DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Control of Communication and Energy Networks

Report

Waste-to-Energy Framework: An intelligent energy recycling management

Ludovica Cartolano
cartolano.1796046@studenti.uniroma1.it

Supervised by
PhD Danilo Menegatti

Master in Control Engineering

Department of Computer, Control and Management Engineering
"Antonio Ruberti" (DIAG)
University of Rome "La Sapienza"
AY 2023/2024

Contents

List of Figures

Chapter 1

Introduction

In the following work it will be described and implemented the problem of Waste to Energy (WTE) framework and its role in the energy network.

How it is possible to infer from article [Binder 2020], the need for an increase in productivity, product quality and process safety along with economic and environmental sustainability has led to progress in software tools and Machine Learning (ML) algorithms which, combined with big-data availability, helped to enhance process monitoring and prediction in industrial settings.

To predict data in process industry, data-driven algorithms are crucial tools for monitoring variables that are difficult, if not impossible, to measure directly, such as heating value of biomass WTE plants. With the help of ML models, one can predict such variables with high accuracy.

In fact, this is the case of study of the article [Kaya et al. 2021], which proposes different ML models to predict the amount of waste to be used for smart energy systems in the European side of Istanbul, Turkey.

Biomass is a key renewable energy source, alongside hydro-power, solar, wind, and geothermal power [Binder 2020]. Even though hydro-power is the most widely used energy source in many countries, the surrounding natural ecosystems are often negatively impacted by hydro-power dams [Binder 2020, Abidur Rahman 2022].
The advantages of WTE include widespread availability, storability (in the form of bio-fuel), and reliability compared to other intermittent renewable sources.
Generally speaking, it is possible to deduce from the work [Abidur Rahman 2022] that integrating renewable energy sources into a single structure is the best way to use them to increase their reliability. For instance, creating a wind farm by hybridizing it with a WTE plant [Alireza Tajeddin 2019].
Despite its benefits, Biomass energy faces challenges such as waste collection, pollutant emissions (NOx, SO2), high investment costs, and the risk of deforestation [Binder 2020, Abidur Rahman 2022]. On the other hand, the increasing global waste due to population growth and urbanization necessitates effective waste management systems [Binder 2020]. Big cities, in fact, suffer from many problems, such as traffic, air and water quality, pollution, waste ect. Smart cities solutions are proposed to overcome these issues using information and communication technologies [Kaya et al. 2021]. In particular, management of Municipal Solid Waste (MSW) with an integrated system is essential to protect human and environmental health, increase resource efficiency, and ensure sustainability [Kaya et al. 2021].

WTE technologies, categorized into: direct combustion, burning organic mate-

rial for heat producing CO2 and water; biological treatment, turning organic materials into fuels through processes like fermentation; chemical treatment, converting biomass into liquid fuels via chemical reactions; and thermo-chemical treatment, creating various fuels from biomass through high-heat processes like gasification and pyrolysis. These methods have gaining popularity for sustainable energy recovery and environmentally friendly waste disposal [Binder 2020, Abidur Rahman 2022].
In a specific application through the use of the Outotec Advanced Staged Gasifier within a Waste-to-Energy plant [Binder 2020]. The solid waste is converted into syngas through gasification in a fluidized bed, followed by combustion in the presence of air and NOx reduction agents. This process generates heat for steam and electricity production. The plant incorporates mechanisms for removing contaminants and recovering heat from flue gas, aiming to minimize environmental impact while maximizing energy efficiency.

In the article [Kaya et al. 2021], the authors have built an ensemble model, Gradient Boosting (GB), to predict the amount of MSW using daily data related to other variables such as seasonality and socio-economic status. Then they have used the calorific index value to predict generated energy from solid waste categorized in 14 different waste types.

In the first stage they have proposed WTE framework by employing ML models for MSW prediction using real-life datasets of four MSW transfer stations via Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) performance metrics.

The models are: Multilayer Perceptron Network (MLP), Support Vector Machine (SVM), Random Forest (RF), Extra Trees (ET), and Gradient Boosting (GB). They have observed that GB produces significantly better results.

In the second stage they have performed the calorific value calculations for the 14 waste types on the European side of Istanbul and combine them with the weighted average for four MSW transfer stations.

By combining these two phases, the WTE framework is obtained. It is then able to make predictions and provide thermal energy potential forecasting service through affine equations for the four MSW transfer stations

For my personal implementation has been used Multilayer Perceptron (MLP) for regression and Support Vector Machine for regression (SVR) that will be trained on dataset created by hand merging the tables described in [Kaya et al. 2021] and one has used Transfer Learning for second stage on pre-trained dataset.

Last, it is possible to integrate the energy produced in the energy network by following the same line as the article [A. Joshi and Glielmo 2023]. This letter offers an innovative exploration into the decentralized scheduling of shared energy storage systems (SBESS) among residential communities, leveraging multi-agent reinforcement learning (MARL) to create efficient and scalable control strategies. By framing the interaction between households as a Markov Game and adopting a consensus-based Tabular Q-learning approach.

Chapter 2

State of the Art

In the next chapter it will be described the Waste-to-Energy Framework.
The problem defined in [Kaya et al. 2021] has been split in two stages:

- First Stage - Prediction of the Daily Solid Waste Amount: In this stage, machine learning algorithms are employed to predict the daily solid waste amounts using datasets from multiple Municipal Solid Waste (MSW) transfer stations. This involves the use of different ML models to achieve accurate predictions, which are crucial for the planning and operational efficiency of waste management.

- Second Stage - Calculation of Calorific Value (CV) for Thermal Energy Potential: After predicting the daily solid waste amount, the next stage involves calculating the calorific value of the waste. This is done by utilizing waste characterization studies that determine the energy potential of the waste. The calorific value represents the amount of energy that can be generated through the thermal conversion of the waste.

These two stages combine predictive analytics with practical energy calculations to optimize the Waste-to-Energy conversion process, aiming to enhance both the economic and environmental outcomes of waste management systems.
Thanks to this the amount of energy that can be obtained from waste by thermal energy conversation can be found in the daily period in advance.

2.1 ML Models for MSW Prediction

The problem can be formulated as a Regression Problem. As it is well known this can be done by using different supervised learning method such as Multilayer Perceptron model, SVM model, and a Transfer Learning approach.

In supervised learning the algorithm learns from labeled training data, helping to predict outcomes from unforeseen data. Given X instances and Y the labels, it involves to learn a function $f : X \to Y$ through a dataset $D = \{(x_i, t_i)_1^N\}$ where t_i are the target values. Learning a function means to compute an approximated function $\hat{f}$ on the bases of the training set D such that it returns values as close as possible to f especially for unforeseen samples x.

Follows the description of the models that have been used for implementation in my code [Cartolano 2024].

2.1.1 Multilayer Perceptron (MLP)

Perceptron is supervised learning algorithm that learn a function $f(\cdot) : \mathbb{R}^m \rightarrow \mathbb{R}^O$ by training on a dataset where m is the number of dimensions for the input and O is the number of dimensions of the output.

It takes as input multiple real-value features $X = x_1, x_2, ..., x_m$, and produces a single binary output based on those inputs. The input depend on weights, which represent the strength of the connection between the input and the neuron. Also consider a bias term, which adjusts the threshold for the output to be activated. As one can see in Fig.(2.1), the output y is calculated using the weighted sum of the inputs ($z = w_0 \sum_1^N w_i x_i$) followed by the application of an activation function. y can learn a non-linear function approximator for regression.

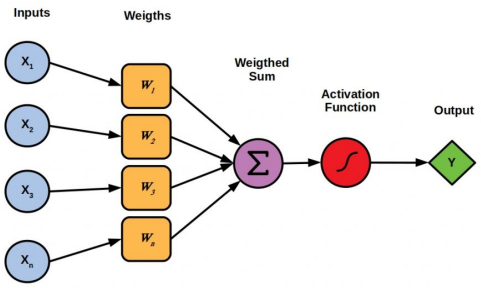


Figure 2.1: Perceptron Network

In Multiplayer Perceptron, between the input and the output layer, there can be two or more non-linear layers, called hidden layers. This structure is shown in Fig.(2.2).

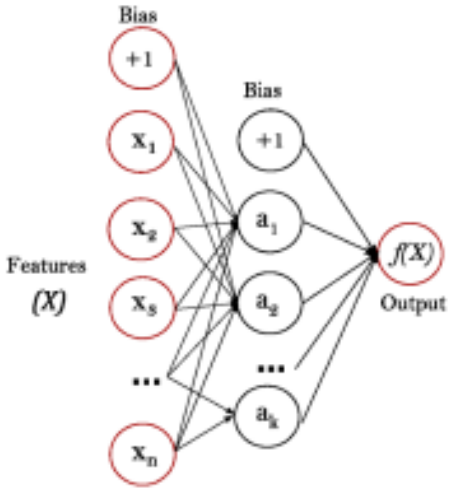


Figure 2.2: Multilayer Perceptron Network

For simplicity, one decides to use two hidden layers.
Follows, MLP hyperparameters and their values used for tuning:

- hidden layer sizes: It specifies the size and structure of the hidden layers within the network. Each tuple within the list represents a different configuration of hidden layers that can be used in the MLP.
  `[(50,), (100,), (50, 50), (100, 50)]`
  The tuple means that there are two hidden layers with different neurons values.

- activation: It is the activation function of hidden layers.
  `['relu', 'tanh']`

- solver: The solver for weight optimization.
  `['sgd', 'adam']`
  Where one wants to choose between Stochastic Gradient Descent (Appendix (A.1)) and Adam optimizer (Appendix A.2).

- momentum: Momentum for gradient descent update. Should be between 0 and 1. Only used when `solver = 'sgd'`.
  `[0.1, 0.5, 0.9]`

- learning rate init: The initial learning rate used. It controls the step-size in updating the weights.
  `[0.01, 0.05, 0.1]`

- learning rate: Learning rate schedule for weights updates "constant" is a constant learning rate given by "learning rate init". "adaptive" keeps the learning rate constant to "learning rate init" as long as training loss keeps decreasing.
  `['constant', 'adaptive']`

2.1.2 Support Vector Machine - SVM

SVM are powerful tools for classification and regression tasks. The key idea is to find a solution through a decision hyperplane that best separates the data points belonging to different classes while finding the maximal margin.

The principal of maximal margin is about finding the hyperplane that separates the classes in the feature space with the largest possible margin while minimizing the classification error. The margin is defined as the distance between the hyperplane and the nearest data points from both classes. As it is possible to see in Fig.(2.3), the maximal margin is illustrated as the widest possible strip that separates classes without containing any data points.

To explain how SVM works, consider a function $f : X \to \{+1, -1\}$, with dataset $D = \{(x_n, t_n)_1^N\}$ linearly separable with $t_n \in \{+1, -1\}$ target values, and a linear model $y(x) = w_0 + w^T x$; so there exists w_0, w such that $y(x_n) > 0$ if $t_n = +1$ and $y(x_n) < 0$ if $t_n = -1$.
Let $x_k \in D$ be the closest point to the hyperplane $\bar{h} : \bar{w}^T x + \bar{w_0} = 0$; the margin is computed as $\frac{|y(x_k)|}{\|w\|}$.
The optimal hyperplane $h*$ that maximizes the margin is given by

$$w*, w*_0 = argmax_{w,w_0} \frac{1}{\|w\|} min[\ t_n(w *^T x_n + w*_0)]$$

When $h*$ is found, then there will be at least two closest points x_{k_+} and x_{k_-} called Support Vectors, such that $w *^T x_{k_+} + w*_0 = +1$ and $w *^T x_{k_-} + w*_0 = -1$. These play a critical role in determining the optimal hyperplane.

