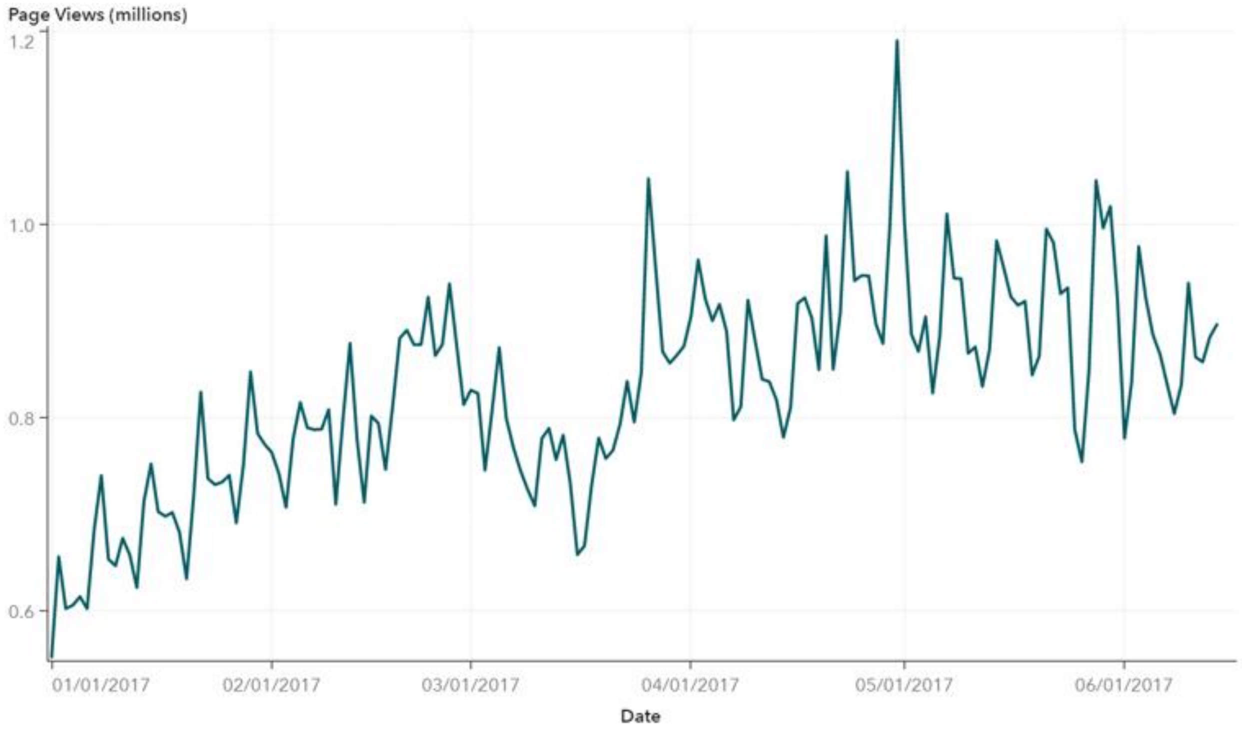
TIME SERIES MINING

Time Series

- A time series is an ordered sequence of *n* real-valued observations

$$T = (t_1, t_2, \dots, t_n), t_i \in \mathbb{R}$$



Time Series

- Many time series are recorded at very frequent time scales
- Stock market data
  - Ticker-level
- Purchases online
  - Real time
- What granularity to make forecasts
  - What is required?
  - What level of noise?
- Grocery store: hourly vs. daily

Mining Time Series

- Descriptive / Time series analysis
  - Model time series to determine its seasonal patterns, trends, relations to external factors

- Predictive / Time series forecasting
  - Use information in a time series to forecast future values of that series
  - Two types of forecasting
    - Linear regression – user specifies model and estimates time series
    - Smoothing – learns patterns from data

Forecasting Single vs Multiple Time Series

- Single
  - Blood pressure
  - Stock market prices
- Multiple
  - Temperature, precipitation, wind speed
  - EKG (Brain waves)
- Typically each series is modeled individually

Predictive Modeling

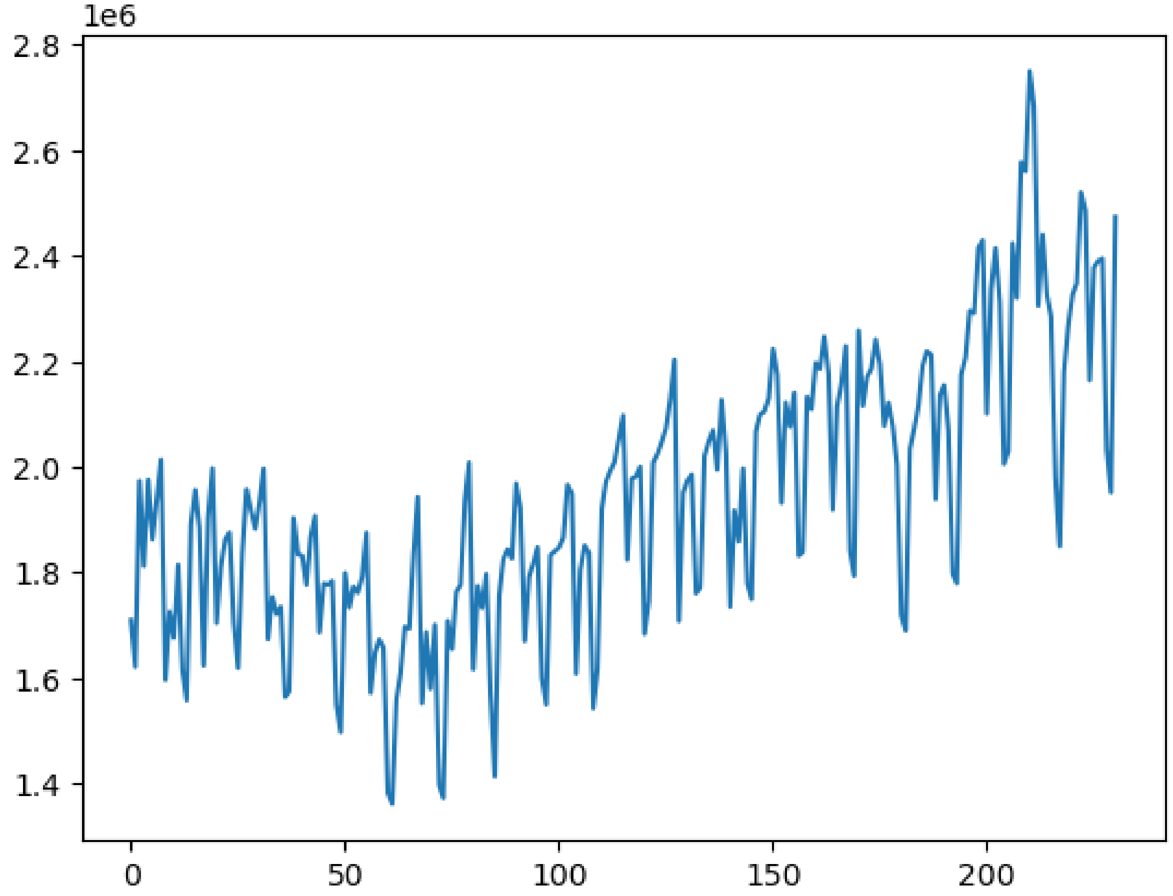
- Four components:
  - Level
    - Average of the time series
  - Trend
    - Change in one period to the next
  - Seasonality
    - Short-term cyclical behavior
  - Noise
    - Random variation from measurement error or other causes not accounted for

Amtrak Data

- Download the Amtrak data (rail_amtrack_ridership)
- Open a Jupyter Notebook

```
import pandas
import numpy as np
from matplotlib import pyplot
data = np.loadtxt('rail_amtrak_ridership.csv',delimiter = ',',skiprows = 1,usecols = 1)
pyplot.plot(data)
```

Results



Basic Model

- Model time series using linear regression
  - Given a time series Y
- Assume observations are at fixed intervals
  - Assign numeric value to those observation $t \in T = \{1, 2, \dots, N\}$

