analysis_nyc_type1

April 23, 2023

1 Analysis Template

1.1 Preprocess

[ ]:

```
# resolve dependency
!pip install pmdarima
```

```
Requirement already satisfied: pmdarima in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (2.0.3)
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(65.6.3)
Requirement already satisfied: joblib>=0.11 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(1.1.1)
Requirement already satisfied: statsmodels>=0.13.2 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(0.13.5)
Requirement already satisfied: scikit-learn>=0.22 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(1.0.2)
Requirement already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(0.29.33)
Requirement already satisfied: urllib3 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(1.26.14)
Requirement already satisfied: numpy>=1.21.2 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(1.22.4)
Requirement already satisfied: pandas>=0.19 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(1.5.3)
Requirement already satisfied: scipy>=1.3.2 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from pmdarima)
(1.7.1)
Requirement already satisfied: python-dateutil>=2.8.1 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from
pandas>=0.19->pmdarima) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from
pandas>=0.19->pmdarima) (2022.7)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from scikit-
learn>=0.22->pmdarima) (2.2.0)
Requirement already satisfied: packaging>=21.3 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from
statsmodels>=0.13.2->pmdarima) (23.0)
Requirement already satisfied: patsy>=0.5.2 in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from
statsmodels>=0.13.2->pmdarima) (0.5.3)
Requirement already satisfied: six in
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-packages (from
patsy>=0.5.2->statsmodels>=0.13.2->pmdarima) (1.16.0)
```

[ ]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
from pandas.plotting import autocorrelation_plot
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
import statsmodels.api as sm
from pmdarima.arima import ADFTest , auto_arima
%matplotlib inline
```

[ ]:

```
data_path = "../data/nypd_assault.csv"
crime = "type1"
target = "count"
date = "date"
city = "nyc"
fig_size = (20,5)
```

[ ]:

```
df_by_day = pd.read_csv(data_path)
df_by_day[date] = pd.to_datetime(df_by_day[date])
df_by_day.set_index(date, inplace=True)
```

1.2 Profiling

1.2.1 By day

[ ]:

```
df_by_day.head()
```

[ ]:

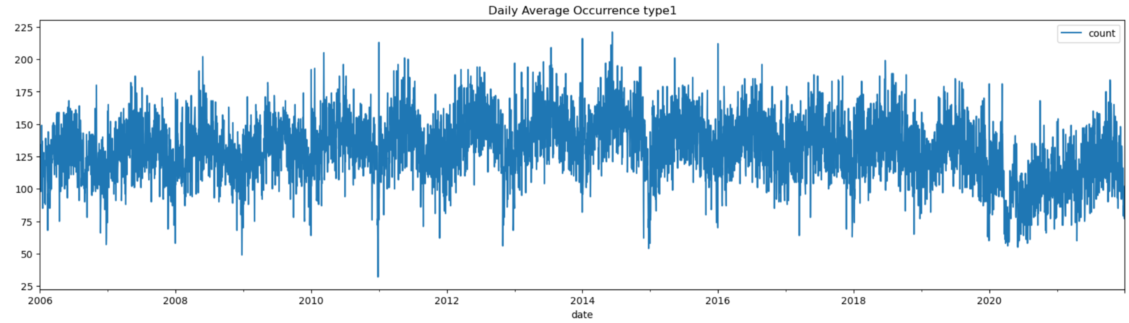
```
            count
date
2006-01-01    107
```

```
2006-01-02    105
2006-01-03    104
2006-01-04    118
2006-01-05    133
```

[ ]:
```
df_by_day.describe()
```

[ ]:
```
             count
count  5844.000000
mean    131.463895
std      24.073300
min      32.000000
25%     116.000000
50%     132.000000
75%     148.000000
max     221.000000
```

[ ]:
```
df_by_day.plot(figsize=fig_size, title="Daily Average Occurrence " + crime)
plt.show()
```



[ ]:
```
df_by_day[target].sort_values(ascending=False).head()
```

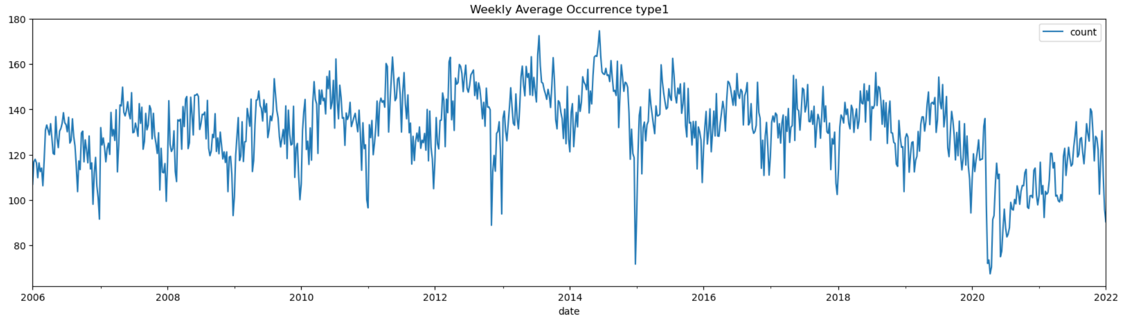
[ ]:
```
date
2014-06-11    221
2014-01-01    216
2011-01-01    213
2016-01-01    212
2014-06-04    211
Name: count, dtype: int64
```

1.2.2 By week

[ ]:
```
df_by_week = pd.DataFrame(df_by_day[target].resample('W').mean())
```

[ ]:

```
df_by_week.plot(
    figsize=fig_size,
    title="Weekly Average Occurrence " + crime)
plt.show()
```



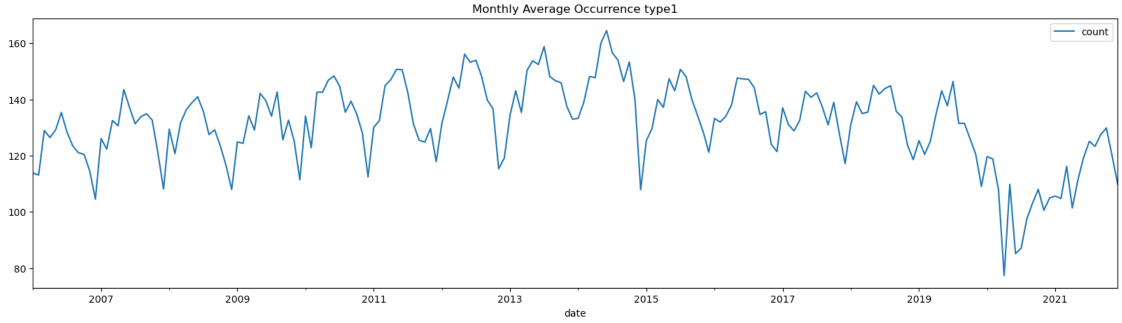
1.2.3 By month

[ ]:

```
df_by_month = pd.DataFrame(df_by_day[target].resample('M').mean())
```

[ ]:

```
df_by_month.plot(
    figsize=fig_size,
    title="Monthly Average Occurrence " + crime)
plt.show()
```



1.3 Analysis

[ ]:

```
#Ho: It is non stationary
#H1: It is stationary

def adfuller_test(count):
    result=adfuller(count)
    labels = ['ADF Test Statistic','p-value','#Lags Used','Number of␣
↪Observations Used']
    for value,label in zip(result,labels):
        print(label+' : '+str(value) )
    if result[1] <= 0.05:
        print("strong evidence against the null hypothesis(Ho), reject the null␣
↪hypothesis. Data has no unit root and is stationary")
    else:
        print("weak evidence against null hypothesis, time series has a unit␣
↪root, indicating it is non-stationary ")
```

1.3.1 Checking stationary

[ ]:

```
adfuller_test(df_by_month[target])
```

```
ADF Test Statistic : -1.8558505343444571
p-value : 0.3531067118505653
#Lags Used : 14
Number of Observations Used : 177
weak evidence against null hypothesis, time series has a unit root, indicating
it is non-stationary
```

1.3.2 Checking seasonality

[ ]:

```
df_by_month['seasonal_first_difference'] = df_by_month[target] -␣
 ↪df_by_month[target].shift(12)
```

[ ]:

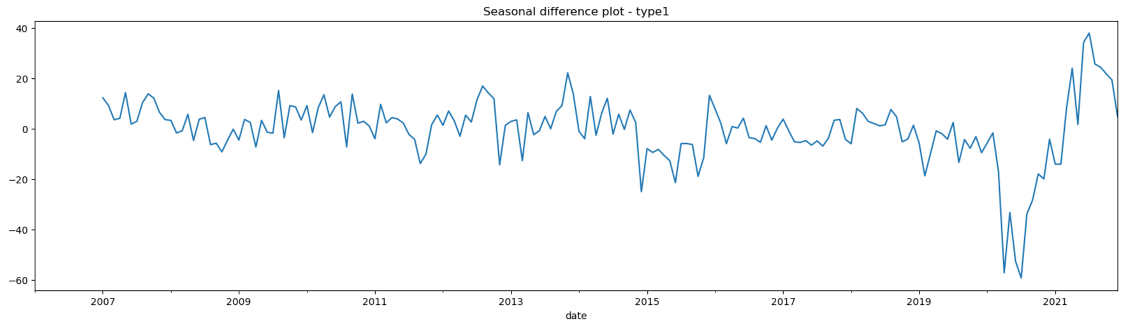
```
adfuller_test(df_by_month['seasonal_first_difference'].dropna())
```