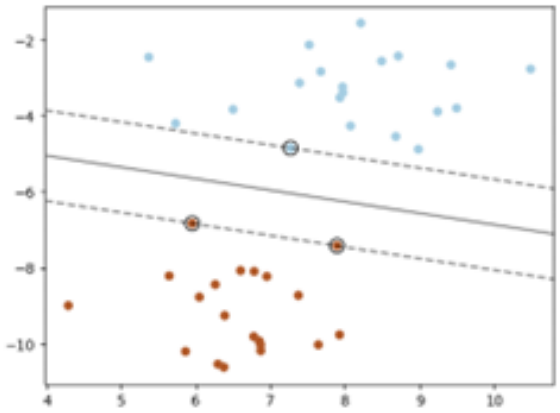


Figure 2.3: Example of Support Vector Machine (SVM)

To use this method in the context of Regression Problem in real-world numbers one needs to add a tolerance margin (Soft Margin). $\xi_n \geq 0$ are introduced, called *slack variables* and the Soft Margin constraint defined as $t_n y(x_n) \geq 1 - \xi_n$ with $n = 1, ..., N$.
However the idea stays the same: minimize the objective function, to which the slack variables have been added and find the hyperplane that maximises the margin

$$w*, w*_0 = argmin \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N} \xi_n$$

SVM can efficiently handle non-linear decision boundaries by using the kernel trick (Appendix (B.1)) which maps the input features into a higher-dimensional space, by computing the dot product of the transformed instances without explicitly calculating the transformation using a function called *kernel* ($k(x, x')$ where x' is an unforeseen instance). This helps the intrinsic linearity of the SVM model to capture complex relationships in the data.
This is done through the so called, basis functions that have been chosen to be:

- Polynomial: $k(x, x') = (\alpha x^T x' + \beta)^d$ with d degree of the polynomial.
- Radial Basis Function (RBF): $k(x, x') = exp(-\alpha|x^T x'|^2)$.

Follows, SVM hyperparameters and their values used for tuning:

- kernel: Chooses between basis functions.
  `['rbf', 'poly']`
- C: Penalty parameter of the error term.
  `[0.01, 1, 10]`
- gamma: Gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'.
  `[0.1, 1, 10`
- degree: The degree of polynomial kernel function.
  `[2, 3, 4]`

2.1.3 Transfer Learning

Transfer learning is a method in machine learning that involves leveraging knowledge from one problem to solve a related but different problem. This technique is particularly beneficial in situations where there is a shortage of labeled data for the target task, but an abundance of data is available for a related task; which is the case of the article [Kaya et al. 2021] considered.

In transfer learning, there are two key areas: the source domain and the target domain. The source domain contains a large amount of labeled data, while the target domain has significantly less. Knowledge is transferred from the source domain, where a model is initially trained, to enhance performance in the target domain. This transfer can improve predictive accuracy, speed up model training, and reduce the amount of data needed for effective training in the target domain.

The domains in transfer learning are characterized by their feature spaces and marginal probability distributions. When these aspects differ between two domains, they are considered distinct. Tasks within these domains are defined by their label spaces and the predictive functions derived from the feature spaces.

There are various forms of transfer learning, including:

- Inductive Transfer Learning: The target task differs from the source task, although the domains may be the same.
- Transductive Transfer Learning: The source and target tasks are the same, but their domains differ, typically in their data distributions.
- Unsupervised Transfer Learning: Both tasks involve unsupervised learning but are not necessarily identical.

Several methods can be employed to implement transfer learning, depending on the problem specifics:

- Instance Transfer involves reusing data instances from the source domain, possibly with adjustments to better suit the target task.
- Feature Representation Transfer aims to find a feature representation that minimizes the differences in data distributions between domains.
- Parameter Transfer involves sharing parameters or priors between models in different domains.
- Relational Knowledge Transfer focuses on transferring relational knowledge from the source to the target domain.

Transfer learning is extensively used in fields such as natural language processing, speech recognition, and image classification. It helps in learning with less labeled data, shortens training times, and enhances the performance of models on the target task. However, differences in feature spaces and data distributions between the source and target domains can make the transfer process challenging.

A way to implement Transfer Learning is by using `TensorFlow`.
TensorFlow provides tools and libraries for building, training, and deploying Neural Networks (NN) efficiently, enabling developers and researchers to create and experiment with complex machine learning models.

Neural Networks are a type of machine learning model inspired by the structure and function of the human brain. They consist of interconnected nodes (neurons) arranged in layers. Each neuron takes input, performs a mathematical operation on it, and passes the result to the next layer. Neural networks can learn from data by adjusting the strengths of the connections (weights) between neurons to minimize the difference between predicted and actual outputs.
Follows, NN hyperparameters and their values used for tuning:

- `units`: it defines the number of neurons (units) in the densely connected hidden layers of the neural network. The range specified for tuning is from 32 to 512, with a step size of 32. This means that during the hyperparameter tuning process, different values within this range will be explored to find the optimal number of units for each hidden layer.

- `dropout`: it is a regularization technique used to prevent over-fitting in neural networks. It randomly drops a fraction of input units during training, effectively "turning off" some neurons, which helps prevent the network from relying too much on any individual neuron. The dropout hyperparameter specifies the dropout rate, which is the fraction of input units to drop during training. The range specified for tuning is from 0.0 to 0.5, with a step size of 0.1.

- `learning_rate`: it determines the step size at which the model weights are updated during training. It's a crucial parameter for controlling the convergence of the model during training. The range specified for tuning is from 1e-4 to 1e-2, with a sampling method set to 'log'. This means that the values will be sampled logarithmically within this range, which is common for learning rates as they often span several orders of magnitude.

Moreover one will use in the NN the Adam optimizer with a tunable learning rate. During the hyperparameter tuning process, the Keras Tuner (which is a library that helps to automatically find the best hyperparameters for the machine learning model when using Tensor Flow) will search through different combinations of these hyperparameters to find the configuration that minimizes the Mean Squared Error loss function (MSE defined in Eq.(2.1)).

2.2 Waste-to-energy Prediction

In the case of study [Kaya et al. 2021], the county-based characteristics of the wastes in European side of Istanbul (as shown in Fig.(2.4)) are presented for winter in "Tables 1, 2, 3" shown in the article [Kaya et al. 2021] and for summer in "Tables 4, 5, 6" shown in the article [Kaya et al. 2021]. An example of these tables is shown in Fig.(2.5).

14 categories were used to categorize the wastes in order to determine their characteristics for the First Stage. Additionally, each county's wastes were analyzed in three socioeconomic classes: Low Income (`Low`), High Income (`High`), and Commercial Area (`Comm.`).

The second stage involves calculating the amount of thermal energy that can be extracted from MSW waste using the calorific index values of waste mixtures.

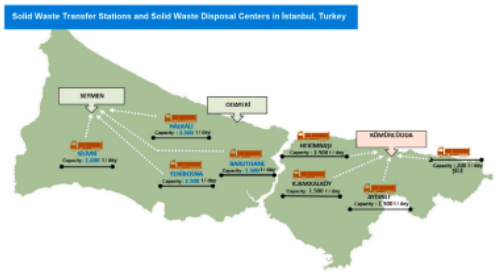


Figure 2.4: Location of waste management centers on the map of Istanbul

Table 1
Winter characteristic - I.

Socio-economic Level		ARNAVUTKOY			AVCILAR			BAGCILAR			BAHCELIEVLER			BAKIRKOY			BASAKSEHIR			BAYRAMPASA			BESIKTAS		
		Low	High	Comm.	Low	High	Comm.	Low	High	Comm.	Low	High	Comm.	Low	High	Comm.	Low	High	Comm.	Low	High	Comm.	Low	High	Comm.
	Waste components		%			%			%			%			%			%			%			%	
1	Paper	5.45	15.1	7.67	4.04	6.05	25.3	4.95	5.94	6.17	17.6	51.1	20.8	9.83	10.8	17.4	18.1	9.87	32.0	9.92	9.59	8.88	17.32	20.26	38.71
2	Glass	4.24	4.37	7.84	2.11	3.94	5.46	1.51	4.62	4.68	5.33	1.58	2.93	7.91	11.1	3.62	3.72	5.41	4.53	3.67	4.35	6.10	5.59	2.92	5.45
3	PET bottles	0.53	1.72	0.59	0.81	0.61	1.69	0.31	0.24	0.46	1.67	2.92	1.36	3.63	1.43	1.56	0.82	1.25	2.15	0.90	1.09	0.37	9.33	3.75	1.21
4	Plastic bag	6.89	13.0	6.96	7.11	27.0	6.30	7.37	5.47	8.36	7.35	11.1	7.68	7.20	7.35	9.09	3.46	7.06	5.66	6.81	7.02	8.66	7.15	5.91	3.67
5	Plastics	6.44	3.43	0.70	1.82	1.38	4.31	0.93	1.47	0.95	1.76	3.49	1.26	2.22	1.98	3.12	1.85	2.73	1.97	1.16	2.50	2.44	2.31	3.16	1.91
6	Metals	0.54	0.45	0.66	0.54	1.16	2.04	1.31	0.38	0.64	1.56	0.45	0.38	1.44	0.98	1.73	2.51	0.81	2.90	1.84	0.70	0.63	1.39	1.92	0.73
7	Organic waste	38.8	46.7	42.2	48.8	40.5	13.9	63.5	50.8	49.4	33.2	9.13	43.4	45.6	47.8	30.0	35.4	52.0	36.3	34.6	40.6	49.0	29.31	43.68	36.37
8	Electronics	0.00	0.18	0.00	0.05	0.16	0.68	0.52	0.00	0.00	0.37	0.20	0.12	0.04	0.00	1.37	0.88	0.16	0.00	0.00	0.07	0.00	0.00	0.74	0.00
9	Hazardous waste	0.52	1.38	1.76	2.34	0.37	0.71	0.79	0.85	0.54	1.00	0.42	0.32	0.44	0.69	0.66	0.56	1.38	0.11	0.79	0.70	0.91	1.09	1.61	0.59
10	Composit	0.76	0.93	0.55	0.71	0.46	0.15	0.85	0.62	0.35	0.57	0.78	0.38	0.55	0.63	0.36	0.61	0.62	2.88	0.43	0.29	0.59	1.11	0.58	0.31
11	Textiles	16.3	3.67	4.02	7.08	2.38	0.12	3.36	9.20	6.86	1.50	3.00	5.82	3.12	4.50	12.8	12.4	3.70	0.23	6.17	16.5	6.98	0.64	3.66	0.18
12	Diapers	10.6	1.04	9.83	11.0	7.31	0.08	6.32	3.99	12.8	4.79	2.43	4.20	2.74	1.15	0.00	6.01	3.32	0.00	14.3	6.54	5.62	3.07	0.83	0.06
13	Other combustible	8.22	7.39	14.5	12.7	8.60	30.6	7.55	16.3	6.48	19.4	13.3	8.28	14.4	11.0	18.1	13.1	10.9	11.2	8.17	9.59	8.95	17.14	10.76	10.46
14	Other non-combustible	0.59	0.54	2.59	0.78	0.00	8.54	0.71	0.00	2.19	3.75	0.00	2.95	0.81	0.39	0.00	0.36	0.69	0.00	11.1	0.35	0.80	4.55	0.22	0.33
	TOTAL	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100

Figure 2.5: Example of the tables shown in the article [Kaya et al. 2021], in particular Table 1

The calorific index indicates the degree of flammability and heat produced when a specific amount of a substance burns. The calorific index values of waste mixtures in the counties, arranged based on seasonality and socioeconomic region, are displayed in "Table 7" of the article [Kaya et al. 2021] which is also shown in Fig.(2.6).

Table 7
Calorific index values (GCV–LCV–ACV) of wastes in the counties according to socio-economic levels (Low–High–Comm.).