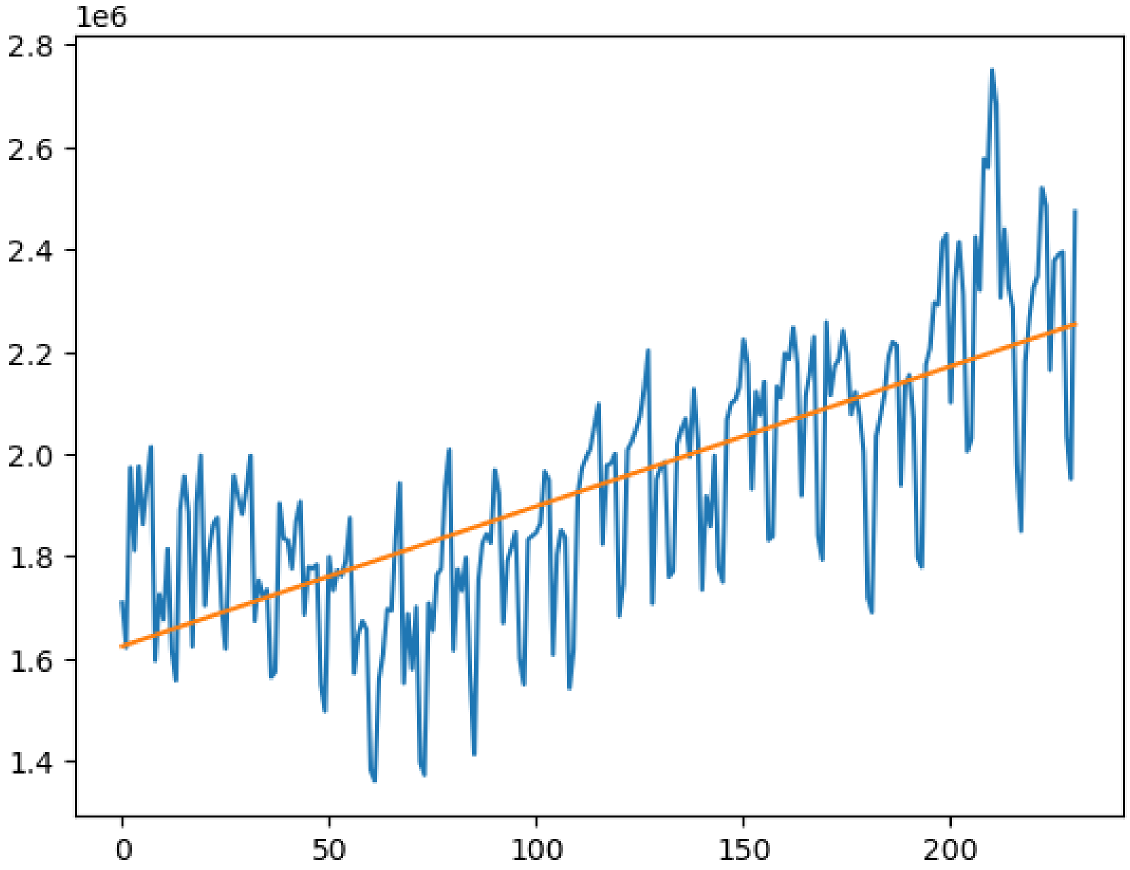
$$y_t = \beta_0 + \beta_1 t + \epsilon$$

- β_0 = Level
- β_1 = Trend
- ϵ = Noise

Basic Model on Amtrak Data

```
N = data.shape[0]
t = np.array(list(range(1,N+1)))
from sklearn.linear_model import LinearRegression
y = data.reshape((N,1))
t = t.reshape((N,1))
# create basic model
reg = LinearRegression()
reg.fit(t,y)
y_pred = reg.predict(t)
pyplot.plot(data)
pyplot.plot(y_pred)
```

Results



Basic Model on Amtrak Data

What is the level (β_0)?

reg.intercept_

```
reg.intercept_
```

```
array([1620967.66681724])
```

What is the trend (β_1)?

reg.coef_

```
reg.coef_
```

```
array([[2738.11837458]])
```

What is the noise (ϵ)?

SSE

np.linalg.norm(y-y_pred,ord=2)

```
np.linalg.norm(y-y_pred, ord=2)
```

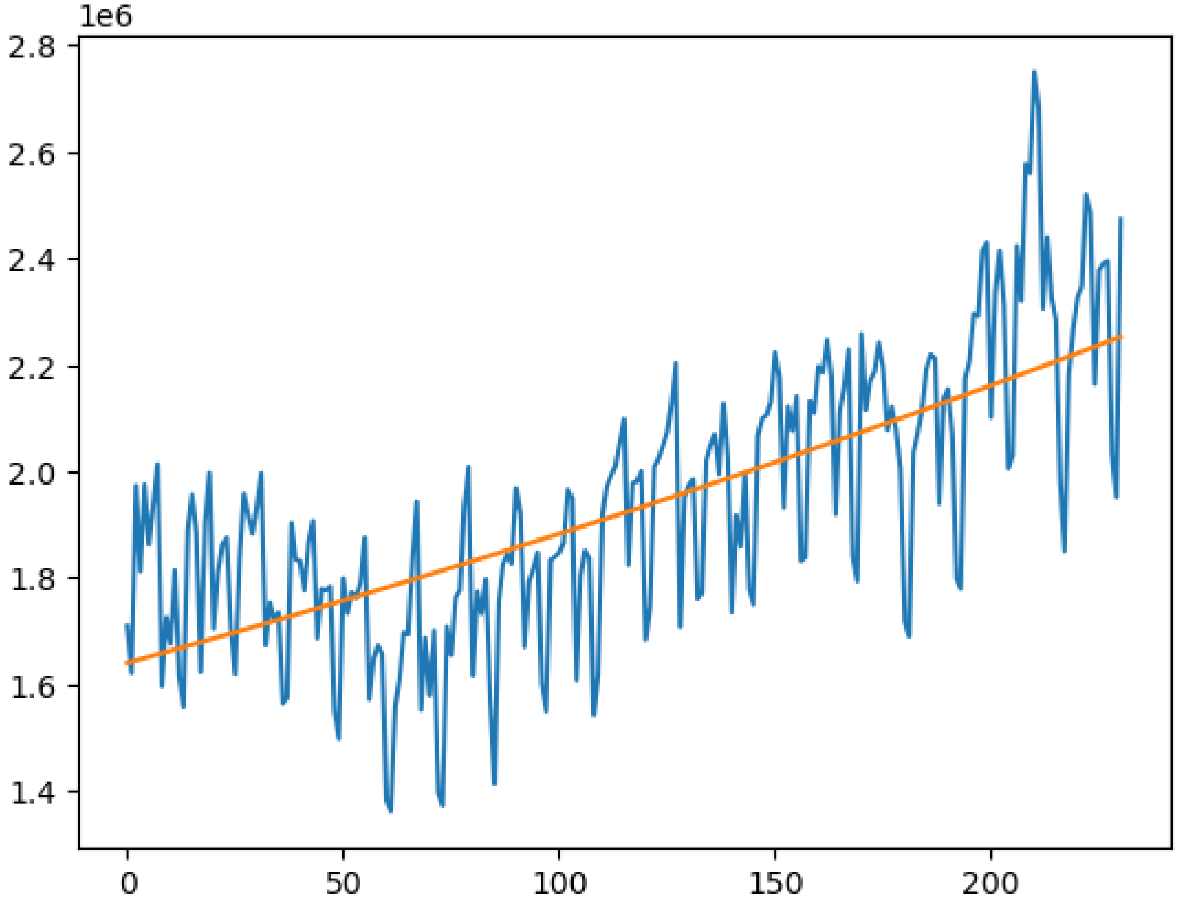
```
2773146.9456016854
```

Exponential Time Series

- Apply natural logarithm to our target variable (y)
- To transform predictions to original space, take exponent

```
reg = LinearRegression()
reg.fit(t, np.log(y))
y_pred = reg.predict(t)
pyplot.plot(data)
pyplot.plot(np.exp(y_pred))
```

