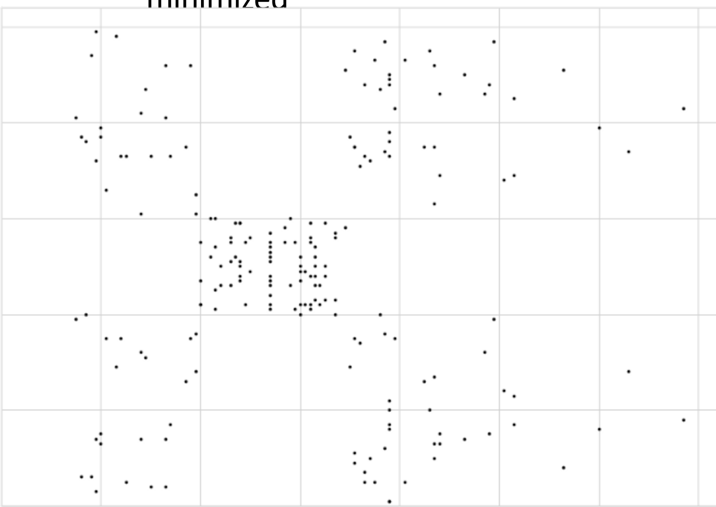
Lec04. Data Analytics for Time Series

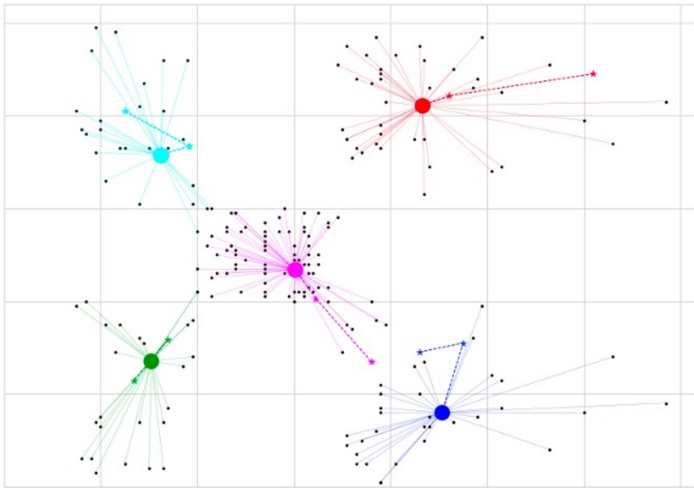
Analysis of special types of data

Week 3 Recap: K-means clustering

- A simple greedy algorithm (usually called Lloyd's algorithm):
  - Divide data into K clusters, each of which has a center (centroid)
  - Each data point belongs to a single cluster only
  - K centroids are chosen such that distance (usually Euclidean) from data points to their centroids is locally minimized



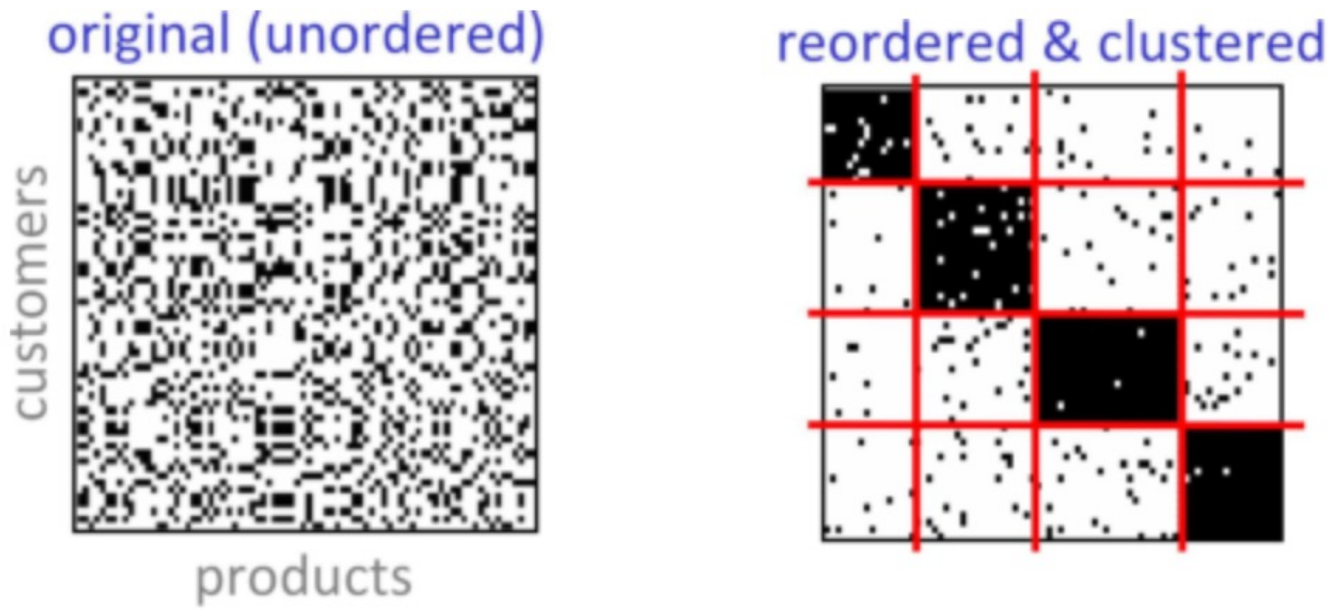
Input



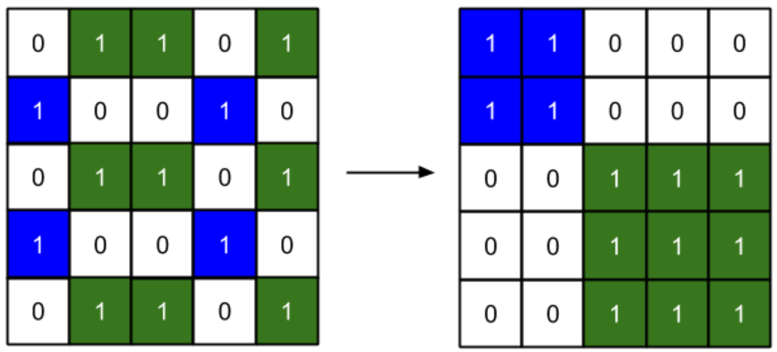
Output

Week 3 Recap: Co-clustering

- Clustering two variables simultaneously



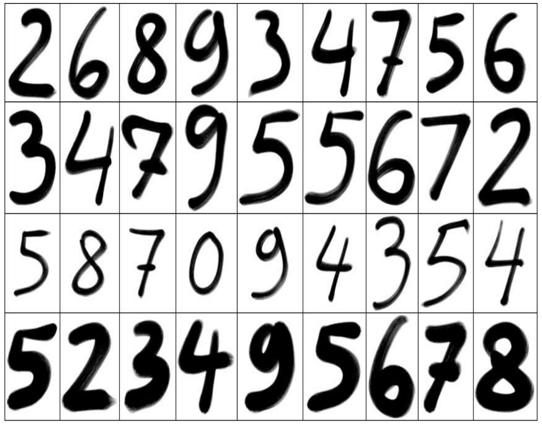
A simple case: similarity values are only 1s and 0s.



Vlachos, Michail et al. "Improving Co-Cluster Quality with Application to Product Recommendations." CIKM (2014).

Week 3 Recap: Classification

❖ Handwritten digit recognition:



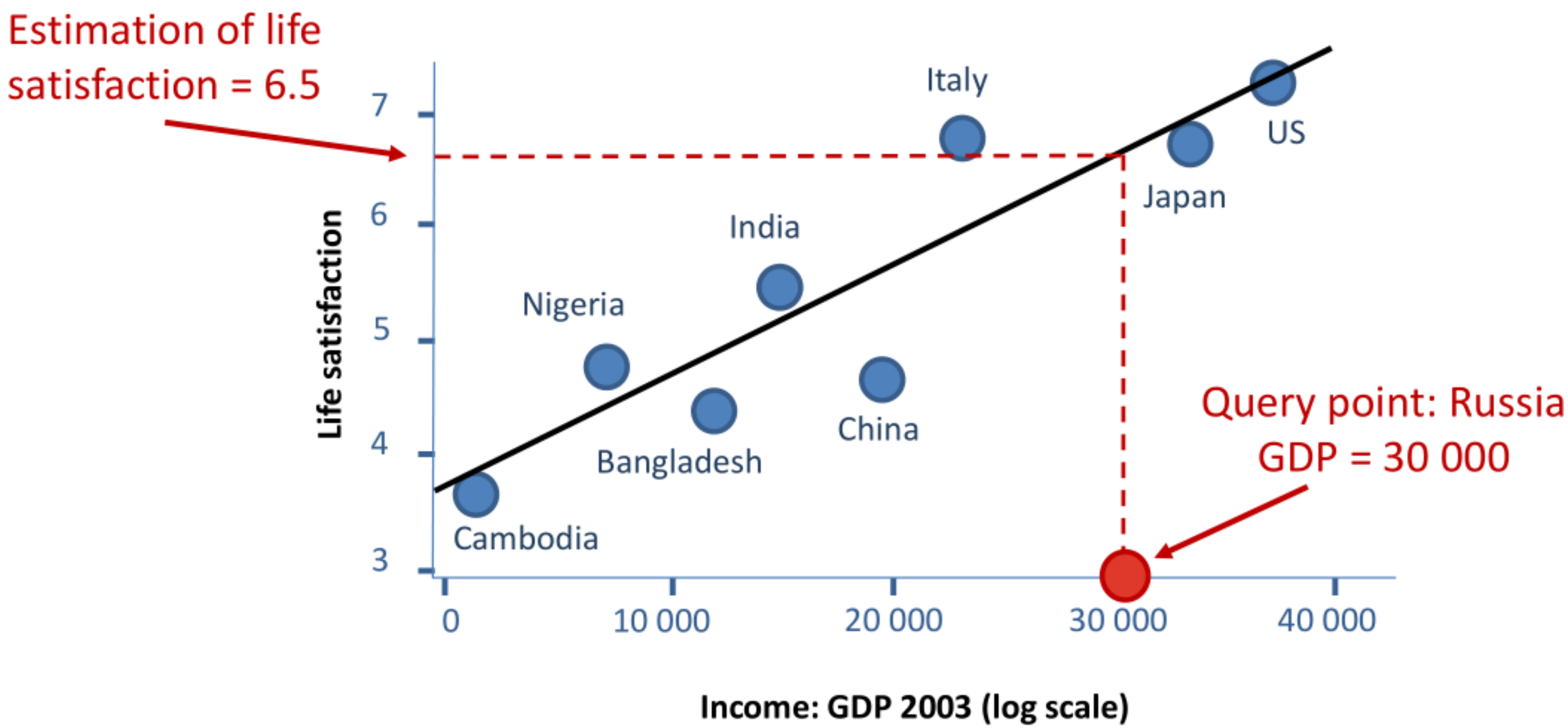
→ Classifier →

2	6	8	9	3	4	7	5	6
3	4	7	9	5	5	6	7	2
5	8	7	0	9	4	3	5	4
5	2	3	4	9	5	6	7	8

❖ Email spam recognition:

Week 3 Recap: Regression

- Maps N-dimensional input $x \in R^N$ to **continuous** values $y \in R$
- e.g. x = [Income (GDP)], y = Continuous value of life satisfaction

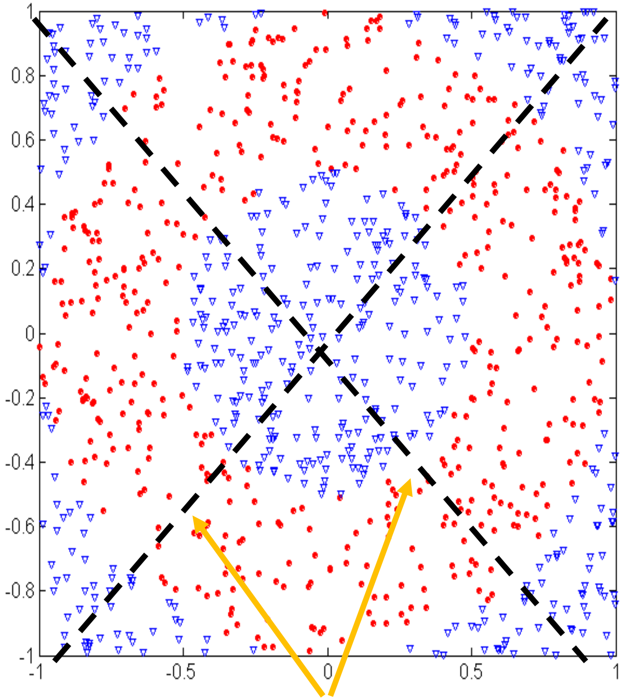


Week 3 Recap: Model Evaluation

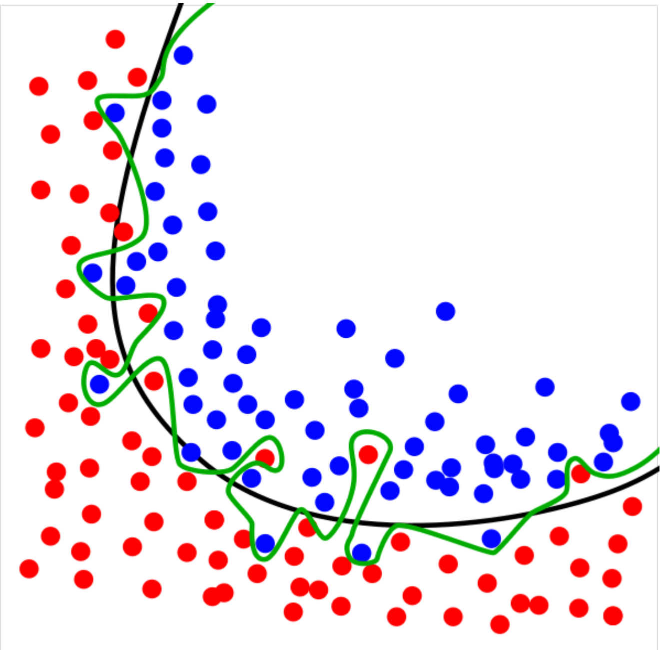
Underfitting

E.g. Regression

Overfitting



Models are too simple!



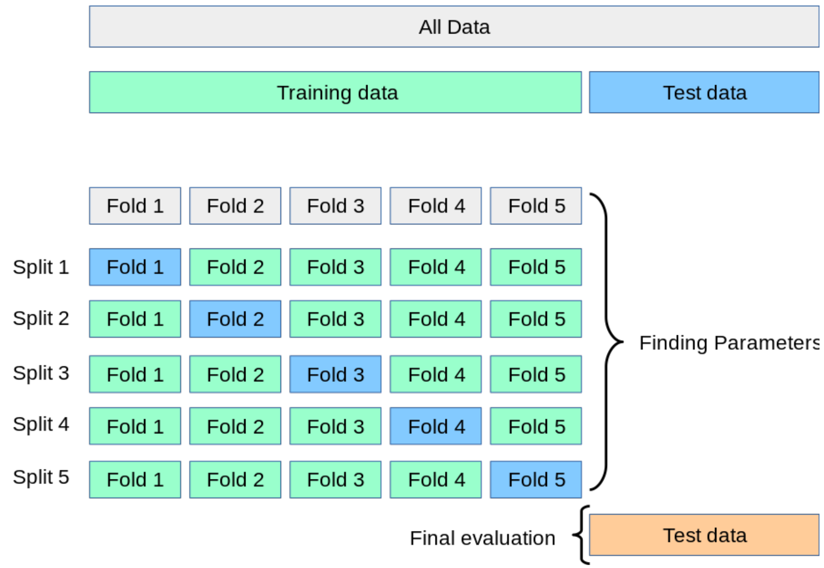
- Green line is overfitting (tailored too much to the training data)
- Black line is what we want

Low variance, high bias

High variance, low bias

Week 3 Recap: K-Fold Cross validation

- Prevent you from over-fitting a single train/test split
- General process:
  - Split your training data into K randomly-assigned folds
  - Reserve one segment as your validation data
  - Train on each of the remaining K-1 folds to find model parameters
  - Take the best parameters based on validation accuracy
- You can do the same process for different train/test splits and take the average result



Course structure

W1. Data Processing with Python

W2. Data Exploration with Python

W3. Data Modeling with Pytyhon

W4. Data Analytics for Timeseries

W5. Holiday

W6-7. Data Analytics for Texts

W8. Data Analytics for Images

W9. Data Analytics for Graphs

W10-11. Data Analytics for Other Data

W12. Revision

Time-Series Data Analytics

I. Definitions and Applications
II. Segmentation
III. Classification
IV. Forecasting
V. Decomposition
VI. Anomaly Detection

I. Definitions and Applications

Time Series (aka sequential data)

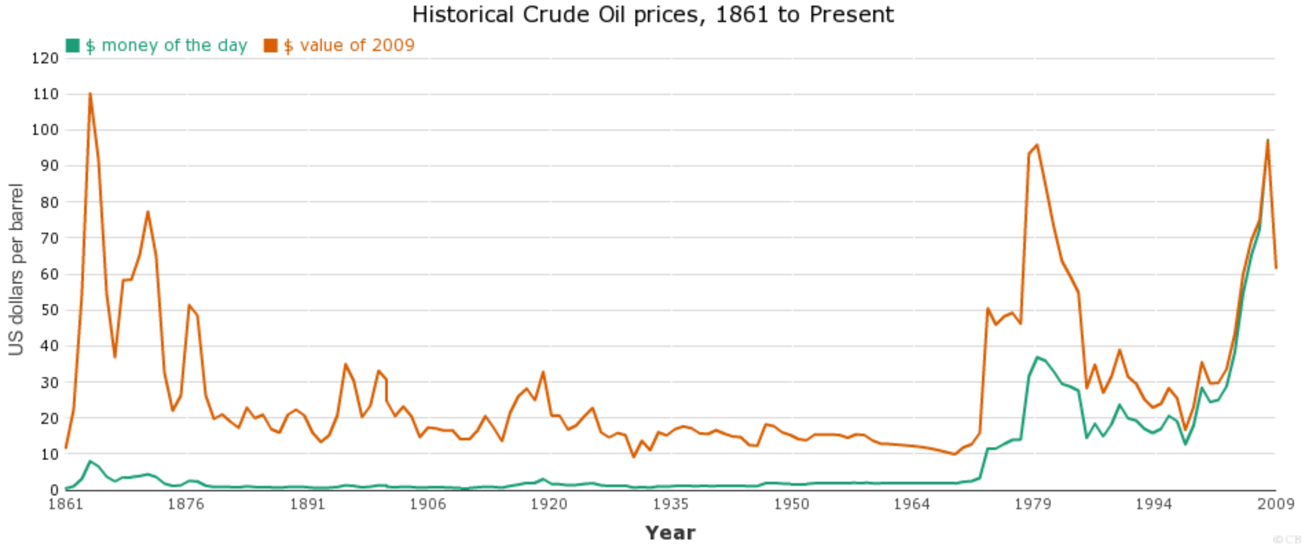
- A series of data entries indexed in time order.
  - Discrete Representation.

    $S = (s_1, s_2, \dots, s_n)$ $\qquad s_{1 \cdot time} < s_{2 \cdot time} < \dots < s_{n \cdot time}$

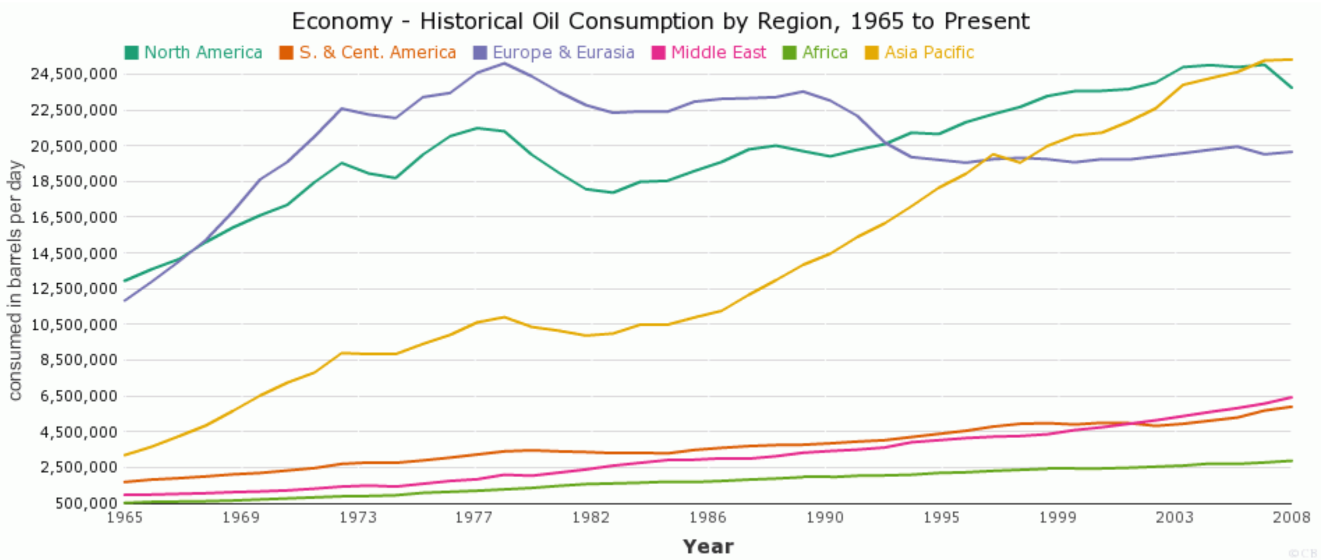
  - Continuous representation.
    - Function of time: $y = f(t)$

- Historical and Sequential data.
- Very frequently plotted via line charts.

