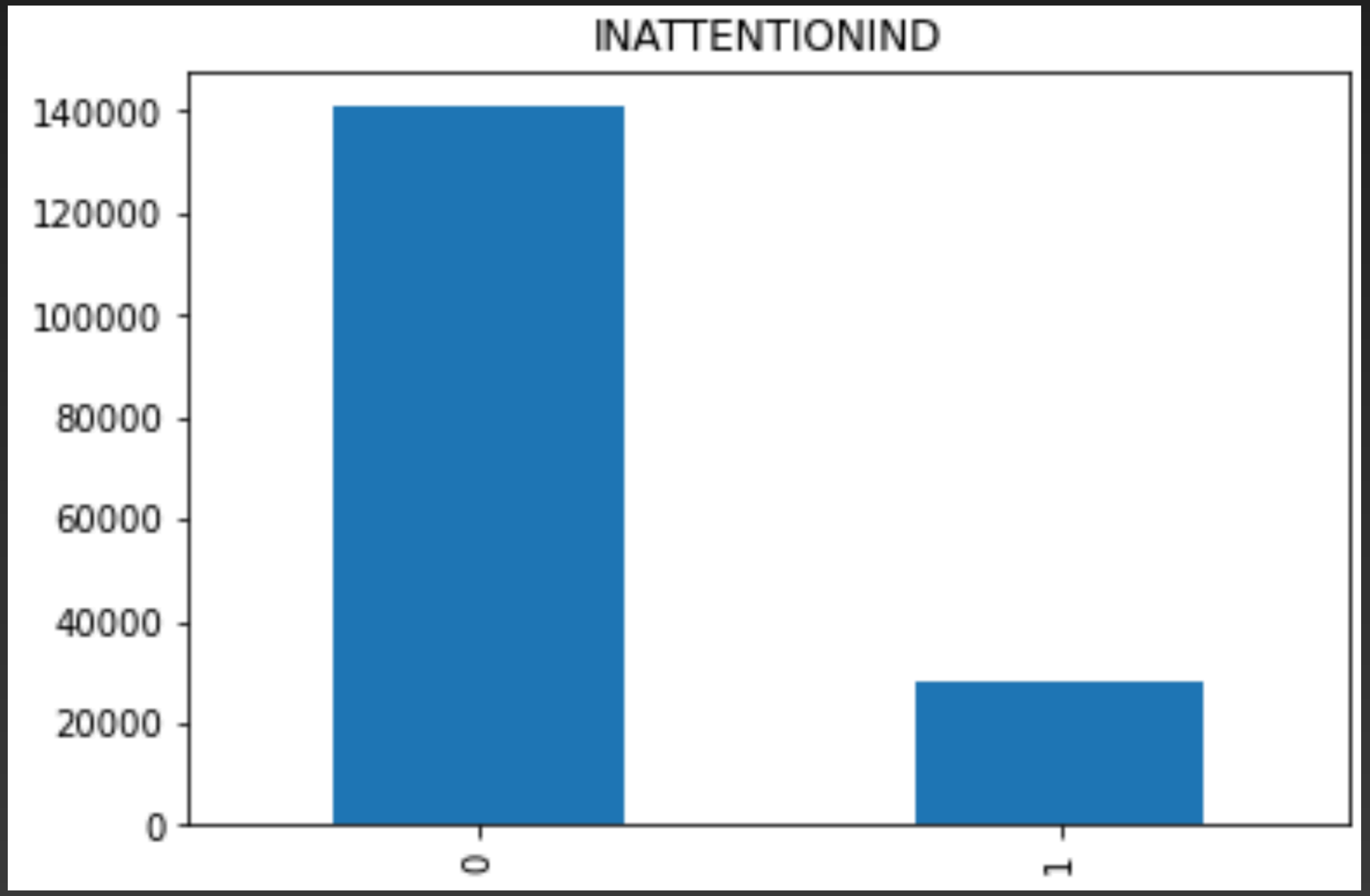
- Similarly, we modify the variable SPEEDING, which originally is encoded as Y if there is evidence that the driver was driving above the speed limit, and as missing value if there is a lack of evidence. We redefine it as 0 (if not evidence of speeding), and as 1 (if evidence of speeding).

