Space Time Exploration of Metro-Police Crime Data April 2021 - March 2023

Installing and Importing all Libraries

In [22]:
```
!pip install fsspec
!pip install jupyterlab-rise
!pip install seaborn
!pip install scipy
!pip install statsmodels
!pip install folium
!pip install scikit-learn
!pip install folium
!pip install scikit-learn
#!pip install contextily as ctx
!pip install seaborn matplotlib contextily

import os
import shutil
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from statsmodels.tsa.seasonal import STL
from statsmodels.api import OLS, add_constant
import folium
from folium.plugins import HeatMap
from sklearn.cluster import DBSCAN
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import DBSCAN
```

```
Requirement already satisfied: fsspec in c:\users\jckat\anaconda3\envs\space_time_exploration_of
_metro_police\lib\site-packages (2024.5.0)
Requirement already satisfied: jupyterlab-rise in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (0.42.0)
Requirement already satisfied: jupyter-server<3,>=2.0.1 in c:\users\jckat\anaconda3\envs\space_t
ime_exploration_of_metro_police\lib\site-packages (from jupyterlab-rise) (2.10.0)
Requirement already satisfied: anyio>=3.1.0 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (4.2.0)
Requirement already satisfied: argon2-cffi in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (21.3.0)
Requirement already satisfied: jinja2 in c:\users\jckat\anaconda3\envs\space_time_exploration_of
_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (3.1.3)
Requirement already satisfied: jupyter-client>=7.4.4 in c:\users\jckat\anaconda3\envs\space_time
_exploration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise)
(8.6.0)
Requirement already satisfied: jupyter-core!=5.0.*,>=4.12 in c:\users\jckat\anaconda3\envs\space
_time_exploration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-r
ise) (5.5.0)
Requirement already satisfied: jupyter-events>=0.6.0 in c:\users\jckat\anaconda3\envs\space_time
_exploration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise)
(0.8.0)
Requirement already satisfied: jupyter-server-terminals in c:\users\jckat\anaconda3\envs\space_t
ime_exploration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-ris
e) (0.4.4)
Requirement already satisfied: nbconvert>=6.4.4 in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (7.1
0.0)
Requirement already satisfied: nbformat>=5.3.0 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (5.9.
2)
Requirement already satisfied: overrides in c:\users\jckat\anaconda3\envs\space_time_exploration
_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (7.4.0)
Requirement already satisfied: packaging in c:\users\jckat\anaconda3\envs\space_time_exploration
_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (23.2)
Requirement already satisfied: prometheus-client in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (0.1
4.1)
Requirement already satisfied: pywinpty in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (2.0.10)
Requirement already satisfied: pyzmq>=24 in c:\users\jckat\anaconda3\envs\space_time_exploration
_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (25.1.2)
Requirement already satisfied: send2trash>=1.8.2 in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (1.
8.2)
Requirement already satisfied: terminado>=0.8.3 in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (0.1
7.1)
Requirement already satisfied: tornado>=6.2.0 in c:\users\jckat\anaconda3\envs\space_time_explor
ation_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (6.3.3)
Requirement already satisfied: traitlets>=5.6.0 in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (5.7.
1)
Requirement already satisfied: websocket-client in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from jupyter-server<3,>=2.0.1->jupyterlab-rise) (0.5
8.0)
Requirement already satisfied: idna>=2.8 in c:\users\jckat\anaconda3\envs\space_time_exploration
_of_metro_police\lib\site-packages (from anyio>=3.1.0->jupyter-server<3,>=2.0.1->jupyterlab-ris
e) (3.7)
Requirement already satisfied: sniffio>=1.1 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from anyio>=3.1.0->jupyter-server<3,>=2.0.1->jupyterlab-r
ise) (1.3.0)
Requirement already satisfied: exceptiongroup>=1.0.2 in c:\users\jckat\anaconda3\envs\space_time
_exploration_of_metro_police\lib\site-packages (from anyio>=3.1.0->jupyter-server<3,>=2.0.1->jup
```

```
yterlab-rise) (1.2.0)
Requirement already satisfied: typing-extensions>=4.1 in c:\users\jckat\anaconda3\envs\space_tim
e_exploration_of_metro_police\lib\site-packages (from anyio>=3.1.0->jupyter-server<3,>=2.0.1->ju
pyterlab-rise) (4.11.0)
Requirement already satisfied: importlib-metadata>=4.8.3 in c:\users\jckat\anaconda3\envs\space_
time_exploration_of_metro_police\lib\site-packages (from jupyter-client>=7.4.4->jupyter-server<
3,>=2.0.1->jupyterlab-rise) (7.0.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\jckat\anaconda3\envs\space_tim
e_exploration_of_metro_police\lib\site-packages (from jupyter-client>=7.4.4->jupyter-server<3,>=
2.0.1->jupyterlab-rise) (2.8.2)
Requirement already satisfied: platformdirs>=2.5 in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from jupyter-core!=5.0.*,>=4.12->jupyter-server<3,>=
2.0.1->jupyterlab-rise) (3.10.0)
Requirement already satisfied: pywin32>=300 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from jupyter-core!=5.0.*,>=4.12->jupyter-server<3,>=2.0.1
->jupyterlab-rise) (305.1)
Requirement already satisfied: jsonschema>=4.18.0 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from jsonschema[format-nongpl]>=4.18.0->jupyter-eve
nts>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (4.19.2)
Requirement already satisfied: python-json-logger>=2.0.4 in c:\users\jckat\anaconda3\envs\space_
time_exploration_of_metro_police\lib\site-packages (from jupyter-events>=0.6.0->jupyter-server<
3,>=2.0.1->jupyterlab-rise) (2.0.7)
Requirement already satisfied: pyyaml>=5.3 in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupy
terlab-rise) (6.0.1)
Requirement already satisfied: referencing in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupy
terlab-rise) (0.30.2)
Requirement already satisfied: rfc3339-validator in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1
->jupyterlab-rise) (0.1.4)
Requirement already satisfied: rfc3986-validator>=0.1.1 in c:\users\jckat\anaconda3\envs\space_t
ime_exploration_of_metro_police\lib\site-packages (from jupyter-events>=0.6.0->jupyter-server<3,
>=2.0.1->jupyterlab-rise) (0.1.1)
Requirement already satisfied: beautifulsoup4 in c:\users\jckat\anaconda3\envs\space_time_explor
ation_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyte
rlab-rise) (4.12.2)
Requirement already satisfied: bleach!=5.0.0 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyter
lab-rise) (4.1.0)
Requirement already satisfied: defusedxml in c:\users\jckat\anaconda3\envs\space_time_exploratio
n_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyterlab
-rise) (0.7.1)
Requirement already satisfied: jupyterlab-pygments in c:\users\jckat\anaconda3\envs\space_time_e
xploration_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->j
upyterlab-rise) (0.1.2)
Requirement already satisfied: markupsafe>=2.0 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyt
erlab-rise) (2.1.3)
Requirement already satisfied: mistune<4,>=2.0.3 in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jup
yterlab-rise) (2.0.4)
Requirement already satisfied: nbclient>=0.5.0 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyt
erlab-rise) (0.8.0)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\jckat\anaconda3\envs\space_time_
exploration_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->
jupyterlab-rise) (1.5.0)
Requirement already satisfied: pygments>=2.4.1 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyt
erlab-rise) (2.15.1)
Requirement already satisfied: tinycss2 in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from nbconvert>=6.4.4->jupyter-server<3,>=2.0.1->jupyterlab-r
```

```
ise) (1.2.1)
Requirement already satisfied: fastjsonschema in c:\users\jckat\anaconda3\envs\space_time_explor
ation_of_metro_police\lib\site-packages (from nbformat>=5.3.0->jupyter-server<3,>=2.0.1->jupyter
lab-rise) (2.16.2)
Requirement already satisfied: argon2-cffi-bindings in c:\users\jckat\anaconda3\envs\space_time_
exploration_of_metro_police\lib\site-packages (from argon2-cffi->jupyter-server<3,>=2.0.1->jupyt
erlab-rise) (21.2.0)
Requirement already satisfied: six in c:\users\jckat\anaconda3\envs\space_time_exploration_of_me
tro_police\lib\site-packages (from websocket-client->jupyter-server<3,>=2.0.1->jupyterlab-rise)
(1.16.0)
Requirement already satisfied: webencodings in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from bleach!=5.0.0->nbconvert>=6.4.4->jupyter-server<3,>=
2.0.1->jupyterlab-rise) (0.5.1)
Requirement already satisfied: zipp>=0.5 in c:\users\jckat\anaconda3\envs\space_time_exploration
_of_metro_police\lib\site-packages (from importlib-metadata>=4.8.3->jupyter-client>=7.4.4->jupyt
er-server<3,>=2.0.1->jupyterlab-rise) (3.17.0)
Requirement already satisfied: attrs>=22.2.0 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from jsonschema>=4.18.0->jsonschema[format-nongpl]>=4.1
8.0->jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (23.1.0)
Requirement already satisfied: importlib-resources>=1.4.0 in c:\users\jckat\anaconda3\envs\space
_time_exploration_of_metro_police\lib\site-packages (from jsonschema>=4.18.0->jsonschema[format-
nongpl]>=4.18.0->jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (6.1.1)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in c:\users\jckat\anaconda3
\envs\space_time_exploration_of_metro_police\lib\site-packages (from jsonschema>=4.18.0->jsonsch
ema[format-nongpl]>=4.18.0->jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (2
023.7.1)
Requirement already satisfied: pkgutil-resolve-name>=1.3.10 in c:\users\jckat\anaconda3\envs\spa
ce_time_exploration_of_metro_police\lib\site-packages (from jsonschema>=4.18.0->jsonschema[forma
t-nongpl]>=4.18.0->jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (1.3.10)
Requirement already satisfied: rpds-py>=0.7.1 in c:\users\jckat\anaconda3\envs\space_time_explor
ation_of_metro_police\lib\site-packages (from jsonschema>=4.18.0->jsonschema[format-nongpl]>=4.1
8.0->jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (0.10.6)
Requirement already satisfied: fqdn in c:\users\jckat\anaconda3\envs\space_time_exploration_of_m
etro_police\lib\site-packages (from jsonschema[format-nongpl]>=4.18.0->jupyter-events>=0.6.0->ju
pyter-server<3,>=2.0.1->jupyterlab-rise) (1.5.1)
Requirement already satisfied: isoduration in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from jsonschema[format-nongpl]>=4.18.0->jupyter-events>=0.
6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (20.11.0)
Requirement already satisfied: jsonpointer>1.13 in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from jsonschema[format-nongpl]>=4.18.0->jupyter-event
s>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (2.4)
Requirement already satisfied: uri-template in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from jsonschema[format-nongpl]>=4.18.0->jupyter-events>=
0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (1.3.0)
Requirement already satisfied: webcolors>=1.11 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from jsonschema[format-nongpl]>=4.18.0->jupyter-events
>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (1.13)
Requirement already satisfied: cffi>=1.0.1 in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from argon2-cffi-bindings->argon2-cffi->jupyter-server<3,>
=2.0.1->jupyterlab-rise) (1.16.0)
Requirement already satisfied: soupsieve>1.2 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from beautifulsoup4->nbconvert>=6.4.4->jupyter-server<3,
>=2.0.1->jupyterlab-rise) (2.5)
Requirement already satisfied: pycparser in c:\users\jckat\anaconda3\envs\space_time_exploration
_of_metro_police\lib\site-packages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->jupyter
-server<3,>=2.0.1->jupyterlab-rise) (2.21)
Requirement already satisfied: arrow>=0.15.0 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from isoduration->jsonschema[format-nongpl]>=4.18.0->jup
yter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise) (1.3.0)
Requirement already satisfied: types-python-dateutil>=2.8.10 in c:\users\jckat\anaconda3\envs\sp
ace_time_exploration_of_metro_police\lib\site-packages (from arrow>=0.15.0->isoduration->jsonsch
ema[format-nongpl]>=4.18.0->jupyter-events>=0.6.0->jupyter-server<3,>=2.0.1->jupyterlab-rise)
(2.9.0.20240316)
```

```
Requirement already satisfied: seaborn in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from seaborn) (1.24.3)
Requirement already satisfied: pandas>=1.2 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from seaborn) (2.0.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from seaborn) (3.7.5)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.1.1)
Requirement already satisfied: cycler>=0.10 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (23.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2.8.2)
Requirement already satisfied: importlib-resources>=3.2.0 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (6.1.1)
Requirement already satisfied: pytz>=2020.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from pandas>=1.2->seaborn) (2023.3)
Requirement already satisfied: zipp>=3.1.0 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from importlib-resources>=3.2.0->matplotlib!=3.6.1,>=3.4->seaborn) (3.17.0)
Requirement already satisfied: six>=1.5 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
Requirement already satisfied: scipy in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (1.10.1)
Requirement already satisfied: numpy<1.27.0,>=1.19.5 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from scipy) (1.24.3)
Requirement already satisfied: statsmodels in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (0.14.1)
Requirement already satisfied: numpy<2,>=1.18 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from statsmodels) (1.24.3)
Requirement already satisfied: scipy!=1.9.2,>=1.4 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from statsmodels) (1.10.1)
Requirement already satisfied: pandas!=2.1.0,>=1.0 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from statsmodels) (2.0.3)
Requirement already satisfied: patsy>=0.5.4 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from statsmodels) (0.5.6)
Requirement already satisfied: packaging>=21.3 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from statsmodels) (23.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from pandas!=2.1.0,>=1.0->statsmodels) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from pandas!=2.1.0,>=1.0->statsmodels) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from pandas!=2.1.0,>=1.0->statsmodels) (2023.3)
Requirement already satisfied: six in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (from patsy>=0.5.4->statsmodels) (1.16.0)
Requirement already satisfied: folium in c:\users\jckat\anaconda3\envs\space_time_exploration_of_metro_police\lib\site-packages (0.16.0)
```

```
Requirement already satisfied: branca>=0.6.0 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from folium) (0.7.2)
Requirement already satisfied: jinja2>=2.9 in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from folium) (3.1.3)
Requirement already satisfied: numpy in c:\users\jckat\anaconda3\envs\space_time_exploration_of_
metro_police\lib\site-packages (from folium) (1.24.3)
Requirement already satisfied: requests in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from folium) (2.31.0)
Requirement already satisfied: xyzservices in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from folium) (2024.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from jinja2>=2.9->folium) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\jckat\anaconda3\envs\space_t
ime_exploration_of_metro_police\lib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from requests->folium) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from requests->folium) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from requests->folium) (2024.2.2)
Requirement already satisfied: scikit-learn in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (1.3.2)
Requirement already satisfied: numpy<2.0,>=1.17.3 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from scikit-learn) (1.24.3)
Requirement already satisfied: scipy>=1.5.0 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from scikit-learn) (1.10.1)
Requirement already satisfied: joblib>=1.1.1 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\jckat\anaconda3\envs\space_time_
exploration_of_metro_police\lib\site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: folium in c:\users\jckat\anaconda3\envs\space_time_exploration_of
_metro_police\lib\site-packages (0.16.0)
Requirement already satisfied: branca>=0.6.0 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from folium) (0.7.2)
Requirement already satisfied: jinja2>=2.9 in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from folium) (3.1.3)
Requirement already satisfied: numpy in c:\users\jckat\anaconda3\envs\space_time_exploration_of_
metro_police\lib\site-packages (from folium) (1.24.3)
Requirement already satisfied: requests in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from folium) (2.31.0)
Requirement already satisfied: xyzservices in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from folium) (2024.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from jinja2>=2.9->folium) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\jckat\anaconda3\envs\space_t
ime_exploration_of_metro_police\lib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from requests->folium) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from requests->folium) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from requests->folium) (2024.2.2)
Requirement already satisfied: scikit-learn in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (1.3.2)
Requirement already satisfied: numpy<2.0,>=1.17.3 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from scikit-learn) (1.24.3)
Requirement already satisfied: scipy>=1.5.0 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from scikit-learn) (1.10.1)
Requirement already satisfied: joblib>=1.1.1 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\jckat\anaconda3\envs\space_time_
exploration_of_metro_police\lib\site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: seaborn in c:\users\jckat\anaconda3\envs\space_time_exploration_o
```

```
f_metro_police\lib\site-packages (0.13.2)
Requirement already satisfied: matplotlib in c:\users\jckat\anaconda3\envs\space_time_exploratio
n_of_metro_police\lib\site-packages (3.7.5)
Collecting contextily
  Using cached contextily-1.5.2-py3-none-any.whl.metadata (2.9 kB)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\jckat\anaconda3\envs\space_time_
exploration_of_metro_police\lib\site-packages (from seaborn) (1.24.3)
Requirement already satisfied: pandas>=1.2 in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from seaborn) (2.0.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from matplotlib) (1.1.1)
Requirement already satisfied: cycler>=0.10 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jckat\anaconda3\envs\space_time_exp
loration_of_metro_police\lib\site-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in c:\users\jckat\anaconda3\envs\space_time_explo
ration_of_metro_police\lib\site-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\jckat\anaconda3\envs\space_time_explora
tion_of_metro_police\lib\site-packages (from matplotlib) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\jckat\anaconda3\envs\space_time_expl
oration_of_metro_police\lib\site-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jckat\anaconda3\envs\space_time_
exploration_of_metro_police\lib\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: importlib-resources>=3.2.0 in c:\users\jckat\anaconda3\envs\space
_time_exploration_of_metro_police\lib\site-packages (from matplotlib) (6.1.1)
Collecting geopy (from contextily)
  Using cached geopy-2.4.1-py3-none-any.whl.metadata (6.8 kB)
Collecting mercantile (from contextily)
  Downloading mercantile-1.2.1-py3-none-any.whl.metadata (4.8 kB)
Collecting rasterio (from contextily)
  Downloading rasterio-1.3.10-cp38-cp38-win_amd64.whl.metadata (15 kB)
Requirement already satisfied: requests in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from contextily) (2.31.0)
Requirement already satisfied: joblib in c:\users\jckat\anaconda3\envs\space_time_exploration_of
_metro_police\lib\site-packages (from contextily) (1.4.2)
Requirement already satisfied: xyzservices in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from contextily) (2024.4.0)
Requirement already satisfied: zipp>=3.1.0 in c:\users\jckat\anaconda3\envs\space_time_explorati
on_of_metro_police\lib\site-packages (from importlib-resources>=3.2.0->matplotlib) (3.17.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\jckat\anaconda3\envs\space_time_explor
ation_of_metro_police\lib\site-packages (from pandas>=1.2->seaborn) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Collecting geographiclib<3,>=1.52 (from geopy->contextily)
  Downloading geographiclib-2.0-py3-none-any.whl.metadata (1.4 kB)
Collecting click>=3.0 (from mercantile->contextily)
  Using cached click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting affine (from rasterio->contextily)
  Downloading affine-2.4.0-py3-none-any.whl.metadata (4.0 kB)
Requirement already satisfied: attrs in c:\users\jckat\anaconda3\envs\space_time_exploration_of_
metro_police\lib\site-packages (from rasterio->contextily) (23.1.0)
Requirement already satisfied: certifi in c:\users\jckat\anaconda3\envs\space_time_exploration_o
f_metro_police\lib\site-packages (from rasterio->contextily) (2024.2.2)
Collecting cligj>=0.5 (from rasterio->contextily)
  Using cached cligj-0.7.2-py3-none-any.whl.metadata (5.0 kB)
Collecting snuggs>=1.4.1 (from rasterio->contextily)
  Downloading snuggs-1.4.7-py3-none-any.whl.metadata (3.4 kB)
Collecting click-plugins (from rasterio->contextily)
  Using cached click_plugins-1.1.1-py2.py3-none-any.whl.metadata (6.4 kB)
```

```
Requirement already satisfied: setuptools in c:\users\jckat\anaconda3\envs\space_time_exploratio
n_of_metro_police\lib\site-packages (from rasterio->contextily) (69.5.1)
Requirement already satisfied: importlib-metadata in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from rasterio->contextily) (7.0.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\jckat\anaconda3\envs\space_t
ime_exploration_of_metro_police\lib\site-packages (from requests->contextily) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jckat\anaconda3\envs\space_time_explorat
ion_of_metro_police\lib\site-packages (from requests->contextily) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\jckat\anaconda3\envs\space_time_ex
ploration_of_metro_police\lib\site-packages (from requests->contextily) (2.2.1)
Requirement already satisfied: colorama in c:\users\jckat\anaconda3\envs\space_time_exploration_
of_metro_police\lib\site-packages (from click>=3.0->mercantile->contextily) (0.4.6)
Downloading contextily-1.5.2-py3-none-any.whl (17 kB)
Using cached geopy-2.4.1-py3-none-any.whl (125 kB)
Downloading mercantile-1.2.1-py3-none-any.whl (14 kB)
Downloading rasterio-1.3.10-cp38-cp38-win_amd64.whl (24.4 MB)
   ---------------------------------------- 0.0/24.4 MB ? eta -:--:--
   ---------------------------------------- 0.0/24.4 MB 991.0 kB/s eta 0:00:25
   ---------------------------------------- 0.1/24.4 MB 1.3 MB/s eta 0:00:19
   ---------------------------------------- 0.2/24.4 MB 1.3 MB/s eta 0:00:19
   ---------------------------------------- 0.3/24.4 MB 1.4 MB/s eta 0:00:17
    --------------------------------------- 0.3/24.4 MB 1.5 MB/s eta 0:00:16
    --------------------------------------- 0.4/24.4 MB 1.5 MB/s eta 0:00:16
    --------------------------------------- 0.5/24.4 MB 1.6 MB/s eta 0:00:15
   - --------------------------------------- 0.6/24.4 MB 1.7 MB/s eta 0:00:14
   - --------------------------------------- 0.7/24.4 MB 1.8 MB/s eta 0:00:14
   - --------------------------------------- 0.9/24.4 MB 1.9 MB/s eta 0:00:13
   - --------------------------------------- 1.0/24.4 MB 2.0 MB/s eta 0:00:12
   - --------------------------------------- 1.1/24.4 MB 2.0 MB/s eta 0:00:12
   -- -------------------------------------- 1.3/24.4 MB 2.1 MB/s eta 0:00:11
   -- -------------------------------------- 1.4/24.4 MB 2.2 MB/s eta 0:00:11
   -- -------------------------------------- 1.6/24.4 MB 2.3 MB/s eta 0:00:10
   -- -------------------------------------- 1.8/24.4 MB 2.4 MB/s eta 0:00:10
   --- ------------------------------------- 1.9/24.4 MB 2.4 MB/s eta 0:00:10
   --- ------------------------------------- 2.0/24.4 MB 2.4 MB/s eta 0:00:10
   --- ------------------------------------- 2.1/24.4 MB 2.4 MB/s eta 0:00:10
   --- ------------------------------------- 2.2/24.4 MB 2.4 MB/s eta 0:00:10
   --- ------------------------------------- 2.2/24.4 MB 2.3 MB/s eta 0:00:10
   --- ------------------------------------- 2.4/24.4 MB 2.3 MB/s eta 0:00:10
   ---- ------------------------------------ 2.5/24.4 MB 2.4 MB/s eta 0:00:10
   ---- ------------------------------------ 2.7/24.4 MB 2.4 MB/s eta 0:00:09
   ---- ------------------------------------ 2.9/24.4 MB 2.5 MB/s eta 0:00:09
   ----- ----------------------------------- 3.1/24.4 MB 2.5 MB/s eta 0:00:09
   ----- ----------------------------------- 3.3/24.4 MB 2.6 MB/s eta 0:00:09
   ----- ----------------------------------- 3.5/24.4 MB 2.7 MB/s eta 0:00:08
   ------ ---------------------------------- 3.8/24.4 MB 2.7 MB/s eta 0:00:08
   ------ ---------------------------------- 4.0/24.4 MB 2.8 MB/s eta 0:00:08
   ------- --------------------------------- 4.3/24.4 MB 2.9 MB/s eta 0:00:07
   ------- --------------------------------- 4.5/24.4 MB 3.0 MB/s eta 0:00:07
   ------- --------------------------------- 4.8/24.4 MB 3.1 MB/s eta 0:00:07
   -------- -------------------------------- 4.9/24.4 MB 3.1 MB/s eta 0:00:07
   -------- -------------------------------- 5.1/24.4 MB 3.1 MB/s eta 0:00:07
   -------- -------------------------------- 5.3/24.4 MB 3.1 MB/s eta 0:00:07
   --------- ------------------------------- 5.5/24.4 MB 3.1 MB/s eta 0:00:07
   --------- ------------------------------- 5.7/24.4 MB 3.2 MB/s eta 0:00:06
   --------- ------------------------------- 5.9/24.4 MB 3.2 MB/s eta 0:00:06
   --------- ------------------------------- 6.1/24.4 MB 3.2 MB/s eta 0:00:06
   ---------- ------------------------------ 6.2/24.4 MB 3.2 MB/s eta 0:00:06
   ---------- ------------------------------ 6.4/24.4 MB 3.2 MB/s eta 0:00:06
   ---------- ------------------------------ 6.6/24.4 MB 3.2 MB/s eta 0:00:06
   ----------- ----------------------------- 6.8/24.4 MB 3.2 MB/s eta 0:00:06
   ----------- ----------------------------- 6.9/24.4 MB 3.2 MB/s eta 0:00:06
   ----------- ----------------------------- 7.1/24.4 MB 3.3 MB/s eta 0:00:06
```

```
---------- ---------------------------- 7.3/24.4 MB 3.3 MB/s eta 0:00:06
---------- ---------------------------- 7.5/24.4 MB 3.3 MB/s eta 0:00:06
---------- ---------------------------- 7.7/24.4 MB 3.3 MB/s eta 0:00:06
---------- ---------------------------- 7.9/24.4 MB 3.4 MB/s eta 0:00:05
---------- ---------------------------- 8.2/24.4 MB 3.4 MB/s eta 0:00:05
---------- ---------------------------- 8.4/24.4 MB 3.4 MB/s eta 0:00:05
---------- ---------------------------- 8.6/24.4 MB 3.4 MB/s eta 0:00:05
---------- ---------------------------- 8.9/24.4 MB 3.5 MB/s eta 0:00:05
---------- ---------------------------- 9.1/24.4 MB 3.5 MB/s eta 0:00:05
---------- ---------------------------- 9.3/24.4 MB 3.5 MB/s eta 0:00:05
---------- ---------------------------- 9.6/24.4 MB 3.6 MB/s eta 0:00:05
---------- ---------------------------- 9.8/24.4 MB 3.6 MB/s eta 0:00:05
---------- ---------------------------- 10.0/24.4 MB 3.6 MB/s eta 0:00:05
---------- ---------------------------- 10.2/24.4 MB 3.6 MB/s eta 0:00:04
---------- ---------------------------- 10.3/24.4 MB 3.6 MB/s eta 0:00:04
---------- ---------------------------- 10.4/24.4 MB 3.6 MB/s eta 0:00:04
---------- ---------------------------- 10.6/24.4 MB 3.7 MB/s eta 0:00:04
---------- ---------------------------- 10.8/24.4 MB 3.8 MB/s eta 0:00:04
---------- ---------------------------- 11.0/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 11.2/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 11.3/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 11.4/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 11.6/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 11.7/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 11.8/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 12.0/24.4 MB 3.8 MB/s eta 0:00:04
---------- ---------------------------- 12.2/24.4 MB 3.8 MB/s eta 0:00:04
---------- ---------------------------- 12.3/24.4 MB 3.9 MB/s eta 0:00:04
---------- ---------------------------- 12.5/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 12.7/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 12.9/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 13.1/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 13.3/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 13.5/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 13.8/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 14.0/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 14.2/24.4 MB 4.1 MB/s eta 0:00:03
---------- ---------------------------- 14.4/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 14.7/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 14.9/24.4 MB 4.1 MB/s eta 0:00:03
---------- ---------------------------- 15.2/24.4 MB 4.1 MB/s eta 0:00:03
---------- ---------------------------- 15.3/24.4 MB 4.1 MB/s eta 0:00:03
---------- ---------------------------- 15.5/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 15.7/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 15.9/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 16.0/24.4 MB 4.0 MB/s eta 0:00:03
---------- ---------------------------- 16.1/24.4 MB 3.9 MB/s eta 0:00:03
---------- ---------------------------- 16.2/24.4 MB 3.9 MB/s eta 0:00:03
---------- ---------------------------- 16.4/24.4 MB 3.9 MB/s eta 0:00:03
---------- ---------------------------- 16.6/24.4 MB 3.9 MB/s eta 0:00:03
---------- ---------------------------- 16.7/24.4 MB 3.9 MB/s eta 0:00:02
---------- ---------------------------- 16.9/24.4 MB 3.9 MB/s eta 0:00:02
---------- ---------------------------- 17.0/24.4 MB 3.9 MB/s eta 0:00:02
---------- ---------------------------- 17.1/24.4 MB 3.9 MB/s eta 0:00:02
---------- ---------------------------- 17.3/24.4 MB 3.8 MB/s eta 0:00:02
---------- ---------------------------- 17.4/24.4 MB 3.8 MB/s eta 0:00:02
---------- ---------------------------- 17.5/24.4 MB 3.8 MB/s eta 0:00:02
---------- ---------------------------- 17.7/24.4 MB 3.7 MB/s eta 0:00:02
---------- ---------------------------- 17.8/24.4 MB 3.7 MB/s eta 0:00:02
---------- ---------------------------- 17.9/24.4 MB 3.7 MB/s eta 0:00:02
---------- ---------------------------- 18.0/24.4 MB 3.7 MB/s eta 0:00:02
---------- ---------------------------- 18.1/24.4 MB 3.6 MB/s eta 0:00:02
---------- ---------------------------- 18.1/24.4 MB 3.6 MB/s eta 0:00:02
```

```
   ---------------------------- ---------- 18.1/24.4 MB 3.5 MB/s eta 0:00:02
   ---------------------------- ---------- 18.3/24.4 MB 3.5 MB/s eta 0:00:02
   ---------------------------- --------- 18.4/24.4 MB 3.5 MB/s eta 0:00:02
   ----------------------------- --------- 18.6/24.4 MB 3.5 MB/s eta 0:00:02
   ----------------------------- --------- 18.7/24.4 MB 3.4 MB/s eta 0:00:02
   ----------------------------- --------- 18.9/24.4 MB 3.4 MB/s eta 0:00:02
   ------------------------------ -------- 19.0/24.4 MB 3.4 MB/s eta 0:00:02
   ------------------------------ -------- 19.1/24.4 MB 3.4 MB/s eta 0:00:02
   ------------------------------ -------- 19.3/24.4 MB 3.3 MB/s eta 0:00:02
   ------------------------------ -------- 19.4/24.4 MB 3.3 MB/s eta 0:00:02
   ------------------------------- ------- 19.6/24.4 MB 3.3 MB/s eta 0:00:02
   ------------------------------- ------- 19.7/24.4 MB 3.3 MB/s eta 0:00:02
   ------------------------------- ------- 19.9/24.4 MB 3.3 MB/s eta 0:00:02
   -------------------------------- ------- 20.1/24.4 MB 3.3 MB/s eta 0:00:02
   -------------------------------- ------ 20.3/24.4 MB 3.3 MB/s eta 0:00:02
   -------------------------------- ------ 20.5/24.4 MB 3.3 MB/s eta 0:00:02
   -------------------------------- ------ 20.6/24.4 MB 3.3 MB/s eta 0:00:02
   --------------------------------- ----- 20.8/24.4 MB 3.3 MB/s eta 0:00:02
   --------------------------------- ----- 21.0/24.4 MB 3.3 MB/s eta 0:00:02
   --------------------------------- ----- 21.2/24.4 MB 3.3 MB/s eta 0:00:01
   ---------------------------------- ---- 21.4/24.4 MB 3.3 MB/s eta 0:00:01
   ---------------------------------- ---- 21.6/24.4 MB 3.4 MB/s eta 0:00:01
   ---------------------------------- ---- 21.8/24.4 MB 3.4 MB/s eta 0:00:01
   ----------------------------------- --- 22.0/24.4 MB 3.4 MB/s eta 0:00:01
   ----------------------------------- --- 22.2/24.4 MB 3.4 MB/s eta 0:00:01
   ----------------------------------- --- 22.5/24.4 MB 3.5 MB/s eta 0:00:01
   ------------------------------------ -- 22.8/24.4 MB 3.5 MB/s eta 0:00:01
   ------------------------------------ -- 23.0/24.4 MB 3.5 MB/s eta 0:00:01
   ------------------------------------- - 23.2/24.4 MB 3.5 MB/s eta 0:00:01
   ------------------------------------- - 23.4/24.4 MB 3.5 MB/s eta 0:00:01
   ------------------------------------- - 23.6/24.4 MB 3.5 MB/s eta 0:00:01
   -------------------------------------  23.8/24.4 MB 3.5 MB/s eta 0:00:01
   -------------------------------------  24.0/24.4 MB 3.5 MB/s eta 0:00:01
   -------------------------------------  24.2/24.4 MB 3.5 MB/s eta 0:00:01
   -------------------------------------  24.4/24.4 MB 3.5 MB/s eta 0:00:01
   -------------------------------------- 24.4/24.4 MB 3.4 MB/s eta 0:00:00
Using cached click-8.1.7-py3-none-any.whl (97 kB)
Using cached cligj-0.7.2-py3-none-any.whl (7.1 kB)
Using cached geographiclib-2.0-py3-none-any.whl (40 kB)
Using cached snuggs-1.4.7-py3-none-any.whl (5.4 kB)
Using cached affine-2.4.0-py3-none-any.whl (15 kB)
Using cached click_plugins-1.1.1-py2.py3-none-any.whl (7.5 kB)
Installing collected packages: snuggs, geographiclib, click, affine, mercantile, geopy, cligj, c
lick-plugins, rasterio, contextily
Successfully installed affine-2.4.0 click-8.1.7 click-plugins-1.1.1 cligj-0.7.2 contextily-1.5.2
geographiclib-2.0 geopy-2.4.1 mercantile-1.2.1 rasterio-1.3.10 snuggs-1.4.7
```

