In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

pd.set_option('display.max_columns', None)
```

In [2]:
```
people = pd.read_csv('Traffic_Crashes_-_People.csv', low_memory=False)
vehicles = pd.read_csv('Traffic_Crashes_-_Vehicles.csv',low_memory=False)
crashes = pd.read_csv('Traffic_Crashes_-_Crashes.csv', low_memory=False)
```

In [3]:
```
people.head()
```

Out[3]:

	PERSON_ID	PERSON_TYPE	CRASH_RECORD_ID	RD_NO	VEHICLE_ID	CRASH_DATE	SEAT_NO	CITY	STATE	ZIPCODE
0	O749947	DRIVER	81dc0de2ed92aa62baccab641fa377be7feb1cc47e6554...	JC451435	834816.0	09/28/2019 03:30:00 AM	NaN	CHICAGO	IL	6065
1	O871921	DRIVER	af84fb5c8d996fcd3aefd36593c3a02e6e7509eeb27568...	JD208731	827212.0	04/13/2020 10:50:00 PM	NaN	CHICAGO	IL	6062
2	O10018	DRIVER	71162af7bf22799b776547132ebf134b5b438dcf3dac6b...	HY484534	9579.0	11/01/2015 05:00:00 AM	NaN	NaN	NaN	Nal
3	O10038	DRIVER	c21c476e2ccc41af550b5d858d22aaac4ffc88745a1700...	HY484750	9598.0	11/01/2015 08:00:00 AM	NaN	NaN	NaN	Nal
4	O10039	DRIVER	eb390a4c8e114c69488f5fb8a097fe629f5a92fd528cf4...	HY484778	9600.0	11/01/2015 10:15:00 AM	NaN	NaN	NaN	Nal

In [4]:
```
people.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1084709 entries, 0 to 1084708
Data columns (total 30 columns):
 #   Column                 Non-Null Count    Dtype
---  ------                 --------------    -----
 0   PERSON_ID              1084709 non-null  object
 1   PERSON_TYPE            1084709 non-null  object
 2   CRASH_RECORD_ID        1084709 non-null  object
 3   RD_NO                  1076708 non-null  object
 4   VEHICLE_ID             1063284 non-null  float64
 5   CRASH_DATE             1084709 non-null  object
 6   SEAT_NO                221576 non-null   float64
 7   CITY                   801391 non-null   object
 8   STATE                  810387 non-null   object
 9   ZIPCODE                731819 non-null   object
 10  SEX                    1068629 non-null  object
 11  AGE                    775285 non-null   float64
 12  DRIVERS_LICENSE_STATE  642907 non-null   object
 13  DRIVERS_LICENSE_CLASS  557675 non-null   object
 14  SAFETY_EQUIPMENT       1081493 non-null  object
 15  AIRBAG_DEPLOYED        1064224 non-null  object
 16  EJECTION               1071334 non-null  object
 17  INJURY_CLASSIFICATION  1084140 non-null  object
 18  HOSPITAL               197075 non-null   object
 19  EMS_AGENCY             125027 non-null   object
 20  EMS_RUN_NO             20326 non-null    object
 21  DRIVER_ACTION          861047 non-null   object
 22  DRIVER_VISION          860762 non-null   object
 23  PHYSICAL_CONDITION     861672 non-null   object
 24  PEDPEDAL_ACTION        20115 non-null    object
 25  PEDPEDAL_VISIBILITY    20072 non-null    object
 26  PEDPEDAL_LOCATION      20116 non-null    object
 27  BAC_RESULT             862160 non-null   object
 28  BAC_RESULT VALUE       1396 non-null     float64
 29  CELL_PHONE_USE         1157 non-null     object
dtypes: float64(4), object(26)
memory usage: 248.3+ MB
```

In [5]:
```
1 vehicles.head()
```

Out[5]:

	CRASH_UNIT_ID	CRASH_RECORD_ID	RD_NO	CRASH_DATE	UNIT_NO	UNIT_TYPE	NUM_PASSENGERS	VEHICLE_ID	CMRC
0	829999	24ddf9fd8542199d832e1c223cc474e5601b356f1d77a6...	JD124535	01/22/2020 06:25:00 AM	1	DRIVER	NaN	796949.0	
1	749947	81dc0de2ed92aa62baccab641fa377be7feb1cc47e6554...	JC451435	09/28/2019 03:30:00 AM	1	DRIVER	NaN	834816.0	
2	749949	81dc0de2ed92aa62baccab641fa377be7feb1cc47e6554...	JC451435	09/28/2019 03:30:00 AM	2	PARKED	NaN	834819.0	
3	749950	81dc0de2ed92aa62baccab641fa377be7feb1cc47e6554...	JC451435	09/28/2019 03:30:00 AM	3	PARKED	NaN	834817.0	
4	871921	af84fb5c8d996fcd3aefd36593c3a02e6e7509eeb27568...	JD208731	04/13/2020 10:50:00 PM	2	DRIVER	NaN	827212.0	

In [6]:
```
1 vehicles.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1002530 entries, 0 to 1002529
Data columns (total 72 columns):
 #   Column                     Non-Null Count    Dtype
---  ------                     --------------    -----
 0   CRASH_UNIT_ID              1002530 non-null  int64
 1   CRASH_RECORD_ID            1002530 non-null  object
 2   RD_NO                      994929 non-null   object
 3   CRASH_DATE                 1002530 non-null  object
 4   UNIT_NO                    1002530 non-null  int64
 5   UNIT_TYPE                  1001029 non-null  object
 6   NUM_PASSENGERS             150023 non-null   float64
 7   VEHICLE_ID                 979473 non-null   float64
 8   CMRC_VEH_I                 18691 non-null    object
 9   MAKE                       979468 non-null   object
 10  MODEL                      979326 non-null   object
 11  LIC_PLATE_STATE            895842 non-null   object
 12  VEHICLE_YEAR               820649 non-null   float64
 13  VEHICLE_DEFECT             979473 non-null   object
 14  VEHICLE_TYPE               979473 non-null   object
 15  VEHICLE_USE                979473 non-null   object
 16  TRAVEL_DIRECTION           979473 non-null   object
 17  MANEUVER                   979473 non-null   object
 18  TOWED_I                    113480 non-null   object
 19  FIRE_I                     732 non-null      object
 20  OCCUPANT_CNT               979473 non-null   float64
 21  EXCEED_SPEED_LIMIT_I       2390 non-null     object
 22  TOWED_BY                   82893 non-null    object
 23  TOWED_TO                   51848 non-null    object
 24  AREA_00_I                  38961 non-null    object
 25  AREA_01_I                  259916 non-null   object
 26  AREA_02_I                  170910 non-null   object
 27  AREA_03_I                  94865 non-null    object
 28  AREA_04_I                  99856 non-null    object
 29  AREA_05_I                  152027 non-null   object
 30  AREA_06_I                  150698 non-null   object
 31  AREA_07_I                  129600 non-null   object
 32  AREA_08_I                  168045 non-null   object
 33  AREA_09_I                  41672 non-null    object
 34  AREA_10_I                  60279 non-null    object
 35  AREA_11_I                  120244 non-null   object
 36  AREA_12_I                  118399 non-null   object
 37  AREA_99_I                  105625 non-null   object
 38  FIRST_CONTACT_POINT        972847 non-null   object
 39  CMV_ID                     10659 non-null    float64
 40  USDOT_NO                   6219 non-null     object
 41  CCMC_NO                    1396 non-null     object
 42  ILCC_NO                    1010 non-null     object
 43  COMMERCIAL_SRC             7613 non-null     object
 44  GVWR                       6198 non-null     object
 45  CARRIER_NAME               10220 non-null    object
 46  CARRIER_STATE              9664 non-null     object
 47  CARRIER_CITY               9491 non-null     object
 48  HAZMAT_PLACARDS_I          217 non-null      object
 49  HAZMAT_NAME                39 non-null       object
 50  UN_NO                      392 non-null      object
 51  HAZMAT_PRESENT_I           7871 non-null     object
 52  HAZMAT_REPORT_I            7627 non-null     object
 53  HAZMAT_REPORT_NO           1 non-null        object
 54  MCS_REPORT_I               7678 non-null     object
 55  MCS_REPORT_NO              5 non-null        object
 56  HAZMAT_VIO_CAUSE_CRASH_I   7749 non-null     object
 57  MCS_VIO_CAUSE_CRASH_I      7620 non-null     object
 58  IDOT_PERMIT_NO             633 non-null      object
 59  WIDE_LOAD_I                91 non-null       object
 60  TRAILER1_WIDTH             2186 non-null     object
 61  TRAILER2_WIDTH             245 non-null      object
 62  TRAILER1_LENGTH            1789 non-null     float64
 63  TRAILER2_LENGTH            47 non-null       float64
 64  TOTAL_VEHICLE_LENGTH       2157 non-null     float64
 65  AXLE_CNT                   3104 non-null     float64
 66  VEHICLE_CONFIG             8877 non-null     object
 67  CARGO_BODY_TYPE            8498 non-null     object
 68  LOAD_TYPE                  8136 non-null     object
 69  HAZMAT_OUT_OF_SERVICE_I    7412 non-null     object
 70  MCS_OUT_OF_SERVICE_I       7608 non-null     object
 71  HAZMAT_CLASS               730 non-null      object
dtypes: float64(9), int64(2), object(61)
memory usage: 550.7+ MB
```

In [7]:
```
1 crashes.head()
```

Out[7]:

	CRASH_RECORD_ID	RD_NO	CRASH_DATE_EST_I	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVIC
0	4fd0a3e0897b3335b94cd8d5b2d2b350eb691add56c62d...	JC343143	NaN	07/10/2019 05:56:00 PM	35	NO CONTROLS	
1	009e9e67203442370272e1a13d6ee51a4155dac65e583d...	JA329216	NaN	06/30/2017 04:00:00 PM	35	STOP SIGN/FLASHER	
2	ee9283eff3a55ac50ee58f3d9528ce1d689b1c4180b4c4...	JD292400	NaN	07/10/2020 10:25:00 AM	30	TRAFFIC SIGNAL	
3	f8960f698e870ebdc60b521b2a141a5395556bc3704191...	JD293602	NaN	07/11/2020 01:00:00 AM	30	NO CONTROLS	
4	8eaa2678d1a127804ee9b8c35ddf7d63d913c14eda61d6...	JD290451	NaN	07/08/2020 02:00:00 PM	20	NO CONTROLS	

In [8]:
```
1 crashes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 490939 entries, 0 to 490938
Data columns (total 49 columns):
 #   Column                         Non-Null Count   Dtype
---  ------                         --------------   -----
 0   CRASH_RECORD_ID                490939 non-null  object
 1   RD_NO                          487252 non-null  object
 2   CRASH_DATE_EST_I               36901 non-null   object
 3   CRASH_DATE                     490939 non-null  object
 4   POSTED_SPEED_LIMIT             490939 non-null  int64
 5   TRAFFIC_CONTROL_DEVICE         490939 non-null  object
 6   DEVICE_CONDITION               490939 non-null  object
 7   WEATHER_CONDITION              490939 non-null  object
 8   LIGHTING_CONDITION             490939 non-null  object
 9   FIRST_CRASH_TYPE               490939 non-null  object
 10  TRAFFICWAY_TYPE                490939 non-null  object
 11  LANE_CNT                       198965 non-null  float64
 12  ALIGNMENT                      490939 non-null  object
 13  ROADWAY_SURFACE_COND           490939 non-null  object
 14  ROAD_DEFECT                    490939 non-null  object
 15  REPORT_TYPE                    478949 non-null  object
 16  CRASH_TYPE                     490939 non-null  object
 17  INTERSECTION_RELATED_I         110783 non-null  object
 18  NOT_RIGHT_OF_WAY_I             23138 non-null   object
 19  HIT_AND_RUN_I                  144927 non-null  object
 20  DAMAGE                         490939 non-null  object
 21  DATE_POLICE_NOTIFIED           490939 non-null  object
 22  PRIM_CONTRIBUTORY_CAUSE        490939 non-null  object
 23  SEC_CONTRIBUTORY_CAUSE         490939 non-null  object
 24  STREET_NO                      490939 non-null  int64
 25  STREET_DIRECTION               490936 non-null  object
 26  STREET_NAME                    490938 non-null  object
 27  BEAT_OF_OCCURRENCE             490934 non-null  float64
 28  PHOTOS_TAKEN_I                 6167 non-null    object
 29  STATEMENTS_TAKEN_I             9907 non-null    object
 30  DOORING_I                      1563 non-null    object
 31  WORK_ZONE_I                    3152 non-null    object
 32  WORK_ZONE_TYPE                 2484 non-null    object
 33  WORKERS_PRESENT_I              756 non-null     object
 34  NUM_UNITS                      490939 non-null  int64
 35  MOST_SEVERE_INJURY             489942 non-null  object
 36  INJURIES_TOTAL                 489953 non-null  float64
 37  INJURIES_FATAL                 489953 non-null  float64
 38  INJURIES_INCAPACITATING        489953 non-null  float64
 39  INJURIES_NON_INCAPACITATING    489953 non-null  float64
 40  INJURIES_REPORTED_NOT_EVIDENT  489953 non-null  float64
 41  INJURIES_NO_INDICATION         489953 non-null  float64
 42  INJURIES_UNKNOWN               489953 non-null  float64
 43  CRASH_HOUR                     490939 non-null  int64
 44  CRASH_DAY_OF_WEEK              490939 non-null  int64
 45  CRASH_MONTH                    490939 non-null  int64
 46  LATITUDE                       488205 non-null  float64
 47  LONGITUDE                      488205 non-null  float64
 48  LOCATION                       488205 non-null  object
dtypes: float64(11), int64(6), object(32)
memory usage: 183.5+ MB
```

Removing unneccesary columns that do not have a direct effect on the features we are looking for

In [9]:
```
not_needed = ['CRASH_DATE_EST_I', 'DATE_POLICE_NOTIFIED', 'STATEMENTS_TAKEN_I','PHOTOS_TAKEN_I','LANE_CNT']
crashes2 = crashes.drop(columns = not_needed, axis=1)
```

In [10]:
```
crashes2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 490939 entries, 0 to 490938
Data columns (total 44 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   CRASH_RECORD_ID                 490939 non-null  object 
 1   RD_NO                           487252 non-null  object 
 2   CRASH_DATE                      490939 non-null  object 
 3   POSTED_SPEED_LIMIT              490939 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE          490939 non-null  object 
 5   DEVICE_CONDITION                490939 non-null  object 
 6   WEATHER_CONDITION               490939 non-null  object 
 7   LIGHTING_CONDITION              490939 non-null  object 
 8   FIRST_CRASH_TYPE                490939 non-null  object 
 9   TRAFFICWAY_TYPE                 490939 non-null  object 
 10  ALIGNMENT                       490939 non-null  object 
 11  ROADWAY_SURFACE_COND            490939 non-null  object 
 12  ROAD_DEFECT                     490939 non-null  object 
 13  REPORT_TYPE                     478949 non-null  object 
 14  CRASH_TYPE                      490939 non-null  object 
 15  INTERSECTION_RELATED_I          110783 non-null  object 
 16  NOT_RIGHT_OF_WAY_I              23138 non-null   object 
 17  HIT_AND_RUN_I                   144927 non-null  object 
 18  DAMAGE                          490939 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE         490939 non-null  object 
 20  SEC_CONTRIBUTORY_CAUSE          490939 non-null  object 
 21  STREET_NO                       490939 non-null  int64  
 22  STREET_DIRECTION                490936 non-null  object 
 23  STREET_NAME                     490938 non-null  object 
 24  BEAT_OF_OCCURRENCE              490934 non-null  float64
 25  DOORING_I                       1563 non-null    object 
 26  WORK_ZONE_I                     3152 non-null    object 
 27  WORK_ZONE_TYPE                  2484 non-null    object 
 28  WORKERS_PRESENT_I               756 non-null     object 
 29  NUM_UNITS                       490939 non-null  int64  
 30  MOST_SEVERE_INJURY              489942 non-null  object 
 31  INJURIES_TOTAL                  489953 non-null  float64
 32  INJURIES_FATAL                  489953 non-null  float64
 33  INJURIES_INCAPACITATING         489953 non-null  float64
 34  INJURIES_NON_INCAPACITATING     489953 non-null  float64
 35  INJURIES_REPORTED_NOT_EVIDENT   489953 non-null  float64
 36  INJURIES_NO_INDICATION          489953 non-null  float64
 37  INJURIES_UNKNOWN                489953 non-null  float64
 38  CRASH_HOUR                      490939 non-null  int64  
 39  CRASH_DAY_OF_WEEK               490939 non-null  int64  
 40  CRASH_MONTH                     490939 non-null  int64  
 41  LATITUDE                        488205 non-null  float64
 42  LONGITUDE                       488205 non-null  float64
 43  LOCATION                        488205 non-null  object 
dtypes: float64(10), int64(6), object(28)
memory usage: 164.8+ MB
```

In [11]:
```
crashes2.head()
```

Out[11]:

	CRASH_RECORD_ID	RD_NO	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEAT
0	4fd0a3e0897b3335b94cd8d5b2d2b350eb691add56c62d...	JC343143	07/10/2019 05:56:00 PM	35	NO CONTROLS	NO CONTROLS	
1	009e9e67203442370272e1a13d6ee51a4155dac65e583d...	JA329216	06/30/2017 04:00:00 PM	35	STOP SIGN/FLASHER	FUNCTIONING PROPERLY	
2	ee9283eff3a55ac50ee58f3d9528ce1d689b1c4180b4c4...	JD292400	07/10/2020 10:25:00 AM	30	TRAFFIC SIGNAL	FUNCTIONING PROPERLY	
3	f8960f698e870ebdc60b521b2a141a5395556bc3704191...	JD293602	07/11/2020 01:00:00 AM	30	NO CONTROLS	NO CONTROLS	
4	8eaa2678d1a127804ee9b8c35ddf7d63d913c14eda61d6...	JD290451	07/08/2020 02:00:00 PM	20	NO CONTROLS	NO CONTROLS	

Looking at columns with yes or no answers. As seen above in crashes.head() columns with I have Yes, No and NaN responses. We will clean up the NaN responses by setting them to 0 which is no and Yes will be 1.

Below we will get a better idea of the number of NaN values that we will need to clean up in our data.

In [12]:
```
I_columns = ['INTERSECTION_RELATED_I','DOORING_I', 'WORK_ZONE_I', 'WORKERS_PRESENT_I', 'NOT_RIGHT_OF_WAY_I', 'HIT_ANI
for col in I_columns:
    print(crashes2[col].value_counts())
    print('NaN count:', crashes2[col].isnull().sum())
```

```
Y    105548
N      5235
Name: INTERSECTION_RELATED_I, dtype: int64
NaN count: 380156
Y    1064
N     499
Name: DOORING_I, dtype: int64
NaN count: 489376
Y    2484
N     668
Name: WORK_ZONE_I, dtype: int64
NaN count: 487787
Y    677
N     79
Name: WORKERS_PRESENT_I, dtype: int64
NaN count: 490183
Y    21108
N     2030
Name: NOT_RIGHT_OF_WAY_I, dtype: int64
NaN count: 467801
Y    138622
N      6305
Name: HIT_AND_RUN_I, dtype: int64
NaN count: 346012
```

Then we will run the code necessary to perform our cleaning of the NaN values in crashes2.

In [13]:
```
for col in I_columns:
    crashes2[col] = crashes2[col].map(lambda x: 1 if x == 'Y' else 0)
    print(crashes2[col].value_counts())
    print('NaN present:', crashes2[col].isnull().sum())
```

```
0    385391
1    105548
Name: INTERSECTION_RELATED_I, dtype: int64
NaN present: 0
0    489875
1      1064
Name: DOORING_I, dtype: int64
NaN present: 0
0    488455
1      2484
Name: WORK_ZONE_I, dtype: int64
NaN present: 0
0    490262
1       677
Name: WORKERS_PRESENT_I, dtype: int64
NaN present: 0
0    469831
1     21108
Name: NOT_RIGHT_OF_WAY_I, dtype: int64
NaN present: 0
0    352317
1    138622
Name: HIT_AND_RUN_I, dtype: int64
NaN present: 0
```

Now, lets recheck the standing of our data after cleaning up our NaNs.

In [14]:
```
1 crashes2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 490939 entries, 0 to 490938
Data columns (total 44 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   CRASH_RECORD_ID                 490939 non-null  object 
 1   RD_NO                           487252 non-null  object 
 2   CRASH_DATE                      490939 non-null  object 
 3   POSTED_SPEED_LIMIT              490939 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE          490939 non-null  object 
 5   DEVICE_CONDITION                490939 non-null  object 
 6   WEATHER_CONDITION               490939 non-null  object 
 7   LIGHTING_CONDITION              490939 non-null  object 
 8   FIRST_CRASH_TYPE                490939 non-null  object 
 9   TRAFFICWAY_TYPE                 490939 non-null  object 
 10  ALIGNMENT                       490939 non-null  object 
 11  ROADWAY_SURFACE_COND            490939 non-null  object 
 12  ROAD_DEFECT                     490939 non-null  object 
 13  REPORT_TYPE                     478949 non-null  object 
 14  CRASH_TYPE                      490939 non-null  object 
 15  INTERSECTION_RELATED_I          490939 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I              490939 non-null  int64  
 17  HIT_AND_RUN_I                   490939 non-null  int64  
 18  DAMAGE                          490939 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE         490939 non-null  object 
 20  SEC_CONTRIBUTORY_CAUSE          490939 non-null  object 
 21  STREET_NO                       490939 non-null  int64  
 22  STREET_DIRECTION                490936 non-null  object 
 23  STREET_NAME                     490938 non-null  object 
 24  BEAT_OF_OCCURRENCE              490934 non-null  float64
 25  DOORING_I                       490939 non-null  int64  
 26  WORK_ZONE_I                     490939 non-null  int64  
 27  WORK_ZONE_TYPE                  2484 non-null    object 
 28  WORKERS_PRESENT_I               490939 non-null  int64  
 29  NUM_UNITS                       490939 non-null  int64  
 30  MOST_SEVERE_INJURY              489942 non-null  object 
 31  INJURIES_TOTAL                  489953 non-null  float64
 32  INJURIES_FATAL                  489953 non-null  float64
 33  INJURIES_INCAPACITATING         489953 non-null  float64
 34  INJURIES_NON_INCAPACITATING     489953 non-null  float64
 35  INJURIES_REPORTED_NOT_EVIDENT   489953 non-null  float64
 36  INJURIES_NO_INDICATION          489953 non-null  float64
 37  INJURIES_UNKNOWN                489953 non-null  float64
 38  CRASH_HOUR                      490939 non-null  int64  
 39  CRASH_DAY_OF_WEEK               490939 non-null  int64  
 40  CRASH_MONTH                     490939 non-null  int64  
 41  LATITUDE                        488205 non-null  float64
 42  LONGITUDE                       488205 non-null  float64
 43  LOCATION                        488205 non-null  object 
dtypes: float64(10), int64(12), object(22)
memory usage: 164.8+ MB
```

Work zone does not seem to apply to many of our rows and therefore might be easier just to remove the column as it is not as pertinent as our other columns

In [15]:
```
crashes2 = crashes2.drop(['WORK_ZONE_TYPE'], axis=1)
crashes2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 490939 entries, 0 to 490938
Data columns (total 43 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   CRASH_RECORD_ID                 490939 non-null  object 
 1   RD_NO                           487252 non-null  object 
 2   CRASH_DATE                      490939 non-null  object 
 3   POSTED_SPEED_LIMIT              490939 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE          490939 non-null  object 
 5   DEVICE_CONDITION                490939 non-null  object 
 6   WEATHER_CONDITION               490939 non-null  object 
 7   LIGHTING_CONDITION              490939 non-null  object 
 8   FIRST_CRASH_TYPE                490939 non-null  object 
 9   TRAFFICWAY_TYPE                 490939 non-null  object 
 10  ALIGNMENT                       490939 non-null  object 
 11  ROADWAY_SURFACE_COND            490939 non-null  object 
 12  ROAD_DEFECT                     490939 non-null  object 
 13  REPORT_TYPE                     478949 non-null  object 
 14  CRASH_TYPE                      490939 non-null  object 
 15  INTERSECTION_RELATED_I          490939 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I              490939 non-null  int64  
 17  HIT_AND_RUN_I                   490939 non-null  int64  
 18  DAMAGE                          490939 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE         490939 non-null  object 
 20  SEC_CONTRIBUTORY_CAUSE          490939 non-null  object 
 21  STREET_NO                       490939 non-null  int64  
 22  STREET_DIRECTION                490936 non-null  object 
 23  STREET_NAME                     490938 non-null  object 
 24  BEAT_OF_OCCURRENCE              490934 non-null  float64
 25  DOORING_I                       490939 non-null  int64  
 26  WORK_ZONE_I                     490939 non-null  int64  
 27  WORKERS_PRESENT_I               490939 non-null  int64  
 28  NUM_UNITS                       490939 non-null  int64  
 29  MOST_SEVERE_INJURY              489942 non-null  object 
 30  INJURIES_TOTAL                  489953 non-null  float64
 31  INJURIES_FATAL                  489953 non-null  float64
 32  INJURIES_INCAPACITATING         489953 non-null  float64
 33  INJURIES_NON_INCAPACITATING     489953 non-null  float64
 34  INJURIES_REPORTED_NOT_EVIDENT   489953 non-null  float64
 35  INJURIES_NO_INDICATION          489953 non-null  float64
 36  INJURIES_UNKNOWN                489953 non-null  float64
 37  CRASH_HOUR                      490939 non-null  int64  
 38  CRASH_DAY_OF_WEEK               490939 non-null  int64  
 39  CRASH_MONTH                     490939 non-null  int64  
 40  LATITUDE                        488205 non-null  float64
 41  LONGITUDE                       488205 non-null  float64
 42  LOCATION                        488205 non-null  object 
dtypes: float64(10), int64(12), object(21)
memory usage: 161.1+ MB
```

Exploration of Crash Data

We will look at the data presented in primary and secondary causes in order to help them improve our model, which will look at how we can prevent crashes. It is important to also remember that for some data, primary contributory cause is not present but there is data present for a secondary cause. For those items we will attempt to replace the datapoint in the secondary contributory cause and move it to primary for ease of use. What we want to look at is overall contributory cause as primary and secondary is not as relevant for our purposes.

Exploring PRIM_CONTRIBUTORY_CAUSE

In [16]:
```
1 crashes2['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[16]:

```
UNABLE TO DETERMINE                                                    181836
FAILING TO YIELD RIGHT-OF-WAY                                           53936
FOLLOWING TOO CLOSELY                                                   51957
NOT APPLICABLE                                                          26344
IMPROPER OVERTAKING/PASSING                                             23310
IMPROPER BACKING                                                        21492
FAILING TO REDUCE SPEED TO AVOID CRASH                                  21170
IMPROPER LANE USAGE                                                     18938
IMPROPER TURNING/NO SIGNAL                                              16271
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                     15350
DISREGARDING TRAFFIC SIGNALS                                             8921
WEATHER                                                                  8506
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER    6130
DISREGARDING STOP SIGN                                                   5425
DISTRACTION - FROM INSIDE VEHICLE                                        3593
EQUIPMENT - VEHICLE CONDITION                                            3079
PHYSICAL CONDITION OF DRIVER                                             2883
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                     2866
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)       2619
DRIVING ON WRONG SIDE/WRONG WAY                                          2313
DISTRACTION - FROM OUTSIDE VEHICLE                                       2165
EXCEEDING AUTHORIZED SPEED LIMIT                                         1982
EXCEEDING SAFE SPEED FOR CONDITIONS                                      1684
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                 1383
ROAD CONSTRUCTION/MAINTENANCE                                            1186
DISREGARDING OTHER TRAFFIC SIGNS                                         1047
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                         913
CELL PHONE USE OTHER THAN TEXTING                                         687
DISREGARDING ROAD MARKINGS                                                675
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                           544
ANIMAL                                                                    413
TURNING RIGHT ON RED                                                      344
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    232
TEXTING                                                                   214
DISREGARDING YIELD SIGN                                                   187
RELATED TO BUS STOP                                                       164
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                     66
PASSING STOPPED SCHOOL BUS                                                 64
OBSTRUCTED CROSSWALKS                                                      31
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                  19
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

We have 181836 rows that have UNABLE TO DETERMINE in this column. Hopefully we can combine this with secondary causes

Exploring SEC_CONTRIBUTORY_CAUSE

In [17]:
```
1 crashes2['SEC_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[17]:

