Road map!

- Module 1- Introduction to Deep Forecasting
- Module 2- Setting up Deep Forecasting Environment
- Module 3- Exponential Smoothing
- Module 4- ARIMA models
- Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

Module 8- Prophet and Neural Prophet

PROPHET

Neural Prophet

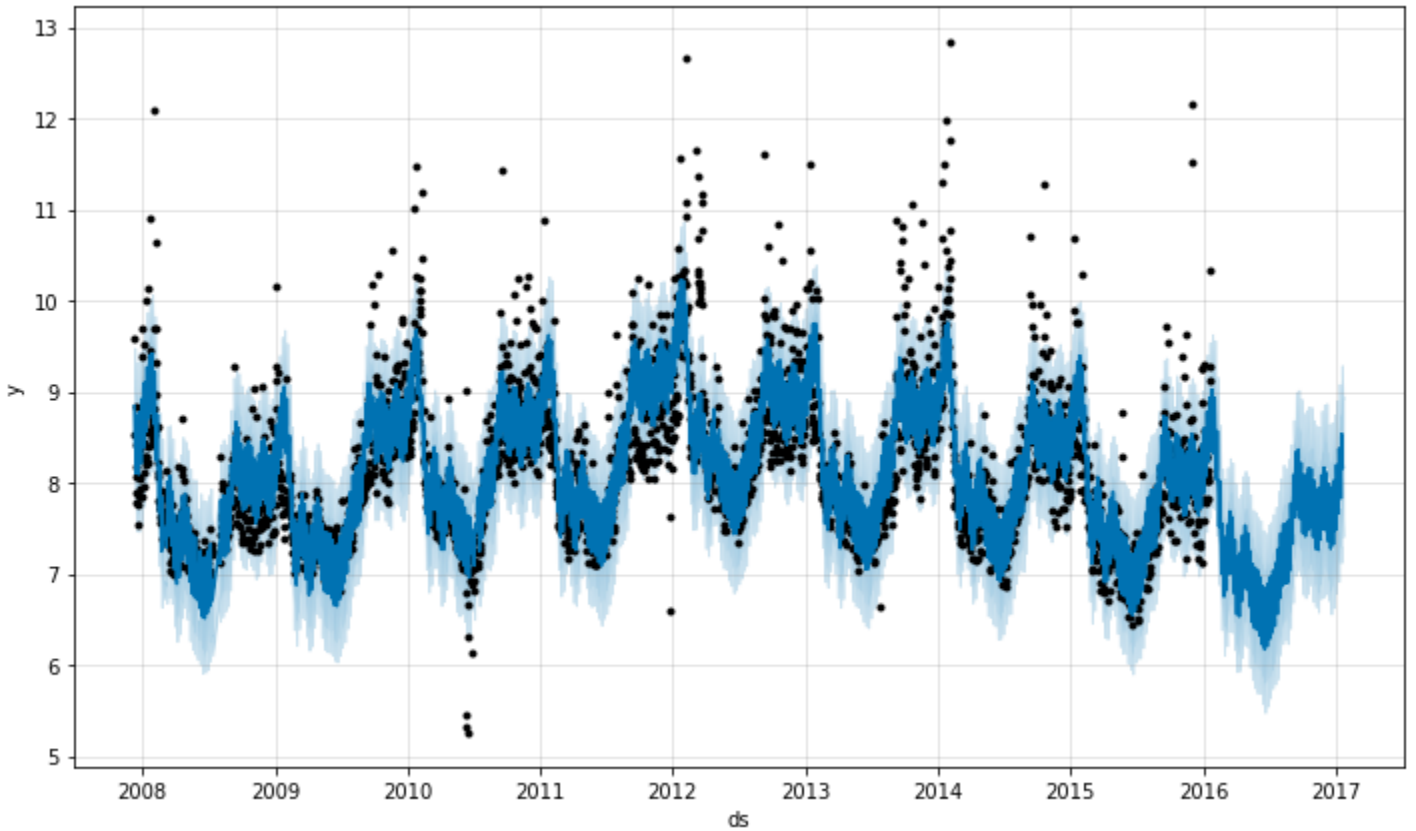
Feature	Prophet	NeuralProphet
Trend + Seasonality	✅	✅
Holiday Effects	✅	✅
Autoregression (past values)	❌	✅
Neural Network	❌	✅ (MLP)

Forecasting at Scale" (Taylor & Letham, 2017)

NeuralProphet: Explainable Forecasting at Scale" (Triebe et al., 2021)

Module 8- Part 1

PROPHET



Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

What is Facebook Prophet?

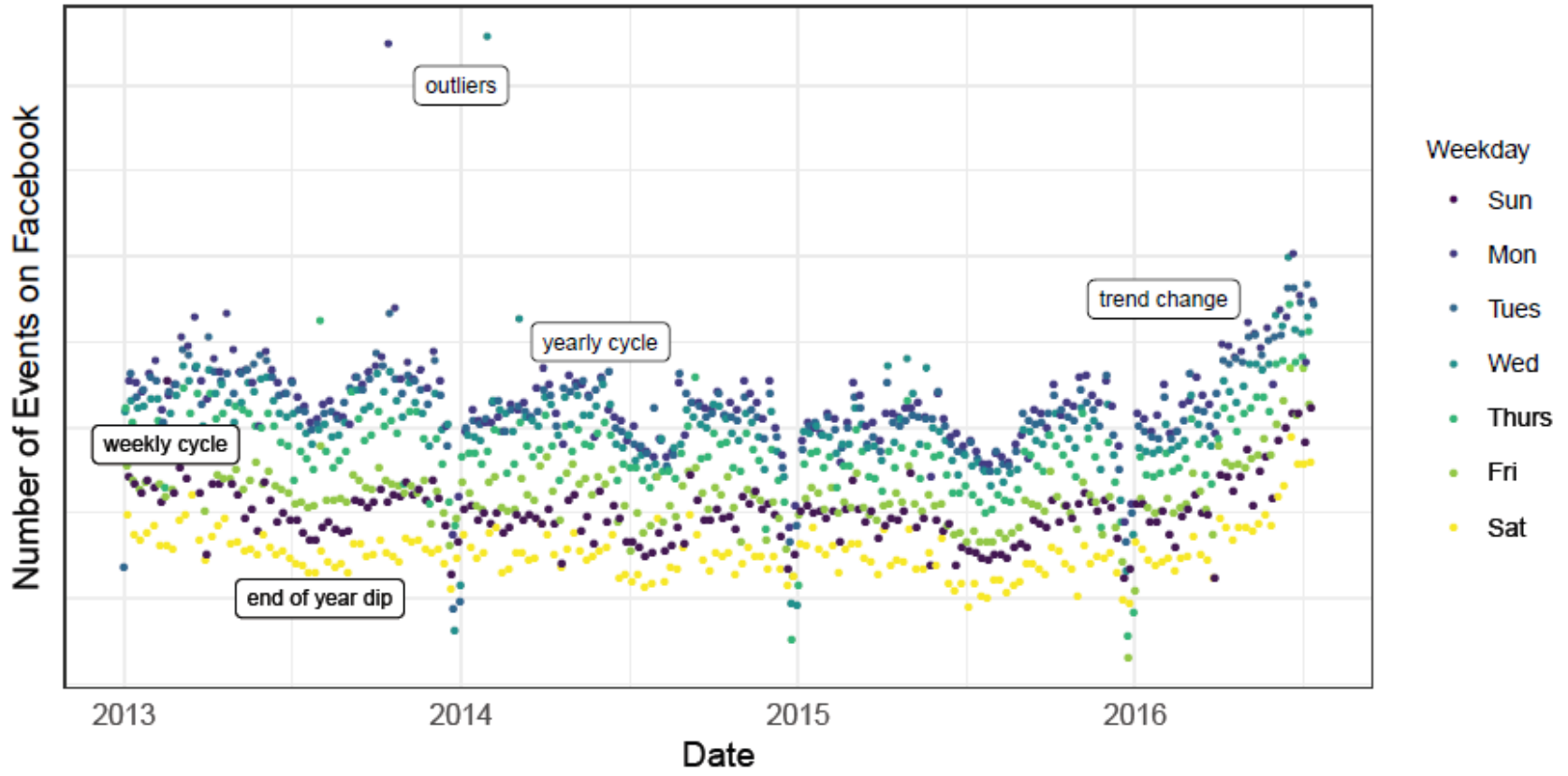
- Prophet is a robust open-source timeseries forecasting tool, available in Python and R.
- Developed by Sean J. Taylor and Benjamin Letham from Facebook (2017)
- Designed to produce forecasts **quickly** and reliably across a wide range of business applications.
- Forecasting at Scale:
    1. Large number of people making forecasts (possibly without training in ts methods)
    2. Large variety of forecasting problems
    3. Large number of forecasts (model evaluation and comparison while using human feedback to fix performance problems)
- Ideal for organizations that need to manage numerous and varied forecasting tasks simultaneously.