Define the path to the main folder

This script efficiently automates the process of combining multiple CSV files into one, simplifying data management and analysis tasks.

1. Define Directory Path: Specify the path to the directory containing CSV files.
2. Initialize List: Create an empty list to hold DataFrames.

Directory Traversal: Use os.walk() to find all CSV files in the directory. * Check each file's extension. * Construct the full file path. * Read each CSV into a DataFrame. * Append the DataFrame to the list. 3. Concatenate DataFrames: Merge all DataFrames into a single DataFrame using pd.concat(). *Save to CSV: Write the merged DataFrame to a new CSV file named metro_police_crime.csv. 4. Confirmation: Print a message confirming the successful merge

In [4]:
```
main_folder = 'C://Users//jckat//Desktop//metro'

# Initialize an empty list to hold DataFrames
data_frames = []

# Walk through the directory
for root, dirs, files in os.walk(main_folder):
    for file in files:
        if file.endswith('.csv'):
            # Create the full file path
            file_path = os.path.join(root, file)
            # Read the CSV file and append the DataFrame to the list
            df = pd.read_csv(file_path)
            data_frames.append(df)

# Concatenate all DataFrames
merged_df = pd.concat(data_frames, ignore_index=True)

# Save the merged DataFrame to a new CSV file
merged_df.to_csv('metro_police_crime.csv', index=False)

print("CSV files have been merged into 'metro_police_crime.csv'")
```

```
CSV files have been merged into 'metro_police_crime.csv'
```

Reading the Metro Crime Data

In [5]:
```
# Specify the path to your CSV file
csv_file_path = 'C://Users//jckat//Desktop//Space data//metro_police_crime.csv'

# Read the CSV file using pandas
data = pd.read_csv(csv_file_path)

# Display the first few rows of the DataFrame
data.head()
```

Out[5]:

	Crime ID	Month	Reported by	Falls within	Longitude
0	bcf33862673ea5cebb2d0814770e3147981dd6c6f88ad6...	2021-04	Metropolitan Police Service	Metropolitan Police Service	0.867037
1	578df7143c18214677518d49cf3834f34016eefaa134d1...	2021-04	Metropolitan Police Service	Metropolitan Police Service	-0.590478
2	NaN	2021-04	Metropolitan Police Service	Metropolitan Police Service	0.135866
3	NaN	2021-04	Metropolitan Police Service	Metropolitan Police Service	0.140192
4	NaN	2021-04	Metropolitan Police Service	Metropolitan Police Service	0.134947

The main purpose of this code is to load data from a CSV file into a pandas DataFrame and display the initial rows for inspection. This is a common first step in data analysis to verify that the data has been read correctly and to get a quick look at its structure.

In [6]:
```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3149321 entries, 0 to 3149320
Data columns (total 12 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   Crime ID               object 
 1   Month                  object 
 2   Reported by            object 
 3   Falls within           object 
 4   Longitude              float64
 5   Latitude               float64
 6   Location               object 
 7   LSOA code              object 
 8   LSOA name              object 
 9   Crime type             object 
 10  Last outcome category  object 
 11  Context                float64
dtypes: float64(3), object(9)
memory usage: 288.3+ MB
None
```

In [9]:
```
data.head()
```

Out[9]:

	Crime ID	Month	Reported by	Falls within	Longitude
0	bcf33862673ea5cebb2d0814770e3147981dd6c6f88ad6...	2021-04-01	Metropolitan Police Service	Metropolitan Police Service	0.867037
1	578df7143c18214677518d49cf3834f34016eefaa134d1...	2021-04-01	Metropolitan Police Service	Metropolitan Police Service	-0.590478
2	NaN	2021-04-01	Metropolitan Police Service	Metropolitan Police Service	0.135866
3	NaN	2021-04-01	Metropolitan Police Service	Metropolitan Police Service	0.140192
4	NaN	2021-04-01	Metropolitan Police Service	Metropolitan Police Service	0.134947

Convert the 'month' column to datetime format

In [8]:
```
data['Month'] = pd.to_datetime(data['Month'], format='%Y-%m')

# Extract year and month for exploration
data['year'] = data['Month'].dt.year
data['Month_num'] = data['Month'].dt.month
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3149321 entries, 0 to 3149320
Data columns (total 14 columns):
 #   Column                 Dtype
---  ------                 -----
 0   Crime ID               object
 1   Month                  datetime64[ns]
 2   Reported by            object
 3   Falls within           object
 4   Longitude              float64
 5   Latitude               float64
 6   Location               object
 7   LSOA code              object
 8   LSOA name              object
 9   Crime type             object
 10  Last outcome category  object
 11  Context                float64
 12  year                   int32
 13  Month_num              int32
dtypes: datetime64[ns](1), float64(3), int32(2), object(8)
memory usage: 312.4+ MB
```

Checking for Missing Values and data types

In [13]:
```
data.isnull().sum()
```

Out[13]:
```
Crime ID                 702265
Month                         0
Reported by                   0
Falls within                  0
Longitude                 64222
Latitude                  64222
Location                      0
LSOA code                 64222
LSOA name                 64222
Crime type                    0
Last outcome category     702265
Context                  3149321
year                          0
Month_num                     0
dtype: int64
```

In [14]:
```
data.dtypes
```

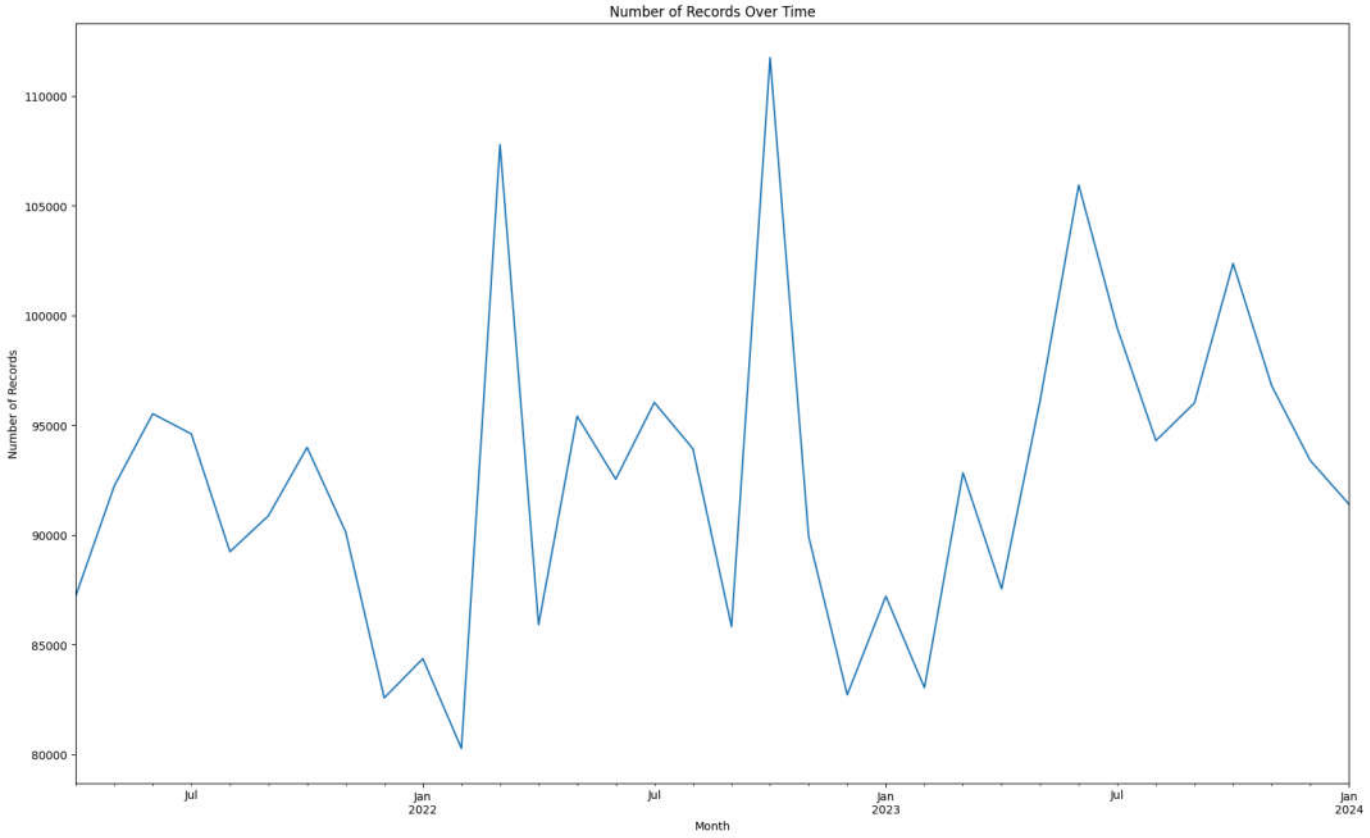
Out[14]:
```
Crime ID                         object
Month                    datetime64[ns]
Reported by                      object
Falls within                     object
Longitude                       float64
Latitude                        float64
Location                         object
LSOA code                        object
LSOA name                        object
Crime type                       object
Last outcome category            object
Context                         float64
year                              int32
Month_num                         int32
dtype: object
```

Basic exploratory data analysis

Plotting the number of records over time

This code snippet generates a plot showing the number of records per month in your dataset and saves this plot as a PNG file in a specified directory. This can be useful for visualizing trends in the data over time and making the plot accessible for future use or sharing.

In [18]:
```
# Create the plot
plt.figure(figsize=(20, 12))
data['Month'].value_counts().sort_index().plot()
plt.title('Number of Records Over Time')
plt.xlabel('Month')
plt.ylabel('Number of Records')

# Define the path to save the plot
save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\Pl

# Save the plot to the specified path
plt.savefig(save_path, bbox_inches='tight')
plt.show()
```



Plotting Geographical Distribution of 'Latitude' and 'Longitude' of Crime Point the Metro Police

In [17]:
```
if 'Latitude' in data.columns and 'Longitude' in data.columns:
    plt.figure(figsize=(20, 10))  # Increase the figure size for a larger plot
    sns.scatterplot(data=data, x='Longitude', y='Latitude', hue='Month', palette='viridis', s=1
    plt.title('Geographical Distribution of Crime Occurrences Recorded by Metro-Police April 20
    plt.xlabel('Longitude')
    plt.ylabel('Latitude')
    plt.legend(title='Month', bbox_to_anchor=(1.05, 1), loc='upper left')

    # Define the path to save the plot
    save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Poli

    # Save the plot to the specified path
    plt.savefig(save_path, bbox_inches='tight')
    plt.show()
```

Space-Time Analysis: Geographical distribution over time

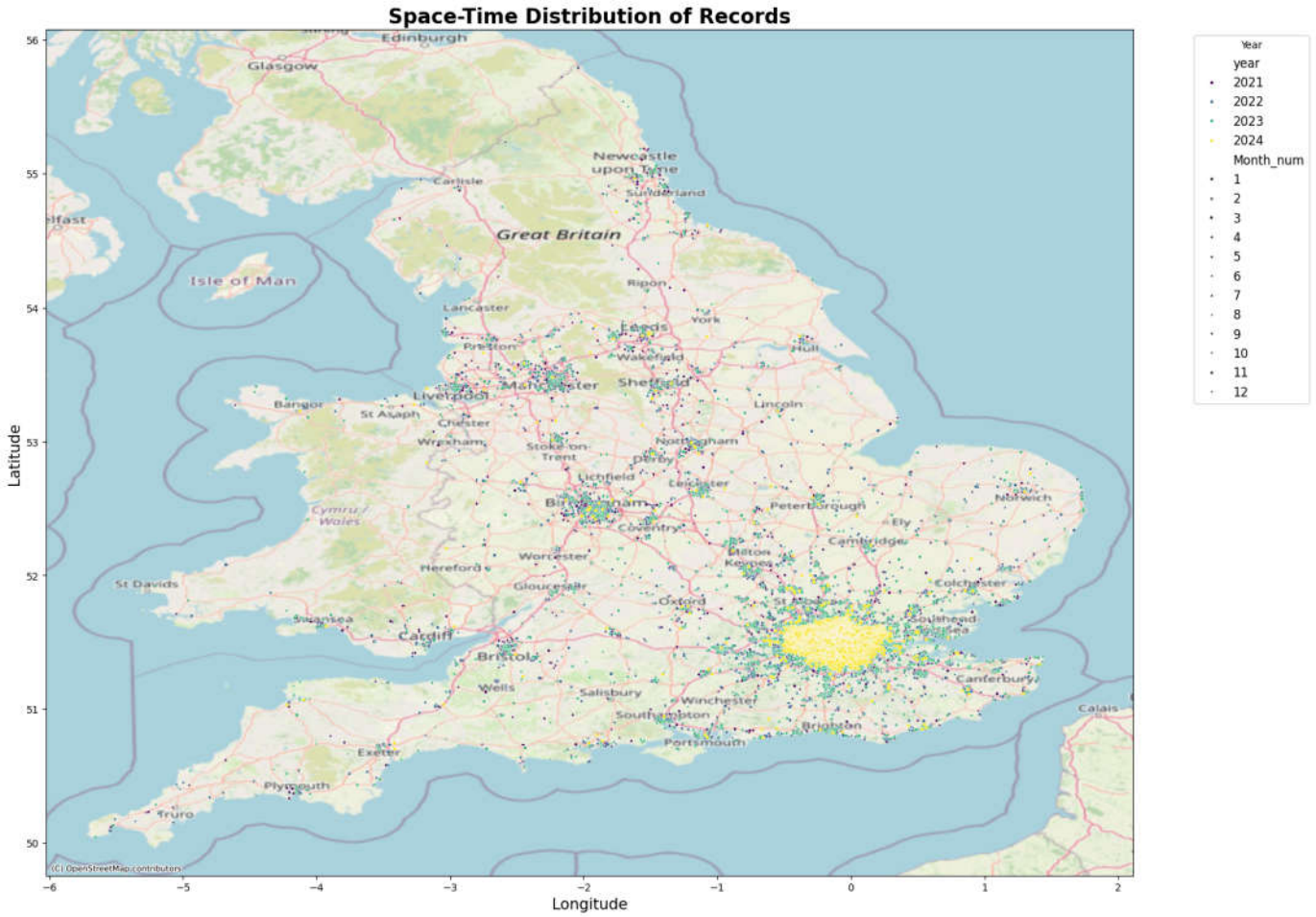
In [19]:
```
# Create the plot
plt.figure(figsize=(20, 10))  # Increase the figure size for a larger plot
sns.scatterplot(data=data, x='Longitude', y='Latitude', hue='year', palette='viridis', s=10, st
plt.title('Space-Time Distribution of Records')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.legend(title='Year', bbox_to_anchor=(1.05, 1), loc='upper left')

# Define the path to save the plot
save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\P

# Save the plot to the specified path
plt.savefig(save_path, bbox_inches='tight')
plt.show()
```

Visualizing Spatiotemporal Distribution of Crime Records with a Base Map

This code creates a scatter plot to visualize the spatiotemporal distribution of crime records, coloring points by year and styling them by month number. It overlays the points on an OpenStreetMap base

map, adds titles and labels, and saves the plot as a PNG file to a specified directory. This visualization helps in understanding how crimes are distributed across different locations and times, making it a valuable tool for analysis.

In [35]:
```
import contextily as ctx

# Create the plot
fig, ax = plt.subplots(figsize=(25, 15))  # Increase the figure size for a larger plot
sns.scatterplot(data=data, x='Longitude', y='Latitude', hue='year', palette='viridis', s=10, st
plt.title('Space-Time Distribution of Records', fontsize=20, fontweight='bold')
plt.xlabel('Longitude', fontsize=15)
plt.ylabel('Latitude', fontsize=15)
plt.legend(title='Year', bbox_to_anchor=(1.05, 1), loc='upper left', fontsize=12)

# Add base map
ctx.add_basemap(ax, crs='EPSG:4326', source=ctx.providers.OpenStreetMap.Mapnik)

# Define the path to save the plot
save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\P

# Save the plot to the specified path
plt.savefig(save_path, bbox_inches='tight')
plt.show()
```



Plotting the Temporal Trend of Crime Records Over Time

This code creates a line plot to visualize the temporal trend of crime records over time, using the Month column to count the number of records per month. The plot is then saved as a PNG file to a specified directory. This visualization helps in understanding the trend and frequency of crime records over different months, providing insights into temporal patterns in the dataset.

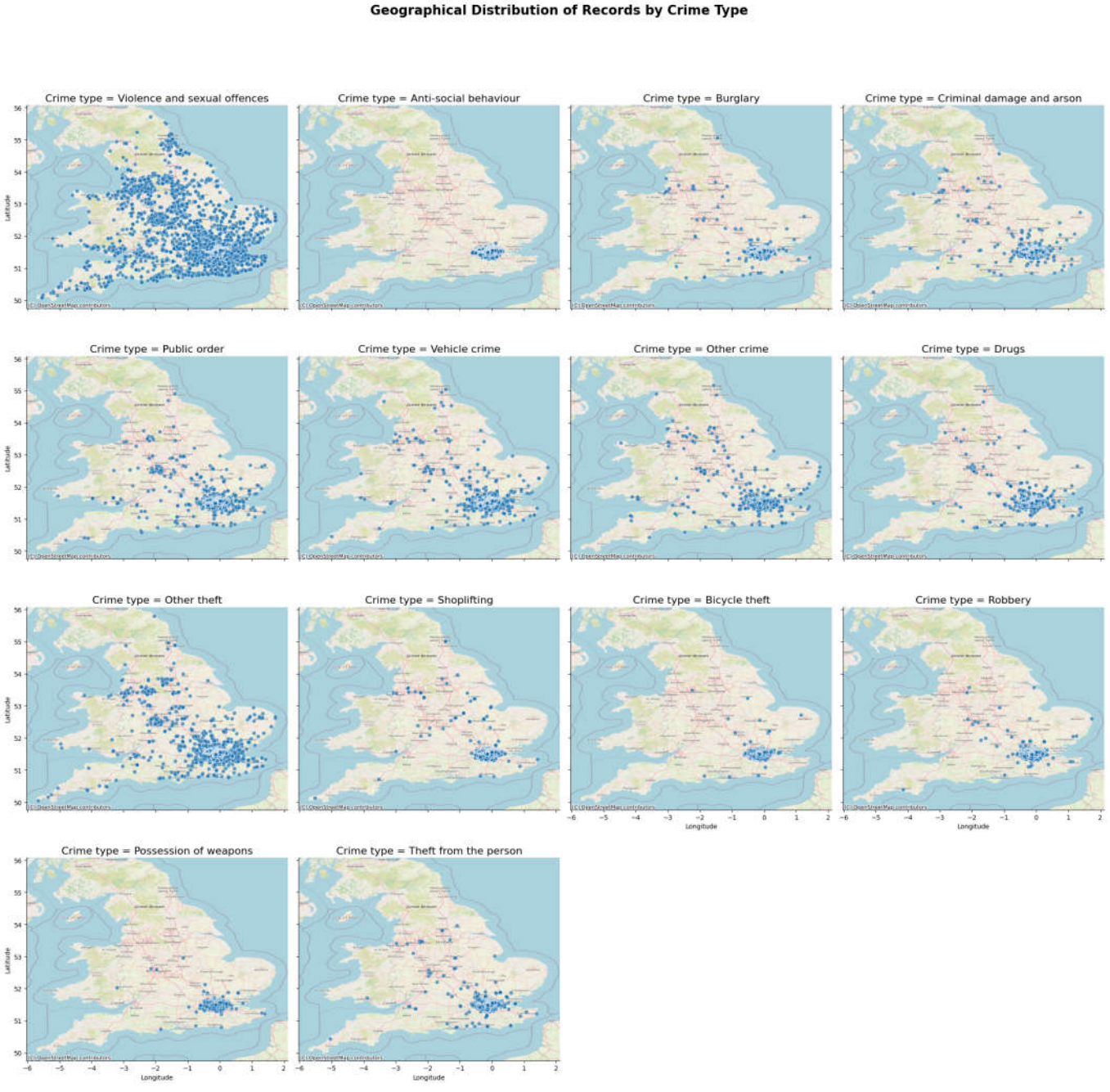
In [32]:
```
import seaborn as sns
import matplotlib.pyplot as plt

if 'Latitude' in data.columns and 'Longitude' in data.columns and 'Crime type' in data.columns:
    unique_crime_types = data['Crime type'].unique()
    palette = sns.color_palette("hsv", len(unique_crime_types))  # Generate a unique color for 

    g = sns.FacetGrid(data, col="Crime type", col_wrap=4, height=6, aspect=1, palette=palette)
    g.map_dataframe(sns.scatterplot, "Longitude", "Latitude", alpha=.7)

    for ax in g.axes.flat:
        ax.set_title(ax.get_title(), fontsize=16)  # Increase font size for each subplot title

    g.add_legend()
    plt.subplots_adjust(top=0.9)
    g.fig.suptitle('Geographical Distribution of Records by Crime Type', fontsize=20, fontweigh
    plt.show()

    # Define the path to save the plot
    save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Poli

    # Save the plot to the specified path
    g.savefig(save_path, bbox_inches='tight')
    plt.show()
```

Visualizing Geographical Distribution of Crime Records by Crime Type with FacetGrid

This code creates multiple scatter plots to visualize the geographical distribution of crime records, separated by crime type. Each plot is overlaid with an OpenStreetMap base map. The resulting visualization is saved as a PNG file to a specified directory, providing a comprehensive view of how different types of crimes are geographically distributed.

In [33]:
```
import contextily as ctx
import matplotlib.pyplot as plt
import seaborn as sns
import contextily as ctx
import os

# Define a unique color palette for the crime types
unique_crime_types = data['Crime type'].unique()
palette = sns.color_palette("husl", len(unique_crime_types))

# Create multiple plots based on the 'Crime type' column
if 'Latitude' in data.columns and 'Longitude' in data.columns and 'Crime type' in data.columns:
```

```
    g = sns.FacetGrid(data, col="Crime type", col_wrap=4, height=6, aspect=1, palette=palette)
    g.map_dataframe(sns.scatterplot, "Longitude", "Latitude", alpha=.7)
    g.add_legend()

    plt.subplots_adjust(top=0.9)
    g.fig.suptitle('Geographical Distribution of Records by Crime Type', fontsize=20, fontweigh

    # Increase the font size of each crime type title
    for ax in g.axes.flat:
        ax.set_title(ax.get_title(), fontsize=16)

    for ax in g.axes.flat:
        ctx.add_basemap(ax, crs='EPSG:4326', source=ctx.providers.OpenStreetMap.Mapnik)

    # Define the path to save the plot
    save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Poli

    # Save the plot to the specified path
    g.savefig(save_path, bbox_inches='tight')
    plt.show()
```



Cleaning and Saving Crime Data for Geographical Analysis

This code checks for the necessary columns in the DataFrame, removes rows with missing values in the specified columns, and saves the cleaned DataFrame to a CSV file in a specified directory. This process ensures that the data is ready for accurate geographical analysis and visualization.

