Introduction

Background

One of the major problems today is Road Safety. Road accidents can be serious or even fatal. The aim of this project is to build a model that predicts the severity of road accidents with attributes as light conditions and weather. The target audience of this project would be the local authorities for them to help and predict the area / address where road accidents may occur in the future. This could later on lead to applying new road safety measures for the specific areas.

Problem

A recent study shows that residential and shopping/commercial locations are more hazardous than village area. The frequency of casualities were high near the zones of residence because of higher exposure. Study also revealed that casuality rate among residential areas are classified as relatively deprived and significantly higher than those from relatively affluent areas. By analysing different factors which cause collision, it represents that car accidents are one of the common types of collision occuring everywhere globally every day.

Data Acquisition and Cleaning

Objectives of this Capstone project are described below :

1. Gather a comprehensive database of road accident statistics with parameters that affect road safety and ultimately accidents
2. Analyse data for the factors which can impact road accident rates (e.g. weather, road surface conditions, intersections, light conditions(visibility)

Data Source We will be using open source data which is published by Leeds city council. (Open Government License)

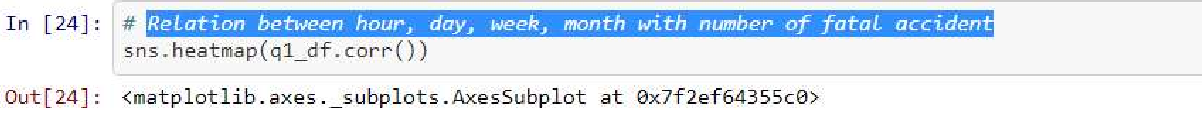
Link : https://datasetsearch.research.google.com/search?query=car%20accidents&docid=yKyJqNCmNypGNKe7AAAAAA%3D%3D and download accident dataset 2019 in CSV format.

Data downloaded from multiple sources (vehicle data, accident data & casualties data) in csv format and then combined into one table, Microsoft SQL Server. There were a lot of data type issues while uploading to DB, and also NULL and missing values for many rows. I wrote SQL script to load all data into DB and modify the column data types while transforming columns to correct values.

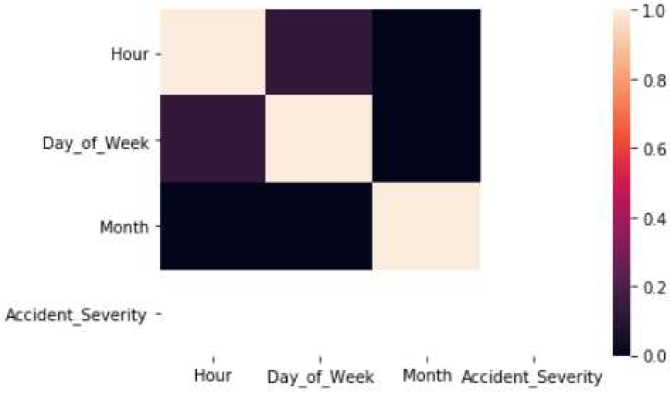
Data Analysis

Below steps are taken to do data analysis

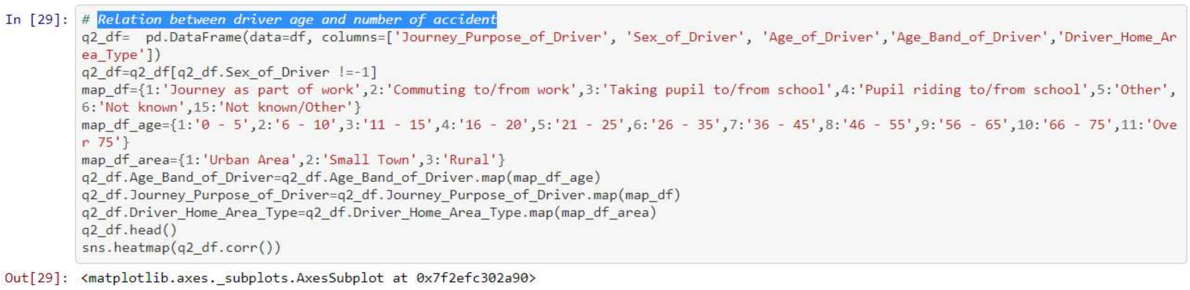
1. Created function to add month column
2. Created function to add hour column
3. Getting a datafame for Q1
4. Getting cases of 'Fatal Accidents' only for Q1
5. Relation between hour, day, week, month with number of fatal accident



