RAPORT Z WYKONANIA PROJEKTU Z PRZEDMIOTU

SYSTEMY BIG DATA W ANALIZIE DANYCH IoT

Temat: Analiza danych z czujników IoT – pomiar oraz predykcja temperatury

Wykonawcy: Wojciech Mętel i Maciej Szymczak

1. Cel Projektu

Celem projektu jest napisanie aplikacji, która będzie zaczytywała dane z zewnętrznej bazy dostarczonej z artykułem projektu, prezentowała wykresy temperatur i przewidywała zmiany.

Projekt będzie realizowany w języku Python w środowisku Jupyter.

2. Omówienie problemu

Źródłem danych do projektu jest anonimowa, amatorska stacja pogodowa znajdująca się w Indiach. Temperatura była mierzona zarówno wewnątrz pomieszczenia, jak i na zewnątrz. Motywacją do wykonania pomiarów były zmiany klimatu związane z globalnym ociepleniem. Pomiary wykonywano na przestrzeni ostatniego kwartału 2018r.

3. Realizacja projektu

1) Utworzenie środowiska i zaczytanie odpowiednich bibliotek potrzebnych do realizacji projektu

W realizacji zostały użyte następujące biblioteki:

Numpy – obliczenia

Pandas – wczytanie i praca z danymi z csv

Holoviews oraz matplotlib – prezentacja danych

Scikit oraz FbProphet – funkcje predykcyjne

```
In [4]: import numpy as np
        import pandas as pd
        import holoviews as hv
        from holoviews import opts
        hv.extension('bokeh')
        from matplotlib import pyplot as plt
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler, LabelEncoder
        import os
        from fbprophet import Prophet
        from fbprophet.plot import add_changepoints_to_plot
```

2) Wczytanie danych

Do realizacji projektu zostały użyte dane dostarczone przez czujnik temperatury IoT w pliku w formacie csv.

In [5]:
```
df = pd.read_csv("IOT-temp.csv")
print(f'IOT-temp.csv : {df.shape}')
df.head(3)
```

```
IOT-temp.csv : (97606, 5)
```

Out[5]:

	id	room_id/id	noted_date	temp	out/in
0	__export__.temp_log_196134_bd201015	Room Admin	08-12-2018 09:30	29	In
1	__export__.temp_log_196131_7bca51bc	Room Admin	08-12-2018 09:30	29	In
2	__export__.temp_log_196127_522915e3	Room Admin	08-12-2018 09:29	41	Out

3) Obróbka danych

Aby przygotować przedstawione dane do odpowiedniej obróbki i prezentacji, format danych został uzupełniony o odpowiednie tabele.

In [9]:
```
df['date'] = pd.to_datetime(df['date'], format='%d-%m-%Y %H:%M')
df['year'] = df['date'].apply(lambda x : x.year)
df['month'] = df['date'].apply(lambda x : x.month)
df['day'] = df['date'].apply(lambda x : x.day)
df['weekday'] = df['date'].apply(lambda x : x.day_name())
df['weekofyear'] = df['date'].apply(lambda x : x.weekofyear)
df['hour'] = df['date'].apply(lambda x : x.hour)
df['minute'] = df['date'].apply(lambda x : x.minute)
df.head(3)
```

Out[9]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute
0	__export__.temp_log_196134_bd201015	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30
1	__export__.temp_log_196131_7bca51bc	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30
2	__export__.temp_log_196127_522915e3	2018-12-08 09:29:00	41	Out	2018	12	8	Saturday	49	9	29

In [10]:
```
def month2seasons(x):
    if x in [12, 1, 2]:
        season = 'Winter'
    elif x in [3, 4, 5]:
        season = 'Summer'
    elif x in [6, 7, 8, 9]:
        season = 'Monsoon'
    elif x in [10, 11]:
        season = 'Post_Monsoon'
    return season
```

In [11]:
```
df['season'] = df['month'].apply(month2seasons)
df.head(3)
```

Out[11]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season
0	__export__.temp_log_196134_bd201015	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30	Winter
1	__export__.temp_log_196131_7bca51bc	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30	Winter
2	__export__.temp_log_196127_522915e3	2018-12-08 09:29:00	41	Out	2018	12	8	Saturday	49	9	29	Winter

