Forecasting satellite images

Nirlipta Pande

Mantle Labs

April 2022-July 2022

Problem Statement

Lack of forecasting techniques in remote sensing domain, especially forecasting the next product

- ❖ Don’t effectively exploit the large amounts of data available
- ❖ Lack of utilization of apt computational resources now available

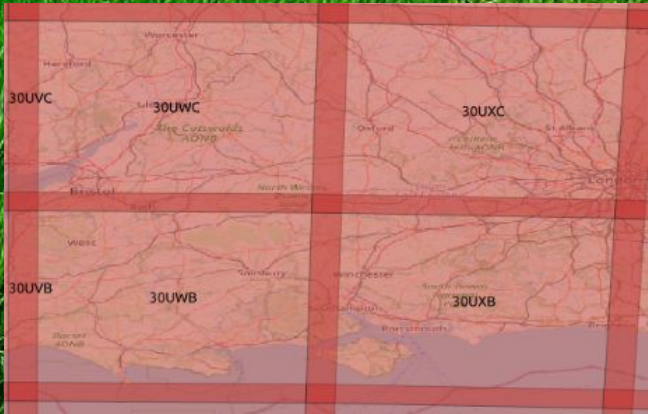
Need to forecast the upcoming product:

- ❖ Gives insight into the near future by showing the ground truth

Time, area and, data under consideration

- ❖ Time period between 01.01.2019 - 31.12.2021
- ❖ Area: Surrey, UK Tile 30UXB (As displayed in the figure below)
- ❖ Data used:
  - ➢ Sentinel 1 and,
  - ➢ Sentinel 2
    - ▪ B02
    - ▪ B03
    - ▪ B04
    - ▪ B05
    - ▪ B06
    - ▪ B07
    - ▪ B08
    - ▪ B11
    - ▪ B12

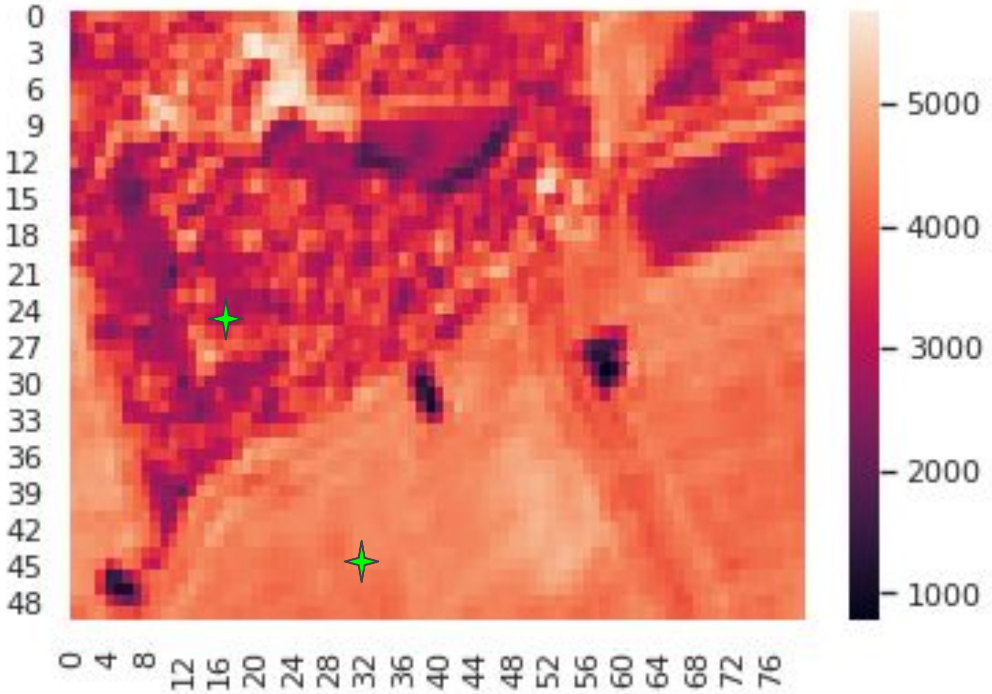
Note: Variations in data and time period in individual cases are mentioned separately



Forecasting Sentinel 1 data

- Data used: Sentinel 1
- Time period: 01.01.2019 - 31.12.2019
- Area: Top left 80 x 50 area of 30UXB tile
- Area under focus (i.e. plotted for every time step)
  - 45_32
  - 25_16
- Models used:
  - Prophet
  - NeuroProphet
  - AutoArima
  - Seasonal Decomposition using LOESS
- Approaches:
  - Ascending and descending orbits separately analyzed
  - Complete data analyzed together

Area under focus
(marked with crosses)



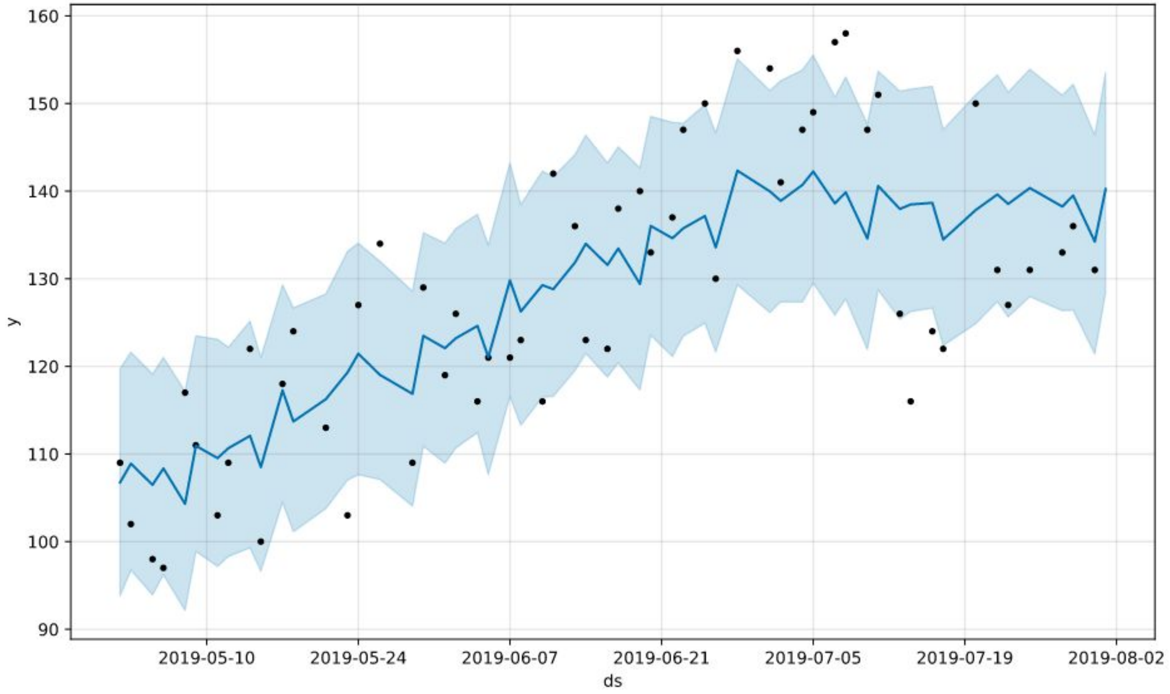
Part 1: Sentinel 1 data

1.1 Prophet by Meta (formerly Facebook)

Methodology and data used

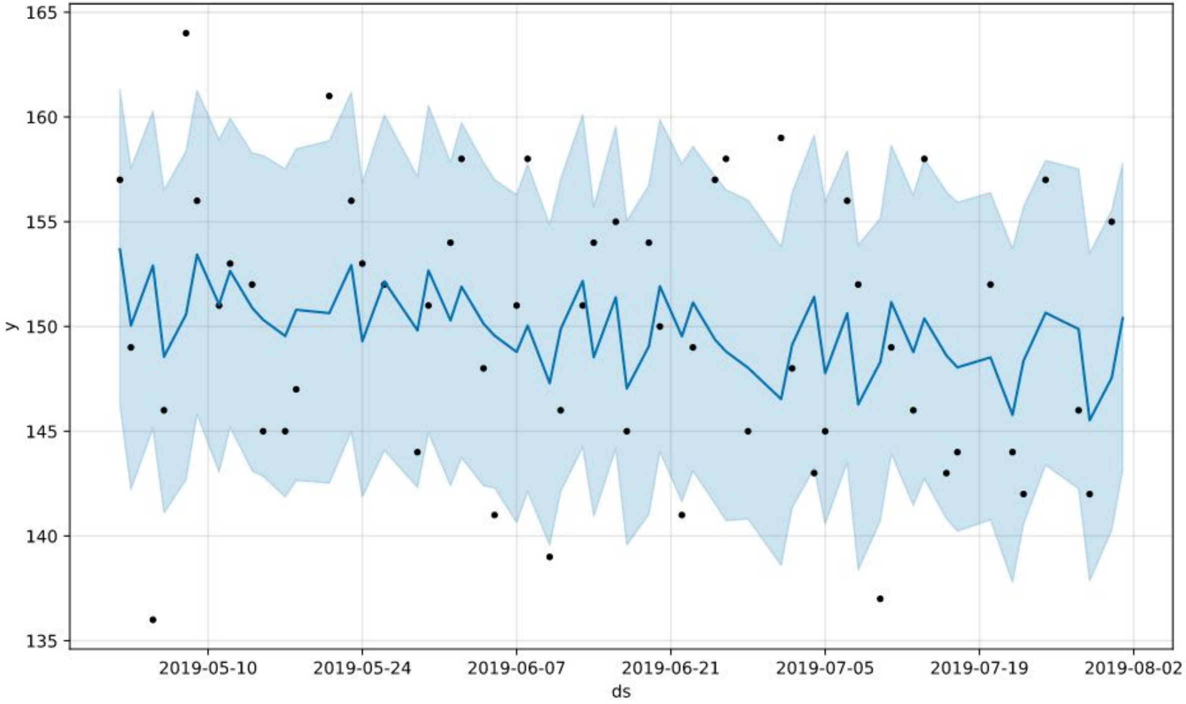
- Prophet by Meta (formerly Facebook)
  - Additive model used as there was no compounding factor adding up
  - It has 4 major parts:
    - An in-built linear trend,
    - The seasonality is modeled using Fourier series ,and
    - Even accounts for weekly components using dummy variables
    - Can account of user provided holidays (essential for data easily changed by human intervention, eg shop sales)
- Sentinel 1
  - Time period 01.01.2019 to 31.12.2019
  - Time period used for training: past 85 days
  - Forecasts obtained at an interval of every 15 days
  - Area considered: 80x50 pixel area of 30UXB (As shown in previous slide)