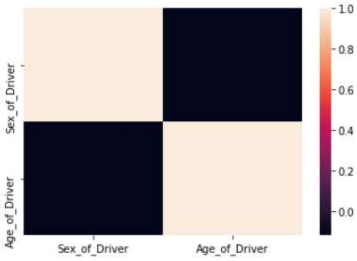
```
In [24]: # Relation between hour, day, week, month with number of fatal accident
         sns.heatmap(q1_df.corr())

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2ef64355c0>
```



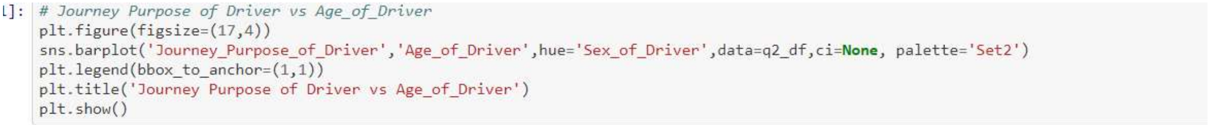
6. Relation between driver age and number of accidents.



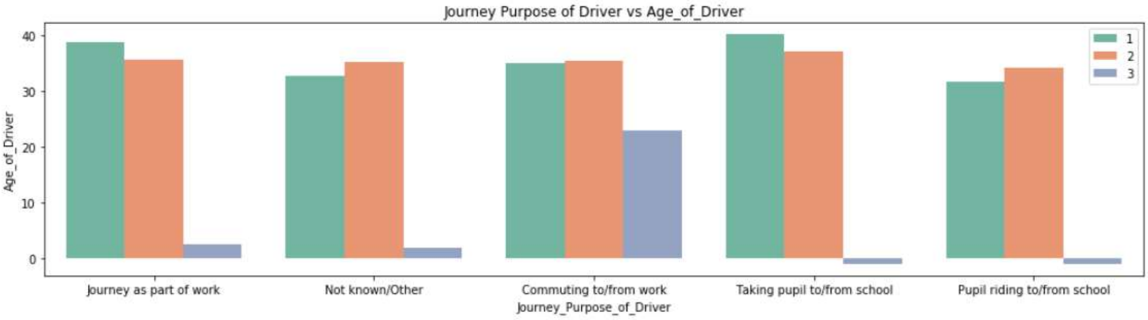
```
In [29]: # Relation between driver age and number of accident
         q2_df=  pd.DataFrame(data=df, columns=['Journey_Purpose_of_Driver', 'Sex_of_Driver', 'Age_of_Driver','Age_Band_of_Driver','Driver_Home_Ar
         ea_Type'])
         q2_df=q2_df[q2_df.Sex_of_Driver !=-1]
         map_df={1:'Journey as part of work',2:'Commuting to/from work',3:'Taking pupil to/from school',4:'Pupil riding to/from school',5:'Other',
         6:'Not known',15:'Not known/Other'}
         map_df_age={1:'0 - 5',2:'6 - 10',3:'11 - 15',4:'16 - 20',5:'21 - 25',6:'26 - 35',7:'36 - 45',8:'46 - 55',9:'56 - 65',10:'66 - 75',11:'Ove
         r 75'}
         map_df_area={1:'Urban Area',2:'Small Town',3:'Rural'}
         q2_df.Age_Band_of_Driver=q2_df.Age_Band_of_Driver.map(map_df_age)
         q2_df.Journey_Purpose_of_Driver=q2_df.Journey_Purpose_of_Driver.map(map_df)
         q2_df.Driver_Home_Area_Type=q2_df.Driver_Home_Area_Type.map(map_df_area)
         q2_df.head()
         sns.heatmap(q2_df.corr())

Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2efc302a90>
```



