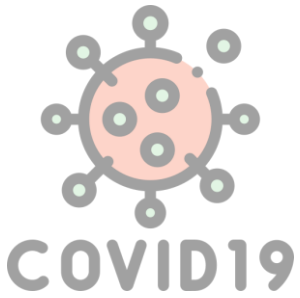
ศึกษาการระบาดของ COVID-19

และเปรียบเทียบการพยากรณ์อนุกรมเวลาด้วยวิธีต่างๆ



นางสาวยุภาภรณ์ วันนา 645020061-2

ข้อมูล

COVID19

- **Corona-2019 Dataset:** ข้อมูลเกี่ยวกับโควิด-19 ไฟล์หลักในชุดข้อมูลนี้คือ covid_19_data.csv โดยมีรายละเอียดข้อมูลดังนี้

Date - วันที่สังเกตใน YYYY/MM/DD เริ่มจาก 2020-01-22 - 2021-05-29

Province/State – จังหวัดหรือรัฐที่สังเกต **Missing

Country/Region - ประเทศที่สังเกต

Confirmed - จำนวนผู้ป่วยที่ได้รับการยืนยันสะสมจนถึงวันนั้น

Deaths - จำนวนผู้เสียชีวิตสะสมจนถึงวันนั้น

Recovered - จำนวนเคสหายสะสมจนถึงวันที่นั้น

	Date	Province/State	Country/Region	Confirmed	Deaths	Recovered
0	01/22/2020	Anhui	Mainland China	1.0	0.0	0.0
1	01/22/2020	Beijing	Mainland China	14.0	0.0	0.0
2	01/22/2020	Chongqing	Mainland China	6.0	0.0	0.0
3	01/22/2020	Fujian	Mainland China	1.0	0.0	0.0
4	01/22/2020	Gansu	Mainland China	0.0	0.0	0.0
...	...	...	...	...	...	...
306424	05/29/2021	Zaporizhia Oblast	Ukraine	102641.0	2335.0	95289.0
306425	05/29/2021	Zeeland	Netherlands	29147.0	245.0	0.0
306426	05/29/2021	Zhejiang	Mainland China	1364.0	1.0	1324.0
306427	05/29/2021	Zhytomyr Oblast	Ukraine	87550.0	1738.0	83790.0
306428	05/29/2021	Zuid-Holland	Netherlands	391559.0	4252.0	0.0

306429 rows × 6 columns

ข้อมูล

- **Population:** ข้อมูลจำนวนประชากรของแต่ละประเทศคือไฟล์ worldometer.csv โดยมีรายละเอียดข้อมูลดังนี้

Country/Region – ประเทศ

Population - จำนวนประชากรของแต่ละประเทศ

	Unnamed: 0	Country/Region	Population
0	0	China	1439323776
1	1	India	1380004385
2	2	United States	331002651
3	3	Indonesia	273523615
4	4	Pakistan	220892340
...	...	...	...
224	224	Tuvalu	11792
225	225	Wallis & Futuna	11239
226	226	Nauru	10824
227	227	Saint Barthelemy	9877
228	228	Saint Helena	6077

229 rows × 3 columns

Data management

➢ **เช็คค่า Missing:**

○ **ตาราง covid_19_data**

```
# ลบคอลัมน์ Province/State ออกเนื่องจากข้อมูลส่วนนี้มีส่วนสำคัญที
covid.drop('Province/State', axis='columns', inplace=True)
```

Province/State มีค่า missing จำนวน 78100 แถว

> ลบคอลัมน์ Province/State ออกเนื่องจากข้อมูลส่วนนี้มีส่วนสำคัญที่จะนำไปวิเคราะห์ต่อหากเติมค่า missing ด้วย mode อาจทำให้ข้อมูลมีความคลาดเคลื่อนได้

○ **ตาราง worldometer**

ไม่พบค่า missing

```
1   print("เช็ค null values:\n",worldometer.isnull().sum())

เช็ค null values:
 Unnamed: 0       0
Country/Region    0
Population        0
dtype: int64
```

Data management

➢ **เช็คประเภทข้อมูล:**

○ **ตาราง covid_19_data**

คอลัมน์ Date เป็นประเภท object ควรเปลี่ยนเป็น datetime

> แปลงประเภทข้อมูลคอลัมน์ "Date" เป็น Datetime

```
# แปลงคอลัมน์ "Date" เป็น Datetime
covid["Date"]=pd.to_datetime(covid["Date"])
```

○ **ตาราง worldometer**

คอลัมน์ Population เป็นประเภท object ควรเปลี่ยนเป็น int

> แปลงประเภทข้อมูลคอลัมน์ "Population" เป็น int

> แปลงปกติด้วยคำสั่ง .astype() ไม่ได้เนื่องจาก value = ‘300,000’

```
Population = worldometer['Population'].tolist()
a = []
for i in Population:
    a_= i.replace(',', '')
    a.append(int(a_))

worldometer['Population'] = a
```

วัตถุประสงค์

- เพื่อศึกษาการระบาดของ COVID-19 ด้วยการทำ Visualizations และการจัดกลุ่มข้อมูลวิธี Clustering

- เพื่อคาดการณ์และพยากรณ์อนุกรมเวลาเพื่อศึกษาผลกระทบและการแพร่กระจายของ COVID-19 ในอนาคต(นับจากวันสิ้นสุดการอัพเดทของข้อมูล)

เทคนิคที่ใช้

- **Clustering:**
  - k-means clustering

- **Time Series Forecasting:** ตัวแบบทาง Mining
  - Regression based on k-nearest neighbors (KNeighborsRegressor)

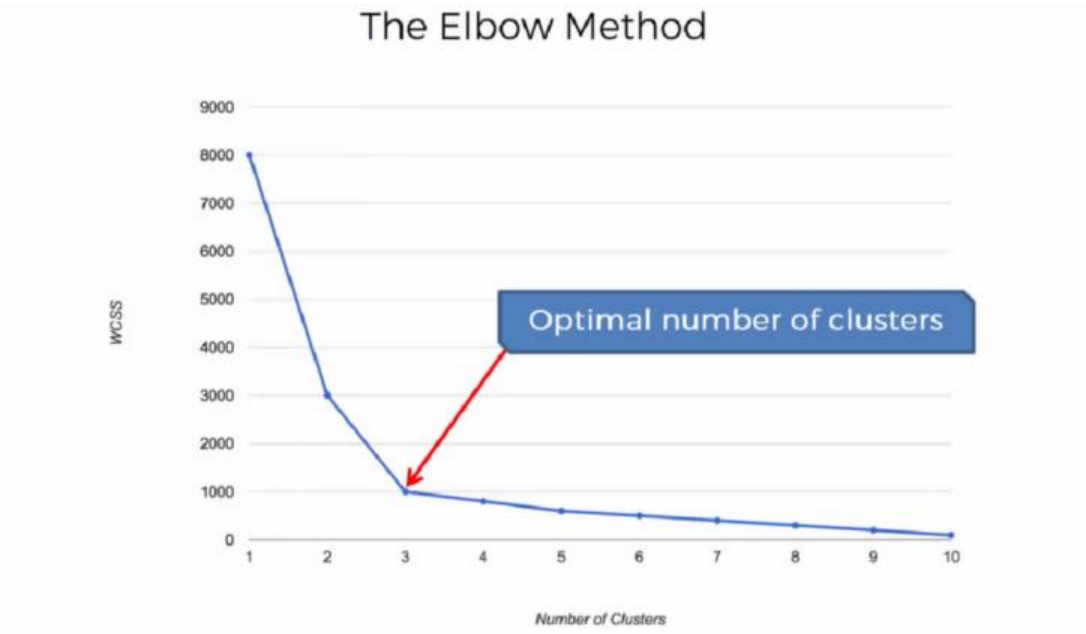
เทคนิคที่ใช้

- **Time Series Forecasting:** ตัวแบบทางสถิติ
  - Facebook's Prophet Model
  - Holt's Linear
  - Auto Regressive Model (AR)
  - Holt's Winter Model
  - ARIMA Model
  - SARIMA Model
  - Moving Average Model (MA)
  - Linear Regression
  - Polynomial Regression

เกณฑ์วัดประสิทธิภาพโมเดล

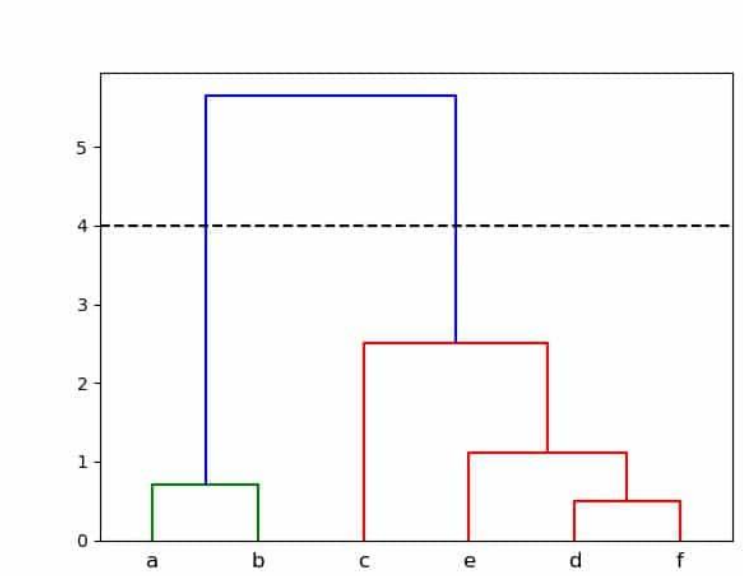
- **Clustering:**
  - วิธีการเลือกจำนวนคลัสเตอร์ที่เหมาะสม

- Within Sum of Squares (WCSS) by Elbow Method



(อ้างอิงจาก: https://ichi.pro/th/withi-kar-leuxk-canwn-khlastexr-thi-hemaa-sm-ni-xal-kx-ri-thum-k-mean-159844816308064)

- Hierarchical Clustering



(อ้างอิงจาก: https://aiaspirant.com/hierarchical-clustering/)

เกณฑ์วัดประสิทธิภาพโมเดล

➢ **Time Series Forecasting:**

- RMSE (Root Mean Square Error) เป็นกฎการให้คะแนนกำลังสองที่วัดขนาดเฉลี่ยของข้อผิดพลาดด้วย มันคือรากที่สองของค่าเฉลี่ยของความแตกต่างกำลังสองระหว่างการทำนายและการสังเกตจริง

  มีค่าอยู่ตั้งแต่ 0 ไปจนถึง ∞ ค่า Error ยิ่งต่ำก็ยิ่งดี โดยปกติใช้สำหรับเปรียบประสิทธิภาพโมเดล

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

  (อ้างอิงจาก: https://www.tpa.or.th/writer/read_this_book_topic.php?bookID=3086&read=true)

- R-Squared คือ ค่าความผันแปรของตัวแปรตอบสนอง (y) ที่สามารถอธิบายได้มีอยู่ในตัวแบบเชิงเส้นนี้ คิดเป็นกี่เปอร์เซ็นต์

  R-Squared = ความผันแปรที่สามารถอธิบายได้ / ความผันแปรทั้งหมด (Explained variation / Total Variation)

