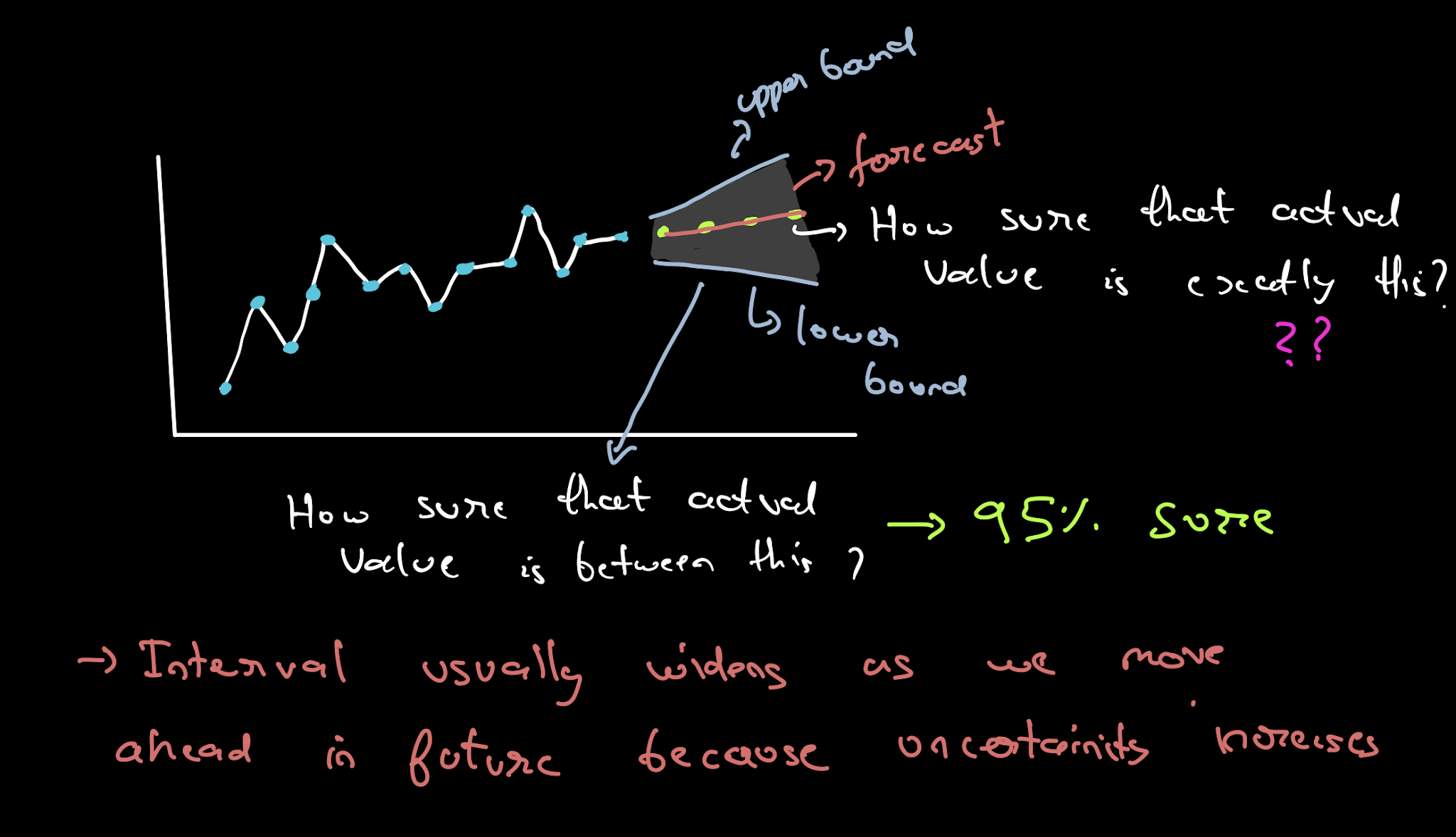
**Time series Analysis**

# **Confidence Intervals**

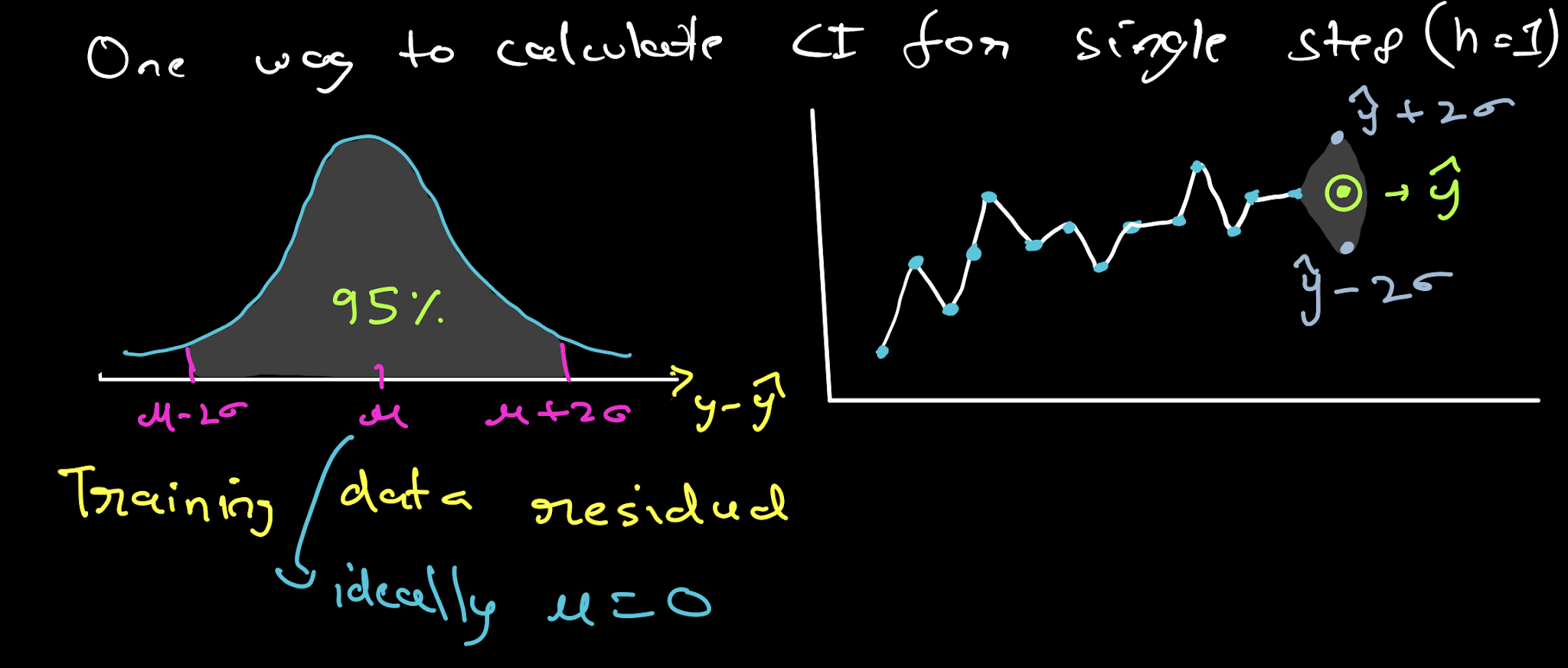
* In addition to making predictions, the models can also provide Confidence Intervals.
* In addition to making predictions, time series models can also provide Confidence Intervals.
* Owing to the **potential error** of our forecast, and the **unpredictability of the future**, we cannot claim to be a hundred percent sure of our forecasts.
* But we can still, based on our study of time series, say that the future value would lie in the **range** of [A, B].
* Naturally A becomes the minimum possible value of our target variable at time t, and B becomes the maximum possible value at time t.
* This [A, B] becomes a range for determining the **real possible value** for a certain prediction.
* It is called the **confidence interval.**
* It can also be used for performance evaluation.



* Confidence Intervals are required for determining the uncertainty of your predictions, after making forecasts.
* Confidence Intervals provide an expected range for the real observation. When making decisions, one can utilize the range of information better, than just a single point.
* Different algorithms have different formulations for CI.
* These can be complicated to derive and are out of scope for this lecture.
* We can get the upper and lower values according to the models available in the statsmodel library.
* It takes a value of a parameter: **alpha**, which is a measure of how much confidence we want in our forecasts.
* One example of a 1-step ahead forecast CI is to just take the residual distribution.
* For example, the 95% confidence interval will be



* Note that for a good forecast 𝑈𝑟𝑒𝑠𝑖𝑑 should be close to 0



* To interpret confidence intervals, let our forecasted value is 𝑦̂. And you pass a value of **alpha = 0.05** for which you get the confidence interval as (m,n).
* It means that the time series model will estimate the upper(n) and lower(m) bound of values around the forecast, where there is only a 5% chance that the real value will **not be** in that range.
* That is, 95% that our forecast will fall within the range (m,n).

