Is Rainfall Getting Heavier?

Building a Weather Forecasting Pipeline with Singapore Weather Station Data

By: Chin Hwee Ong (@ongchinhwee)

7 February 2021
FOSDEM Python devroom

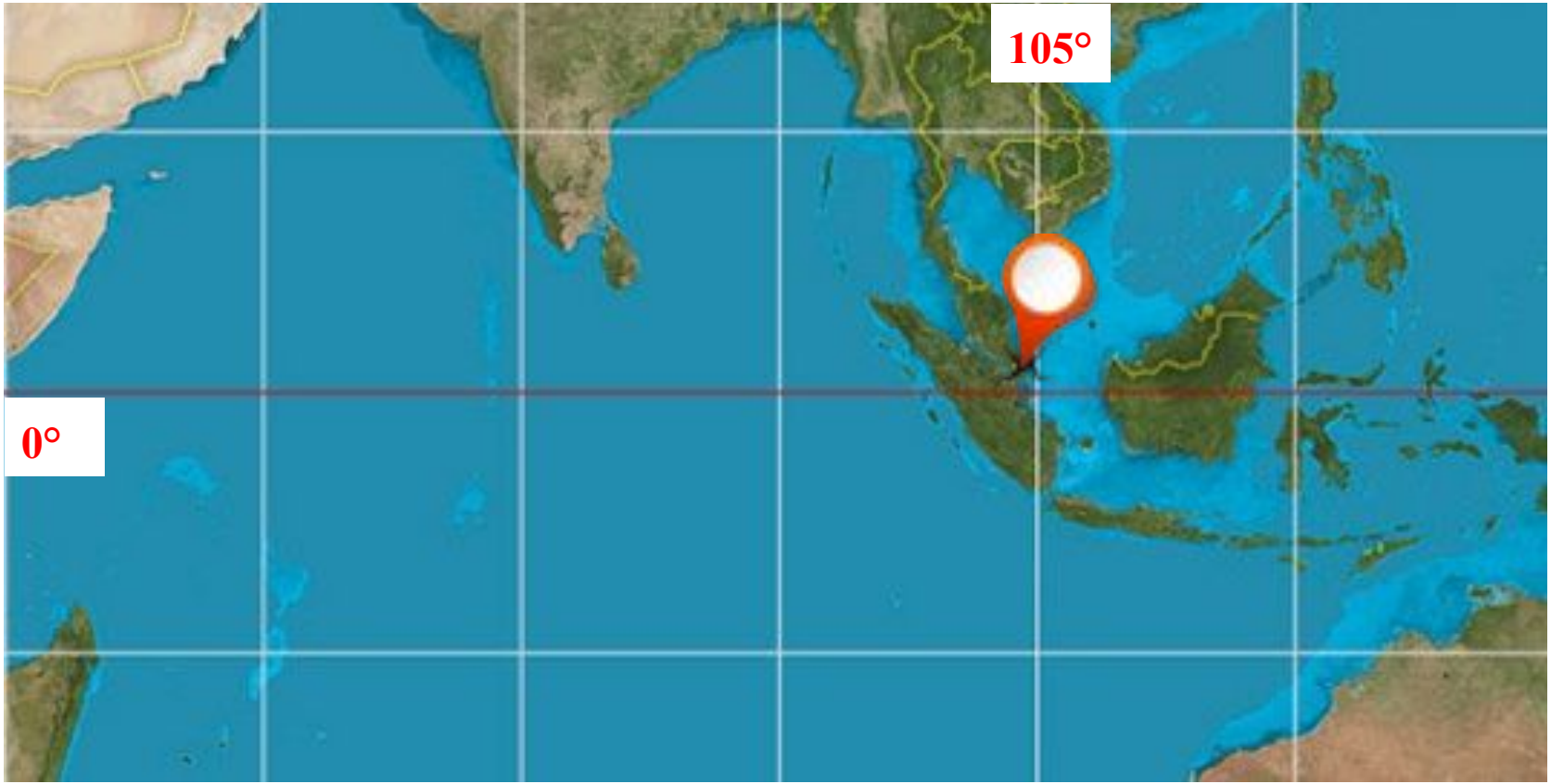
About me

Ong Chin Hwee **王敬惠**

- Data Engineer
- Based in **sunny Singapore** ☀️
- Aerospace Engineering + Computational Modelling
- Loves (and contributes to) pandas

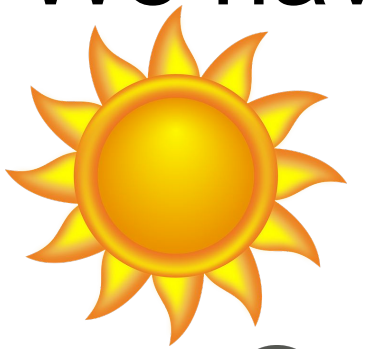
@ongchinhwee

Singapore 新加坡:
1°17'22.81"N, 103°
51'0.25"E
北纬1度，经纬103度



Singapore is a **tropical** country

We have our “four seasons”:



1. **Cold and Rainy**
2. Warm and Dry
3. **<u>Extremely Hot</u>**
4. **Hot and Stormy**

@ongchinhwee

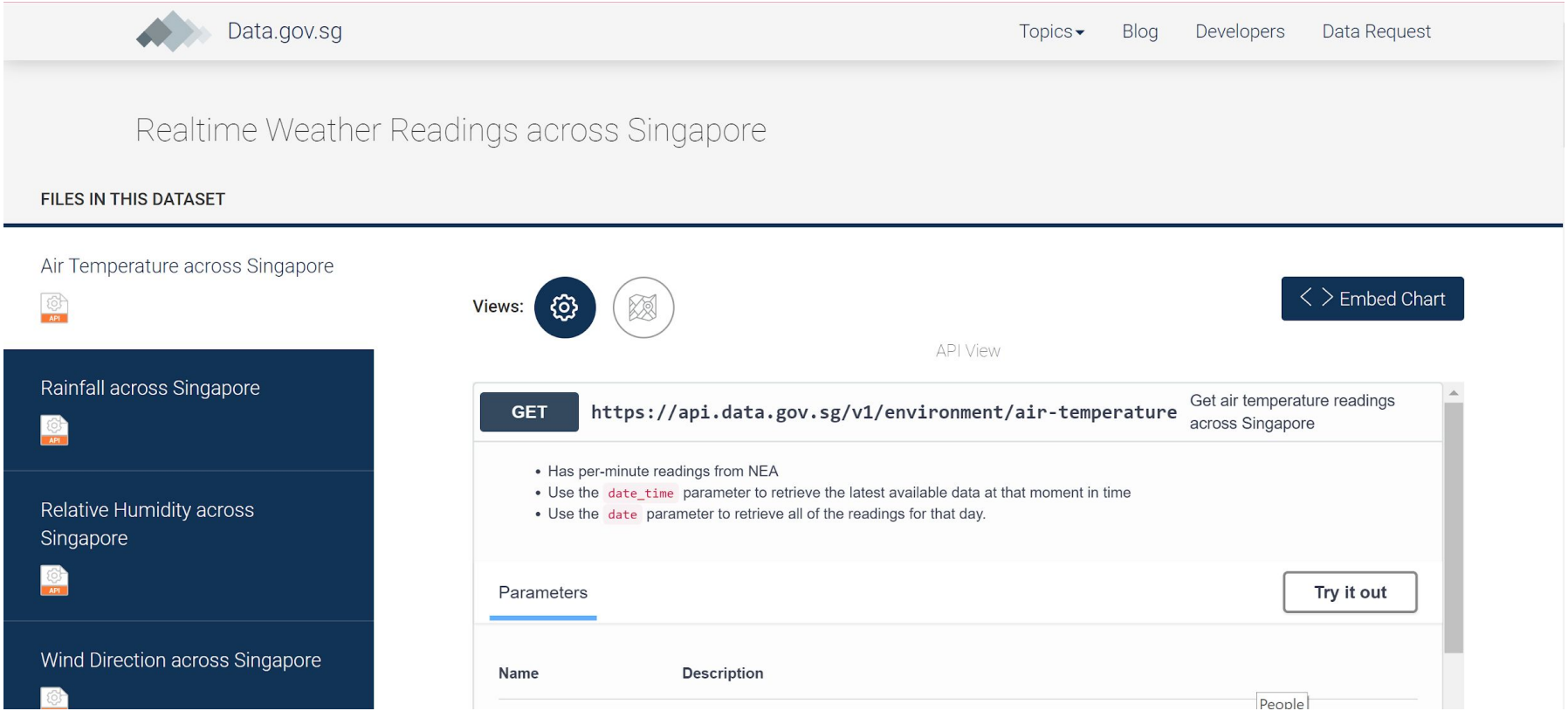
Since 2018, Singapore had **more than 20 flash floods**.

Majority of the floods were caused by **intense rain**.

Source: PUB Singapore
(https://www.pub.gov.sg/drainage/floodmanagement/recentflashfloods)

Could we **predict heavier rainfall** with weather data?

Extracting Weather Data



Data.gov.sg - Singapore's Open Data Portal

@ongchinhwee

Realtime Weather Readings across Singapore

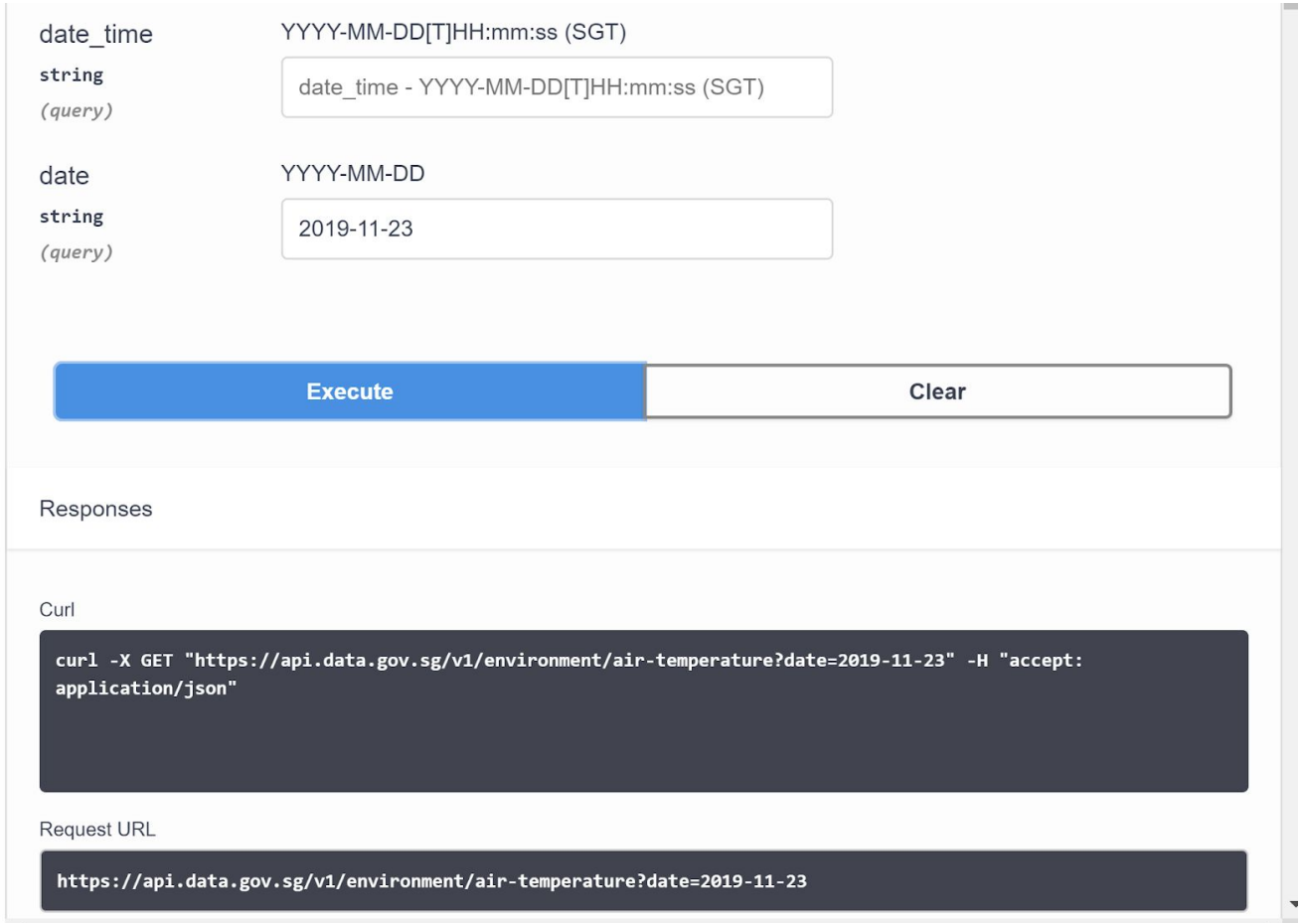
Real-time API on Data.gov.sg (Singapore's open data portal)