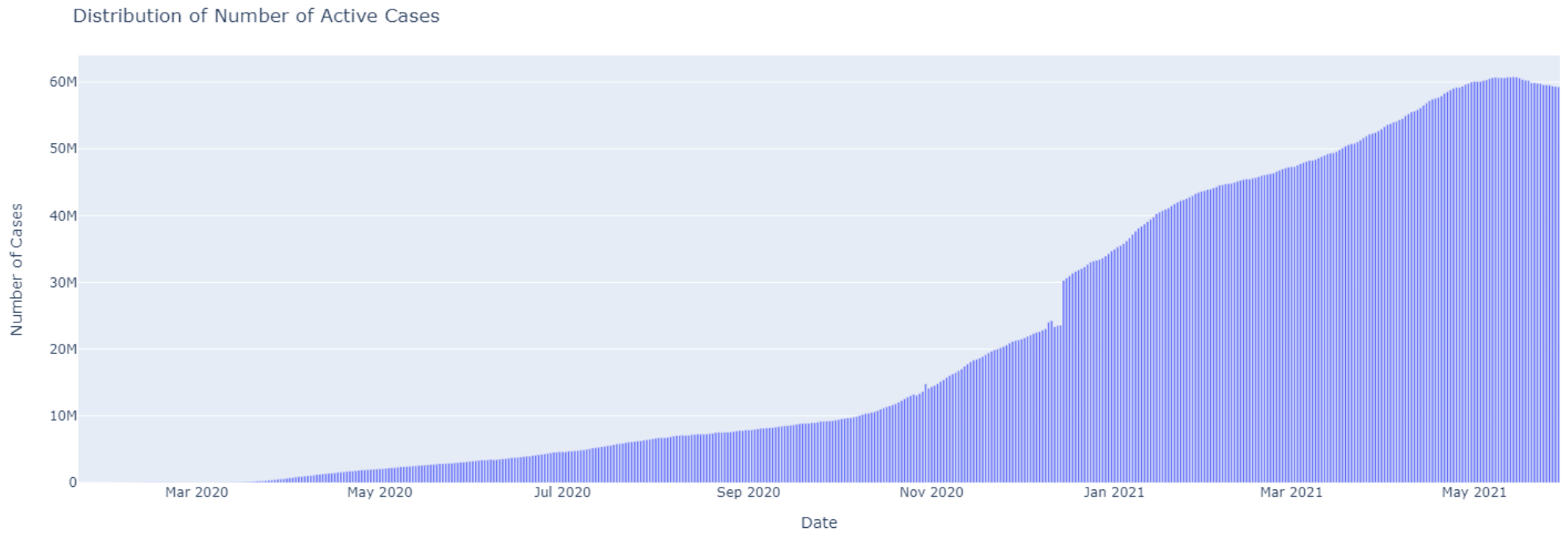
  ค่า R-Squared จะมีค่าอยู่ระหว่าง 0% - 100%

  - 0% แสดงให้เห็นว่า ตัวแบบที่ได้มาไม่สามารถอธิบายความผันแปรของค่าตัวแปรตอบสนอง ต่างที่กระจายรอบค่าเฉลี่ยได้เลย

  - 100% แสดงให้เห็นว่า ตัวแบบที่ได้มาสามารถอธิบายความผันแปรของค่าตัวแปรตอบสนอง ต่างที่กระจายรอบค่าเฉลี่ยได้เป็นอย่างดี (อ้างอิงจาก: https://www.tpa.or.th/writer/read_this_book_topic.php?bookID=3086&read=true)

Visualization

1.การวิเคราะห์ตามวันที่

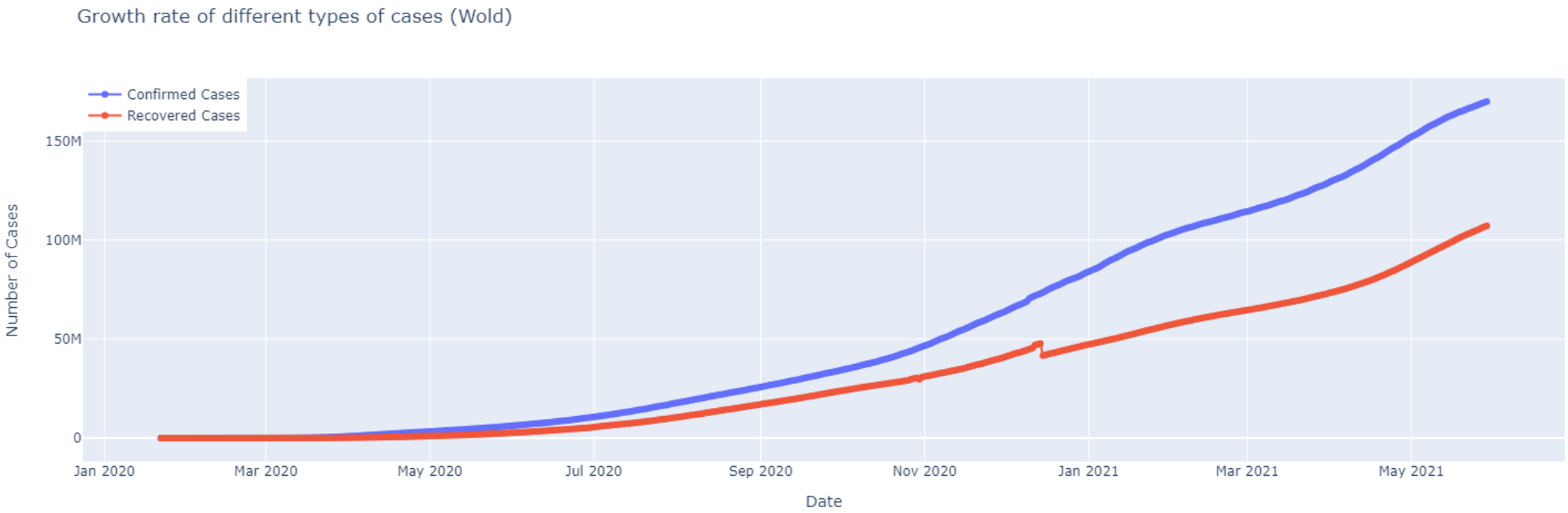


#Active Cases = Number of Confirmed Cases - Number of Recovered Cases - Number of Death Cases

#การเพิ่มจำนวน Active Cases จะบ่งบอกได้ว่าจำนวนของ Recovered case หรือ Death case ลดลงมากเมื่อเทียบกับจำนวน Confirmed Cases

Visualization

1.การวิเคราะห์ตามวันที่



#สรุปได้ว่า จากกราฟอัตราการเพิ่มขึ้นในแต่ละวันของ Confirmed และ Recovered จะแนวโน้มเพิ่มขึ้นเรื่อยๆ

Visualization

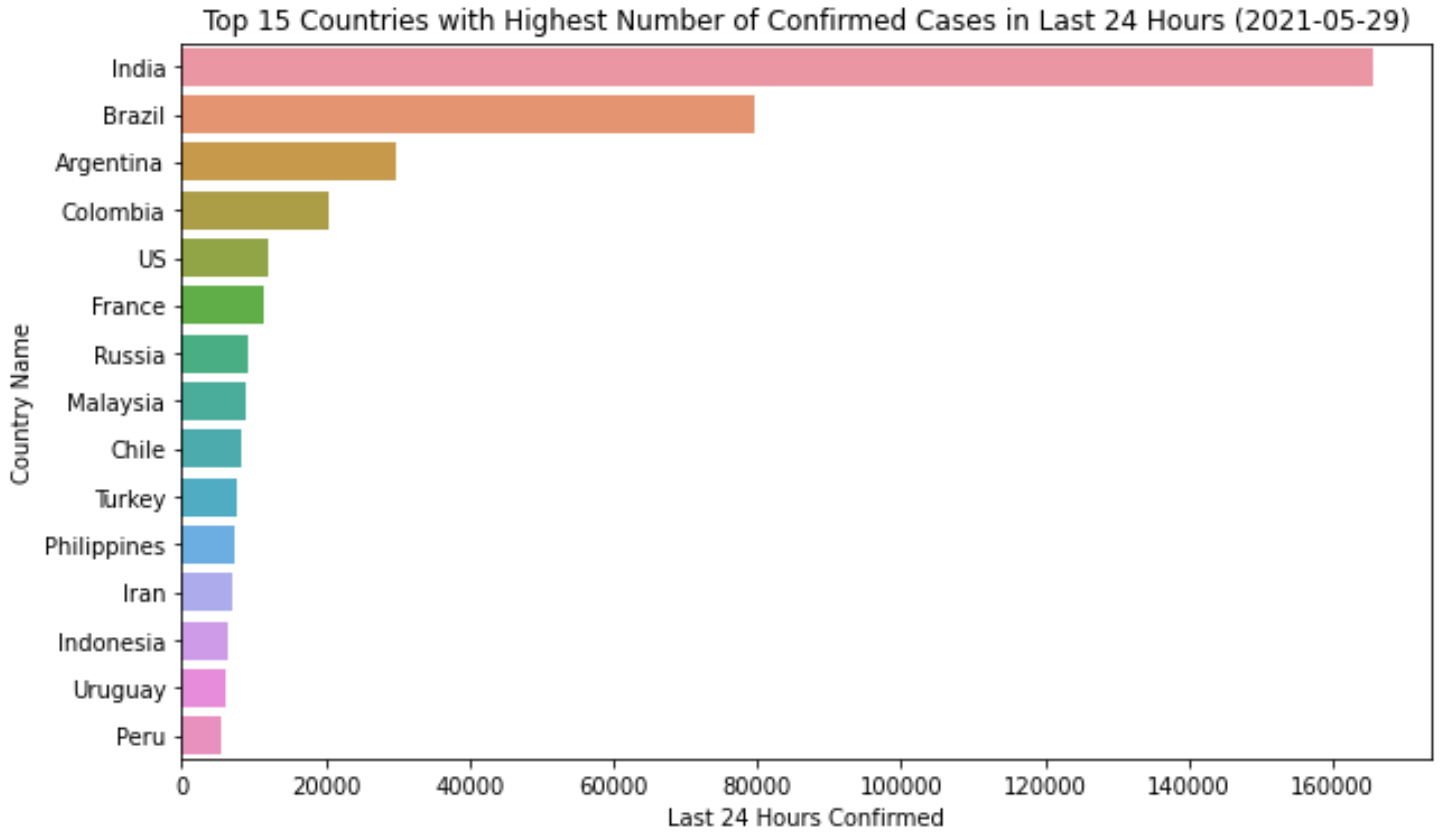
1.การวิเคราะห์ตามวันที่

#Confirmed case proportion=
(Number of Confirmed Cases / Max value of Number of Confirmed Cases) x 100
#Mortality rate =
(Number of Death Cases / Number of Confirmed Cases) x 100
#Recovery rate=
(Number of Recovery Cases / Number of Confirmed Cases) x 100

#สรุปผล
#Mortality rate แสดงให้เห็นว่าใช้เวลานานพอสมควรกว่าจะมีแนวโน้มของจำนวนคนตายที่ลดลงบ่งบอกได้ถึงการจัดการรับมือกับเชื้อโควิดที่ดีขึ้น
#Recovery rate แสดงให้เห็นว่าช่วงเดือนพฤษภาคม - มิถุนายน บ่งบอกได้ถึงความรุนแรงของเชื้อโควิดว่าอาจเกิดการกลายพันธ์ได้ทำให้มีจำนวนผู้เสียที่สูงมาก
หรือเป็นช่วงเวลาที่เกิดการรุนแรงที่สุดของเชื่อโควิด
Confirmed case proportion เมื่อนำมาเปรียบเทียบกับกราฟแถวที่สองและสามแล้วแสดงให้เห็นว่าช่วงที่เกิดความรุนแรงของเชื้อโควิดว่าอาจเกิดการกลายพันธ์ได้ทำให้มีจำนวนผู้เสียที่สูงมากเป็นจริงไม่มีความเกี่ยวข้องกับจำนวนผู้ที่ติดเพิ่มขึ้น

