```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
df=pd.read_csv("/content/RTA Dataset.csv")
df
```

	Time	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level
0	17:02:00	Monday	18-30	Male	Above high school
1	17:02:00	Monday	31-50	Male	Junior high school
2	17:02:00	Monday	18-30	Male	Junior high school
3	1:06:00	Sunday	18-30	Male	Junior high school
4	1:06:00	Sunday	18-30	Male	Junior high school
...	...	...	...	...	...
12311	16:15:00	Wednesday	31-50	Male	NaN
12312	18:00:00	Sunday	Unknown	Male	Elementary school
12313	13:55:00	Sunday	Over 51	Male	Junior high school
12314	13:55:00	Sunday	18-30	Female	Junior high school
12315	13:55:00	Sunday	18-30	Male	Junior high school

12316 rows × 32 columns

```
df.columns
```

```
Index(['Time', 'Day_of_week', 'Age_band_of_driver', 'Sex_of_driver',
       'Educational_level', 'Vehicle_driver_relation', 'Driving_experience',
       'Type_of_vehicle', 'Owner_of_vehicle', 'Service_year_of_vehicle',
       'Defect_of_vehicle', 'Area_accident_occured', 'Lanes_or_Medians',
       'Road_allignment', 'Types_of_Junction', 'Road_surface_type',
       'Road_surface_conditions', 'Light_conditions', 'Weather_conditions',
       'Type_of_collision', 'Number_of_vehicles_involved',
       'Number_of_casualties', 'Vehicle_movement', 'Casualty_class',
       'Sex_of_casualty', 'Age_band_of_casualty', 'Casualty_severity',
       'Work_of_casuality', 'Fitness_of_casuality', 'Pedestrian_movement',
       'Cause_of_accident', 'Accident_severity'],
      dtype='object')
```

```
df.dtypes
```

```
Time                           object
Day_of_week                    object
Age_band_of_driver             object
Sex_of_driver                  object
Educational_level              object
Vehicle_driver_relation        object
Driving_experience             object
Type_of_vehicle                object
Owner_of_vehicle               object
Service_year_of_vehicle        object
Defect_of_vehicle              object
Area_accident_occured          object
Lanes_or_Medians               object
Road_allignment                object
Types_of_Junction              object
```

```
     Road_surface_type              object
     Road_surface_conditions        object
     Light_conditions               object
     Weather_conditions             object
     Type_of_collision              object
     Number_of_vehicles_involved     int64
     Number_of_casualties            int64
     Vehicle_movement               object
     Casualty_class                 object
     Sex_of_casualty                object
     Age_band_of_casualty           object
     Casualty_severity              object
     Work_of_casuality              object
     Fitness_of_casuality           object
     Pedestrian_movement            object
     Cause_of_accident              object
     Accident_severity              object
     dtype: object
```

```
df.isna().sum()
```

```
     Time                              0
     Day_of_week                       0
     Age_band_of_driver                0
     Sex_of_driver                     0
     Educational_level               741
     Vehicle_driver_relation         579
     Driving_experience              829
     Type_of_vehicle                 950
     Owner_of_vehicle                482
     Service_year_of_vehicle        3928
     Defect_of_vehicle              4427
     Area_accident_occured           239
     Lanes_or_Medians                385
     Road_allignment                 142
     Types_of_Junction               887
     Road_surface_type               172
     Road_surface_conditions           0
     Light_conditions                  0
     Weather_conditions                0
     Type_of_collision               155
     Number_of_vehicles_involved       0
     Number_of_casualties              0
     Vehicle_movement                308
     Casualty_class                    0
     Sex_of_casualty                   0
     Age_band_of_casualty              0
     Casualty_severity                 0
     Work_of_casuality              3198
     Fitness_of_casuality           2635
     Pedestrian_movement               0
     Cause_of_accident                 0
     Accident_severity                 0
     dtype: int64
```

```
df=df.drop(['Service_year_of_vehicle','Defect_of_vehicle','Work_of_casuality','Fitness_of_casuality','Time'],axis=1)
```

```
df
```