NOT APPLICABLE	198979
UNABLE TO DETERMINE	175393
FAILING TO REDUCE SPEED TO AVOID CRASH	20317
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE	15397
FAILING TO YIELD RIGHT-OF-WAY	14714
FOLLOWING TOO CLOSELY	13502
IMPROPER OVERTAKING/PASSING	7256
IMPROPER LANE USAGE	7050
WEATHER	6334
IMPROPER TURNING/NO SIGNAL	4846
IMPROPER BACKING	4229
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER	3304
DISREGARDING TRAFFIC SIGNALS	1860
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)	1657
DISTRACTION - FROM INSIDE VEHICLE	1570
PHYSICAL CONDITION OF DRIVER	1520
EXCEEDING AUTHORIZED SPEED LIMIT	1473
DISREGARDING STOP SIGN	1465
EXCEEDING SAFE SPEED FOR CONDITIONS	1438
EQUIPMENT - VEHICLE CONDITION	994
DRIVING ON WRONG SIDE/WRONG WAY	909
DISTRACTION - FROM OUTSIDE VEHICLE	895
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)	846
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)	668
ROAD CONSTRUCTION/MAINTENANCE	647
DISREGARDING ROAD MARKINGS	544
ROAD ENGINEERING/SURFACE/MARKING DEFECTS	543
DISREGARDING OTHER TRAFFIC SIGNS	516
CELL PHONE USE OTHER THAN TEXTING	409
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST	272
ANIMAL	240
BICYCLE ADVANCING LEGALLY ON RED LIGHT	215
RELATED TO BUS STOP	196
TURNING RIGHT ON RED	186
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)	156
DISREGARDING YIELD SIGN	139
TEXTING	96
PASSING STOPPED SCHOOL BUS	66
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT	50
OBSTRUCTED CROSSWALKS	48

Name: SEC_CONTRIBUTORY_CAUSE, dtype: int64

Now we will review the values in primary contributory causes under 'unable to determine in relation to secondary contributory causes in order to potentially help streamline our crash data.

In [18]:
```
utd = crashes2[crashes2['PRIM_CONTRIBUTORY_CAUSE'] == 'UNABLE TO DETERMINE']
utd['SEC_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[18]:
```
UNABLE TO DETERMINE                                                    110017
NOT APPLICABLE                                                          66714
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                       995
FAILING TO REDUCE SPEED TO AVOID CRASH                                    531
WEATHER                                                                   496
FOLLOWING TOO CLOSELY                                                     400
IMPROPER LANE USAGE                                                       399
FAILING TO YIELD RIGHT-OF-WAY                                             375
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                           242
PHYSICAL CONDITION OF DRIVER                                              232
IMPROPER BACKING                                                          186
IMPROPER OVERTAKING/PASSING                                               165
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER    138
IMPROPER TURNING/NO SIGNAL                                                 98
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                       77
EQUIPMENT - VEHICLE CONDITION                                              66
EXCEEDING AUTHORIZED SPEED LIMIT                                           65
ANIMAL                                                                     62
ROAD CONSTRUCTION/MAINTENANCE                                              52
DISTRACTION - FROM INSIDE VEHICLE                                          50
DISREGARDING TRAFFIC SIGNALS                                               49
DISREGARDING STOP SIGN                                                     46
DISTRACTION - FROM OUTSIDE VEHICLE                                         42
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                     40
DRIVING ON WRONG SIDE/WRONG WAY                                            37
EXCEEDING SAFE SPEED FOR CONDITIONS                                        33
RELATED TO BUS STOP                                                        29
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)         29
DISREGARDING YIELD SIGN                                                    29
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                   26
DISREGARDING OTHER TRAFFIC SIGNS                                           25
DISREGARDING ROAD MARKINGS                                                 21
CELL PHONE USE OTHER THAN TEXTING                                          16
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                  11
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                          10
TURNING RIGHT ON RED                                                       10
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)     8
OBSTRUCTED CROSSWALKS                                                       7
PASSING STOPPED SCHOOL BUS                                                  5
TEXTING                                                                     3
Name: SEC_CONTRIBUTORY_CAUSE, dtype: int64
```

In [19]:
```
apl = crashes2[crashes2['PRIM_CONTRIBUTORY_CAUSE'] == 'NOT APPLICABLE']
apl['SEC_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[19]:
```
NOT APPLICABLE                                                         24705
UNABLE TO DETERMINE                                                     1071
WEATHER                                                                   59
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                       50
FAILING TO YIELD RIGHT-OF-WAY                                             50
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                    46
FOLLOWING TOO CLOSELY                                                     37
FAILING TO REDUCE SPEED TO AVOID CRASH                                    34
IMPROPER BACKING                                                          26
RELATED TO BUS STOP                                                       25
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                      21
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER    19
IMPROPER LANE USAGE                                                       18
EQUIPMENT - VEHICLE CONDITION                                             18
DISREGARDING TRAFFIC SIGNALS                                              14
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                  14
IMPROPER OVERTAKING/PASSING                                               13
PHYSICAL CONDITION OF DRIVER                                              12
DISTRACTION - FROM OUTSIDE VEHICLE                                        12
DISTRACTION - FROM INSIDE VEHICLE                                         12
IMPROPER TURNING/NO SIGNAL                                                 9
ROAD CONSTRUCTION/MAINTENANCE                                              9
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                  9
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                          7
EXCEEDING SAFE SPEED FOR CONDITIONS                                        7
DISREGARDING STOP SIGN                                                     7
ANIMAL                                                                     6
DRIVING ON WRONG SIDE/WRONG WAY                                            6
DISREGARDING OTHER TRAFFIC SIGNS                                           5
OBSTRUCTED CROSSWALKS                                                      5
CELL PHONE USE OTHER THAN TEXTING                                          5
TEXTING                                                                    4
EXCEEDING AUTHORIZED SPEED LIMIT                                           4
DISREGARDING ROAD MARKINGS                                                 3
PASSING STOPPED SCHOOL BUS                                                 1
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)     1
Name: SEC_CONTRIBUTORY_CAUSE, dtype: int64
```

We will now create a copy of crashes2 named crashes3 to see how our data has changed and so we can implement some changes will still maintaining a previous version we can refer to if needed in the future

In [20]:
```
crashes3 = crashes2.copy()
crashes3.head()
```

Out[20]:

	CRASH_RECORD_ID	RD_NO	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEAT
0	4fd0a3e0897b3335b94cd8d5b2d2b350eb691add56c62d...	JC343143	07/10/2019 05:56:00 PM	35	NO CONTROLS	NO CONTROLS	
1	009e9e67203442370272e1a13d6ee51a4155dac65e583d...	JA329216	06/30/2017 04:00:00 PM	35	STOP SIGN/FLASHER	FUNCTIONING PROPERLY	
2	ee9283eff3a55ac50ee58f3d9528ce1d689b1c4180b4c4...	JD292400	07/10/2020 10:25:00 AM	30	TRAFFIC SIGNAL	FUNCTIONING PROPERLY	
3	f8960f698e870ebdc60b521b2a141a5395556bc3704191...	JD293602	07/11/2020 01:00:00 AM	30	NO CONTROLS	NO CONTROLS	
4	8eaa2678d1a127804ee9b8c35ddf7d63d913c14eda61d6...	JD290451	07/08/2020 02:00:00 PM	20	NO CONTROLS	NO CONTROLS	

In [21]:
```
crashes3['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[21]:
```
UNABLE TO DETERMINE                                                    181836
FAILING TO YIELD RIGHT-OF-WAY                                           53936
FOLLOWING TOO CLOSELY                                                   51957
NOT APPLICABLE                                                          26344
IMPROPER OVERTAKING/PASSING                                             23310
IMPROPER BACKING                                                        21492
FAILING TO REDUCE SPEED TO AVOID CRASH                                  21170
IMPROPER LANE USAGE                                                     18938
IMPROPER TURNING/NO SIGNAL                                              16271
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                     15350
DISREGARDING TRAFFIC SIGNALS                                             8921
WEATHER                                                                  8506
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER    6130
DISREGARDING STOP SIGN                                                   5425
DISTRACTION - FROM INSIDE VEHICLE                                        3593
EQUIPMENT - VEHICLE CONDITION                                            3079
PHYSICAL CONDITION OF DRIVER                                             2883
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                     2866
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)       2619
DRIVING ON WRONG SIDE/WRONG WAY                                          2313
DISTRACTION - FROM OUTSIDE VEHICLE                                       2165
EXCEEDING AUTHORIZED SPEED LIMIT                                         1982
EXCEEDING SAFE SPEED FOR CONDITIONS                                      1684
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                 1383
ROAD CONSTRUCTION/MAINTENANCE                                            1186
DISREGARDING OTHER TRAFFIC SIGNS                                         1047
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                         913
CELL PHONE USE OTHER THAN TEXTING                                         687
DISREGARDING ROAD MARKINGS                                                675
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                           544
ANIMAL                                                                    413
TURNING RIGHT ON RED                                                      344
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    232
TEXTING                                                                   214
DISREGARDING YIELD SIGN                                                   187
RELATED TO BUS STOP                                                       164
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                     66
PASSING STOPPED SCHOOL BUS                                                 64
OBSTRUCTED CROSSWALKS                                                      31
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                  19
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

In [ ]:
```

```

In [22]:
```
for index, row in crashes3.iterrows():
    if crashes3.loc[index,'PRIM_CONTRIBUTORY_CAUSE'] == 'UNABLE TO DETERMINE':
        if (crashes3.loc[index,'SEC_CONTRIBUTORY_CAUSE'] != 'UNABLE TO DETERMINE') & (crashes3.loc[index,'SEC_CONTRI
            crashes3.loc[index,'PRIM_CONTRIBUTORY_CAUSE'] = crashes3.loc[index,'SEC_CONTRIBUTORY_CAUSE']
```

In [23]:
```
1 crashes3['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[23]:
```
UNABLE TO DETERMINE                                                    176731
FAILING TO YIELD RIGHT-OF-WAY                                           54311
FOLLOWING TOO CLOSELY                                                   52357
NOT APPLICABLE                                                          26344
IMPROPER OVERTAKING/PASSING                                             23475
FAILING TO REDUCE SPEED TO AVOID CRASH                                  21701
IMPROPER BACKING                                                        21678
IMPROPER LANE USAGE                                                     19337
IMPROPER TURNING/NO SIGNAL                                              16369
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                     16345
WEATHER                                                                  9002
DISREGARDING TRAFFIC SIGNALS                                             8970
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER    6268
DISREGARDING STOP SIGN                                                   5471
DISTRACTION - FROM INSIDE VEHICLE                                        3643
EQUIPMENT - VEHICLE CONDITION                                            3145
PHYSICAL CONDITION OF DRIVER                                             3115
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                     2943
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)       2648
DRIVING ON WRONG SIDE/WRONG WAY                                          2350
DISTRACTION - FROM OUTSIDE VEHICLE                                       2207
EXCEEDING AUTHORIZED SPEED LIMIT                                         2047
EXCEEDING SAFE SPEED FOR CONDITIONS                                      1717
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                 1409
ROAD CONSTRUCTION/MAINTENANCE                                            1238
DISREGARDING OTHER TRAFFIC SIGNS                                         1072
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                         923
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                           786
CELL PHONE USE OTHER THAN TEXTING                                         703
DISREGARDING ROAD MARKINGS                                                696
ANIMAL                                                                    475
TURNING RIGHT ON RED                                                      354
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    240
TEXTING                                                                   217
DISREGARDING YIELD SIGN                                                   216
RELATED TO BUS STOP                                                       193
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                    106
PASSING STOPPED SCHOOL BUS                                                 69
OBSTRUCTED CROSSWALKS                                                      38
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                  30
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

In [ ]:
```
1
```

In [24]:
```
for index, row in crashes3.iterrows():
if crashes3.loc[index,'PRIM_CONTRIBUTORY_CAUSE'] == 'NOT APPLICABLE':
    if (crashes3.loc[index,'SEC_CONTRIBUTORY_CAUSE'] != 'UNABLE TO DETERMINE') & (crashes3.loc[index,'SEC_CONTRIBUTOR
        crashes3.loc[index,'PRIM_CONTRIBUTORY_CAUSE'] = crashes3.loc[index,'SEC_CONTRIBUTORY_CAUSE']
```

In [25]:
```
1 crashes3['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[25]:

UNABLE TO DETERMINE	176731
FAILING TO YIELD RIGHT-OF-WAY	54361
FOLLOWING TOO CLOSELY	52394
NOT APPLICABLE	25776
IMPROPER OVERTAKING/PASSING	23488
FAILING TO REDUCE SPEED TO AVOID CRASH	21735
IMPROPER BACKING	21704
IMPROPER LANE USAGE	19355
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE	16395
IMPROPER TURNING/NO SIGNAL	16378
WEATHER	9061
DISREGARDING TRAFFIC SIGNALS	8984
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER	6287
DISREGARDING STOP SIGN	5478
DISTRACTION - FROM INSIDE VEHICLE	3655
EQUIPMENT - VEHICLE CONDITION	3163
PHYSICAL CONDITION OF DRIVER	3127
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)	2964
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)	2648
DRIVING ON WRONG SIDE/WRONG WAY	2356
DISTRACTION - FROM OUTSIDE VEHICLE	2219
EXCEEDING AUTHORIZED SPEED LIMIT	2051
EXCEEDING SAFE SPEED FOR CONDITIONS	1724
ROAD ENGINEERING/SURFACE/MARKING DEFECTS	1423
ROAD CONSTRUCTION/MAINTENANCE	1247
DISREGARDING OTHER TRAFFIC SIGNS	1077
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST	930
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)	786
CELL PHONE USE OTHER THAN TEXTING	708
DISREGARDING ROAD MARKINGS	699
ANIMAL	481
TURNING RIGHT ON RED	354
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)	241
TEXTING	221
RELATED TO BUS STOP	218
DISREGARDING YIELD SIGN	216
BICYCLE ADVANCING LEGALLY ON RED LIGHT	152
PASSING STOPPED SCHOOL BUS	70
OBSTRUCTED CROSSWALKS	43
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT	39

Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64

Drop SEC_CONTRIBUTORY_CAUSE and verify!

In [26]:
```
pcc = crashes3.drop(['SEC_CONTRIBUTORY_CAUSE'], axis=1)
pcc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 490939 entries, 0 to 490938
Data columns (total 42 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   CRASH_RECORD_ID                 490939 non-null  object 
 1   RD_NO                           487252 non-null  object 
 2   CRASH_DATE                      490939 non-null  object 
 3   POSTED_SPEED_LIMIT              490939 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE          490939 non-null  object 
 5   DEVICE_CONDITION                490939 non-null  object 
 6   WEATHER_CONDITION               490939 non-null  object 
 7   LIGHTING_CONDITION              490939 non-null  object 
 8   FIRST_CRASH_TYPE                490939 non-null  object 
 9   TRAFFICWAY_TYPE                 490939 non-null  object 
 10  ALIGNMENT                       490939 non-null  object 
 11  ROADWAY_SURFACE_COND            490939 non-null  object 
 12  ROAD_DEFECT                     490939 non-null  object 
 13  REPORT_TYPE                     478949 non-null  object 
 14  CRASH_TYPE                      490939 non-null  object 
 15  INTERSECTION_RELATED_I          490939 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I              490939 non-null  int64  
 17  HIT_AND_RUN_I                   490939 non-null  int64  
 18  DAMAGE                          490939 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE         490939 non-null  object 
 20  STREET_NO                       490939 non-null  int64  
 21  STREET_DIRECTION                490936 non-null  object 
 22  STREET_NAME                     490938 non-null  object 
 23  BEAT_OF_OCCURRENCE              490934 non-null  float64
 24  DOORING_I                       490939 non-null  int64  
 25  WORK_ZONE_I                     490939 non-null  int64  
 26  WORKERS_PRESENT_I               490939 non-null  int64  
 27  NUM_UNITS                       490939 non-null  int64  
 28  MOST_SEVERE_INJURY              489942 non-null  object 
 29  INJURIES_TOTAL                  489953 non-null  float64
 30  INJURIES_FATAL                  489953 non-null  float64
 31  INJURIES_INCAPACITATING         489953 non-null  float64
 32  INJURIES_NON_INCAPACITATING     489953 non-null  float64
 33  INJURIES_REPORTED_NOT_EVIDENT   489953 non-null  float64
 34  INJURIES_NO_INDICATION          489953 non-null  float64
 35  INJURIES_UNKNOWN                489953 non-null  float64
 36  CRASH_HOUR                      490939 non-null  int64  
 37  CRASH_DAY_OF_WEEK               490939 non-null  int64  
 38  CRASH_MONTH                     490939 non-null  int64  
 39  LATITUDE                        488205 non-null  float64
 40  LONGITUDE                       488205 non-null  float64
 41  LOCATION                        488205 non-null  object 
dtypes: float64(10), int64(12), object(20)
memory usage: 157.3+ MB
```

Now it will be important to remove the rows in our dataset where the cause of the crash was not determined or not applicable. This will help improve the data for modeling since it helps to know what caused a crash to help prevent them in the future.

In [27]:
```
causes = pcc.loc[(pcc['PRIM_CONTRIBUTORY_CAUSE'] != 'NOT APPLICABLE')]
causes = causes.loc[(pcc['PRIM_CONTRIBUTORY_CAUSE'] != 'UNABLE TO DETERMINE')]
causes['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[27]:
```
FAILING TO YIELD RIGHT-OF-WAY                                                54361
FOLLOWING TOO CLOSELY                                                        52394
IMPROPER OVERTAKING/PASSING                                                  23488
FAILING TO REDUCE SPEED TO AVOID CRASH                                       21735
IMPROPER BACKING                                                             21704
IMPROPER LANE USAGE                                                          19355
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          16395
IMPROPER TURNING/NO SIGNAL                                                   16378
WEATHER                                                                       9061
DISREGARDING TRAFFIC SIGNALS                                                  8984
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER  6287
DISREGARDING STOP SIGN                                                        5478
DISTRACTION - FROM INSIDE VEHICLE                                             3655
EQUIPMENT - VEHICLE CONDITION                                                 3163
PHYSICAL CONDITION OF DRIVER                                                  3127
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2964
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2648
DRIVING ON WRONG SIDE/WRONG WAY                                               2356
DISTRACTION - FROM OUTSIDE VEHICLE                                            2219
EXCEEDING AUTHORIZED SPEED LIMIT                                              2051
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1724
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                      1423
ROAD CONSTRUCTION/MAINTENANCE                                                 1247
DISREGARDING OTHER TRAFFIC SIGNS                                              1077
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                              930
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                786
CELL PHONE USE OTHER THAN TEXTING                                              708
DISREGARDING ROAD MARKINGS                                                     699
ANIMAL                                                                         481
TURNING RIGHT ON RED                                                           354
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    241
TEXTING                                                                        221
RELATED TO BUS STOP                                                            218
DISREGARDING YIELD SIGN                                                        216
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         152
PASSING STOPPED SCHOOL BUS                                                      70
OBSTRUCTED CROSSWALKS                                                           43
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       39
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Injury Results and Columns

It is important to look at our values for injuries to see if we need to make any adjustments

In [28]:
```
causes['INJURIES_TOTAL'].value_counts()

```

Out[28]:
```
0.0     243293
1.0      33092
2.0       7870
3.0       2434
4.0        889
5.0        338
6.0        142
7.0         49
8.0         16
9.0         11
10.0         6
11.0         5
15.0         4
12.0         2
21.0         2
13.0         1
19.0         1
Name: INJURIES_TOTAL, dtype: int64
```

In [29]:
```
print('NaN count:', causes['INJURIES_TOTAL'].isnull().sum())
```

