ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

ĐỒ ÁN 2

ĐỀ TÀI: Tìm hiểu về công cụ dự báo Prophet

Giảng viên hướng dẫn: **ThS. Nguyễn Công Hoan**

Lớp: SE122.L11.PMCL

Sinh viên thực hiện:

Phan Huỳnh Minh Duy 17520405

LỜI CẢM ƠN

Trong suốt thời gian từ khi bắt đầu thực hiện đồ án, em đã nhận được sự quan tâm, giúp đỡ của quý thầy cô, gia đình và bạn bè.

Em xin chân thành cảm ơn thầy Nguyễn Công Hoan đã trực tiếp hướng dẫn, giúp đỡ về kiến thức, tài liệu và cách thực hiện để hoàn thành đề tài này.

Mặc dù có nhiều cố gắng trong suốt quá trình thực hiện, song có thể còn có những mặt hạn chế, sai sót. Em mong nhận được ý kiến đóng góp và chỉ dẫn của thầy và các bạn để đề tài được hoàn thiện.

Xin trân trọng cảm ơn!

Sinh viên thực hiện

Phan Huỳnh Minh Duy

Tp. Hồ Chí Minh, ngày 08 tháng 01 năm 2021

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

MỤC LỤC

Chương 1: Giới thiệu chung 3
1.1. Lý do chọn đề tài 3
1.2. Mục tiêu 4
1.3. Đối tượng nghiên cứu 4
Chương 2: Kiến thức nền tảng 4
1.1. Tổng quan về Time-series 4
1.2. Thành phần của Time-series: 8
1.3. Đánh giá lỗi: 13
1.4. Ứng dụng của Time-series Data và Time-series Forecasting: 14
Chương 3: Công cụ dự báo Prophet 15
1.1. Tổng quan về Facebook Prophet: 15
1.2. Yêu cầu và cài đặt: 17
1.3. Tìm hiểu mã nguồn của Prophet: 19
1.4. Thực hiện dự đoán bằng Prophet: 25
Chương 4: Kết luận 38
Chương 5: Một số thuật ngữ trong báo cáo 39
Chương 6: Tài liệu tham khảo 39

Chương 1: Giới thiệu chung

1.1.Lý do chọn đề tài

Hiện nay, công nghệ đang phát triển với tốc độ vượt bậc và trở thành một phần thiết yếu trong cuộc sống. Nhờ vào công nghệ, chúng ta có thể truy cập nhanh chóng đến nhiều lĩnh vực đời sống, giúp công việc trở nên dễ dàng và tiện lợi hơn.

Dựa trên yêu cầu của con người đến việc dự đoán các lĩnh vực trong đời sống, các thuật toán và công cụ để dự đoán trên thiết bị công nghệ được sử dụng thường xuyên và phát triển hơn. Chẳng hạn như dự đoán biến động chứng khoán, dự đoán thời tiết, dự đoán lượng truy cập của một trang web…

Đó cũng là lý do ra đời của công cụ dự báo Prophet, cũng chính là công cụ được tìm hiểu trong đề tài này.

1.2.Mục tiêu

- ❖ Hiểu kiến thức về Time-series Data.
- ❖ Áp dụng Time-series Data để thực hiện Time-series Forecasting.
- ❖ Tìm hiểu và áp dụng công cụ Facebook Prophet.

1.3.Đối tượng nghiên cứu

- ❖ Time-series Data & Time-series Forecasting
- ❖ Công cụ tìm hiểu: Facebook Prophet
- ❖ Ngôn ngữ: Python

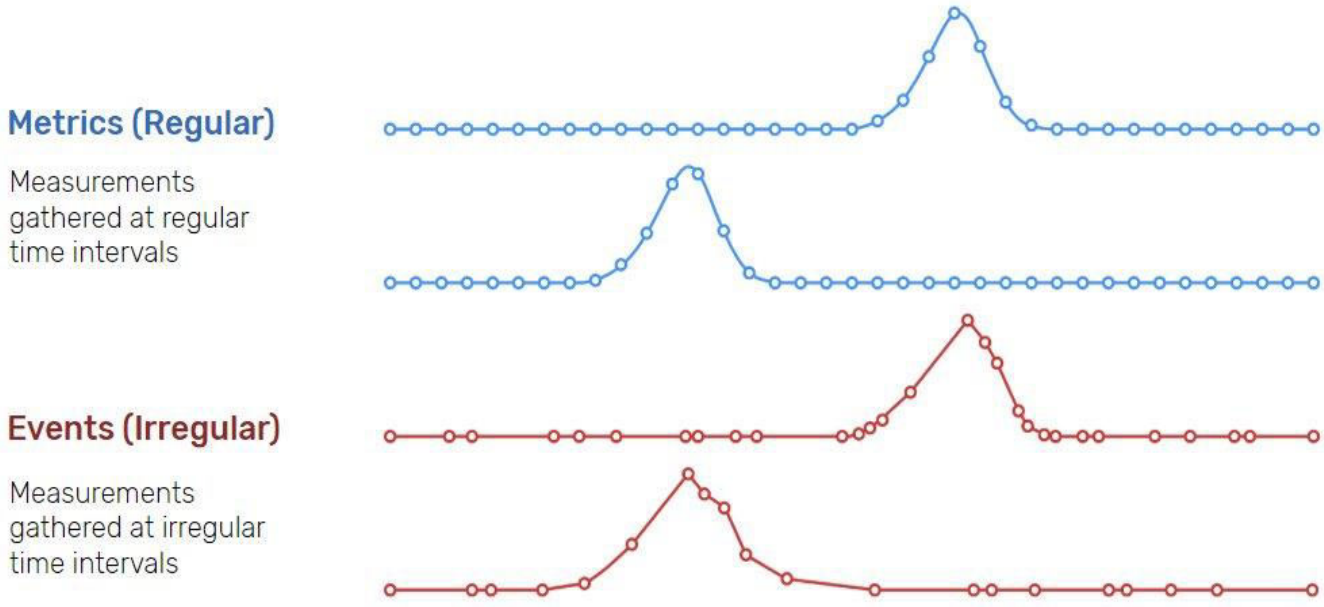
Chương 2: Kiến thức nền tảng

1.1. Tổng quan về Time-series

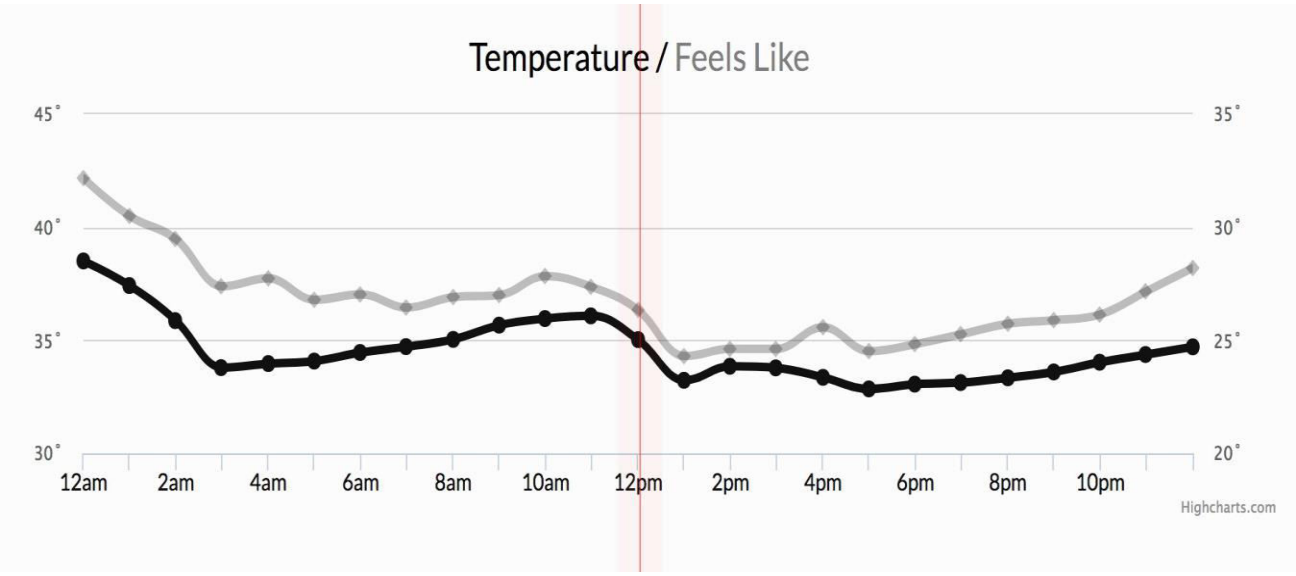
Time-series Data: là một chuỗi các điểm dữ liệu, thu được thông qua phép đo lặp lại theo thời gian.

Time-series Data có 2 loại chính:

- **Thông thường (Regular - Metrics)**: dữ liệu được ghi lại theo những khoảng thời gian đều nhau.
- **Bất thường (Irregular – Events):** dữ liệu được ghi lại khi có sự kiện hoặc biến động bất thường.

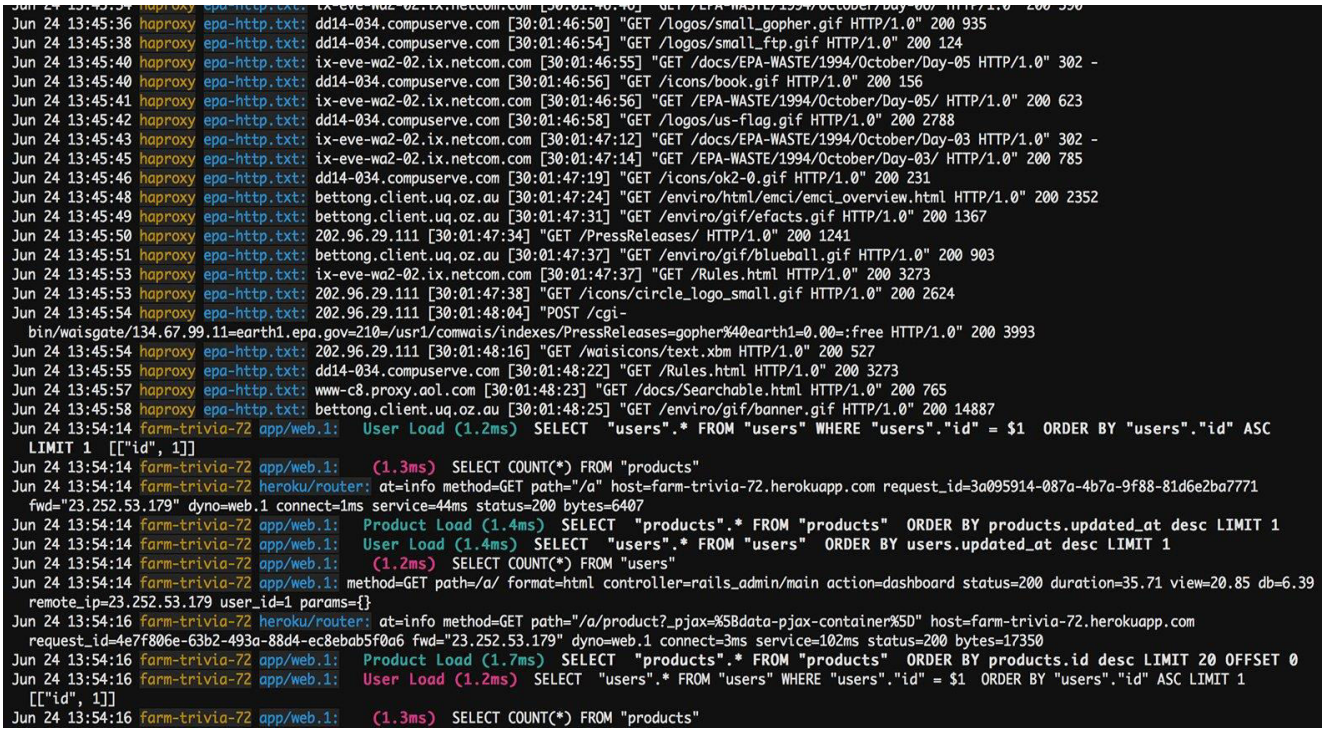


<u>Ví dụ 1:</u> Time-series Data về Nhiệt độ trong ngày, các điểm dữ liệu được đo cách nhau 2 giờ đồng hồ. Đây là **dữ liệu dạng thông thường**.



