```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Get read-only pandas dataframe object for dataset with given name.
df = vizierdb.get_data_frame('traffic_crashes')

# Use the columns attribute to list all columns
print('The initial shape of the dataframe is:')
# columns_list = df.columns.tolist()
print(df.shape)

# The attributes in column_to_drop are unnecessary for analysis, we will delete them.
column_to_drop = ['CRASH_RECORD_ID', 'RD_NO', 'CRASH_DATE_EST_I', 'REPORT_TYPE',
'STREET_NO', 'PHOTOS_TAKEN_I', 'STATEMENTS_TAKEN_I',
                  'WORKERS_PRESENT_I', 'INJURIES_UNKNOWN',
'INJURIES_INCAPACITATING', 'INJURIES_NON_INCAPACITATING',
'INJURIES_REPORTED_NOT_EVIDENT',
                  'INJURIES_NO_INDICATION', 'DAMAGE', 'DATE_POLICE_NOTIFIED',
'NUM_UNITS', 'STREET_DIRECTION', 'STREET_NAME', 'LANE_CNT', 'SEC_CONTRIBUTORY_CAUSE',
'DOORING_I']
# Use the drop() method to delete these unnecessary columns
df = df.drop(columns=column_to_drop)

print('The shape of the dataframe after drop the unneccessary columns is:')
# columns_list_1 = df.columns.tolist()
# print(columns_list_1)
print(df.shape)


# Use isnull() method to find missing values
missing_values = df.isnull()

# Count the number of missing values in each column
missing_count_per_column = missing_values.sum()

# Count the number of missing values in each column
total_missing_count = missing_values.sum().sum()

# Output the number of missing values in each column and the total number of missing values

print("\n Number of missing values per column:",missing_count_per_column)

print("\n Total number of missing values： ", total_missing_count)
```

```
# Remove features with too many missing values and fill features with fewer missing values with ‘unknown’.
missing_value_columns = ['INTERSECTION_RELATED_I','NOT_RIGHT_OF_WAY_I','HIT_AND_RUN_I','WORK_ZONE_I','WORK_ZONE_TYPE']
df = df.drop(columns=missing_value_columns)

# Fill missing values in INJURIES_TOTAL and INJURIES_FATAL using mean
df['INJURIES_TOTAL'].fillna(df['INJURIES_TOTAL'].mean(), inplace=True)
df['INJURIES_FATAL'].fillna(df['INJURIES_FATAL'].mean(), inplace=True)

# Fill missing values with 0 in the other features
# df.fillna(0, inplace=True)

# Get all different values of POSTED_SPEED_LIMIT feature
speed_limit_values = df['POSTED_SPEED_LIMIT'].unique()

# Output all different POSTED_SPEED_LIMIT values

print('\n These are the speed limit values:',speed_limit_values)



# We have speed limits that are not logged correctly, so we will drop them.
# There wasn't a lot so this will not effect our data
list_ = [3, 9, 0, 11]
for n in list_:
    df.drop(index=df[df['POSTED_SPEED_LIMIT'] == n].index, inplace=True)

print('\n The shape of the dataframe after drop the not logged correctly example and the five columns who has too much missing values:', df.shape)