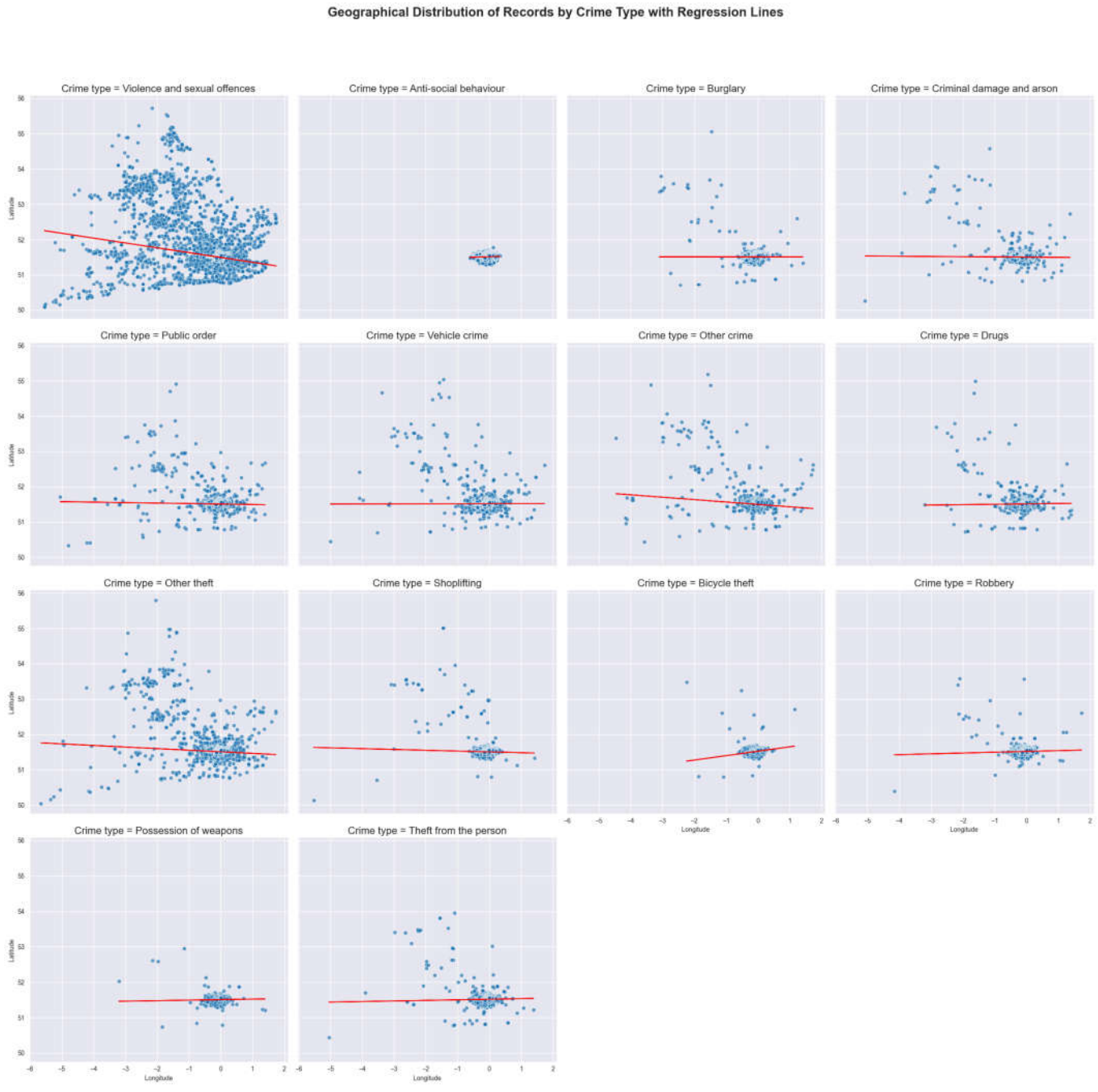
In [37]:
```
# Check if the necessary columns are in the DataFrame
required_columns = ['Latitude', 'Longitude', 'Crime type']
if all(column in data.columns for column in required_columns):
    # Remove rows with missing values in 'Latitude', 'Longitude', or 'Crime type'
    data_clean = data.dropna(subset=required_columns)

# Save the cleaned DataFrame to a file
save_path = "C:\\Users\\jckat\\OneDrive\\Documents\\GitHub\\Space_TIme_Exploration_of_Metro_Pol
data_clean.to_csv(f"{save_path}cleaned_data.csv", index=False)
```

Visualizing Geographical Distribution of Crime Records with Regression Lines by Crime Type

This code visualizes the geographical distribution of crime records by crime type, including regression lines to show trends within each type. Each scatter plot is overlaid with a linear regression line and displayed in a FacetGrid, separated by crime type. The plot is saved as a PNG file to a specified directory, providing insights into spatial trends for different crime types.

In [77]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import statsmodels.api as sm

# Assuming 'data' is your DataFrame

# Check if the necessary columns are in the DataFrame
if 'Latitude' in data.columns and 'Longitude' in data.columns and 'Crime type' in data.columns:
    # Remove rows with missing values in 'Latitude', 'Longitude', or 'Crime type'
    data_clean = data.dropna(subset=['Latitude', 'Longitude', 'Crime type'])

    unique_crime_types = data_clean['Crime type'].unique()
    palette = sns.color_palette("hsv", len(unique_crime_types))  # Generate a unique color for

    # Create a FacetGrid
    g = sns.FacetGrid(data_clean, col="Crime type", col_wrap=4, height=6, aspect=1, palette=pal

    # Function to plot scatter plot and regression line
    def scatter_with_regression(data, color, **kws):
        sns.scatterplot(x='Longitude', y='Latitude', data=data, color=color, alpha=0.7)
        # Perform linear regression
        X = sm.add_constant(data['Longitude'])
        y = data['Latitude']
        model = sm.OLS(y, X).fit()
        # Plot regression line in red
        plt.plot(data['Longitude'], model.predict(X), color='red')

    # Map the function to the FacetGrid
    g.map_dataframe(scatter_with_regression)

    # Adjust font size for each subplot title
    for ax in g.axes.flat:
        ax.set_title(ax.get_title(), fontsize=16)

    g.add_legend()
    plt.subplots_adjust(top=0.9)
```

```
    g.fig.suptitle('Geographical Distribution of Records by Crime Type with Regression Lines',
    plt.show()

    # Define the path to save the plot
    save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Poli

    # Save the plot to the specified path
    g.savefig(save_path, bbox_inches='tight')
    plt.show()
```

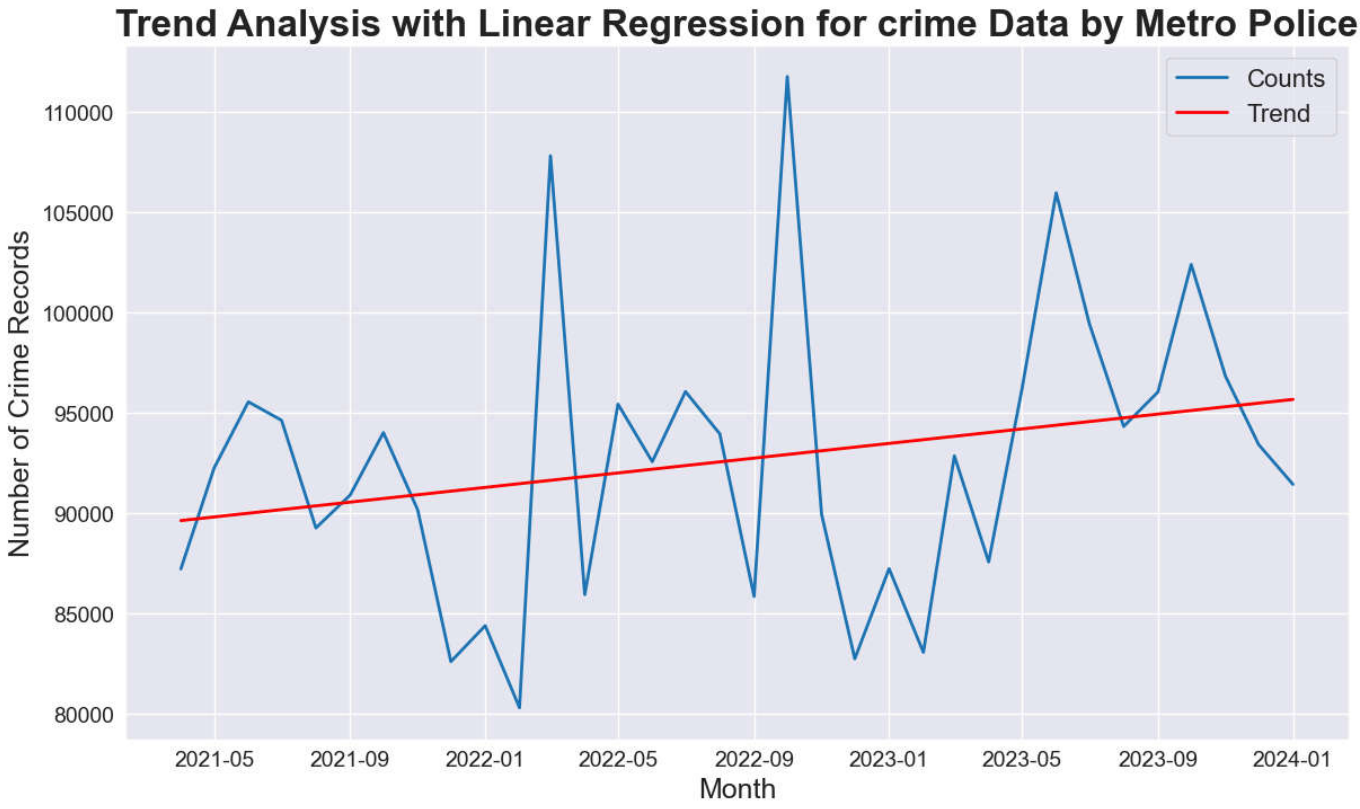


Trend Analysis with Linear Regression for Crime Data by Metro Police

This code performs a trend analysis using linear regression on the monthly counts of crime records. It plots the actual counts along with the linear regression trend line, saving the plot as a PNG file to a specified directory. Additionally, it prints the summary of the regression model for further statistical insights.

In [43]:
```
# Trend Analysis: Linear Regression
data_grouped = data.groupby('Month').size().reset_index(name='counts')
```

```python
data_grouped['timestamp'] = data_grouped['Month'].apply(lambda x: x.timestamp())

# Add a constant to the predictor
X = add_constant(data_grouped['timestamp'])
y = data_grouped['counts']

# Fit the linear regression model
model = OLS(y, X).fit()

# Add the trend predictions to the DataFrame
data_grouped['trend'] = model.predict(X)

# Plot the counts and the trend
plt.figure(figsize=(14, 8))  # Increase the plot size
plt.plot(data_grouped['Month'], data_grouped['counts'], label='Counts', linewidth=2)
plt.plot(data_grouped['Month'], data_grouped['trend'], label='Trend', color='red', linewidth=2)
plt.title('Trend Analysis with Linear Regression for crime Data by Metro Police', fontsize=24,
plt.xlabel('Month', fontsize=18)  # Increase the x-axis label font size
plt.ylabel('Number of Crime Records', fontsize=18)  # Increase the y-axis label font size
plt.xticks(fontsize=14)  # Increase the x-axis ticks font size
plt.yticks(fontsize=14)  # Increase the y-axis ticks font size
plt.legend(fontsize=16)  # Increase the legend font size

# Define the path to save the plot
save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\P

# Save the plot to the specified path
plt.savefig(save_path, bbox_inches='tight')

# Show the plot
plt.show()

# Print the summary of the regression model
print(model.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 counts   R-squared:                       0.064
Model:                            OLS   Adj. R-squared:                  0.035
Method:                 Least Squares   F-statistic:                     2.187
Date:                Thu, 16 May 2024   Prob (F-statistic):              0.149
Time:                        14:47:45   Log-Likelihood:                -348.60
No. Observations:                  34   AIC:                             701.2
Df Residuals:                      32   BIC:                             704.3
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -2.29e+04   7.81e+04     -0.293      0.771   -1.82e+05    1.36e+05
timestamp  6.957e-05    4.7e-05      1.479      0.149   -2.63e-05       0.000
==============================================================================
Omnibus:                        4.640   Durbin-Watson:                   2.046
Prob(Omnibus):                  0.098   Jarque-Bera (JB):                3.187
Skew:                           0.684   Prob(JB):                        0.203
Kurtosis:                       3.617   Cond. No.                     1.07e+11
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are
strong multicollinearity or other numerical problems.
```

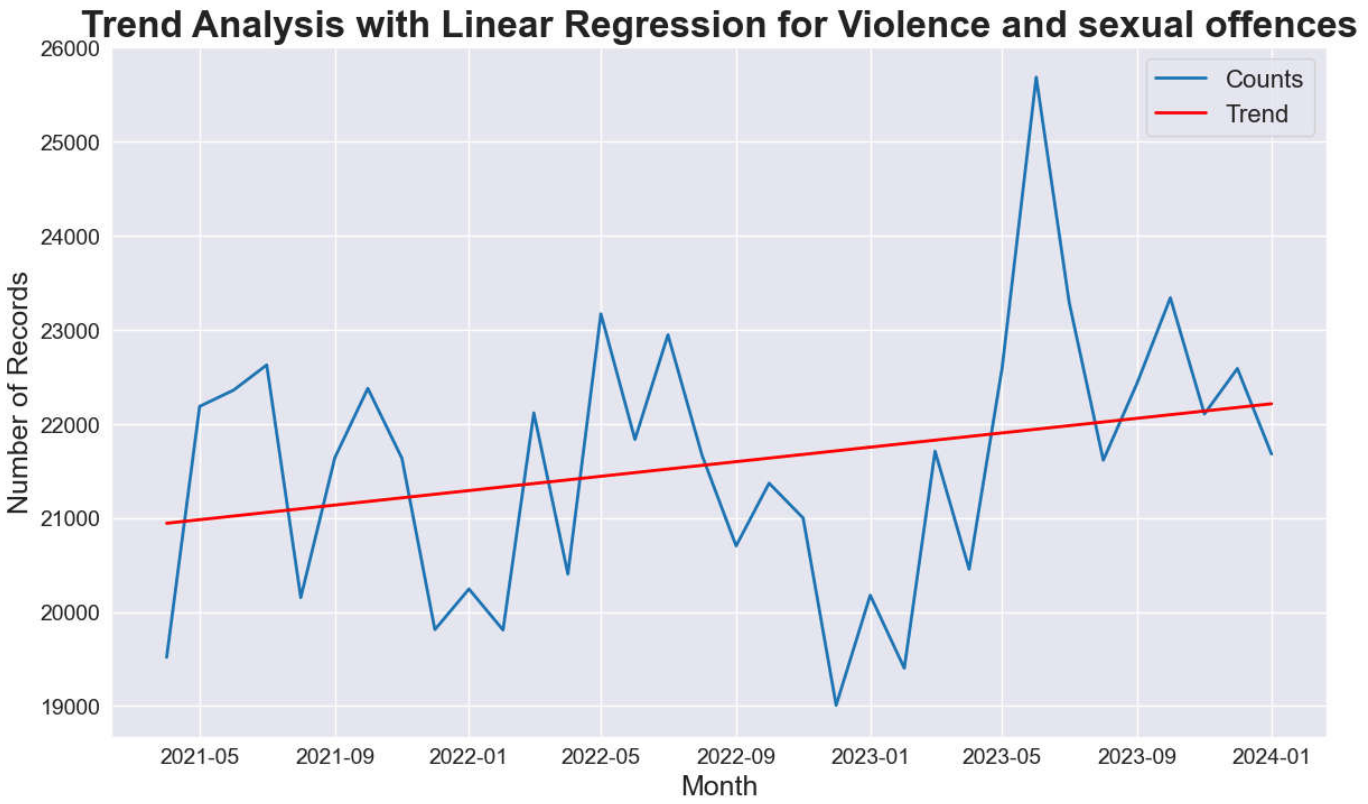
Trend Analysis with Linear Regression by Crime Type

This code performs a trend analysis using linear regression on the monthly counts of crime records for each crime type. It plots the actual counts along with the linear regression trend line for each crime type, saving each plot as a PNG file in a specified directory. Additionally, it prints the summary of the regression model for each crime type for further statistical insights.

In [46]:

```python
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.api import OLS, add_constant

# Assuming 'data' is your DataFrame

# Ensure 'Month' is in datetime format
data['Month'] = pd.to_datetime(data['Month'])

# Function to perform trend analysis and plot for each crime type
def Linear_trend_analysis_by_crime_type(data, crime_type, save_path):
    # Filter data for the specific crime type
    data_crime = data[data['Crime type'] == crime_type]

    # Trend Analysis: Linear Regression
    data_grouped = data_crime.groupby('Month').size().reset_index(name='counts')
    data_grouped['timestamp'] = data_grouped['Month'].apply(lambda x: x.timestamp())

    # Add a constant to the predictor
    X = add_constant(data_grouped['timestamp'])
    y = data_grouped['counts']

    # Fit the linear regression model
    model = OLS(y, X).fit()

    # Add the trend predictions to the DataFrame
```

```
    data_grouped['trend'] = model.predict(X)

    # Plot the counts and the trend
    plt.figure(figsize=(14, 8))  # Increase the plot size
    plt.plot(data_grouped['Month'], data_grouped['counts'], label='Counts', linewidth=2)
    plt.plot(data_grouped['Month'], data_grouped['trend'], label='Trend', color='red', linewidt
    plt.title(f'Trend Analysis with Linear Regression for {crime_type}', fontsize=24, fontweigh
    plt.xlabel('Month', fontsize=18)  # Increase the x-axis label font size
    plt.ylabel('Number of Records', fontsize=18)  # Increase the y-axis label font size
    plt.xticks(fontsize=14)  # Increase the x-axis ticks font size
    plt.yticks(fontsize=14)  # Increase the y-axis ticks font size
    plt.legend(fontsize=16)  # Increase the legend font size

    # Save the plot to the specified path
    plot_save_path = f"{save_path}Trend_Analysis_{crime_type.replace(' ', '_')}.png"
    plt.savefig(plot_save_path, bbox_inches='tight')

    # Show the plot
    plt.show()

    # Print the summary of the regression model
    print(f"Summary for {crime_type}:")
    print(model.summary())
    print("\n")