	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_driver_relation	Driving_experie
0	Monday	18-30	Male	Above high school	Employee	1-
1	Monday	31-50	Male	Junior high school	Employee	Above 1
2	Monday	18-30	Male	Junior high school	Employee	1-
3	Sunday	18-30	Male	Junior high school	Employee	5-1
4	Sunday	18-30	Male	Junior high school	Employee	2-
...	...	...	...	...	...	
12311	Wednesday	31-50	Male	NaN	Employee	2-

```
df.isna().sum()
```

```
Day_of_week                     0
Age_band_of_driver              0
Sex_of_driver                   0
Educational_level             741
Vehicle_driver_relation       579
Driving_experience            829
Type_of_vehicle               950
Owner_of_vehicle              482
Area_accident_occured         239
Lanes_or_Medians              385
Road_allignment               142
Types_of_Junction             887
Road_surface_type             172
Road_surface_conditions         0
Light_conditions                0
Weather_conditions              0
Type_of_collision             155
Number_of_vehicles_involved     0
Number_of_casualties            0
Vehicle_movement              308
Casualty_class                  0
Sex_of_casualty                 0
Age_band_of_casualty            0
Casualty_severity               0
Pedestrian_movement             0
Cause_of_accident               0
Accident_severity               0
dtype: int64
```

```
df.info()
```

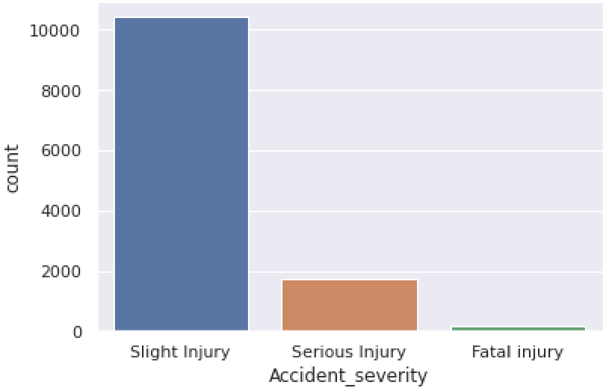
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 27 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Day_of_week                  12316 non-null  object
 1   Age_band_of_driver           12316 non-null  object
 2   Sex_of_driver                12316 non-null  object
 3   Educational_level            11575 non-null  object
 4   Vehicle_driver_relation      11737 non-null  object
 5   Driving_experience           11487 non-null  object
 6   Type_of_vehicle              11366 non-null  object
 7   Owner_of_vehicle             11834 non-null  object
 8   Area_accident_occured        12077 non-null  object
 9   Lanes_or_Medians             11931 non-null  object
 10  Road_allignment              12174 non-null  object
 11  Types_of_Junction            11429 non-null  object
 12  Road_surface_type            12144 non-null  object
 13  Road_surface_conditions      12316 non-null  object
 14  Light_conditions             12316 non-null  object
 15  Weather_conditions           12316 non-null  object
 16  Type_of_collision            12161 non-null  object
 17  Number_of_vehicles_involved  12316 non-null  int64 
 18  Number_of_casualties         12316 non-null  int64 
 19  Vehicle_movement             12008 non-null  object
 20  Casualty_class               12316 non-null  object
 21  Sex_of_casualty              12316 non-null  object
 22  Age_band_of_casualty         12316 non-null  object
 23  Casualty_severity            12316 non-null  object
 24  Pedestrian_movement          12316 non-null  object
 25  Cause_of_accident            12316 non-null  object
 26  Accident_severity            12316 non-null  object
dtypes: int64(2), object(25)
memory usage: 2.5+ MB
```

```
df['Accident_severity'].value_counts()
```

```
Slight Injury     10415
Serious Injury     1743
Fatal injury        158
Name: Accident_severity, dtype: int64
```

```
sns.countplot('Accident_severity',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd1011a880>
```



```
categorical=[i for i in df.columns if df[i].dtype=='O']
print('The categorical variables are ',categorical)
for i in categorical:
  df[i].fillna(df[i].mode()[0],inplace=True)
```

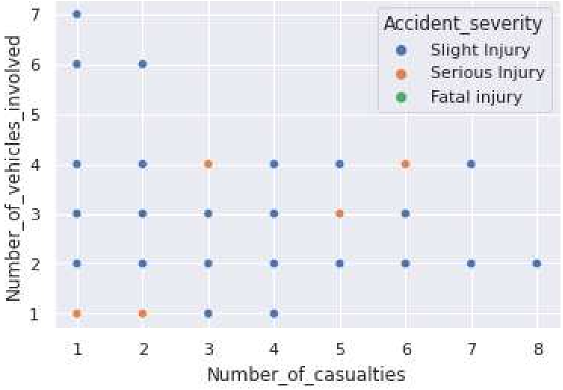
```
The categorical variables are  ['Day_of_week', 'Age_band_of_driver', 'Sex_of_driver', 'Educational_level', 'Vehicl
```