Input Time Series
(Without ascending descending separation)



Pixel 45_32

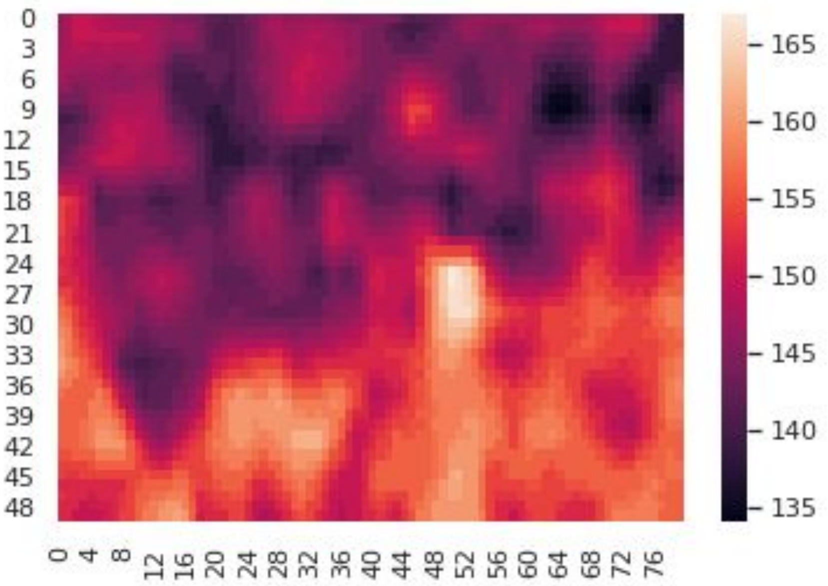
Input Time Series
(Without ascending descending separation)



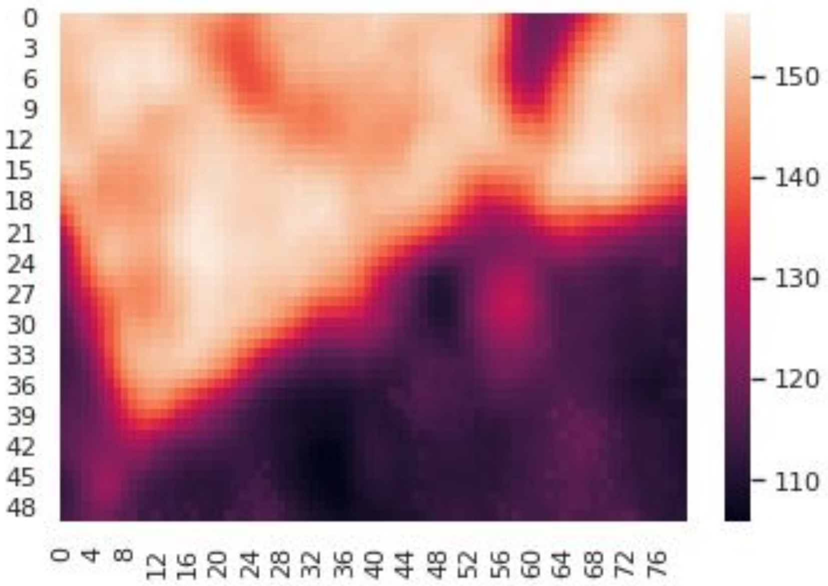
Pixel 25_16

Results (01.08.2019)

Target

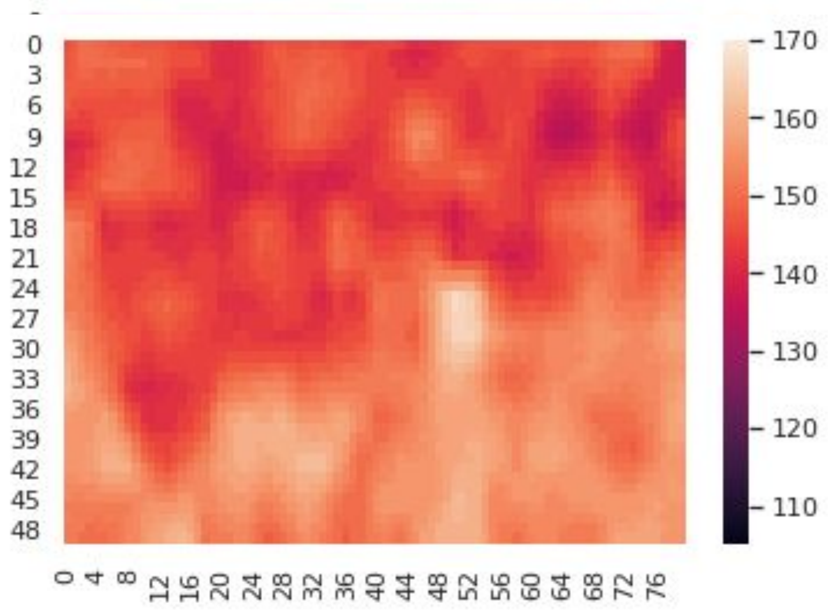


Forecast

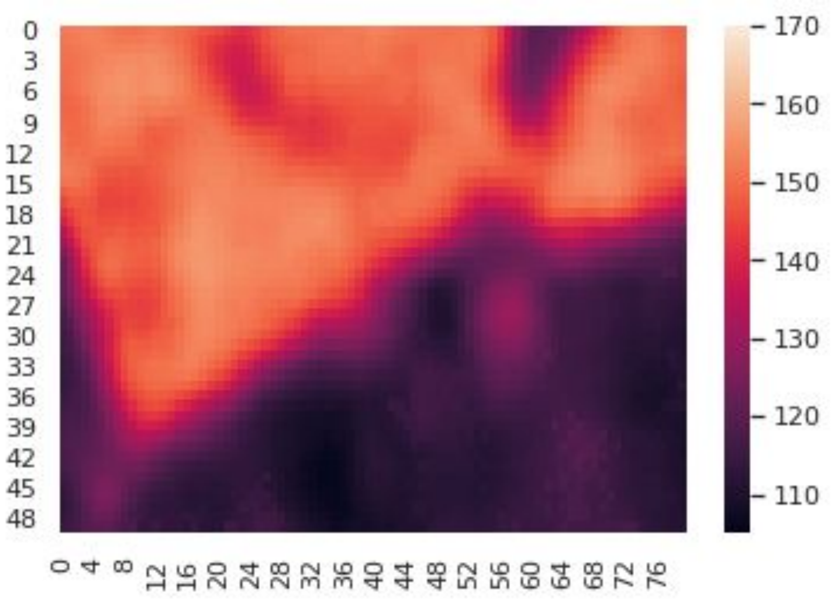


Results (01.08.2019)

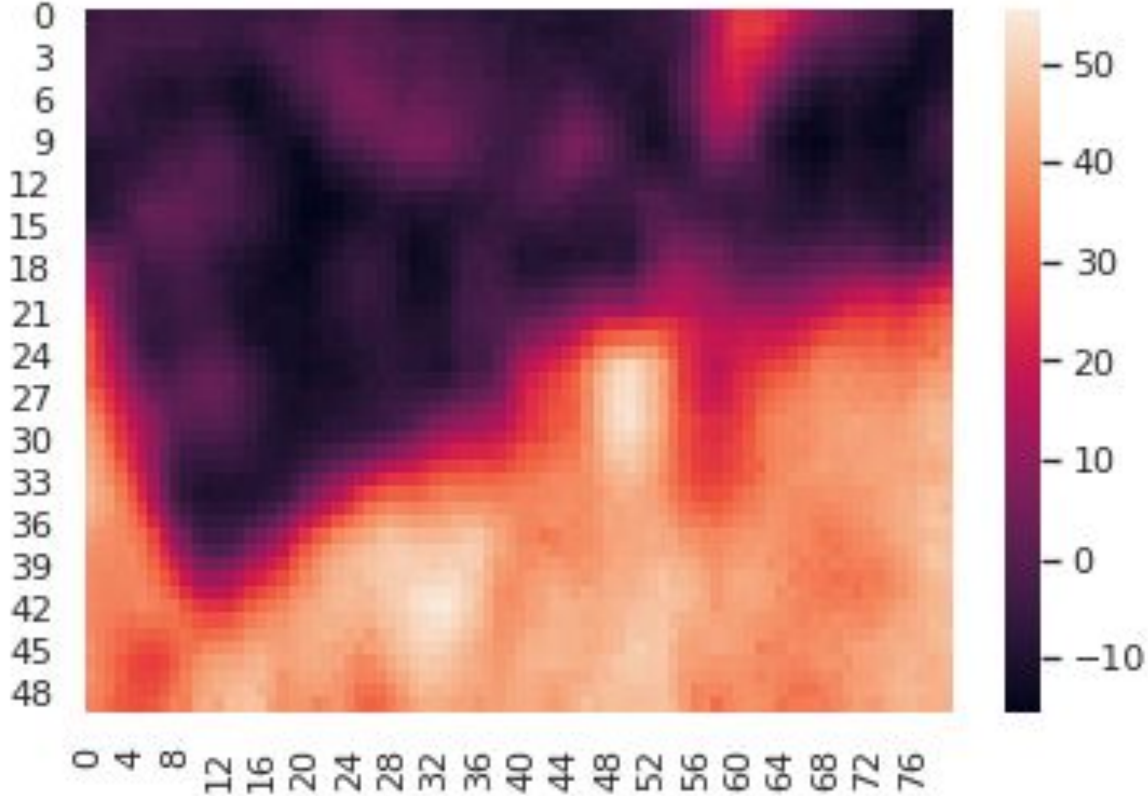
Target



Forecast

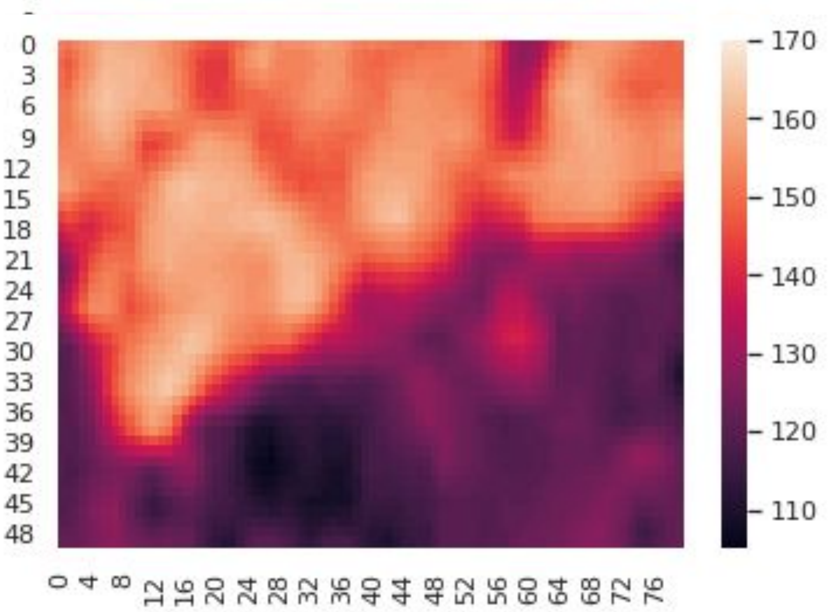


Errors (01.08.2019)

