

Vehicle crashes dataset of New York City

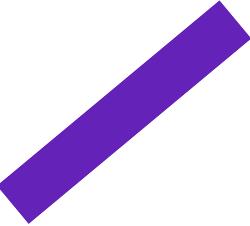
Niha R Gali

About

Problem Statement

Understanding the vehicle crashes dataset of New York and develop a multiclass classifier to classify the contributing factor for those crashes

The Motor Vehicle Collisions vehicle table contains details on each vehicle involved in the crash. Each row represents a motor vehicle involved in a crash. The data in this table goes back to April 2016 when crash reporting switched to an electronic system.

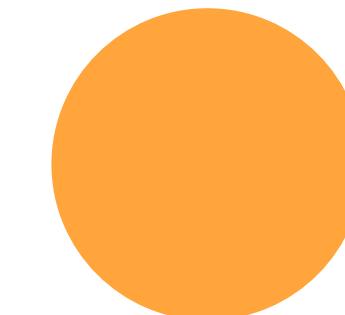


Data Source

Dataset link:

<https://data.cityofnewyork.us/Public-Safety/Motor-Vehicle-Collisions-Vehicles/bm4k-52h4>

- Size : 114.4MB
- Rows : 3.7M
- Columns : 25
- Each row represents motor vehicle involved in a crash.



Attributes

- UNIQUE_ID Unique record code generated by system.
- COLLISION_ID Crash identification code. Foreign Key, matches unique_id from the Crash table.
- CRASH_DATE Occurrence date of collision
- CRASH_TIME Occurrence time of collision
- VEHICLE_ID Vehicle identification code assigned by system
- STATE_REGISTRATION State where vehicle is registered.
- VEHICLE_TYPE Type of vehicle based on the selected vehicle category

Attributes

- VEHICLE_MAKE Vehicle make
- VEHICLE_MODEL Vehicle model
- VEHICLE_YEAR Year the vehicle was manufactured
- TRAVEL_DIRECTION Direction vehicle was traveling
- VEHICLE_OCCUPANTS Number of vehicle occupants
- DRIVER_SEX Gender of driver
- DRIVER_LICENSE_STATUS License, permit, unlicensed

Attributes

- DRIVER_LICENSE_JURISDICTION State where driver license was issued.
- PRE_CRASH Pre-crash action: Going straight, making right turn, passing, backing, etc.
- POINT_OF_IMPACT Location on the vehicle of the initial point of impact
- VEHICLE_DAMAGE Location on the vehicle where most of the damage occurred
- VEHICLE_DAMAGE_1 Additional damage locations on the vehicle
- VEHICLE_DAMAGE_2 Additional damage locations on the vehicle
- VEHICLE_DAMAGE_3 Additional damage locations on the vehicle

Attributes

- PUBLIC_PROPERTY_DAMAGE Public property damaged (Yes or No)
- PUBLIC_PROPERTY_DAMAGE_TYPE Type of public property damaged (ex. Sign, fence, light post, etc.)
- CONTRIBUTING_FACTOR_1 Factors contributing to the collision for designated vehicle
- CONTRIBUTING_FACTOR_2 Factors contributing to the collision for designated vehicle

Questions for analysis

- Understanding the contributing factor
- To find the extent of property damage
- What part of vehicle was damaged mostly ?
- What vehicles were involved in the accident ?
- Identify drivers license and sex distribution
- Accident distribution over time

Data Cleaning and Engineering

- There are very few rows like unique_id, collision_id, date and time with no missing data.
- Columns such as VEHICLE_MODEL, VEHICLE_DAMAGE_1, VEHICLE_DAMAGE_2, VEHICLE_DAMAGE_3, PUBLIC_PROPERTY_DAMAGE_TYPE have more than 65% of missing values.
- Reduced the number of rows with values 'Unspecified' by 49.8%.
- More than 90% of the data has 'No' that means in more than 90% of crashes there was no property damage.
- Most of the damages have occurred at the front end of the car compared to other parts.
- A great portion of cars that were involved in an accident were either going straight or were parked.