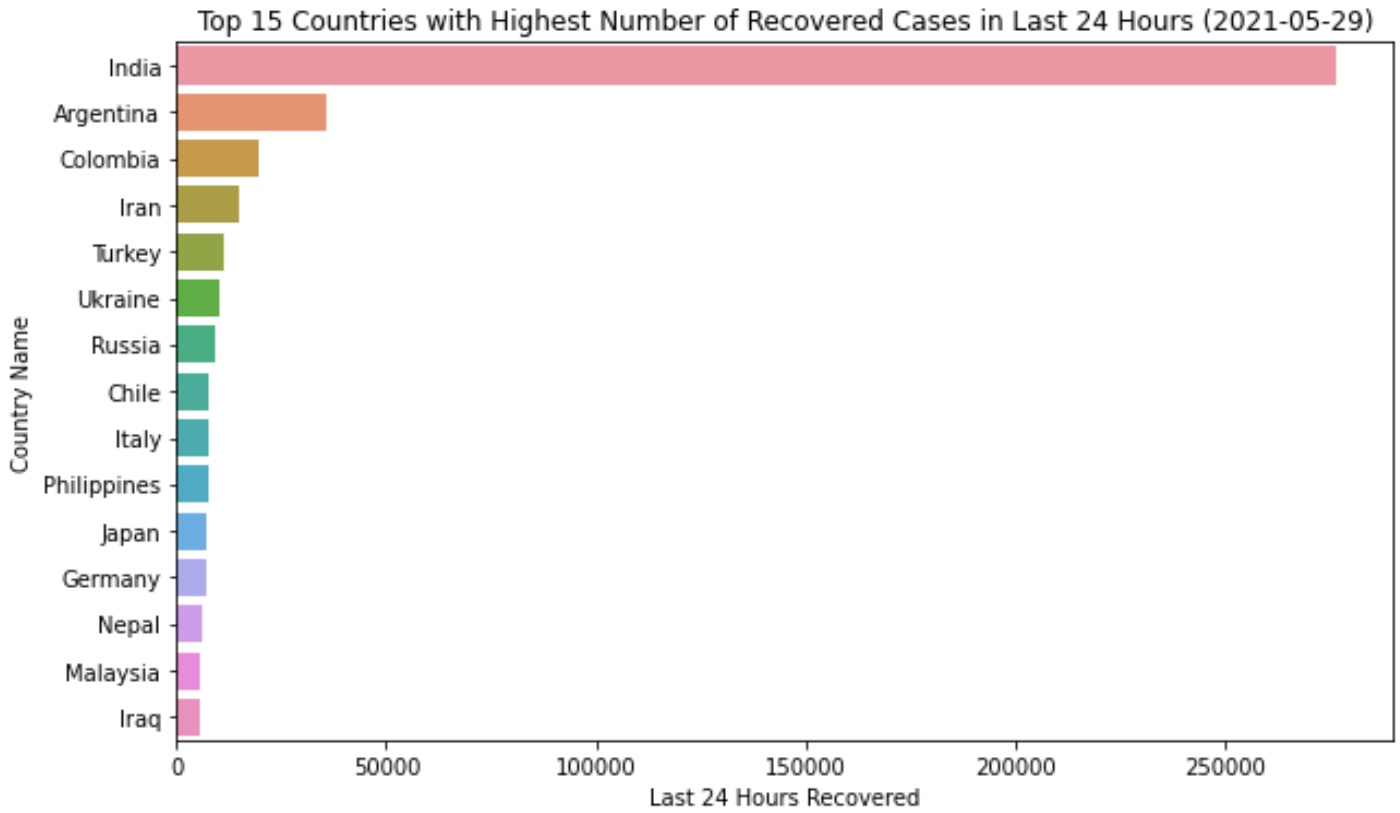
Visualization

1.การวิเคราะห์ตามวันที่



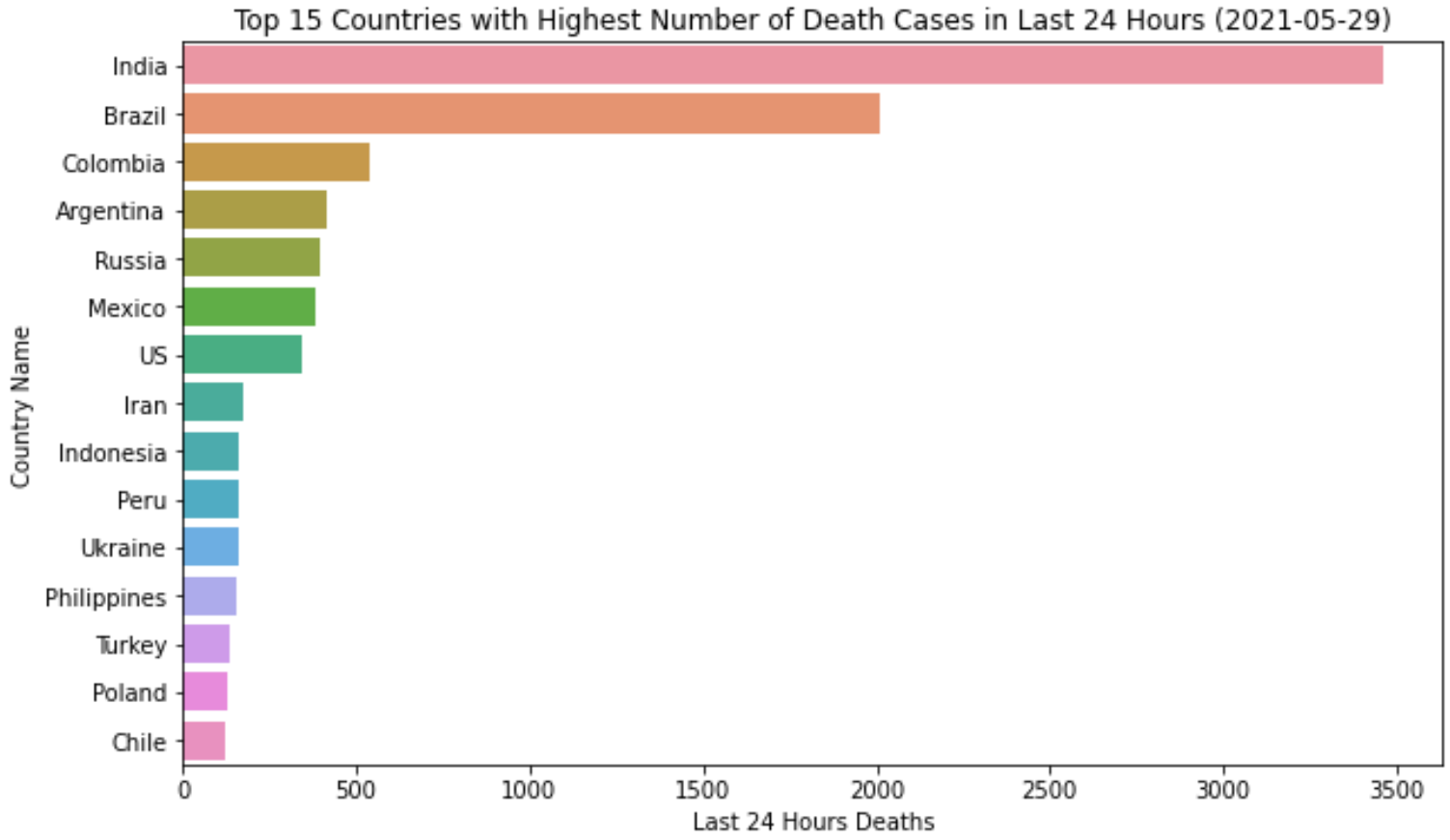
Visualization

1.การวิเคราะห์ตามวันที่



Visualization

1.การวิเคราะห์ตามวันที่



Visualization

2.การวิเคราะห์ตามประเทศ

หน่วยเป็น % แล้ว

สัดส่วน (%) ของประเทศใน Confirmed

Recovered และ Death Cases

#สัดส่วนต่อผลรวมทุกประเทศใน 1 วันสุดท้าย

Country Name	Proportion of Confirmed	Proportion of Recovered	Proportion of Deaths
India	34.430314	54.434397	32.946106
Brazil	16.569093	0.694050	19.158256
Argentina	6.206079	7.018125	3.951628
Colombia	4.262169	3.873325	5.141878
US	2.490667	0.000000	3.266045
France	2.397288	0.134358	0.628452
Russia	1.903980	1.801812	3.761188
Malaysia	1.875903	1.088849	0.933156
Chile	1.708694	1.515957	1.133118
Turkey	1.592230	2.202522	1.304513
Philippines	1.544813	1.463357	1.485431
Iran	1.478054	2.911939	1.647305
Indonesia	1.365333	1.067179	1.542563
Uruguay	1.242422	0.721040	0.552276

Visualization

2.การวิเคราะห์ตามประเทศ

หน่วยเป็น % แล้ว

สัดส่วน (%) ของประเทศใน Confirmed Recovered และ Death Cases

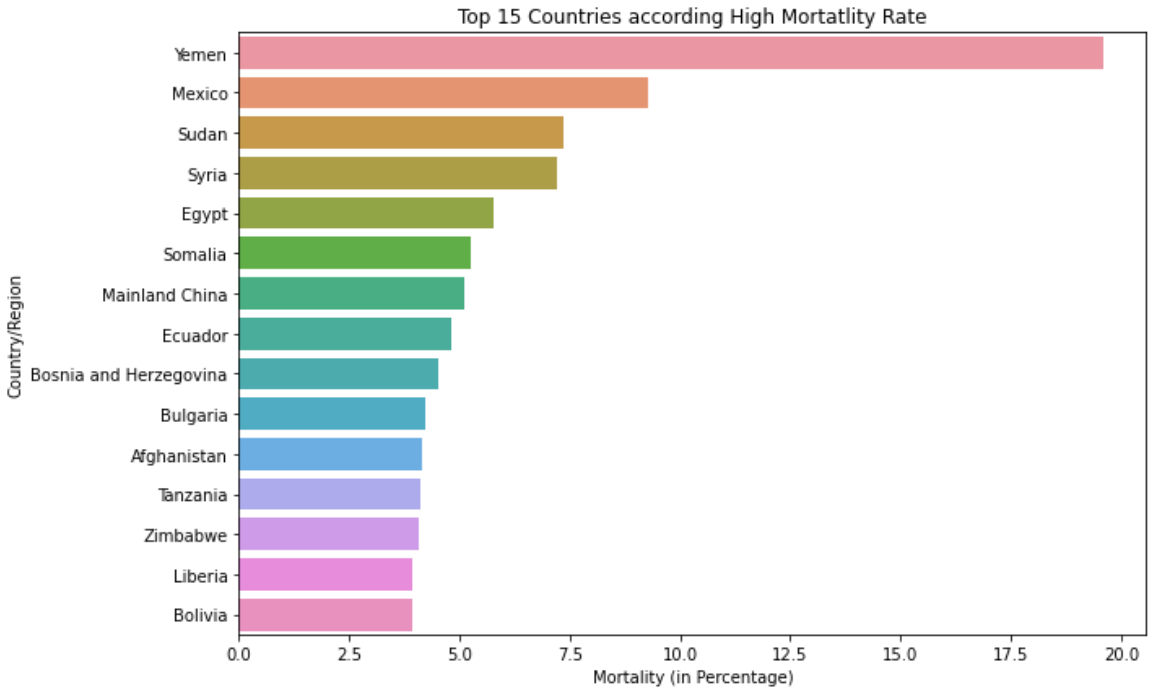
#สัดส่วนต่อประชากรของแต่ละประเทศใน 1 วันสุดท้าย

