Boosting Research for a Smart and Carbon Neutral Built Environment with Digital Twins – **SmartWins**

CERTH
CENTRE FOR RESEARCH & TECHNOLOGY HELLAS

Machine Learning Models: Integration and Deployment

Aristeidis Mystakidis,

PHDc, MBA, MSc, MEng,

CERTH

CERTH SmartWins Training Sessions: Day3
25 April 2024
Thessaloniki

Funded by the European Union

This project has received funding from the European Union's Horizon research and innovation programme under grant agreement No 101078997

SmartWins

Importance of Energy Forecasting

Why accurate energy forecasting is crucial for various sectors (energy utilities, manufacturing, transportation, etc.)?

What is the impact on decision-making, resource allocation, and cost management?

Energy Forecasting

- is critical in the design and operation of power systems.
- helps energy suppliers to anticipate electricity use and prepare for future power demands.
- supports distribution and transmission system operators
- has received a lot of attention in recent years

Types of Energy Forecasting Techniques

Energy (or Power) Load/Generation Forecasting (ELF or EGF) types:

- PV Generation Forecasting
- Wind Generation Forecasting
- Electricity Load Forecasting
- Thermal (Heating – Cooling) Load Forecasting

Infrastructures:

- Smart Buildings
- Universities
- Laboratories
- PV Parks
- Wind Turbines
- Machines (Fridge, Washing Machine, Oven etc.)

Models

Statistical
- ARIMA
- Exponential Smoothing

Machine (Shallow) Learning
- Popular Tree-based models like XGBoost, Random Forest, CatBoost, LGBM
- Low memory, high speed models like LGBM, Ridge regression
- Other classic ML models like Support Vector Regressor

Deep Learning
- Long-short term memory recurrent neural network (LSTM RNNs)
- Gated recurrent unit recurrent neural network (GRU RNNs)
- Multi-layer Perceptron (MLP)
- Convolutional neural network (CNN)
- Sequence to Sequence recurrent neural networks (Seq2Seq RNNs)

Parameters to be investigated

Weather Data	Time Features - Seasonality	Miscellaneous	Sensors
Solar Radiation	Hour of day	Occupancy	Inside Temperature
Outside Temperature	Day of the week	Reheat Coefficient	Door Open/Closed
Wind Speed	Minute of hour	Covid - 19	
Relative Humidity	Day of month	Holidays	
Cloud Cover	Month or year	Building Details	

Data preparation

Are the requested data available on a real time basis?

- Analyze the energy data source
  - Generation (PV, Wind etc.)
  - Load (Electricity, Heating, Cooling etc.)
- Weather Data preparation
  - Check weather APIs (MeteoStat, VisualCrossing etc.)
  - Decide about weather forecast
- Other sources (occupancy, holidays, sensors etc.)
- Check time periods - resolution
- Validate each dataset