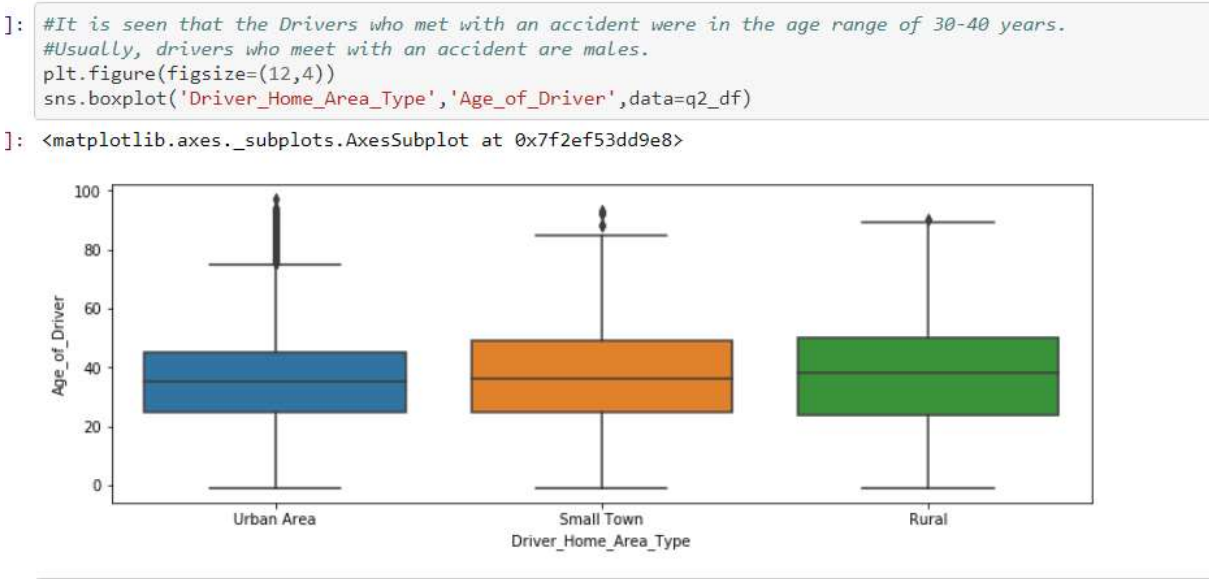
7. Journey purpose of driver vs age and sex



```
[ ]: # Journey Purpose of Driver vs Age_of_Driver
     plt.figure(figsize=(17,4))
     sns.barplot('Journey_Purpose_of_Driver','Age_of_Driver',hue='Sex_of_Driver',data=q2_df,ci=None, palette='Set2')
     plt.legend(bbox_to_anchor=(1,1))
     plt.title('Journey Purpose of Driver vs Age_of_Driver')
     plt.show()
```



8. It is observed that the drivers who met with an accident were in the age range of 30-40 years. Usually drivers who met with and accident are males.

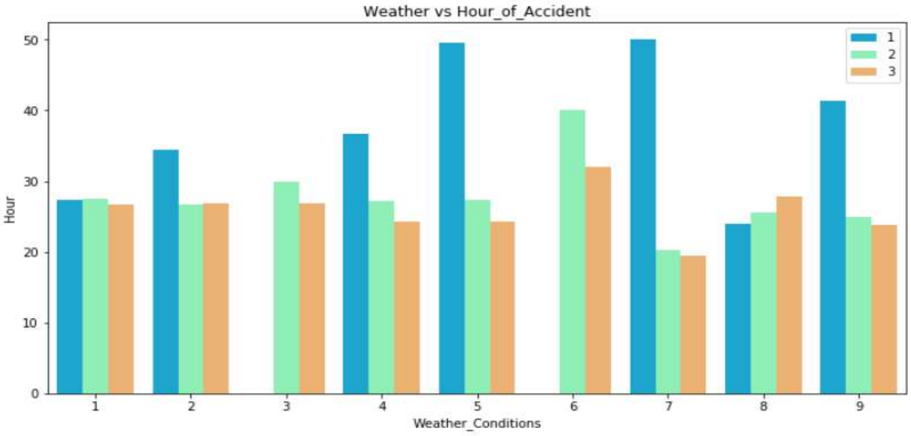


```
 ]: #It is seen that the Drivers who met with an accident were in the age range of 30-40 years.
    #Usually, drivers who meet with an accident are males.
    plt.figure(figsize=(12,4))
    sns.boxplot('Driver_Home_Area_Type','Age_of_Driver',data=q2_df)
```