Open government data available under the Singapore Open Data License

(Almost) minute-by-minute weather station readings

“Let’s try to scrap weather data for a specific weather station!”

“How about we scrap multi-day data from the API?”



Responses

Curl

```
curl -X GET "https://api.data.gov.sg/v1/environment/air-temperature?date=2019-11-23" -H "accept: application/json"
```

Request URL

```
https://api.data.gov.sg/v1/environment/air-temperature?date=2019-11-23
```

```
        "id": "S100",
        "device_id": "S100",
        "name": "Woodlands Road",
        "location": {
          "latitude": 1.4172,
          "longitude": 103.74855
        }
      },
      {
        "id": "S115",
        "device_id": "S115",
        "name": "Tuas South Avenue 3",
        "location": {
          "latitude": 1.29377,
          "longitude": 103.61843
        }
      }
    ],
    "reading_type": "DBT 1M F",
    "reading_unit": "deg C"
  },
  "items": [
    {
      "timestamp": "2019-11-23T00:01:00+08:00",
      "readings": [
        {
          "station_id": "S109",
          "value": 25.2
        },
        {
          "station_id": "S117",
          "value": 26
        },
```

```
      "id": "S100",
      "device_id": "S100",
      "name": "Woodlands Road",
      "location": {
        "latitude": 1.4172,
        "longitude": 103.74855
      }
    },
    {
      "id": "S115",
      "device_id": "S115",
      "name": "Tuas South Avenue 3",
      "location": {
        "latitude": 1.29377,
        "longitude": 103.61843
      }
    }
  ],
  "reading_type": "DBT 1M F",
  "reading_unit": "deg C"
},
"items": [
  {
    "timestamp": "2019-11-23T00:01:00+08:00",
    "readings": [
      {
        "station_id": "S109",
        "value": 25.2
      },
      {
        "station_id": "S117",
        "value": 26
      },
```


Nested JSON format!

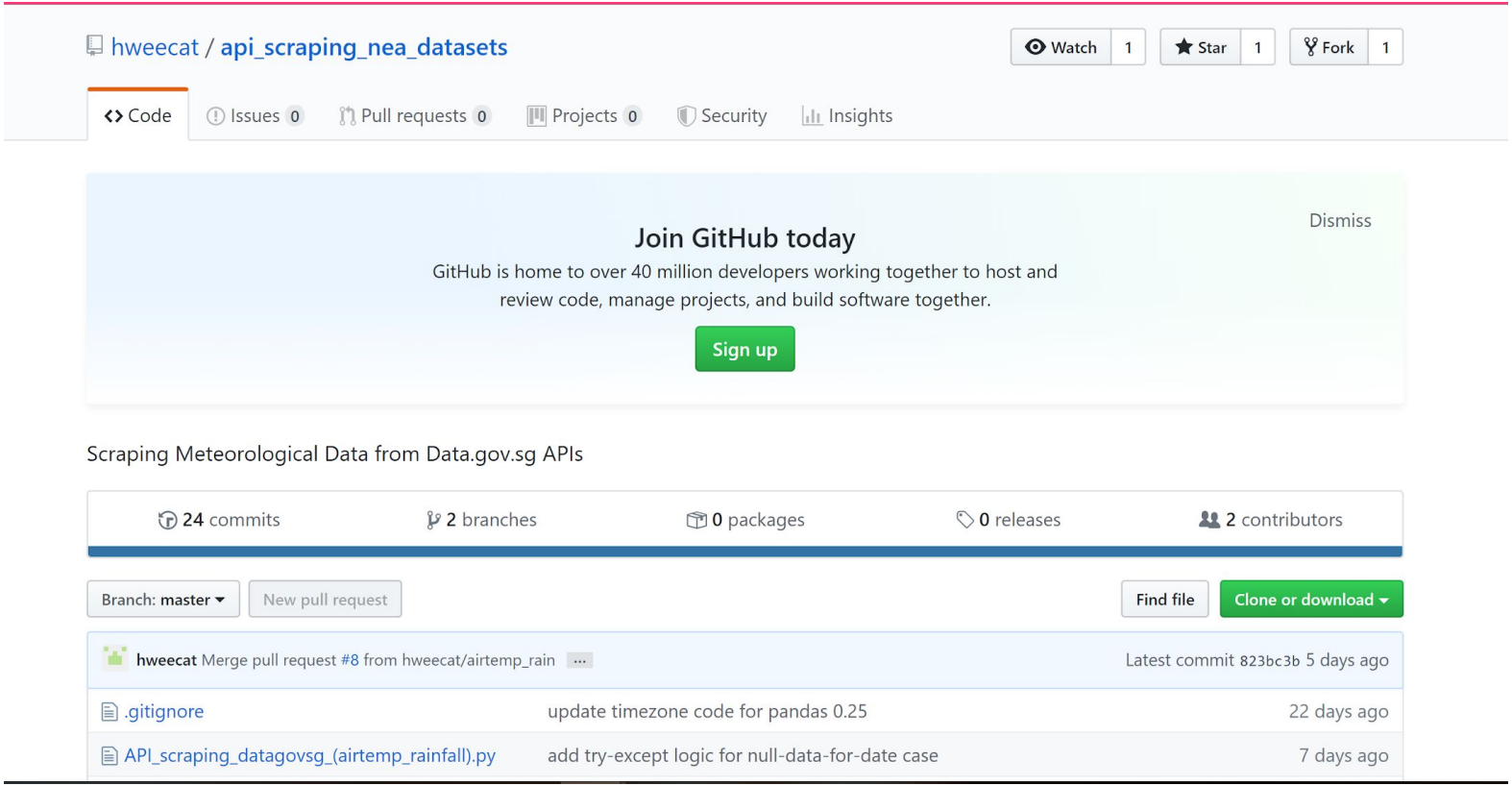
@ongchinhwee

```
        "id": "S100",
        "device_id": "S100",
        "name": "Woodlands Road",
        "location": {
          "latitude": 1.4172,
          "longitude": 103.74855
        }
      },
      {
        "id": "S115",
        "device_id": "S115",
        "name": "Tuas South Avenue 3",
        "location": {
          "latitude": 1.29377,
          "longitude": 103.61843
        }
      }
    ],
    "reading_type": "DBT 1M F",
    "reading_unit": "deg C"
  },
  "items": [
    {
      "timestamp": "2019-11-23T00:01:00+08:00",
      "readings": [
        {
          "station_id": "S109",
          "value": 25.2
        },
        {
          "station_id": "S117",
          "value": 26
        },
```


[ ]

	index	readings	timestamp
0	0	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:05:00+08:00
1	1	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:10:00+08:00
2	2	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:15:00+08:00
3	3	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-06-01 00:20:00+08:00
4	4	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:25:00+08:00
...	...	...	...
60435	277	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-12-31 23:35:00+08:00
60436	278	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:40:00+08:00
60437	279	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-12-31 23:45:00+08:00
60438	280	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:50:00+08:00
60439	281	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:55:00+08:00

60440 rows × 3 columns



"Scraping Meteorological Data from Data.gov.sg APIs" Project

@ongchinhwee

Data.gov.sg Weather Data API Scraping

Scraping weather data from APIs via **“Requests” library**

“Requests”:

Python library for humans to send HTTP requests

@ongchinhwee

Data.gov.sg Weather Data API Scraping

Currently supported Data.gov.sg APIs:

1. **Air Temperature (in °C)**
2. **Rainfall (in mm)**
3. **Relative Humidity**
4. **Wind Direction**
5. **Wind Speed**

Scrap data for **continuous time range** + **specific weather station**

@ongchinhwee

Design Considerations

Slow connection

- retry mechanism

```
from retrying import retry

@retry(wait_exponential_multiplier=1000, wait_exponential_max=10000)

def get_rainfall_data_from_date(date):
```

Design Considerations

Slow connection

API working but **no data for specific date**

Code	Details
200	Response body Download

```
{
  "metadata": {
    "stations": []
  },
  "items": [],
  "api_info": {
    "status": "healthy"
  }
}
```

@ongchinhwee

Design Considerations

Slow connection

API working but **no data for specific date**

- Return **empty DataFrame** with same column names as if there were data for specific date

Design Considerations

Slow connection

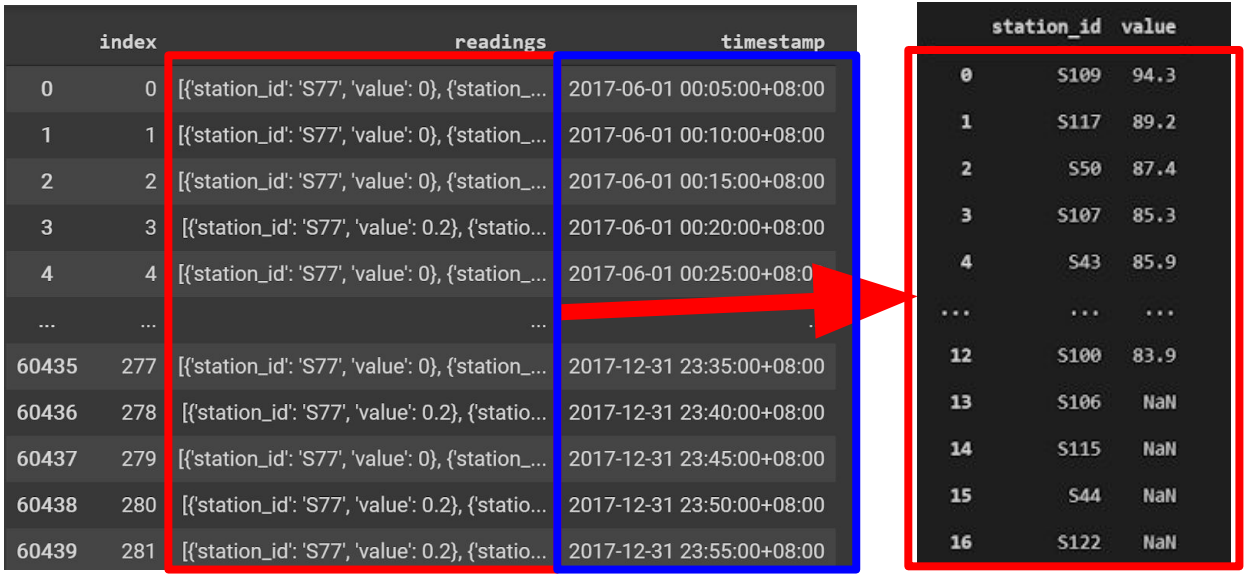
API working but **no data for specific date**