```
ADF Test Statistic : -3.516023909094369
p-value : 0.007584977307881228
#Lags Used : 12
Number of Observations Used : 167
strong evidence against the null hypothesis(Ho), reject the null hypothesis.
Data has no unit root and is stationary
```

[ ]:

```
df_by_month['seasonal_first_difference'].plot(figsize=fig_size, title='Seasonal␣
 ↪difference plot - ' + crime)
```

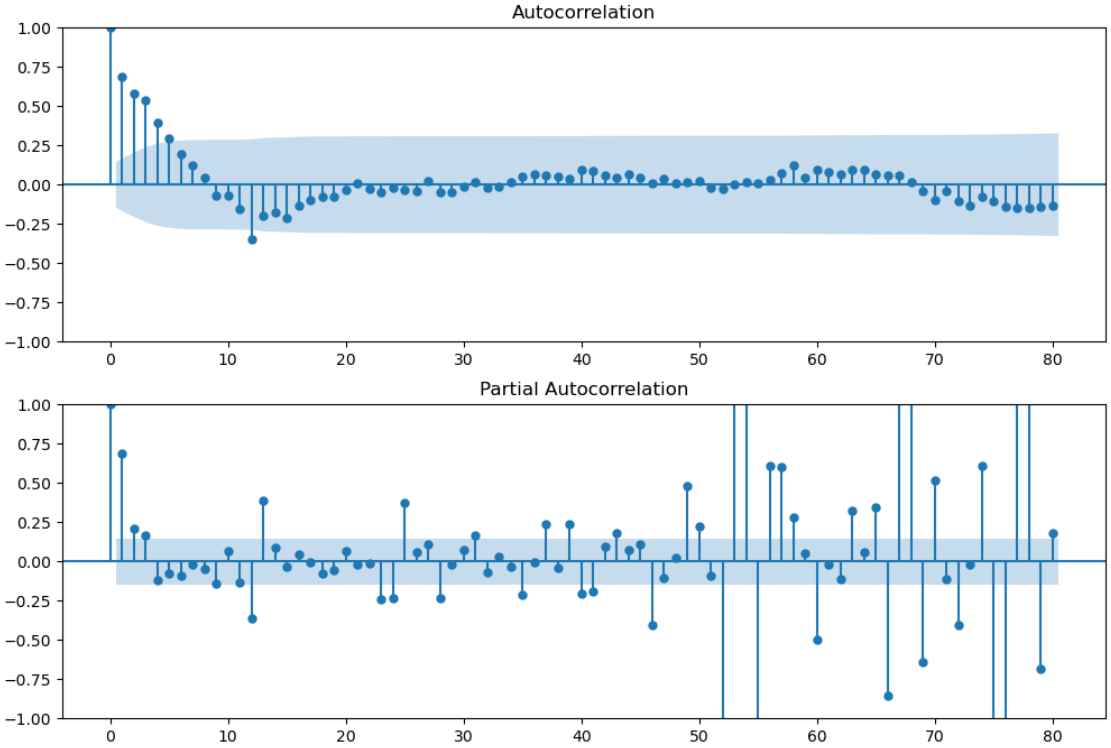
[ ]: <Axes: title={'center': 'Seasonal difference plot - type1'}, xlabel='date'>



1.3.3 Auto Regressive Model

```
[ ]: fig = plt.figure(figsize=(12,8))
     ax1 = fig.add_subplot(211)
     fig = sm.graphics.tsa.plot_acf(df_by_month['seasonal_first_difference'].iloc[13:
      ↪],lags=80,ax=ax1)
     ax2 = fig.add_subplot(212)
     fig = sm.graphics.tsa.plot_pacf(df_by_month['seasonal_first_difference'].
      ↪iloc[13:],lags=80,ax=ax2)
```