<u>Ví dụ 2:</u> Time-series Data về biến động của việc sử dụng ổ đĩa, các điểm dữ liệu được đo cách nhau 8 tiếng đồng hồ. Đây là **dữ liệu dạng thông thường**.

Ví dụ 3: Time-series Data về log của 1 hệ thống, các điểm dữ liệu được ghi lại khi có lệnh được gọi và thực hiện. Đây là **dữ liệu dạng bất thường**.



```
Jun 24 13:45:36 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:50] "GET /logos/small_gopher.gif HTTP/1.0" 200 935
Jun 24 13:45:38 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:54] "GET /logos/small_ftp.gif HTTP/1.0" 200 124
Jun 24 13:45:40 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:55] "GET /docs/EPA-WASTE/1994/October/Day-05 HTTP/1.0" 302 -
Jun 24 13:45:40 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:56] "GET /icons/book.gif HTTP/1.0" 200 156
Jun 24 13:45:41 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:56] "GET /EPA-WASTE/1994/October/Day-05/ HTTP/1.0" 200 623
Jun 24 13:45:42 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:58] "GET /logos/us-flag.gif HTTP/1.0" 200 2788
Jun 24 13:45:43 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:12] "GET /docs/EPA-WASTE/1994/October/Day-03 HTTP/1.0" 302 -
Jun 24 13:45:45 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:14] "GET /EPA-WASTE/1994/October/Day-03/ HTTP/1.0" 200 785
Jun 24 13:45:46 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:47:19] "GET /icons/ok2-0.gif HTTP/1.0" 200 231
Jun 24 13:45:48 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:24] "GET /enviro/html/emci/emci_overview.html HTTP/1.0" 200 2352
Jun 24 13:45:49 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:31] "GET /enviro/gif/efacts.gif HTTP/1.0" 200 1367
Jun 24 13:45:50 haproxy epa-http.txt: 202.96.29.111 [30:01:47:34] "GET /PressReleases/ HTTP/1.0" 200 1241
Jun 24 13:45:51 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:37] "GET /enviro/gif/blueball.gif HTTP/1.0" 200 903
Jun 24 13:45:53 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:37] "GET /Rules.html HTTP/1.0" 200 3273
Jun 24 13:45:53 haproxy epa-http.txt: 202.96.29.111 [30:01:47:38] "GET /icons/circle_logo_small.gif HTTP/1.0" 200 2624
Jun 24 13:45:54 haproxy epa-http.txt: 202.96.29.111 [30:01:48:04] "POST /cgi-
  bin/waisgate/134.67.99.11=earth1.epa.gov=210=/usr1/comwais/indexes/PressReleases=gopher%40earth1=0.00=:free HTTP/1.0" 200 3993
Jun 24 13:45:54 haproxy epa-http.txt: 202.96.29.111 [30:01:48:16] "GET /waisicons/text.xbm HTTP/1.0" 200 527
Jun 24 13:45:55 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:48:22] "GET /Rules.html HTTP/1.0" 200 3273
Jun 24 13:45:57 haproxy epa-http.txt: www-c8.proxy.aol.com [30:01:48:23] "GET /docs/Searchable.html HTTP/1.0" 200 765
Jun 24 13:45:58 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:48:25] "GET /enviro/gif/banner.gif HTTP/1.0" 200 14887
Jun 24 13:54:14 farm-trivia-72 app/web.1:   User Load (1.2ms)  SELECT  "users".* FROM "users" WHERE "users"."id" = $1  ORDER BY "users"."id" ASC
  LIMIT 1  [["id", 1]]
Jun 24 13:54:14 farm-trivia-72 app/web.1:    (1.3ms)  SELECT COUNT(*) FROM "products"
Jun 24 13:54:14 farm-trivia-72 heroku/router: at=info method=GET path="/a" host=farm-trivia-72.herokuapp.com request_id=3a095914-087a-4b7a-9f88-81d6e2ba7771
  fwd="23.252.53.179" dyno=web.1 connect=1ms service=44ms status=200 bytes=6407
Jun 24 13:54:14 farm-trivia-72 app/web.1:   Product Load (1.4ms)  SELECT  "products".* FROM "products"  ORDER BY products.updated_at desc LIMIT 1
Jun 24 13:54:14 farm-trivia-72 app/web.1:   User Load (1.4ms)  SELECT  "users".* FROM "users"  ORDER BY users.updated_at desc LIMIT 1
Jun 24 13:54:14 farm-trivia-72 app/web.1:    (1.2ms)  SELECT COUNT(*) FROM "users"
Jun 24 13:54:14 farm-trivia-72 app/web.1: method=GET path=/a/ format=html controller=rails_admin/main action=dashboard status=200 duration=35.71 view=20.85 db=6.39
  remote_ip=23.252.53.179 user_id=1 params={}
Jun 24 13:54:16 farm-trivia-72 heroku/router: at=info method=GET path="/a/product?_pjax=%5Bdata-pjax-container%5D" host=farm-trivia-72.herokuapp.com
  request_id=4e7f806e-63b2-493a-88d4-ec8ebab5f0a6 fwd="23.252.53.179" dyno=web.1 connect=3ms service=102ms status=200 bytes=17350
Jun 24 13:54:16 farm-trivia-72 app/web.1:   Product Load (1.7ms)  SELECT  "products".* FROM "products"  ORDER BY products.id desc LIMIT 20 OFFSET 0
Jun 24 13:54:16 farm-trivia-72 app/web.1:   User Load (1.2ms)  SELECT  "users".* FROM "users" WHERE "users"."id" = $1  ORDER BY "users"."id" ASC LIMIT 1
  [["id", 1]]
Jun 24 13:54:16 farm-trivia-72 app/web.1:    (1.3ms)  SELECT COUNT(*) FROM "products"
```

Ví dụ 4: Time-series Data về những lần login của 1 tài khoản, dữ liệu được ghi lại khi có phiên đăng nhập. Đây là **dữ liệu dạng bất thường.**

	A	B	C	D	E	F	G	H	I	J	K
1	CreationDate	UserIds	Operations	AuditD							
4	2018-06-13T18:49:52	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T18:49:52","Id":"60541b58-09ef-4348-8209-54399df9b7a1",							
5	2018-06-13T18:49:52	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T18:49:52","Id":"6bb03c99-2a41-4841-a320-5f16950ff013","W							
23	2018-06-13T15:14:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T15:14:00","Id":"53c782fd-0b59-47a1-8053-3f66fa1521fc","(							
24	2018-06-13T15:14:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T15:14:00","Id":"926467f6-8ea9-47df-ba70-cb38677ade29",							
25	2018-06-13T19:02:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T19:02:00","Id":"cfce27d0-1f02-48c3-872a-a2abddf2ccef","(							
29	2018-06-13T17:05:21	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T17:05:21","Id":"b6cef964-397d-49a8-bc69-e5b5406bdca1",							
38	2018-06-14T09:38:46	Unknown	UserLoggedIn	{"CreationTime":"2018-06-14T09:38:46","Id":"2b489f58-1b2a-4d2c-9a17-138d95eaf876",							
41	2018-06-13T17:05:21	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T17:05:21","Id":"da979913-f545-4ffe-941e-42c2543a3f0f","(							
57	2018-06-13T15:14:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T15:14:00","Id":"926467f6-8ea9-47df-ba70-cb38677ade29",							
58	2018-06-13T15:14:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T15:14:00","Id":"53c782fd-0b59-47a1-8053-3f66fa1521fc","(							
80	2018-06-13T15:14:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T15:14:00","Id":"53c782fd-0b59-47a1-8053-3f66fa1521fc","(							
81	2018-06-13T15:14:00	Unknown	UserLoggedIn	{"CreationTime":"2018-06-13T15:14:00","Id":"926467f6-8ea9-47df-ba70-cb38677ade29",							
98	2018-06-13T12:44:16	Unknown	UserLoginFailed	{"CreationTime":"2018-06-13T12:44:16","Id":"ad81b8b3-fb10-4e10-89d0-ad04f1fedf36",							

Time-series Analysis: dựa vào Time-series đã có thực hiện phân thích và thống kê các thông số.

Time-series Forecasting: sử dụng các yếu tố và thành phần của Time-series để thực hiện dự đoán trước về một giá trị của đối tượng đó trong khoảng thời gian nhất định trong tương lai.

Hiện tại có khá nhiều mô hình thực hiện Time-series Forecasting. Tùy theo đối tượng cần theo dõi và dự đoán mà kết quả của các loại mô hình này sẽ khác nhau. Một số mô hình tiêu biểu có:

- **Autoregressive – AR:** sử dụng các giá trị trước đó làm đầu vào cho phương trình hồi quy để dự đoán giá trị tiếp theo.

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

- **Moving Average – MA:** thay vì sử dụng các giá trị trước đó, mô hình này sử dụng các lỗi dự đoán (***εt)*** để dự đoán giá trị tiếp theo

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

- **Autoregressive Integrated Moving Average – ARIMA:** là sự kết hợp của AR và MA nhưng có vài thay đổi để tách biệt với chúng

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

- **Decomposable:** là mô hình dạng phân rã, sử dụng các thành phần để dự đoán giá trị. Công thức được đề cập ở phần sau.