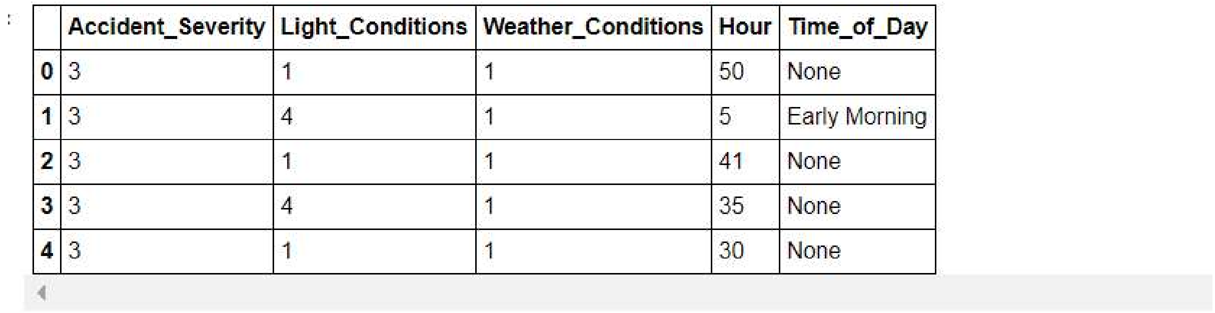
```
 ]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2ef53dd9e8>
```

9. Weather vs Hour of Accident

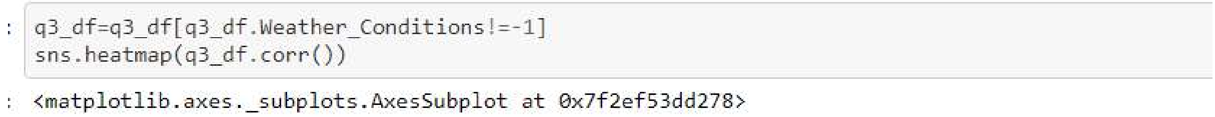
Accidents usually take place in the afternoon: refer fig: Weather vs Hour_of_Accident. Accidents with Slight severity occurred the most. Accidents usually took place when the Weather conditions were fine and also there were not any high winds, meaning which the weather conditions didn’t not effectively contribute to occurrences of accidents



10. weather impact the number or severity of an accident

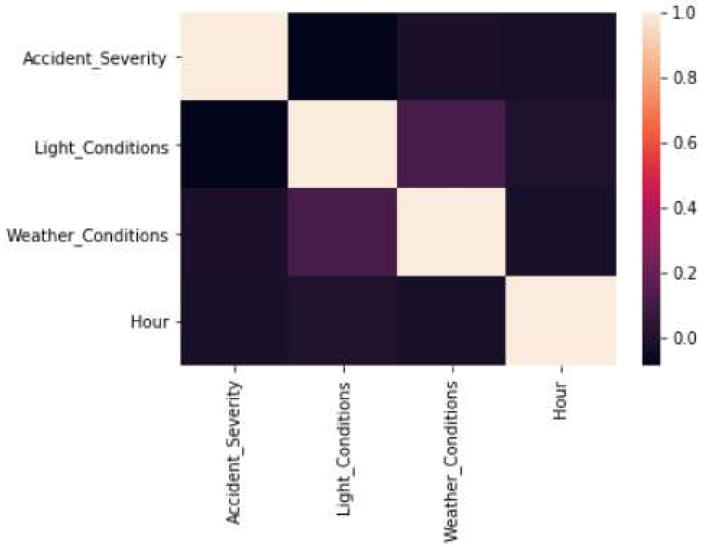


	Accident_Severity	Light_Conditions	Weather_Conditions	Hour	Time_of_Day
0	3	1	1	50	None
1	3	4	1	5	Early Morning
2	3	1	1	41	None
3	3	4	1	35	None
4	3	1	1	30	None

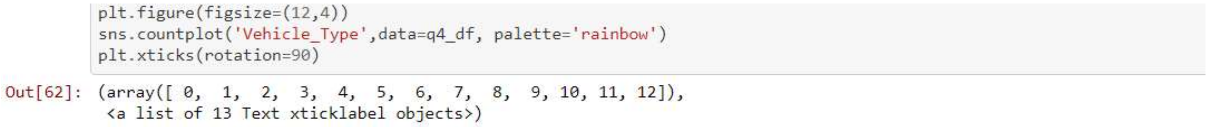


```
q3_df=q3_df[q3_df.Weather_Conditions!=-1]
sns.heatmap(q3_df.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2ef53dd278>
```