Time Series



Historical Crude Oil prices, 1861 to Present
$ money of the day
$ value of 2009
US dollars per barrel
0
10
20
30
40
50
60
70
80
90
100
110
120
1861
1876
1891
1906
1920
1935
1950
1964
1979
1994
2009
Year



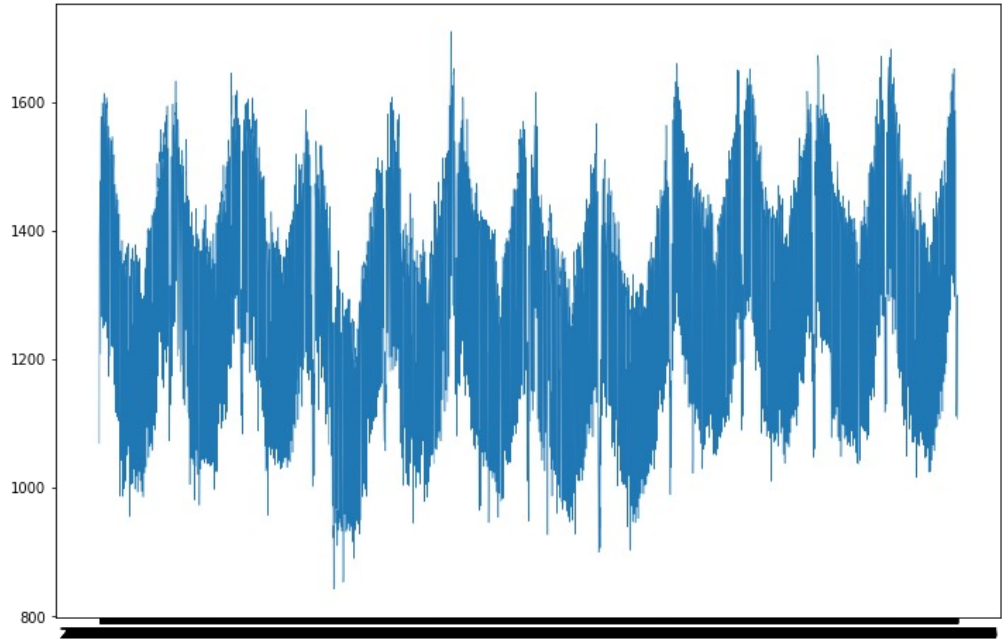
Economy - Historical Oil Consumption by Region, 1965 to Present
North America
S. & Cent. America
Europe & Eurasia
Middle East
Africa
Asia Pacific
consumed in barrels per day
24,500,000
22,500,000
20,500,000
18,500,000
16,500,000
14,500,000
12,500,000
10,500,000
8,500,000
6,500,000
4,500,000
2,500,000
500,000
1965
1969
1973
1977
1982
1986
1990
1995
1999
2003
2008
Year

Time Series Applications

- Economic and Sales Forecasting.
- Weather Forecasting.
- Budget and Stock Market Analysis.
- Process and Quality Control.
- Moving Object's Analysis.
- Recommendation Systems.
- Pattern Recognition.
- etc.

Exploratory Analysis of Time Series

- Examine a time-series with statistics:
  - **Pros:** Summarize the values
  - **Cons:** do not consider the timestamps

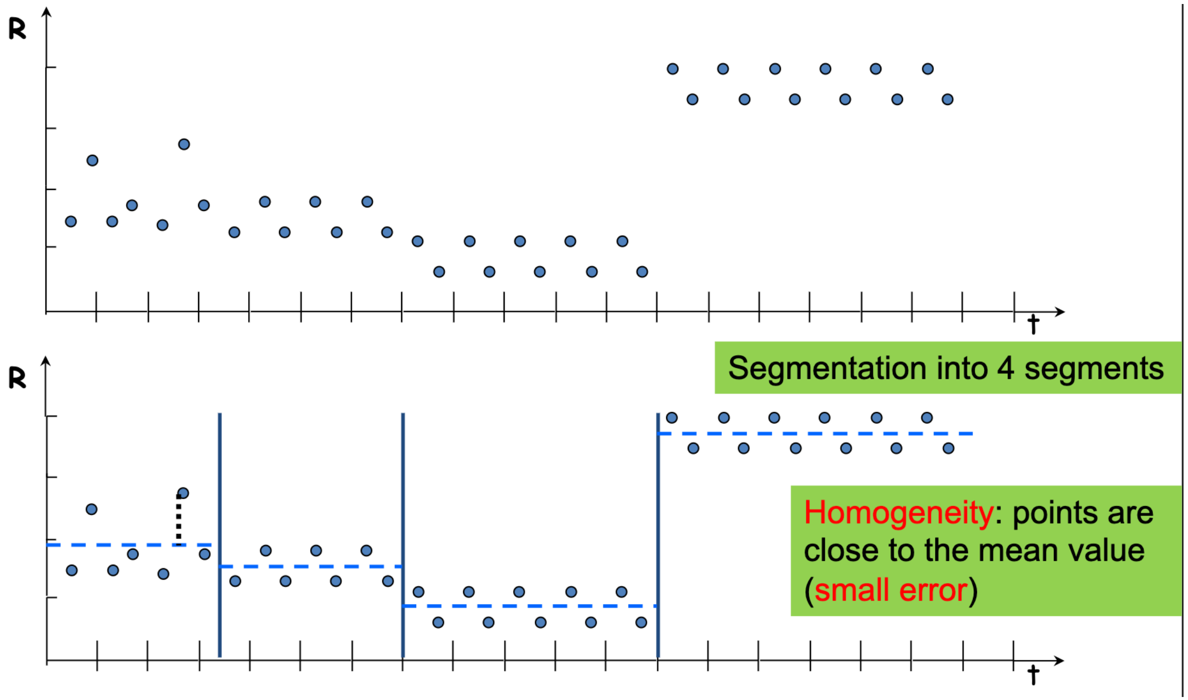


count	4383.000000
mean	1338.675836
std	165.775710
min	842.395000
25%	1217.859000
50%	1367.123000
75%	1457.761000
max	1709.568000

II. Segmentation

Time Series Segmentation

- **Goal:** discover structure in the time series and provide a concise summary
  - Useful for really long time series
  - Divide and conquer: Make it easier to analyze
- **How:** given a time series S, segment it into K disjoint segments (or partitions) that are as homogeneous as possible
  - Data points in the same segment are ``similar''
  - Similar to clustering but only allow grouping along the time dimension



K-segmentation problem

❖ **Problem**: Given a time series S of length N and a value K, find a K-segmentation $P = \{p_1, \dots, p_k\}$ of S such that the sum of square errors is minimized:

$$E(P) = \sum_{p \in P} \sum_{s \in P} (s - \mu_p)^2$$

where μ_p is the mean of segment p.

❖ **Optimal solution:** using Bellman's algorithm based on dynamic programming

- Construct the solution of the problem by using solutions to problems of smaller size
- Build the solution bottom up from smaller to larger instances

$$E(P_{opt}(S[1,..,n],k) = \min_{j<n}(E\underbrace{\left(P_{opt}(S[1,..,j],k-1)\right)}_{\text{Smaller size}} + E\underbrace{\left(P_{opt}(S[j+1,..,n],1)\right)}_{\text{Bottom-up}})$$

https://justinwillmert.com/articles/2014/bellman-k-segmentation-algorithm/

Other segmentation algorithms

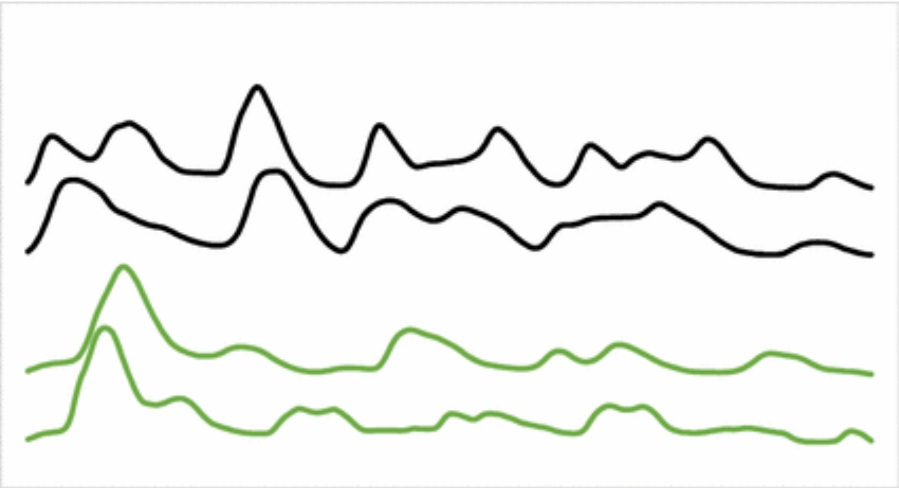
- Bottom-up greedy (BU): O(nlogn)
  - Keogh and Smyth'97, Keogh and Pazzani'98
- Top-down greedy (TD): O(nlogn)
  - Douglas and Peucker'73, Shatkay and Zdonik'96, Lavrenko et. al'00
- Global Iterative Replacement (GiR): O(nI)
  - Himberg et. al '01
- Local Iterative Replacement (LiR): O(nI)
  - Himberg et. al '01
- DnS approximation algorithm
- …

III. Classification

Time Series Classification

- Assigning time series pattern to a specific category.
  - Given a time series $\mathrm{Y} = (y_1, y_2, \dots, y_n)$ and a list of categories $C = (c_1, c_2, \dots, c_k)$, we want to assign Y to it the best matching category c_i.
  - Needs a similarity/distance measure.

- **Applications**:
  - Identify a word based on series of hand movements in sign language.
  - Handwriting classification.
  - Moving pattern similarity.