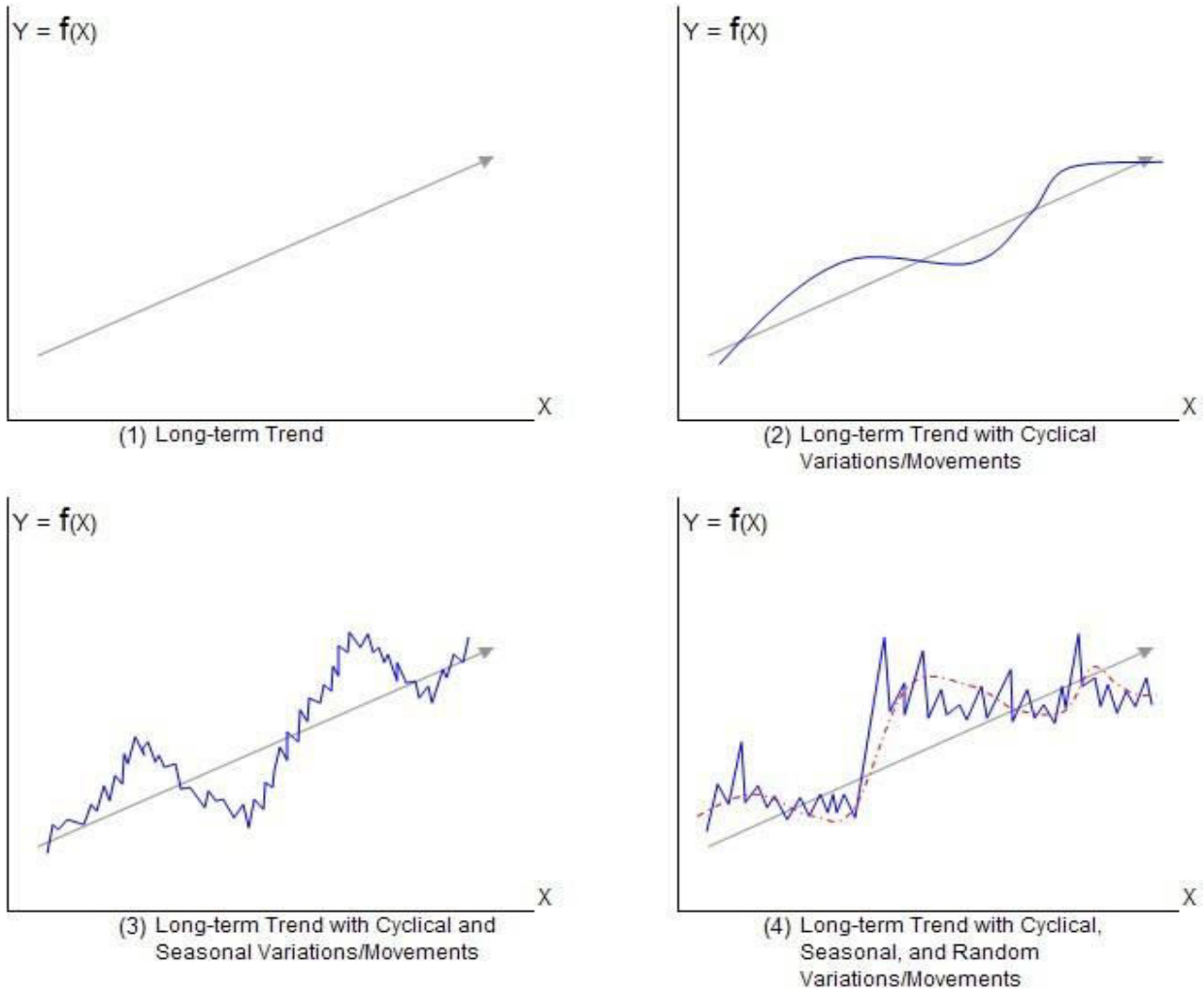
Ngoài ra còn các mô hình dự đoán khác, tùy trường hợp ta sẽ sử dụng mô hình phù hợp nhất với đối tượng cần dự đoán.

1.2. Thành phần của Time-series:

Thông thường các thành phần hợp thành time-series gồm có: **trend (T), seasonality (S), cyclical (C)** và **irregular (I).** Tùy theo cách phát triển của time-series riêng biệt mà các thành phần có thể thay đổi hoặc kết hợp với nhau. Đối với mô hình phân rã có 2 dạng kết hợp là:

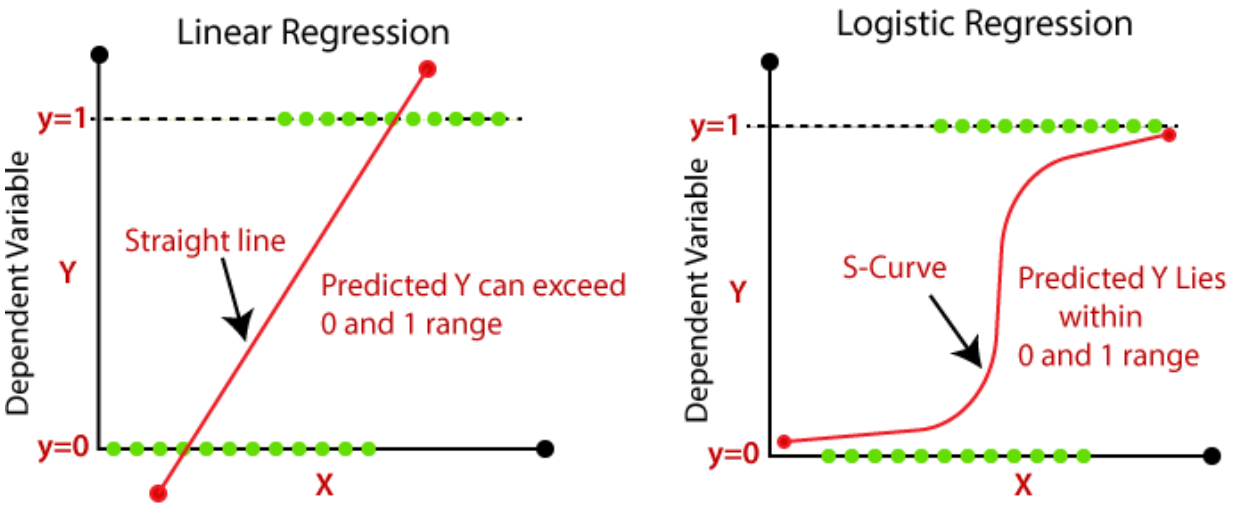
Additive model: $y(t) = T(t) + S(t) + C(t) + I(t)$

Multipicative model: $y(t) = T(t) * S(t) * C(t) * I(t)$



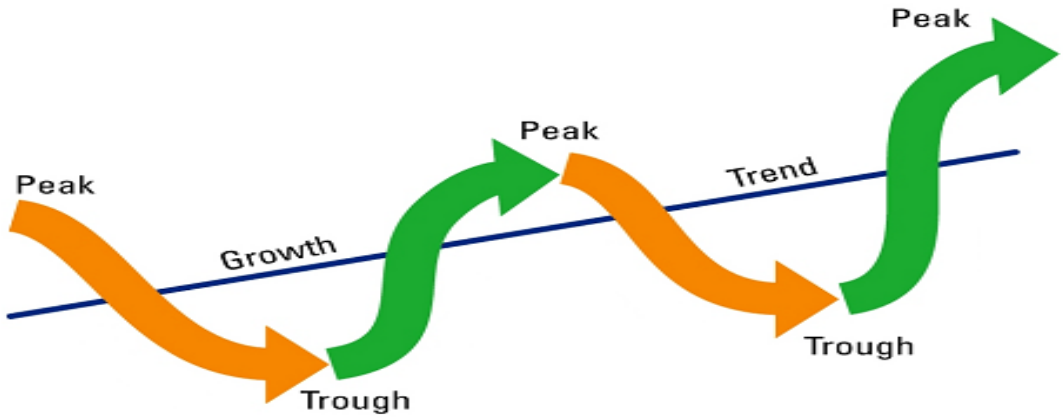
Hình ảnh thể hiện các thành phần của 1 time-series

❖ **<u>Trend (T) – xu hướng:</u>** thể hiện chiều hướng biến động, tăng hoặc giảm của time-series. Xu hướng có thể được thể hiện bằng dạng tuyến tính hoặc phi tuyến tính dựa vào giá trị growth thuộc dạng linear hay logistic. Đa số các dữ liệu thực tế đời sống có xu hướng dạng phi tuyến tính.

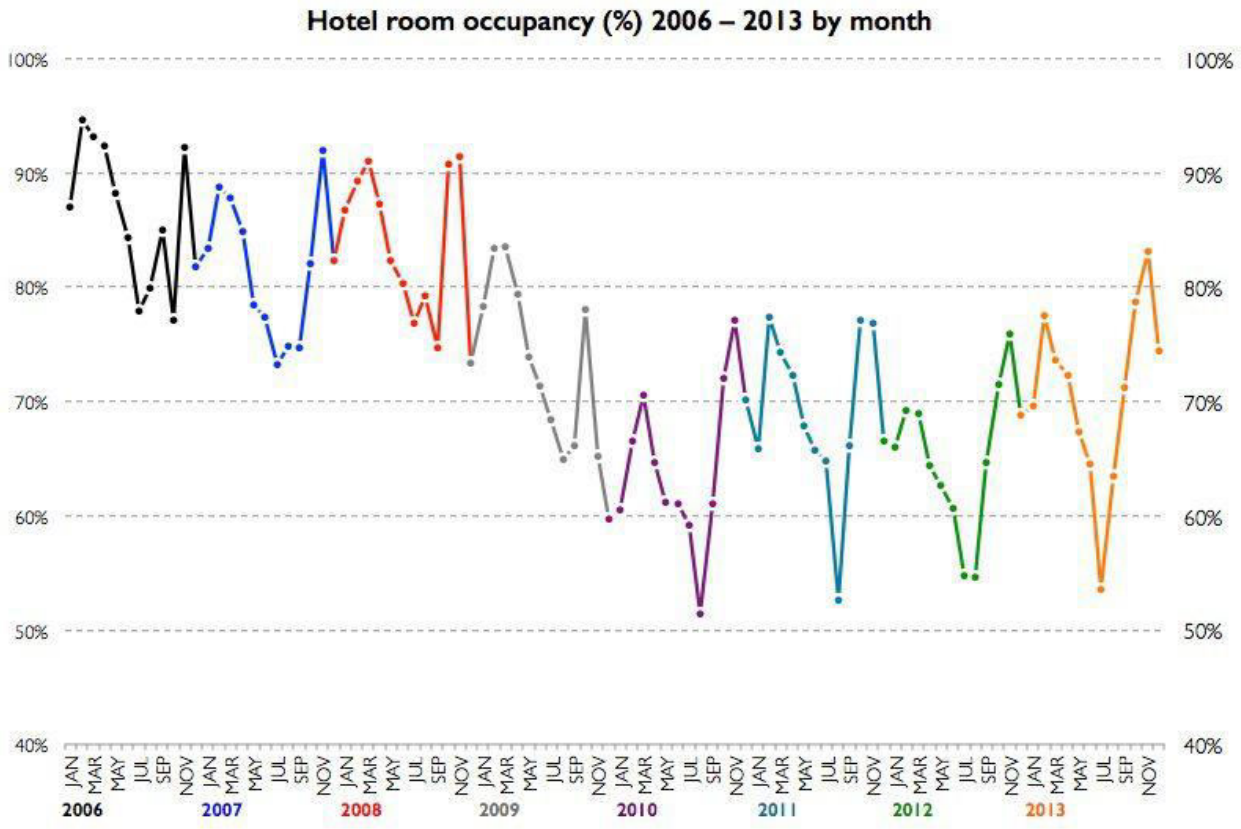


Tuyến tính (trái) và phi tuyến tính

❖ **<u>Cyclical (C) – chu kỳ:</u>** các chu kỳ là các đường cong thể hiện sự lên xuống so với đường xu hướng. Chu kỳ chỉ ra sự biến động theo chu kỳ của đối tượng.