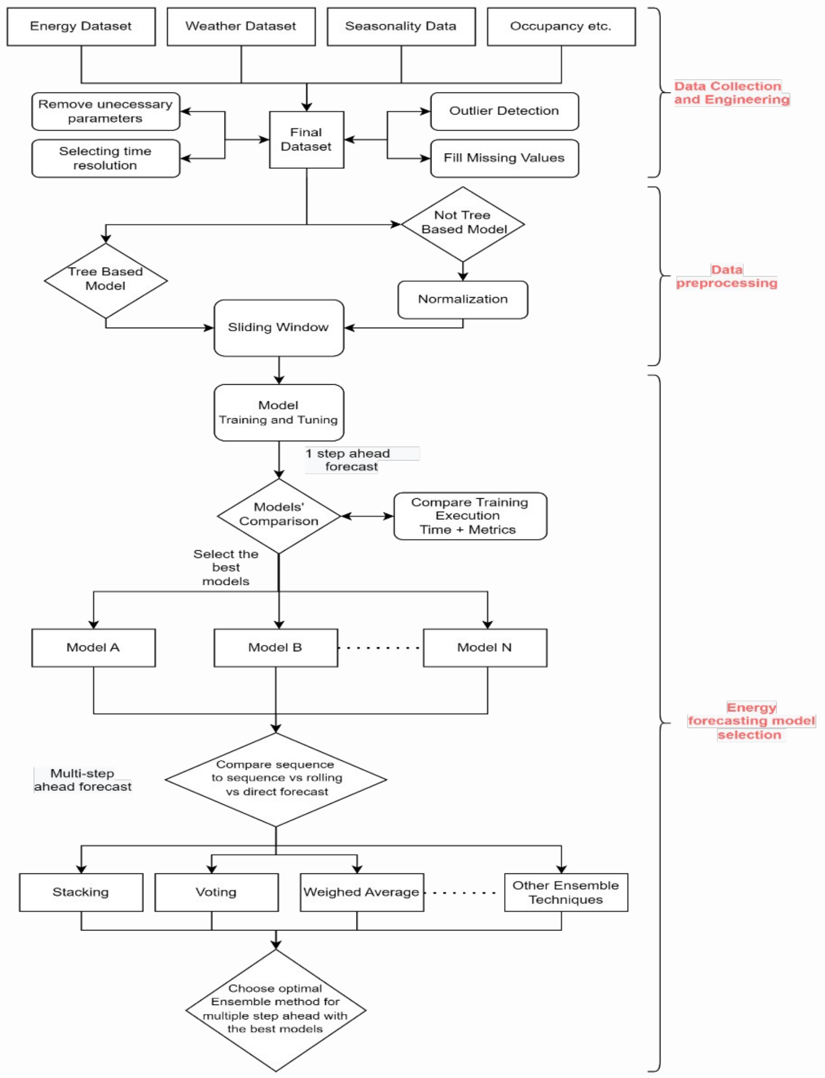
Data merging from various sources (Energy, Weather etc.)

- Define the communication with the Platform
- Understand the data sources:
  - APIs
  - Databases
  - .csv, .xlsx files etc.
- Choose merging technique
  - Inner/outer/right/left joining
  - concatenation
  - appending
- Identify merging key (Timestamp etc.)
- Validate the merged dataset

Timestamp formatting

- Understand time zones
- Local to UTC or UTC to Local: Converting timestamps between local time and UTC (Coordinated Universal Time) is a common task in handling time-related data
- Consider daylight saving time changes
- Historical time zone differences
- Use libraries and modules like:
  - datetime
  - pytz
  - strftime()
- Deliver the prediction on the requested timestamp format and time zone

Typical workflow of experimental forecasting modelling



Energy Dataset
Weather Dataset
Seasonality Data
Occupancy etc.
Remove unecessary parameters
Selecting time resolution
Final Dataset
Outlier Detection
Fill Missing Values
Data Collection and Engineering
Not Tree Based Model
Tree Based Model
Normalization
Sliding Window
Data preprocessing
Model Training and Tuning
1 step ahead forecast
Models' Comparison
Compare Training Execution Time + Metrics
Select the best models
Model A
Model B
Model N
Energy forecasting model selection
Multi-step ahead forecast
Compare sequence to sequence vs rolling vs direct forecast
Stacking
Voting
Weighed Average
Other Ensemble Techniques
Choose optimal Ensemble method for multiple step ahead with the best models

Source: CERTH material

Overall Architecture:

- Data Collection and Engineering
- Data preprocessing
- Energy forecasting model selection for One Step ahead
- Best multi-step ahead strategy identification
- Best possible ensemble technique selection

Energy Forecasting from different pilot sites

The problem to be solved:

- Generation Forecasting →PV, Wind
- Load Forecasting→ Heating, Cooling, Electricity
- 1st Step: One hour, 15 minutes, One Day
- Multi-step: 24 hours ahead (24 or 96 steps)

Technologies tested

- Machine (Shallow) Learning Models
  - Popular Tree-based models like XGB, CatBoost, Random Forest
  - Low memory, high speed models like LGBM
  - Other classic ML models like Support Vector Regressor
- Deep Learning Models
  - Long-short term memory recurrent neural network
  - Gated recurrent unit recurrent neural network
  - Multi-layer Perceptron
  - Sequence to Sequence recurrent neural networks