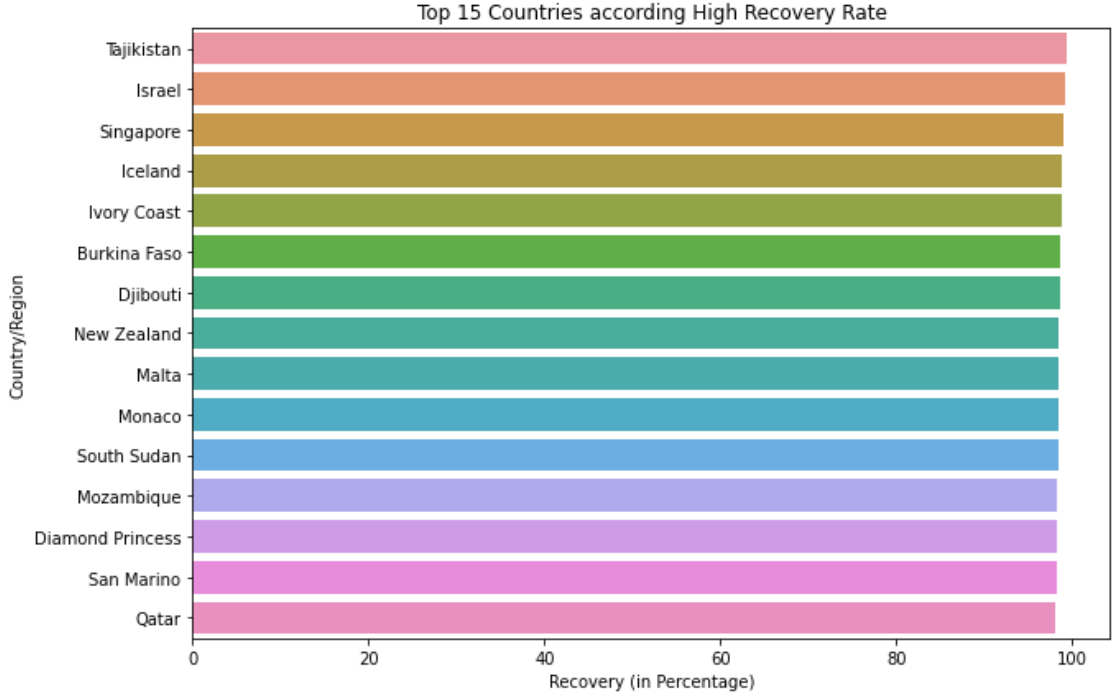
Country Name	Proportion of Confirmed	Proportion of Recovered	Proportion of Deaths
Maldives	0.196469	0.253633	0.000370
Bahrain	0.192351	0.148862	0.000940
Uruguay	0.171977	0.105362	0.001670
Argentina	0.066026	0.078822	0.000918
Chile	0.042979	0.040254	0.000623
Colombia	0.040277	0.038640	0.001061
Brazil	0.037481	0.001657	0.000947
Suriname	0.035798	0.021479	0.001023
Paraguay	0.034055	0.031223	0.001332
Malaysia	0.027869	0.017077	0.000303
Trinidad and Tobago	0.027224	0.022222	0.000857
Cabo Verde	0.027159	0.028238	0.000360
Kuwait	0.026554	0.026788	0.000094
Georgia	0.024717	0.014790	0.000301
Netherlands	0.019761	0.000175	0.000047
Lithuania	0.019102	0.063917	0.000367
Bolivia	0.018367	0.012987	0.000497
United Arab Emirates	0.018321	0.017987	0.000051
Denmark	0.018024	0.015020	0.000069

Visualization

2.การวิเคราะห์ตามประเทศ

15 อันดับสูงสุดประเทศตามอัตราการเสียชีวิตและอัตราการฟื้นตัวที่มี Confirmed Cases มากกว่า 500 ราย

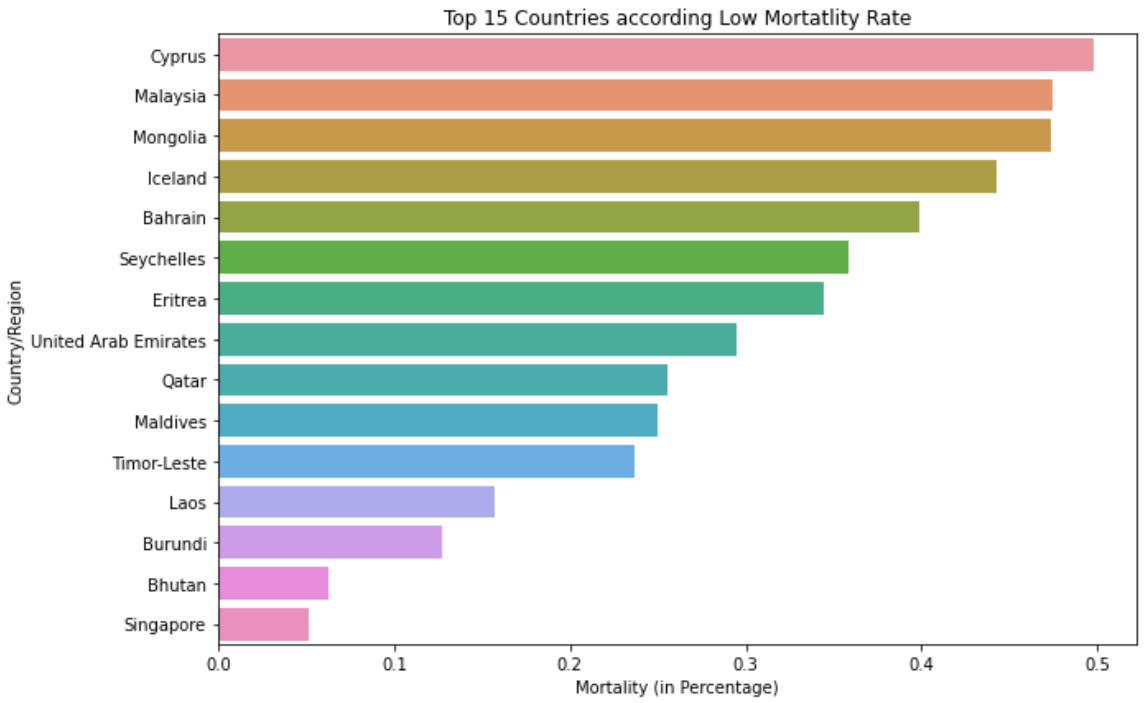


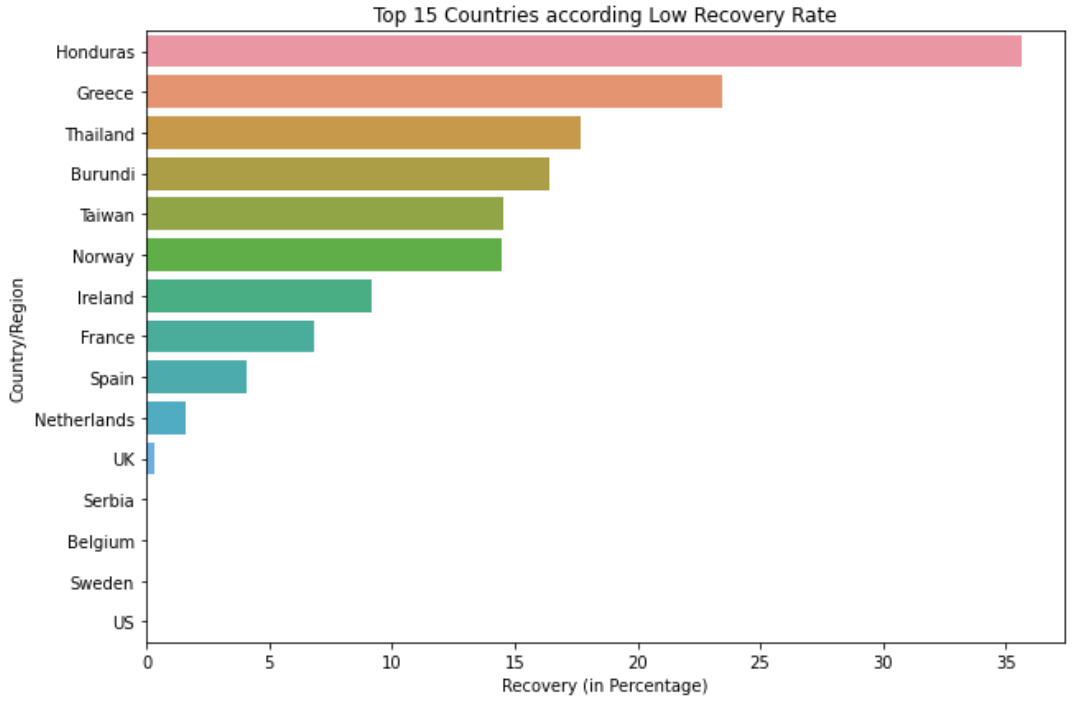


Visualization

2.การวิเคราะห์ตามประเทศ

15 อันดับสูงสุดประเทศตามอัตราการเสียชีวิตและอัตราการฟื้นตัวที่มี Confirmed Cases น้อยกว่า 500 ราย





Visualization

1. เลือกข้อมูล

จัดกลุ่มประเทศโดยพิจารณา

- Mortality rate
- Recovery rate

เนื่องจากทราบว่า COVID-19 มีอัตราการเสียชีวิตที่แตกต่างกันในแต่ละประเทศโดยพิจารณาจากปัจจัยต่างๆ และอัตราการฟื้นตัวก็เช่นกัน อาจเกี่ยวเนื่องกับแนวทางปฏิบัติในการควบคุมการระบาดใหญ่ของแต่ละประเทศ

Country/Region	Mortality	Recovery
US	1.787282	0.000000
India	1.168576	91.251129
Brazil	2.799103	88.007382
France	1.914692	6.833678
Turkey	0.902811	97.293743
...	...	...
Vanuatu	25.000000	75.000000
Marshall Islands	0.000000	100.000000
Samoa	0.000000	100.000000
Kiribati	0.000000	0.000000
Micronesia	0.000000	100.000000