```
NaN count: 277
```

In [30]:
```
injury = causes[causes['INJURIES_TOTAL'].isna()]
injury
```

Out[30]:

	CRASH_RECORD_ID	RD_NO	CRASH_DATE	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	V
1218	83d05299efc278e3cf3095d1c0972faf3fb5714dfb434f...	JD210607	04/16/2020 11:05:00 PM	10	OTHER	OTHER	
3917	3ece743dad3f0948035957ddf0bd5a703492bc5d385ebc...	JC276163	05/24/2019 01:30:00 AM	10	NO CONTROLS	NO CONTROLS	
3970	9833fcfd2cff7ffb134ffa63eaa8397d26c2ccae7c49bc...	JC355633	07/19/2019 02:00:00 PM	10	NO CONTROLS	NO CONTROLS	
6276	c1109fab24b79660baf0375731af68788821188673c8e0...	JC146249	02/07/2019 12:00:00 PM	30	NO CONTROLS	NO CONTROLS	
12672	9eb385de99935fd761c6a7785627f26ceb6e5db8bb5d90...	JD301333	07/18/2020 03:12:00 AM	30	NO CONTROLS	NO CONTROLS	
...	...	...	...	...	...	...	
481792	fb03e3c84d227069d78ffb9888a3b07cce36abe3b584e0...	JB417772	09/01/2018 11:00:00 AM	15	NO CONTROLS	NO CONTROLS	
481911	fafbe57fed61cef5bb75417b22d04adde611051af0eff0...	JD414269	10/29/2020 11:17:00 AM	15	NO CONTROLS	NO CONTROLS	
485629	fd1761860f9d437d0880a636e5db40bfa7cf3505693a46...	JB261064	05/12/2018 04:56:00 PM	30	NO CONTROLS	NO CONTROLS	
488732	fecfdcb303b6b97260affee2be26dc3c61388bc0d3dfa5...	JC224212	04/14/2019 10:50:00 AM	30	NO CONTROLS	NO CONTROLS	
488959	fef1a57e01711bea9e4a1f82d66ea13eeda25a6a366fce...	JB147110	02/07/2018 11:00:00 AM	5	NO CONTROLS	NO CONTROLS	

277 rows × 42 columns

In [31]:
```
injury['CRASH_TYPE'].value_counts()
```

Out[31]:
```
NO INJURY / DRIVE AWAY              197
INJURY AND / OR TOW DUE TO CRASH     80
Name: CRASH_TYPE, dtype: int64
```

In [32]:
```
injury['HIT_AND_RUN_I'].value_counts()

```

Out[32]:
```
0    208
1     69
Name: HIT_AND_RUN_I, dtype: int64
```

Most of the crashes with NaN involve situations in which there was not a connection to individuals that where injured or that the individual involved ran away from the accident. Since it removes a confounding variable it is best to remove.

In [33]:
```
causes = causes.dropna(subset = ['MOST_SEVERE_INJURY'])
causes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 288152 entries, 0 to 490937
Data columns (total 42 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   CRASH_RECORD_ID                 288152 non-null  object 
 1   RD_NO                           286109 non-null  object 
 2   CRASH_DATE                      288152 non-null  object 
 3   POSTED_SPEED_LIMIT              288152 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE          288152 non-null  object 
 5   DEVICE_CONDITION                288152 non-null  object 
 6   WEATHER_CONDITION               288152 non-null  object 
 7   LIGHTING_CONDITION              288152 non-null  object 
 8   FIRST_CRASH_TYPE                288152 non-null  object 
 9   TRAFFICWAY_TYPE                 288152 non-null  object 
 10  ALIGNMENT                       288152 non-null  object 
 11  ROADWAY_SURFACE_COND            288152 non-null  object 
 12  ROAD_DEFECT                     288152 non-null  object 
 13  REPORT_TYPE                     280921 non-null  object 
 14  CRASH_TYPE                      288152 non-null  object 
 15  INTERSECTION_RELATED_I          288152 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I              288152 non-null  int64  
 17  HIT_AND_RUN_I                   288152 non-null  int64  
 18  DAMAGE                          288152 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE         288152 non-null  object 
 20  STREET_NO                       288152 non-null  int64  
 21  STREET_DIRECTION                288150 non-null  object 
 22  STREET_NAME                     288152 non-null  object 
 23  BEAT_OF_OCCURRENCE              288148 non-null  float64
 24  DOORING_I                       288152 non-null  int64  
 25  WORK_ZONE_I                     288152 non-null  int64  
 26  WORKERS_PRESENT_I               288152 non-null  int64  
 27  NUM_UNITS                       288152 non-null  int64  
 28  MOST_SEVERE_INJURY              288152 non-null  object 
 29  INJURIES_TOTAL                  288152 non-null  float64
 30  INJURIES_FATAL                  288152 non-null  float64
 31  INJURIES_INCAPACITATING         288152 non-null  float64
 32  INJURIES_NON_INCAPACITATING     288152 non-null  float64
 33  INJURIES_REPORTED_NOT_EVIDENT   288152 non-null  float64
 34  INJURIES_NO_INDICATION          288152 non-null  float64
 35  INJURIES_UNKNOWN                288152 non-null  float64
 36  CRASH_HOUR                      288152 non-null  int64  
 37  CRASH_DAY_OF_WEEK               288152 non-null  int64  
 38  CRASH_MONTH                     288152 non-null  int64  
 39  LATITUDE                        286390 non-null  float64
 40  LONGITUDE                       286390 non-null  float64
 41  LOCATION                        286390 non-null  object 
dtypes: float64(10), int64(12), object(20)
memory usage: 94.5+ MB
```

Next, we will go ahead and drop the columns associated with injury so that we can focus on what was the most severe injury to occur in the crashes. This will help in focusing our data exploration and our goal to highlight the most severe injuries caused in accidents so that we can try to reduce these fatal occurances and increase safety on the road.

In [34]:
```
causes = causes.drop(columns=['INJURIES_FATAL', 'INJURIES_INCAPACITATING', 'INJURIES_NON_INCAPACITATING',
                             'INJURIES_REPORTED_NOT_EVIDENT', 'INJURIES_NO_INDICATION','INJURIES_UNKNOWN'], axis=1)
causes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 288152 entries, 0 to 490937
Data columns (total 36 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   CRASH_RECORD_ID           288152 non-null  object 
 1   RD_NO                     286109 non-null  object 
 2   CRASH_DATE                288152 non-null  object 
 3   POSTED_SPEED_LIMIT        288152 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE    288152 non-null  object 
 5   DEVICE_CONDITION          288152 non-null  object 
 6   WEATHER_CONDITION         288152 non-null  object 
 7   LIGHTING_CONDITION        288152 non-null  object 
 8   FIRST_CRASH_TYPE          288152 non-null  object 
 9   TRAFFICWAY_TYPE           288152 non-null  object 
 10  ALIGNMENT                 288152 non-null  object 
 11  ROADWAY_SURFACE_COND      288152 non-null  object 
 12  ROAD_DEFECT               288152 non-null  object 
 13  REPORT_TYPE               280921 non-null  object 
 14  CRASH_TYPE                288152 non-null  object 
 15  INTERSECTION_RELATED_I    288152 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I        288152 non-null  int64  
 17  HIT_AND_RUN_I             288152 non-null  int64  
 18  DAMAGE                    288152 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE   288152 non-null  object 
 20  STREET_NO                 288152 non-null  int64  
 21  STREET_DIRECTION          288150 non-null  object 
 22  STREET_NAME               288152 non-null  object 
 23  BEAT_OF_OCCURRENCE        288148 non-null  float64
 24  DOORING_I                 288152 non-null  int64  
 25  WORK_ZONE_I               288152 non-null  int64  
 26  WORKERS_PRESENT_I         288152 non-null  int64  
 27  NUM_UNITS                 288152 non-null  int64  
 28  MOST_SEVERE_INJURY        288152 non-null  object 
 29  INJURIES_TOTAL            288152 non-null  float64
 30  CRASH_HOUR                288152 non-null  int64  
 31  CRASH_DAY_OF_WEEK         288152 non-null  int64  
 32  CRASH_MONTH               288152 non-null  int64  
 33  LATITUDE                  286390 non-null  float64
 34  LONGITUDE                 286390 non-null  float64
 35  LOCATION                  286390 non-null  object 
dtypes: float64(4), int64(12), object(20)
memory usage: 81.3+ MB
```

Remove columns that are missing data. We will see our numbers before removing rows with missing data and then after removing the rows.

In [35]:
```
causes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 288152 entries, 0 to 490937
Data columns (total 36 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   CRASH_RECORD_ID           288152 non-null  object 
 1   RD_NO                     286109 non-null  object 
 2   CRASH_DATE                288152 non-null  object 
 3   POSTED_SPEED_LIMIT        288152 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE    288152 non-null  object 
 5   DEVICE_CONDITION          288152 non-null  object 
 6   WEATHER_CONDITION         288152 non-null  object 
 7   LIGHTING_CONDITION        288152 non-null  object 
 8   FIRST_CRASH_TYPE          288152 non-null  object 
 9   TRAFFICWAY_TYPE           288152 non-null  object 
 10  ALIGNMENT                 288152 non-null  object 
 11  ROADWAY_SURFACE_COND      288152 non-null  object 
 12  ROAD_DEFECT               288152 non-null  object 
 13  REPORT_TYPE               280921 non-null  object 
 14  CRASH_TYPE                288152 non-null  object 
 15  INTERSECTION_RELATED_I    288152 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I        288152 non-null  int64  
 17  HIT_AND_RUN_I             288152 non-null  int64  
 18  DAMAGE                    288152 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE   288152 non-null  object 
 20  STREET_NO                 288152 non-null  int64  
 21  STREET_DIRECTION          288150 non-null  object 
 22  STREET_NAME               288152 non-null  object 
 23  BEAT_OF_OCCURRENCE        288148 non-null  float64
 24  DOORING_I                 288152 non-null  int64  
 25  WORK_ZONE_I               288152 non-null  int64  
 26  WORKERS_PRESENT_I         288152 non-null  int64  
 27  NUM_UNITS                 288152 non-null  int64  
 28  MOST_SEVERE_INJURY        288152 non-null  object 
 29  INJURIES_TOTAL            288152 non-null  float64
 30  CRASH_HOUR                288152 non-null  int64  
 31  CRASH_DAY_OF_WEEK         288152 non-null  int64  
 32  CRASH_MONTH               288152 non-null  int64  
 33  LATITUDE                  286390 non-null  float64
 34  LONGITUDE                 286390 non-null  float64
 35  LOCATION                  286390 non-null  object 
dtypes: float64(4), int64(12), object(20)
memory usage: 81.3+ MB
```

In [36]:
```
causes = causes.dropna()
causes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 277195 entries, 0 to 490937
Data columns (total 36 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   CRASH_RECORD_ID           277195 non-null  object 
 1   RD_NO                     277195 non-null  object 
 2   CRASH_DATE                277195 non-null  object 
 3   POSTED_SPEED_LIMIT        277195 non-null  int64  
 4   TRAFFIC_CONTROL_DEVICE    277195 non-null  object 
 5   DEVICE_CONDITION          277195 non-null  object 
 6   WEATHER_CONDITION         277195 non-null  object 
 7   LIGHTING_CONDITION        277195 non-null  object 
 8   FIRST_CRASH_TYPE          277195 non-null  object 
 9   TRAFFICWAY_TYPE           277195 non-null  object 
 10  ALIGNMENT                 277195 non-null  object 
 11  ROADWAY_SURFACE_COND      277195 non-null  object 
 12  ROAD_DEFECT               277195 non-null  object 
 13  REPORT_TYPE               277195 non-null  object 
 14  CRASH_TYPE                277195 non-null  object 
 15  INTERSECTION_RELATED_I    277195 non-null  int64  
 16  NOT_RIGHT_OF_WAY_I        277195 non-null  int64  
 17  HIT_AND_RUN_I             277195 non-null  int64  
 18  DAMAGE                    277195 non-null  object 
 19  PRIM_CONTRIBUTORY_CAUSE   277195 non-null  object 
 20  STREET_NO                 277195 non-null  int64  
 21  STREET_DIRECTION          277195 non-null  object 
 22  STREET_NAME               277195 non-null  object 
 23  BEAT_OF_OCCURRENCE        277195 non-null  float64
 24  DOORING_I                 277195 non-null  int64  
 25  WORK_ZONE_I               277195 non-null  int64  
 26  WORKERS_PRESENT_I         277195 non-null  int64  
 27  NUM_UNITS                 277195 non-null  int64  
 28  MOST_SEVERE_INJURY        277195 non-null  object 
 29  INJURIES_TOTAL            277195 non-null  float64
 30  CRASH_HOUR                277195 non-null  int64  
 31  CRASH_DAY_OF_WEEK         277195 non-null  int64  
 32  CRASH_MONTH               277195 non-null  int64  
 33  LATITUDE                  277195 non-null  float64
 34  LONGITUDE                 277195 non-null  float64
 35  LOCATION                  277195 non-null  object 
dtypes: float64(4), int64(12), object(20)
memory usage: 78.2+ MB
```

Now we have been able to even out our data so that is easier to work with.

Additional Cleaning by Removal

We will remove additional columns that are not relevant to the question that we are trying to answer. While these columns would be useful to address other questions, they will not benefit our goals. The columns to be removed are:

- Longitude
- Latitude
- Location
- Street Direction
- Street Number
- Crash Date
- Rd No
- Beat of Occurance

In [37]:
```
causes = causes.drop(columns=['LONGITUDE','LATITUDE','LOCATION','STREET_DIRECTION','STREET_NO',
                              'CRASH_DATE', 'RD_NO', 'BEAT_OF_OCCURRENCE'], axis=1)
causes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 277195 entries, 0 to 490937
Data columns (total 28 columns):
 #   Column                   Non-Null Count   Dtype  
---  ------                   --------------   -----  
 0   CRASH_RECORD_ID          277195 non-null  object 
 1   POSTED_SPEED_LIMIT       277195 non-null  int64  
 2   TRAFFIC_CONTROL_DEVICE   277195 non-null  object 
 3   DEVICE_CONDITION         277195 non-null  object 
 4   WEATHER_CONDITION        277195 non-null  object 
 5   LIGHTING_CONDITION       277195 non-null  object 
 6   FIRST_CRASH_TYPE         277195 non-null  object 
 7   TRAFFICWAY_TYPE          277195 non-null  object 
 8   ALIGNMENT                277195 non-null  object 
 9   ROADWAY_SURFACE_COND     277195 non-null  object 
 10  ROAD_DEFECT              277195 non-null  object 
 11  REPORT_TYPE              277195 non-null  object 
 12  CRASH_TYPE               277195 non-null  object 
 13  INTERSECTION_RELATED_I   277195 non-null  int64  
 14  NOT_RIGHT_OF_WAY_I       277195 non-null  int64  
 15  HIT_AND_RUN_I            277195 non-null  int64  
 16  DAMAGE                   277195 non-null  object 
 17  PRIM_CONTRIBUTORY_CAUSE  277195 non-null  object 
 18  STREET_NAME              277195 non-null  object 
 19  DOORING_I                277195 non-null  int64  
 20  WORK_ZONE_I              277195 non-null  int64  
 21  WORKERS_PRESENT_I        277195 non-null  int64  
 22  NUM_UNITS                277195 non-null  int64  
 23  MOST_SEVERE_INJURY       277195 non-null  object 
 24  INJURIES_TOTAL           277195 non-null  float64
 25  CRASH_HOUR               277195 non-null  int64  
 26  CRASH_DAY_OF_WEEK        277195 non-null  int64  
 27  CRASH_MONTH              277195 non-null  int64  
dtypes: float64(1), int64(11), object(16)
memory usage: 61.3+ MB
```

In [38]:
```
causes.describe()
```

Out[38]:

	POSTED_SPEED_LIMIT	INTERSECTION_RELATED_I	NOT_RIGHT_OF_WAY_I	HIT_AND_RUN_I	DOORING_I	WORK_ZONE_I	WORKERS_PRESENT_I	NUM
count	277195.000000	277195.000000	277195.000000	277195.000000	277195.000000	277195.000000	277195.000000	277195.
mean	28.733963	0.266073	0.038854	0.210747	0.001634	0.006302	0.001775	2.
std	6.026797	0.441903	0.193246	0.407840	0.040393	0.079137	0.042093	0.
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.
25%	30.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2.
50%	30.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2.
75%	30.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2.
max	99.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	18.

Working with Categorical Variables

With our now refined and slimmed causes dataset, we can work on fine tuning other items in the set, namely working on the categorical variables present.

Below we will look at the columns that will require adjustments.

In [39]:
```
for col in causes.columns:
    print(causes[col].value_counts())
```

```
6c00c84505b2c5c7af38dc51cf0c5c9726f4b296c56d4a8287fbd9c86e3daeafc2ccb5d5128167202d95ff940d18a8d3d2faa4cbe9b438bf69ee631b6b483a0f    1
e06670da489f2bf499f3cd961188b7d08c4ffb62b9e19d8e8591fb345fa27aba2b7206ea32709c6d12cfc678de5100ab8f6ef98c74bb9a757710e10dec21e9a3    1
5a8b25936ce8412680d731524e2f2d728dd2fd2bd394df9febcb9a0ffa3caaa3ec66eb97d59217a3e97d0eac423e4ef1dd6d0446c9af36573880d4117f7f5560    1
0e1bae401b8ecc5d6055ce9e03d703b613b4c914fe862c6fbfe2e8dad863b633b7da8080e7286eceadde345400c357b64bc575c0e9cd6406b642d2c9681c3116    1
4839abe9fd072b69f636b2a3c86c8a6b39d5d9454ed9594f1014ce41145bc16e52b54316f63117e8df7e8fc3d37e3d66237215434d37aec97758b63f4eb85892    1

..
2e3ecc93216ba8ac25bbe128ac2f3cf6572b73ea3c18236690f5a045d3599f4ddb26e856e48814e31a8f50bd15ec7413a9e7ba1193a499921e0d4e8a03eaac59    1
c4795be894651075b9ed060fdc1505d52c5ae4594981e6f33737eeb262aeee319237b808f0306a45469b3a7dd82bcbf2b997adea0e0c91aa9c9fabe9bb67adc8    1
324e6004361fba5aebfabc9179bee94075ae4d14cd2abc021bcad85629912994664da7bd577ee11969a08c18711d6479c1d54ccb961b2fc54eee9c31bcec8aff    1
beab4aed90b029c921687bf9f73a974d48cfbd649e264b2fdaebd8ccecc68dfd28cafb8dbea6550d8102d67ef8e672a05e3031e029b0d4f6e12b1211293a4650    1
ab8e015b5e310ae266490c3b5f9f12f2efc1a561f724fb23b43c8b3a25bca4b43fe83014d4c373774890438e2584dda75ae454f1f55e516bf398e3c601286cf4    1
Name: CRASH_RECORD_ID, Length: 277195, dtype: int64
30    209449
35     20654
25     14818
20      8864
15      8490
10      4440
0       3246
40      3096
45      1895
5       1713
55       234
50        69
3         61
9         39
39        35
60        14
2         14
1          9
99         9
32         8
33         7
34         5
24         4
11         4
36         3
6          3
70         2
7          2
65         2
12         1
38         1
4          1
49         1
63         1
31         1
Name: POSTED_SPEED_LIMIT, dtype: int64
NO CONTROLS                 142913
TRAFFIC SIGNAL               91477
STOP SIGN/FLASHER            32993
UNKNOWN                       5065
OTHER                         1732
LANE USE MARKING               908
YIELD                          493
OTHER REG. SIGN                358
OTHER WARNING SIGN             331
RAILROAD CROSSING GATE         216
PEDESTRIAN CROSSING SIGN       159
POLICE/FLAGMAN                 131
SCHOOL ZONE                    111
DELINEATORS                     96
FLASHING CONTROL SIGNAL         88
OTHER RAILROAD CROSSING         80
RR CROSSING SIGN                20
NO PASSING                      16
BICYCLE CROSSING SIGN            8
Name: TRAFFIC_CONTROL_DEVICE, dtype: int64
NO CONTROLS                 145541
FUNCTIONING PROPERLY        115988
UNKNOWN                      10499
OTHER                         2254
```

```
FUNCTIONING IMPROPERLY        1758
NOT FUNCTIONING                954
WORN REFLECTIVE MATERIAL       157
MISSING                         44
Name: DEVICE_CONDITION, dtype: int64
CLEAR                       220282
RAIN                         27023
SNOW                         12716
CLOUDY/OVERCAST               9335
UNKNOWN                       5351
OTHER                          944
FOG/SMOKE/HAZE                 533
SLEET/HAIL                     479
FREEZING RAIN/DRIZZLE          361
BLOWING SNOW                   104
SEVERE CROSS WIND GATE          66
BLOWING SAND, SOIL, DIRT         1
Name: WEATHER_CONDITION, dtype: int64
DAYLIGHT                    184476
DARKNESS, LIGHTED ROAD       62776
DARKNESS                     12855
DUSK                          8834
DAWN                          4797
UNKNOWN                       3457
Name: LIGHTING_CONDITION, dtype: int64
REAR END                        75593
TURNING                         47157
SIDESWIPE SAME DIRECTION        45000
PARKED MOTOR VEHICLE            39651
ANGLE                           34980
FIXED OBJECT                    11495
PEDESTRIAN                       6169
SIDESWIPE OPPOSITE DIRECTION     4039
PEDALCYCLIST                     4011
HEAD ON                          2735
OTHER OBJECT                     1991
REAR TO FRONT                    1858
REAR TO SIDE                     1101
OTHER NONCOLLISION                708
REAR TO REAR                      301
ANIMAL                            236
OVERTURNED                        150
TRAIN                              20
Name: FIRST_CRASH_TYPE, dtype: int64
NOT DIVIDED                          123340
DIVIDED - W/MEDIAN (NOT RAISED)       54805
ONE-WAY                               31181
DIVIDED - W/MEDIAN BARRIER            19298
PARKING LOT                           15594
FOUR WAY                              10940
OTHER                                  7720
ALLEY                                  3987
CENTER TURN LANE                       2954
T-INTERSECTION                         2299
UNKNOWN                                1697
RAMP                                    982
DRIVEWAY                                889
UNKNOWN INTERSECTION TYPE               587
FIVE POINT, OR MORE                     274
Y-INTERSECTION                          254
TRAFFIC ROUTE                           213
NOT REPORTED                             91
ROUNDABOUT                               55
L-INTERSECTION                           35
Name: TRAFFICWAY_TYPE, dtype: int64
STRAIGHT AND LEVEL       269134
STRAIGHT ON GRADE          3997
CURVE, LEVEL               2418
STRAIGHT ON HILLCREST      1027
CURVE ON GRADE              465
CURVE ON HILLCREST          154
Name: ALIGNMENT, dtype: int64
DRY                 208640
WET                  42040
SNOW OR SLUSH        12357
UNKNOWN              10747
ICE                   2662
OTHER                  621
SAND, MUD, DIRT        128
Name: ROADWAY_SURFACE_COND, dtype: int64
NO DEFECTS           237729
UNKNOWN               33270
RUT, HOLES             2366
OTHER                  1747
WORN SURFACE           1162
SHOULDER DEFECT         655
DEBRIS ON ROADWAY       266
```