	WINTER									SUMMER								
	Low			High			Comm.			Low			High			Comm.		
	Calorific value (CV)		ACV	Calorific value (CV)		ACV	Calorific value (CV)		ACV	Calorific value (CV)		ACV	Calorific value (CV)		ACV	Calorific value (CV)		ACV
	GCV	LCV		GCV	LCV		GCV	LCV		GCV	LCV		GCV	LCV		GCV	LCV	
ARNAVUTKOY	4389	1743	3066	5062	1730	3396	4627	1479	3053	3604	1325	2464.5	3898	1258	2578	3717	1523	2620
AVCILAR	4840	1498	3169	4784	1692	3238	4028	2118	3073	4143	1167	2655	4293	1717	3005	3878	1294	2586
BAGCILAR	5199	1476	3292.5	4777	1383	3080	5249	1524	3382	3866	1325	2595.5	2420	987	1703.5	4406	1351	2878.5
BAHCELIEVLER	4344	1751	3047.5	4647	2517	3582	4951	1635	3293	3739	812	2275.5	3031	966	1998.5	3888	1137	2512.5
BAKIRKOY	5327	1815	3571	4915	1587	3251	4577	2419	3498	3606	1701	2698.5	2898	1508	2203	4306	1949	3127.5
BASAKSEHIR	6406	2743	4574.5	5878	2504	4191	5156	2773	3964.5	3800	1410	2605	4444	1733	3088.5	4533	2345	3439
BAYRAMPASA	4665	2203	3434	5057	2629	3553	5108	1368	3238	3601	1232	2416.5	3781	1314	2547.5	3982	1972	2977
BESIKTAS	5077	2693	3885	4645	1894	3269.5	4683	1497	3091	3000	1354	2177	3806	1698	2752	3507	1514	2510.5
BEYLIKDUZU	5052	2166	3609	5196	1710	3453	4428	1585	3006.5	3828	1463	2645.5	3597	1167	2382	2933	1586	2259.5
BEYOGLU	5132	2262	3697	5468	1794	3631	5379	1720	3545	3182	1528	2355	4488	2322	3405	4046	1791	2918.5
CATALCA	4495	2699	3572	4418	1551	2884.5	4776	1608	3192	2935	1456	2195.5	2925	1417	2671	3182	1751	2466.5
EYUP	5177	2019	3598	5106	2163	3634.5	4915	1962	3438.5	3742	1120	2421	4011	1764	2887.5	4267	1472	2869.5
ESENLER	3738	2260	2999	5273	1342	3307.5	4596	1108	2852	4141	1201	2671	4185	1733	2959	2737	693	1715
ESENYURT	4459	1718	3113.5	4614	1734	3174	4701	1838	3269	3628	1463	2645.5	3597	1167	2382	2933	1586	2259.5
FATIH	4284	1777	3030.5	4602	1439	3020.5	4622	1499	3060.5	4072	1076	2574	3944	1982	3013	3998	2078	3038.5
FATIHSMANPASA	4956	1077	3016.5	4602	1439	3020.5	4622	1499	3060.5	4072	1076	2574	3944	1982	3013	3998	2078	3038.5
GUNGOREN	4984	1353	3118.5	5049	1869	3459.5	4807	1955	3381	3744	1307	2525.5	2860	1572	2967.5	3640	1544	2592
KAGITHANE	5603	2138	3870.5	4698	1454	3076	4863	1981	3422	5104	1378	3237	4014	1894	2954	4209	1752	2953.5
KUCUKCEKMECE	5300	1145	3222.5	5057	2095	3576	4584	1364	2974	3471	1402	2436.5	3980	1483	2731.5	4025	1337	2681
SARIYER	5452	1366	3359	4948	1717	3332.5	4512	1398	2955	3794	1696	2745	4270	2192	3231	4069	2147	3108
SILIVRI	4438	1977	3207.5	4512	1122	2817	4018	1528	2773	2643	864	1753.5	1855	659	1257	2792	978	1885
SULTANGAZI	5436	2040	3738	5135	1285	3210	5118	2244	3681	3421	1396	2408.5	2595	904	1749.5	3174	1167	2170.5
SISLI	4685	1549	3117	4879	2163	3521	4845	1571	3208	3321	1075	2198	4227	2230	3228.5	4108	1973	3040.5
ZEYTINBURNU	4844	1782	3313	5016	1538	3277	5028	1660	3344	4331	1834	3082.5	3512	1320	2416	3931	1565	2748

Figure 2.6: Table 7 shown in the article [Kaya et al. 2021]

In this table, the gross calorific value (GCV) is calculated by reheating all combustion products to their initial pre-combustion temperature, especially any vapor that may have been created. Conversely, the Lower Calorific Value (LCV) is calculated by deducting the heat generated during the vaporization of water from the higher value. To get more accurate results, the authors have also utilized Average Calorific Value (ACV), which is determined by averaging GCV and LCV.

2.2.1 Model Evaluation

In machine learning, regression evaluation plays a critical role in understanding, diagnosing, and improving the performance of models that predict continuous or quantitative outcomes. The effectiveness of a regression model is largely determined through its evaluation, which helps stakeholders understand the model's accuracy, reliability, and utility in making decisions.

To evaluate and compare the results between the models it will be used Mean Squared Error (MSE):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i) \tag{2.1}$$

Another method that can be used is the Root Mean Squared Error (RMSE) :

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)} \tag{2.2}$$

with N is the total number of observations (data points), y_i is the actual value of an observation, and $\hat{y}_i$ is the predicted value of the observation.

Regression evaluation in machine learning serves several key roles; among them:
Assessing model accuracy: It quantifies how closely the model's predictions match actual observed outcomes using metrics like MSE, RMSE.
Model selection: Evaluation metrics help in comparing different models or configurations to choose the best performing one.
Diagnosing model behavior: It identifies problems like under-fitting and over-fitting, guiding necessary adjustments to improve the model.
Improving model performance: Systematic evaluation supports the iterative tuning of model parameters and feature selection to enhance accuracy.
Building trust and confidence: Reliable evaluation results ensure stakeholders can trust the model's predictions, crucial for decision-making processes.

Chapter 3

The Implementation

Follows a discussion on my personal implementation [Cartolano 2024] and results given the problem illustrated in the previous chapters.

3.1 Define the Machine Learning Problem

Given the context [Kaya et al. 2021], the general Machine Learning problem will be the prediction of the calorific value of waste based on its socioeconomic factors, season, county and waste amount.
This is done by following two steps:

1. Predicting Daily Solid Waste Amount.
2. Finding Calorific Value of Composite MSW on the basis of the predicted Waste Amount.

3.2 The Dataset

The structure of the datasets has been simplified.

3.2.1 Consolidate the Dataset

Since the data was spread in different tables Table 1-2-3-4-5-6 (in article [Kaya et al. 2021]) for Amount of Waste and Table 7 (in article [Kaya et al. 2021]) for Calorific Value of waste, it has been merged into a single, coherent dataset each.

To do that it has been considered as common identifiers for Waste Components:

- "Paper, Glass, PET Bottles, Plastic Bags, Plastics, Metals, Electronics, Hazardous Waste, Compost, Textiles, Diapers, Other Combustible, Other Non-Combustible"

their correspondent Socioeconomic Level categorized in

- "Low", "High", "Comm" (Commercial Area).

the season the wastes have been produced

- "Winter" and "Summer"

and the county of Istanbul considered using the following code:

- **A1**: ARNAVUTKOY, **A2**: AVCILAR, **B1**: BAGCILAR, **B2**: BAHCELIEVLER, **B3**: BAKIRKOY, **B4**: BASAKSEHIR, **B5**: BAYRAMPASA, **B6**: BESIKTAS, **B7**: BEYLIKDUZU, **B8**: BEYOGLU, **C1**: CATALCA, **E1**: ESENLER, **E2**: ESENYURT, **E3**: EYUP, **F1**: FATIH, **G1**: GAZIOSMANPASA, **G2**: GUNGOREN, **K1**: KAGITHANE, **K2**: KUCUKCEKMECE, **S1**: SARIYER, **S2**: SILIVRI, **S3**: SULTANGAZI, **S4**: SISLI, **Z1**: ZEYTINBURNU.

It has been done the same thing for Calorific Values considering the same identifiers but "Waste Components" excluded.

3.2.2 Load the Data

After importing libraries, it has been loaded the dataset corresponding to the Amount of Waste "`my_dataset_WA.csv`" and the one for the dataset corresponding to the Average Calorific Value "`my_dataset_CV.csv`".

3.2.3 Data Exploration

One has printed some information of the datasets considered to understand their characteristics, distributions, missing values and any correlation between the features.

Dataset for Amount of Waste

In the preliminary analysis it is possible to observe the columns names of the dataset:
`Index(['Waste Components', 'County', 'Socio-economic Lvl', 'Waste Amount', 'Season'], dtype='object')`.
In Fig.(3.1) it is possible to see ten random rows of the Waste Amount dataset.

Waste Amount Dataset

	Waste Components	County	Socio-economic Lvl	Waste Amount	Season
1198	Electronics	B6	High	0.10	Summer
526	Electronics	G1	High	0.28	Winter
393	PET bottles	E1	Low	0.28	Winter
1407	PET bottles	E3	Low	2.41	Summer
433	Plastics	B7	High	2.15	Winter
1720	Glass	S3	High	1.95	Summer
1090	Plastic Bag	B2	High	4.65	Summer
429	Plastic Bag	G1	Low	7.83	Winter
1800	Metals	G2	Low	0.56	Summer
530	Hazardous Waste	B7	Comm	0.93	Winter

Figure 3.1: Display few random rows of the data frame for Amount of Waste

Follow some basic descriptive statistics of the Waste Amount dataset:

- `count = 2016.000000`
- `mean = 7.044564`
- `std = 12.388274`
- `min = 0.000000`
- `25% = 0.640000`
- `50% = 2.250000`

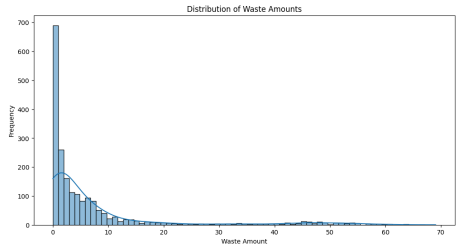
- `75% = 6.865000`
- `max = 69.070000`

In particular, `mean` is the mean value, `std` is the standard deviation and `50%` is the median or 50th percentile which is the middle value, dividing the dataset into two halves.

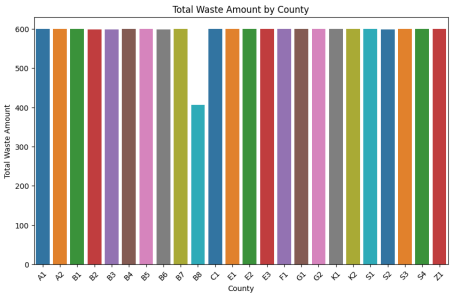
The mean gives an overall idea of the value of the dataset but can be skewed by outliers. Standard deviation is a measure of the amount of dispersion of the dataset. Low standard deviation indicates that the values tend to be close to the mean of the set, high values on the other hand indicate that the values are spread out over a wider range. Last, the median is a measure of a central tendency.

The case of the Amount of Waste dataset, one needs to consider of course only the numerical column of `Waste Amount`. The mean the average waste amounts in kilos is 7.05. The standard deviation is relatively high compared to the mean so, this indicates that the waste amounts vary widely from the average. The median is less then the mean which suggests that the distribution of the waste amount is skewed to the right (positively), meaning that more observations are clustered at the lower end of the scale.

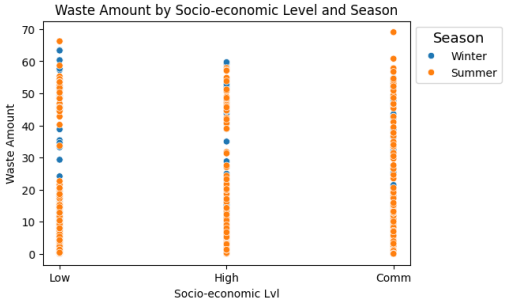
After this, one has checked if there are any missing values; there are none.



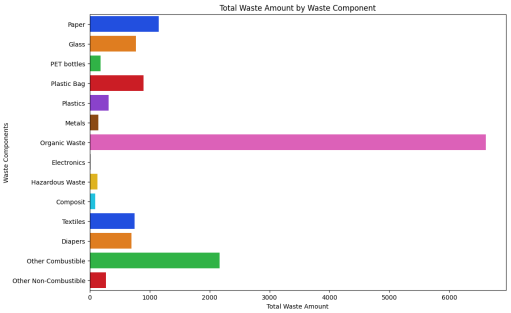
(a) Histogram for target variable: Waste Amount



(b) Bar Plot of Waste Amount by County



(c) Scatter Plot of Waste Amount by Socio-economic Level and Season



(d) Bar Plot for Waste Amount by Waste Components

Figure 3.2: Visualisation of the dataset Amount of Waste

In Fig.(3.2a) it is represented a histogram with a fitted line that represents a probability distribution of Waste Amount. Each bar's height corresponds to the frequency of waste amounts falling within the range of the bar. The first bar, for instance, is the tallest, indicating that the majority of waste amount measurements fall within that first bin (which is from 0 to around 2.5, if we assume equal bin width).