195 rows × 2 columns

ขั้นตอนการจัดกลุ่มด้วยวิธี Clustering

2. แปลงข้อมูล

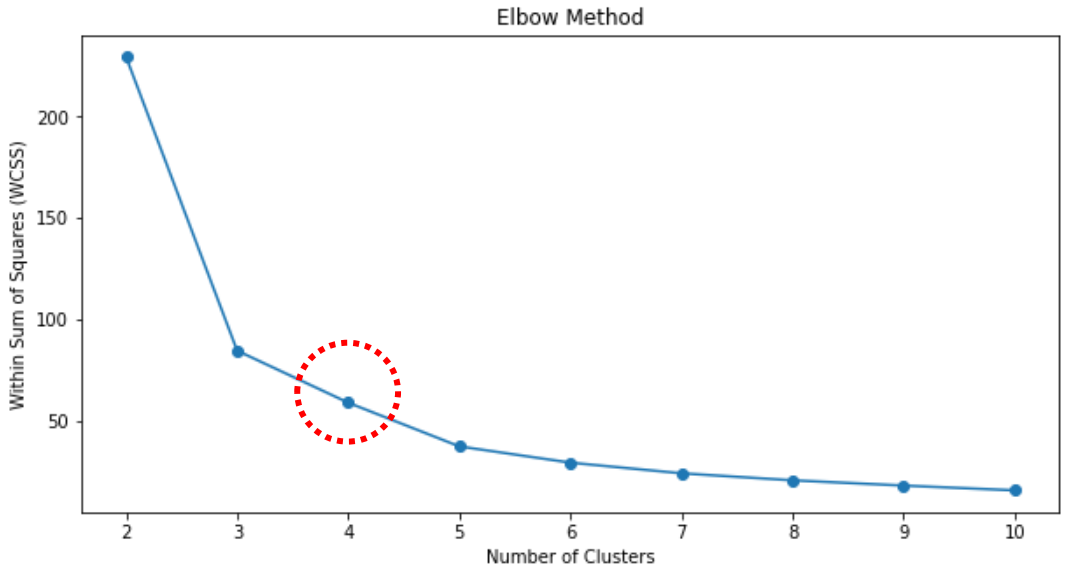
```
#Standard Scaling since K-Means Clustering is a distance based alogrithm
X=std.fit_transform(X)
X[0:5]
```

```
array([[-0.13259083, -3.19381604],
       [-0.34746838,  0.37668599],
       [ 0.21881596,  0.24976367],
       [-0.08834139, -2.92642579],
       [-0.43976895,  0.61312324]])
```

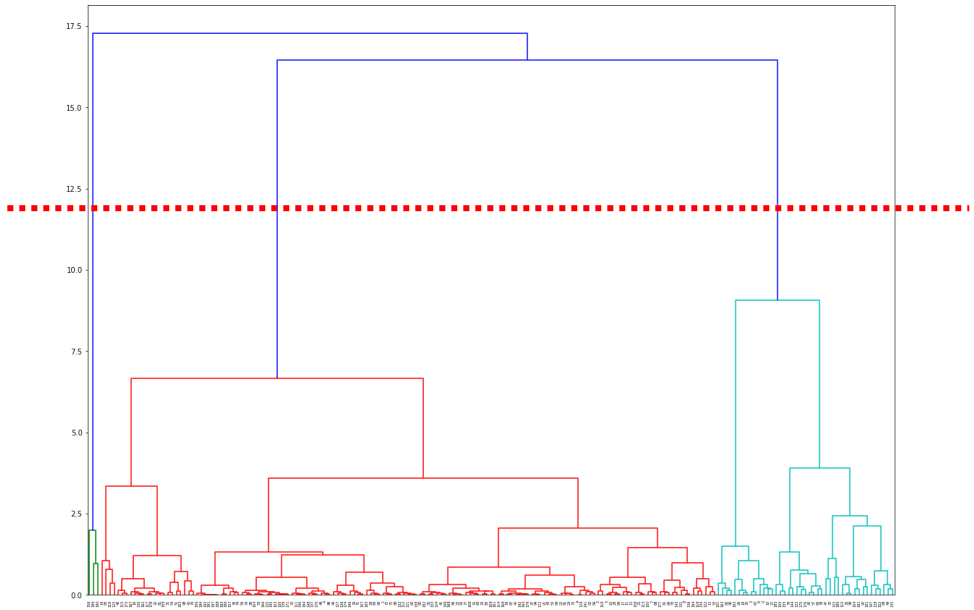
ขั้นตอนการจัดกลุ่มด้วยวิธี Clustering

2. เลือกจำนวนกลุ่มที่ดีที่สุด

- Within Sum of Squares (WCSS) by Elbow Method



- Hierarchical Clustering



ทั้ง 2 วิธีการ Elbow Method และ Hierarchical Clustering แสดงให้เห็นว่า K=3 จะแก้ไขจำนวนคลัสเตอร์

ขั้นตอนการจัดกลุ่มด้วยวิธี Clustering

3.สร้างโมเดล KMeans และจัดกลุ่มข้อมูล

```
#ทั้ง 2 วิธีการ Elbow Method และ Hierarchical Clustering แสดงให้เห็นว่า K=3 จะแก้ไขจำนวนคลัสเตอร์
```

-สร้างโมเดล KMeans และจัดกลุ่มข้อมูล

```
# สร้างตัวแบบ
clf_final=KMeans(n_clusters=3,init='k-means++',random_state=6)
clf_final.fit(X)
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=6, tol=0.0001, verbose=0)
```

```
# ทำนายกลุ่มข้อมูล
countrywise["Clusters"]=clf_final.predict(X)
```

Country/Region	Confirmed	Recovered	Deaths	Mortality	Recovery	Clusters
India	27894800.00	25454320.00	325972.00	1.17	91.25	1.00
Brazil	16471600.00	14496224.00	461057.00	2.80	88.01	1.00
Turkey	5235978.00	5094279.00	47271.00	0.90	97.29	1.00
Russia	4995613.00	4616422.00	118781.00	2.38	92.41	1.00
Italy	4213055.00	3845087.00	126002.00	2.99	91.27	1.00
Argentina	3732263.00	3288467.00	77108.00	2.07	88.11	1.00
Germany	3684672.00	3479700.00	88413.00	2.40	94.44	1.00
Colombia	3363061.00	3141549.00	87747.00	2.61	93.41	1.00
Iran	2893218.00	2425033.00	79741.00	2.76	83.82	1.00
Poland	2871371.00	2636675.00	73682.00	2.57	91.83	1.00
Mexico	2411503.00	1924865.00	223455.00	9.27	79.82	1.00
Ukraine	2257904.00	2084477.00	52414.00	2.32	92.32	1.00
Peru	1947555.00	1897522.00	68878.00	3.54	97.43	1.00
Indonesia	1809926.00	1659974.00	50262.00	2.78	91.72	1.00
Czech Republic	1660935.00	1617498.00	30101.00	1.81	97.38	1.00
US	33251939.00	0.00	594306.00	1.79	0.00	2.00
France	5719877.00	390878.00	109518.00	1.91	6.83	2.00
UK	4496823.00	15481.00	128037.00	2.85	0.34	2.00
Spain	3668658.00	150376.00	79905.00	2.18	4.10	2.00
Netherlands	1671967.00	26810.00	17889.00	1.07	1.60	2.00
Sweden	1068473.00	0.00	14451.00	1.35	0.00	2.00
Belgium	1059763.00	0.00	24921.00	2.35	0.00	2.00
Serbia	712046.00	0.00	6844.00	0.96	0.00	2.00
Switzerland	693023.00	317600.00	10805.00	1.56	45.83	2.00
Greece	400395.00	93764.00	12024.00	3.00	23.42	2.00
Ireland	254870.00	23364.00	4941.00	1.94	9.17	2.00
Honduras	236952.00	84389.00	6296.00	2.66	35.61	2.00
Thailand	151842.00	26873.00	988.00	0.65	17.70	2.00
Norway	124655.00	17998.00	783.00	0.63	14.44	2.00
Finland	92244.00	46000.00	948.00	1.03	49.87	2.00
Yemen	6731.00	3399.00	1319.00	19.60	50.50	0.00
MS Zaandam	9.00	7.00	2.00	22.22	77.78	0.00
Vanuatu	4.00	3.00	1.00	25.00	75.00	0.00

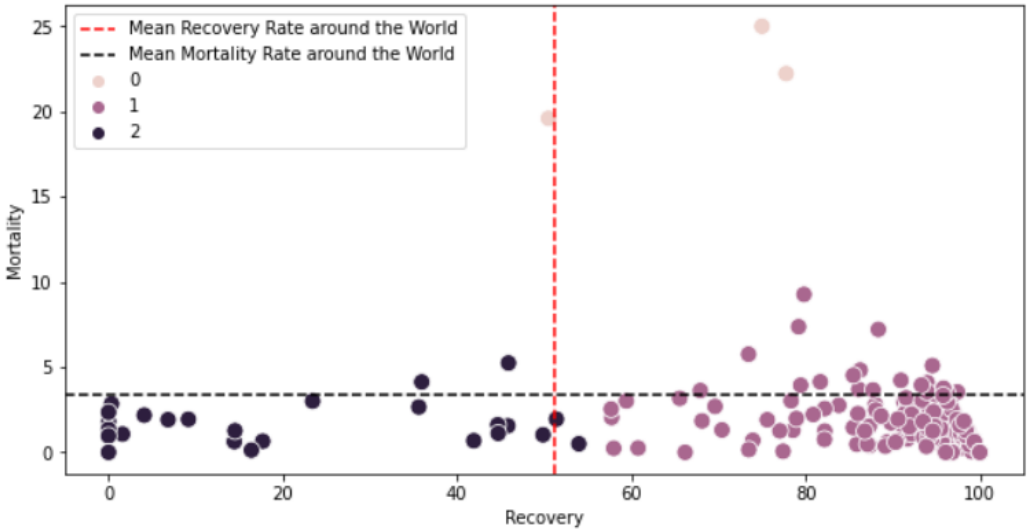
Summary of Clusters

แสดงแค่ Top – 15

ของแต่ละกลุ่ม

ขั้นตอนการจัดกลุ่มด้วยวิธี Clustering

4. Visualization ผลการจัดกลุ่มข้อมูล



- Cluster 0 คือรัฐบาลบริหารงานไม่ดี และประชาชนไม่ดูแลตัวเอง กลุ่มประเทศที่มีอัตราการเสียชีวิตสูงเมื่อเทียบกับกลุ่มอื่นและมีอัตราการฟื้นตัวที่ดีพอสมควร
- Cluster 1 คือรัฐบาลบริหารงานที่ดี กลุ่มประเทศที่มีอัตราการเสียชีวิตต่ำและอัตราการฟื้นตัวสูงจริงๆ เหล่านี้เป็นชุดของประเทศที่สามารถควบคุม COVID-19 ได้โดยปฏิบัติตามแนวทางการควบคุมการระบาดใหญ่อย่างเข้มงวด
- Cluster 2 คือรัฐบาลบริหารงานไม่ดี แต่ประชาชนดูแลตัวเองได้ กลุ่มประเทศที่มีอัตราการเสียชีวิตต่ำและอัตราการฟื้นตัวที่ต่ำ ประเทศเหล่านี้จำเป็นต้องเร่งอัตราการหมุนเวียนเพื่อเอาตัวรอด บางประเทศมีผู้ติดเชื้อจำนวนมาก แต่อัตราการเสียชีวิตต่ำถือเป็นสัญญาณที่ดี แต่บางประเทศก็พบข้อมูลผิดปรกติตรงที่ข้อมูลไม่มีอัตราการฟื้นตัถ้าแปลความตรงตัวถือว่ากำลังประสบการระบาดครั้งใหญ่ที่เลวร้ายที่สุด

Few Countries belonging to Cluster 0: ['Yemen', 'MS Zaandam', 'Vanuatu']

Few Countries belonging to Cluster 1: ['India', 'Brazil', 'Turkey', 'Russia', 'Italy', 'Argentina', 'Germany', 'Colombia', 'Iran', 'Poland', 'Mexico', 'Ukraine', 'Peru', 'Indonesia', 'Czech Republic']

Few Countries belonging to Cluster 2: ['US', 'France', 'UK', 'Spain', 'Netherlands', 'Sweden', 'Belgium', 'Serbia', 'Switzerland', 'Greece', 'Ireland', 'Honduras', 'Thailand', 'Norway', 'Finland']