Results



What is the noise (ϵ)?

```
np.linalg.norm(y-np.exp(y_pred),ord=2)
```

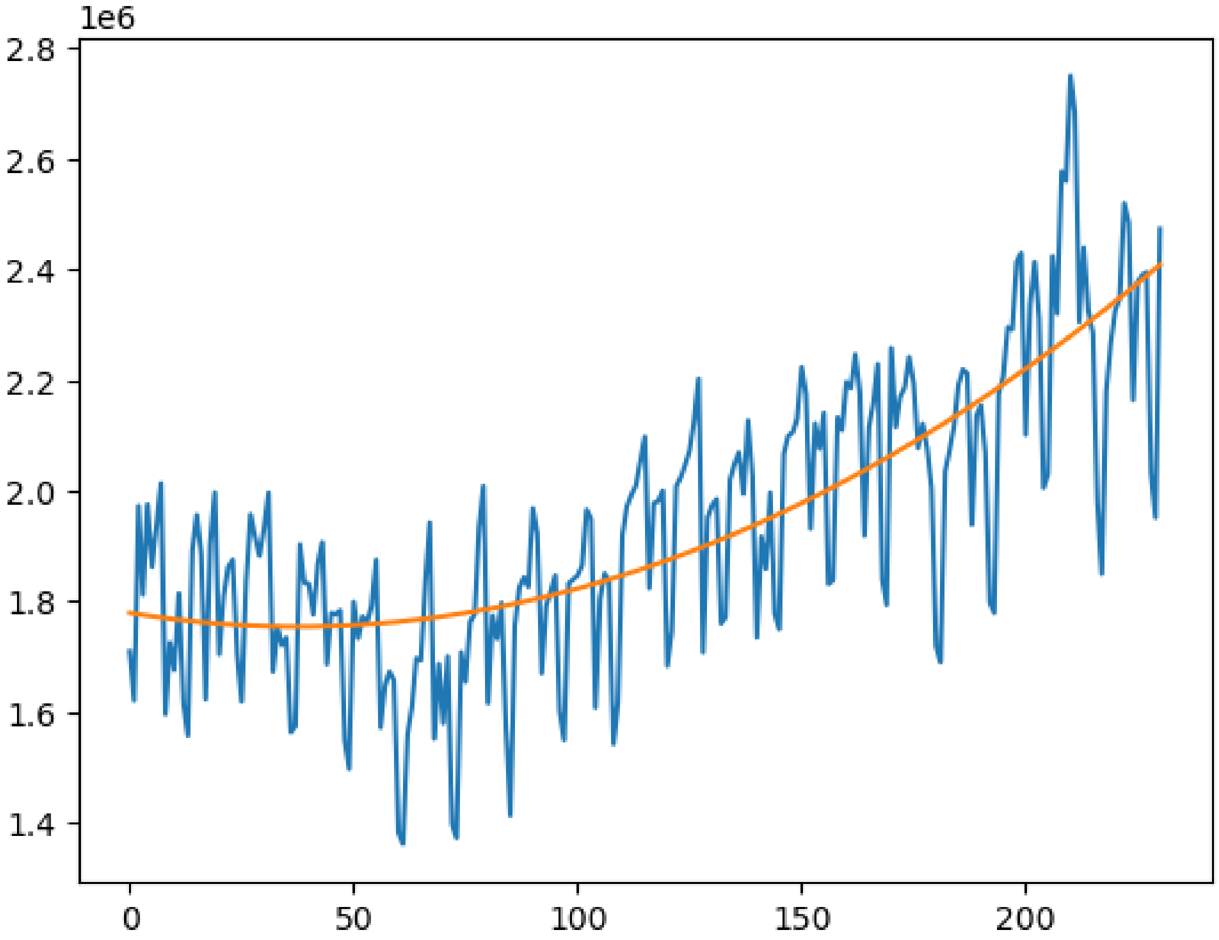
```
2737384.085111154
```

Polynomial Time Series

- Add predictor that is polynomial to t, e.g. t^2
- Build regression model on t and t^2
- Any trend shape can fit, as long as it has a mathematical representation

```
t2 = t*t
reg = LinearRegression()
reg.fit(np.concatenate((t,t2),axis=1),y)
y_pred = reg.predict(np.concatenate((t,t2
pyplot.plot(data)
pyplot.plot(y_pred)
```

Results



What is the noise (ϵ)?

```
np.linalg.norm(y-y_pred,ord=2)
```

2559328.6732444405

Seasonality

- Amtrak exhibits strong monthly seasonality

- Create a new categorical variable for the season of the observation

  month = t % 12

- Convert categorical variable into dummy variables
  - For m variables, we create m-1 variables

Seasonality

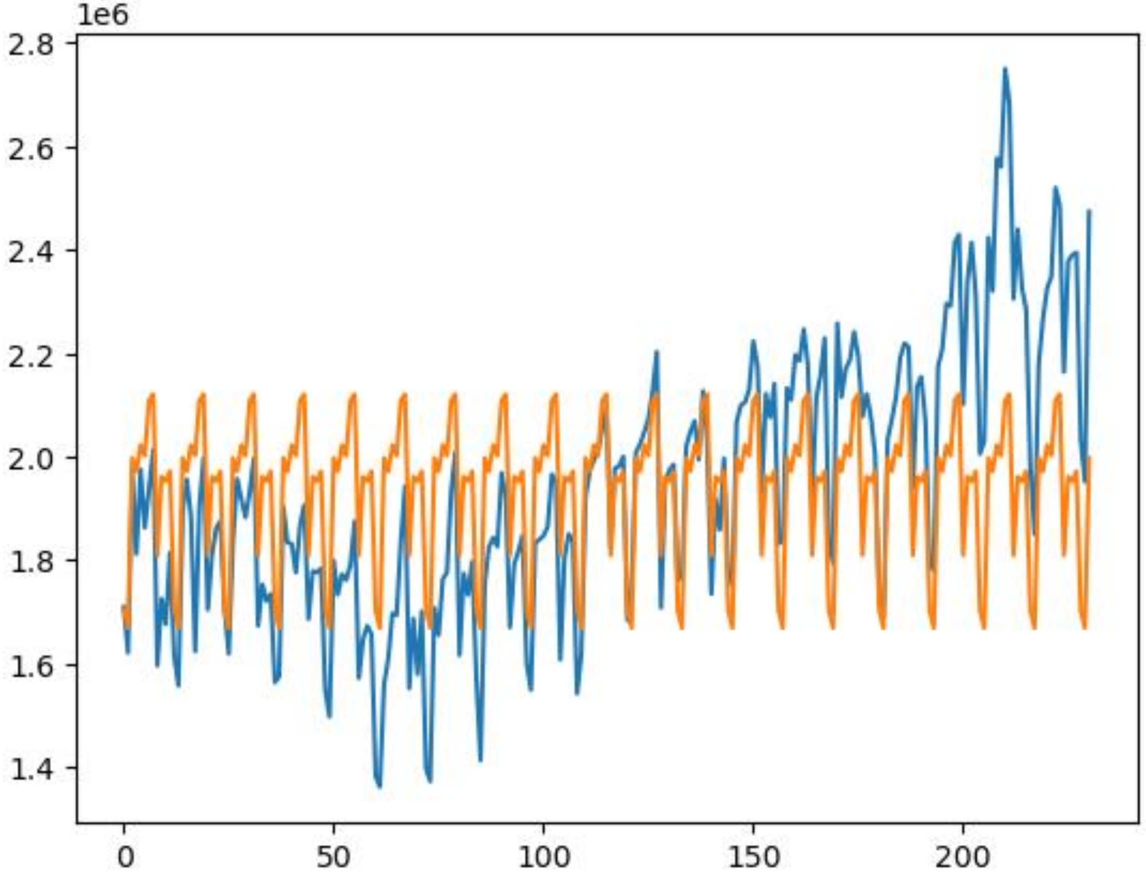
```
month = t % 12                    # convert to months


from sklearn.preprocessing import OneHotEncoder
hot = OneHotEncoder()             # create dummy variables
hot.fit(month)
onehot = hot.transform(month)
onehot = onehot.todense()
onehot = onehot[:,:11]            # keep m-1 dummy variables
```