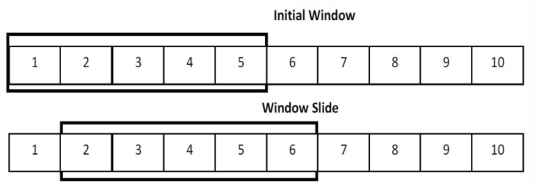
Core Techniques utilized

- Sliding window
- Rolling (Recursive) or Direct Strategy

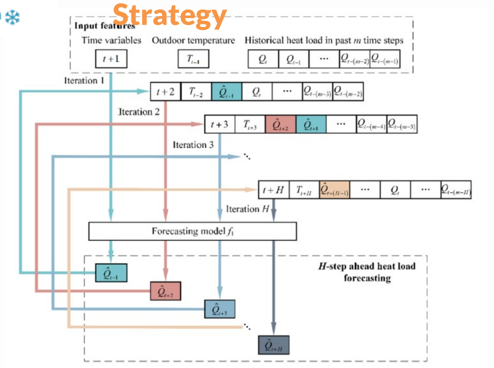
Examples of Final Implementations utilized

1) Dynamical Weighed Average Ensemble Model based on LSTM, XGBoost and Random Forest's behaviour and Recursive Strategy or 2) Direct Strategy with 96 different Light Gradient Boosting model's implementation on an increased sized dataset.

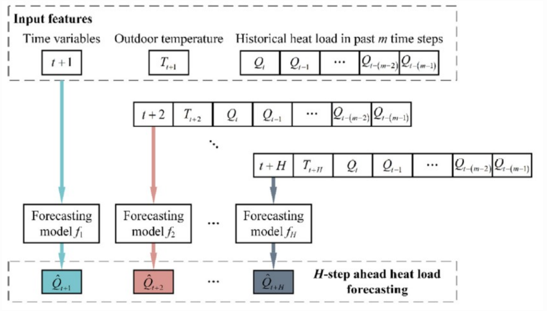
Sliding Window technique



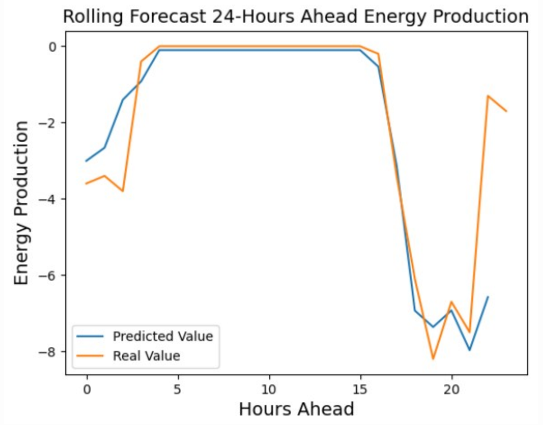
Recursive Strategy



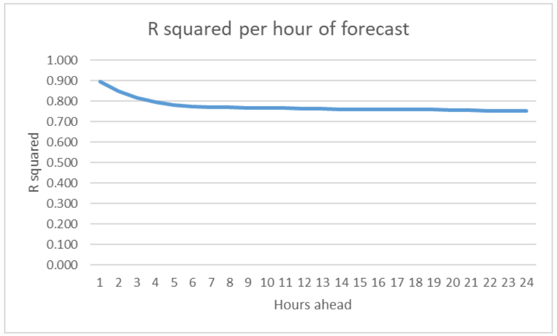
Direct Strategy



One Day ahead forecast example



Results' example



Model Save and Load (1/2)

Saving a scikit-learn model:

- Use the joblib library's dump function to save the model to a file
- Example: joblib.dump(model, 'model.pkl')

Loading a scikit-learn model:

- Use the joblib library's load function to load the model from the saved file
- Example: model = joblib.load('model.pkl')

Saving a TensorFlow model:

- Use the model.save() method to save the entire model or model.save_weights() to save only the model weights
- Example: model.save('model_tf') or model.save_weights('model_weights_tf.h5')

Loading a TensorFlow model:

- Use the tf.keras.models.load_model() function to load the saved model
- Example: model = tf.keras.models.load_model('model_tf')

Model Save and Load (2/2)

Benefits of Model Saving and Loading

- Enables reusability and reproducibility of trained models
- Facilitates deployment of trained models in production environments
- Saves time and computational resources by avoiding retraining the model from scratch

