Phương pháp Naive24

Cuộc thi Dự đoán lưu lượng server - AIviVN

KhaPham

Ngày 10 tháng 06 năm 2019

Mục lục

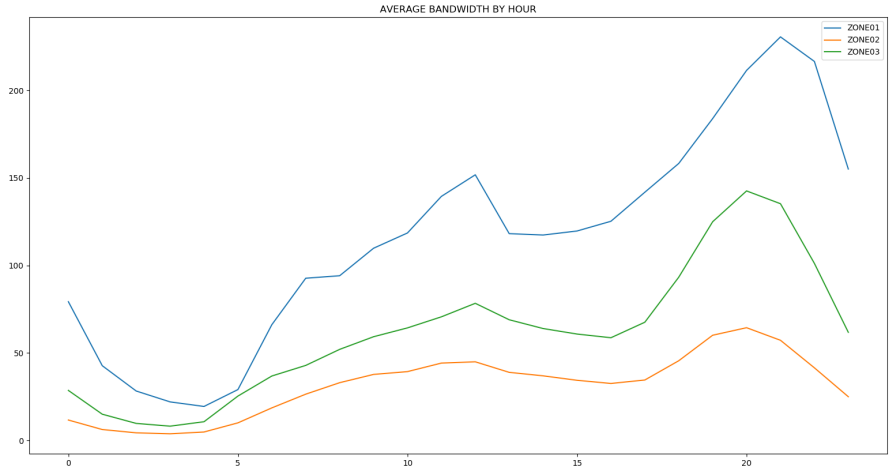
1 Thống kê và xử lý dữ liệu

Dữ liệu được cung cấp là số liệu về lưu lượng và số lượng người dùng tối đa của 501 server theo giờ từ ngày 01/10/2017 đến ngày 09/03/2018. Nếu viết dưới dạng time-series thì ta sẽ có 501 time-series, trong đó đa số các time-series có độ dài khoảng 12500. Tại một số thời điểm (các thời điểm này khác nhau tuỳ theo server), dữ liệu bị mất.

Việc làm đầu tiên của mình đó là định dạng lại tập dữ liệu theo dạng time-series như hình dưới đây.

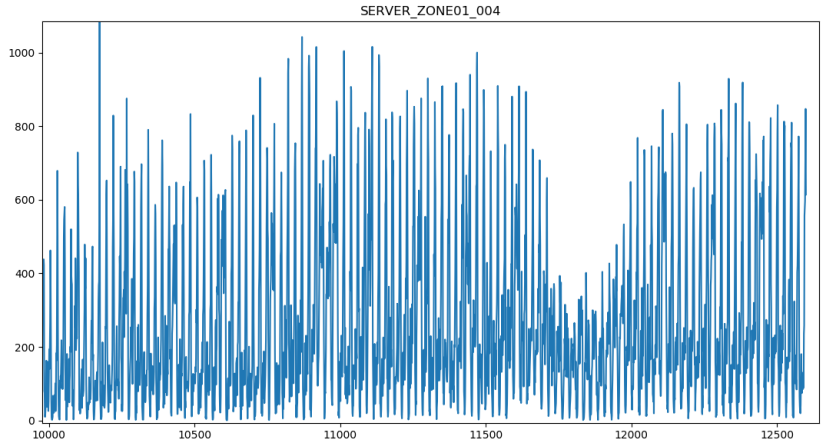
SERVER_NAME	ZONE	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SERVER_ZONE01_001	1	26.90693	13.08857	12.36472	9.641088	6.984784	18.14098	108.2638	97.9313	22.00865	17.02772	14.23683	24.03253	37.54782	53.7255	54.83936	25.97164
SERVER_ZONE01_002	1	61.48169	1.145404	3.607371	9.024135	1.198343	65.46652	63.83677	37.7813	38.25543	43.83543	86.40275	74.98993	51.77588	126.9781	39.30924	103.2022
SERVER_ZONE01_003	1	20.60984	41.33793	35.40337	6.74104	69.70008	34.75453	81.56033	64.45312	68.99272	96.00297	133.0648	56.7671	64.79722	54.80458	84.77855	92.55001
SERVER_ZONE01_004	1	97.73738	36.99003	12.281	1.81173	9.09204	4.742476	100.2506	89.67275	82.67576	244.5349	181.8659	237.6051	325.9315	304.1006	351.7008	142.6894
SERVER_ZONE01_005	1	113.2507	53.74656	60.59822	63.23632	36.40085	18.04801	48.4294	164.6443	256.7981	263.9964	222.4198	219.1628	174.5732	245.2007	326.4098	249.3765
SERVER_ZONE01_006	1	112.607	55.96841	39.79051	55.35848	11.68184	62.83245	256.6116	174.2461	218.0088	276.3045	363.4749	622.3792	400.2153	275.6483	333.1957	363.3452
SERVER_ZONE01_009	1																
SERVER_ZONE01_010	1																
SERVER_ZONE01_011	1																
SERVER_ZONE01_017	1	50.02676	34.41626	6.038094	16.84769	2.328055	16.66725	37.79165	148.1004	189.8257	310.245	331.0916	370.9644	546.9965	378.511	447.8067	276.2004

