Environment Set-Up

Load relevant Python Packages

In [15]:
```
reset -fs
```

In [16]:
```
# Importing the most important modules
import pandas as pd
from datetime import datetime
```

Preparing the Results for Visualization

To assure that the result data is saved in the right format and directory to be visualized 2 dataframes are set up:

- The first one will include all the predictions for each model for both the validation and the test set and for each timestep.
- The second one will include all the actual observed values for each model for both the validation and the test set and for each timestep.

In [17]:
```
columnnames = ["date","Step 1","Step 2","Step 3","Step 4","Step 5","Step 6","Step 7","Step 8","Step 9",
               "Step 10","Step 11","Step 12","Step 13","Step 14","Step 15","Step 16","Step 17",
               "Step 18", "Model", "Dataset"]

# setting up source_pred
df_pred = pd.read_csv("./Results/naive_shift_validation_predictions.csv")
df_naive_test_pred = pd.read_csv("./Results/naive_shift_test_predictions.csv")
df_mov_av_val_pred = pd.read_csv("./Results/moving_average_validation_predictions.csv")
df_mov_av_test_pred = pd.read_csv("./Results/moving_average_test_predictions.csv")
df_prophet_val_pred = pd.read_csv("./Results/prophet_validation_predictions.csv")
df_prophet_test_pred = pd.read_csv("./Results/prophet_test_predictions.csv")
df_multi_lstm_peephole_val_pred = pd.read_csv("./Results/multi_lstm_peephole_validation_predictions.csv")
df_multi_lstm_peephole_test_pred = pd.read_csv("./Results/multi_lstm_peephole_test_predictions.csv")
df_uni_lstm_peephole_val_pred = pd.read_csv("./Results/uni_lstm_peephole_validation_predictions.csv")
df_uni_lstm_peephole_test_pred = pd.read_csv("./Results/uni_lstm_peephole_test_predictions.csv")
df_multi_lstm_val_pred = pd.read_csv("./Results/multi_lstm_validation_predictions.csv")
df_multi_lstm_test_pred = pd.read_csv("./Results/multi_lstm_test_predictions.csv")
df_uni_lstm_val_pred = pd.read_csv("./Results/uni_lstm_validation_predictions.csv")
df_uni_lstm_test_pred = pd.read_csv("./Results/uni_lstm_test_predictions.csv")

# assuring coherence between the datasets
df_mov_av_val_pred.drop(columns = ["y_all_pred Step 19","y_all_pred Step 20"], inplace = True)
df_mov_av_test_pred.drop(columns = ["y_all_pred Step 19","y_all_pred Step 20"], inplace = True)
df_prophet_val_pred.columns = df_pred.columns
df_prophet_test_pred.columns = df_pred.columns
df_multi_lstm_peephole_val_pred.columns = df_pred.columns
df_multi_lstm_peephole_test_pred.columns = df_pred.columns
df_uni_lstm_peephole_val_pred.columns = df_pred.columns
df_uni_lstm_peephole_test_pred.columns = df_pred.columns
df_multi_lstm_val_pred.columns = df_pred.columns
df_multi_lstm_test_pred.columns = df_pred.columns
df_uni_lstm_val_pred.columns = df_pred.columns
df_uni_lstm_test_pred.columns = df_pred.columns
df_multi_lstm_peephole_val_pred["date"] = df_pred.date.iloc[0:1420]
df_multi_lstm_peephole_test_pred["date"] = df_naive_test_pred.date.iloc[0:1420]
df_uni_lstm_peephole_val_pred["date"] = df_pred.date.iloc[0:1420]
df_uni_lstm_peephole_test_pred["date"] = df_naive_test_pred.date.iloc[0:1420]
df_multi_lstm_val_pred["date"] = df_pred.date.iloc[0:1420]
df_multi_lstm_test_pred["date"] = df_naive_test_pred.date.iloc[0:1420]
df_uni_lstm_val_pred["date"] = df_pred.date.iloc[0:1420]
df_uni_lstm_test_pred["date"] = df_naive_test_pred.date.iloc[0:1420]

# creating selectors for dataset variable
df_pred["Dataset"] = "Validation"
df_naive_test_pred["Dataset"] = "Test"
df_mov_av_val_pred["Dataset"] = "Validation"
df_mov_av_test_pred["Dataset"] = "Test"
df_prophet_val_pred["Dataset"] = "Validation"
df_prophet_test_pred["Dataset"] = "Test"
df_multi_lstm_peephole_val_pred["Dataset"] = "Validation"
df_multi_lstm_peephole_test_pred["Dataset"] = "Test"
df_uni_lstm_peephole_val_pred["Dataset"] = "Validation"
df_uni_lstm_peephole_test_pred["Dataset"] = "Test"
df_multi_lstm_val_pred["Dataset"] = "Validation"
df_multi_lstm_test_pred["Dataset"] = "Test"
df_uni_lstm_val_pred["Dataset"] = "Validation"
df_uni_lstm_test_pred["Dataset"] = "Test"


# stacking all the predictions from the different models on the validation and test sets in one datafra
me
df_pred = df_pred.append(df_naive_test_pred,)
df_pred = df_pred.append(df_mov_av_val_pred)
df_pred = df_pred.append(df_mov_av_test_pred)
df_pred = df_pred.append(df_prophet_val_pred)
df_pred = df_pred.append(df_prophet_test_pred)
df_pred = df_pred.append(df_multi_lstm_peephole_val_pred)
df_pred = df_pred.append(df_multi_lstm_peephole_test_pred)
df_pred = df_pred.append(df_uni_lstm_peephole_val_pred)
df_pred = df_pred.append(df_uni_lstm_peephole_test_pred)
df_pred = df_pred.append(df_multi_lstm_val_pred)
df_pred = df_pred.append(df_multi_lstm_test_pred)
df_pred = df_pred.append(df_uni_lstm_val_pred)
df_pred = df_pred.append(df_uni_lstm_test_pred)
df_pred.columns = columnnames
df_pred.head(2)
```