```
df.isna().sum()
```

```
Day_of_week                 0
Age_band_of_driver          0
Sex_of_driver               0
Educational_level           0
Vehicle_driver_relation     0
Driving_experience          0
Type_of_vehicle             0
Owner_of_vehicle            0
Area_accident_occured       0
Lanes_or_Medians            0
Road_allignment             0
Types_of_Junction           0
Road_surface_type           0
Road_surface_conditions     0
Light_conditions            0
Weather_conditions          0
Type_of_collision           0
Number_of_vehicles_involved 0
Number_of_casualties        0
Vehicle_movement            0
Casualty_class              0
Sex_of_casualty             0
Age_band_of_casualty        0
Casualty_severity           0
Pedestrian_movement         0
Cause_of_accident           0
Accident_severity           0
dtype: int64
```

```
#plotting relationship between no of casualities and no of vehciles involved
sns.scatterplot(x=df['Number_of_casualties'],y=df['Number_of_vehicles_involved'],hue=df['Accident_severity'])
```

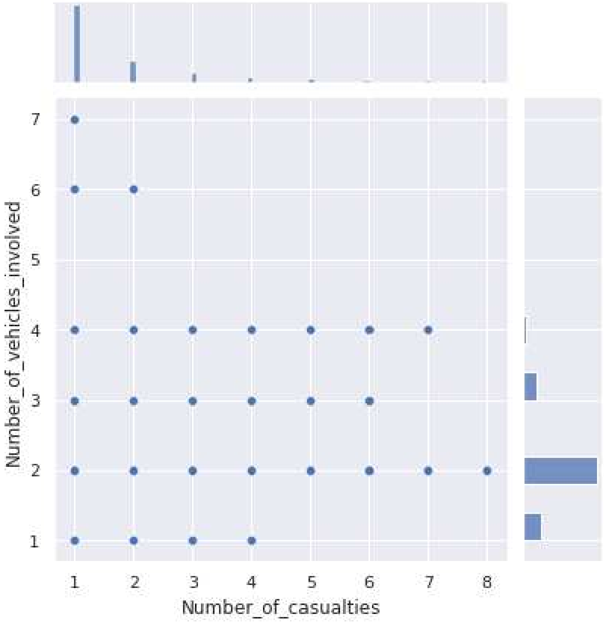
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd100c9e20>
```



Joint plot

```
sns.jointplot(x='Number_of_casualties',y='Number_of_vehicles_involved',data=df)
```

```
<seaborn.axisgrid.JointGrid at 0x7fcd1003ac70>
```

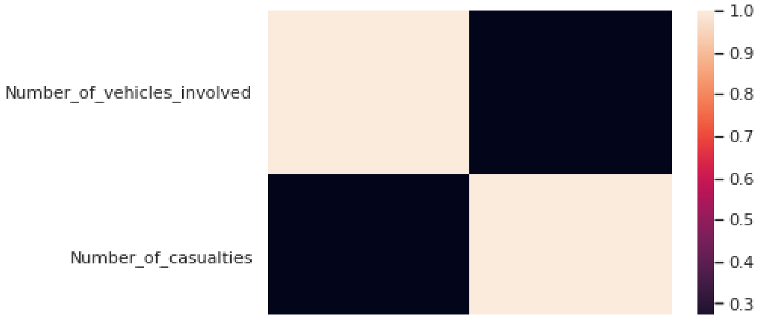


```
#checking the correlation between numerical columns
df.corr()
```

	Number_of_vehicles_involved	Number_of_casualties
Number_of_vehicles_involved	1.000000	0.213427
Number_of_casualties	0.213427	1.000000

```
#plotting correlation using heat map
sns.heatmap(df.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd10106d60>
```



```
#storing numerical value column to a new variable
numerical=[i for i in df.columns if df[i].dtypes!='O']
print(numerical)
```