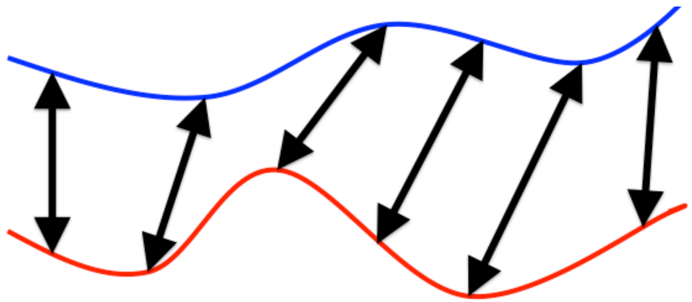


Basic Approach

Apply traditional classification algorithms, e.g. k-nearest neighbors (KNN)

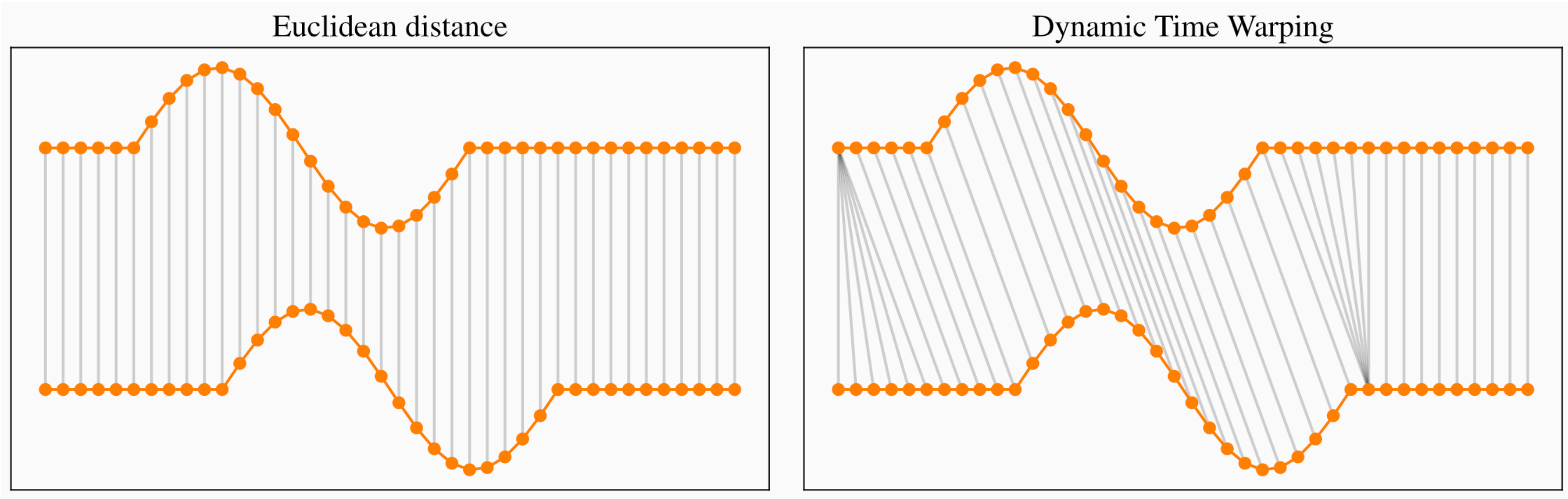
1. Consider each time series as a multi-dimensional item
   - Each time step is an attribute
2. Measure distance between two time series:
   - Traditional measures:
     - Euclidian distance
     - Hamming distance
     - Manhattan distanc
     - Minkowski distance
   - However, these measures do not capture temporal patterns

→ Need better distance measure



Time Series Similarity

- **Dynamic Time Warping (DTW):** ”best-practice” distance measure.
  - Suppose we want to measure the distance between two series $\mathrm{A} = (a_1, a_2, \dots, a_n)$ and $\mathrm{B} = (b_1, b_2, \dots, b_n)$. Let $M(A, B)$ be the pointwise distance matrix between *A* and *B*. The **DTW** distance between series is the path through *M* that minimizes the total distance (Path between the points).
  - If $\mathbf{P}$ is the space of all possible paths, the DTW path $\mathbf{P}^*$is the path that has the minimum distance.
    - DTW can be calculated using dynamic programing.



Other classification algorithms

- Shapelet
  - Time series often exhibits characteristic data shapes that are indictive of the class, e.g. brain of heart data
  - → Can use the geometry property of time series subsequences for a classifier
- Segmentation-based approaches
  - Segment the time series into distinct intervals
  - E.g. Bag-of-Patterns method, Time Series forest method.
- Deep learning
  - Recurrent neural networks
  - LSTM
  - …

IV. Forecasting

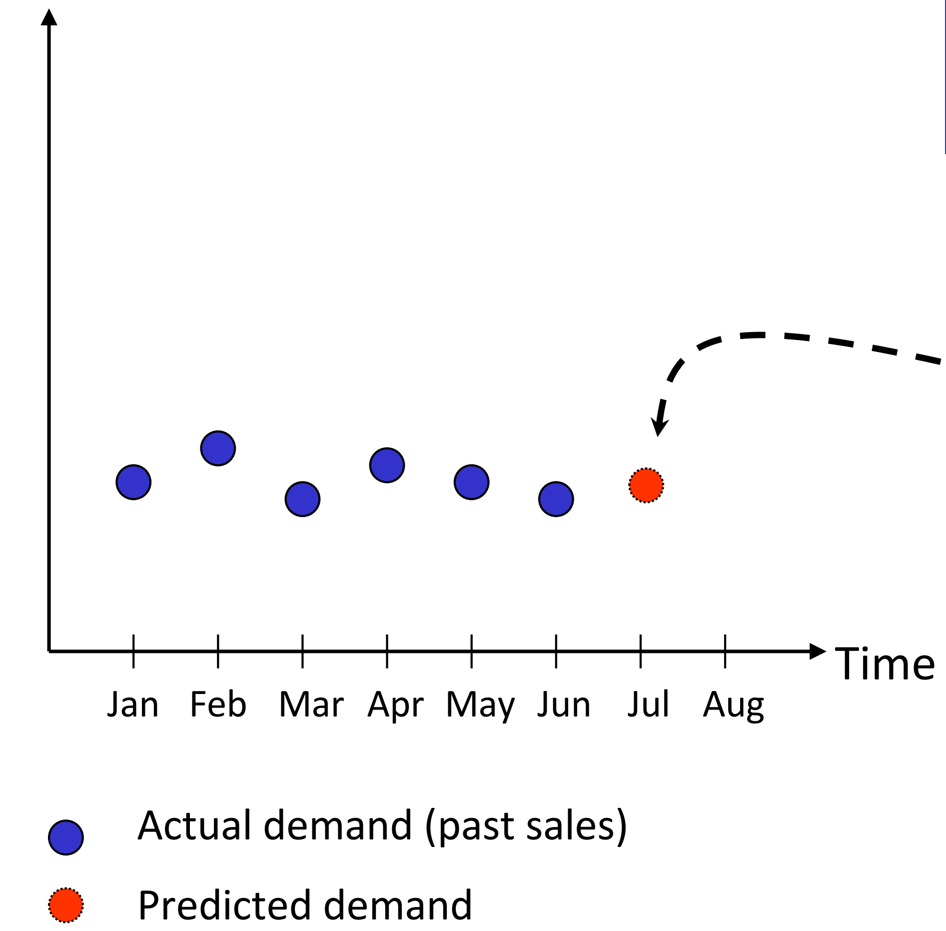
Forecasting Basics

- What is forecasting?
  - Forecasting is a tool used for predicting future based on past information.
- Applications:
  - Strategic planning (long range planning)
  - Finance and accounting (budgets and cost controls)
  - Marketing (future sales, new products)
  - Production and operations

Forecasting: Example

We try to predict the future by looking back at the past

Demand for Mercedes E Class



Predicted demand looking back six months

Forecasting Fundamentals

1. Basic Principles
2. Models
3. Accuracy

1. Basic Principles

- **Assumptions:**
  - Historical data contains **some patterns** → support predicting the future
- **Best practices:**
  - Forecasts are rarely perfect
  - Forecasts are more accurate for shorter than longer time periods
  - Forecasts are more accurate for grouped data than for individual items
  - Every forecast should include an error estimate

2. Forecasting Models

- **Techniques:**
  - Moving Average
  - Weighted Moving Average
  - Exponential Smoothing
  - Exponential Smoothing with Trend
  - Seasonality
  - Causal Model

Time-Series: Moving Average Method

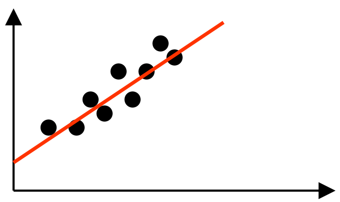
- Moving Average (MA) is a series of arithmetic means
- **Calculation**: the average value over a set time period (e.g. last four weeks)

$$F_{t+1} = \frac{\sum_{t-n+1}^{t} A_i}{n}$$

Where F_t is the forecast value at time t

A_t is the actual value at time t

- **Limitations**:
  - Do not forecast trends well
  - **Trends:** persistent, overall upward and downward pattern
    - typically over several years
    - e.g. population, technology, age, culture