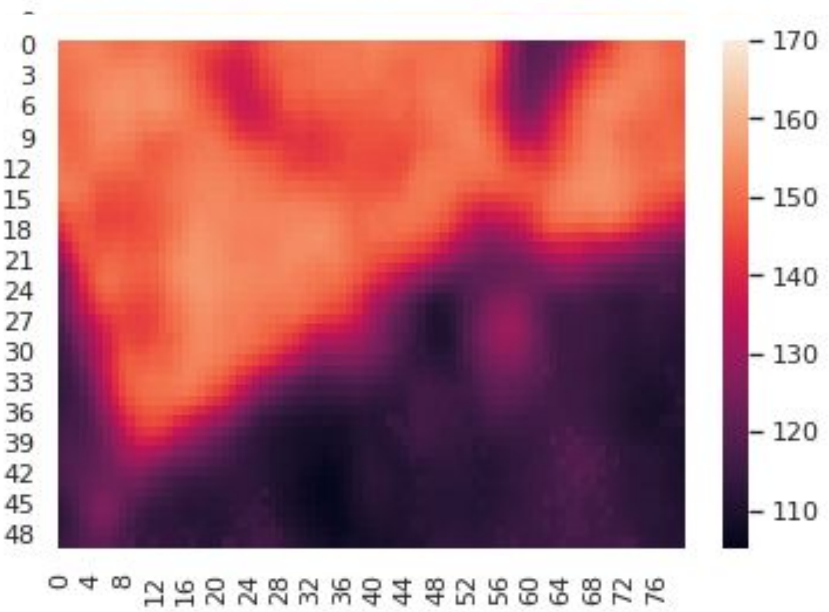


Results (01.08.2019)

Monkey

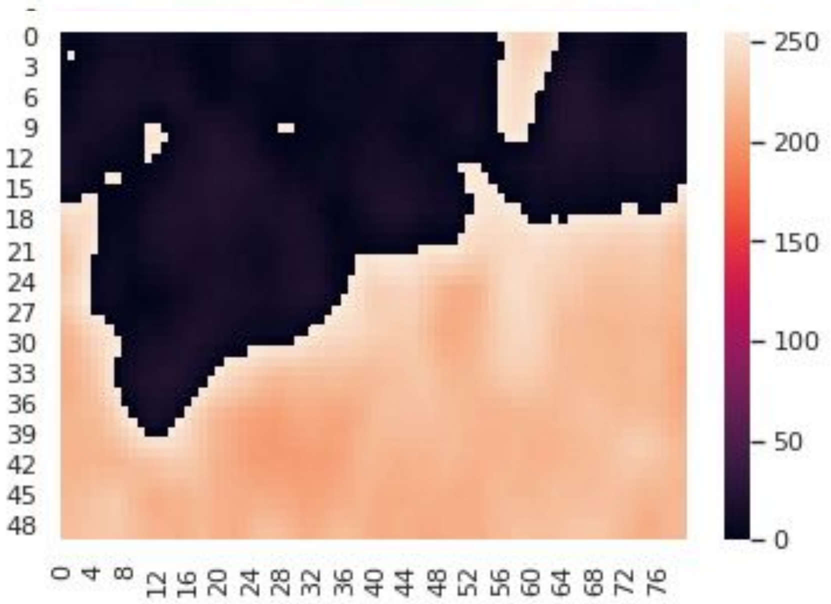


Forecast

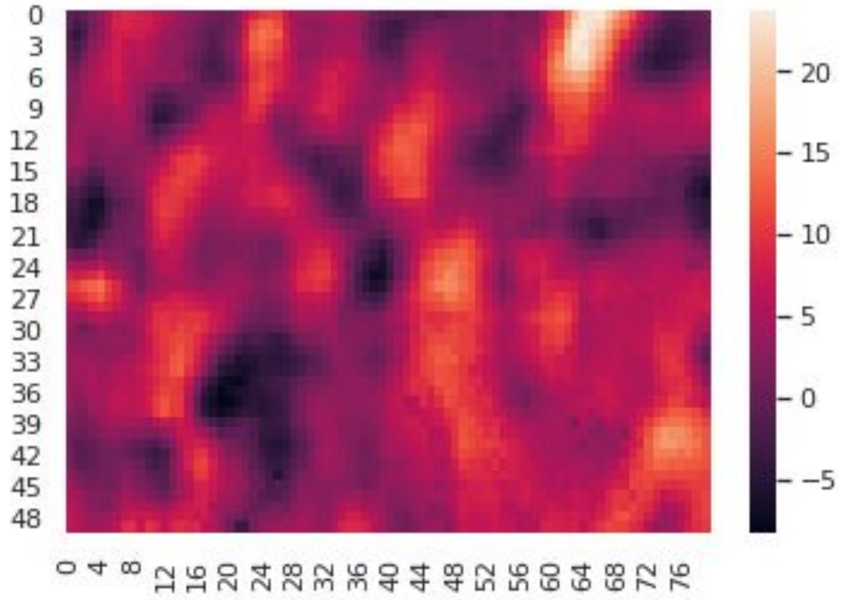


Errors (01.08.2019)

