How to Load & Run Models

Objective

Forecasting Problem where you want to predict the average tensiometer value (soil moisture) for a given day

Independent Features

Feature	Unit of Measure	Summary Values	
Temperature	Celsius (°C)	min, max, avg, med	
Air Humidity	Relative Humidity (% RH)	min, max, avg, med	
Solar Radiation	Surface Power Density (W/m²)	min, max, avg, med	
Rain	Millimetres per Hour (mm/h)	max, med, sum	
Irrigation	Liters per Minute (l/min)	sum	
Tensiometer	Millibar (mBar)	min, max, avg, med, sum	AR component
Tens. Residuals of LM	Millibar (mBar)	avg	MA component

All these features are measured for the 3 days preceding the one you want to predict, for a total of 66 features

Target Feature: Average Tensiometer Value for the given day

1) Load Libraries and Functions

Simply load the libraries and the manually defined functions by running the Python code

Libraries

```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import datetime
```

Functions

```
def calculate_aic(y_true, y_pred, num_features):
    resid = y_true - y_pred
    sse = np.sum(resid ** 2)
    aic = len(y_true) * np.log(sse / len(y_true)) + 2 * num_features
    return aic
```

2) Pre-process the Train Data

Run the manually defined function to pre-process the training data: data are cleaned, aggregated on a daily basis and shifted properly

```
data = pre_processing(file = 'data_sensors_rovere.csv')
```

Raw Data

	reading_id	sensor_id	timestamp	value	unit_id	reading_state_id	location_id
0	5216799	102	2023-04-28 00:00:00+00:00	13.28	1	1	58
1	5200138	101	2023-04-28 00:00:00+00:00	0.00	4	1	58
2	5325730	104	2023-04-28 00:00:00+00:00	62.64	7	1	58
3	5306003	55	2023-04-28 00:00:00+00:00	1.96	5	1	20
4	5216799	102	2023-04-28 00:00:00+00:00	13.28	1	1	58

Clean Data

	sensor_id	avg_tens	avg_tens_lag1	min_hum_lag1	min_temp_lag1
0	61	84.451389	79.012346	38.033333	11.066667
1	61	84.706667	84.451389	47.766667	11.550000
2	61	79.666667	84.706667	41.900000	10.350000
3	61	84.792453	79.666667	31.420000	10.080000
4	61	92.128205	84.792453	34.400000	12.833333

Steps in details:

a) Load the raw data with the following features required (the database table is indicated):

   i. *reading_id*: identification code associated with the reading
   ii. *timestamp*: date and time at which the sensor reading took place
   iii. *sensor_id*: identification code associated with the sensor
   iv. *value*: reading value for a given sensor_id at a specific timestamp
   v. *description*: type of sensor, defined in the **sensor** table
   vi. *group*: cluster to which the sensor belongs, defined as in the **names.csv** file

reading table

b) Round down each timestamp to the nearest date, thus cutting off the hour component of each day

c) Duplicate the dataset so that tensiometers in the same group with different depths are correctly processed separately from each other (this step could be changed to make the code more usable)

d) Aggregate the dataset values by date, sensor_id and group by calculating the following summary measures for each sensor and day: minimum, maximum, mean, median and sum.

e) Perform a pivotal transformation based on date and group, creating a new dataset where the unique values of description become the columns and the summary measures populate the cells.

f) Delete summary measurements that are not of interest for the analysis, such as wind measurements. Therefore, leave only the measures in according to the features table.

g) Remap the sensors according to the artificially created groups in step c)

h) Linearly impute any missing values, making sure that these correspond to an average between the previous day and the day after the one not present.

i) Recreate a dataset of values shifted up to 3 days from the one of interest

j) Delete data on superfluous dates, i.e. the first 3 days (28, 29, 30 April) and the last one (15 October)

k) For each sensor, train a linear regression model in order to calculate the residuals that will then be used as components in the ARIMAX models. For each model, feature selection was by bidirectional stepwise selection based on Akaike's information criterion (AIC).

