Predicting the Future with Python
(in 11 lines of code)

•••

Chris Moffitt @chris1610
pbpython.com
June 8, 2017

pymntos
Twin Cities Python User Group

I see an awesome
presentation in your future...

Who am I?

Python user since ~2000

Experience with unix admin, django development, data science tools

Blog author - Practical Business Python (pbpython.com)

Interested in using power of python to solve problems (aka avoid Excel hell)

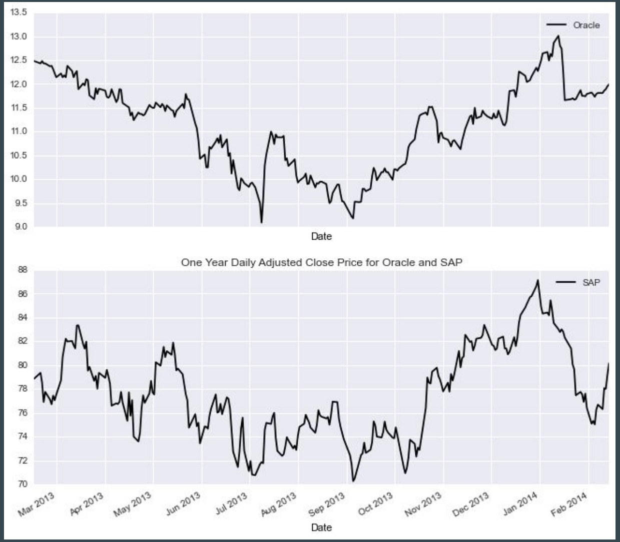
chris@moffitts.net

@chris1610

What is Time Series Forecasting?

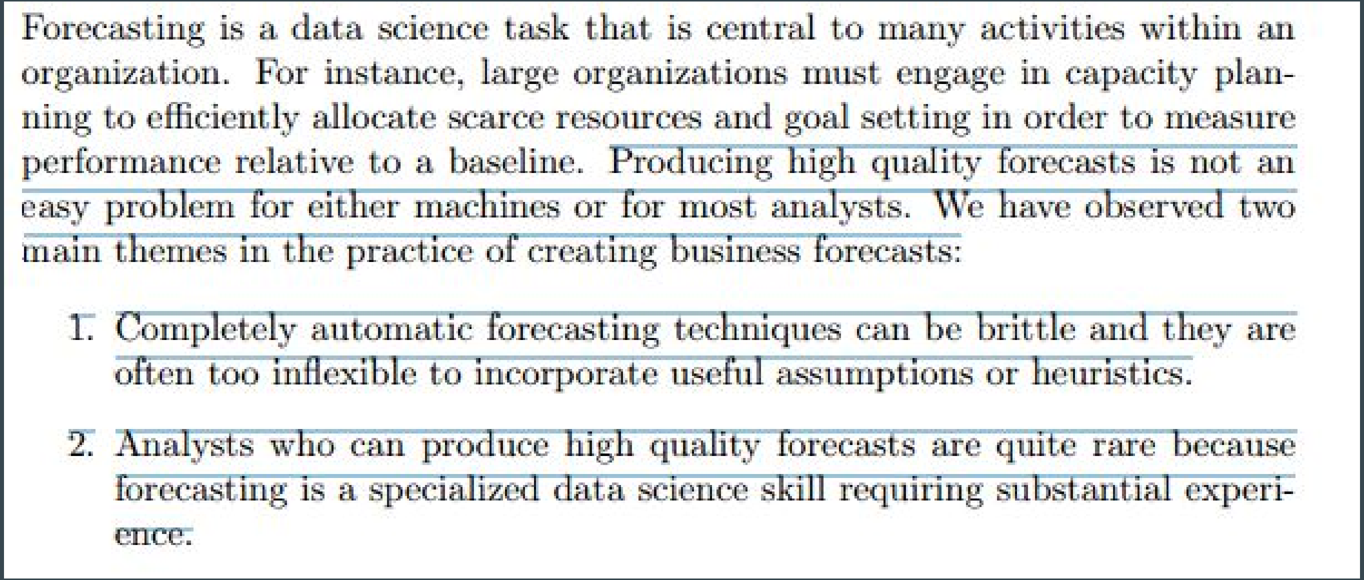
“Time series forecasting is the use of a model to predict future values based on previously observed values.”