Overlaid on the histogram, there is a smooth curve that represents a probability density function (PDF) fitted to the data. This curve shows the shape of the assumed distribution of the data. The distribution seems to suggest that while most of the data is clustered around a lower waste amount, there are a few instances with much larger waste amounts. Overall, most of the waste amounts are low, with fewer high waste amounts. The distribution is not symmetric and is skewed to the right, indicating that higher waste amounts are less frequent but do occur.

In Fig.(3.2b) it is possible to see a Bar-Plot that puts in relationship Waste Amount with County. Notice that the County with code `B8` produces less waste than the other counties.

In Fig.(3.2c), the plot provided appears to be a type of scatter plot that is displaying Waste Amount as it relates to Socio-economic Level and differentiates between two seasons: Winter and Summer. There is a distinction in waste amounts between the socio-economic levels, with the Low and High levels having a wide range of waste amounts while the Commercial category has a more concentrated range, suggesting a more consistent waste output. The seasons do not show a clear pattern of difference in waste production within each socio-economic level, as both Winter and Summer seasons are interspersed throughout. There does not appear to be a seasonal effect that is visually discernible from this plot.

In the last one, Fig.(3.2d), it is represented the distribution of Waste Amount by Waste Components. It catches the eye that the majority of the waste is produced by organic waste and the minimum is produced by electronics.

Dataset for Average Calorific Value of Wastes

It has been done the same data exploration for this second dataset.
In the preliminary analysis it is possible to observe the columns names of the dataset: `Index(['County', 'Socio-economic Lvl', 'Avg Calorific Value', 'Season'], dtype='object')`.
In Fig.(3.3) it is possible to see ten random rows of the Calorific Value dataset.

Avarege Calorific Value of the Wastes Dataset

	County	Socio-economic Lvl	Avg Calorific Value	Season
117	G1	Low	3024.0	Summer
19	B5	High	3233.0	Winter
82	B2	High	1998.5	Summer
97	B7	High	2382.0	Summer
56	K2	Comm	2974.0	Winter
12	B3	Low	3571.0	Winter
132	S2	Low	1753.5	Summer
65	S3	Comm	3681.0	Winter
66	S4	Low	3117.0	Winter
18	B5	Low	3434.0	Winter

Figure 3.3: Display few random rows of the data frame for Calorific Value

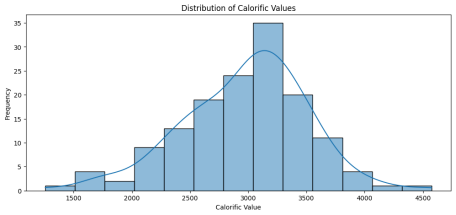
Follow some basic descriptive statistics of the Calorific Values dataset:

- `count = 144.000000`
- `mean = 2974.284722`
- `std = 532.415011`
- `min = 1257.000000`

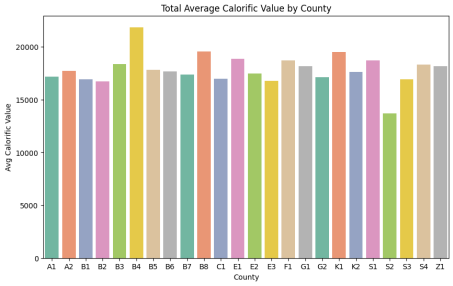
- 25% = 2639.125000
- 50% = 3044.000000
- 75% = 3308.875000
- max = 4574.500000

In this case, considering of course only the numerical column of "Avg Calorific Value, the mean is 2974.284722. The standard deviation is low compared to the mean so, this indicates that the data points are clustered closely to the mean. The median as almost the same value as the mean which suggests that the distribution is basically symmetrically distributed.

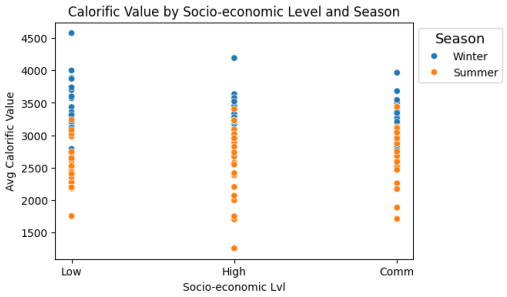
In the code [Cartolano 2024], it is shown that there are no missing values.
It has been also considered the visualization of the dataset distribution of Average Calorific Value of the Wastes with respect to different features through different plots.



(a) Histogram for target variable: Average Calorific Value



(b) Bar Plot of Calorific Value by County



(c) Scatter Plot of Calorific Value by Socio-economic Level and Season

Figure 3.4: Visualisation of the dataset Calorific Values

In Fig.(3.4a) it is shown a histogram with a curve overlaid, illustrating the distribution of calorific values.
The bars show the number of observations that fall within each bin range. The highest bar appears in the 3000-3500 range, indicating this range of calorific values has the highest frequency in the dataset.
The line graph overlaying the histogram seems to be a best-fit curve, which suggests how the data points are distributed across the calorific value spectrum.
The histogram shows that the distribution of calorific values is somewhat symmetrical around the 3000 mark, which could suggest a normal distribution of values

around a mean of approximately 3000 (peak of the curve) and so fewer instances of lower calorific values.

In Fig.(3.4b) it is possible to see an bar-plot that puts in relationship Average Calorific Value with County. Notice that the County with code `B4` produced more calories than the other counties.

In Fig.(3.4c) is a scatter plot that presents 'Avg Calorific Value' in relation to 'Socio-economic Level', with data points color-coded for two seasons, Winter and Summer. Each dot on the plot represents an individual measurement of calorific value within the dataset. The blue dots indicate measurements taken in Winter, while the orange dots represent measurements from Summer. At all socio-economic levels, there's a broad range of calorific values, but the Commercial category seems to have less variation in average calorific values. Some of the highest calorific values are found in the 'Low' and 'High' socio-economic levels during Winter, indicated by the few blue dots towards the top of these categories. The 'Comm' category has the least variation in calorific values, and there are few outlier points. This might suggest that there is a more consistent calorific value in commercial areas.

3.3 First Stage: Predicting Daily Solid Waste Amount

3.3.1 Select Features and Target Variables

From the dataset `my_dataset_WA.csv` one can identify which columns can serve as features (independent variables) and which can be the target (dependent variables).

- **Features**: "Socio-economic Level", "Waste Components", "Season" and "County".
- **Target Variable**: For this predictive model, the target value is "Waste Amount".

3.3.2 Data Preprocessing

Data Preprocessing is fundamental to prepare the data for modeling. It includes encoding categorical variables, normalize/standardize numerical values and splitting the dataset into training and testing.

First one wants to prepare the data by dividing it in features (`"X"`) and target variables (`"y"`).

In Fig.(3.5) it is shown the structure of the data by displaying ten random rows for features and target variables.

Then, before feeding the data into the models considered one should ensure that the categorical variables are properly encoded and the numerical data is normalised/scaled.

It has been defined the categorical transformer by using a pipeline that ensures that some steps are executed in the right order. The first step is `'imputer'` which uses the transformer `SimpleImputer(strategy = 'most_frequent')` which replaces missing values with the most frequent one in each column. The second step is `'onehot'` encoder which converts categorical variables into a form that could be provided to ML algorithms to do a better job in prediction. It creates a binary column for each category and returns a matrix with the results. The transformer

```
Features
     Waste Components County Socio-economic Lvl  Season
1198      Electronics     B6               High  Summer
526       Electronics     G1               High  Winter
393       PET bottles     E1                Low  Winter
1407      PET bottles     E3                Low  Summer
433          Plastics     B7               High  Winter
1720            Glass     S3               High  Summer
1090      Plastic Bag     B2               High  Summer
429       Plastic Bag     G1                Low  Winter
1800           Metals     G2                Low  Summer
530   Hazardous Waste     B7               Comm  Winter
Target Variables
1198    0.10
526     0.28
393     0.28
1407    2.41
433     2.15
1720    1.95
1090    4.65
429     7.83
1800    0.56
530     0.93
Name: Waste Amount, dtype: float64
```

Figure 3.5: Display of features and target values

`'OneHotEncoder(handle_unknown = 'ignore')'` tells the encoder to ignore any unseen labels that might appear in future data. This is useful for maintaining model robustness when it encounters new, unseen categories in the data.

It has been defined also the numerical transformer by using another a pipeline. The first step is `'imputer'` which uses the transformer `SimpleImputer(strategy = 'mean')` which deals with missing values with the mean value of each column. The second step is `'scaler'` which uses the transformer `'StandardScaler()'` that standardizes the features by removing the mean and scaling to unit variance. This standardization is important because it ensures that each feature contributes equally to the model.

Complete the preprocessing integration through a `preprocessor` that contains the values of the categorical columns after encoding and numerical columns after the standardization.

3.3.3 Split the Data

The dataset `"my_dataset_WA-csv"`, after the dropping of the target features `"Waste Amount"` has been split into train (`(X_train, y_train`) and test (`(X_test, y_test`) by using the 70:30 rule. This has been done to prepare the data for model training.

In Fig.(3.6) it is shown for features and target value the size and the first sample for train and test sets.

3.3.4 Model Implementation

In this section it will be discussed how the machine learning models chosen has been trained for predicting solid amount of waste.

Multilayer Perceptron Regressor (MLP)

After defining the MLP model through `MLPRegressor()` it has been done some hyperparameter tuning to look for the optimal parameters to get the best training model for the implementation.

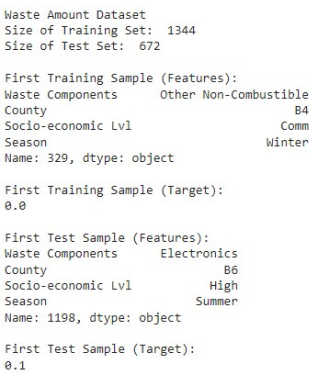


Figure 3.6: Display the size and the first sample of training and test for features and target values

After having defined the hyperparameters as it has been done in Ch.2.1.1, one has used `"RandomizedSearchCV()"` to define the model and then train it.

`"RandomizedSearchCV()"` uses cross validation to asses how well the results will generalize to an independent dataset made of unseen values and to mitigate problems like over-fitting or under-fitting

The best parameters for MLP are

- `'mlp__solver': 'sgd'`;
- `'mlp__momentum': 0.1`;
- `'mlp__learning_rate_init': 0.05`;
- `'mlp__learning_rate': 'constant'`;
- `'mlp__hidden_layer_sizes': (100,)` and `'mlp__activation': 'tanh'`.

The "Best Score" attribute specifically holds the highest mean cross-validated score achieved by any of the parameter combinations tested during search. It allows to understand how well the best found model is expected to perform on unseen data.

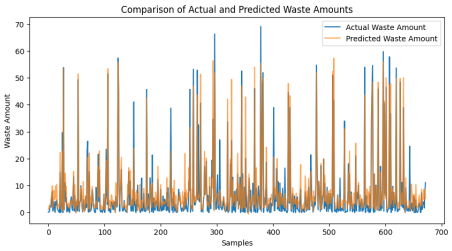
MLP Best score is `0.847` which suggests that the best MLP model configuration found performs very well and accurately predicts 84,7% of the times on average across cross-validation folds.

This model has training time of `108.78` seconds.

Once the model has been trained, it has been used the model to predict the values of `Waste Amount` and then it has been evaluated it using MSE which gets the value `18.26` and RMSE has `4.27`.

Fig.(3.7a) is a line chart comparing two sets of data: the 'Actual Waste Amount' and the 'Predicted Waste Amount' across numerous samples. Blue Line (Actual Waste Amount) represents the actual waste amounts recorded for each sample. This is the observed data. Orange Line (Predicted Waste Amount) shows the waste amounts predicted by a model or method for each sample.

The lines for actual and predicted waste amounts seem to follow a somewhat similar pattern, indicating that the prediction model has some degree of accuracy. There are noticeable deviations between the actual and predicted values in some samples, suggesting that the model might not be perfectly accurate in every instance.



