Chicago Crime - Time Series Analysis

We'll conduct a time series analysis on the original Chicago Crime Dataset. We won't worry about balancing here, since the goal is to visualize the data through time, not prediction. First, load the data:

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from datetime import datetime
```

In [2]:
```
df = pd.read_csv("C:\\Users\\mbadi\\Desktop\\Summer_Research\\Crimes_-_2001_to_present.csv")
```

In [3]:
```
df.shape
```

Out[3]:
```
(6868001, 30)
```

We'll want to consider the range of our time stamps.

In [4]:
```
df["Date"].min(), df["Date"].max()
```

Out[4]:
```
('01/01/2001 01:00:00 AM', '12/31/2018 12:55:00 AM')
```

We're working with a 17 year range of time.

Let's consider the proportion of unique time-stamps to the total number of time-stamps.

In [5]:
```
len(set(df["Date"]))/len(df["Date"])
```

Out[5]:
```
0.40165617331738884
```

Now let's verify whether there are missing values:

In [6]:
```
df["Date"].isnull().sum()
```

Out[6]:
```
0
```

No missing date values, which means that a large number of time-stamps repeat.

A good starting metric might be the rate of arrests per month. This can be compared to, say, the rate of crimes per month that involved narcotics or of those that occured on the sidewalk. Previous analysis indicates that these factors are all positively correlated, so now we can explore how their relationship changes through time.

Let's whittle down the dataset to those columns that we're interested in.

In [7]:
```
cols_to_keep = ["Date", "Primary Type", "Location Description", "Arrest"]

crime_type = ["THEFT","BATTERY","NARCOTICS","CRIMINAL DAMAGE","ASSAULT","OTHER
OFFENSE","CRIMINAL TRESPASS",
              "BURGLARY","MOTOR VEHICLE THEFT","DECEPTIVE PRACTICE","ROBBERY",
"PROSTITUTION",
              "WEAPONS VIOLATION"]

crime_loc = ["STREET","RESIDENCE","APARTMENT","SIDEWALK","OTHER","PARKING LOT/
GARAGE(NON.RESID.)","ALLEY",
             "SCHOOL, PUBLIC, BUILDING","RESIDENCE-GARAGE","SMALL RETAIL STOR
E","RESIDENCE PORCH/HALLWAY",
             "VEHICLE NON-COMMERCIAL","RESTAURANT"]
```

In [8]:
```
# df = df.loc[df["Primary Type"].isin(crime_type) &
#             df["Location Description"].isin(crime_loc)]