# show the type of every features, and then perform next step processing based on the data type of the feature.
print('\n The type of every features are:',df.dtypes)

# Iterate through each column, print the unique values of each feature
for column in df.columns:
    unique_values = df[column].unique()
    print(f"Unique values for the feature '{column}':")
    print(unique_values)
    print("\n")
```

```
# normalize the numerical features
# normalize POSTED_SPEED_LIMIT
min_val_psl = min(df['POSTED_SPEED_LIMIT'])
max_val_psl = max(df['POSTED_SPEED_LIMIT'])
normalized_psl = [(x - min_val_psl) / (max_val_psl - min_val_psl) for x in df['POSTED_SPEED_LIMIT']]

# normalize INJURIES_TOTAL
min_val_it = min(df['INJURIES_TOTAL'])
max_val_it = max(df['INJURIES_TOTAL'])
normalized_it = [(x - min_val_it) / (max_val_it - min_val_it) for x in df['INJURIES_TOTAL']]

df['POSTED_SPEED_LIMIT'] = normalized_psl
df['INJURIES_TOTAL'] = normalized_it

# Classify features into categories

# Classify the MOST_SEVERE_INJURY feature into categories
severity_mapping = {
    'NO INDICATION OF INJURY': 'NO_INJURY',
    'NONINCAPACITATING INJURY': 'MINOR_INJURY',
    'REPORTED, NOT EVIDENT': 'NOT_EVIDENT',
    'INCAPACITATING INJURY': 'INCAPACITATING',
    'UNKNOWN': 'UNKNOWN',
    'FATAL': 'FATAL'
}
df['MOST_SEVERE_INJURY'] = df['MOST_SEVERE_INJURY'].map(severity_mapping)

# Classify CRASH_TYPE features into categories
crash_type_mapping = {
    'NO INJURY / DRIVE AWAY': 'NO_INJURY',
    'INJURY AND / OR TOW DUE TO CRASH': 'INJURY_OR_TOW'
}
df['CRASH_TYPE'] = df['CRASH_TYPE'].map(crash_type_mapping)

# One-hot encoding of TRAFFIC_CONTROL_DEVICE features
df = pd.get_dummies(df, columns=['TRAFFIC_CONTROL_DEVICE'], prefix='TRAFFIC_CONTROL')


# features is a list of features to be Label Encoded
features = ['DEVICE_CONDITION', 'WEATHER_CONDITION', 'LIGHTING_CONDITION', 'FIRST_CRASH_TYPE',
            'TRAFFICWAY_TYPE', 'ALIGNMENT', 'ROADWAY_SURFACE_COND', 'ROAD_DEFECT']
```

```
# Label Encode each feature and add to the original DataFrame
for feature in features:
    df[feature + '_encoded'] = pd.factorize(df[feature])[0]

# Show the encoded results
print('\n The df after encoded is:',df.head())
```

Screenshot:

```
The initial shape of the dataframe is:
(7822, 49)
The shape of the dataframe after drop the unneccessary columns is:
(7822, 28)
Number of missing values per column:
CRASH_DATE                   0
POSTED_SPEED_LIMIT           0
TRAFFIC_CONTROL_DEVICE       0
DEVICE_CONDITION             0
WEATHER_CONDITION            0
LIGHTING_CONDITION           0
FIRST_CRASH_TYPE             0
TRAFFICWAY_TYPE              0
ALIGNMENT                    0
ROADWAY_SURFACE_COND         0
ROAD_DEFECT                  0
CRASH_TYPE                   0
INTERSECTION_RELATED_I    6038
NOT_RIGHT_OF_WAY_I        7492
HIT_AND_RUN_I             5403
PRIM_CONTRIBUTORY_CAUSE      0
BEAT_OF_OCCURRENCE           0
WORK_ZONE_I               7770
WORK_ZONE_TYPE            7787
MOST_SEVERE_INJURY          20
INJURIES_TOTAL              20
INJURIES_FATAL              20
CRASH_HOUR                   0
CRASH_DAY_OF_WEEK            0
CRASH_MONTH                  0
LATITUDE                    64
LONGITUDE                   64
LOCATION                    64
dtype: int64

Total number of missing values:
34742
These are the speed limit values:
[30 35 20 15 25  5 55 10 45 50  0 40 11  3  9]
The shape of the dataframe after drop the not logged correctly example and the five columns who has too much missing values:
(7737, 23)
The type of every features are:
CRASH_DATE                 object