Key Features of Prophet

- Easy to Use with Analyst in the Loop: Prophet facilitates **active engagement** from analysts, requiring only basic familiarity with time series models while still benefiting from their domain expertise.
- Additive Model Components: Combines trends, seasonality, and holidays into an additive model that adjusts to changes in the time series.
- Automatic Trend and Seasonality Adjustments: Robust to dramatic shifts in trend and seasonality
- Robust to Data Anomalies: Prophet's non-recursive nature means it does not rely on lagged observations, which enhances its ability to handle datasets with missing entries. Additionally, its model formulation helps manage and mitigate the impact of outliers.

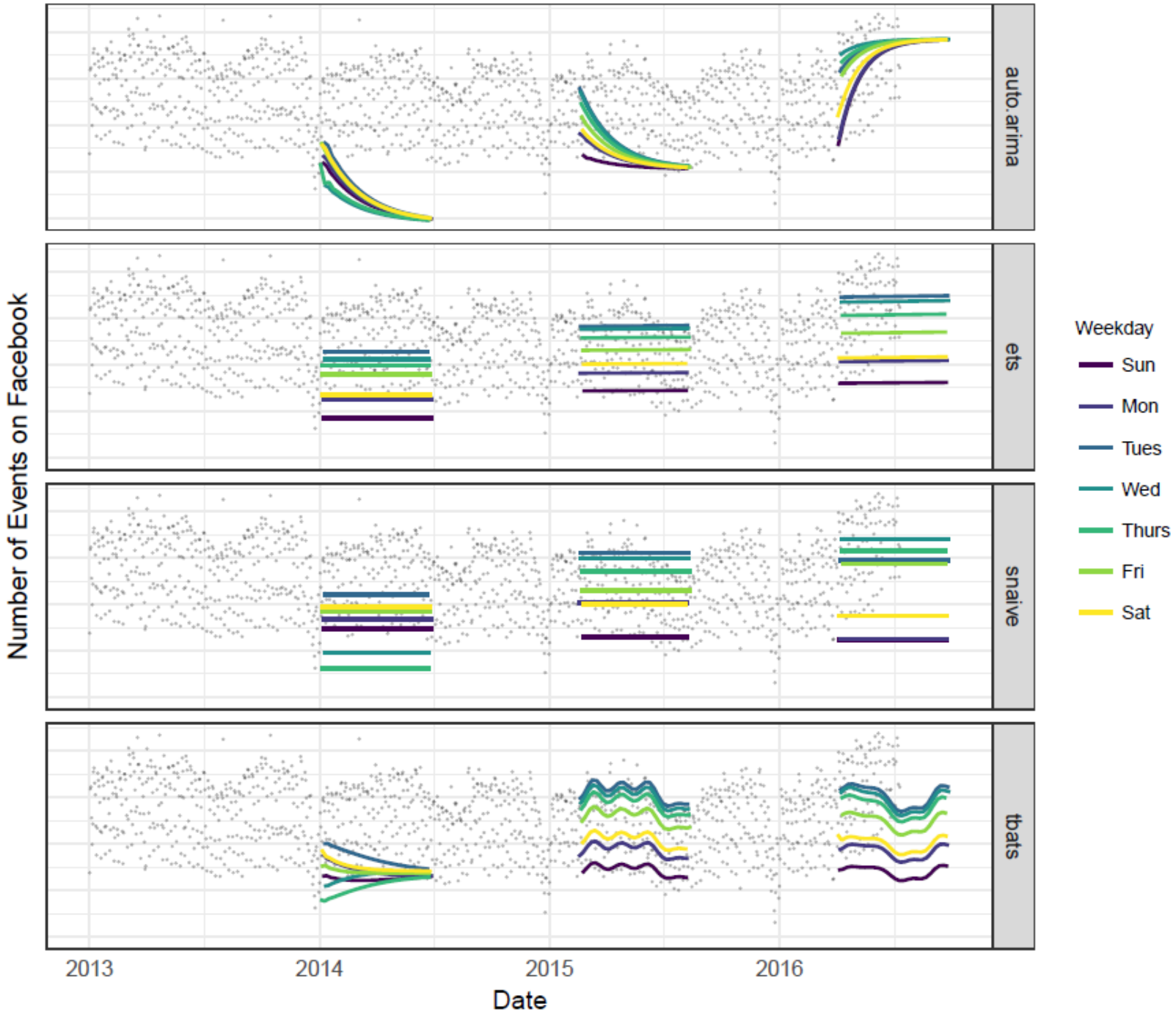
Business Time Series

- The features of this time series are representative of many business time series: multiple strong seasonalities, trend changes, outliers, and holiday.