# **Facebook’s Prophet**

* One of the drawbacks of the SARIMAX Model is that one cannot have multiple seasonality and can only select one value.
* To overcome this disability, we're studying an open-source tool/library called prophet which is developed by Facebook

**Following are the features of Facebook's Prophet:**

* Provides intuitive parameters which can be **easily tuned**
* It is **robust to missing data and shifts in the trend** and typically **handles outliers** well.
* It can account for **multiple seasonalities**. This is possible because, under the hood, the math of seasonalities is based on **Fourier transforms**, which help incorporate this.
* The Prophet uses a decomposable time series model with three main model components
* They are combined in the following equation:

**y(t)= g(t) + s(t) + h(t) + εt**

* **g(t):** piecewise linear or logistic growth curve for modeling non-periodic changes in time series (**trend**)
* **s(t):** periodic changes (e.g. weekly/yearly **seasonality**)
* **h(t):** effects of **holidays** (user provided) with irregular schedules
* **εt:** **error term** accounts for any unusual changes not accommodated by the model.
* For using Prophet, the data set should contain only two columns with column names as ‘ds’ and ‘y’
* ‘ds’ should always be in ‘date-time’ format. And ‘y’ represents a feature that you want to forecast.
* To install prophet to your environment, you can use the following environments:

!pip install pystan~=2.14

!pip install fbprophet

→ from fbprophet import Prophet

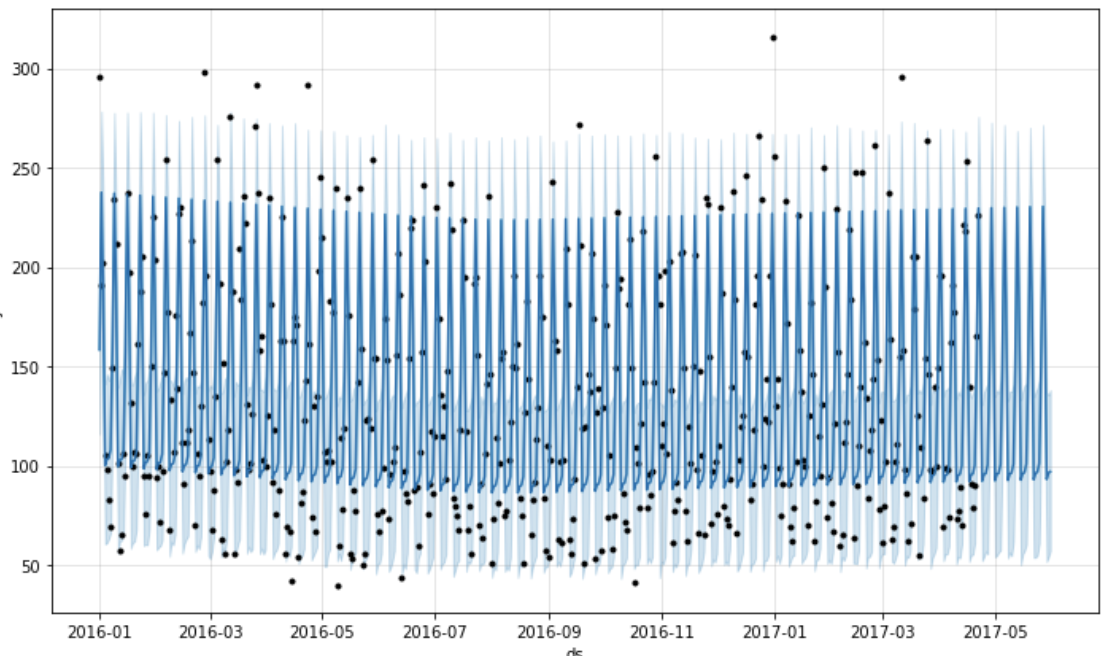
m = Prophet()

m.fit(df[['ds', 'y']][:-39]) #here we are leaving last 39 observations because we will predict it in 'future'

future = m.make\_future\_dataframe(periods=39,freq="D")

forecast = m.predict(future)

fig = m.plot(forecast)