Considerations

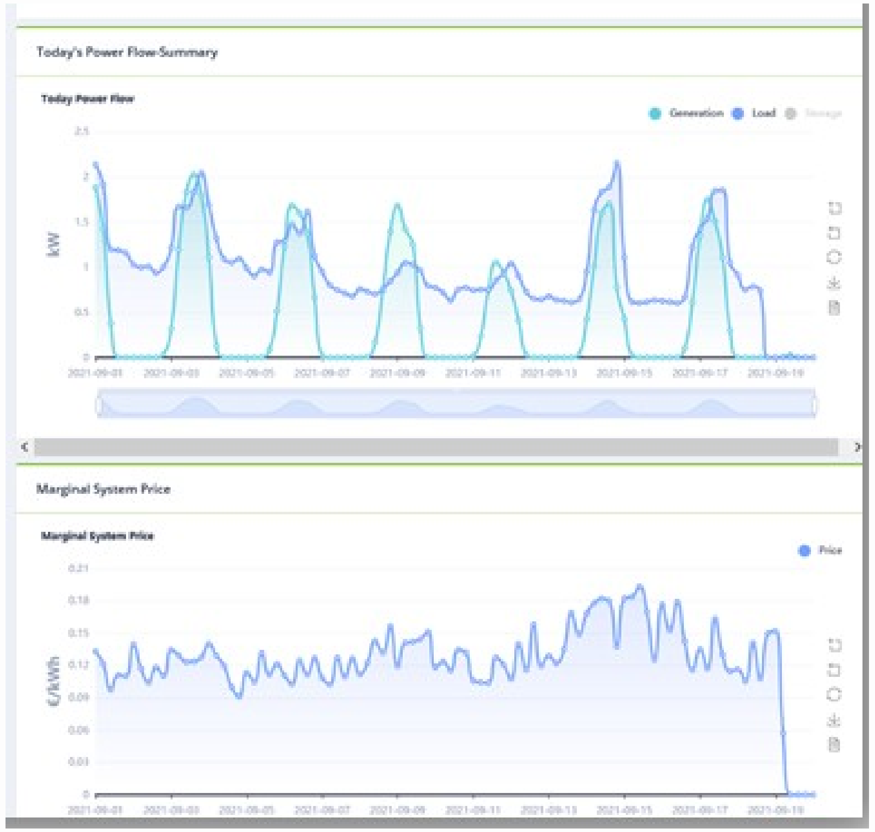
- Ensure compatibility between the library versions used for training and loading models
- Pay attention to file size and storage requirements, especially for large models in TensorFlow

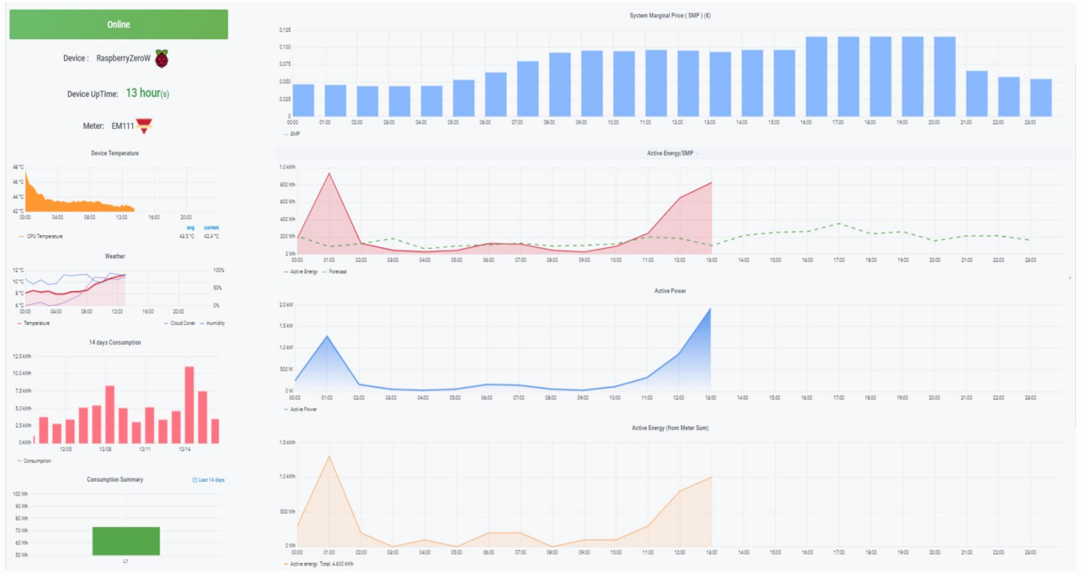
Best Practices

- Save both the model architecture and weights for TensorFlow models to ensure complete reproducibility
- Version control saved models along with the code using tools like Git for better tracking and collaboration
- It is better for the models and the data to be saved on a different Git repository