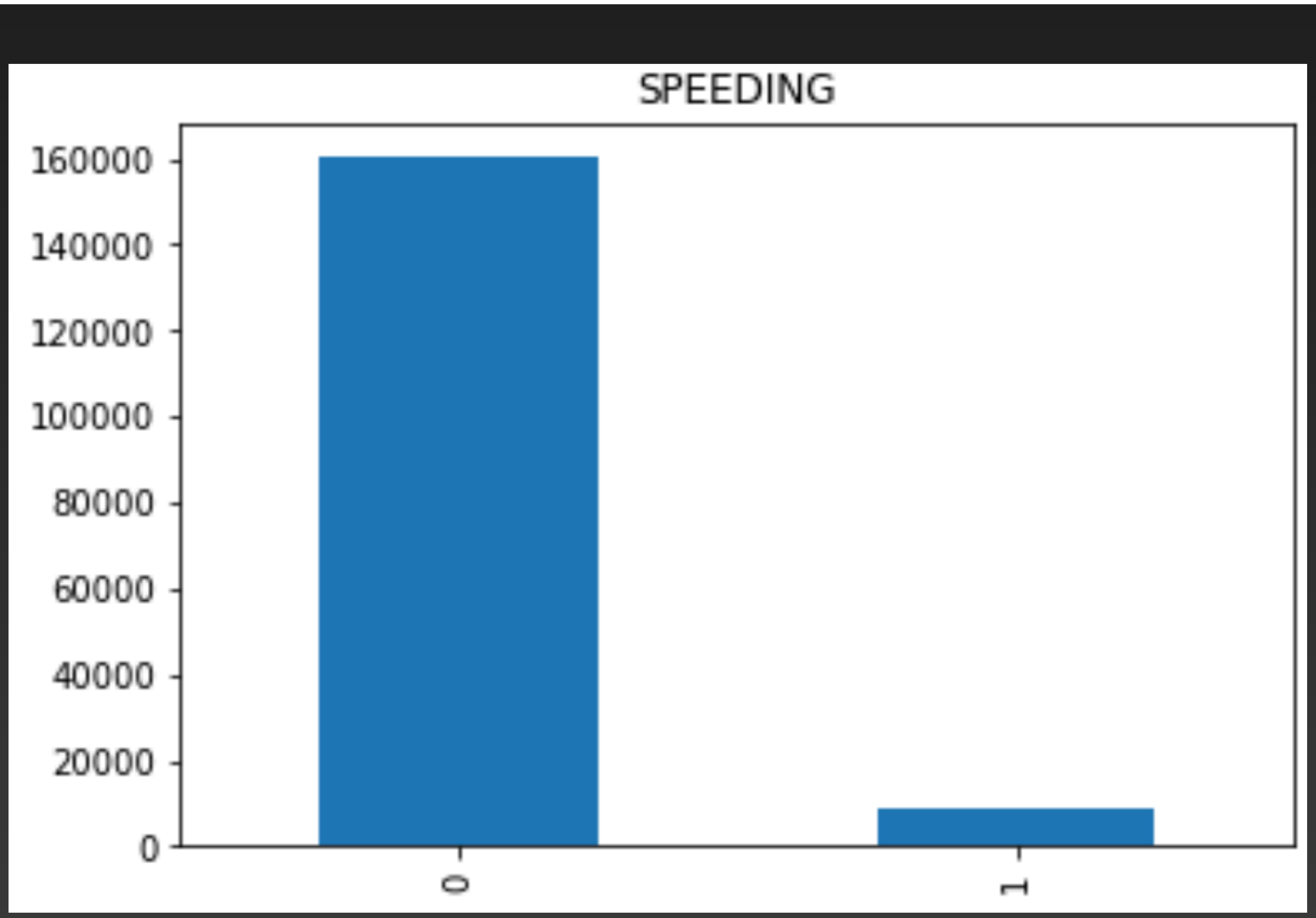
- We also modify the variable INATTENTIONIND, which originally is encoded as Y if there is evidence that the driver was not paying attention while driving, and as a missing value if there is no such evidence. We again redefine it as 0 (if not evidence of inattention), and as 1 (if evidence of inattention). Now, we substitute the entries for which the road condition appears as "Unknown" or "Other" for missing values. We also substitute the entries for which the LIGHTCOND appears as "Unknown" or "Other" for missing values. Next, we gather all the different entries for which conditions were dark without proper lighting ("Dark - Unknown Lighting", "Dark - No Street Lights", "Dark - Street Lights Off") for "Dark - No Lights”. Then, we replace the entries for which the WEATHER appears as "Unknown" or "Other" for missing values. In addition, we replace entries for which weather appears as "Partly Cloudy" for "Clear". Since we do not expect partly cloudy weather to have a different impact on accident severity than clear weather.