Nested JSON to pandas **DataFrame** conversion

- Extract desired **station** and **readings**
- Concatenate them back with **timestamp**

Design Considerations

Nested JSON to pandas **DataFrame** conversion



	index	readings	timestamp
0	0	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:05:00+08:00
1	1	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:10:00+08:00
2	2	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:15:00+08:00
3	3	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-06-01 00:20:00+08:00
4	4	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:25:00+08:0
...	...	...	.
60435	277	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-12-31 23:35:00+08:00
60436	278	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:40:00+08:00
60437	279	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-12-31 23:45:00+08:00
60438	280	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:50:00+08:00
60439	281	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:55:00+08:00

	station_id	value
0	S109	94.3
1	S117	89.2
2	S50	87.4
3	S107	85.3
4	S43	85.9
...	...	...
12	S100	83.9
13	S106	NaN
14	S115	NaN
15	S44	NaN
16	S122	NaN

@ongchinhwee

Design Considerations

Nested JSON to pandas **DataFrame** conversion

	index	readings	timestamp
0	0	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:05:00+08:00
1	1	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:10:00+08:00
2	2	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:15:00+08:00
3	3	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-06-01 00:20:00+08:00
4	4	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-06-01 00:25:00+08:00
...	...	...	...
60435	277	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-12-31 23:35:00+08:00
60436	278	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:40:00+08:00
60437	279	[{'station_id': 'S77', 'value': 0}, {'station_...	2017-12-31 23:45:00+08:00
60438	280	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:50:00+08:00
60439	281	[{'station_id': 'S77', 'value': 0.2}, {'statio...	2017-12-31 23:55:00+08:00

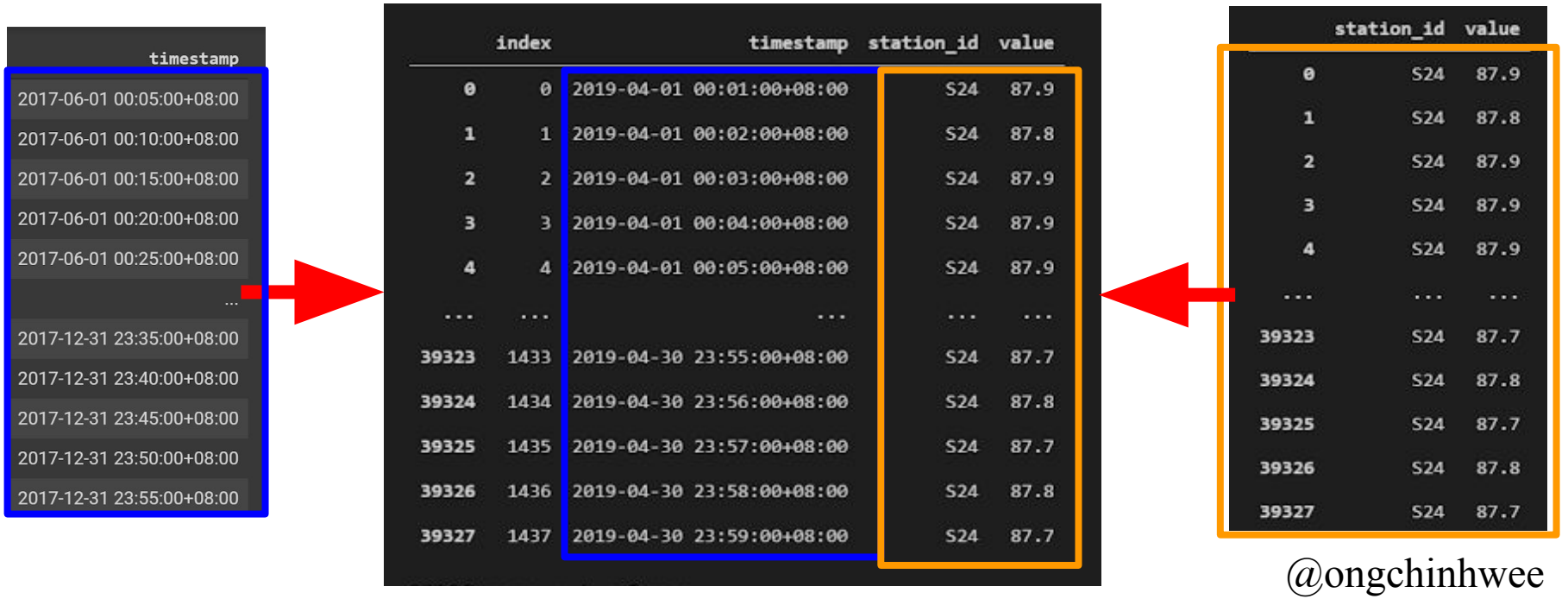
	station_id	value
0	S109	94.3
1	S117	89.2
2	S50	87.4
3	S107	85.3
4	S43	85.9
...	...	...
12	S100	83.9
13	S106	NaN
14	S115	NaN
15	S44	NaN
16	S122	NaN

	station_id	value
0	S24	87.9
1	S24	87.8
2	S24	87.9
3	S24	87.9
4	S24	87.9
...	...	...
39323	S24	87.7
39324	S24	87.8
39325	S24	87.7
39326	S24	87.8
39327	S24	87.7

@ongchinhwee

Design Considerations

Nested JSON to pandas **DataFrame** conversion



timestamp
2017-06-01 00:05:00+08:00
2017-06-01 00:10:00+08:00
2017-06-01 00:15:00+08:00
2017-06-01 00:20:00+08:00
2017-06-01 00:25:00+08:00
...
2017-12-31 23:35:00+08:00
2017-12-31 23:40:00+08:00
2017-12-31 23:45:00+08:00
2017-12-31 23:50:00+08:00
2017-12-31 23:55:00+08:00

	index	timestamp	station_id	value
0	0	2019-04-01 00:01:00+08:00	S24	87.9
1	1	2019-04-01 00:02:00+08:00	S24	87.8
2	2	2019-04-01 00:03:00+08:00	S24	87.9
3	3	2019-04-01 00:04:00+08:00	S24	87.9
4	4	2019-04-01 00:05:00+08:00	S24	87.9
...	...	...	...	...
39323	1433	2019-04-30 23:55:00+08:00	S24	87.7
39324	1434	2019-04-30 23:56:00+08:00	S24	87.8
39325	1435	2019-04-30 23:57:00+08:00	S24	87.7
39326	1436	2019-04-30 23:58:00+08:00	S24	87.8
39327	1437	2019-04-30 23:59:00+08:00	S24	87.7

	station_id	value
0	S24	87.9
1	S24	87.8
2	S24	87.9
3	S24	87.9
4	S24	87.9
...	...	...
39323	S24	87.7
39324	S24	87.8
39325	S24	87.7
39326	S24	87.8
39327	S24	87.7

@ongchinhwee

Singapore Rainfall Data:
A 4-Year Time Series Analysis

@ongchinhwee

Time Series Analysis of Singapore Rainfall Data

Selected **weather station**:

Changi Weather Station **(ID: S24)**

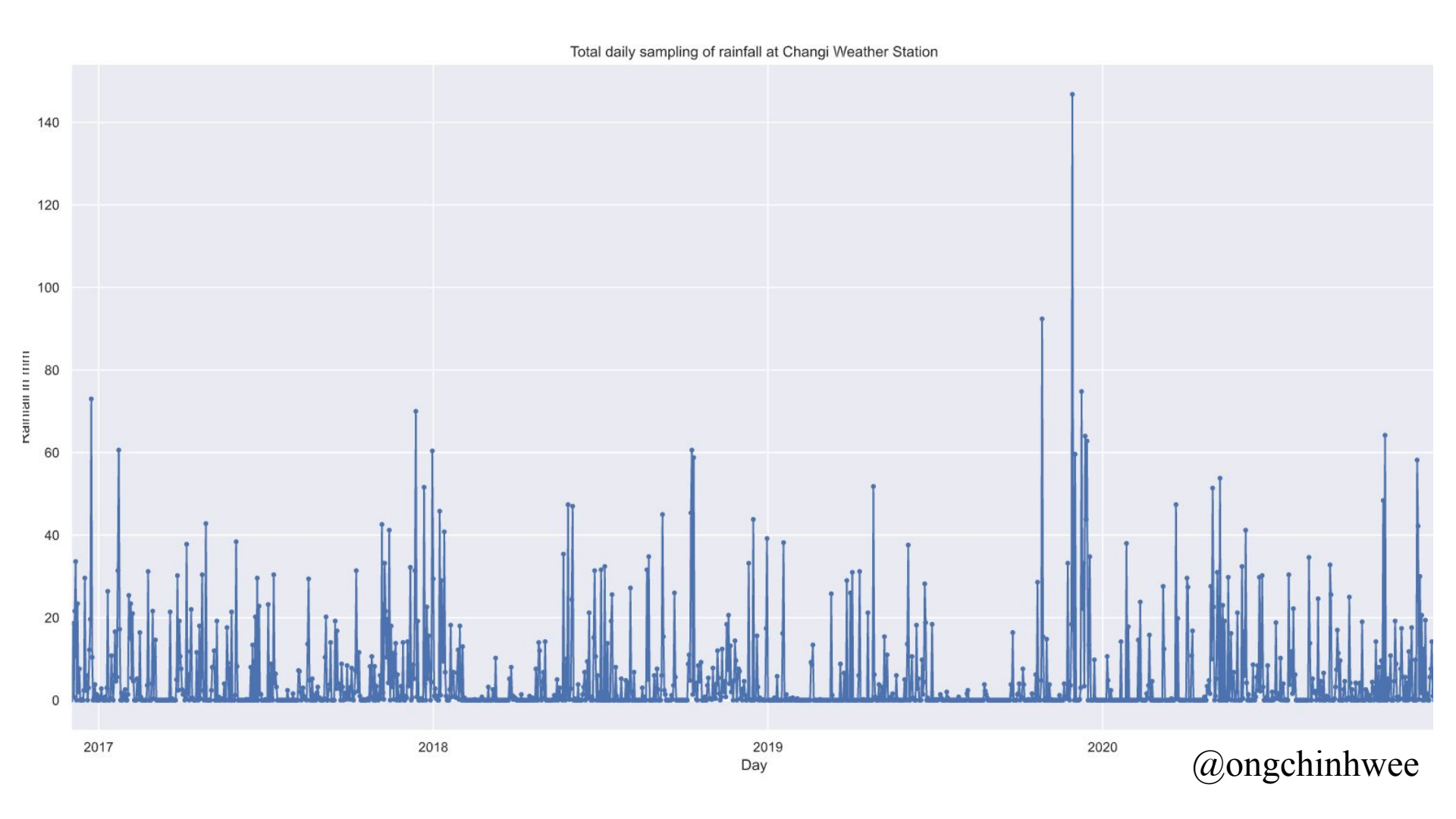
Analysis **timeframe**:

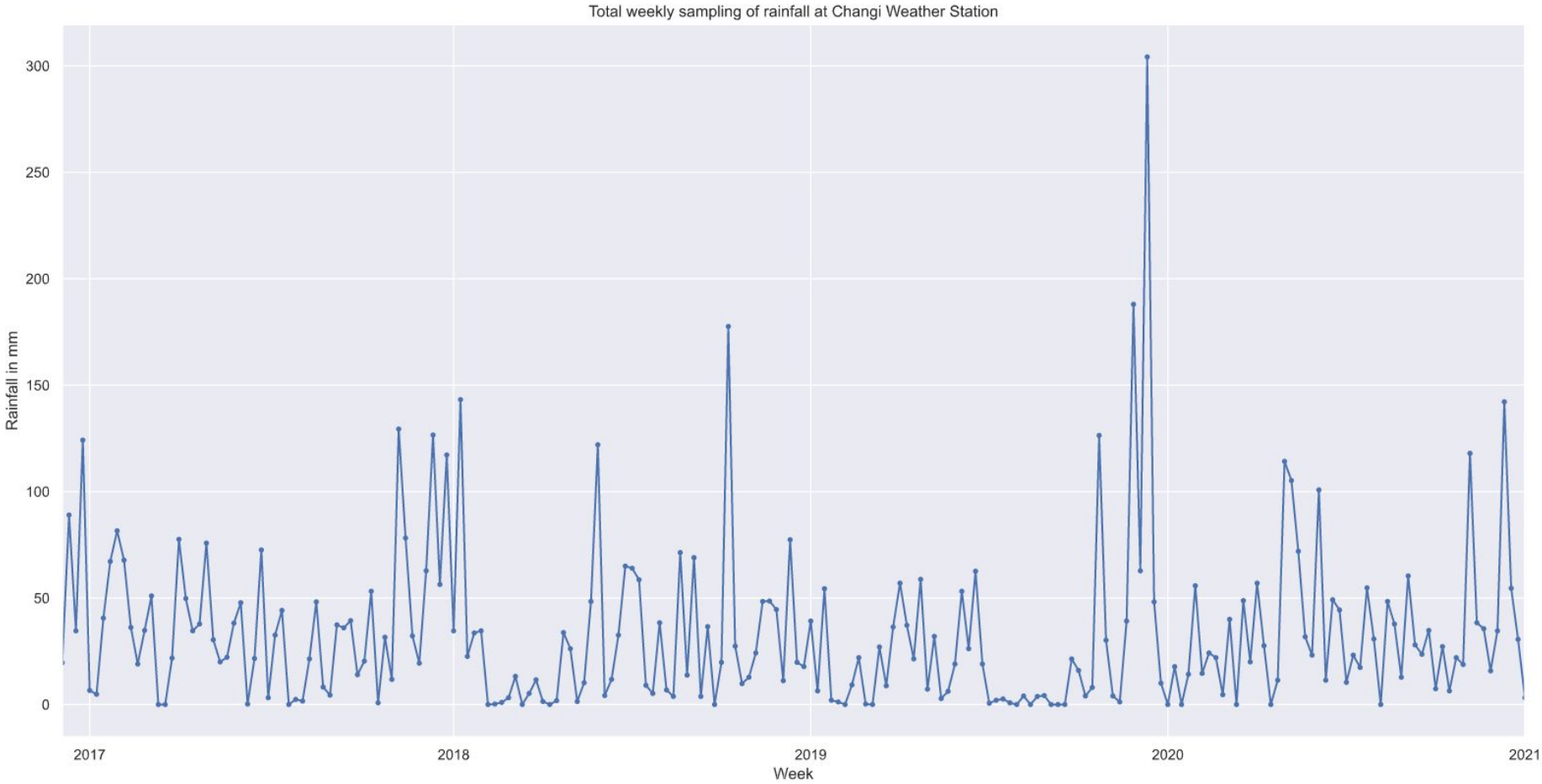
2 Dec 2016 to 31 Dec 2020 **(~4 years)**

Objective:

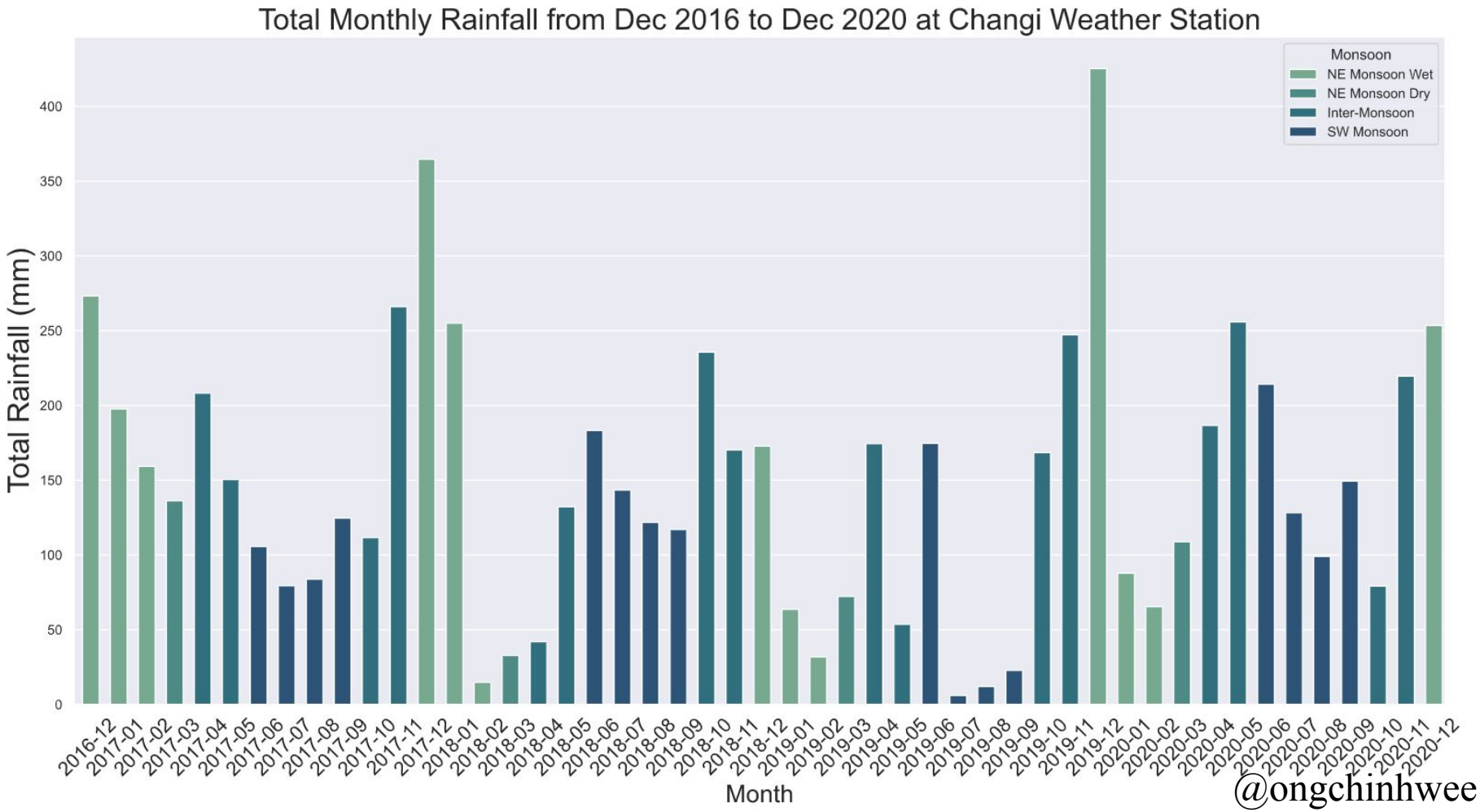
- Extract **trend** and **seasonality** from 5-minute rainfall data

@ongchinhwee

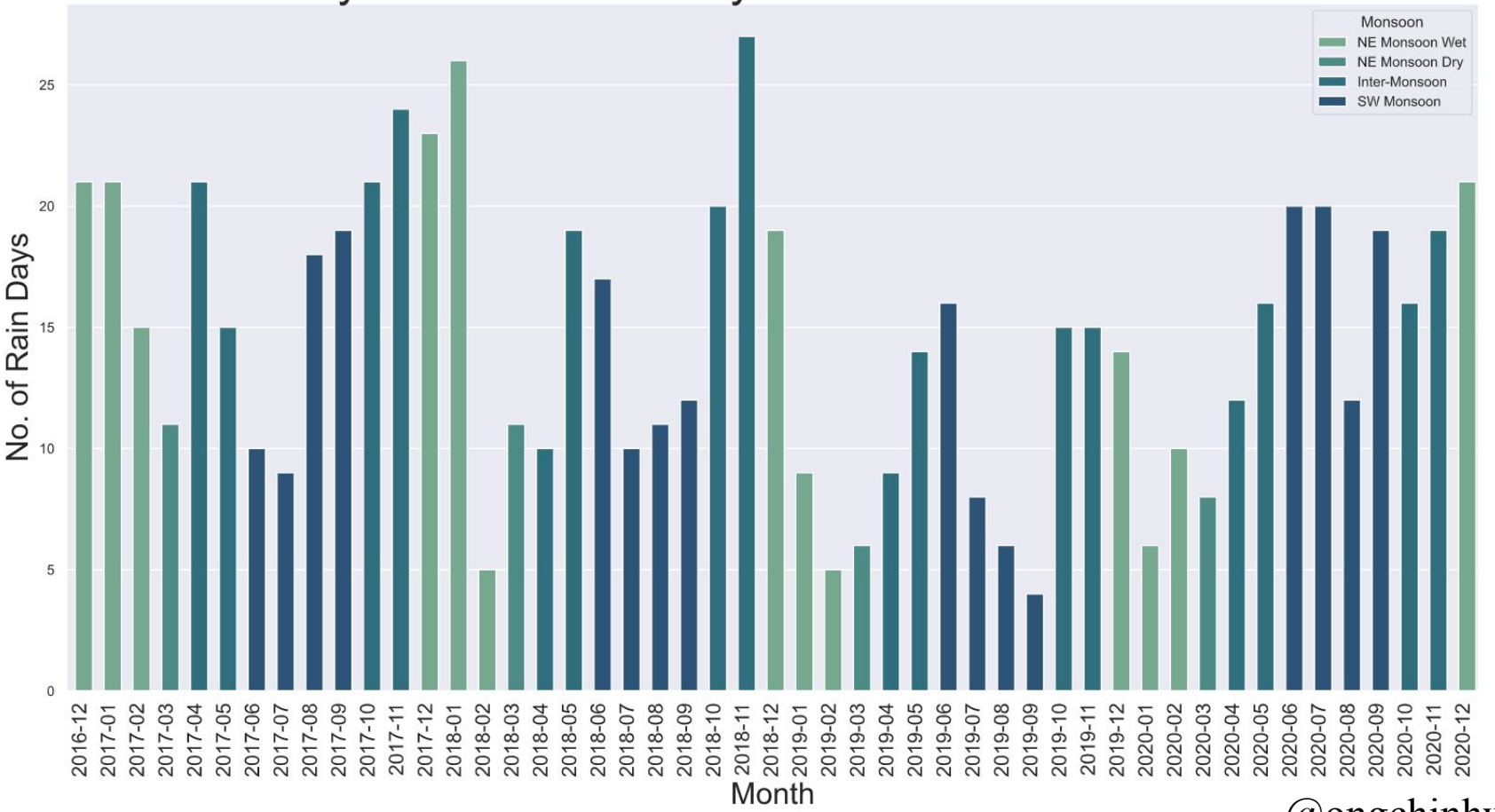

Total daily sampling of rainfall at Changi Weather Station
140
120
100
80
60
40
20
0
2017
2018
2019
2020
Day
@ongchinhwee



