GENERAL ASSEMBLY

TIME SERIES MODELLING

Angelo Klin
Katra Analytics

LEARNING OBJECTIVES

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

PRE-WORK

PRE-WORK REVIEW

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

```
conda install statsmodels
```

OPENING

TIME SERIES MODELLING

TIME SERIES MODELLING

- 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
- With a sequence of values (a time series), we will use the techniques in this class to predict a future value

TIME SERIES MODELLING

- There are many times when you may want to use a series of values to predict a future value

  - The number of sales in a future month
  - Anticipated website traffic when buying a server
  - Financial forecasting
  - The number of visitors to your store during the holidays

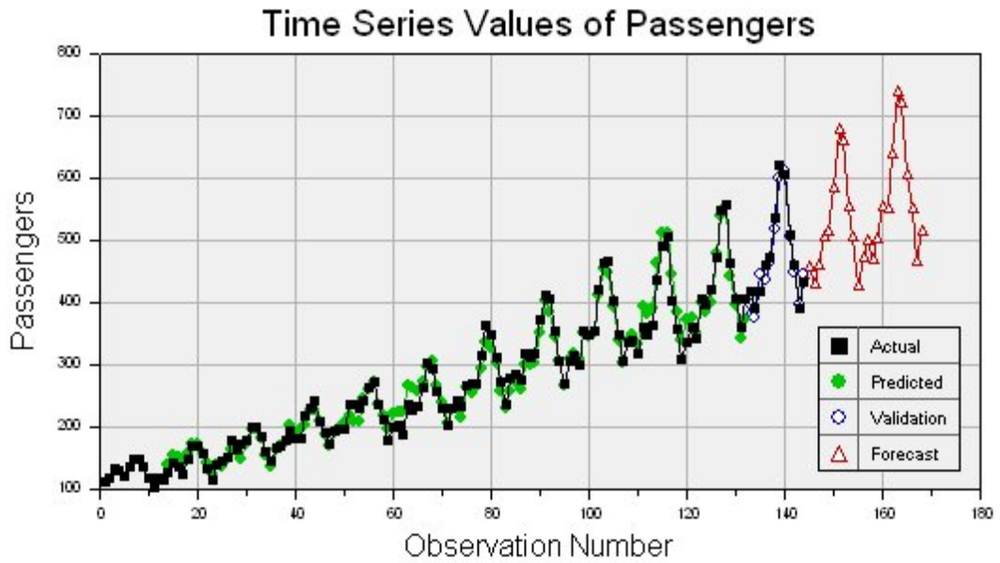
WHAT ARE TIME SERIES MODELS?

WHAT ARE TIME SERIES MODELS?

- 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

- **Unlike** previous models, we will use the earlier in time **outcome** variables as **inputs** predictions



WHAT ARE TIME SERIES MODELS?

- **Like** previous modelling 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

WHAT ARE TIME SERIES MODELS?

- **Unlike** previous modelling exercises, we will not 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
- Suppose we did select a random 80% sample of data points for training and a random 20% for testing
  - What could go wrong?

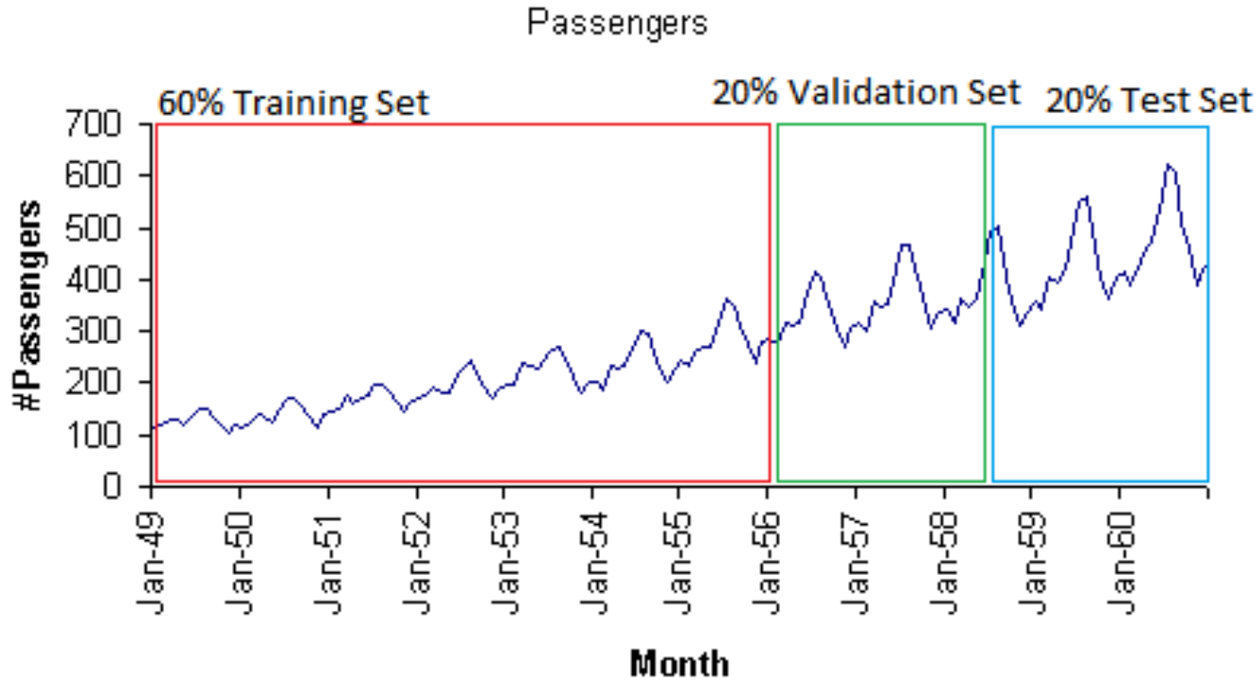
WHAT ARE TIME SERIES MODELS?

- The training data set would likely contain data from before AND after a test data set

- This would not be possible in real life (you can not use future, unseen data points when building your model)
  - Therefore, it is not a valid test of how our model would perform in practice

WHAT ARE TIME SERIES MODELS?

- 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



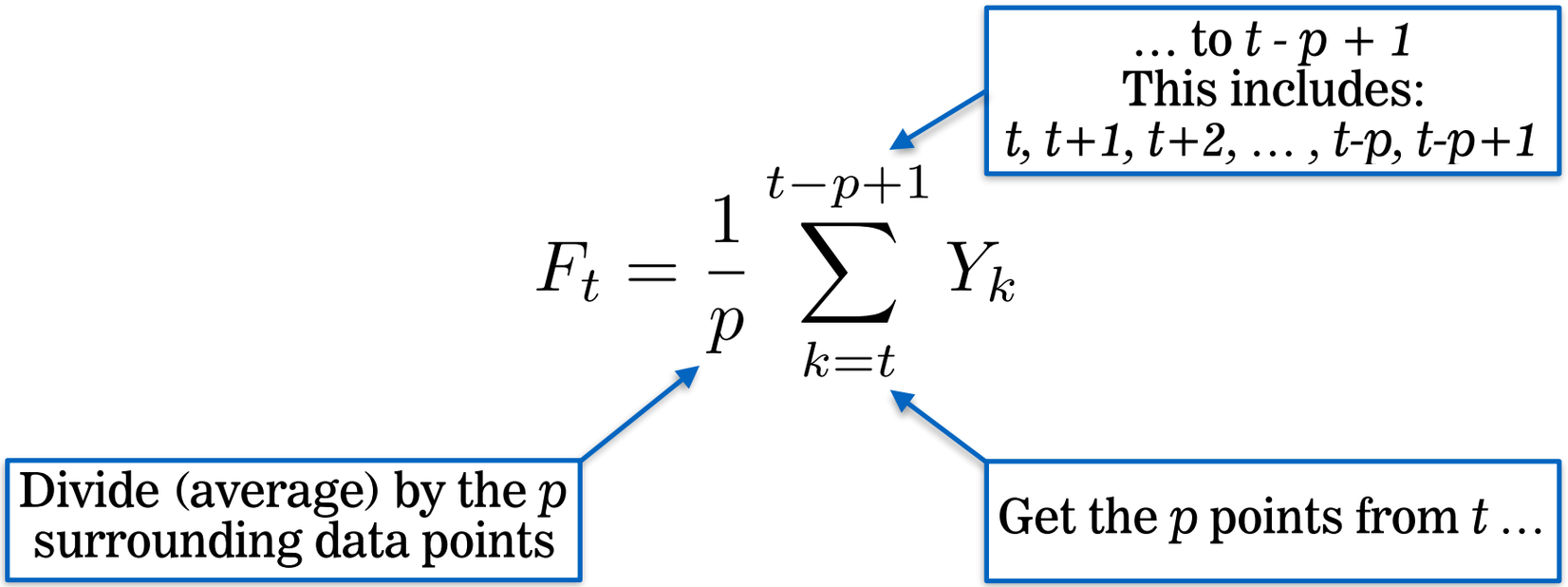
ACTIVITY: KNOWLEDGE CHECK

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

1. In our last class, we saw a few statistics for analysing time series
2. We looked at moving averages to evaluate the local behaviour of the time series
3. Redefine the moving average and its purpose

PROPERTIES FOR TIME SERIES PREDICTION

- A **moving average** is an average of p surrounding data points in time



ACTIVITY: KNOWLEDGE CHECK

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

1. We previously looked at auto-correlation to compute the relationship of the data with prior values
2. Recall the definition of autocorrelation and its purpose

PROPERTIES FOR TIME SERIES PREDICTION

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

$$r_k = \frac{\sum_{t=k+1}^{n} (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{n} (y_t - \bar{y})^2}$$

- We fix a lag, k, which is how many time points earlier we should use to compute the correlation

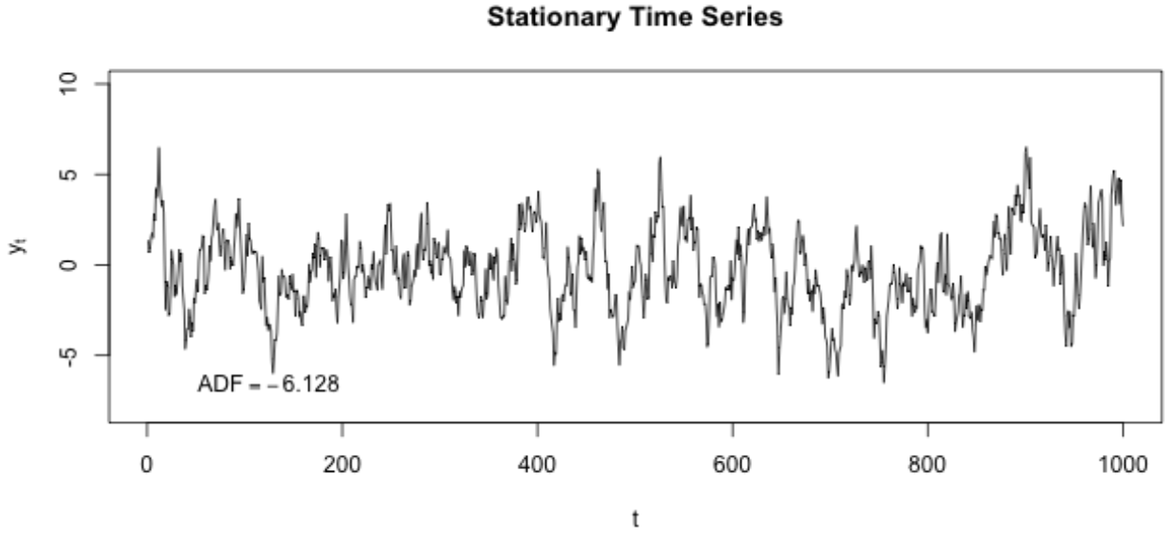
PROPERTIES FOR TIME SERIES PREDICTION

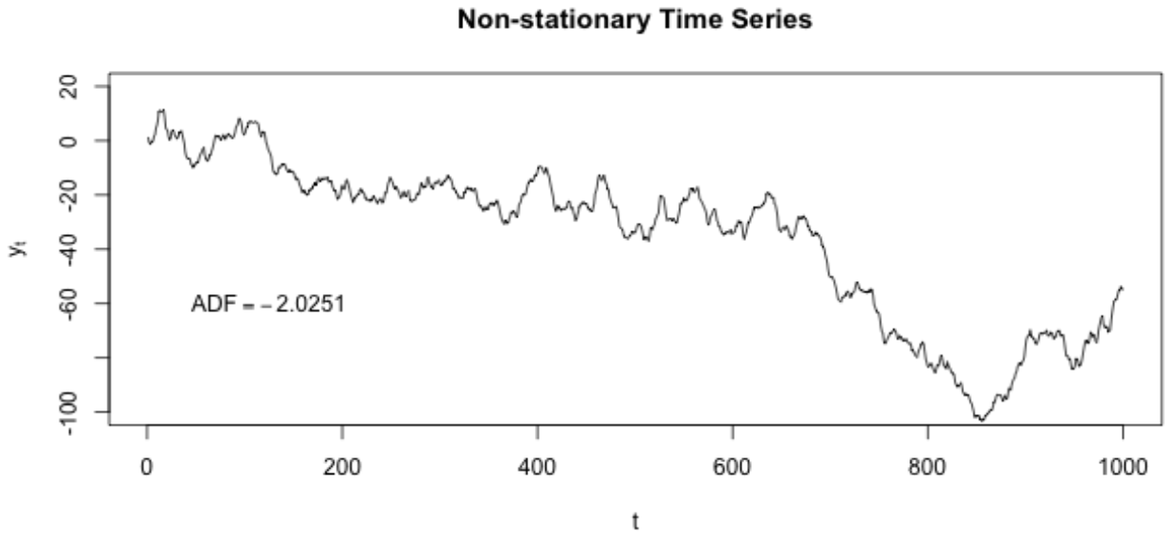
- We can use these values to assess how we plan to model our time series
- Typically, for a high quality model, we require some autocorrelation in our data
- We can compute autocorrelation at various lag values to determine how far back in time we need to go

PROPERTIES FOR TIME SERIES PREDICTION

- Many models make an assumption of stationarity, assuming the mean and variance of our values is the same throughout
- While the values (e.g. of sales) may shift up or down over time, the mean and variance of sales is constant (i.e. there are not 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

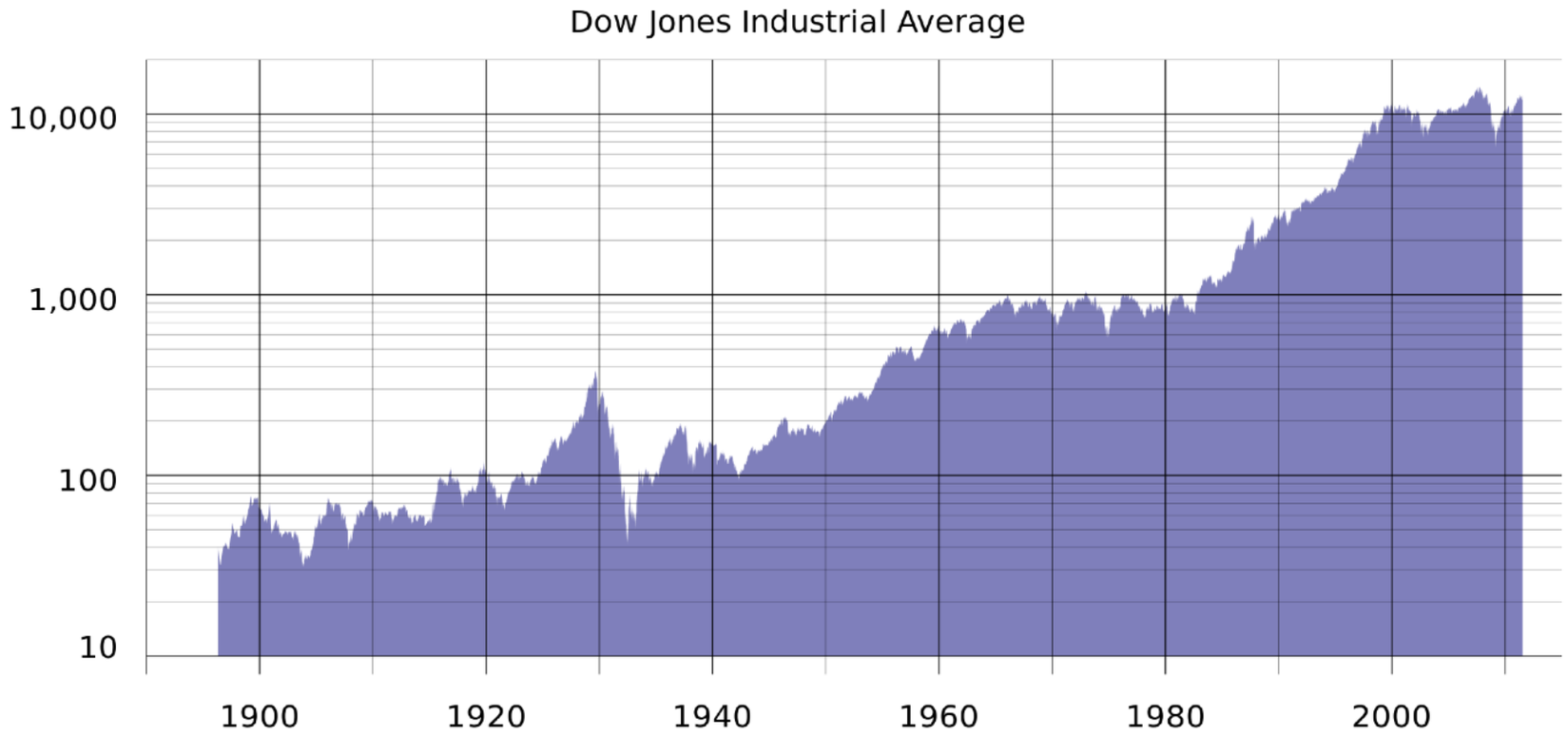
PROPERTIES FOR TIME SERIES PREDICTION





PROPERTIES FOR TIME SERIES PREDICTION

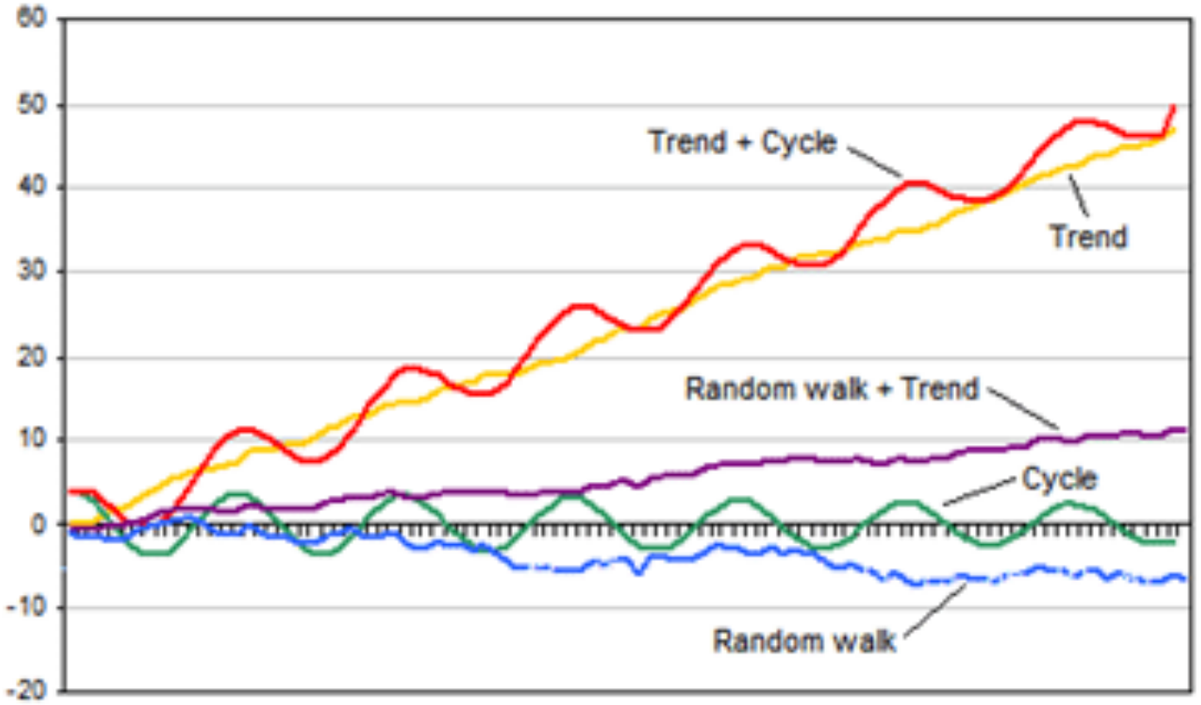
- For example, typical stock or market performance is not stationary
  - In this plot of Dow Jones performance since 1986, the mean is clearly increasing over time



PROPERTIES FOR TIME SERIES PREDICTION

- Below are simulated examples of non-stationary time series and why they might occur

Table 1 Non-stationary behavior



PROPERTIES FOR TIME SERIES PREDICTION

- Often, if these assumptions do not hold, we can alter our data to make them true
  - Two common methods are de-trending and differencing
- De-trending would mean to remove any major trends in our data
- We could do this is many ways, but the simplest is to fit a line to the trend and make a new series that is the difference between the line and the true series

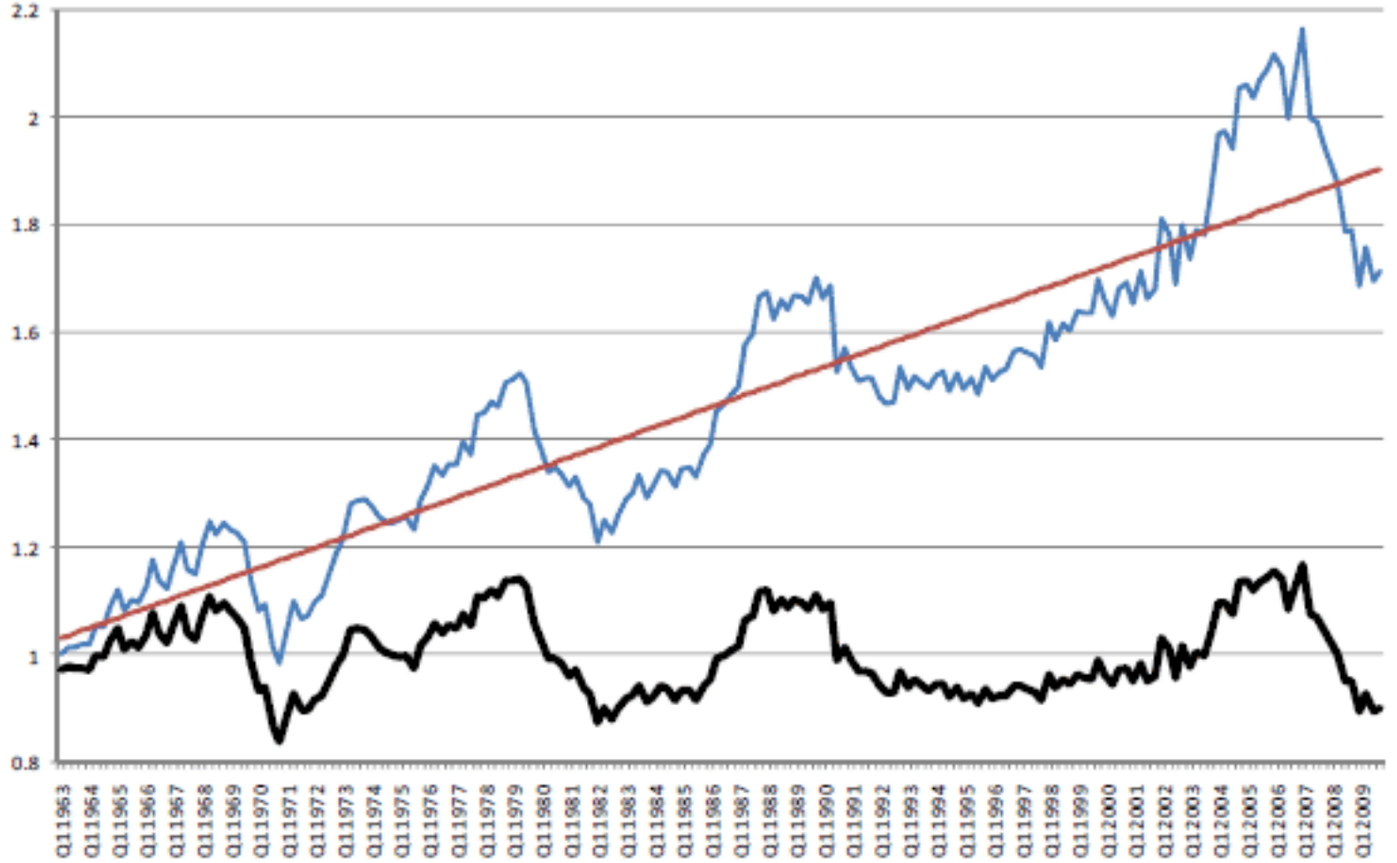
PROPERTIES FOR TIME SERIES PREDICTION

- For example, there is a clear upward (non-stationary) trend in google searches for “iPhone”

- If we fit a line to this data first, we can create a new series that is the difference between the true number of searches and the predicted searches

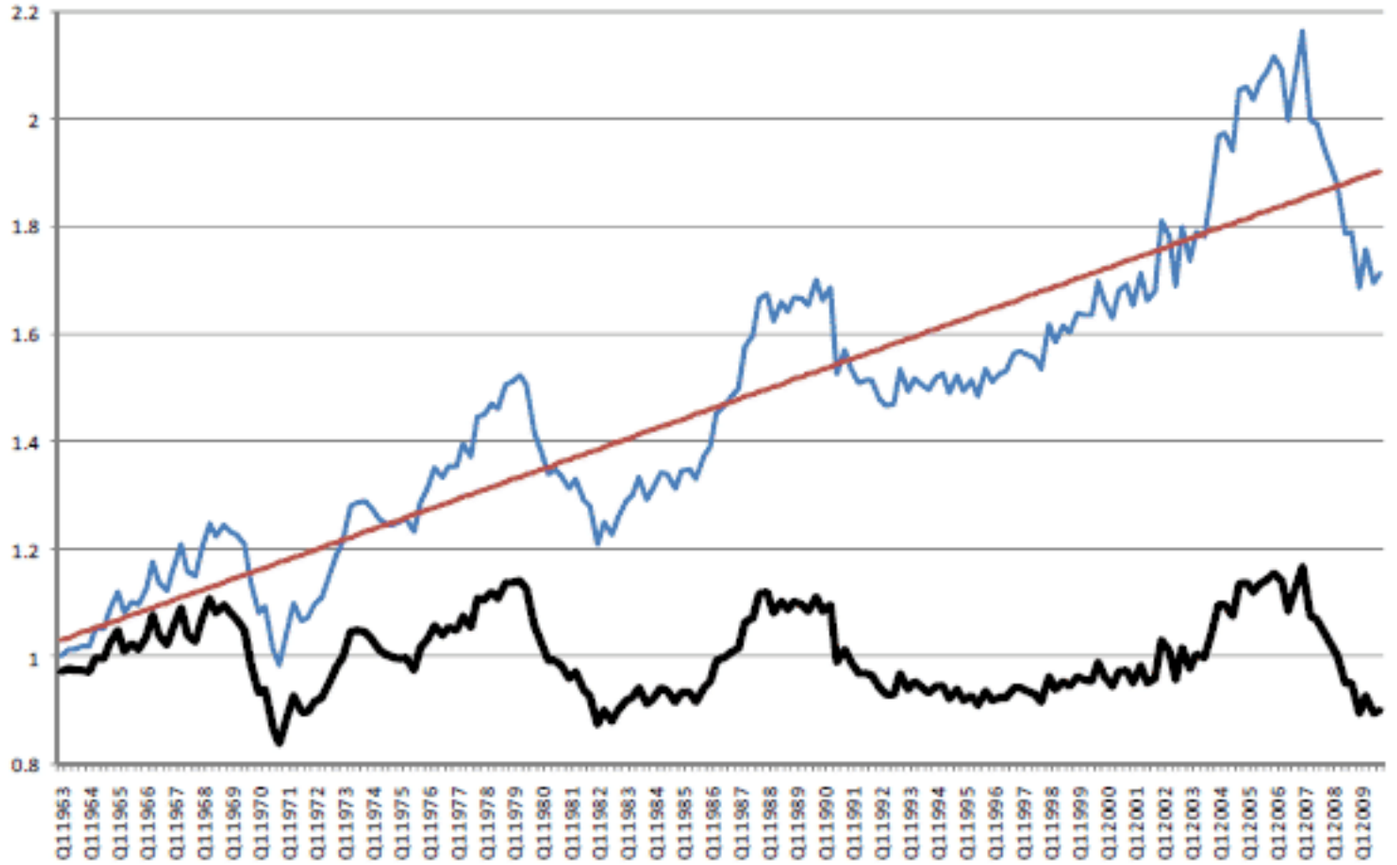
PROPERTIES FOR TIME SERIES PREDICTION

- Below is an example where we look at US housing prices over time
  - Clearly, there is an upward trend, making the time series non-stationary (ie: the mean house price is increasing)



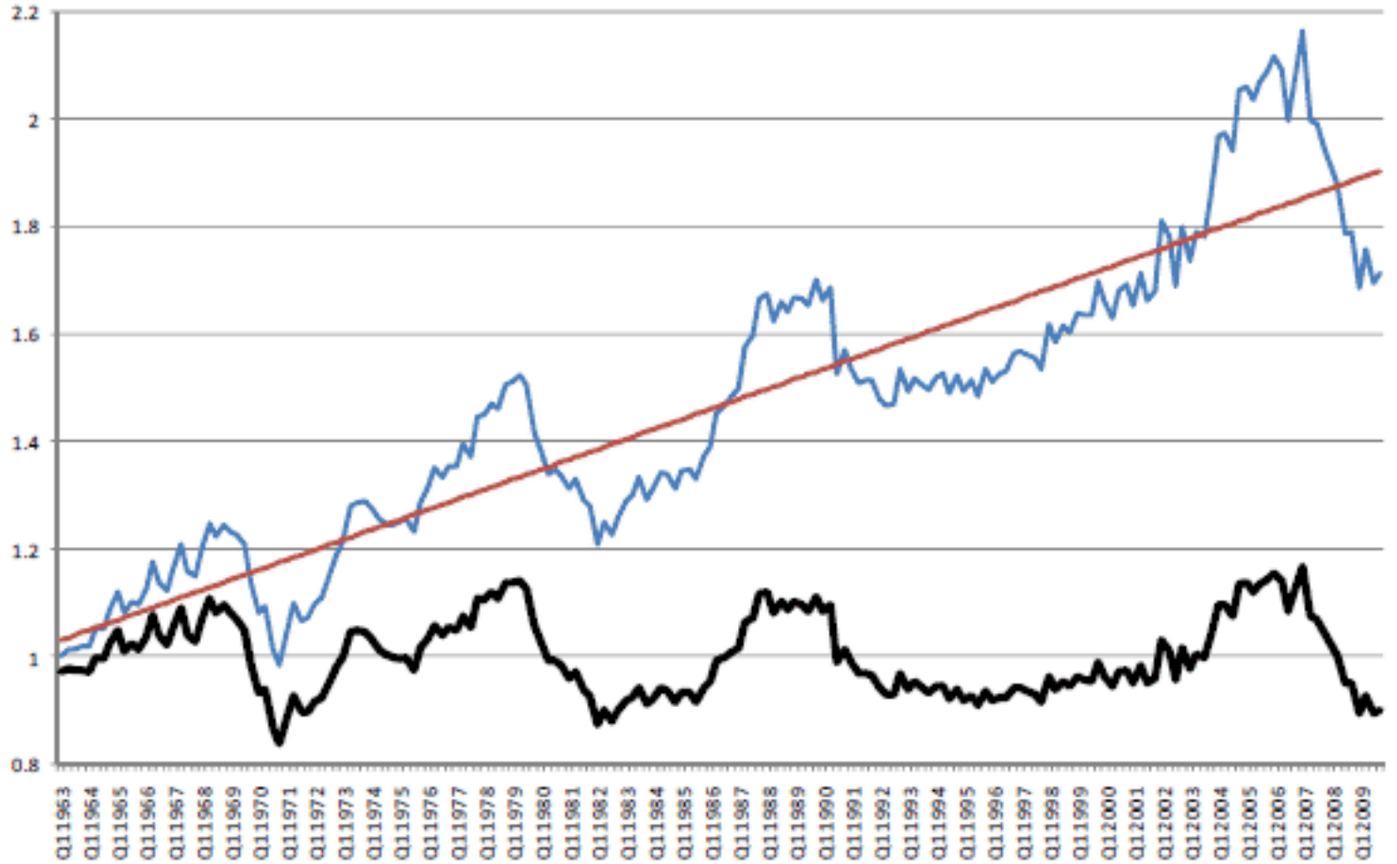
PROPERTIES FOR TIME SERIES PREDICTION

- We can fit a line that represents the trend
  - With our trend line, we can subtract the trend line value from the original value to get the bottom figure



PROPERTIES FOR TIME SERIES PREDICTION

- The data now has a fixed mean and will be easier to model
  - This pattern is similar to mean-scaling our features in earlier models with StandardScaler

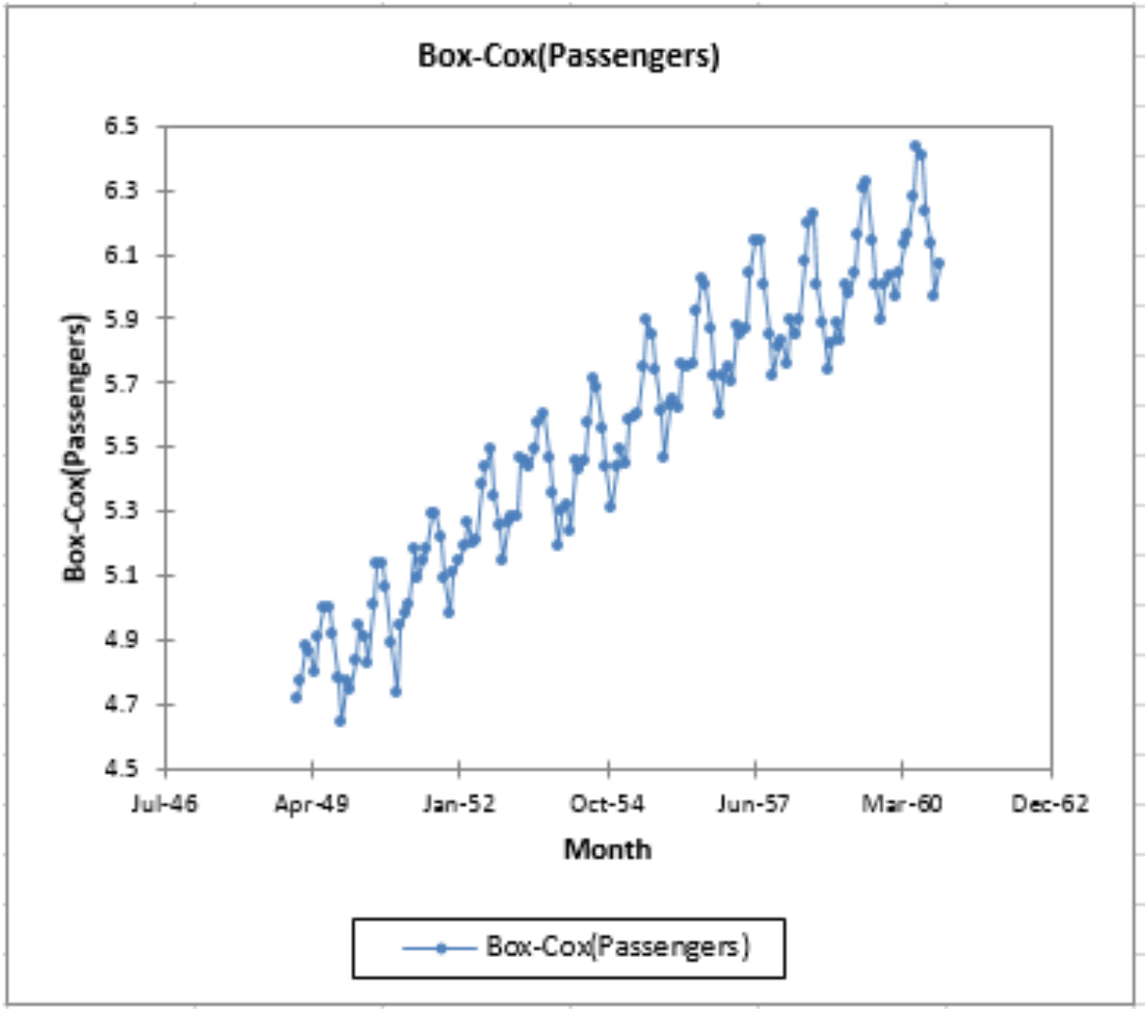


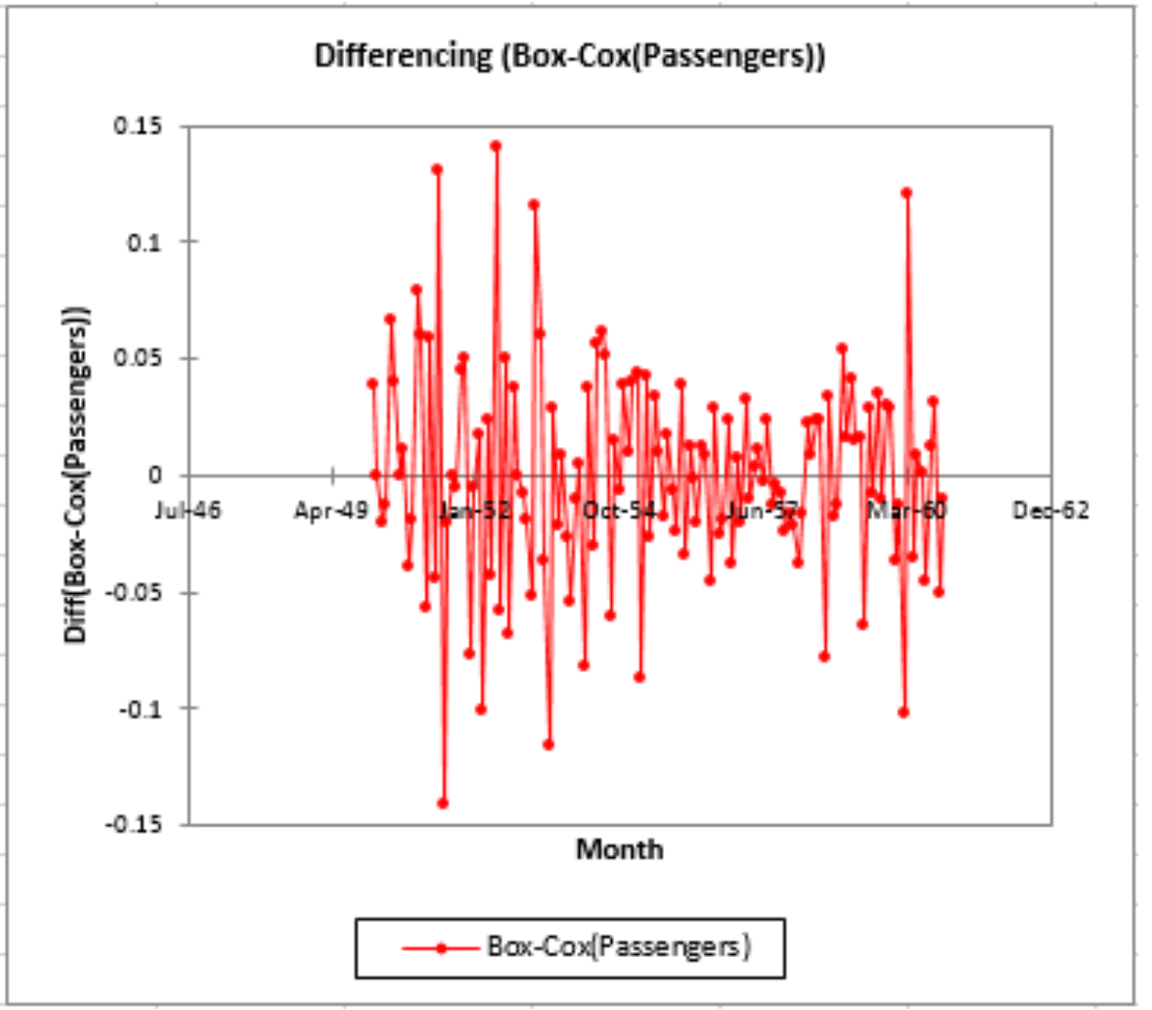
PROPERTIES FOR TIME SERIES PREDICTION

- A simpler method is **differencing**
  - This is very closely related to the diff() function we saw in the last class

- Instead of predicting the series (again our non-stationary series), we can predict the difference between two consecutive values

PROPERTIES FOR TIME SERIES PREDICTION





ACTIVITY: KNOWLEDGE CHECK

EXERCISE

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

Non-stationary data is the most common; almost any interesting data set is non-stationary

1. Can you think of some interesting data sets that might be stationary?

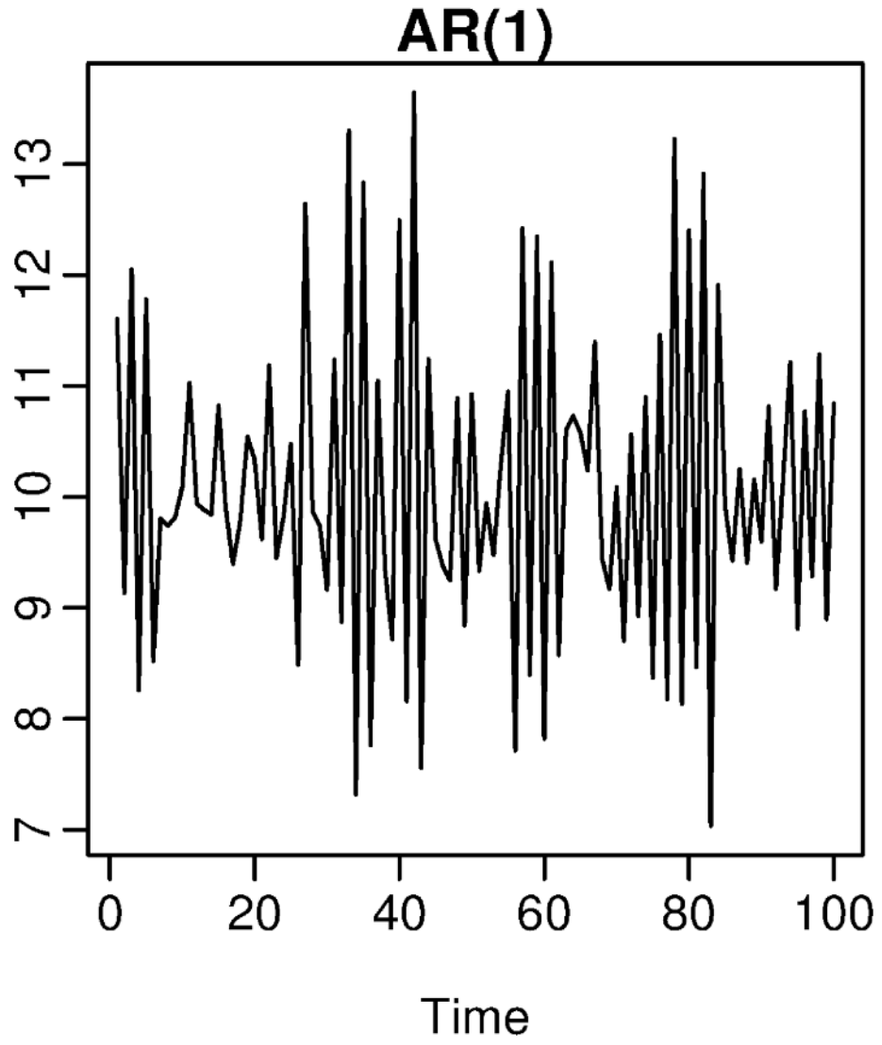
TIME SERIES MODELS

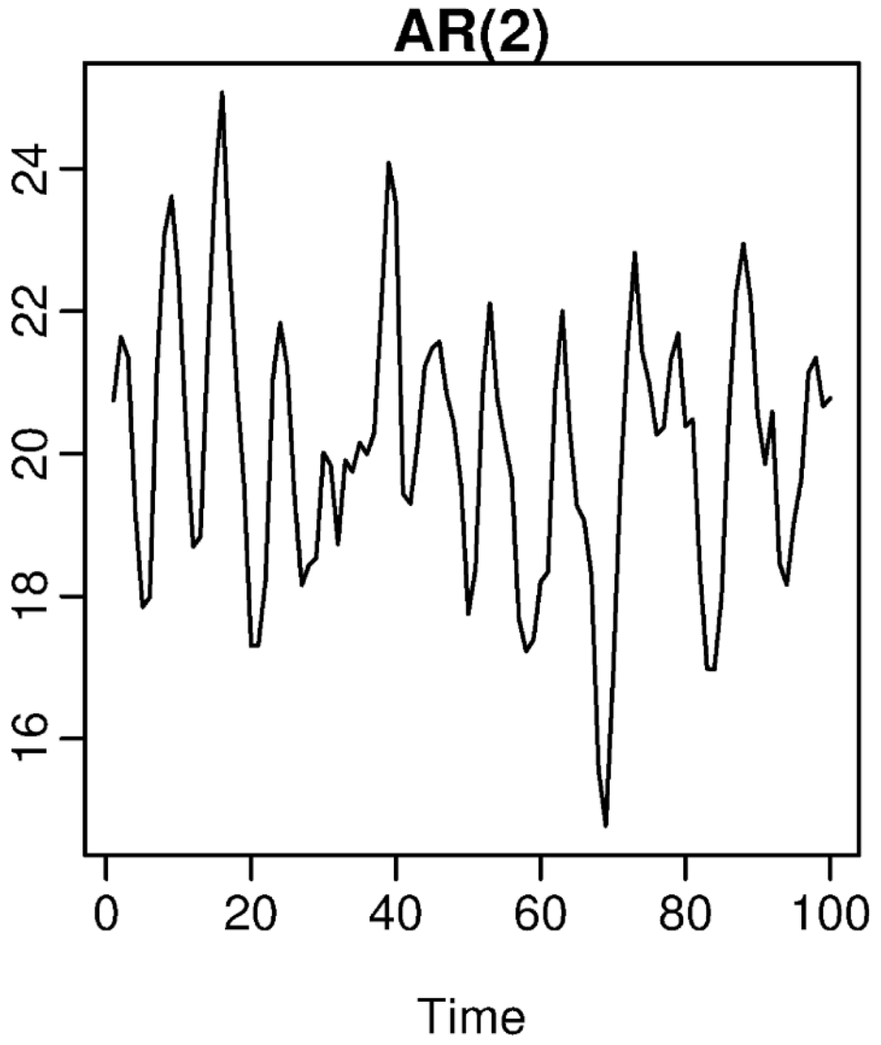
- In the rest of this lesson, we are going to build up to the **ARIMA** time series model
- This model combines the ideas of differencing and two models we will see
  - AR - AutoRegressive Models
  - MA - Moving Average Models

AUTOREGRESSIVE MODELS

- Autoregressive models are those that 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
- If we are attempting to predict weekly sales, we use the sales from a previous week as input
- Typically, AR models are noted *AR(p)* where *p* indicates the number of previous time points to incorporate, with *AR(1)* being the most common

AUTOREGRESSIVE MODELS





AUTOREGRESSIVE MODELS

- In an autoregressive model, similar to standard regression, we are learning regression coefficients for each of the p previous values
  - Therefore, we will learn p coefficients or β values

- If we have a time series of sales per week, y_i, we can regress each y_i from the last p values

$$y_i = \alpha + \beta_1 y_{i-1} + \beta_2 y_{i-2} + \cdots + \beta_p y_{i-p} + \epsilon$$

- As with standard regression, our model assumes that each outcome variable is a linear combination of the inputs and a random error term

AUTOREGRESSIVE MODELS

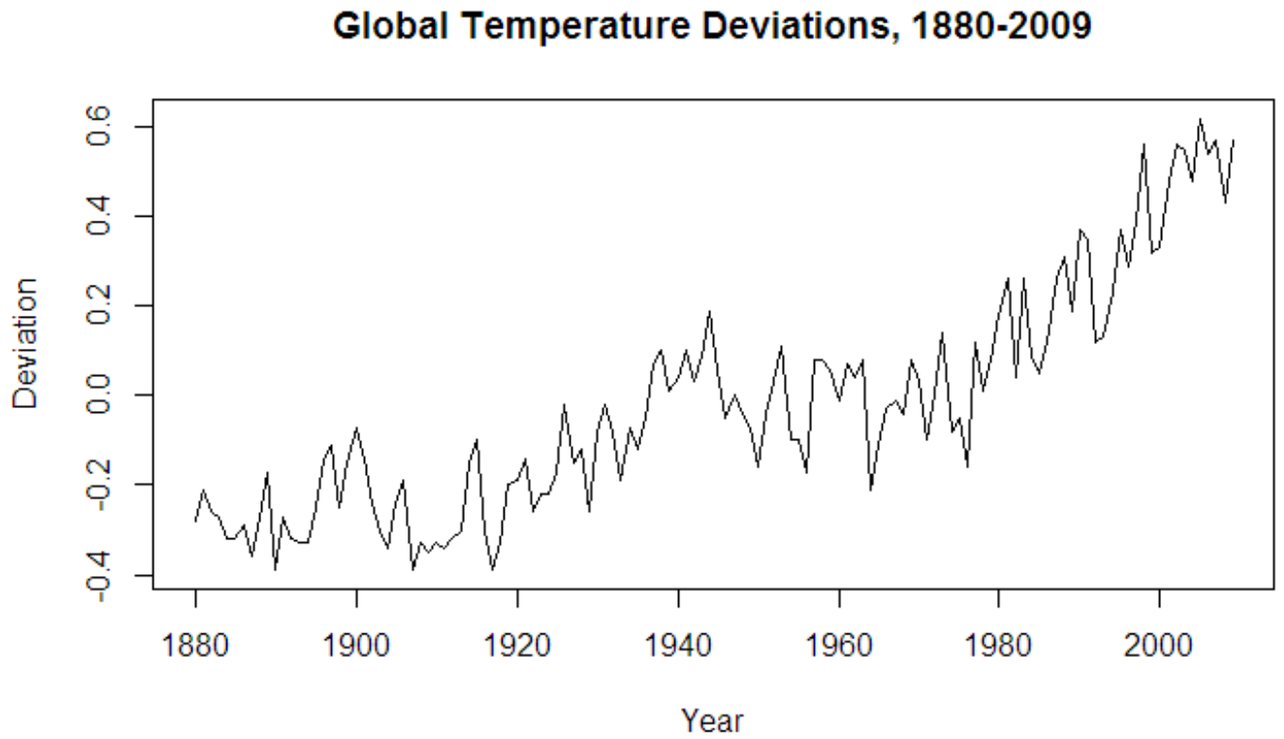
- For an *AR(1)* model, we will learn a single coefficient

- This coefficient, β, will tell us the relationship between the previous value, Y_{t-1} and the next value, Y_t

$$Y_t = \beta Y_{t-1}$$

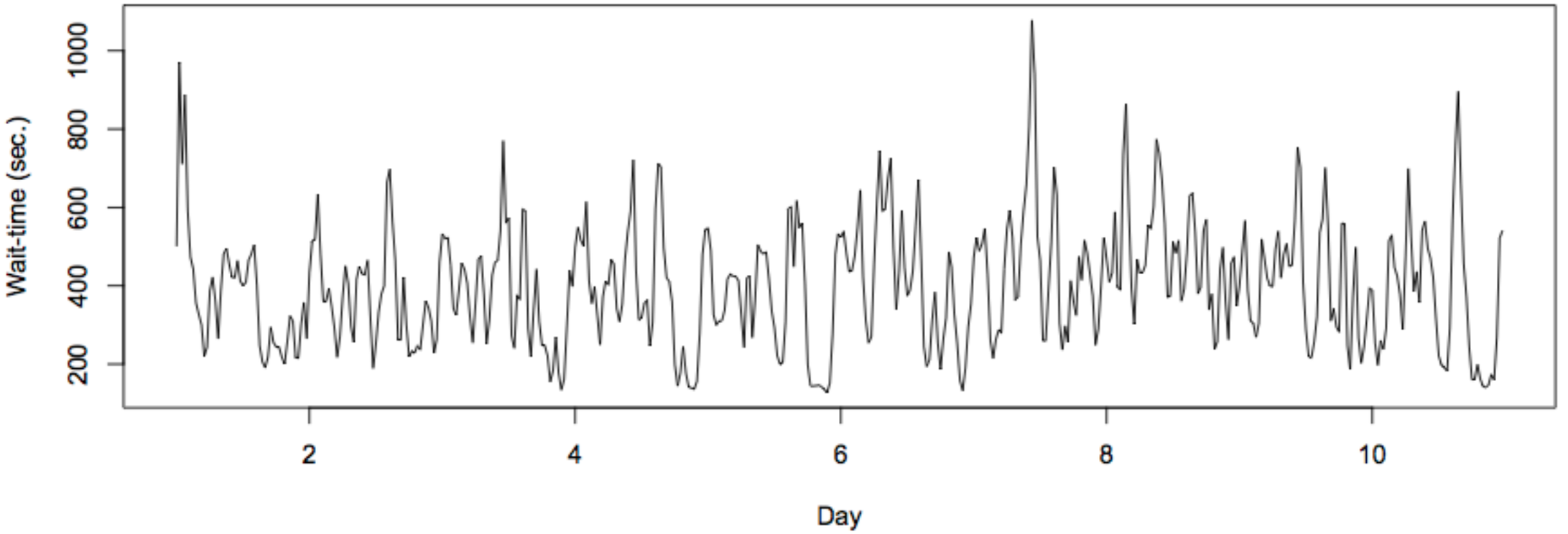
AUTOREGRESSIVE MODELS

- A value > 1 would indicate a growth over previous values
  - This would typically represent non-stationary data, since if we compound the increases, the values are continually increasing



AUTOREGRESSIVE MODELS

- Values between 1 and -1 represent increasing and decreasing patterns from previous patterns



AUTOREGRESSIVE MODELS

- As with other models, interpretation of the model becomes more complex as we add more factors
- Going from *AR(1)* to *AR(2)* can add significant multi-collinearity
- Recall that **autocorrelation** is the correlation of a value with its series **lagged** behind
- A model with high correlation implies that the data is highly dependent on previous values and an autoregressive model would perform well

AUTOREGRESSIVE MODELS

- 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

ACTIVITY: KNOWLEDGE CHECK

EXERCISE

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

1. If we observe an autocorrelation near 1 for lag 1, what do we expect the single coefficient in an AR(1) model to be? >1, between 0 and 1 or <1?
2. What if we observe an autocorrelation of 0?

MOVING AVERAGE 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

- We will attempt to predict the next value based on the overall average and how off our previous predictions were

MOVING AVERAGE MODELS

- This model is useful for handling specific or abrupt changes in a system
- AR models slowly incorporate changes in the system by combining previous values
- MA models use prior errors to quickly incorporate changes
- This is useful for modelling a sudden occurrence - something going out of stock or a sudden rise in popularity affecting sales

MOVING AVERAGE MODELS

- 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

- If we have a time series of sales per week, y_i, we can regress each y_i from the last q error terms

$$y_i = mean + \beta_1 \epsilon_{i-1} + \beta_2 \epsilon_{i-2} + \cdots + \beta_q \epsilon_{i-q}$$

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

MOVING AVERAGE MODELS

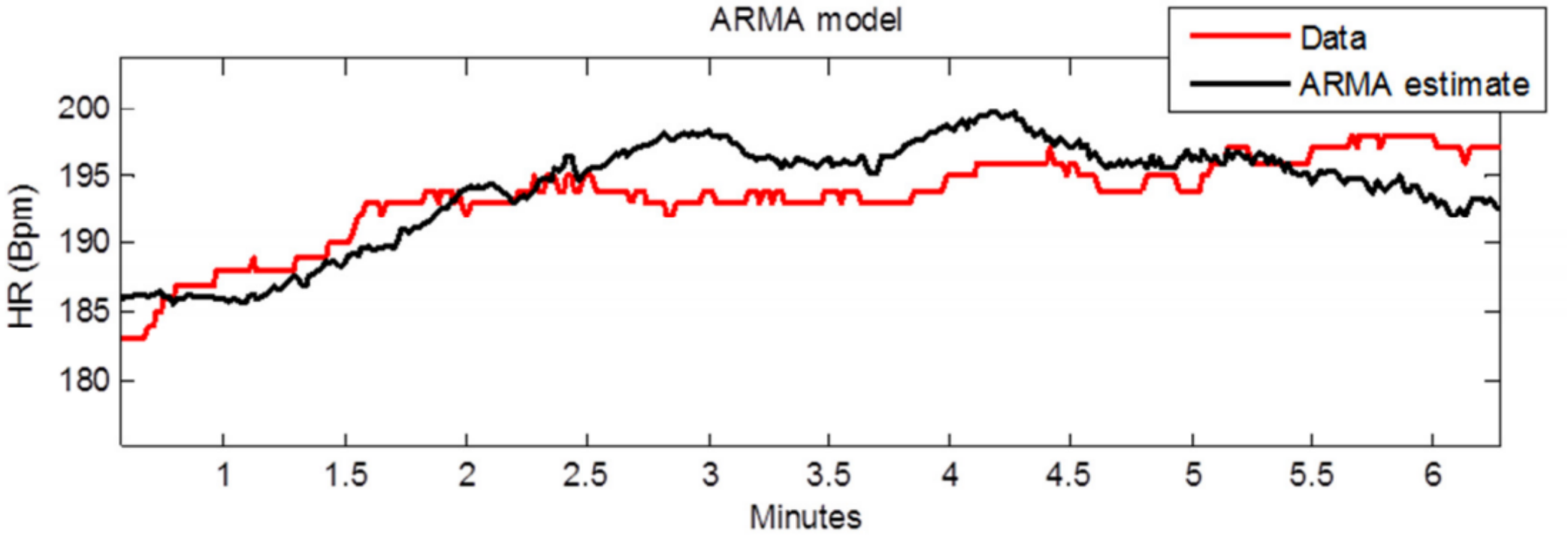
- Of course, we do not have error terms when we start - where do they come from?
- This requires a more complex fitting procedure than we have seen previously
- We need to iteratively fit a model (perhaps with random error terms), compute the errors and then refit, again and again

MOVING AVERAGE MODELS

- In this model, we learn q coefficients
- In an *MA(1)* model, we learn one coefficient
- This value indicates the impact of how our previous error term on the next prediction

ARMA MODELS

- **ARMA** (pronounced "R-mah") models combine the autoregressive and moving average models
- An *ARMA(p, q)* model is simply a combination (sum) of an *AR(p)* model and *MA(q)* model



ARMA MODELS

- We specify two model settings, p and q, which correspond to combining an $AR(p)$ model with a $MA(q)$ model
- Incorporating both models allows us to mix two types of effects
  - AR models slowly incorporate changes in preferences, tastes and patterns
  - Moving average models base their prediction on the prior error, allowing to correct sudden changes based on random events - supply, popularity spikes, etc

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)

ARIMA MODELS

- Recall the pandas diff() function
  - This computes the difference between two consecutive values

- In an ARIMA model, we attempt to predict this difference instead of the actual values

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

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

ARIMA MODELS

- An ARIMA model has three parameters and is specified *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

- d was 1 in our prior example
  - For d=2, our model would be

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

ARIMA MODELS

- 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 over time, 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

DEMONSTRATION

TIME SERIES MODELLING IN STATSMODELS

TIME SERIES MODELLING IN STATSMODELS

- To explore time series models, we will continue to use the Rossmann sales data

- This data set has sales data for every Rossmann store for a 3-year period and indicators for holidays and basic store information

TIME SERIES MODELLING IN STATSMODELS

- In the last class, we saw that we could plot the sales data at a particular store to identify how the sales changed over time

- We also computed autocorrelation for the data at varying lag periods. This helps us identify if previous time points are predictive of future data and which time points are most important - the previous day, week or month

TIME SERIES MODELLING IN STATSMODELS

```
import pandas as pd
# Load the data and set the DateTime index
data = pd.read_csv("../../../data/rossmann.csv", skipinitialspace = True)
data["Date"] = pd.to_datetime(data["Date"])
data.set_index("Date", inplace = True)
# Filter to Store 1
store1_data = data[data.Store == 1]
# Filter to open days
store1_open_data = store1_data[store1_data.Open == 1]
# Plot the sales over time
store1_open_data[["Sales"]].plot()
```

ACTIVITY: KNOWLEDGE CHECK

EXERCISE

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

1. Compute the autocorrelation of Sales in Store 1 for lag 1 and 2
2. Will we be able to use a predictive model, particularly an autoregressive one?

ACTIVITY: KNOWLEDGE CHECK

EXERCISE

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

1. To start to diagnose the model, we want to look at residuals
2. What are residuals?
3. In linear regression, what did we expect of residuals?

AR, MA AND ARMA MODELS IN STATSMODELS

- Residuals are the errors of the model or how off our predictions are
- Ideally, we want randomly distributed errors that are small
- If the errors are large, our model does not perform well
- If the errors have a pattern, particularly over time, we may have overlooked something in the model or have periods of time that are different than the rest of the data set

AR, MA AND ARMA MODELS IN STATSMODELS

- We can use statsmodels to plot the residuals

```
model.resid.plot()
```

- Here we see large spikes at the end of each year, indicating that our model does not account for the holiday spikes

- Our model considers a short period of time, so it does not take into account the longer seasonal pattern

ACTIVITY: KNOWLEDGE CHECK

EXERCISE

DIRECTIONS: ANSWER THE FOLLOWING QUESTIONS

1. Take a moment to look at the coefficients of our new model
2. Offer an interpretation of this model

ARIMA MODELS IN STATSMODELS

- Increasing p increases the dependency on previous values further (longer lag)
  - But our autocorrelation plots show this is not necessary past a certain point

- Increasing q increases the likelihood of an unexpected jump at a handful of points
  - The autocorrelation plots show this does not help past a certain point

- Increasing d increases differencing, but d=1 moves our data towards stationarity (other than a few points)
  - d=2 would imply an exponential trend which we do not have here

ARIMA MODELS IN STATSMODELS

- There are variants of ARIMA that will better handle the seasonal aspect of our data
  - This is referred to as Seasonal ARIMA

- These models fit two ARIMA models, one on the current frequency (daily in our example) and another on the seasonal frequency (maybe monthly or yearly patterns)

- Additionally, issues with seasonality could be handled by preprocessing tricks such as de-trending

INDEPENDENT PRACTICE

WALMART SALES DATA

ACTIVITY: WALMART SALES DATA

EXERCISE

DIRECTIONS: COMPLETE THE FOLLOWING TASKS (50 MINUTES)

We will analyse the weekly sales data from Walmart over a two year period from 2010 to 2012. The data is separated by store and department, but we will focus on analysing one store for simplicity

1. Filter the dataframe to Store 1 sales and aggregate over departments to compute the total sales per store
2. Plot the rolling_mean for Weekly_Sales. What general trends do you observe?
3. Compute the 1, 2, 52 autocorrelations for Weekly_Sales and/or create an autocorrelation plot
4. What does the autocorrelation plot say about the type of model you want to build?

ACTIVITY: WALMART SALES DATA

EXERCISE

DIRECTIONS: COMPLETE THE FOLLOWING TASKS (50 MINUTES)

5. Split the weekly sales data in a training and test set - using 75% of the data for training
6. Create an AR(1) model on the training data and compute the mean absolute error of the predictions
7. Plot the residuals - where are their significant errors?
8. Compute and AR(2) model and an ARMA(2, 2) model - does this improve your mean absolute error on the held out set?
9. Finally, compute an ARIMA model to improve your prediction error - iterate on the p, q, and parameters comparing the model's performance

TOPIC REVIEW

TOPIC 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)

TOPIC REVIEW

- 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

BEFORE NEXT CLASS

BEFORE NEXT CLASS

DUE DATE

- Project:
  - Final Project, part 3

Q & A

EXIT TICKETS

DON'T FORGET TO FILL OUT YOUR EXIT TICKET

Exit Ticket Link

What's the lesson number?	16
What was the topic of the lesson?	Time Series Modelling

CREDITS AND REFERENCES

CREDITS AND REFERENCES

- ARIMA model overview
- Time Series Analysis in Python with statsmodels (.pdf)
- Investopedia: Stationarity
- First Place Entry in Walmart Sales Prediction
- Google Search Terms predict market movements