Scalers can be saved too!

Platform Integration

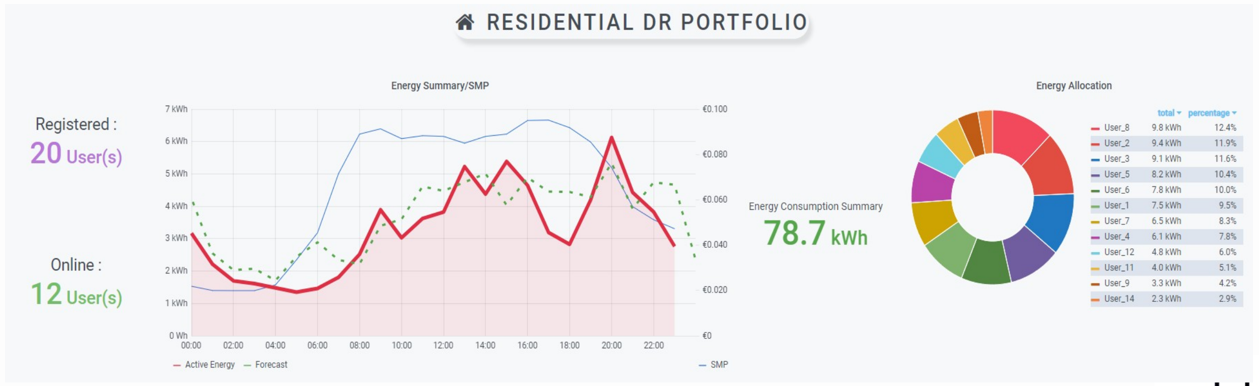



Online

Device : RaspberryZeroW

Device UpTime: 13 hour(s)

Meter: EM111


RESIDENTIAL DR PORTFOLIO

Registered : 20 User(s)

Online : 12 User(s)

Energy Consumption Summary 78.7 kWh

Energy Allocation

	total	percentage
User_8	9.8 kWh	12.4%
User_2	9.4 kWh	11.9%
User_3	9.1 kWh	11.6%
User_5	8.2 kWh	10.4%
User_6	7.8 kWh	10.0%
User_1	7.5 kWh	9.5%
User_7	6.5 kWh	8.3%
User_4	6.1 kWh	7.8%
User_12	4.8 kWh	6.0%
User_11	4.0 kWh	5.1%
User_9	3.3 kWh	4.2%
User_14	2.3 kWh	2.9%

Possible integration cases:

Real time integration:

- Pilot site provides real time energy data → Depending on the time step (5 minutes, 15 minutes, 1-hour etc.), the dataset is updated, and each prediction is based on the recent energy values.
- Usually, Sliding Window is performed including all or most of the final parameters.

No real time integration:

- Pilot site does not provide real time energy data → Dataset is not updated (at least not regularly), and each prediction is not based on the recent energy values.
- In such cases, either simple regression models were implemented, or partial Sliding Window is performed including parameters like weather, seasonality (not Energy).
- No real time integration is a great tool if the data management system is down as a backup on a real time integration implementation

Synthesize real time lagged values

In time-series data, relationship between independent variables and the predicted value can vary over time