```
Name: ROAD_DEFECT, dtype: int64
NOT ON SCENE (DESK REPORT)    147142
ON SCENE                      129893
AMENDED                          160
Name: REPORT_TYPE, dtype: int64
NO INJURY / DRIVE AWAY              196813
INJURY AND / OR TOW DUE TO CRASH     80382
Name: CRASH_TYPE, dtype: int64
0    203441
1     73754
Name: INTERSECTION_RELATED_I, dtype: int64
0    266425
1     10770
Name: NOT_RIGHT_OF_WAY_I, dtype: int64
0    218777
1     58418
Name: HIT_AND_RUN_I, dtype: int64
OVER $1,500      167753
$501 - $1,500     75044
$500 OR LESS      34398
Name: DAMAGE, dtype: int64
FAILING TO YIELD RIGHT-OF-WAY                                                52127
FOLLOWING TOO CLOSELY                                                        50757
IMPROPER OVERTAKING/PASSING                                                  22572
IMPROPER BACKING                                                             21001
FAILING TO REDUCE SPEED TO AVOID CRASH                                       20823
IMPROPER LANE USAGE                                                          18557
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          15769
IMPROPER TURNING/NO SIGNAL                                                   15751
WEATHER                                                                       8722
DISREGARDING TRAFFIC SIGNALS                                                  8520
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER     5956
DISREGARDING STOP SIGN                                                        5239
DISTRACTION - FROM INSIDE VEHICLE                                             3530
EQUIPMENT - VEHICLE CONDITION                                                 2990
PHYSICAL CONDITION OF DRIVER                                                  2987
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2863
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2496
DRIVING ON WRONG SIDE/WRONG WAY                                               2203
DISTRACTION - FROM OUTSIDE VEHICLE                                            2130
EXCEEDING AUTHORIZED SPEED LIMIT                                              1946
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1675
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                      1382
ROAD CONSTRUCTION/MAINTENANCE                                                 1212
DISREGARDING OTHER TRAFFIC SIGNS                                              1032
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                              891
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                745
CELL PHONE USE OTHER THAN TEXTING                                              679
DISREGARDING ROAD MARKINGS                                                     675
ANIMAL                                                                         469
TURNING RIGHT ON RED                                                           339
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    235
TEXTING                                                                        215
RELATED TO BUS STOP                                                            210
DISREGARDING YIELD SIGN                                                        208
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         145
PASSING STOPPED SCHOOL BUS                                                      68
OBSTRUCTED CROSSWALKS                                                           40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
WESTERN AVE     7890
PULASKI RD      6801
ASHLAND AVE     6393
CICERO AVE      6285
HALSTED ST      5786
                ...
94TH PL            1
STONE ST           1
WILDWOOD AVE       1
MC ALPIN AVE       1
ELSTON PKWY        1
Name: STREET_NAME, Length: 1417, dtype: int64
0    276742
1       453
Name: DOORING_I, dtype: int64
0    275448
1      1747
Name: WORK_ZONE_I, dtype: int64
0    276703
1       492
Name: WORKERS_PRESENT_I, dtype: int64
2     243951
3      16198
1      12955
4       3032
5        702
6        224
```

```
7         69
8         33
9         16
10         7
11         4
12         2
18         1
15         1
Name: NUM_UNITS, dtype: int64
NO INDICATION OF INJURY     234898
NONINCAPACITATING INJURY     23595
REPORTED, NOT EVIDENT        13177
INCAPACITATING INJURY         5292
FATAL                          233
Name: MOST_SEVERE_INJURY, dtype: int64
0.0     234898
1.0      31380
2.0       7343
3.0       2248
4.0        812
5.0        309
6.0        125
7.0         41
8.0         14
9.0          9
10.0         5
11.0         5
21.0         2
12.0         1
13.0         1
15.0         1
19.0         1
Name: INJURIES_TOTAL, dtype: int64
16    21703
17    21465
15    21256
14    18685
18    17525
13    16956
12    16074
8     15336
11    14468
9     13101
10    12885
19    12811
7     12214
20    10109
21     8896
22     8229
23     6916
6      6054
0      5300
1      4282
2      3782
5      3422
3      2983
4      2743
Name: CRASH_HOUR, dtype: int64
6    45759
7    40814
5    40411
4    39829
3    39752
2    38113
1    32517
Name: CRASH_DAY_OF_WEEK, dtype: int64
10    26595
12    25581
11    24703
9     24533
8     23770
1     23679
2     23227
7     22530
3     21828
6     21404
5     21149
4     18196
Name: CRASH_MONTH, dtype: int64
```

Before making our changes we will make a copy of our causes database in case we need to go back and redo our work. We will name this as cat_data.

In [40]:
```
cat_data = causes.copy()
```

Binary Columns

Binary Columns

There are some columns that be best slimmed down into binary columns for ease of use. Those columns include alignment of the road, the weather condition and physical condition, and whether a crash resulted in an injury or not.

Road Alignment Binary

We will create a binary column where if the road is straight, it will receive a value of 1 and if not will receive a 0

In [41]:
```
cat_data['STRAIGHT_ALIGNMENT'] = cat_data['ALIGNMENT'].map(lambda x: 1 if 'STRAIGHT' in x else 0)
cat_data = cat_data.drop(['ALIGNMENT'], axis=1)
cat_data['STRAIGHT_ALIGNMENT'].value_counts()
```

Out[41]:
```
1    274158
0      3037
Name: STRAIGHT_ALIGNMENT, dtype: int64
```

Roadway Surface Condition

We will consolidate the roadway conditions. We will perform two different steps for this column. We will combine the Dry and Unknown sections since we have no way of knowing what the actual conditions were in 'Unknown' however since a majority of the conditions are dry, it can be safe to assume that we can concat both. Also, we will be assigning 0 if there was a roadway surface condition other than Dry and 1 for Dry.

In [42]:
```
cat_data['DRY_ROADWAY_SURFACE_COND'] = cat_data['ROADWAY_SURFACE_COND'].map(lambda x: 1 if ('DRY' in x or 'UNKNOWN' in
cat_data = cat_data.drop(['ROADWAY_SURFACE_COND'], axis = 1)
cat_data['DRY_ROADWAY_SURFACE_COND'].value_counts()
```

Out[42]:
```
1    219387
0     57808
Name: DRY_ROADWAY_SURFACE_COND, dtype: int64
```

Road Defects

We will consolidate all the road defects to just highlight whether the road where the incident occured had any defects present. Similar to Roadway Surface Conditions we will group unknown with no defects for simplicity and since the majority of our data represents no road defects present.

In [43]:
```
cat_data['ROAD_DEFECT'] = cat_data['ROAD_DEFECT'].map(lambda x: 0 if ('NO DEFECTS' in x or 'UNKNOWN' in x ) else 1)
cat_data['ROAD_DEFECT'].value_counts()
```

Out[43]:
```
0    270999
1      6196
Name: ROAD_DEFECT, dtype: int64
```

Crash Type

We will differeniate wheter our incident has no injury and assign it 1, and if it did result in an injury it will be assigned 0. This will help in our analysis later as well.

In [44]:
```
cat_data['NO_INJ_CRASH_TYPE'] = cat_data['CRASH_TYPE'].map(lambda x: 1 if ('NO INJURY' in x ) else 0)
cat_data = cat_data.drop(['CRASH_TYPE'], axis = 1)
cat_data['NO_INJ_CRASH_TYPE'].value_counts()
```

Out[44]:
```
1    196813
0     80382
Name: NO_INJ_CRASH_TYPE, dtype: int64
```

Traffice Control Device

There are a number of traffic control devices outlined in our data. In order to simplify our columns, it will be easier to consolidate our devices to encompass 3 distinct catagories:

- Signals
- Signs
- Other

While we will take some liberties as to which gets catagorized into each, we can safely say that each catagory in general will contain the devices that are appropriate for that catagory.

In [45]:
```
cat_data['TRAFFIC_CONTROL_DEVICE'].value_counts()
```

Out[45]:
```
NO CONTROLS                 142913
TRAFFIC SIGNAL               91477
STOP SIGN/FLASHER            32993
UNKNOWN                       5065
OTHER                         1732
LANE USE MARKING               908
YIELD                          493
OTHER REG. SIGN                358
OTHER WARNING SIGN             331
RAILROAD CROSSING GATE         216
PEDESTRIAN CROSSING SIGN       159
POLICE/FLAGMAN                 131
SCHOOL ZONE                    111
DELINEATORS                     96
FLASHING CONTROL SIGNAL         88
OTHER RAILROAD CROSSING         80
RR CROSSING SIGN                20
NO PASSING                      16
BICYCLE CROSSING SIGN            8
Name: TRAFFIC_CONTROL_DEVICE, dtype: int64
```

Placing relevant variables into 'Signal'

In [46]:
```
cat_data['TRAFFIC_CONTROL_DEVICE'] = cat_data['TRAFFIC_CONTROL_DEVICE'].apply(lambda x: 'SIGNAL' if 'SIGNAL' in x els
cat_data['TRAFFIC_CONTROL_DEVICE'].value_counts()
```

Out[46]:
```
NO CONTROLS                 142913
SIGNAL                       91565
STOP SIGN/FLASHER            32993
UNKNOWN                       5065
OTHER                         1732
LANE USE MARKING               908
YIELD                          493
OTHER REG. SIGN                358
OTHER WARNING SIGN             331
RAILROAD CROSSING GATE         216
PEDESTRIAN CROSSING SIGN       159
POLICE/FLAGMAN                 131
SCHOOL ZONE                    111
DELINEATORS                     96
OTHER RAILROAD CROSSING         80
RR CROSSING SIGN                20
NO PASSING                      16
BICYCLE CROSSING SIGN            8
Name: TRAFFIC_CONTROL_DEVICE, dtype: int64
```

Placing relevant variables into 'Sign'

In [47]:
```
cat_data['TRAFFIC_CONTROL_DEVICE'] = cat_data['TRAFFIC_CONTROL_DEVICE'].apply(lambda x: 'SIGN' if ' SIGN' in x else x)
cat_data['TRAFFIC_CONTROL_DEVICE'].value_counts()
```

Out[47]:
```
NO CONTROLS                 142913
SIGNAL                       91565
SIGN                         33869
UNKNOWN                       5065
OTHER                         1732
LANE USE MARKING               908
YIELD                          493
RAILROAD CROSSING GATE         216
POLICE/FLAGMAN                 131
SCHOOL ZONE                    111
DELINEATORS                     96
OTHER RAILROAD CROSSING         80
NO PASSING                      16
Name: TRAFFIC_CONTROL_DEVICE, dtype: int64
```

Creating a dictonary by which we can assign the aforementioned traffic devices into their respective sign, signal, and other categories by assigning the appropriate keys and values. We will also introduce one final element in order to further simplify our data, which is to create a seperate element for no control.

In [48]:
```
traffic_dict = {'NO CONTROLS': 'NO CONTROLS','SIGN': 'SIGN', 'UNKNOWN': 'OTHER',
                'OTHER': 'OTHER', 'YIELD':'SIGN', 'RAILROAD CROSSING GATE': 'SIGNAL',
                'POLICE/FLAGMAN': 'OTHER', 'DELINEATORS': 'OTHER', 'SCHOOL ZONE': 'SIGN',
                'OTHER RAILROAD CROSSING': 'SIGNAL', 'NO PASSING': 'SIGN', 'SIGNAL':'SIGNAL'}
cat_data['TRAFFIC_CONTROL_DEVICE'] = cat_data['TRAFFIC_CONTROL_DEVICE'].map(traffic_dict)
cat_data['TRAFFIC_CONTROL_DEVICE'].value_counts()
```

Out[48]:
```
NO CONTROLS    142913
SIGNAL          91861
SIGN            34489
OTHER            7024
Name: TRAFFIC_CONTROL_DEVICE, dtype: int64
```

Device Condition

We will follow the same method as we did previously with Traffic Control device. We will group Device condition to three groups:

- No Controls
- Functioning Properly
- Functioning Improperly/Missing

It is important to remember that the condition of the traffic device may play an important role as to why severe and fatal accidents occur in Chicago. We will keep this factor in mind for our other following columns such as Weather Condition and Lighting

In [49]:
```
cat_data['DEVICE_CONDITION'].value_counts()
```

Out[49]:
```
NO CONTROLS                 145541
FUNCTIONING PROPERLY        115988
UNKNOWN                      10499
OTHER                         2254
FUNCTIONING IMPROPERLY        1758
NOT FUNCTIONING                954
WORN REFLECTIVE MATERIAL       157
MISSING                         44
Name: DEVICE_CONDITION, dtype: int64
```

In [50]:
```
device_dict = {'NO CONTROLS': 'NO CONTROLS',
               'FUNCTIONING PROPERLY': 'FUNCTIONING PROPERLY',
               'UNKNOWN': 'NO CONTROLS',
               'FUNCTIONING IMPROPERLY':'FUNCTIONING IMPROPERLY/ MISSING',
               'NOT FUNCTIONING':'FUNCTIONING IMPROPERLY/ MISSING',
               'WORN REFLECTIVE MATERIAL': 'FUNCTIONING PROPERLY',
               'MISSING': 'FUNCTIONING IMPROPERLY/ MISSING',
               'OTHER': 'FUNCTIONING PROPERLY'}
cat_data['DEVICE_CONDITION'] = cat_data['DEVICE_CONDITION'].map(device_dict)
cat_data['DEVICE_CONDITION'].value_counts()
```

Out[50]:
```
NO CONTROLS                       156040
FUNCTIONING PROPERLY              118399
FUNCTIONING IMPROPERLY/ MISSING     2756
Name: DEVICE_CONDITION, dtype: int64
```

Weather Conditions

We will split this column into Clear, Precipitation or Other

In [51]:
```
cat_data['WEATHER_CONDITION'].value_counts()
```

Out[51]:
```
CLEAR                      220282
RAIN                        27023
SNOW                        12716
CLOUDY/OVERCAST              9335
UNKNOWN                      5351
OTHER                         944
FOG/SMOKE/HAZE                533
SLEET/HAIL                    479
FREEZING RAIN/DRIZZLE         361
BLOWING SNOW                  104
SEVERE CROSS WIND GATE         66
BLOWING SAND, SOIL, DIRT        1
Name: WEATHER_CONDITION, dtype: int64
```

In [52]:
```
weather_dict = {'CLEAR': 'CLEAR', 'RAIN':'PRECIPITATION','SNOW':'PRECIPITATION',
                'CLOUDY/OVERCAST': 'OTHER', 'OTHER':'CLEAR', 'FREEZING RAIN/DRIZZLE':'PRECIPITATION',
                'OTHER':'OTHER', 'FOG/SMOKE/HAZE':'OTHER', 'SLEET/HAIL':'PRECIPITATION', 'BLOWING SNOW':'PRECIPITATIO
                'SEVERE CROSS WIND GATE':'OTHER', 'UNKNOWN': 'CLEAR'}

cat_data['WEATHER_CONDITION'] = cat_data['WEATHER_CONDITION'].map(weather_dict)
cat_data['WEATHER_CONDITION'].value_counts()
```

Out[52]:
```
CLEAR            225633
PRECIPITATION     40683
OTHER             10878
Name: WEATHER_CONDITION, dtype: int64
```

Lighting

We will divide this column into Light, Variable Light, and Darkness. We will group the unknown with 'Light' since it is the majority in our dataset.

In [53]:
```
cat_data['LIGHTING_CONDITION'].value_counts()
```

Out[53]:
```
DAYLIGHT                  184476
DARKNESS, LIGHTED ROAD     62776
DARKNESS                   12855
DUSK                        8834
DAWN                        4797
UNKNOWN                     3457
Name: LIGHTING_CONDITION, dtype: int64
```

In [54]:
```
lighting_dict = {'DAYLIGHT':'LIGHT','DARKNESS, LIGHTED ROAD':'VARIABLE LIGHT', 'DARKNESS':'DARKNESS',
                 'DUSK':'VARIABLE LIGHT', 'DAWN': 'VARIABLE LIGHT', 'UNKNOWN': 'LIGHT'}
cat_data['LIGHTING_CONDITION'] = cat_data['LIGHTING_CONDITION'].map(lighting_dict)
cat_data['LIGHTING_CONDITION'].value_counts()
```

Out[54]:
```
LIGHT            187933
VARIABLE LIGHT    76407
DARKNESS          12855
Name: LIGHTING_CONDITION, dtype: int64
```

Accident Data Columns

In this section we will work on consolidating some of our additional accident data columns that highlight points such as the type of accident, the type of intersection it occured at, the amount of damage involved and the severity of injuries. This will, just like our previous work help streamline our data for modeling.

First Crash Type

We will consolidate our data to remove some points that can be considered similar or redunant in nature such as SIDEWIPE and SIDESWIPE IN OPPOSITE DIRECTION. Below are the categories we are originally given and their consolidation will follow.

In [55]:
```
cat_data['FIRST_CRASH_TYPE'].value_counts()
```

Out[55]:
```
REAR END                       75593
TURNING                        47157
SIDESWIPE SAME DIRECTION       45000
PARKED MOTOR VEHICLE           39651
ANGLE                          34980
FIXED OBJECT                   11495
PEDESTRIAN                      6169
SIDESWIPE OPPOSITE DIRECTION    4039
PEDALCYCLIST                    4011
HEAD ON                         2735
OTHER OBJECT                    1991
REAR TO FRONT                   1858
REAR TO SIDE                    1101
OTHER NONCOLLISION               708
REAR TO REAR                     301
ANIMAL                           236
OVERTURNED                       150
TRAIN                             20
Name: FIRST_CRASH_TYPE, dtype: int64
```

In [56]:
```
cat_data['FIRST_CRASH_TYPE'] = cat_data['FIRST_CRASH_TYPE'].apply(lambda x: 'SIDESWIPE' if 'SIDESWIPE' in x else x)
cat_data['FIRST_CRASH_TYPE'].value_counts()
```

Out[56]:
```
REAR END                75593
SIDESWIPE               49039
TURNING                 47157
PARKED MOTOR VEHICLE    39651
ANGLE                   34980
FIXED OBJECT            11495
PEDESTRIAN               6169
PEDALCYCLIST             4011
HEAD ON                  2735
OTHER OBJECT             1991
REAR TO FRONT            1858
REAR TO SIDE             1101
OTHER NONCOLLISION        708
REAR TO REAR              301
ANIMAL                    236
OVERTURNED                150
TRAIN                      20
Name: FIRST_CRASH_TYPE, dtype: int64
```

In [57]:
```
cat_data['FIRST_CRASH_TYPE'] = cat_data['FIRST_CRASH_TYPE'].apply(lambda x: 'PERSON/ANIMAL' if
                                                                  ('PED' in x or 'ANIMAL' in x) else x)
cat_data['FIRST_CRASH_TYPE'].value_counts()
```

Out[57]:
```
REAR END                75593
SIDESWIPE               49039
TURNING                 47157
PARKED MOTOR VEHICLE    39651
ANGLE                   34980
FIXED OBJECT            11495
PERSON/ANIMAL           10416
HEAD ON                  2735
OTHER OBJECT             1991
REAR TO FRONT            1858
REAR TO SIDE             1101
OTHER NONCOLLISION        708
REAR TO REAR              301
OVERTURNED                150
TRAIN                      20
Name: FIRST_CRASH_TYPE, dtype: int64
```

In [58]:
```
cat_data['FIRST_CRASH_TYPE'] = cat_data['FIRST_CRASH_TYPE'].apply(lambda x: 'PARKED CAR/OBJECT'if
                                                                  ('PARKED' in x or 'OBJECT' in x) else x)
cat_data['FIRST_CRASH_TYPE'].value_counts()
```

Out[58]:
```
REAR END              75593
PARKED CAR/OBJECT     53137
SIDESWIPE             49039
TURNING               47157
ANGLE                 34980
PERSON/ANIMAL         10416
HEAD ON                2735
REAR TO FRONT          1858
REAR TO SIDE           1101
OTHER NONCOLLISION      708
REAR TO REAR            301
OVERTURNED              150
TRAIN                    20
Name: FIRST_CRASH_TYPE, dtype: int64
```

Most Severe Injury

We will consolidate our data on the most severe injuries per crash into 4 catagories:

- No Injury
- Minor Injuries
- Major Injuries
- Fatal

We will accomplish this by setting each injury in their respective place within an injury dictionary.

In [59]:
```
cat_data['MOST_SEVERE_INJURY'].value_counts()
```

Out[59]:
```
NO INDICATION OF INJURY     234898
NONINCAPACITATING INJURY     23595
REPORTED, NOT EVIDENT        13177
INCAPACITATING INJURY         5292
FATAL                          233
Name: MOST_SEVERE_INJURY, dtype: int64
```

In [60]:
```
injury_dict = {'NO INDICATION OF INJURY' : 'NO INJURY', 'NONINCAPACITATING INJURY': 'MINOR INJURIES',
               'REPORTED, NOT EVIDENT': 'MINOR INJURIES', 'INCAPACITATING INJURY' : 'MAJOR INJURIES',
               'FATAL': 'FATAL'}

cat_data['MOST_SEVERE_INJURY'] = cat_data['MOST_SEVERE_INJURY'].map(injury_dict)
cat_data['MOST_SEVERE_INJURY'].value_counts()
```

Out[60]:
```
NO INJURY         234898
MINOR INJURIES     36772
MAJOR INJURIES      5292
FATAL                233
Name: MOST_SEVERE_INJURY, dtype: int64
```

Trafficway Type

Simialar to First Crash Type we will consolidate some categories that can be considered similar or redundant.

In [61]:
```
cat_data['TRAFFICWAY_TYPE'].value_counts()
```

Out[61]:
```
NOT DIVIDED                         123340
DIVIDED - W/MEDIAN (NOT RAISED)      54805
ONE-WAY                              31181
DIVIDED - W/MEDIAN BARRIER           19298
PARKING LOT                          15594
FOUR WAY                             10940
OTHER                                 7720
ALLEY                                 3987
CENTER TURN LANE                      2954
T-INTERSECTION                        2299
UNKNOWN                               1697
RAMP                                   982
DRIVEWAY                               889
UNKNOWN INTERSECTION TYPE              587
FIVE POINT, OR MORE                    274
Y-INTERSECTION                         254
TRAFFIC ROUTE                          213
NOT REPORTED                            91
ROUNDABOUT                              55
L-INTERSECTION                          35
Name: TRAFFICWAY_TYPE, dtype: int64
```

In [62]:
```
cat_data['TRAFFICWAY_TYPE'] = cat_data['TRAFFICWAY_TYPE'].apply(lambda x:
                                                                'INTERSECTION' if 'INTERSECTION' in x else x)
cat_data['TRAFFICWAY_TYPE'].value_counts()
```

Out[62]:
```
NOT DIVIDED                         123340
DIVIDED - W/MEDIAN (NOT RAISED)      54805
ONE-WAY                              31181
DIVIDED - W/MEDIAN BARRIER           19298
PARKING LOT                          15594
FOUR WAY                             10940
OTHER                                 7720
ALLEY                                 3987
INTERSECTION                          3175
CENTER TURN LANE                      2954
UNKNOWN                               1697
RAMP                                   982
DRIVEWAY                               889
FIVE POINT, OR MORE                    274
TRAFFIC ROUTE                          213
NOT REPORTED                            91
ROUNDABOUT                              55
Name: TRAFFICWAY_TYPE, dtype: int64
```

In [63]:
```
intersect_dict = {'ROUNDABOUT': 'INTERSECTION', 'FIVE POINT, OR MORE': 'INTERSECTION',
                  'UNKNOWN':'NOT DIVIDED', 'NOT REPORTED': 'NOT DIVIDED'}