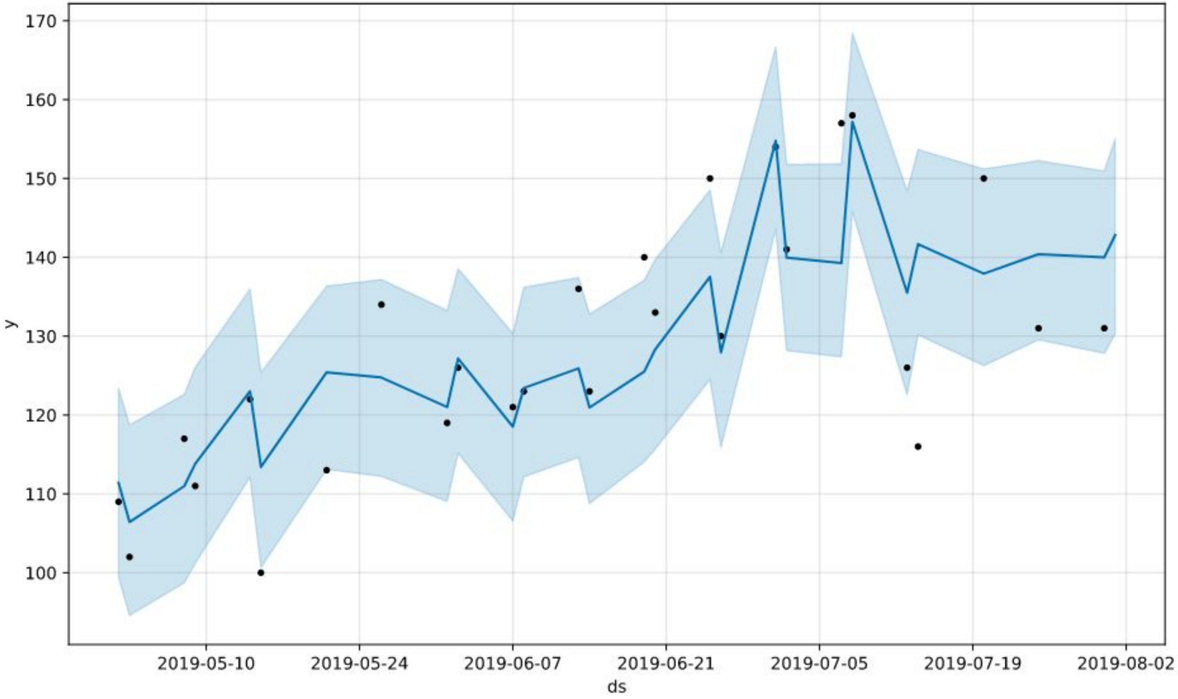
Monkey - Target



Monkey - Forecast

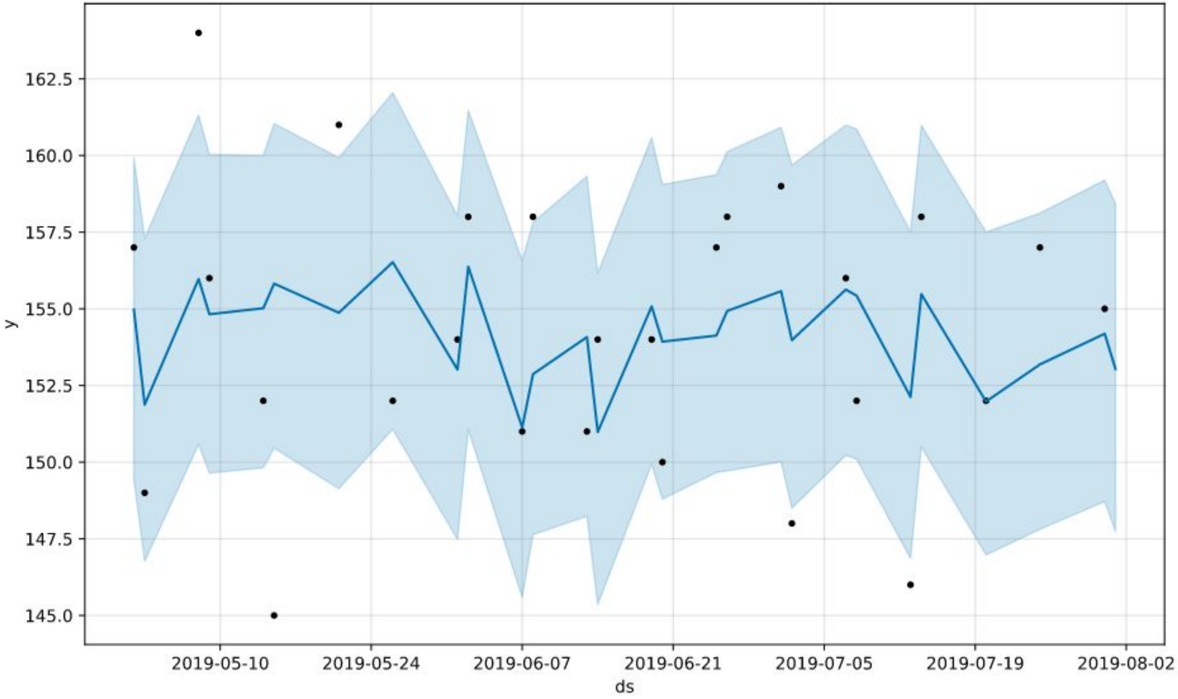


Input Time Series (01.08.2019) (Ascending Orbit only)



Pixel 45_32

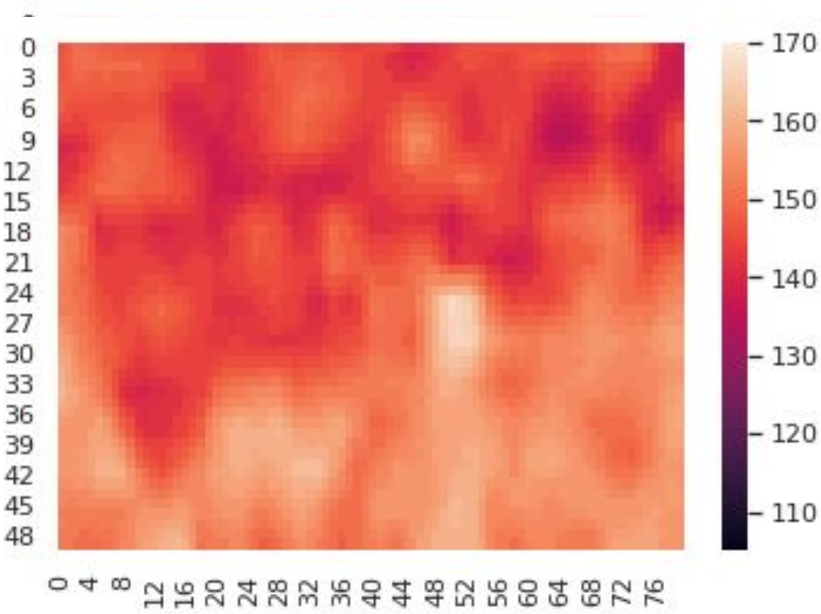
Input Time Series (01.08.2019) (Ascending Orbit only)