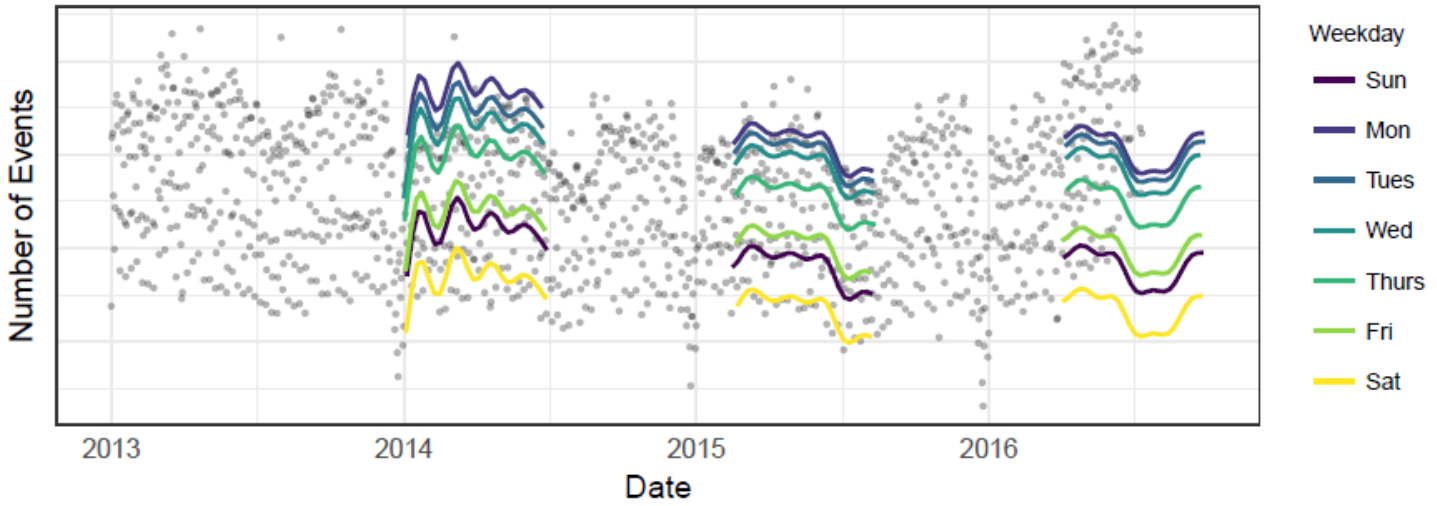


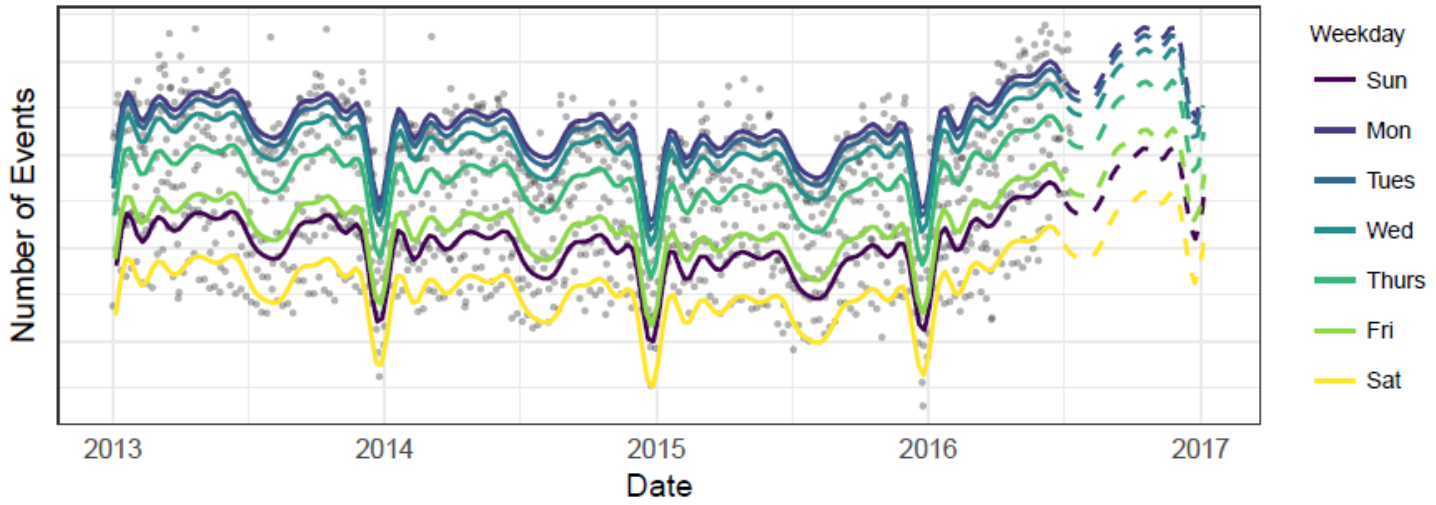
Benchmarks?!

- Forecasts made at 3 points in the history, each using only the portion of the time series up to that point.
- (It seems that) other models have failed to:
  - Capture trend change near cut off points
  - Capture mixed seasonalities
  - Handle end-of-year dip

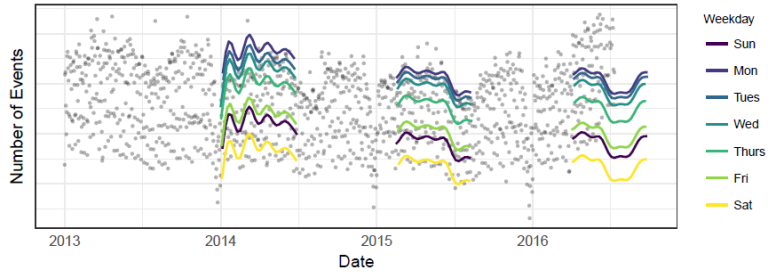


Business Time Series with Prophet





Business Time Series with Prophet



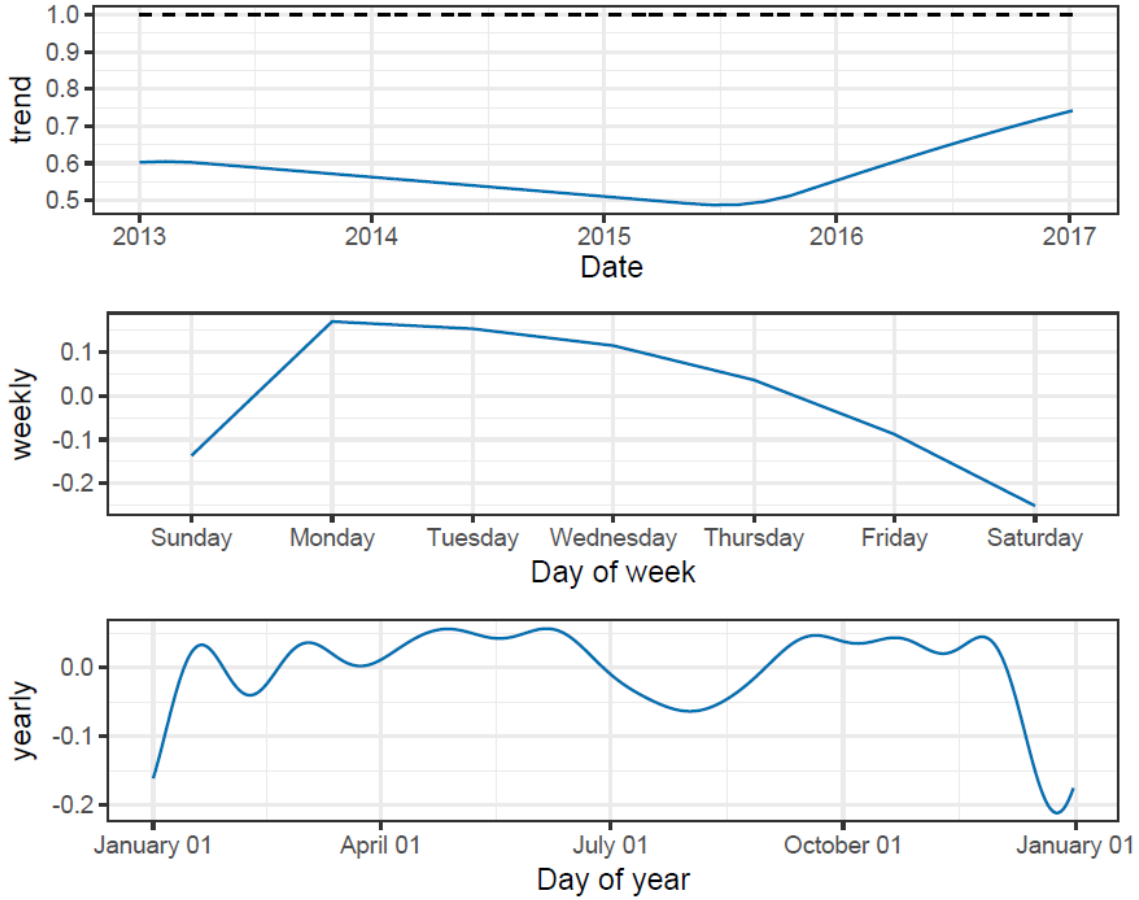


Figure 6: Components of the Prophet forecast in Fig. 5.

Core Components of Prophet

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

- Trend Component: Captures non-periodic changes and identifies points where the time series have abrupt changes in trajectory.
- Seasonality Component: Models periodic changes using Fourier series to provide a flexible representation of periodic effects.
- Holidays Component: Allows for the specification of irregular events that can affect the time series.
- Error term: Represents any idiosyncratic changes not captured by the model.
- Similar to Generalized Additive Models (GAM) with Non-Recursive nature.

Core Components of Prophet

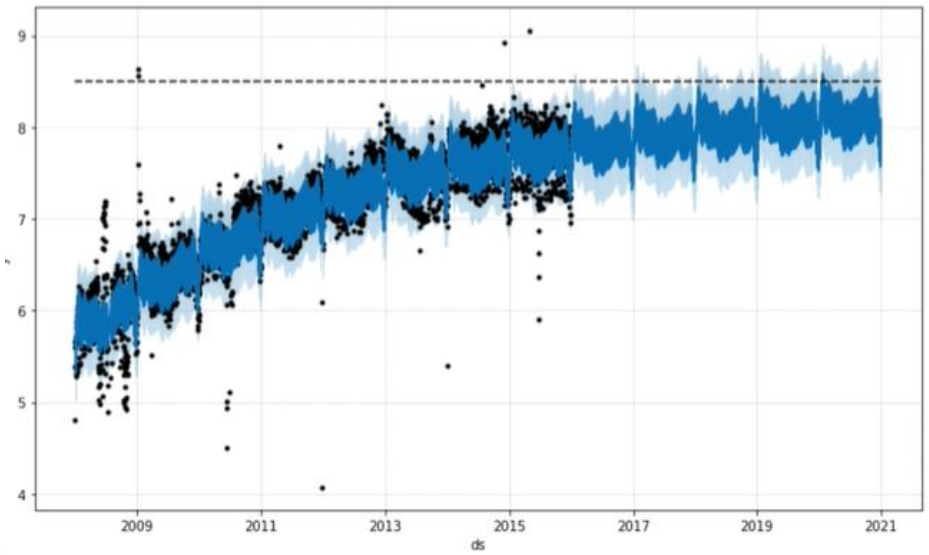
$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