Seasonality (cont.)

```
# Build regression model for seasonality
reg = LinearRegression()
reg.fit(onehot,y)
y_pred = reg.predict(onehot)
```

Results

pyplot.plot(data)
pyplot.plot(y_pred)

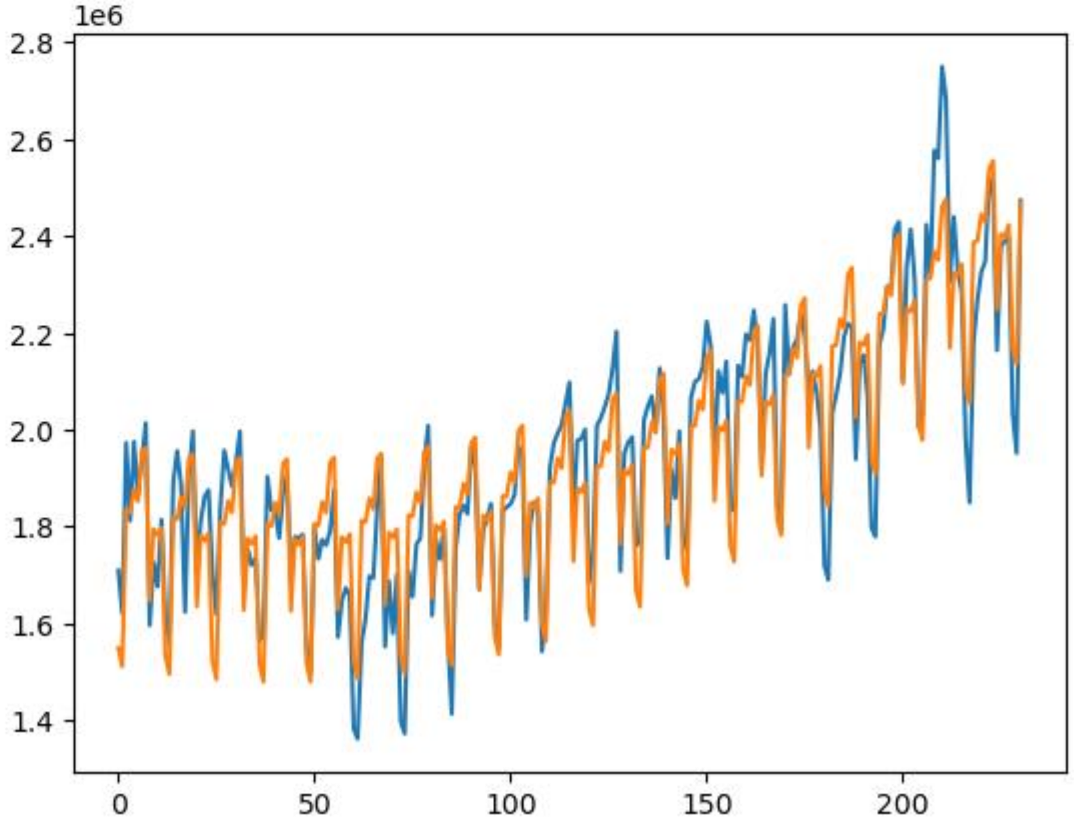


Modeling Trend and Seasonality

- Combine t and t^2 for trend and 11 dummy variables for seasonality

```
reg = LinearRegression()
reg.fit(np.concatenate((t,t2,onehot),axis=1),y)
y_pred = reg.predict(np.concatenate((t,t2,onehot),
pyplot.plot(data)
pyplot.plot(y_pred)
```

Results



What is the noise (ϵ)?

```
np.linalg.norm(y-y_pred,ord=2)
```

```
1400365.5781889278
```

Autocorrelation

- In time series, observations in neighboring periods tend to be correlated
- Autocorrelation – correlation between values of a time series in neighboring periods
  - Relationship between time series and itself
- Lagged series – copy of original series that is move forward 1 or more time periods

y2 = y.reshape((231,))

np.corrcoef(y2[:230],y2[1:231])   # Lag 1

```
y2 = y.reshape((231,))
np.corrcoef(y2[:230],y2[1:231])
```

```
array([[1.        , 0.76584186],
       [0.76584186, 1.        ]])
```

Autocorrelation – Interesting Patterns

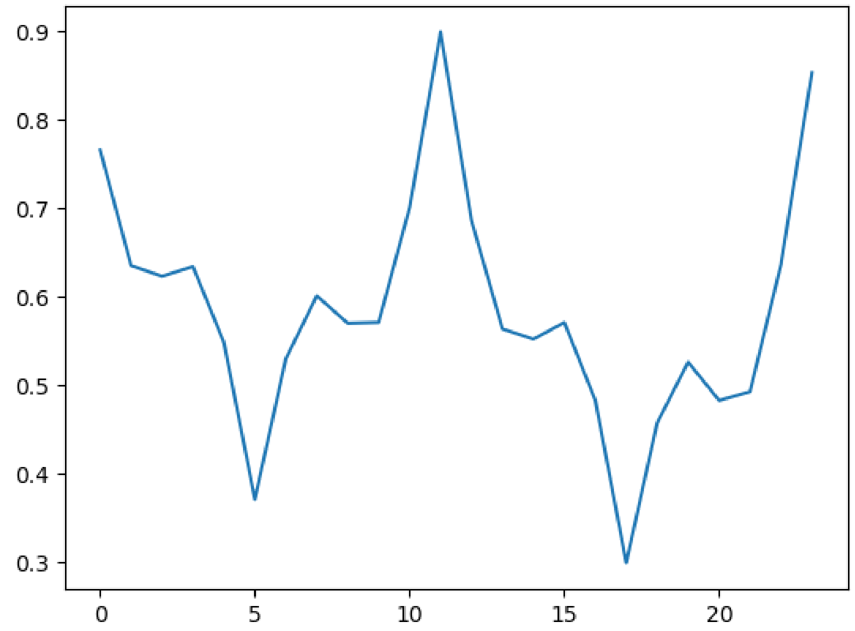
- Strong autocorrelation (positive or negative) at a lag $k > 1$
  - Typically reflects a cyclical pattern

- Positive lag-1 autocorrelation (stickiness)
  - Consecutive variables move in the same direction

- Negative lag-1 autocorrelation
  - Swings in the series – high values are immediately followed by low values

Autocorrelation - Amtrak

```
y2 = y.reshape((231,))
error = []
for k in range(1,25):
    error.append(np.corrcoef(y2[:231-k],y2[k:231])[0,1])
pyplot.plot(error)
```

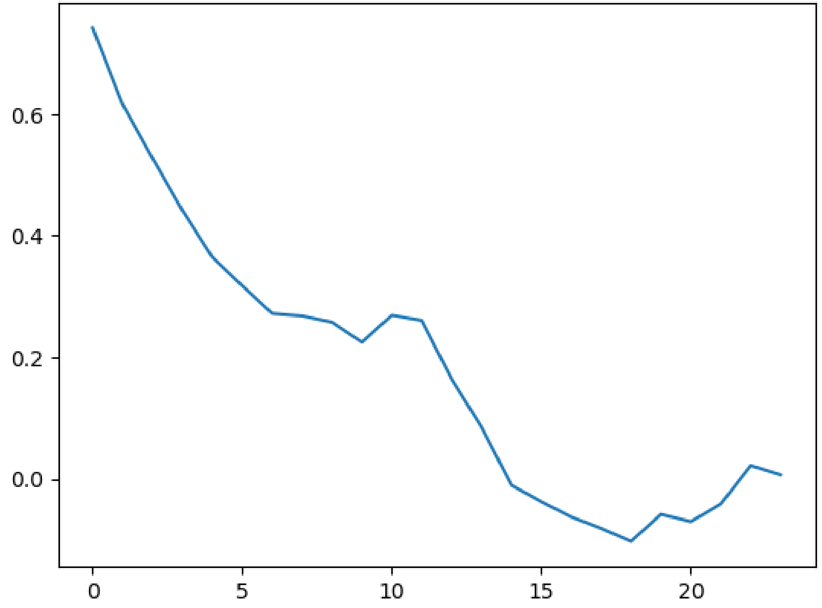


- Observation: Strong correlation when k = 12, 24, etc.
  - Indicates seasonal pattern