POSTED_SPEED_LIMIT          int16
TRAFFIC_CONTROL_DEVICE     object
DEVICE_CONDITION           object
WEATHER_CONDITION          object
LIGHTING_CONDITION         object
FIRST_CRASH_TYPE           object
TRAFFICWAY_TYPE            object
ALIGNMENT                  object
ROADWAY_SURFACE_COND       object
```

```
PRIM_CONTRIBUTORY_CAUSE     object
BEAT_OF_OCCURRENCE         float32
MOST_SEVERE_INJURY          object
INJURIES_TOTAL             float32
INJURIES_FATAL             float32
CRASH_HOUR                   int16
CRASH_DAY_OF_WEEK            int16
CRASH_MONTH                  int16
LATITUDE                   float32
LONGITUDE                  float32
LOCATION                    object
dtype: object

Unique values for the feature 'CRASH_DATE':
['12/13/2016 02:00:00 PM' '04/11/2022 03:00:00 AM'
 '12/09/2022 02:25:00 PM' ... '01/01/2018 02:00:00 AM'
 '05/18/2023 04:18:00 PM' '07/09/2019 11:30:00 PM']



Unique values for the feature 'POSTED_SPEED_LIMIT':
[30 35 20 15 25  5 55 10 45 50 40]


Unique values for the feature 'TRAFFIC_CONTROL_DEVICE':
['NO CONTROLS' 'TRAFFIC SIGNAL' 'STOP SIGN/FLASHER' 'UNKNOWN' 'OTHER'
 'DELINEATORS' 'RAILROAD CROSSING GATE' 'LANE USE MARKING'
 'OTHER REG. SIGN' 'PEDESTRIAN CROSSING SIGN' 'YIELD' 'POLICE/FLAGMAN'
 'SCHOOL ZONE' 'RR CROSSING SIGN' 'OTHER WARNING SIGN'
 'OTHER RAILROAD CROSSING' 'FLASHING CONTROL SIGNAL']



Unique values for the feature 'DEVICE_CONDITION':
['NO CONTROLS' 'FUNCTIONING PROPERLY' 'UNKNOWN' 'OTHER'
 'FUNCTIONING IMPROPERLY' 'NOT FUNCTIONING' 'WORN REFLECTIVE MATERIAL'
 'MISSING']



Unique values for the feature 'WEATHER_CONDITION':
['CLEAR' 'OTHER' 'RAIN' 'SNOW' 'CLOUDY/OVERCAST' 'UNKNOWN' 'SLEET/HAIL'
 'FREEZING RAIN/DRIZZLE' 'FOG/SMOKE/HAZE' 'BLOWING SNOW']



Unique values for the feature 'LIGHTING_CONDITION':
['DAYLIGHT' 'DARKNESS, LIGHTED ROAD' 'DAWN' 'DUSK' 'DARKNESS' 'UNKNOWN']


Unique values for the feature 'FIRST_CRASH_TYPE':
['TURNING' 'SIDESWIPE SAME DIRECTION' 'PARKED MOTOR VEHICLE' 'REAR END'
 'ANGLE' 'SIDESWIPE OPPOSITE DIRECTION' 'FIXED OBJECT' 'PEDESTRIAN'
```

```
['NO INDICATION OF INJURY' 'NONINCAPACITATING INJURY'
 'REPORTED, NOT EVIDENT' 'INCAPACITATING INJURY' None 'FATAL']



Unique values for the feature 'INJURIES_TOTAL':
[ 0.          3.          1.          2.          0.18892592  4.
  6.          5.         10.        ]



Unique values for the feature 'INJURIES_FATAL':
[0.0000000e+00 7.6903356e-04 1.0000000e+00]


Unique values for the feature 'CRASH_HOUR':
[14  3  7  6 10 13 17 15 11  9 12  8 19 23  2 16 21 22 18  0 20  5  1  4]


Unique values for the feature 'CRASH_DAY_OF_WEEK':
[3 2 6 7 5 1 4]


Unique values for the feature 'CRASH_MONTH':
[12  4  1  3  7  9  8  6  5 10 11  2]


Unique values for the feature 'LATITUDE':
[41.90076  41.891975 41.707806 ... 41.732323 41.921925 41.88364 ]


Unique values for the feature 'LONGITUDE':
[-87.62262  -87.74585  -87.690155 ... -87.55254  -87.6487   -87.745544]


Unique values for the feature 'LOCATION':
['POINT (-87.622617710596 41.900761132456)'
 'POINT (-87.745847370613 41.891975891594)'
 'POINT (-87.690154606691 41.707805222578)' ...
 'POINT (-87.552546294868 41.732321664578)'
 'POINT (-87.648697090238 41.921925643987)'
 'POINT (-87.74554121068 41.883639357187)']



The df after encoded is:
CRASH_DATE  ...  ROAD_DEFECT_encoded
0  12/13/2016 02:00:00 PM  ...                    0
1  04/11/2022 03:00:00 AM  ...                    0
2  12/09/2022 02:25:00 PM  ...                    0
3  01/26/2016 07:30:00 AM  ...                    0
4  03/14/2020 06:10:00 AM  ...                    0
```