Sau đó, mình vẽ một biểu đồ đơn giản về lưu lượng trung bình theo từng giờ của các server trong từng khu vực.



Quan sát biểu đồ này, ta nhận thấy rằng từ 1 giờ đến 5 giờ sáng là khoảng thời gian có ít lượt truy cập vào server nhất. Nhận xét này sẽ giúp ích cho ta trong việc xây dựng thuật toán sau này.

Tiếp theo, ta quan sát đồ thị của từng time-series. Nếu quan sát đủ nhiều, ta sẽ nhận thấy rằng nhiều server có hiện tượng sụt giảm lưu lượng đáng kể từ điểm thời gian thứ 11600 đến 12000, chẳng hạn như đồ thị dưới đây.



Rõ ràng khoảng thời gian này là một "ngoại lệ" (outlier) trong các time-series và nên bị loại bỏ. Cách loại bỏ của mình đó là cắt khoảng thời gian này (chú ý rằng số lượng điểm thời gian bị cắt phải chia hết cho 24) và ghép khoảng thời gian sau đó nối tiếp với khoảng thời gian trước.

Ngoài ra, trong mỗi time-series có một số thời điểm dữ liệu bị mất. Cách mình "điền vào chỗ trống" cũng rất đơn giản: nếu dữ liệu lúc 8 giờ của hôm nay bị mất thì ta sẽ điền vào đó dữ liệu lúc 8 giờ của ngày hôm qua. Nói cách khác, nếu điểm thời gian thứ i bị thiếu thì ta sẽ điền vào đó số liệu tại điểm thời gian thứ $i - 24$.

2 Thuật toán

2.1 Metric tính sai số

Metric tính sai số dùng trong cuộc thi là MAPE, xác định bởi

$$\text{MAPE}\left(y_{\text{true}}, y_{\text{pred}}\right) = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y^i_{\text{true}} - y^i_{\text{pred}}}{y^i_{\text{true}}}\right|,$$

trong đó y_{true} là chuỗi giá trị đúng, y_{pred} là chuỗi giá trị dự đoán; y^i_{true} và y^i_{pred} lần lượt là giá trị đúng và giá trị dự đoán tại thời điểm i; n là độ dài của y_{true} và y_{pred}.

Metric MAPE có lẽ không phải là metric tốt nhất cho bài toán này, vì nó có xu hướng "khuếch đại" sai số khi giá trị đúng là rất nhỏ (chẳng hạn nếu giá trị đúng là 0.1, giá trị dự đoán là 0.5 thì sai số lên đến 400%). Dựa vào biểu đồ về lưu lượng trung bình ở trên, ta có thể thấy rằng vào các thời điểm từ 1 đến 5 giờ sáng, lưu lượng của các server là rất nhỏ và có thể bằng 0. Hiện tượng này làm cho khoảng thời gian khuya và sáng sớm rất "nhạy cảm" với metric MAPE.