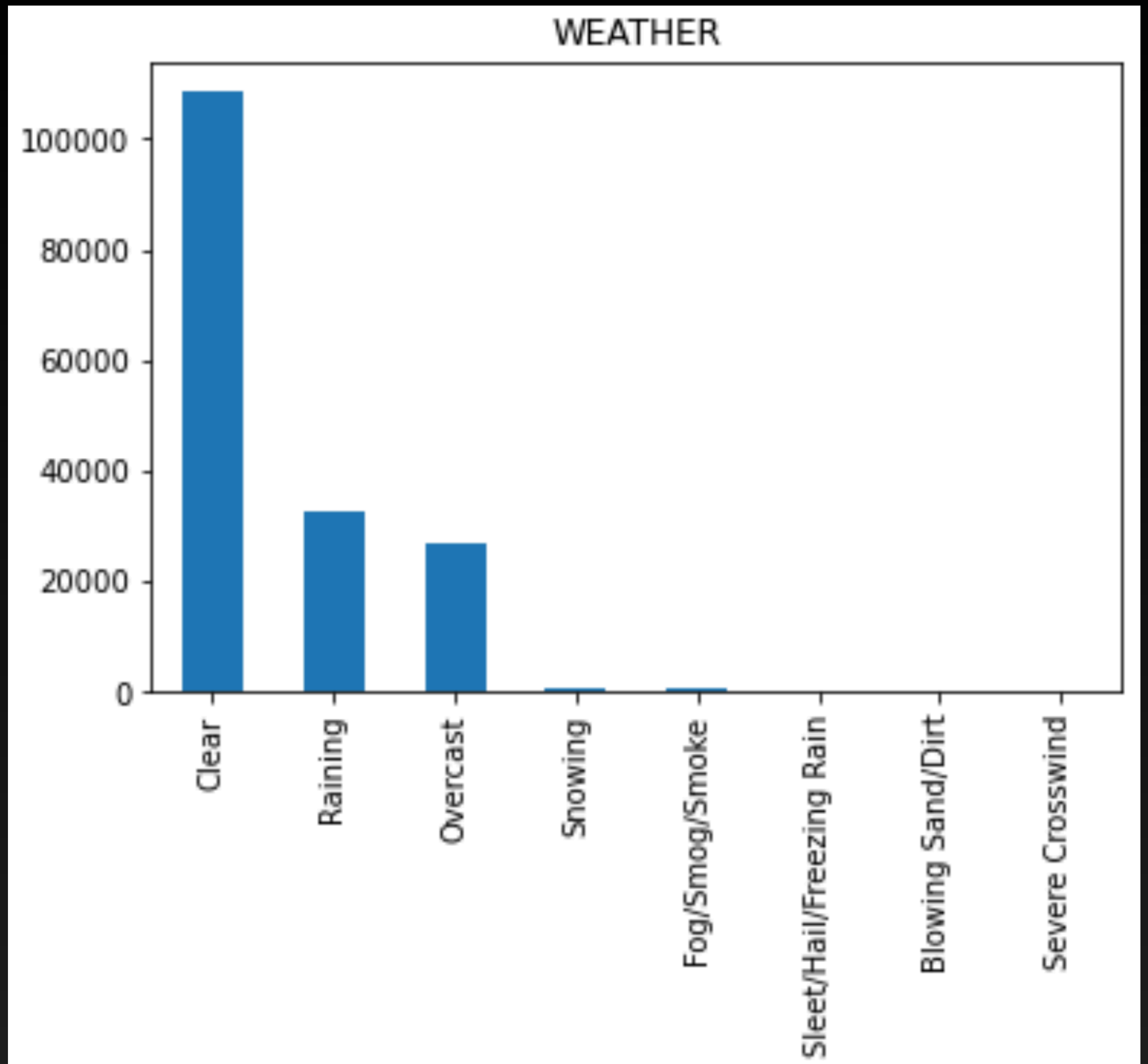
- Finally, we remove all the unnecessary columns in our dataset and after that, we remove every row containing any missing values. Our new dataset contains 169,247 rows and 8 columns. After removing all the missing values, we have not lost a dramatic amount of information. Let us plot the distributions of each of the variables in our new modified dataset.