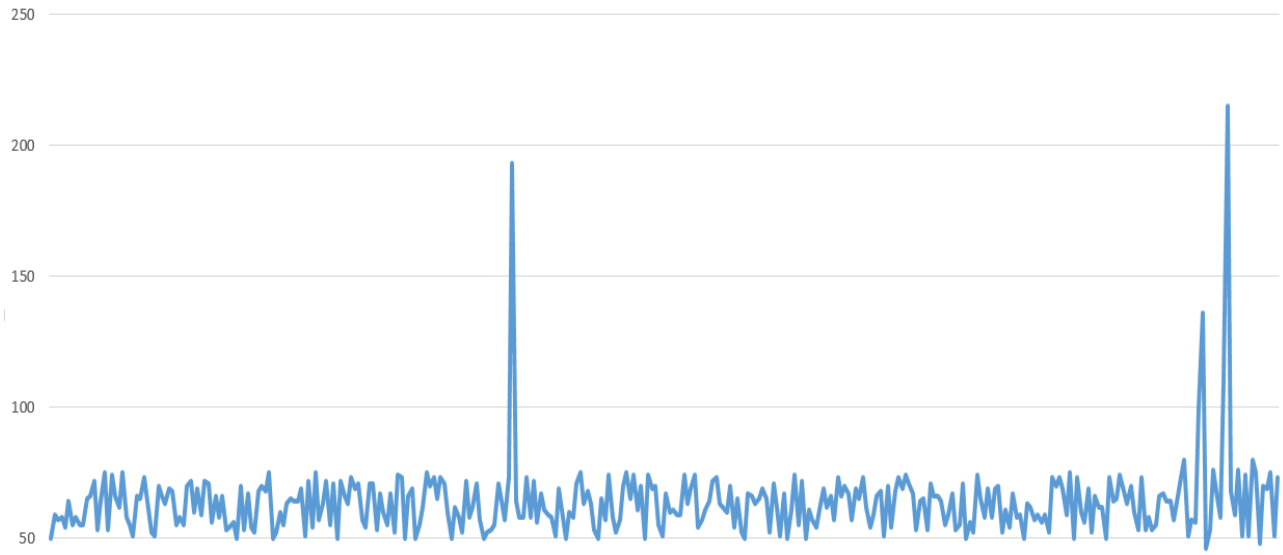


❖ **<u>Seasonality (S) - mùa:</u>** các điểm nhấp nhô hơn so với các chu kỳ là các điểm ảnh hưởng bởi mùa. Cách gọi "mùa" chỉ sự thất thường trong một khoảng thời gian so với các điểm khác trong cùng chu kỳ. Những biến động "mùa" có thể do tự nhiên (thời tiết, mùa) hoặc do con người tạo ra. Chẳng hạn như: trồng trọ phụ thuộc vào thời tiết là yếu tố tự nhiên; còn doanh thu mua bán có thể phụ thuộc vào tuần sale là yếu tố con người tạo ra.



Lượng khách ở khách sạn thường sẽ giảm vào khoảng tháng 7-9 so với các tháng khác trong năm

❖ **<u>Irregular (I) – bất thường:</u>** hay còn gọi là Random (ngẫu nhiên) Là các điểm bất thường trên đồ thị, thể hiện sự thay đổi đột ngột không thể dự đoán trước, không lặp lại theo quy luật, có mức độ khác biệt. Là yếu tố hoàn toàn ngẫu nhiên. Chẳng hạn: biểu đồ chi tiêu của 1 người sẽ có sự khác biệt trong 1 ngày họ tổ chức tiệc sinh nhật.



Những điểm cao bất thường không theo quy luật

1.3. Đánh giá lỗi:

Để đánh giá mô hình và phương pháp dự đoán, ta áp dụng các mô hình dự đoán tìm ra kết quả dự đoán, sau đó khi có dữ liệu thực tế ta sẽ đem so sánh chúng với nhau.

$$\varepsilon_t = x(t) - f(t)$$

với t là mốc thời gian t, x(t) là giá trị thực tế, còn f(t) là giá trị dự đoán

Sau đó tìm **Mean absolute error (MAE)** và **Mean absolute percentage error (MAPE)**:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\varepsilon_t|$$

$$MAPE = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{\varepsilon_t}{y_t}\right|$$

Kết quả các chỉ số MAE và MAPE càng nhỏ thì dự đoán càng hiệu quả.

1.4. Ứng dụng của Time-series Data và Time-series Forecasting:

Time-series Data và Time-series Forecasting có vai trò quan trọng trong việc quản lý và dự đoán thông tin trong nhiều lĩnh vực.

- **Thời tiết:** dự đoán thời tiết, mực nước, độ ẩm, dự báo thiên tai,… Các dự báo này vô cùng quan trọng đối với đời sống con người và các lĩnh vực liên quan như nông nghiệp, du lịch, giao thông.
- **Kinh tế / tài chính:** dự đoán biến động thị trường, dự đoán giá vàng, dự đoán doanh thu, dự đoán cung cầu.
- **Du lịch:** dự đoán lượng khách du lịch, dự đoán doanh thu từ du lịch.
- **Giao thông / vận tải:** dự đoán lưu lượng lưu thông.
- **Xã hội:** dự đoán tỉ lệ gia tăng dân số, dự đoán phân bố dân cư.
- **Công nghệ thông tin:** dự đoán lượng truy cập trang web, dự đoán số lượng post trên diễn đàn, dự đoán số người sử dụng ứng dụng.

Ngoài ra còn những lĩnh vực và ứng dụng khác trong đời sống.

Chương 3: Công cụ dự báo Prophet

1.1. Tổng quan về Facebook Prophet:

Prophet là một thư viện mã nguồn mở (open source library) được phát triển bởi đội Core Data Science của Facebook. Bản đầu tiên v0.1 được ra mắt tháng 3/2017. Bản released mới nhất hiện tại của Prophet là bản **v0.7.1**.

Trang chủ của Facebook Prophet: https://facebook.github.io/prophet/

Prophet được phát triển trên 2 ngôn ngữ là **Python** và **R**. Tuy là 2 ngôn ngữ khác nhau nhưng đều có chung nền tảng Stan (nền tảng thống kê và hỗ trợ xác suất thống kê hiệu suất cao). Để sử dụng Prophet trên cả 2 ngôn ngữ thì cần cài đặt Stan cho ngôn ngữ tương ứng trước: đối với Python là **PyStan** và đối với R là **RStan**. Trong báo cáo này, em sẽ tìm hiểu Prophet qua ngôn ngữ **Python**.

Prophet có dạng model phân rã dựa trên **Additive Model** với **trend-cyclical** (trong các model phân rã thường kết hợp trend và cyclical), có **seasonality** theo ngày, tuần, năm. Cùng với đó là thành phần **holidays** thể hiện những ảnh hưởng của khoảng thời gian lễ, đặc biệt. Chúng được kết hợp theo công thức sau:

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

Trong đó: $g(t)$ là thành phần trend-cyclical, $s(t)$ là thành phần seasonality theo định kỳ có quy luật, $h(t)$ là thành phần holidays được người dùng hoặc thư viện cung cấp, ε_t là chỉ số lỗi.

Theo các nhà nghiên cứu của Facebook, họ xem vấn đề của time-series forecasting nằm ở việc tính toán và thể hiện "đường cong" của đồ thị, khác với các mô hình khác thực hiện việc giải thích rõ ràng cấu trúc phụ thuộc trong dữ liệu.

Vì thế, họ sử dụng mô hình phân rã và bỏ qua một số lợi thế của các mô hình có tính suy luận chặt chẽ như ARIMA. Dù vậy, cách làm này cũng phát huy một vài ưu điểm:

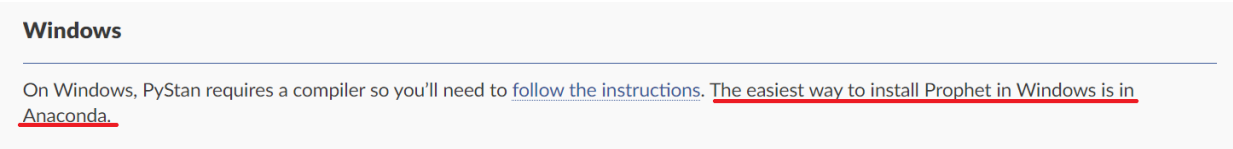
- ❖ Tính linh hoạt (Flexibility): dễ dàng điều chỉnh seasonality với các khoảng khác nhau (theo ngày, theo tuần, theo năm)
- ❖ Không như mô hình ARIMA, các phép tính toán không cần thực hiện trong không gian chính quy (không gian Metrics: https://en.wikipedia.org/wiki/Metric_space) và không cần nội suy các giá trị bị thiếu.
- ❖ Fit nhanh chóng (trong forecasting, **fit** là thuật ngữ dùng để chỉ việc tính toán các giá trị thành phần dựa vào dữ liệu đầu vào sao cho phù hợp)
- ❖ Có các giá trị thành phần có thể thay đổi được, giúp phù hợp hơn với đối tượng cần dự đoán.

Mục tiêu ban đầu của dự án Prophet là dự đoán số liệu liên quan đến Facebook, sau đó quy mô của họ mở rộng hơn trở thành công cụ hỗ trợ được sử dụng bởi nhiều người. Đó là lý do mà mô hình của họ có hướng hơi thiên về yếu tố sao cho người dùng dễ dàng tiếp cận và tìm hiểu (có thành phần holidays, các giá trị có thể thay đổi).

1.2. Yêu cầu và cài đặt:

Prophet version mới nhất v0.7.1 có các yêu cầu sau về ngôn ngữ và Package:

- Ngôn ngữ **Python 3.0.0** trở lên
- Để download và cài các gói package cần có **PyPi** hoặc **Anaconda3** (được Prophet khuyên dùng)



- Các package và version tối thiểu:

Package	Version tối thiểu
Cython	0.22
cmdstanpy	0.9.5
pystan	2.14
numpy	1.15.4
pandas	1.0.4
matplotlib	2.0.0
LunarCalendar	0.0.9
convertdate	2.1.2
holidays	0.10.2
python-dateutil	2.8.0
tqdm	4.36.1
plotly	3.5.0

Để cài Python, download gói cái đặt tại **python.org** và tiến hành cài đặt như bình thường.

Các bản Python 3.3 trở lên đều có tích hợp sẵn PyPi.

Để cài các package, chạy python và nhập:

```
$ pip install [tên package]
```

Ví dụ để cài pystan nhập:

```
$ pip install pystan
```

Hoặc nếu sử dụng Anaconda chạy Anaconda và nhập:

```
conda install -c conda-forge [tên package]
```

Ví dụ để cài pystan nhập:

```
conda install -c conda-forge pystan
```

Sau khi đã cài đầy đủ các package, cài fbprophet:

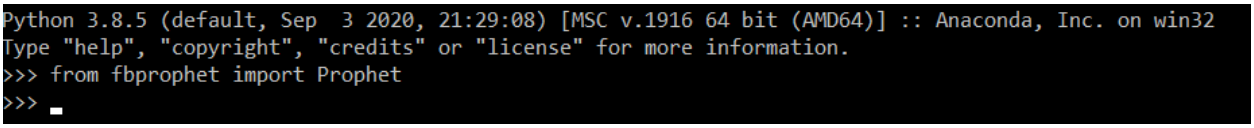
```
$ pip install fbprophet
```

Hoặc

```
conda install -c conda-forge fbprophet
```

Sau khi đã cài đặt xong tất cả có thể test bằng cách nhập như sau:

```
from fbprophet import Prophet
```