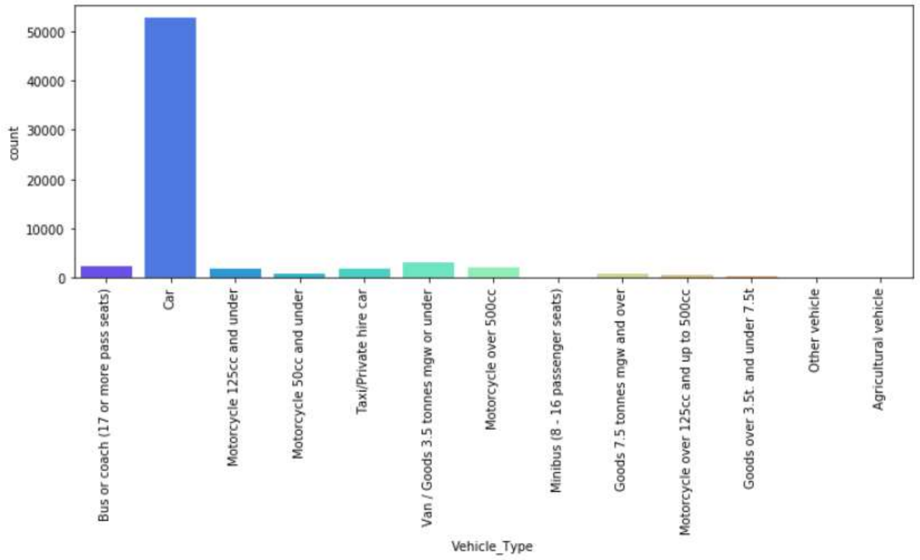


11. Are certain vehicle types are safer than others ?



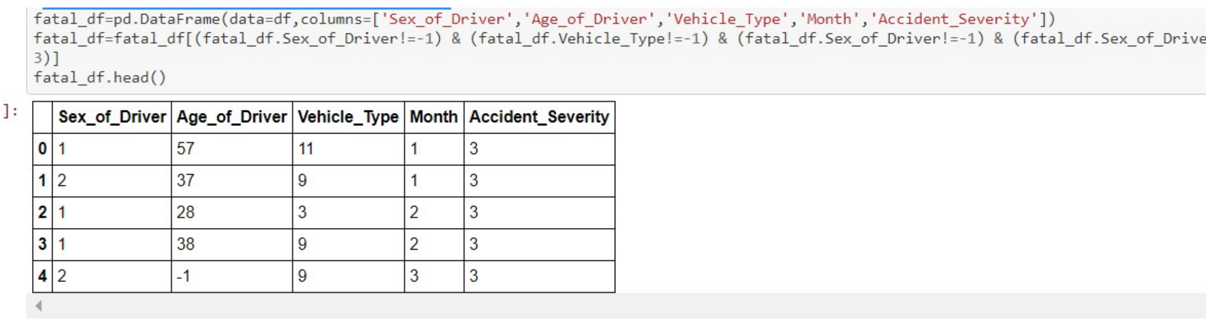
```
plt.figure(figsize=(12,4))
sns.countplot('Vehicle_Type',data=q4_df, palette='rainbow')
plt.xticks(rotation=90)
```

```
Out[62]: (array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12]),
          <a list of 13 Text xticklabel objects>)
```