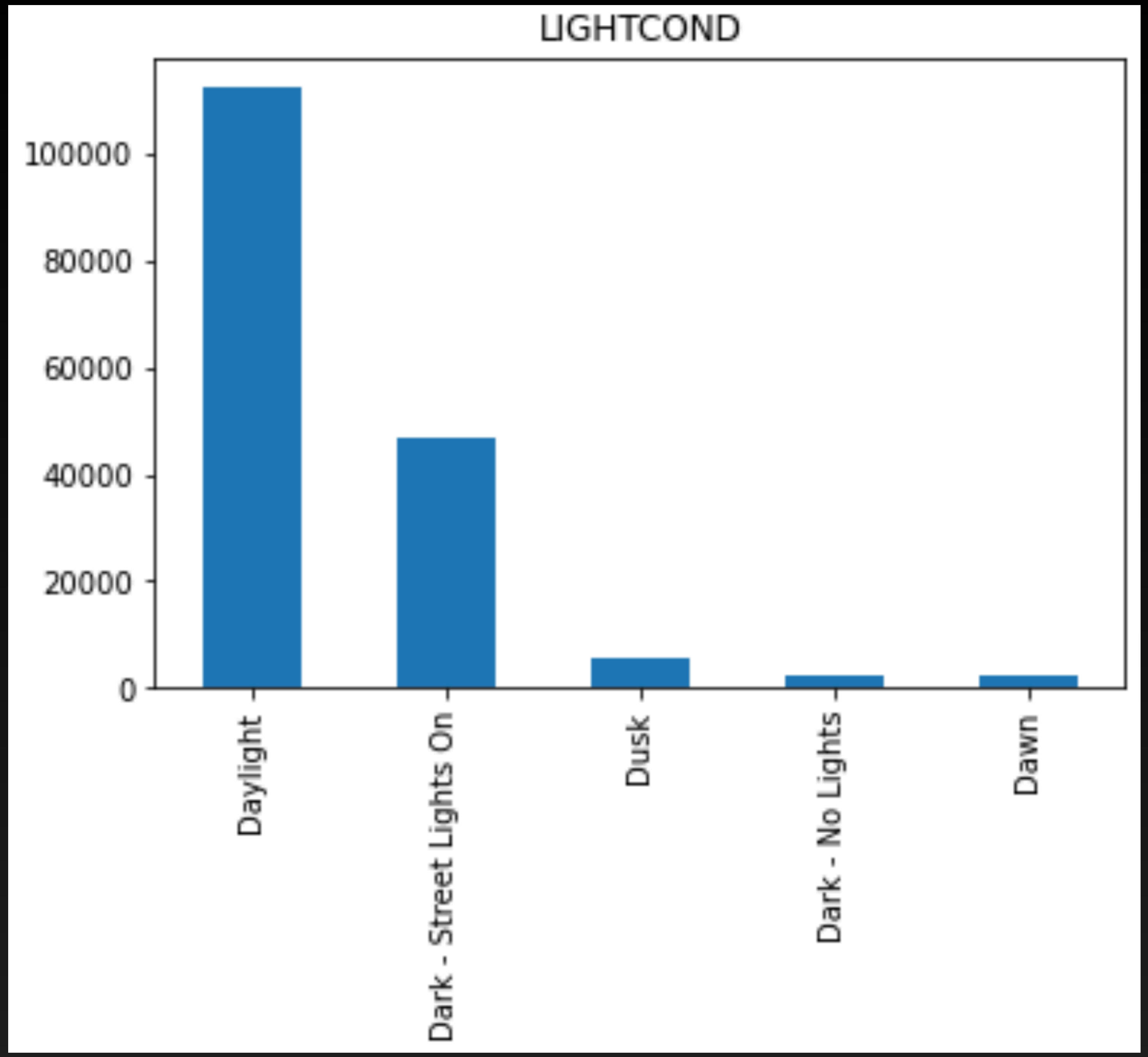

SEVERITYCODE
0
20000
40000
60000
80000
100000
0
1

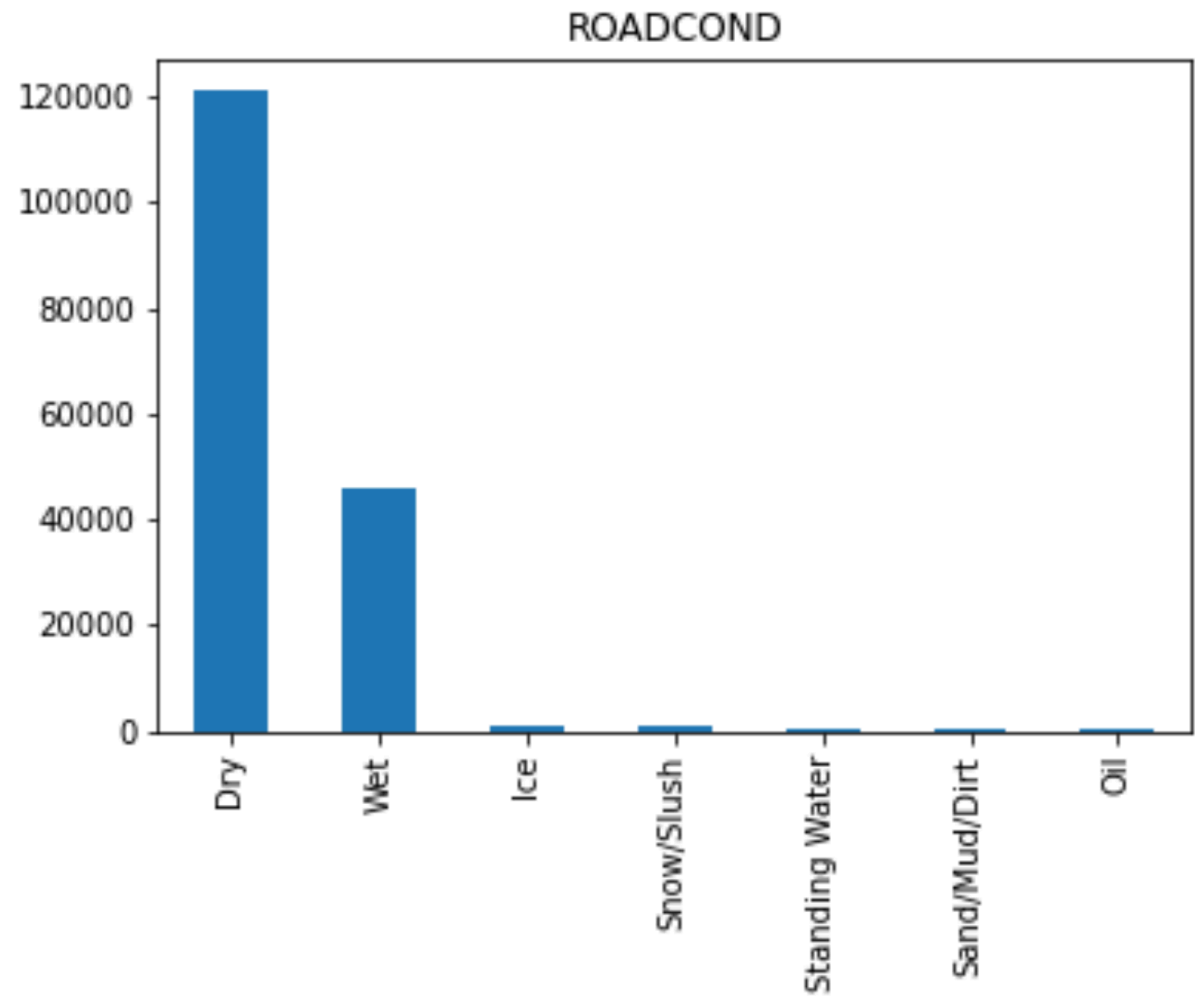

UNDERINFL
0
20000
40000
60000
80000
100000
120000
140000
160000
0
1

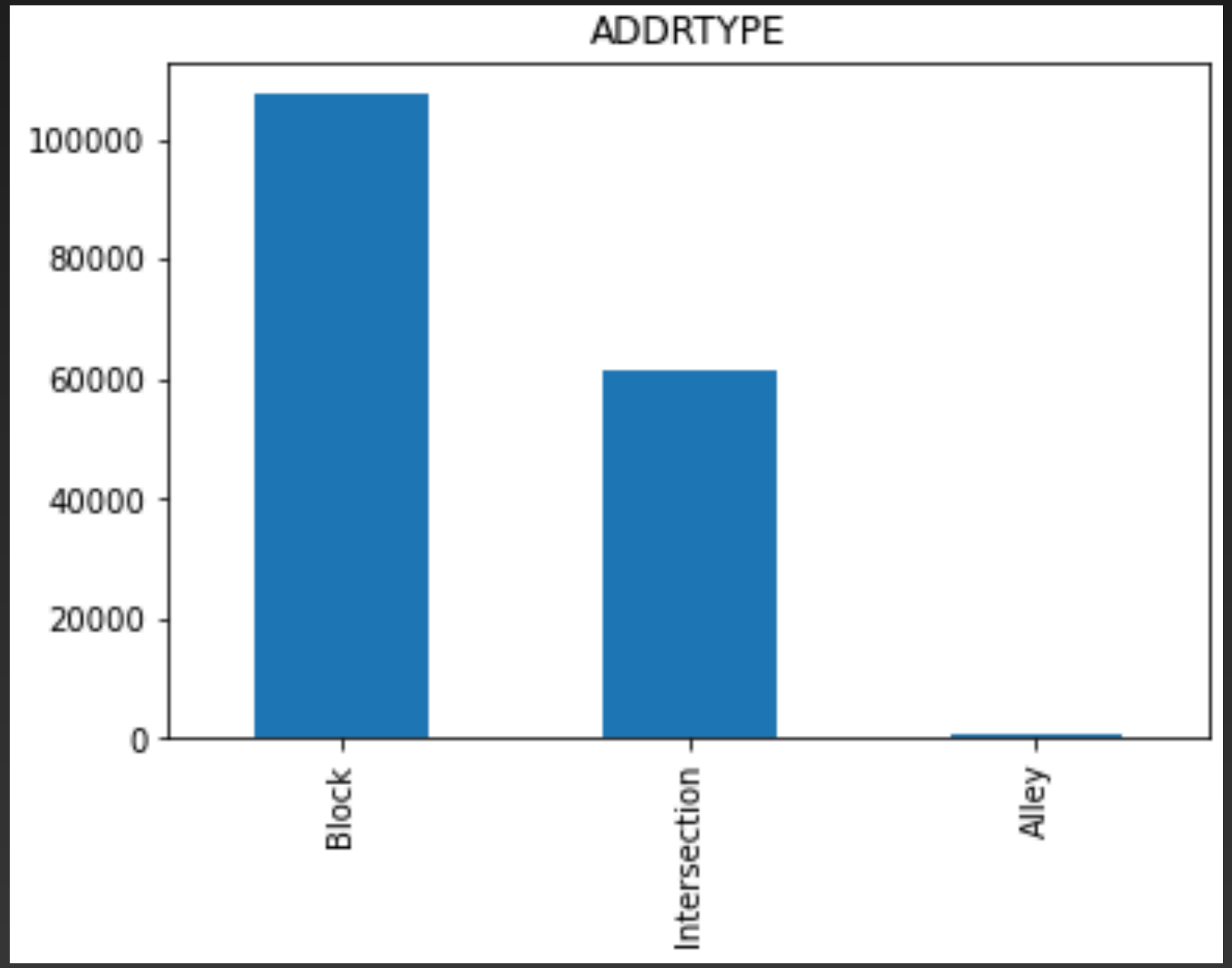

INATTENTIONIND
0
20000
40000
60000
80000
100000
120000
140000
0
1


SPEEDING
0
20000
40000
60000
80000
100000
120000
140000
160000
0
1


WEATHER
0
20000
40000
60000
80000
100000
Clear
Raining
Overcast
Snowing
Fog/Smog/Smoke
Sleet/Hail/Freezing Rain
Blowing Sand/Dirt
Severe Crosswind