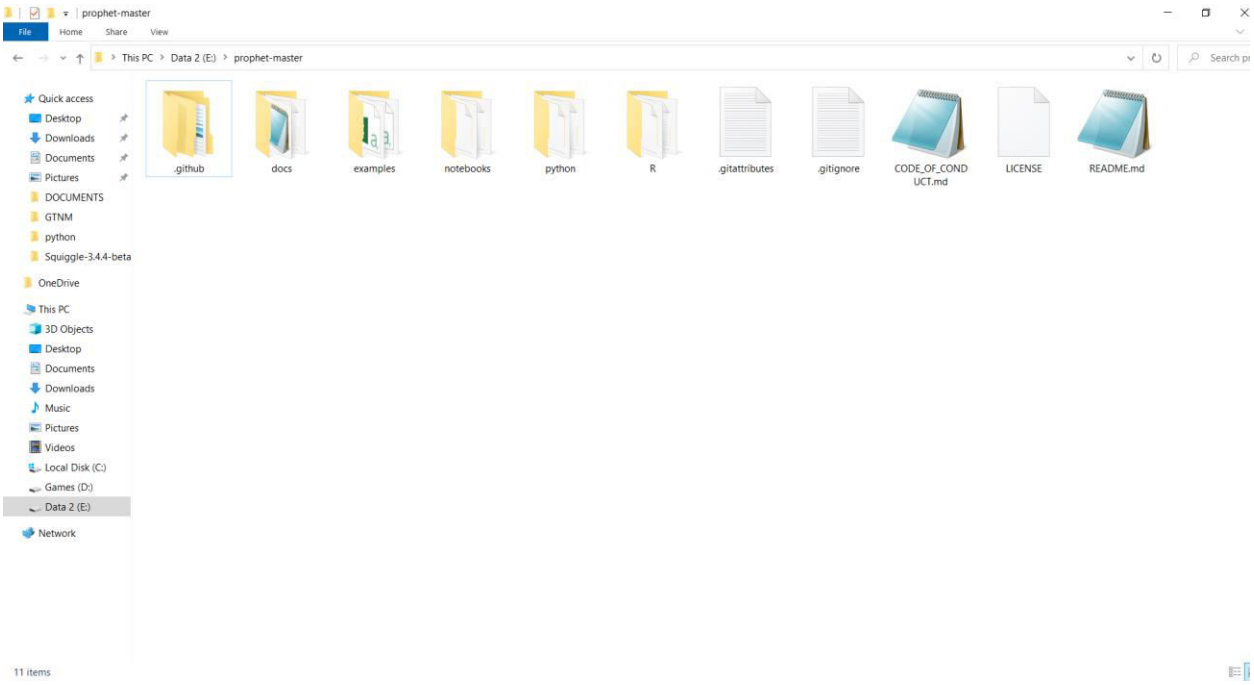


Nếu không hiện bất cứ lỗi nào như ảnh trên là Prophet đã cài đặt thành công.

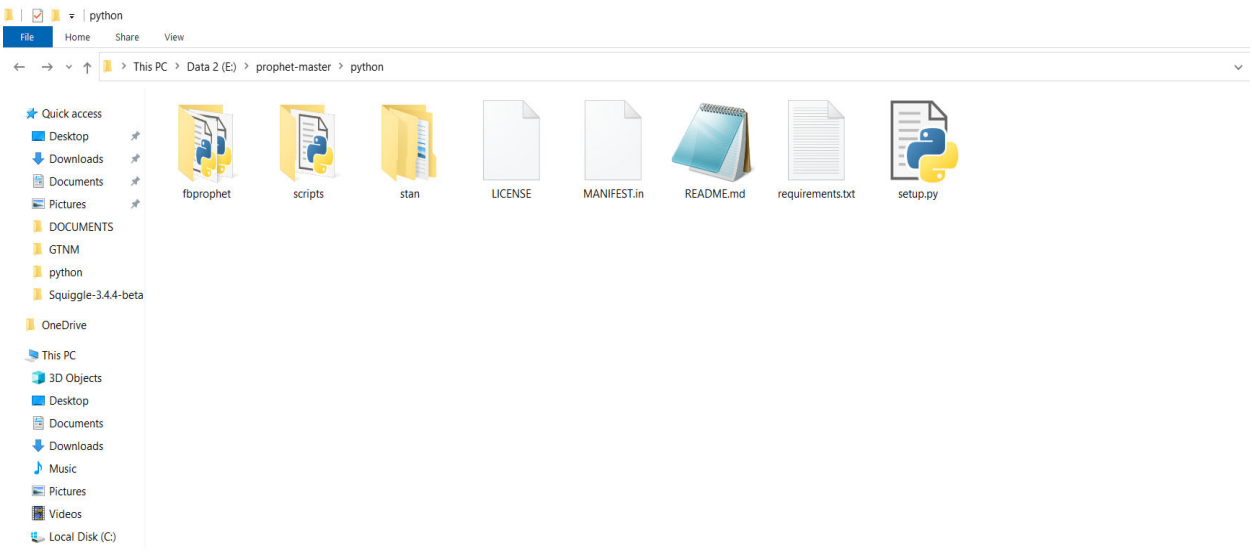
1.3. Tìm hiểu mã nguồn của Prophet:

Để tìm hiểu cấu trúc mã nguồn của Prophet, có thể clone hoặc download về từ GitHub của Prophet (https://github.com/facebook/prophet).

Bên trong folder prophet-master sẽ gồm các files và folders sau:



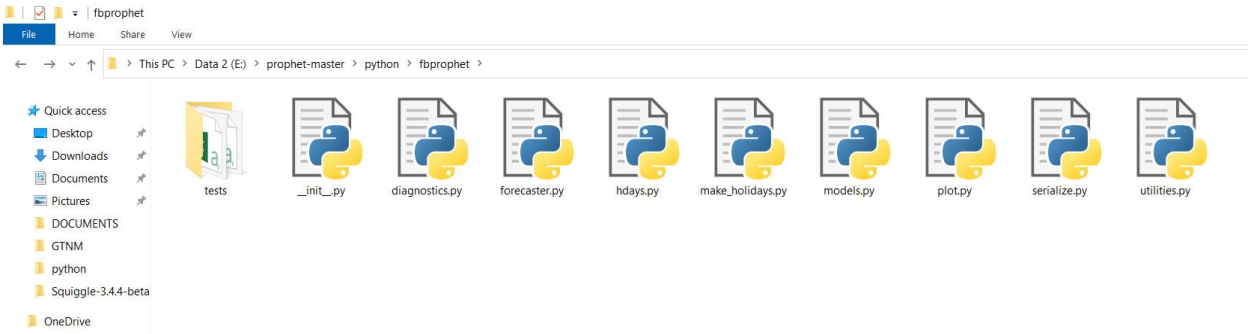
Trong đó các folders **docs, notebooks** và **examples** chứa các tài liệu và input ví dụ của Prophet. Mã nguồn chính của Prophet nằm trong folder **python** và **R** tương ứng với ngôn ngữ phát triển. Ta sẽ xét folder **python** ứng với ngôn ngữ Python.



File **setup.py** và **requirement.txt** là file cài đặt và yêu cầu tối thiểu.

Folder **stan** là nền stan của mã nguồn.

Mã nguồn của class Prophet và các hàm nằm trong thư mục **fbprophet**, ta sẽ xét folder này.



Thư mục **test** chứa các file test và input test.

File **_init_.py** thể hiện version của Prophet.

File **forecaster.py** chứa **class Prophet()** cùng các tham số và hàm của nó.

❖ **Các tham số (parameters):**

Tên tham số	Kiểu	Chú thích
growth	String	'linear' hoặc 'logistic' để thể hiện trend dạng tương ứng
changepoints	List of dates	danh sách các ngày có changepoint (thể hiện sự thay đổi của đồ thị)
yearly_seasonality	String	'auto' hoặc 'true' hoặc 'false' thể hiện seasonality tương ứng
weekly_seasonality	String	như yearly_seasonality
daily_seasonality	String	như yearly_seasonality
holidays	pandas DataFrame	đưa vào danh sách holiday thể hiện thành phần holidays
seasonality_mode	String	biểu thị dạng additive hoặc multiplicative
seasonality_prior_scale	Float	giá trị thay đổi mức ảnh hưởng của seasonality (càng lớn càng ảnh hưởng nhiều)
holidays_prior_scale	Float	giá trị thay đổi mức ảnh hưởng của holidays (càng lớn càng ảnh hưởng nhiều)
changepoint_prior_scale	Float	giá trị thay đổi cách chọn lọc changepoint (càng lớn càng chọn lọc nhiều changepoint)
stan_backend	Stan	Stan backend của Prophet

❖ **Các hàm (function):**

Tên hàm	Chú thích
load_stan_backend	Load stan backend của Prophet
validate_inputs	Kiểm tra các giá trị input của Prophet
validate_column_name	Kiểm tra tên các giá trị seasonality, holidays
setup_dataframe	Chuẩn bị dataframe để fit hoặc dự đoán
initialize_scales	Điều chỉnh các tham số liên quan đến scale để fit dataframe
set_changepoints	Quét các changepoints và gán vào tham số changepoints
fourier_series	Hàm tính chuỗi Fourier dùng để xét các tham số seasonality

make_seasonality_features	Liệt kê các seasonality và giá trị của nó vào 1 dataframe
construct_holiday_dataframe	Kết hợp & kiểm tra các holidays của Prophet với các holidays người dùng thêm vào (tự tạo / từ thư viện khác)
make_holidays_features	Liệt kê các holidays và giá trị của nó vào 1 dataframe
add_regressor	Thêm hàm hồi quy dùng để fit hoặc dự đoán vào object Prophet
add_seasonality	Thêm 1 seasonality vào object Prophet
add_country_holidays	Thêm holidays của 1 quốc gia vào object Prophet (đây là holidays của Prophet, không phải từ thư viện ngoài hay tự tạo)
add_group_components	Tạo 1 nhóm gồm những thành phần liên quan đến nhau (VD: 1 group cùng seasonality)
parse_seasonality_args	Trả về giá trị chuỗi Fourier của seasonality
set_auto_seasonality	Tự điều chỉnh các giá trị yearly_seasonality, weekly_seasonality, daily_seasonality sao cho phù hợp với dataframe
linear_growth_init	Tính các giá trị linear growth (tăng trưởng tuyến tính) cho thành phần trend phù hợp với dataframe
logistic_growth_init	Tính các giá trị logistic growth (tăng trưởng phi tuyến tính) cho thành phần trend phù hợp với dataframe
flat_growth_init	Tính các giá trị flat growth (khi các giá trị đều nhau) cho thành phần trend phù hợp với dataframe
fit	Fit object Prophet với dataframe truyền vào
predict	Tính toán và đưa ra các giá trị dự đoán
piecewise_linear	Tính vector tuyến tính từng đoạn (chia đoạn trên đồ thị)
piecewise_logistic	Tính vector phi tuyến tính từng đoạn (chia đoạn trên đồ thị)
predict_trend	Tính toán và đưa ra giá trị dự đoán của thành phần trend
predict_seasonality_components	Tính toán và đưa ra giá trị dự đoán của thành phần seasonality
sample_model	Tính giá trị theo công thức mô hình phân rã của Prophet
predict_uncertainty	Tính miền giá trị lệch có thể có
sample_predictive_trend	Mô phỏng giá trị trend bằng mô hình ngoại suy
make_future_dataframe	Trả về dataframe chứa khoảng thời gian trong tương lai cần dự đoán (VD: thêm ngày vào dataframe đã fit)

plot	Vẽ đồ thị thể hiện giá trị cho object Prophet
plot_component	Vẽ đồ thị thể hiện tất cả giá trị thành phần (trend, seasonality, holidays)

File **diagnostics.py** chứa các hàm hỗ trợ việc phân tích sau khi dự đoán như tính lỗi, tính độ lệch…

❖ **Các hàm (function):**

Tên hàm	Chú thích
generate_cutoffs	Cắt dataframe ra những khoảng thời gian nhất định để phục vụ việc phân tích, xem xét
cross_validation	Trả về dataframe chứa các giá trị dự đoán trong các khoảng cutoff
single_cutoff_forecast	Trả về giá trị dự đoán của 1 giá trị cutoff
prophet_copy	Copy 1 prophet object
mse	Hàm tính lỗi Mean squared error
rmse	Hàm tính lỗi Root mean squared error
mae	Hàm tính lỗi Mean absolute error
mape	Hàm tính lỗi Mean absolute percentage error
mdape	Hàm tính lỗi Median absolute percentage error
coverage	Hàm tính Coverage
performance_metrics	Tổng hợp các giá trị lỗi vào 1 dataframe

File **hdays.py** chứa các class holidays của các nước, dữ liệu về các ngày lễ được tổng hợp từ Wikipedia. Ngoài file hdays.py thì các ngày lễ các nước khác được thêm vào từ thư viện **holidays** của Python. VD: Việt Nam có class Vietnam chứa hàm và thuộc tính định nghĩa các ngày lễ sau: Tết Nguyên Đán 29/12-5/1 Âm

lịch (được tính dựa vào thư viện **LunarCalendar** của Python), Tết Dương Lịch 1/1, giỗ Tổ Vua Hùng 10/3 Âm lịch, ngày Thống nhất Đất nước 30/4, Quốc tế Lao động 1/5, Quốc khánh 2/9. Để gọi class Vietnam gọi hàm add_country_holiday(country_name= "VN").