- Stocks
- web traffic
- Sales
- Store visits



http://kldavenport.com/dynamic-time-series-modeling/

Forecasting is a common need but it’s hard to do


Forecasting is a data science task that is central to many activities within an organization. For instance, large organizations must engage in capacity planning to efficiently allocate scarce resources and goal setting in order to measure performance relative to a baseline. Producing high quality forecasts is not an easy problem for either machines or for most analysts. We have observed two main themes in the practice of creating business forecasts:

1. Completely automatic forecasting techniques can be brittle and they are often too inflexible to incorporate useful assumptions or heuristics.

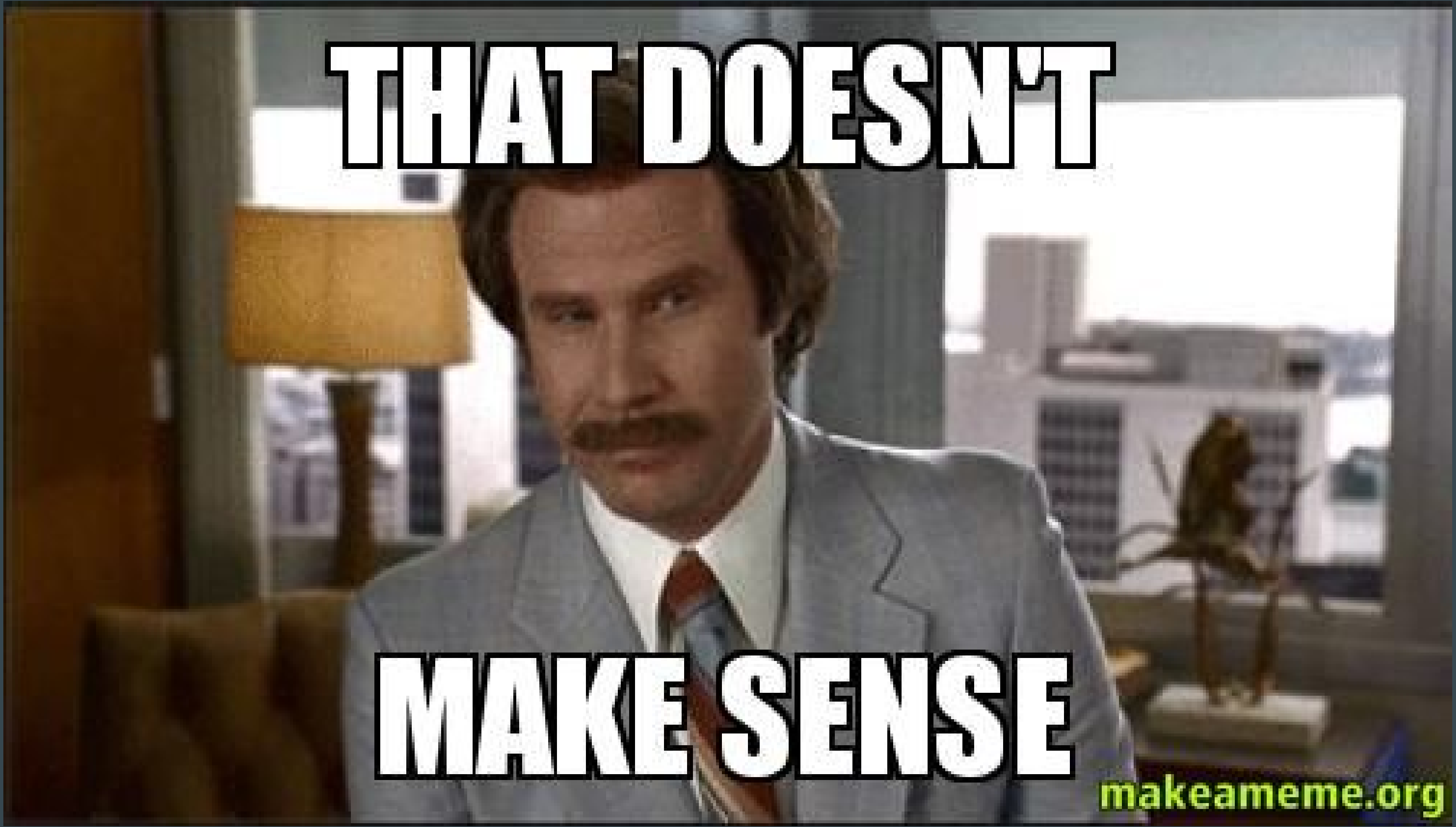
2. Analysts who can produce high quality forecasts are quite rare because forecasting is a specialized data science skill requiring substantial experience.