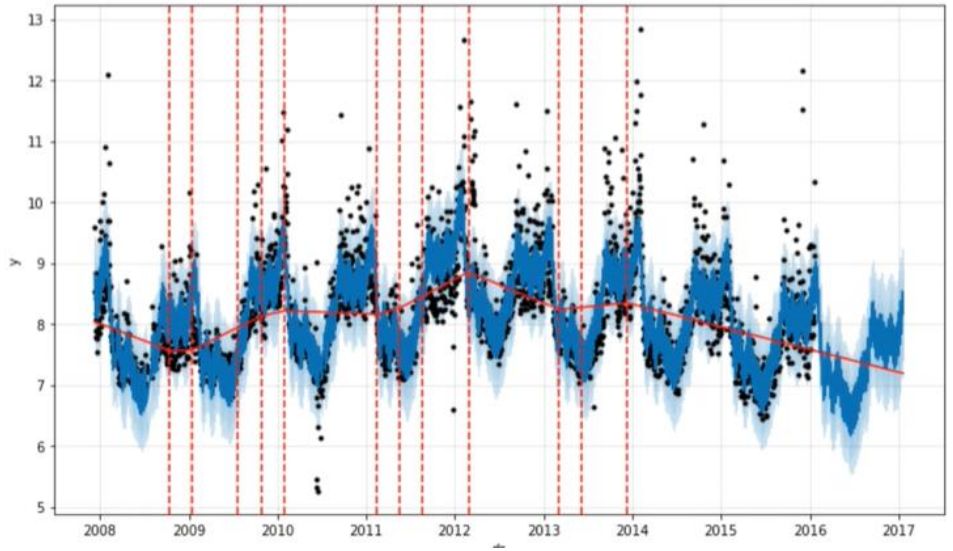
- In essence, framing the forecasting problem as a curve-fitting exercise.
- Inherently different from models that explicitly account for the temporal dependence structure in the data.
- Practical advantages are:
  - Flexibility
  - Doesn't require regular time series → no need to interpolate missing values or outliers
  - Fast fitting → allows for exploring many model specifications
  - Interpretable (also easily extendable to include more components)

The Trend Model

- Prophet uses a piecewise linear or logistic growth curve to model non-periodic changes, making it adaptable to different growth scenarios.
- It implements two trend models
  1. Nonlinear, Saturating Growth
  2. Linear Trend with Changepoints

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$





Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

1. Nonlinear, Saturating Growth

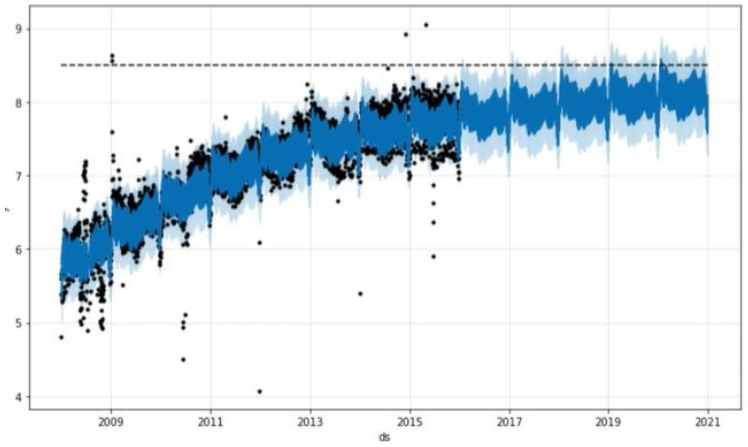
- Saturation: Accounts for carrying capacity in time series data.
- The most basic form is using the logistic growth model:

$$g(t) = \frac{C}{1 + \exp(-k(t - m))}$$

- C is the carrying capacity, k the growth rate, and m an offset parameter.
- Updates: Time varying Carrying capacity + Time varying Growth rate

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})(t - (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma})))}$$

- This feature is particularly useful when forecasting business metrics that have natural saturation points, such as market size or total population.



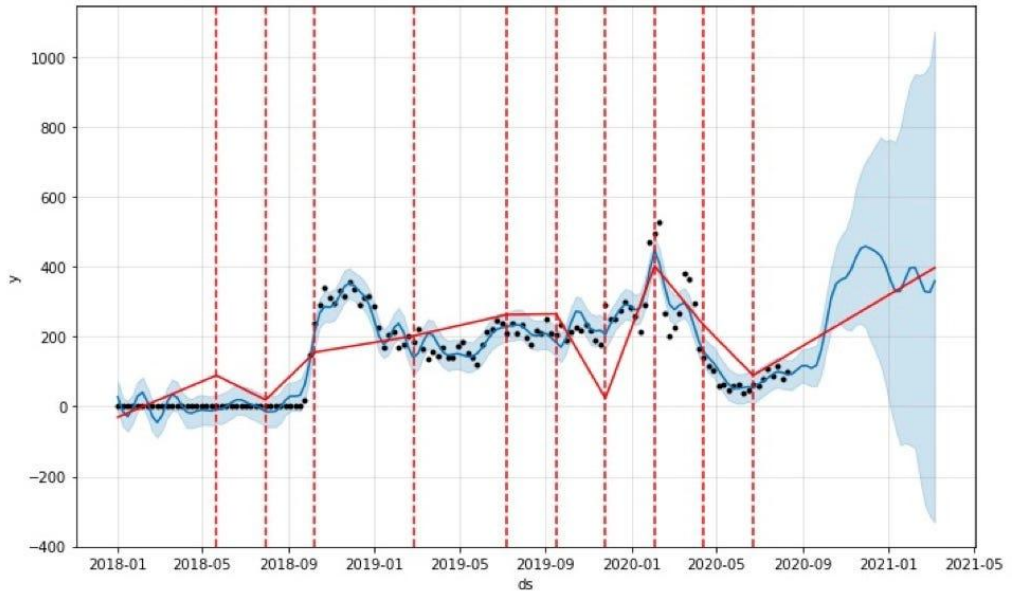
$$g(t) = \frac{C(t)}{1 + \exp(-(k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})(t - (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma})))}$$

2. Linear Trend with Changepoints

- For forecasting problems that do not exhibit saturating growth, a piece-wise constant rate of growth provides a parsimonious and often useful model

$$g(t) = (k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})t + (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma})$$

- Changepoints could be specified by the analyst using known dates of product launches and other growth-altering events.
- Automatic changepoint selection could be done by selecting a set of prior candidates.



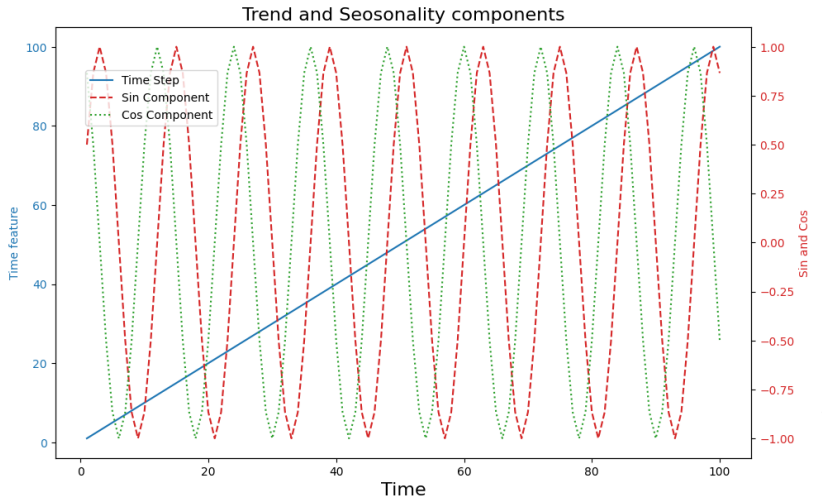
Seasonality Component

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

- Business time series often have multi-period seasonality because of the human behaviors they represent (5-day work week, vacations, school breaks, etc.)
- Prophet handles daily, weekly, and yearly seasonality, as well as custom seasonalities, allowing accurate forecasting for various temporal granularities.
- This is done by specifying seasonality models that are periodic functions of t → Fourier Series approximating arbitrary smooth seasonal effects:

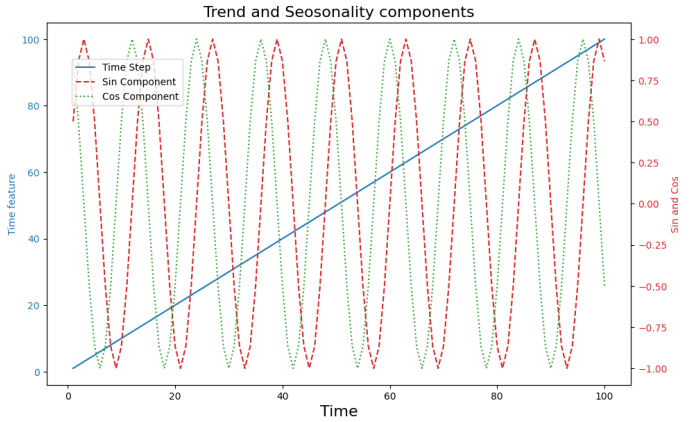
$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right)$$

- P is the regular period we expect the time series to have.