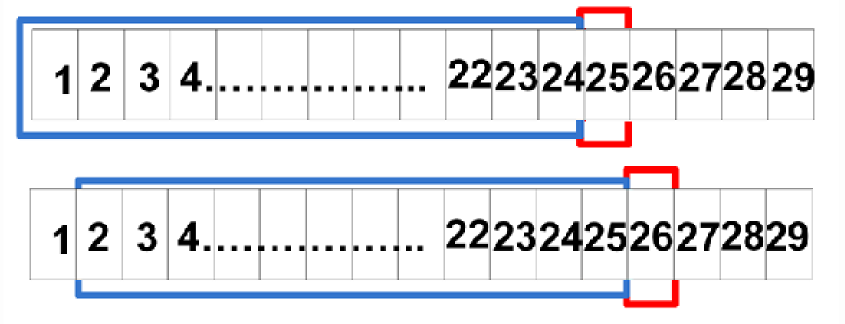
To capture these dynamics, time lagged features are synthesized → incorporating past values of the variables as inputs to the model

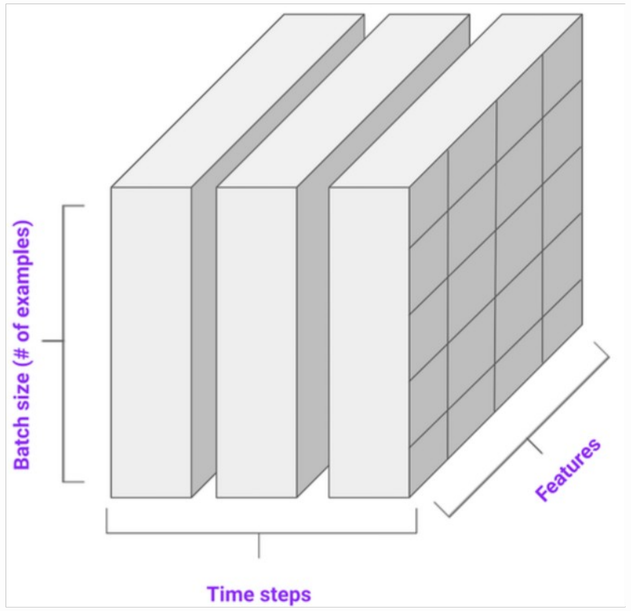
Enhances learning from historical patterns → improves predictive performance.

Most prominent technique: Sliding Window

Often time-series data are shifted (24/48/96 steps) ahead based on the asset's resolution, in order to produce datasets with the information of the previous day

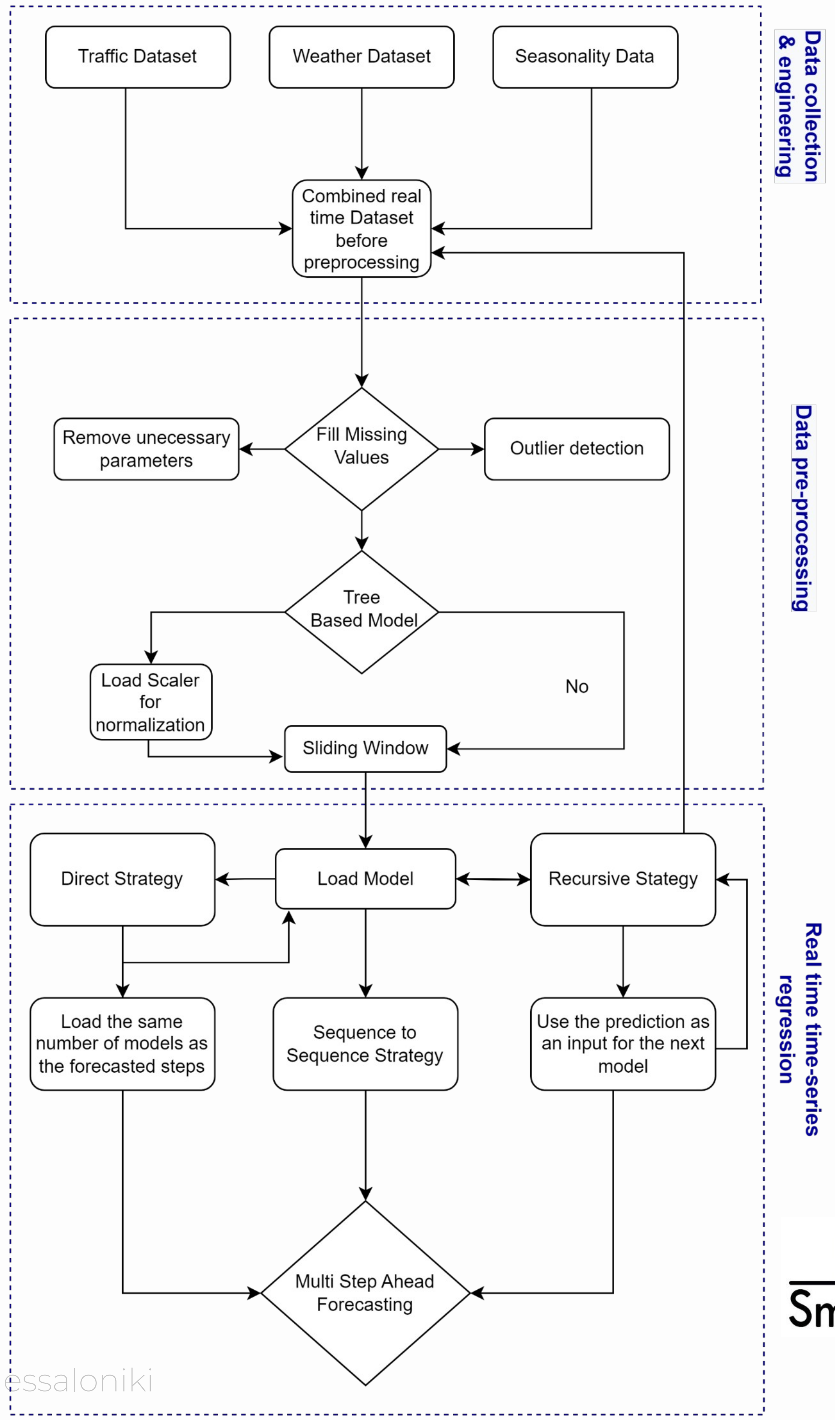
- 3D for LSTMs
  - **time-steps**
  - **rows**
  - **parameters**
- 2D for others
- (Rows, parameters for all time-steps)