df = df[cols_to_keep]
df.isnull().sum()
```

Out[8]:
```
Date                       0
Primary Type               0
Location Description    4958
Arrest                     0
dtype: int64
```

Only Location Description has missing values. We can simply drop those rows, since they are so few. But first, we'll create a new "Date" column, the entries of which are DateTime values and not simply strings as in the original "Date" column. The DateTime datatype is much easier to work with when we attempt to group instances by month. The following code blocks create the new DateTime column and also add dummy variables to the dataset. This will now let us group the data by month, and display the sum of each category of crime for each month.

In [9]:
```
time_stamps = np.array(df["Date"])
```

In [10]:
```
dates = np.array([time_stamps[i][:10] for i in range(len(time_stamps))])
```

In [11]:
```
dates = [datetime.strptime(date, '%m/%d/%Y') for date in dates]
```

In [12]:
```
df["DateTime"] = pd.Series(dates)
```

In [13]:
```
# df.dropna(axis=0, how='any', inplace=True)
```

In [14]:
```
dummies = pd.get_dummies(df[["Primary Type", "Location Description"]])
```

In [15]:
```
df = df.join(dummies)
```

In [16]:
```
df.drop(["Primary Type", "Location Description"], axis=1, inplace=True)
```

In [17]:
```
del dummies
```

We've set up our dataset the way we intended. Now let's group the instances by month.

In [18]:
```
df["DateTimeIndex"] = pd.DatetimeIndex(df["DateTime"])
```

In [19]:
```
df.set_index(df["DateTimeIndex"], inplace=True)
```

In [20]:
```
formatted_df = df.drop(["Date", "DateTime"], axis=1, inplace=False).copy()
del df
```

In [21]:
```
crime_type_cols = ["Primary Type_" + cols for cols in crime_type]
crime_loc_cols = ["Location Description_" + cols for cols in crime_loc]
top_26_cols = crime_type_cols + crime_loc_cols + ["Arrest"]
```

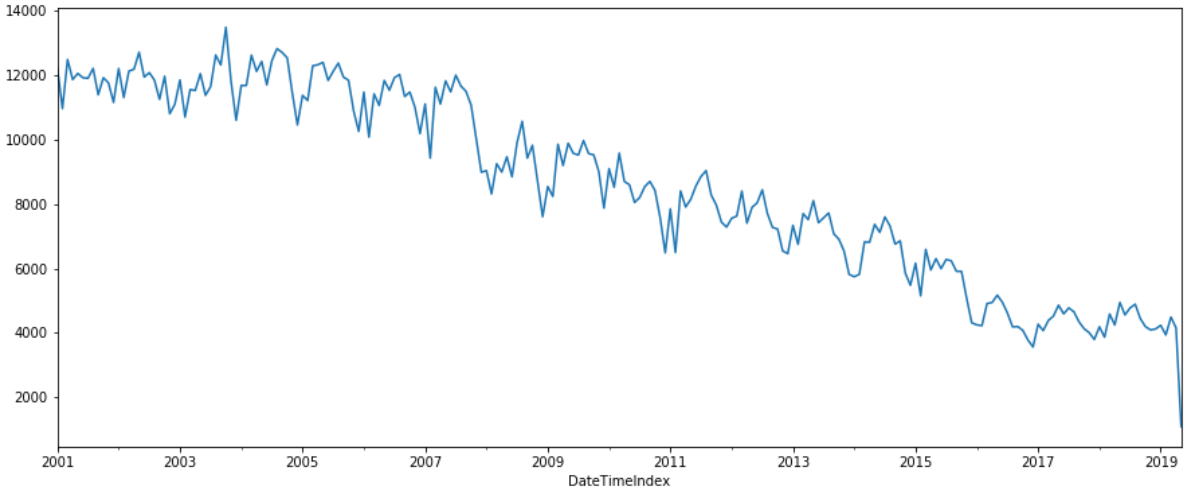
In [22]:
```
formatted_df = formatted_df[top_26_cols].copy()
```

In [23]:
```
formatted_df = formatted_df.resample('M').sum()
```

Now all the data has been collapsed into individual months, the values of which are the sums of all the values that occured in that month. Let's proceed to visualize some trends.

In [24]:
```
formatted_df["Arrest"].plot(figsize=(15,6))
```

Out[24]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25bb19e55f8>
```



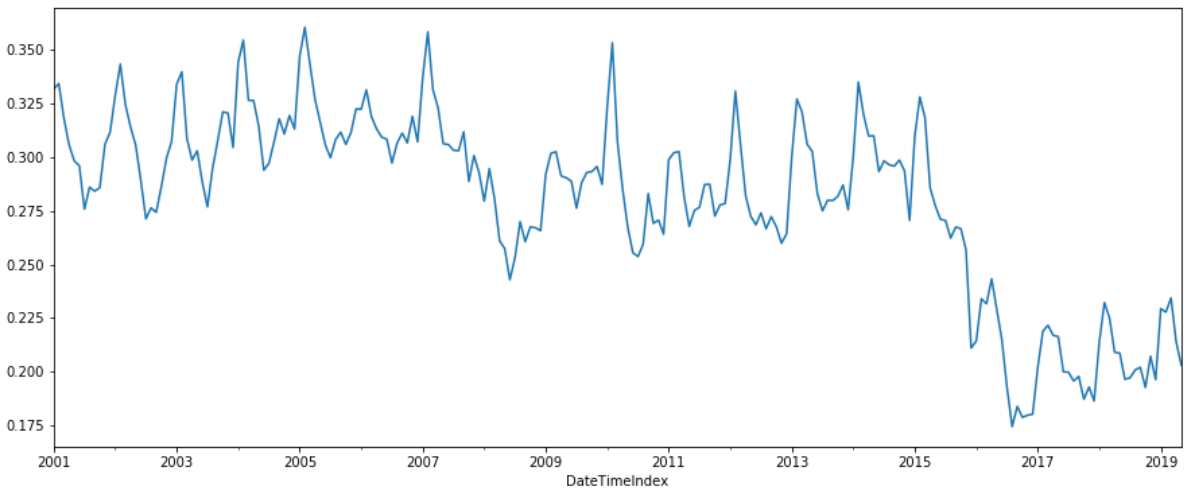
The number of arrests per month has a downward trend. But this isn't surprising, since the source website indicates a decrease in overall crime as well. A more telling metric might be the ratio of # arrests to # crimes.