cat_data['TRAFFICWAY_TYPE'] = cat_data['TRAFFICWAY_TYPE'].map(intersect_dict).fillna(cat_data['TRAFFICWAY_TYPE'])
cat_data['TRAFFICWAY_TYPE'].value_counts()
```

Out[63]:
```
NOT DIVIDED                        125128
DIVIDED - W/MEDIAN (NOT RAISED)     54805
ONE-WAY                             31181
DIVIDED - W/MEDIAN BARRIER          19298
PARKING LOT                         15594
FOUR WAY                            10940
OTHER                                7720
ALLEY                                3987
INTERSECTION                         3504
CENTER TURN LANE                     2954
RAMP                                  982
DRIVEWAY                              889
TRAFFIC ROUTE                         213
Name: TRAFFICWAY_TYPE, dtype: int64
```

Damage

For the column damage we will do some cleaning up by removing some placemarkers and making our data easier to use.

In [64]:
```
cat_data['DAMAGE'].value_counts()
```

Out[64]:
```
OVER $1,500      167753
$501 - $1,500     75044
$500 OR LESS      34398
Name: DAMAGE, dtype: int64
```

In [65]:
```
cat_data['DAMAGE_DOLLARS'] = cat_data['DAMAGE'].map(lambda x:
                                                    x.replace('$','').replace(',',"").replace('-','TO'))
cat_data = cat_data.drop(columns = ['DAMAGE'])
cat_data['DAMAGE_DOLLARS'].value_counts()
```

Out[65]:
```
OVER 1500      167753
501 TO 1500     75044
500 OR LESS     34398
Name: DAMAGE_DOLLARS, dtype: int64
```

Review our columns after our changes

In [66]:
```
cat_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 277195 entries, 0 to 490937
Data columns (total 28 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   CRASH_RECORD_ID           277195 non-null  object
 1   POSTED_SPEED_LIMIT        277195 non-null  int64
 2   TRAFFIC_CONTROL_DEVICE    276287 non-null  object
 3   DEVICE_CONDITION          277195 non-null  object
 4   WEATHER_CONDITION         277194 non-null  object
 5   LIGHTING_CONDITION        277195 non-null  object
 6   FIRST_CRASH_TYPE          277195 non-null  object
 7   TRAFFICWAY_TYPE           277195 non-null  object
 8   ROAD_DEFECT               277195 non-null  int64
 9   REPORT_TYPE               277195 non-null  object
 10  INTERSECTION_RELATED_I    277195 non-null  int64
 11  NOT_RIGHT_OF_WAY_I        277195 non-null  int64
 12  HIT_AND_RUN_I             277195 non-null  int64
 13  PRIM_CONTRIBUTORY_CAUSE   277195 non-null  object
 14  STREET_NAME               277195 non-null  object
 15  DOORING_I                 277195 non-null  int64
 16  WORK_ZONE_I               277195 non-null  int64
 17  WORKERS_PRESENT_I         277195 non-null  int64
 18  NUM_UNITS                 277195 non-null  int64
 19  MOST_SEVERE_INJURY        277195 non-null  object
 20  INJURIES_TOTAL            277195 non-null  float64
 21  CRASH_HOUR                277195 non-null  int64
 22  CRASH_DAY_OF_WEEK         277195 non-null  int64
 23  CRASH_MONTH               277195 non-null  int64
 24  STRAIGHT_ALIGNMENT        277195 non-null  int64
 25  DRY_ROADWAY_SURFACE_COND  277195 non-null  int64
 26  NO_INJ_CRASH_TYPE         277195 non-null  int64
 27  DAMAGE_DOLLARS            277195 non-null  object
dtypes: float64(1), int64(15), object(12)
memory usage: 61.3+ MB
```

In [67]:
```
1 cat_data.head()
```

Out[67]:

	CRASH_RECORD_ID	POSTED_SPEED_LIMIT	TRAFFIC_CONTROL_DEVICE	DEVICE_CONDITION	WEATHER_CONDITION	LIGHTI
0	4fd0a3e0897b3335b94cd8d5b2d2b350eb691add56c62d...	35	NO CONTROLS	NO CONTROLS	CLEAR	
1	009e9e67203442370272e1a13d6ee51a4155dac65e583d...	35	SIGN	FUNCTIONING PROPERLY	CLEAR	
2	ee9283eff3a55ac50ee58f3d9528ce1d689b1c4180b4c4...	30	SIGNAL	FUNCTIONING PROPERLY	CLEAR	
7	f636d4a51a88015ac89031159b1f1952b8d92e49d11aeb...	30	NO CONTROLS	NO CONTROLS	CLEAR	
10	0209e21f298984f7375742b7ef27c9880b485f41123a12...	30	SIGNAL	FUNCTIONING PROPERLY	CLEAR	

Classification of Primary Contributory Cause

In order to have a more efficient model it is important to consolidate the causes of accidents to a number that is manageable. We will combine the causes based on their similarities in factors such as external influences, reckless driving behavior and aggressive driving. Such as the case was for our previous data, we will be working of a copy of our cat_data dataset in case we need to recall it later on and to preserve the work that we have done so far to clean our data. We are a few steps from having our data set for modeling!

In [68]:
```
1 prime_class = cat_data.copy()
```

In [69]:
```
1 prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[69]:
```
FAILING TO YIELD RIGHT-OF-WAY                                                52127
FOLLOWING TOO CLOSELY                                                        50757
IMPROPER OVERTAKING/PASSING                                                  22572
IMPROPER BACKING                                                             21001
FAILING TO REDUCE SPEED TO AVOID CRASH                                       20823
IMPROPER LANE USAGE                                                          18557
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          15769
IMPROPER TURNING/NO SIGNAL                                                   15751
WEATHER                                                                       8722
DISREGARDING TRAFFIC SIGNALS                                                  8520
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER  5956
DISREGARDING STOP SIGN                                                        5239
DISTRACTION - FROM INSIDE VEHICLE                                             3530
EQUIPMENT - VEHICLE CONDITION                                                 2990
PHYSICAL CONDITION OF DRIVER                                                  2987
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2863
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2496
DRIVING ON WRONG SIDE/WRONG WAY                                               2203
DISTRACTION - FROM OUTSIDE VEHICLE                                            2130
EXCEEDING AUTHORIZED SPEED LIMIT                                              1946
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1675
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                      1382
ROAD CONSTRUCTION/MAINTENANCE                                                 1212
DISREGARDING OTHER TRAFFIC SIGNS                                              1032
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                              891
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                745
CELL PHONE USE OTHER THAN TEXTING                                              679
DISREGARDING ROAD MARKINGS                                                     675
ANIMAL                                                                         469
TURNING RIGHT ON RED                                                           339
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    235
TEXTING                                                                        215
RELATED TO BUS STOP                                                            210
DISREGARDING YIELD SIGN                                                        208
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         145
PASSING STOPPED SCHOOL BUS                                                      68
OBSTRUCTED CROSSWALKS                                                           40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Creating DISREGARDING SIGNS/SIGNALS

In [70]:
```
prime_class['PRIM_CONTRIBUTORY_CAUSE'] = prime_class['PRIM_CONTRIBUTORY_CAUSE'].apply(lambda x:
                                                                                      'DISREGARDING SIGNS/SIGNALS' i
                                                                                      'DISREGARDING' in x else x)

prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[70]:
```
FAILING TO YIELD RIGHT-OF-WAY                                                52127
FOLLOWING TOO CLOSELY                                                        50757
IMPROPER OVERTAKING/PASSING                                                  22572
IMPROPER BACKING                                                             21001
FAILING TO REDUCE SPEED TO AVOID CRASH                                       20823
IMPROPER LANE USAGE                                                          18557
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          15769
IMPROPER TURNING/NO SIGNAL                                                   15751
DISREGARDING SIGNS/SIGNALS                                                   15674
WEATHER                                                                       8722
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER     5956
DISTRACTION - FROM INSIDE VEHICLE                                             3530
EQUIPMENT - VEHICLE CONDITION                                                 2990
PHYSICAL CONDITION OF DRIVER                                                  2987
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2863
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2496
DRIVING ON WRONG SIDE/WRONG WAY                                               2203
DISTRACTION - FROM OUTSIDE VEHICLE                                            2130
EXCEEDING AUTHORIZED SPEED LIMIT                                              1946
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1675
ROAD ENGINEERING/SURFACE/MARKING DEFECTS                                      1382
ROAD CONSTRUCTION/MAINTENANCE                                                 1212
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                              891
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                745
CELL PHONE USE OTHER THAN TEXTING                                              679
ANIMAL                                                                         469
TURNING RIGHT ON RED                                                           339
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    235
TEXTING                                                                        215
RELATED TO BUS STOP                                                            210
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         145
PASSING STOPPED SCHOOL BUS                                                      68
OBSTRUCTED CROSSWALKS                                                           40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Combining and Creating External Factors

In [71]:
```
prime_class['PRIM_CONTRIBUTORY_CAUSE'] = prime_class['PRIM_CONTRIBUTORY_CAUSE'].apply(lambda x:
                                                                                      'EXTERNAL FACTORS' if
                                                                                      'ROAD' in x else x)
prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[71]:
```
FAILING TO YIELD RIGHT-OF-WAY                                                52127
FOLLOWING TOO CLOSELY                                                        50757
IMPROPER OVERTAKING/PASSING                                                  22572
IMPROPER BACKING                                                             21001
FAILING TO REDUCE SPEED TO AVOID CRASH                                       20823
IMPROPER LANE USAGE                                                          18557
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          15769
IMPROPER TURNING/NO SIGNAL                                                   15751
DISREGARDING SIGNS/SIGNALS                                                   15674
WEATHER                                                                       8722
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER     5956
DISTRACTION - FROM INSIDE VEHICLE                                             3530
EQUIPMENT - VEHICLE CONDITION                                                 2990
PHYSICAL CONDITION OF DRIVER                                                  2987
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2863
EXTERNAL FACTORS                                                              2594
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2496
DRIVING ON WRONG SIDE/WRONG WAY                                               2203
DISTRACTION - FROM OUTSIDE VEHICLE                                            2130
EXCEEDING AUTHORIZED SPEED LIMIT                                              1946
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1675
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST                              891
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                745
CELL PHONE USE OTHER THAN TEXTING                                              679
ANIMAL                                                                         469
TURNING RIGHT ON RED                                                           339
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    235
TEXTING                                                                        215
RELATED TO BUS STOP                                                            210
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         145
PASSING STOPPED SCHOOL BUS                                                      68
OBSTRUCTED CROSSWALKS                                                           40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

In [72]:
```
prime_class['PRIM_CONTRIBUTORY_CAUSE'] = prime_class['PRIM_CONTRIBUTORY_CAUSE'].apply(lambda x:
                                                                                      'EXTERNAL FACTORS' if
                                                                                      'ANIMAL' in x else x)
prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[72]:
```
FAILING TO YIELD RIGHT-OF-WAY                                                52127
FOLLOWING TOO CLOSELY                                                        50757
IMPROPER OVERTAKING/PASSING                                                  22572
IMPROPER BACKING                                                             21001
FAILING TO REDUCE SPEED TO AVOID CRASH                                       20823
IMPROPER LANE USAGE                                                          18557
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          15769
IMPROPER TURNING/NO SIGNAL                                                   15751
DISREGARDING SIGNS/SIGNALS                                                   15674
WEATHER                                                                       8722
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER     5956
EXTERNAL FACTORS                                                              3954
DISTRACTION - FROM INSIDE VEHICLE                                             3530
EQUIPMENT - VEHICLE CONDITION                                                 2990
PHYSICAL CONDITION OF DRIVER                                                  2987
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2863
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2496
DRIVING ON WRONG SIDE/WRONG WAY                                               2203
DISTRACTION - FROM OUTSIDE VEHICLE                                            2130
EXCEEDING AUTHORIZED SPEED LIMIT                                              1946
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1675
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                745
CELL PHONE USE OTHER THAN TEXTING                                              679
TURNING RIGHT ON RED                                                           339
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    235
TEXTING                                                                        215
RELATED TO BUS STOP                                                            210
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         145
PASSING STOPPED SCHOOL BUS                                                      68
OBSTRUCTED CROSSWALKS                                                           40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Combining Improper Driving Behaviors

In [73]:
```
prime_class['PRIM_CONTRIBUTORY_CAUSE'] = prime_class['PRIM_CONTRIBUTORY_CAUSE'].apply(lambda x:
                                                                                      'IMPROPER' if
                                                                                      'IMPROPER' in x else x)
prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[73]:
```
IMPROPER                                                                     77881
FAILING TO YIELD RIGHT-OF-WAY                                                52127
FOLLOWING TOO CLOSELY                                                        50757
FAILING TO REDUCE SPEED TO AVOID CRASH                                       20823
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                                          15769
DISREGARDING SIGNS/SIGNALS                                                   15674
WEATHER                                                                       8722
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER     5956
EXTERNAL FACTORS                                                              3954
DISTRACTION - FROM INSIDE VEHICLE                                             3530
EQUIPMENT - VEHICLE CONDITION                                                 2990
PHYSICAL CONDITION OF DRIVER                                                  2987
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)                          2863
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)            2496
DRIVING ON WRONG SIDE/WRONG WAY                                               2203
DISTRACTION - FROM OUTSIDE VEHICLE                                            2130
EXCEEDING AUTHORIZED SPEED LIMIT                                              1946
EXCEEDING SAFE SPEED FOR CONDITIONS                                           1675
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)                                745
CELL PHONE USE OTHER THAN TEXTING                                              679
TURNING RIGHT ON RED                                                           339
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)    235
TEXTING                                                                        215
RELATED TO BUS STOP                                                            210
BICYCLE ADVANCING LEGALLY ON RED LIGHT                                         145
PASSING STOPPED SCHOOL BUS                                                      68
OBSTRUCTED CROSSWALKS                                                           40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                                       36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Creating Reckless Driving Behavior Dictionary

We will create a dictionary that combines categories that can be considered to be reckless driving behaviors for simplicity.

In [74]:
```
reckless_dict = {'TEXTING': 'RECKLESS DRIVING BEHAVIOR',
                 'DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)':
                 'RECKLESS DRIVING BEHAVIOR',
                  'HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)':'RECKLESS DRIVING BEHAVIOR',
                 'CELL PHONE USE OTHER THAN TEXTING': 'RECKLESS DRIVING BEHAVIOR',
                'UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)': 'RECKLESS DRIVING BEHAVIOR',
                'OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER' :
                 'RECKLESS DRIVING BEHAVIOR',
                'FOLLOWING TOO CLOSELY': 'RECKLESS DRIVING BEHAVIOR',
                'PASSING STOPPED SCHOOL BUS' : 'RECKLESS DRIVING BEHAVIOR',
                'PHYSICAL CONDITION OF DRIVER': 'RECKLESS DRIVING BEHAVIOR',
                'DRIVING ON WRONG SIDE/WRONG WAY':'RECKLESS DRIVING BEHAVIOR',
                'EXCEEDING AUTHORIZED SPEED LIMIT':'RECKLESS DRIVING BEHAVIOR',
                'EXCEEDING SAFE SPEED FOR CONDITIONS': 'RECKLESS DRIVING BEHAVIOR'}

prime_class['PRIM_CONTRIBUTORY_CAUSE'] =prime_class['PRIM_CONTRIBUTORY_CAUSE'].map(reckless_dict).fillna(prime_class
prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()

```

Out[74]:
```
IMPROPER                                                77881
RECKLESS DRIVING BEHAVIOR                               69962
FAILING TO YIELD RIGHT-OF-WAY                           52127
FAILING TO REDUCE SPEED TO AVOID CRASH                  20823
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE                     15769
DISREGARDING SIGNS/SIGNALS                              15674
WEATHER                                                  8722
EXTERNAL FACTORS                                         3954
DISTRACTION - FROM INSIDE VEHICLE                        3530
EQUIPMENT - VEHICLE CONDITION                            2990
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)     2863
DISTRACTION - FROM OUTSIDE VEHICLE                       2130
TURNING RIGHT ON RED                                      339
RELATED TO BUS STOP                                       210
BICYCLE ADVANCING LEGALLY ON RED LIGHT                    145
OBSTRUCTED CROSSWALKS                                      40
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT                  36
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Creating External Factors Dictionary

We will create a dictionary that combines categories that can be considered to be external factors for simplicity.

In [75]:
```
external_dict = {'OBSTRUCTED CROSSWALKS': 'EXTERNAL FACTORS',
                 'DISTRACTION - FROM OUTSIDE VEHICLE': 'EXTERNAL FACTORS',
                 'EQUIPMENT - VEHICLE CONDITION': 'EXTERNAL FACTORS',
                 'DISTRACTION - FROM INSIDE VEHICLE': 'EXTERNAL FACTORS',
                 'WEATHER': 'EXTERNAL FACTORS',
                'VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)': 'EXTERNAL FACTORS',
                'DRIVING SKILLS/KNOWLEDGE/EXPERIENCE':'EXTERNAL FACTORS',
                'BICYCLE ADVANCING LEGALLY ON RED LIGHT' :'EXTERNAL FACTORS',
                'MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT' :'EXTERNAL FACTORS'}


prime_class['PRIM_CONTRIBUTORY_CAUSE'] = prime_class['PRIM_CONTRIBUTORY_CAUSE'].map(external_dict).fillna(prime_class[
prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[75]:
```
IMPROPER                                77881
RECKLESS DRIVING BEHAVIOR               69962
FAILING TO YIELD RIGHT-OF-WAY           52127
EXTERNAL FACTORS                        40179
FAILING TO REDUCE SPEED TO AVOID CRASH  20823
DISREGARDING SIGNS/SIGNALS              15674
TURNING RIGHT ON RED                      339
RELATED TO BUS STOP                       210
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Creating Improper/Agressive Dictionary¶

We will create a dictionary that combines categories that can be considered to be improper and aggressive driving for simplicity.

In [76]:
```
improper_aggressive_dict = {'IMPROPER' : 'IMPROPER/AGGRESSIVE DRIVING',
            'FAILING TO YIELD RIGHT-OF-WAY' : 'IMPROPER/AGGRESSIVE DRIVING',
            'EXTERNAL FACTORS': 'EXTERNAL FACTORS/ OTHER',
            'FAILING TO REDUCE SPEED TO AVOID CRASH': 'IMPROPER/AGGRESSIVE DRIVING',
            'DISREGARDING SIGNS/SIGNALS': 'RECKLESS DRIVING BEHAVIOR',
            'RELATED TO BUS STOP': 'IMPROPER/AGGRESSIVE DRIVING',
            'TURNING RIGHT ON RED': 'RECKLESS DRIVING BEHAVIOR'}

prime_class['PRIM_CONTRIBUTORY_CAUSE'] = prime_class['PRIM_CONTRIBUTORY_CAUSE'].map(improper_aggressive_dict).fillna(pr
prime_class['PRIM_CONTRIBUTORY_CAUSE'].value_counts()
```

Out[76]:
```
IMPROPER/AGGRESSIVE DRIVING    151041
RECKLESS DRIVING BEHAVIOR       85975
EXTERNAL FACTORS/ OTHER         40179
Name: PRIM_CONTRIBUTORY_CAUSE, dtype: int64
```

Exploration of Data

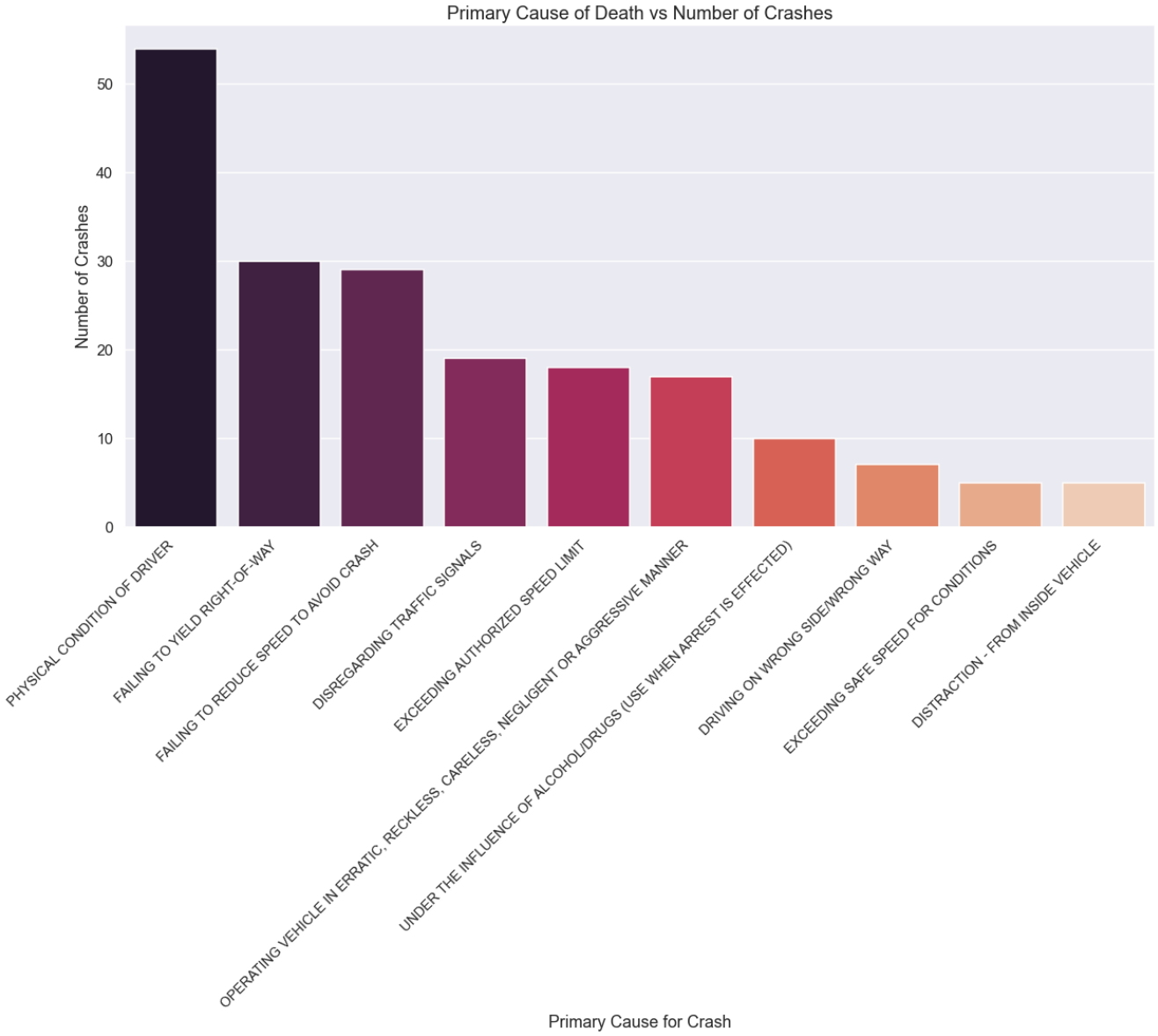
We will look at trends that we will later expand upon, most importantly, looking at the factors that contribute to fatal car accidents. While other types of accidents are important to explore, it is believed that saving lifes should be most important as improving these factors could have a trickle down effect.

We will accomplish this by creating and evaluating our dataset 'fatality'.