Autocorrelation of residuals

- If we have adequately modeled the seasonal pattern, then residual should show no autocorrelation

```
y2 = y-y_pred
y2 = y2.reshape((231,))
error = []
for k in range(1,25):
    error.append(np.corrcoef(y2[:231-k],y2[k:231])[0,1])
pyplot.plot(error)
```



Autocorrelation of Residual - Amtrak

- For lag > 1, autocorrelation is low
  - Modeled the seasonality of the data
- Strong positive autocorrelation at lag 1
  - Positive relationship between neighboring residuals

Autoregressive models

- Directly account for autocorrelation in the model
- Similar to linear regression, except predictors are past values of the series

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \epsilon$$

- ARIMA models – Autoregressive integrated moving average models
  - Creates larger set of more flexible models
  - Requires more statistical expertise to chose the order of the model

Implementation

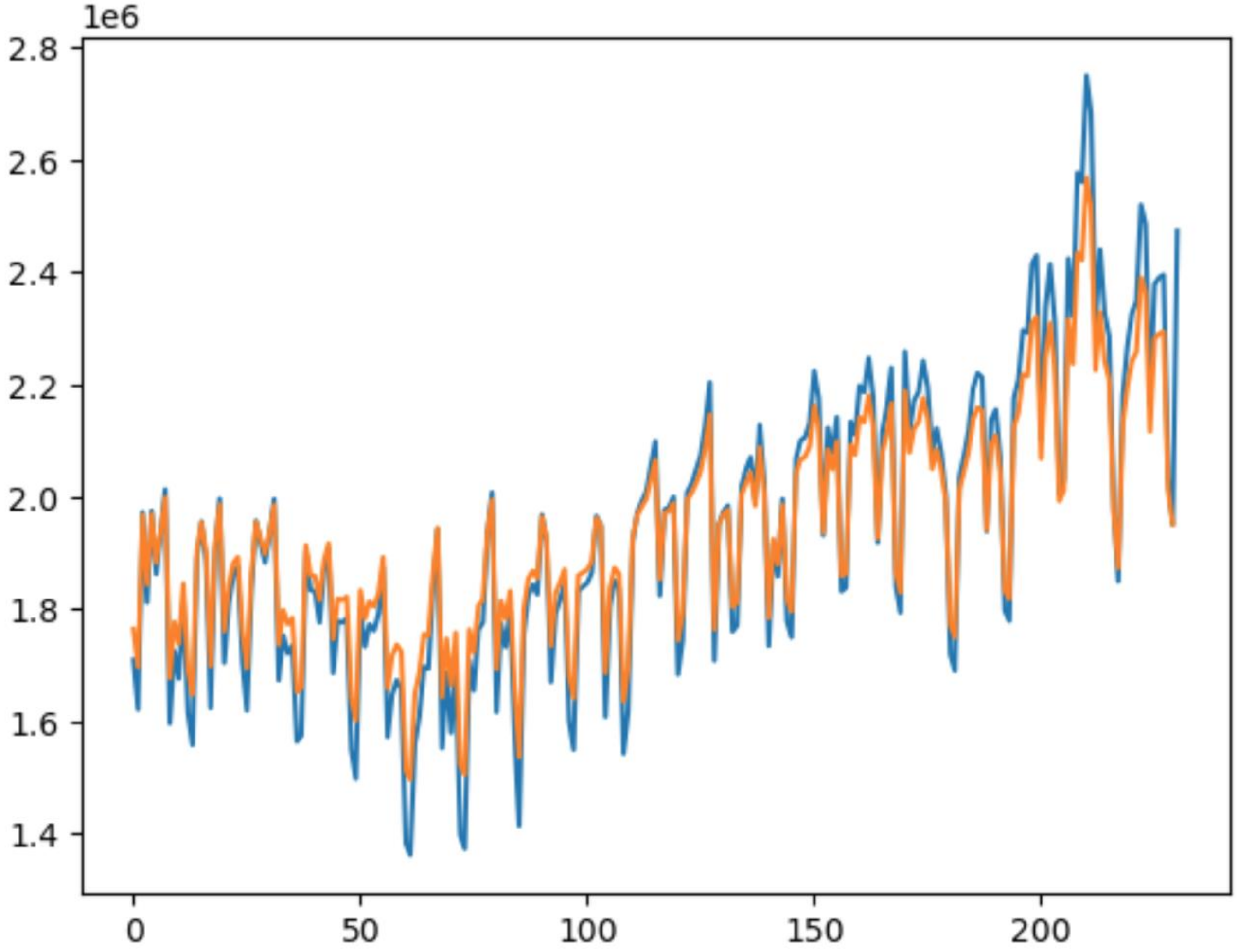
statsmodels

- Python package statsmodels

```
conda install anaconda::statsmodels
```

```
from statsmodels.tsa.ar_model import AutoReg

# Fit an autoregressive model
lag = 1
model = AutoReg(data, lags=lag)
model_fit = model.fit()

# Generate predictions based on the autoregressive model
y_pred = model_fit.predict(start=lag, end=len(data)-1)

pyplot.plot(data)
pyplot.plot(y_pred)
```

Results



DATA VISUALIZATION

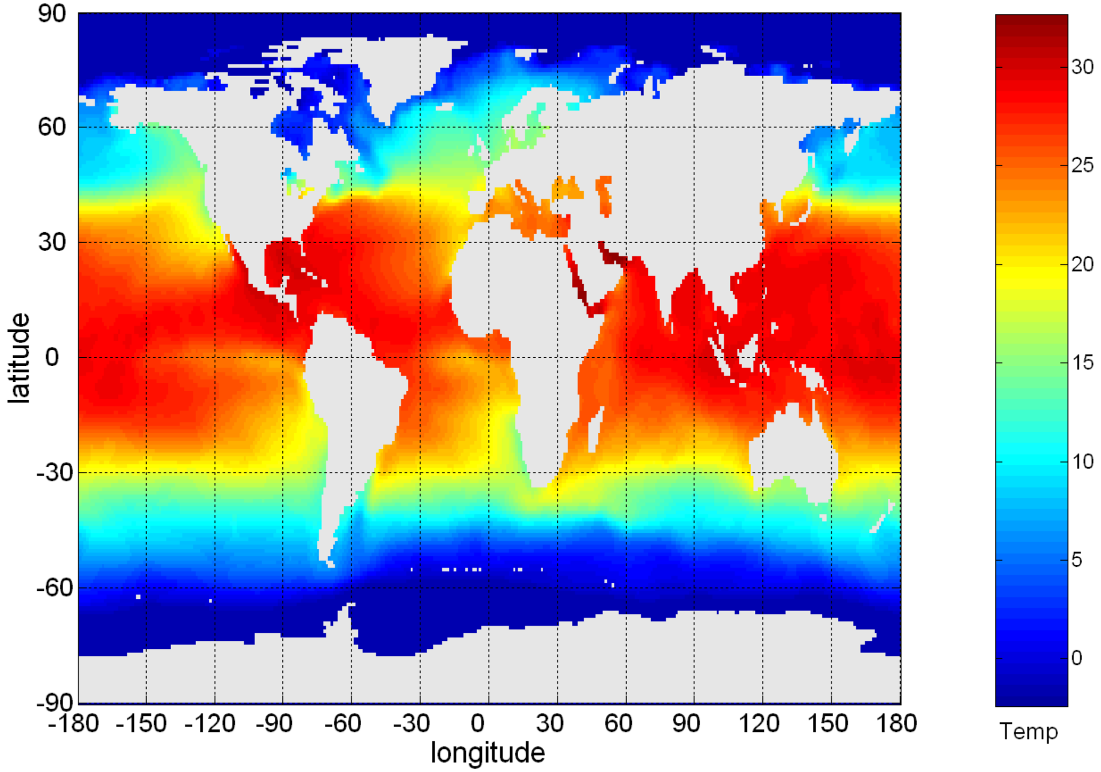
Visualization

Visualization is the conversion of data into a visual or tabular format so that the characteristics of the data and the relationships among data items or attributes can be analyzed or reported.