Data Cleaning and Engineering

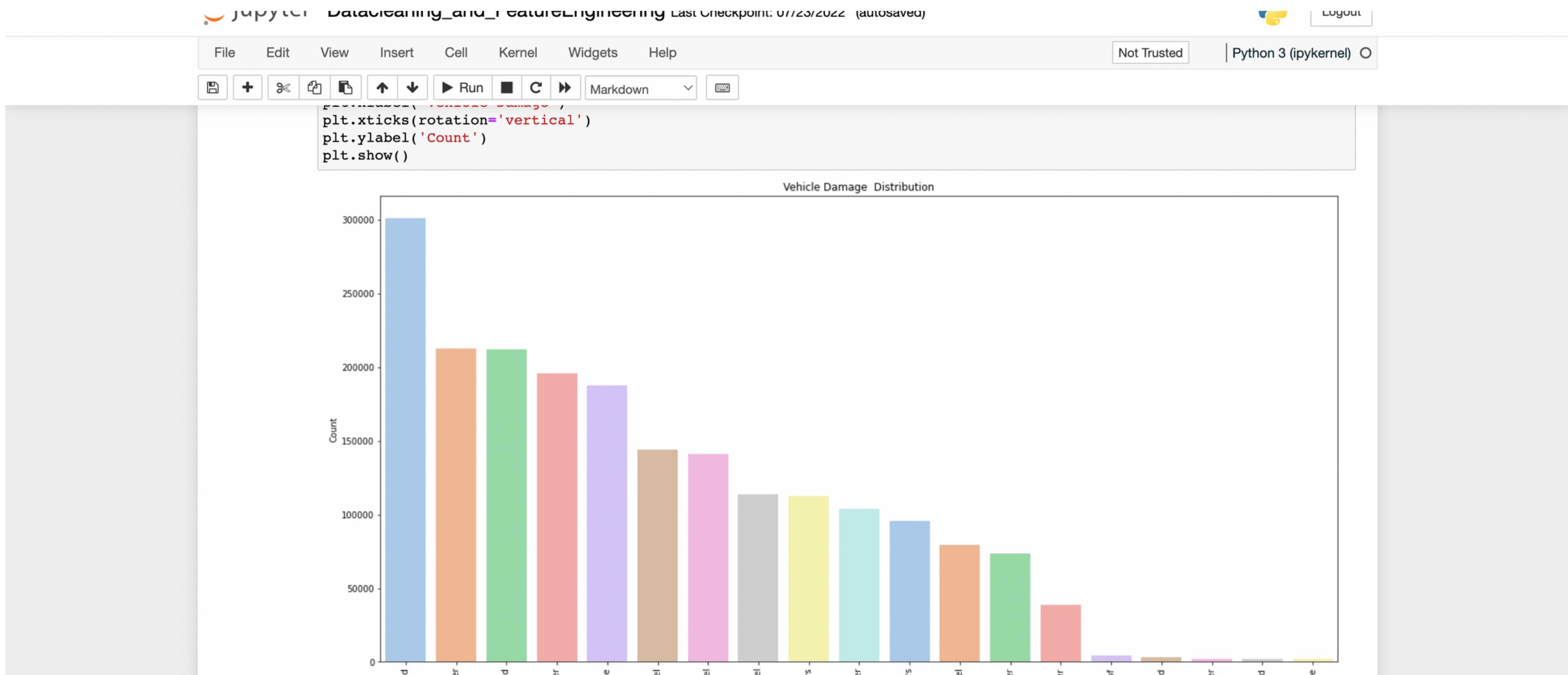
- About 95% of people who were involved in an accident are licensed.
- Majority of accidents were caused by Males.
- In 80% of the cases there is only one person in the vehicle which was involved in an accident.
- Dropping the unnecessary columns
 - UNIQUE_ID, COLLISION_ID, and VEHICLE_ID as they are unique and don't add value for this analysis.
 - CRASH_DATE, CRASH_TIME, VEHICLE_YEAR, CONTRIBUTING_FACTOR_2, and VEHICLE_MAKE as we extracted the main information from them.
 - VEHICLE_MODEL, PUBLIC_PROPERTY_DAMAGE, PUBLIC_PROPERTY_DAMAGE_TYPE as most of the data is missing.
 - STATE_REGISTRATION, DRIVER_LICENSE_STATUS, TRAVEL_DIRECTION, and DRIVER_LICENSE_JURISDICTION as data is highly skewed
 - VEHICLE_DAMAGE_cleaned as POINT_OF_IMPACT and VEHICLE_DAMAGE_cleaned very highly correlated.

Exploratory Data Analysis

Observations

- I observed that during the years 2017, 2018 there was an average of 175,000 accident.
- There is a decrease in the number of accidents in 2020 and 2021 due to the pandemic.
- I observed that there was a sudden spike in the number of accidents in the months of May and June.
- Contrast to common conception most of the accidents took place in the afternoon.
- Lack of attention was the major contributor for accidents followed by following too closely, not DUI or DWI.
- For the accidents having damages on the front end the driver was going straight.
- For the accidents having damages on the back end the driver was mostly backing the car.
- When there is only one person in the vehicle lack of attention and over speeding were the causes for accidents.
- Compared to females, men cause a lot of accidents due to distraction.

Exploratory Data Analysis



Exploratory Data Analysis

Gmail YouTube ENMG 652 Manag... Maps (1) Tu Chai... Untitled document...

jupyter Datacleaning_and_FeatureEngineering Last Checkpoint: 07/23/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