https://facebookincubator.github.io/prophet/static/prophet_paper_20170113.pdf

That’s why Facebook developed prophet

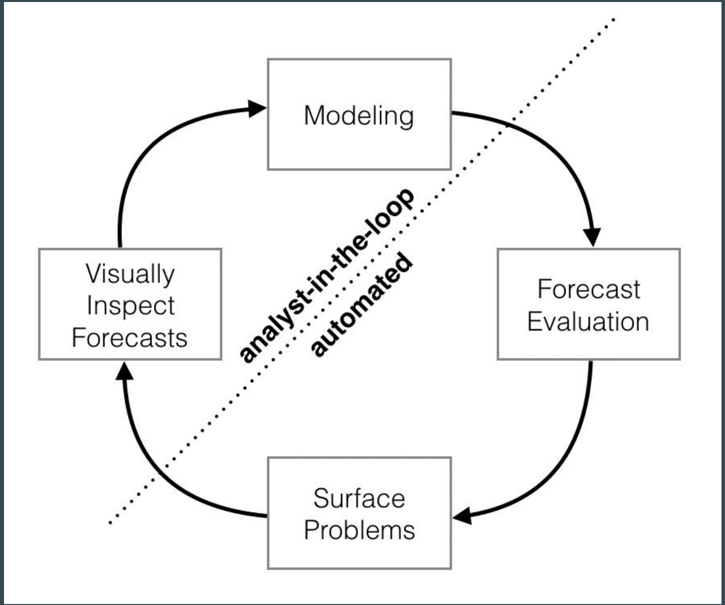
“Prophet is a procedure for forecasting time series data. It is based on an additive model where non-linear trends are fit with yearly and weekly seasonality, plus holidays. It works best with daily periodicity data with at least one year of historical data. Prophet is robust to missing data, shifts in the trend, and large outliers.”

https://facebookincubator.github.io/prophet/


THAT DOESN'T
MAKE SENSE
makeameme.org

What is prophet?

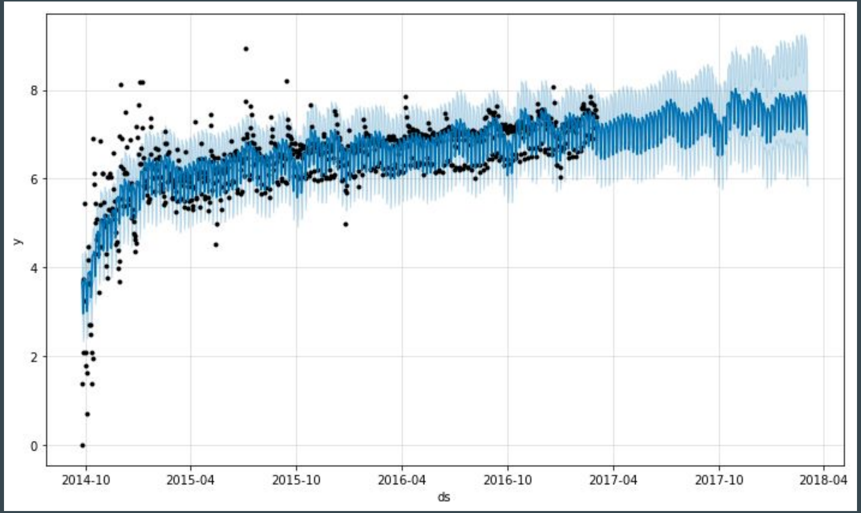
- Used internally by Facebook to make time series forecasts
- Open Source
- Accurate and Fast
- Fully Automatic
- Tunable options
- Built on stan but has python and R bindings



https://research.fb.com/prophet-forecasting-at-scale/

Why I like prophet

- Powerful forecasting algorithms
  - See this post for a more complex approach: https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python/
- Only need basic python knowledge to build a prediction (11 lines of code)
- Options to tweak over time
- Utilizes familiar python data science stack:
  - pandas
  - matplotlib
  - scikit-learn