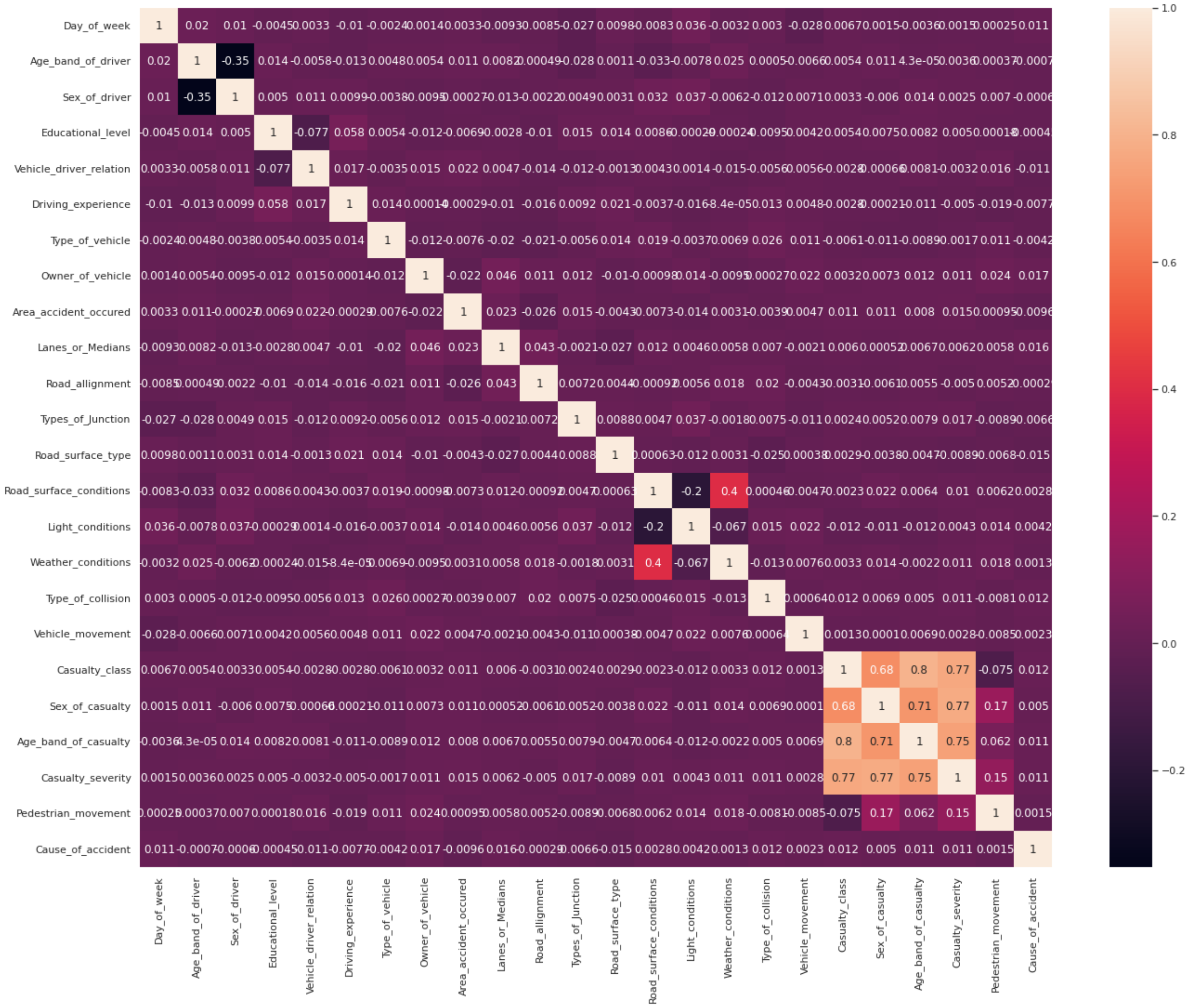
```
['Number_of_vehicles_involved', 'Number_of_casualties']
```

```
#importing label encoding
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()


#creating a new dataframe
df1=pd.DataFrame()


#adding all categorical columns
for i in categorical:
  if i!='Accident_severity':
    df1[i]=le.fit_transform(df[i])


df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 24 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Day_of_week              12316 non-null  int64
 1   Age_band_of_driver       12316 non-null  int64
 2   Sex_of_driver            12316 non-null  int64
 3   Educational_level        12316 non-null  int64
 4   Vehicle_driver_relation  12316 non-null  int64
 5   Driving_experience       12316 non-null  int64
 6   Type_of_vehicle          12316 non-null  int64
 7   Owner_of_vehicle         12316 non-null  int64
 8   Area_accident_occured    12316 non-null  int64
 9   Lanes_or_Medians         12316 non-null  int64
 10  Road_allignment          12316 non-null  int64
 11  Types_of_Junction        12316 non-null  int64
 12  Road_surface_type        12316 non-null  int64
 13  Road_surface_conditions  12316 non-null  int64
 14  Light_conditions         12316 non-null  int64
 15  Weather_conditions       12316 non-null  int64
 16  Type_of_collision        12316 non-null  int64
 17  Vehicle_movement         12316 non-null  int64
 18  Casualty_class           12316 non-null  int64
 19  Sex_of_casualty          12316 non-null  int64
 20  Age_band_of_casualty     12316 non-null  int64
 21  Casualty_severity        12316 non-null  int64
 22  Pedestrian_movement      12316 non-null  int64
 23  Cause_of_accident        12316 non-null  int64
dtypes: int64(24)
memory usage: 2.3 MB
```

```
plt.figure(figsize=(22,17))
sns.set(font_scale=1)
sns.heatmap(df1.corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd13adf2b0>
```



