Time Series Data, Part 2

Ivan Corneillet

Data Scientist

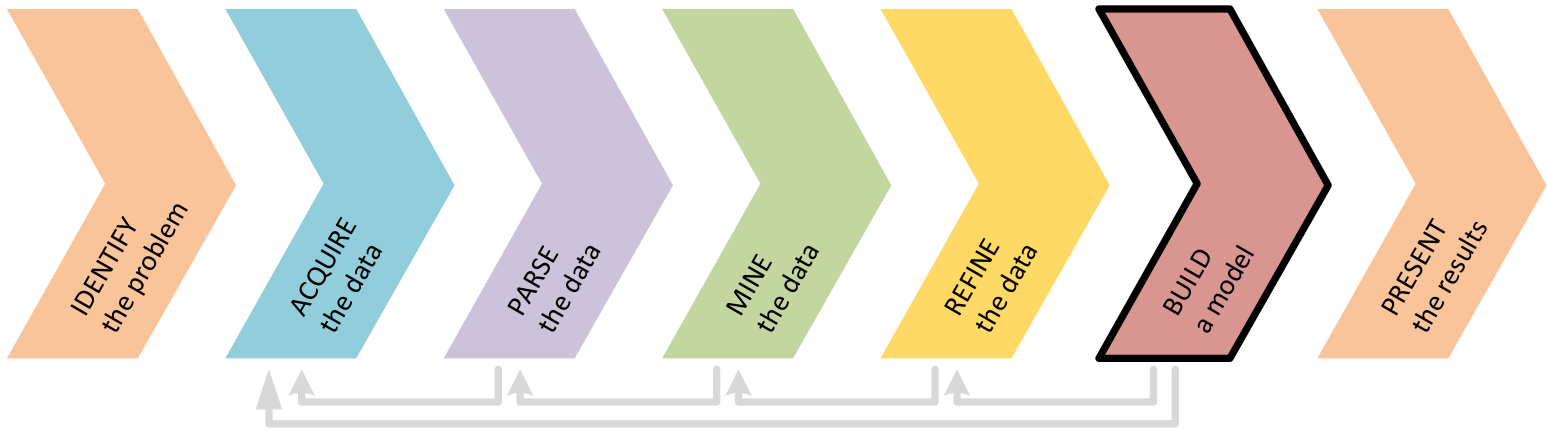
GENERAL ASSEMBLY

Final Project Countdown

Final Project, Part 3	Due today
Final Project, Part 4	April 26; due in 1 week
Final Project, Part 5	April 28; due in 1.5 weeks

In the last class, we focused on exploring time series data and common statistics for time series analysis. In this class, we will advance those techniques to show how to predict or forecast forward from time series data

Unit 1 – Research Design and Data Analysis	*Research Design*	*Data Visualization in Pandas*	*Statistics*	*Exploratory Data Analysis in Pandas*
Unit 2 – Foundations of Modeling	Linear Regression	Classification Models	Evaluating Model Fit	Presenting Insights from Data Models
Unit 3 – Data Science in the Real World	*Decision Trees and Random Forests*	*Time Series Data*	*Natural Language Processing*	*Databases*



Learning Objectives

After this lesson, you should be able to:

- Model and predict from time series data using AR, ARMA, or ARIMA models
- Specifically, coding these models in *statsmodels*

Outline

- Review
- Time Series Modeling
  - Training, validation, and testing sets
  - Autocorrelation
- Stationarity
- Autoregressive (AR) Models
- Moving Average (MA) Models
- ARMA Models
- ARIMA Models
- Codealong
- Lab
- Office hours in class for final projects
- Review

Review

Review

- We use time series analysis to identify changes in values over time
- We want to identify whether changes are true trends or seasonal changes
- Rolling means give us a local statistic of an average in time, smoothing out random fluctuations and removing outliers
- Autocorrelations are a measure of how much a data point is dependent on previous data points

Pre-Work

Pre-Work

Before this lesson, you should already be able to:

- Prior definition and Python functions for moving averages and autocorrelation
- Prior exposure to linear regression with discussion of coefficients and residuals

Time Series Modeling

Time Series Modeling

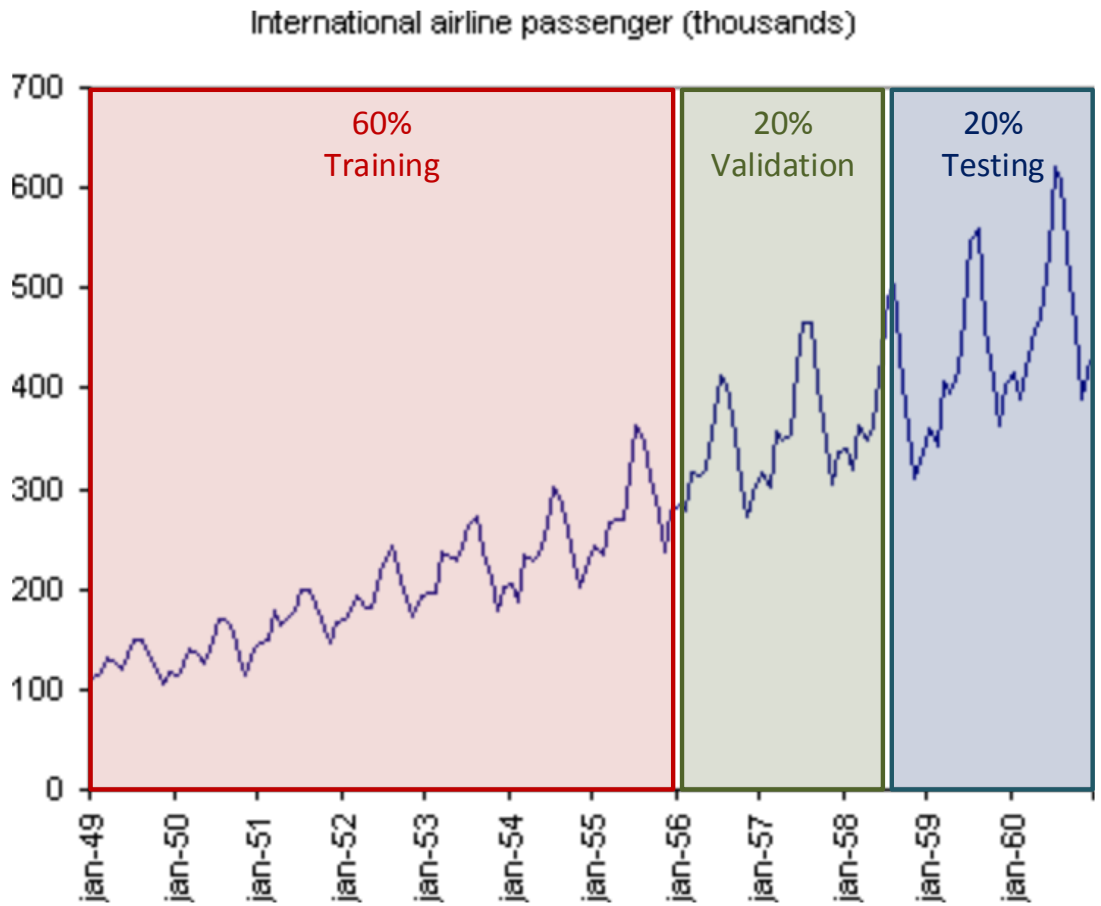
- There are many times when you may want to use a time series to predict a future value. E.g.,
  - Next month's sales
  - Anticipated website traffic when buying a server
  - Financial forecasting
  - Number of visitors to your store during the holidays

Time series models are models that will be used to predict a future value in the time series

- **Like** other predictive models, we will use prior history to predict the future
- **Like** previous modeling exercises, we will have to evaluate the different types of models to ensure we have chosen the best one
  - We will want to evaluate on a held-out set or test data to ensure our model performs well on unseen data
- **Unlike** previous models, we will use the earlier in time outcome variables as inputs for predictions
- **Unlike** previous modeling exercises, we won't be able to use standard cross-validation for evaluation
  - Since there is a time component to our data, we cannot choose training and test examples at random

Training, validation, and testing sets

- Instead, we will exclusively train on values earlier (in time) in our data and test our model on values at the end of the data period



Autocorrelation (cont.)

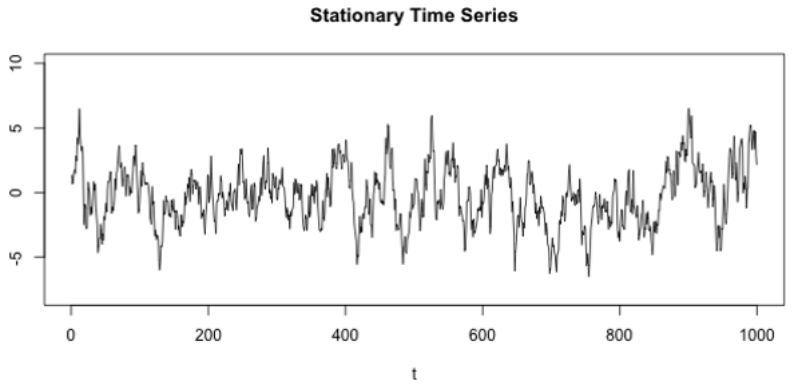
- *Autocorrelation* is how correlated a variable is with itself. Specifically, how related are variables earlier in time with variables later in time

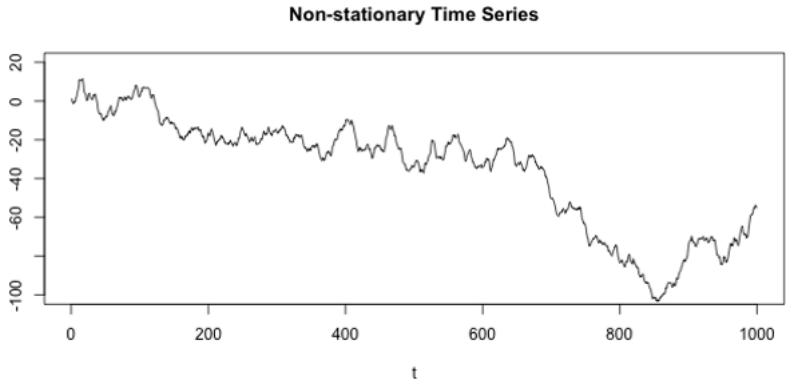
- Typically, for a high quality model, we require some autocorrelation in our data
- We can compute autocorrelation at various lag values to determine how far back in time we need to go

Stationarity

Many models make an assumption of *stationarity*, assuming the mean and variance of our values is the *same* throughout

- E.g., while sales may shift up or down over time, the mean and variance of sales is constant. I.e., there aren't many dramatic swings up or down
- These assumptions may not represent real world data; we must be aware of that when we are breaking the assumptions of our model

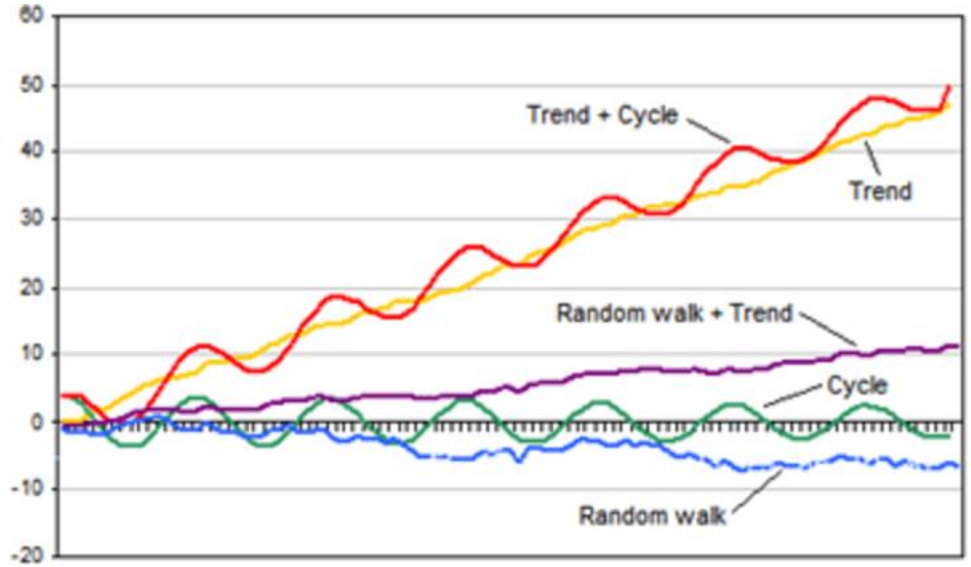




E.g., typical stock or market performance is not stationary. In this plot of the S&P 500 performance since 1993, the mean is increasing over time

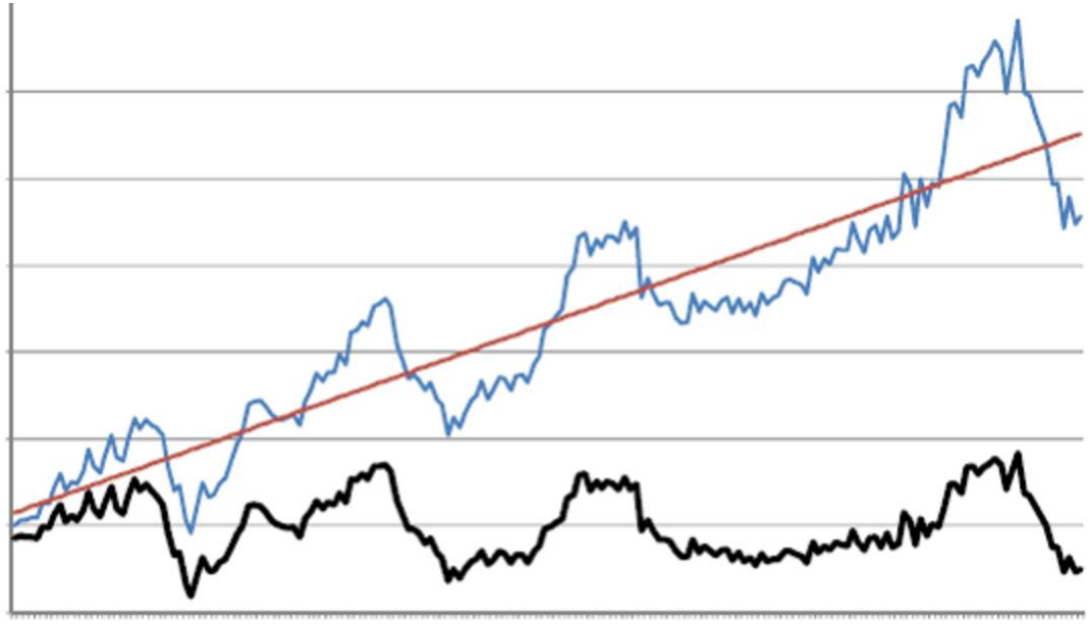
Non-stationary time series

- E.g., some simulated examples of non-stationary time series



By removing major trends in the data, *detrending* can make time series stationarity

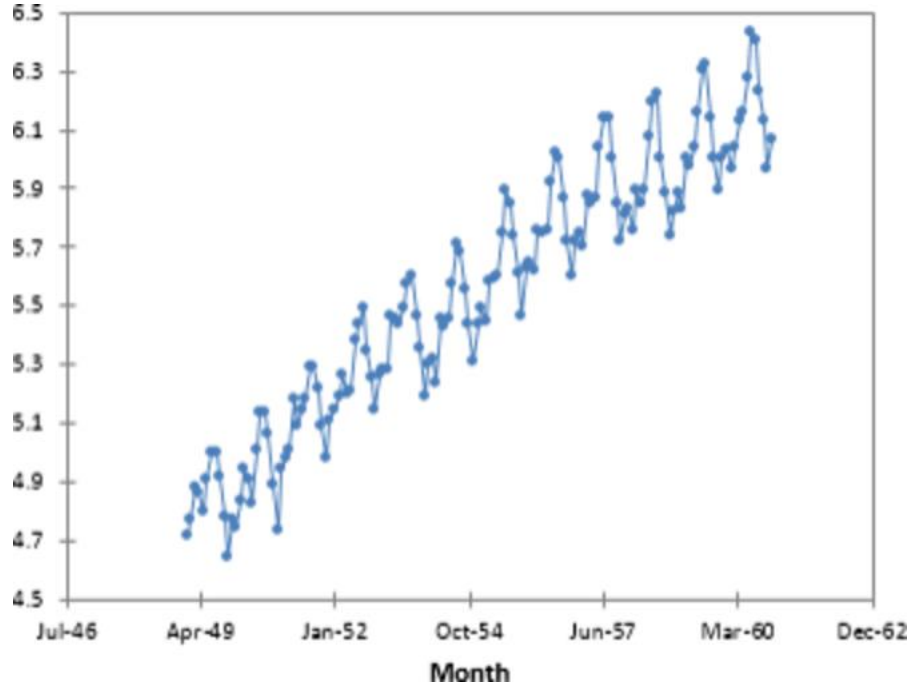
- The simplest way to do this is to fit a line to the trend and make a new series that is the difference between the line and the true series



A simpler method is *differencing*: Instead of predicting the series (again our non-stationary series), we can predict the difference between two consecutive values

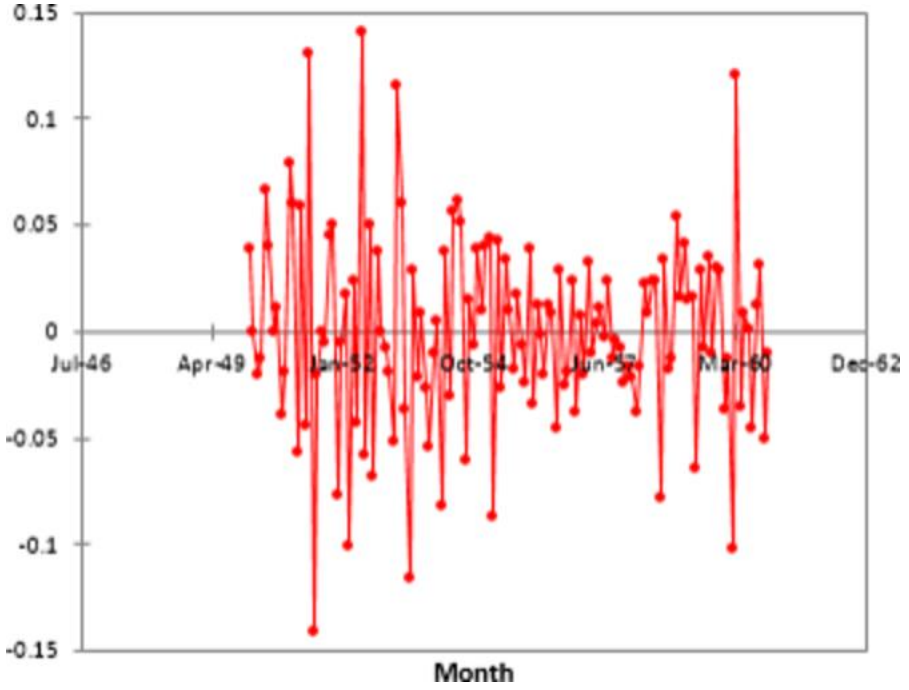
Before differencing

(non-stationary series)



After differencing

(stationary series)



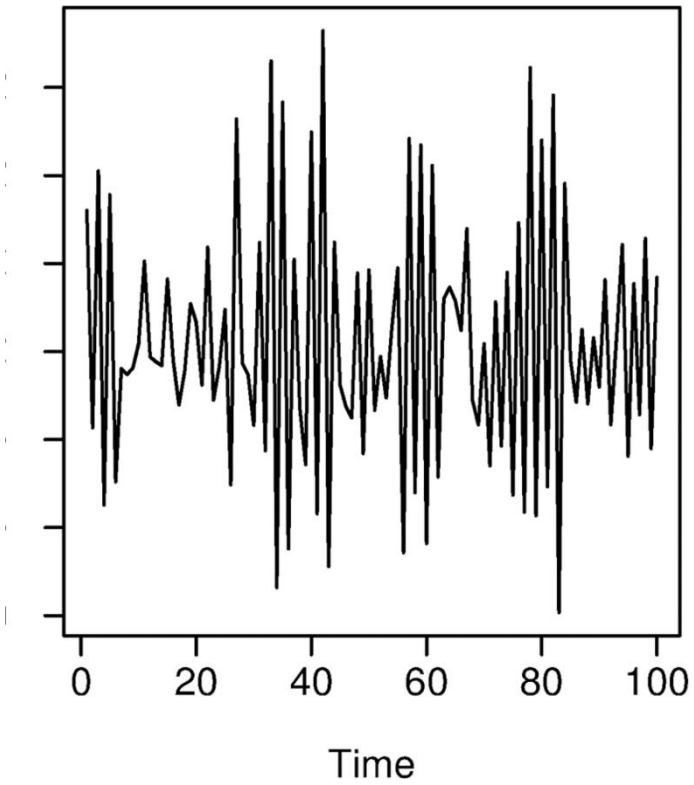
Autoregressive (AR) Models

Autoregressive (AR) models use data from previous time points to predict the next

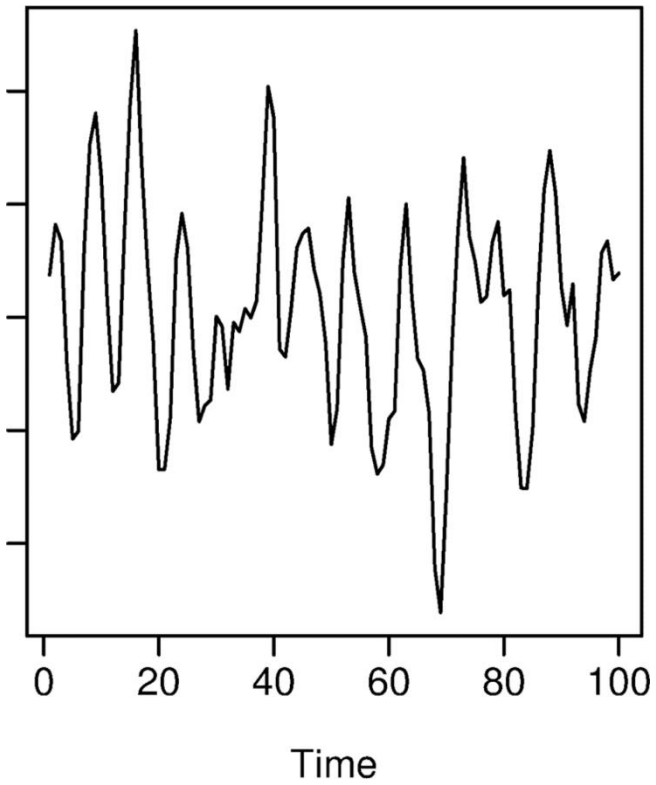
- This is very similar to previous regression models, except as input, we take the previous outcome
- E.g., if we are attempting to predict weekly sales, we use the sales from a previous week as input

Typically, AR models are denoted $AR(p)$ where p indicates the number of previous time points to incorporate; $AR(1)$ is the most common

$AR(1)$



$AR(2)$



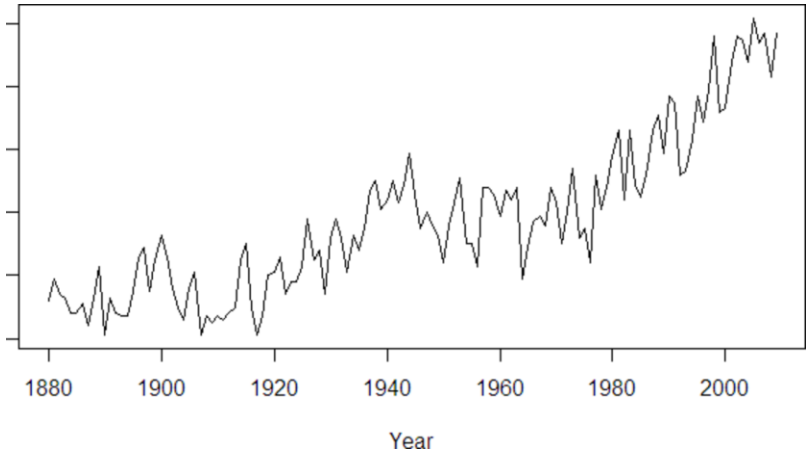
In an autoregressive model, we are learning regression coefficients for each of the p previous values. Therefore, we will learn p coefficients (or β) (+ intercept)

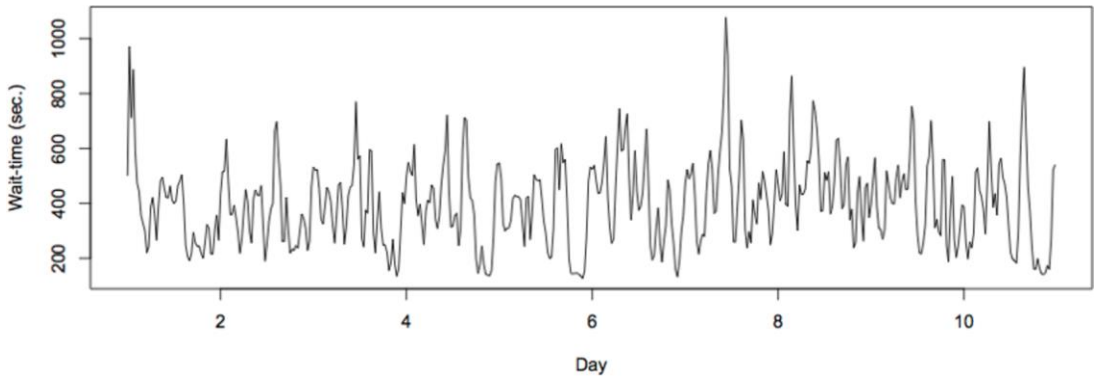
$$y_t = \beta_0 + \beta_1 y_{t-1} + \cdots + \beta_p y_{t-p} + \varepsilon_t$$

- As with standard regression, our model assumes that each outcome variable is a linear combination of the inputs and a random error term
- As with other models, interpretation of the model becomes more complex as we add more factors
- A model with high autocorrelation implies that the data is highly dependent on previous values and an autoregressive model would perform well

For an AR(1) model, we will learn a single coefficient (+ intercept): $\hat{y}_t = \hat{\beta}_0 + \hat{\beta}_1 \hat{y}_{t-1}$

- $\hat{\beta}_1 > 1$ suggests growth over previous values. This would typically represent non-stationary data, since if we compound the increases, the values are continually increasing
- $-1 < \hat{\beta}_1 < 1$ suggests increasing and decreasing patterns from previous patterns





Autoregressive (AR) models (cont.)

- Autoregressive models are useful for learning falls or rises in our series
- This will weight together the last few values to make a future prediction
- Typically, this model type is useful for small-scale trends such as an increase in demand or change in tastes that will gradually increase or decrease the series

Moving Average (MA) Models

Moving average (MA) models, as opposed to AR models, do not take the previous outputs (or values) as inputs; they take the previous error terms

- MA models attempt to predict the next value based on the overall average and how off our previous predictions were
- AR models slowly incorporate changes in the system by combining previous values; MA models use prior errors to quickly incorporate changes
- This model is useful for handling specific or abrupt changes in a system, e.g., something going out of stock or a sudden rise in popularity affecting sales

As in AR models, we have an order term, q, and we refer to our model as $MA(q)$. The moving average model is dependent on the last q errors

$$y_t = mean + \beta_1 y_{t-1} + \cdots + \beta_q y_{t-q} + \varepsilon_t$$

- We include the mean of the time series (that's why it's called a moving average…) as we assume the model takes the mean value of the series and randomly jumps around it

Of course, we don’t have error terms when we start – where do they come from?

- This requires a more complex fitting procedure than we have seen previously
- We need to iteratively fit a model (perhaps with random error terms), compute the errors, and then refit, again and again

ARMA Models

ARMA (pronounced ‘R-mah’) models combine the autoregressive (AR) and moving average (MA) models

- An $ARMA(p, q)$ model is simply a combination (sum) of an $AR(p)$ model and $MA(q)$ model
  - We specify two model settings, p and q
- Incorporating both models allows us to mix two types of effects:
  - AR models slowly incorporate changes. E.g., in preferences and tastes
  - MA models base their prediction on the prior error, allowing to correct sudden changes based on random events. E.g., supply and popularity spikes

ARIMA Models

ARIMA (pronounced ‘uh-ri-mah’) is an autoregressive integrated moving average model

- In this model, we learn an $ARMA(p, q)$ model to predict the difference of the series (as opposed to the value of the series)

$$y_t - y_{t-1} = ARIMA(p, q)$$

- This handles the stationarity assumption we wanted for our data. Instead of *detrending* or *differencing* manually, the model does this

An ARIMA model has in fact three parameters and is specified as $ARIMA(p, d, q)$

- p is the order of the autoregressive component
- q is the order of the moving average component
- d is the degree of differencing.
  - In our prior example, $d = 1$. For $d = 2$, our model would be

$$(y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) = ARIMA(p, 2, q)$$

ARIMA models (cont.)

- Compared to an ARMA model, ARIMA models do not rely on the underlying series being stationary
  - The differencing operation can convert the series to one that is stationary
- Instead of attempting to predict values overtime, our new series is the difference in values over time
  - Since ARIMA models include differencing, they can be used on a broader set of data without the assumption of a constant mean

Codealong

Review

Review

- Time-series models use previous values to predict future values, also known as forecasting
- AR and MA model are simple models on previous values or previous errors respectively
- ARMA combines these two types of models to account for both gradual shifts (due to AR models) and abrupt changes (MA models)
- ARIMA models train ARMA models on differenced data to account for non-stationary data
- Note that none of these models may perform well for data that has more random variation
  - For example, for something like iPhone sales (or searches) which may be sporadic, with short periods of increases, these models may not work well

Pre-Work

Pre-Work

Before the next lesson, you should already be able to:

- Install *gensim* with `conda install sqlite`
- Recall and apply dataframe manipulation in *pandas* (subsetting by rows and columns; grouping aggregation with *GroupBy*)

Q & A

Exit Ticket

Don’t forget to fill out your exit ticket here