```
/Users/xuyanchong/opt/anaconda3/lib/python3.9/site-
packages/statsmodels/graphics/tsaplots.py:348: FutureWarning: The default method
'yw' can produce PACF values outside of the [-1,1] interval. After 0.13, the
default will change tounadjusted Yule-Walker ('ywm'). You can use this method
now by setting method='ywm'.
  warnings.warn(
```

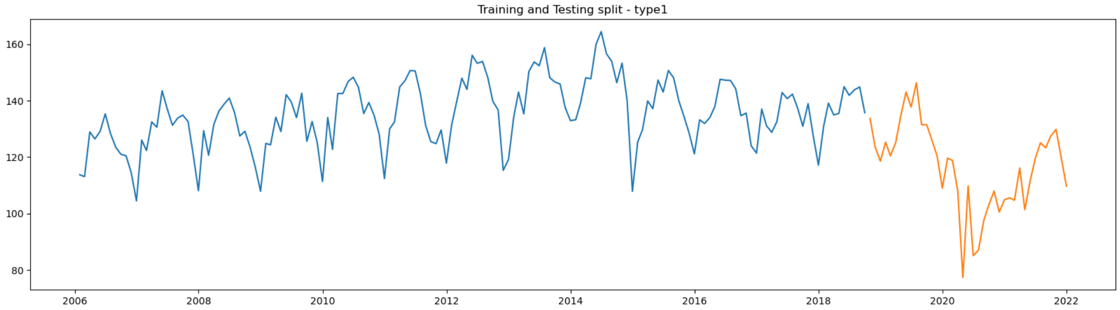


1.3.4 Implementing Seasonal Arima Model

[ ]:
```
adf_test=ADFTest(alpha=0.05)
adf_test.should_diff(df_by_month[target])
```

[ ]:
```
(0.01, False)
```

[ ]:
```
start=int(df_by_month.shape[0]*0.8)
train=df_by_month[:start]
test=df_by_month[start:]
plt.figure(figsize=fig_size)
plt.plot(train[target])
plt.plot(test[target])
plt.title('Training and Testing split - '+ crime)
plt.show()
```



```
[ ]: model=auto_arima(train[target],start_p=0,d=1,start_q=0,
               max_p=10,max_d=10,max_q=10, start_P=0,
               D=1, start_Q=0, max_P=10,max_D=10,
               max_Q=10, m=12, seasonal=True,
               error_action='warn',trace=True,
               supress_warnings=True,stepwise=True,
               random_state=20,n_fits=50)
```

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,1,0)[12]             : AIC=1001.756, Time=0.02 sec
 ARIMA(1,1,0)(1,1,0)[12]             : AIC=939.636, Time=0.16 sec
 ARIMA(0,1,1)(0,1,1)[12]             : AIC=895.705, Time=0.33 sec
 ARIMA(0,1,1)(0,1,0)[12]             : AIC=956.026, Time=0.05 sec
 ARIMA(0,1,1)(1,1,1)[12]             : AIC=897.560, Time=0.59 sec
 ARIMA(0,1,1)(0,1,2)[12]             : AIC=897.538, Time=2.41 sec
 ARIMA(0,1,1)(1,1,0)[12]             : AIC=922.332, Time=0.14 sec
 ARIMA(0,1,1)(1,1,2)[12]             : AIC=899.294, Time=8.19 sec
 ARIMA(0,1,0)(0,1,1)[12]             : AIC=928.833, Time=0.32 sec
 ARIMA(1,1,1)(0,1,1)[12]             : AIC=894.280, Time=0.93 sec
 ARIMA(1,1,1)(0,1,0)[12]             : AIC=956.773, Time=0.19 sec
 ARIMA(1,1,1)(1,1,1)[12]             : AIC=895.795, Time=1.74 sec
 ARIMA(1,1,1)(0,1,2)[12]             : AIC=895.739, Time=21.69 sec
 ARIMA(1,1,1)(1,1,0)[12]             : AIC=920.270, Time=1.98 sec
 ARIMA(1,1,1)(1,1,2)[12]             : AIC=inf, Time=62.84 sec
 ARIMA(1,1,0)(0,1,1)[12]             : AIC=906.736, Time=2.30 sec
 ARIMA(2,1,1)(0,1,1)[12]             : AIC=894.901, Time=3.52 sec
 ARIMA(1,1,2)(0,1,1)[12]             : AIC=894.978, Time=3.57 sec
 ARIMA(0,1,2)(0,1,1)[12]             : AIC=895.509, Time=3.01 sec
 ARIMA(2,1,0)(0,1,1)[12]             : AIC=903.013, Time=2.43 sec
 ARIMA(2,1,2)(0,1,1)[12]             : AIC=896.866, Time=4.01 sec
 ARIMA(1,1,1)(0,1,1)[12] intercept   : AIC=894.087, Time=2.32 sec
 ARIMA(1,1,1)(0,1,0)[12] intercept   : AIC=958.715, Time=0.09 sec
 ARIMA(1,1,1)(1,1,1)[12] intercept   : AIC=895.619, Time=2.64 sec
 ARIMA(1,1,1)(0,1,2)[12] intercept   : AIC=895.561, Time=28.71 sec
 ARIMA(1,1,1)(1,1,0)[12] intercept   : AIC=922.041, Time=2.32 sec
 ARIMA(1,1,1)(1,1,2)[12] intercept   : AIC=897.077, Time=54.58 sec
 ARIMA(0,1,1)(0,1,1)[12] intercept   : AIC=897.075, Time=0.67 sec
 ARIMA(1,1,0)(0,1,1)[12] intercept   : AIC=908.535, Time=0.61 sec
 ARIMA(2,1,1)(0,1,1)[12] intercept   : AIC=inf, Time=1.55 sec
 ARIMA(1,1,2)(0,1,1)[12] intercept   : AIC=inf, Time=1.56 sec
 ARIMA(0,1,0)(0,1,1)[12] intercept   : AIC=930.742, Time=0.46 sec
 ARIMA(0,1,2)(0,1,1)[12] intercept   : AIC=896.221, Time=0.78 sec
 ARIMA(2,1,0)(0,1,1)[12] intercept   : AIC=904.748, Time=0.72 sec
 ARIMA(2,1,2)(0,1,1)[12] intercept   : AIC=894.755, Time=1.49 sec

Best model:  ARIMA(1,1,1)(0,1,1)[12] intercept
Total fit time: 218.919 seconds
```