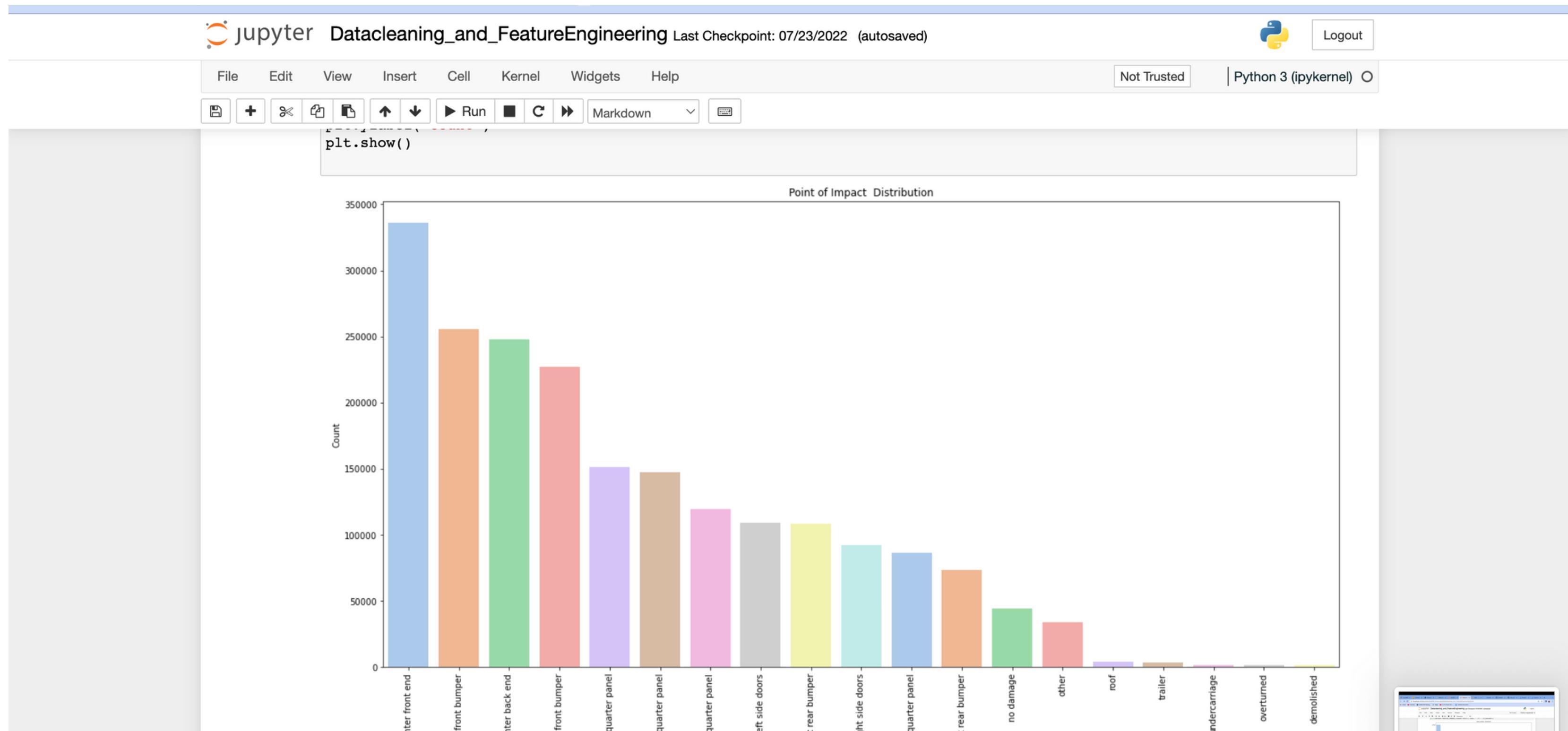
Run

print(dataset.VEHICLE_DAMAGE_cleaned.isnull().sum() * 100 / len(dataset))