@ongchinhwee


Total Monthly Rainfall from Dec 2016 to Dec 2020 at Changi Weather Station
Monsoon
NE Monsoon Wet
NE Monsoon Dry
Inter-Monsoon
SW Monsoon
Total Rainfall (mm)
0
50
100
150
200
250
300
350
400
Month
2016-12
2017-01
2017-02
2017-03
2017-04
2017-05
2017-06
2017-07
2017-08
2017-09
2017-10
2017-11
2017-12
2018-01
2018-02
2018-03
2018-04
2018-05
2018-06
2018-07
2018-08
2018-09
2018-10
2018-11
2018-12
2019-01
2019-02
2019-03
2019-04
2019-05
2019-06
2019-07
2019-08
2019-09
2019-10
2019-11
2019-12
2020-01
2020-02
2020-03
2020-04
2020-05
2020-06
2020-07
2020-08
2020-09
2020-10
2020-11
2020-12
@ongchinhwee

Monthly number of rain days from Dec 2016 to Dec 2020



Time Series Analysis for Forecasting

Analyse and forecast time series using **“statsmodels.tsa”**

“statsmodels” library:

Python library for statistical models, tests and exploration

“statsmodels.tsa”:

Model classes and functions for Time Series Analysis

Time Series Analysis for Forecasting

Stationarity: Stationary vs Non-Stationary

- Augmented Dickey-Fuller (ADF) Test

Patterns: Trend, Seasonality, Cycles (and Noise)

- Moving Averages
- STL Decomposition

Autocorrelation: Relationship between a time series and a lagged version of itself

Augmented Dickey-Fuller (ADF) Stationary Test

```
from statsmodels.tsa.stattools import adfuller
def ADF_test(timeseries):
    dftest = adfuller(timeseries.dropna(), autolag="AIC")
    print("Test statistic = {:.3f}".format(dftest[0]))
    print("P-value = {:.3f}".format(dftest[1]))
    print("Critical values :")
    for k, v in dftest[4].items():
        print(
f"\t{k}%: {v:.3f} - The data is {"not" if v < dftest[0] else ""} stationary
with {100 - int(k[:-1])}% confidence")
```

@ongchinhwee

Augmented Dickey-Fuller (ADF) Stationary Test

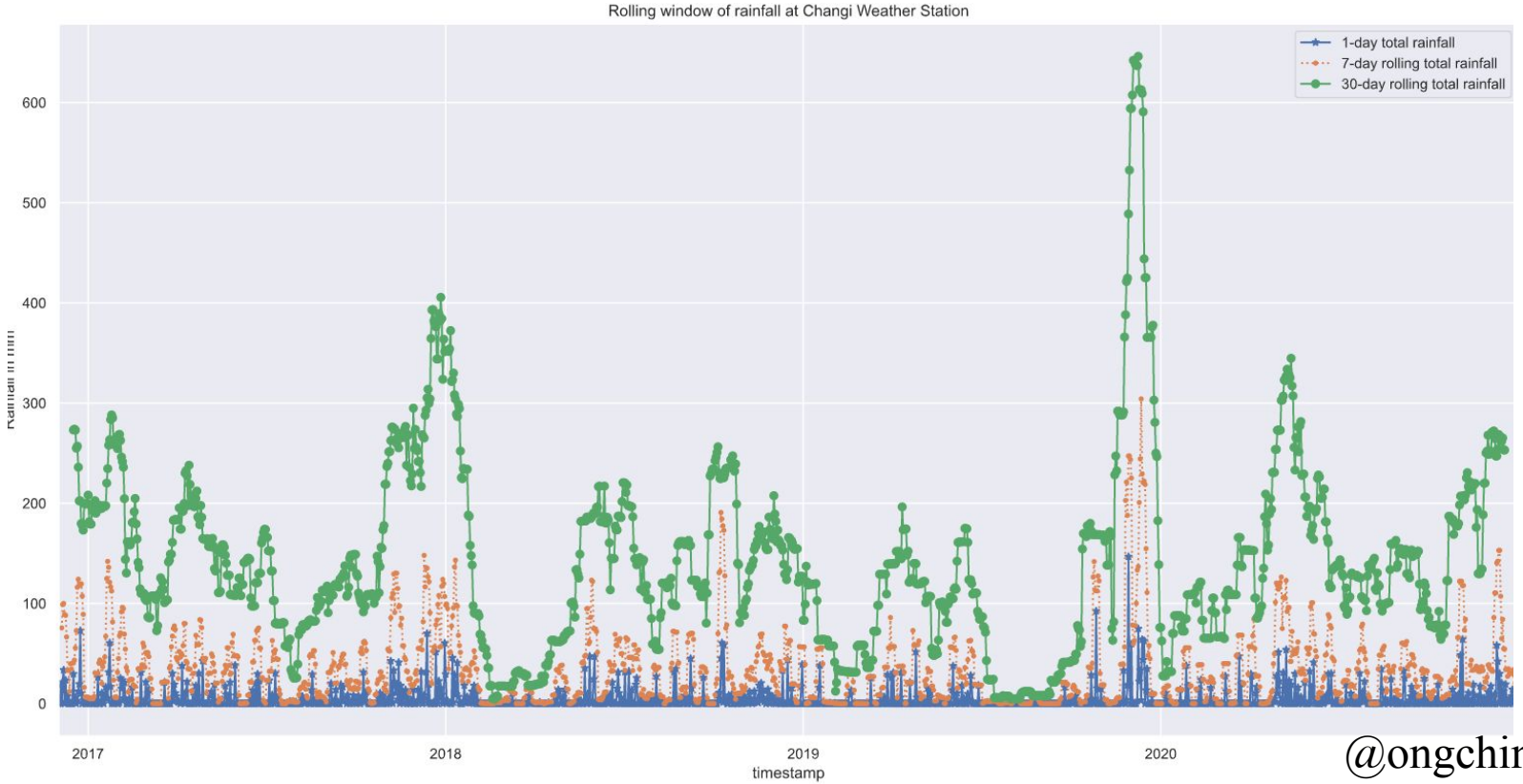
Total Daily Rainfall

```
Test statistic = -5.710
P-value = 0.000
Critical values :
    1%: -3.585 - The data is
stationary with 99% confidence
    5%: -2.928 - The data is
stationary with 95% confidence
    10%: -2.602 - The data is
stationary with 90% confidence
```

Monthly Daily Rainfall

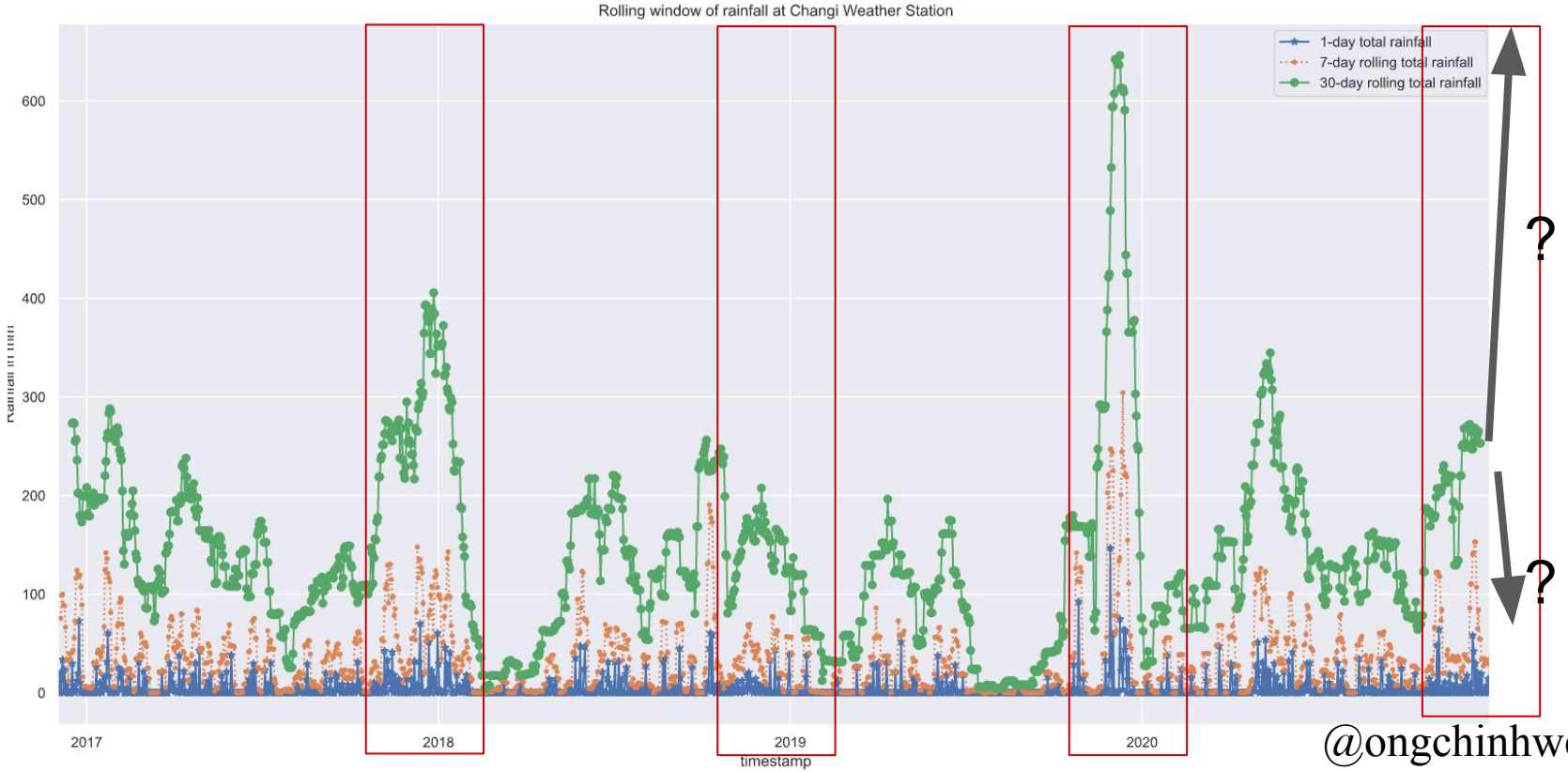
```
Test statistic = -13.590
P-value = 0.000
Critical values :
    1%: -3.435 - The data is
stationary with 99% confidence
    5%: -2.864 - The data is
stationary with 95% confidence
    10%: -2.256 - The data is
stationary with 90% confidence
```