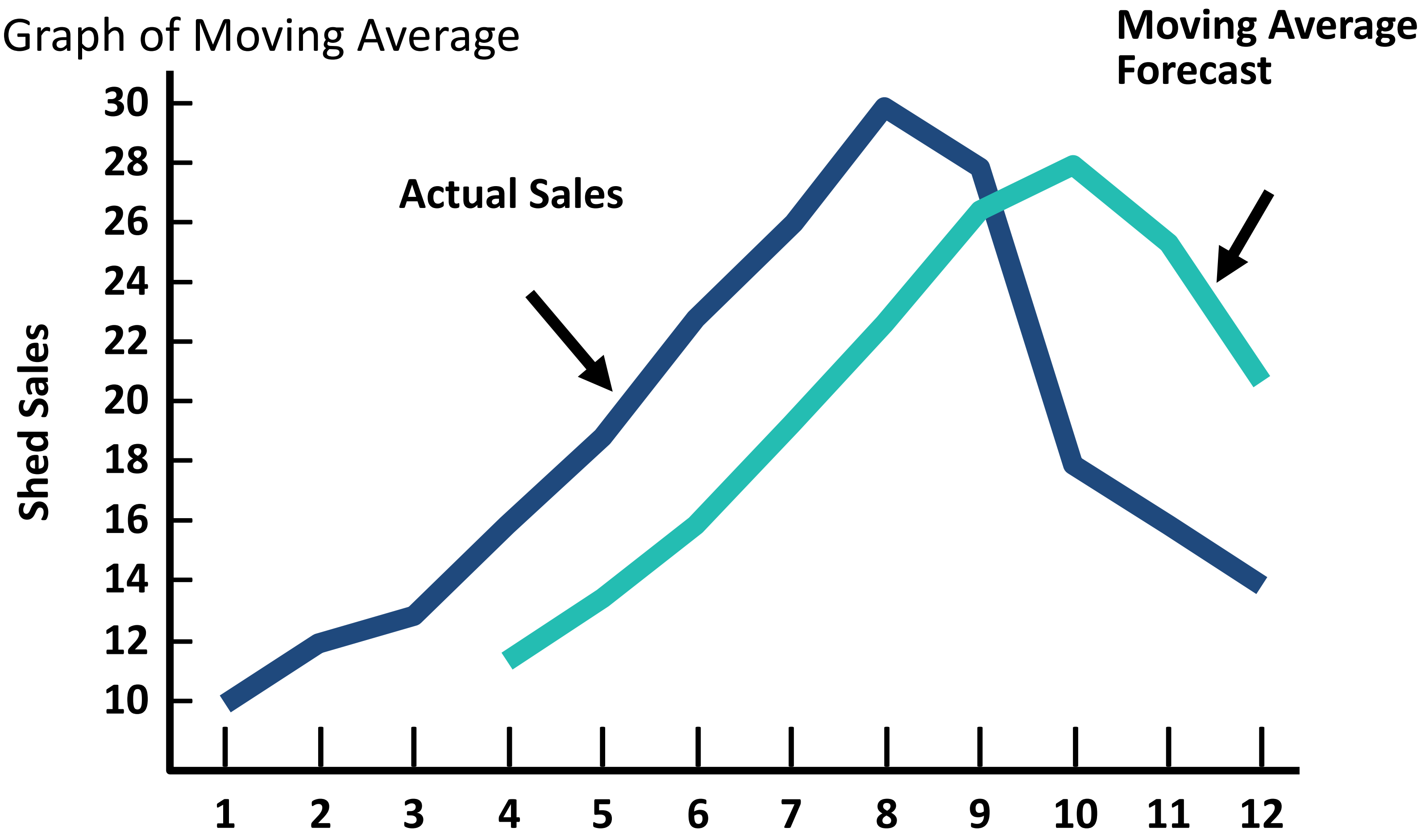
Time-Series: Moving Average Method

- Example

Month	Actual Shed Sales	3-Month Moving Average
January	10	
February	12	
March	13	
April	16	$(10 + 12 + 13)/3 = 11\ ^2/_3$
May	19	$(12 + 13 + 16)/3 = 13\ ^2/_3$
June	23	$(13 + 16 + 19)/3 = 16$
July	26	$(16 + 19 + 23)/3 = 19\ ^1/_3$

Time-Series: Moving Average Method

❖ Graph of Moving Average



MV is lagged behind actual data

Weighted Moving Average

- **Limitation of Moving Average:**
  - Weighs all periods equally → less responsive to trends
- **Weighted Moving Average:** used when some trend might be present
  - e.g. older data usually less important
- **Calculation:**

$$F_{t+1} = \sum w_t A_t$$

  - All weights must add to 1, e.g. $w_t = 0.5, w_{t-1} = 0.3, w_{t-2} = 0.2$
    - indicates more weight on recent data
- **Why WMA?**
  - Give more importance to what happened recently, without losing the impact of the past
  - Ability to vary the weights

Time-Series: Exponential Smoothing

- Use a weighted combination of last forecast and last actual value

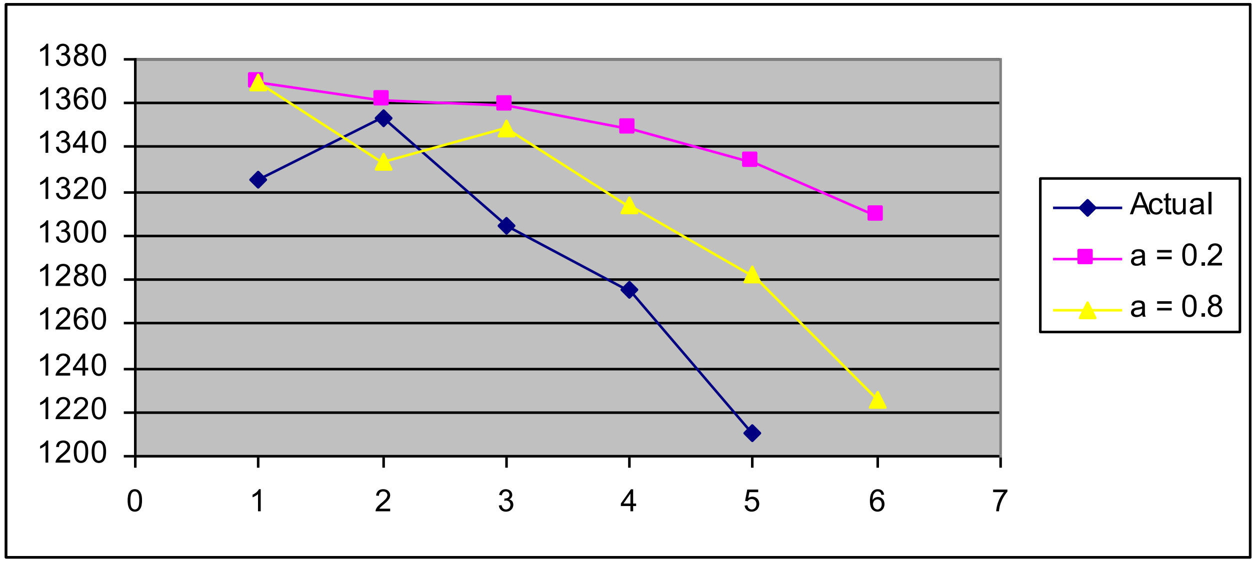
$$F_{t+1} = \alpha A_t + (1-\alpha)F_t$$

$$= \alpha A_t + \alpha(1-\alpha)A_{t-1} + \cdots + \alpha(1-\alpha)^t A_0 \qquad \text{where } \alpha \in [0,1]$$

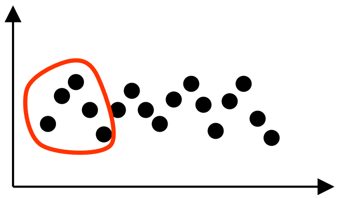
  - A form of weighted moving average:
    - Most recent data weighted most
    - Weights decline exponentially
- Why exponential smoothing?
  - More responsive to trend
  - Require minimum amount of data needed

Exponential Smoothing: α trade-off

- Trade-off between trend and random variation: $F_{t+1} = F_t + \alpha(A_t - F_t)$
  - Higher α values (e.g. 0.7 or 0.8) may place too much weight on last period's random variation
    - Good at capturing long-term trend
    - Sensitive to short-term fluctuations



Time-series: Seasonality



- **Seasonality**: regular pattern of up and down fluctuations
  - typically over 1 year
  - e.g. weather, customs
- E.g. A university must develop forecasts for the next year's quarterly enrollments. It has collected quarterly enrollments for the past two years. What is the forecast for each quarter of next year?

Quarter	Year 1	Year 2	Year3
Fall	24000	26000	?
Winter	23000	22000	?
Spring	19000	19000	?
Summer	14000	17000	?

Time-series: Seasonality

- The multiplicative seasonal model can adjust trend data for seasonal variations in demand (jet skis, snow mobiles)

- **Steps:**
  1. Calculate the average demand per season
     - E.g.: average quarterly demand
  2. Calculate a seasonal index for each season of each year:
     - Divide the actual demand of each season by the average demand per season for that year
  3. Average the indexes by season
     - E.g.: take the average of all Spring indexes, then of all Summer indexes, ...
  4. Forecast demand for the next year & divide by the number of seasons
     - Use regular forecasting method & divide by four for average quarterly demand
  5. Multiply next year's average seasonal demand by each average seasonal index
     - Result is a forecast of demand for each season of next year

Time-series: Seasonality

- Step 1. Calculate the average demand for each year
- Step 2. Calculate seasonal indexes
- Step 3. Average the indexes
- Step 4. Forecast demand for the next year
- Step 5. Multiple next year's average seasonal demand by each average seasonal index

Quarter	Year 1	Seasonal Index	Year 2	Seasonal Index	Avg. Index	Year3
Fall	24000	**1.2**	26000	**1.24**	**1.22**	**26840**
Winter	23000	**1.15**	22000	**1.05**	…	**…**
Spring	19000	**0.95**	19000	…		
Summer	14000	**0.7**	17000	…		
Average	**20000**		**21000**			**22000**

Forecasting: Causal Model

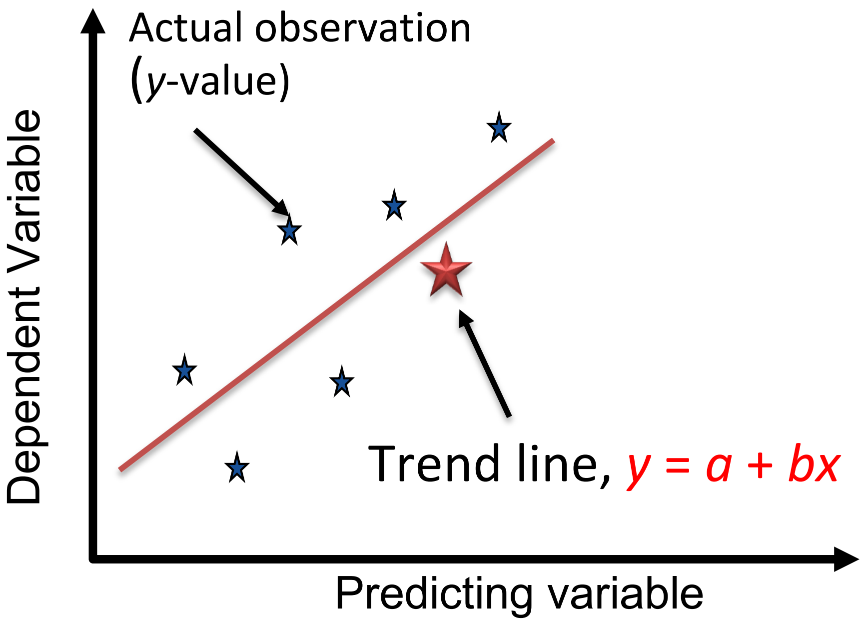
- Causal models establish a cause-and-effect relationship between independent and dependent variables

A maker of golf shirts has been tracking the relationship between sales and advertising dollars. Use linear regression to find out what sales might be if the company invested $53,000 in advertising next year.