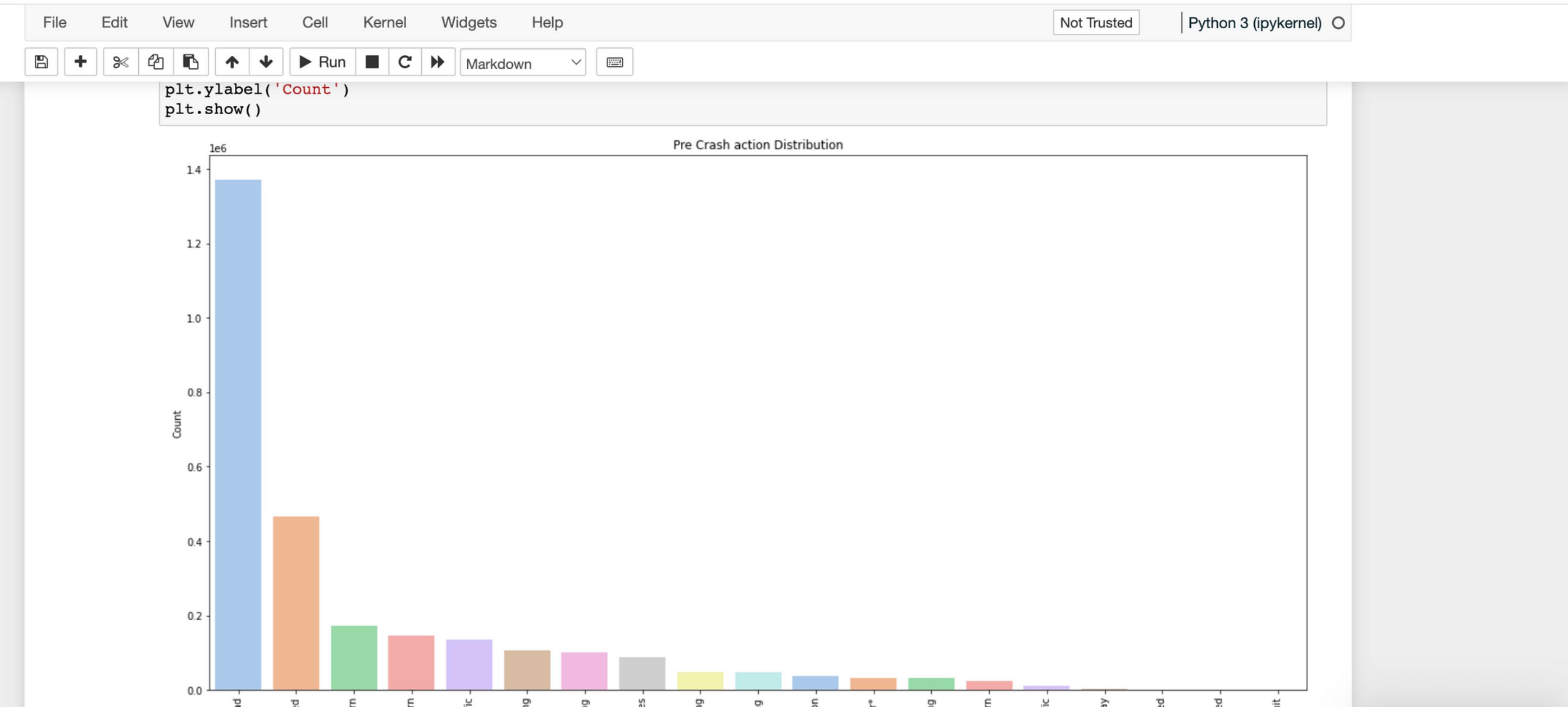
Road condition Distribution

Road Condition	Count
Center Front End	~300,000
Left Front Bumper	~215,000
Center Back End	~215,000
Right Front Bumper	~195,000
No Damage	~185,000
Front Quarter Panel	~145,000
Front Quarter Panel	~140,000
Left Side Doors	~115,000
Left Rear Bumper	~105,000
Right Side Doors	~95,000
Right Quarter Panel	~80,000
Right Rear Bumper	~75,000
Other	~40,000
Roof	~5,000
Demolished	~3,000
Trailer	~2,000
Overtured	~1,000
Undercarriage	~1,000

Exploratory Data Analysis



Exploratory Data Analysis



Machine Learning

Categorical columns :

'VEHICLE_TYPE','DRIVER_SEX','POINT_OF_IMPACT','MAKE','Month','Week','Hour'

Numerical columns : 'how_old'

x(independentvariables)

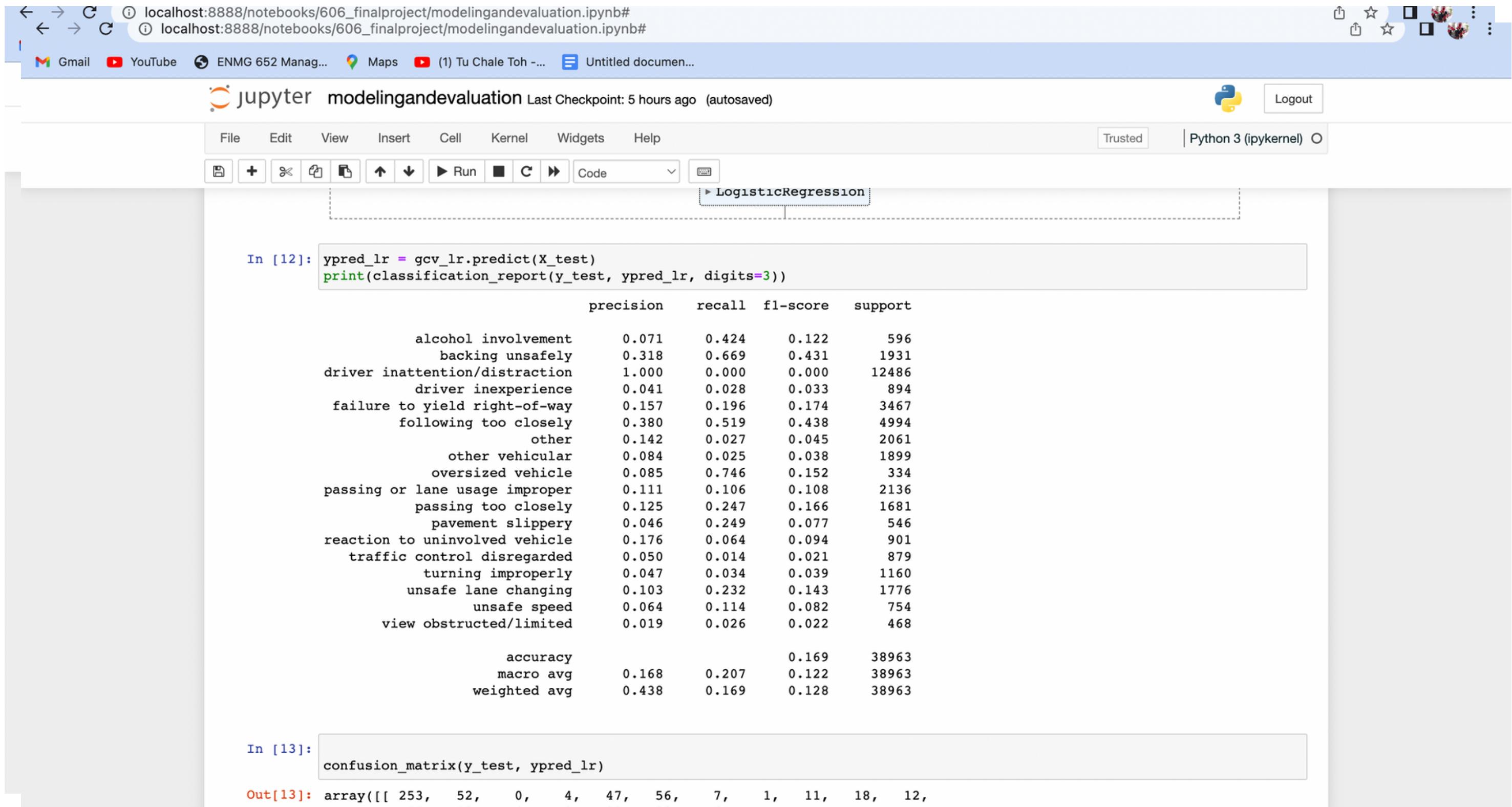
: 'VEHICLE_TYPE','DRIVER_SEX','POINT_OF_IMPACT','MAKE','Month','Week','Hour','how_old'