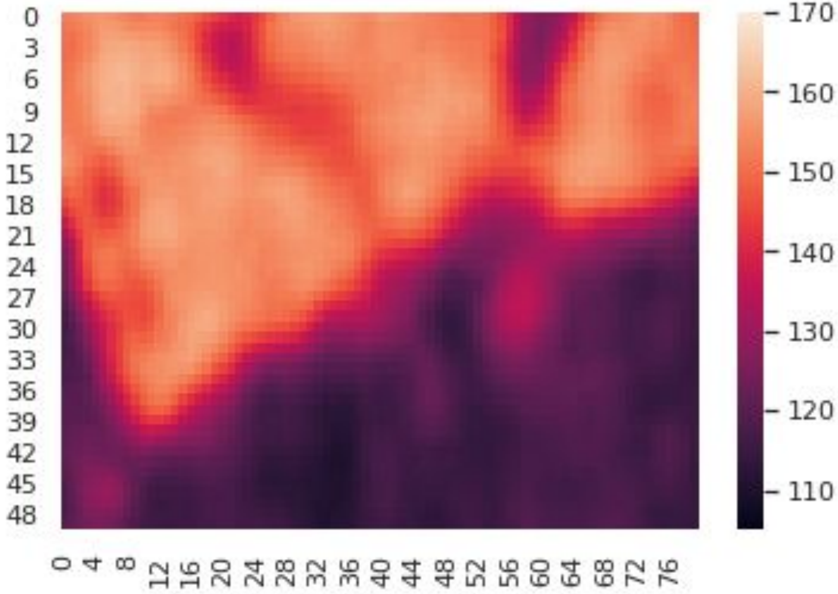
Pixel 25_16

Results

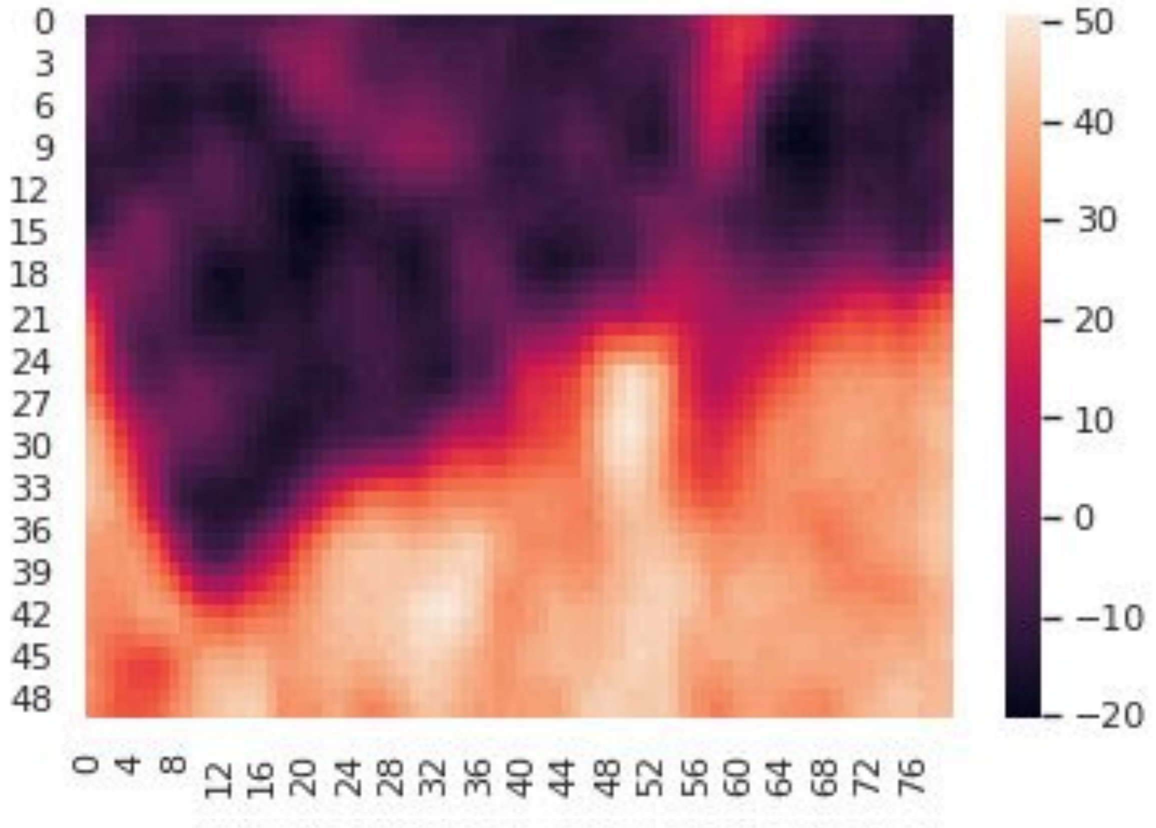
Target



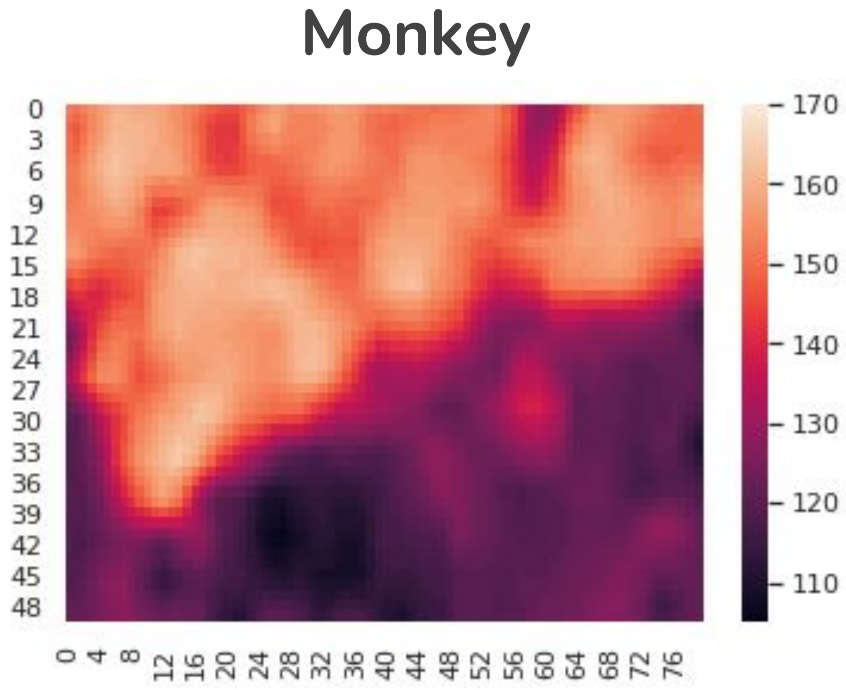
Forecast

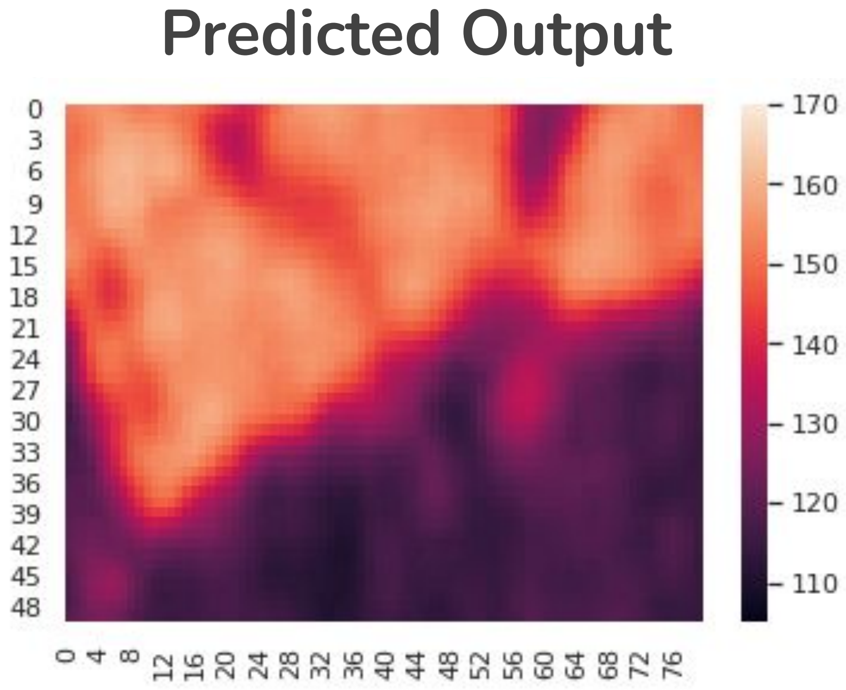


Errors (01.08.2019)



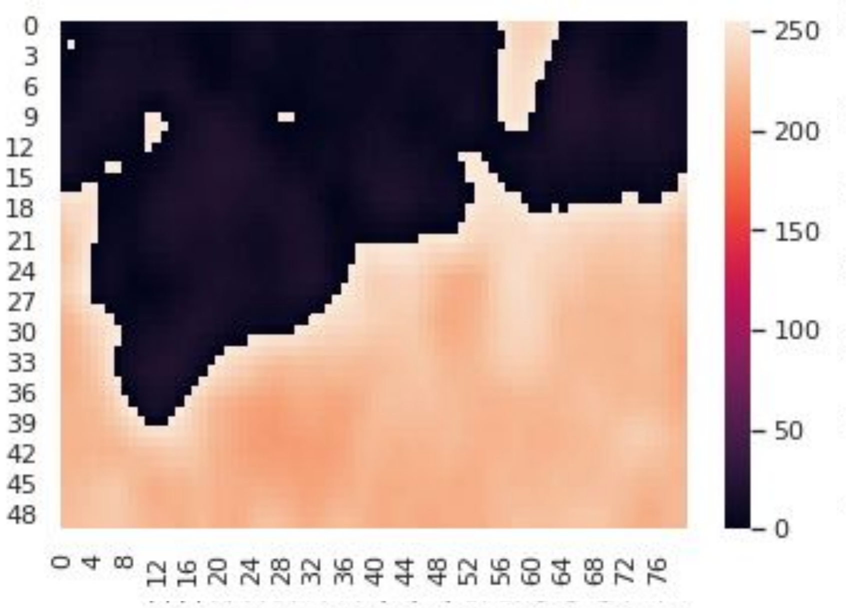
Results


Monkey
0
3
6
9
12
15
18
21
24
27
30
33
36
39
42
45
48
0
4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
76
170
160
150
140
130
120
110

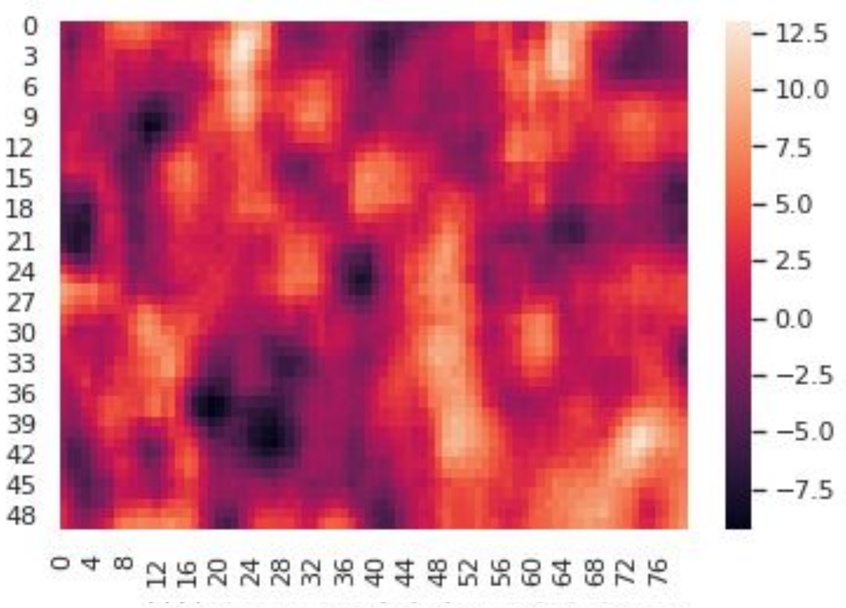

Predicted Output
0
3
6
9
12
15
18
21
24
27
30
33
36
39
42
45
48
0
4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
76
170
160
150
140
130
120
110

Results

Monkey



Forecast



Part 1: Sentinel 1 data

1.2 AutoArima

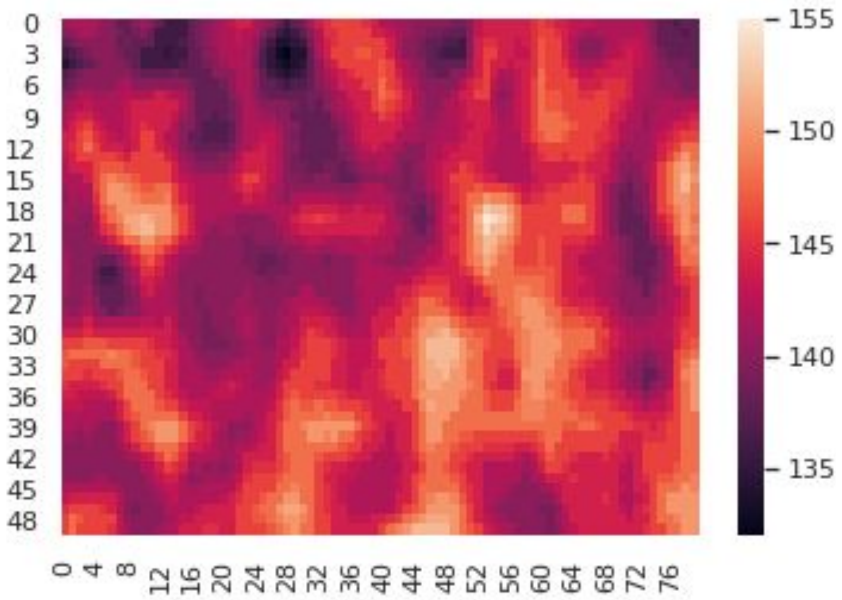
by Rob J. Hyndman et al.

Methodology and data used

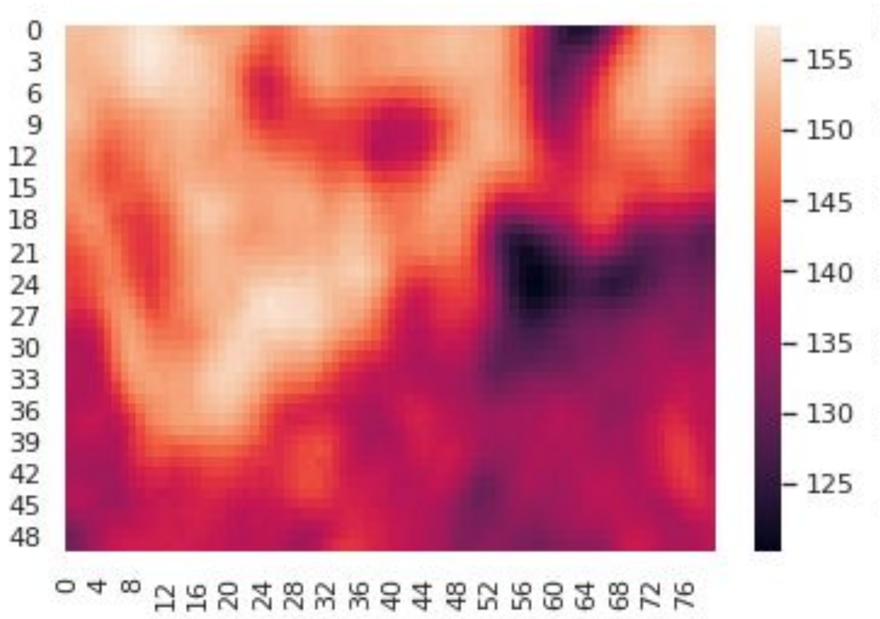
- AutoArima by Rob J. Hyndman et al.
  - Variation of the Hyndman-Khandakar algorithm
  - An ARIMA model is obtained by combining
    - unit root tests,
    - minimisation of the Akaike information criterion with correlation (AICc) ,and
    - Maximum likelihood estimation (MLE)
  -
- Sentinel 1
  - Time period 01.01.2019 to 31.12.2019
  - Time period used for training: past 85 days
  - Forecasts obtained at an interval of every 15 days
  - Area considered: 80x50 pixel area of 30UXB (As shown in previous slide)

Results
(27.03.2019)