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

1. เตรียมข้อมูล

ตัวแบบทำนาย 1วันถัดไป X ต่อเนื่อง

เขียนโคดแปลงข้อมูลให้อยู่ในรูปแบบที่สามารถสร้างตัวแบบทำนายได้

datewise.head() *#ข้อมูลที่ใช่ในสร้างตัวแบบทำนาย*

Date	Confirmed	Recovered	Deaths	Days Since
2020-01-22	557.0	30.0	17.0	0 days
2020-01-23	1097.0	60.0	34.0	1 days
2020-01-24	941.0	39.0	26.0	2 days
2020-01-25	1437.0	42.0	42.0	3 days
2020-01-26	2118.0	56.0	56.0	4 days

แปลงข้อมูล

df *#ข้อมูลที่จะใช้ในการสร้าง model ทำนาย*

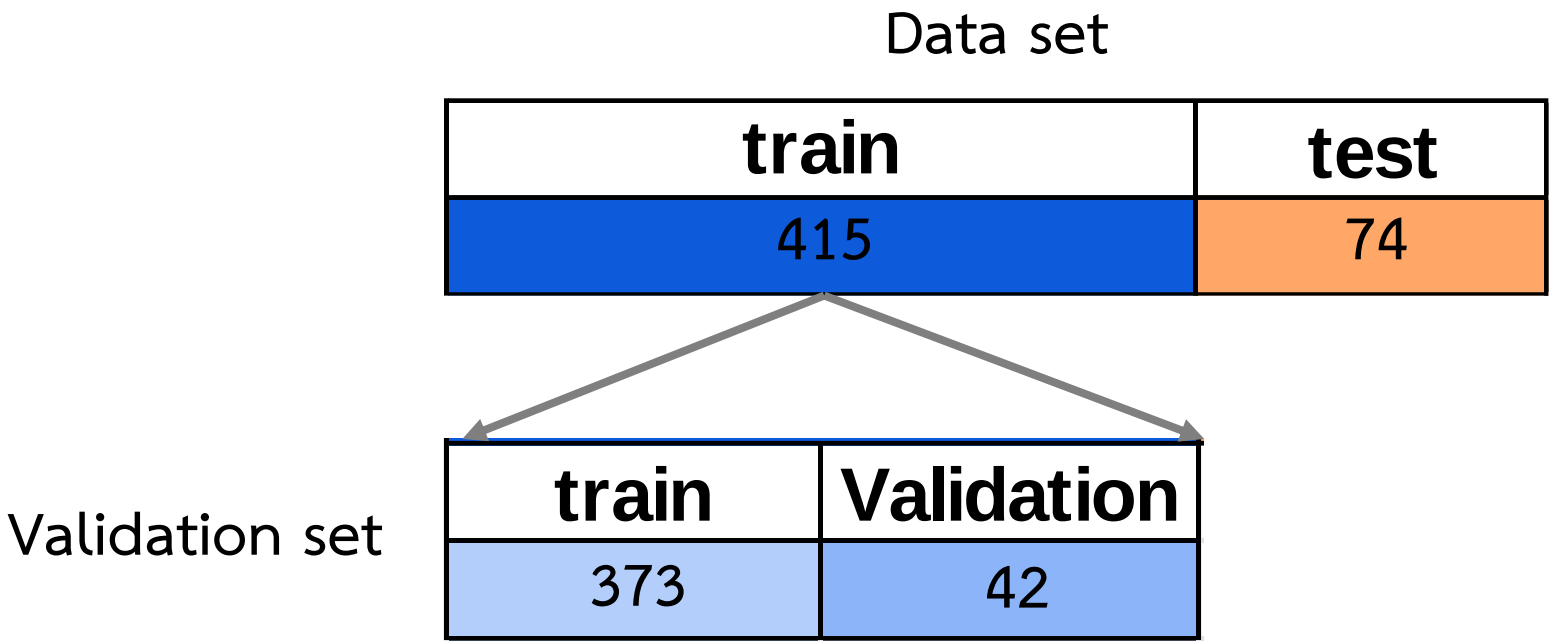
	x1	x2	x3	x4	x5	y
0	557.0	1097.0	941.0	1437.0	2118.0	2927.0
1	1097.0	941.0	1437.0	2118.0	2927.0	5578.0
2	941.0	1437.0	2118.0	2927.0	5578.0	6165.0
3	1437.0	2118.0	2927.0	5578.0	6165.0	8235.0
4	2118.0	2927.0	5578.0	6165.0	8235.0	9925.0
...	...	...	...	...	...	...
484	165182316.0	165808192.0	166385987.0	166862062.0	167316362.0	167848207.0
485	165808192.0	166385987.0	166862062.0	167316362.0	167848207.0	168416423.0
486	166385987.0	166862062.0	167316362.0	167848207.0	168416423.0	168970791.0
487	166862062.0	167316362.0	167848207.0	168416423.0	168970791.0	169470725.0
488	167316362.0	167848207.0	168416423.0	168970791.0	169470725.0	169951560.0

489 rows × 6 columns

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

2. แบ่งข้อมูล Train Test

ตัวแบบทำนาย 1วันถัดไป X ต่อเนื่อง



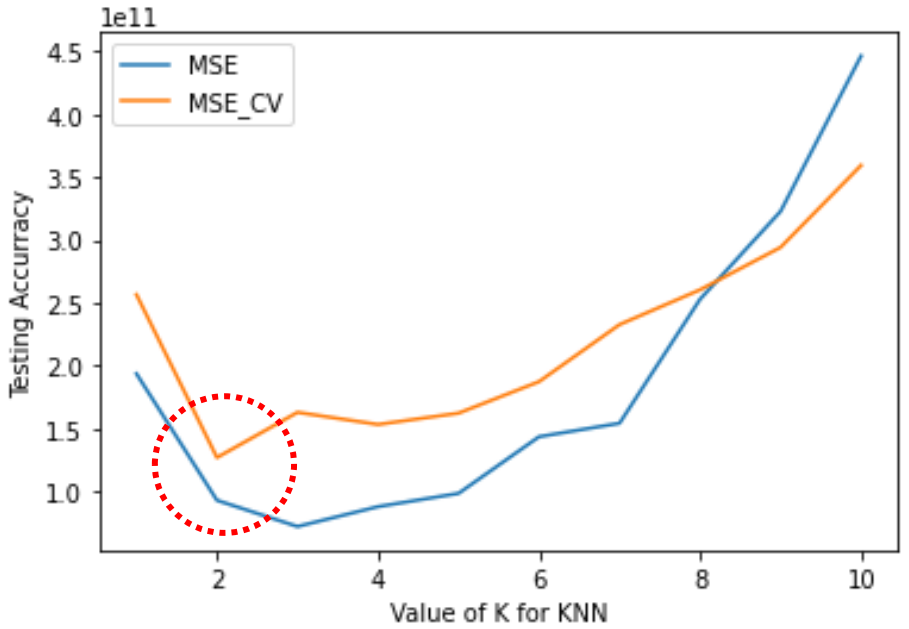
ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

2. Parameter Tuning by k= 10 fold cross validation

ตัวแบบทำนาย 1วันถัดไป X ต่อเนื่อง

Fine –Tune: n_neighbors ตั้งแต่ 1-10

```
k = 1 MSE : 193769094910.6905 MSE_CV: 256730673883.0252
k = 2 MSE : 93034867965.6012 MSE_CV: 127045583820.4504
k = 3 MSE : 72198416402.88655 MSE_CV: 163022105379.16443
k = 4 MSE : 88003205440.40923 MSE_CV: 153382976298.98386
k = 5 MSE : 98532341849.42876 MSE_CV: 162362465641.0729
k = 6 MSE : 143640253021.48254 MSE_CV: 187448918232.9581
k = 7 MSE : 154348293700.59406 MSE_CV: 232867846938.8461
k = 8 MSE : 253022060824.8285 MSE_CV: 260541334342.70148
k = 9 MSE : 322934449746.8035 MSE_CV: 294297659736.3819
k = 10 MSE : 446640582497.32794 MSE_CV: 359558746773.3213
```



แสดงให้เห็นว่า : n_neighbors =2 ดีที่สุด

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

COVID19

3. Training model by best parametor

4. Prediction

```
print("Mean Squared Error = ", mean_squared_error(predict_df.Dependent_Test, predict_df.Dependent_Predicted))
print("Root Mean Squared Error = ", np.sqrt(mean_squared_error(predict_df.Dependent_Test, predict_df.Dependent_Predicted)))
#We see that the mse value decreases in the use of optimum parameters.
#เราเห็นว่าค่า mse ลดลงเมื่อใช้พารามิเตอร์ที่เหมาะสมที่สุด
```

```
Mean Squared Error =  93034867965.6012
Root Mean Squared Error =  305016.1765638032
```

```
r2_score_full = r2_score(predict_df_full.Dependent_Test, predict_df_full.Dependent_Predicted)
r2_score_full
```

```
0.9999892097275556
```

	Dependent_Test	Dependent_Predicted
156	10706794.0	10687694.5
209	23459139.0	23463506.5
276	45116309.0	45130309.5
351	91689011.0	91710451.0
345	87329386.0	87008057.5
...	...	...
204	22174876.0	22185147.5
216	25262187.0	25417079.5
235	30555734.0	30539109.0
84	2471984.0	2474970.5
34	88368.0	85068.5

74 rows × 2 columns

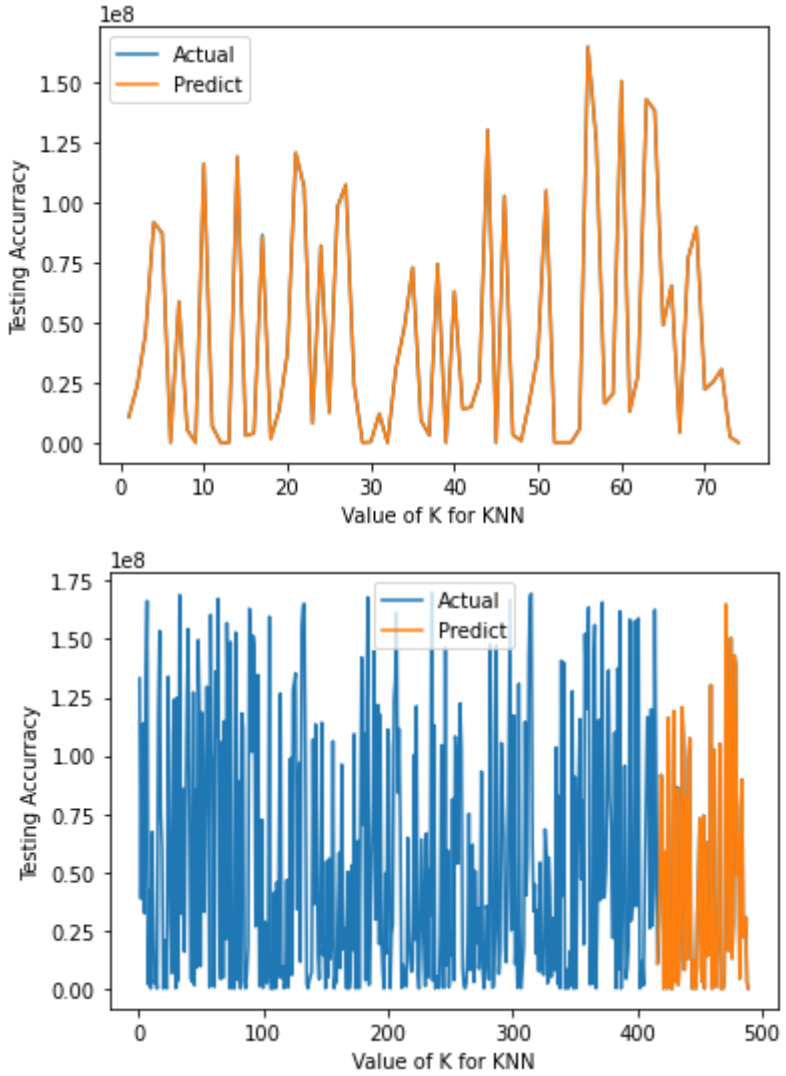
ตัวแบบทำนาย 1วันถัดไป X ต่อเนื่อง

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

4. Prediction

ตัวแบบทำนาย 1วันถัดไป

X ต่อเนื่อง



	Dependent_Test	Dependent_Predicted
156	10706794.0	10687694.5
209	23459139.0	23463506.5
276	45116309.0	45130309.5
351	91689011.0	91710451.0
345	87329386.0	87008057.5
...	...	...
204	22174876.0	22185147.5
216	25262187.0	25417079.5
235	30555734.0	30539109.0
84	2471984.0	2474970.5
34	88368.0	85068.5