* In the plot, black dots are actual visits, deep blue lines are the predicted visits and light blue lines are the 95% confidence interval around the prediction.
* You can see that the lines are flat and the model is not able to capture the seasonality properly so it is not a good fit.
* Here light blue lines are 95% confidence intervals around the predictions.
* Here we also didn't do anything explicitly for Nan values it was handled by the prophet.
* We can use ‘**add\_regressor’** for adding features to the prophet model.
  + There is another interesting parameter here: ‘**changepoint\_prior\_scale’**
* If the trend changes are overfitting (too much flexibility) or underfitting (very less flexibility)
* We can adjust the strength of sparse prior using this parameter.
* Default value: 0.05
* Increasing this will make the trend more flexible.

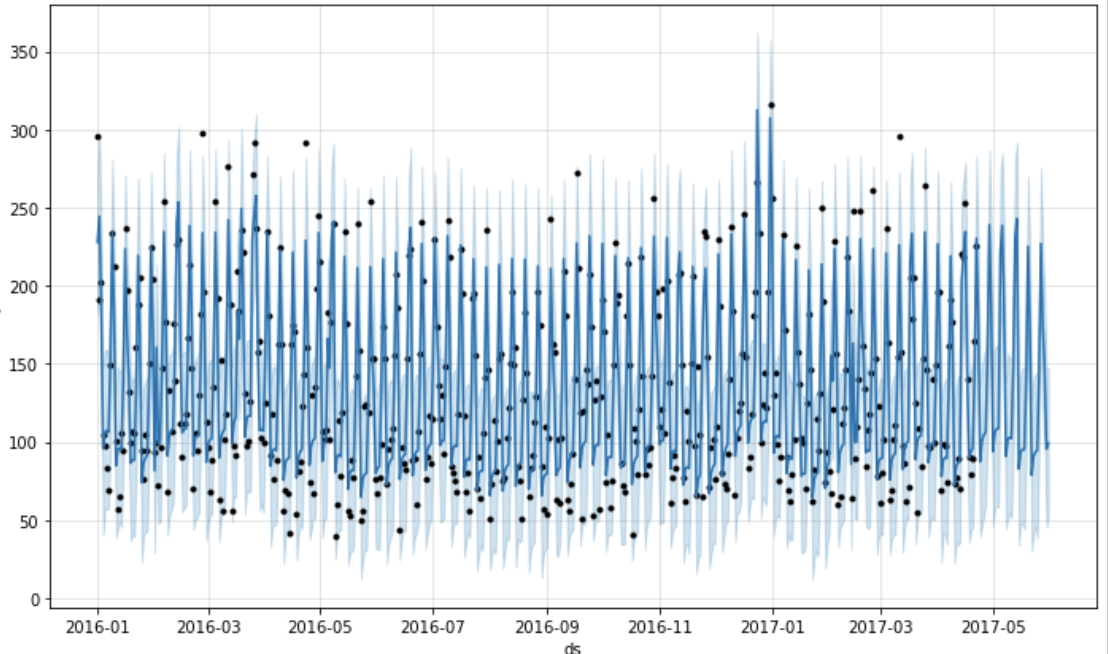
→ model2=Prophet(interval\_width=0.95, yearly\_seasonality=True, weekly\_seasonality=True,changepoint\_prior\_scale=4)

model2.add\_regressor('holiday') #adding holidays data in the model3

model2.fit(df[:-39])

forecast2 = model2.predict(df)

fig = model2.plot(forecast2)