Seasonality Component

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi n t}{P}\right) + b_n \sin\left(\frac{2\pi n t}{P}\right) \right)$$



- P is the regular period we expect the time series to have.
- Fitting seasonality requires estimating 2N parameters $(a_1, \ldots a_n, b_1, \ldots, b_n)$
- This is done by constructing a matrix of seasonality vectors i.e. features!
- For example, with yearly seasonality and $N = 10$, the features are:

$$X(t) = \left[\cos\left(\frac{2\pi(1)t}{365.25}\right), \ldots, \sin\left(\frac{2\pi(10)t}{365.25}\right) \right]$$

- And the seasonal component is:

$$s(t) = X(t)\boldsymbol{\beta}$$

Holiday Component

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

- Holidays and events provide large, somewhat predictable shocks to many business time series and often do not follow a periodic pattern.
  - Thanksgiving in US on 4th Thursday in November
  - Super Bowl, Sunday in Jan or Feb.
  - Lunar calendar, etc.
- So, Holiday effects are not well modeled by a smooth cycle.

Holiday	Country	Year	Date
Thanksgiving	US	2015	26 Nov 2015
Thanksgiving	US	2016	24 Nov 2016
Thanksgiving	US	2017	23 Nov 2017
Thanksgiving	US	2018	22 Nov 2018
Christmas	*	2015	25 Dec 2015
Christmas	*	2016	25 Dec 2016
Christmas	*	2017	25 Dec 2017
Christmas	*	2018	25 Dec 2018

Holiday Component

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

- Prophet allows the analyst to provide a custom list of past and future events.
- Assumption: the effects of holidays are independent.
- One-hot encoding every holiday:
  - For each holiday i, D_i is the set of past and future dates for that holiday.
  - $Z(t)$ is a matrix of regressors, including indicator functions representing if time t is during holiday i, and assign each holiday a parameter κ_i which is the corresponding change in the forecast.

$$Z(t) = [\mathbf{1}(t \in D_1), \ldots, \mathbf{1}(t \in D_L)]$$

- And the holiday component is: $h(t) = Z(t)\boldsymbol{\kappa}$

Model Fitting

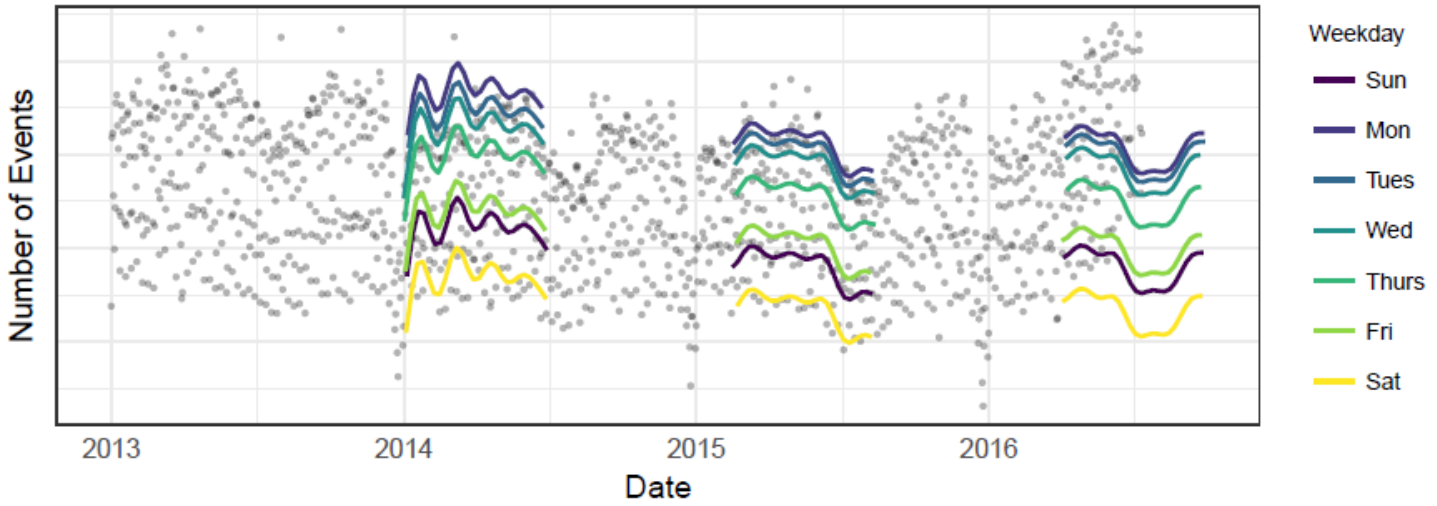
$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

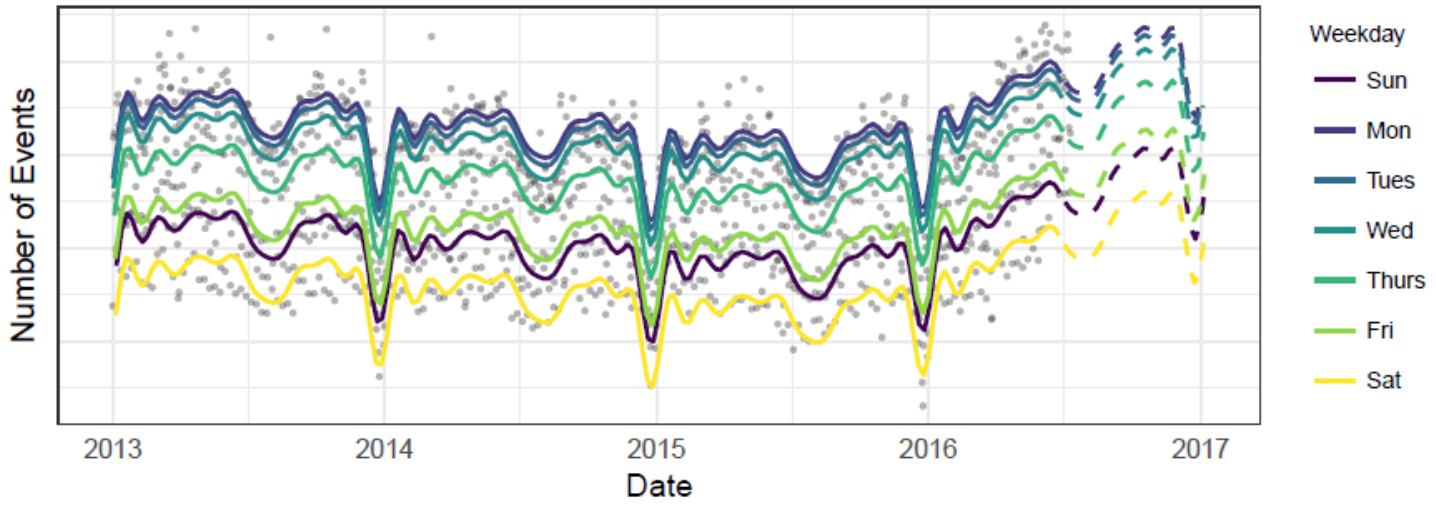
- **Matrix Formulation**: Seasonality, holidays, and change point effects are consolidated into matrix forms X and A, allowing the model to be efficiently expressed and computed.
- **Stan Code**: The entire forecasting model is implemented in Stan, a Bayesian library for statistical modeling and high-performance statistical computation.
- **Prior Selection**: Priors for the model parameters are chosen to reflect reasonable beliefs about the data without being overly restrictive. (including client on the modeling process)
- **Likelihood Functions**: Two forms of likelihood—logistic and linear—are used to capture different types of data behavior, with normal distributions assumed for the noise.
- **Full Posterior Inference**: Stan is also capable of conducting full posterior inference to estimate the uncertainty in the model parameters.

Listing 1: Example Stan code for our complete model.