y(dependent variable) : 'CONTRIBUTING_FACTOR_1'

Models and Approach

- Logistic regression
- Decision tree
- Random forest
- Gradient Boosting Classifier
- Ada Boost Classifier

Logistic Regression



The screenshot shows a Jupyter Notebook interface running on a local host. The title bar indicates the URL is `localhost:8888/notebooks/606_finalproject/modelingandevaluation.ipynb#`. The notebook has two cells visible:

In [12]:

```
ypred_lr = gcv_lr.predict(X_test)
print(classification_report(y_test, ypred_lr, digits=3))
```

The output of this cell is a classification report table:

	precision	recall	f1-score	support
alcohol involvement	0.071	0.424	0.122	596
backing unsafely	0.318	0.669	0.431	1931
driver inattention/distraction	1.000	0.000	0.000	12486
driver inexperience	0.041	0.028	0.033	894
failure to yield right-of-way	0.157	0.196	0.174	3467
following too closely	0.380	0.519	0.438	4994
other	0.142	0.027	0.045	2061
other vehicular	0.084	0.025	0.038	1899
oversized vehicle	0.085	0.746	0.152	334
passing or lane usage improper	0.111	0.106	0.108	2136
passing too closely	0.125	0.247	0.166	1681
pavement slippery	0.046	0.249	0.077	546
reaction to uninvolved vehicle	0.176	0.064	0.094	901
traffic control disregarded	0.050	0.014	0.021	879
turning improperly	0.047	0.034	0.039	1160
unsafe lane changing	0.103	0.232	0.143	1776
unsafe speed	0.064	0.114	0.082	754
view obstructed/limited	0.019	0.026	0.022	468
accuracy			0.169	38963
macro avg	0.168	0.207	0.122	38963
weighted avg	0.438	0.169	0.128	38963