In [77]:
```
fatality = cat_data[cat_data['MOST_SEVERE_INJURY'] == 'FATAL']
```

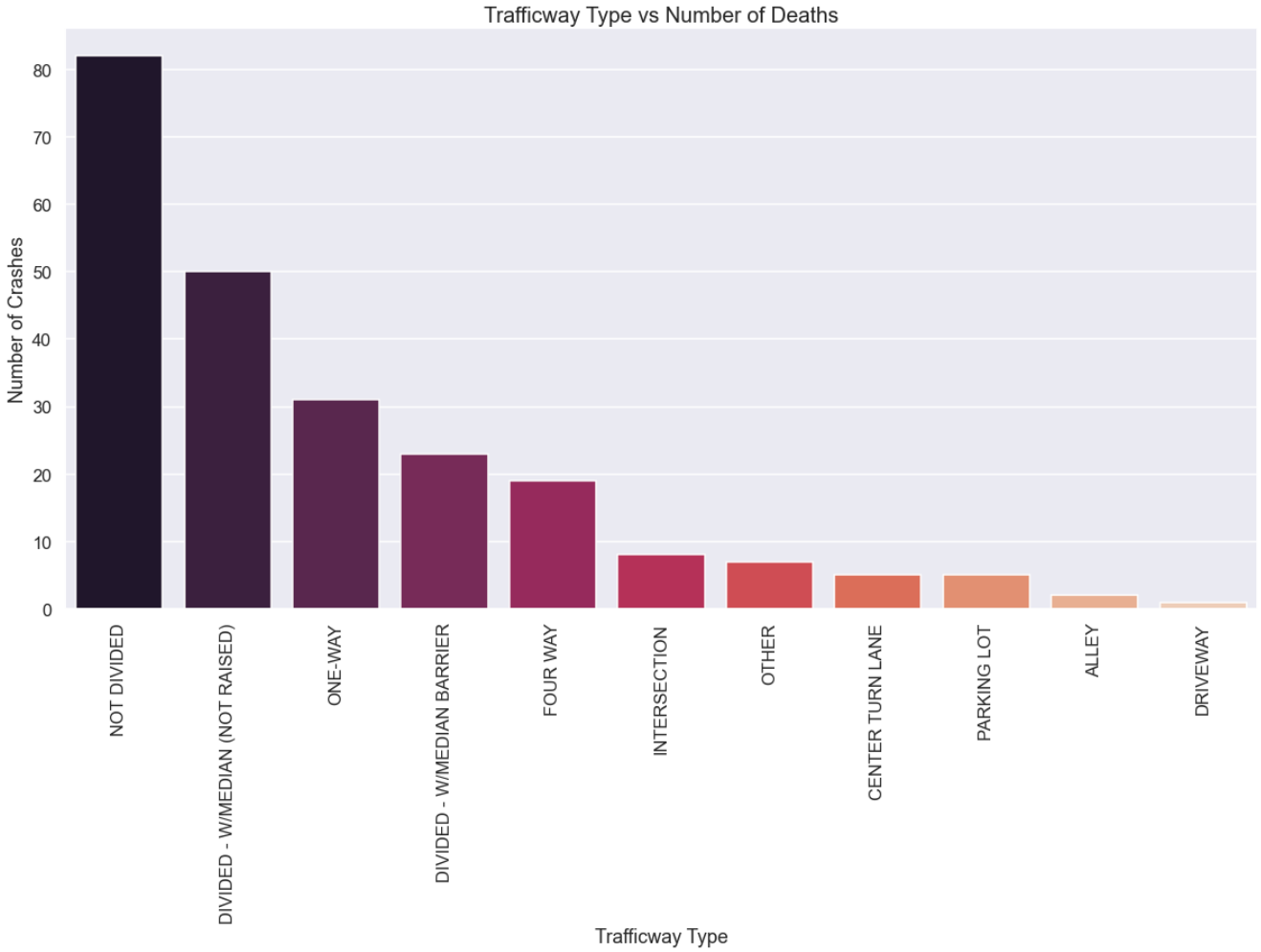
In [78]:

```
sns.set_context("talk")
sns.set_style("darkgrid")
plt.figure(figsize =(20,10))
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small'
)

ax = sns.countplot(x="PRIM_CONTRIBUTORY_CAUSE", data=fatality,
                   order = fatality['PRIM_CONTRIBUTORY_CAUSE'].value_counts().head(10).index, palette = 'rocket')
plt.xlabel('Primary Cause for Crash')
plt.ylabel('Number of Crashes')
plt.title('Primary Cause of Death vs Number of Crashes', fontsize=20)
plt.show()
```



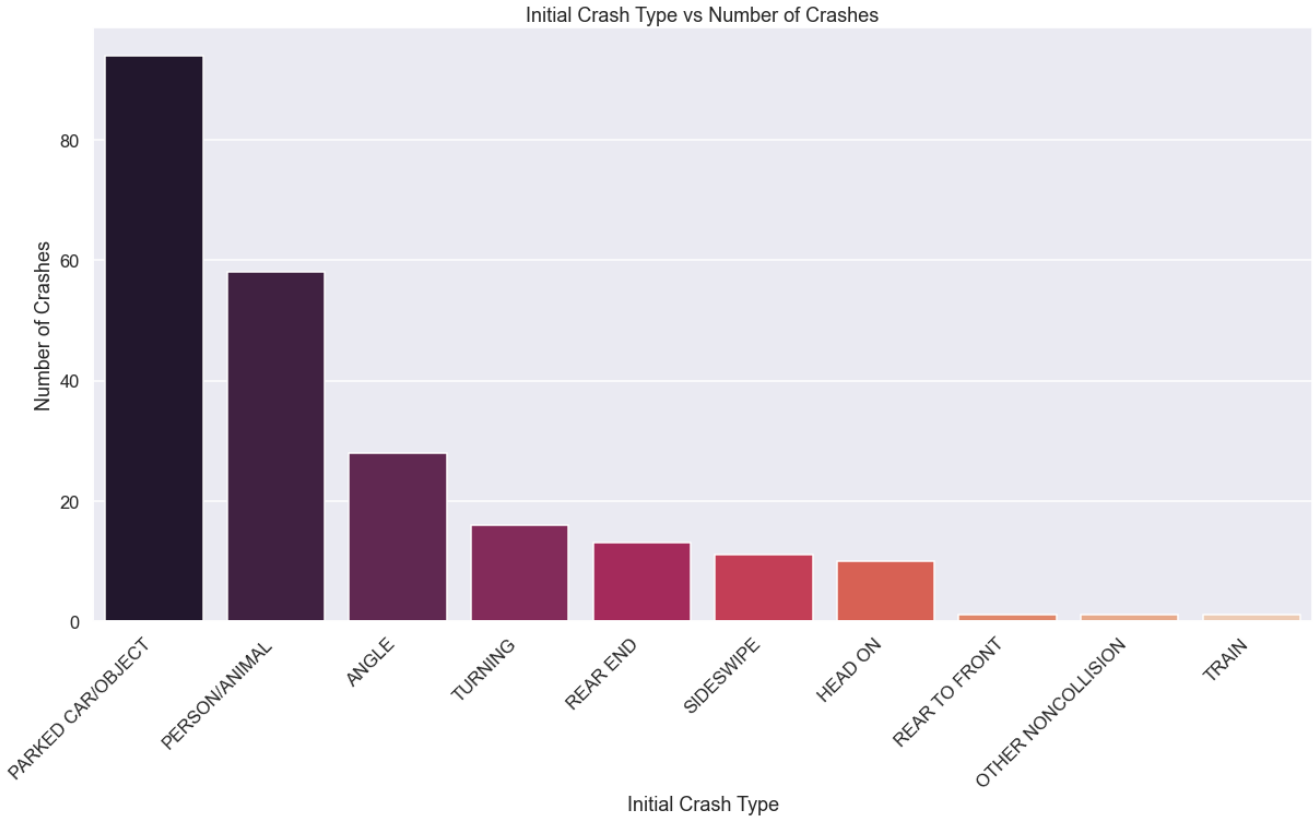
In [79]:

```
plt.figure(figsize =(20,10))
plt.xticks(rotation=90)

ax = sns.countplot(x="TRAFFICWAY_TYPE", data=fatality,
                   order = fatality['TRAFFICWAY_TYPE'].value_counts().index, palette = 'rocket')
plt.xlabel('Trafficway Type')
plt.ylabel('Number of Crashes')
plt.title('Trafficway Type vs Number of Deaths', fontsize=20)
plt.show()
```



In [80]:

```
plt.figure(figsize =(20,10))
plt.xticks(rotation=45, horizontalalignment='right')

ax = sns.countplot(x="FIRST_CRASH_TYPE", data=fatality,
                   order = fatality['FIRST_CRASH_TYPE'].value_counts().index, palette = 'rocket')
plt.xlabel('Initial Crash Type')
plt.ylabel('Number of Crashes')
plt.title('Initial Crash Type vs Number of Crashes')
plt.show()
```

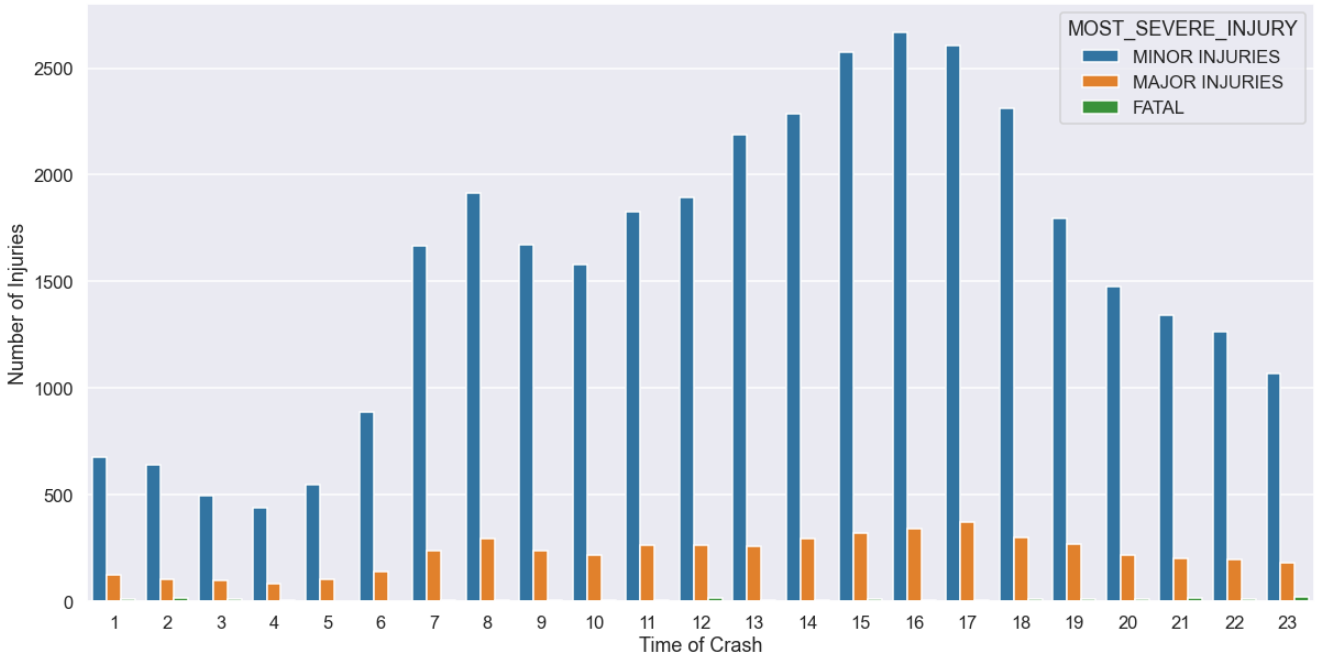


Injuries and Time of Accidents as a Factor

In [81]:
```
inj_crashes = cat_data[cat_data['MOST_SEVERE_INJURY'] != 'NO INJURY']
inj_crashes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 42297 entries, 7 to 490927
Data columns (total 28 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   CRASH_RECORD_ID           42297 non-null  object
 1   POSTED_SPEED_LIMIT        42297 non-null  int64
 2   TRAFFIC_CONTROL_DEVICE    42138 non-null  object
 3   DEVICE_CONDITION          42297 non-null  object
 4   WEATHER_CONDITION         42297 non-null  object
 5   LIGHTING_CONDITION        42297 non-null  object
 6   FIRST_CRASH_TYPE          42297 non-null  object
 7   TRAFFICWAY_TYPE           42297 non-null  object
 8   ROAD_DEFECT               42297 non-null  int64
 9   REPORT_TYPE               42297 non-null  object
 10  INTERSECTION_RELATED_I    42297 non-null  int64
 11  NOT_RIGHT_OF_WAY_I        42297 non-null  int64
 12  HIT_AND_RUN_I             42297 non-null  int64
 13  PRIM_CONTRIBUTORY_CAUSE   42297 non-null  object
 14  STREET_NAME               42297 non-null  object
 15  DOORING_I                 42297 non-null  int64
 16  WORK_ZONE_I               42297 non-null  int64
 17  WORKERS_PRESENT_I         42297 non-null  int64
 18  NUM_UNITS                 42297 non-null  int64
 19  MOST_SEVERE_INJURY        42297 non-null  object
 20  INJURIES_TOTAL            42297 non-null  float64
 21  CRASH_HOUR                42297 non-null  int64
 22  CRASH_DAY_OF_WEEK         42297 non-null  int64
 23  CRASH_MONTH               42297 non-null  int64
 24  STRAIGHT_ALIGNMENT        42297 non-null  int64
 25  DRY_ROADWAY_SURFACE_COND  42297 non-null  int64
 26  NO_INJ_CRASH_TYPE         42297 non-null  int64
 27  DAMAGE_DOLLARS            42297 non-null  object
dtypes: float64(1), int64(15), object(12)
memory usage: 9.4+ MB
```

In [82]:
```
plt.figure(figsize =(20,10))
ax = sns.countplot(x="CRASH_HOUR", hue = 'MOST_SEVERE_INJURY',data=inj_crashes,order = range(1,24))
plt.xlabel('Time of Crash')
plt.ylabel('Number of Injuries')
plt.show()
```

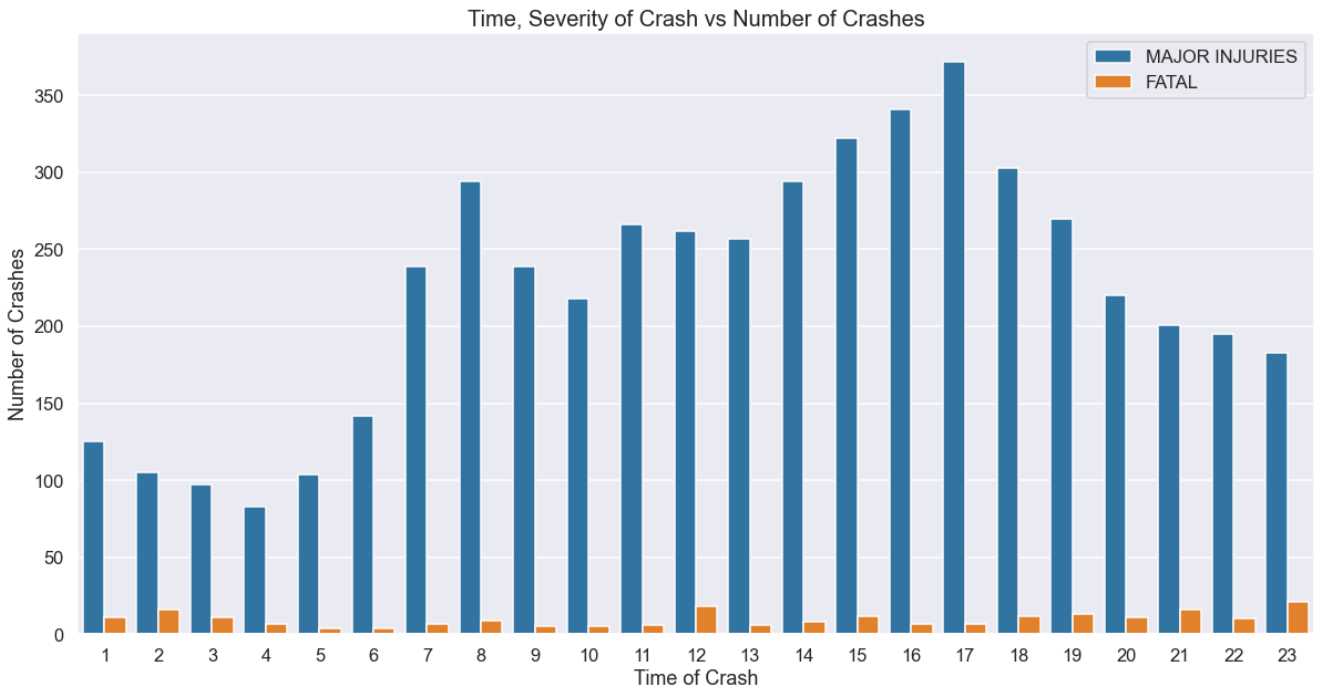


Focusing on Major Injury and Fatality

In [83]:
```
major_injury_crashes = cat_data[(cat_data['MOST_SEVERE_INJURY'] != 'NO INJURY')
                                & (cat_data['MOST_SEVERE_INJURY'] != 'MINOR INJURIES')]
major_injury_crashes.info()
```

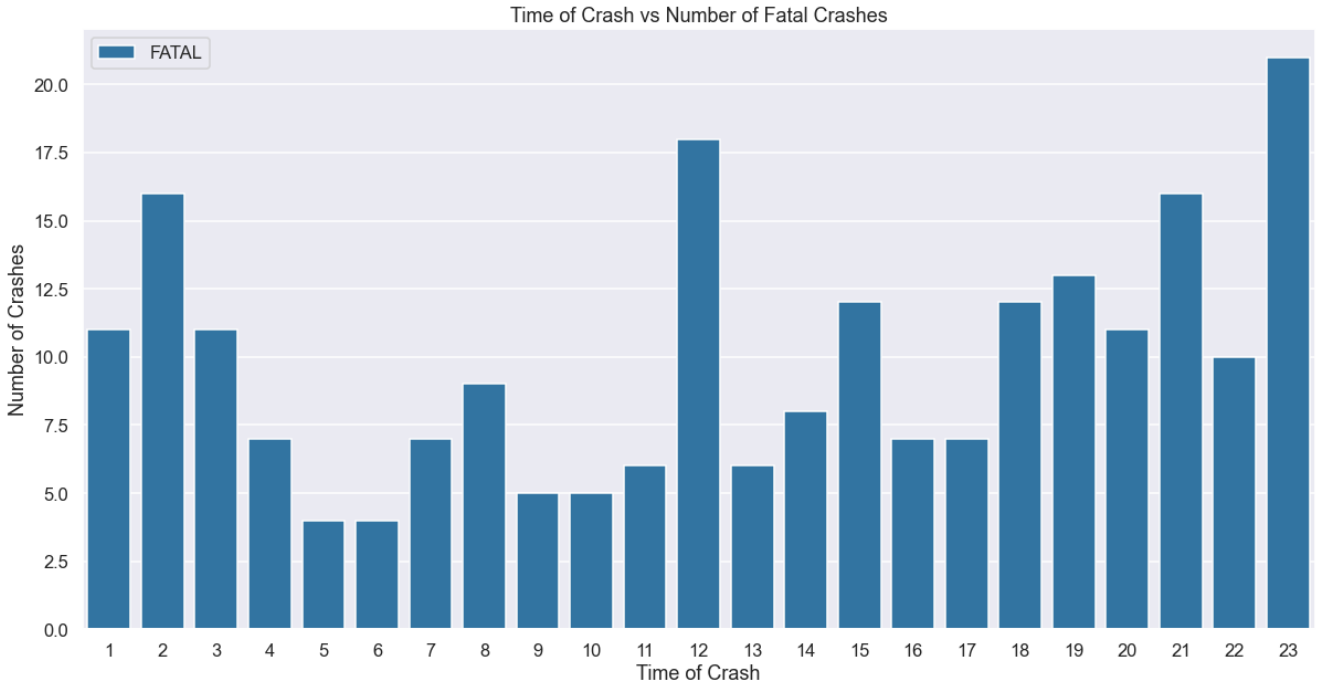
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5525 entries, 101 to 490537
Data columns (total 28 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   CRASH_RECORD_ID           5525 non-null   object
 1   POSTED_SPEED_LIMIT        5525 non-null   int64
 2   TRAFFIC_CONTROL_DEVICE    5498 non-null   object
 3   DEVICE_CONDITION          5525 non-null   object
 4   WEATHER_CONDITION         5525 non-null   object
 5   LIGHTING_CONDITION        5525 non-null   object
 6   FIRST_CRASH_TYPE          5525 non-null   object
 7   TRAFFICWAY_TYPE           5525 non-null   object
 8   ROAD_DEFECT               5525 non-null   int64
 9   REPORT_TYPE               5525 non-null   object
 10  INTERSECTION_RELATED_I    5525 non-null   int64
 11  NOT_RIGHT_OF_WAY_I        5525 non-null   int64
 12  HIT_AND_RUN_I             5525 non-null   int64
 13  PRIM_CONTRIBUTORY_CAUSE   5525 non-null   object
 14  STREET_NAME               5525 non-null   object
 15  DOORING_I                 5525 non-null   int64
 16  WORK_ZONE_I               5525 non-null   int64
 17  WORKERS_PRESENT_I         5525 non-null   int64
 18  NUM_UNITS                 5525 non-null   int64
 19  MOST_SEVERE_INJURY        5525 non-null   object
 20  INJURIES_TOTAL            5525 non-null   float64
 21  CRASH_HOUR                5525 non-null   int64
 22  CRASH_DAY_OF_WEEK         5525 non-null   int64
 23  CRASH_MONTH               5525 non-null   int64
 24  STRAIGHT_ALIGNMENT        5525 non-null   int64
 25  DRY_ROADWAY_SURFACE_COND  5525 non-null   int64
 26  NO_INJ_CRASH_TYPE         5525 non-null   int64
 27  DAMAGE_DOLLARS            5525 non-null   object
dtypes: float64(1), int64(15), object(12)
memory usage: 1.2+ MB
```

In [84]:
```
plt.figure(figsize =(20,10))
ax = sns.countplot(x="CRASH_HOUR", hue = 'MOST_SEVERE_INJURY',data=major_injury_crashes,order = range(1,24))
plt.xlabel('Time of Crash')
plt.ylabel('Number of Crashes')
plt.title('Time, Severity of Crash vs Number of Crashes', fontsize=20)
plt.legend(loc = 'upper right')
plt.show()
```



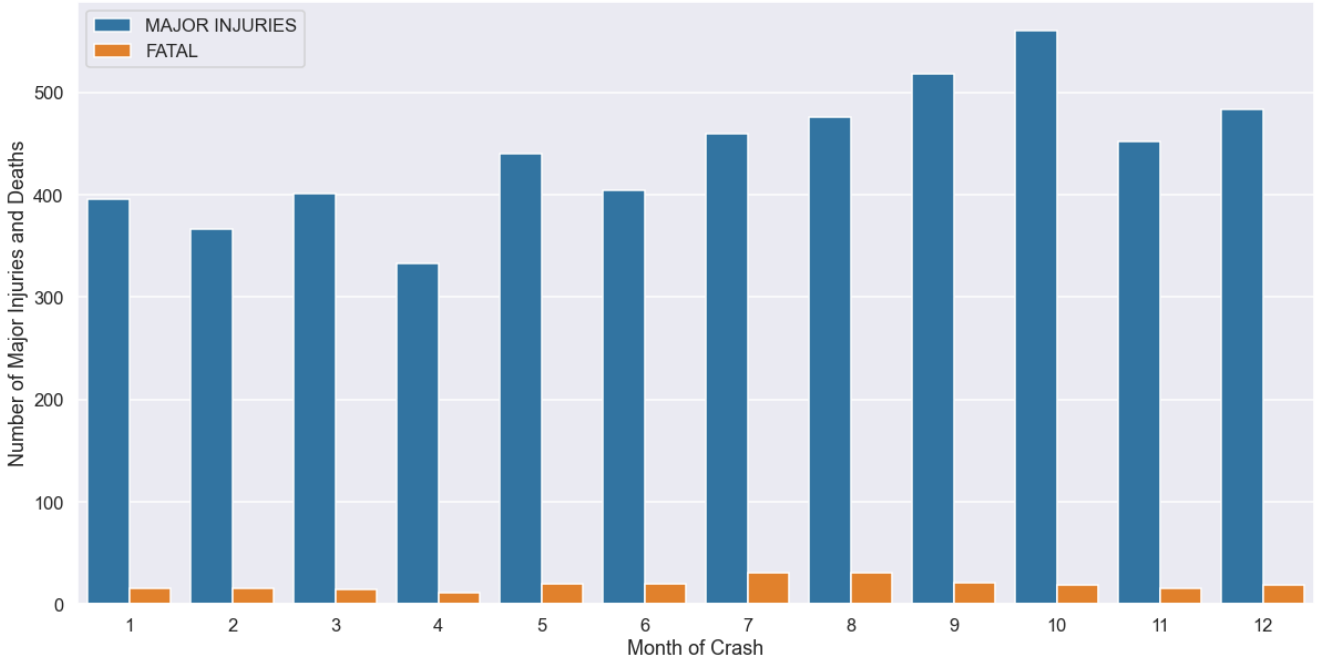
Focuing on Fatal Accidents and Time of Crash

In [85]:
```
plt.figure(figsize =(20,10))
ax = sns.countplot(x="CRASH_HOUR",hue = 'MOST_SEVERE_INJURY',data=fatality,
                   order = range(1,24))