- Visualization of data is one of the most powerful and appealing techniques for data exploration.
  - Humans have a well developed ability to analyze large amounts of information that is presented visually
  - Can detect general patterns and trends
  - Can detect outliers and unusual patterns

Example: Sea Surface Temperature

- The following shows the Sea Surface Temperature (SST) for July 1982
  - Thousands of data points are summarized in a single figure



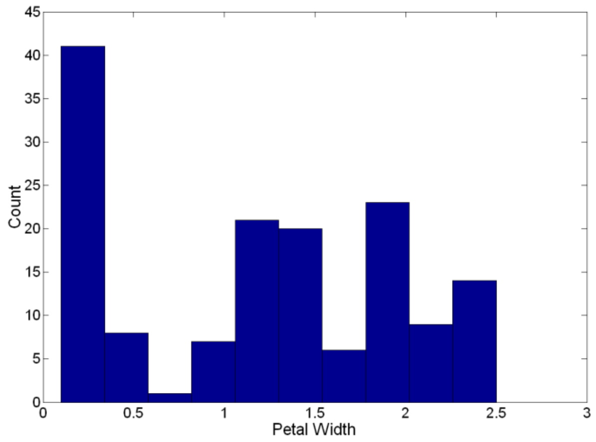
Iris Sample Data Set

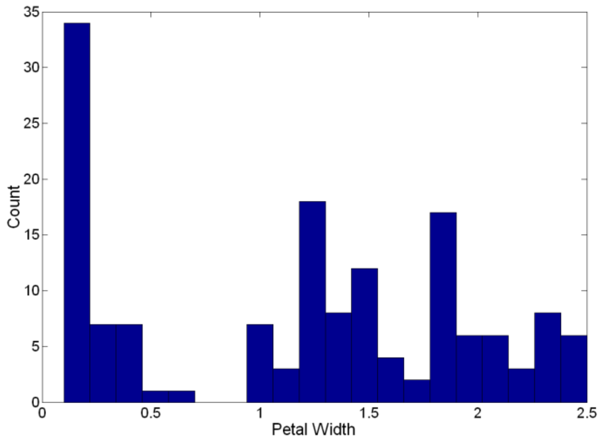
- Many of the exploratory data techniques are illustrated with the Iris Plant data set.
  - Can be obtained from the UCI Machine Learning Repository http://www.ics.uci.edu/~mlearn/MLRepository.html
  - From the statistician Douglas Fisher
  - Three flower types (classes):
    - Setosa
    - Virginica
    - Versicolour
  - Four (non-class) attributes
    - Sepal width and length
    - Petal width and length

Virginica. Robert H. Mohlenbrock. USDA NRCS. 1995. Northeast wetland flora: Field office guide to plant species. Northeast National Technical Center, Chester, PA. Courtesy of USDA NRCS Wetland Science Institute.

Visualization Techniques: Histograms

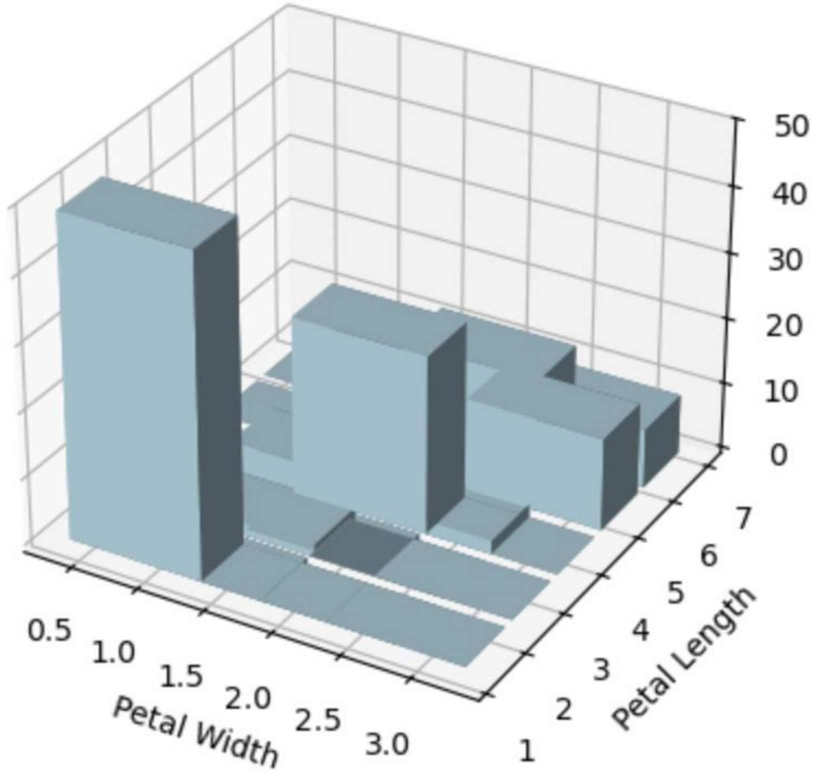
- Histogram
  - Usually shows the distribution of values of a single variable
  - Divide the values into bins and show a bar plot of the number of objects in each bin.
  - The height of each bar indicates the number of objects
  - Shape of histogram depends on the number of bins
- Example: Petal Width (10 and 20 bins, respectively)





Two-Dimensional Histograms

- Show the joint distribution of the values of two attributes
- Example: petal width and petal length