In [25]:
```
arrest_proportion = [formatted_df["Arrest"][i]/formatted_df.iloc[i,:13].sum()
                     for i in range(formatted_df.shape[0])]
arrest_proportion = pd.DataFrame(arrest_proportion, index=formatted_df.index,
columns=["Arrest Rate"])
```

In [26]:
```
formatted_df = formatted_df.join(arrest_proportion)
```

In [27]:
```
formatted_df["Arrest Rate"].plot(figsize=(15,6))
```

Out[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25bf58e85f8>
```



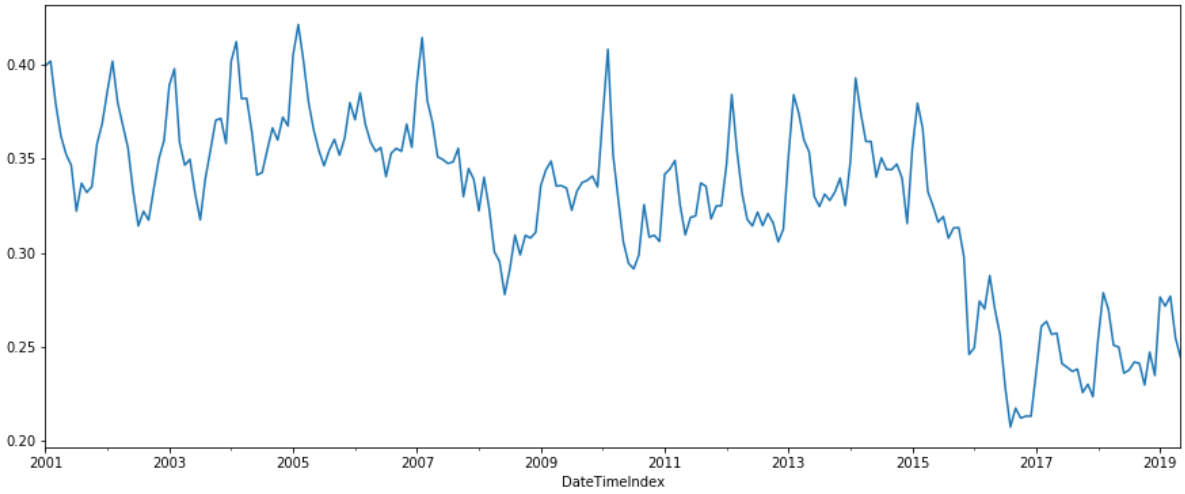
The graph above charts the ratio of arrests to the sum of all the crime types for each time stamp. There's a noticable seasonal trend, which is expected. Observe the sharp drop in the arrest ratio over the course of 2016. There's a less sharp temporary decrease in the ratio sometime in 2008.

In [28]:
```
arrest_by_loc = [formatted_df["Arrest"][i]/formatted_df.iloc[i, 13:26].sum()
                 for i in range(formatted_df.shape[0])]
arrest_by_loc = pd.DataFrame(arrest_by_loc, index=formatted_df.index, columns=
["Arrest Rate by Location"])
```

In [29]:
```
formatted_df = formatted_df.join(arrest_by_loc)
```

In [30]:
```
formatted_df["Arrest Rate by Location"].plot(figsize=(15,6))
```

Out[30]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25bb3b8ad68>
```



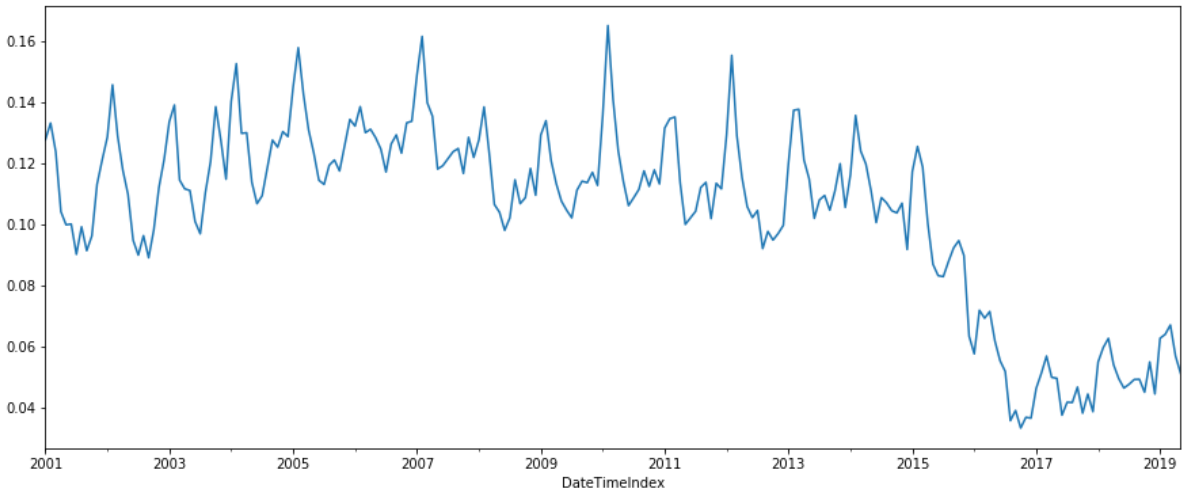
^This graph charts the arrest ratio with respect to the sum of all crime locations for each timespamps. I did this simply because the sum in the previous chart and this one did not agree. Regardless, we see a near-identical trend. Let's look at the trend of the proportion of crimes involving narcotics.