plt.xlabel('Time of Crash')
plt.ylabel('Number of Crashes')
plt.title('Time of Crash vs Number of Fatal Crashes')
plt.legend(loc = 'upper left')
plt.show()
```



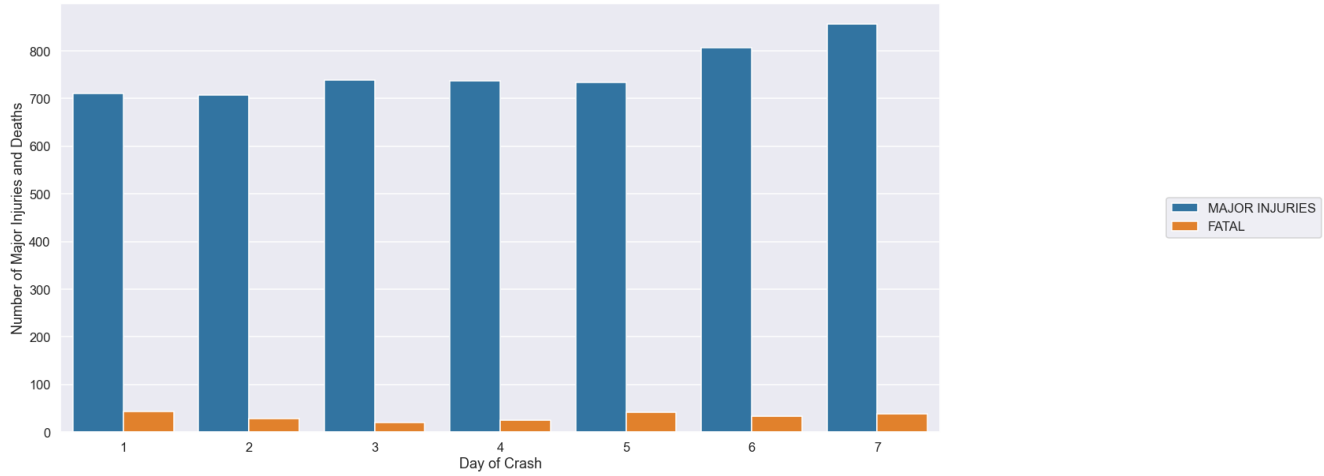
Major Injuries and Fatal Crashes by Month

In [86]:
```
plt.figure(figsize =(20,10))
ax = sns.countplot(x="CRASH_MONTH", hue = 'MOST_SEVERE_INJURY',data=major_injury_crashes,order = range(1,13))
plt.xlabel('Month of Crash')
plt.ylabel('Number of Major Injuries and Deaths')
plt.legend(loc = 'upper left')

plt.show()
```



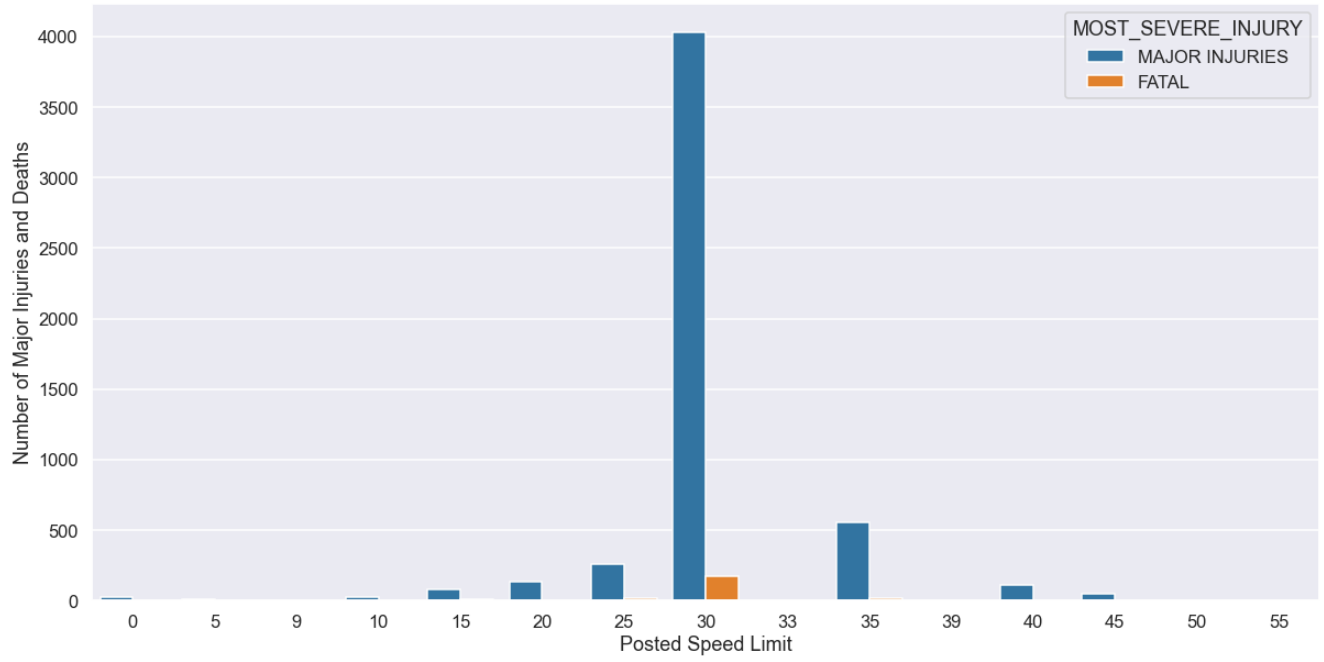
Major Injuries and Fatal Crashes by Week

In [87]:
```
plt.figure(figsize =(20,10))
ax = sns.countplot(x="CRASH_DAY_OF_WEEK", hue = 'MOST_SEVERE_INJURY',data=major_injury_crashes,order = range(1,8))
plt.xlabel('Day of Crash')
plt.ylabel('Number of Major Injuries and Deaths')
plt.legend(fancybox=True,loc='center left', bbox_to_anchor=(1.25, 0.5), ncol=1)
plt.show()
```



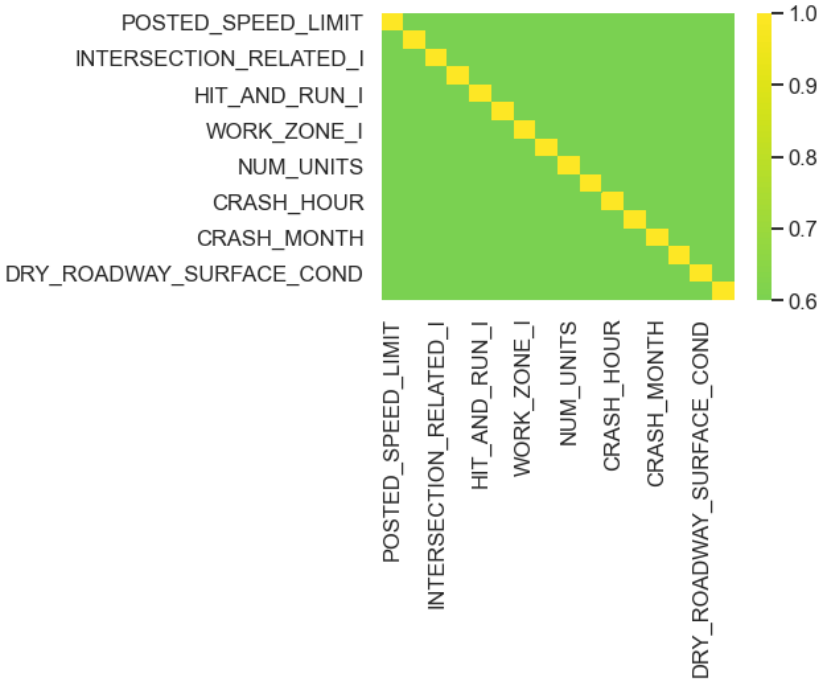
Major Injuries and Fatal Crashes by Posted Speed Limit

In [88]:
```
plt.figure(figsize =(20,10))
ax = sns.countplot(x="POSTED_SPEED_LIMIT", hue = 'MOST_SEVERE_INJURY',data=major_injury_crashes)
plt.xlabel('Posted Speed Limit')
plt.ylabel('Number of Major Injuries and Deaths')
plt.show()
```



In [89]:
```
1 sns.heatmap(prime_class.corr(), center=0, cmap='viridis',vmin = 0.60)
```

Out[89]: <AxesSubplot:>



Modeling

In [90]:
```
# importing our necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.neighbors import KNeighborsClassifier # K-nearest Neighbors
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import precision_score, recall_score, accuracy_score, f1_score, roc_curve, auc, r2_score, mean_sq
from sklearn.metrics import plot_confusion_matrix,confusion_matrix, classification_report
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn import preprocessing
from sklearn.utils import class_weight
from sklearn.ensemble import BaggingClassifier, RandomForestClassifier
import xgboost as xgb
%matplotlib inline
```

Next we will drop some columns and create our predictor and result variables. It will be important at this point to create our dummies for our data.

In [91]:
```
model_data = prime_class.drop(columns = ['CRASH_RECORD_ID', 'STREET_NAME'])
model_data.head()
model_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 277195 entries, 0 to 490937
Data columns (total 26 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   POSTED_SPEED_LIMIT        277195 non-null  int64  
 1   TRAFFIC_CONTROL_DEVICE    276287 non-null  object 
 2   DEVICE_CONDITION          277195 non-null  object 
 3   WEATHER_CONDITION         277194 non-null  object 
 4   LIGHTING_CONDITION        277195 non-null  object 
 5   FIRST_CRASH_TYPE          277195 non-null  object 
 6   TRAFFICWAY_TYPE           277195 non-null  object 
 7   ROAD_DEFECT               277195 non-null  int64  
 8   REPORT_TYPE               277195 non-null  object 
 9   INTERSECTION_RELATED_I    277195 non-null  int64  
 10  NOT_RIGHT_OF_WAY_I        277195 non-null  int64  
 11  HIT_AND_RUN_I             277195 non-null  int64  
 12  PRIM_CONTRIBUTORY_CAUSE   277195 non-null  object 
 13  DOORING_I                 277195 non-null  int64  
 14  WORK_ZONE_I               277195 non-null  int64  
 15  WORKERS_PRESENT_I         277195 non-null  int64  
 16  NUM_UNITS                 277195 non-null  int64  
 17  MOST_SEVERE_INJURY        277195 non-null  object 
 18  INJURIES_TOTAL            277195 non-null  float64
 19  CRASH_HOUR                277195 non-null  int64  
 20  CRASH_DAY_OF_WEEK         277195 non-null  int64  
 21  CRASH_MONTH               277195 non-null  int64  
 22  STRAIGHT_ALIGNMENT        277195 non-null  int64  
 23  DRY_ROADWAY_SURFACE_COND  277195 non-null  int64  
 24  NO_INJ_CRASH_TYPE         277195 non-null  int64  
 25  DAMAGE_DOLLARS            277195 non-null  object 
dtypes: float64(1), int64(15), object(10)
memory usage: 57.1+ MB
```

In [92]:
```
X = model_data.drop('PRIM_CONTRIBUTORY_CAUSE', axis=1)
y = model_data['PRIM_CONTRIBUTORY_CAUSE']
```

In [93]:
```
model_d = pd.get_dummies(X)
```

In [94]:
```
1 model_d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 277195 entries, 0 to 490937
Data columns (total 65 columns):
 #   Column                                          Non-Null Count   Dtype
---  ------                                          --------------   -----
 0   POSTED_SPEED_LIMIT                              277195 non-null  int64
 1   ROAD_DEFECT                                     277195 non-null  int64
 2   INTERSECTION_RELATED_I                          277195 non-null  int64
 3   NOT_RIGHT_OF_WAY_I                              277195 non-null  int64
 4   HIT_AND_RUN_I                                   277195 non-null  int64
 5   DOORING_I                                       277195 non-null  int64
 6   WORK_ZONE_I                                     277195 non-null  int64
 7   WORKERS_PRESENT_I                               277195 non-null  int64
 8   NUM_UNITS                                       277195 non-null  int64
 9   INJURIES_TOTAL                                  277195 non-null  float64
 10  CRASH_HOUR                                      277195 non-null  int64
 11  CRASH_DAY_OF_WEEK                               277195 non-null  int64
 12  CRASH_MONTH                                     277195 non-null  int64
 13  STRAIGHT_ALIGNMENT                              277195 non-null  int64
 14  DRY_ROADWAY_SURFACE_COND                        277195 non-null  int64
 15  NO_INJ_CRASH_TYPE                               277195 non-null  int64
 16  TRAFFIC_CONTROL_DEVICE_NO CONTROLS              277195 non-null  uint8
 17  TRAFFIC_CONTROL_DEVICE_OTHER                    277195 non-null  uint8
 18  TRAFFIC_CONTROL_DEVICE_SIGN                     277195 non-null  uint8
 19  TRAFFIC_CONTROL_DEVICE_SIGNAL                   277195 non-null  uint8
 20  DEVICE_CONDITION_FUNCTIONING IMPROPERLY/ MISSING  277195 non-null  uint8
 21  DEVICE_CONDITION_FUNCTIONING PROPERLY           277195 non-null  uint8
 22  DEVICE_CONDITION_NO CONTROLS                    277195 non-null  uint8
 23  WEATHER_CONDITION_CLEAR                         277195 non-null  uint8
 24  WEATHER_CONDITION_OTHER                         277195 non-null  uint8
 25  WEATHER_CONDITION_PRECIPITATION                 277195 non-null  uint8
 26  LIGHTING_CONDITION_DARKNESS                     277195 non-null  uint8
 27  LIGHTING_CONDITION_LIGHT                        277195 non-null  uint8
 28  LIGHTING_CONDITION_VARIABLE LIGHT               277195 non-null  uint8
 29  FIRST_CRASH_TYPE_ANGLE                          277195 non-null  uint8
 30  FIRST_CRASH_TYPE_HEAD ON                        277195 non-null  uint8
 31  FIRST_CRASH_TYPE_OTHER NONCOLLISION             277195 non-null  uint8
 32  FIRST_CRASH_TYPE_OVERTURNED                     277195 non-null  uint8
 33  FIRST_CRASH_TYPE_PARKED CAR/OBJECT              277195 non-null  uint8
 34  FIRST_CRASH_TYPE_PERSON/ANIMAL                  277195 non-null  uint8
 35  FIRST_CRASH_TYPE_REAR END                       277195 non-null  uint8
 36  FIRST_CRASH_TYPE_REAR TO FRONT                  277195 non-null  uint8
 37  FIRST_CRASH_TYPE_REAR TO REAR                   277195 non-null  uint8
 38  FIRST_CRASH_TYPE_REAR TO SIDE                   277195 non-null  uint8
 39  FIRST_CRASH_TYPE_SIDESWIPE                      277195 non-null  uint8
 40  FIRST_CRASH_TYPE_TRAIN                          277195 non-null  uint8
 41  FIRST_CRASH_TYPE_TURNING                        277195 non-null  uint8
 42  TRAFFICWAY_TYPE_ALLEY                           277195 non-null  uint8
 43  TRAFFICWAY_TYPE_CENTER TURN LANE                277195 non-null  uint8
 44  TRAFFICWAY_TYPE_DIVIDED - W/MEDIAN (NOT RAISED)  277195 non-null  uint8
 45  TRAFFICWAY_TYPE_DIVIDED - W/MEDIAN BARRIER      277195 non-null  uint8
 46  TRAFFICWAY_TYPE_DRIVEWAY                        277195 non-null  uint8
 47  TRAFFICWAY_TYPE_FOUR WAY                        277195 non-null  uint8
 48  TRAFFICWAY_TYPE_INTERSECTION                    277195 non-null  uint8
 49  TRAFFICWAY_TYPE_NOT DIVIDED                     277195 non-null  uint8
 50  TRAFFICWAY_TYPE_ONE-WAY                         277195 non-null  uint8
 51  TRAFFICWAY_TYPE_OTHER                           277195 non-null  uint8
 52  TRAFFICWAY_TYPE_PARKING LOT                     277195 non-null  uint8
 53  TRAFFICWAY_TYPE_RAMP                            277195 non-null  uint8
 54  TRAFFICWAY_TYPE_TRAFFIC ROUTE                   277195 non-null  uint8
 55  REPORT_TYPE_AMENDED                             277195 non-null  uint8
 56  REPORT_TYPE_NOT ON SCENE (DESK REPORT)          277195 non-null  uint8
 57  REPORT_TYPE_ON SCENE                            277195 non-null  uint8
 58  MOST_SEVERE_INJURY_FATAL                        277195 non-null  uint8
 59  MOST_SEVERE_INJURY_MAJOR INJURIES               277195 non-null  uint8
 60  MOST_SEVERE_INJURY_MINOR INJURIES               277195 non-null  uint8
 61  MOST_SEVERE_INJURY_NO INJURY                    277195 non-null  uint8
 62  DAMAGE_DOLLARS_500 OR LESS                      277195 non-null  uint8
 63  DAMAGE_DOLLARS_501 TO 1500                      277195 non-null  uint8
 64  DAMAGE_DOLLARS_OVER 1500                        277195 non-null  uint8
dtypes: float64(1), int64(15), uint8(49)
memory usage: 48.9 MB
```

In [95]:
```
model_d.head()
```

Out[95]:

	POSTED_SPEED_LIMIT	ROAD_DEFECT	INTERSECTION_RELATED_I	NOT_RIGHT_OF_WAY_I	HIT_AND_RUN_I	DOORING_I	WORK_ZONE_I	WORKERS_PRESENT
0	35	0	0	0	0	0	0	
1	35	0	1	0	0	0	0	
2	30	0	0	0	0	0	0	
7	30	0	0	0	0	0	0	
10	30	0	1	0	0	0	0	

In [96]:
```
model_d.columns
```

Out[96]:
```
Index(['POSTED_SPEED_LIMIT', 'ROAD_DEFECT', 'INTERSECTION_RELATED_I',
       'NOT_RIGHT_OF_WAY_I', 'HIT_AND_RUN_I', 'DOORING_I', 'WORK_ZONE_I',
       'WORKERS_PRESENT_I', 'NUM_UNITS', 'INJURIES_TOTAL', 'CRASH_HOUR',
       'CRASH_DAY_OF_WEEK', 'CRASH_MONTH', 'STRAIGHT_ALIGNMENT',
       'DRY_ROADWAY_SURFACE_COND', 'NO_INJ_CRASH_TYPE',
       'TRAFFIC_CONTROL_DEVICE_NO CONTROLS', 'TRAFFIC_CONTROL_DEVICE_OTHER',
       'TRAFFIC_CONTROL_DEVICE_SIGN', 'TRAFFIC_CONTROL_DEVICE_SIGNAL',
       'DEVICE_CONDITION_FUNCTIONING IMPROPERLY/ MISSING',
       'DEVICE_CONDITION_FUNCTIONING PROPERLY', 'DEVICE_CONDITION_NO CONTROLS',
       'WEATHER_CONDITION_CLEAR', 'WEATHER_CONDITION_OTHER',
       'WEATHER_CONDITION_PRECIPITATION', 'LIGHTING_CONDITION_DARKNESS',
       'LIGHTING_CONDITION_LIGHT', 'LIGHTING_CONDITION_VARIABLE LIGHT',
       'FIRST_CRASH_TYPE_ANGLE', 'FIRST_CRASH_TYPE_HEAD ON',
       'FIRST_CRASH_TYPE_OTHER NONCOLLISION', 'FIRST_CRASH_TYPE_OVERTURNED',
       'FIRST_CRASH_TYPE_PARKED CAR/OBJECT', 'FIRST_CRASH_TYPE_PERSON/ANIMAL',
       'FIRST_CRASH_TYPE_REAR END', 'FIRST_CRASH_TYPE_REAR TO FRONT',
       'FIRST_CRASH_TYPE_REAR TO REAR', 'FIRST_CRASH_TYPE_REAR TO SIDE',
       'FIRST_CRASH_TYPE_SIDESWIPE', 'FIRST_CRASH_TYPE_TRAIN',
       'FIRST_CRASH_TYPE_TURNING', 'TRAFFICWAY_TYPE_ALLEY',
       'TRAFFICWAY_TYPE_CENTER TURN LANE',
       'TRAFFICWAY_TYPE_DIVIDED - W/MEDIAN (NOT RAISED)',
       'TRAFFICWAY_TYPE_DIVIDED - W/MEDIAN BARRIER',
       'TRAFFICWAY_TYPE_DRIVEWAY', 'TRAFFICWAY_TYPE_FOUR WAY',
       'TRAFFICWAY_TYPE_INTERSECTION', 'TRAFFICWAY_TYPE_NOT DIVIDED',
       'TRAFFICWAY_TYPE_ONE-WAY', 'TRAFFICWAY_TYPE_OTHER',
       'TRAFFICWAY_TYPE_PARKING LOT', 'TRAFFICWAY_TYPE_RAMP',
       'TRAFFICWAY_TYPE_TRAFFIC ROUTE', 'REPORT_TYPE_AMENDED',
       'REPORT_TYPE_NOT ON SCENE (DESK REPORT)', 'REPORT_TYPE_ON SCENE',
       'MOST_SEVERE_INJURY_FATAL', 'MOST_SEVERE_INJURY_MAJOR INJURIES',
       'MOST_SEVERE_INJURY_MINOR INJURIES', 'MOST_SEVERE_INJURY_NO INJURY',
       'DAMAGE_DOLLARS_500 OR LESS', 'DAMAGE_DOLLARS_501 TO 1500',
       'DAMAGE_DOLLARS_OVER 1500'],
      dtype='object')
```

Here we will create the model for K-Nearest Neighbors by creating our train test split with 25% set for testing. Then we will follow up with instantiating the KNeighborsClassifer and predict on our test set.

In [97]:
```
X_train, X_test, y_train, y_test = train_test_split(model_d, y, test_size=0.25, random_state=123)
```

In [101]:
```
scaler = StandardScaler()
scaled_df = scaler.fit_transform(model_d)

scaled_data_train = scaler.fit_transform(X_train)
scaled_data_test = scaler.transform(X_test)

scaled_df_train = pd.DataFrame(scaled_data_train, columns=X_train.columns)
scaled_df_train.head()
```

Out[101]:

	POSTED_SPEED_LIMIT	ROAD_DEFECT	INTERSECTION_RELATED_I	NOT_RIGHT_OF_WAY_I	HIT_AND_RUN_I	DOORING_I	WORK_ZONE_I	WORKERS_PRESENT_
0	0.209674	-0.150974	-0.602378	-0.201103	-0.517656	-0.039933	-0.080059	-0.04228
1	0.209674	-0.150974	-0.602378	-0.201103	-0.517656	-0.039933	-0.080059	-0.04228
2	0.209674	-0.150974	-0.602378	-0.201103	-0.517656	-0.039933	-0.080059	-0.04228
3	1.871434	-0.150974	-0.602378	-0.201103	-0.517656	-0.039933	-0.080059	-0.04228
4	0.209674	-0.150974	-0.602378	-0.201103	-0.517656	-0.039933	-0.080059	-0.04228

In [128]:
```
clf = KNeighborsClassifier()

clf.fit(scaled_data_train, y_train)

test_preds = clf.predict(scaled_data_test)
```

In [129]:
```
def metrics(labels, preds):
    print("Precision Score: {}".format(precision_score(labels, preds, average= 'weighted')))
    print("Recall Score: {}".format(recall_score(labels, preds,average='weighted')))
    print("Accuracy Score: {}".format(accuracy_score(labels, preds)))
    print("F1 Score: {}".format(f1_score(labels, preds,average= 'weighted')))

metrics(y_test, test_preds)
```

```
Precision Score: 0.6308677704679002
Recall Score: 0.6431117332140435
Accuracy Score: 0.6431117332140435
F1 Score: 0.6339990945641581
```

Decision Trees

In [130]:
```
# Train a DT classifier
classifier = DecisionTreeClassifier(random_state=42)
classifier.fit(X_train, y_train)
```

Out[130]:
```
DecisionTreeClassifier(random_state=42)
```

In [131]:
```
y_pred = classifier.predict(X_test)
```

In [132]:
```
metrics(y_test, y_pred)
```

```
Precision Score: 0.5615663523375126
Recall Score: 0.5564726763733965
Accuracy Score: 0.5564726763733965
F1 Score: 0.5588829351463644
```

In [*]:
```
plt.figure(figsize=(10,10), dpi=500)
tree.plot_tree(classifier,
               feature_names=model_d.columns,
               class_names=np.unique(y).astype('str'),
               filled=True, rounded=True)
plt.show()
```

In [ ]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(classifier, X_test, y_test,
                      cmap=plt.cm.Blues,normalize='true')
plt.show()
```

In [ ]:
```

```

In [ ]:
```

```

Random Forest Model

In [102]:
```
tree_clf = DecisionTreeClassifier(criterion='gini', max_depth=5,class_weight = 'balanced', random_state = 42)
tree_clf.fit(X_train, y_train)
```

