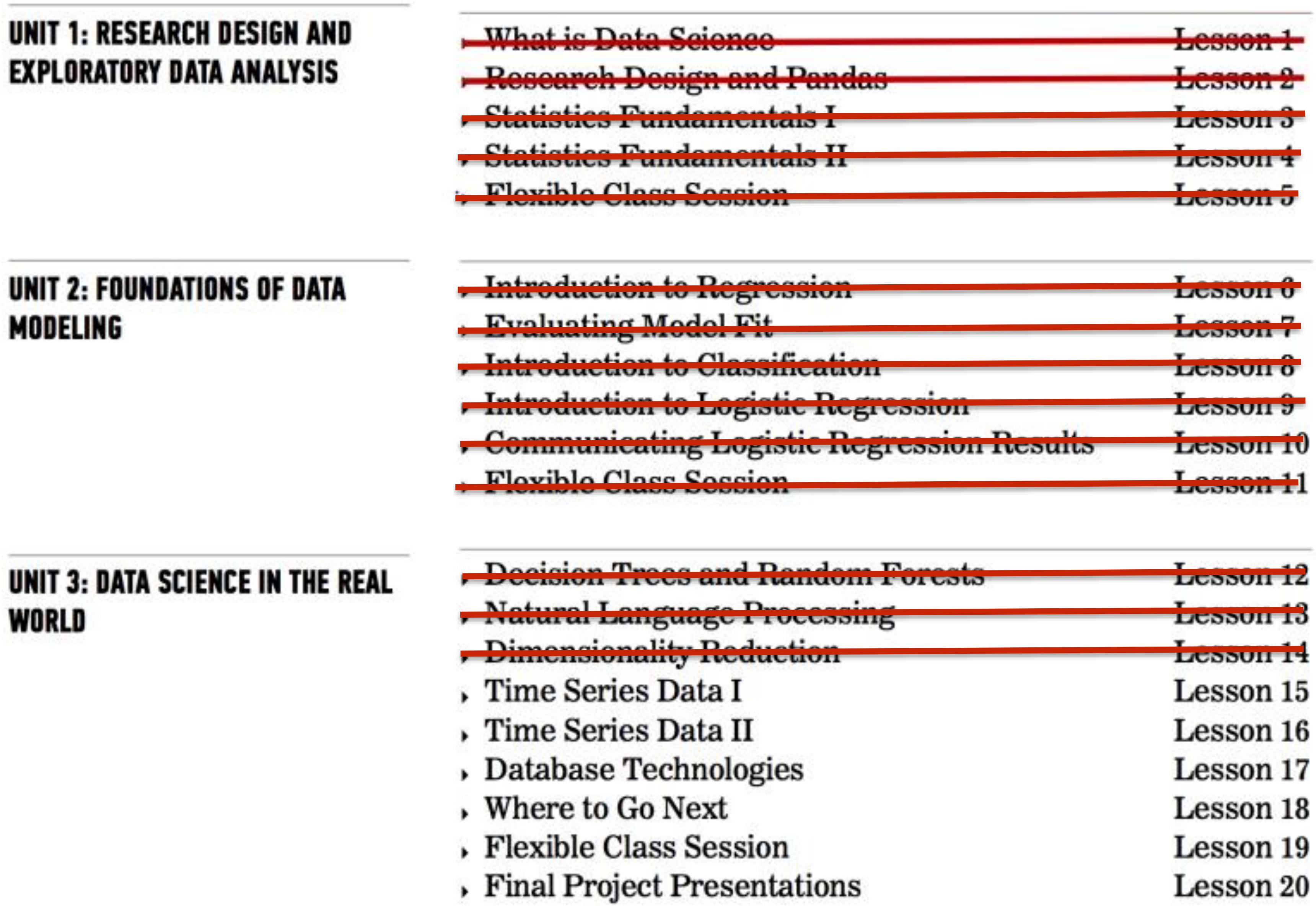
GENERAL ASSEMBLY

TIME SERIES ANALYSIS I

Abbas Chokor, Ph.D.

Staff Data Scientist, Seagate Technology

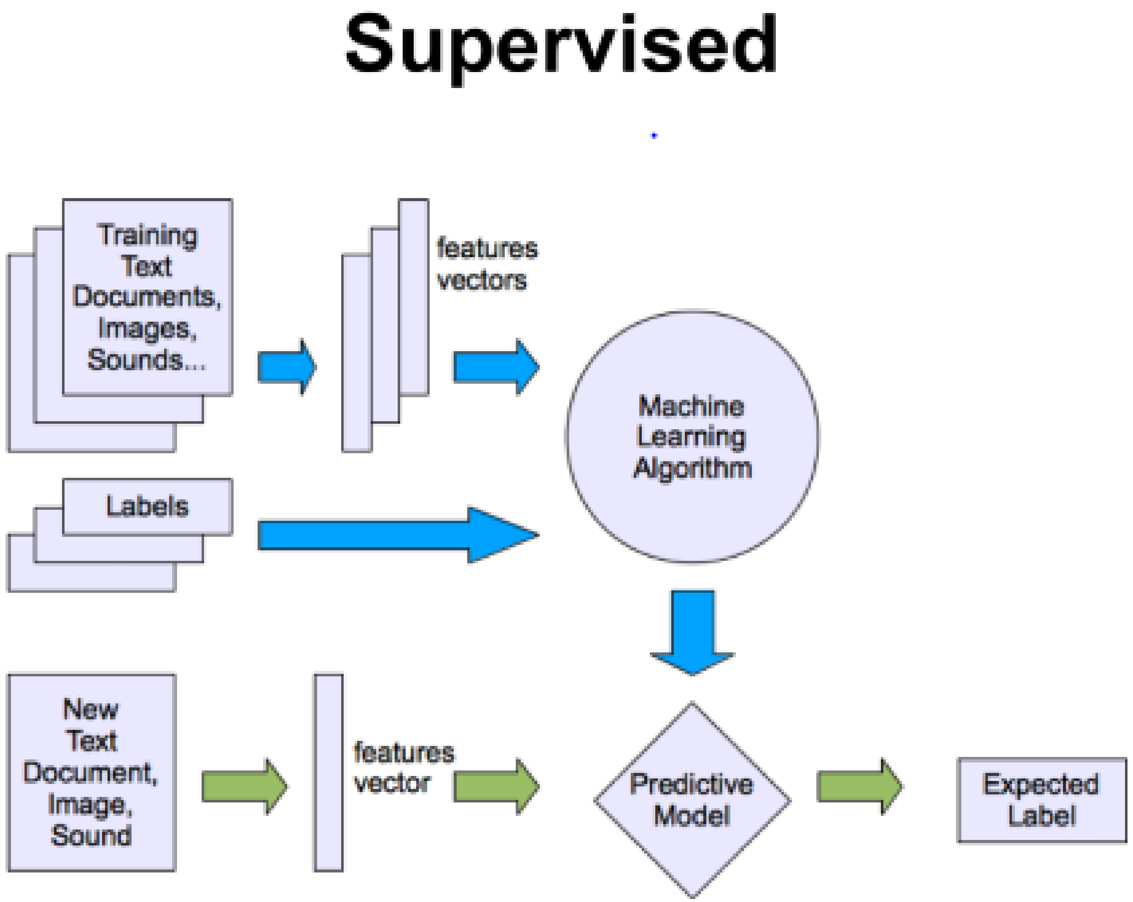
OUR PROGRESS SO FAR

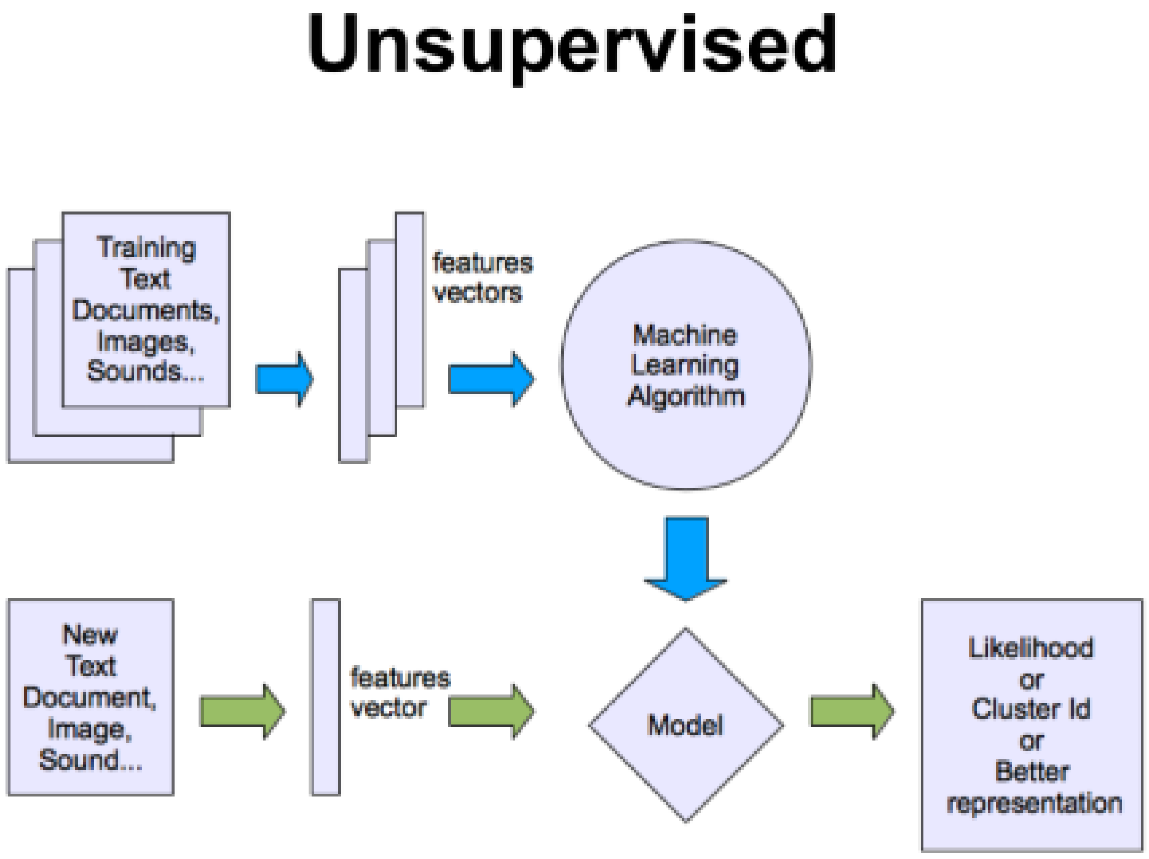


UNIT 1: RESEARCH DESIGN AND EXPLORATORY DATA ANALYSIS	~~What is Data Science~~	~~Lesson 1~~
	~~Research Design and Pandas~~	~~Lesson 2~~
	~~Statistics Fundamentals I~~	~~Lesson 3~~
	~~Statistics Fundamentals II~~	~~Lesson 4~~
	~~Flexible Class Session~~	~~Lesson 5~~
UNIT 2: FOUNDATIONS OF DATA MODELING	~~Introduction to Regression~~	~~Lesson 6~~
	~~Evaluating Model Fit~~	~~Lesson 7~~
	~~Introduction to Classification~~	~~Lesson 8~~
	~~Introduction to Logistic Regression~~	~~Lesson 9~~
	~~Communicating Logistic Regression Results~~	~~Lesson 10~~
	~~Flexible Class Session~~	~~Lesson 11~~
UNIT 3: DATA SCIENCE IN THE REAL WORLD	~~Decision Trees and Random Forests~~	~~Lesson 12~~
	~~Natural Language Processing~~	~~Lesson 13~~
	~~Dimensionality Reduction~~	~~Lesson 14~~
	Time Series Data I	Lesson 15
	Time Series Data II	Lesson 16
	Database Technologies	Lesson 17
	Where to Go Next	Lesson 18
	Flexible Class Session	Lesson 19
	Final Project Presentations	Lesson 20

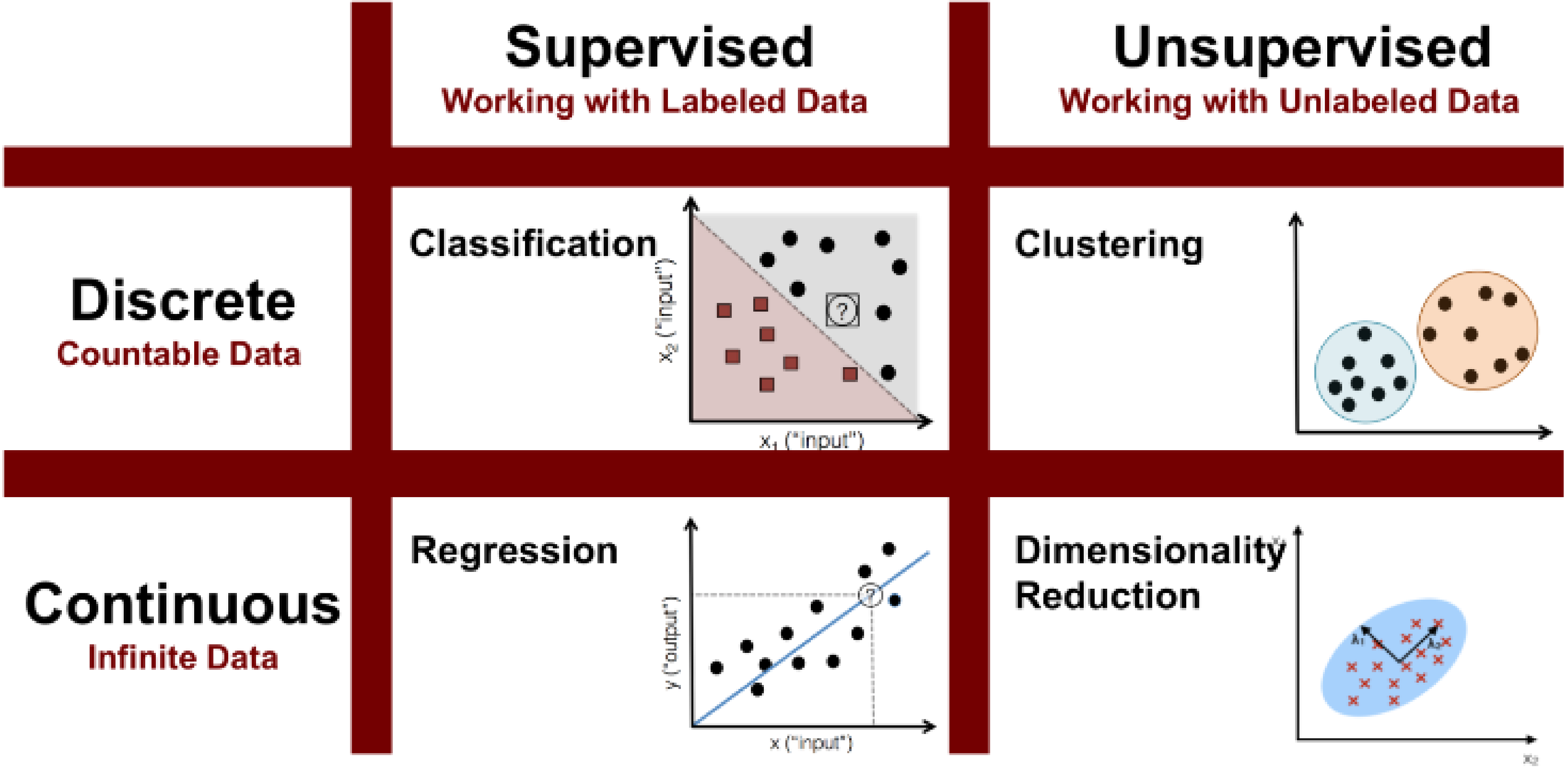
Today’s Class

TYPES OF MACHINE LEARNING


Supervised
Training Text Documents, Images, Sounds...
features vectors
Labels
Machine Learning Algorithm
New Text Document, Image, Sound
features vector
Predictive Model
Expected Label


Unsupervised
Training Text Documents, Images, Sounds...
features vectors
Machine Learning Algorithm
New Text Document, Image, Sound...
features vector
Model
Likelihood or Cluster Id or Better representation

TYPES OF MACHINE LEARNING



	Supervised Working with Labeled Data	**Unsupervised** Working with Unlabeled Data
Discrete Countable Data	Classification	Clustering
Continuous Infinite Data	Regression	Dimensionality Reduction

WHAT DID WE LEARN?

- Understand what *latent* variables are
- Dimensionality Reduction (PCA, LSI, LDA)
- Understand the uses of *latent variables* in language processing
- Use the *word2vec* and *LDA* algorithms of `genism`

LEARNING OBJECTIVES

- Understand what time series data is and what is unique about it
- Perform time series analysis in Pandas including rolling mean/median and autocorrelation

COURSE

PRE-WORK

PRE-WORK REVIEW

- Load data with Pandas
- Plotting data with Seaborn
- Understand correlation

OPENING

TIME SERIES ANALYSIS

TIME SERIES ANALYSIS

- In this class, we’ll discuss analyzing data that is changing over time.

- In most of our previous examples, we didn’t care which data points were collected earlier or later than others.

- We made assumptions that the data was *not* changing over time.

- This class will focus on statistics around data that is changing over time and how to measure that change.

INTRODUCTION

WHAT IS TIME SERIES DATA?

WHAT IS TIME SERIES DATA?

- Time series data is any data where the individual data points change over time.

- This is fairly common in sales and other business cases where data would likely change according to seasons and trends.

- Time series data is also useful for studying social phenomena. For instance, there is statistically more crime in the summer, which is a seasonal trend.

WHAT IS TIME SERIES DATA?

- Most datasets are likely to have an important time component, but typically we assume that it’s fairly minimal.

- For example, if we were analyzing salaries in an industry, it’s clear that salaries shift over time and vary with the economic period.

- However, if we are examining the problem on a smaller scale (e.g. 3-5 years), the effect of time on salaries is much smaller than other factors, like industry or position.

WHAT IS TIME SERIES DATA?

- When the time component *is* important, we need to focus on identifying the aspects of the data that are influenced by time and those that aren’t.

- Typically, time series data will be a sequence of values. We will be interested in studying the changes to this series and how related individual values are.

- For example, how much does this week’s sales affect next week’s? How much does today’s stock price affect tomorrow’s?

WHAT IS TIME SERIES DATA?

- Time series analysis is useful in many fields: sales analysis, stock market trends, studying economic phenomena, social science problems, etc.

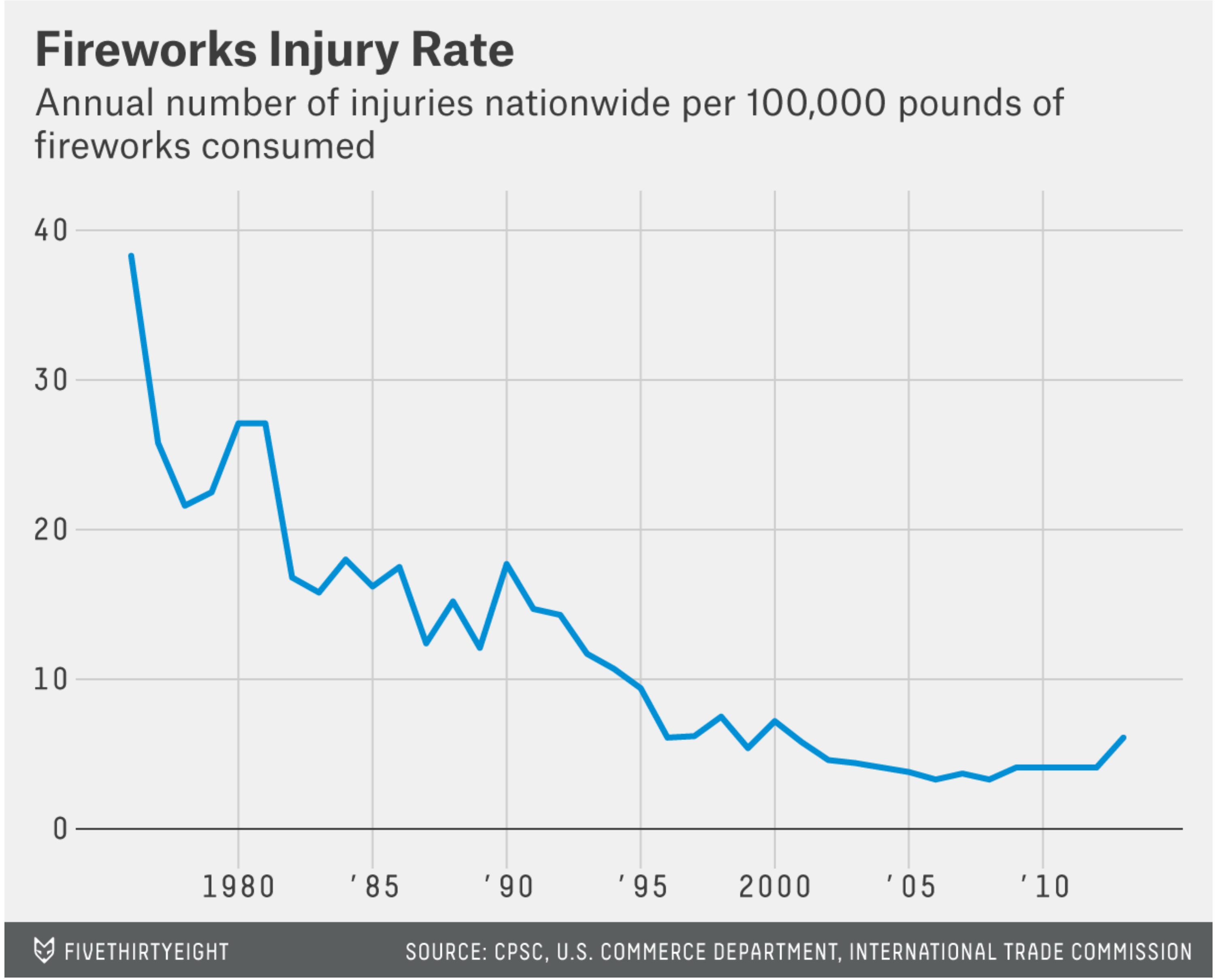
- Typically, we are interested in separating the effects of time into two components:

  - Trends - significant increases or decreases over time

  - Seasonality - regularly repeating increases or decreases

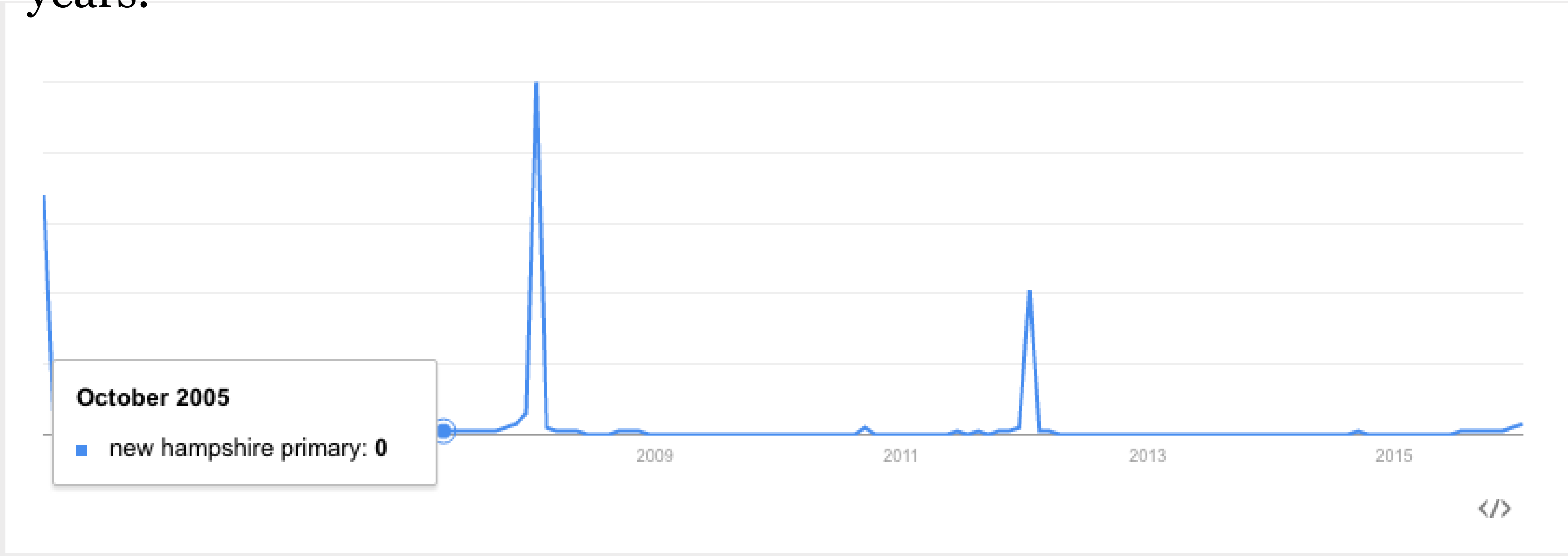
WHAT IS TIME SERIES DATA?

- This plot of fireworks injury rates has an overall *trend* of fewer injuries with no *seasonal* pattern.



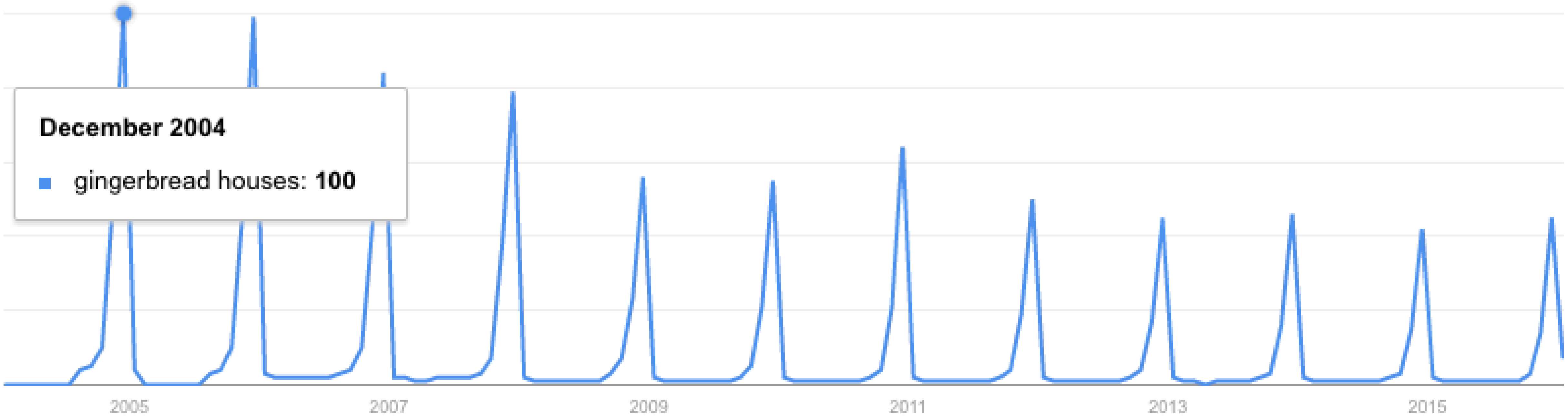
WHAT IS TIME SERIES DATA?

- Meanwhile, the number of searches for the New Hampshire Primary has a clear *seasonal* component - it peaks every four years and on election years.



WHAT IS TIME SERIES DATA?

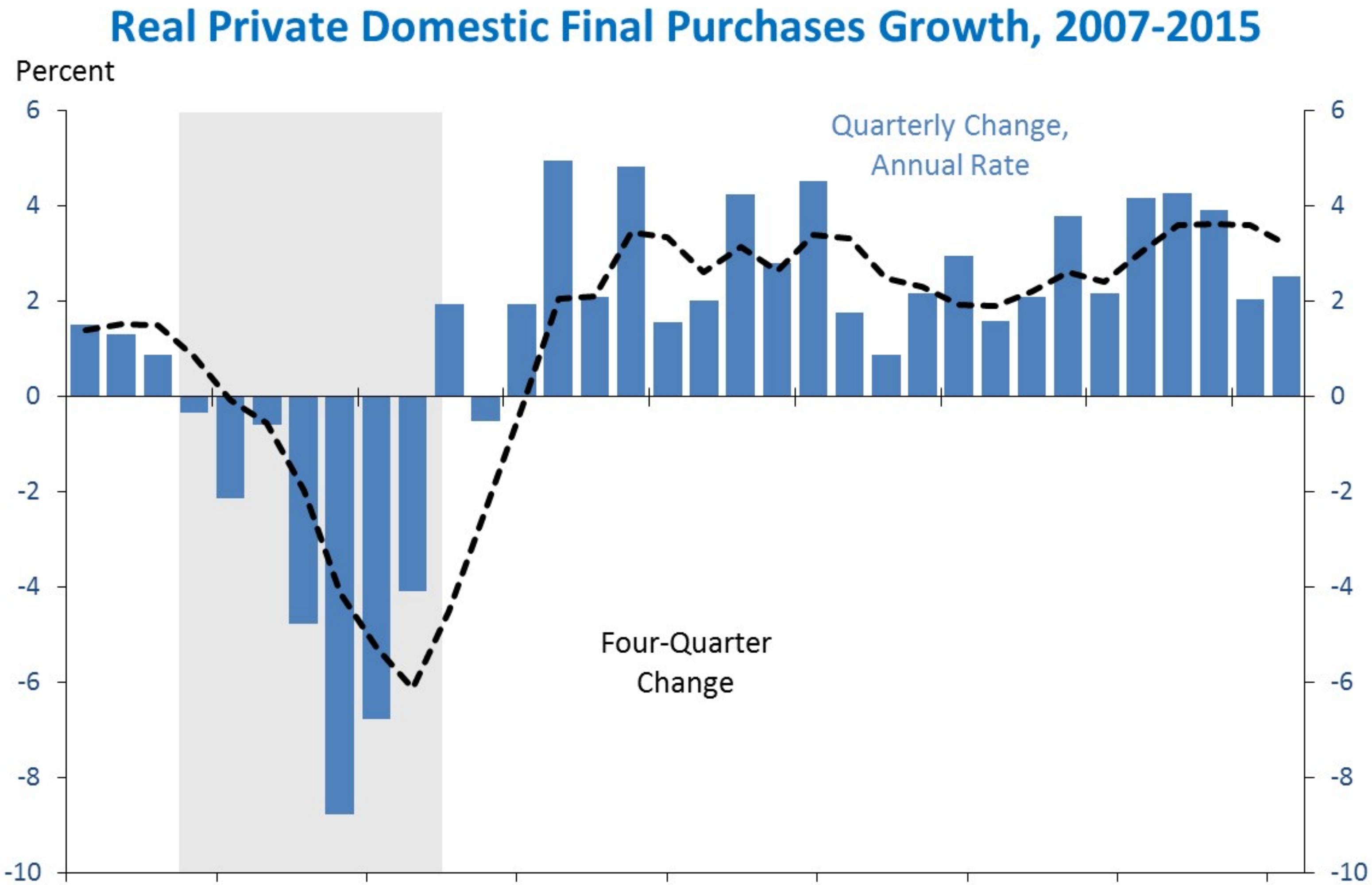
- Similarly, searches for ‘gingerbread houses’ spike every year around the holiday season.



- These spikes recur on a fixed time-scale, making them *seasonal* patterns.

WHAT IS TIME SERIES DATA?

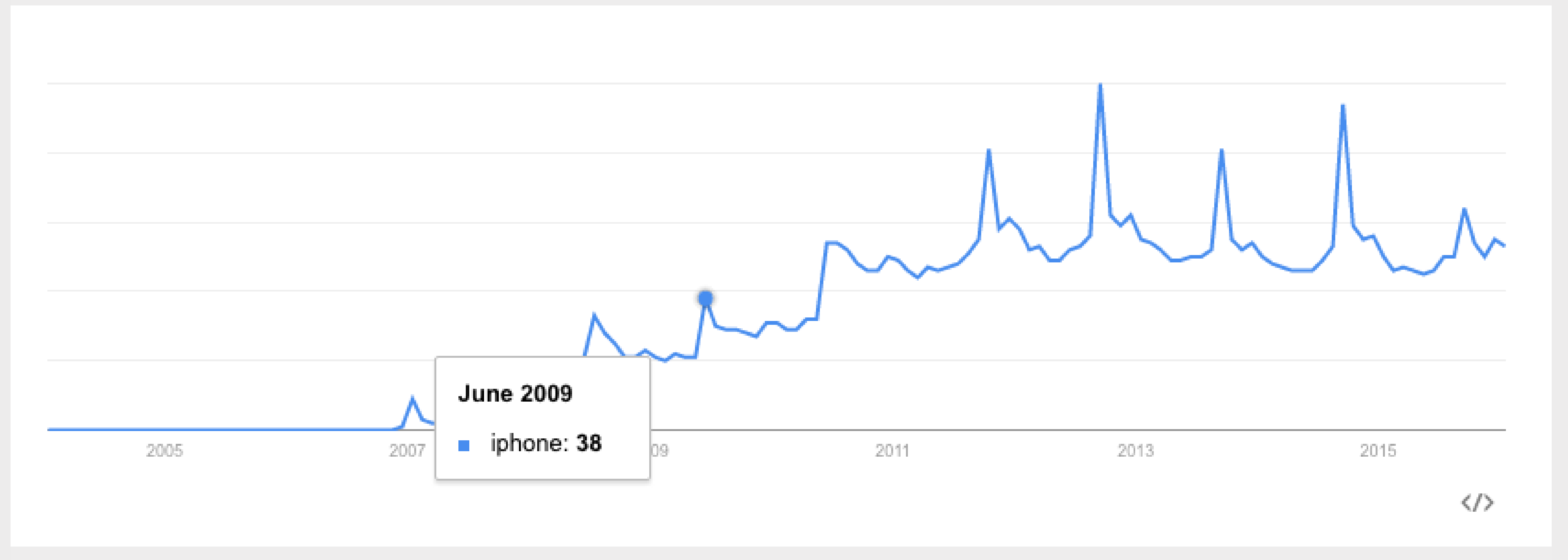
- Many other types of regularly occurring up or down swings may occur without a fixed timescale or *period* (e.g. growth vs. recession for economic trends).



Note: Shading denotes recession. Private domestic final purchases represent the sum of consumption and fixed investment.
Source: Bureau of Economic Analysis; CEA calculations.

WHAT IS TIME SERIES DATA?

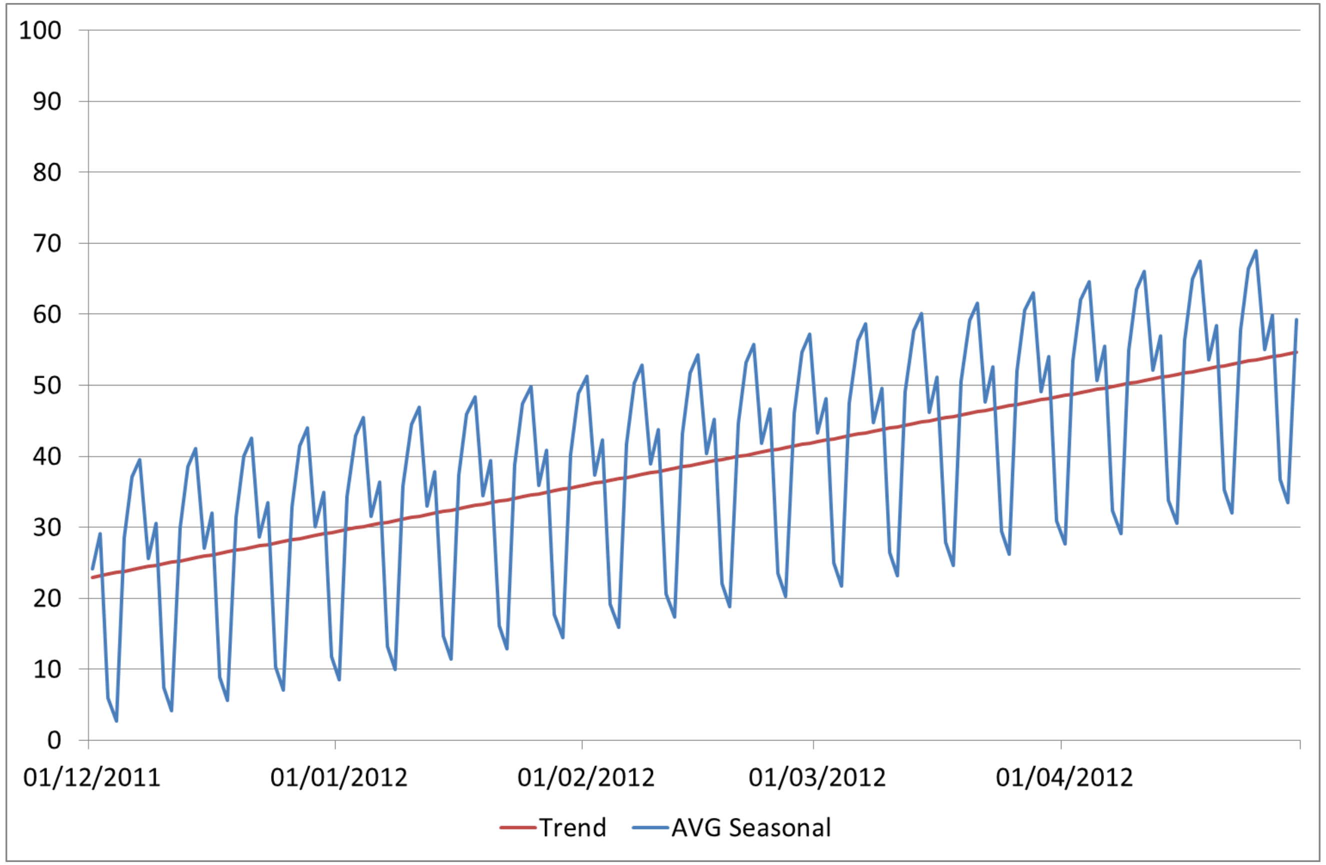
- Searches for “iphone” have both a general trend upwards (indicating more popularity for the phone) as well as a seasonal spike in September (which is when Apple typically announces new versions).



WHAT IS TIME SERIES DATA?

- Most often, we're interested in studying the *trend* and not the *seasonal* fluctuations.

- Therefore it is important to identify whether we think a change is due to an ongoing trend or seasonal change.



INTRODUCTION

COMMON ANALYSIS FOR TIME SERIES DATA

MOVING AVERAGES

- A *moving average* replaces each data point with an average of *k* consecutive data points in time.

- Typically, this is *k/2* data points prior to and following a given time point, but it could also be the *k* preceding points.

- These are often referred to as the “rolling” average.

- The measure of average could be mean or median.

- The formula for the rolling *mean* is $F_t = \frac{1}{p} \sum_{k=t}^{t-p+1} Y_k$

MOVING AVERAGES

- A rolling mean would average all values in the window, but can be skewed by outliers (extremely small or large values).

- This may be useful if we are looking to identify atypical periods or we want to evaluate these odd periods.

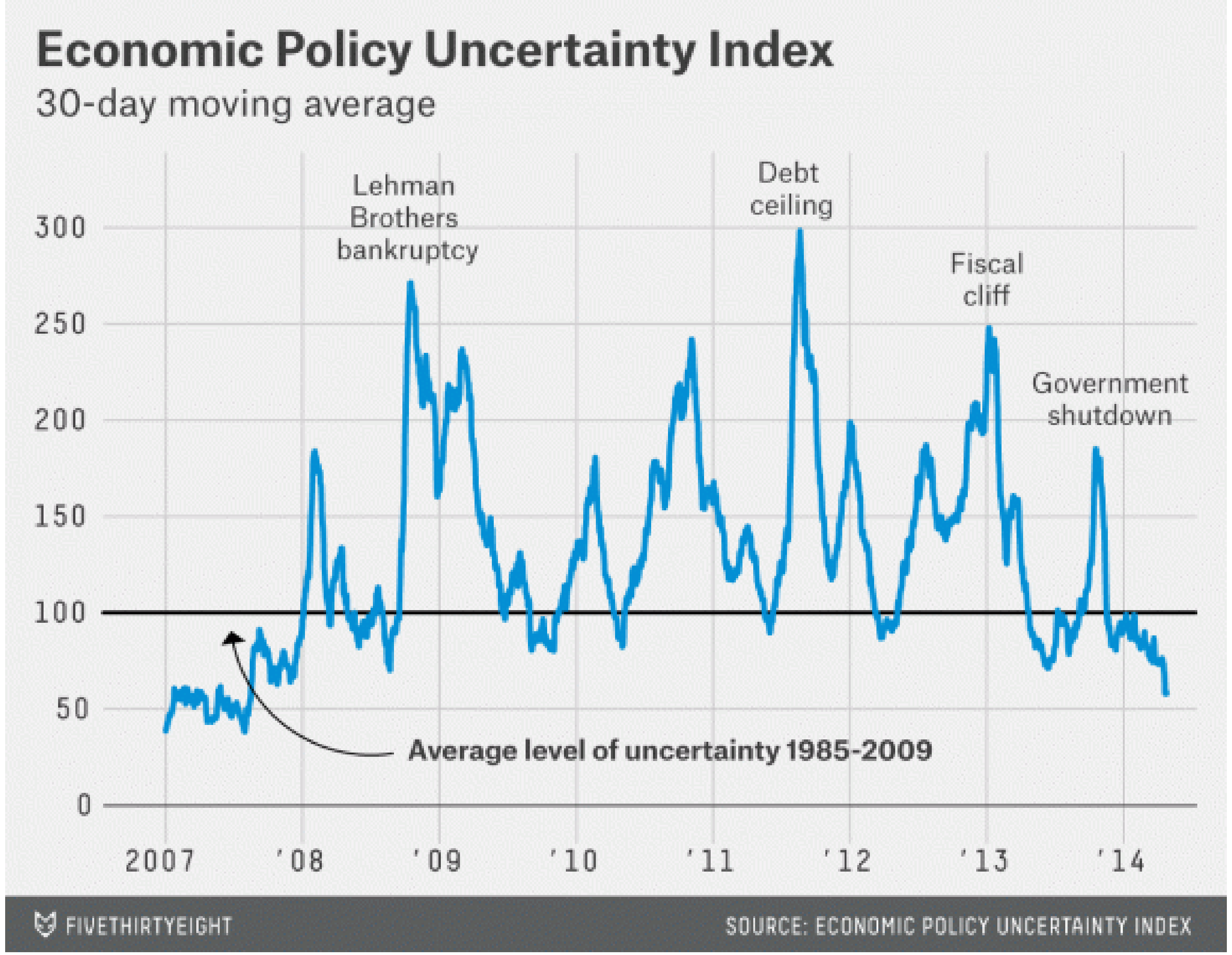
- For example, this would be useful if we are trying to identify particularly successful or unsuccessful sales days.

- The rolling median would provide the 50 percentile value for the period and would possibly be more representative of a “typical” day.

WHAT IS TIME SERIES DATA?

- This plot shows the 30-day moving average of the Economic Uncertainty Index.

- Plotting the moving average allows us to more easily visualize trends by smoothing out random fluctuations and removing outliers.



MOVING AVERAGES

- While this statistic weights all data evenly, it may make sense to weight data closer to our date of interest higher.

- We do this by taking a *weighted moving average*, where we assign particular weights to certain time points.

- Various formulas or schemes can be used to weight the data points.

MOVING AVERAGES

- A common weighting scheme is an *exponential weighted moving average (EWMA)* where we add a *decay* term to give less and less weight to older data points.

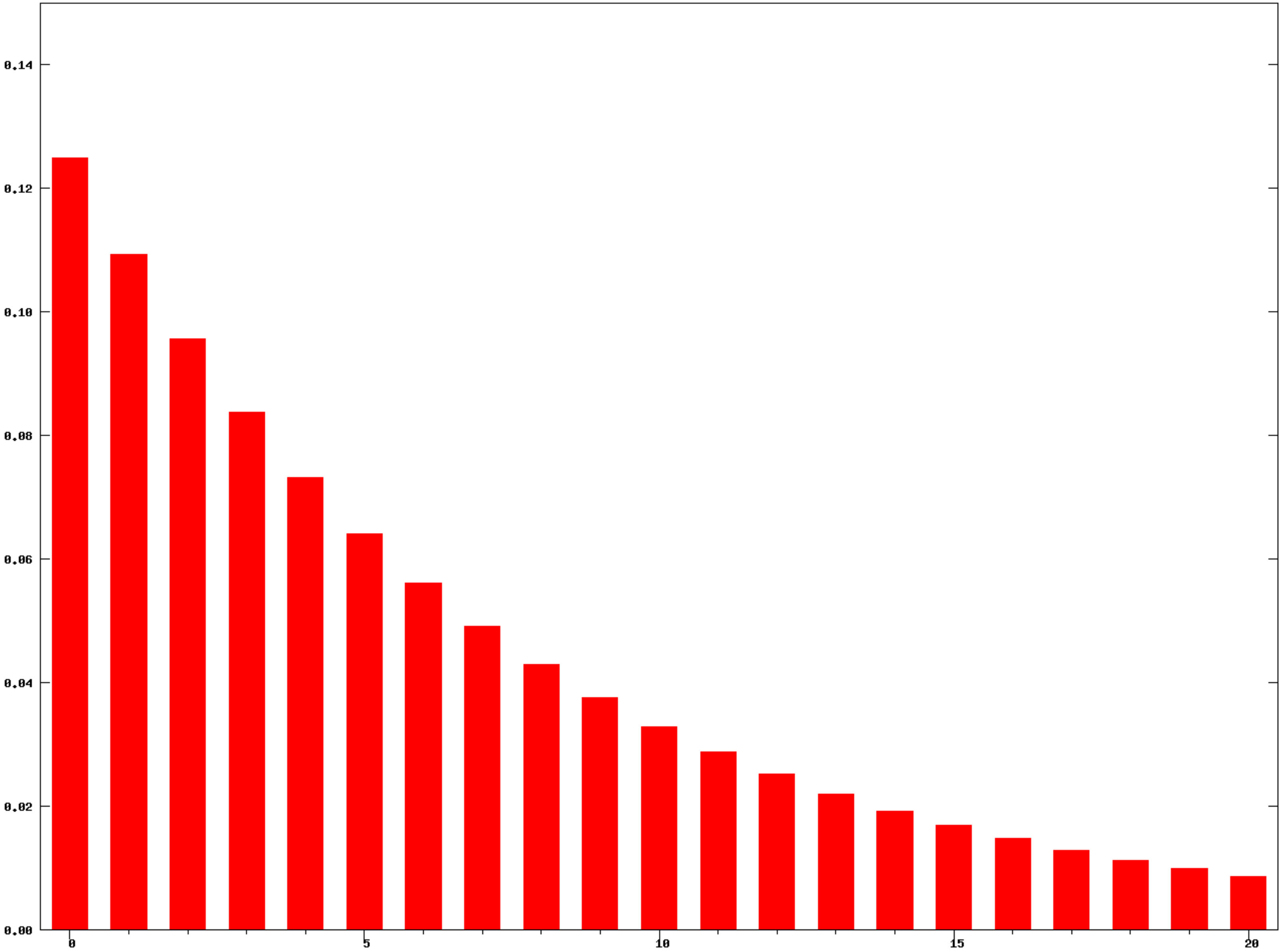
- The EWMA can be calculated recursively for a series Y.

For $t = 1$, $EWMA_1 = Y_1$

For $t > 1$, $EWMA_t = \alpha \bullet Y_t + (1 - \alpha) \bullet EWMA_{t-1}$

MOVING AVERAGES

- The weights for an exponential weighted moving average with k = 15.



AUTOCORRELATION

- In previous classes, we have been concerned with how two variables are correlated (e.g. height and weight, education and salary).

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

AUTOCORRELATION

- To compute autocorrelation, we fix a “lag” *k*. This is how many time points earlier we should use to compute the correlation.

- A lag of 1 computes how correlated a value is with the prior one. A lag of 10 computes how correlated a value is with one 10 time points earlier.

DEMO

EXPLORING ROSSMANN DRUGSTORE SALES DATA

EXPLORING ROSSMANN DRUGSTORE SALES DATA

- Open Lab 15 Demo Code

- We will be using data made available by a German drugstore, Rossmann.

- This data contains the daily sales made at the drugstore as well as whether there was a sale or holiday affecting the sales data.

LOADING THE DATA

- As always, use Pandas to load our data.

```
import pandas as pd
```

```
data = pd.read_csv('../../assets/dataset/rossmann.csv', 
skipinitialspace=True, low_memory=False)
```

LOADING THE DATA

- Because we are most interested in the `Date` column, we can process it as a `DateTime` type and set it as the index of our dataframe.

```
data['Date'] = pd.to_datetime(data['Date'])
data.set_index('Date', inplace=True)

data['Year'] = data.index.year
data['Month'] = data.index.month
```

LOADING THE DATA

- This allows us to easily filter by date. For example, to a particular year:

```
data['2014']
```

- We can also filter to a particular month:

```
data['2015-05']
```

LOADING THE DATA

- There are over a million sales data points in this dataset, so for some analysis we will focus on just one store.

```
store1_data = data[data.Store == 1]
```

PLOTTING THE SALES DATA

- As we begin to study the sales from this drug store, we will also want to know both the time dependent elements of sales as wells as whether promotions or holidays affected sales.

- To start, we can simply compare the average sales on those events.

- To compare sales on holidays, we can compare the sales using box plots. This allows us to compare the distribution of sales on holidays against all other days.

PLOTTING THE SALES DATA

- On state holidays the store is closed (so there should be 0 sales).

- On school holidays, the sales are relatively similar.

```
sb.factorplot(
    col='Open',
    x='SchoolHoliday',
    y='Sales',
    data=store1_data,
    kind='box'
)
```

PLOTTING THE SALES DATA

- We can see there *is* a difference in sales on promotion days.

```
sb.factorplot(
    col='Open',
    x='Promo',
    y='Sales',
    data=store1_data,
    kind='box'
)
```

- Why is it important to separate out days where the store is closed?

PLOTTING THE SALES DATA

- Because there aren’t any promotions on those days either, so including them will bias your sales data on days without promotions.

- Let’s compare sales across days of the week.

```
sb.factorplot(
    col='Open',
    x='DayOfWeek',
    y='Sales',
    data=store1_data,
    kind='box',
)
```

PLOTTING THE SALES DATA

- Lastly, we want to identify larger scale trends in our data.
- How did sales change from 2014 to 2015? Were any particularly interesting outliers in terms of sales or customer visits?
- To plot the sales over time:

```
# Filter to days store 1 was open
store1_open_data = store1_data[store1_data.Open==1]
store1_open_data[['Sales']].plot()
```

PLOTTING THE SALES DATA

- To plot customer visits over time over time:

```
store1_open_data[['Customers']].plot()
```

- We can see that there are large spikes of sales and customers towards the end of 2013 and 2014 leading into the first quarter of 2014 and 2015.

PLOTTING THE SALES DATA

- To filter to the 2015 data:

```
store1_data_2015 = store1_data['2015']
store1_data_2015[store1_data_2015.Open==1][['Sales']].plot()
```

COMPUTING AUTOCORRELATION

- To measure how much the sales are correlated with each other, we want to compute the *autocorrelation* of the "Sales" column.

- In Pandas, we do this with the `autocorr` function. `autocorr` takes one argument, `lag`. This is how many points prior should be used to compute the correlation.

```
data['Sales'].resample('D', how='mean').autocorr(lag=1)
```

- As with correlation between different variables, as this number moves closer to 1, the data is more correlated.

AGGREGATES OF SALES OVER TIME

- If we want to investigate trends over time in sales, we will start by computing simple aggregations. What were the mean and median sales in each year and each month?

- In Pandas, this is performed using the `resample` function, which is very similar to the `groupby` function. It allows us to group over different time periods.

AGGREGATES OF SALES OVER TIME

- We can use `data.resample` and provide the following arguments.
  - A level on which to roll up to: ‘D’ for day, ‘W’ for week, ‘M’ for month, ‘A’ for year.
  - The aggregation to perform: ‘mean’, ‘median’, ‘sum’, etc.

AGGREGATES OF SALES OVER TIME

```
data[['Sales']].resample('A', how=['median', 'mean'])
```

```
data[['Sales']].resample('M', how=['median', 'mean'])
```

- Here we see that December 2013 and 2014 were the highest average sales months.

AGGREGATES OF SALES OVER TIME

- While identifying the monthly averages are useful, we often want to compare the sales data of a date to a smaller window.

- To understand holidays’ sales, we want to compare the sales data of late December to a few days surrounding it.

- We can do this using rolling averages.

AGGREGATES OF SALES OVER TIME

- In Pandas, we can compute the rolling average using the `pd.rolling_mean` or `pd.rolling_median` functions.

```
pd.rolling_mean(data[['Sales']], window=3, center=True, freq='D')
```

- This computes a rolling mean of sales using the sales on each day, the day preceding, and the day following (`window=3` and `center=True`).

AGGREGATES OF SALES OVER TIME

- `rolling_mean` (as well as `rolling_median`) takes the series to aggregate and three important parameters:
  - `window` - the number of days to include in the average
  - `center` - whether the window should be centered on the date or use data prior to that date
  - `freq` - what level to roll up the averages to (as used in resample); `‘D’` for day, `‘W’` for week, `‘M’` for month, `‘A’` for year

AGGREGATES OF SALES OVER TIME

- We can use our index filtering to just look at 2015:

```
pd.rolling_mean(data[['Sales']], window=3, center=True, freq='D')['2015']
```

- Instead of plotting the full time series, we can plot the rolling mean instead. This smooths random changes in sales as well as removing outliers, helping us identify larger trends.

```
pd.rolling_mean(data[['Sales']], window=10, center=True, freq='D').plot()
```

AGGREGATES OF SALES OVER TIME

- As we discussed earlier, this averages all values in the window evenly. However we may want to weight closer values more.

- For example, for a centered weighted average of 10 days, we want to put emphasis on +/- 1 day versus +/- 5 days.

- One option to do that is the `ewma` function or the `exponential weighted moving average` function.

```
pd.ewma(data['Sales'], span=10)
```

PANDAS WINDOW FUNCTIONS

- Pandas `rolling_mean` and `rolling_median` are only two examples of Pandas window function capabilities.

- Window functions operate on a set of N consecutive rows (a window) and produce output.

- In addition, there are `rolling_sum`, `rolling_min`, `rolling_max`, and many more.

PANDAS WINDOW FUNCTIONS

- Another common window function is `diff`, which takes the difference over time.

- `pd.diff` takes one argument, `periods`, which is how many rows prior to use for the difference.

- For example, if we want to compute the difference in sales, day by day:

```
data['Sales'].diff(periods=1)
```

PANDAS WINDOW FUNCTIONS

- However, if we wanted to compare the same day in the prior week, we could set `periods=7`.

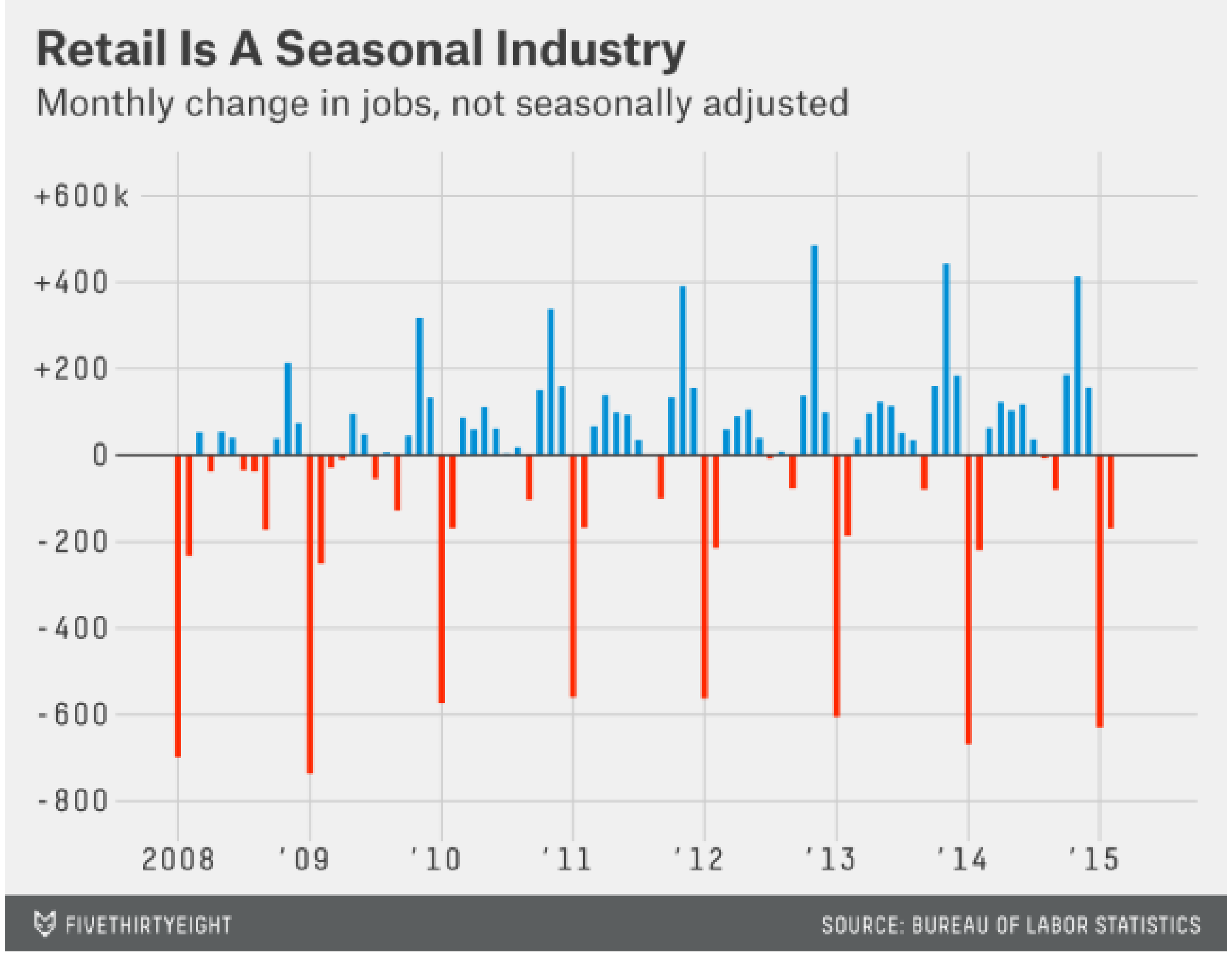
```
data['Sales'].diff(periods=7)
```

- This would compute the difference in sales, from every day to the same day in the previous week.

WHAT IS TIME SERIES DATA?

- Difference functions allow us to identify seasonal changes as we see repeated up or down swings.

- This plot of the month to month change (`diff`) in retail jobs helps identify the seasonal component of the number of retail jobs.



PANDAS EXPANDING FUNCTIONS

- In addition to the set of “rolling” functions, Pandas also provides a similar set of “expanding” functions.

- Instead of using a window of N values, “expanding” functions use all values up until that time.

PANDAS EXPANDING FUNCTIONS

- We can compute the average sales from the first date *until* the date specified.

```
pd.expanding_mean(data['Sales'], freq='d')
```

- We can also compute the sum of average sales per store up until that date.

```
pd.expanding_sum(data['Sales'], freq='d')
```

INDEPENDENT PRACTICE

TIME SERIES EXERCISES

ACTIVITY: TIME SERIES EXERCISES

EXERCISE

DIRECTIONS (Open Lab 15 Starter Code)

1. Plot the distribution of sales by month and compare the effect of promotions.
2. Are sales more correlated with the prior date, a similar date last year, or a similar date last month?
3. Plot the 15 day rolling mean of customers in the stores.
4. Identify the date with largest drop in sales from the same date in the previous month.
5. Compute the total sales up until Dec. 2014.
6. When were the largest differences between 15-day moving/rolling averages? **HINT**: Using `rolling_mean` and `diff`

DELIVERABLE

Plots and answers to the above questions

INDEPENDENT PRACTICE REVIEW

TIME SERIES EXERCISES

TIME SERIES EXERCISES REVIEW

- Plot the distribution of sales by month and compare the effect of promotions.

```
sb.factorplot(
    col='Open',
    hue='Promo',
    x='Month',
    y='Sales',
    data=store1_data,
    kind='box'
)
```

TIME SERIES EXERCISES REVIEW

- Are sales more correlated with the prior date, a similar date last year, or a similar date last month?

```
# Compare the following:
average_daily_sales = data[['Sales', 'Open']].resample('D', how='mean')

average_daily_sales['Sales'].autocorr(lag=1)

average_daily_sales['Sales'].autocorr(lag=30)

average_daily_sales['Sales'].autocorr(lag=365)
```

TIME SERIES EXERCISES REVIEW

- Plot the 15 day rolling mean of customers in the stores.

```
pd.rolling_mean(data[['Sales']], window=15, freq='D').plot()
```

TIME SERIES EXERCISES REVIEW

- Identify the date with largest drop in sales from the same date in the previous month.

```
average_daily_sales = data[['Sales', 'Open']].resample('D', how='mean')
average_daily_sales['DiffVsLastWeek'] =
average_daily_sales[['Sales']].diff(periods=7)

average_daily_sales.sort(['DiffVsLastWeek']).head
```

- Unsurprisingly this day is December 25th and 26th in 2014 and 2015, when the store is closed and there were many sales in the preceding week.

TIME SERIES EXERCISES REVIEW

- Compute the total sales up until Dec. 2014.

```
total_daily_sales = data[['Sales']].resample('D', how='sum')
pd.expanding_sum(total_daily_sales)['2014-12']
```

- Note that this is **NOT**

```
pd.expanding_sum(data['Sales'], freq='D')
```

- We don’t want to average over stores first!

TIME SERIES EXERCISES REVIEW

- When were the largest differences between 15-day moving/rolling averages? **HINT**: Using `rolling_mean` and `diff`

```
pd.rolling_mean(data[['Sales']], window=15, 
freq='D').diff(1).sort('Sales')
```

- Unsurprisingly, they occur at the beginning of every year after the holiday season.

CONCLUSION

TOPIC REVIEW

CONCLUSION

- 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.

COURSE

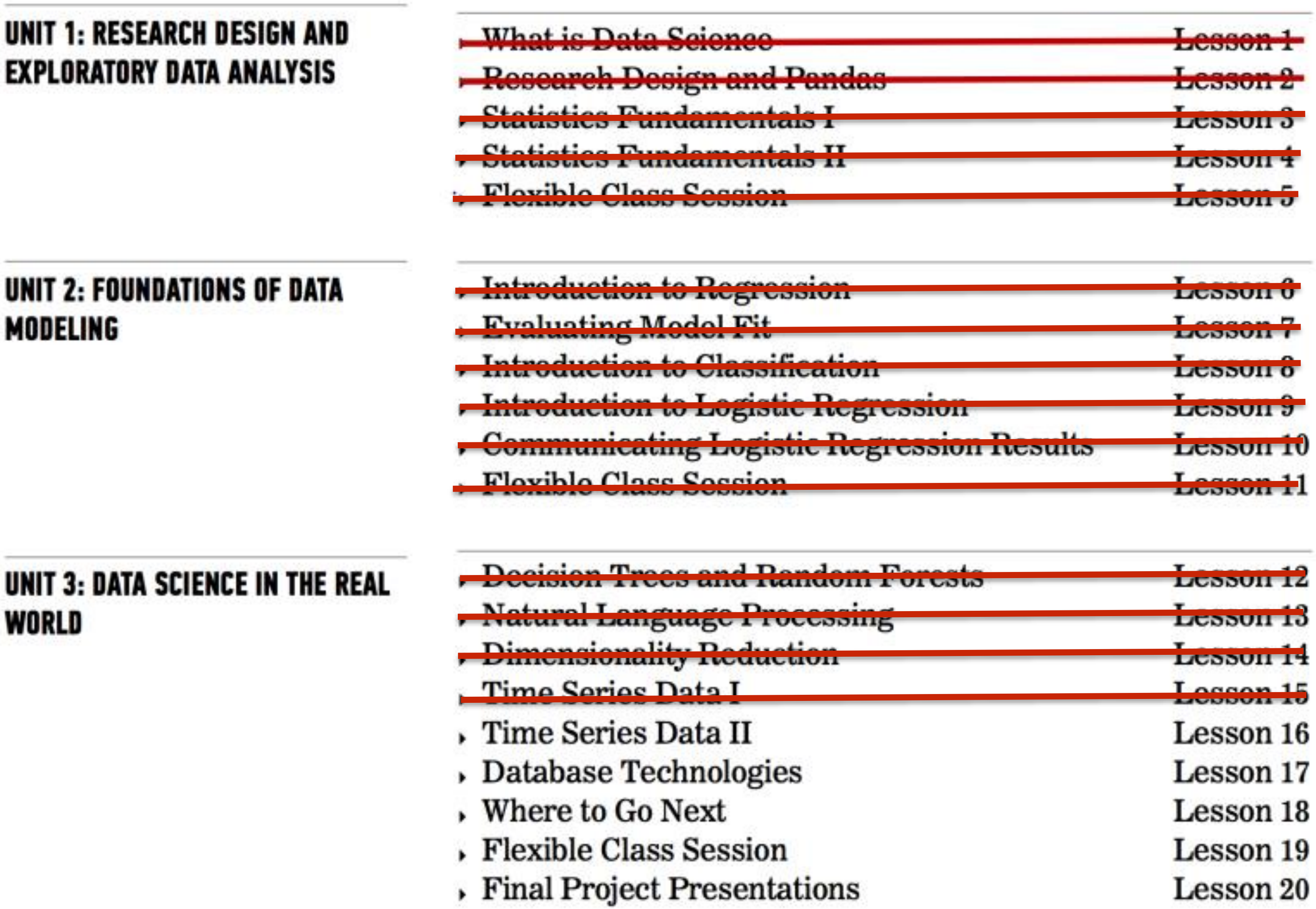
BEFORE NEXT CLASS

BEFORE NEXT CLASS

DUE DATE

- Project: Final Project, Part 3

OUR PROGRESS SO FAR



UNIT 1: RESEARCH DESIGN AND EXPLORATORY DATA ANALYSIS

~~What is Data Science~~	~~Lesson 1~~
~~Research Design and Pandas~~	~~Lesson 2~~
~~Statistics Fundamentals I~~	~~Lesson 3~~
~~Statistics Fundamentals II~~	~~Lesson 4~~
~~Flexible Class Session~~	~~Lesson 5~~

UNIT 2: FOUNDATIONS OF DATA MODELING

~~Introduction to Regression~~	~~Lesson 6~~
~~Evaluating Model Fit~~	~~Lesson 7~~
~~Introduction to Classification~~	~~Lesson 8~~
~~Introduction to Logistic Regression~~	~~Lesson 9~~
~~Communicating Logistic Regression Results~~	~~Lesson 10~~
~~Flexible Class Session~~	~~Lesson 11~~

UNIT 3: DATA SCIENCE IN THE REAL WORLD

~~Decision Trees and Random Forests~~	~~Lesson 12~~
~~Natural Language Processing~~	~~Lesson 13~~
~~Dimensionality Reduction~~	~~Lesson 14~~
~~Time Series Data I~~	~~Lesson 15~~
Time Series Data II	Lesson 16
Database Technologies	Lesson 17
Where to Go Next	Lesson 18
Flexible Class Session	Lesson 19
Final Project Presentations	Lesson 20

Next Class

LESSON

EXIT TICKET

DON’T FORGET TO FILL OUT YOUR EXIT TICKET