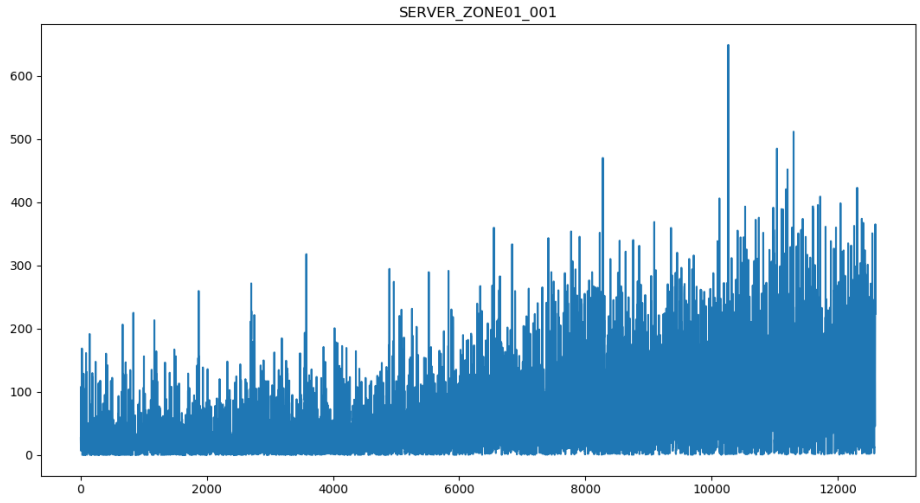
Theo mình, metric sMAPE xác định bởi

$$\text{sMAPE}\left(y_{\text{true}}, y_{\text{pred}}\right) = \frac{2}{n}\sum_{i=1}^{n}\frac{\left|y^i_{\text{true}} - y^i_{\text{pred}}\right|}{|y^i_{\text{true}}| + |y^i_{\text{pred}}|}$$

sẽ phù hợp hơn trong bài toán này.

2.2 Tập huấn luyện (Training set)

Dựa vào đồ thị của các time-series, mình nhận thấy rằng khoảng 2000 giá trị cuối của series có nét tương đồng, theo nghĩa xu hướng (trend) trong khoảng thời gian này là không thật rõ ràng.



Do đó, với mỗi time-series, mình chọn tập huấn luyện là 2000 giá trị cuối của time-series đó (tức là dùng khoảng 3 tháng cuối để dự đoán 1 tháng tiếp theo) để dự đoán các giá

trị về lưu lượng. Đối với các time-series về số lượng người dùng, mình chọn 1000 giá trị cuối cùng.

2.3 Tập xác minh (Validation set)

Khoảng thời gian cần dự đoán là 31 ngày, tương ứng với 744 giờ, do đó mình chọn tập xác minh có độ dài tương tự.

2.4 Các thuật toán cơ bản

2.4.1 Thuật toán Naive24Min

Đây là thuật toán mình dùng làm baseline. Sở dĩ mình nghĩ tới thuật toán này là vì tính nhạy cảm của metric MAPE đối với các giá trị nhỏ như đã nói ở trên.

Một cách ngắn gọn, thuật toán này có thể được giải thích như sau: dự đoán của giờ thứ i sẽ là giá trị nhỏ nhất của các giá trị của giờ thứ i trong tập huấn luyện.

Đoạn code cho thuật toán này cũng rất đơn giản (chú ý seasonal ở đây được chọn là 24).

```
def forward_min(self, data, predict_len):
    min_list = [0] * self.seasonal
    data_len = len(data)
    for remainder in range(self.seasonal):
    min_list[remainder] = np.min([data[i] for i in range(data_len) if
                                  int(i % self.seasonal) == remainder])

    prediction = [0] * predict_len
    for i in range(predict_len):
    remainder_cur = int((data_len + i) % self.seasonal)
    prediction[i] = min_list[remainder_cur]

    return prediction
```

2.4.2 Thuật toán Naive24Mean

Tương tự thuật toán Naive24Min, thuật toán Naive24Mean cũng rất đơn giản: dự đoán của giờ thứ i sẽ là trung bình của các giá trị của giờ thứ i trong tập huấn luyện.

Tuy nhiên, không phải giờ nào mình cũng sử dụng thuật toán này. Thuật toán Naive24Mean (với một số tinh chỉnh khi thay đổi giá trị trung bình thành giá trị quantile) sẽ được dùng cho một số giờ nhất định, các giờ còn lại mình dùng thuật toán Naive24Min (cũng với một số tinh chỉnh).