LIGHTCOND
0
20000
40000
60000
80000
100000
Daylight
Dark - Street Lights On
Dusk
Dark - No Lights
Dawn


ROADCOND
0
20000
40000
60000
80000
100000
120000
Dry
Wet
Ice
Snow/Slush
Standing Water
Sand/Mud/Dirt
Oil

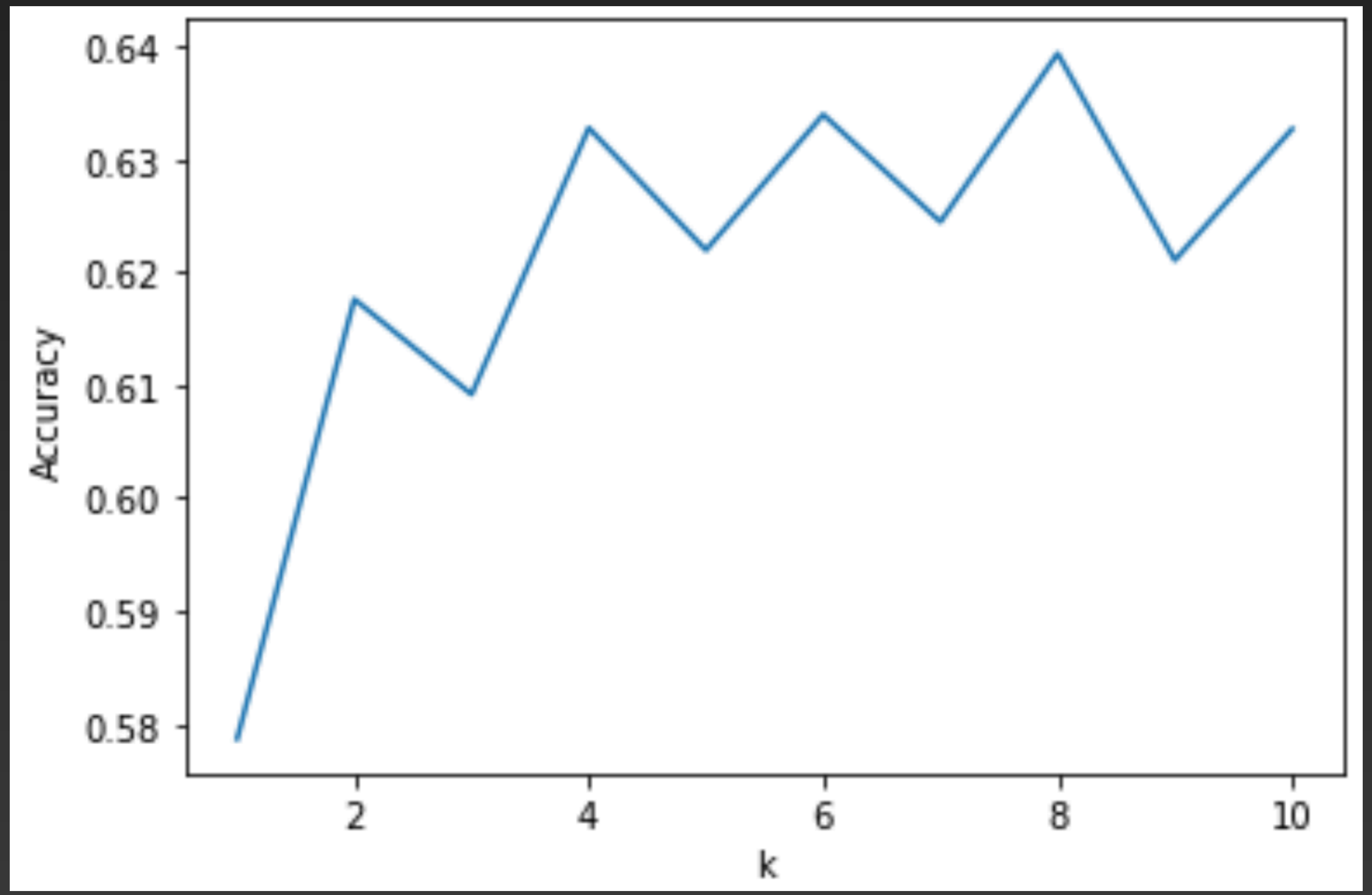

ADDRTYPE
0
20000
40000
60000
80000
100000
Block
Intersection
Alley

3. Methodology

3.2 Algorithms

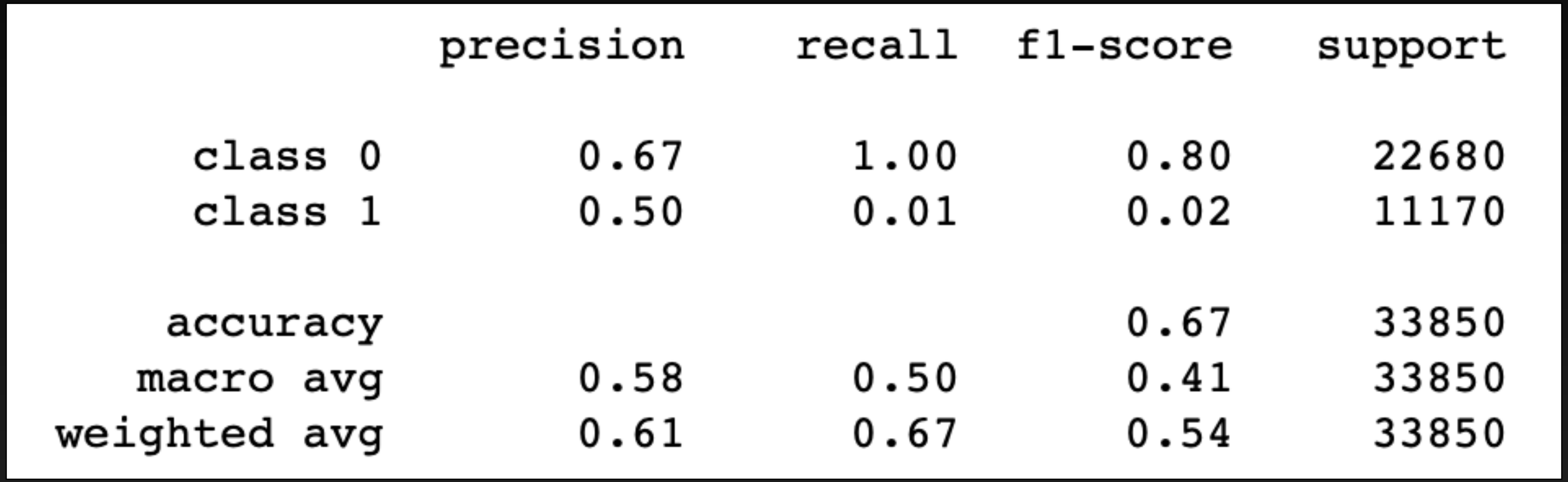
- We first split our data into training and testing sets, composed of 80% and 20% of the data respectively. We train 3 different Machine Learning models: Random Tree Classifier, Logistic Regression, and K-Nearest Neighbors

- From the scikit-learn library, we use the Decision Tree Classifier library to train a classification random tree