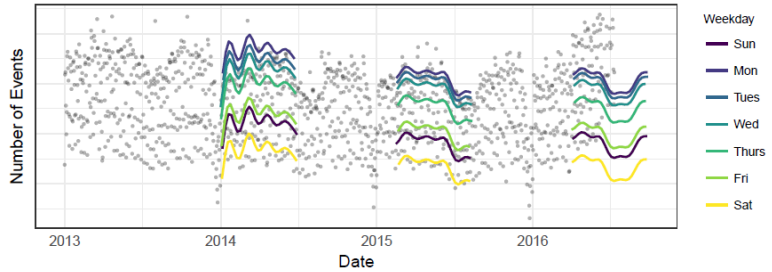
```
model {
  // Priors
  k ~ normal(0, 5);
  m ~ normal(0, 5);
  epsilon ~ normal(0, 0.5);
  delta ~ double_exponential(0, tau);
  beta ~ normal(0, sigma);

  // Logistic likelihood
  y ~ normal(C ./ (1 + exp(-(k + A * delta) .* (t - (m + A * gamma)))) +
             X * beta, epsilon);
  // Linear likelihood
  y ~ normal((k + A * delta) .* t + (m + A * gamma) + X * beta, sigma);
}
```

Business Time Series with Prophet





Business Time Series with Prophet



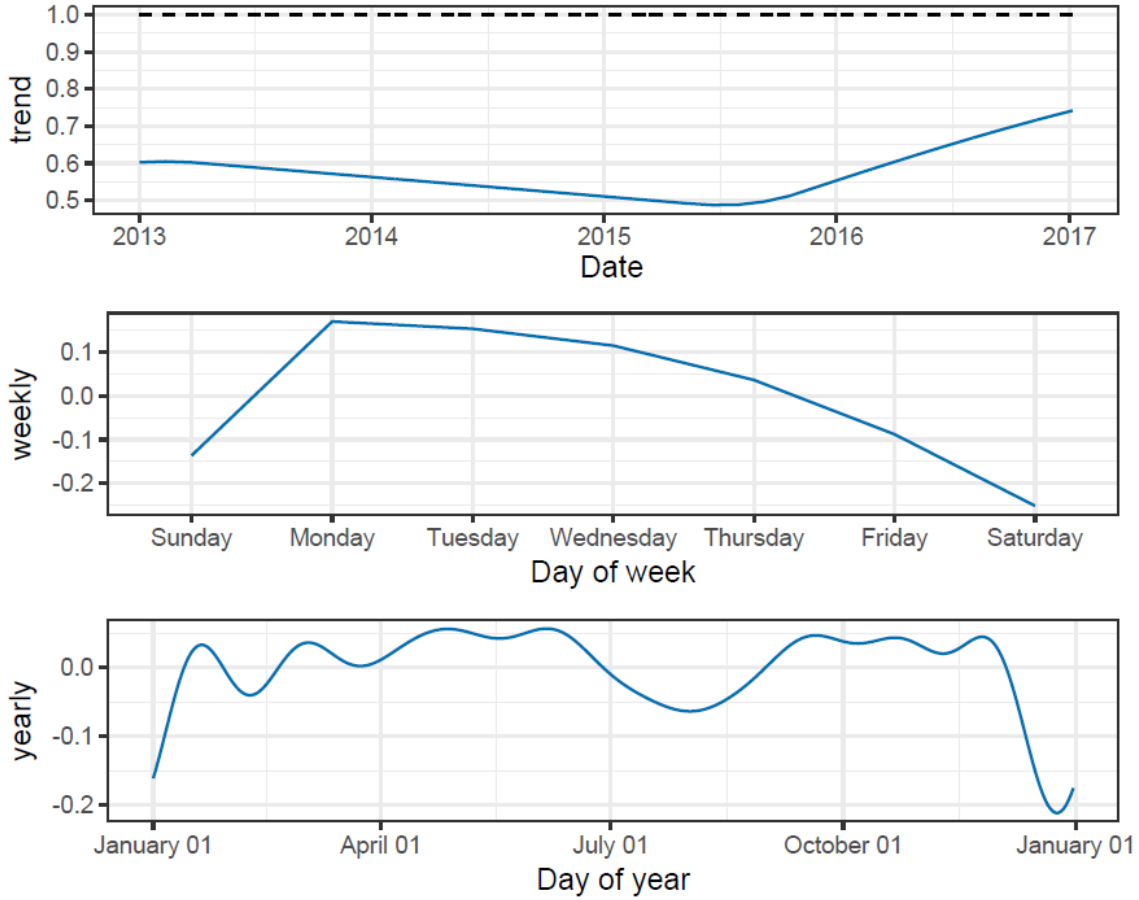


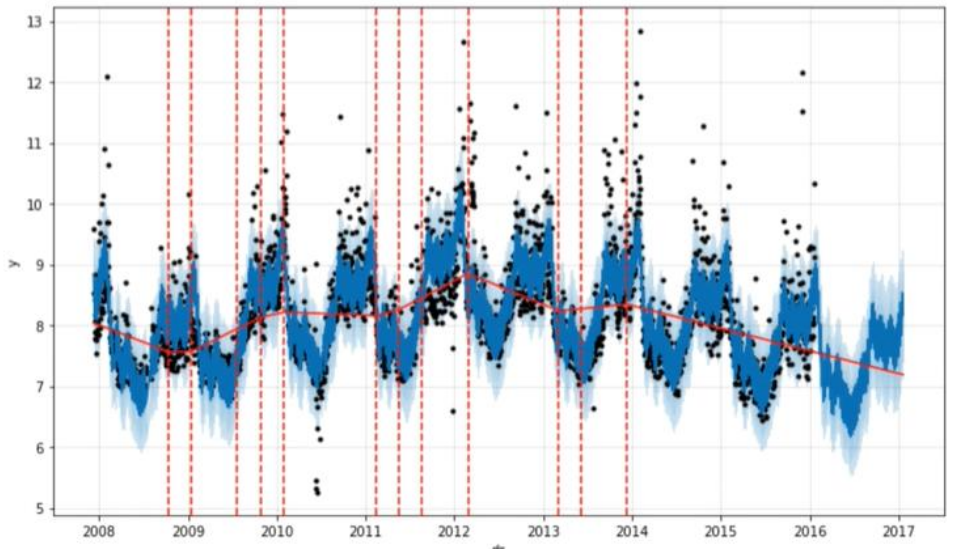
Figure 6: Components of the Prophet forecast in Fig. 5.

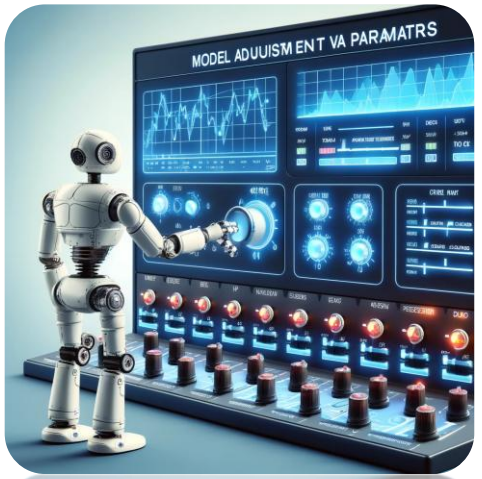
The Role of the Analyst in Prophet's Forecasting

- Domain Knowledge Utilization: Analysts can input their extensive domain knowledge into the Prophet model to refine forecasts without needing statistical expertise.
  - Capacities: Integration of market size data to set the carrying capacity for forecasts.
  - Changepoints: Direct specification of known events that impact the time series.
  - Holidays and Seasonality: Input of regional growth impacts from holidays based on experience.
- Model Adjustment via Parameters

The Role of the Analyst in Prophet's Forecasting

- Domain Knowledge Utilization
- Model Adjustment via Parameters: Parameters in Prophet allow analysts to apply their insights to adjust the model's sensitivity to trends and seasonality.
  - Smoothing Parameters: Adjusting priors to change trend flexibility and influence the strength of seasonality.
  - Visualization Tools: Essential for validating model fit and identifying areas for refinement.