3) Pre-process the Target Data

Run the manually defined function to pre-process the target data: data are cleaned, aggregated on a daily basis and shifted properly

```
X_target = pre_processing_target(75, target_example, data)
```

Raw data must refer to observations from 3 days before the date on which the tensiometer value is predicted. In addition, the raw data must refer only to the target tensiometer values and their associated sensors, i.e. those of the nearest irrigator and weather station.

To understand precisely which irrigator and weather station are associated with the tensiometer, please refer to the **names.csv** file, but insert just one tensiometer per group.

Steps in details:

a) Enter the raw data for a specific group, considering only the target tensiometer, with the values of 3 days before the date on which the tensiometer value is predicted; the following features are required:
   i. *timestamp*: date and time at which the sensor reading took place
   ii. *sensor_id*: identification code associated with the target tensiometer
   iii. *value*: reading value for a given sensor_id at a specific timestamp
   iv. *description*: type of sensor, defined in the **sensor** table

b) Excluding the steps involving the group variable (steps c, g, h, j), the steps are the same as in the pre-processing for train data.

Example

If we want to predict the value of tensiometer number **75** (depth 60) for **10 May 2023**, we will have to take into account its values over the previous 3 days (7, 8, 9 May) and those of the associated sensors:

	timestamp	sensor_id	value	description	group
0	2023-05-07 00:00:00+00:00	55	0.34	Avg wind speed (1h)	3
1	2023-05-07 00:00:00+00:00	102	17.76	Avg Temperature (1h)	3
2	2023-05-07 00:00:00+00:00	104	72.06	Avg Relative Humidity (1h)	3
3	2023-05-07 00:00:00+00:00	58	0.00	Avg solar radiation (1h)	3
4	2023-05-07 00:00:00+00:00	101	0.00	Rain accumulated (1h)	3
5	2023-05-07 00:00:24+00:00	126	0.00	irrigation	3
6	2023-05-07 00:01:26+00:00	126	0.00	irrigation	3
7	2023-05-07 00:02:28+00:00	126	0.00	irrigation	3

- **Tensiometer**: sensor 75
- Temperature: sensor 102
- Air Humidity: sensor 104
- Rain: sensor 101
- Irrigation: sensor 126
- Wind Speed: sensor 55 (not essential)

Data for sensor 73 (depth 30) are then not entered.

Output of the data required for prediction after pre-processing:

	avg_tens_lag1	min_hum_lag1	min_temp_lag1	min_solar_lag1	min_tens_lag1	max_hum_lag1	max_temp_lag1	max_solar_lag1	max_rain_lag1	max_tens_lag1	...	med_temp_lag3	med_solar_lag3	med_rain_lag3	med_tens_lag3	sum_rain_lag3	sum_irr_lag3	sum_tens_lag3	residuals_lag1	residuals_lag2	residuals_lag3
0	98.268908	65.6	14.2	0.0	95.0	96.433333	21.02	636.0	0.48	100.0	...	19.273333	0.0	0.0	93.0	0.0	0.0	13409.0	-57.983409	-60.38565	-62.483086

1 rows × 66 columns

It actually corresponds to a single one-dimensional array associated with the forecast date, i.e. **10 May**

4) Forecasting

Run the model_predict function indicating the reference sensor, the pre-processed data and the variable selection strategy.

```
prediction = model_predict(75, X_target, data, fs_strategy = 'aic')
```

you will obtain the tensiometer value for the target date

123.371436

Steps in details:

a) Enter the following parameters in the appropriate function:

   i. *sensor_id*: identification code associated with the target tensiometer

   ii. *X_target*: features associated to the target tensiometer, corresponds to the output of the function **pre_processing_target**

   iii. *data_train*: preprocessed dataset on which the model will be trained, corresponds to the output of the function **pre_processing**

   iv. *fs_strategy*: strategy for feature selection, it can take the value *'aic'* for a selection based on Akaike's criterion or the value *'pls'* for a strategy based on Partial Least Squares type dimensionality reduction

b) Define a dictionary for the tensiometer clusters associated with each target tensiometer, based on the results of the report

c) Search for the best subset that shows the best performance in terms of RMSE on the train dataset. For the PLS strategy, new features are also calculated again based on RMSE minimization.

d) Perform the feature selection

e) Run the model

f) Compute the prediction for the day of interest (forecasting)