Analyzing Rainfall with Moving Averages



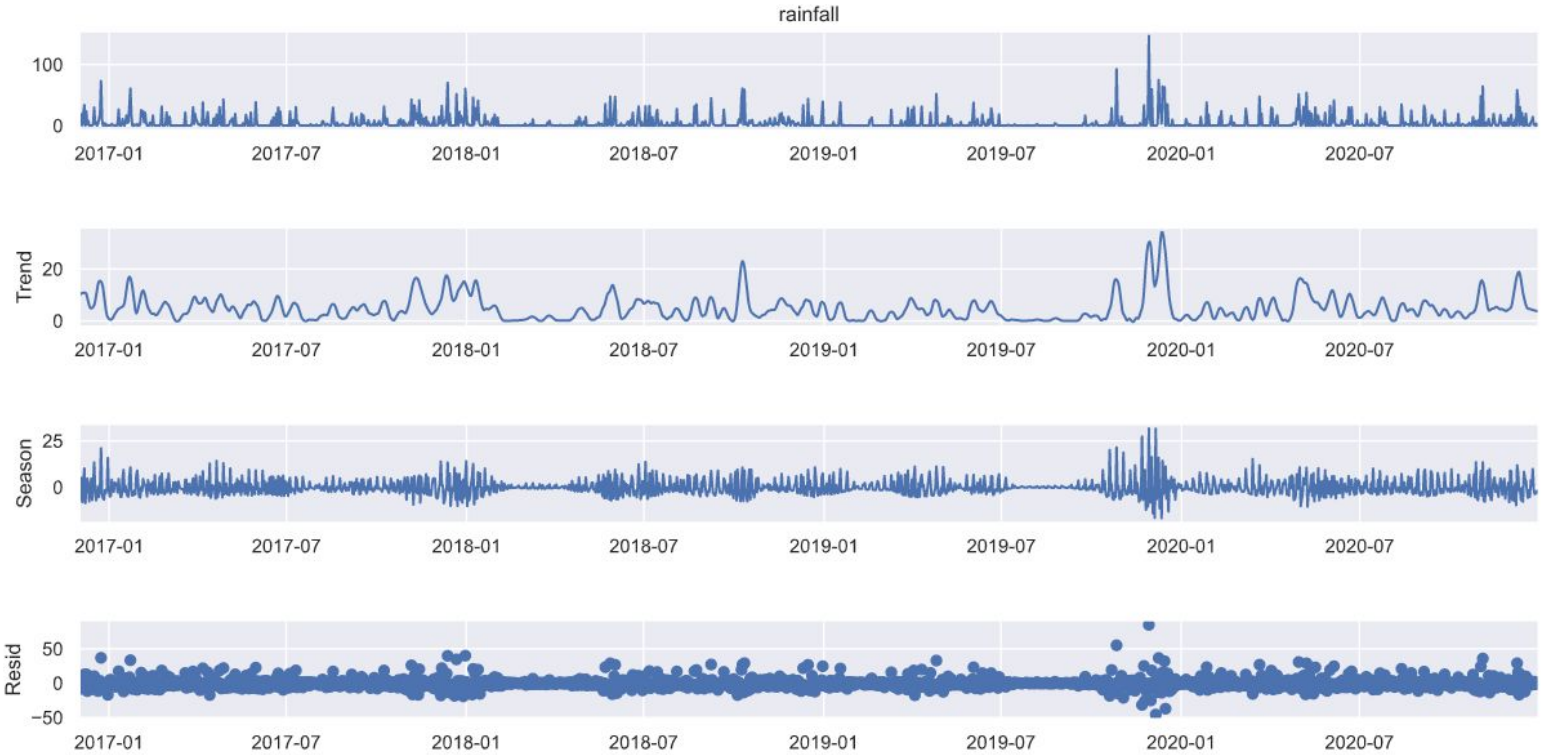
@ongchinhwee

Analyzing Rainfall with Moving Averages



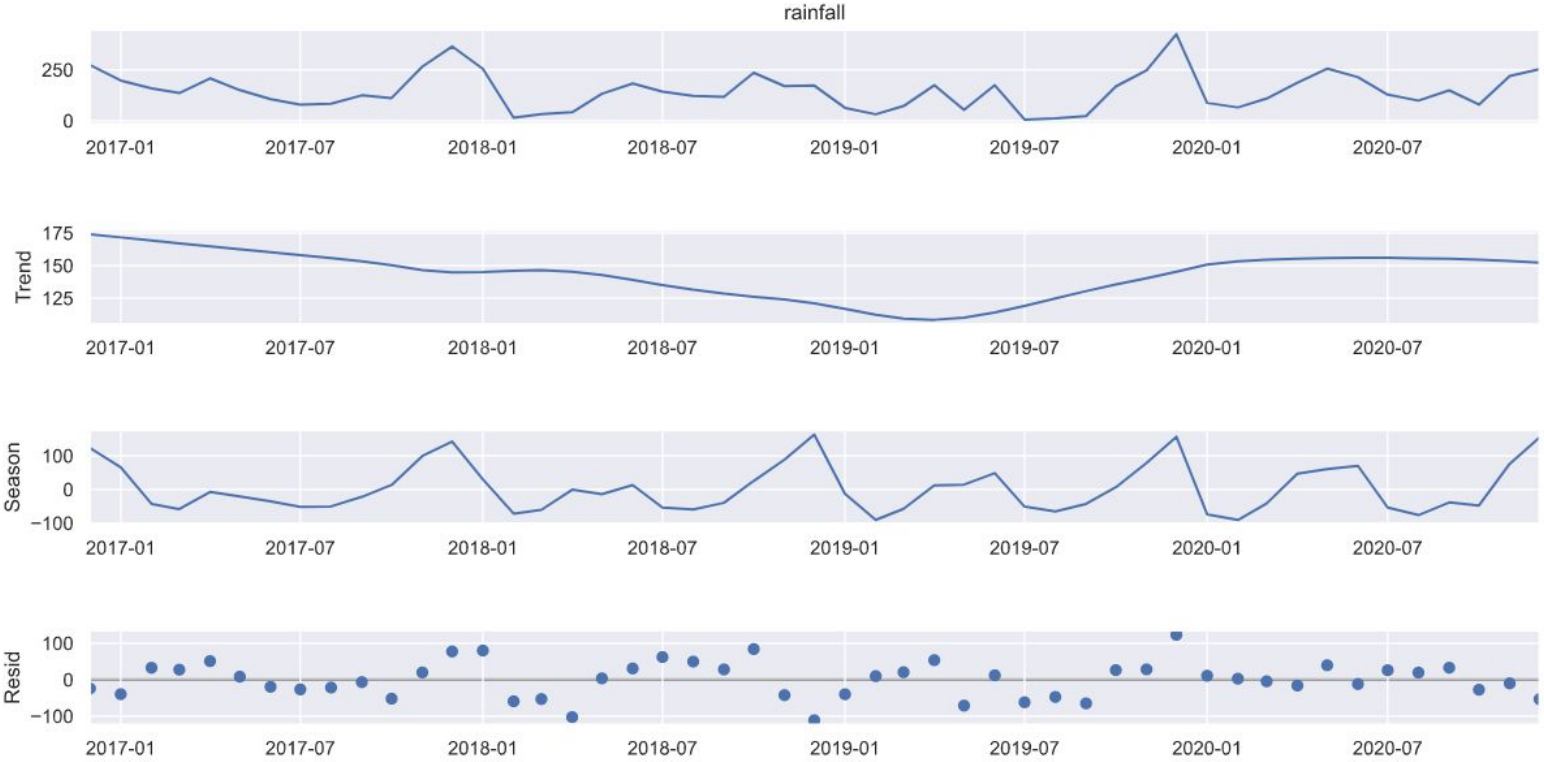
@ongchinhwee

STL Decomposition of Daily Rainfall



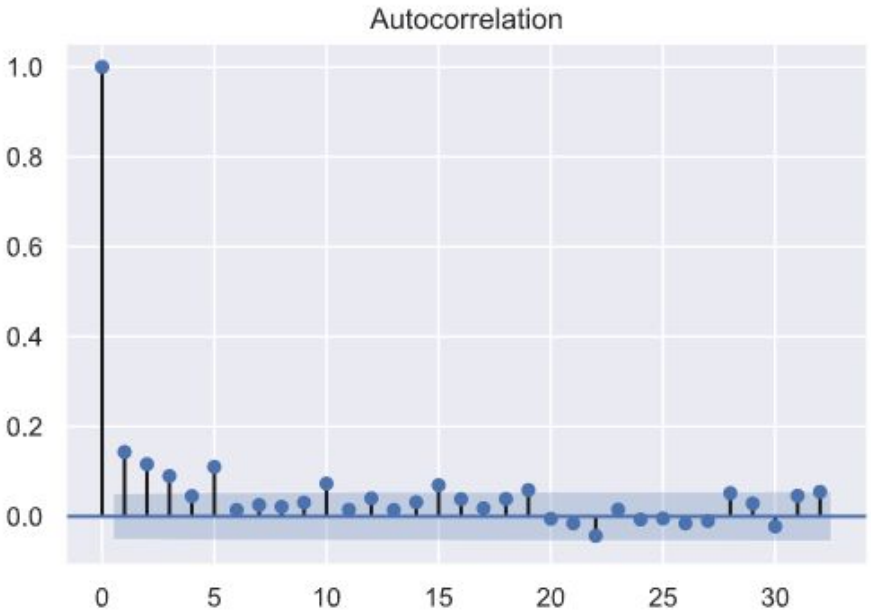
@ongchinhwee

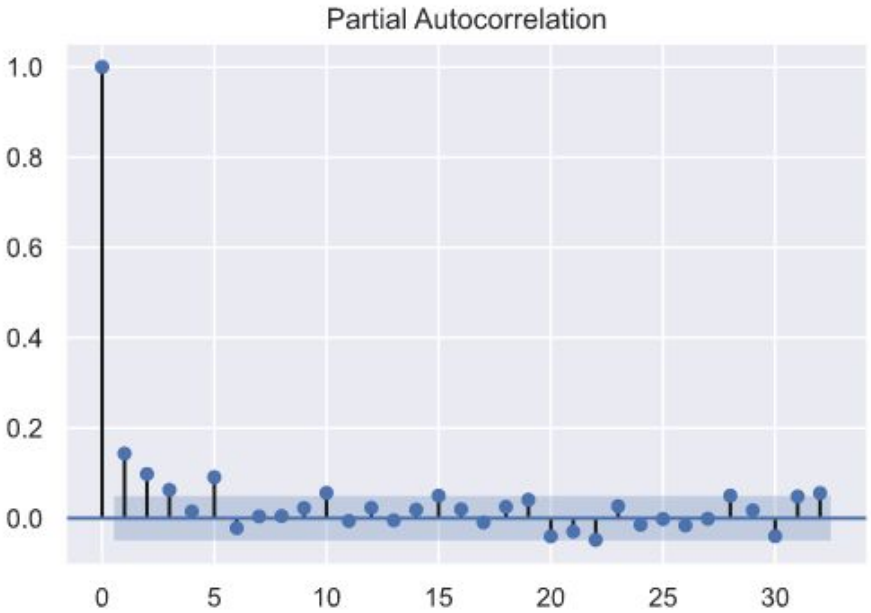
STL Decomposition of Monthly Rainfall



@ongchinhwee

Autocorrelation of Daily Rainfall

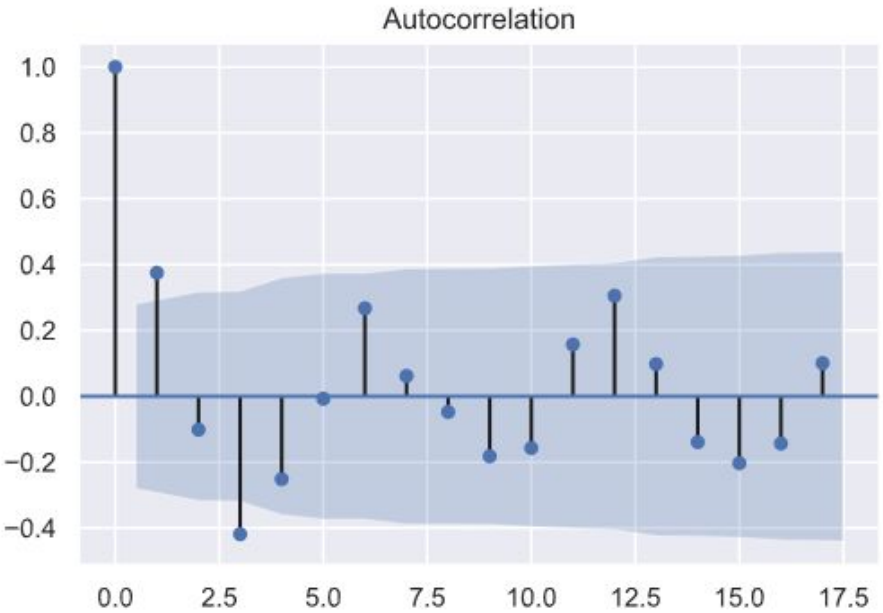


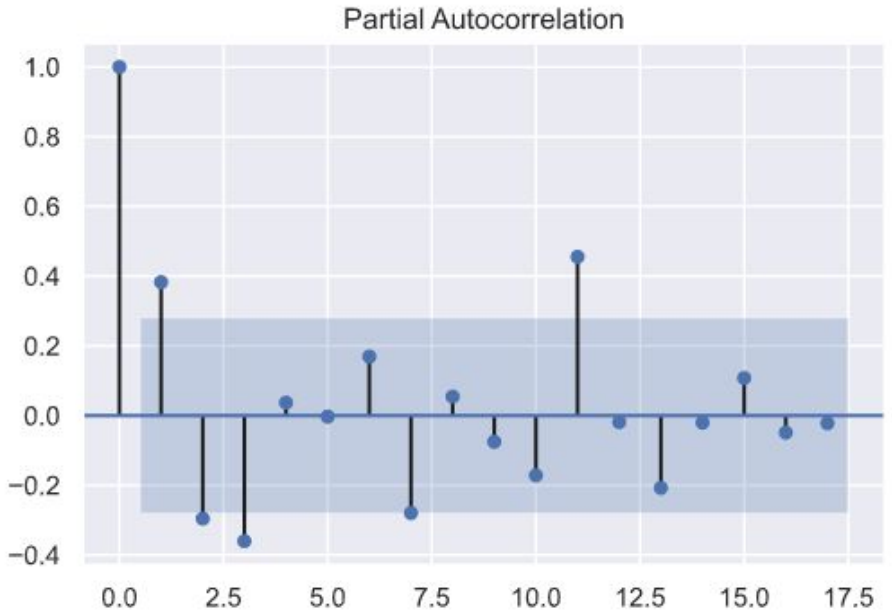


Low correlation between daily rainfall and its own lagged values.

@ongchinhwee

Autocorrelation of Monthly Rainfall





Most positive: **1st cofficient (r_1)**; Most negative: **3rd coefficient (r_3)**

@ongchinhwee

Recap:
Could we **predict heavier rainfall** with weather data?

Rainfall Forecasting with ARIMA models

ARIMA(p,d,q) model

(AutoRegressive Integrated Moving Average)

where:

p: order of the **autoregressive** part;

d: degree of first **differencing** involved;

q: order of the **moving average** part.