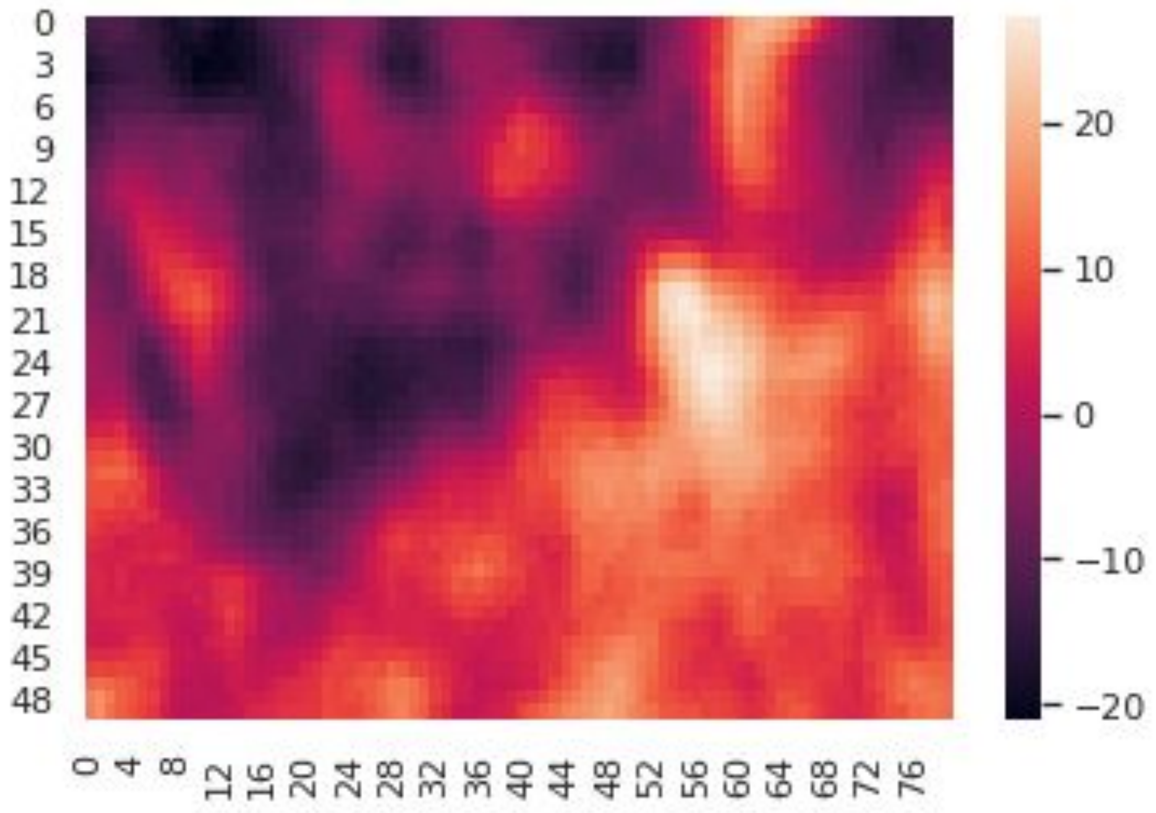
Target



Forecast

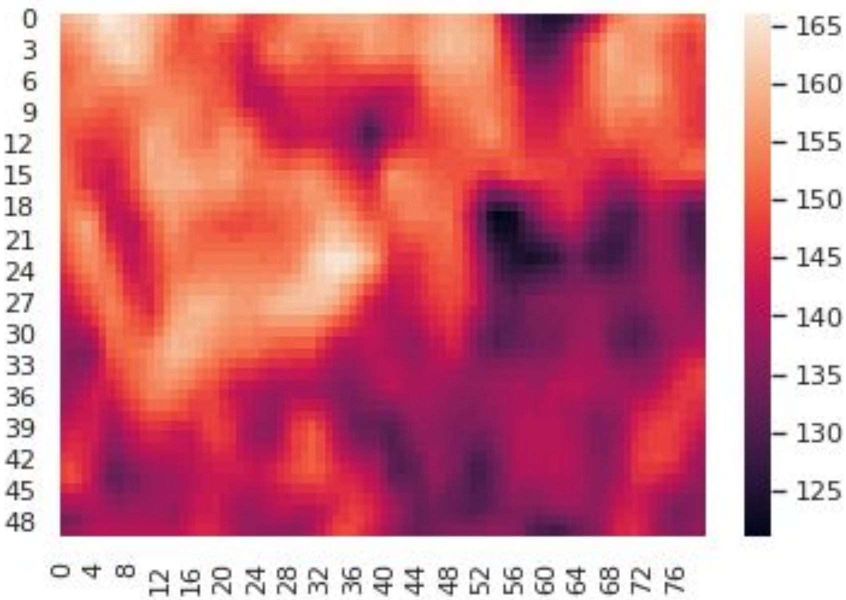


Errors (27.03.2019)

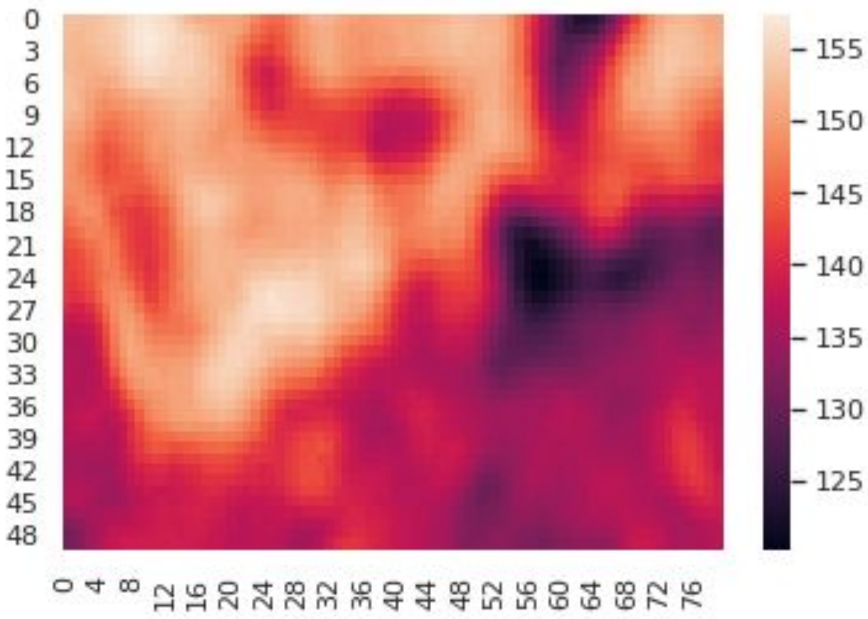


Results
(27.03.2019)

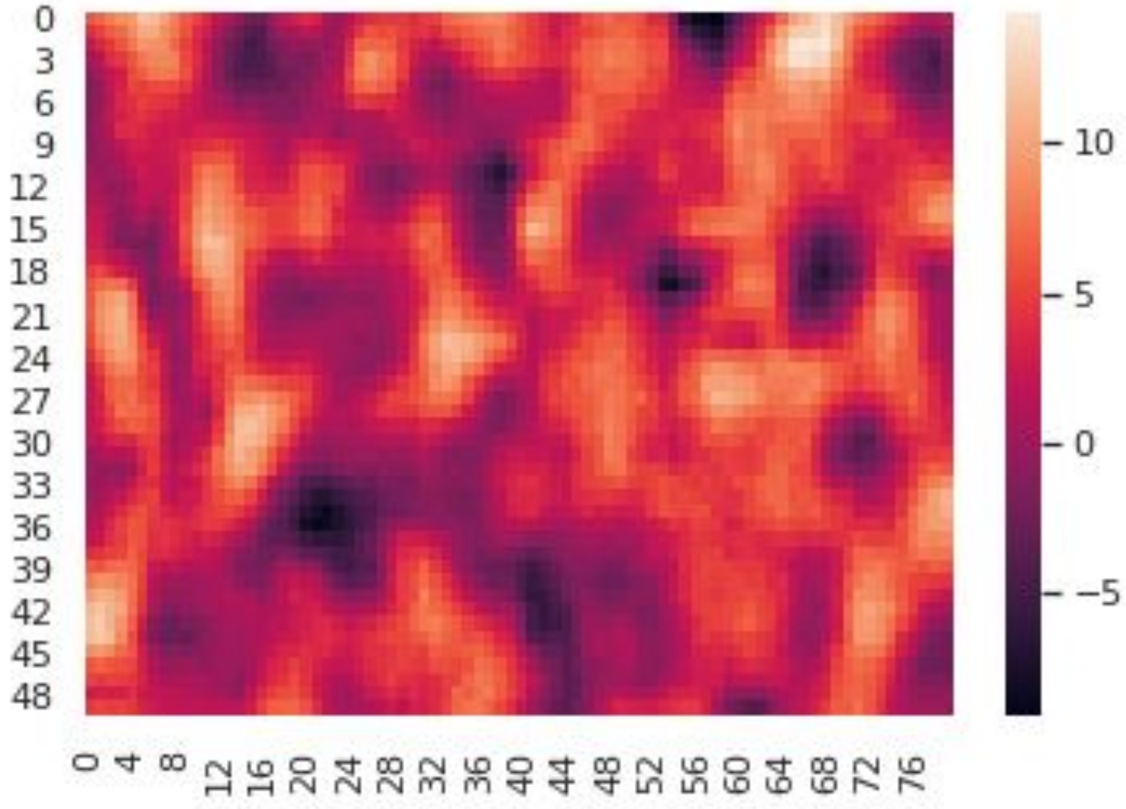
Monkey



Forecast



Errors (27.03.2019)