import chi2 test

```
from pandas.core.internals.blocks import F
from sklearn.feature_selection import chi2
f_p_values=chi2(df1,df['Accident_severity'])
```

Double-click (or enter) to edit

Form a new df to evaluate f_p_scores for feature selection

```
f_p_values=pd.DataFrame({'features':df1.columns,'Fscore':f_p_values[0],'pvalues':f_p_values[1]})
f_p_values
```

	features	Fscore	pvalues
0	Day_of_week	0.158221	0.923938
1	Age_band_of_driver	8.915392	0.011589
2	Sex_of_driver	0.143189	0.930908
3	Educational_level	0.174585	0.916409
4	Vehicle_driver_relation	5.345345	0.069067
5	Driving_experience	4.499679	0.105416
6	Type_of_vehicle	1.077671	0.583427
7	Owner_of_vehicle	1.104262	0.575722
8	Area_accident_occured	3.616540	0.163937
9	Lanes_or_Medians	3.281615	0.193824
10	Road_allignment	0.131931	0.936163
11	Types_of_Junction	3.086487	0.213687
12	Road_surface_type	6.994806	0.030276
13	Road_surface_conditions	0.615103	0.735245
14	Light_conditions	16.082824	0.000322
15	Weather_conditions	1.149345	0.562889

Sort By Ascending Order

17	Vehicle_movement	2.200712	0.332753

```
#select features with high fscores and low pvalues
#so sort it by pvalues in asc order
f_p_values.sort_values(by='pvalues',ascending=True)
```

	features	Fscore	pvalues
14	Light_conditions	16.082824	0.000322
20	Age_band_of_casualty	13.778413	0.001019
1	Age_band_of_driver	8.915392	0.011589

Drop the column having less Pvalue

```
df2=df.drop(['Owner_of_vehicle','Type_of_vehicle','Road_surface_conditions','Pedestrian_movement','Casualty_severity','E
           'Day_of_week','Sex_of_driver','Road_allignment','Sex_of_casualty'],axis=1)
df2
```

	Age_band_of_driver	Vehicle_driver_relation	Driving_experience	Area_accident_occured	Lanes_or_Medians
0	18-30	Employee	1-2yr	Residential areas	Two-way (divided with broken lines road marking)
1	31-50	Employee	Above 10yr	Office areas	Undivided Two way
2	18-30	Employee	1-2yr	Recreational areas	other
3	18-30	Employee	5-10yr	Office areas	other
4	18-30	Employee	2-5yr	Industrial areas	other
...	...	...	...	...	...
12311	31-50	Employee	2-5yr	Outside rural areas	Undivided Two way
12312	Unknown	Employee	5-10yr	Outside rural areas	Two-way (divided with broken lines road marking)
12313	Over 51	Employee	5-10yr	Outside rural areas	Two-way (divided with broken lines road marking)
12314	18-30	Employee	Above 10yr	Office areas	Undivided Two way
12315	18-30	Employee	5-10yr	Outside rural areas	Undivided Two way

12316 rows × 17 columns

Now store categorical column to a new variable

```
categorical1=[i for i in df2.columns if df2[i].dtype=='O']
print(categorical1)
```

```
['Age_band_of_driver', 'Vehicle_driver_relation', 'Driving_experience', 'Area_accident_occured', 'Lanes_or_Medians
```

Converting the categorical features into integers by get dummies

```
dummy=pd.get_dummies(df[['Age_band_of_driver','Vehicle_driver_relation','Driving_experience','Area_accident_occured',\
                         'Lanes_or_Medians','Types_of_Junction','Road_surface_type','Light_conditions','Weather_conditio
                         'Type_of_collision','Vehicle_movement','Casualty_class','Age_band_of_casualty','Cause_of_accide
dummy
```