File **make_holidays.py** chứa hàm để lấy tên các ngày lễ và tạo dataframe gồm các ngày lễ của quốc gia được gọi.

❖ **Các hàm (function):**

Tên hàm	Chú thích
get_holiday_name	Trả về 1 tập tên các ngày lễ của quốc gia được gọi
make_holiday_df	Tạo dataframe gồm các tên ngày lễ và ngày cụ thể

File **models.py** chứa các class về Stan Backend và hàm để gọi Stan Backend.

File **plot.py** chứa hàm về plot để vẽ đồ thị cho object Prophet.

❖ **Các hàm (function):**

Tên hàm	Chú thích
plot	Vẽ đồ thị thể hiện các giá trị cho object Prophet
plot_components	Vẽ đồ thị thể hiện tất cả giá trị thành phần (trend, seasonality, holidays)
plot_forecast_component	Vẽ đồ thị thể hiện 1 giá trị thành phần cụ thể (trend hoặc seasonality hoặc holidays)
seasonality_plot_df	Trả về dataframe với giá trị thành phần seasonality cụ thể (yearly, weekly, daily)
plot_weekly	Vẽ đồ thị thể hiện weekly seasonality
plot_yearly	Vẽ đồ thị thể hiện yearly seasonality
plot_seasonality	Vẽ đồ thị thể hiện 1 seasonality cụ thể (yearly hoặc weekly hoặc daily)
add_changepoints_to_plot	Thêm các changepoints vào đồ thị
plot_cross_validation_metric	Vẽ đồ thị thể hiện các giá trị tính lỗi
plot_plotly	Vẽ đồ thị bằng thư viện plotly của Python

File **utilities.py** chứa các hàm liên quan đến hồi quy và ma trận.

1.4. Thực hiện dự đoán bằng Prophet:

Sau khi đã tìm hiểu về cấu trúc mã nguồn với các class và hàm quan trọng giờ có thể bắt đầu thực hiện dự đoán với Prophet.

Phần thực hiện dự đoán được viết bằng ngôn ngữ Python, có nhiều công cụ để viết file .py như **NotePad++** hay **Atom**. Ở phần này, em sẽ thực hiện trên **Jupyter Notebook** (được hỗ trợ khi cài đặt Anaconda3).

Dữ liệu nguồn được lấy trong folder example của Prophet "example_wp_log_peyton_manning.csv". Dữ liệu nguồn dạng csv với 2 column: date và y. Có thể download file thực hiện test.py và dữ liệu nguồn tại: https://github.com/minhduystg1999/prophet-python

	date	y
2	12/10/2007	9.590761
3	12/11/2007	8.51959
4	12/12/2007	8.183677
5	12/13/2007	8.072467
6	12/14/2007	7.893572
7	12/15/2007	7.783641
8	12/16/2007	8.414052
9	12/17/2007	8.829226
10	12/18/2007	8.382518
11	12/19/2007	8.069655
12	12/20/2007	7.879291
13	12/21/2007	7.761745
14	12/22/2007	7.529406
15	12/23/2007	8.385261
16	12/24/2007	8.620111
17	12/25/2007	7.852439
18	12/26/2007	7.853993
19	12/27/2007	8.051978
20	12/28/2007	7.926603
21	12/29/2007	7.838343
22	12/30/2007	9.703145
23	12/31/2007	9.385973
24	1/1/2008	8.2938
25	1/2/2008	8.434681
26	1/3/2008	8.262043

❖ **Đầu tiên import các thư viện cần thiết:**

Sau khi tạo file .py, ta bắt đầu import các thư viện.

```
import numpy as np
import pandas as pd
from scipy import stats
import plotly.graph_objects as go
from fbprophet import Prophet
```

```
1  import numpy as np
2  import pandas as pd
3  from scipy import stats
4  import plotly.graph_objects as go
5  from fbprophet import Prophet
6  from fbprophet.plot import add_changepoints_to_plot
7  from fbprophet.diagnostics import cross_validation
8  from fbprophet.diagnostics import performance_metrics
9  from fbprophet.plot import plot_cross_validation_metric
10
```

numpy: liên quan đến các phép tính đa chiều, mảng.

pandas: hỗ trợ dataframe và đọc xuất file.

scipy: liên quan đến các công thức toán học, xác suất, thống kê.

plotly: liên quan đến việc vẽ các đồ thị

fbprophet: class Prophet cần để định nghĩa object Prophet, ngoài ra là các hàm liên quan sẽ được sử dụng.

❖ **Nhập file .csv và in ra màn hình:**

```
df = pd.read_csv('[filepath]')
print(df)
```

Thay **[filepath]** bằng đường dẫn đến file input.
Sau đó chạy file test.py

```python
import numpy as np
import pandas as pd
from scipy import stats
import plotly.graph_objects as go
from fbprophet import Prophet
from fbprophet.plot import add_changepoints_to_plot
from fbprophet.diagnostics import cross_validation
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric


df = pd.read_csv('example_wp_log_peyton_manning.csv')

print(df)

```

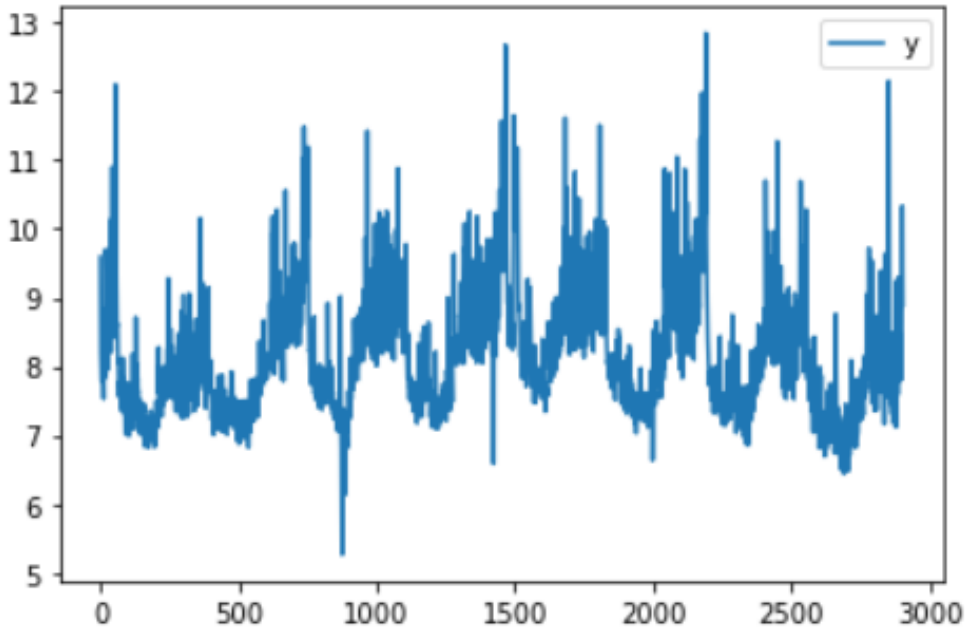
Kết quả:

	date	y
0	12/10/2007	9.590761
1	12/11/2007	8.519590
2	12/12/2007	8.183677
3	12/13/2007	8.072467
4	12/14/2007	7.893572
...	...	...
2900	1/16/2016	7.817223
2901	1/17/2016	9.273878
2902	1/18/2016	10.333775
2903	1/19/2016	9.125871

Thử vẽ đồ thị thể hiện input:

`df.plot()`

Kết quả:



❖ **Tạo Prophet object và fit dataframe:**

Đầu tiên cần chỉnh các column của dataframe ban đầu theo dạng của Prophet là **[‘ds’, ‘y’]**.

```
df.columns = [‘ds’, ‘y’]
df['ds'] = pd.to_datetime(df['ds'])
```

Do ta dự đoán dữ liệu dựa trên dữ liệu sẵn có đầu vào và sau đó cần tính lỗi nên ta sẽ tạo training dataframe cắt bớt dữ liệu từ dataframe ban đầu. Ta sẽ cắt bớt 150 ngày cuối.

```
prediction_size = 150
train_df = df[:-prediction_size]
```

Tạo object Prophet và **fit** training dataframe.

```
p = Prophet()
p.fit(train_df)
```

```
11
12 df = pd.read_csv('example_wp_log_peyton_manning.csv')
13
14 print(df)
15
16 df.plot()
17
18 df.columns = ['ds', 'y']
19 df['ds'] = pd.to_datetime(df['ds'])
20
21 prediction_size = 150
22 train_df = df[:-prediction_size]
23
24 p = Prophet()
25 p.fit(train_df)
26
```

❖ **Tạo future dataframe và thực hiện dự đoán:**

Ta dùng hàm **make_future_dataframe** để thêm số lượng tháng mà ta đã cắt bớt đi để tiến hành dự đoán và gán nó vào biến mới future.

```
future = p.make_future_dataframe(periods=prediction_size, freq='M')
```

Ở đây **periods = 150** và **freq = 'D'** tương ứng cho **150 ngày**. Đối với các time-series data dạng tháng thì **freq = 'M'**, còn dạng năm là **freq = 'Y'**.

Sau đó chỉ cần dự đoán bằng hàm **predict(dataframe)** và in ra kết quả ta vừa dự đoán.

```
forecast = p.predict(future)
print(forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail(10))
```

Kết quả của predict là dataframe với nhiều cột, ở đây ta nên chú ý 3 cột là **'yhat'**, **'yhat_lower'** và **'yhat_upper'**. Vì thế ta nên in ra màn hình 3 cột này.