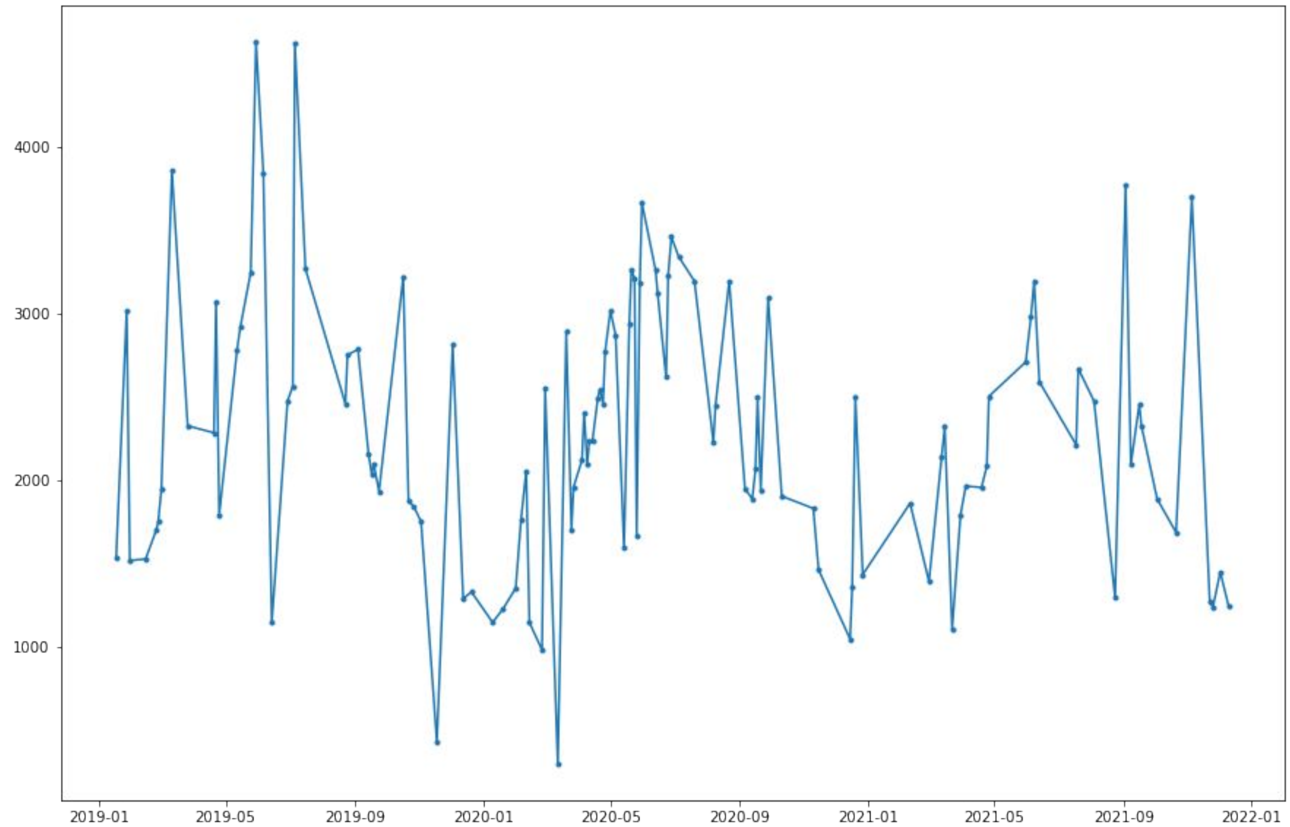
Forecasting Sentinel 2 data

- Data used: Sentinel 2
- Time period: 01.01.2019 - 31.12.2021
- Area: Top left 80 x 50 area of 30UXB tile
- Area under focus (i.e. plotted for every time step)
  - 45_32
  - 25_16
- Models used:
  - Seasonal Decomposition using LOESS
  - CNNs
- Approaches:
  - Ascending and descending orbits separately analyzed
  - Complete data analyzed together

Part 2: Sentinel 2 data

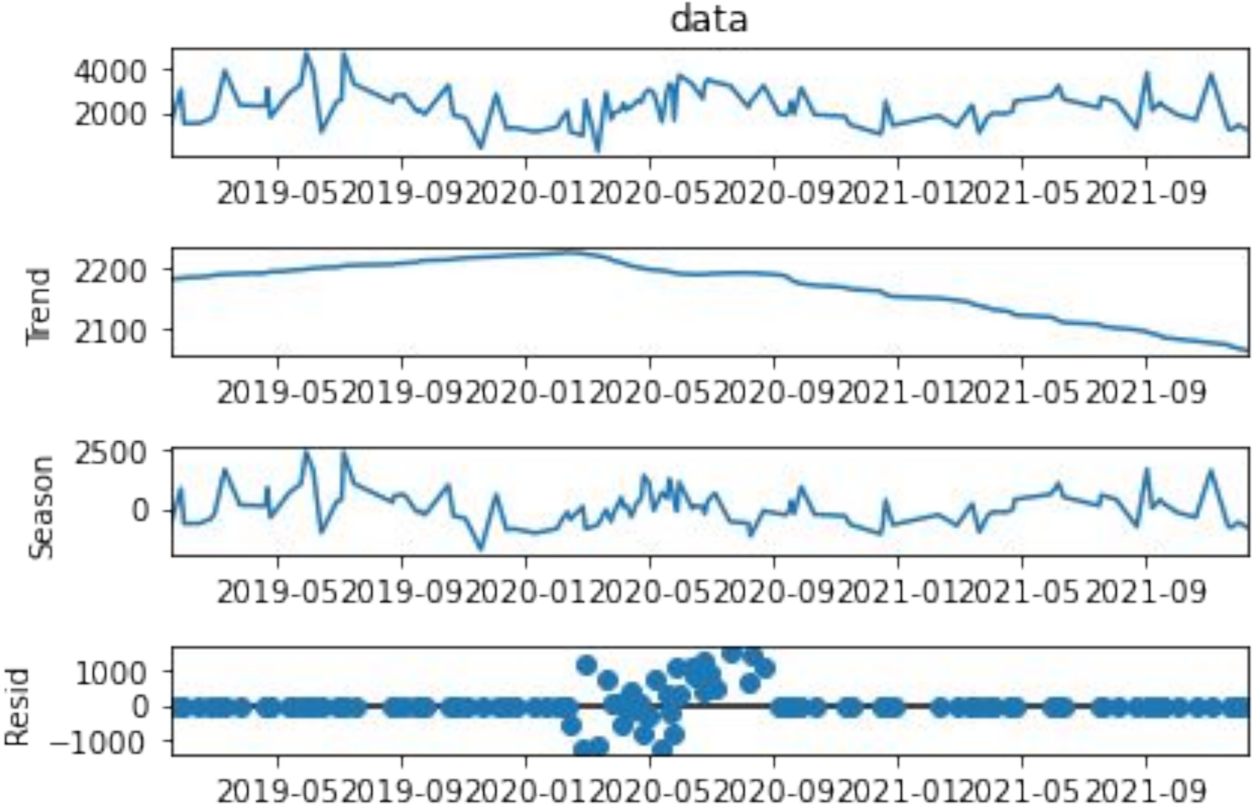
2.1 STL decomposition

Band 08
25_16

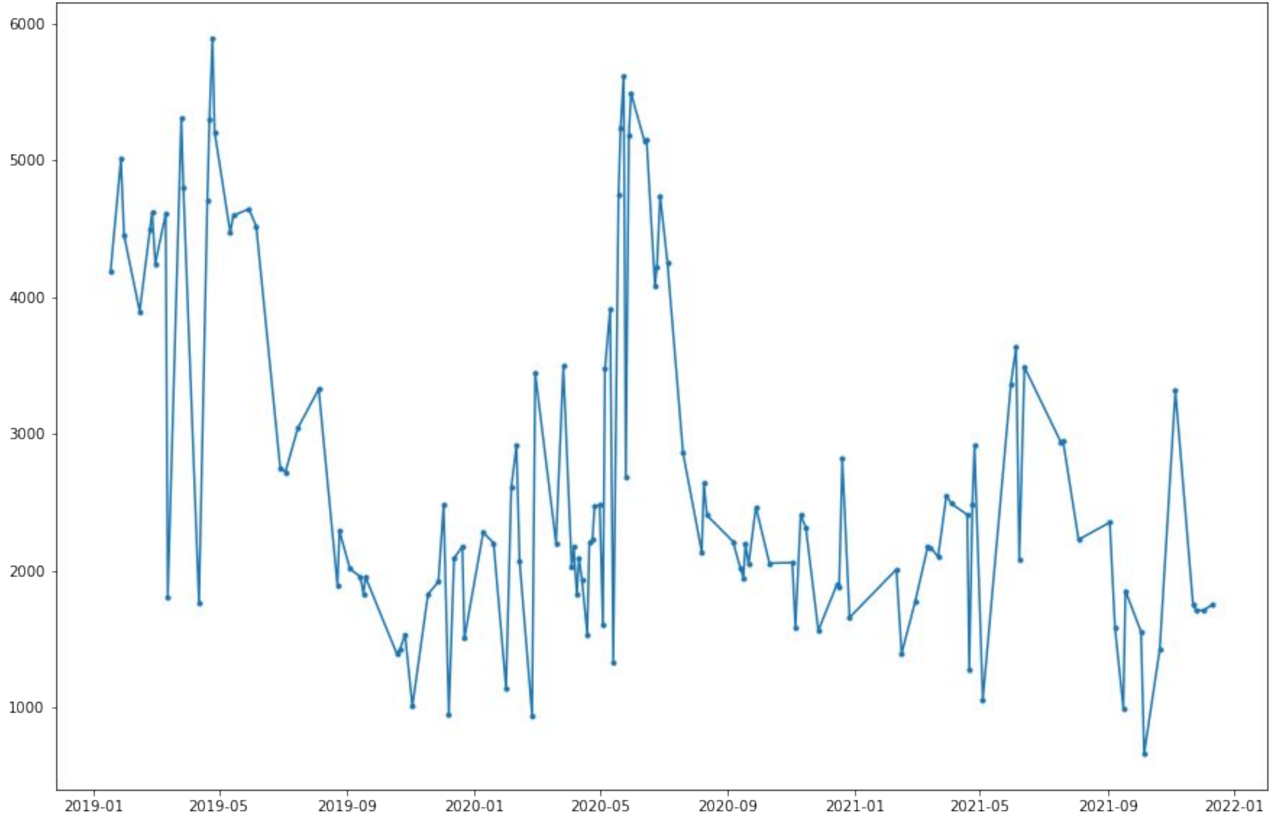


4000
3000
2000
1000
2019-01
2019-05
2019-09
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01