	Age_band_of_driver_31-50	Age_band_of_driver_Over 51	Age_band_of_driver_Under 18	Age_band_of_driver_Unknown	Vehi
0	0	0	0	0	
1	1	0	0	0	
2	0	0	0	0	
3	0	0	0	0	
4	0	0	0	0	
...	...	...	...	...	
12311	1	0	0	0	
12312	0	0	0	1	
12313	0	1	0	0	

Concatinate

```
dfe=pd.concat([df2,dummy],axis=1)
dfe
```

	Age_band_of_driver	Vehicle_driver_relation	Driving_experience	Area_accident_occured	Lanes_or_Medians	Ty
0	18-30	Employee	1-2yr	Residential areas	Two-way (divided with broken lines road marking)	
1	31-50	Employee	Above 10yr	Office areas	Undivided Two way	
2	18-30	Employee	1-2yr	Recreational areas	other	
3	18-30	Employee	5-10yr	Office areas	other	
4	18-30	Employee	2-5yr	Industrial areas	other	
...	...	...	...	...	...	
12311	31-50	Employee	2-5yr	Outside rural areas	Undivided Two way	
12312	Unknown	Employee	5-10yr	Outside rural areas	Two-way (divided with broken lines road marking)	
12313	Over 51	Employee	5-10yr	Outside rural areas	Two-way (divided with broken lines road marking)	
12314	18-30	Employee	Above 10yr	Office areas	Undivided Two way	
12315	18-30	Employee	5-10yr	Outside rural areas	Undivided Two way	

12316 rows × 119 columns

```
df2=dfe.drop(['Age_band_of_driver','Vehicle_driver_relation','Driving_experience','Area_accident_occured',\
            'Lanes_or_Medians','Types_of_Junction','Road_surface_type','Light_conditions','Weather_conditions','Type_o
            'Vehicle_movement','Casualty_class','Age_band_of_casualty','Cause_of_accident'],axis=1)
df2
```

	Number_of_vehicles_involved	Number_of_casualties	Accident_severity	Age_band_of_driver_31-50	Age_band_of_d
0	2	2	Slight Injury	0	
1	2	2	Slight Injury	1	
2	2	2	Serious Injury	0	
3	2	2	Slight Injury	0	
4	2	2	Slight Injury	0	
...	...	...	...	...	
12311	2	1	Slight Injury	1	
12312	2	1	Slight Injury	0	
12313	1	1	Serious Injury	0	
12314	2	1	Slight Injury	0	

seprate X and Y

```
x=df2.drop(['Accident_severity'],axis=1)
x
```

	Number_of_vehicles_involved	Number_of_casualties	Age_band_of_driver_31-50	Age_band_of_driver_Over 51	Age_ban
0	2	2	0	0	
1	2	2	1	0	
2	2	2	0	0	
3	2	2	0	0	
4	2	2	0	0	
...	...	...	...	...	
12311	2	1	1	0	
12312	2	1	0	0	
12313	1	1	0	1	
12314	2	1	0	0	
12315	2	1	0	0	

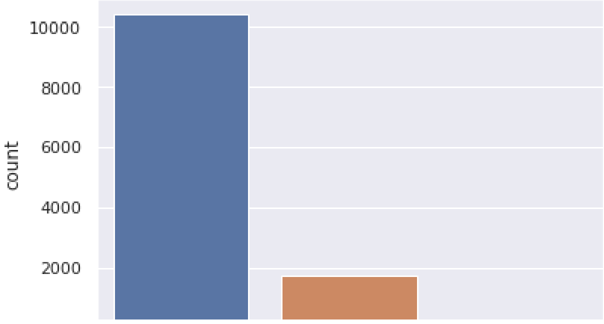
12316 rows × 104 columns

```
y=df2['Accident_severity']
y
```

```
0          Slight Injury
1          Slight Injury
2         Serious Injury
3          Slight Injury
4          Slight Injury
               ...
12311      Slight Injury
12312      Slight Injury
12313     Serious Injury
12314      Slight Injury
12315      Slight Injury
Name: Accident_severity, Length: 12316, dtype: object
```

```
sns.countplot('Accident_severity',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd147239a0>
```