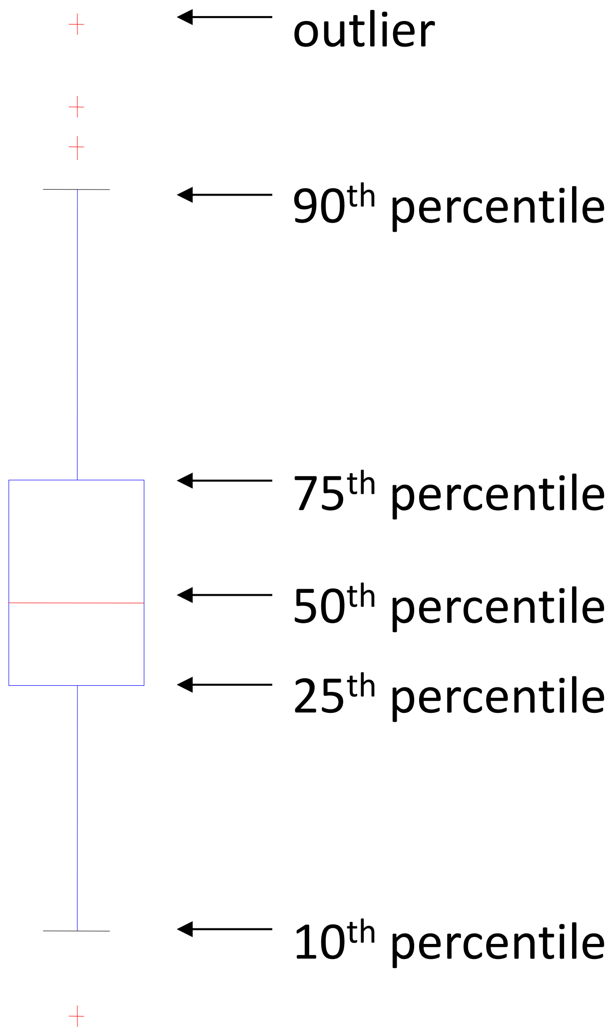


- Implementation

bar3D

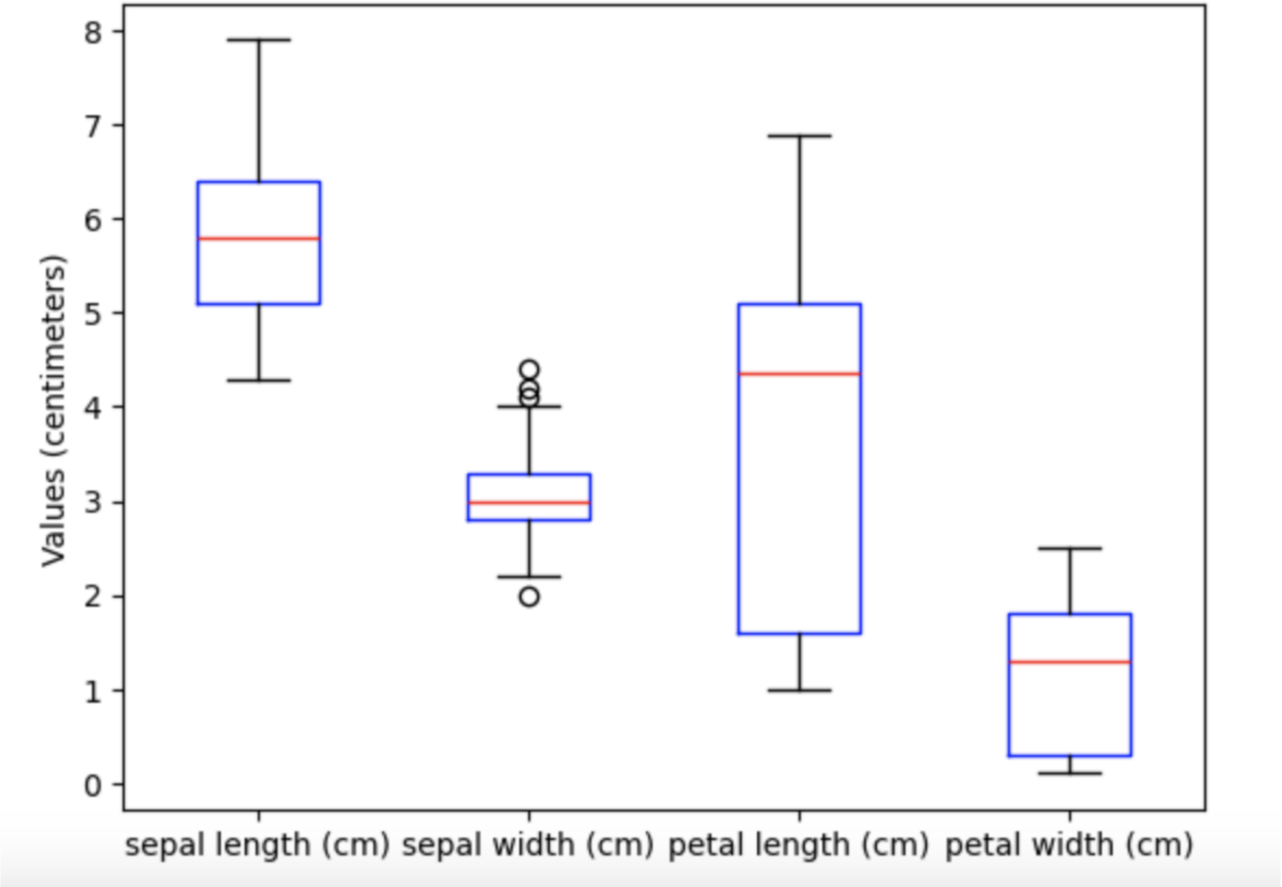
Visualization Techniques: Box Plots

- Box Plots
  - Invented by J. Tukey
  - Another way of displaying the distribution of data
  - Following figure shows the basic part of a box plot



Example of Box Plots

- Box plots can be used to compare attributes

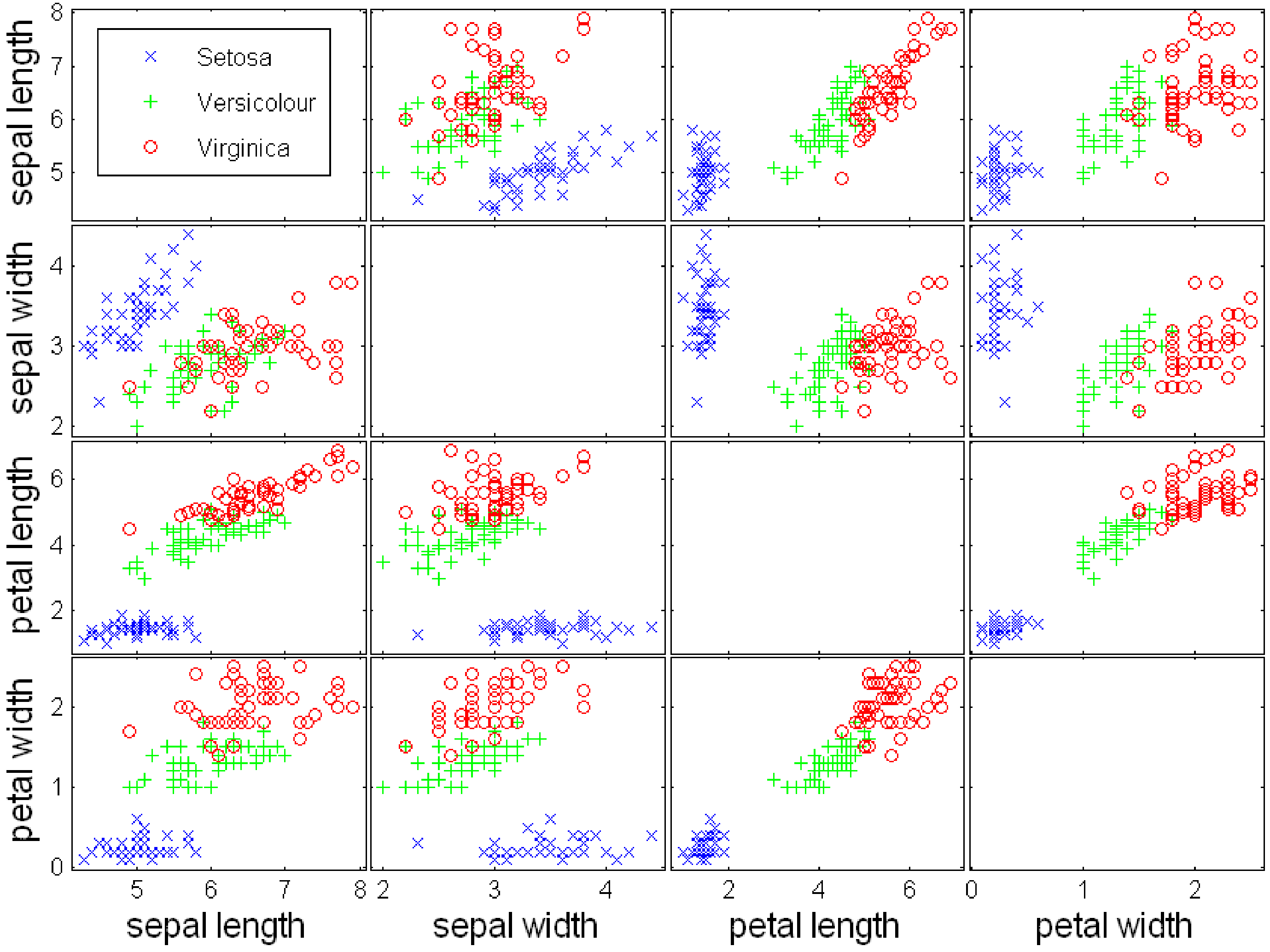


Implementation
plt.boxplot

Visualization Techniques: Scatter Plots

- Scatter plots
  - Attributes values determine the position
  - Two-dimensional scatter plots most common, but can have three-dimensional scatter plots
  - Often additional attributes can be displayed by using the size, shape, and color of the markers that represent the objects
  - It is useful to have arrays of scatter plots can compactly summarize the relationships of several pairs of attributes
    - See example on the next slide