Rainfall Forecasting with ARIMA models

1. Apply **rolling forecast technique** with ARIMA(p, d, q) on time series data
2. Minimise **root-mean-squared-error (RMSE)**
3. Use optimized order parameters (p, d, q) to run **rolling forecast for next N cycles**
    a. Daily Forecast: N = 61
    b. Monthly Forecast: N = 13

Forecasting of Daily Rainfall with ARIMA

SARIMAX Results

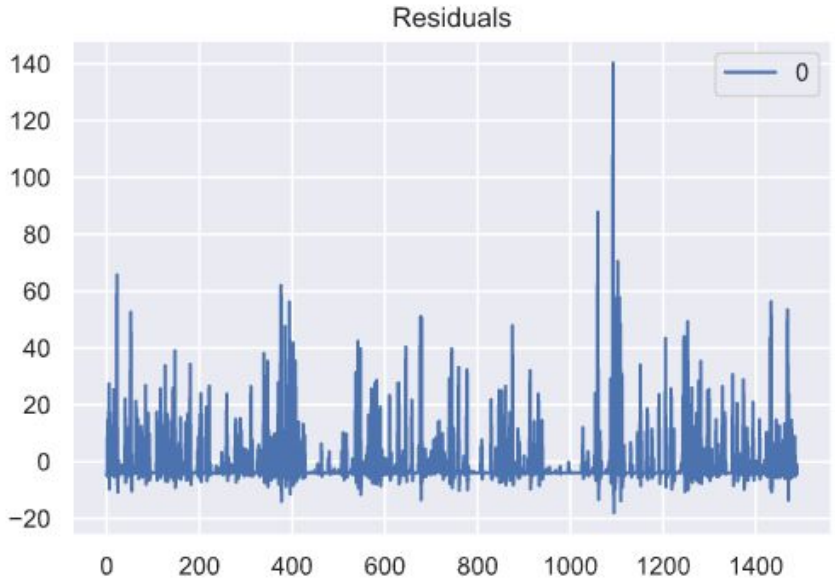
Dep. Variable:	y	No. Observations:	1490
Model:	ARIMA(0, 0, 2)	Log Likelihood	-5687.965
Date:	Thu, 14 Jan 2021	AIC	11383.930
Time:	23:48:12	BIC	11405.156
Sample:	0	HQIC	11391.840
	- 1490		
Covariance Type:	opg		

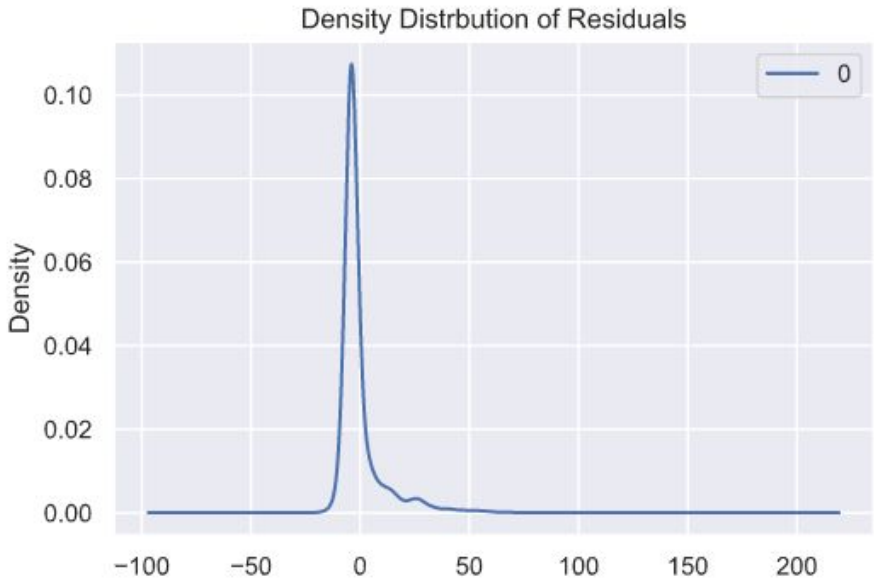
	coef	std err	z	P>|z|	[0.025	0.975]
const	4.8190	0.542	8.890	0.000	3.757	5.881
ma.L1	0.1197	0.020	5.900	0.000	0.080	0.159
ma.L2	0.0972	0.018	5.291	0.000	0.061	0.133
sigma2	121.0694	1.816	66.679	0.000	117.511	124.628

Ljung-Box (L1) (Q):	0.06	Jarque-Bera (JB):	51835.43
Prob(Q):	0.81	Prob(JB):	0.00
Heteroskedasticity (H):	1.66	Skew:	4.14
Prob(H) (two-sided):	0.00	Kurtosis:	30.68