# Define the base path to save the plots
save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\Pl

# Loop through each unique crime type and perform the trend analysis
for crime_type in data['Crime type'].unique():
    Linear_trend_analysis_by_crime_type(data, crime_type, save_path)
```



Summary for Violence and sexual offences:

OLS Regression Results

Dep. Variable:	counts	R-squared:	0.075
Model:	OLS	Adj. R-squared:	0.046
Method:	Least Squares	F-statistic:	2.579
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.118
Time:	15:09:01	Log-Likelihood:	-292.78
No. Observations:	34	AIC:	589.6
Df Residuals:	32	BIC:	592.6
Df Model:	1		
Covariance Type:	nonrobust		

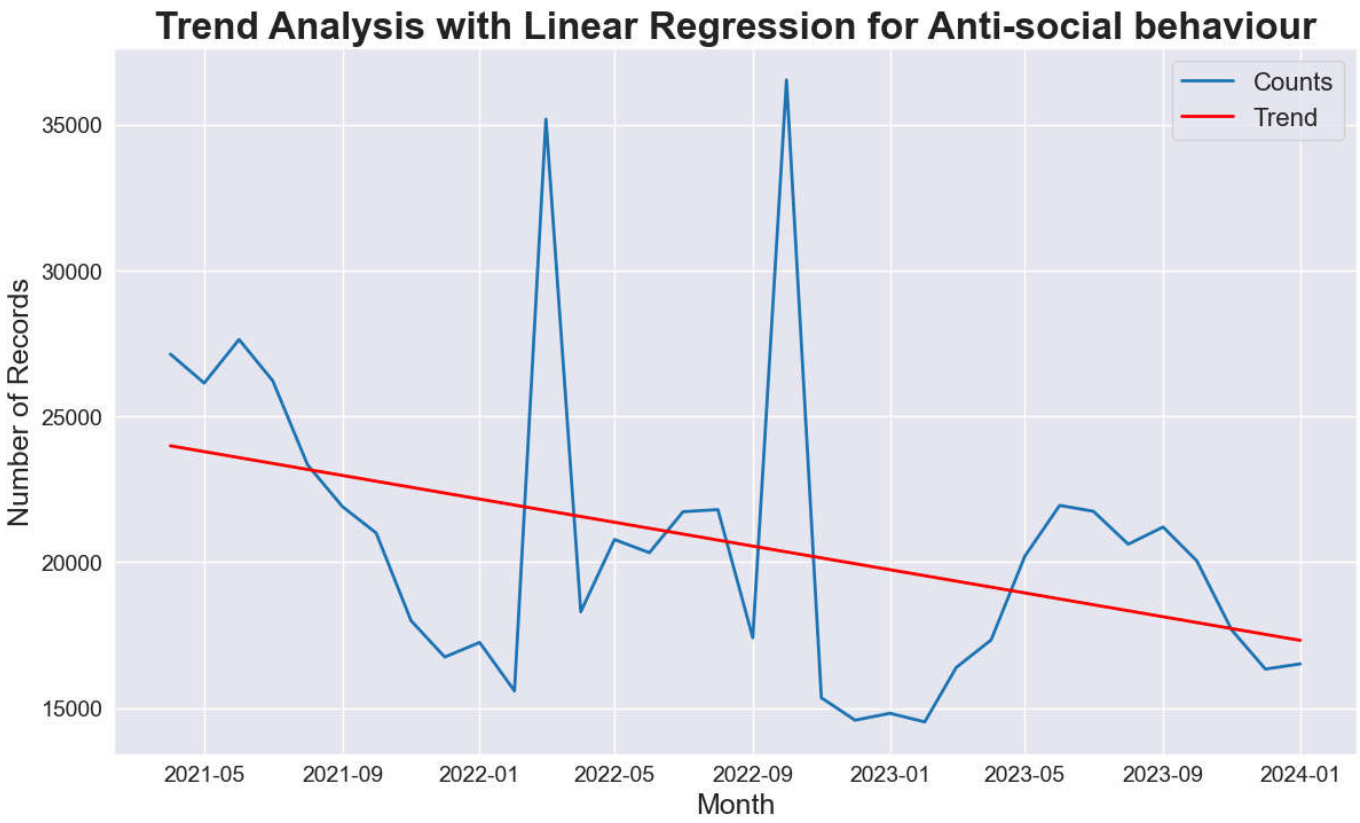
	coef	std err	t	P>|t|	[0.025	0.975]
const	-2712.2348	1.51e+04	-0.179	0.859	-3.35e+04	2.81e+04
timestamp	1.463e-05	9.11e-06	1.606	0.118	-3.93e-06	3.32e-05

Omnibus:	1.180	Durbin-Watson:	1.364
Prob(Omnibus):	0.554	Jarque-Bera (JB):	0.495
Skew:	0.272	Prob(JB):	0.781
Kurtosis:	3.230	Cond. No.	1.07e+11

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Anti-social behaviour:

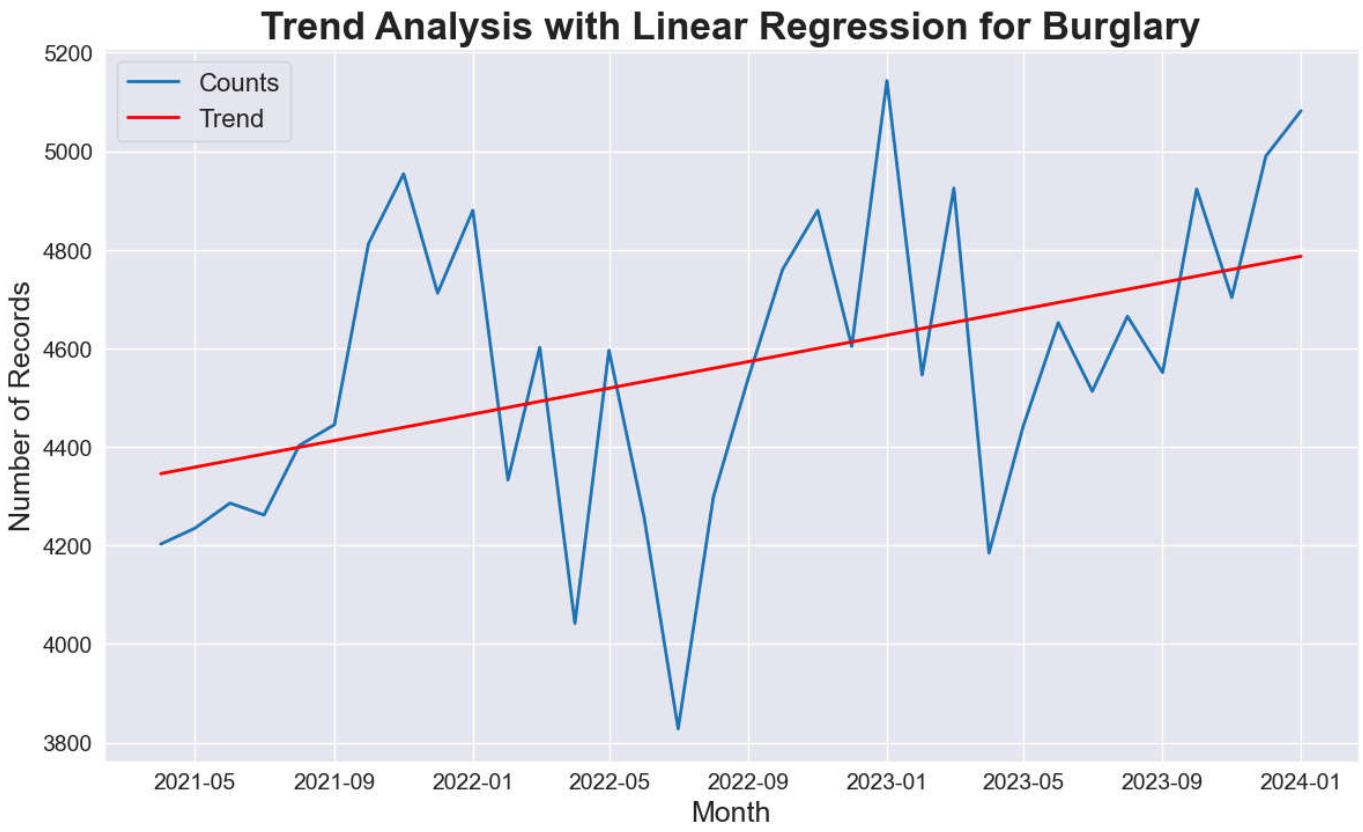
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.146
Model:	OLS	Adj. R-squared:	0.119
Method:	Least Squares	F-statistic:	5.453
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0260
Time:	15:09:03	Log-Likelihood:	-336.41
No. Observations:	34	AIC:	676.8
Df Residuals:	32	BIC:	679.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.481e+05	5.46e+04	2.713	0.011	3.69e+04	2.59e+05
timestamp	-7.677e-05	3.29e-05	-2.335	0.026	-0.000	-9.8e-06

Omnibus:	18.853	Durbin-Watson:	1.998
Prob(Omnibus):	0.000	Jarque-Bera (JB):	25.552
Skew:	1.500	Prob(JB):	2.83e-06
Kurtosis:	6.006	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Burglary:

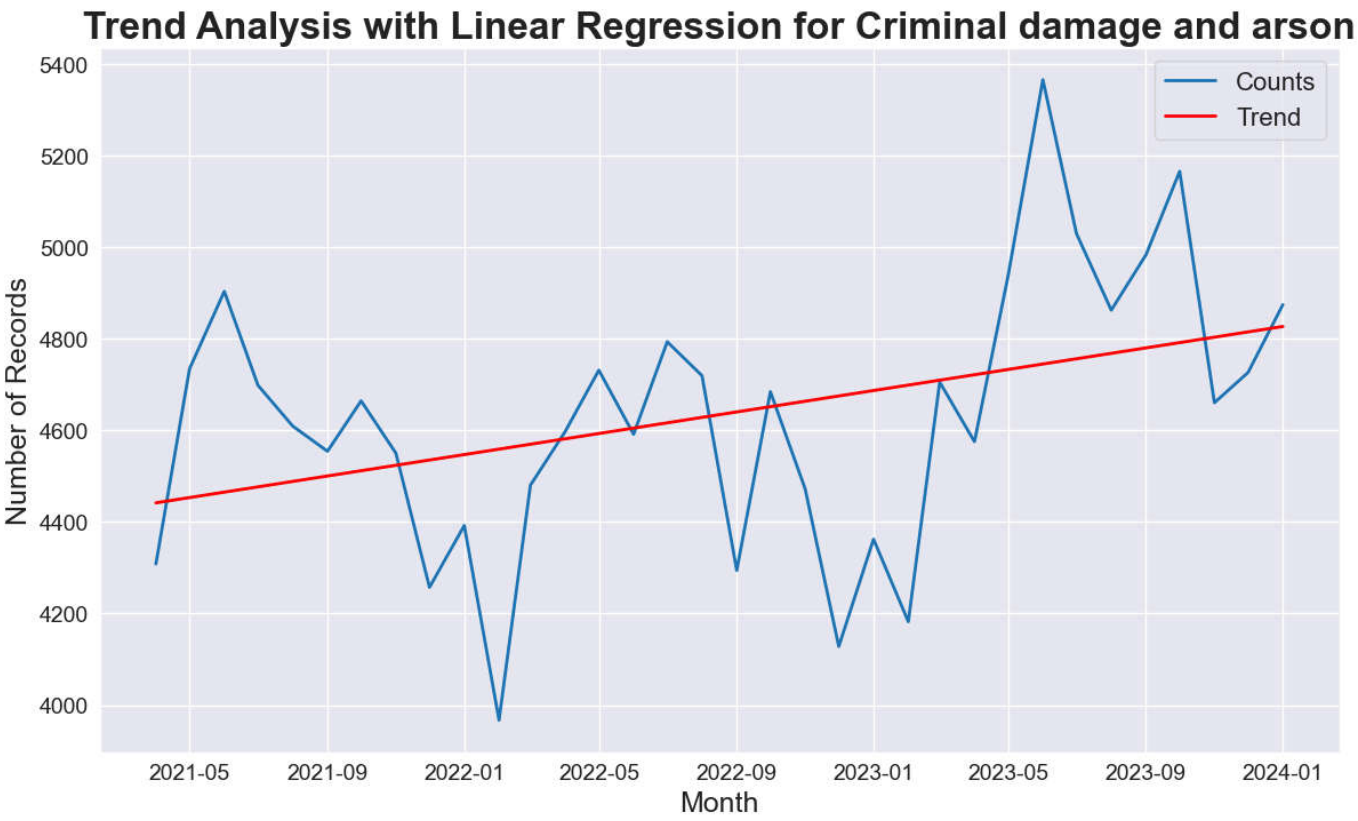
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.182
Model:	OLS	Adj. R-squared:	0.157
Method:	Least Squares	F-statistic:	7.138
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0118
Time:	15:09:04	Log-Likelihood:	-239.52
No. Observations:	34	AIC:	483.0
Df Residuals:	32	BIC:	486.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-3870.6569	3158.287	-1.226	0.229	-1.03e+04	2562.563
timestamp	5.081e-06	1.9e-06	2.672	0.012	1.21e-06	8.95e-06

Omnibus:	0.549	Durbin-Watson:	1.430
Prob(Omnibus):	0.760	Jarque-Bera (JB):	0.238
Skew:	-0.205	Prob(JB):	0.888
Kurtosis:	2.997	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are
strong multicollinearity or other numerical problems.



Summary for Criminal damage and arson:

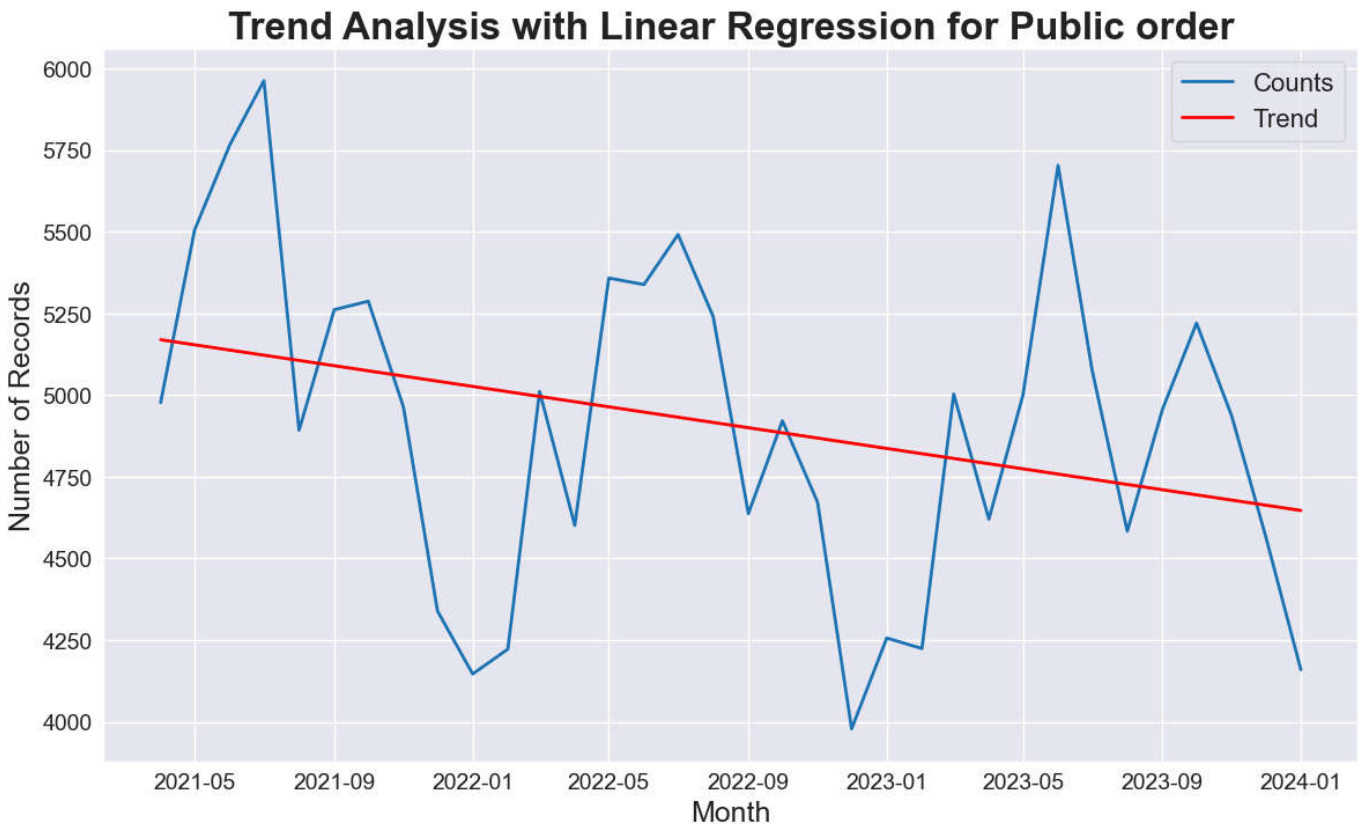
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.153
Model:	OLS	Adj. R-squared:	0.126
Method:	Least Squares	F-statistic:	5.762
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0224
Time:	15:09:05	Log-Likelihood:	-238.53
No. Observations:	34	AIC:	481.1
Df Residuals:	32	BIC:	484.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2728.9925	3067.745	-0.890	0.380	-8977.784	3519.799
timestamp	4.434e-06	1.85e-06	2.400	0.022	6.71e-07	8.2e-06

Omnibus:	0.727	Durbin-Watson:	1.047
Prob(Omnibus):	0.695	Jarque-Bera (JB):	0.347
Skew:	-0.247	Prob(JB):	0.841
Kurtosis:	3.023	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Public order:

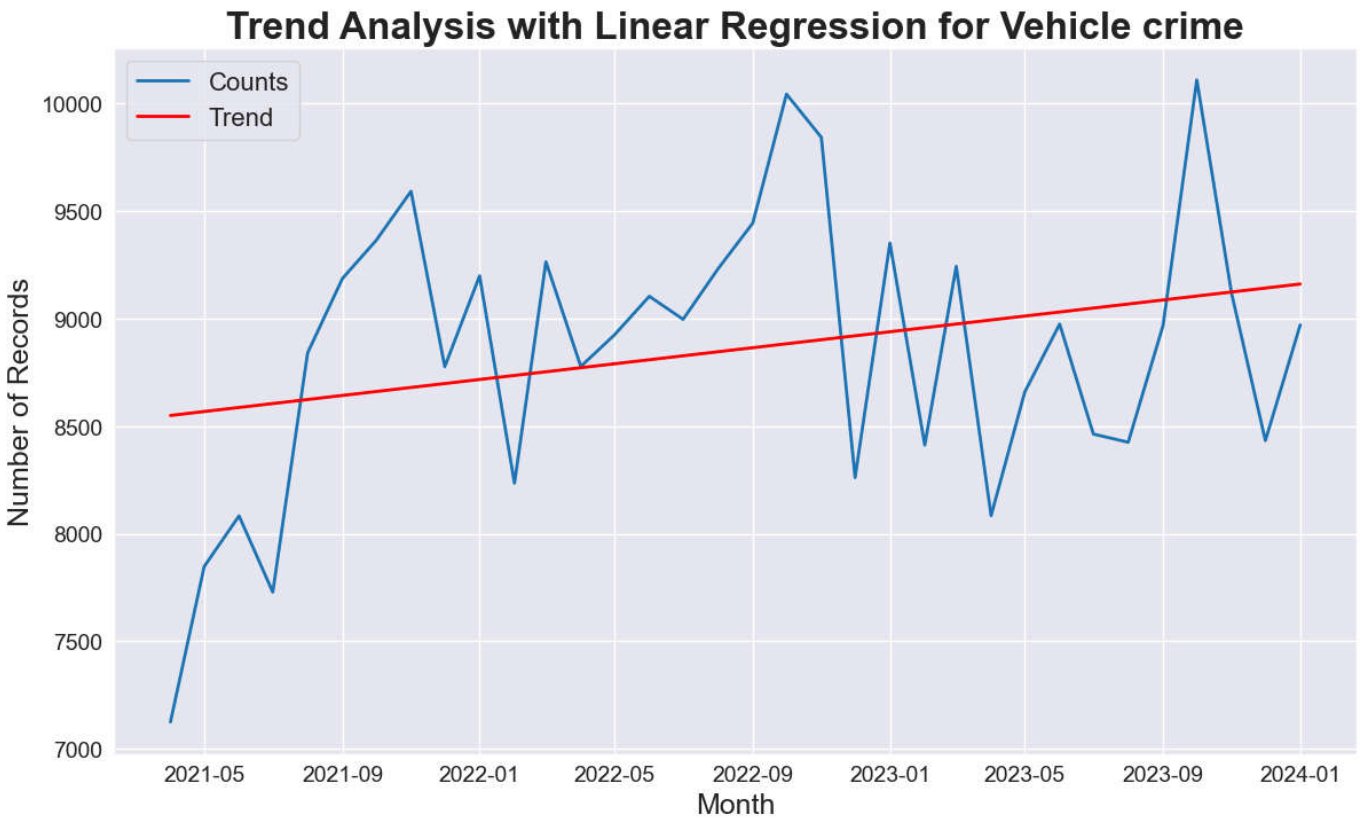
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.098
Model:	OLS	Adj. R-squared:	0.070
Method:	Least Squares	F-statistic:	3.491
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0709
Time:	15:09:06	Log-Likelihood:	-257.44
No. Observations:	34	AIC:	518.9
Df Residuals:	32	BIC:	521.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.49e+04	5349.876	2.786	0.009	4005.732	2.58e+04
timestamp	-6.019e-06	3.22e-06	-1.869	0.071	-1.26e-05	5.42e-07

Omnibus:	0.713	Durbin-Watson:	0.975
Prob(Omnibus):	0.700	Jarque-Bera (JB):	0.748
Skew:	-0.145	Prob(JB):	0.688
Kurtosis:	2.333	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Vehicle crime:

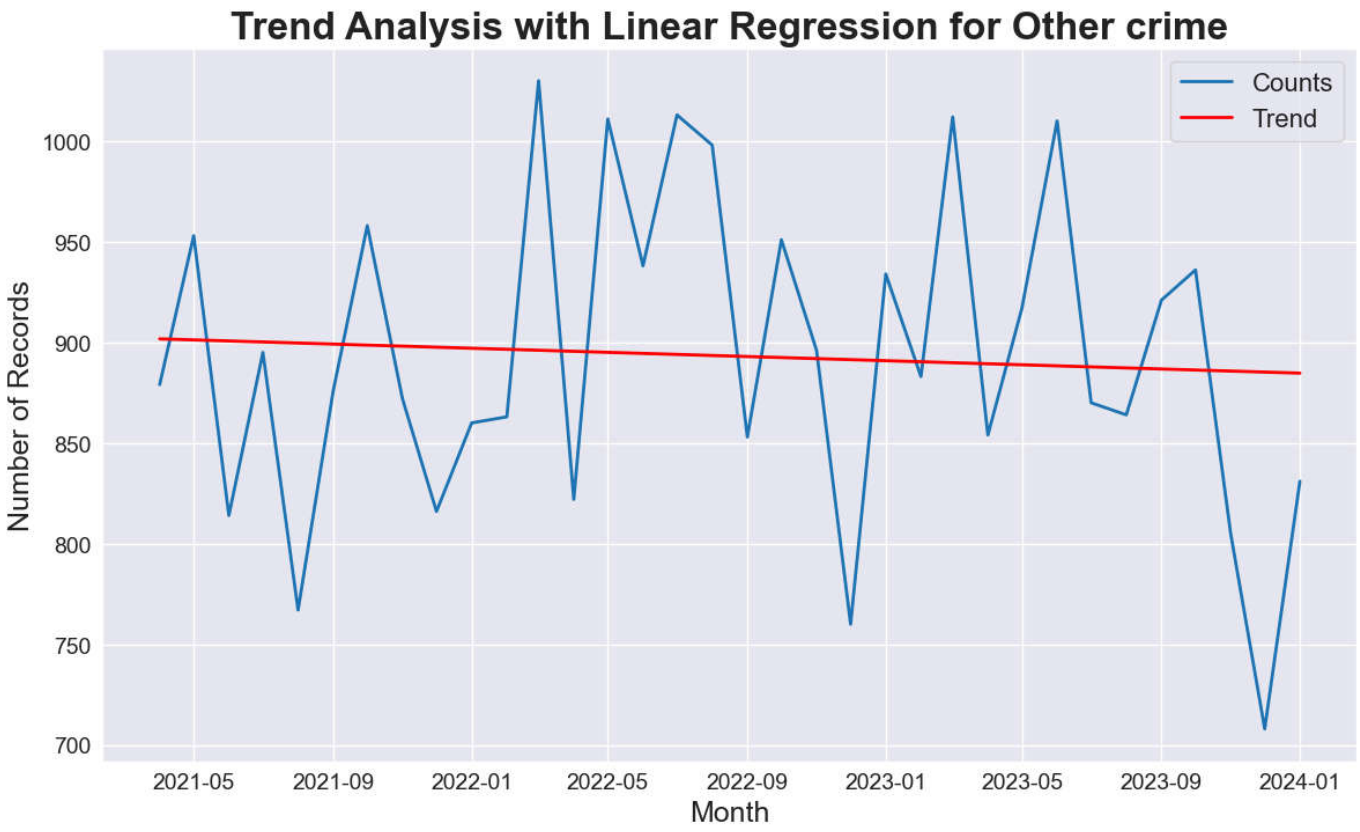
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.079
Model:	OLS	Adj. R-squared:	0.050
Method:	Least Squares	F-statistic:	2.739
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.108
Time:	15:09:07	Log-Likelihood:	-266.85
No. Observations:	34	AIC:	537.7
Df Residuals:	32	BIC:	540.7
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2821.7351	7056.158	-0.400	0.692	-1.72e+04	1.16e+04
timestamp	7.031e-06	4.25e-06	1.655	0.108	-1.62e-06	1.57e-05

Omnibus:	0.806	Durbin-Watson:	1.250
Prob(Omnibus):	0.668	Jarque-Bera (JB):	0.780
Skew:	-0.118	Prob(JB):	0.677
Kurtosis:	2.297	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Other crime:

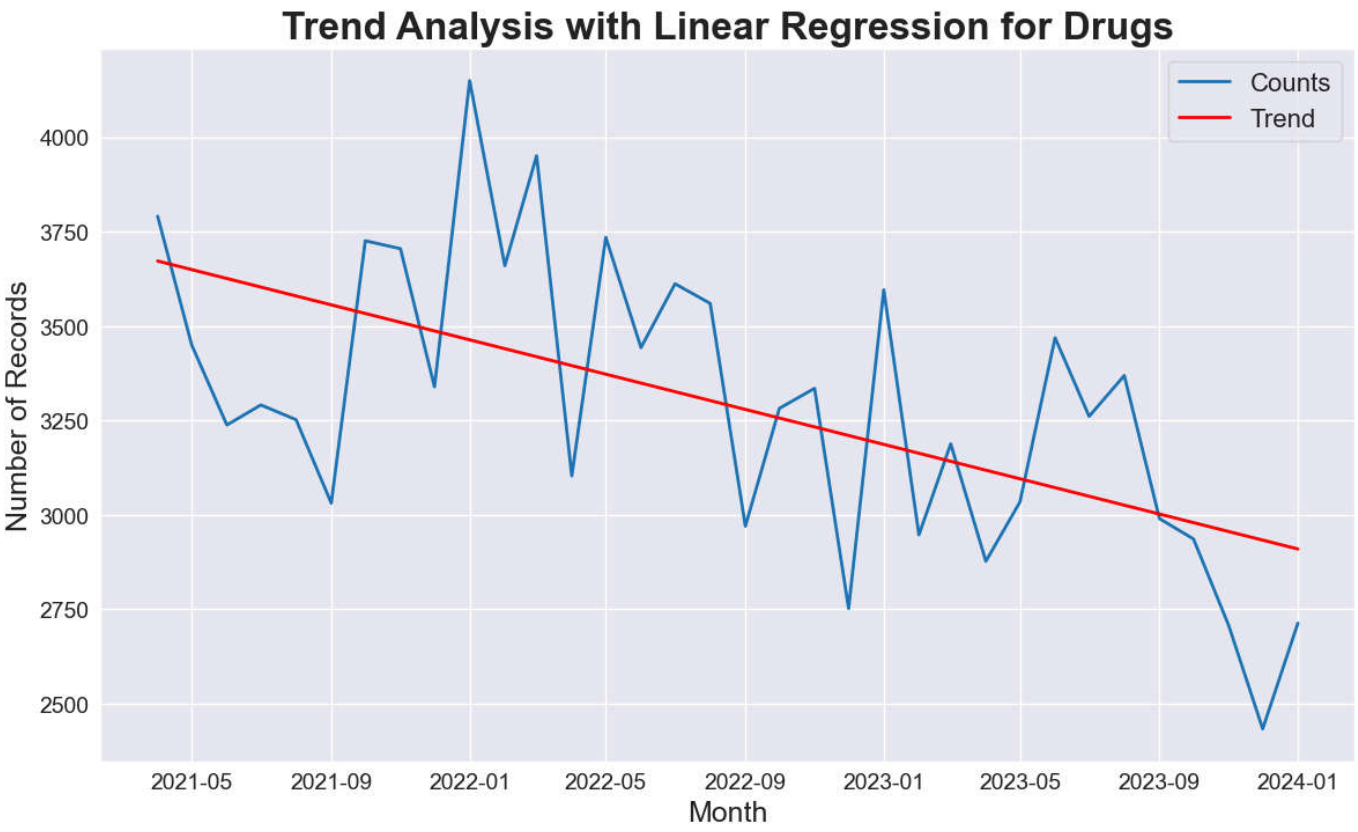
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.004
Model:	OLS	Adj. R-squared:	-0.027
Method:	Least Squares	F-statistic:	0.1343
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.716
Time:	15:09:08	Log-Likelihood:	-196.41
No. Observations:	34	AIC:	396.8
Df Residuals:	32	BIC:	399.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1218.9176	888.858	1.371	0.180	-591.628	3029.463
timestamp	-1.961e-07	5.35e-07	-0.366	0.716	-1.29e-06	8.94e-07

Omnibus:	0.288	Durbin-Watson:	1.934
Prob(Omnibus):	0.866	Jarque-Bera (JB):	0.474
Skew:	-0.087	Prob(JB):	0.789
Kurtosis:	2.448	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Drugs:

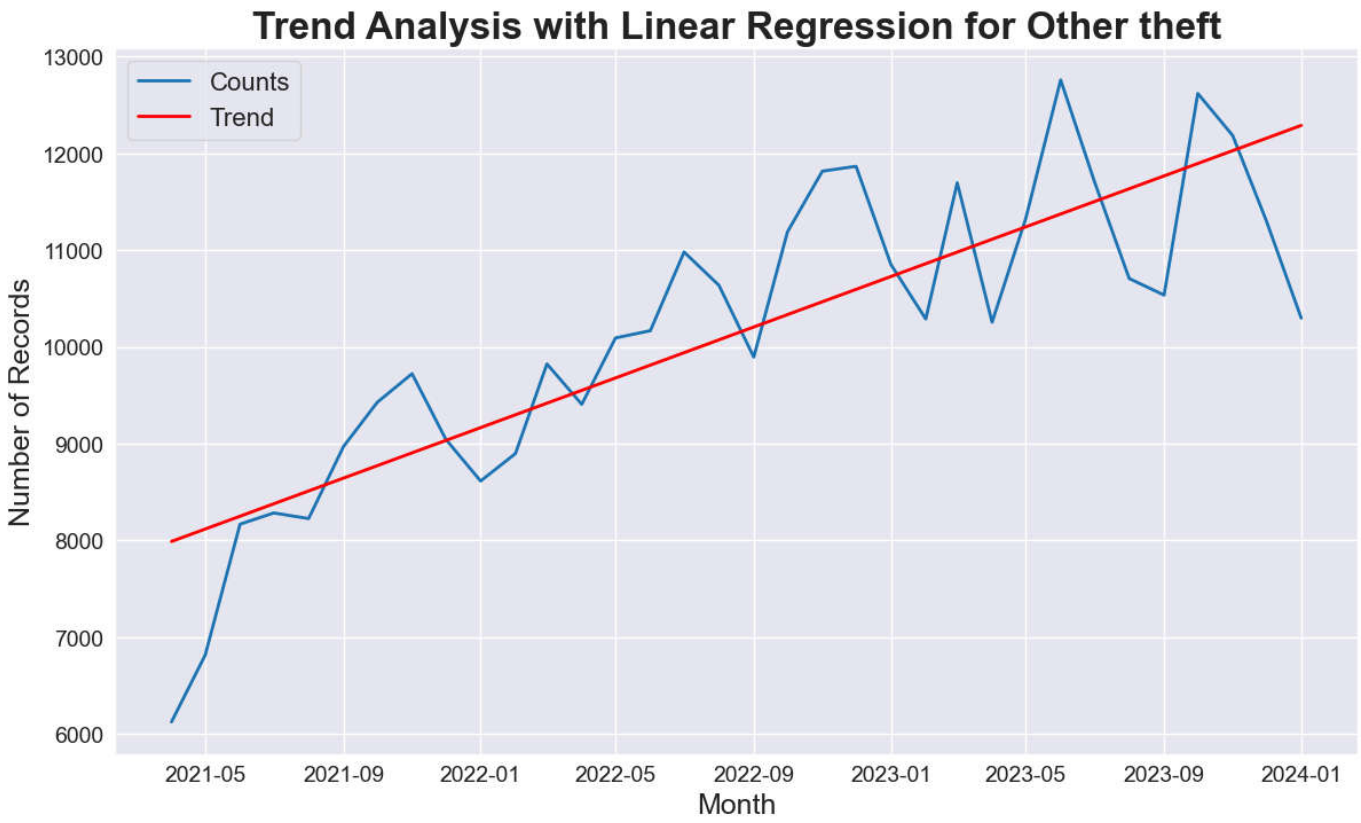
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.351
Model:	OLS	Adj. R-squared:	0.331
Method:	Least Squares	F-statistic:	17.31
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.000222
Time:	15:09:09	Log-Likelihood:	-243.07
No. Observations:	34	AIC:	490.1
Df Residuals:	32	BIC:	493.2
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.788e+04	3506.654	5.099	0.000	1.07e+04	2.5e+04
timestamp	-8.786e-06	2.11e-06	-4.161	0.000	-1.31e-05	-4.48e-06

Omnibus:	1.695	Durbin-Watson:	1.770
Prob(Omnibus):	0.428	Jarque-Bera (JB):	1.169
Skew:	0.170	Prob(JB):	0.557
Kurtosis:	2.158	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Other theft:

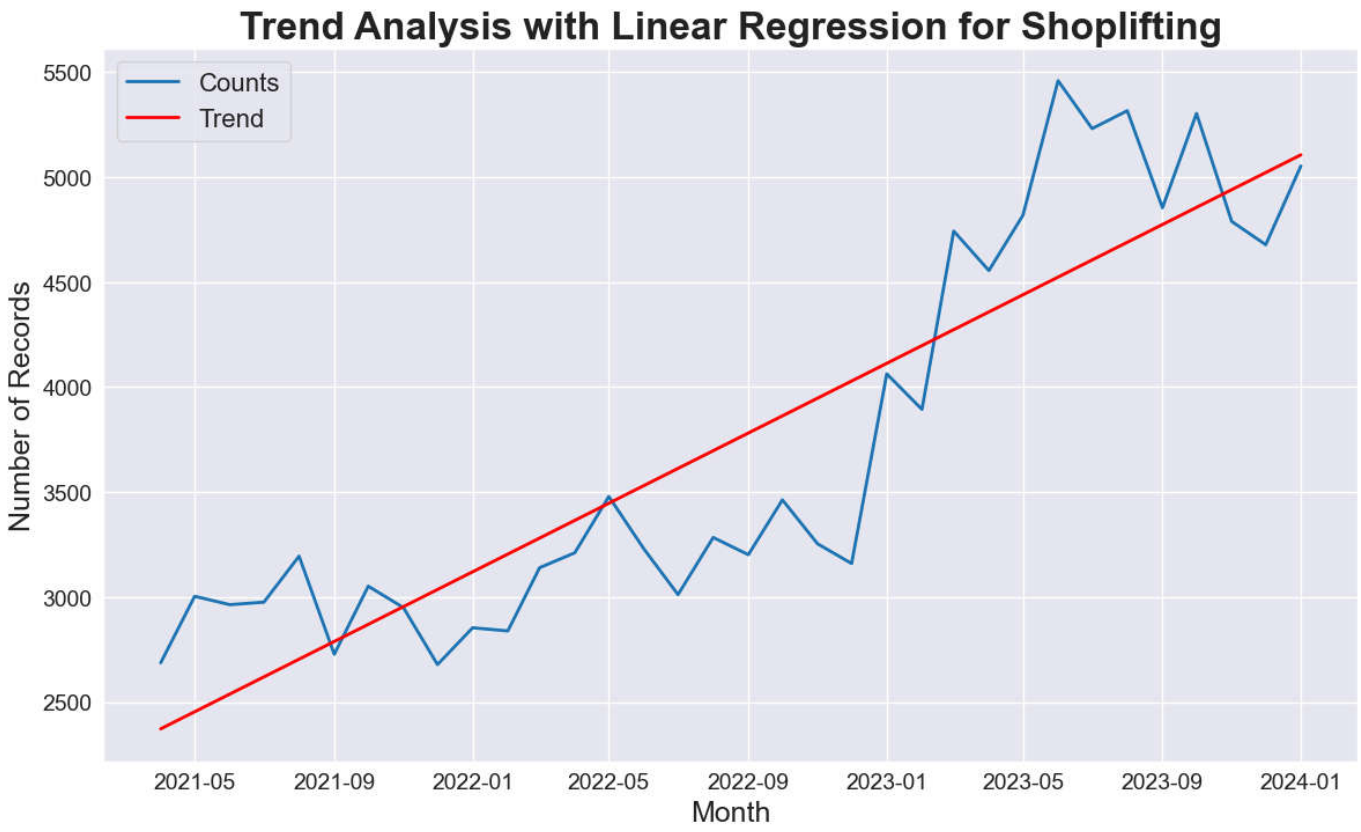
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.696
Model:	OLS	Adj. R-squared:	0.686
Method:	Least Squares	F-statistic:	73.11
Date:	Thu, 16 May 2024	Prob (F-statistic):	9.02e-10
Time:	15:09:10	Log-Likelihood:	-277.38
No. Observations:	34	AIC:	558.8
Df Residuals:	32	BIC:	561.8
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-7.211e+04	9619.681	-7.496	0.000	-9.17e+04	-5.25e+04
timestamp	4.953e-05	5.79e-06	8.551	0.000	3.77e-05	6.13e-05

Omnibus:	1.725	Durbin-Watson:	1.047
Prob(Omnibus):	0.422	Jarque-Bera (JB):	1.453
Skew:	-0.491	Prob(JB):	0.484
Kurtosis:	2.755	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Shoplifting:

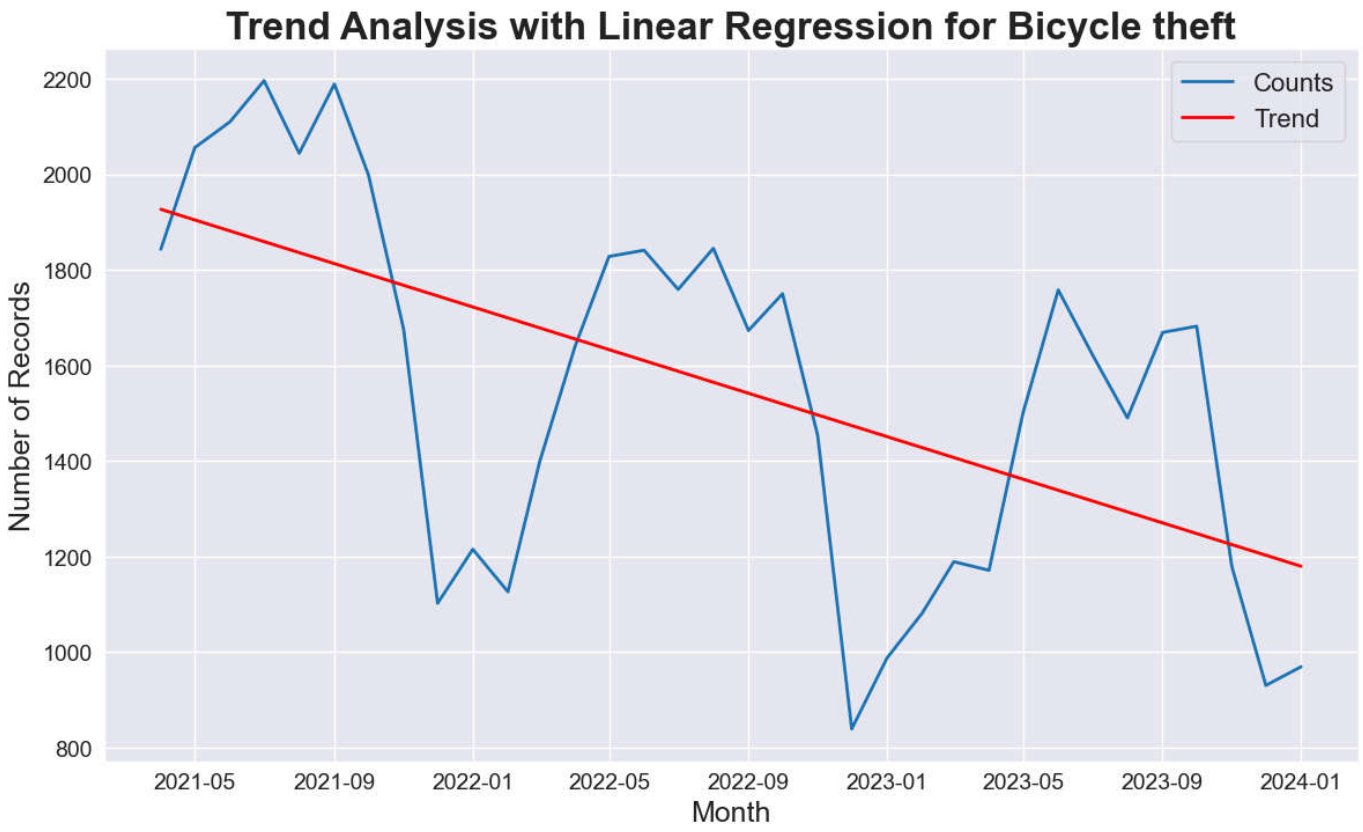
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.783
Model:	OLS	Adj. R-squared:	0.776
Method:	Least Squares	F-statistic:	115.2
Date:	Thu, 16 May 2024	Prob (F-statistic):	3.88e-12
Time:	15:09:11	Log-Likelihood:	-254.24
No. Observations:	34	AIC:	512.5
Df Residuals:	32	BIC:	515.5
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-4.853e+04	4869.807	-9.966	0.000	-5.85e+04	-3.86e+04
timestamp	3.148e-05	2.93e-06	10.735	0.000	2.55e-05	3.75e-05

Omnibus:	0.990	Durbin-Watson:	0.625
Prob(Omnibus):	0.610	Jarque-Bera (JB):	0.849
Skew:	0.095	Prob(JB):	0.654
Kurtosis:	2.249	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Bicycle theft:

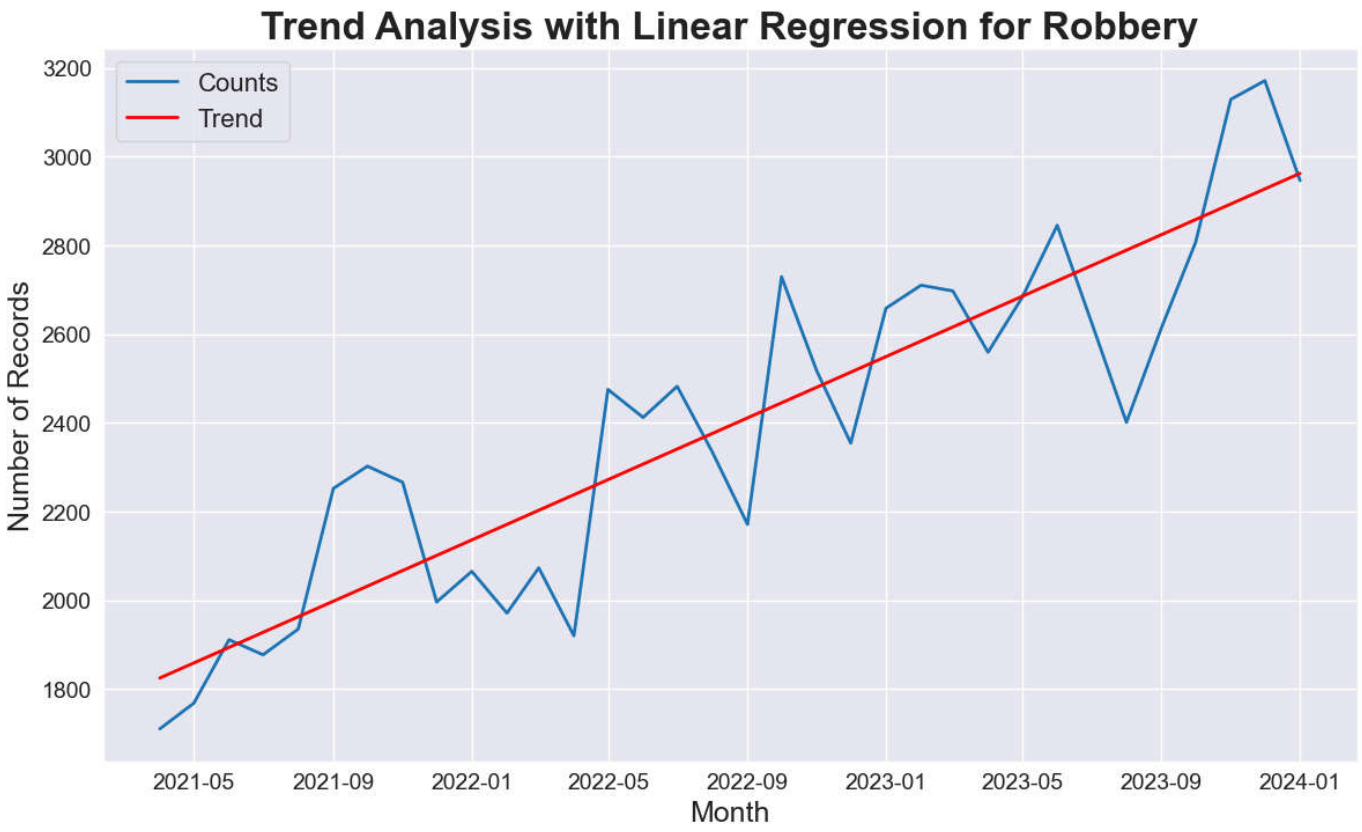
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.329
Model:	OLS	Adj. R-squared:	0.309
Method:	Least Squares	F-statistic:	15.72
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.000386
Time:	15:09:12	Log-Likelihood:	-244.01
No. Observations:	34	AIC:	492.0
Df Residuals:	32	BIC:	495.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.585e+04	3604.827	4.396	0.000	8502.788	2.32e+04
timestamp	-8.607e-06	2.17e-06	-3.965	0.000	-1.3e-05	-4.19e-06

Omnibus:	3.592	Durbin-Watson:	0.543
Prob(Omnibus):	0.166	Jarque-Bera (JB):	2.781
Skew:	-0.566	Prob(JB):	0.249
Kurtosis:	2.175	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Robbery:

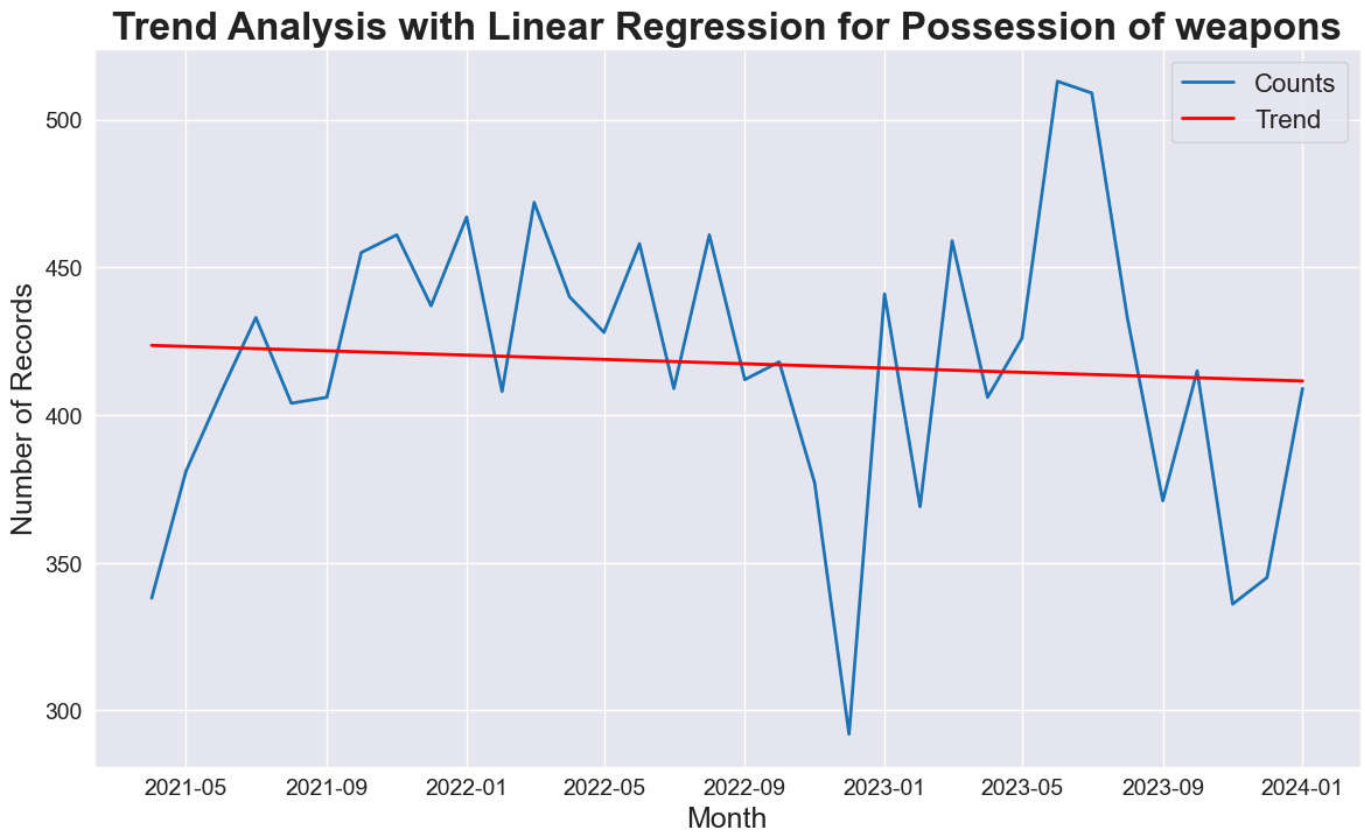
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.795
Model:	OLS	Adj. R-squared:	0.788
Method:	Least Squares	F-statistic:	124.0
Date:	Thu, 16 May 2024	Prob (F-statistic):	1.53e-12
Time:	15:09:13	Log-Likelihood:	-223.19
No. Observations:	34	AIC:	450.4
Df Residuals:	32	BIC:	453.4
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1.936e+04	1954.059	-9.909	0.000	-2.33e+04	-1.54e+04
timestamp	1.31e-05	1.18e-06	11.135	0.000	1.07e-05	1.55e-05

Omnibus:	0.728	Durbin-Watson:	1.447
Prob(Omnibus):	0.695	Jarque-Bera (JB):	0.746
Skew:	-0.126	Prob(JB):	0.689
Kurtosis:	2.320	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



Summary for Possession of weapons:

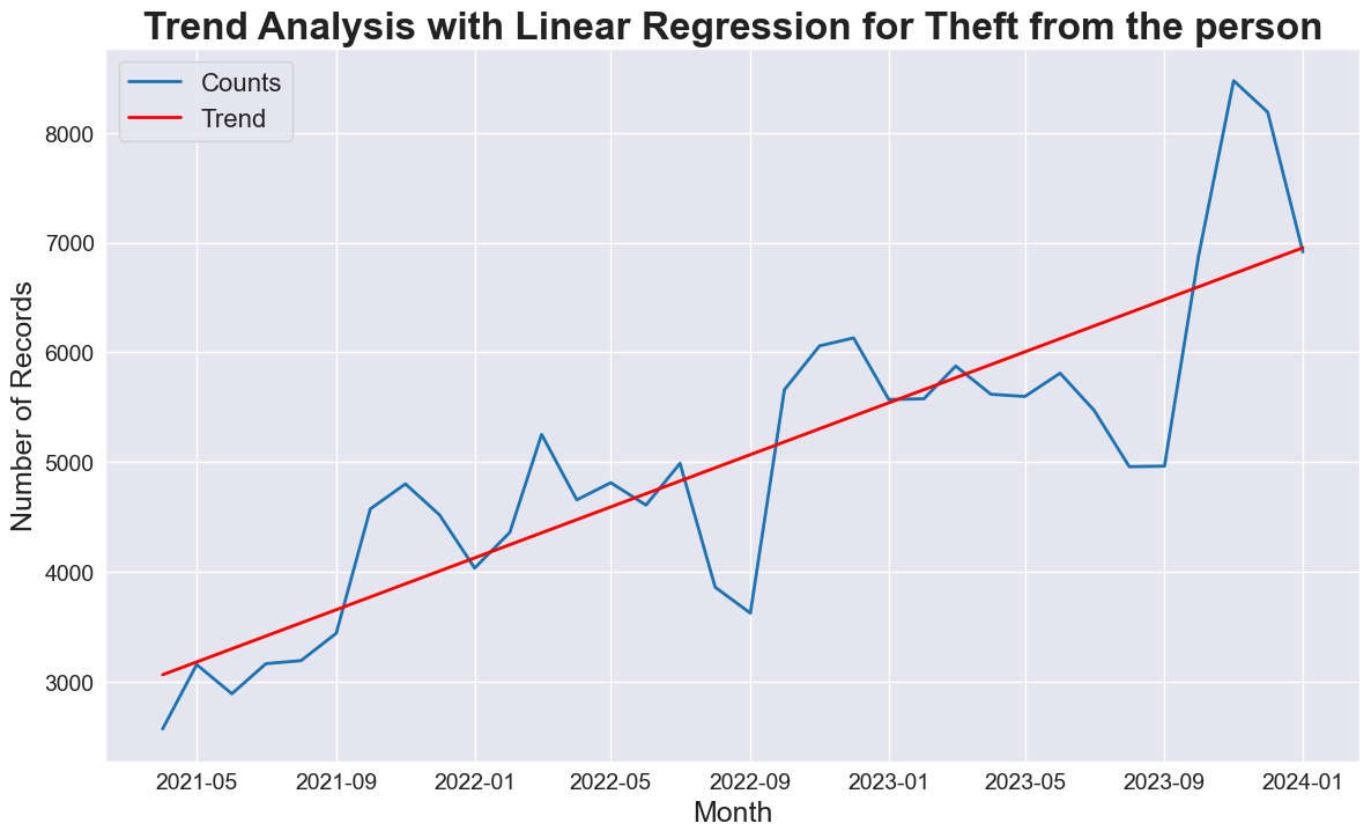
OLS Regression Results

Dep. Variable:	counts	R-squared:	0.006
Model:	OLS	Adj. R-squared:	-0.025
Method:	Least Squares	F-statistic:	0.1853
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.670
Time:	15:09:14	Log-Likelihood:	-179.21
No. Observations:	34	AIC:	362.4
Df Residuals:	32	BIC:	365.5
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	648.2562	536.041	1.209	0.235	-443.624	1740.136
timestamp	-1.389e-07	3.23e-07	-0.430	0.670	-7.96e-07	5.19e-07

Omnibus:	1.733	Durbin-Watson:	1.358
Prob(Omnibus):	0.420	Jarque-Bera (JB):	0.817
Skew:	-0.330	Prob(JB):	0.665
Kurtosis:	3.376	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.