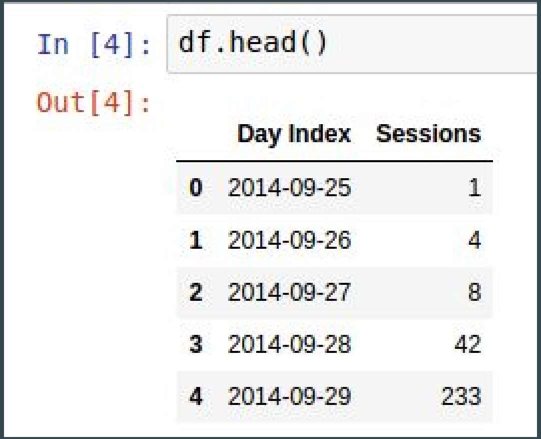


Installation Process

```
conda create -n pymntos python=3

source activate pymntos

conda install pandas

conda install xlrd

conda install jupyter

conda install -c conda-forge fbprophet
```

Get coding (use jupyter notebook)

```
import pandas as pd

import numpy as np

from fbprophet import Prophet

# Read in the source data - downloaded from google analytics
df = pd.read_excel('http://bit.ly/2r8dnji')
```



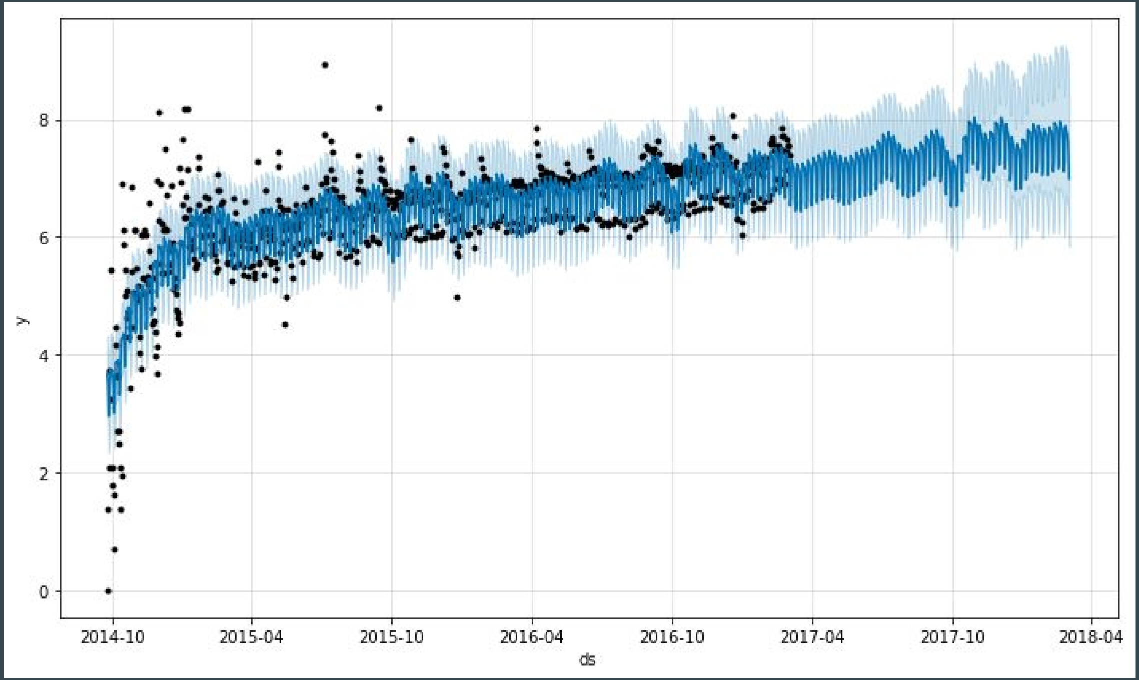
```
In [4]: df.head()

Out[4]:
     Day Index   Sessions
0   2014-09-25          1
1   2014-09-26          4
2   2014-09-27          8
3   2014-09-28         42
4   2014-09-29        233
```