In [13]:

```
confusion_matrix(y_test, ypred_lr)
```

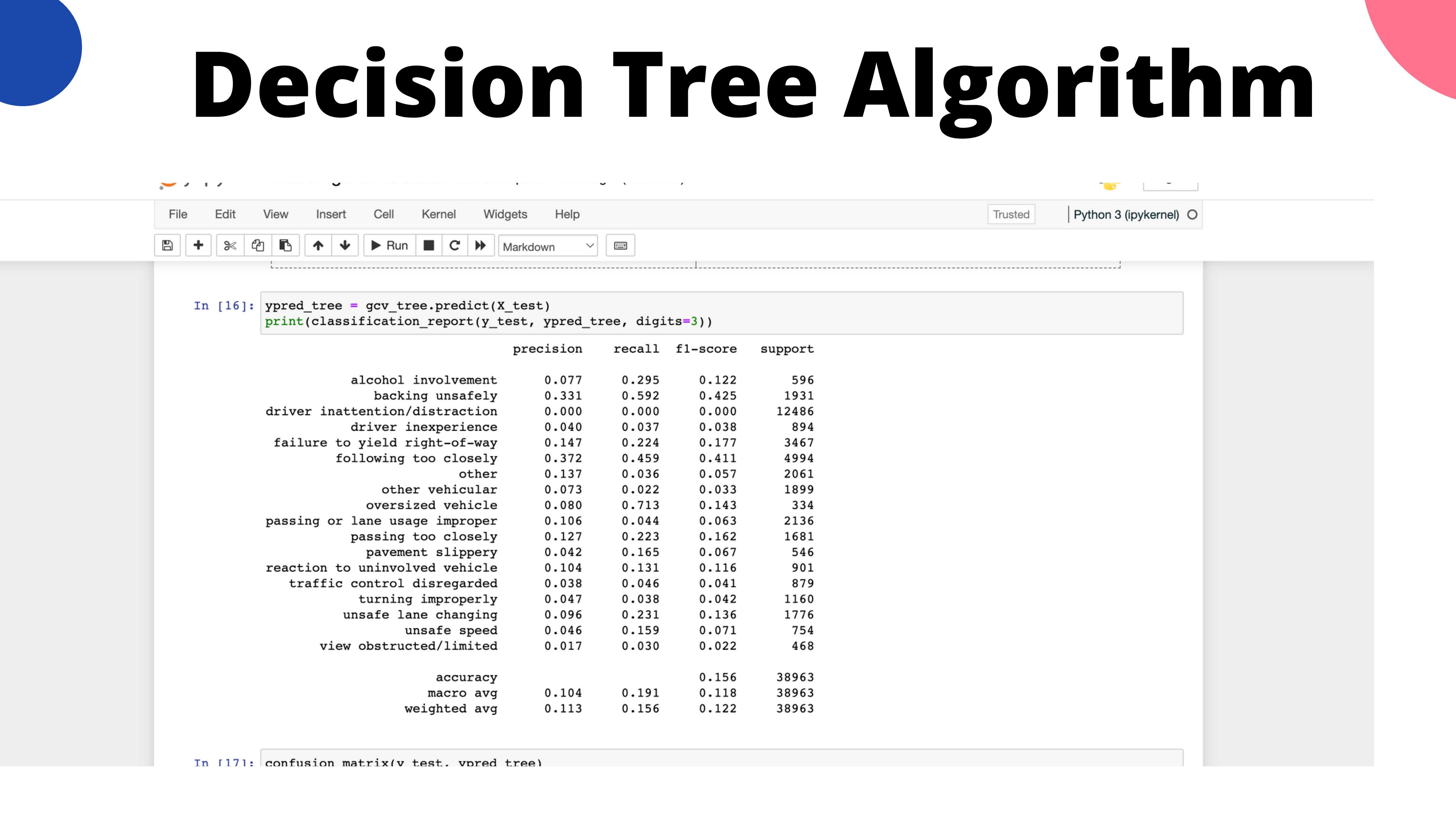
The output of this cell is:

```
Out[13]: array([[ 253,     52,      0,      4,     47,     56,     7,     1,    11,    18,    12,
```

Observations

- When we use a logistic regression to classify the contributing factor we get a macro average of 0.312
- When it comes to driver's inattention the model has a 60% precision.
- When it comes to failure to yield right-of-way the model has a 21% precision.
- When it comes to following too closely the model has a 42.5% precision.
- When it comes to improper lane usage the model has a 15.7% precision.
- For others the model has a 16.7% precision.

Decision Tree Algorithm



```
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel) O
```

In [16]:

```
ypred_tree = gcv_tree.predict(X_test)
print(classification_report(y_test, ypred_tree, digits=3))
```

	precision	recall	f1-score	support
alcohol involvement	0.077	0.295	0.122	596
backing unsafely	0.331	0.592	0.425	1931
driver inattention/distraction	0.000	0.000	0.000	12486
driver inexperience	0.040	0.037	0.038	894
failure to yield right-of-way	0.147	0.224	0.177	3467
following too closely	0.372	0.459	0.411	4994
other	0.137	0.036	0.057	2061
other vehicular	0.073	0.022	0.033	1899
oversized vehicle	0.080	0.713	0.143	334
passing or lane usage improper	0.106	0.044	0.063	2136
passing too closely	0.127	0.223	0.162	1681
pavement slippery	0.042	0.165	0.067	546
reaction to unininvolved vehicle	0.104	0.131	0.116	901
traffic control disregarded	0.038	0.046	0.041	879
turning improperly	0.047	0.038	0.042	1160
unsafe lane changing	0.096	0.231	0.136	1776
unsafe speed	0.046	0.159	0.071	754
view obstructed/limited	0.017	0.030	0.022	468
accuracy			0.156	38963
macro avg	0.104	0.191	0.118	38963
weighted avg	0.113	0.156	0.122	38963

In [17]:

```
confusion_matrix(y_test, ypred_tree)
```

Observations

- When we use a decision tree to classify the contributing factor we get a macro average of 0.295
- When it comes to driver's inattention the model has a 56% precision.
- When it comes to failure to yield right-of-way the model has a 21% precision.
- When it comes to following too closely the model has a 42% precision.
- When it comes to improper lane usage the model has a 15% precision.
- For others the model has a 13.8% precision.