Implementation steps

- Real time data collection
- Fast preprocessing
- Real time timeseries regression



Traffic Dataset
Weather Dataset
Seasonality Data
Combined real time Dataset before preprocessing
Data collection & engineering
Remove unecessary parameters
Fill Missing Values
Outlier detection
Tree Based Model
Load Scaler for normalization
No
Sliding Window
Data pre-processing
Direct Strategy
Load Model
Recursive Stategy
Load the same number of models as the forecasted steps
Sequence to Sequence Strategy
Use the prediction as an input for the next model
Multi Step Ahead Forecasting
Real time time-series regression

Model Versioning, Management, Monitoring and Maintenance

Importance of versioning and managing energy forecasting models:

- Tracking changes to model configurations and hyperparameters
- Ensuring reproducibility and consistency in forecasting results
- Tools and best practices for model versioning and management (GitLab, GitHub etc.)

Specific infrastructure requirements for deploying energy forecasting models:

- Scalability to handle large volumes of data
- Reliability to ensure continuous operation
- Cost considerations, especially for cloud-based solutions vs. on-premises infrastructure

Model Versioning, Management, Monitoring and Maintenance

Specific monitoring considerations for deployed energy forecasting models:

- Monitoring forecast performance over time
- Detecting and addressing concept drift in data distributions
- Ensuring fairness and transparency in model predictions

Consider getting help and documentation from various sources

- AI based tools (ChatGPT, Claude 2, Bing AI)
- Websites (Stack Overflow, Medium, Reddit, Kaggle)
- Online Platforms (Udemy, W3Schools, Coursera)

Retrain cycles after deployment is important!

- Retrain after specific period (every month, 3 months, year etc.)
- Retrain after a validation week, if metrics are worse (be aware of holidays, covid, emergencies etc.)
- Retrain and compare a secondary solution as a backup

Boosting Research for a Smart and Carbon Neutral Built Environment with Digital Twins – **SmartWins**

Project Partners

ktu kaunas university of technology 1922

CERTH CENTRE FOR RESEARCH & TECHNOLOGY HELLAS

POLITECNICO MILANO 1863

Contecht

INNOTROPE

Funded by the European Union

This project has received funding from the European Union's Horizon research and innovation programme under grant agreement No 101078997

SmartWins