(a) Comparison of Actual and Predicted Waste Amounts

(b) Actual vs. Predicted Waste Amounts

Figure 3.7: Visualisation of the predicted data with respect to the actual ones

In Fig.(3.7b) is a scatter plot with a line of best fit, comparing the Actual vs. Predicted Waste Amounts.The blue dots represents an individual prediction by the model versus the actual value. For perfect predictions, dots would lie exactly on the line where the actual value equals the predicted value. The red line is a regression line that indicates the general trend of the predictions relative to the actual values. If predictions were perfect, all blue dots would be on this line.
The general trend suggested by the line of best fit is that the predictions are positively correlated with the actual waste amounts, which means that as the actual waste amount increases, so does the predicted waste amount.
There are some points clustered towards the lower end of the actual waste amounts, indicating a lot of data with lower values and possibly some level of over-prediction for these lower actual values since many of them are above the line of best fit.
The overall fit seems reasonable but not perfect.

Support Vector Machine Regressor (SVR)

The other model considered is the SVM model through `SVR()` it has been done some hyperparameter tuning.

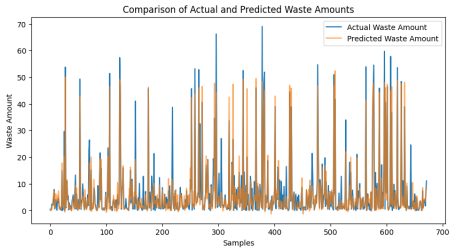
One defines the parameters for the tuning as described in Ch.2.1.2 and the use `"RandomizedSearchCV()"` to define the model and then train it as one have done before.

The best parameters for SVR are

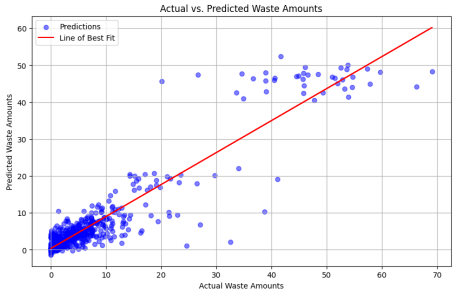
- `'svr__kernel': 'poly'`;
- `'svr__gamma': 10`;
- `'svr__degree': 2`;
- `'svr__C': 0.01`.

MLP Best score is `0.859` which suggests that the best SVR model configuration found, performs with a prediction accuracy of 85,9% of the times on average across cross-validation folds.
This model has training time of `11.27` seconds.
The prediction of `Waste Amount` has been evaluated by using MSE which gets the value `16.96` and RMSE has `4.11`.

It is possible to draw the same conclusions from Fig.(3.8) as from the MLP case.



(a) Comparison of Actual and Predicted Waste Amounts

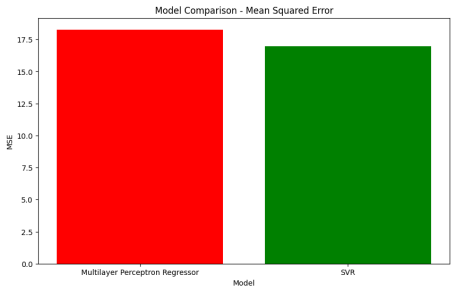


(b) Actual vs. Predicted Waste Amounts

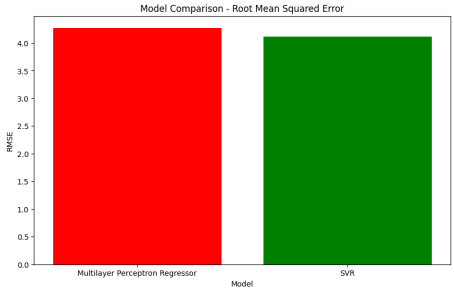
Figure 3.8: Visualisation of the predicted data with respect to the actual ones

3.3.5 Best Model

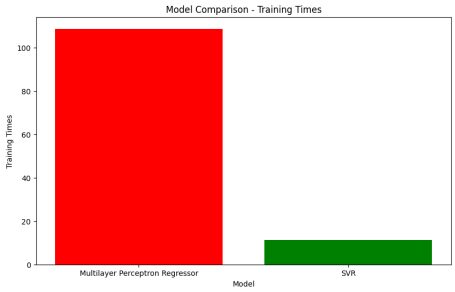
In Fig.(3.9a) it is possible to see in green the MSE; RMSE and training time of SVR and in red the MSE; RMSE and training time of MLP for Waste Amount prediction. It is obvious that even if MLP is slower it performs definitely better than SVR. Using the MLP model one puts the predictions of `"Waste Amount"` in terms of `"County"`, `"Socio-economic Level"` and `Season` in a table on the CVS file `"my_dataset_pred_WA.csv"`.



(a) MSE evaluation between MLP and SVR



(b) RMSE evaluation between MLP and SVR



(c) Training time between MLP and SVR

Figure 3.9: Visualisation of the evaluation indices

3.4 Second Stage: Calculating Average Calorific Value (CV) of the Composite MSW

For the second stage one will need the average calorific values (CV) by county and socio-economic level for each season (`"my_dataset_CV.csv"`). This stage calculates the CV of the composite MSW based on the predicted waste amounts and their CVs.

First thing to do is to load the dataset for predicted Amount of Waste by the CVS file `"my_dataset_pred_WA.csv"`.

Then it is good practice to merge the two datasets for predicted waste amount and average calorific values on `"County"`, `"Socio-economic Level"`, `"Season"` like shown in Fig.(3.10) where ten random rows of the table are displayed.

Average Calorific Values + Predicted Waste Amount Dataset

	County	Socio-economic Lvl	Avg Calorific Value	Season	Predicted WA Values
361	A2	Comm	2586.0	Summer	0.624987
158	C1	Comm	3192.0	Winter	0.885565
481	B8	Comm	2918.5	Summer	-0.481435
639	S3	Low	2403.5	Summer	0.930791
275	S1	Low	3359.0	Winter	4.409273
362	A2	Comm	2586.0	Summer	-0.158862
310	S3	Comm	3681.0	Winter	0.849037
199	E3	Comm	3204.0	Winter	8.878899
544	E3	Low	2645.5	Summer	44.247502
90	B5	Comm	3238.0	Winter	0.300386

Figure 3.10: Display few random rows of the merged data frame for predicted Amount of Waste and Calorific Values

3.4.1 Transfer Learning

For this section it has been used transfer learning by using two techniques: best model from Stage 1 and Neural Networks by using Tensor Flow.

Prepare the Data

From the merged dataset, one has separated target value (`Avg Calorific Value`) from features (`County`, `Socio-economic Lvl`, `Season` and `Predicted WA Values`). In Fig.(3.11a) one can see ten random rows for features and target values.

Follows preprocessing of categorical and numerical data just as one has done in Ch.(3.3.2) and split between training samples and test samples just as in Ch.(3.3.3). In Fig.(3.11b) it is possible to see information about test and split.

Best Model from Stage 1

By using the best model from Stage 1, one will predict the Calorific Values and evaluate through MSE (214032.3) and RMSE (462.6).

In Fig.(3.12a) shows two lines representing the actual and predicted values across a number of samples. The actual calorific values are shown in blue, and the predicted values are in orange. Both lines follow a similar pattern, indicating that the prediction model is following the trend of the actual values, although the predicted values deviate from the actual ones.

```
Features
     Waste Components County Socio-economic Lvl  Season
1198      Electronics     B6              High  Summer
526       Electronics     G1              High  Winter
393       PET bottles     E1               Low  Winter
1407      PET bottles     E3               Low  Summer
433          Plastics     B7              High  Winter
1720            Glass     S3              High  Summer
1090      Plastic Bag     B2              High  Summer
429       Plastic Bag     G1               Low  Winter
1800           Metals     G2               Low  Summer
530   Hazardous Waste     B7              Comm  Winter
Target Variables
1198    0.10
526     0.28
393     0.28
1407    2.41
433     2.15
1720    1.95
1090    4.65
429     7.83
1800    0.56
530     0.93
Name: Waste Amount, dtype: float64
```

(a) Display of features and target variables

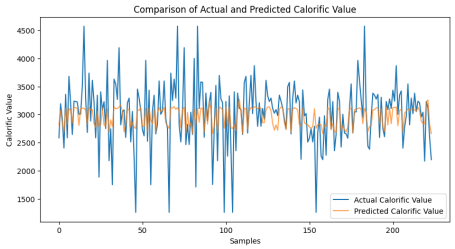
```
Average Calorific Values + Predicted Waste Amount Dataset
Size of Training Set:  448
Size of Test Set:  224

First Training Sample (Features):
County                       K1
Socio-economic Lvl         Comm
Season                   Winter
Predicted WA Values    0.123086
Name: 259, dtype: object

First Training Sample (Target):
3422.0

First Test Sample (Features):
County                       A2
Socio-economic Lvl         Comm
Season                   Summer
Predicted WA Values    0.624987
Name: 361, dtype: object

First Test Sample (Target):
2586.0
```

(b) Display of training samples and test samples

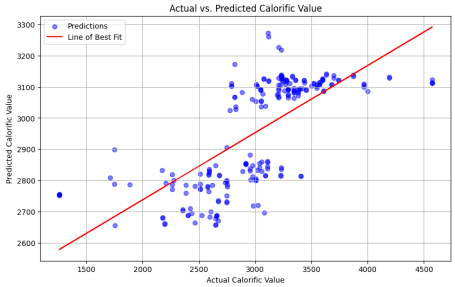
Figure 3.11: Visualisation of the data frame

In Fig. (3.12b) is a scatter plot with a line of best fit through the data points. The distribution of points suggests a correlation between the actual and predicted values, with the line of best fit indicating the average trend of the predictions. However, there is some spread around the line, which suggests that while the model has predictive power, there are variations in its accuracy.

In general, it is obvious that this prediction works very bad, with score `0.394`, which means 39,4 % of right answers.



(a) Comparison of Actual and Predicted Calorific Values



(b) Actual vs. Predicted Calorific Values

Figure 3.12: Visualisation of the predicted data with respect to the actual ones

Tensor Flow

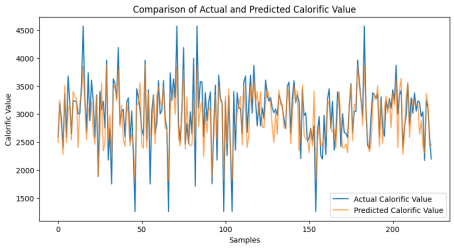
First thing done was to pre-process the data so that it can fit the requirements of the Neural Network model called by Tensor Flow.
It has been added some parameter tuning to get the best values of them which mitigate over-fitting. These are:

- `units`: 256
- `dropout`: 0.2
- `learning_rate`: 0.00712

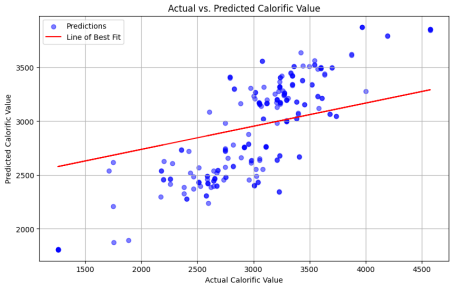
and training time: 5.335 seconds.
The model prediction has been evaluated on MSE = 107752.3 and RMSE = 328.2.

In Fig.(3.13a) the lines follow a similar pattern throughout the graph, which indicates the predicted values are generally in alignment with the actual values.

In Fig.(3.13b) the scatter of points around this line suggesting the average accuracy of the model's predictions relative to the actual values. Deviations from the line indicate prediction errors.