Trong đó: **'yhat'** là giá trị dự đoán dựa vào mô hình phân rã của Prophet; **'yhat_lower'** và **'yhat_upper'** lần lượt là biên dưới và biên trên của 'yhat', tức là khả năng dao động thấp nhất hoặc cao nhất của 'yhat'.

```
25
26 future = p.make_future_dataframe(periods=prediction_size, freq='D')
27 forecast = p.predict(future)
28 print(forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail(10))
29
```

Kết quả ta có được giá trị dự đoán của những ngày cuối:

	ds	yhat	yhat_lower	yhat_upper
2895	2016-01-10	8.331660	7.767044	8.998898
2896	2016-01-11	8.637315	7.985801	9.272623
2897	2016-01-12	8.441010	7.841793	9.042477
2898	2016-01-13	8.290986	7.649342	8.936600
2899	2016-01-14	8.312079	7.687082	8.927345
2900	2016-01-15	8.332458	7.696423	8.969635
2901	2016-01-16	8.126265	7.442366	8.769566
2902	2016-01-17	8.490562	7.803089	9.065046
2903	2016-01-18	8.793215	8.187262	9.426912
2904	2016-01-19	8.592981	7.957910	9.154725

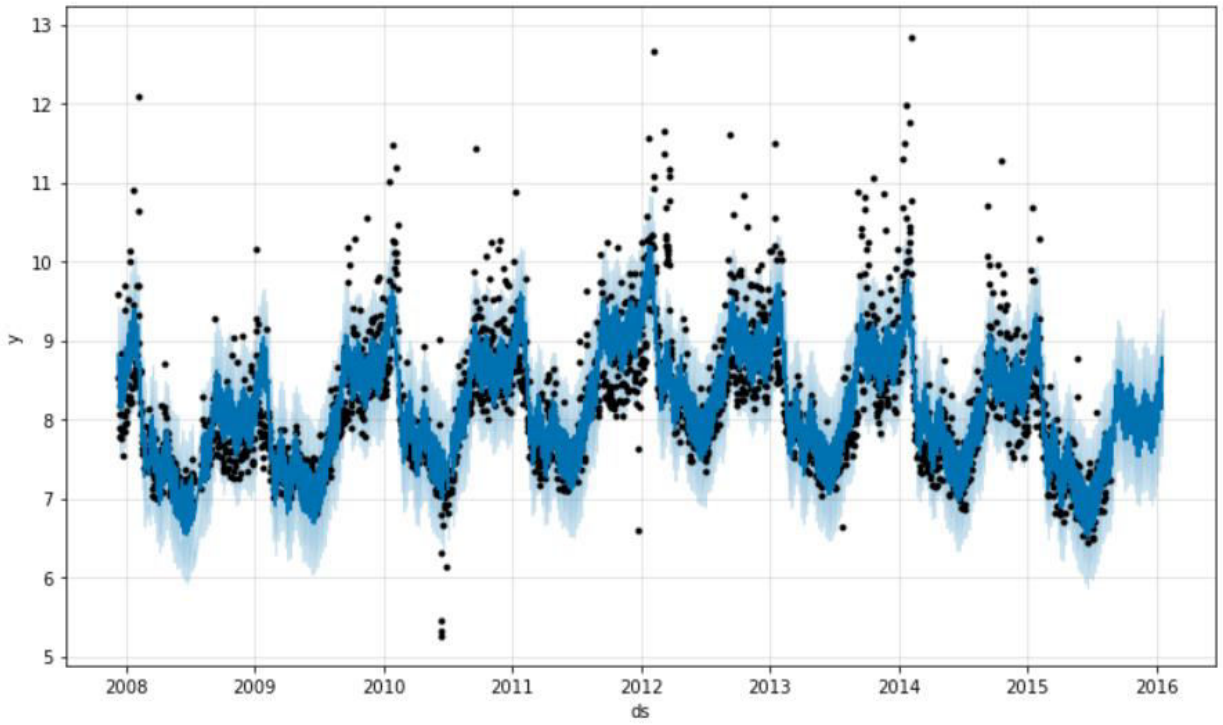
❖ **Vẽ đồ thị:**

Vẽ đồ thị bằng hàm **plot** và vẽ đồ thị các thành phần bằng hàm **plot_components**.

```
p.plot(forecast)
```

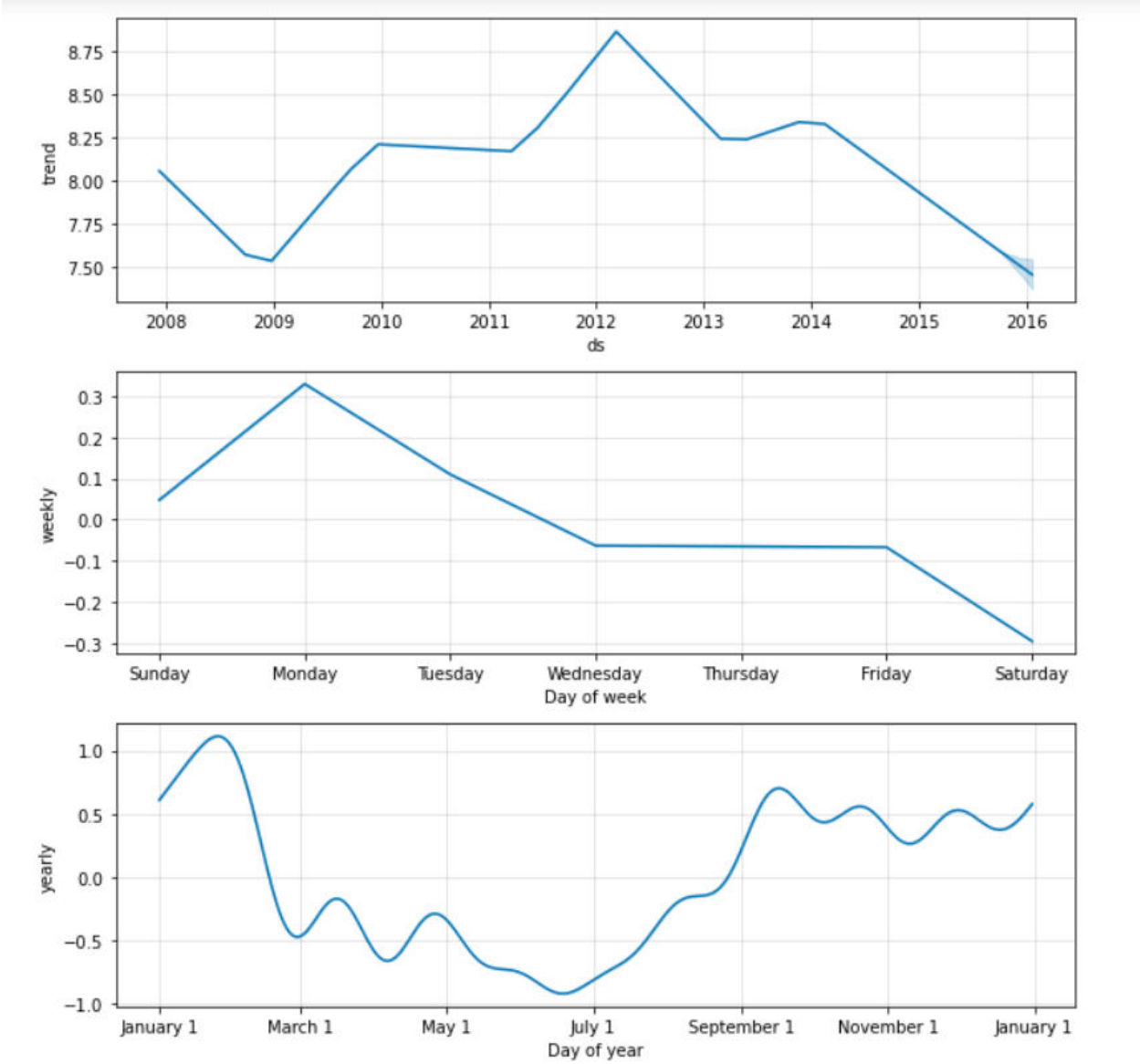
```
p.plot_components(forecast)
```

Kết quả:



Đồ thị từ hàm plot

Đường xanh đậm thể hiện giá trị yhat dự đoán được còn phần rìa sắc nhạt hơn là miền dao động của yhat, cao nhất là yhat_upper, thấp nhất là yhat_lower. Các chấm đen là các giá trị ban đầu trong training dataframe. Giai đoạn năm cuối 2015 đầu 2016 còn lại chính là 150 ngày được dự đoán (không có chấm đen nào).



Ba đồ thị thành phần từ hàm plot_components

Đồ thị đầu tiên thể hiện thành phần **trend** theo mô hình phân rã của Prophet, đoạn xanh cuối thành phần trend là phần chúng ta dự đoán. Đồ thị thứ 2 và 3 các thành phần **weekly seasonality** và **yearly_seasonality**. Nếu có thêm thành phần holidays thì sẽ có đồ thị của thành phần đó (dữ liệu này không có thành phần holidays).

❖ **Tạo comparision dataframe để so sánh kết quả:**

Do các hàm tính lỗi của prophet chỉ tính được lỗi đối với dữ liệu có trước nên với cách làm cắt bớt giá trị ta không thể so sánh kết quả và xem các chỉ số lỗi. Vì thế tiếp theo sẽ tạo một hàm mới giúp tạo 1 dataframe chứa cả giá trị dự đoán được và giá trị có trước đó để so sánh. Cách làm đơn giản là nối giá trị của 2 dataframe với nhau.

```
def make_comparison_dataframe(initial, forecast):
return forecast.set_index('ds')[['yhat', 'yhat_lower', 'yhat_upper']].join(initial.set_index('ds'))
```

```
31
32 def make_comparison_dataframe(initial, forecast):
33     return forecast.set_index('ds')[['yhat', 'yhat_lower', 'yhat_upper']].join(initial.set_index('ds'))
34
35 cmp_df = make_comparison_dataframe(df, forecast)
```

initial là dataframe ban đầu.

forecast là dataframe nhận được sau khi dùng Prophet predict.

Kết quả khi in 10 dòng cuối của **cmp_df (comparison dataframe vừa tạo):**

ds	yhat	yhat_lower	yhat_upper	y
2016-01-10	8.331660	7.721263	8.945197	8.281724
2016-01-11	8.637315	8.039227	9.295162	8.470730
2016-01-12	8.441010	7.813670	9.014481	8.135054
2016-01-13	8.290986	7.681640	8.895751	8.067149
2016-01-14	8.312079	7.729032	8.938441	8.023552
2016-01-15	8.332458	7.715825	8.941316	8.021913
2016-01-16	8.126265	7.499723	8.748125	7.817223
2016-01-17	8.490562	7.900544	9.176804	9.273878
2016-01-18	8.793215	8.184117	9.409908	10.333775
2016-01-19	8.592981	8.004455	9.226525	9.125871

Cột y là giá trị thực tế ta có từ dữ liệu ban đầu, các cột còn lại là giá trị dự đoán

Sau đó viết hàm tính lỗi (dựa theo hàm tính MAE và MAPE của Prophet):

```
def calculate_forecast_errors(df, prediction_size):
 df = df.copy()
 df['e'] = df['y'] - df['yhat']
 df['p'] = 100 * df['e'] / df['y']
 predicted_part = df[-prediction_size:]
 error_mean = lambda error_name:
np.mean(np.abs(predicted_part[error_name]))
 return {'MAPE': error_mean('p'), 'MAE': error_mean('e')}


for err_name, err_value in calculate_forecast_errors(cmp_df,
prediction_size).items():
print(err_name, err_value)
```

```
39
40 def calculate_forecast_errors(df, prediction_size):
41     df = df.copy()
42     df['e'] = df['y'] - df['yhat']
43     df['p'] = 100 * df['e'] / df['y']
44     predicted_part = df[-prediction_size:]
45     error_mean = lambda error_name: np.mean(np.abs(predicted_part[error_name]))
46     return {'MAPE': error_mean('p'), 'MAE': error_mean('e')}
47
48 for err_name, err_value in calculate_forecast_errors(cmp_df, prediction_size).items():
49     print(err_name, err_value)
50
```

Kết quả:

```
MAPE 3.748094209242616
MAE 0.3290916379744892 6
```

❖ **Thực hiện dự đoán thật:**

Những bước trên là để làm quen với việc sử dụng Prophet và dựa vào dữ liệu cắt bớt để xem chênh lệch và chỉ số lỗi. Tiếp theo ta sẽ thực hiện dự đoán thật cho 365 ngày tiếp theo kể từ ngày cuối của dataframe ban đầu là 19/9/2016.