In [31]:
```
narcotics_proportion = [formatted_df["Primary Type_NARCOTICS"][i]/formatted_df
.iloc[i,:13].sum()
                        for i in range(formatted_df.shape[0])]
narcotics_proportion = pd.DataFrame(narcotics_proportion, index=formatted_df.i
ndex, columns=["Narcotics Rate"])
```

In [32]:
```
formatted_df = formatted_df.join(narcotics_proportion)
```

In [33]:
```
formatted_df["Narcotics Rate"].plot(figsize=(15,6))
```

Out[33]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25c26799ef0>
```



^We have the same sharp decline in the proportion of narcotics crimes as with the rate of arrests around 2016. It was shown in a previous Notebook that arrests most strongly correlate with narcotics.

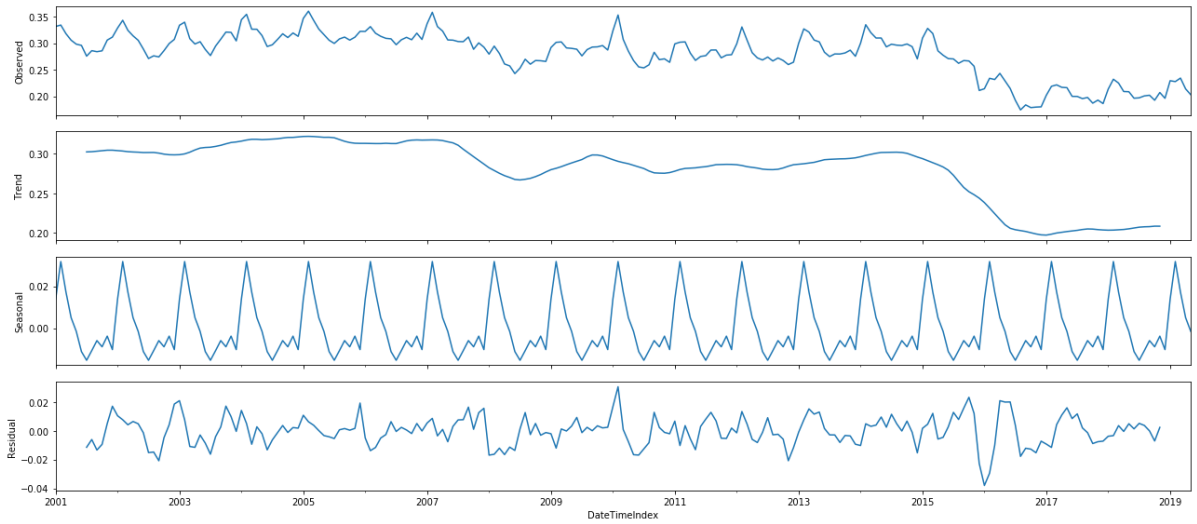
Time Series Decomposition

Let's see if we can't visualize our variables above by decomposing them into trend, seasonality, and noise. We'll start with the rate of arrests with respect to crime type.

In [34]:
```
import statsmodels.api as sm
from pylab import rcParams
rcParams['figure.figsize'] = 18, 8
```

In [35]:
```
decomposition1 = sm.tsa.seasonal_decompose(
    pd.DataFrame(formatted_df["Arrest Rate"], index=formatted_df.index).copy
(),
    model='additive')
```

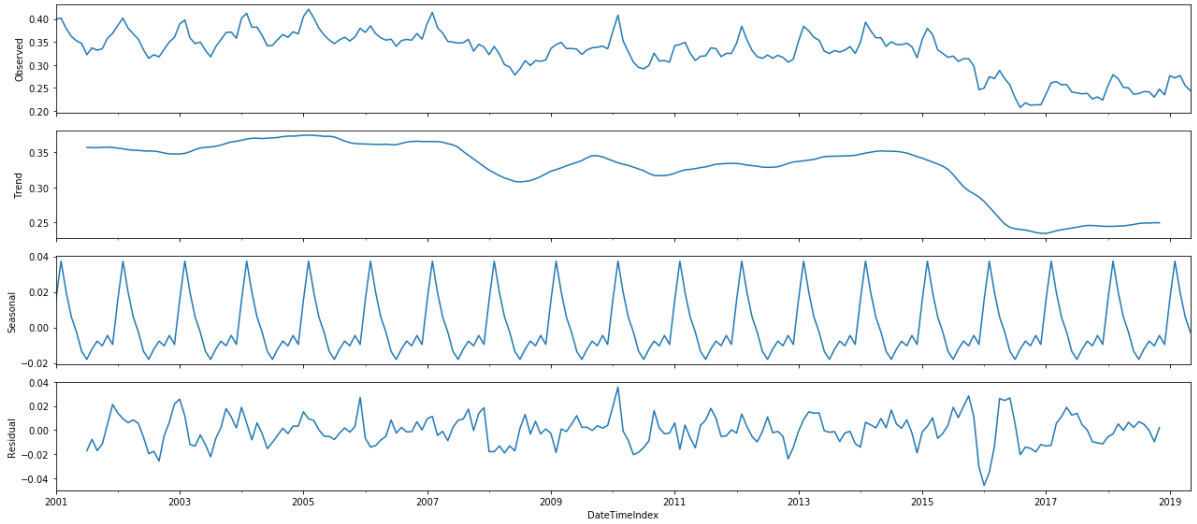
In [36]:
```
fig = decomposition1.plot()
plt.show()
```