Format the data

```
# Convert to log format
df['Sessions'] = np.log(df['Sessions'])

# Need to name the columns like this in order for prophet to work
df.columns = ["ds", "y"]

# Create the model
m1 = Prophet()
m1.fit(df)

# Predict out a year
future1 = m1.make_future_dataframe(periods=365)
forecast1 = m1.predict(future1)
```

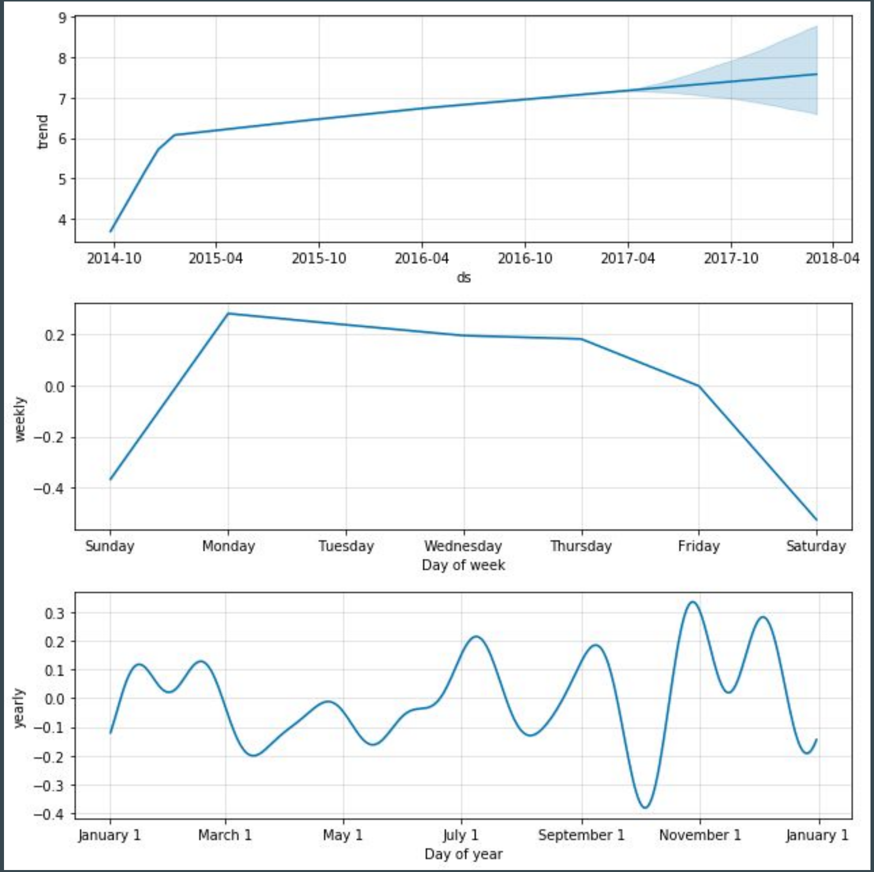
Behold the forecast

```
m1.plot(forecast1);
```



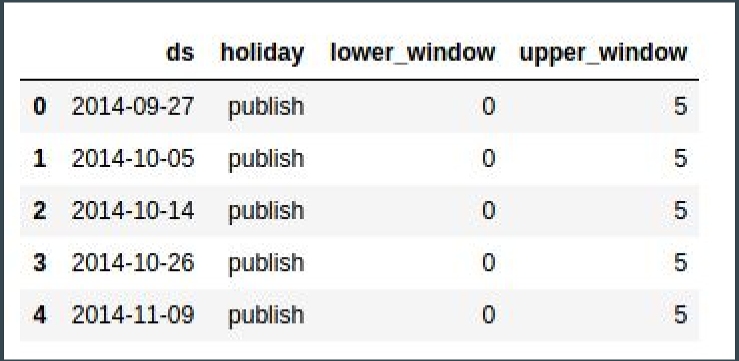
Bonus Analysis

```
m1.plot_components(forecast1);
```