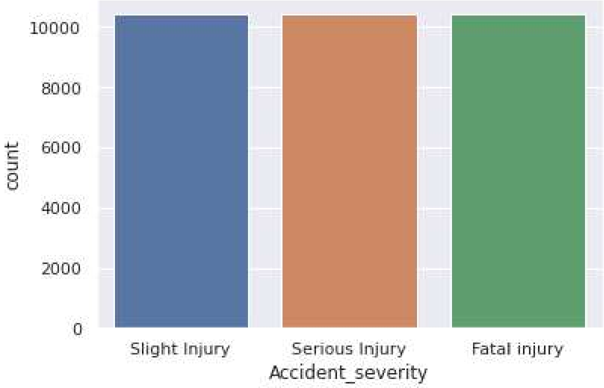
OverSampling

```
from imblearn.over_sampling import SMOTE
sampling=SMOTE()
xo,yo=sampling.fit_resample(x,y)
y1=pd.DataFrame(yo)
y1.value_counts()
```

```
Accident_severity
Fatal injury        10415
Serious Injury      10415
Slight Injury       10415
dtype: int64
```

```
sns.countplot('Accident_severity',data=y1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd135e34f0>
```



Splitting data into traing and testing

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(xo,yo,test_size=0.30,random_state=42)
x_train
```

	Number_of_vehicles_involved	Number_of_casualties	Age_band_of_driver_31-50	Age_band_of_driver_Over 51	Age_ban
1365	2	1	1	0	
22920	1	2	0	0	

```
x_test
```

	Number_of_vehicles_involved	Number_of_casualties	Age_band_of_driver_31-50	Age_band_of_driver_Over 51	Age_ban
29518	2	1	0	0	
10388	2	2	1	0	
8426	2	2	1	0	
16296	1	1	0	0	
27990	3	1	0	0	
...	...	...	...	...	
31135	2	1	0	1	
15063	1	1	0	0	
12917	2	1	0	0	
30794	2	2	0	0	
7834	2	1	0	1	

9374 rows × 104 columns

```
y_train
```

```
1365      Slight Injury
22920    Serious Injury
23609    Serious Injury
575       Slight Injury
3204      Slight Injury
              ...
29802    Serious Injury
5390      Slight Injury
860       Slight Injury
15795      Fatal injury
23654    Serious Injury
Name: Accident_severity, Length: 21871, dtype: object
```

```
y_test
```

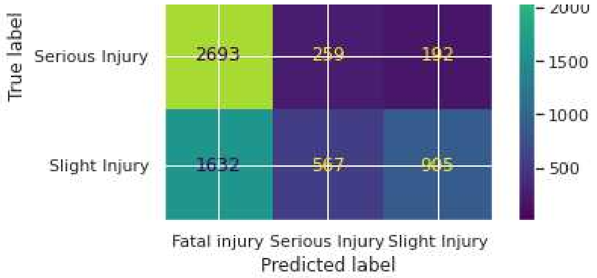
```
29518    Serious Injury
10388     Slight Injury
8426      Slight Injury
16296      Fatal injury
27990    Serious Injury
              ...
31135    Serious Injury
15063      Fatal injury
12917      Fatal injury
30794    Serious Injury
7834      Slight Injury
Name: Accident_severity, Length: 9374, dtype: object
```

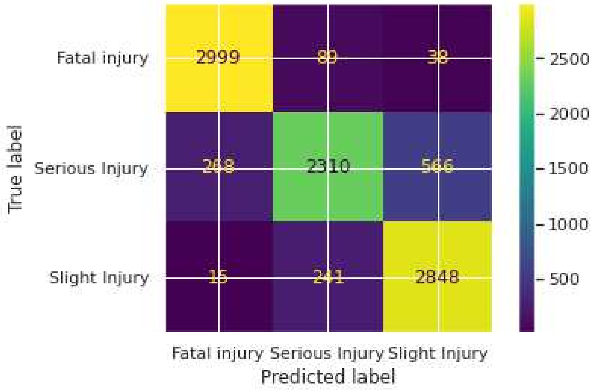
Model creation

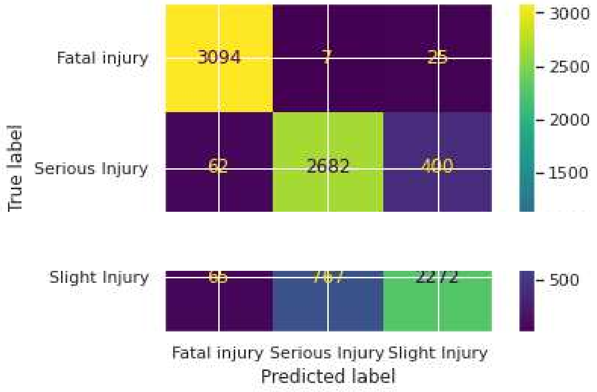
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
k_model=KNeighborsClassifier(n_neighbors=5)
nb_model=GaussianNB()
svm_model=SVC()
tree_model=DecisionTreeClassifier(criterion='entropy')
random_model=RandomForestClassifier(n_estimators=5,criterion='entropy')
lst_model=[k_model,nb_model,svm_model,tree_model,random_model]

from sklearn.metrics import confusion_matrix,classification_report
for i in lst_model:
  print(i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print("*********************************************************************************")
  print(confusion_matrix(y_test,y_pred))
  print("*********************************************************************************")
  print(classification_report(y_test,y_pred))
```