74 rows × 2 columns

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

4. Prediction 74 วันถัดไป

ตัวแบบทำนาย 1วันถัดไป

X ต่อเนื่อง

ชุดข้อมูล testset 74 วันสุดท้ายของชุดข้อมูล

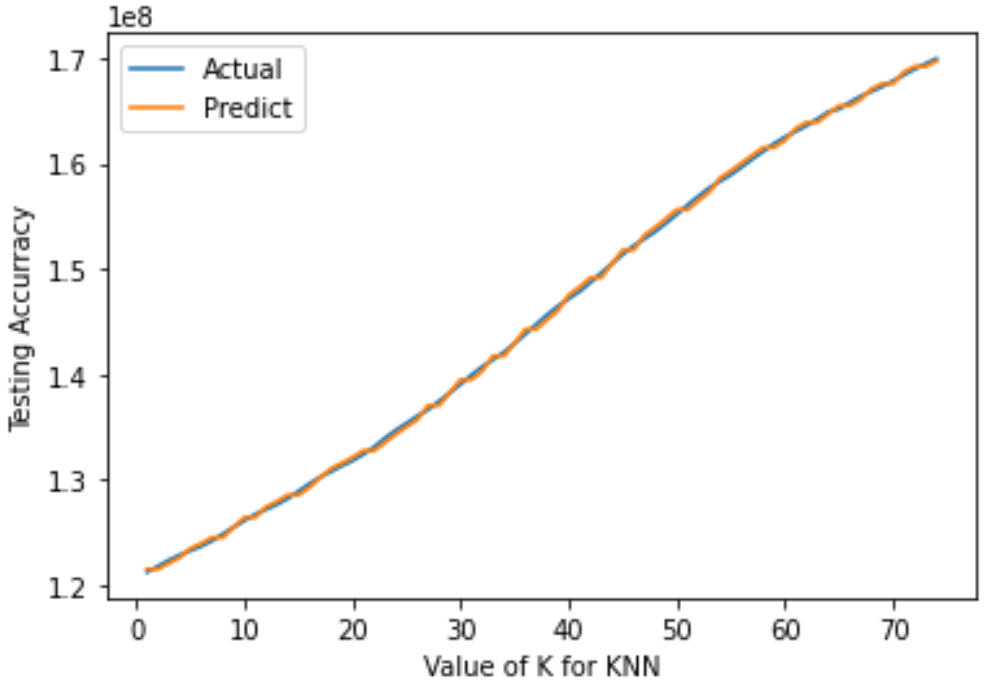
```
predict_df_Exp = pd.DataFrame({"Dependent_Test" : Y_Test, "Dependent_Predicted" : Y_pred})
predict_df_Exp
```

	Dependent_Test	Dependent_Predicted
415	121238378.0	121515239.5
416	121792101.0	121515239.5
417	122353780.0	122072940.5
418	122852600.0	122603190.0
419	123275915.0	123484325.5
...	...	...
484	167848207.0	167582284.5
485	168416423.0	168693607.0
486	168970791.0	169220758.0
487	169470725.0	169220758.0
488	169951560.0	169711142.5

74 rows × 2 columns

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

4. Prediction 74 วันถัดไป



ความแม่นยำในการทำนาย 74 วันถัดไป

```
print("Mean Squared Error = ", mean_squared_error(predict_df.Dependent_Test, predict_df.Dependent_Predicted))
print("Root Mean Squared Error = ", np.sqrt(mean_squared_error(predict_df.Dependent_Test, predict_df.Dependent_Predicted)))
#We see that the mse value decreases in the use of optimum parameters.
#เราเห็นว่าค่า mse ลดลงเมื่อใช้พารามิเตอร์ที่เหมาะสมที่สุด
```

```
Mean Squared Error =  93034867965.6012
Root Mean Squared Error =  305016.1765638032
```

```
r2_score_Exp = r2_score(predict_df_Exp.Dependent_Test, predict_df_Exp.Dependent_Predicted)
r2_score_Exp
```

```
0.9995247159784287
```

ตัวแบบทำนาย 1วันถัดไป

X ต่อเนื่อง

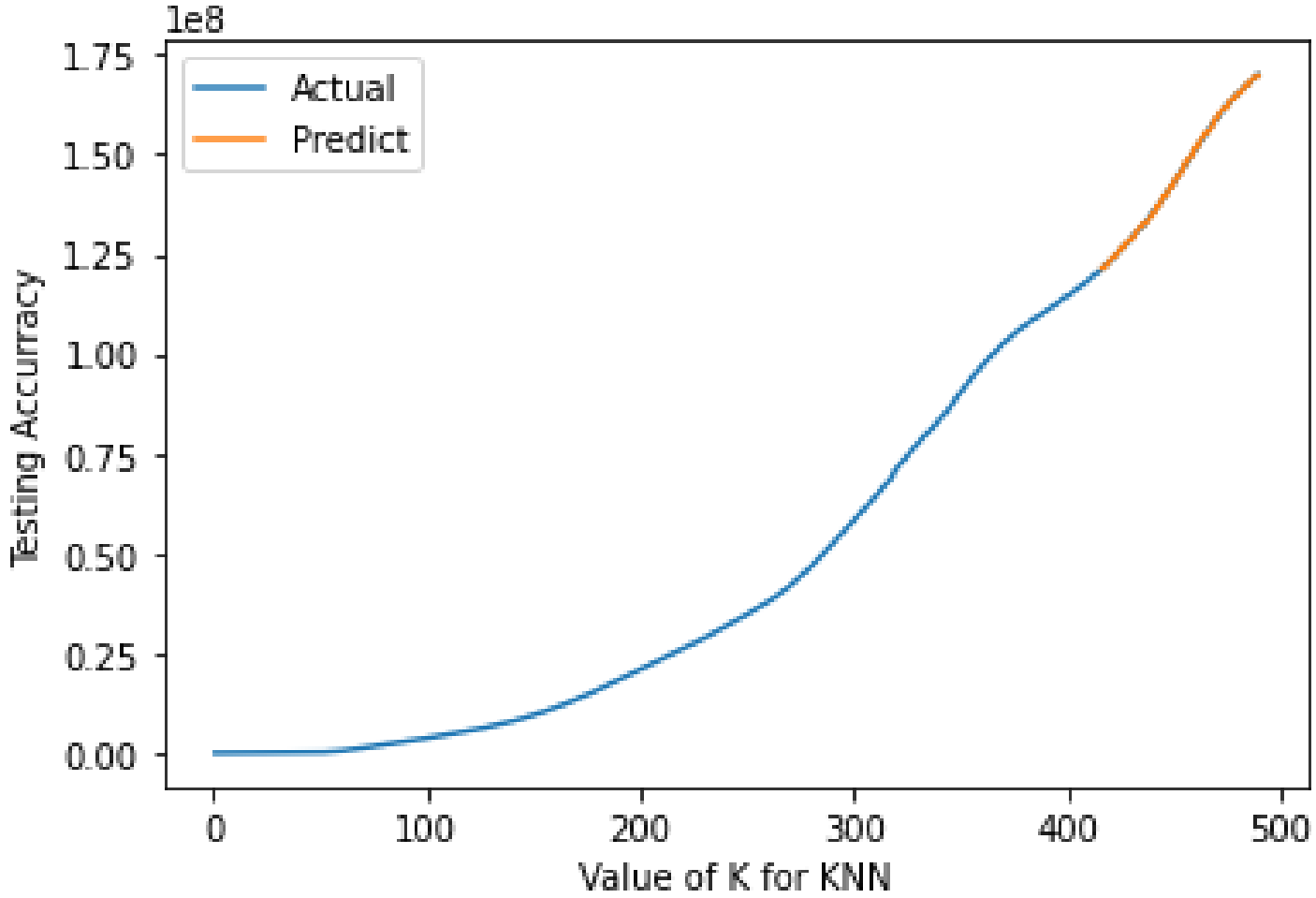
ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

4. Prediction 74 วันถัดไป

ตัวแบบทำนาย 1วันถัดไป

X ต่อเนื่อง

กราฟแสดงการทำนายข้อมูล 74 วันสุดท้าย(74 วันถัดไป)



ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

1. เตรียมข้อมูล

ตัวแบบทำนาย 5 วันถัดไป X ไม่ต่อเนื่อง

เขียนโคดแปลงข้อมูลให้อยู่ในรูปแบบที่สามารถสร้างตัวแบบทำนายได้

datewise.head() *#ข้อมูลที่ใช้ในสร้างตัวแบบทำนาย*

Date	Confirmed	Recovered	Deaths	Days Since
2020-01-22	557.0	30.0	17.0	0 days
2020-01-23	1097.0	60.0	34.0	1 days
2020-01-24	941.0	39.0	26.0	2 days
2020-01-25	1437.0	42.0	42.0	3 days
2020-01-26	2118.0	56.0	56.0	4 days