In [12]:
```
def hours2timing(x):
    if x in [22,23,0,1,2,3]:
        timing = 'Night'
    elif x in range(4, 12):
        timing = 'Morning'
    elif x in range(12, 17):
        timing = 'Afternoon'
    elif x in range(17, 22):
        timing = 'Evening'
    else:
        timing = 'X'
    return timing
```

In [13]:
```
df['timing'] = df['hour'].apply(hours2timing)
df.head(3)
```

Out[13]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
0	__export__.temp_log_196134_bd201015	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30	Winter	Morning
1	__export__.temp_log_196131_7bca51bc	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30	Winter	Morning
2	__export__.temp_log_196127_522915e3	2018-12-08 09:29:00	41	Out	2018	12	8	Saturday	49	9	29	Winter	Morning

In [14]:
```
df[df.duplicated()]
```

Out[14]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
11	__export__.temp_log_196108_4a983c7e	2018-12-08 09:25:00	42	Out	2018	12	8	Saturday	49	9	25	Winter	Morning

In [15]:
```
df[df['id']=='__export__.temp_log_196108_4a983c7e']
```

Out[15]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
10	__export__.temp_log_196108_4a983c7e	2018-12-08 09:25:00	42	Out	2018	12	8	Saturday	49	9	25	Winter	Morning
11	__export__.temp_log_196108_4a983c7e	2018-12-08 09:25:00	42	Out	2018	12	8	Saturday	49	9	25	Winter	Morning

In [16]:
```
df.drop_duplicates(inplace=True)
df[df.duplicated()]
```

Out[16]:

id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing

In [17]:
```
df.loc[df['date']=='2018-09-12 03:09:00', ].sort_values(by='id').head(5)
```

Out[17]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
61229	__export__.temp_log_101144_ff2f0b97	2018-09-12 03:09:00	29	Out	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61258	__export__.temp_log_101502_172517d2	2018-09-12 03:09:00	29	In	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61255	__export__.temp_log_104868_a5e526b3	2018-09-12 03:09:00	28	In	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61231	__export__.temp_log_108845_062b2592	2018-09-12 03:09:00	28	In	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61272	__export__.temp_log_112303_fca608f4	2018-09-12 03:09:00	29	In	2018	9	12	Wednesday	37	3	9	Monsoon	Night

In [18]:
```
df['id'].apply(lambda x : x.split('_')[6]).nunique() == len(df)
```

Out[18]:
```
True
```

In [19]:
```
df['id'] = df['id'].apply(lambda x : int(x.split('_')[6]))
df.head(3)
```

```
In [18]: df['id'].apply(lambda x : x.split('_')[6]).nunique() == len(df)
Out[18]: True
```

```
In [19]: df['id'] = df['id'].apply(lambda x : int(x.split('_')[6]))
         df.head(3)
```

Out[19]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
0	196134	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30	Winter	Morning
1	196131	2018-12-08 09:30:00	29	In	2018	12	8	Saturday	49	9	30	Winter	Morning
2	196127	2018-12-08 09:29:00	41	Out	2018	12	8	Saturday	49	9	29	Winter	Morning

```
In [20]: df.loc[df['date'] == '2018-09-12 03:09:00', ].sort_values(by ='id').head(5)
```

Out[20]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
61273	17002	2018-09-12 03:09:00	29	Out	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61275	17003	2018-09-12 03:09:00	28	Out	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61267	17006	2018-09-12 03:09:00	28	Out	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61269	17009	2018-09-12 03:09:00	28	Out	2018	9	12	Wednesday	37	3	9	Monsoon	Night
61271	17010	2018-09-12 03:09:00	29	Out	2018	9	12	Wednesday	37	3	9	Monsoon	Night

```
In [21]: df.loc[df['id'].isin(range(4000, 4011))].sort_values(by='id')
```