Let's verify the consistency of the above trend by plotting that of arrest rate by location.

In [37]:
```
decomposition2 = sm.tsa.seasonal_decompose(
    pd.DataFrame(formatted_df["Arrest Rate by Location"], index=formatted_df.i
ndex).copy(),
    model='additive')
```

In [38]:
```
fig = decomposition2.plot()
plt.show()
```

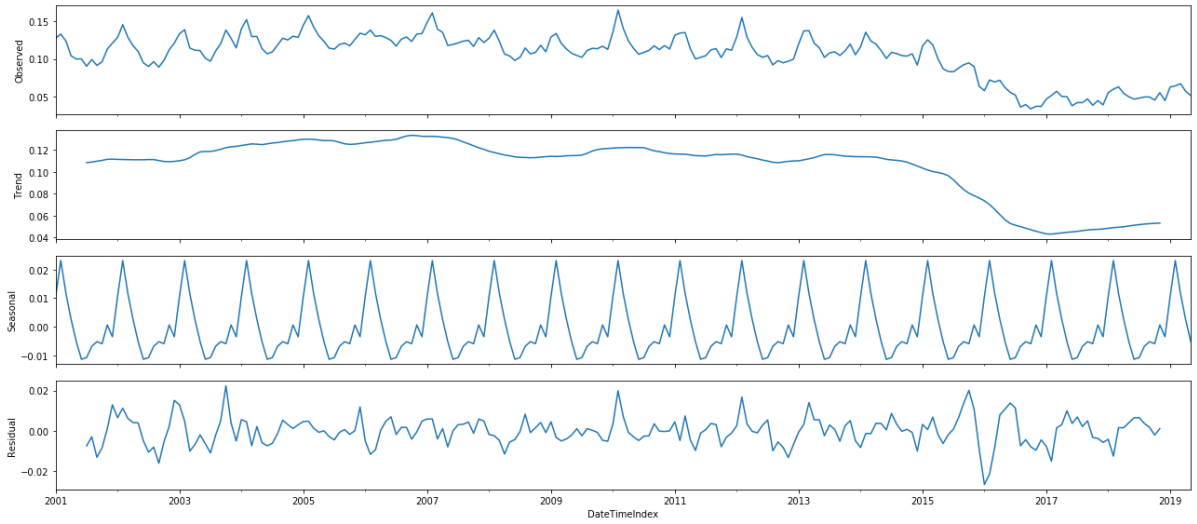


The y-axis values are a bit off, but the trends are virtually indistinguishable. Both trends show a drastic drop in the arrest rate in 2016, and both trends are reliably cyclical with respect to the seasons.

Finally, let's look at the rate of crimes pertaining to narcotics.

In [39]:
```
decomposition3 = sm.tsa.seasonal_decompose(
    pd.DataFrame(formatted_df["Narcotics Rate"], index=formatted_df.index).copy(),
    model='additive')
```

In [40]:
```
fig = decomposition3.plot()
plt.show()
```



It might be worth exploring what about 2016 caused the marked drops in the arrest rate. I suspect it was the drop in the rate of crimes pertaining to narcotics, which is the crime type that sees the most arrests.

Works Cited

Some of the source code adapted from this site was quite useful: [https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b (https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b)](https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b)