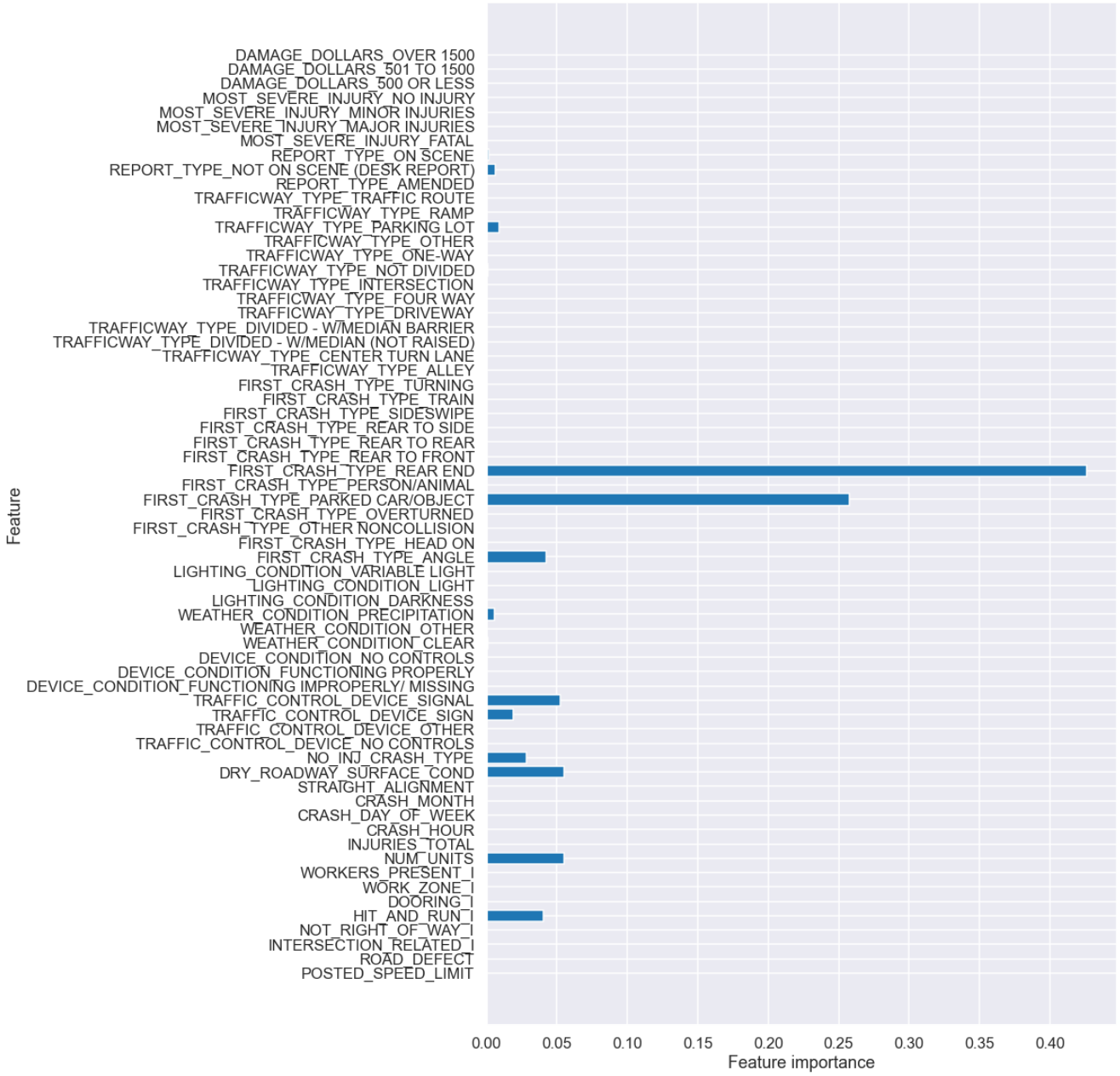
Out[102]:
```
DecisionTreeClassifier(class_weight='balanced', max_depth=5, random_state=42)
```

In [103]:
```
tree_clf.feature_importances_
```

Out[103]:
```
array([0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       4.02630015e-02, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       5.49112265e-02, 0.00000000e+00, 8.34381632e-05, 0.00000000e+00,
       0.00000000e+00, 0.00000000e+00, 5.52371201e-02, 2.85782082e-02,
       0.00000000e+00, 0.00000000e+00, 1.89849074e-02, 5.21104028e-02,
       0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 5.97473106e-04,
       0.00000000e+00, 5.60835534e-03, 0.00000000e+00, 0.00000000e+00,
       0.00000000e+00, 4.20487967e-02, 0.00000000e+00, 0.00000000e+00,
       4.02198671e-04, 2.57519418e-01, 0.00000000e+00, 4.26303402e-01,
       0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       0.00000000e+00, 1.06756815e-03, 0.00000000e+00, 1.51892821e-04,
       8.60252383e-03, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       6.14444166e-03, 1.38562479e-03, 0.00000000e+00, 0.00000000e+00,
       0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
       0.00000000e+00])
```

In [104]:

```
def plot_feature_importances(model):
    n_features = X_train.shape[1]
    plt.figure(figsize=(12,20))
    plt.barh(range(n_features), model.feature_importances_, align='center')
    plt.yticks(np.arange(n_features), scaled_df_train.columns.values)
    plt.xlabel('Feature importance')
    plt.ylabel('Feature')

plot_feature_importances(tree_clf)
```



In [105]:
```
pred = tree_clf.predict(X_test)

# Confusion matrix and classification report
print(confusion_matrix(y_test, pred))
print(classification_report(y_test, pred))
```

```
[[ 4926  3091  2013]
 [ 6055 26944  4894]
 [ 3018  5053 13305]]
                           precision    recall  f1-score   support

    EXTERNAL FACTORS/ OTHER       0.35      0.49      0.41     10030
IMPROPER/AGGRESSIVE DRIVING       0.77      0.71      0.74     37893
  RECKLESS DRIVING BEHAVIOR       0.66      0.62      0.64     21376

                   accuracy                           0.65     69299
                  macro avg       0.59      0.61      0.60     69299
               weighted avg       0.67      0.65      0.66     69299
```

In [106]:
```
# Instantiate a BaggingClassifier
bagged_tree =  BaggingClassifier(DecisionTreeClassifier(criterion='gini',
                                                         max_depth=5, class_weight = 'balanced',
                                                         random_state = 42), n_estimators=20)
```

In [107]:
```
bagged_tree.fit(X_train, y_train)
```

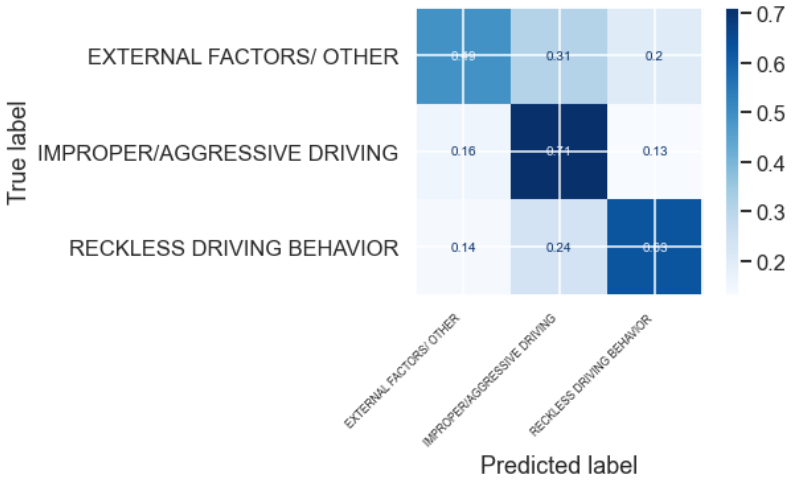
Out[107]:
```
BaggingClassifier(base_estimator=DecisionTreeClassifier(class_weight='balanced',
                                                        max_depth=5,
                                                        random_state=42),
                  n_estimators=20)
```

In [108]:
```
bagged_tree.score(X_train, y_train)

```

Out[108]:
```
0.6488244122061031
```

In [109]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(bagged_tree, X_test, y_test,
                      cmap=plt.cm.Blues,normalize='true')
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small'
)
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



In [110]:
```
# Instantiate and fit a RandomForestClassifier
forest = RandomForestClassifier(n_estimators=100, max_depth= 5, class_weight = 'balanced', random_state = 42)
forest.fit(X_train, y_train)
```

Out[110]:
```
RandomForestClassifier(class_weight='balanced', max_depth=5, random_state=42)
```

In [111]:
```
forest.score(X_train, y_train)
```

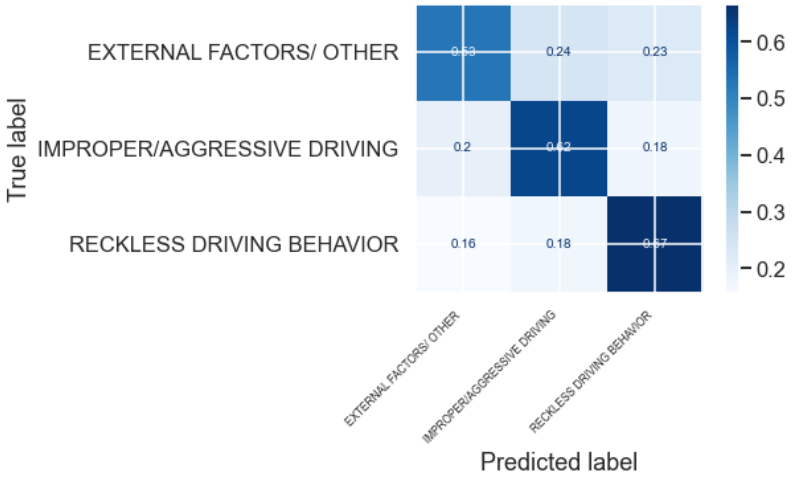
Out[111]:
```
0.6195934505714396
```

In [112]:
```
forest.score(X_test, y_test)
```

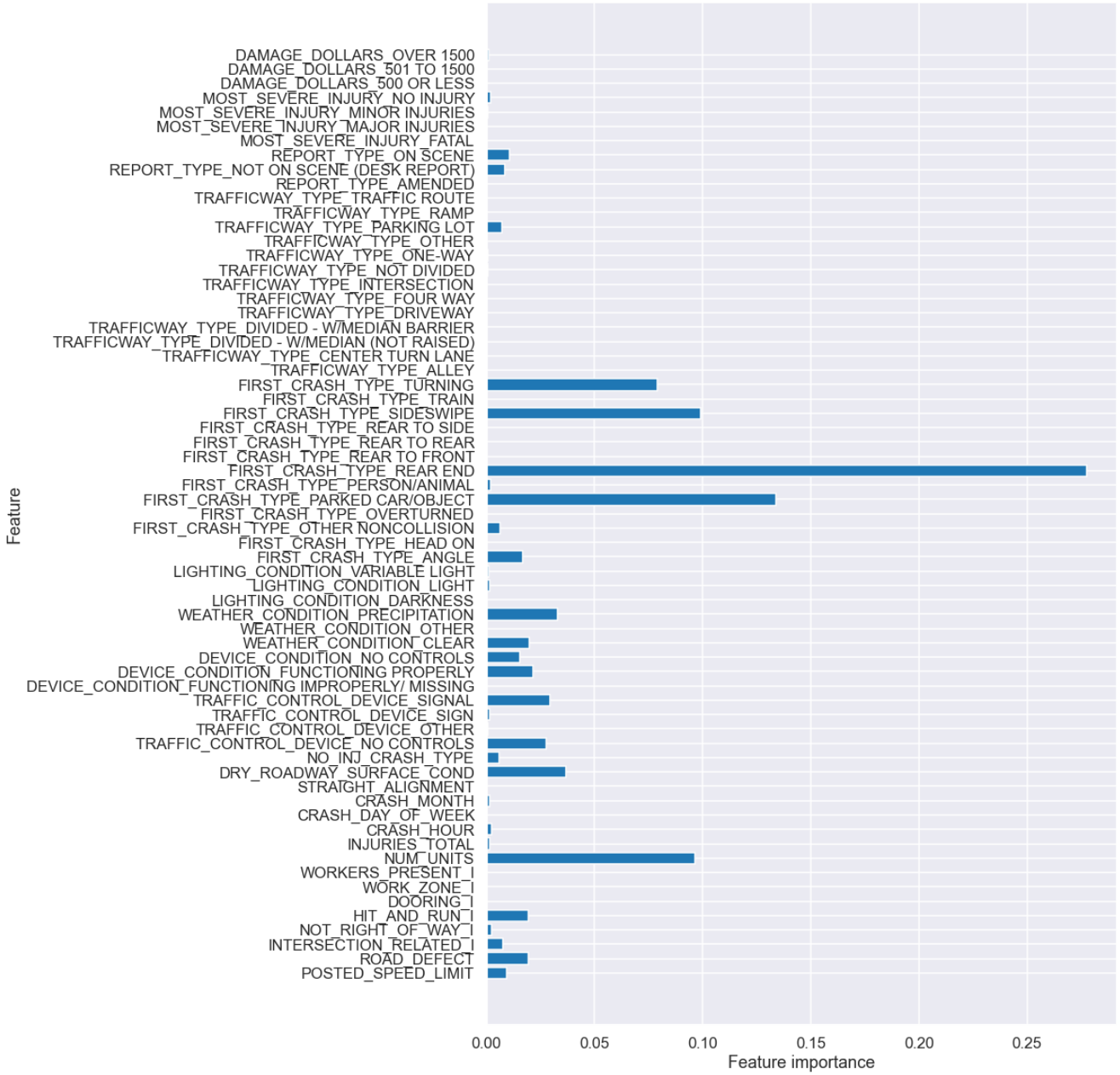
Out[112]:
```
0.6222889219180652
```

In [113]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(forest, X_test, y_test,
                      cmap=plt.cm.Blues,normalize='true')
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small')
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



In [114]:
```
1 plot_feature_importances(forest)
```



Gridsearch Model

In [115]:
```
dt_clf = DecisionTreeClassifier(random_state = 42, class_weight = 'balanced')

dt_cv_score = cross_val_score(dt_clf, X_train, y_train, cv=3)
mean_dt_cv_score = np.mean(dt_cv_score)

print(f"Mean Cross Validation Score: {mean_dt_cv_score :.2%}")
```

```
Mean Cross Validation Score: 55.68%
```

In [116]:
```
dt_param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 2, 3, 4, 5, 6],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 3, 4, 5, 6]}
```

In [117]:
```
# Instantiate GridSearchCV
dt_grid_search = GridSearchCV(dt_clf, dt_param_grid, cv=3, return_train_score=True)

# Fit to the data
dt_grid_search.fit(X_train, y_train)
```

Out[117]:
```
GridSearchCV(cv=3,
             estimator=DecisionTreeClassifier(class_weight='balanced',
                                              random_state=42),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [None, 2, 3, 4, 5, 6],
                         'min_samples_leaf': [1, 2, 3, 4, 5, 6],
                         'min_samples_split': [2, 5, 10]},
             return_train_score=True)
```

In [119]:
```
dt_gs_training_score = np.mean(dt_grid_search.cv_results_['mean_train_score'])

# Mean test score
dt_gs_testing_score = dt_grid_search.score(X_test, y_test)

print(f"Mean Training Score: {dt_gs_training_score :.2%}")
print(f"Mean Test Score: {dt_gs_testing_score :.2%}")
print("Best Parameter Combination Found During Grid Search:")
dt_grid_search.best_params_
```

```
Mean Training Score: 66.24%
Mean Test Score: 65.17%
Best Parameter Combination Found During Grid Search:
```

Out[119]:
```
{'criterion': 'gini',
 'max_depth': 6,
 'min_samples_leaf': 4,
 'min_samples_split': 2}
```

In [120]:
```
rf_clf = RandomForestClassifier(random_state = 42, class_weight = 'balanced')
mean_rf_cv_score = np.mean(cross_val_score(rf_clf, X_train, y_train, cv=3))

print(f"Mean Cross Validation Score for Random Forest Classifier: {mean_rf_cv_score :.2%}")
```

```
Mean Cross Validation Score for Random Forest Classifier: 66.52%
```

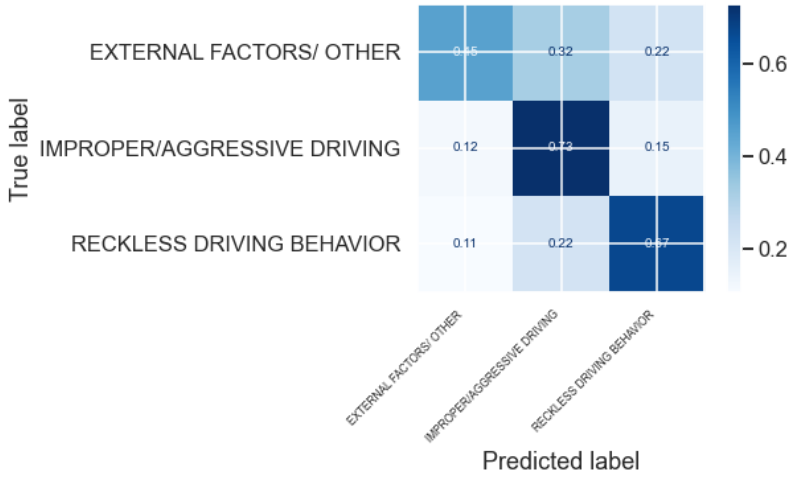
In [121]:
```
rf_param_grid = {
    'n_estimators': [10, 30, 100],
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 2, 6, 10],
    'min_samples_split': [5, 10],
    'min_samples_leaf': [3, 6]
}
```

In [122]:
```
rf_grid_search = GridSearchCV(rf_clf, rf_param_grid, cv=3)
rf_grid_search.fit(X_train, y_train)

print(f"Training Accuracy: {rf_grid_search.best_score_ :.2%}")
print("")
print(f"Optimal Parameters: {rf_grid_search.best_params_}")
```

```
Training Accuracy: 66.86%

Optimal Parameters: {'criterion': 'gini', 'max_depth': None, 'min_samples_leaf': 3, 'min_samples_split': 5, 'n_estimat
ors': 100}
```

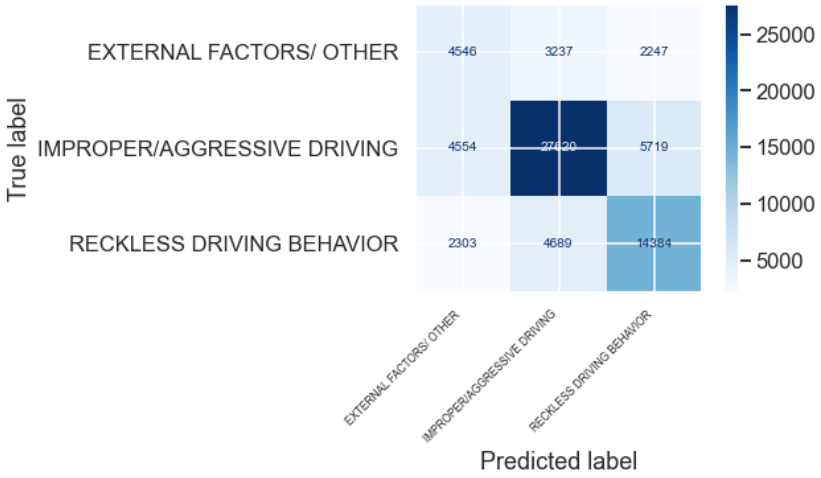
In [124]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(rf_grid_search, X_test, y_test,
                      cmap=plt.cm.Blues,normalize='true')
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small')
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



In [125]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(rf_grid_search, X_test, y_test,
                      cmap=plt.cm.Blues)
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small')
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



XGBoost Model

In [118]:
```
clf = xgb.XGBClassifier(random_state = 42)
clf.fit(X_train, y_train)
training_preds = clf.predict(X_train)
val_preds = clf.predict(X_test)
training_accuracy = accuracy_score(y_train, training_preds)
val_accuracy = accuracy_score(y_test, val_preds)

print("Training Accuracy: {:.4}%".format(training_accuracy * 100))
print("Validation accuracy: {:.4}%".format(val_accuracy * 100))
```

```
Training Accuracy: 70.9%
Validation accuracy: 69.83%
```

In [123]:
```
param_grid = {
    "early_stopping_rounds": [10],
    "learning_rate": [0.1],
    'max_depth': [5,6],
    'min_child_weight': [10],
    'subsample': [0.75],
    'n_estimators': [200, 250],}
```

In [*]:
```
grid_clf = GridSearchCV(clf, param_grid, scoring='accuracy', cv=None, n_jobs=1)
grid_clf.fit(X_train, y_train)

best_parameters = grid_clf.best_params_

print("Grid Search found the following optimal parameters: ")
for param_name in sorted(best_parameters.keys()):
    print("%s: %r" % (param_name, best_parameters[param_name]))

training_preds = grid_clf.predict(X_train)
val_preds = grid_clf.predict(X_test)
training_accuracy = accuracy_score(y_train, training_preds)
val_accuracy = accuracy_score(y_test, val_preds)

print("")
print("Training Accuracy: {:.4}%".format(training_accuracy * 100))
print("Validation accuracy: {:.4}%".format(val_accuracy * 100))
```

In [ ]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(grid_clf, X_train, y_train,
                      cmap=plt.cm.Blues,normalize='true')
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small')
plt.show()
```

In [ ]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(grid_clf,X_test, y_test,
                      cmap=plt.cm.Blues,normalize='true')
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small')
plt.show()
```

In [ ]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(grid_clf, X_test, y_test, values_format = '.0f',
                      cmap=plt.cm.Blues)
plt.xticks(
    rotation=45,
    horizontalalignment='right',
    fontsize='small')
plt.show()
```

K-Nearest Neighbor Model

In [ ]:
```
def best_k(X_train, y_train, X_test, y_test, min_k=1, max_k=10):
    best_k = 0
    best_score = 0.0
    for k in range(min_k, max_k+1, 2):
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(X_train, y_train)
        preds = knn.predict(X_test)
        f1 = f1_score(y_test, preds,average= 'weighted')
        if f1 > best_score:
            best_k = k
            best_score = f1

    print("Best Value for k: {}".format(best_k))
    print("F1-Score: {}".format(best_score))
```

In [ ]:
```
best_k(scaled_data_train, y_train, scaled_data_test, y_test)
```

In [ ]:
```
X_train = scaled_data_train
X_test = scaled_data_test
```

In [ ]:
```
plt.figure(figsize =(10,10))
plot_confusion_matrix(clf, X_test, y_test,
                      cmap=plt.cm.Blues,normalize='true')
plt.show()
```

In [ ]:
```

```

In [ ]:
```
clf = KNeighborsClassifier(n_neighbors = 9)

# Fit the classifier
clf.fit(X_train, y_train)

# Predict on the test set
test_preds = clf.predict(X_test)
```

In [ ]:
```
metrics(y_test, test_preds)
```

In [ ]:
```

```

Conclusions

From the information derived from classification and exploring trends supported by modeling, we believe that the City of Chicago should implement factors that will help reduce the amount of reckless driving behavior seen. For example, it may be beneifical to add more sobriety checkpoints on the weekends when there seems to be more fatal accidents than any other day of the week. Also using speed markers that make drivers aware of their speed or red light cameras at busy intersections could reduce the amount of fatal accidents related to failing to yield to the right-of-way and failure to reduce speed to avoid an accident. These free factors are the greatest in contributing to fatal accidents.

Future Work

Future work that will be useful is to continue to test our models with both historical data (data before 2015) and current crash incident data. This will allow us to continuously evaluate our models based on the data received and can help review whether the implementations recommended above proved effective in reducing fatal accidents and had a trickle down effect to less severe incidents. The City of Chicago could also coordinate with other large cities such as Los Angeles and New York City to see if there are factors that Chicago may be missing that can help improve driver's safety. It may also be beneficial to look at other cities with low fatality resulting accidents and see if any preventive measure they have in place can be scaled to a city such as Chicago and obtain a reduction of accidents. With the level of data it would be helpful to use models that are computationally less expensive as the size of our data made models like KNN and Decision trees impractical.

In [ ]:
```
1
```