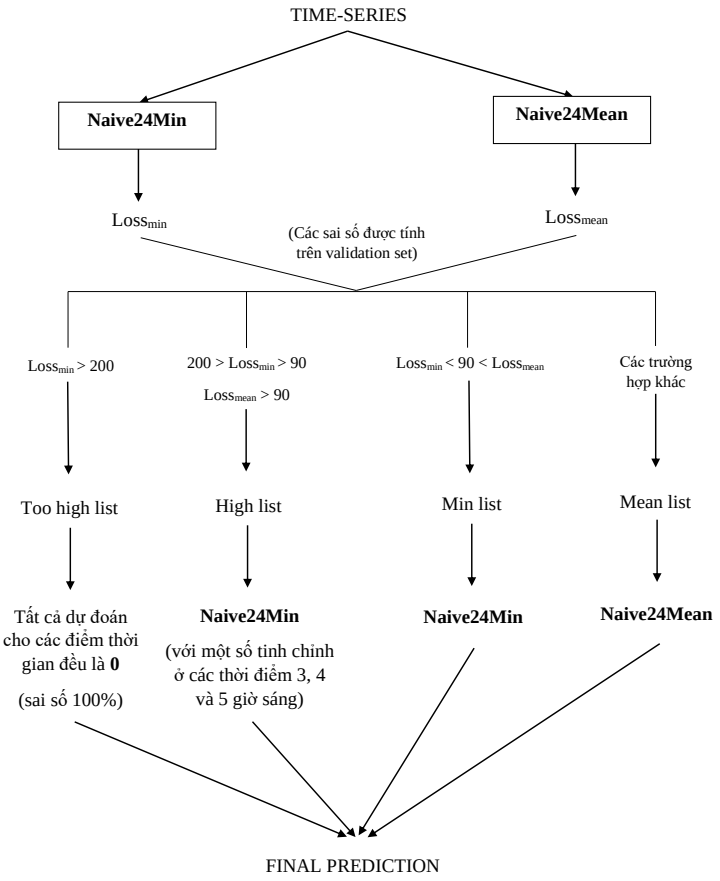
Đoạn code cụ thể như sau. Việc thêm nhiễu vào có thể giúp sai số giảm đi đôi chút.

```
def forward_mean(self, data, predict_len):
    min_list = [0] * self.seasonal
    data_len = len(data)
    remainder_data = self.seasonal - data_len % self.seasonal
    convert_list = [0] * self.seasonal
    for i in range(self.seasonal):
        convert_list[i] = (i - remainder_data) % self.seasonal
    for remainder in range(self.seasonal):
        if remainder in [convert_list[i] for i in [15, 16, 17, 18, 19, 20,
                                                   21, 22, 6, 7, 8, 9, 10]]:
            noise = np.random.normal(0.9, 0.1)
            data_cur = [data[i] for i in range(data_len) if
                        int(i % self.seasonal) == remainder] * noise
            min_list[remainder] = np.quantile(data_cur, 0.45) * noise
        else:
            noise = np.random.normal(1.2, 0.1)
            data_cur = [data[i] for i in range(data_len) if
                        int(i % self.seasonal) == remainder] * noise
            min_list[remainder] = np.quantile(data_cur, 0.05)

    prediction = [0] * predict_len
    for i in range(predict_len):
        remainder_cur = int((data_len + i) % self.seasonal)
```

```
            prediction[i] = min_list[remainder_cur]

return prediction
```

2.5 Thuật toán hoàn chỉnh

Sơ đồ dưới đây mô tả thuật toán hoàn chỉnh để dự đoán lưu lượng. Sơ đồ thuật toán dự đoán người dùng cũng tương tự và có phần đơn giản hơn.



Sơ đồ thuật toán hoàn chỉnh cho bài toán dự đoán lưu lượng

2.6 Một kỹ thuật nhỏ để tính trung bình

Kỹ thuật trung bình này mình lấy ý tưởng từ đáp án đoạt giải 3 trong cuộc thi M4 của đội đến từ ProLogistica Soft (các bạn quan tâm có thể xem thêm ở đây: https://github.com/M4Competition/M4-methods).

Giả sử mình có hai bài nộp với dự đoán, sai số lần lượt là $\text{pred}_1, \text{loss}_1$ và $\text{pred}_2, \text{loss}_2$. Khi đó dự đoán trung bình sẽ là

$$\text{pred}_{\text{average}} = \frac{e^{\frac{1}{\text{loss}_1}}}{e^{\frac{1}{\text{loss}_1}} + e^{\frac{1}{\text{loss}_2}}}\text{pred}_1 + \frac{e^{\frac{1}{\text{loss}_2}}}{e^{\frac{1}{\text{loss}_1}} + e^{\frac{1}{\text{loss}_2}}}\text{pred}_2.$$

Kỹ thuật trung bình này giúp sai số giảm đi khoảng $0.5 - 1\%$.

3 Kết quả

Với các thuật toán và kỹ thuật ở trên thì mình đạt sai số 53.5% trên validation set. Đối với public test set, sai số là 70.35%. Với private test set, sai số là 68.95%.