Predictive Data Modelling

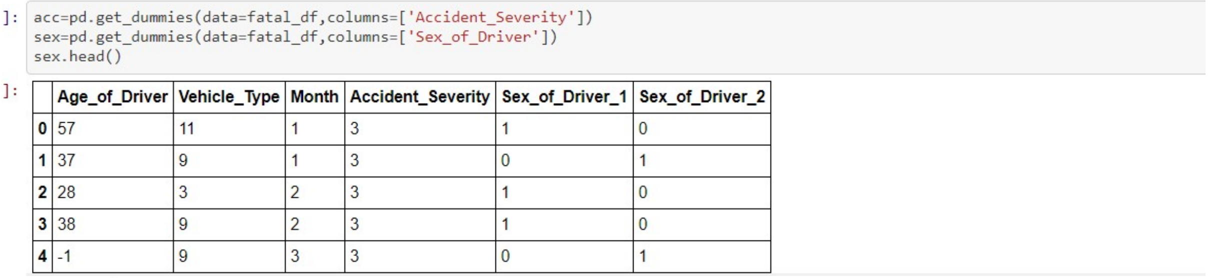
let's begin with prediction of fatal accidents, where please do note that Accident_Severity_1 corresponds to fatal accident and Sex_of_Driver_1 corresponds to male driver.



```
fatal_df=pd.DataFrame(data=df,columns=['Sex_of_Driver','Age_of_Driver','Vehicle_Type','Month','Accident_Severity'])
fatal_df=fatal_df[(fatal_df.Sex_of_Driver!=-1) & (fatal_df.Vehicle_Type!=-1) & (fatal_df.Sex_of_Driver!=-1) & (fatal_df.Sex_of_Drive
3)]
fatal_df.head()
```

]:

	Sex_of_Driver	Age_of_Driver	Vehicle_Type	Month	Accident_Severity
0	1	57	11	1	3
1	2	37	9	1	3
2	1	28	3	2	3
3	1	38	9	2	3
4	2	-1	9	3	3

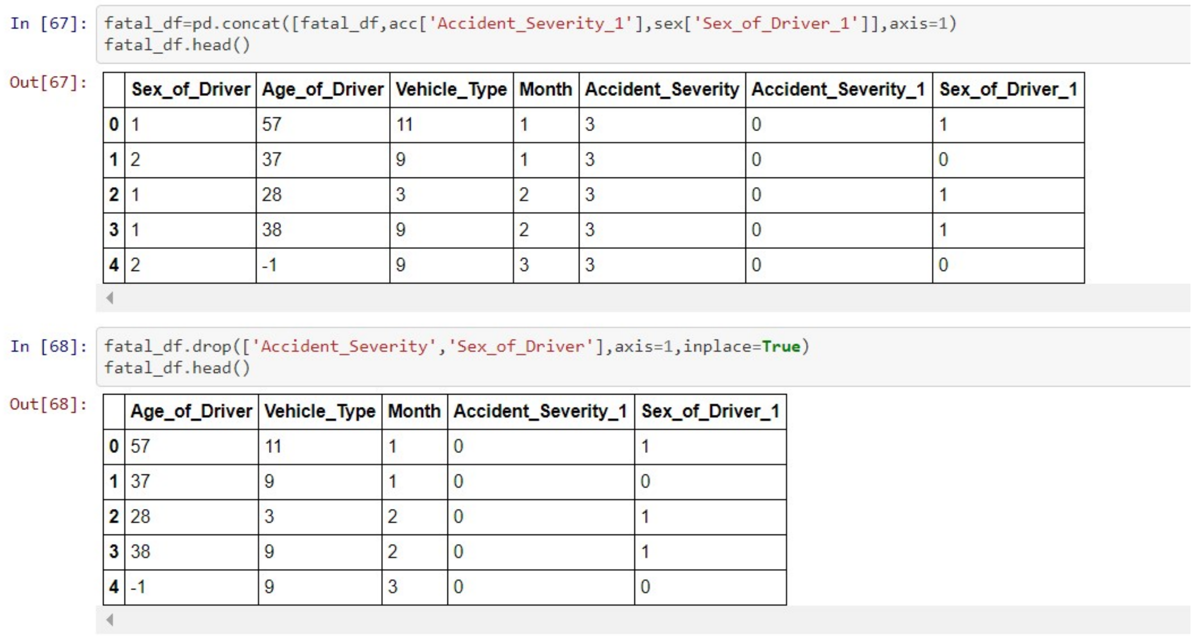


]:

```
acc=pd.get_dummies(data=fatal_df,columns=['Accident_Severity'])
sex=pd.get_dummies(data=fatal_df,columns=['Sex_of_Driver'])
sex.head()
```

]:

	Age_of_Driver	Vehicle_Type	Month	Accident_Severity	Sex_of_Driver_1	Sex_of_Driver_2
0	57	11	1	3	1	0
1	37	9	1	3	0	1
2	28	3	2	3	1	0
3	38	9	2	3	1	0
4	-1	9	3	3	0	1