Out[17]:

	date	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	...	Step 11	Step 12	Step 13
0	2019-03-17 06:00:00	0.327166	0.327166	0.327166	0.327166	0.327166	0.327166	0.327166	0.327166	0.327166	...	0.327166	0.327166	0.327166
1	2019-03-17 06:10:00	0.453624	0.453624	0.453624	0.453624	0.453624	0.453624	0.453624	0.453624	0.453624	...	0.453624	0.453624	0.453624

2 rows × 21 columns

In [18]:
```
# setting up source_test
df_test = pd.read_csv("./Results/naive_shift_validation_values.csv")
df_naive_test_test = pd.read_csv("./Results/naive_shift_test_values.csv")
df_mov_av_val_test = pd.read_csv("./Results/moving_average_validation_values.csv")
df_mov_av_test_test = pd.read_csv("./Results/moving_average_test_values.csv")
df_prophet_val_test = pd.read_csv("./Results/prophet_validation_values.csv")
df_prophet_test_test = pd.read_csv("./Results/prophet_test_values.csv")
df_multi_lstm_peephole_val_test = pd.read_csv("./Results/multi_lstm_peephole_validation_values.csv")
df_multi_lstm_peephole_test_test = pd.read_csv("./Results/multi_lstm_peephole_test_values.csv")
df_uni_lstm_peephole_val_test = pd.read_csv("./Results/uni_lstm_peephole_validation_values.csv")
df_uni_lstm_peephole_test_test = pd.read_csv("./Results/uni_lstm_peephole_test_values.csv")
df_multi_lstm_val_test = pd.read_csv("./Results/multi_lstm_validation_values.csv")
df_multi_lstm_test_test = pd.read_csv("./Results/multi_lstm_test_values.csv")
df_uni_lstm_val_test = pd.read_csv("./Results/uni_lstm_validation_values.csv")
df_uni_lstm_test_test = pd.read_csv("./Results/uni_lstm_test_values.csv")

# assuring coherence between the datasets
df_mov_av_val_test.drop(columns = ["y_all_observed Step 1","y_all_observed Step 2"], inplace = True)
df_mov_av_test_test.drop(columns = ["y_all_observed Step 1","y_all_observed Step 2"], inplace = True)
df_mov_av_val_test.columns = df_test.columns
df_mov_av_test_test.columns = df_test.columns
df_prophet_val_test.columns = df_test.columns
df_prophet_test_test.columns = df_test.columns
df_multi_lstm_peephole_val_test.columns = df_test.columns
df_multi_lstm_peephole_test_test.columns = df_test.columns
df_uni_lstm_peephole_val_test.columns = df_test.columns
df_uni_lstm_peephole_test_test.columns = df_test.columns
df_multi_lstm_val_test.columns = df_test.columns
df_multi_lstm_test_test.columns = df_test.columns
df_uni_lstm_val_test.columns = df_test.columns
df_uni_lstm_test_test.columns = df_test.columns
df_multi_lstm_peephole_val_test["date"] = df_test.date.iloc[0:1420]
df_multi_lstm_peephole_test_test["date"] = df_naive_test_test.date.iloc[0:1420]
df_uni_lstm_peephole_val_test["date"] = df_test.date.iloc[0:1420]
df_uni_lstm_peephole_test_test["date"] = df_naive_test_test.date.iloc[0:1420]
df_multi_lstm_val_test["date"] = df_test.date.iloc[0:1420]
df_multi_lstm_test_test["date"] = df_naive_test_test.date.iloc[0:1420]
df_uni_lstm_val_test["date"] = df_test.date.iloc[0:1420]