	Sales $ (Y)	Adv.$ (X)
1	130	32
2	151	52
3	150	50
4	158	55
5	?	53

Causal Model: Approach

- Causal models establish a cause-and-effect relationship between independent and dependent variables
- Typical approach: regression model



Infer the regression variables:

$$b = \frac{\sum xy - n\bar{x}\bar{y}}{\sum x^2 - n\bar{x}^2}$$

$$a = \bar{y} - b\bar{x}$$

3. Forecasting: Evaluation

- Mean Absolute Deviation (MAD)
  - Measures the total error in a forecast without regard to sign
  - Higher MAD implies worse performance

$$\mathbf{MAD} = \frac{\sum \left|\mathbf{actual} - \mathbf{forecast}\right|}{\mathbf{n}}$$

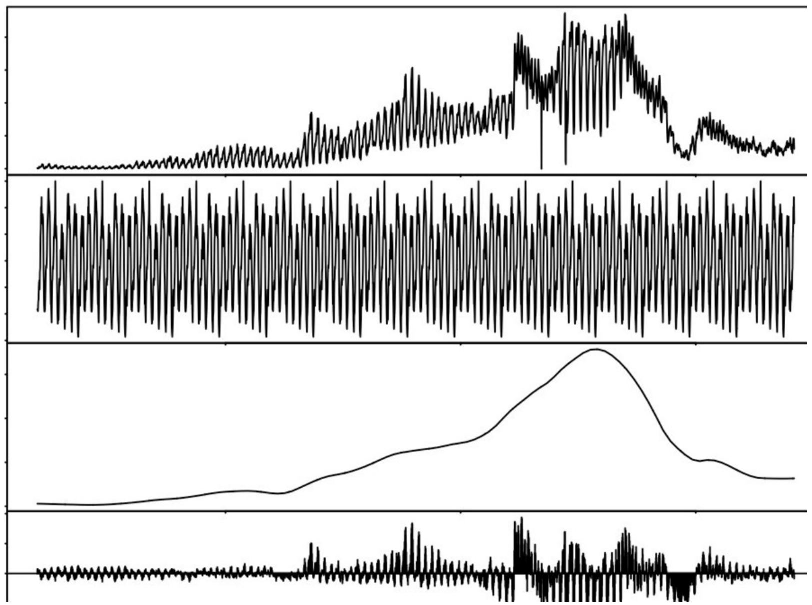
- Mean Square Error (MSE)
  - Penalizes larger errors

$$\mathbf{MSE} = \frac{\sum \left(\mathbf{actual\ -\ forecast}\right)^2}{\mathbf{n}}$$

V. Decomposition

Time Series Decomposition

- Sometimes, a time series is too complex for segmentation, classification, or forecasting
  - It is better to understand short-term, long-term and recurring patterns first
  - **Approach**: decompose a time series into several components, each representing one of the underlying patterns.



Original time series =

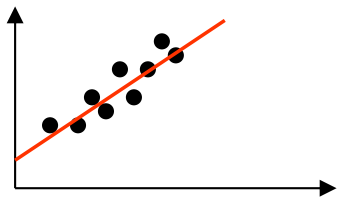
Seasonality component +

Trend component +

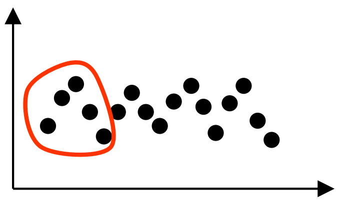
Residue component

3 components of time series

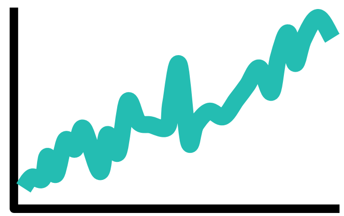
- **Trends:** persistent, overall upward and downward pattern
  - typically over several years
  - e.g. population, technology, age, culture



- **Seasonality:** regular pattern of up and down fluctuations
  - typically over 1 year
  - e.g. weather, customs

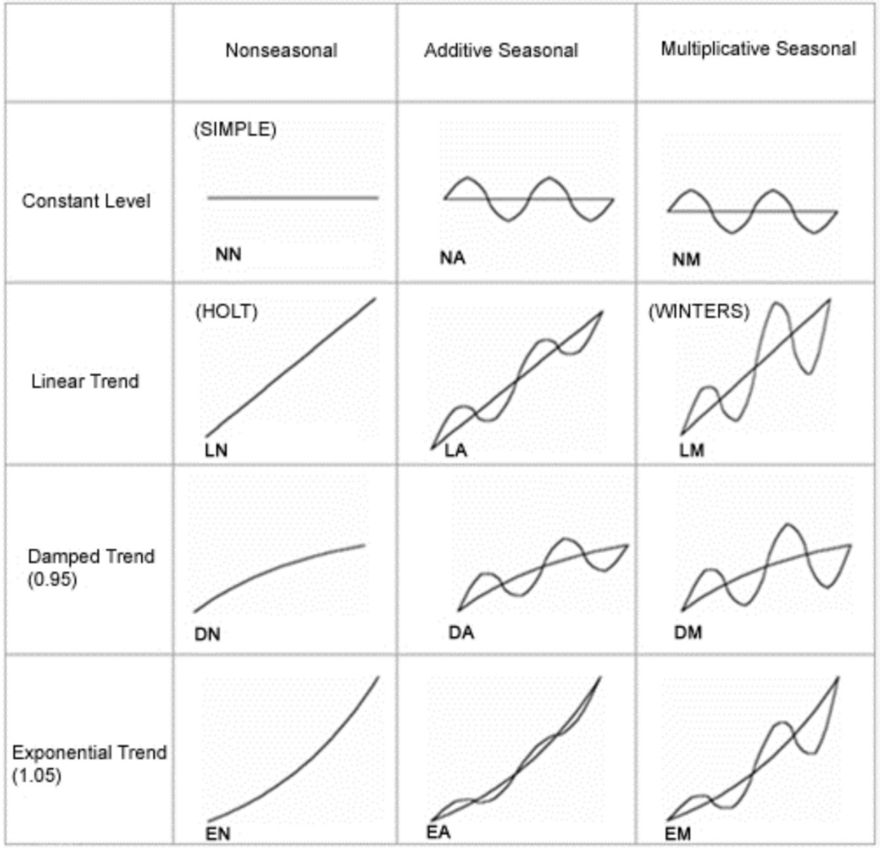


- **Random variation:** erratic, unsystematic, “residual” fluctuations
  - unforeseen events (stocks)
  - Short duration and norepeating



Trend

- ❖ Denote an evolution of the average value over time.
- ❖ Types of trend:
  - ➢ Constant
  - ➢ Linear
  - ➢ Exponential
  - ➢ Logarithmic (aka dampened)
- ❖ **Approach**: use the appropriate regression function (e.g. linear, logistic, exponential …)



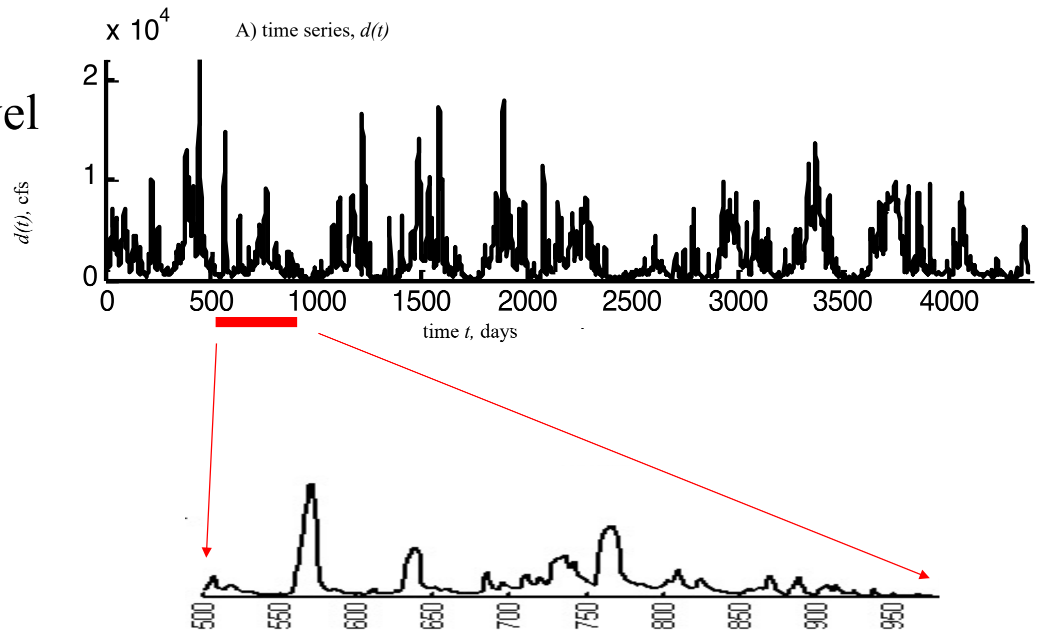
Find trend and seasonality in Python

```
from statsmodels.tsa.seasonal import seasonal_decompose
decomposition = seasonal_decompose(df['value'].ffill(axis=0), period=12, extrapolate_trend='freq')
```

Seasonality test by auto-correlation

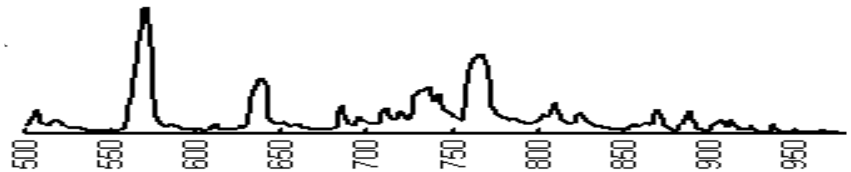
- The auto-correlation is defined as the correlation of the series over time
  - **Informal:** Does the future correlate with the past?
  - **Formal:** Does the value at time t depends on the value at time t−j for all j.

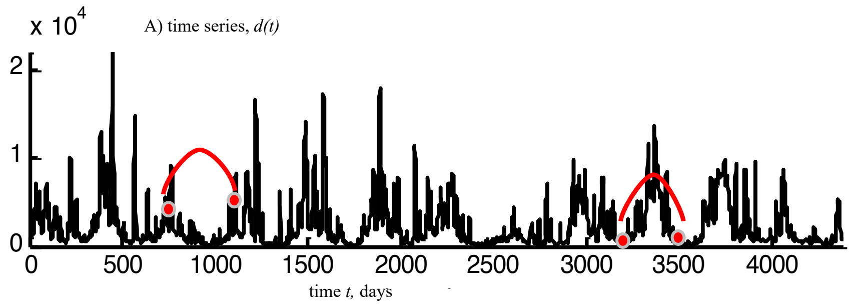
River Water Level



If the river is high, it usually stays high for a few days.
If it is low, it usually stays low for a few days.