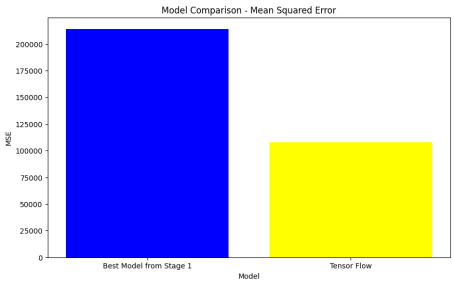
(a) Comparison of Actual and Predicted Calorific Values



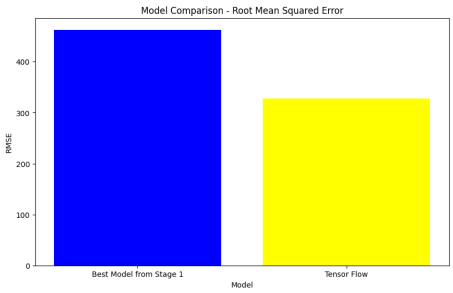
(b) Actual vs. Predicted Calorific Values

Figure 3.13: Visualisation of the predicted data with respect to the actual ones

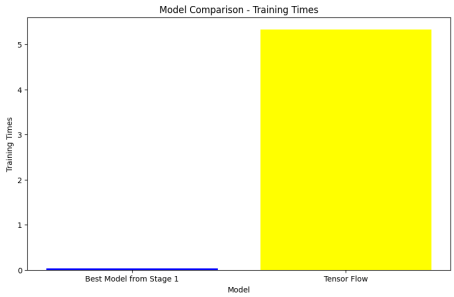
3.4.2 Best Model for Transfer Learning



(a) MSE evaluation between Best Model form Stage 1 and Tensor Flow



(b) RMSE evaluation between Best Model form Stage 1 and Tensor Flow



(c) Training time between Best Model form Stage 1 and Tensor Flow

Figure 3.14: Visualisation of the evaluation indices

It is clear from Fig.(3.14a) and Fig.(3.14b) that the model that best predicts the Calorific Values is NN by using Tensor Flow even it it makes more time.

3.4.3 "By Hand" Method

For sake of completeness, one has calculated the Total Calorific Value from the predicted waste amount values that one have got from Stage 1.
This is done by using the following formula:
`data_merged['TotalCalorificValue'] = data_merged['Avg Calorific Value'] * data_merged['Predicted WA Values']`
and the predictions have been saved in a CVS file which has the structure shown in Fig.(3.15).

County	Socio-economic Lvl	Season	TotalCalorificValue
A1	Comm	Summer	88144.66106
A1	Comm	Winter	52046.35287
A1	High	Summer	28374.24084
A1	High	Winter	90952.67949
A1	Low	Summer	2616.513817
A1	Low	Winter	42010.90314
A2	Comm	Summer	73439.17402
A2	Comm	Winter	17028.29128
A2	High	Summer	25124.25335
A2	High	Winter	234376.1396
A2	Low	Summer	82429.51017
A2	Low	Winter	16521.60671
B1	Comm	Summer	26683.2744
B1	Comm	Winter	26889.48504
B1	High	Summer	109922.7891
B1	High	Winter	25976.92087
B1	Low	Summer	49700.23014
B1	Low	Winter	59901.88608
B2	Comm	Summer	159966.9094

Figure 3.15: Display the first twenty rows of the data frame containing Total Calorific Values

Fig.(3.16) is used to compare the total calorific values across different counties, differentiated by season and socio-economic levels.
Considering county grouping, there are six bars, each corresponding to a different combination of season (Summer or Winter) and socio-economic level (Low, High, or Comm for Commercial).
Legend (Season and Socio-Economic Level):

- Blue for Summer, Commercial
- Green for Summer, High
- Orange for Summer, Low
- Purple for Winter, Commercial
- Red for Winter, High
- Brown for Winter, Low

The variation in bar height within a group indicates how the total calorific value changes with the socio-economic level and season within a particular county.
Notice that county `C1` produces the highest amount of wastes in winter by people with social-economic level Low; on the other hand, in county `B8` commercial activities in summer produce so little wastes that one gets negative Total Calorific Value.

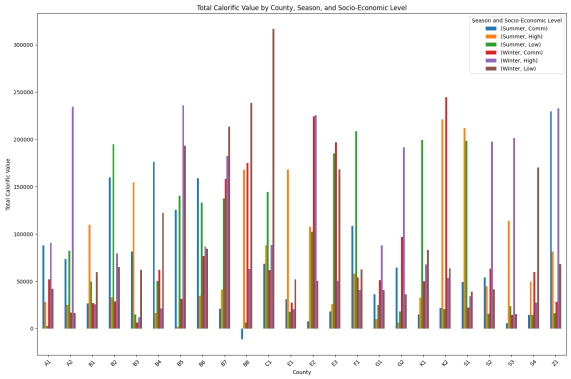


Figure 3.16: Total Calorific Value by County, Season, and Socio-Economic Level

Chapter 4

The Role of WTE in the Energy Network

By considering the scenario described in the article [A. Joshi and Glielmo 2023], it will be given an explanation of the role of WTE in the Energy Network.

In a decentralized energy grid, as shown in Fig.(4.1), individual households or agents may share a common energy storage system (such as batteries). This allows for more effective utilization of storage capacity, reducing costs compared to scenarios where each household maintains its own separate storage system.

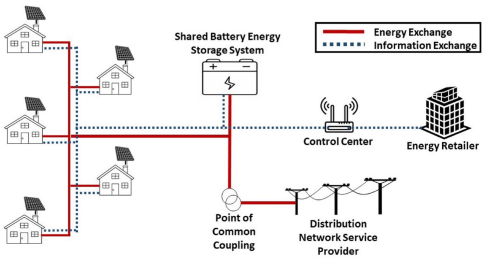


Figure 4.1: Residential households in a neighbourhood

In this approach [A. Joshi and Glielmo 2023], each agent (or household) operates as a learning entity in a multi-agent reinforcement learning environment. The agents use Q-learning (Appendix (C)), a type of reinforcement learning algorithm, to independently determine their optimal actions based on their own observations and the shared state of the system.

The objective in using Q-learning is to optimize the charging and discharging schedules of the shared energy storage. By learning the best strategies for storing or releasing energy, the system can maximize the use of renewable energy, minimize costs, and maintain the stability of the grid.

While each agent operates independently, the consensus aspect of the Q-learning approach ensures that all agents' strategies are aligned towards a common goal (consensus Appendix (D.1)). This alignment is crucial to prevent conflicts and ensure that the shared storage is used efficiently without overcharging or depleting it unnecessarily.

Integrating WTE into this framework helps to balance the intermittent nature of renewable energy sources like wind and solar ones which depend on the weather

conditions. WTE provides a more consistent and controllable energy source that complements the variability of other renewable energies.

By optimizing the use of shared storage and incorporating WTE, the energy system can reduce overall energy costs for participants and increase the efficiency of the grid.

4.1 The Power Grid Model

Let us have a look at the grid. To make things easier with respect to the article [Binder 2020], consider a typical energy generation setup combined with a Waste-To-Energy (WTE) plant which introduces complexity but also potential benefits to grid management. The combined system can optimize the use of variable waste-derived energy alongside stable conventional generation, like a coal, gas, or nuclear power plant.

Follows an easy discussion on how this set up can be modeled and analyzed, especially focusing on power flow equations, swing equations and system stability.

The Power Flow Equations

Power flow analysis is crucial to determine the voltage at various buses and the power flow through transmission lines in a network.

In a grid with both a conventional generator and a WTE plant, each power source is modeled as a generator bus in power flow analysis because one needs to account for multiple generation sources.
The basic power flow equations for a bus i in a typical AC network:

Active Power (P) Equation:

$$P_i = V_i \sum_{j=1}^{n} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \tag{4.1}$$

Reactive Power (Q) Equation:

$$Q_i = V_i \sum_{j=1}^{n} V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \tag{4.2}$$

where,

- V_i and V_j are the voltage magnitudes at buses i and j, respectively.
- $\theta_{ij} = \theta_i - \theta_j$ is the phase angle difference between the buses.
- G_{ij} and B_{ij} are the conductance and susceptance of the line connecting buses i and j, respectively.

For a WTE plant integrated into the grid, it would typically be modeled as a generation bus, where the power output is controllable and voltage is maintained within certain limits.
At bus k (conventional generator) and bus m (WTE plant), one can write Active Power as

$$P_m = P_{\text{WTE}} + V_m \sum_{j=1}^{n} V_j (G_{mj} \cos\theta_{mj} + B_{mj} \sin\theta_{mj}) \tag{4.3}$$

and Reactive Power as

$$Q_m = Q_{\text{WTE}} + V_m \sum_{j=1}^{n} V_j (G_{mj} \sin\theta_{mj} - B_{mj} \cos\theta_{mj}) \tag{4.4}$$

where

- P_{WTE} represent the power generated by the WTE facility and it is a function of the waste input rate, conversion efficiency, and operational status of the WTE plant as described in article [Kaya et al. 2021].
- The value of Q_{WTE} depends on the design of the WTE facility.

Swing Equations

The swing or power angle equations are used to model the dynamics of generators connected to the grid and their response to disturbances.

If the WTE plant is sizable and its generation is variable (depending on waste supply and processing rate), it could affect grid stability.

The stability of each generator connected to the grid is critical, especially with two types of generation units that have different dynamic characteristics.

Swing Equation for Conventional Generator at bus k:

$$\frac{d^2\delta_k}{dt^2} = \frac{\omega_{\text{base}}}{2H_k} \left(P_{m_k} - P_{e_k} - D_k \frac{d\delta_k}{dt} \right) \tag{4.5}$$

where,

- $\frac{d^2\delta_k}{dt^2}$: angular acceleration of the generator's rotor, indicating how the rotor's angle changes with respect to time.
- $2H_k$: twice the inertia constant of the generator, indicating its resistance to changes in rotational speed.
- ω_{base}: the synchronous speed of the grid.
- P_{m_k}: mechanical power input to the generator, typically from a prime mover.
- P_{e_k}: electrical power output, which is the power delivered to the electrical grid.
- D_k: damping coefficient, which accounts for losses such as mechanical friction and electrical resistance.
- $\frac{d\delta_k}{dt}$: angular velocity of the generator's rotor, indicating the speed at which the rotor is rotating.

Swing Equation for WTE Generator at bus m:

$$\frac{d^2\delta_m}{dt^2} = \frac{\omega_{\text{base}}}{2H_m} \left(P_{m_m} - P_{e_m} - D_m \frac{d\delta_m}{dt} \right) \tag{4.6}$$

where,

- $\frac{d^2\delta_m}{dt^2}$: angular acceleration of the WTE generator's rotor at bus m.
- $2H_m$: twice the inertia constant of the WTE generator.
- P_{m_m}: mechanical power input specific to the WTE process.
- P_{e_m}: electrical power output from the WTE generator.
- D_m: damping coefficient for the WTE generator.
- $\frac{d\delta_m}{dt}$: angular velocity of the WTE generator's rotor.

The stability of the power grid is affected by the introduction of a WTE plant in several ways.

Steady-State stability refers to the power system's ability to maintain equilibrium under normal conditions. It deals with the grid's reaction to minor changes in load or generation. The grid should be analyzed by solving power flow equations to ensure voltage and frequency remain within the desired operation range of the WTE plant.

Dynamic Stability assesses the power system's ability to withstand and damp out oscillations following small disturbances like rapid changes in energy production from a WTE plant. The analysis involves simulating the system's response to oscillatory disturbances which are needed to verify that the WTE plant does not negatively affect the grid's dynamic behavior and maintain the system's oscillations within acceptable limits.

In particular, Transient Stability Analysis is crucial in assessing the grid's ability to maintain stability when subjected to severe transient disturbances, such as faults, sudden large load changes, or loss of generation. The analysis can determine how quickly and effectively the system can return to steady state. This is especially pertinent given the variable nature of WTE generation.

Voltage Stability studies focus on the system's ability to maintain acceptable voltage levels at all buses under normal conditions and after being subjected to disturbances. This is interesting since WTE plants may operate with different voltage characteristics compared to conventional power plants.

Frequency Analysis crucial for ensuring the reliability and efficiency of power delivery. With WTE plants, there might be different types of power electronics used for energy conversion processes that can introduce harmonics into the grid.

Cyber-Physical Security aspects of integrating WTE plants into the grid could be crucial because they are increasingly reliant on digital communication and control technologies. This includes securing the communication channels and ensuring robust control against potential cyber threats.

4.2 Methodology for Q-Learning Consensus

In the proposed approach [A. Joshi and Glielmo 2023], the model includes agents (households), each with control over charging or discharging a shared energy storage system (SBESS). The interaction is modeled as a Markov Game (Appendix (C.1)) where each agent's action impacts the state of the shared system.

Shared Energy Storage System (SBESS)

Households collectively own a SBESS, which acts as the only controllable load. It is modeled using an energy reservoir model, where the control action is the net power to be injected or withdrawn from the SBESS, and its state is represented by the available energy stored. Additionally, the energy management includes waste-to-energy (WTE) processes which convert household waste into energy.