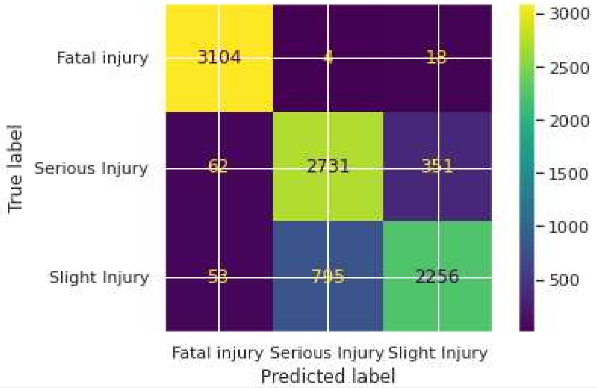
```
                precision    recall  f1-score   support

  Fatal injury       0.42      0.99      0.59      3126
Serious Injury       0.31      0.08      0.13      3144
 Slight Injury       0.82      0.29      0.43      3104

      accuracy                           0.45      9374
     macro avg       0.51      0.45      0.38      9374
  weighted avg       0.51      0.45      0.38      9374

SVC()
*********************************************************************************
[[2999   89   38]
 [ 268 2310  566]
 [  15  241 2848]]
********************************************************************************
                precision    recall  f1-score   support

  Fatal injury       0.91      0.96      0.94      3126
Serious Injury       0.88      0.73      0.80      3144
 Slight Injury       0.83      0.92      0.87      3104

      accuracy                           0.87      9374
     macro avg       0.87      0.87      0.87      9374
  weighted avg       0.87      0.87      0.87      9374

DecisionTreeClassifier(criterion='entropy')
*********************************************************************************
[[3096    8   22]
 [  62 2664  418]
 [  74  751 2279]]
********************************************************************************
                precision    recall  f1-score   support

  Fatal injury       0.96      0.99      0.97      3126
Serious Injury       0.78      0.85      0.81      3144
 Slight Injury       0.84      0.73      0.78      3104

      accuracy                           0.86      9374
     macro avg       0.86      0.86      0.86      9374
  weighted avg       0.86      0.86      0.86      9374

RandomForestClassifier(criterion='entropy', n_estimators=5)
*********************************************************************************
[[3106    6   14]
 [  53 2745  346]
 [  63  712 2329]]
********************************************************************************
                precision    recall  f1-score   support

  Fatal injury       0.96      0.99      0.98      3126
Serious Injury       0.79      0.87      0.83      3144
 Slight Injury       0.87      0.75      0.80      3104

      accuracy                           0.87      9374
     macro avg       0.87      0.87      0.87      9374
  weighted avg       0.87      0.87      0.87      9374
```

```
from sklearn.metrics import accuracy_score,ConfusionMatrixDisplay
for i in lst_model:
  print(i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print("************************************************")
  print(accuracy_score(y_test,y_pred))
  print("*****************************************")
  print(ConfusionMatrixDisplay.from_predictions(y_test,y_pred))
```


True label
Serious Injury
Slight Injury
2693
259
192
1632
567
905
2000
1500
1000
500
Fatal injury
Serious Injury
Slight Injury
Predicted label


True label
Fatal injury
Serious Injury
Slight Injury
2999
89
38
268
2310
566
15
241
2848
2500
2000
1500
1000
500
Fatal injury
Serious Injury
Slight Injury
Predicted label


True label
Fatal injury
Serious Injury
Slight Injury
3094
7
25
62
2682
400
65
767
2272
3000
2500
2000
1500
500
Fatal injury
Serious Injury
Slight Injury
Predicted label


True label
Fatal injury
Serious Injury
Slight Injury
3104
4
18
62
2731
351
53
795
2256
3000
2500
2000
1500
1000
500
Fatal injury
Serious Injury
Slight Injury
Predicted label