Random Forest Algorithm

jupyter modelingandevaluation Last Checkpoint: 5 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

`RandomForestClassifier(class_weight='balanced_subsample', max_depth=15,
max_samples=1000)`

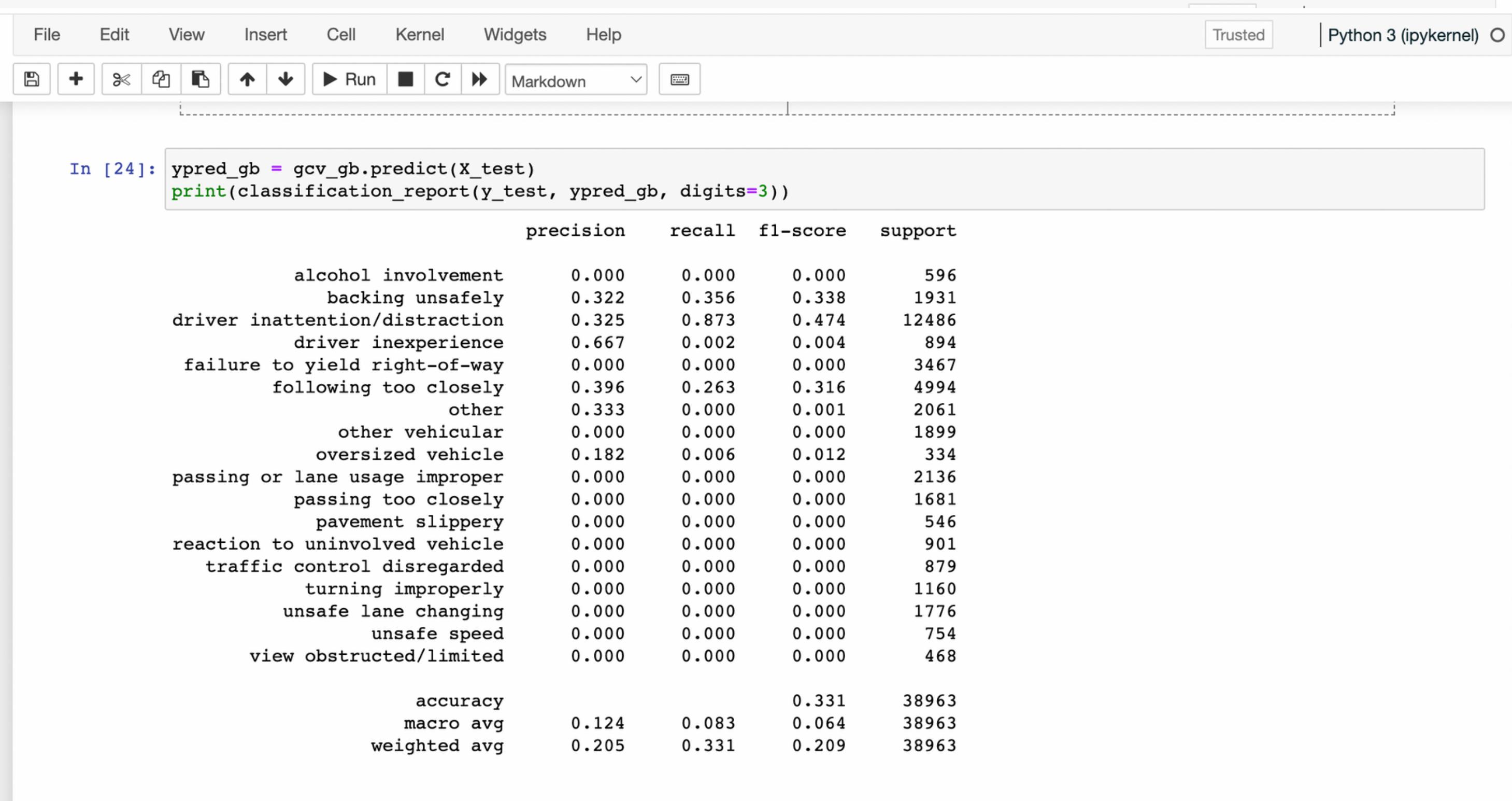
In [20]: `ypred_rf = gcv_rf.predict(X_test)
print(classification_report(y_test, ypred_rf, digits=3))`

	precision	recall	f1-score	support
alcohol involvement	0.086	0.012	0.021	596
backing unsafely	0.280	0.247	0.262	1931
driver inattention/distraction	0.323	0.670	0.436	12486
driver inexperience	0.000	0.000	0.000	894
failure to yield right-of-way	0.120	0.056	0.076	3467
following too closely	0.336	0.452	0.385	4994
other	0.097	0.015	0.026	2061
other vehicular	0.046	0.006	0.011	1899
oversized vehicle	0.112	0.063	0.080	334
passing or lane usage improper	0.094	0.028	0.043	2136
passing too closely	0.132	0.048	0.070	1681
pavement slippery	0.034	0.002	0.003	546
reaction to unininvolved vehicle	0.045	0.002	0.004	901
traffic control disregarded	0.023	0.001	0.002	879
turning improperly	0.096	0.008	0.014	1160
unsafe lane changing	0.089	0.030	0.045	1776
unsafe speed	0.020	0.003	0.005	754
view obstructed/limited	0.048	0.002	0.004	468
accuracy			0.297	38963
macro avg	0.110	0.091	0.083	38963
weighted avg	0.202	0.297	0.220	38963