แปลงข้อมูล

df *#ข้อมูลที่จะใช้ในการสร้าง model ทำนาย*

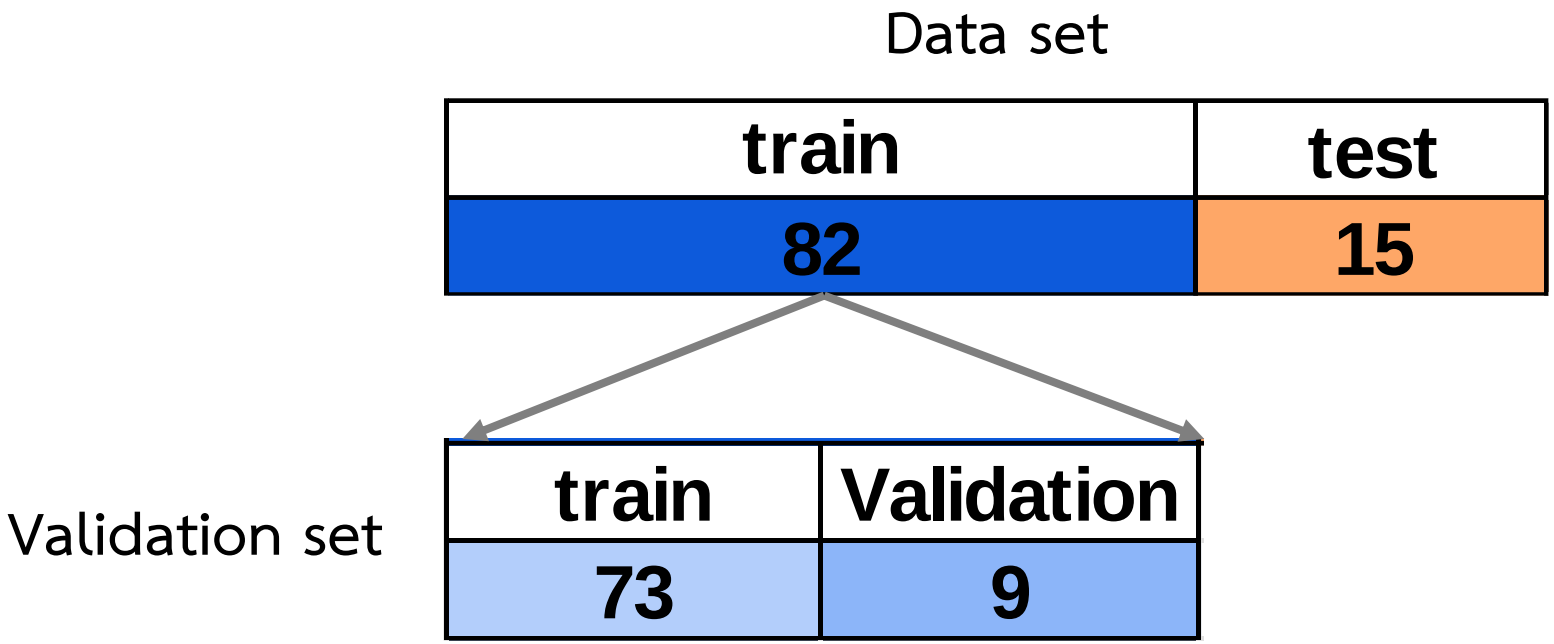
	x1	x2	x3	x4	x5	y
0	557.0	1097.0	941.0	1437.0	2118.0	12038.0
1	2927.0	5578.0	6165.0	8235.0	9925.0	30818.0
2	12038.0	16787.0	19881.0	23892.0	27636.0	44803.0
3	30818.0	34392.0	37121.0	40151.0	42763.0	71226.0
4	44803.0	45222.0	60370.0	66887.0	69033.0	76843.0
...	...	...	...	...	...	...
92	147872402.0	148716872.0	149622864.0	150520466.0	151399480.0	156070729.0
93	152196159.0	152870507.0	153552097.0	154359533.0	155200757.0	159690774.0
94	156070729.0	156902287.0	157688833.0	158330372.0	158952301.0	163069932.0
95	159690774.0	160450873.0	161176148.0	161894208.0	162521726.0	165808192.0
96	163069932.0	163609626.0	164231811.0	164902903.0	165182316.0	168416423.0

97 rows × 6 columns

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

2. แบ่งข้อมูล Train Test

ตัวแบบทำนาย 5 วันถัดไป X ไม่ต่อเนื่อง



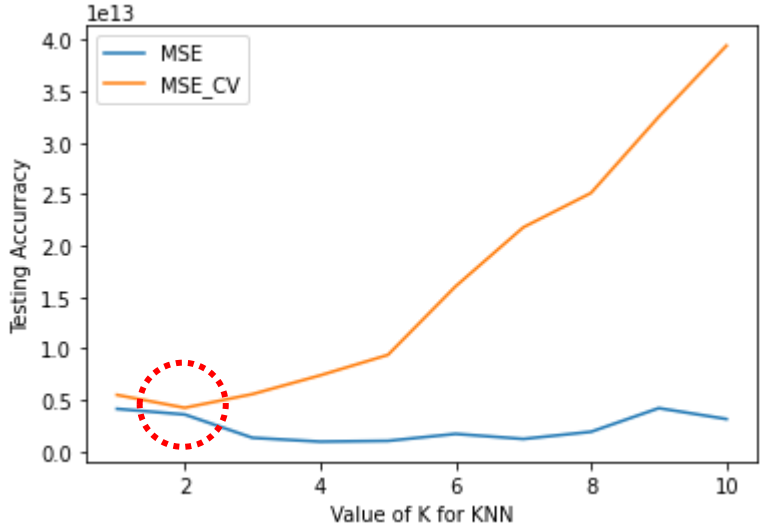
ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

2. Parameter Tuning by k= 10 fold cross validation

ตัวแบบทำนาย 5 วันถัดไป X ไม่ต่อเนื่อง

Fine –Tune: n_neighbors ตั้งแต่ 1-10

```
k = 1 MSE : 4141180673262.3335 MSE_CV: 5495772699781.691
k = 2 MSE : 3592262372508.6113 MSE_CV: 4244060419413.8203
k = 3 MSE : 1328062236487.3225 MSE_CV: 5572258255480.457
k = 4 MSE : 953159297490.0903 MSE_CV: 7396795888320.467
k = 5 MSE : 1028868222365.1128 MSE_CV: 9384080573200.3
k = 6 MSE : 1706645136790.0244 MSE_CV: 16053182859910.365
k = 7 MSE : 1216414465871.7598 MSE_CV: 21788661117660.6
k = 8 MSE : 1912272193442.481 MSE_CV: 25131314581822.023
k = 9 MSE : 4209522162085.2056 MSE_CV: 32531352318806.523
k = 10 MSE : 3151454428500.633 MSE_CV: 39454621723575.21
```



แสดงให้เห็นว่า : n_neighbors =2 ดีที่สุด

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

COVID19

3. Training model by best parametor

4. Prediction

```
print("Mean Squared Error = ", mean_squared_error(predict_df.Dependent_Test, predict_df.Dependent_Predicted))
print("Root Mean Squared Error = ", np.sqrt(mean_squared_error(predict_df.Dependent_Test, predict_df.Dependent_Predicted)))
#We see that the mse value decreases in the use of optimum parameters.
#เราเห็นว่าค่า mse ลดลงเมื่อใช้พารามิเตอร์ที่เหมาะสมที่สุด
```

```
Mean Squared Error =  3592262372508.6113
Root Mean Squared Error =  1895326.4553919495
```

```
r2_score_full = r2_score(predict_df_full.Dependent_Test, predict_df_full.Dependent_Predicted)
r2_score_full
```

```
0.9983232907313568
```

ตัวแบบทำนาย 5 วันถัดไป X ไม่ต่อเนื่อง

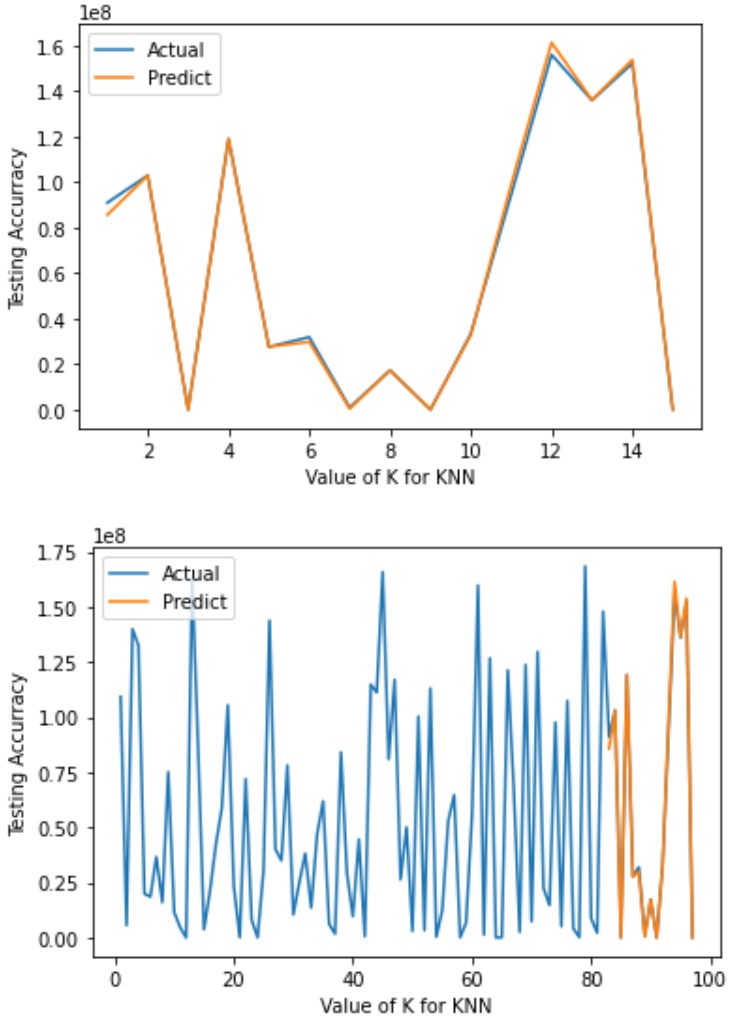
	Dependent_Test	Dependent_Predicted
69	90983503.0	85711002.5
73	103027363.0	102891874.5
2	44803.0	51022.0
81	119062678.0	119077182.5
44	27617704.0	27705395.5
47	31937561.0	29808437.5
12	932440.0	465657.0
36	17326649.0	17312159.5
0	12038.0	51022.0
48	33423047.0	32762228.0
70	94598958.0	98964130.5
92	156070729.0	161380353.0
87	136046628.0	136175300.0
91	152196159.0	153781588.0
5	81397.0	98071.5

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธี KNeighborsRegressor

4. Prediction

ตัวแบบทำนาย 5 วันถัดไป

X ไม่ต่อเนื่อง



	Dependent_Test	Dependent_Predicted
69	90983503.0	85711002.5
73	103027363.0	102891874.5
2	44803.0	51022.0
81	119062678.0	119077182.5
44	27617704.0	27705395.5
47	31937561.0	29808437.5
12	932440.0	465657.0
36	17326649.0	17312159.5
0	12038.0	51022.0
48	33423047.0	32762228.0
70	94598958.0	98964130.5
92	156070729.0	161380353.0
87	136046628.0	136175300.0
91	152196159.0	153781588.0
5	81397.0	98071.5

ขั้นตอนการคาดการณ์และพยากรณ์อนุกรมเวลาด้วยวิธีทางสถิติ

COVID19

- **Time Series Forecasting:** ตัวแบบทางสถิติ
  - Facebook's Prophet Model
  - Holt's Linear
  - Auto Regressive Model (AR)
  - Holt's Winter Model
  - ARIMA Model
  - SARIMA Model
  - Moving Average Model (MA)
  - Linear Regression
  - Polynomial Regression

	Model Name	Root Mean Squared Error	R-Squared
8	Facebook's Prophet Model	1.025041e+06	0.999624
2	Holt's Linear	1.696112e+06	0.848627
4	Auto Regressive Model (AR)	2.350964e+06	0.709175
7	SARIMA Model	2.357813e+06	0.707478
3	Holt's Winter Model	2.594640e+06	0.645763
5	Moving Average Model (MA)	2.901479e+06	0.557026
6	ARIMA Model	3.160138e+06	0.474526
1	Polynomial Regression	2.736296e+07	-38.397214
0	Linear Regression	3.354151e+07	-58.197677

ผลลัพธ์เพิ่มเติม

Link: https://nbviewer.org/github/Peckkie/DPDM2021/blob/main/Mini_Project_COVID19.ipynb

ประโยชน์

- ได้สารสนเทศเกี่ยวกับสถานการณ์การแพร่ระบาดของ COVID-19 จากทั่วโลกเพื่อวางแผนรับมือกับสถานการณ์ในอนาคตเกี่ยวกับ COVID-19 ได้อย่างทันท่วงที

- นำผลการทำนายจำนวนคนที่คาดว่าตรวจโควิดแล้วจะติดโควิด เพื่อใช้ในการวางแผนจัดการเกี่ยวกับการเตรียมเครื่องมือและโรงพยาบาลให้พร้อมรักษาผู้ป่วยที่คาดว่าจะติดเชื้อในอนาคต