Out[21]:

	id	date	temp	place	year	month	day	weekday	weekofyear	hour	minute	season	timing
84141	4000	2018-09-09 16:24:00	29	Out	2018	9	9	Sunday	36	16	24	Monsoon	Afternoon
84142	4002	2018-09-09 16:24:00	29	Out	2018	9	9	Sunday	36	16	24	Monsoon	Afternoon
84144	4004	2018-09-09 16:23:00	28	Out	2018	9	9	Sunday	36	16	23	Monsoon	Afternoon
84128	4006	2018-09-09 16:24:00	28	Out	2018	9	9	Sunday	36	16	24	Monsoon	Afternoon
84132	4007	2018-09-09 16:24:00	29	Out	2018	9	9	Sunday	36	16	24	Monsoon	Afternoon
84136	4009	2018-09-09 16:24:00	28	Out	2018	9	9	Sunday	36	16	24	Monsoon	Afternoon
84137	4010	2018-09-09 16:24:00	28	Out	2018	9	9	Sunday	36	16	24	Monsoon	Afternoon

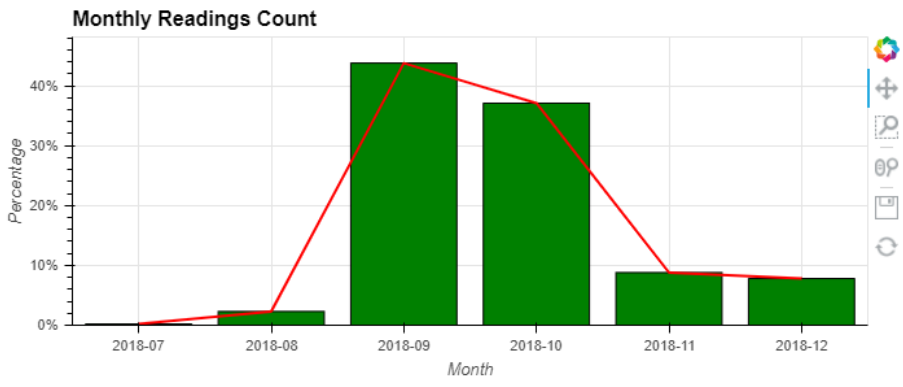
Jak widać dane trzeba było odpowiednio uporządkować. Wystąpił też problem z powieleniem się danych co trzeba było odnaleźć i usunąć duplikaty. Wzięliśmy także pod uwagę miejsce, w którym zamontowany był czujnik temperatury. Jako, że było to w Indiach, dane zostały podzielone na pory roku i dnia odpowiednie dla miejsca przeprowadzenia pomiarów.

4) Prezentacja danych z czujnika

Po odpowiedniej obróbce danych można przedstawić odczytane parametry:

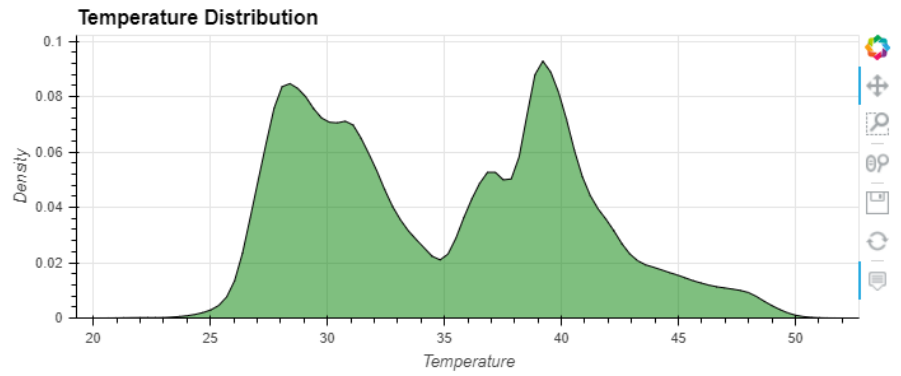
4.1) Podział ze względu na ilość w miesiącu

Out[22]:



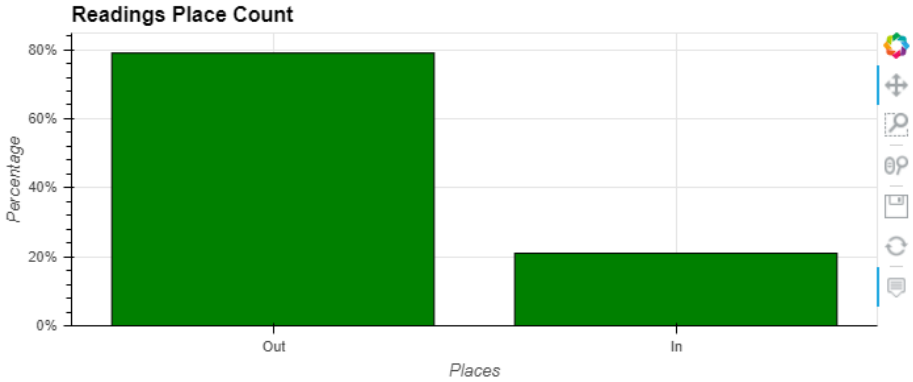
4.2) Dystrybucja temperatury

Out[23]:



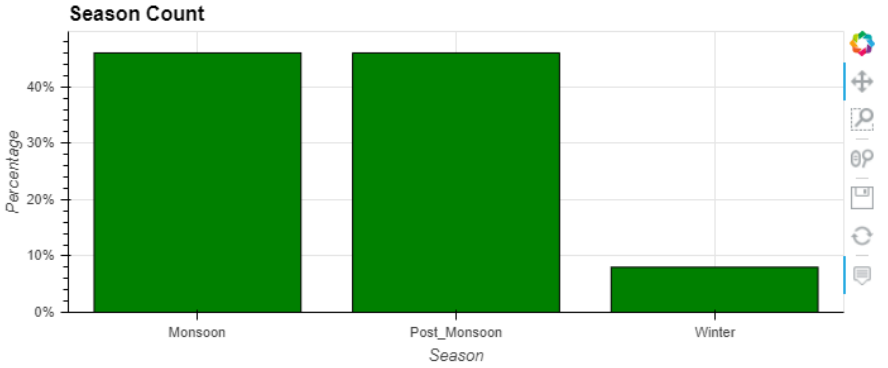
4.3) Miejsce pochodzenia pomiaru (z wewnątrz pokoju administracyjnego czy z wewnątrz)

Out[24]:



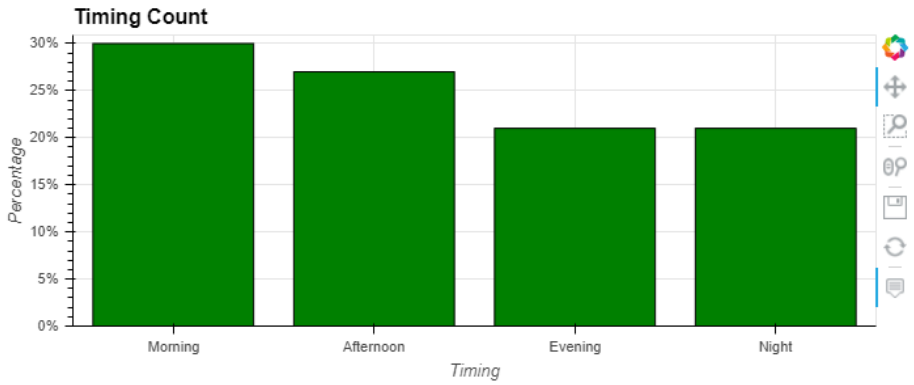
4.4) Ilość pomiarów w danej porze roku

Out[25]:



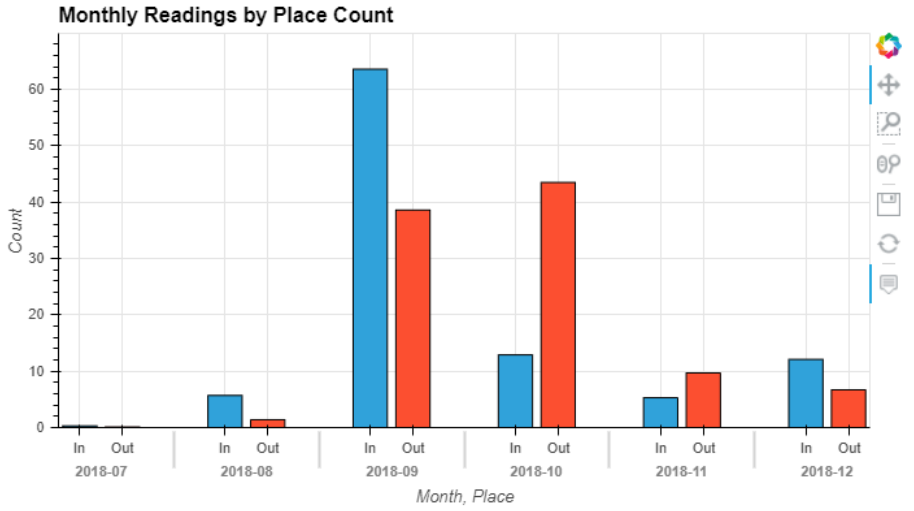
4.5) Ilość pomiarów w danej porze dnia

Out[26]:



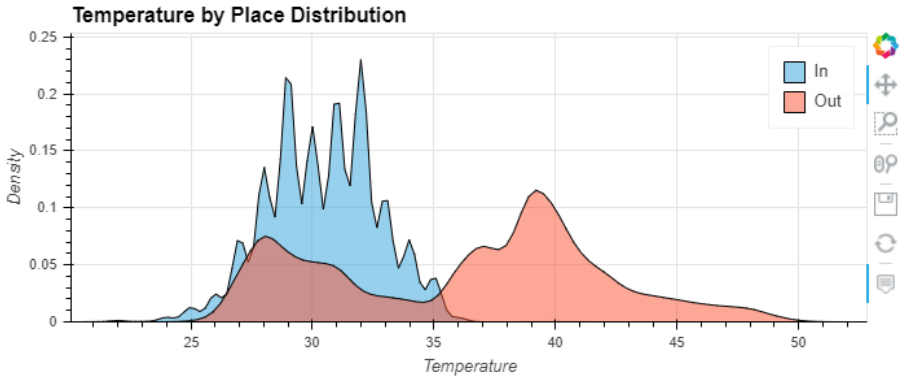
4.6) Ilość pomiarów w miesiącu z podziałem na miejsce (w środku czy na zewnątrz)

Out[27]:



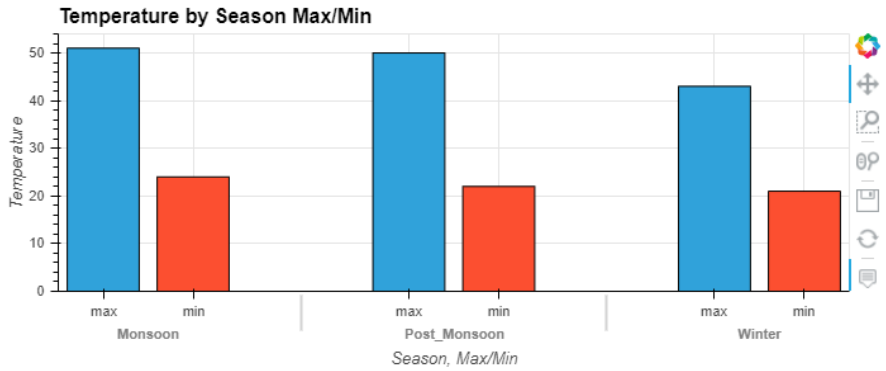
4.7) Dystrybucja temperatury z podziałem na miejsce

Out[28]:



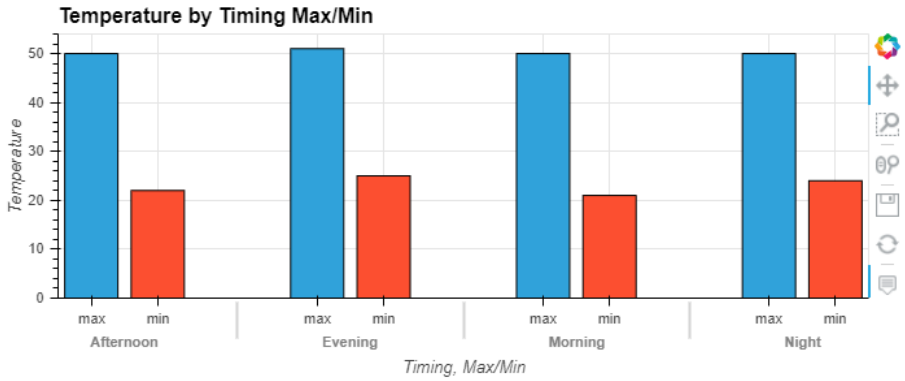
4.8) Minimalna i maksymalna temperatura w danej porze roku

Out[29]:



4.9) Minimalna i maksymalna temperatura w danej porze dnia

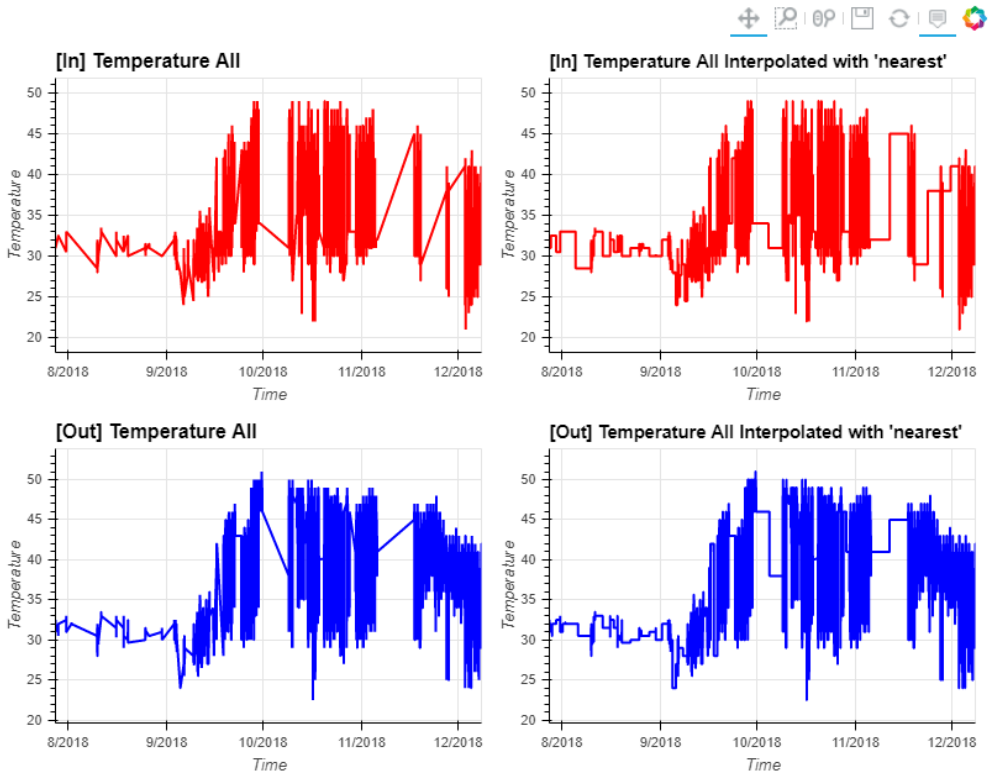
Out[30]:



5) Problemy z danymi

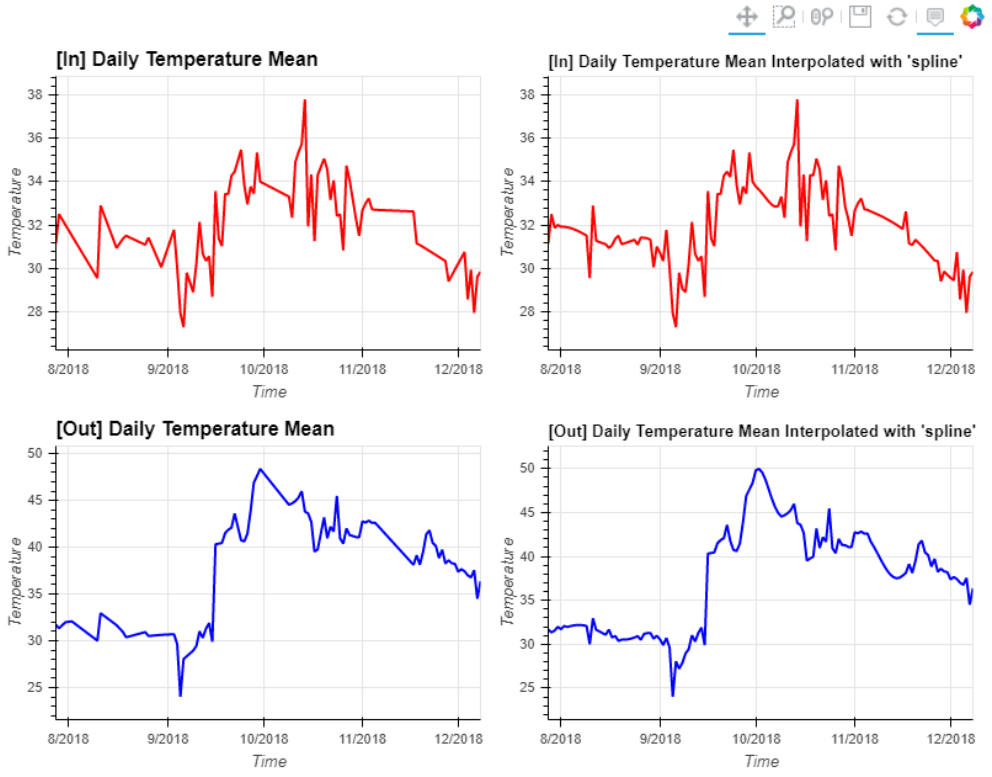
Jako, że w dostarczonej bazie danych z pomiarami brakowało niektórych rekordów (mógł nastąpić problem z łączem internetowym lub błąd czujnika) należało uzupełnić bazę danymi uśrednionymi. Początkowo przyjęta została metoda przybliżenia do najbliższego ale nie dała ona satysfakcjonujących rezultatów:

Out[36]:



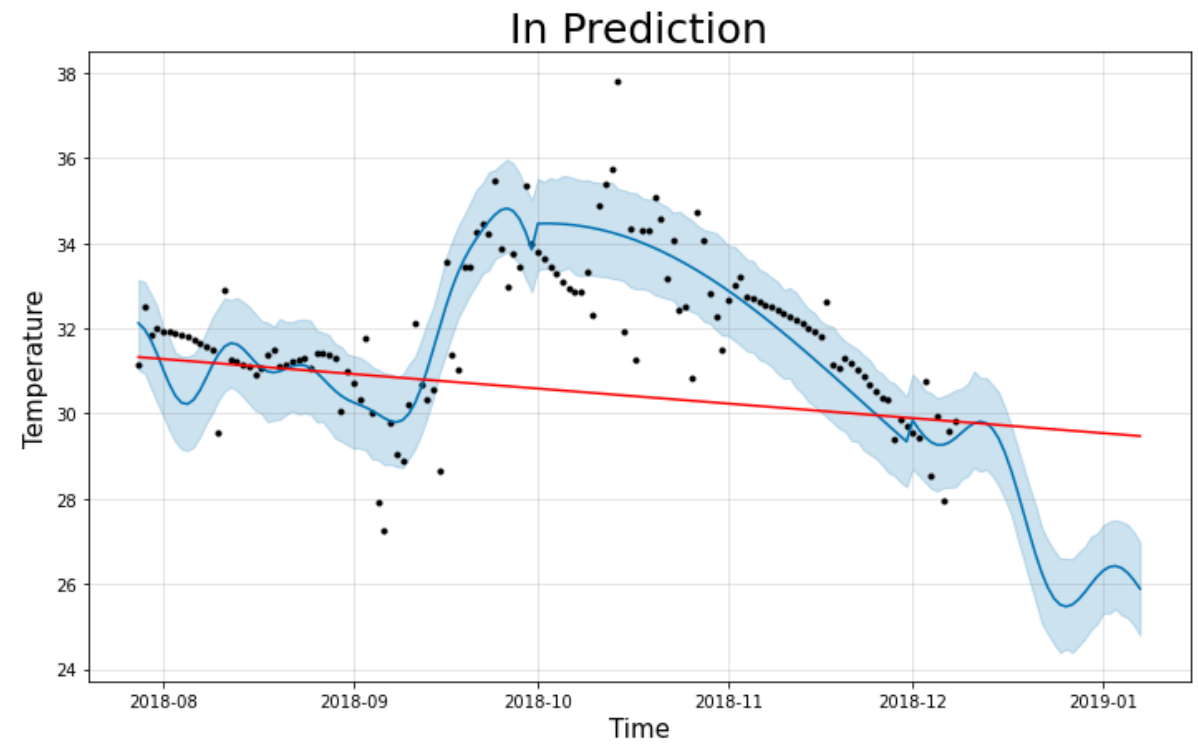
Dopiero zastosowanie przybliżenia metodą sklejania (spline) dało następujący rezultat:

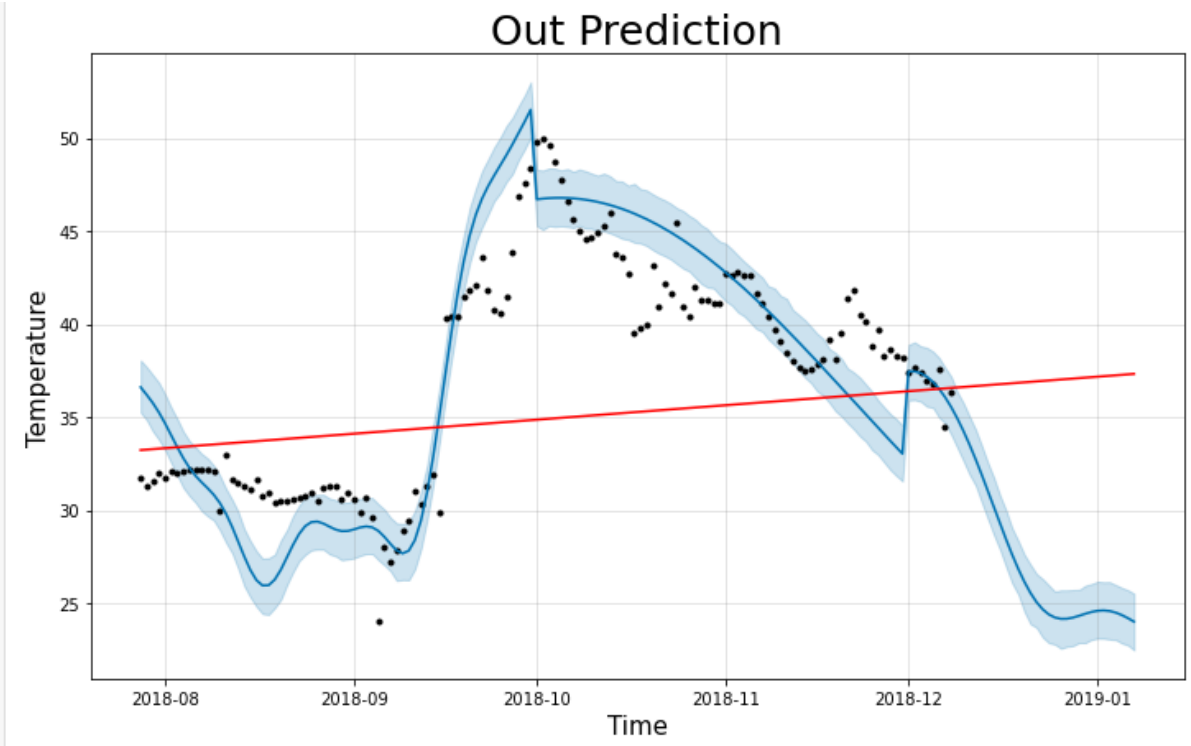
Out[37]:



6) Predykcja

Funkcja predykcyjna została wykonana w oparciu o dokumentację biblioteki FbProphet.
Predykcja została wykonana dla 30 następnych dni.


In Prediction
Temperature
Time
38
36
34
32
30
28
26
24
2018-08
2018-09
2018-10
2018-11
2018-12
2019-01


Out Prediction
Temperature
Time
50
45
40
35
30
25
2018-08
2018-09
2018-10
2018-11
2018-12
2019-01

4. Wnioski

- Podstawowym zadaniem przy realizacji projektu była praca z dodatkowymi bibliotekami oraz ich dokumentacją. Podczas wykonywania musieliśmy powtórzyć wiedzę z już poznanych bibliotek a także skorzystać z nowych (holoviews - interaktywne wykresy, fb prophet - zbiór funkcji predykcyjnych). Pozwoliło to na zarówno rozwinięcie się pod kątem programistycznym jak i nauczyło nas korzystania z nowych rozwiązań w oparciu o ich dokumentację.

- Jednym z napotkanych problemów była kompatybilność wsteczna bibliotek - okazało się, iż do poprawnego działania fb-prophet wymaga starszej wersji biblioteki pandas.

- Obróbka i analiza danych były powtórzeniem i rozszerzeniem informacji oraz umiejętności zdobytych podczas kursu.