df_uni_lstm_test_test["date"] = df_naive_test_test.date.iloc[0:1420]

# creating selectors for dataset variable
df_test["Dataset"] = "Validation"
df_naive_test_test["Dataset"] = "Test"
df_mov_av_val_test["Dataset"] = "Validation"
df_mov_av_test_test["Dataset"] = "Test"
df_prophet_val_test["Dataset"] = "Validation"
df_prophet_test_test["Dataset"] = "Test"
df_multi_lstm_peephole_val_test["Dataset"] = "Validation"
df_multi_lstm_peephole_test_test["Dataset"] = "Test"
df_uni_lstm_peephole_val_test["Dataset"] = "Validation"
df_uni_lstm_peephole_test_test["Dataset"] = "Test"
df_multi_lstm_val_test["Dataset"] = "Validation"
df_multi_lstm_test_test["Dataset"] = "Test"
df_uni_lstm_val_test["Dataset"] = "Validation"
df_uni_lstm_test_test["Dataset"] = "Test"

# stacking all the actual values from the different models on the validation and test sets in one dataf
rame
df_test = df_test.append(df_naive_test_test)
df_test = df_test.append(df_mov_av_val_test)
df_test = df_test.append(df_mov_av_test_test)
df_test = df_test.append(df_prophet_val_test)
df_test = df_test.append(df_prophet_test_test)
df_test = df_test.append(df_multi_lstm_peephole_val_test)
df_test = df_test.append(df_multi_lstm_peephole_test_test)
df_test = df_test.append(df_uni_lstm_peephole_val_test)
df_test = df_test.append(df_uni_lstm_peephole_test_test)
df_test = df_test.append(df_multi_lstm_val_test)
df_test = df_test.append(df_multi_lstm_test_test)
df_test = df_test.append(df_uni_lstm_val_test)
df_test = df_test.append(df_uni_lstm_test_test)
df_test.columns = columnnames
df_test.head(2)
```

Out[18]:

	date	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	...	Step 11	Step 12	Step 13
0	2019-03-17 06:00:00	0.453624	0.472353	0.466946	0.439173	0.286735	0.265595	0.349561	0.339105	0.344043	...	0.246061	0.242016	0.254804
1	2019-03-17 06:10:00	0.472353	0.466946	0.439173	0.286735	0.265595	0.349561	0.339105	0.344043	0.264401	...	0.242016	0.254804	0.263315

2 rows × 21 columns

Exporting the Stacked Results

Both stacked dataframes are saved and can now be explored interactively via Bokeh.

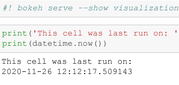
In [19]:
```
df_test.to_csv("./Results/values.csv", index = False)
df_pred.to_csv("./Results/predictions.csv", index = False)

print('This cell was last run on: ')
print(datetime.now())
```

```
This cell was last run on:
2020-11-26 12:12:17.490096
```

Interactive Visualization of predictions

Execution of the code below will start an interactive plot (as demonstrated below) in the default browser. On the right 3 dropdown selectors enable the user to choose of which model for which dataset and for which timestep the predictions shall be shown. To stop the interactive plot the kernel needs to be restarted.

```
! bokeh serve --show visualization.py
```



In [20]:
```
#! bokeh serve --show visualization.py
```

In [21]:
```
print('This cell was last run on: ')
print(datetime.now())
```

```
This cell was last run on:
2020-11-26 12:12:17.509143
```