Seasonality types



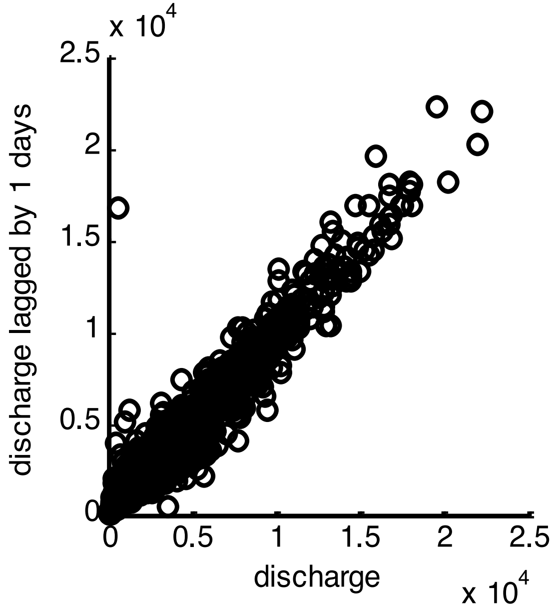


Short-term correlation
what ever the river was doing yesterday, its probably doing today, too
because water takes time to drain away

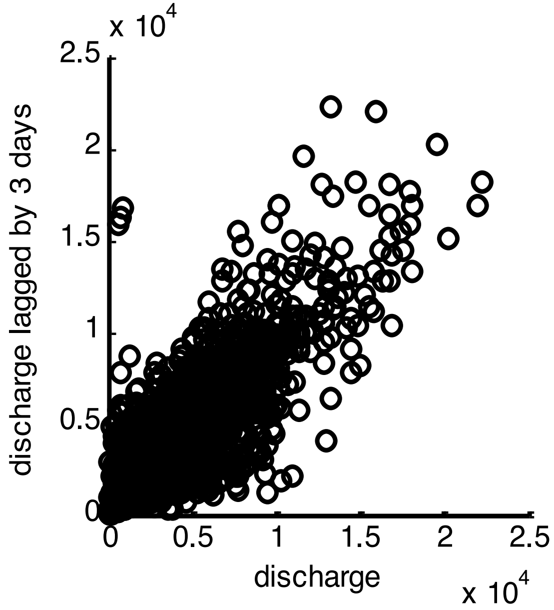
Long-term correlation
what ever the river was doing this time last year, its probably doing today, too
because seasons repeat

Auto Correlation with different periods

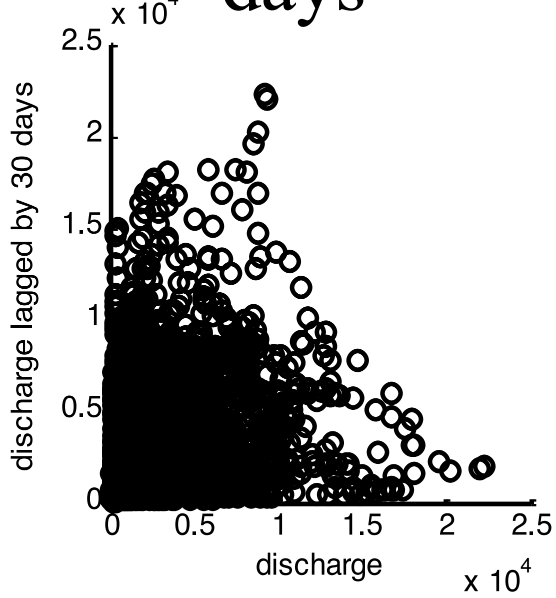
1 day



3 days



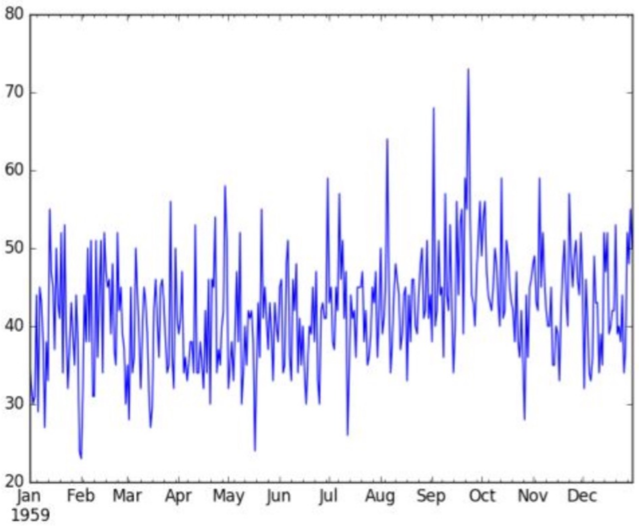
30 days



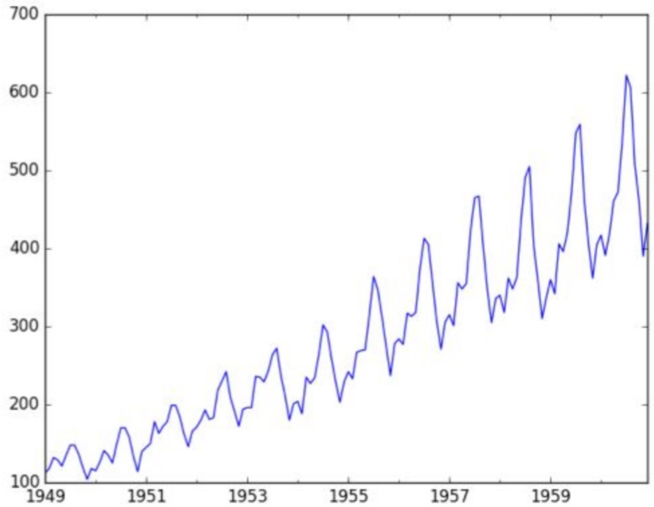
Help to recognize which period has short-term correlation

```
import statsmodels.api as sm
fig, axes = plt.subplots(1, 2, figsize=(15,8))
fig = sm.graphics.tsa.plot_acf(df['value'], lags=12, ax=axes[0])
```

When to decompose?



Stationary



Non-Stationary

A stationary time series is one whose statistical properties such as mean, variance, autocorrelation, etc. do not change over time

1. **If stationary:** can do forecasting, classification accurately
2. **If non-stationary:** better to decompose first:
   - Additive model: Time Series = Trend + Seasonality + Residual
   - The **residual component** is often stationary → work on it like case 1
   - Rebuild **the time series afterward.**

Test for stationarity

- Hard to conclude from the plots and basic statistics might not be enough
- **Approach:** Augmented Dickey-Fuller test
  - Model the timeseries and test if $\gamma = 1$ with the following equation:

$$\Delta y_t = \gamma y_{t-1} + \sum_{j=1}^{p} \delta_j \Delta y_{t-j} + \epsilon_t$$

  - If we confidence that $\gamma = 1$ to some degree then the time-series is non-stationary (the future can be only predicted with the last observation)

```
from statsmodels.tsa.stattools import adfuller

result = adfuller(df['value'].ffill(0))
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
```

```
ADF Statistic: 3.145186
p-value: 1.000000
Critical Values:
        1%: -3.466
        5%: -2.877
        10%: -2.575
```

If p-value > 0.05: non-stationary
If p-value <=0.05: stationary

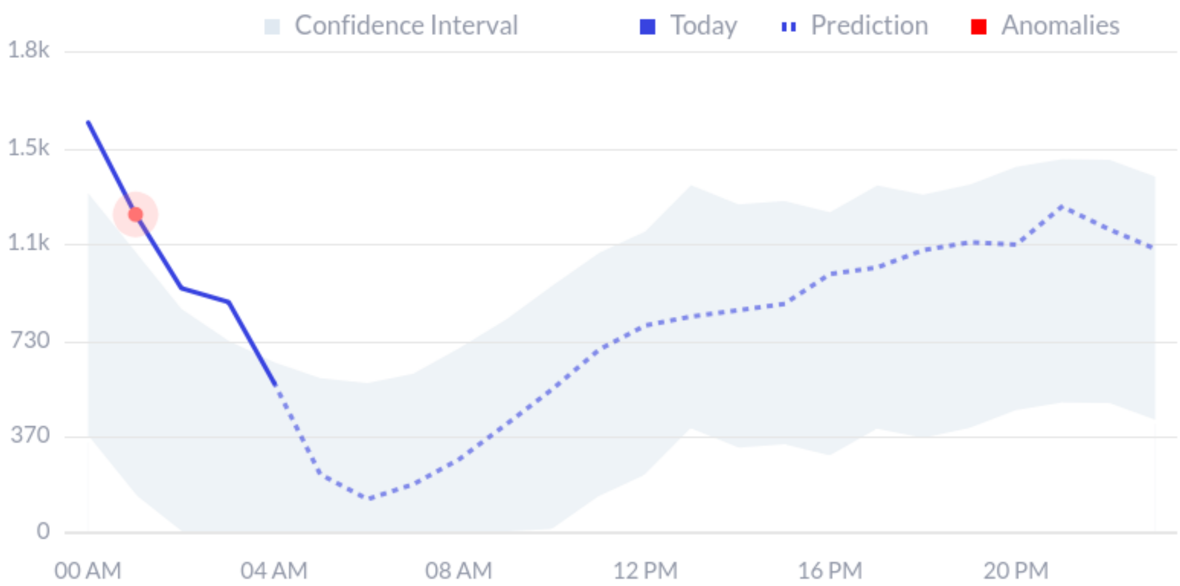
Forecasting with stationary time series

1. Moving Average (MA) process (of order q)
   - Current value is a mean value with some noises added up over time:
     $$y_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}$$
   - μ: mean, θ_i: regression parameters, ϵ_i: random white noises (generated from a normal distribution with mean zero and variance one)
2. Auto Regressive (AR) process (of order p)
   - Current value is a linear function of the observations at the prior time steps:
     $$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \epsilon_t$$
3. Auto Regressive Moving Average (ARMA) process with order (p,q)
   - Combine MA and AR processes:
     $$y_t = \mu + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$
4. Auto Regressive Integrated Moving Average (ARIMA) process
   - Combine MA and AR processes on the differencing version of time series:
     $$\Delta^d y_t = \mu + \phi_1 \Delta y_{t-1} + \cdots + \phi_p \Delta y_{t-p} + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$
   - $\Delta y_t = y_t - y_{t-1}$. Hyperparameter: p,q,d
   - Working on Δy_t makes sure the model is stationary

VI. Anomaly Detection

Anomaly Detection in Time Series

- Finding **outlier** data points relative to some standard or usual pattern.
  - Such as unexpected spikes, drops, trend changes and level shifts.
  - Basically, an anomaly detection algorithm should either **label** each time point with *anomaly/not anomaly*, or **forecast** a signal for some point and test if this point value varies from the forecasted enough to deem it as an anomaly.



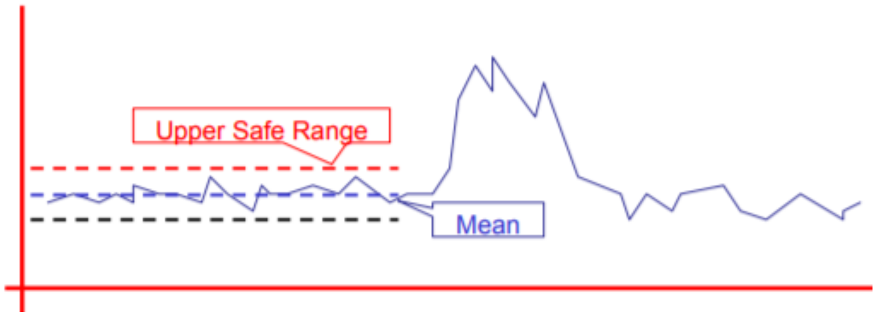
Examples:

- Growth of users in a short period of time that looks like a spike.
- When your server goes down and you see zero or a really low number of users for some short period of time.