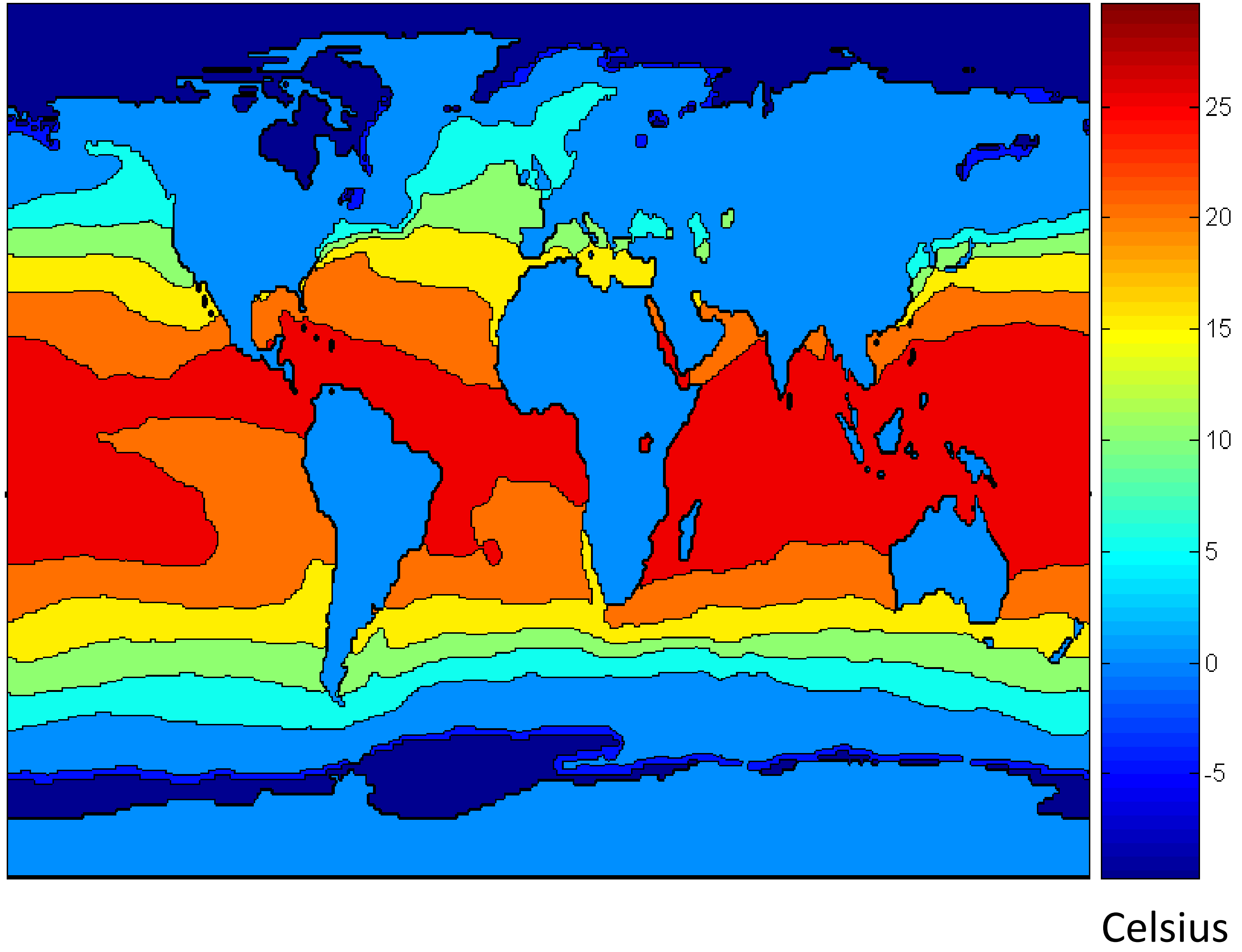
Scatter Plot Array of Iris Attributes



Visualization Techniques: Contour Plots

- Contour plots
  - Useful when a continuous attribute is measured on a spatial grid
  - They partition the plane into regions of similar values
  - The contour lines that form the boundaries of these regions connect points with equal values
  - The most common example is contour maps of elevation
  - Can also display temperature, rainfall, air pressure, etc.
    - An example for Sea Surface Temperature (SST) is provided on the next slide

Contour Plot Example: SST Dec, 1998

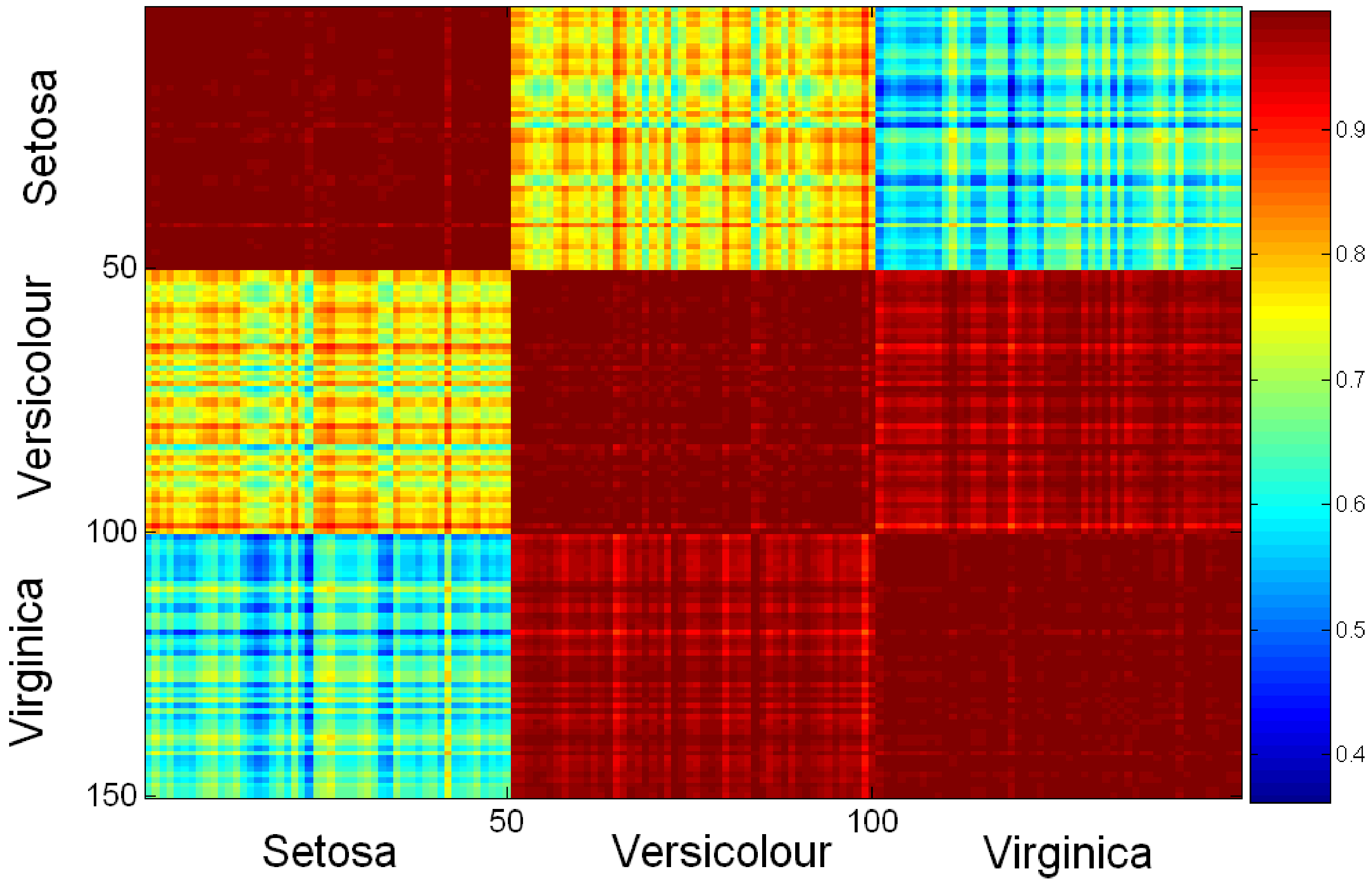


ax.contourf

Visualization Techniques: Matrix Plots

- Matrix plots
  - Can plot the data matrix
  - This can be useful when objects are sorted according to class
  - Typically, the attributes are normalized to prevent one attribute from dominating the plot
  - Plots of similarity or distance matrices can also be useful for visualizing the relationships between objects
  - Examples of matrix plots are presented on the next two slides

Visualization of the Iris Correlation Matrix



Visualize High-dimensional Data

- T-distributed neighbor embedding (t-SNE) is a dimensionality reduction technique that helps users visualize high-dimensional data sets.

- Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualization similarly to t-SNE

conda install -c conda-forge umap-learn