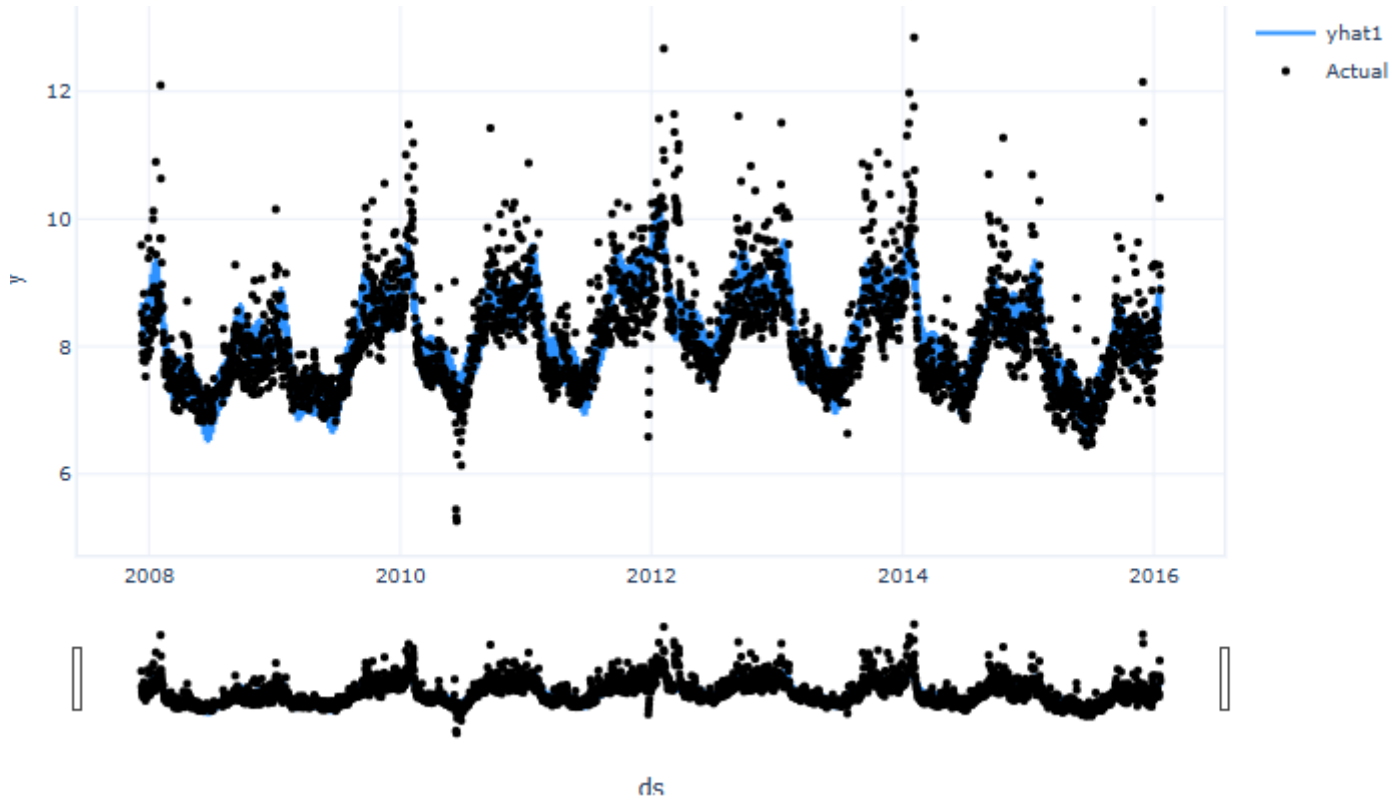


Conclusion

- Prophet's Innovation: Analyst-in-the-Loop Modeling Approach, combines the precision of statistical forecasts with the nuanced insights of judgmental forecasts.
- Versatile Model Adjustments: Analysts can apply their domain knowledge through intuitive parameters affecting capacities, changepoints, and seasonalities.
- Automated vs. Judgmental Forecasting: Prophet marries the scalability of automated forecasting with the depth of judgmental forecasting.
- Automated Forecast Evaluation: Critical for scaling analyst-in-the-loop, automated tools assess forecast quality, guiding where human input is most beneficial.

Module 8- Part 2

Neural Prophet



Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

NeuralProphet - Model Intuition

- What is NeuralProphet ?
  - NeuralProphet = Prophet + Autoregression + Neural Networks
  - Designed for time series forecasting, just like Facebook Prophet
  - Adds power and flexibility:
    - Captures short-term dependencies (AR)
    - Handles complex patterns using a neural net
  - Built on PyTorch; open-source and customizable

NeuralProphet - Model Structure

- Model Components
  - **Trend (g(t))**
    - Piecewise linear or logistic growth
    - Can incorporate changepoints
  - **Seasonality (s(t))**
    - Modeled with Fourier series (like Prophet)
    - Supports yearly, weekly, daily patterns
  - **Holiday Effects (h(t))**
    - Optional holiday regressors (country calendars or custom)
  - **Autoregression (AR(t))**
    - Captures short-term memory using lagged values $y_{t-1}, \ldots, y_{t-n}$
  - **Neural Net (AR)**
    - Nonlinear mapping from lags to forecast

Core Formula: $y(t) = g(t) + s(t) + h(t) + AR(t) + \varepsilon_t$ Where:

- $g(t)$: Trend
- $s(t)$: Seasonality
- $h(t)$: Holiday effects
- $AR(t)$: Autoregressive component (learned via NN)
- ε_t: Noise

Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

What makes it Neural? Autoregressive component (AR-Net)

How It Works:

- **Input:** Sequence of lagged target values $[y_{t-1}, y_{t-2}, \ldots, y_{t-n}]$.
- **Architecture:**
  - Default: Single linear layer mapping input to output.
  - Configurable: Add hidden layers with sizes defined in `ar_layers`.
- **Activation:** ReLU applied if hidden layers are added.
- **Output:** Autoregressive contribution to the forecast.

Regularization:

- **AR Regularization (`ar_reg`):**
  - Applies L1/L2 penalties to AR weights to induce sparsity and prevent overfitting.
  - Not enabled by default; users can set `ar_reg` to a value greater than zero to activate.

Integration:

- The AR component's output is combined additively with trend, seasonality, and holiday effects to form the final forecast.

Minimal example

```
pip install neuralprophet
```

```
from neuralprophet import NeuralProphet
```

After importing the package, you can use NeuralProphet in your code:

```
m = NeuralProphet()
metrics = m.fit(df)
forecast = m.predict(df)
```

You can visualize your results with the inbuilt plotting functions:

```
fig_forecast = m.plot(forecast)
fig_components = m.plot_components(forecast)
fig_model = m.plot_parameters()
```

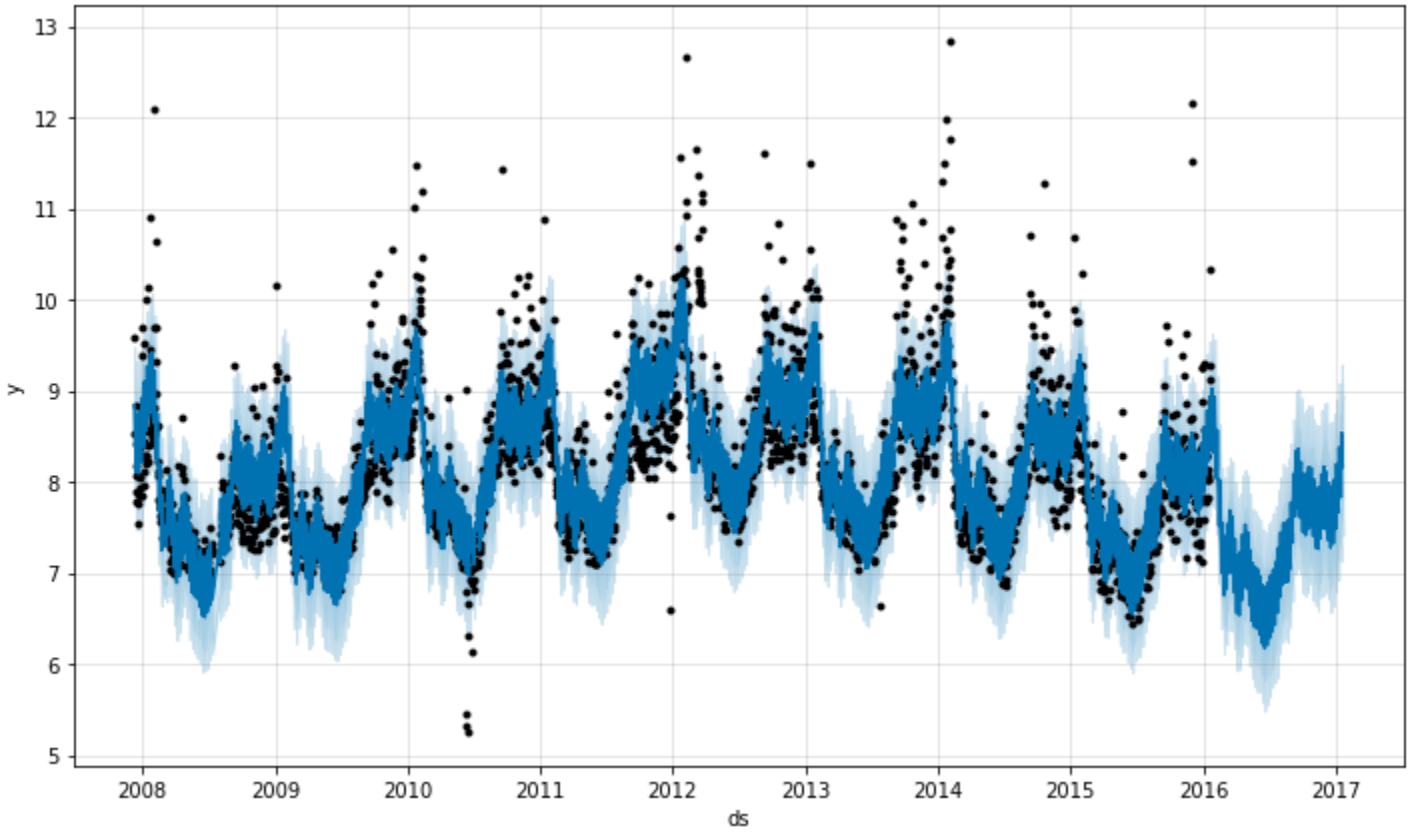
https://neuralprophet.com/how-to-guides/feature-guides/Migration_from_Prophet.html