In [67]:

```
fatal_df=pd.concat([fatal_df,acc['Accident_Severity_1'],sex['Sex_of_Driver_1']],axis=1)
fatal_df.head()
```

Out[67]:

	Sex_of_Driver	Age_of_Driver	Vehicle_Type	Month	Accident_Severity	Accident_Severity_1	Sex_of_Driver_1
0	1	57	11	1	3	0	1
1	2	37	9	1	3	0	0
2	1	28	3	2	3	0	1
3	1	38	9	2	3	0	1
4	2	-1	9	3	3	0	0

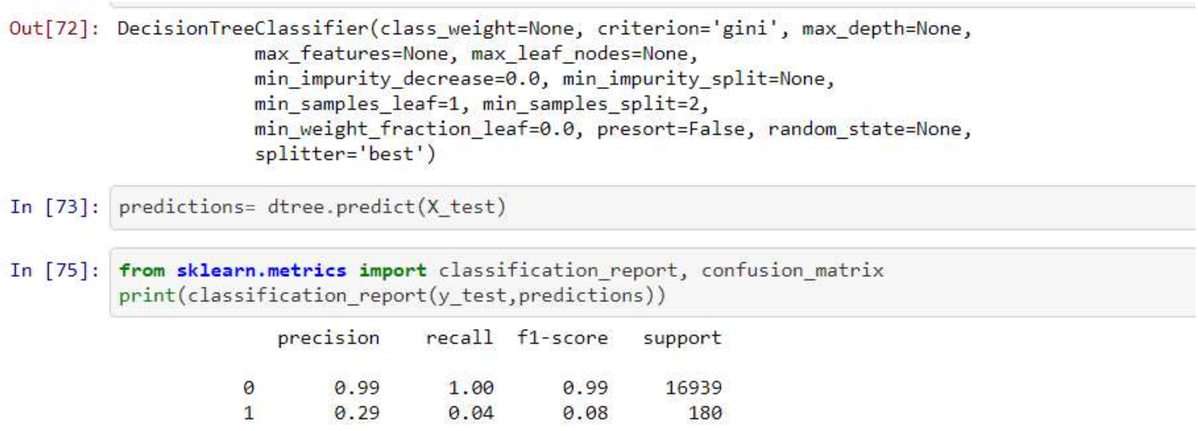
In [68]:

```
fatal_df.drop(['Accident_Severity','Sex_of_Driver'],axis=1,inplace=True)
fatal_df.head()
```

Out[68]:

	Age_of_Driver	Vehicle_Type	Month	Accident_Severity_1	Sex_of_Driver_1
0	57	11	1	0	1
1	37	9	1	0	0
2	28	3	2	0	1
3	38	9	2	0	1
4	-1	9	3	0	0

Using Decision Tree



Out[72]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [73]:

```
predictions= dtree.predict(X_test)
```

In [75]:

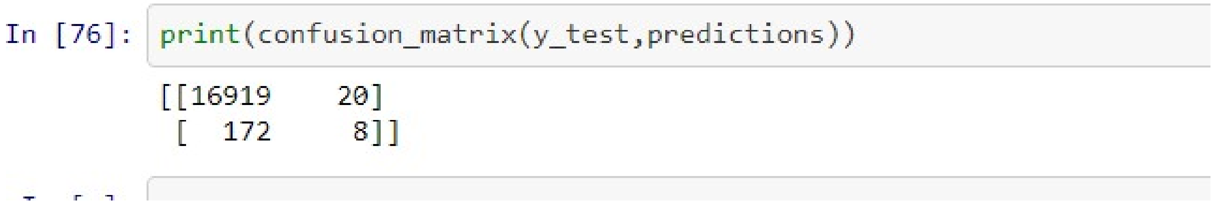
```
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_test,predictions))
```

```
             precision    recall  f1-score   support

          0       0.99      1.00      0.99     16939
          1       0.29      0.04      0.08       180
```

It seems like the model didn't do well

Though the precision is good, it is noticed that the model had better predictions for only case:0

Also, checking the recall, it is noticed that case:1 is neglected



```
In [76]: print(confusion_matrix(y_test,predictions))

[[16919    20]
 [  172     8]]
```

Hence, better data engineering is required to obtain predictions…