Residual Errors for ARIMA Daily Rainfall Model

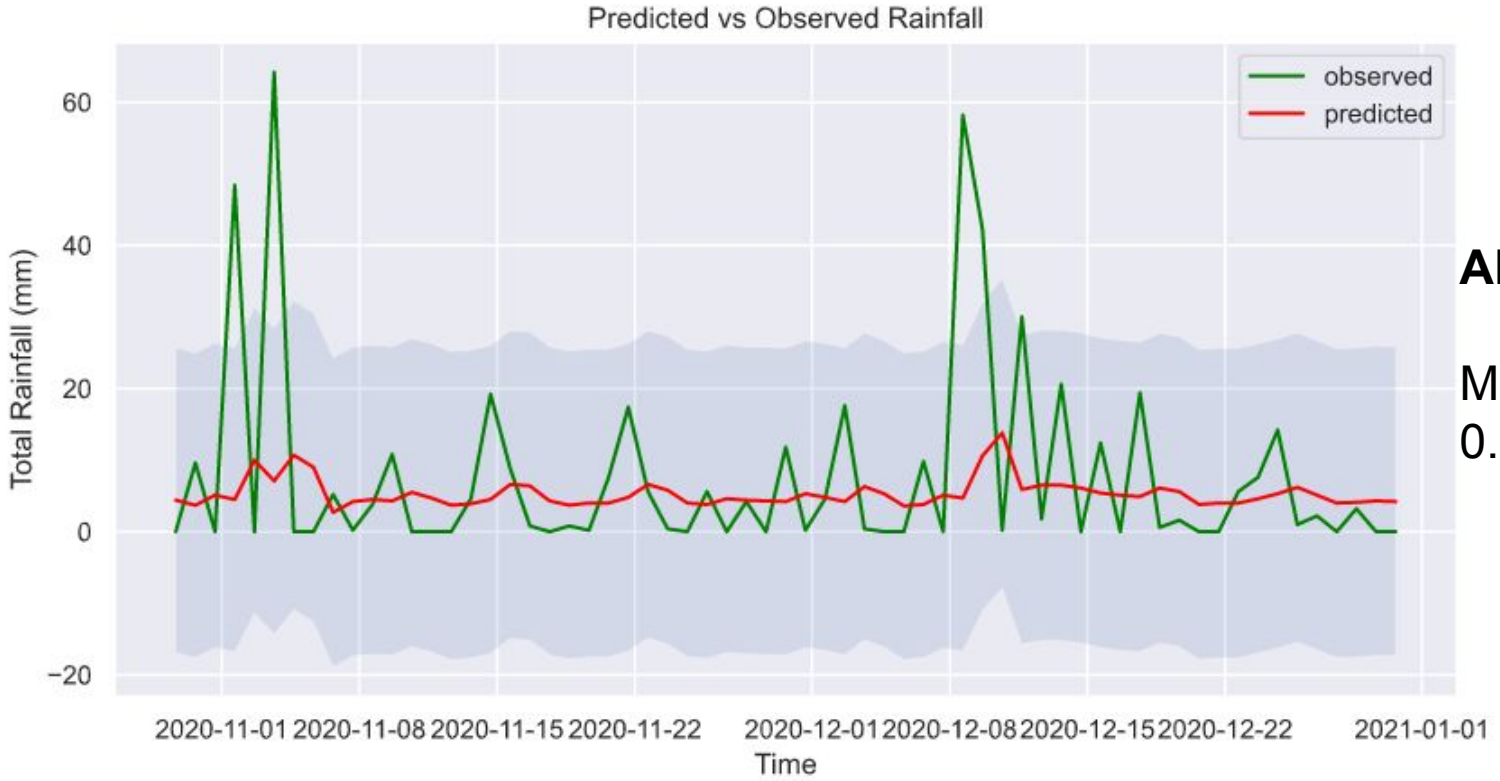
ARIMA(0,0,2)





@ongchinhwee

Forecasting of Daily Rainfall with ARIMA



ARIMA(0,0,2)

Min-max error: 0.757

@ongchinhwee

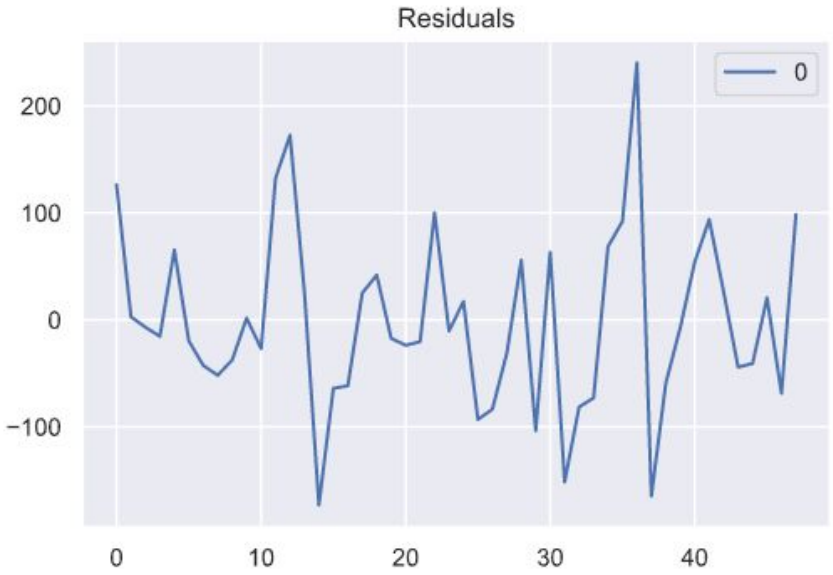
Forecasting of Monthly Rainfall with ARIMA

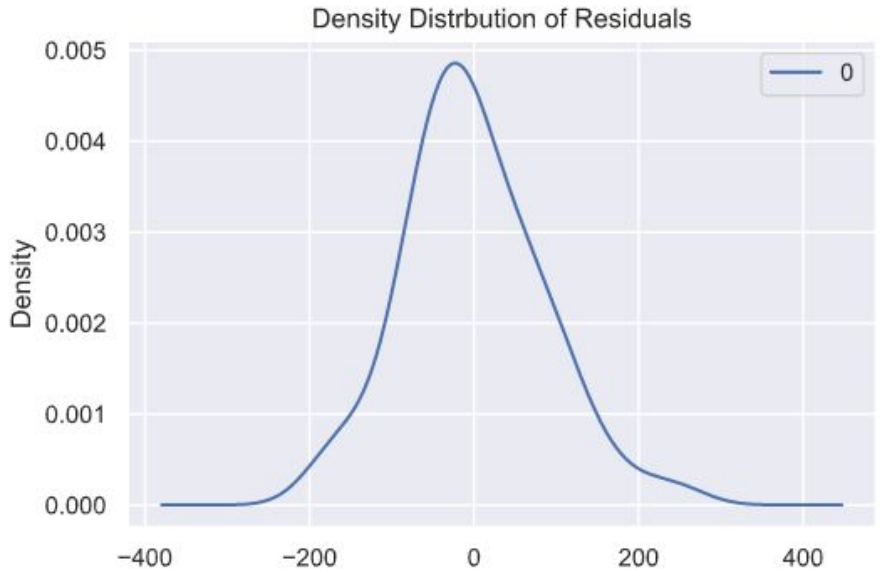
```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                   48
Model:                 ARIMA(1, 0, 0)   Log Likelihood                -280.016
Date:                Thu, 14 Jan 2021   AIC                            566.033
Time:                        23:50:13   BIC                            571.646
Sample:                             0   HQIC                           568.154
                                  - 48
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const        146.8506     21.313      6.890      0.000     105.077     188.624
ar.L1          0.3792      0.116      3.269      0.001       0.152       0.607
sigma2      6810.5983   1297.728      5.248      0.000    4267.097    9354.099
===================================================================================
Ljung-Box (L1) (Q):                   0.42   Jarque-Bera (JB):                 1.77
Prob(Q):                              0.52   Prob(JB):                         0.41
Heteroskedasticity (H):               1.35   Skew:                             0.41
Prob(H) (two-sided):                  0.56   Kurtosis:                         3.48
===================================================================================
```

@ongchinhwee

Residual Errors for ARIMA Monthly Rainfall Model

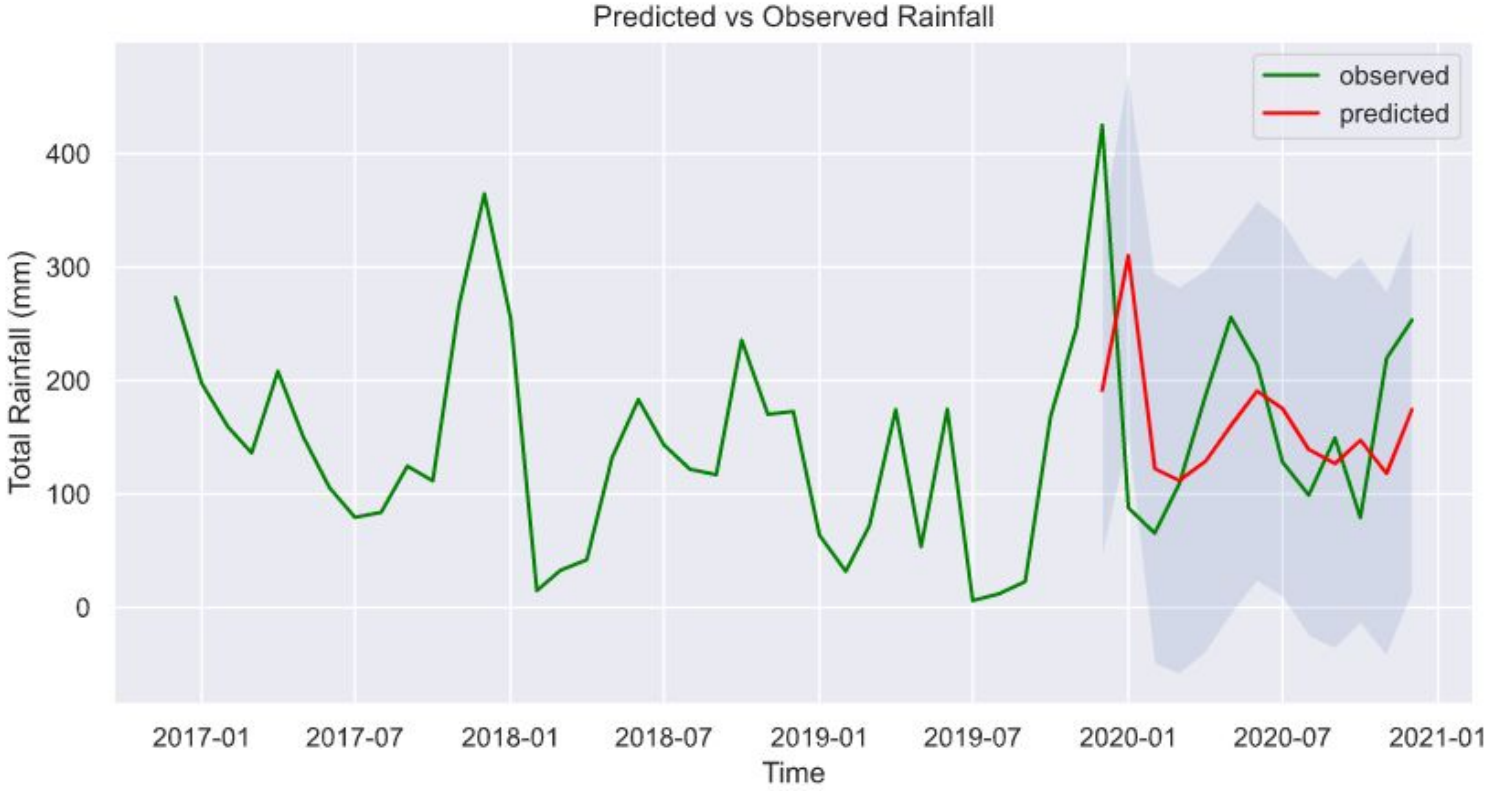
ARIMA(1,0,0)





@ongchinhwee

Forecasting of Monthly Rainfall with ARIMA



ARIMA(1,0,0)

Min-max error: 0.346

@ongchinhwee

Key Takeaways

- With **climate change**, rainfall patterns are becoming **more extreme** and **more challenging to predict**
  - **Highest rainfall** in December 2019 (NE Monsoon)
  - **Higher-than-expected rainfall** in May 2020 (Inter-Monsoon) - also **earlier-than-expected monsoon**
- Rainfall data from weather station + ARIMA **may not be sufficient enough** to predict **more “erratic” spikes in daily rainfall**


SALE

Reach out to me!


: ongchinhwee


: @ongchinhwee


: hweecat


: https://ongchinhwee.me

And check out my project on:


hweecat/api-scraping-nea-datasets