Observations

- When we use a Random Forest to classify the contributing factor we get a macro average of 0.329
- When it comes to driver's inattention the model has a 54% precision.
- When it comes to failure to yield right-of-way the model has a 21% precision.
- When it comes to following too closely the model has a 42% precision.
- When it comes to improper lane usage the model has a 16.6% precision.
- For others the model has a 31% precision

Gradient Booster Algorithm



The image shows a Jupyter Notebook interface with a title bar and a code cell containing Python code and its output.

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [24]:

```
ypred_gb = gcv_gb.predict(X_test)
print(classification_report(y_test, ypred_gb, digits=3))
```

	precision	recall	f1-score	support
alcohol involvement	0.000	0.000	0.000	596
backing unsafely	0.322	0.356	0.338	1931
driver inattention/distraction	0.325	0.873	0.474	12486
driver inexperience	0.667	0.002	0.004	894
failure to yield right-of-way	0.000	0.000	0.000	3467
following too closely	0.396	0.263	0.316	4994
other	0.333	0.000	0.001	2061
other vehicular	0.000	0.000	0.000	1899
oversized vehicle	0.182	0.006	0.012	334
passing or lane usage improper	0.000	0.000	0.000	2136
passing too closely	0.000	0.000	0.000	1681
pavement slippery	0.000	0.000	0.000	546
reaction to unininvolved vehicle	0.000	0.000	0.000	901
traffic control disregarded	0.000	0.000	0.000	879
turning improperly	0.000	0.000	0.000	1160
unsafe lane changing	0.000	0.000	0.000	1776
unsafe speed	0.000	0.000	0.000	754
view obstructed/limited	0.000	0.000	0.000	468
accuracy			0.331	38963
macro avg	0.124	0.083	0.064	38963
weighted avg	0.205	0.331	0.209	38963

Observations

- When we use a gradient boosting to classify the contributing factor we get a macro average of 0.260
- When it comes to driver's inattention the model has a 51% precision.
- When it comes to failure to yield right-of-way the model has a 0% precision.
- When it comes to following too closely the model has a 45.4% precision.
- When it comes to improper lane usage the model has a 0% precision.
- For others the model has a 33.3% precision.

Ada Boost Classifier Algorithm

JUPYTER modelingandevaluation Last Checkpoint: 5 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

abc = AdaBoostClassifier(base_estimator=tree, n_estimators=500, learning_rate=0.1, random_state=1)
pipe4 = Pipeline([('data_processing', processing_pipeline), ('tree', abc)])
pipe4.fit(X_training, y_training)
abc_predicted = pipe4.predict(X_test)

In [29]: `print(classification_report(y_test, abc_predicted, digits=3))`

	precision	recall	f1-score	support
alcohol involvement	0.000	0.000	0.000	596
backing unsafely	0.319	0.068	0.112	1931
driver inattention/distraction	0.321	0.976	0.483	12486
driver inexperience	0.000	0.000	0.000	894
failure to yield right-of-way	0.000	0.000	0.000	3467
following too closely	0.361	0.042	0.075	4994
other	0.000	0.000	0.000	2061
other vehicular	0.000	0.000	0.000	1899
oversized vehicle	0.375	0.009	0.018	334
passing or lane usage improper	0.000	0.000	0.000	2136
passing too closely	0.000	0.000	0.000	1681
pavement slippery	0.000	0.000	0.000	546
reaction to unininvolved vehicle	0.000	0.000	0.000	901
traffic control disregarded	0.000	0.000	0.000	879
turning improperly	0.000	0.000	0.000	1160
unsafe lane changing	0.000	0.000	0.000	1776
unsafe speed	0.000	0.000	0.000	754
view obstructed/limited	0.000	0.000	0.000	468
accuracy			0.322	38963
macro avg	0.076	0.061	0.038	38963
weighted avg	0.168	0.322	0.170	38963

Observations

- When we use a Adaboosting classifier to classify the contributing factor we get a macro average of 0.192
- When it comes to driver's inattention the model has a 54% precision.
- When it comes to failure to yield right-of-way the model has a 0% precision.
- When it comes to following too closely the model has a 43.6% precision.
- When it comes to improper lane usage the model has a 0% precision.
- For others the model has a 0% precision.

Summary

I would use random forest or logistic regression as the final model. As both of these model identify the less frequently occurring classes more accurately compared to the complex algorithms like gradient boosting algortihm or adaboosting algorithm.

Future Scope

I would use the model that identifies the less frequently occurring classes more accurately compared to the complex algorithms. For the future of the project, I would like to identify various data sources that would help us look at a bigger picture of what the major contributing factors of the accident are and merge them with the existing dataset. Also, build machine learning models that could more accurately classify the contributing factor for an accident.