- Green energy surplus for each household n at time t is given by $r^n_{\uparrow,t} = \max(r^n_t - d^n_t, 0) \in \mathbb{R}^+$.
- Uncontrollable charging of the SBESS using the green surplus and energy from WTE is $u^n_{r,t}$, constrained by $u^n_{r,t} \leq r^n_{\uparrow,t}$.
- Controllable action $u^n_t \in \mathbb{R}$:
  - $u^n_t > 0$ indicates charging the SBESS using grid energy.
  - $u^n_t < 0$ indicates discharging the SBESS.

  Overall, since the agents can independently control the SBESS, the individual control action has to be bounded, $\forall n \in N$, as $\underline{u}_n \leq u^t_n + u^{r,t}_n \leq \overline{u}_n$. The collective control action is bounded by $\underline{u} \leq \sum_{n\in N}(u^t_n + u^{r,t}_n) \leq \overline{u}$.
- Individual control actions are bounded for each agent, and collective actions are also bounded across all households.
- Energy generated from waste by each household n at time t is represented by $w_{n,t}$.

About the charging and discharging of the SBESS.

- The change in level-of-charge (LoC) of the SBESS is given by

$$\mathrm{LoC}_{t+1} = \mathrm{LoC}_t + \eta \sum_{n\in N}(u^n_t + u^n_{r,t} + w_{n,t}) \tag{4.7}$$

  where η varies based on whether the action is charging or discharging.

$$\eta = \begin{cases} \eta_{\mathrm{ch}}, & \text{if } \sum_{n\in N}(u^n_t + u^n_{r,t}) > 0, \\ \frac{1}{\eta_{\mathrm{dch}}}, & \text{otherwise} \end{cases}$$

  where, conversion efficiencies η_{ch} and $\frac{1}{\eta_{dch}}$ are used for charging and discharging, respectively.
  The LoC must always remain within the bounds of the SBESS capacity: $0 \leq \mathrm{LoC} \leq \mathrm{LoC}_t \leq B$, where B is the capacity of the SBESS.

Economic Considerations

Households exchange energy with the grid and their costs are influenced by the time-of-use electricity price and the feed-in tariff for selling surplus green energy, including that generated from WTE processes.

$$b_{t,n} = (d_{t,n} - r_{t,n} - w_{t,n} + u_{t,n})^+ c_{ToU} - (r^{\uparrow}_{t,n} + w_{t,n})c_{FiT} \tag{4.8}$$

where,

- $d_{t,n}$ is the demand of household n at time t.
- $r_{t,n}$ is the renewable energy generation (e.g., from solar) of household n at time t.
- $w_{t,n}$ is the energy generated from waste by household n at time t.
- $u_{t,n}$ represents other controllable inputs or withdrawals of energy (e.g., from/to a battery or the grid).
- c_{ToU} is the time-of-use tariff, charged for energy taken from the grid.
- c_{FiT} is the feed-in tariff, credited for energy fed back to the grid from renewable sources and now also from waste.

The goal is to minimize the individual electricity bills through decentralized control of the SBESS and WTE processes, without direct coordination mechanisms, while satisfying all system constraints.

4.2.1 System Model Markov Game

The energy generation model needs to include both WTE and RES. WTE could include the conversion of municipal solid waste or other organic waste into electricity through thermal or biochemical processes. This adds complexity to the model as WTE typically has different generation profiles and reliability factors compared to RES.

State of the Environment includes all variables that define the current situation of the energy system, such as the charge level of the battery, demand profiles, generation from renewable sources and WTE, pricing information...

$\forall t \in T$, where T is the set of times, the state is defined as

$$s_t = \{\mathrm{LoC}_t, c_t^{\mathrm{ToU}}, c_t^{\mathrm{FiT}}, d_{1,t}, \ldots, d_{N,t}, r_{1,t}, \ldots, r_{N,t}, w_{1,t}, \ldots, w_{N,t}\} \tag{4.9}$$

where,

- LoC_t is the level of charge of the shared energy storage at time t given by Eq.(4.7).
- c_t^{ToU} is the time-of-use electricity price at time t.
- c_t^{FiT} is the feed-in tariff for selling energy back to the grid at time t.
- $d_{n,t}$ is the demand of household n at time t.
- $r_{n,t}$ is the renewable energy generated by household n at time t.
- $w_{n,t}$ is the energy generated from waste by household n at time t.

Observations o_n^t are the perceivable subset of the state space that each agent uses to make decisions. Not all state variables might be observable by all agents due to privacy or technical constraints.

$$o_{n,t} = \{\hat{\mathrm{LoC}}_t, c_t^{\mathrm{ToU}}\} \tag{4.10}$$

Each agent n receives local observations related to the state, which may include:

- Local Energy Usage: Each household observes its own energy demand.
- Local Generation Rates: How much energy each household is generating from both RES and WET.
- Shared Storage Information: The level of charge in the shared battery, which might be fully or partially observable.
- $\hat{\text{LoC}}_t$: Estimated or discretized level of charge visible to agent n.
- c_t^{ToU}: Time-of-Use cost at time t.

Control Actions a_n^t are represented by each household which can decide how much power to draw from or supply to the battery based on its current needs and the state of the system.
Agents' decisions consider the optimal use of energy from both RES and WTE, especially in terms of storage management and grid interaction.

$$a_{n,t} \in \{-u_n, -u_n + \Delta u, \ldots, -\Delta u, 0, \Delta u, \ldots, u_n - \Delta u, u_n\} \tag{4.11}$$

where Δu is the step size for discretizing the action space.

Rewards given to the agents are based on several criteria, including economic benefits, the efficiency of energy use, and adherence to constraints like battery capacity and power flow limits.
$r_{n,t+1}$ is a composite measure designed to reflect the immediate cost or benefit of an action taken by agent n at time t, defined as:

$$r_{n,t+1} = w_a r_{a,t+1} + w_{\text{LoC}} r_{\text{LoC},t+1} + w_b r_{b,t+1} + w_w r_{w,t+1} \tag{4.12}$$

where,

- $r_{w,t+1}$: additional reward/penalty related to the optimal use or management of energy from waste.
- w_w: weight associated with the WTE component of the reward.
- $r_{a,t+1}$, $r_{\text{LoC},t+1}$, and $r_{b,t+1}$: rewards or penalties related to the aggregate control action, level-of-charge constraints, and the electricity bill, respectively.
- w_a, w_{LoC}, w_b: weights reflecting the relative importance of each component of the reward.

In particular, according to [A. Joshi and Glielmo 2023]:
Reward for aggregate control action:

$$r_{a,t+1}^n = \begin{cases} -[a_{t,n}]_+/u_n & \text{if } \sum_{n\in N}(a_{t,n} + ur_{t,n}) > u, \\ -[a_{t,n}]_-/u_n & \text{if } \sum_{n\in N}(a_{t,n} + ur_{t,n}) < u, \end{cases}$$

where agents are penalized in proportion to the magnitude of their control action if the aggregate control action results in overcharging or over-discharging the system.
Reward for Level-of-Charge compliance:

$$r_{LoC,t+1}^n = \begin{cases} -[a_{t,n}]_+/u_n & \text{if } LoC_t > LoC, \\ -[a_{t,n}]_-/u_n & \text{if } LoC_t < LoC, \end{cases} \tag{4.13}$$

which penalizes agents based on their control action's impact on the state of charge exceeding or not meeting predefined thresholds.

4.2.2 Consensus Q-Learning

In order to achieve overall energy efficiency and sustainability in a multi-agent reinforcement learning context, it is necessary to handle non-stationarity and the requirement to properly manage the conversion of WTE while coordinating with other renewable energy sources (such solar, wind, etc.).

The authors A. Joshi and Glielmo 2023 expand the consensus Q-learning technique to take into account the complexity of WET in addition to RES in order to manage these complexities successfully.

The traditional Hyper-Q function (Appendix (C)), poses scalability challenges, particularly when integrating the diverse and fluctuating energy outputs from both RES and WET. To address this, one can propose a modified version of the Hyper-Q function that considers both energy sources

$$Q_n(o_n, a_n) \approx Q_n(s, a_n, a_{-n}), \quad \forall n \in N$$

where o_n includes observations related to both RES and WET outputs.

During the offline training phase, agents share information about their respective Q-functions related to energy decisions, enabling them to reach a consensus that optimally balances energy production from waste and renewables.

The Q-learning algorithm considered, employs an ϵ-greedy strategy to explore and exploit actions to get to the best optimal policy π^* by explore and exploit the energy management strategies.
The updating rule for the modified Q-function is given by:

$$Q^n_{t+1}(o^n_t, a^n_t) = Q^n_t(o^n_t, a^n_t) + \alpha_n \cdot TD^n_{t+1} \tag{4.14}$$

where:

- α_n is the learning rate for agent n, controlling how quickly the agent learns from new information.
- TD^n_{t+1} is the Temporal Difference updating rule.

Follows,

$$TD^n_{t+1} = r^n_{t+1} + \gamma \max_{a' \in A} Q_n(o^n_{t+1}, a') - Q_n(o^n_t, a^n_t) \tag{4.15}$$

where,

- r^n_{t+1} is the reward received by agent n at time $t+1$.
- γ is the discount factor, reflecting the importance of future rewards.
- o^n_t and a^n_t are the observation and action at time t for agent n.
- o^n_{t+1} is the new observation after action a^n_t is taken.

Consensus Mechanism

The consensus mechanism, described in Appendix (D.1) ensures that all agents in the network update their estimates in a way that converges towards a common understanding or policy:

$$Q^n_{t+1} = W_{n,n} Q^n_t + \sum_{n' \in \mathrm{NEI}_n} W_{n,n'} Q^{n'}_t \tag{4.16}$$

where,

- $W_{n,n}$ and $W_{n,n'}$ are weights that define how much agent n is influenced by its own previous estimate and the estimates of its neighbors n', respectively.
- NEI_n denotes the set of neighbors of agent n.

The convergence of the Q-values is hypothesized based on the weighted averaging process and the bounded nature of updates [A. Joshi and Glielmo 2023]:

$$\lim_{t \to \infty} \|Q_t - Q^*\| = 0$$

This equation implies that as time progresses, the Q-values for all agents are expected to converge to an optimal Q-function Q^*, which represents the best cooperative strategy for managing both Renewable Energy Sources (RES) and Waste-to-Energy Technologies (WET) efficiently.

Chapter 5

Conclusions

This report investigates the Waste-to-Energy (WTE) framework within an intelligent energy recycling management system. By employing various machine learning models, this study predicts municipal solid waste (MSW) quantities and calorific values in Istanbul, aiming to significantly enhance the efficiency and sustainability of energy recovery from waste [Kaya et al. 2021].

A two-stage predictive framework was implemented, which showcased improvements in WTE prediction of waste through the integration of Machine Learning (ML) technologies [Cartolano 2024]. Stage 1 focused on predicting daily MSW volumes using Multilayer Perceptron (MLP) and Support Vector Machine for Regression (SVR) models. Notably, MLP demonstrated lower Mean Square Error (MSE) and Root MSE compared to SVR, despite requiring more training time. Stage 2 involved the calculation of calorific values based on predicted waste amounts, enabling precise estimations of potential energy outputs, which supports strategic energy production and resource management. In the context of Stage 2, it has been further explored the application of transfer learning to enhance the adaptability of the predictive models; which is a critical aspect in contexts where data scarcity can limit technology deployment. Transfer learning was particularly effective when applied to Neural Networks using the TensorFlow framework, outperforming the predictions on calorific values made by the best model from Stage 1 (MLP).

Additionally, the integration of WTE energy into local energy networks represents a pivotal advancement towards sustainable urban development. Employing decentralized energy strategies through multi-agent reinforcement learning (MARL) enables WTE systems to complement intermittent renewable sources such as solar and wind. This integration not only helps reduce urban carbon footprints but also increases the reliability and efficiency of renewable energy sources, in alignment with global sustainability goals [A. Joshi and Glielmo 2023]. Moreover, the research also examined the economic impacts of advanced WTE systems. By optimizing energy storage and consumption strategies (the energy network provides a SBESS), the framework facilitates reduced energy costs and promotes economic stability in urban environments.

