I did a search among the training dataset,

```
print(train.isnull().sum())
```

✓ 0.0s

```
_id                    0
OFFENSE_CODE           0
OFFENSE_DESCRIPTION    0
DISTRICT               0
REPORTING_AREA         0
SHOOTING               0
OCCURRED_ON_DATE       0
YEAR                   0
MONTH                  0
DAY_OF_WEEK            0
HOUR                   0
STREET                 0
Severe_crimes          0
dtype: int64
```

```
# find duplicate rows
print(train.duplicated().sum())
```

✓ 0.0s

```
0
```

```
# remove id column
train = train.drop(columns=['_id'])
# find duplicate rows
print(train.duplicated().sum())
```

✓ 0.0s

```
185
```

```
# remove duplicate rows
train = train.drop_duplicates()
# find duplicate rows
print(train.duplicated().sum())
```

✓ 0.0s

```
0
```

Thanks to Boston Government, this dataset is really clean and there are no missing values in the dataset, there are some duplicates but I managed to remove them.

I remove year variable and the year part in the OCCURRED_ON_DATE column since the test data will be on 2024, I don't want the year to have influence on the model. The OCCURRED_ON_DATE column is also changed to datetime format for better use of model.

These dataset are highly involved with non-numerical values, so a encoder is necessary, I choose the label encoder from Sklearn kit and used it on the training set first and save the encoder use joblib for later use(decode the data or use it on test and val data).

After checking all variables' encoder with the training dataset, I use the same encoder for the test and validation set and do the same process individually just to prevent data leakage.

```
# do the same for test and val data
test = test.drop(columns=['_id', 'REPORTING_AREA', 'SHOOTING', 'YEAR'])
test['OCCURRED_ON_DATE'] = pd.to_datetime(test['OCCURRED_ON_DATE'])
test['DAY_OF_WEEK'] = test['OCCURRED_ON_DATE'].dt.dayofweek
test['OCCURRED_ON_DATE'] = test['OCCURRED_ON_DATE'].dt.strftime('%m-%d')
test['OFFENSE_DESCRIPTION'] = le.transform(test['OFFENSE_DESCRIPTION'])
test['DISTRICT'] = le.transform(test['DISTRICT'])
test['STREET'] = le.transform(test['STREET'])

val = val.drop(columns=['_id', 'REPORTING_AREA', 'SHOOTING', 'YEAR'])
val['OCCURRED_ON_DATE'] = pd.to_datetime(val['OCCURRED_ON_DATE'])
val['DAY_OF_WEEK'] = val['OCCURRED_ON_DATE'].dt.dayofweek
val['OCCURRED_ON_DATE'] = val['OCCURRED_ON_DATE'].dt.strftime('%m-%d')
val['OFFENSE_DESCRIPTION'] = le.transform(val['OFFENSE_DESCRIPTION'])
val['DISTRICT'] = le.transform(val['DISTRICT'])
val['STREET'] = le.transform(val['STREET'])

# save the processed data
train.to_csv('../data/processed/train_data_processed.csv', index=False)
test.to_csv('../data/processed/test_data_processed.csv', index=False)
val.to_csv('../data/processed/val_data_processed.csv', index=False)
```