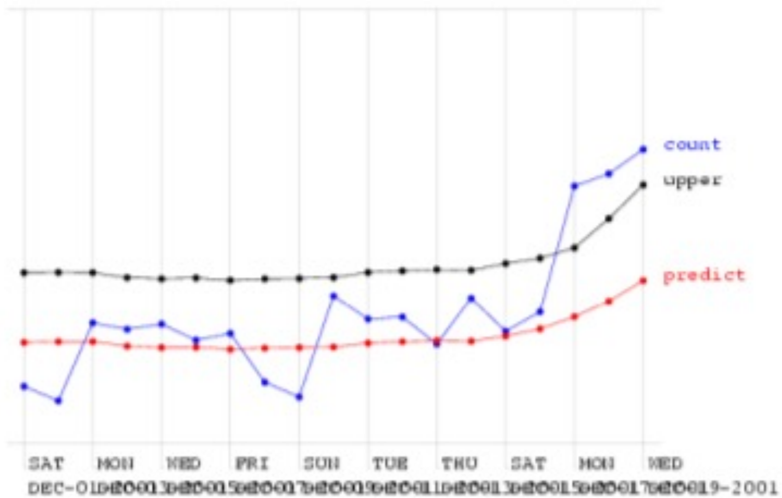
Anomaly Detection Methods

- **Simple Statistic Method:** The simplest approach to identifying irregularities in data is to flag the data points that deviate from common statistical properties of a distribution, including mean, median, mode, standard deviation, and quantiles.

Easy but cannot cope with trend and seasonality.

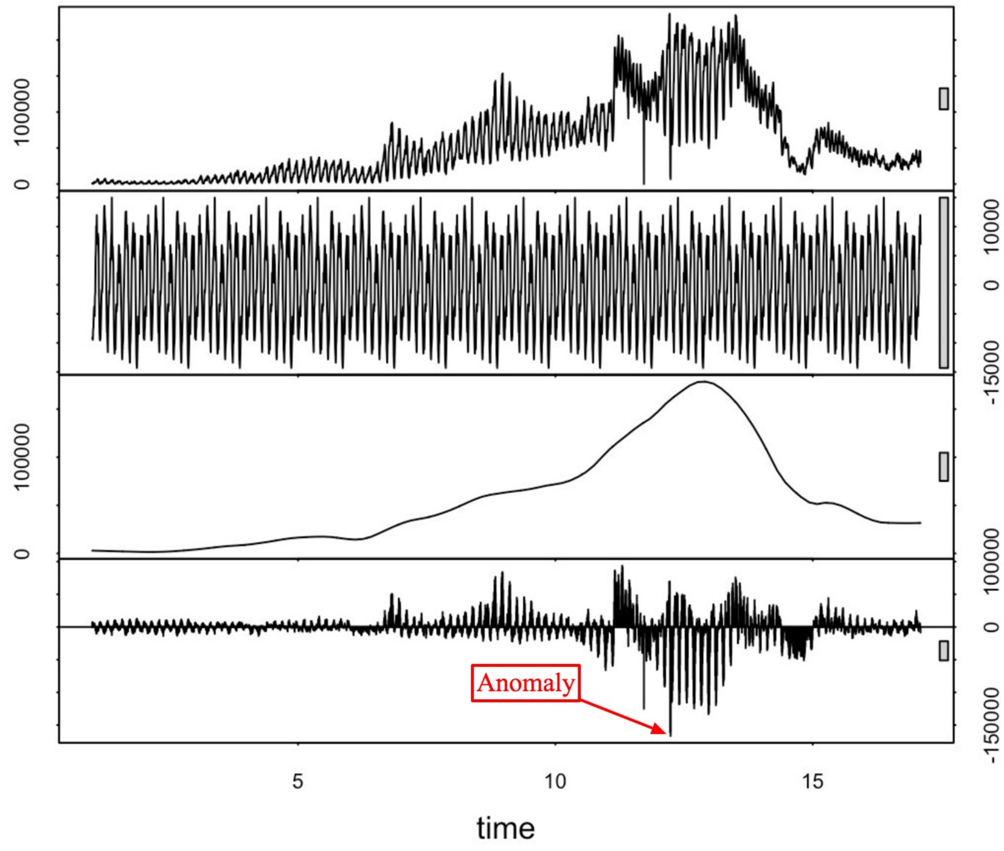


Use **Moving Average** Instead.



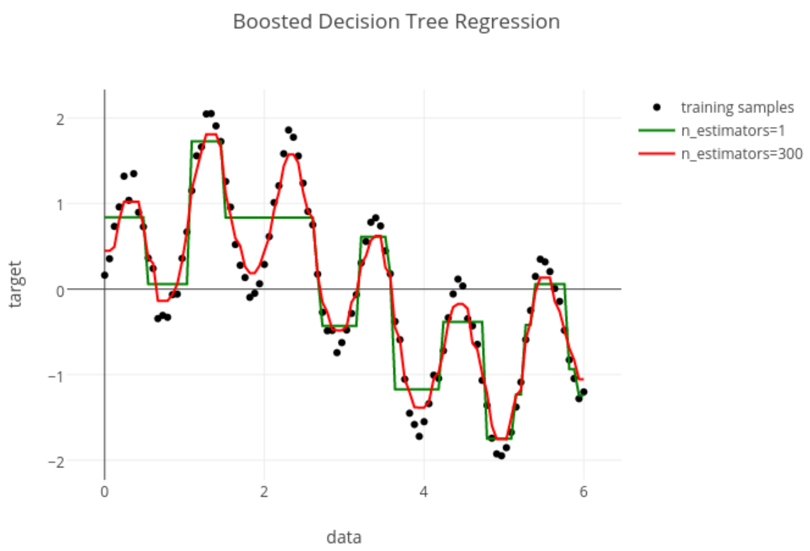
Anomaly Detection Methods

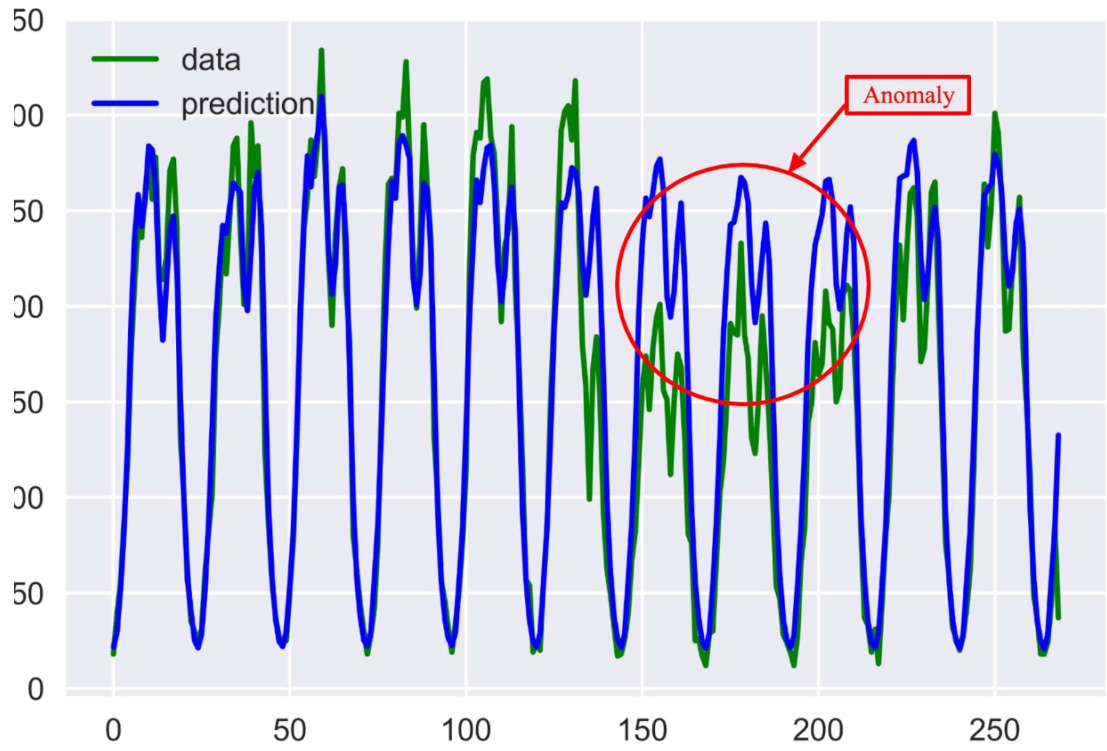
- **STL decomposition:** Seasonal-Trend decomposition.
  - Deconstructs a time series into several components, each representing one of the underlying categories of patterns.
  - STL decomposes the original time series into tree parts:
    - (1) Seasonal
    - (2) Trend
    - (3) Residue



Anomaly Detection Methods

- **Classification and Regression Trees**
  - Use supervised learning to teach trees to classify anomaly and non-anomaly data points.
  - Use unsupervised learning to predict the next data point in the series and have some confidence interval.





Summary: Time Series Analysis

- **Exploratory data analysis:** examine a time series with a line chart or statistics.
- **Segmentation:** Splitting a time-series into a sequence of segments. Represent a time-series as a sequence of individual segments, each with its own characteristic properties.
- **Classification:** Assigning time series pattern to a specific category → Natural phenomena, astronomical phenomena, animal movement.
- **Forecasting:** is the use of a model to predict future values based on previously observed values.
- **Decomposition:** deals with complex timeseries to help forecasting easier
- **Anomaly Detection:** Finding outlier data points relative to some standard or usual pattern.

References

[1] Yan, Zhixian, et al. "SeMiTri: a framework for semantic annotation of heterogeneous trajectories." Proceedings of the 14th international conference on extending database technology. ACM, 2011.

[2] Bagnall, A., Lines, J., Bostrom, A. et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Min Knowl Disc (2017)

[3] http://timeseriesclassification.com/index.php

[4] https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc4.htm

[5] http://www.statgraphics.com/time-series-analysis-and-forecasting

[6] https://www.wessa.net/download/stl.pdf

[7] Yu Zheng. Trajectory Data Mining: An Overview. ACM Transactions on Intelligent Systems and Technology. 2015, vol. 6, issue 3.

[8] Yu Zheng, et al. Understanding Mobility Based on GPS Data. UbiComp 2008

[9] https://www.ldeo.columbia.edu/users/menke/edawm/eda_lectures/lec17.pptx

[10] https://slideplayer.com/slide/5994126/

[11] http://cs-people.bu.edu/evimaria/cs565-10/lect19.pdf

[12] https://developer.ibm.com/learningpaths/get-started-time-series-classification-api/what-is-time-series-classification

[13] https://subscription.packtpub.com/book/data/9781789346466/8/ch08lvl1sec54/the-augmented-dickey-fuller-test

[14] https://www.machinelearningplus.com/time-series/augmented-dickey-fuller-test/

[15] https://otexts.com/fpp2/AR.html