### 

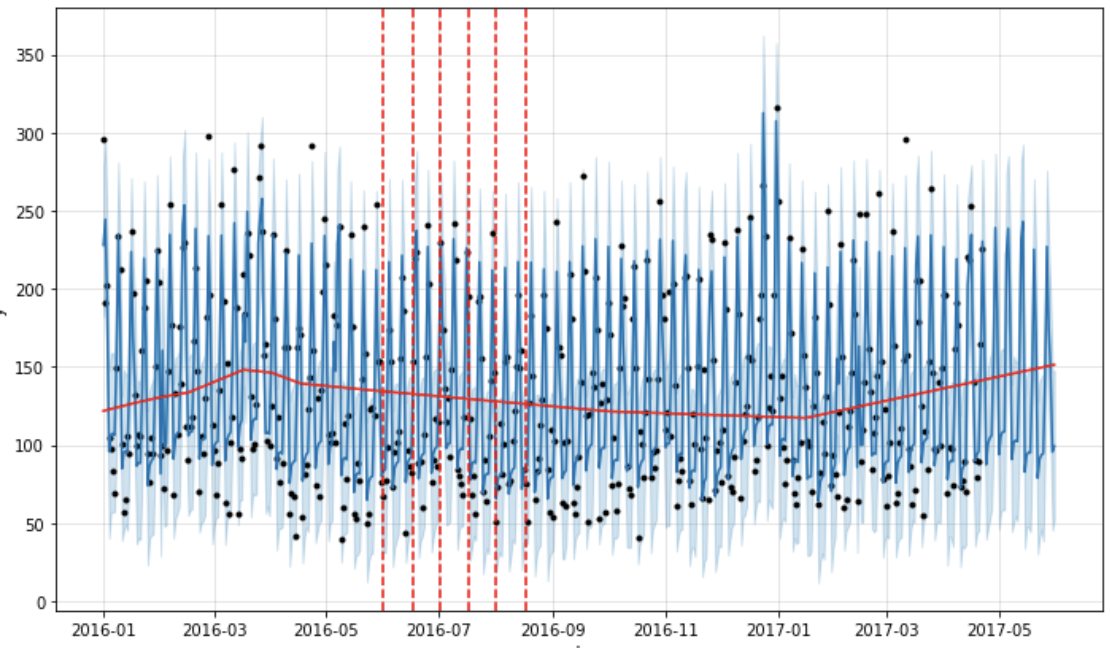
### **Changing Trends**

* Prophet automatically detects the changepoints and will allow the trend to adapt appropriately.
* Prophet detects changepoints by first specifying a large number of potential changepoints at which the rate is allowed to change.

→ from fbprophet.plot import add\_changepoints\_to\_plot

fig = m.plot(forecast2)

a = add\_changepoints\_to\_plot(fig.gca(), m, forecast2)



* The vertical lines in this figure indicate where the potential change-points were placed.
* Facebook prophet provides automated methods to forecast.
* The model has easily interpretable parameters that can be changed by the analyst to impose assumptions on the forecast.

### **Benefits of Prophet**

* Prophet is a simple library and is great for beginners.
* It works best with time series that have strong seasonal effects.
* Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.
* We can add multiple regressors or exogenous variables.
* It forms a very good baseline model because almost no feature engineering is required.
* Interpretability is one of the key advantages of the Prophet.
* If your time series follows some business cycles, you can obtain very decent performance quickly.
* It can also be helpful while detecting change points.

# Change Points

* A sudden change in a time series that changes the hidden trends and statistical characteristics of the data.
* A change point divides a time series into two segments with distinct statistical characteristics.

A graph of a graph of a graph

Description automatically generated with medium confidence

* To detect the change point:
  + Walk through the series with a window of fixed size
  + For each step, compute the **cost** of all elements in the window. There are many options for this cost function.
  + Wherever the cost locally peaks, the center of the window can be considered as a changepoint.
  + We can add other conditions of threshold cost, etc to avoid making too many detections.
  + Cost is high than a certain threshold, change point is present, and vice versa.

A graph of a wave

Description automatically generated

* There are various factors on which one can identify a change point. Some of them are:
  + Mean
  + Variance
  + Periodicity
  + Pattern
* **Change in Mean:**
  + This is the most common and probably the easiest one to interpret and identify a change point.
  + Change in mean often occurs when a time series is the build-up of constant segments having different mean values.
  + Change points are defined as the first time step in each new segment starting with the second segment, so the number of change points is always one fewer than the number of segments.

A diagram of a signal

Description automatically generated

* **Change in Variance**
  + Change in variance is another simple way of determining change points.
  + Here, the mean of the signal stays constant, but there are several segments with different variance values.
  + This can be interpreted as a sudden noise in the signal.

A blue and white graph

Description automatically generated

* **Change in Periodicity (Frequency)**
  + Change in periodicity (also called a change in frequency) is concerned with the changes occurring when the frequency changes suddenly.
  + Detection of this kind of change is usually done in the frequency domain, for example by using Fourier transform or wavelet transform.

A graph showing a signal

Description automatically generated