```
Summary for Theft from the person:
                            OLS Regression Results
==============================================================================
Dep. Variable:                 counts   R-squared:                       0.715
Model:                            OLS   Adj. R-squared:                  0.706
Method:                 Least Squares   F-statistic:                     80.22
Date:                Thu, 16 May 2024   Prob (F-statistic):           3.13e-10
Time:                        15:09:16   Log-Likelihood:                -272.40
No. Observations:                  34   AIC:                             548.8
Df Residuals:                      32   BIC:                             551.9
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -6.94e+04   8308.122     -8.353      0.000   -8.63e+04   -5.25e+04
timestamp   4.48e-05      5e-06      8.956      0.000    3.46e-05     5.5e-05
==============================================================================
Omnibus:                        0.566   Durbin-Watson:                   0.972
Prob(Omnibus):                  0.754   Jarque-Bera (JB):                0.058
Skew:                          -0.029   Prob(JB):                        0.972
Kurtosis:                       3.193   Cond. No.                     1.07e+11
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Trend Analysis, Mann-Kendall Trend Test, and Seasonal Decomposition for Crime Data by Crime Type

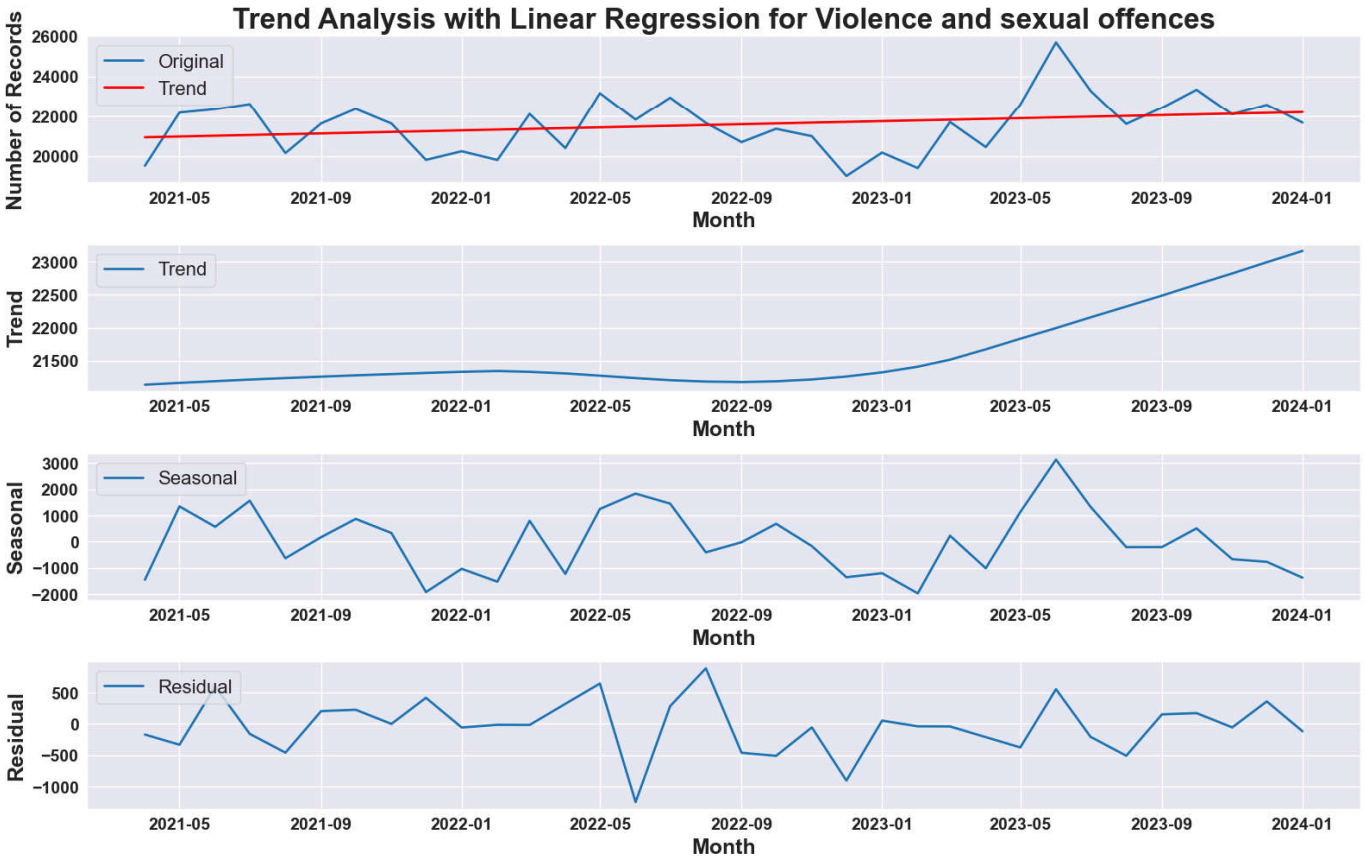
This code performs trend analysis using linear regression, the Mann-Kendall trend test, and seasonal decomposition on the monthly counts of crime records for each crime type. It plots the original counts, trend, and decomposition components (trend, seasonal, and residual) for each crime type and saves each plot as a PNG file in a specified directory. Additionally, it prints the summary of the regression model and the results of the Mann-Kendall trend test for each crime type.

In [47]:
```python
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.api import OLS, add_constant
from statsmodels.tsa.seasonal import STL
from scipy import stats

# Assuming 'data' is your DataFrame

# Ensure 'Month' is in datetime format
data['Month'] = pd.to_datetime(data['Month'])

# Function to perform trend analysis, Mann-Kendall Trend Test, and seasonal decomposition for e
def trend_analysis_by_crime_type(data, crime_type, save_path):
    # Filter data for the specific crime type
    data_crime = data[data['Crime type'] == crime_type]

    # Trend Analysis: Linear Regression
    data_grouped = data_crime.groupby('Month').size().reset_index(name='counts')
    data_grouped['timestamp'] = data_grouped['Month'].apply(lambda x: x.timestamp())
```

```
    # Add a constant to the predictor
    X = add_constant(data_grouped['timestamp'])
    y = data_grouped['counts']

    # Fit the linear regression model
    model = OLS(y, X).fit()

    # Add the trend predictions to the DataFrame
    data_grouped['trend'] = model.predict(X)

    # Mann-Kendall Trend Test
    mk_result = stats.kendalltau(data_grouped['timestamp'], data_grouped['counts'])
    print(f"Mann-Kendall Trend Test for {crime_type}: tau={mk_result.correlation}, p-value={mk_

    # Seasonal Decomposition
    stl = STL(data_grouped['counts'], seasonal=13, period=12)
    result = stl.fit()
    seasonal, trend, resid = result.seasonal, result.trend, result.resid

    # Plot the counts, trend, and decomposition
    plt.figure(figsize=(16, 12))  # Increase the plot size

    # Original Data and Trend
    plt.subplot(511)
    plt.plot(data_grouped['Month'], data_grouped['counts'], label='Original', linewidth=2)
    plt.plot(data_grouped['Month'], data_grouped['trend'], label='Trend', color='red', linewidt
    plt.title(f'Trend Analysis with Linear Regression for {crime_type}', fontsize=24, fontweigh
    plt.xlabel('Month', fontsize=18, fontweight='bold')
    plt.ylabel('Number of Records', fontsize=18, fontweight='bold')
    plt.xticks(fontsize=14, fontweight='bold')
    plt.yticks(fontsize=14, fontweight='bold')
    plt.legend(fontsize=16)

    # Seasonal Decomposition Plots
    plt.subplot(512)
    plt.plot(data_grouped['Month'], trend, label='Trend', linewidth=2)
    plt.xlabel('Month', fontsize=18, fontweight='bold')
    plt.ylabel('Trend', fontsize=18, fontweight='bold')
    plt.xticks(fontsize=14, fontweight='bold')
    plt.yticks(fontsize=14, fontweight='bold')
    plt.legend(loc='upper left', fontsize=16)

    plt.subplot(513)
    plt.plot(data_grouped['Month'], seasonal, label='Seasonal', linewidth=2)
    plt.xlabel('Month', fontsize=18, fontweight='bold')
    plt.ylabel('Seasonal', fontsize=18, fontweight='bold')
    plt.xticks(fontsize=14, fontweight='bold')
    plt.yticks(fontsize=14, fontweight='bold')
    plt.legend(loc='upper left', fontsize=16)

    plt.subplot(514)
    plt.plot(data_grouped['Month'], resid, label='Residual', linewidth=2)
    plt.xlabel('Month', fontsize=18, fontweight='bold')
    plt.ylabel('Residual', fontsize=18, fontweight='bold')
    plt.xticks(fontsize=14, fontweight='bold')
    plt.yticks(fontsize=14, fontweight='bold')
    plt.legend(loc='upper left', fontsize=16)

    plt.tight_layout()

    # Save the plot to the specified path
    plot_save_path = f"{save_path}Trend_Analysis_{crime_type.replace(' ', '_')}.png"
    plt.savefig(plot_save_path, bbox_inches='tight')
```

```
    # Show the plot
    plt.show()

    # Print the summary of the regression model
    print(f"Summary for {crime_type}:")
    print(model.summary())
    print("\n")

# Define the base path to save the plots
save_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\P

# Loop through each unique crime type and perform the trend analysis
for crime_type in data['Crime type'].unique():
    trend_analysis_by_crime_type(data, crime_type, save_path)
```

```
Mann-Kendall Trend Test for Violence and sexual offences: tau=0.16221033868092694, p-value=0.177
32914743926997
```



Summary for Violence and sexual offences:

OLS Regression Results

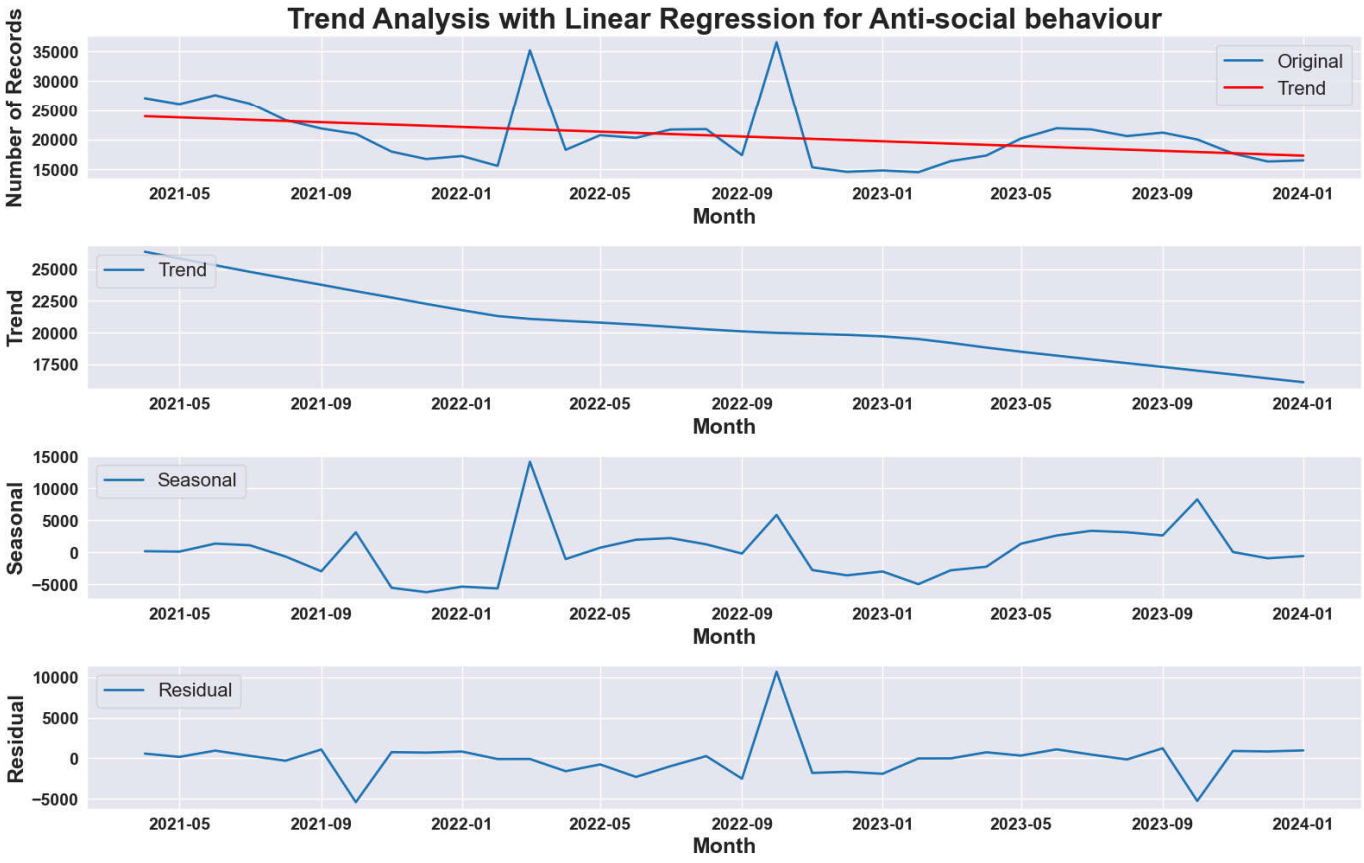
Dep. Variable:	counts	R-squared:	0.075
Model:	OLS	Adj. R-squared:	0.046
Method:	Least Squares	F-statistic:	2.579
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.118
Time:	15:11:55	Log-Likelihood:	-292.78
No. Observations:	34	AIC:	589.6
Df Residuals:	32	BIC:	592.6
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2712.2348	1.51e+04	-0.179	0.859	-3.35e+04	2.81e+04
timestamp	1.463e-05	9.11e-06	1.606	0.118	-3.93e-06	3.32e-05

Omnibus:	1.180	Durbin-Watson:	1.364
Prob(Omnibus):	0.554	Jarque-Bera (JB):	0.495
Skew:	0.272	Prob(JB):	0.781
Kurtosis:	3.230	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Anti-social behaviour: tau=-0.30837789661319076, p-value=0.010328626
81112093



Summary for Anti-social behaviour:

OLS Regression Results

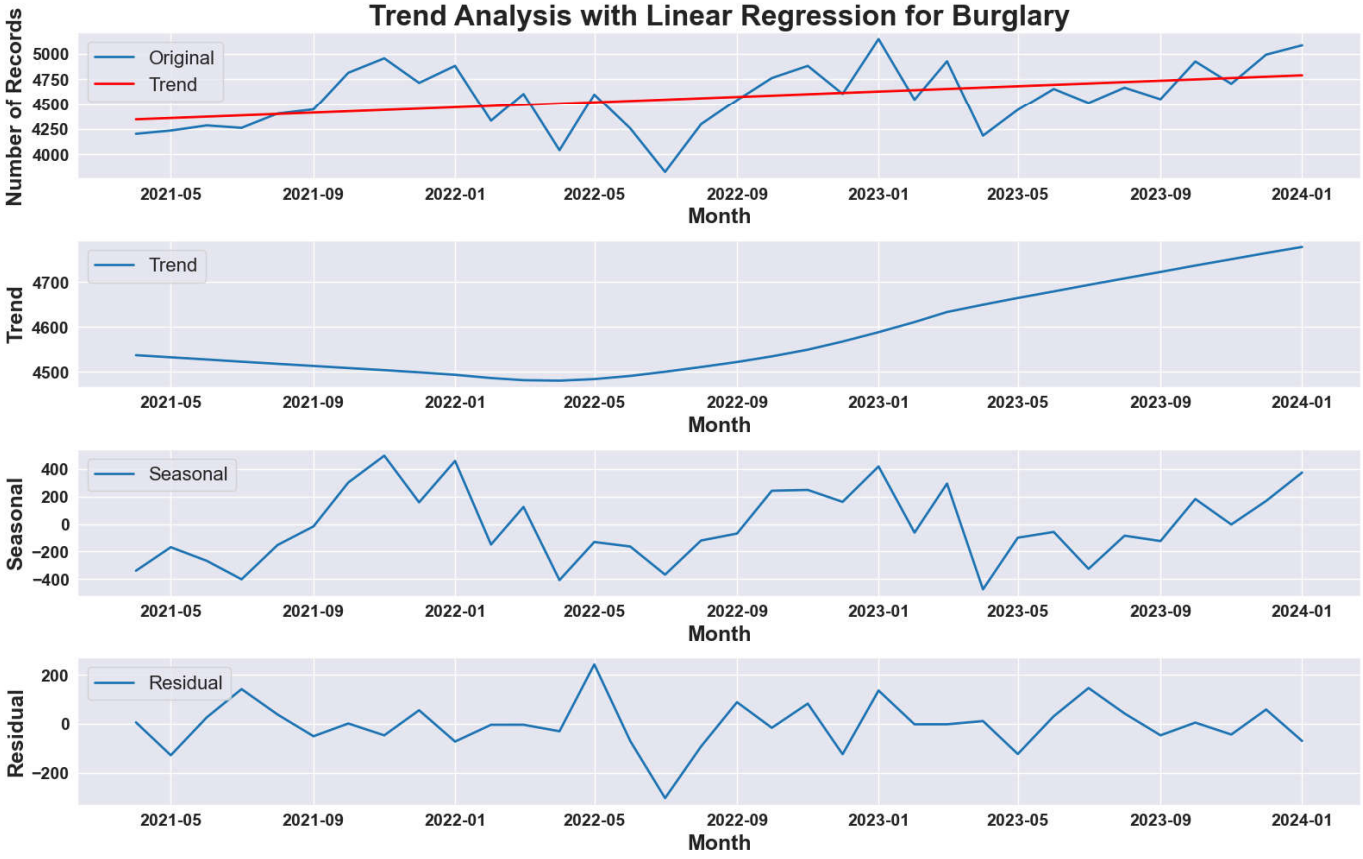
Dep. Variable:	counts	R-squared:	0.146
Model:	OLS	Adj. R-squared:	0.119
Method:	Least Squares	F-statistic:	5.453
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0260
Time:	15:11:59	Log-Likelihood:	-336.41
No. Observations:	34	AIC:	676.8
Df Residuals:	32	BIC:	679.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.481e+05	5.46e+04	2.713	0.011	3.69e+04	2.59e+05
timestamp	-7.677e-05	3.29e-05	-2.335	0.026	-0.000	-9.8e-06

Omnibus:	18.853	Durbin-Watson:	1.998
Prob(Omnibus):	0.000	Jarque-Bera (JB):	25.552
Skew:	1.500	Prob(JB):	2.83e-06
Kurtosis:	6.006	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Burglary: tau=0.3318466976036767, p-value=0.005821858644766352



Summary for Burglary:

OLS Regression Results

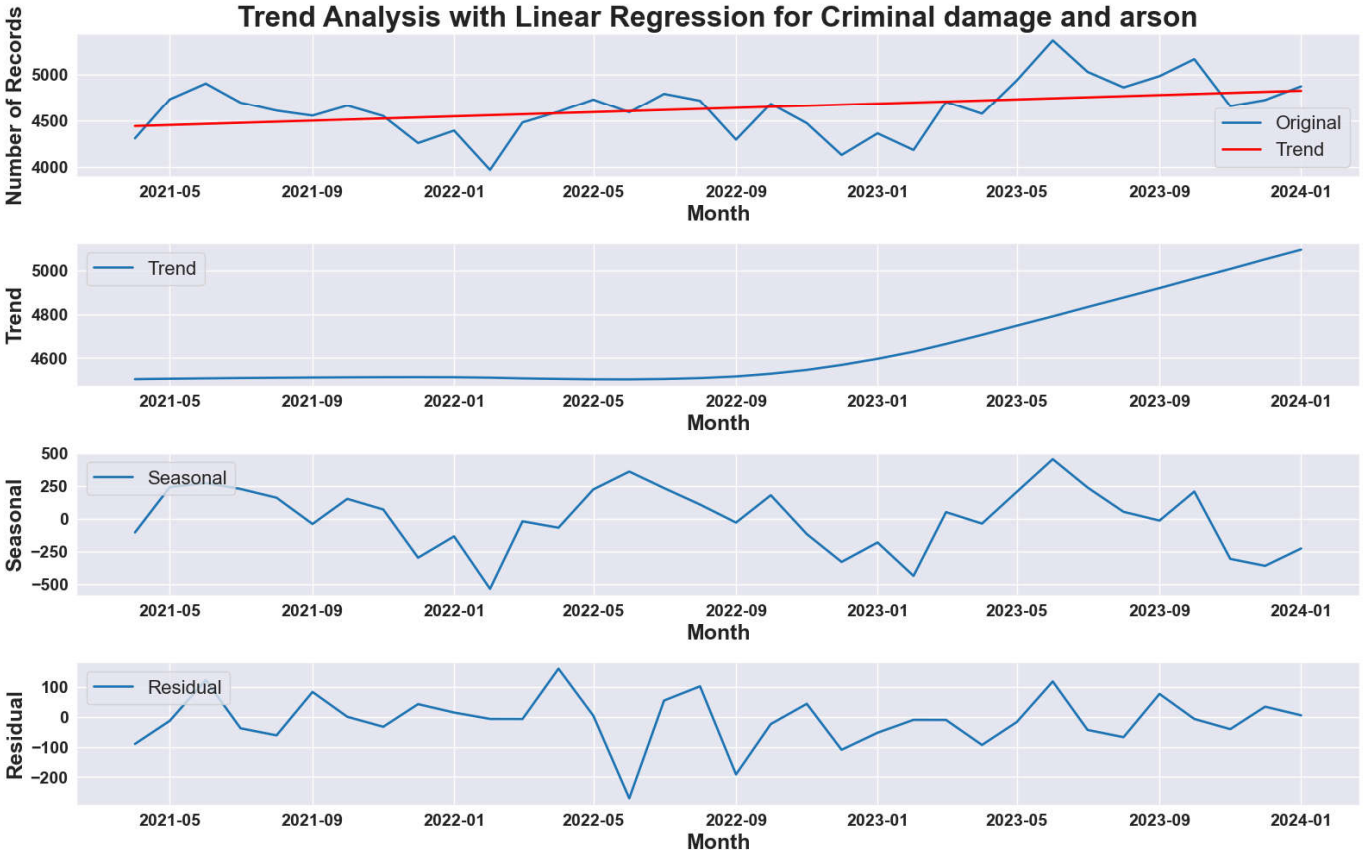
Dep. Variable:	counts	R-squared:	0.182
Model:	OLS	Adj. R-squared:	0.157
Method:	Least Squares	F-statistic:	7.138
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0118
Time:	15:12:03	Log-Likelihood:	-239.52
No. Observations:	34	AIC:	483.0
Df Residuals:	32	BIC:	486.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-3870.6569	3158.287	-1.226	0.229	-1.03e+04	2562.563
timestamp	5.081e-06	1.9e-06	2.672	0.012	1.21e-06	8.95e-06

Omnibus:	0.549	Durbin-Watson:	1.430
Prob(Omnibus):	0.760	Jarque-Bera (JB):	0.238
Skew:	-0.205	Prob(JB):	0.888
Kurtosis:	2.997	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Criminal damage and arson: tau=0.23351158645276296, p-value=0.052136849055109694



Summary for Criminal damage and arson:

OLS Regression Results

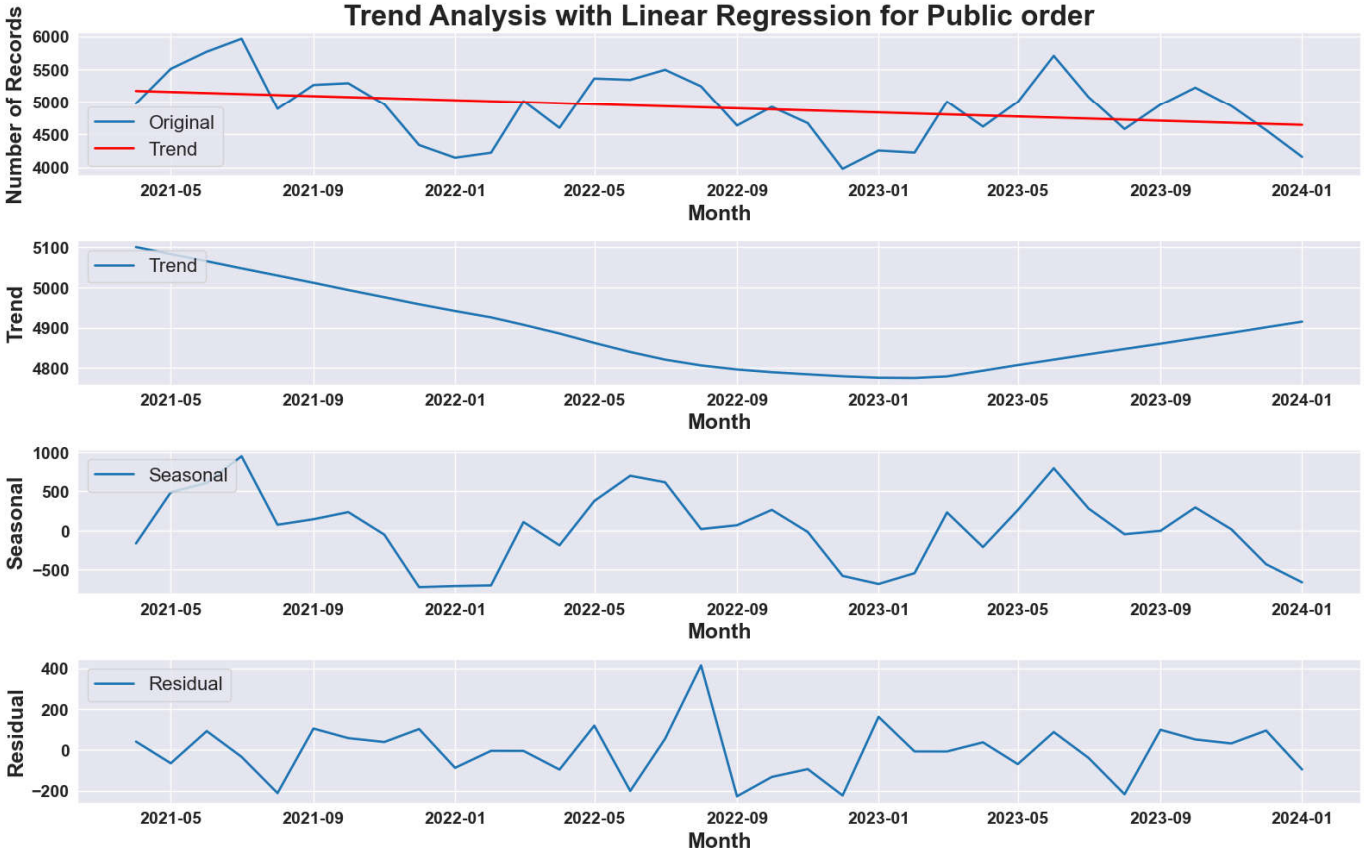
Dep. Variable:	counts	R-squared:	0.153
Model:	OLS	Adj. R-squared:	0.126
Method:	Least Squares	F-statistic:	5.762
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0224
Time:	15:12:06	Log-Likelihood:	-238.53
No. Observations:	34	AIC:	481.1
Df Residuals:	32	BIC:	484.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2728.9925	3067.745	-0.890	0.380	-8977.784	3519.799
timestamp	4.434e-06	1.85e-06	2.400	0.022	6.71e-07	8.2e-06

Omnibus:	0.727	Durbin-Watson:	1.047
Prob(Omnibus):	0.695	Jarque-Bera (JB):	0.347
Skew:	-0.247	Prob(JB):	0.841
Kurtosis:	3.023	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Public order: tau=-0.22281639928698754, p-value=0.063874416294357



Summary for Public order:

OLS Regression Results

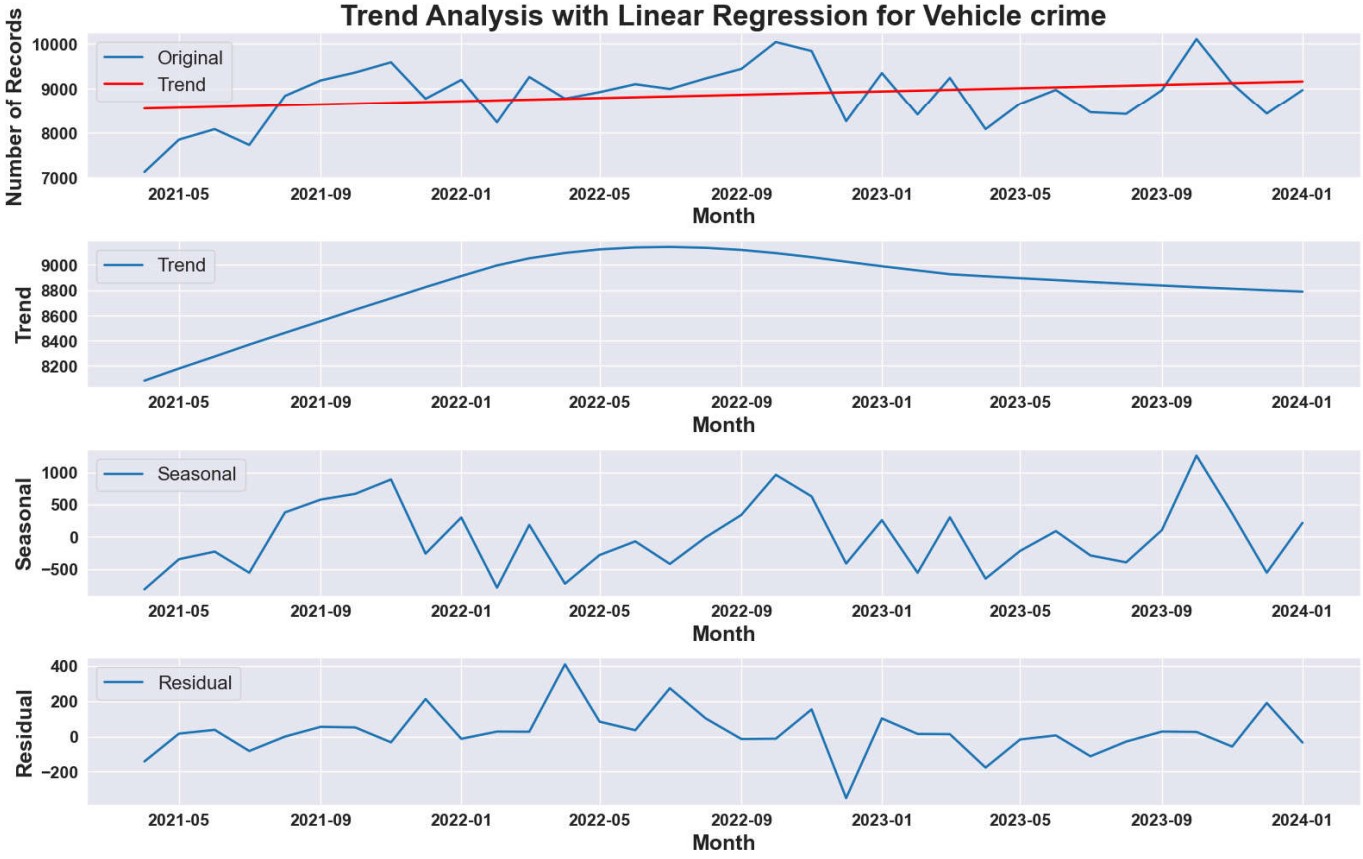
Dep. Variable:	counts	R-squared:	0.098
Model:	OLS	Adj. R-squared:	0.070
Method:	Least Squares	F-statistic:	3.491
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.0709
Time:	15:12:09	Log-Likelihood:	-257.44
No. Observations:	34	AIC:	518.9
Df Residuals:	32	BIC:	521.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.49e+04	5349.876	2.786	0.009	4005.732	2.58e+04
timestamp	-6.019e-06	3.22e-06	-1.869	0.071	-1.26e-05	5.42e-07

Omnibus:	0.713	Durbin-Watson:	0.975
Prob(Omnibus):	0.700	Jarque-Bera (JB):	0.748
Skew:	-0.145	Prob(JB):	0.688
Kurtosis:	2.333	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Vehicle crime: tau=0.14438502673796794, p-value=0.22983658719268163



Summary for Vehicle crime:

OLS Regression Results

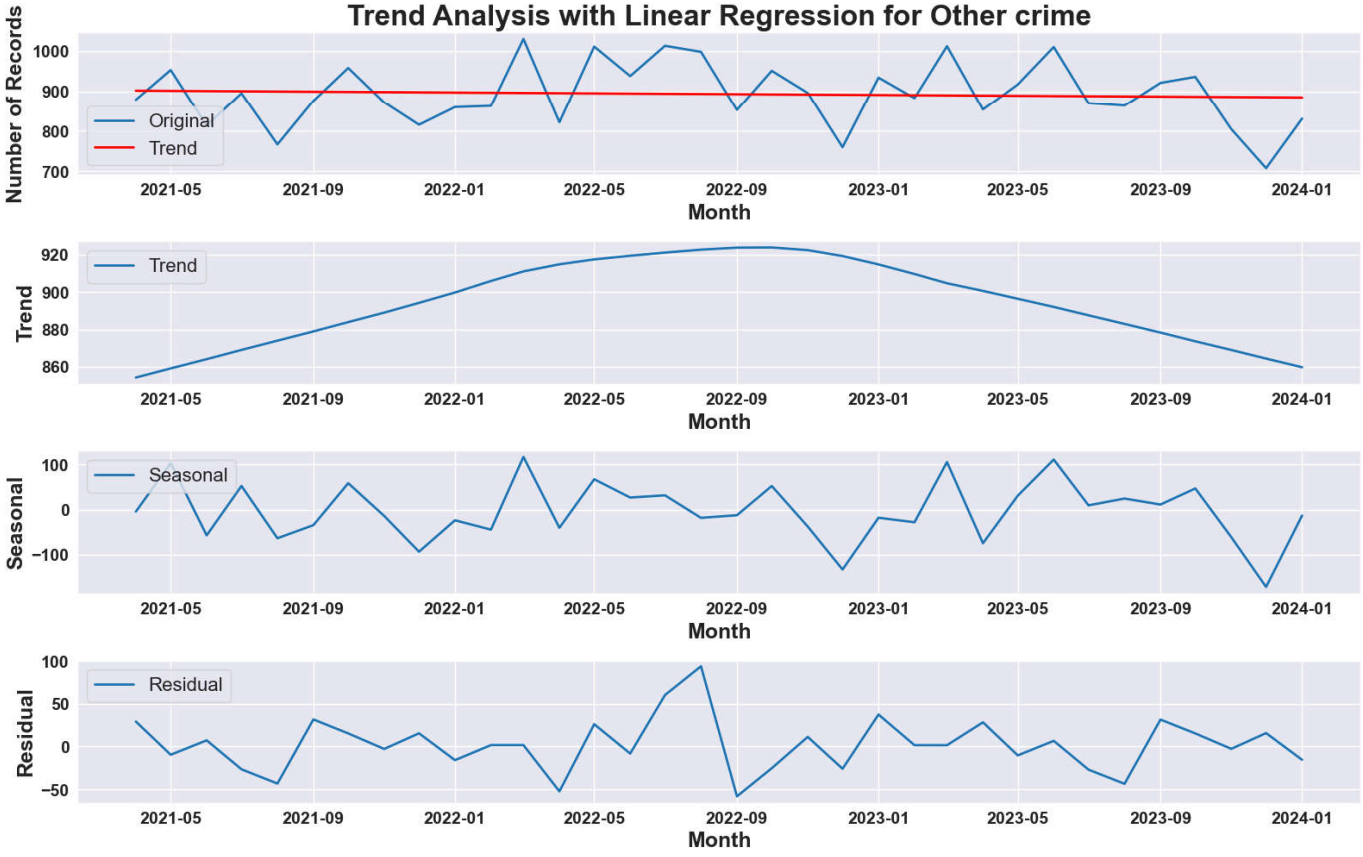
Dep. Variable:	counts	R-squared:	0.079
Model:	OLS	Adj. R-squared:	0.050
Method:	Least Squares	F-statistic:	2.739
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.108
Time:	15:12:13	Log-Likelihood:	-266.85
No. Observations:	34	AIC:	537.7
Df Residuals:	32	BIC:	540.7
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2821.7351	7056.158	-0.400	0.692	-1.72e+04	1.16e+04
timestamp	7.031e-06	4.25e-06	1.655	0.108	-1.62e-06	1.57e-05

Omnibus:	0.806	Durbin-Watson:	1.250
Prob(Omnibus):	0.668	Jarque-Bera (JB):	0.780
Skew:	-0.118	Prob(JB):	0.677
Kurtosis:	2.297	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Other crime: tau=-0.055258467023172914, p-value=0.645833696254254



Summary for Other crime:

OLS Regression Results

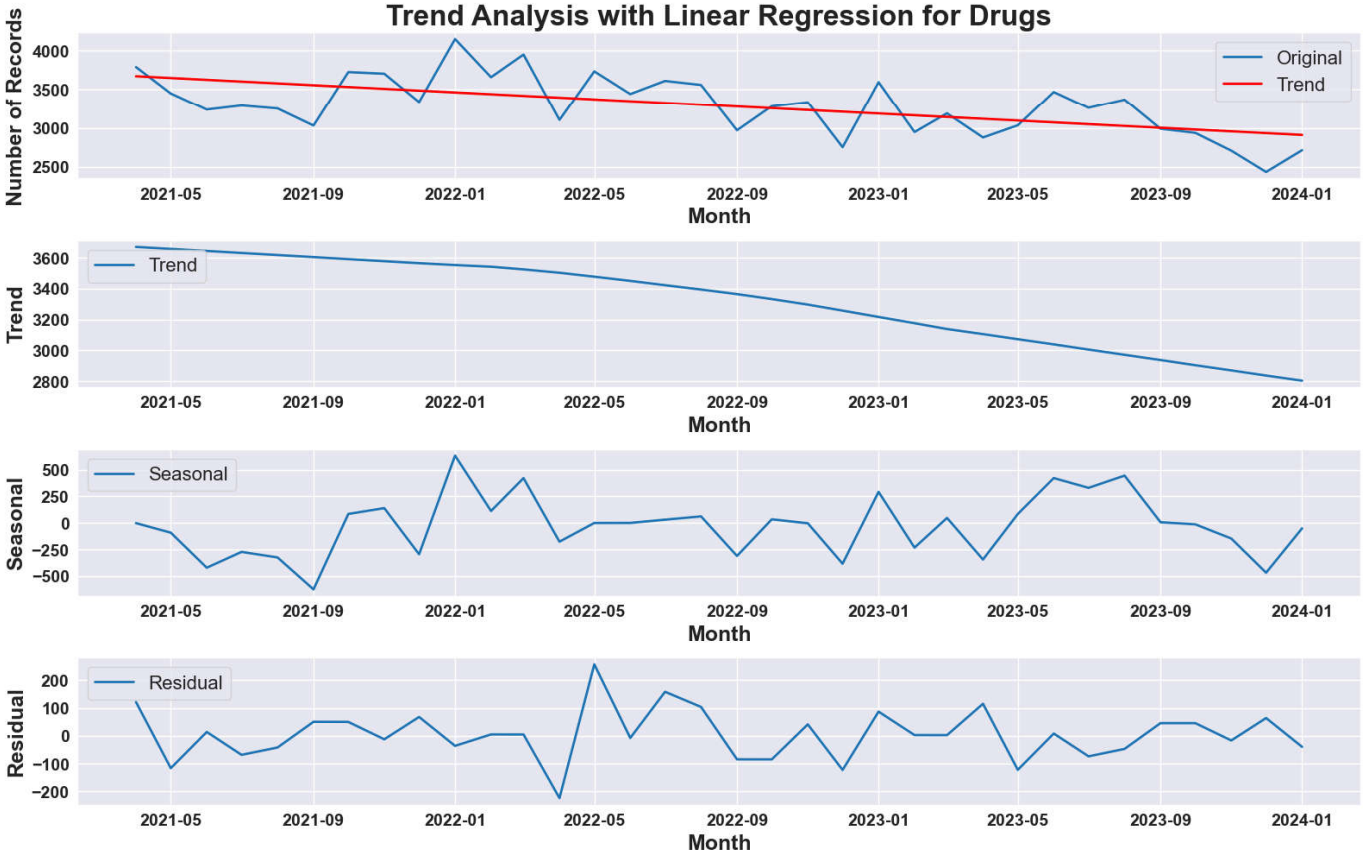
Dep. Variable:	counts	R-squared:	0.004
Model:	OLS	Adj. R-squared:	-0.027
Method:	Least Squares	F-statistic:	0.1343
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.716
Time:	15:12:16	Log-Likelihood:	-196.41
No. Observations:	34	AIC:	396.8
Df Residuals:	32	BIC:	399.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1218.9176	888.858	1.371	0.180	-591.628	3029.463
timestamp	-1.961e-07	5.35e-07	-0.366	0.716	-1.29e-06	8.94e-07

Omnibus:	0.288	Durbin-Watson:	1.934
Prob(Omnibus):	0.866	Jarque-Bera (JB):	0.474
Skew:	-0.087	Prob(JB):	0.789
Kurtosis:	2.448	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Drugs: tau=-0.42959001782531203, p-value=0.0003533318117686386



Summary for Drugs:

OLS Regression Results

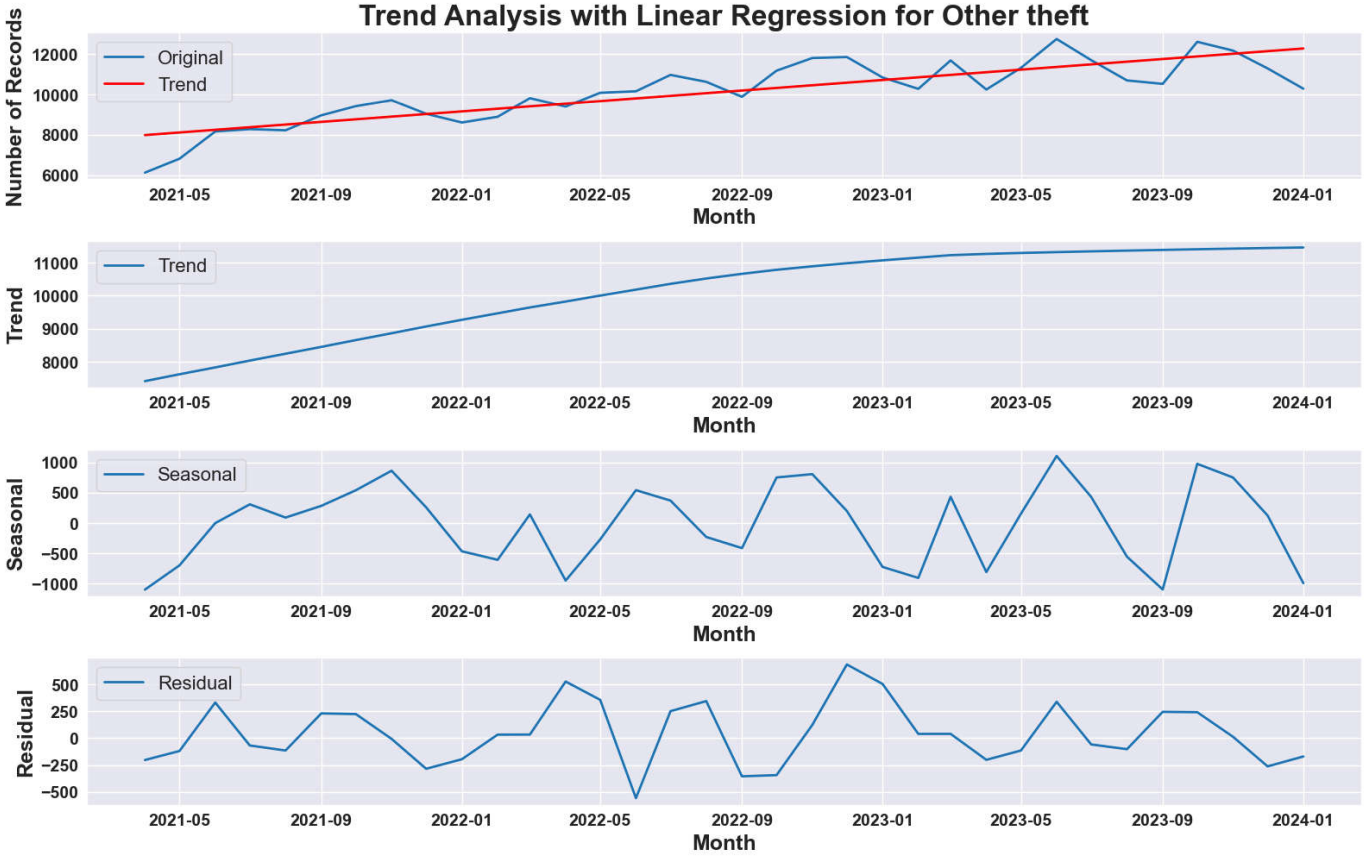
Dep. Variable:	counts	R-squared:	0.351
Model:	OLS	Adj. R-squared:	0.331
Method:	Least Squares	F-statistic:	17.31
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.000222
Time:	15:12:20	Log-Likelihood:	-243.07
No. Observations:	34	AIC:	490.1
Df Residuals:	32	BIC:	493.2
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.788e+04	3506.654	5.099	0.000	1.07e+04	2.5e+04
timestamp	-8.786e-06	2.11e-06	-4.161	0.000	-1.31e-05	-4.48e-06

Omnibus:	1.695	Durbin-Watson:	1.770
Prob(Omnibus):	0.428	Jarque-Bera (JB):	1.169
Skew:	0.170	Prob(JB):	0.557
Kurtosis:	2.158	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Other theft: tau=0.6755793226381462, p-value=1.9266250487277596e-08



Summary for Other theft:

OLS Regression Results

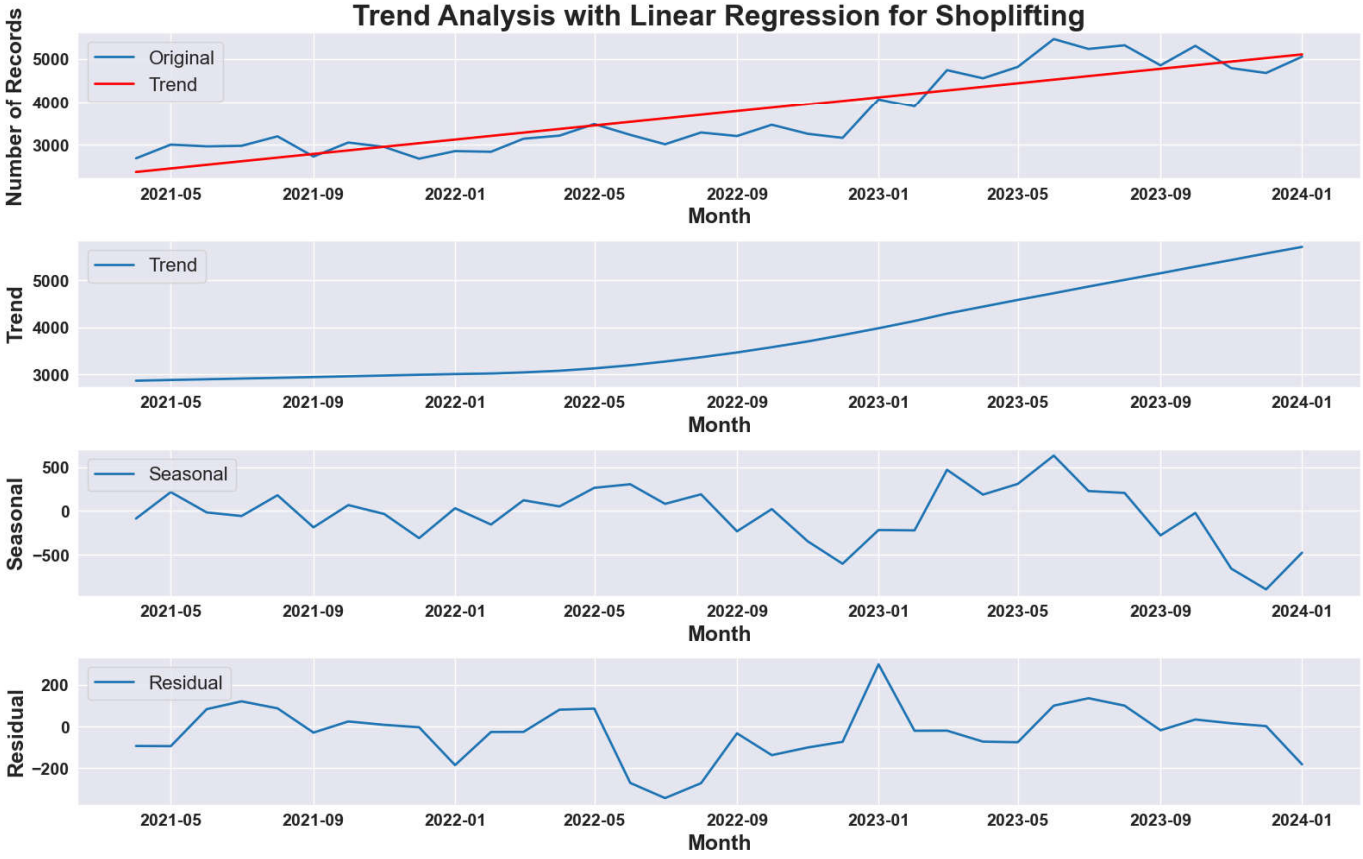
Dep. Variable:	counts	R-squared:	0.696
Model:	OLS	Adj. R-squared:	0.686
Method:	Least Squares	F-statistic:	73.11
Date:	Thu, 16 May 2024	Prob (F-statistic):	9.02e-10
Time:	15:12:23	Log-Likelihood:	-277.38
No. Observations:	34	AIC:	558.8
Df Residuals:	32	BIC:	561.8
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-7.211e+04	9619.681	-7.496	0.000	-9.17e+04	-5.25e+04
timestamp	4.953e-05	5.79e-06	8.551	0.000	3.77e-05	6.13e-05

Omnibus:	1.725	Durbin-Watson:	1.047
Prob(Omnibus):	0.422	Jarque-Bera (JB):	1.453
Skew:	-0.491	Prob(JB):	0.484
Kurtosis:	2.755	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Shoplifting: tau=0.696969696969697, p-value=6.777078240227204e-09



Summary for Shoplifting:

OLS Regression Results

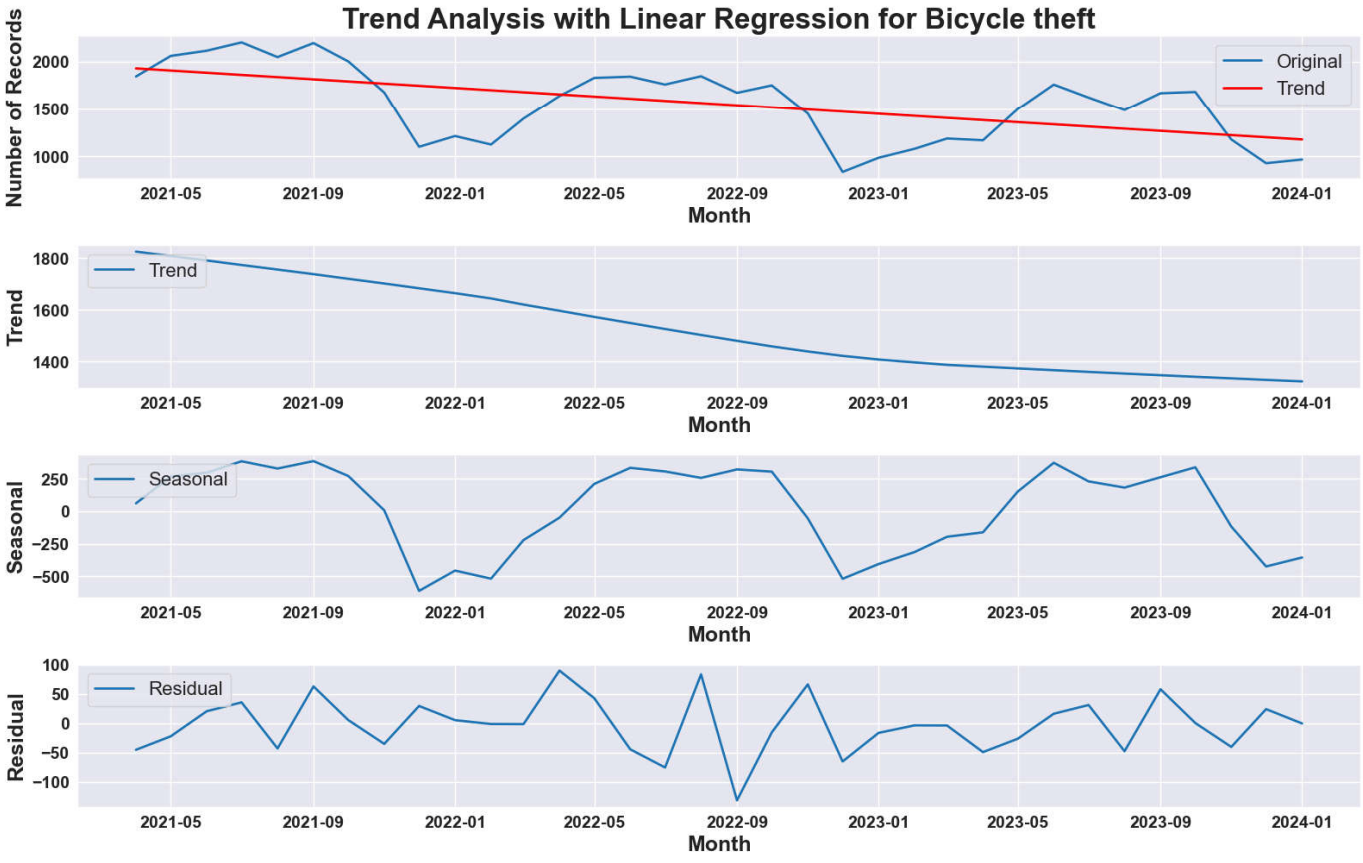
Dep. Variable:	counts	R-squared:	0.783
Model:	OLS	Adj. R-squared:	0.776
Method:	Least Squares	F-statistic:	115.2
Date:	Thu, 16 May 2024	Prob (F-statistic):	3.88e-12
Time:	15:12:26	Log-Likelihood:	-254.24
No. Observations:	34	AIC:	512.5
Df Residuals:	32	BIC:	515.5
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-4.853e+04	4869.807	-9.966	0.000	-5.85e+04	-3.86e+04
timestamp	3.148e-05	2.93e-06	10.735	0.000	2.55e-05	3.75e-05

Omnibus:	0.990	Durbin-Watson:	0.625
Prob(Omnibus):	0.610	Jarque-Bera (JB):	0.849
Skew:	0.095	Prob(JB):	0.654
Kurtosis:	2.249	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Bicycle theft: tau=-0.41176470588235303, p-value=0.00061605333831548
18



Summary for Bicycle theft:

OLS Regression Results

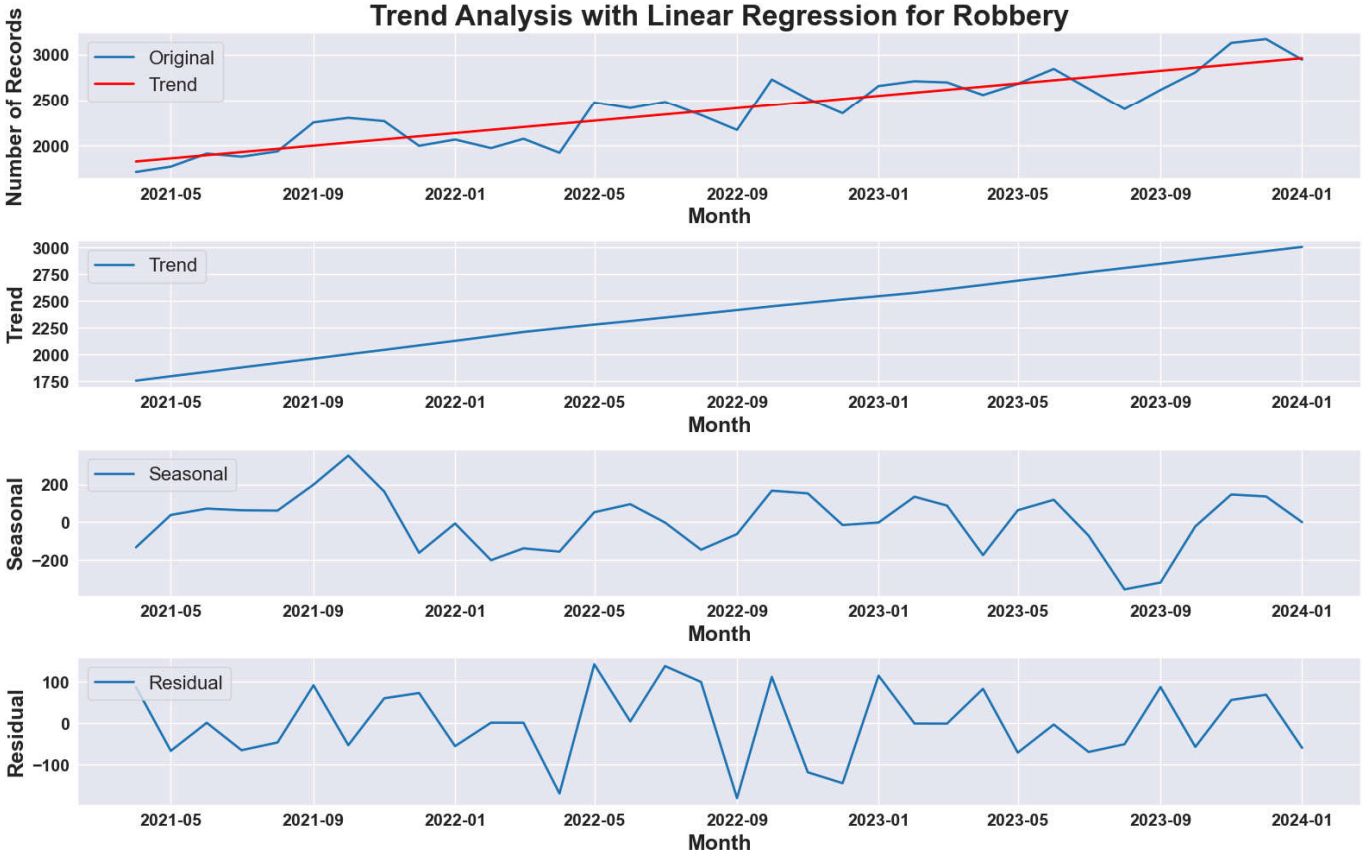
Dep. Variable:	counts	R-squared:	0.329
Model:	OLS	Adj. R-squared:	0.309
Method:	Least Squares	F-statistic:	15.72
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.000386
Time:	15:12:30	Log-Likelihood:	-244.01
No. Observations:	34	AIC:	492.0
Df Residuals:	32	BIC:	495.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.585e+04	3604.827	4.396	0.000	8502.788	2.32e+04
timestamp	-8.607e-06	2.17e-06	-3.965	0.000	-1.3e-05	-4.19e-06

Omnibus:	3.592	Durbin-Watson:	0.543
Prob(Omnibus):	0.166	Jarque-Bera (JB):	2.781
Skew:	-0.566	Prob(JB):	0.249
Kurtosis:	2.175	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Robbery: tau=0.714795008912656, p-value=2.771297472310716e-09



Summary for Robbery:

OLS Regression Results

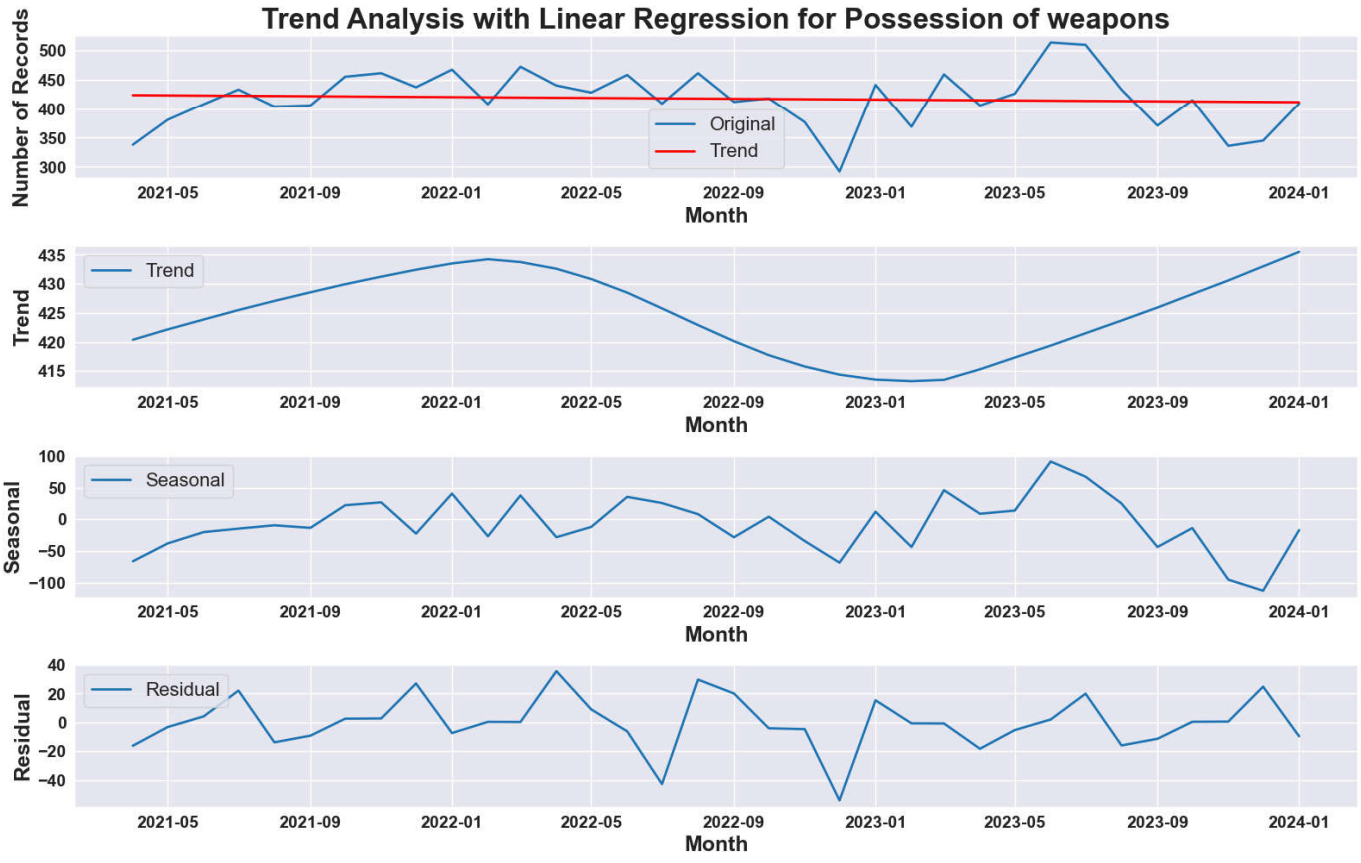
Dep. Variable:	counts	R-squared:	0.795
Model:	OLS	Adj. R-squared:	0.788
Method:	Least Squares	F-statistic:	124.0
Date:	Thu, 16 May 2024	Prob (F-statistic):	1.53e-12
Time:	15:12:34	Log-Likelihood:	-223.19
No. Observations:	34	AIC:	450.4
Df Residuals:	32	BIC:	453.4
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1.936e+04	1954.059	-9.909	0.000	-2.33e+04	-1.54e+04
timestamp	1.31e-05	1.18e-06	11.135	0.000	1.07e-05	1.55e-05

Omnibus:	0.728	Durbin-Watson:	1.447
Prob(Omnibus):	0.695	Jarque-Bera (JB):	0.746
Skew:	-0.126	Prob(JB):	0.689
Kurtosis:	2.320	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Possession of weapons: tau=-0.03222950821147798, p-value=0.7894793972367413



Summary for Possession of weapons:

OLS Regression Results

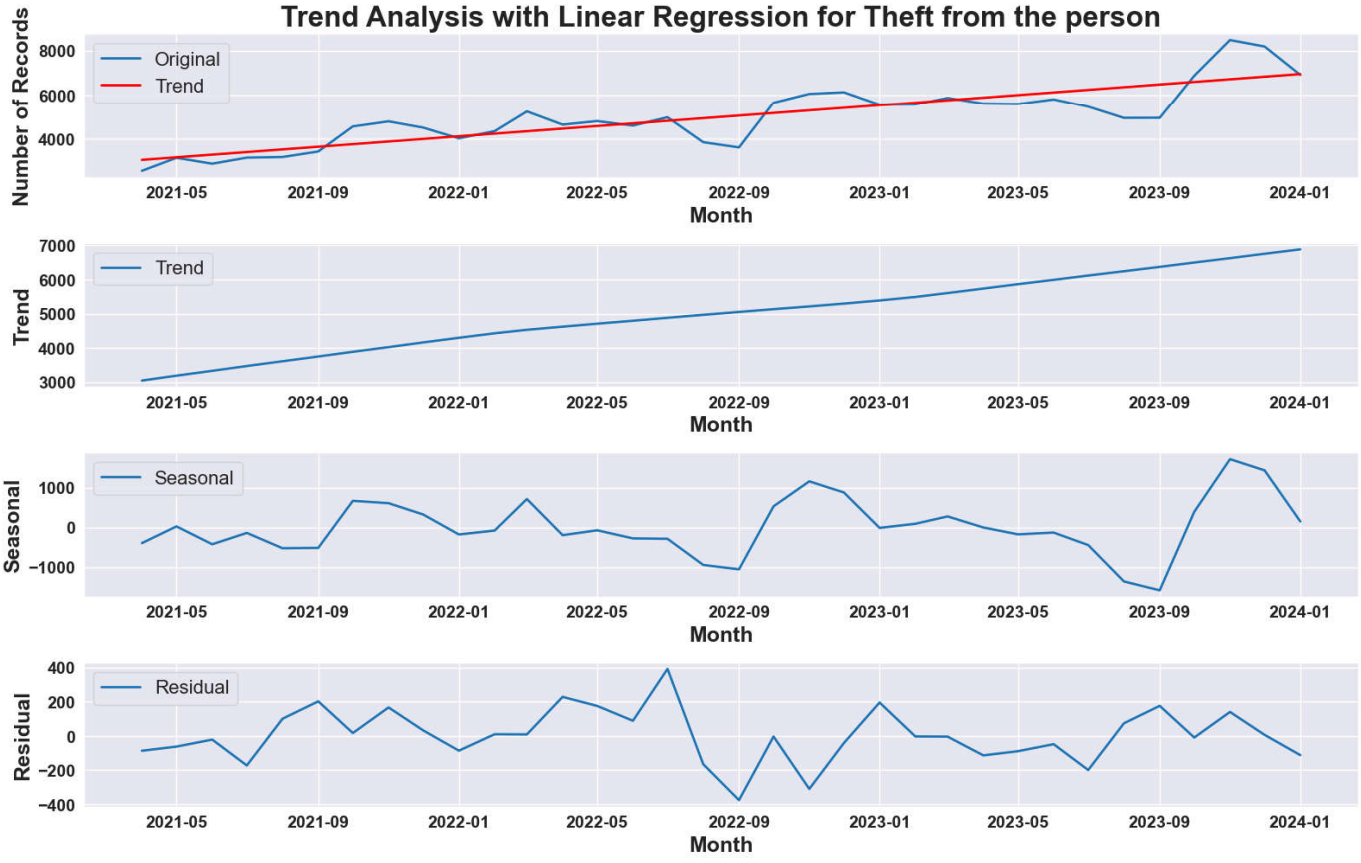
Dep. Variable:	counts	R-squared:	0.006
Model:	OLS	Adj. R-squared:	-0.025
Method:	Least Squares	F-statistic:	0.1853
Date:	Thu, 16 May 2024	Prob (F-statistic):	0.670
Time:	15:12:37	Log-Likelihood:	-179.21
No. Observations:	34	AIC:	362.4
Df Residuals:	32	BIC:	365.5
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	648.2562	536.041	1.209	0.235	-443.624	1740.136
timestamp	-1.389e-07	3.23e-07	-0.430	0.670	-7.96e-07	5.19e-07

Omnibus:	1.733	Durbin-Watson:	1.358
Prob(Omnibus):	0.420	Jarque-Bera (JB):	0.817
Skew:	-0.330	Prob(JB):	0.665
Kurtosis:	3.376	Cond. No.	1.07e+11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are strong multicollinearity or other numerical problems.