In conclusion, this study highlights the indispensable role of machine learning in bridging waste management with energy recovery. Leveraging intelligent technologies, the WTE framework offers promising pathways for enhancing smart city solutions and sustainable waste management practices, while also fostering significant environmental and economic benefits.

Appendix A

A.1 Stochastic Gradient Descent Solver

Stochastic Gradient Descent (SGD) is a widely used optimization technique in machine learning, particularly for training large-scale models. It is an iterative method for optimizing an objective function, typically a loss function, that is composed of a sum of differentiable functions.

The key feature of SGD is to update of the model parameters using only a single sample, or a small batch of samples, from the dataset at each iteration. This contrasts with traditional Gradient Descent (GD), which uses the entire dataset to compute the gradient of the loss function. The update rule for parameters when using SGD is given by:

$$\theta_{t+1} = \theta_t - \eta_t \nabla L(\theta_t; x_i, y_i),$$

where θ_t are the parameters at iteration t, η_t is the learning rate, $\nabla L(\theta_t; x_i, y_i)$ is the gradient of the loss function computed at the sample (x_i, y_i).

The learning rate η_t is a critical hyperparameter in SGD. It can be constant or vary during training. Popular methods for adjusting the learning rate include time-based decay, step decay, and exponential decay. Adjusting the learning rate is crucial for convergence and to avoid oscillations in the parameter space.

SGD has several advantages:

- Efficiency: Suitable for large datasets and high-dimensional optimization problems.
- Online Learning: Capable of learning on-the-fly as new data arrives.

Despite its advantages, SGD faces several challenges. Among them:

- Sensitivity to hyperparameters: Particularly the learning rate and batch size.
- Convergence: May converge to local minima or saddle points, especially in complex loss landscapes.
- Noise: The stochastic nature introduces noise in the gradient estimates, which can affect the stability of the convergence.

To address these challenges, several variants and improvements of SGD use the following hyperparameters:

- Momentum: Helps accelerate SGD in the right direction and dampens oscillations.

- Nesterov Accelerated Gradient: A modification of momentum that looks ahead in the update direction.
- Adaptive Learning Rate Methods: Such as AdaGrad, RMSprop, and Adam, which adjust the learning rate based on past gradients.

SGD remains a cornerstone of machine learning optimization due to its simplicity and effectiveness. Ongoing research continues to refine its performance and applicability across various domains.

A.2 Adam Optimizer Solver

Adam (Adaptive Moment Estimation) is an optimization algorithm used in machine learning to update network weights in an iterative way, based on training data. It combines ideas from two other extensions of stochastic gradient descent: AdaGrad and RMSProp. Adam is appreciated for its effectiveness in handling sparse gradients on noisy problems.

Adam maintains two variables, v and m, that are estimates of the first moments (the mean) and the second moments (the uncentered variance) of the gradients, respectively. The variables are updated as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t,$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$

where g_t is the gradient at time step t, β_1 and β_2 are exponential decay rates for these moment estimates (typically set to 0.9 and 0.999 respectively). The moment estimates are then bias-corrected:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},$$
$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}.$$

The parameters are updated by:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t,$$

where η is the step size (learning rate), and ϵ is a small scalar (e.g., 10^{-8}) added to improve numerical stability.

Adam has several advantages:

- Efficiency: Adam combines the advantages of AdaGrad, which works well with sparse gradients, and RMSProp, which works well in online and non-stationary settings.
- Adaptive Learning Rate: Adjusts the learning rates based on the average of recent gradients in the model, making it suitable for many different types of machine learning problems without requiring much manual tuning of the learning rate.
- Robustness: Particularly useful in large models and large data scenarios.

While Adam is robust and versatile, it might not converge to the optimal solution under certain conditions, and may require more tuning in terms of decay rates and initialization.

Adam is widely used in the fields of deep learning and artificial intelligence for its simplicity and effectiveness in handling large-scale data and model sizes. Despite some potential for non-optimal convergence, it remains a popular choice for training neural networks.

Appendix B

B.1 Kernel Trick

The kernel trick is a method used in machine learning to enable algorithms that only support linear class separation to operate in higher-dimensional spaces without explicitly performing the transformation. This technique is commonly associated with Support Vector Machines (SVMs) but is applicable to any algorithm that relies on dot products between vectors.

The kernel trick involves substituting the standard dot product with a kernel function. This kernel function implicitly maps data into a higher-dimensional feature space without explicitly performing the transformation. Mathematically, if we have a mapping $\phi : \mathcal{X} \to \mathcal{F}$ that transforms original feature space $\mathcal{X}$ into a higher-dimensional feature space $\mathcal{F}$, the kernel function K is defined as:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle,$$

where $x, x' \in \mathcal{X}$ and $\langle \cdot, \cdot \rangle$ denotes the dot product in $\mathcal{F}$.

Several kernel functions are commonly used:

- Linear Kernel: $k(x, x') = x^\top x'$
- Polynomial Kernel: $k(x, x') = (1 + x^\top x')^d$, where d is the degree of the polynomial. It involves creating new features that are powers of the original features, allowing the model to capture polynomial relationships.
- Radial Basis Function (RBF) Kernel: $k(x, x') = \exp(-\gamma \|x - x'\|^2)$, where γ is a parameter that determines the spread of the kernel. It introduces a similarity measure based on the distance between data points, effectively capturing local patterns in the data.
- Sigmoid Kernel: $k(x, x') = \tanh(\alpha x^\top x' + c)$.

The kernel trick has several advantages:

- Computational Efficiency: Allows algorithms to operate in high-dimensional spaces without the need to compute the coordinates of the data in that space, which can be computationally expensive.
- Flexibility: Different kernels can be chosen based on the problem at hand, allowing for custom solutions tailored to specific data characteristics.

- Enhanced Performance: By allowing linear classifiers to achieve non-linear classification, it enhances the performance of models on complex datasets.

The kernel trick is widely used in various applications, including:

- Support Vector Machines: Enhancing their capability to separate data linearly by transforming it into a higher-dimensional space.
- Principal Component Analysis (PCA): Kernel PCA uses the kernel trick to perform PCA in a high-dimensional feature space, achieving better data separation.
- Pattern Recognition and Anomaly Detection: Helps in complex scenarios where data is not linearly separable in the original feature space.

The kernel trick is a pivotal technique in machine learning, providing a powerful way to enhance the capability of linear models and allowing them to perform non-linear classification. Its versatility and efficiency make it an invaluable tool in the arsenal of machine learning practitioners.

Appendix C

Preliminaries of Reinforcement Learning

In Reinforcement Learning (RL), the agents learn to make decisions by taking actions in an environment to maximise some notion of cumulative reward. The agent learns from trial and error, adjusting its strategy to improve its long-term reward.

This is done through exploration and exploitation balance which makes the agent to avoid to be stuck in a behaviour that does not maximises the reward.
Exploration means that the agents seek out in the environment new knowledge by making new decisions and taking new paths even if this means accepting lower immediate reward but it might help it to make better decisions in the future.
Exploitation, instead, means that the agent uses the knowledge that it already has to make decisions which it knows it will maximise the immediate reward.

C.1 Markov Game

The multi-agent environment is formalized as a Markov game, defined as:

$$MDP =< S, A, \delta, r >$$

where,

- **N**: number of agents;
- **S**: state space;
- **A**=$A_1 \times ... \times A_N$: joint action space;
- $\delta : S \times A \times S \rightarrow [\ 0, 1]$: state transition probability function;
- $r_n : S \times A \times S \rightarrow \mathbb{R}$: reward function for agent n;
- $\gamma \in [\ 0, 1]$: discount factor

A MDP assumes the Markov Property, which implies that the future state depends only on the current state at time t and the action taken at time t and not the sequence of events that preceded it: $x_{t+1} = \delta(x_t, a_t)$.

The goal of the learning agents(s) is to find an optimal ("joint") policy $\pi^* : S \rightarrow A$ (also called behaviour function) that dictates the best action to take in each state

to maximise the cumulative reward.
Accordingly, the agent can learn a function called as the Hyper action-value function or Hyper-Q function $Q_n(s, a)$ from which it could determine the optimal policy.

$Q_n^\pi(s, a)$ is the expected utility of taking action a in state s under policy π for agent n

$$Q^\pi(s, a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \mid s^t = s, a_n^t = a_n, \pi\right]$$

C.2 Q-Learning

It is one of the most widely used RL algorithms for solving MDP.
The algorithm consists in iterative update the estimate of the Q-function using, for example, an ϵ-greedy strategy (balance between exploration and exploitation).
The update equation is given by:

$$Q_{\text{new}}(s_t, a_t) = Q(s_t, a_t) + \alpha\left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)\right] \tag{C.1}$$

where,

- α is the learning rate, which controls how much the Q-values are updated at each step.
- r_t is the reward received after taking action a_t.
- γ is the discount factor, used to decrease the importance of future rewards.
- s_t and s_{t+1} are the current and next states, respectively.
- a_t is the action taken at state s_t.
- $\max_{a'} Q(s_{t+1}, a')$ is the maximum predicted reward achievable in the next state s_{t+1}, which helps in estimating the future value.

The Q-learning updating rule Eq.(C.1) converges to the optimal action-value $Q^*, \forall s^t \in S, \forall a^t \in A \; as \; t \to \infty$ as long as the rewards are bounded; all actions are repeatedly sampled in all states and $\sum_t \alpha^t = \infty$, $\sum_t (\alpha^t)^2 < \infty$ [A. Joshi and Glielmo 2023].

Appendix D

Consensus Mechanism

D.1 Distributed Averaging Consensus

In a multi-agent context, consensus is a common goal one wants the agents to achieve.

The exchange of information among the learning agents, in the [A. Joshi and Glielmo 2023] context, is modelled as a fixed undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices or nodes, each node represent each agent, and $\mathcal{E} \subseteq \{1, ..., n, ..., N\} \times \{1, ..., n, ..., N\}$ is the set of edges over the vertices, which represent the relationships between the agents. To facilitate consensus among agents, a distributed averaging approach is used. Each agent n updates its estimate of the Q-function based on its own update and the estimates from neighboring agents, using a weighted average, according to [A. Joshi and Glielmo 2023]:

$$x_n^{k+1} = W_{n,n} x_n^k + \sum_{n' \in N_{\text{EIn}}} W_{n,n'} x_{n'}^k$$

where,

- x_n^k is the estimate of the Q-function for agent n at iteration k.
- $W_{n,n'}$ is the weight matrix representing the influence of agent n' on agent n.
- N_{EIn} is the set of neighbors of agent n, denoting the agents that directly influence the state of agent n through their actions or states.

The distributed averaging (consensus) mechanism ensures that all agents' Q-functions converge towards a common function, which promotes cooperative behavior and robustness against changes in the environment.
This is possible if and only if $\mathbf{1}^T W = \mathbf{1}^T$, $W\mathbf{1} = \mathbf{1}$, and $\sigma(W\mathbf{1}\mathbf{1}^T/N) < 1$

Bibliography

Binder, James Clovis Kabugoa Sirkka-Liisa Jämsä-Jounelaa Robert Schiemannb Christian (2020). "Industry 4.0 based process data analytics platform: A waste-to-energy plant case study". In: *Electrical Power and Energy Systems* 115, p. 105508.

Kaya, Kiymet et al. (2021). "Waste-to-Energy Framework: An intelligent energy recycling management". In: *Sustainable Computing: Informatics and Systems* 30, p. 100548.

Abidur Rahman Omar Farrok, Md Mejbaul Haque (2022). "Environmental impact of renewable energy source based electrical power plants: Solar, wind, hydroelectric, biomass, geothermal, tidal, ocean, and osmotic". In: *Renewable and Sustainable Energy Reviews* 161, p. 112279.

Alireza Tajeddin, Elham Roohi (2019). "Designing a reliable wind farm through hybridization with biomass energy". In: *Applied Thermal Engineering* 154, pp. 171–179.

A. Joshi, M. Tipaldi and L. Glielmo (2023). "A Consensus Q-Learning Approach for Decentralized Control of Shared Energy Storage". In: *IEEE Control Systems Letters* 7, pp. 3447–3452.

Cartolano, Ludovica (2024). *CCEN_Cartolano1796046_WES_IMPLEMENTATION.py*. Python code for CCEN Implementation task.