Takeaway

- Use Prophet for simple, interpretable trend/seasonality
- Use NeuralProphet when AR terms or complex patterns matter
- Still interpretable, but more flexible

Module 8- Python Part 1.1

PROPHET



Jon M. Huntsman School of Business
UtahStateUniversity

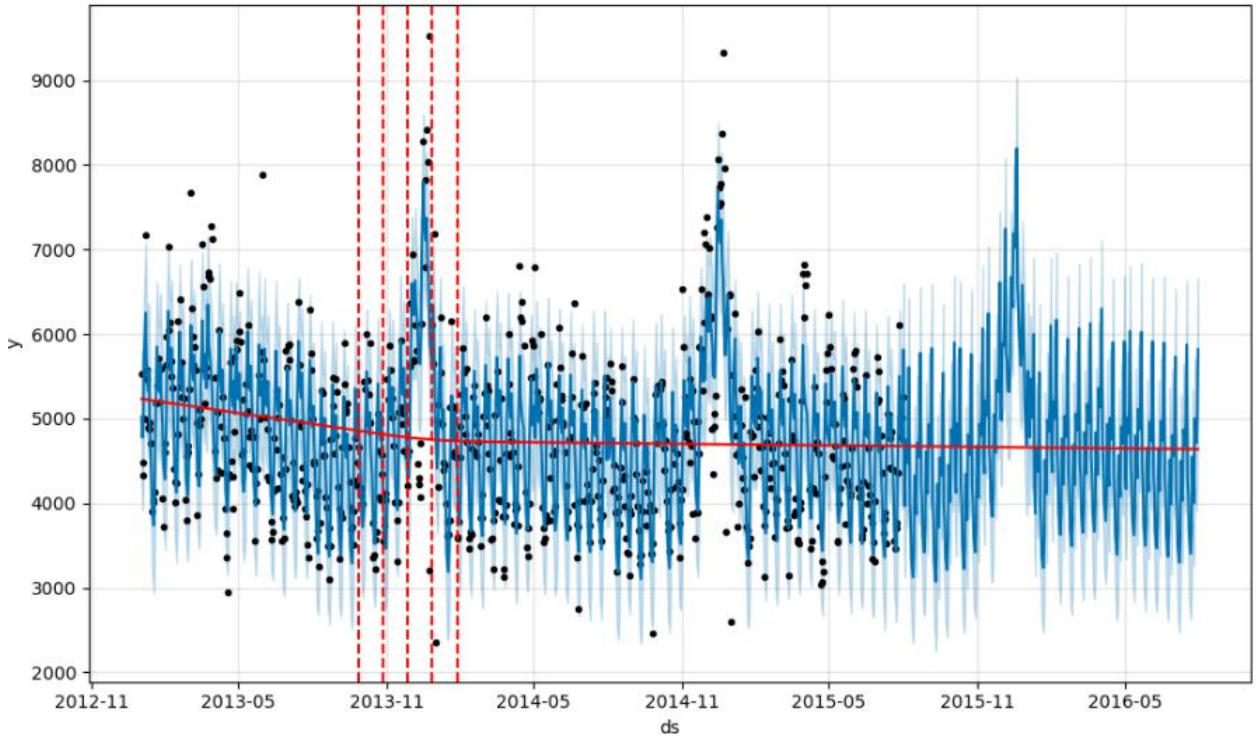
Pedram Jahangiry

Module 8- Python Part 1.2

PROPHET

ROSSMANN

kaggle

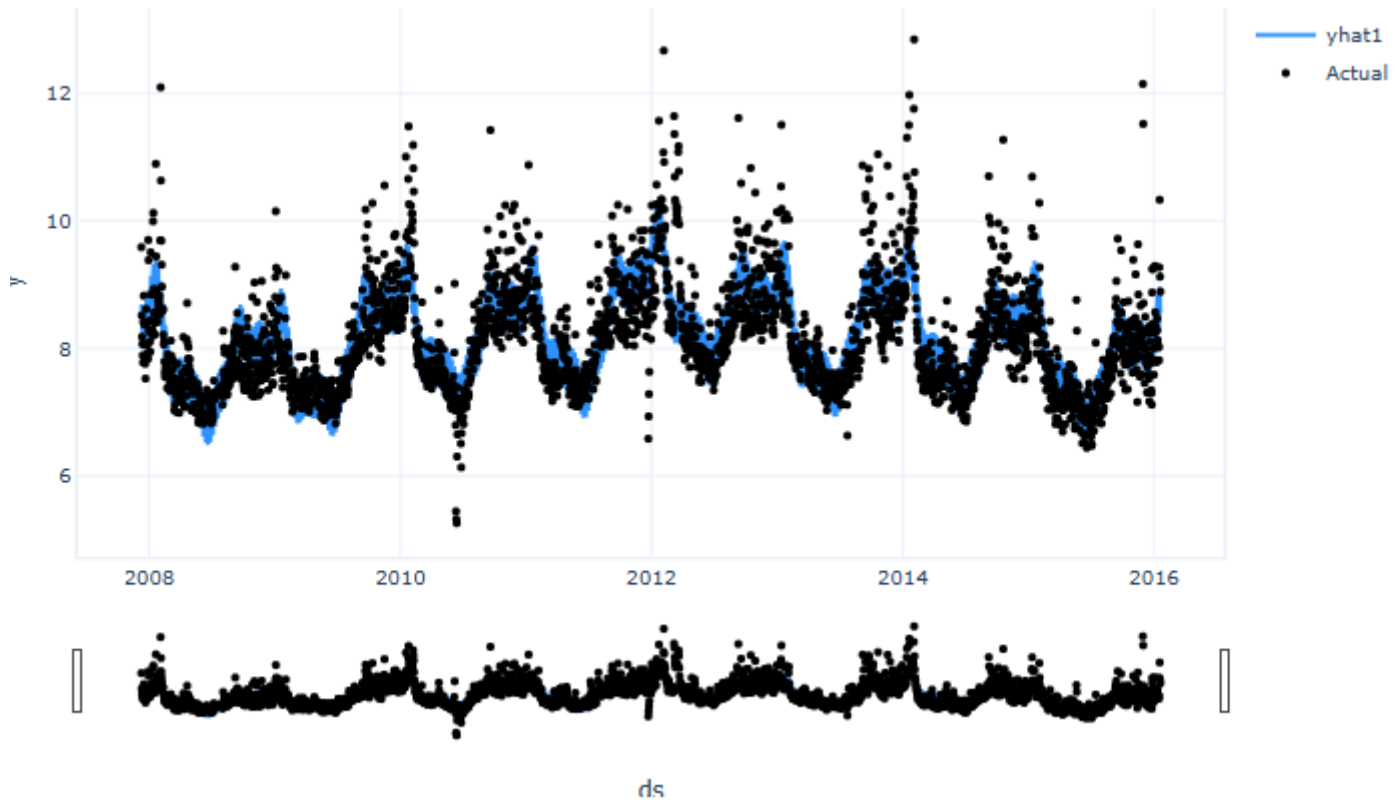


Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

Module 8- Python Part 2

Neural Prophet



Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Road map!

- ✓ Module 1- Introduction to Deep Forecasting
- ✓ Module 2- Setting up Deep Forecasting Environment
- ✓ Module 3- Exponential Smoothing
- ✓ Module 4- ARIMA models
- ✓ Module 5- Machine Learning for Time series Forecasting
- ✓ Module 6- Deep Neural Networks
- ✓ Module 7- Deep Sequence Modeling (RNN, LSTM)
- ✓ Module 8- Prophet and Neural Prophet