How can I tailor it? Use Holiday Effects...

```
articles = pd.DataFrame({
  'holiday': 'publish',
  'ds': pd.to_datetime(['2014-09-27', '2014-10-05', '2014-10-14', '2014-10-26', '2014-11-9',
                        '2014-11-18', '2014-11-30', '2014-12-17', '2014-12-29', '2015-01-06',
                        '2015-01-20', '2015-02-02', '2015-02-16', '2015-03-23', '2015-04-08',
                        '2015-05-04', '2015-05-17', '2015-06-09', '2015-07-02', '2015-07-13',
                        '2015-08-17', '2015-09-14', '2015-10-26', '2015-12-07', '2015-12-30',
                        '2016-01-26', '2016-04-06', '2016-05-16', '2016-06-15', '2016-08-23',
                        '2016-08-29', '2016-09-06', '2016-11-21', '2016-12-19', '2017-01-17',
                        '2017-02-06', '2017-02-21', '2017-03-06']),
  'lower_window': 0,
  'upper_window': 5,
})
```

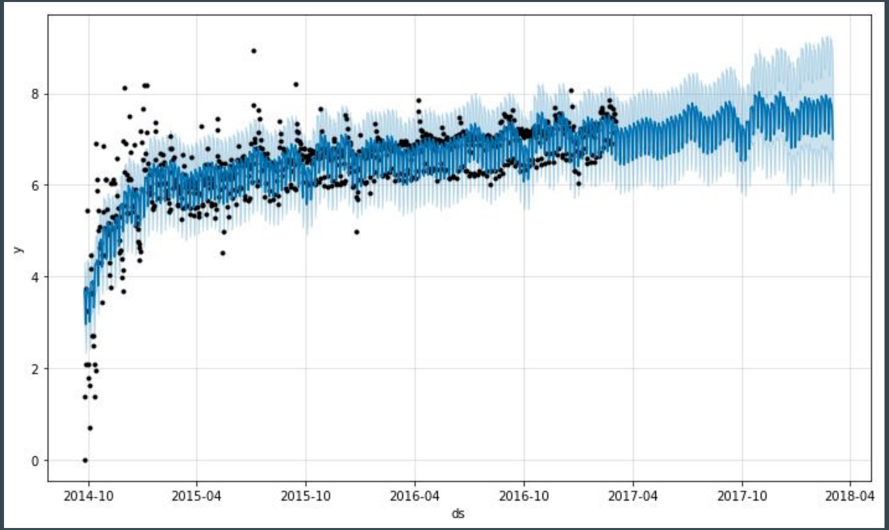


	ds	holiday	lower_window	upper_window
0	2014-09-27	publish	0	5
1	2014-10-05	publish	0	5
2	2014-10-14	publish	0	5
3	2014-10-26	publish	0	5
4	2014-11-09	publish	0	5

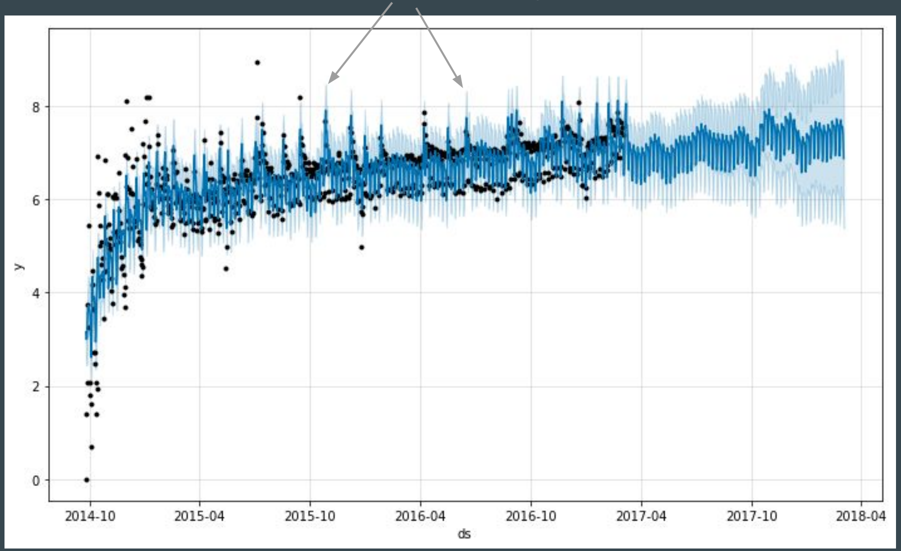
Create a new model with the holidays

```
m2 = Prophet(holidays=articles).fit(df)
future2 = m2.make_future_dataframe(periods=365)
forecast2 = m2.predict(future2)
m2.plot(forecast2);
```