model. The criterion chosen for the classifier is "entropy" and the max depth set to 6. Next, we use the K-Nearest Neighbor classifier to train the K-Nearest Neighbor machine learning model. The optimal value of the number of neighbors used (k), when we try different tuning values up to 10 NN, as shown below, is found at k=10.



4. Results

4.1 Random Tree

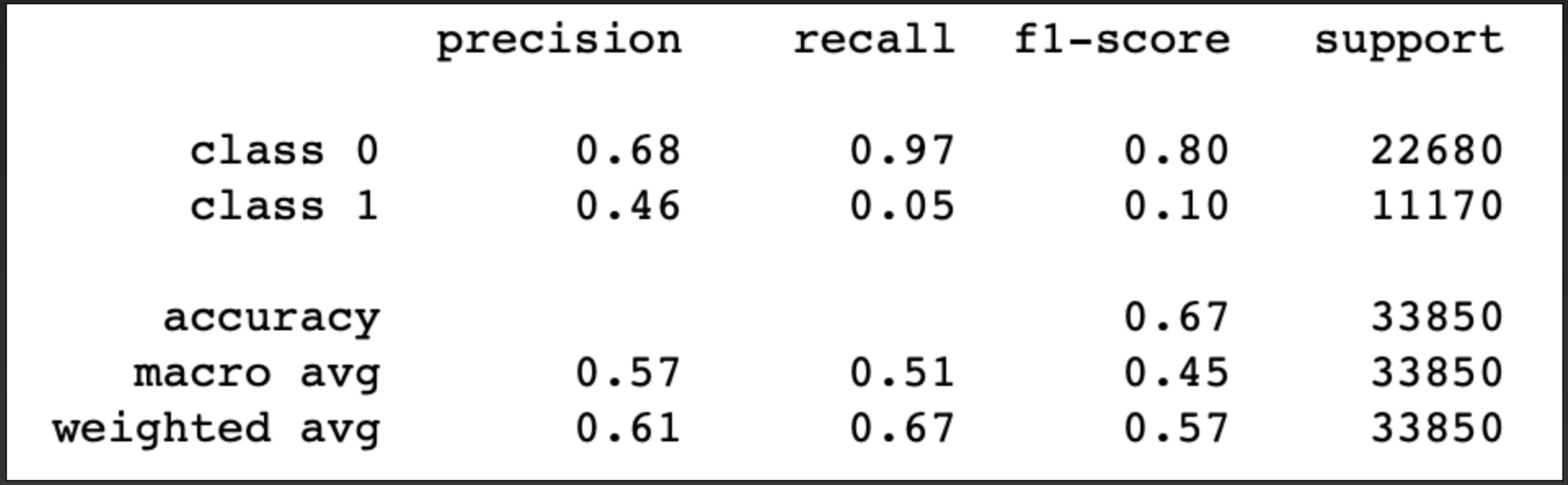


```
              precision    recall  f1-score   support

     class 0       0.67      1.00      0.80     22680
     class 1       0.50      0.01      0.02     11170

    accuracy                           0.67     33850
   macro avg       0.58      0.50      0.41     33850
weighted avg       0.61      0.67      0.54     33850
```