Band 08
25_16

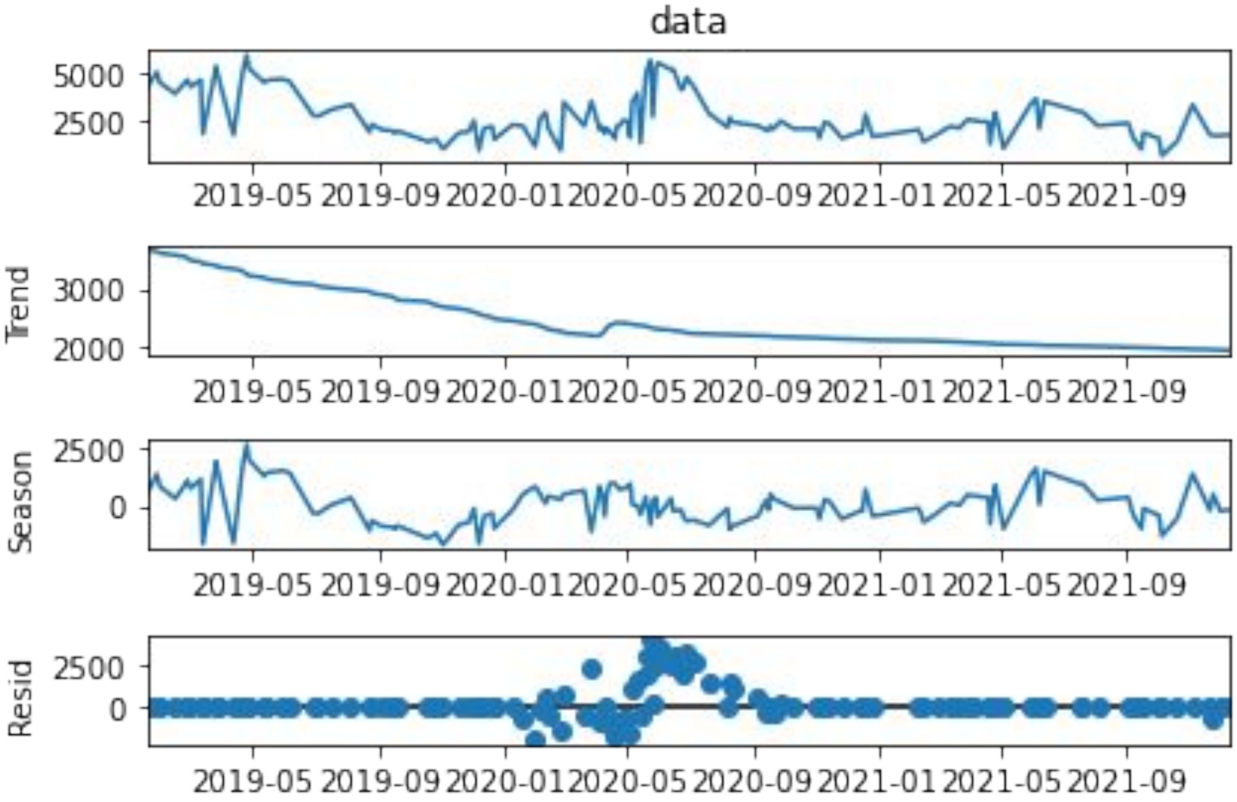

data
4000
2000
Trend
2200
2100
Season
2500
0
Resid
1000
0
−1000
2019-05
2019-09
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09

Band 08
45_32



6000
5000
4000
3000
2000
1000
2019-01
2019-05
2019-09
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01

Band 08
45_32


data
5000
2500
2019-05 2019-09 2020-01 2020-05 2020-09 2021-01 2021-05 2021-09
Trend
3000
2000
2019-05 2019-09 2020-01 2020-05 2020-09 2021-01 2021-05 2021-09
Season
2500
0
2019-05 2019-09 2020-01 2020-05 2020-09 2021-01 2021-05 2021-09
Resid
2500
0
2019-05 2019-09 2020-01 2020-05 2020-09 2021-01 2021-05 2021-09

Part 2: Sentinel 2 data

2.2 CNN

Methodology and data used

- Fawaz et al conducted a review in Time Series Classification
  - End-to-end deep learning achieve the current SOTA for TSC with architectures such as CNNs and deep Residual Networks.
  - FCN, ResNets and Encoders are top performers (Ref fig below)
- Sentinel 2
  - Time period 01.01.2019 to 31.12.2021
  - Bands used:
    - Input : B02 to B08, B10 and, B11
    - Output: B08
  - Area: 80x50 pixel area of 30UXB

Themes (#)	MLP	FCN	ResNet	Encoder	MCNN	t-LeNet	MCDCNN	Time-CNN	TWIESN
DEVICE (6)	0.0	50.0	**83.3**	0.0	0.0	0.0	0.0	0.0	0.0
ECG (7)	14.3	**71.4**	28.6	42.9	0.0	0.0	14.3	0.0	0.0
IMAGE (29)	6.9	34.5	**48.3**	10.3	0.0	0.0	6.9	10.3	0.0
MOTION (14)	14.3	28.6	**71.4**	21.4	0.0	0.0	0.0	0.0	0.0
SENSOR (16)	6.2	37.5	**75.0**	31.2	6.2	6.2	6.2	0.0	12.5
SIMULATED (6)	0.0	33.3	**100.0**	33.3	0.0	0.0	0.0	0.0	0.0
SPECTRO (7)	14.3	14.3	**71.4**	0.0	0.0	0.0	0.0	28.6	28.6

Table 4: Deep learning algorithms' performance grouped by themes. Each entry is the percentage of dataset themes an algorithm is most accurate for. Bold indicates the best model.

Length	MLP	FCN	ResNet	Encoder	MCNN	t-LeNet	MCDCNN	Time-CNN	TWIESN
<81	5.43	3.36	**2.43**	2.79	8.21	8.0	3.07	3.64	5.5
81-250	4.16	**1.63**	1.79	3.42	7.89	8.32	5.26	4.47	5.53
251-450	3.91	2.73	**1.64**	3.32	8.05	8.36	6.0	4.68	4.91
451-700	4.85	2.69	**1.92**	3.85	7.08	7.08	5.62	4.92	4.31
701-1000	4.6	1.9	**1.6**	3.8	7.4	8.5	5.2	6.0	4.5
>1000	3.29	2.71	**1.43**	3.43	7.29	8.43	4.86	5.71	6.0

Table 5: Deep learning algorithms' average ranks grouped by the datasets' length. Bold indicates the best model.

CNN architecture

```
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 9, 1)]            0

 conv1d (Conv1D)             (None, 9, 128)            1152

 batch_normalization (BatchN  (None, 9, 128)           512
 ormalization)

 activation (Activation)     (None, 9, 128)            0

 conv1d_1 (Conv1D)           (None, 9, 256)            164096

 batch_normalization_1 (Batc  (None, 9, 256)           1024
 hNormalization)

 activation_1 (Activation)   (None, 9, 256)            0

 conv1d_2 (Conv1D)           (None, 9, 128)            98432

 batch_normalization_2 (Batc  (None, 9, 128)           512
 hNormalization)

 activation_2 (Activation)   (None, 9, 128)            0

 global_average_pooling1d (G  (None, 128)              0
 lobalAveragePooling1D)

 dense (Dense)               (None, 1)                 129

=================================================================
```

Results

```
Epoch 1/500
12171/12171 [==============================] - 68s 5ms/step - loss: 1147570.8750 - mae: 843.6104
Epoch 2/500
12171/12171 [==============================] - 66s 5ms/step - loss: 1024646.1250 - mae: 807.0136
Epoch 3/500
12171/12171 [==============================] - 66s 5ms/step - loss: 1012440.8125 - mae: 800.1705
Epoch 4/500
12171/12171 [==============================] - 66s 5ms/step - loss: 1005781.0625 - mae: 796.0993
Epoch 5/500
12171/12171 [==============================] - 66s 5ms/step - loss: 1002032.1250 - mae: 793.4509
Epoch 6/500
12171/12171 [==============================] - 65s 5ms/step - loss: 999194.8125 - mae: 791.3301
Epoch 7/500
12171/12171 [==============================] - 67s 5ms/step - loss: 996811.7500 - mae: 789.6002
Epoch 8/500
12171/12171 [==============================] - 67s 5ms/step - loss: 994671.4375 - mae: 787.7321
Epoch 9/500
12171/12171 [==============================] - 66s 5ms/step - loss: 993262.4375 - mae: 786.6482
Epoch 10/500
12171/12171 [==============================] - 65s 5ms/step - loss: 991872.6250 - mae: 785.7442
Epoch 11/500
12171/12171 [==============================] - 66s 5ms/step - loss: 990154.6250 - mae: 784.4467
Epoch 12/500
12171/12171 [==============================] - 66s 5ms/step - loss: 988880.0625 - mae: 783.6118
Epoch 13/500
12171/12171 [==============================] - 66s 5ms/step - loss: 987647.9375 - mae: 782.6048
Epoch 14/500
12171/12171 [==============================] - 66s 5ms/step - loss: 986683.0000 - mae: 781.8907
```

Epochs 1-14

```
Epoch 137/500
12171/12171 [==============================] - 68s 6ms/step - loss: 960877.3750 - mae: 760.6036
Epoch 138/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960758.3750 - mae: 760.6694
Epoch 139/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960799.5000 - mae: 760.5240
Epoch 140/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960714.2500 - mae: 760.4146
Epoch 141/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960702.3750 - mae: 760.3890
Epoch 142/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960473.2500 - mae: 760.0776
Epoch 143/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960372.6250 - mae: 760.0784
Epoch 144/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960246.0000 - mae: 760.1436
Epoch 145/500
12171/12171 [==============================] - 68s 6ms/step - loss: 960347.7500 - mae: 760.2277
Epoch 146/500
12171/12171 [==============================] - 68s 6ms/step - loss: 960318.8750 - mae: 760.1035
Epoch 147/500
12171/12171 [==============================] - 68s 6ms/step - loss: 960267.0000 - mae: 760.0806
Epoch 148/500
12171/12171 [==============================] - 67s 6ms/step - loss: 960192.7500 - mae: 760.0928
Epoch 149/500
 8892/12171 [===================>.........] - ETA: 18s - loss: 959787.4375 - mae: 759.3056
```

Epochs 137-149

Observations

- Data values are 10^3
- MAE and MSE approaches 10^3
- Errors change saturates in 3rd epoch

Future Endeavors

- Z-normalize data, as suggested by Fawaz et al.
- Encoders might be worthwhile to run
- Statistical models like SARIMAX are easier to interpret so should be used as baseline models
- Some good background reading in this direction are:
    - Deep learning for time series classification: a review
    - Time Series Forecasting With Deep Learning: A Survey
    - N-BEATS
    - Forecasting: Principles and Practice

For more results and detailed understanding, please refer to the associated github repository: https://github.com/NirliptaPande/Vienna