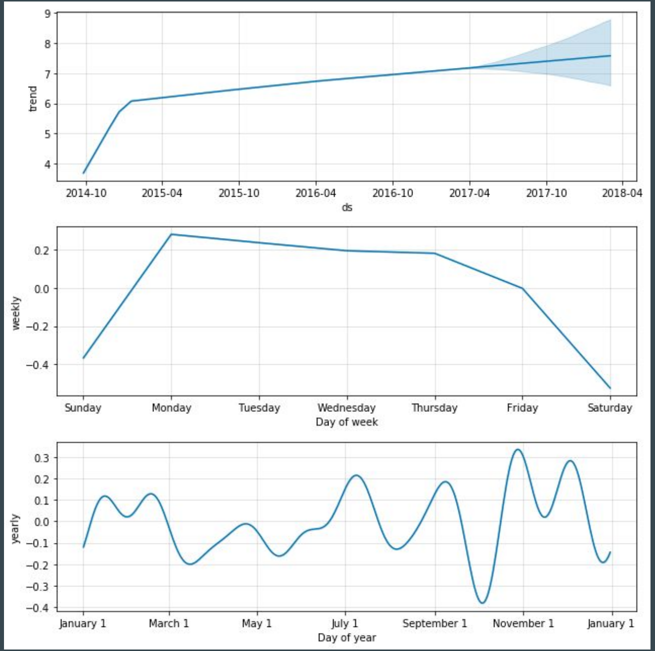
No holiday

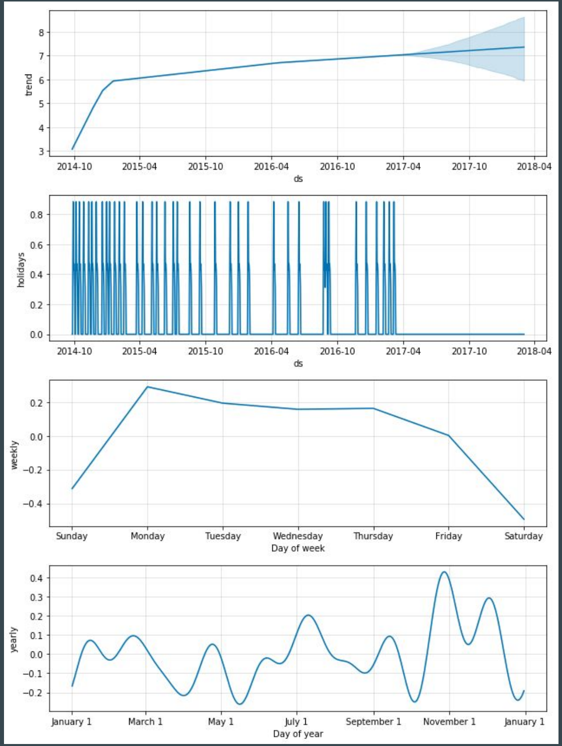


With holiday



Holiday Effects





Can include "future holidays" to make the prediction more accurate

Full Code Example

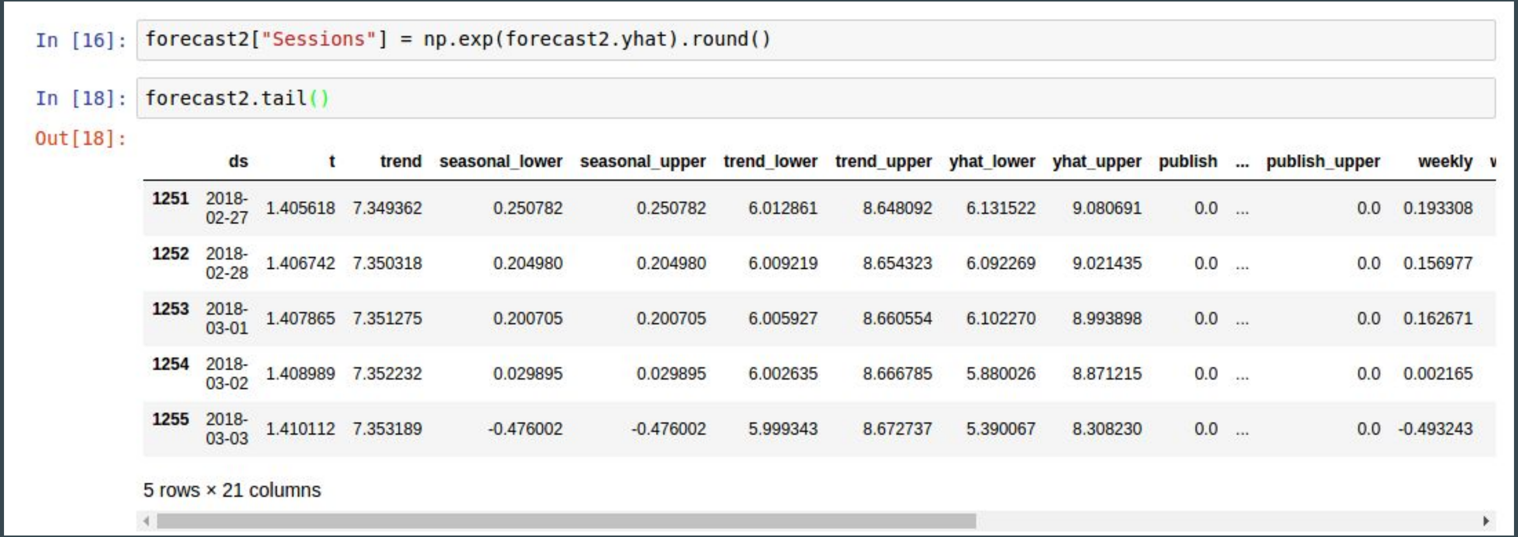
```
import pandas as pd
import numpy as np
from fbprophet import Prophet

# Read in the source data - downloaded from google analytics
df = pd.read_excel('https://github.com/chris1610/pbpython/blob/master/data/All-Web-Site-Data-Audience-Overview.xlsx?raw=True')

# Convert to log format
df['Sessions'] = np.log(df['Sessions'])

# Need to name the columns like this in order for prophet to work
df.columns = ["ds", "y"]

# Create the model
m1 = Prophet()
m1.fit(df)

# Predict out a year
future1 = m1.make_future_dataframe(periods=365)
forecast1 = m1.predict(future1)

m1.plot(forecast1)
```

11 lines of code
(not including comments)

Additional Notes

The forecast is just a pandas dataframe so “raw” data is available

Don’t forget to convert results back using `np.exp`



```
In [16]: forecast2["Sessions"] = np.exp(forecast2.yhat).round()
```

```
In [18]: forecast2.tail()
```

Out[18]:

	ds	t	trend	seasonal_lower	seasonal_upper	trend_lower	trend_upper	yhat_lower	yhat_upper	publish	...	publish_upper	weekly
1251	2018-02-27	1.405618	7.349362	0.250782	0.250782	6.012861	8.648092	6.131522	9.080691	0.0	...	0.0	0.193308
1252	2018-02-28	1.406742	7.350318	0.204980	0.204980	6.009219	8.654323	6.092269	9.021435	0.0	...	0.0	0.156977
1253	2018-03-01	1.407865	7.351275	0.200705	0.200705	6.005927	8.660554	6.102270	8.993898	0.0	...	0.0	0.162671
1254	2018-03-02	1.408989	7.352232	0.029895	0.029895	6.002635	8.666785	5.880026	8.871215	0.0	...	0.0	0.002165
1255	2018-03-03	1.410112	7.353189	-0.476002	-0.476002	5.999343	8.672737	5.390067	8.308230	0.0	...	0.0	-0.493243

5 rows × 21 columns

I will make the notebook available on github for those interested