4.2 Logistic Regression



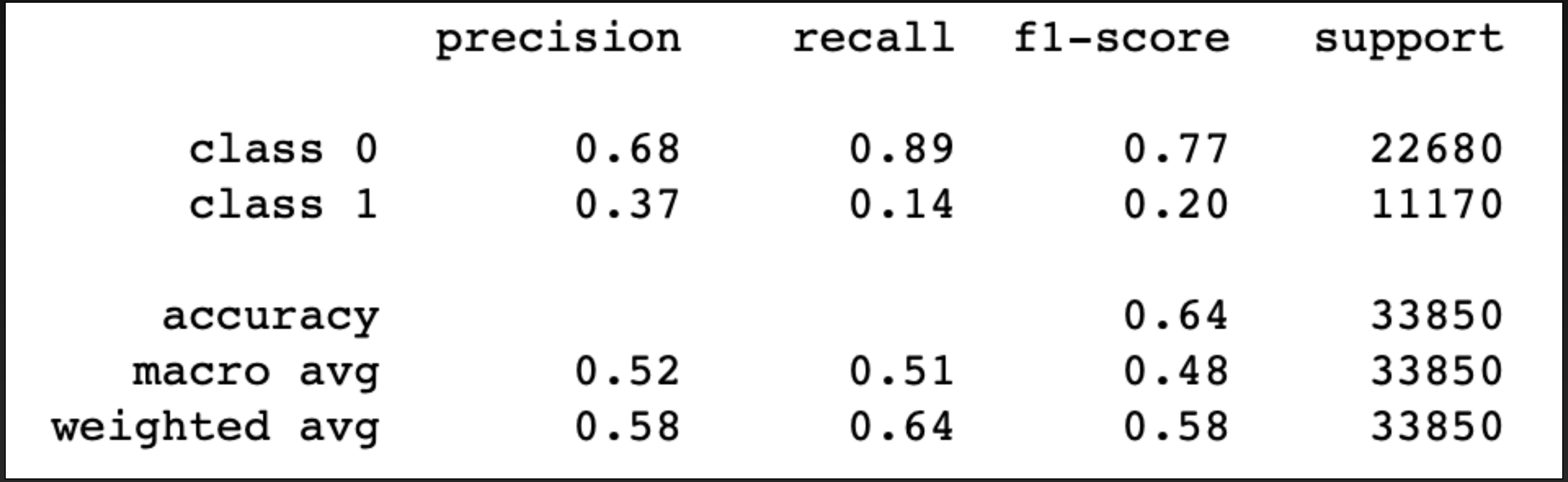
```
              precision    recall  f1-score   support

     class 0       0.68      0.97      0.80     22680
     class 1       0.46      0.05      0.10     11170

    accuracy                           0.67     33850
   macro avg       0.57      0.51      0.45     33850
weighted avg       0.61      0.67      0.57     33850
```

4. Results

4.3 K-Nearest Neighbors



```
              precision    recall  f1-score   support

     class 0       0.68      0.89      0.77     22680
     class 1       0.37      0.14      0.20     11170

    accuracy                           0.64     33850
   macro avg       0.52      0.51      0.48     33850
weighted avg       0.58      0.64      0.58     33850
```

5. Discussion

- Precision is obtained by dividing true positives by the sum of true positive plus false positives. The recall is obtained by dividing true positives by true positive plus false negatives. f1-score is a measure of the accuracy of the model, which is the harmonic mean of the model's precision and recall. We can see that in terms of overall accuracy, the model that gives the best results are both the random tree and logistic regression models, both with an accuracy score of 0.67. However, both the overall precision and the overall f1-score given by the KNN model is better than the one offered by any of the other two methods.

6. Conclusions

- We see that the trained models could have performed much better if various improvements were made. On the one hand, dealing with an unbalanced dataset regarding the target variables produces a biased model that is biased towards the higher prevalence of one of the groups. Balancing this dataset would be a huge improvement for our project. On the other hand, if we had fewer missing variables or unknown entries, our information regarding predictive features would be much richer. This is left as further research.