Do mỗi object prophet chỉ fit 1 dataframe nên ta phải tạo lại object Prophet mới.

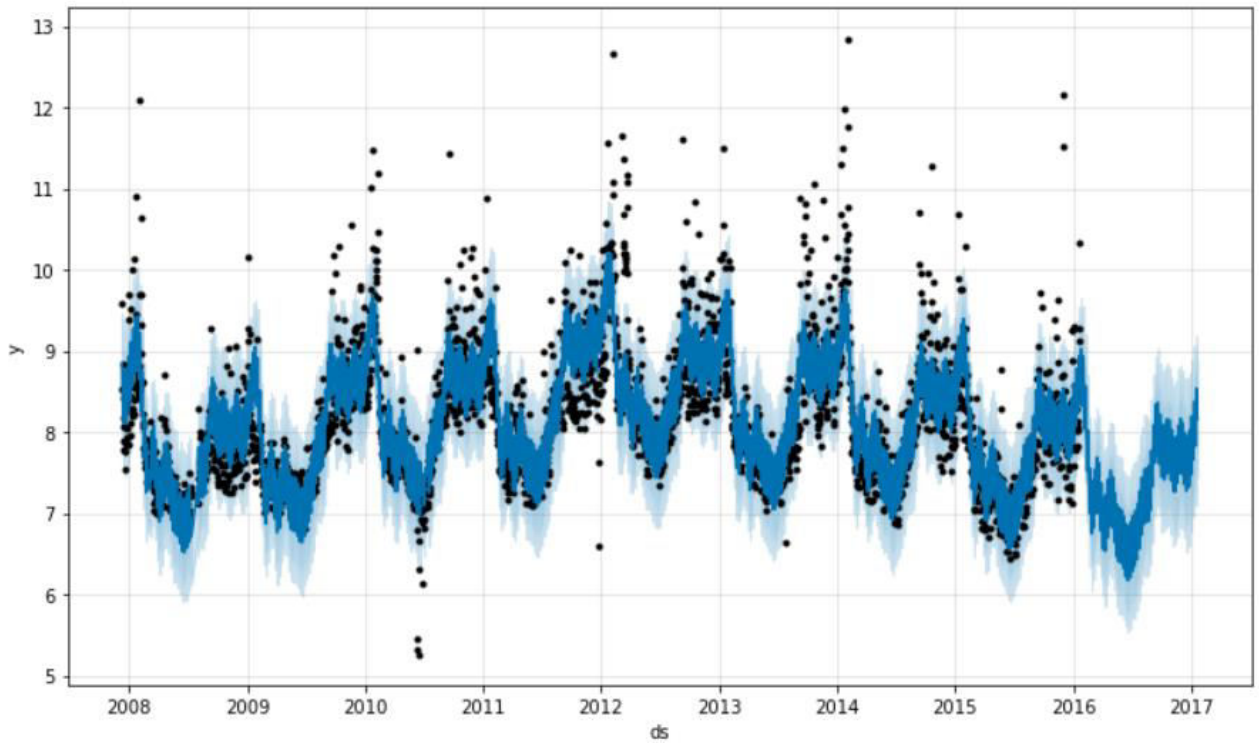
```
p2 = Prophet()
p2.fit(df)
future2 = p2.make_future_dataframe(periods=365, freq='D')
forecast2 = p2.predict(future2)
print(forecast2[['ds', 'yhat', 'yhat_lower', 'yhat_upper']])
p2.plot(forecast2)
p2.plot_components(forecast2)
```

Kết quả:

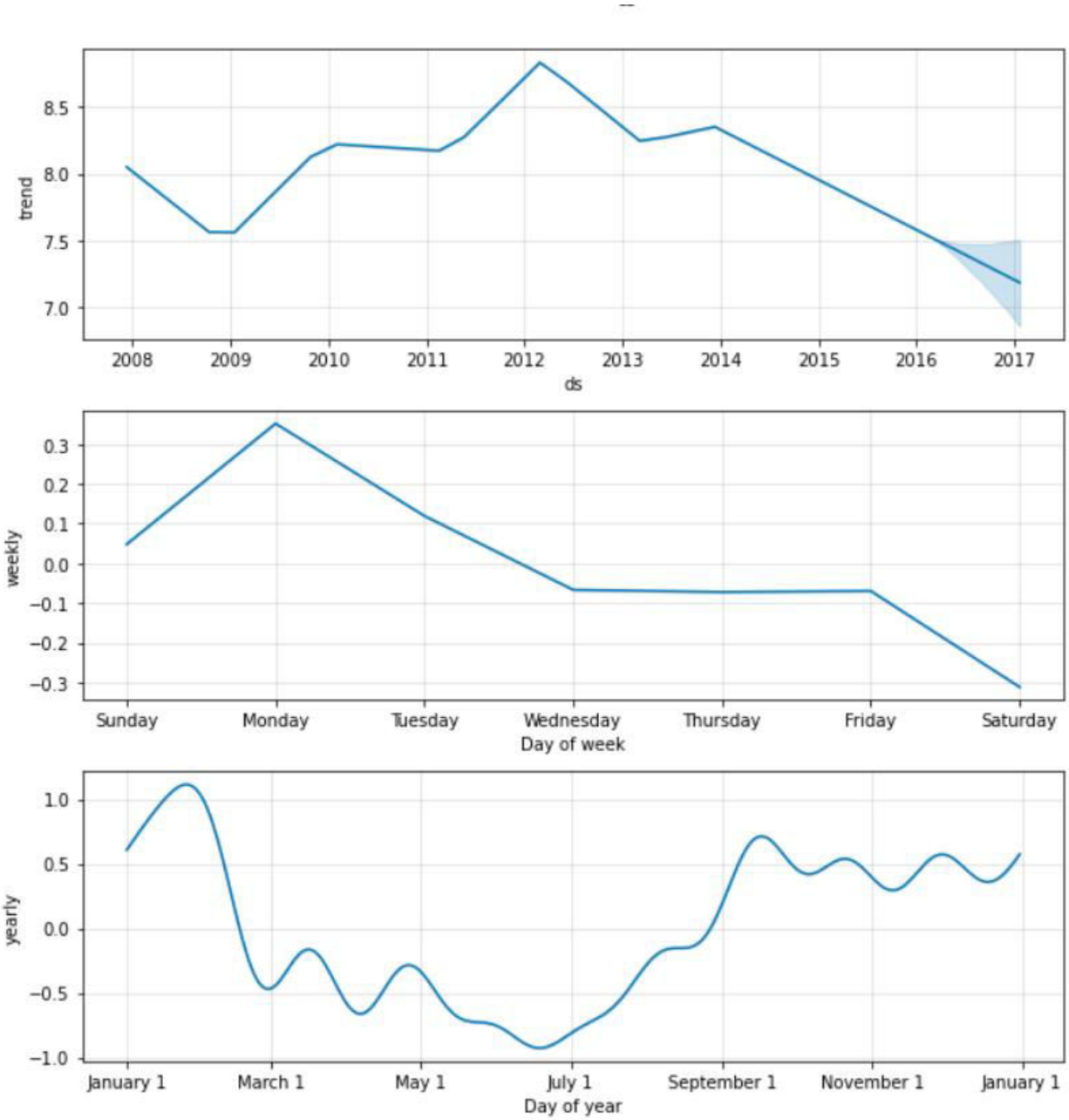
```
            ds      yhat  yhat_lower  yhat_upper
0   2007-12-10  8.852316    8.240698    9.465021
1   2007-12-11  8.600734    7.993005    9.281901
2   2007-12-12  8.396443    7.775475    8.993070
3   2007-12-13  8.374283    7.746394    9.006527
4   2007-12-14  8.362135    7.693413    8.964830
...        ...       ...         ...         ...
3265 2017-01-15  8.210039    7.482467    8.868440
3266 2017-01-16  8.535119    7.833482    9.186205
3267 2017-01-17  8.322532    7.606445    9.057181
3268 2017-01-18  8.155151    7.429920    8.872493
3269 2017-01-19  8.167128    7.435895    8.853421

[3270 rows x 4 columns]
```

Dataframe chứa kết quả dự đoán, ta có thể thấy các giá trị cuối là những ngày thuộc năm 2017



Plot tổng thể, đoạn được dự đoán từ đầu năm 2016-2017 không có các chấm đen



Plot thành phần, Prophet dự đoán 2017 trend sẽ đi xuống rõ rệt

Chương 4: Kết luận

Time-series Data và Time-series Forecasting có vai trò quan trọng trong việc quản lý và dự đoán thông tin trong nhiều lĩnh vực trong đời sống. Tùy theo đối tượng cần theo dõi và dự đoán mà có các mô hình dự đoán khác nhau.

Prophet là thư viện mã nguồn mở được phát triển bởi đội Core Data Science của Facebook. Prophet có nền tảng Stan và được phát triển trên 2 ngôn ngữ là Python và R.

Prophet có mô hình dự đoán dạng phân rã, với các thành phần chính là trend-cyclical, seasonality, holidays và chỉ số lỗi.

Prophet là công cụ dễ tiếp cận, cho phép người dùng tùy chỉnh một số tham số và thành phần để phù hợp hơn với đối tượng, hiện tượng mà họ dự đoán. Ngoài ra các hàm hỗ trợ tìm lỗi, tìm độ lệch, áp dụng thành phần holidays cũng giúp người dùng Prophet dễ dàng hơn trong việc theo dõi, nghiên cứu và phát triển.

Chương 5: Một số thuật ngữ trong báo cáo

- Time-series Data: trang 5
- Time-series Analysis và Time-series Forecasting: trang 8
- Mô hình phân rã – Decomposable: trang 8
- Trend: trang 10
- Cyclical: trang 10
- Seasonality: trang 11
- Irregular: trang 12
- Linear regression: https://en.wikipedia.org/wiki/Linear_regression
- Logistics regression: https://en.wikipedia.org/wiki/Logistic_regression
- MAE: https://en.wikipedia.org/wiki/Mean_absolute_error
- MAPE: https://en.wikipedia.org/wiki/Mean_absolute_percentage_error
- Nội suy – Interpolation: https://en.wikipedia.org/wiki/Interpolation
- Ngoại suy – Extrapolation: https://en.wikipedia.org/wiki/Extrapolation

Chương 6: Tài liệu tham khảo

[1] Forecasting at scale – Sean J. Taylor, Benjamin Letham: https://peerj.com/preprints/3190/

[2] Hỗ trợ Prophet trên ngôn ngữ R (Python không có trang tra cứu tương ứng): https://rdrr.io/cran/prophet/src/R/prophet.R

[3] Trang chủ của Facebook Prophet: https://facebook.github.io/

[4] Forecasting: Principle and Practice – Rob J. Hyndman, George Athanasopoulos: https://otexts.com/fpp2/

[5] https://stackoverflow.com/

[6] https://www.xenonstack.com/blog/time-series-deep-learning/

[7] Time Series for Beginners – Jack Esprabens, Ari Arango, Joshua Kim: https://bookdown.org/JakeEsprabens/431-Time-Series/

[8] https://en.wikipedia.org

==========HẾT==========