Mann-Kendall Trend Test for Theft from the person: tau=0.6648841354723708, p-value=3.2111302497211145e-08



```
Summary for Theft from the person:
                            OLS Regression Results
==============================================================================
Dep. Variable:                 counts   R-squared:                       0.715
Model:                            OLS   Adj. R-squared:                  0.706
Method:                 Least Squares   F-statistic:                     80.22
Date:                Thu, 16 May 2024   Prob (F-statistic):           3.13e-10
Time:                        15:12:40   Log-Likelihood:                -272.40
No. Observations:                  34   AIC:                             548.8
Df Residuals:                      32   BIC:                             551.9
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -6.94e+04   8308.122     -8.353      0.000   -8.63e+04   -5.25e+04
timestamp   4.48e-05      5e-06      8.956      0.000    3.46e-05     5.5e-05
==============================================================================
Omnibus:                        0.566   Durbin-Watson:                   0.972
Prob(Omnibus):                  0.754   Jarque-Bera (JB):                0.058
Skew:                          -0.029   Prob(JB):                        0.972
Kurtosis:                       3.193   Cond. No.                     1.07e+11
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.07e+11. This might indicate that there are
strong multicollinearity or other numerical problems.
```

The Mann-Kendall Trend Test result for the crime type two key values: tau (τ) and p-value. Here's an explanation of what these values mean and their implications:

Mann-Kendall Trend Test

The Mann-Kendall Trend Test is a non-parametric test used to identify trends in time series data. It assesses whether there is a statistically significant trend (increasing or decreasing) in the data over time.

Result Explanation

1. **Tau (τ) Value**: 0.16221033868092694
   - **Interpretation**: Tau is a measure of the strength and direction of the trend. It ranges from -1 to 1.
     - **Positive Tau**: Indicates an increasing trend.
     - **Negative Tau**: Indicates a decreasing trend.
     - **Tau Close to 0**: Indicates little to no trend.
   - In this case, τ = 0.162 indicates a weak positive trend, suggesting that "Violence and sexual offences" might have been increasing slightly over time.
2. **P-value**: 0.17732914743926997
   - **Interpretation**: The p-value indicates the probability that the observed trend is due to random chance.
     - **Low P-value (< 0.05)**: Indicates strong evidence against the null hypothesis (no trend), suggesting that the trend is statistically significant.

- **High P-value (≥ 0.05)**: Indicates weak evidence against the null hypothesis, suggesting that the trend is not statistically significant.

Hot Spot Analysis: Creating a Heat Map of Crime Concentrations

This code creates a heat map to visualize crime concentrations based on the latitude and longitude of crime records. It processes the data to remove any rows with missing geographical coordinates, centers the map based on the average location, and generates a heat map to identify crime hot spots. The resulting map is saved as an HTML file.

In [49]:
```
import pandas as pd
import folium
from folium.plugins import HeatMap

# Assuming 'data' is your DataFrame with 'Latitude' and 'Longitude' columns
# Check for NaN values in the latitude and longitude columns
print(data[['Latitude', 'Longitude']].isna().sum())

# Remove rows with NaN values in the latitude and longitude columns
data_clean = data.dropna(subset=['Latitude', 'Longitude'])

# Recheck to ensure there are no NaN values left
print(data_clean[['Latitude', 'Longitude']].isna().sum())

# Create a map centered at a given location
map_center = [data_clean['Latitude'].mean(), data_clean['Longitude'].mean()]
crime_map = folium.Map(location=map_center, zoom_start=12)

# Prepare the heat data
heat_data = [[row['Latitude'], row['Longitude']] for index, row in data_clean.iterrows()]

# Create the heatmap and add it to the map
HeatMap(heat_data).add_to(crime_map)

# Save the heat map as an HTML file
crime_map.save(r'C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Polic
```

```
Latitude     64222
Longitude    64222
dtype: int64
Latitude     0
Longitude    0
dtype: int64
```

In [54]:
```
import folium
from folium.plugins import HeatMap
import os
import pandas as pd

# Assuming 'data' is your DataFrame

# Base directory to save the HTML files
base_path = r"C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\H
if not os.path.exists(base_path):
    os.makedirs(base_path)

# Check if the necessary columns are in the DataFrame
if 'Latitude' in data.columns and 'Longitude' in data.columns and 'Crime type' in data.columns:
    # Loop through each unique crime type
```

```
    for crime_type in data['Crime type'].unique():
        # Filter data for the specific crime type
        data_crime = data[data['Crime type'] == crime_type]

        # Remove rows with NaN or non-numeric values in 'Latitude' or 'Longitude'
        data_crime = data_crime.dropna(subset=['Latitude', 'Longitude'])
        data_crime['Latitude'] = pd.to_numeric(data_crime['Latitude'], errors='coerce')
        data_crime['Longitude'] = pd.to_numeric(data_crime['Longitude'], errors='coerce')
        data_crime = data_crime.dropna(subset=['Latitude', 'Longitude'])

        # Determine the center of the map
        if not data_crime.empty:
            map_center = [data_crime['Latitude'].mean(), data_crime['Longitude'].mean()]

            # Create a Folium map centered around the mean coordinates
            crime_map = folium.Map(location=map_center, zoom_start=12)

            # Create a list of locations for the heat map
            heat_data = [[row['Latitude'], row['Longitude']] for index, row in data_crime.iterr

            # Add the heat map to the Folium map
            if heat_data:
                HeatMap(heat_data).add_to(crime_map)

                # Define the path to save the map
                save_path = os.path.join(base_path, f"Crime_Heat_Map_{crime_type.replace(' ', '_

                # Save the map to an HTML file
                crime_map.save(save_path)

                # Optionally, print the path where the map is saved
                print(f"Heat map for {crime_type} saved to {save_path}")
            else:
                print(f"No valid data for heat map for {crime_type}.")
```

```
Heat map for Violence and sexual offences saved to C:\Users\jckat\OneDrive\Documents\GitHub\Spac
e_TIme_Exploration_of_Metro_Police\HTML\Crime_Heat_Map_Violence_and_sexual_offences.html
Heat map for Anti-social behaviour saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_
Exploration_of_Metro_Police\HTML\Crime_Heat_Map_Anti-social_behaviour.html
Heat map for Burglary saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_o
f_Metro_Police\HTML\Crime_Heat_Map_Burglary.html
Heat map for Criminal damage and arson saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_T
Ime_Exploration_of_Metro_Police\HTML\Crime_Heat_Map_Criminal_damage_and_arson.html
Heat map for Public order saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explorati
on_of_Metro_Police\HTML\Crime_Heat_Map_Public_order.html
Heat map for Vehicle crime saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explorat
ion_of_Metro_Police\HTML\Crime_Heat_Map_Vehicle_crime.html
Heat map for Other crime saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploratio
n_of_Metro_Police\HTML\Crime_Heat_Map_Other_crime.html
Heat map for Drugs saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_M
etro_Police\HTML\Crime_Heat_Map_Drugs.html
Heat map for Other theft saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploratio
n_of_Metro_Police\HTML\Crime_Heat_Map_Other_theft.html
Heat map for Shoplifting saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploratio
n_of_Metro_Police\HTML\Crime_Heat_Map_Shoplifting.html
Heat map for Bicycle theft saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explorat
ion_of_Metro_Police\HTML\Crime_Heat_Map_Bicycle_theft.html
Heat map for Robbery saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of
_Metro_Police\HTML\Crime_Heat_Map_Robbery.html
Heat map for Possession of weapons saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_
Exploration_of_Metro_Police\HTML\Crime_Heat_Map_Possession_of_weapons.html
Heat map for Theft from the person saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_
Exploration_of_Metro_Police\HTML\Crime_Heat_Map_Theft_from_the_person.html
```

Hot Spot Analysis: Creating Heat Maps for Crime Types

This code generates heat maps for different crime types, visualizing crime concentrations based on latitude and longitude. Each heat map is saved as an HTML file in a specified directory, providing a visual representation of hot spots for each crime type.

In [59]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import DBSCAN
import os

# Load the CSV file into a DataFrame
file_path = r'C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\m
data = pd.read_csv(file_path)

# Display basic information about the DataFrame
print(data.info())
print(data.head())

# Convert the 'Month' column to datetime format and extract year and month
data['Month'] = pd.to_datetime(data['Month'], format='%Y-%m')
data['year'] = data['Month'].dt.year
data['month_num'] = data['Month'].dt.month

# Check for missing values and remove rows with missing Latitude or Longitude
print(data.isnull().sum())
data = data.dropna(subset=['Latitude', 'Longitude'])

# Create output directory if it doesn't exist
output_dir = r'C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police\
if not os.path.exists(output_dir):
    os.makedirs(output_dir)

# Apply DBSCAN to find clusters of crimes by Crime type
for crime_type in data['Crime type'].unique():
    crime_data = data[data['Crime type'] == crime_type]
    coords = crime_data[['Latitude', 'Longitude']].to_numpy()

    # DBSCAN clustering
    db = DBSCAN(eps=0.01, min_samples=5).fit(coords)
    data.loc[data['Crime type'] == crime_type, 'cluster'] = db.labels_  # Directly updating the

    # Use the updated DataFrame for plotting
    crime_data_with_clusters = data[data['Crime type'] == crime_type]

    # Plot clusters
    plt.figure(figsize=(12, 8))
    plt.scatter(crime_data_with_clusters['Longitude'], crime_data_with_clusters['Latitude'],
                c=crime_data_with_clusters['cluster'], cmap='viridis', s=10)
    plt.title(f'Crime Clusters for {crime_type}')
    plt.xlabel('Longitude')
    plt.ylabel('Latitude')
    plt.colorbar(label='Cluster')

    # Save the plot
    filename = f"crime_clusters_{crime_type.replace(' ', '_')}.png"
    filepath = os.path.join(output_dir, filename)
    plt.savefig(filepath)
    plt.close()
```

```
    print(f"Saved cluster plot for {crime_type} at {filepath}")

print("All clustering operations completed.")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3149321 entries, 0 to 3149320
Data columns (total 12 columns):
 #   Column                 Dtype
---  ------                 -----
 0   Crime ID               object
 1   Month                  object
 2   Reported by            object
 3   Falls within           object
 4   Longitude              float64
 5   Latitude               float64
 6   Location               object
 7   LSOA code              object
 8   LSOA name              object
 9   Crime type             object
 10  Last outcome category  object
 11  Context                float64
dtypes: float64(3), object(9)
memory usage: 288.3+ MB
None
                                          Crime ID    Month  \
0  bcf33862673ea5cebb2d0814770e3147981dd6c6f88ad6...  2021-04
1  578df7143c18214677518d49cf3834f34016eefaa134d1...  2021-04
2                                              NaN  2021-04
3                                              NaN  2021-04
4                                              NaN  2021-04

                  Reported by                 Falls within  Longitude  \
0  Metropolitan Police Service  Metropolitan Police Service   0.867037
1  Metropolitan Police Service  Metropolitan Police Service  -0.590478
2  Metropolitan Police Service  Metropolitan Police Service   0.135866
3  Metropolitan Police Service  Metropolitan Police Service   0.140192
4  Metropolitan Police Service  Metropolitan Police Service   0.134947

    Latitude                      Location  LSOA code  \
0  51.141024  On or near Christchurch Road  E01024029
1  51.862951       On or near Church Croft  E01017658
2  51.587336    On or near Gibbfield Close  E01000027
3  51.582311        On or near Hatch Grove  E01000027
4  51.588063         On or near Mead Grove  E01000027

                  LSOA name                    Crime type  \
0              Ashford 005E  Violence and sexual offences
1       Aylesbury Vale 009D  Violence and sexual offences
2  Barking and Dagenham 001A         Anti-social behaviour
3  Barking and Dagenham 001A         Anti-social behaviour
4  Barking and Dagenham 001A         Anti-social behaviour

                          Last outcome category  Context
0  Investigation complete; no suspect identified      NaN
1                      Status update unavailable      NaN
2                                            NaN      NaN
3                                            NaN      NaN
4                                            NaN      NaN
Crime ID                  702265
Month                          0
Reported by                    0
Falls within                   0
Longitude                  64222
Latitude                   64222
Location                       0
LSOA code                  64222
LSOA name                  64222
```

```
Crime type                    0
Last outcome category     702265
Context                  3149321
year                           0
month_num                      0
dtype: int64
Saved cluster plot for Violence and sexual offences at C:\Users\jckat\OneDrive\Documents\GitHub
\Space_TIme_Exploration_of_Metro_Police\DBSCAN\crime_clusters_Violence_and_sexual_offences.png
Saved cluster plot for Anti-social behaviour at C:\Users\jckat\OneDrive\Documents\GitHub\Space_T
Ime_Exploration_of_Metro_Police\DBSCAN\crime_clusters_Anti-social_behaviour.png
Saved cluster plot for Burglary at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explorati
on_of_Metro_Police\DBSCAN\crime_clusters_Burglary.png
Saved cluster plot for Criminal damage and arson at C:\Users\jckat\OneDrive\Documents\GitHub\Spa
ce_TIme_Exploration_of_Metro_Police\DBSCAN\crime_clusters_Criminal_damage_and_arson.png
Saved cluster plot for Public order at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explo
ration_of_Metro_Police\DBSCAN\crime_clusters_Public_order.png
Saved cluster plot for Vehicle crime at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Expl
oration_of_Metro_Police\DBSCAN\crime_clusters_Vehicle_crime.png
Saved cluster plot for Other crime at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explor
ation_of_Metro_Police\DBSCAN\crime_clusters_Other_crime.png
Saved cluster plot for Drugs at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_
of_Metro_Police\DBSCAN\crime_clusters_Drugs.png
Saved cluster plot for Other theft at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explor
ation_of_Metro_Police\DBSCAN\crime_clusters_Other_theft.png
Saved cluster plot for Shoplifting at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Explor
ation_of_Metro_Police\DBSCAN\crime_clusters_Shoplifting.png
Saved cluster plot for Bicycle theft at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Expl
oration_of_Metro_Police\DBSCAN\crime_clusters_Bicycle_theft.png
Saved cluster plot for Robbery at C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploratio
n_of_Metro_Police\DBSCAN\crime_clusters_Robbery.png
Saved cluster plot for Possession of weapons at C:\Users\jckat\OneDrive\Documents\GitHub\Space_T
Ime_Exploration_of_Metro_Police\DBSCAN\crime_clusters_Possession_of_weapons.png
Saved cluster plot for Theft from the person at C:\Users\jckat\OneDrive\Documents\GitHub\Space_T
Ime_Exploration_of_Metro_Police\DBSCAN\crime_clusters_Theft_from_the_person.png
All clustering operations completed.
```

KNOX TEST

The code aims to detect spatiotemporal clustering of crime events using the Knox test. It preprocesses the data, samples it for practical execution, and performs the test to determine if there is significant clustering beyond what would be expected by chance. The results include observed and expected counts, the Knox statistic, and a p-value indicating the significance of the clustering.

In [67]:
```python
import pandas as pd
import numpy as np
from scipy.spatial import cKDTree
from scipy.stats import chi2

def knox_test(data, spatial_threshold, temporal_threshold):
    coords = data[['Latitude', 'Longitude']].to_numpy()
    # Convert times to days since a base time (e.g., the minimum date in your dataset)
    base_time = data['Month'].min()
    times = (data['Month'] - base_time).dt.days

    # Use cKDTree for efficient spatial proximity search
    tree = cKDTree(coords)
    spatial_pairs = list(tree.query_pairs(spatial_threshold))

    # Temporal proximity using a manual filter for pairs found in the spatial step
    close_in_time_count = sum(abs(times[i] - times[j]) < temporal_threshold for i, j in spatial_
    observed = close_in_time_count
```

```
    # Expected count based on the product of probabilities
    num_pairs = len(spatial_pairs)
    total_pairs = len(data) * (len(data) - 1) / 2
    prob_close_in_time = close_in_time_count / total_pairs
    expected = num_pairs * prob_close_in_time

    # Knox statistic calculation
    knox_stat = (observed - expected) ** 2 / expected if expected != 0 else float('inf')
    p_value = 1 - chi2.cdf(knox_stat, df=1)  # df=1 for simple hypothesis

    return observed, expected, knox_stat, p_value

# Load the CSV file into a DataFrame
file_path = r'C:\Users\jckat\Desktop\Space data\metro_police_crime.csv'
data = pd.read_csv(file_path)

# Convert the 'month' column to datetime format
data['Month'] = pd.to_datetime(data['Month'], format='%Y-%m')

# Perform the Knox test on a sample of data for feasibility
sampled_data = data.sample(frac=0.01).reset_index(drop=True)  # Reset index and drop the old in
observed, expected, knox_stat, p_value = knox_test(sampled_data, spatial_threshold=0.01, tempor

print(f"Knox Test: Observed={observed}, Expected={expected}, Statistic={knox_stat}, P-value={p_
```

```
Knox Test: Observed=73685, Expected=325.2683147227824, Statistic=16545264.29210452, P-value=0.0
```

The results of the Knox test are as follows:

- **Observed**: 73685
- **Expected**: 325.27
- **Statistic**: 16545264.29
- **P-value**: 0.0

Here's what these results mean:

Observed and Expected Counts

- **Observed**: This is the actual number of event pairs that are close in both space and time (within the given spatial and temporal thresholds). In this case, there are 73,685 such pairs.
- **Expected**: This is the expected number of event pairs that would be close in both space and time if the events were distributed randomly. The expected count is calculated based on the probability of two events being close in time, multiplied by the number of spatially close pairs. In this case, the expected number is approximately 325.27.

Knox Statistic

- The **Knox Statistic** measures the discrepancy between the observed and expected counts.
- A very high Knox statistic (16,545,264.29 in this case) indicates a large difference between the observed and expected counts.

P-value

- The **P-value** is the probability of observing a Knox statistic as extreme as (or more extreme than) the one calculated, assuming the null hypothesis (no clustering) is true.

- A p-value of 0.0 (which effectively means it is extremely close to 0) indicates that the observed clustering is highly statistically significant. In other words, it is very unlikely that the observed clustering occurred by chance.

Interpretation

- **Significant Clustering**: The large difference between the observed and expected counts, along with the extremely low p-value, suggests that there is significant spatiotemporal clustering of crime events in the dataset.
- **Implications**: This result implies that crimes are not randomly distributed in space and time. Instead, they tend to cluster, which could be due to various factors such as social, economic, or environmental conditions.

Summary

The Knox test results indicate a highly significant spatiotemporal clustering of crimes, with the observed number of close pairs vastly exceeding the expected number under a random distribution. The statistical evidence (p-value of 0.0) strongly suggests that this clustering is not due to random chance, highlighting patterns that might be valuable for further investigation by law enforcement or policy makers.

Logistics Regression

The code aims to build and evaluate a machine learning model for classifying crime types based on geographical and temporal data. It involves:

1. **Loading Data**: Reads crime data from a CSV file.
2. **Data Preprocessing**: Converts dates, extracts features, encodes the target variable, handles missing values, and scales features.
3. **Data Splitting**: Divides the data into training and test sets.
4. **Model Training**: Trains a logistic regression model on the training data.
5. **Prediction and Evaluation**: Predicts crime types on the test set and evaluates the model using metrics like precision, recall, and F1-score.
6. **Saving Results**: Exports the classification report to a text file for documentation.

The overall goal is to accurately classify crime types and evaluate the model's performance for potential use in understanding crime patterns.

In [76]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
import os

# Load the CSV file into a DataFrame
file_path = r'C:\Users\jckat\Desktop\Space data\metro_police_crime.csv'
data = pd.read_csv(file_path)

# Convert the 'month' column to datetime format and extract relevant features
```

```
data['Month'] = pd.to_datetime(data['Month'], format='%Y-%m')
data['Month_num'] = data['Month'].dt.month
data['year'] = data['Month'].dt.year
X = data[['Latitude', 'Longitude', 'Month_num', 'year']]
y = data['Crime type']

# Encode categorical target variable
y = pd.factorize(y)[0]

# Handle missing values
imputer = SimpleImputer(strategy='mean')
X = imputer.fit_transform(X)

# Feature scaling
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the Logistic Regression model
# Adjust class_weight to 'balanced' to handle class imbalance
model = LogisticRegression(max_iter=1000, class_weight='balanced')
model.fit(X_train, y_train)

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
report = classification_report(y_test, y_pred, zero_division=0)

# Print the classification report
print(report)

# Define the path to export the result
output_dir = r'C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_of_Metro_Police'
output_file = os.path.join(output_dir, 'classification_report.txt')

# Create directory if it doesn't exist
if not os.path.exists(output_dir):
    os.makedirs(output_dir)

# Export the classification report to a text file
with open(output_file, 'w') as file:
    file.write(report)

print(f"Classification report saved to {output_file}")
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	146796
1	0.25	0.05	0.08	141002
2	0.00	0.00	0.00	30783
3	0.05	0.01	0.02	31906
4	0.08	0.00	0.00	33096
5	0.11	0.02	0.03	60217
6	0.02	0.21	0.03	6078
7	0.04	0.15	0.07	22332
8	0.20	0.02	0.04	68609
9	0.05	0.21	0.09	25323
10	0.02	0.29	0.04	10705
11	0.02	0.03	0.03	16259
12	0.01	0.08	0.01	2829
13	0.08	0.31	0.12	33930
accuracy			0.05	629865
macro avg	0.07	0.10	0.04	629865
weighted avg	0.10	0.05	0.04	629865

```
Classification report saved to C:\Users\jckat\OneDrive\Documents\GitHub\Space_TIme_Exploration_o
f_Metro_Police\classification_report.txt
```

The classification report provides metrics for evaluating the performance of the logistic regression model in classifying different crime types.

Metrics Explained

- **Precision**: The proportion of true positive predictions among all positive predictions. Higher precision indicates fewer false positives.
- **Recall**: The proportion of true positive predictions among all actual positives. Higher recall indicates fewer false negatives.
- **F1-score**: The harmonic mean of precision and recall, providing a balance between the two.
- **Support**: The number of actual occurrences of the class in the test set.

Analysis of Results

Class-wise Performance

- **Class 0**: Precision, recall, and F1-score are all 0.00, indicating the model failed to identify any instances of this class correctly.
- **Class 1**: Precision is 0.25, recall is 0.05, and F1-score is 0.08. The model is somewhat precise when it predicts this class, but it rarely predicts it.
- **Class 2, 3, 4, and 5**: These classes have very low precision, recall, and F1-scores, indicating poor performance.
- **Class 6**: High recall (0.21) but low precision (0.02) and F1-score (0.03). The model predicts this class often, but with many false positives.
- **Class 7**: Similar to Class 6 with better recall (0.15) but low precision (0.04) and F1-score (0.07).
- **Class 8**: Better precision (0.20) but poor recall (0.02) and F1-score (0.04).
- **Class 9**: Recall is relatively high (0.21) but precision (0.05) and F1-score (0.09) are low.
- **Class 10**: High recall (0.29) but poor precision (0.02) and F1-score (0.04).
- **Class 11, 12**: Very low performance in all metrics.

- **Class 13**: Better performance with precision (0.08), recall (0.31), and F1-score (0.12), indicating some level of effectiveness in predicting this class.

Overall Performance

- **Accuracy**: 0.05, meaning the model correctly predicts the crime type 5% of the time.
- **Macro Average**: Average of precision, recall, and F1-score across all classes without considering class imbalance. Precision is 0.07, recall is 0.10, and F1-score is 0.04, indicating generally poor performance.
- **Weighted Average**: Average of precision, recall, and F1-score across all classes, weighted by the number of instances in each class. Precision is 0.10, recall is 0.05, and F1-score is 0.04, reflecting poor overall model performance due to class imbalance and misclassifications.

Summary

The model exhibits poor performance across most classes, with very low precision, recall, and F1-scores. It particularly struggles with high class imbalance, as seen from the large discrepancies in support values. Some classes like 6, 7, 9, 10, and 13 have relatively higher recall, indicating the model is more likely to predict these classes, but with low precision, leading to many false positives.

Random Forest classifier

This code builds a Random Forest classifier to predict crime types, trains it on the training data, and evaluates its performance on the test data. The classification report provides detailed metrics to understand the model's effectiveness in classifying different crime types.

In [73]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

# Random Forest Classifier
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

# Evaluation
print(classification_report(y_test, y_pred_rf))
```

	precision	recall	f1-score	support
0	0.35	0.44	0.39	146796
1	0.47	0.57	0.51	141002
2	0.16	0.11	0.13	30783
3	0.13	0.08	0.10	31906
4	0.13	0.09	0.10	33096
5	0.26	0.22	0.24	60217
6	0.17	0.12	0.14	6078
7	0.18	0.12	0.14	22332
8	0.26	0.23	0.24	68609
9	0.38	0.44	0.41	25323
10	0.11	0.06	0.08	10705
11	0.12	0.07	0.09	16259
12	0.04	0.02	0.03	2829
13	0.30	0.30	0.30	33930
accuracy			0.33	629865
macro avg	0.22	0.21	0.21	629865
weighted avg	0.31	0.33	0.32	629865

The classification report evaluates the performance of the Random Forest classifier on the test data, summarizing precision, recall, F1-score, and support for each class.

Key Metrics Explained:

- **Precision**: Measures the proportion of true positive predictions out of all positive predictions. Higher precision indicates fewer false positives.
- **Recall**: Measures the proportion of true positive predictions out of all actual positive instances. Higher recall indicates fewer false negatives.
- **F1-score**: The harmonic mean of precision and recall, providing a balance between them.
- **Support**: The number of actual occurrences of each class in the test set.

Overall Metrics:

- **Accuracy**: The proportion of correct predictions out of all predictions. In this case, an accuracy of 33% indicates that the model correctly predicts 33% of the instances.
- **Macro Average**: The unweighted average of precision, recall, and F1-score across all classes. It treats all classes equally, regardless of their frequency.
- **Weighted Average**: The average of precision, recall, and F1-score, weighted by the number of instances in each class. It gives more importance to classes with more instances.

Interpretation:

1. **Major Class Performance**: The model performs relatively better for the most frequent classes, likely due to having more data to learn from.
2. **Minor Class Performance**: The model struggles with less frequent classes, as indicated by lower precision, recall, and F1-scores. This is often due to fewer training examples for these classes.
3. **Class Imbalance**: Significant variation in support values indicates class imbalance, where some classes are much more frequent than others. This can lead to biased predictions favoring the majority classes.

4. **Overall Performance**: With an accuracy of 33% and weighted averages for precision, recall, and F1-score around 0.31 to 0.33, the model's performance is moderate, showing room for improvement, particularly in handling class imbalance and improving predictions for less frequent classes.

In [ ]: