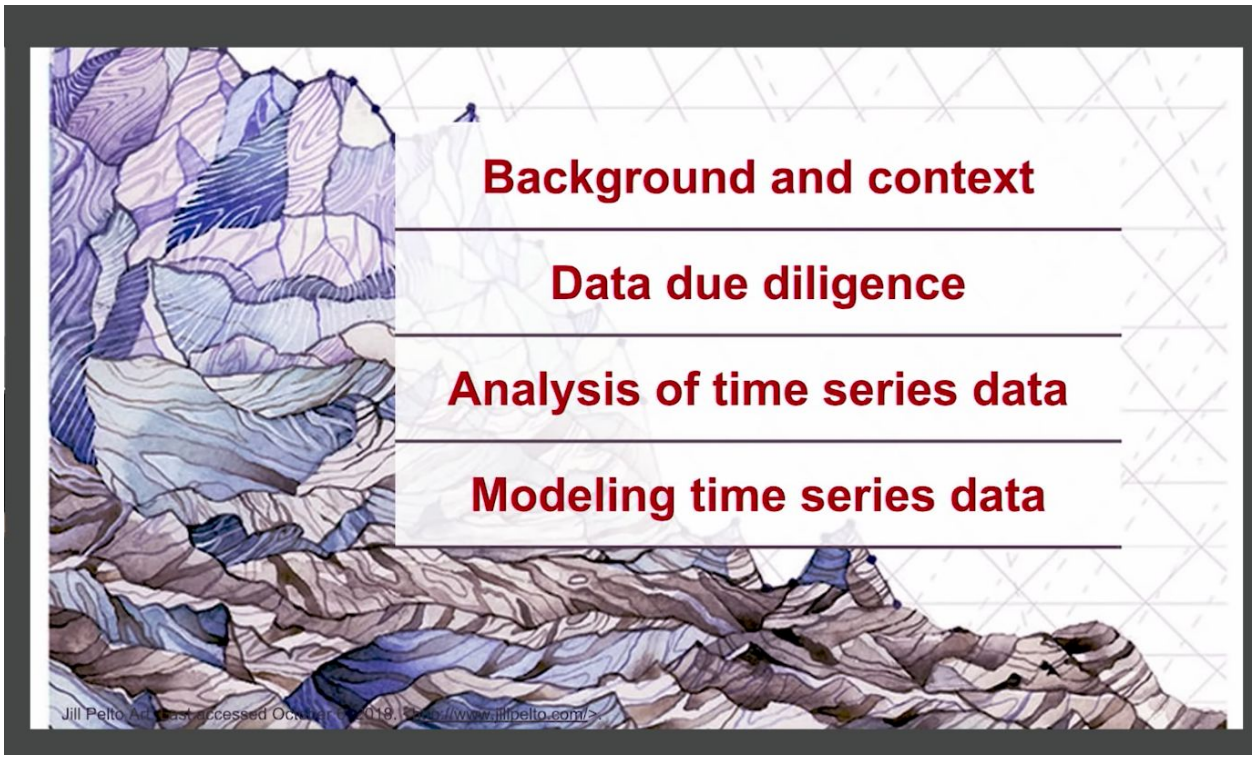
Tamara Louie: Applying Statistical Modeling & Machine Learning to Perform Time-Series Forecasting

PyData channel

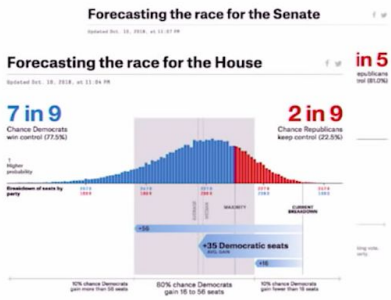
https://www.youtube.com/watch?v=JntA9XaTebs

https://colab.research.google.com/drive/1oBXX3k3OOjdivwlDhpLmAqfq5B5dzFR1#scrollTo=PiFJF8dl-65Q



Why is forecasting important (or at least, interesting)?

Applications in many fields, including politics, finance, health, etc.



Source of images, from left to right. 1. FiveThirtyEight. Last updated October 10, 2018. Last accessed October 10, 2018. Link. 2. Bloomberg. Last accessed October 10, 2018. Link. FiveThirtyEight. Last updated April 2, 2018. Last updated October 10, 2018. Link.

What is time-series data? Do I need time-series data to answer this question?

- Time-series data can be defined in many ways. I tend to describe it as "data collected on the same metric or same object at regular or irregular time intervals."
- In terms of whether one needs time-series data depends on the question.
- I think about whether there is an inherent relationship or structure between data at various time points (e.g., is there a time-dependence), and whether we can leverage that time-ordered information.

Source of images. Alice Oglethorpe, Fitbit. Link. Last accessed October 10, 2018.

Is additional processing necessary?

- **Are there nulls?** Yes, there are a lack of entries (i.e., rows) for some dates.
- **What are the data types? Units?** Need to change the dates to date type. Units appear okay.

Look at stationarity

Most time-series models assume that the underlying time-series data is **stationary**. This assumption gives us some nice statistical properties that allow us to use various models for forecasting.

Stationarity is a statistical assumption that a time-series has:

- **Constant mean**
- **Constant variance**
- **Autocovariance does not depend on time**

More simply put, if we are using past data to predict future data, we should assume that the data will follow the same general trends and patterns as in the past. This general statement holds for most training data and modeling tasks.

There are some good diagrams and explanations on stationarity here and here.

Sometimes we need to transform the data in order to make it stationary. However, this transformation then calls into question if this data is truly stationary and is suited to be modeled using these techniques.

Looking at our data:

- Rolling mean and standard deviation look like they change over time. There may be some de-trending and removing seasonality involved. Based on the **Dickey-Fuller test**, because p = 0.31, we fail to reject the null hypothesis (that the time series is not stationary) at the p = 0.05 level, thus concluding that we fail to reject the null hypothesis that our **time series is not stationary**.

Correct for stationarity

It is common for time series data to have to be corrected for non-stationarity.

2 common reasons behind non-stationarity are:

1. **Trend** – mean is not constant over time.
2. **Seasonality** – variance is not constant over time.

There are ways to correct for trend and seasonality, to make the time series stationary.

What happens if you do not correct these things?

Many things can happen, including:

- Variance can be mis-specified
- Model fit can be worse.
- Not leveraging the valuable time-dependent nature of the data.

Here are some resources on the pitfalls of using traditional methods for time series analysis.

Quora link

Quora link

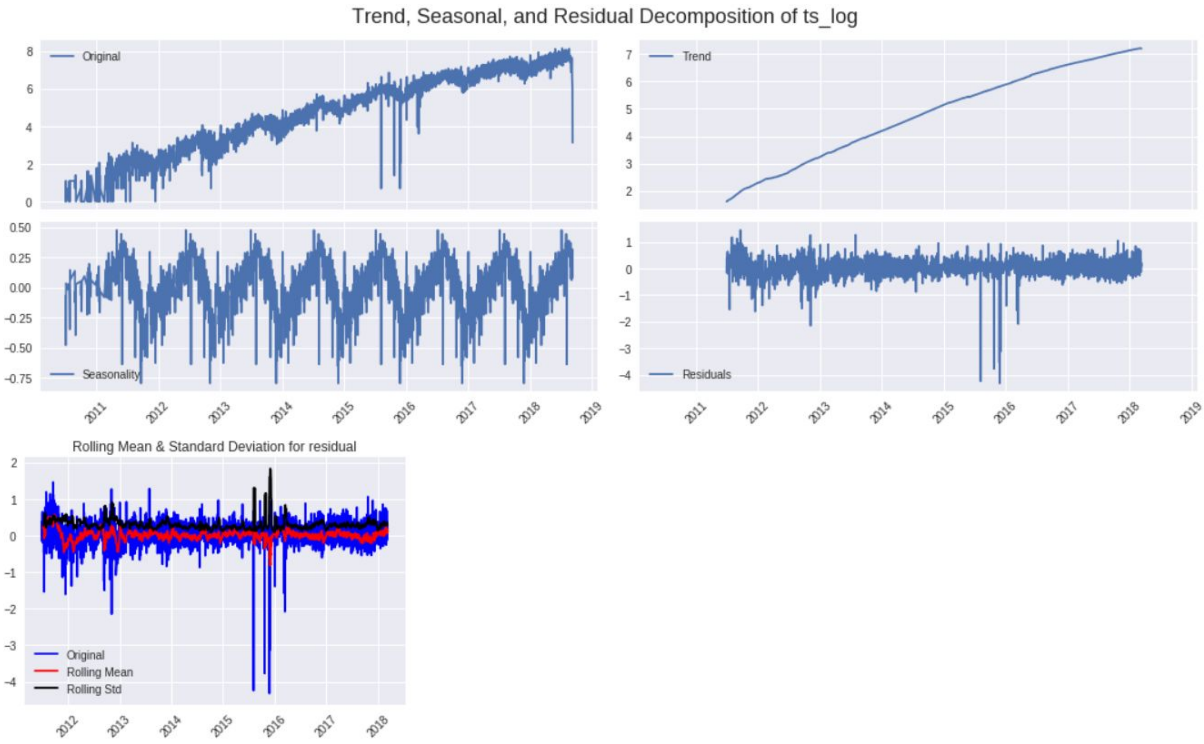
Eliminating trend and seasonality

- **Transformation**
  - *Examples.* Log, square root, etc.
  - We are going to look at the log.
- **Smoothing**
  - *Examples.* Weekly average, monthly average, rolling averages.
  - We are going to look at the weekly average.
- **Differencing**
  - *Examples.* First-order differencing.
  - We are going to look at first-order differencing.
- **Polynomial Fitting**
  - *Examples.* Fit a regression model.
- **Decomposition**

Decomposition: trend, seasonality, residuals

Looking at our data:

- De-trending and de-seasonalizing made the data (i.e., the residuals) more stationary over time. Based on the **Dickey-Fuller test**, because p = < 0.05, we fail to reject the null hypothesis (that the time series is not stationary) at the p = 0.05 level, thus concluding that the **time series is stationary**.



Let us model some time-series data! Finally! ARIMA models.

We will be doing an example here! We can use ARIMA models when we know there is dependence between values and we can leverage that information to forecast.

ARIMA = Auto-Regressive Integrated Moving Average.

Assumptions. The time-series is stationary.

Depends on:

1. Number of AR (Auto-Regressive) terms (p).
2. Number of I (Integrated or Difference) terms (d).
3. Number of MA (Moving Average) terms (q).

ACF and PACF Plots

How do we determine p, d, and q? For p and q, we can use ACF and PACF plots (below).

Autocorrelation Function (ACF). Correlation between the time series with a lagged version of itself (e.g., correlation of Y(t) with Y(t-1)).

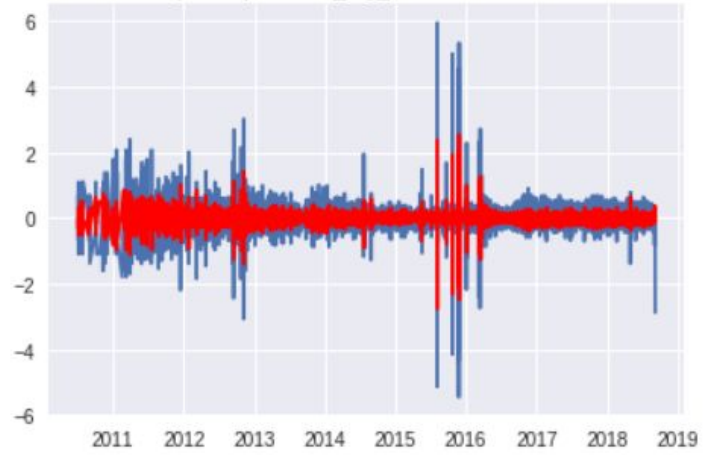
Partial Autocorrelation Function (PACF). Additional correlation explained by each successive lagged term.

How do we interpret ACF and PACF plots?

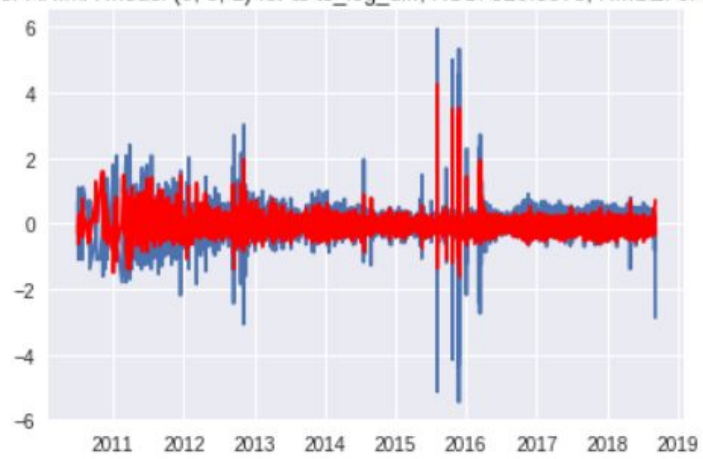
- p – Lag value where the PACF chart crosses the upper confidence interval for the first time.
- q – Lag value where the ACF chart crosses the upper confidence interval for the first time.

```
# Note: I do the differencing in the transformation of the data 'ts_log_diff'
# AR model with 1st order differencing - ARIMA (1,0,0)
model_AR = run_arima_model(df = df_example_transform,
                           ts = 'ts_log_diff',
                           p = 1,
                           d = 0,
                           q = 0)

# MA model with 1st order differencing - ARIMA (0,0,1)
model_MA = run_arima_model(df = df_example_transform,
                           ts = 'ts_log_diff',
                           p = 0,
                           d = 0,
                           q = 1)

# ARMA model with 1st order differencing - ARIMA (1,0,1)
model_MA = run_arima_model(df = df_example_transform,
                           ts = 'ts_log_diff',
                           p = 1,
                           d = 0,
                           q = 1)
```

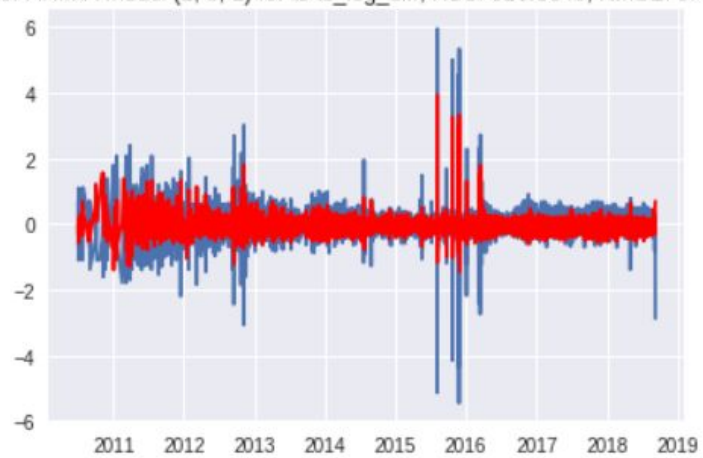
For ARIMA model (1, 0, 0) for ts ts_log_diff, RSS: 638.4617, RMSE: 0.4825



For ARIMA model (0, 0, 1) for ts ts_log_diff, RSS: 529.3078, RMSE: 0.4393



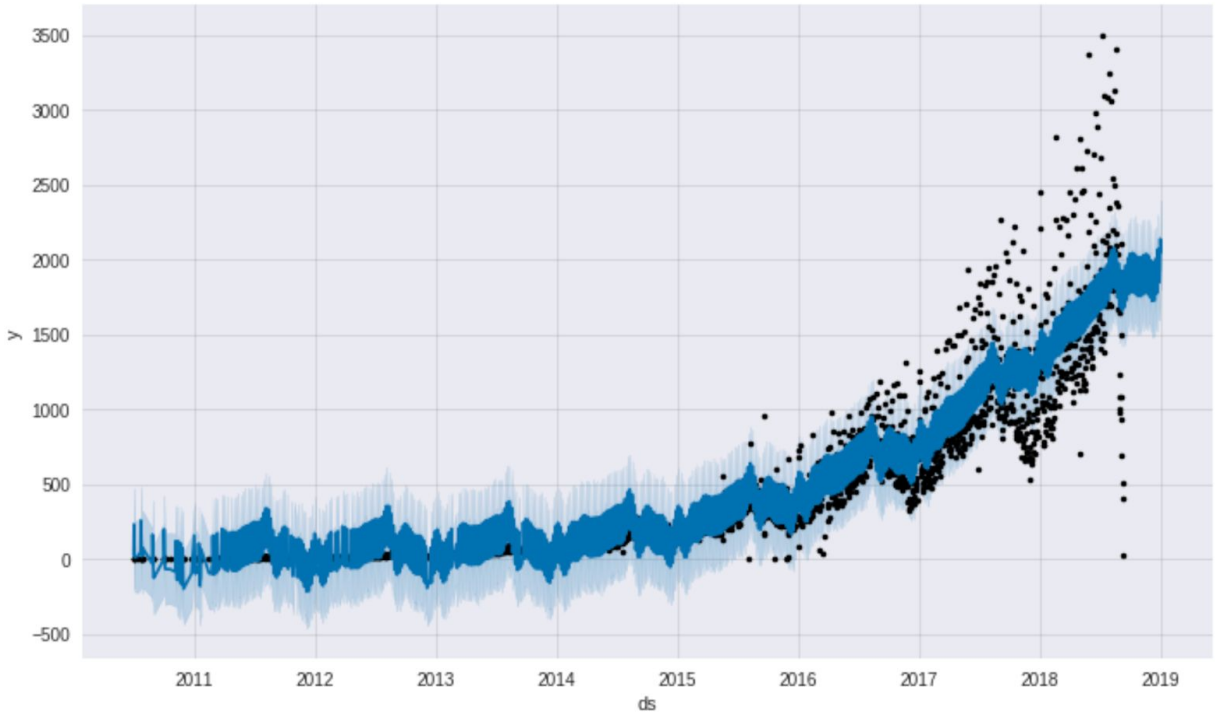
For ARIMA model (1, 0, 1) for ts ts_log_diff, RSS: 526.8646, RMSE: 0.4383



Let us model some time-series data! Finally! Facebook Prophet package.

We will be doing an example here! Installing the necessary packages might take a couple of minutes. In the meantime, I can talk a bit about Facebook Prophet, a tool that allows folks to forecast using additive or component models relatively easily. It can also include things like:

- Day of week effects
- Day of year effects
- Holiday effects
- Trend trajectory
- Can do MCMC sampling

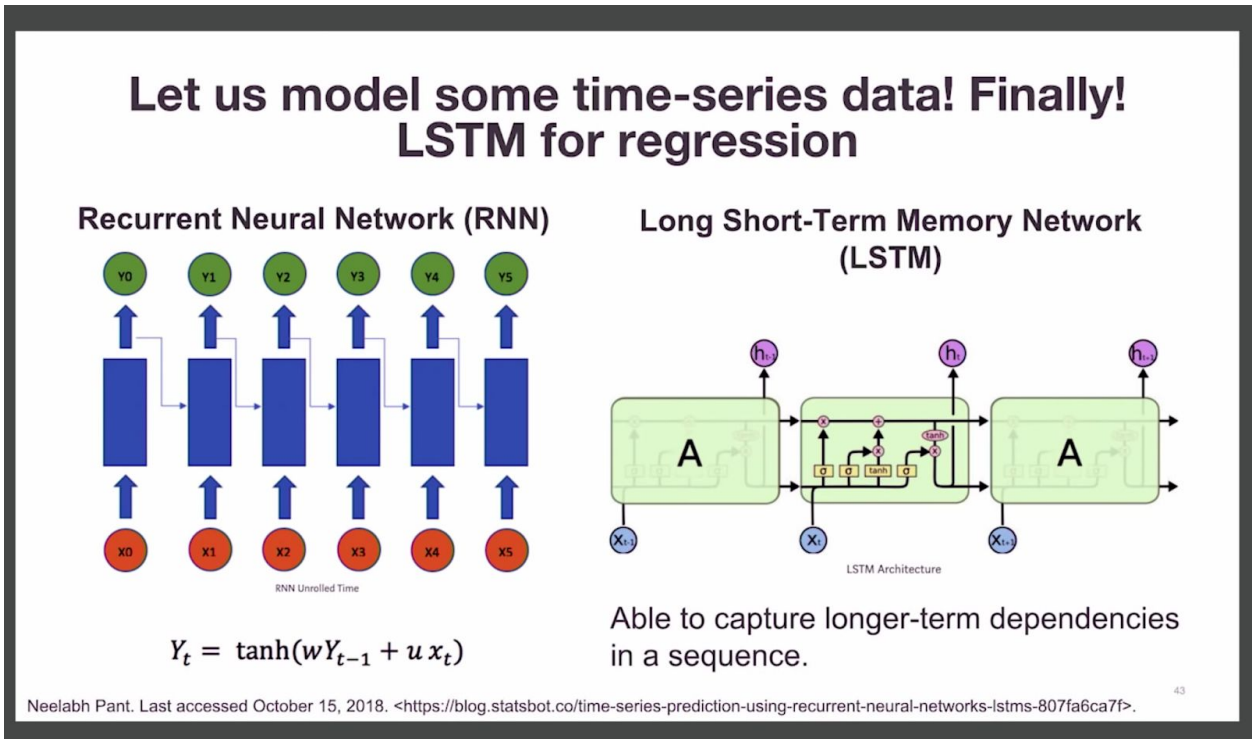


Let us model some time-series data! Finally! LSTM for regression

We will be going through an example here.

Also, here are some resources on recurrent neural networks (RNN) and Long Short-Term Memory networks (LSTMs):

- Link 1
- Link 2
- Link 3



```
Epoch 1/5
 - 4s - loss: 8.1068e-04
Epoch 2/5
 - 3s - loss: 4.2630e-04
Epoch 3/5
 - 3s - loss: 4.2030e-04
Epoch 4/5
 - 3s - loss: 4.2552e-04
Epoch 5/5
 - 3s - loss: 4.1484e-04
Train Score: 81.10 RMSE
Test Score: 380.51 RMSE
```