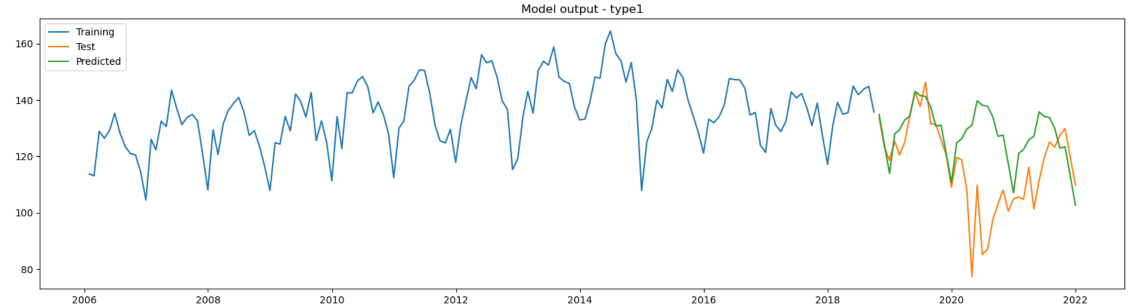
```
[ ]: model.summary()
```

```
[ ]: <class 'statsmodels.iolib.summary.Summary'>
"""
                                     SARIMAX Results
================================================================================
==========
Dep. Variable:                                  y   No. Observations:
153
Model:             SARIMAX(1, 1, 1)x(0, 1, 1, 12)   Log Likelihood
-442.043
Date:                            Sun, 23 Apr 2023   AIC
894.087
Time:                                    01:34:52   BIC
908.795
Sample:                                01-31-2006   HQIC
900.064
                                     - 09-30-2018
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -0.0330      0.020     -1.658      0.097      -0.072       0.006
ar.L1          0.3398      0.095      3.586      0.000       0.154       0.526
ma.L1         -0.8941      0.056    -15.964      0.000      -1.004      -0.784
ma.S.L12      -0.8122      0.085     -9.562      0.000      -0.979      -0.646
sigma2        29.2120      3.730      7.831      0.000      21.901      36.523
================================================================================
===
Ljung-Box (L1) (Q):                   0.21   Jarque-Bera (JB):
20.56
Prob(Q):                              0.65   Prob(JB):
0.00
Heteroskedasticity (H):               1.55   Skew:
-0.59
```

```
Prob(H) (two-sided):                  0.14   Kurtosis:
4.47
================================================================================
===

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-
step).
"""
```

```
[ ]: prediction = pd.DataFrame(model.predict(n_periods = train.shape[0]),index=test.
      ↪index)
     prediction.columns = ['predicted_crime']
     plt.figure(figsize=fig_size)
     plt.plot(train[target],label="Training")
     plt.plot(test[target],label="Test")
     plt.plot(prediction,label="Predicted")
     plt.legend(loc = 'upper left')
     plt.savefig('../output/%s_%s_pred.jpg' % (city,crime))
     plt.title('Model output - '+crime)
     plt.show()
```



```
[ ]: np.sqrt(np.square(np.subtract(test[target].values,prediction['predicted_crime'].
      ↪values)).mean())
```

```
[ ]: 19.612174106151354
```