university of groningen

faculty of science and engineering

Technical Report: A Three-day Forecasting of O_3 and NO_2 levels in Utrecht, The Netherlands

Machine Learning for Industry Practical

WBAI070-05.2024-2025

Author(s):	Arsenijs Golicins Nika Krivic Iulia Capralova

Contents

Executive Summary

This project focuses on building a scalable automated air pollution forecasting system to predict NO_2 and O_3 levels in Utrecht over a three-day period. Pollutant prediction is based on forecasted meteorological variables. After comparing several models models—including LSTM, MLP, and Linear Regression—we found that the MLP outperformed the others, making it the most suitable choice for deployment. The deployed model features a user-friendly interface that provides accessible predictions, along with an alert mechanism to notify users of unusual or extreme conditions.

1 Introduction

1.1 Problem Statement

In today's highly industrialized and rapidly developing world, technological advancement and economic growth have significantly improved people's well-being and living standards worldwide [1]. However, this progress comes at a cost, as industrial activities, urbanization, and an increase in transportation have led to increased air pollution [2]. Factories, power plants, and vehicles release a wide array of pollutants into the atmosphere, leading to reduced air quality across the globe. This remains a major concern for global public health as it leads to a number of illnesses and premature deaths [3]. Around 92% of the global population now lives in regions where air quality fails to meet health-based standards [4].

Despite ongoing efforts in the Netherlands to reduce air pollutants each year, the country's emissions remain comparable to those of other industrialized nations [5]. The cities are becoming the epicenter of pollution and Utrecht is one of the country's leaders in traffic use and industrial activities. Among the pollutants emitted, O_3 and NO_2, are particularly at risk to human health and contribute to global warming. Accurate short-term forecasting of these pollutants is important in avoiding exposure risks, and taking needed measures to improve air quality. Since the dynamics of pollutant formation are influenced by many weather and human factors, reliable forecasting becomes a complex task. The aim of this study is to address this challenge by developing a predictive model that can forecast the concentrations of O_3 and NO_2 in Utrecht for the next three days.

1.2 Background

1.2.1 Health Implication

An imbalance in the atmosphere of substances such as nitrogen dioxide (NO_2) and ozone (O_3) has serious consequences for both nature and human health.

A primary[1] pollutant NO_2 mainly comes from burning fossil fuels such as coal, gas, and oil, especially in car engines. It reacts with water, oxygen, and other chemicals to form nitric acid, a key component of acid rain [6]. It pollutes freshwater resources and damages forests and agricultural land. Additionally, it plays a main role in the formation of ground-level ozone (O_3).

While in the stratosphere[2] ozone (O_3) protects life on Earth by absorbing harmful ultraviolet (UV)

[1] A primary pollutant refers to an air contaminant that is released directly from a specific source.

[2] The stratosphere is the second layer of the Earth's atmosphere, 30 km above the ground.

radiation, it becomes a harmful secondary[3] pollutant in the troposphere[4]. Ground-level ozone is a key component of smog [7], a harmful mixture of pollutants that forms in the atmosphere, especially in urban areas. Once inhaled, this gas becomes a powerful respiratory irritant that can cause a number of health problems. Due to its low solubility in water, it can penetrate deeply into the lungs where it forms secondary oxidation products. Consequently, these products cause oxidative stress, inflammation, and potential damage to the respiratory tract [8].

Given the harmful effects of ground-level ozone and nitrogen dioxide on human health and ecosystems, the World Health Organization (WHO) has developed air quality guidelines (AQG) to reduce these risks. They are represented in Table 1. The AQG establishes recommended maximum levels of O_3 and NO_2 that should not be exceeded to protect public health. Recent reports show that these have not yet been reached in the Netherlands and many other EU countries.[9]

	Averaging time	**AQG level ($\mu g\ m^{-3}$)**
NO_2	24-hour	25
	Annual	10
O_3	8-hour	100
	Peak-season[a]	60

Table 1: AQG levels for O_3 and NO_2 pollutants [10]
[a]Indicates the average daily maximum 8-hour mean in the consecutive months with the highest six-month running average.

1.2.2 Atmospheric Interaction

Before analyzing the data and predicting NO_2 and O_3 values, it is important to understand the chemistry behind pollutant formation and their interactions. This can explain the patterns and cause-and-effect relationships among themselves and between the features in our dataset.

According to Sagar V. Krupa [11], ozone (O_3) in the troposphere is formed primarily from nitrogen oxides (NO_x) released by the combustion of fossil fuels. Under the influence of sunlight, NO_x produces ground-level ozone. Firstly, the ultraviolet (UV) rays from sunlight cause nitrogen dioxide (NO_2) to break apart into nitric oxide (NO) and a single oxygen atom (O). This is represented by the following reaction:

$$NO_2 + h\nu(\lambda < 430\,\text{nm}) \rightarrow NO + O \tag{1}$$

where $h\nu$ represents the energy of a photon. The free oxygen atom, obtained as a result of photolysis[5], has unpaired electrons, which makes it unstable. To complete its outer electron shells, it easily reacts with a naturally found oxygen molecule (O_2) to form a more stable compound. This process creates ozone (O_3):

$$O + O_2 + M \rightarrow O_3 + M \tag{2}$$

[3]A secondary pollutant is formed from primary pollutants
[4]The troposphere is the lowest layer of Earth's atmosphere, where weather events occur.
[5]Photolysis – the decomposition or separation of molecules by the action of light.

Here, M (usually N_2 or O_2) represents a molecule that absorbs the excess energy released during ozone formation, thereby stabilizing the newly created ozone molecule.

In the absence of UV light (night time), the NO_2 remains stable, thus different reactions take place. Specifically, it reacts with O_3, leading to the generation of nitrate (NO_3) and oxygen (O_2):

$$NO_2 + O_3 \rightarrow NO_3 + O_2 \tag{3}$$

Therefore, it is clear that pollutants such as NO_2 and O_3 are chemically interconnected and influence one another.

1.2.3 Air Pollution in Connection to Meteorological Parameters

Although pollutant concentrations may appear to follow simple rules (Eq. 1, 2, 3), many complex factors are involved. They are governed by complex interactions within Earth's dynamic atmosphere, where local meteorological conditions significantly influence their distribution.

The report on tropospheric ozone in the EU [12] describes episodes of high-level O_3 over regions in Europe occurring during anticyclonic conditions, increased sunlight, high temperatures, and low wind speed. Other previous studies [13, 14] have shown a correlation between pollutant levels and the daily average values of maximum temperature, solar radiation, and zonal wind speed.

Current research suggests the most important factors are:

- Solar radiation and temperature are directly related, with higher levels favoring ozone formation, making sunshine duration a suitable predictor.
- Wind speed, as more stagnant atmospheric conditions give more time for NO_2 accumulation and ozone production.
- Dew point temperature, as it influences the cloud cover.

The present paper considers some additional measurements (Section 2.3). This includes the air pressure (as the wind speed is proportional to the air pressure gradient as air travels from high to low-pressure areas), as well as the wind direction and maximum wind gust (since these introduce unstable atmospheric conditions).

Studies indicate that meteorological influences on pollution vary by season and location (e.g., e.g. China [13], US East Coast [14]). Interactions among meteorological factors require careful analysis to identify redundancy in predicting concentrations. Machine learning methods excel in this area, enhancing pattern recognition across seasons and shorter time frames.

1.2.4 Previous Research

Existing studies on pollutant concentration prediction have been mainly focused on applying simpler Artificial Neural Networks (ANNs) and other machine learning methods, often achieving high accuracy with region-specific data [15, 16, 17].

These models work with different data types and varying time periods between the measurements (e.g. hourly, daily) as well as varied amounts of data. They can reach different prediction horizons, the majority making a short-term forecast with rough temporal resolution [16]. The models also differ in the features included, some combining chemical and meteorological predictors [17], some

meteorological and transportational (traffic) [16] etc. Many researchers note problems with multicollinearity (see Section 1.2.3) and suggest dimensionality reduction and feature extraction (mostly PCA). [15]

Yafouz et al. [15] compare various ML algorithms, including tree regression, support vector regression, ensemble regression, and ANNs, for short-term ozone concentration prediction.

Du et al. [16] trained on 10 years of data (O_3 concentration) and used frequent pattern data mining algorithms to reveal relationships between the parameters. They used the XGBoost ML algorithm and reached a very accurate, long-term forecasting with a high temporal resolution.

Over the last few years, deep learning methods have also been applied to this problem. In particular, Freeman et al. [17] developed a deep learning model including RNN and LSTM structures that predicts O_3 levels up to 72 hours with very low errors compared to previous studies. They also emphasise a reduction in variable numbers, going from 25 to 5 features using a decision tree for feature importance.

1.3 Project Objectives

The main aim of this project is to create a machine learning model that accurately and confidently makes a three-day forecast of O_3 and NO_2 levels, based on the data collected in a weather station near Utrecht, The Netherlands. This includes exploratory data analysis to assess intercorrelations among meteorological parameters and their relationships with pollutant levels, helping to identify seasonal and daily trends. Insights from this analysis guide model selection, training, and optimization. A data preprocessing and feature engineering pipelines will be developed based on the data obtained. Accurate predictions from the model could be valuable for sectors like environmental management and public health officials.

1.4 Structure of the Report

The report is structured in the following way. Section 2 covers data collection, sources, preprocessing techniques, and feature engineering, detailing how features and pollutant data were gathered and prepared for modeling. Section 3 discusses model training and evaluation, including a brief theoretical description of the chosen models, selected architecture, and comparison of various forecasting methods. Section 4 focuses on the deployment, monitoring, and maintenance of the selected model. Section 5 presents the results, highlighting model performance and the challenges encountered. Finally, Section 6 provides the conclusion, summarizing the project's achievements and future directions.

2 Data Collection and Preprocessing

2.1 Chapter Highlights

In this section, we inspected the dataset and, specifically, analyzed both weekly and yearly patterns. We observed distinct daily fluctuations as well as seasonal trends in pollutant levels. It was noted that ozone levels tend to peak during the warmer months, while nitrogen dioxide levels are higher during colder months, often exceeding recommended limits recommended by WHO. Additionally, we examined the data distribution, which was right-skewed, indicating the need for transformations to normalize the data. Finally, the correlation matrix revealed strong intercorrelations between certain meteorological features.

2.2 Data Sources and Format

The datasets were obtained from the databases of the Royal Netherlands Meteorological Institute (KNMI), which is part of the Ministry of Infrastructure and Water Management of the Netherlands [18]. In particular, data from Air Quality Monitoring Network (Luchtmeetnet) was used to obtain the data for O_3 and NO_2 [19]. In Utrecht, this network has three automatic measuring stations, out of which one of them, namely, Utrecht-Griftpark was selected [20].

Both O_3 and NO_2 were measured in the $\mu g/m^3$ units, once for every hour in a day. The selected range of measurements were chosen by us to be from the 1st of January 2016, to the 31st of Deccember 2021, skipping the year 2019, as there were a lot of missing values. The Utrecht-Griftpark station is labelled as "staad" (city) station, and is located right next to a motor vehicle road. All of the Luchtmeetnet stations have chemiluminescence as their measurement principle.

Another part of the data, namely a wide range of weather variables, such as radiation, temperature, humidity, air pressure, sunshine duration etc. were obtained from another station's dataset of KNMI, known as "weerstations" (Weather stations) [21]. The closest weather station located to our chosen air quality station was located in the De Bilt, which is a municipality adjacent to the city of Utrecht.

2.3 Data Inspection

2.3.1 Dataset contents

In this project we are interested in two pollutants:

- O_3: Ozone ($\mu g/m^3$)
- NO_2: Nitrogen dioxide ($\mu g/m^3$)

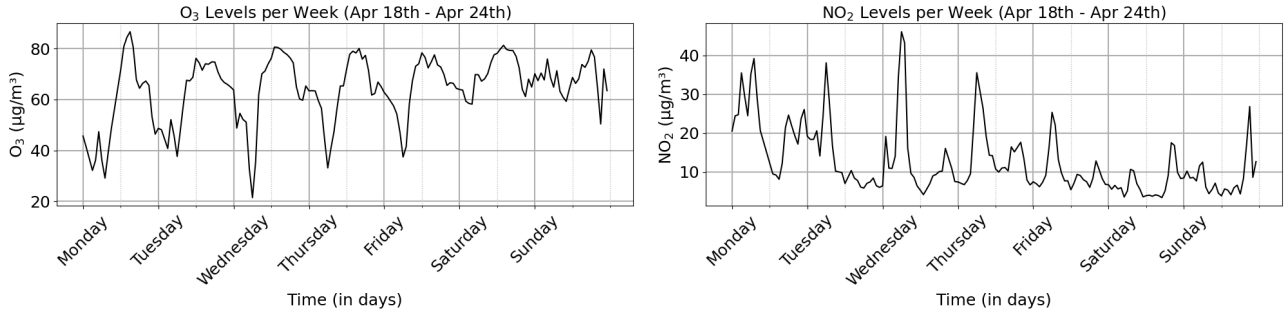
We consider the following meteorological variables as their possible predictors. We renamed the explanatory variables' abbreviations from Dutch to English to ensure consistency between acronyms and descriptions.

- MWD: Mean wind direction (in degrees) during the 10-minute period preceding the time of observation (360=north; 90=east; 180=south; 270=west; 0=calm; 990=variable)
- MWS: Hourly mean wind speed (in 0.1 m/s)
- MWS10: Mean wind speed (in 0.1 m/s) during the 10-minute period preceding the time of observation
- WG: Maximum wind gust (in 0.1 m/s) during the hourly division
- T: Temperature (in 0.1 degrees Celsius) at 1.50 m at the time of observation
- TD: Dew point temperature (in 0.1 degrees Celsius) at 1.50 m at the time of observation
- S: Sunshine duration (in 0.1 hour) during the hourly division; calculated from global radiation (-1 for $<$ 0.05 hour)
- GR: Global radiation (in J/cm^2) during the hourly division
- P: Air pressure (in 0.1 hPa) reduced to mean sea level; at the time of observation

2.3.2 Pollutants Over Time

To better understand the patterns in ozone (O_3) and nitrogen dioxide (NO_2) levels, we visualize their distributions over varying time periods.

Focusing on a single week (Apr 18th - Apr 24th), we examine daily and weekly pollutant dynamics. As shown in Figure 1a, ozone concentrations peak during daylight, especially around midday (12 PM). Conversely, NO_2 levels are higher at the beginning and end of the day (Figure 1b), likely due to traffic emissions during rush hours [22]. These trends align with the photochemical reactions discussed in Section 1.2.2, where morning NO_2 emissions under sunlight initiate O_3 formation.

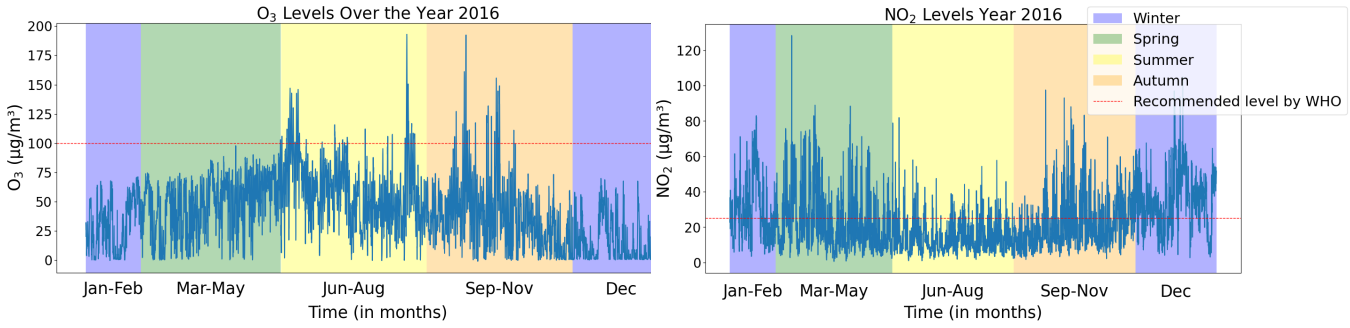


(a) (b)

Figure 1: (a) Ozone (O_3) and (b) Nitrogen dioxide (NO_2) levels across the sixth week of the year 2016 (Apr 18th - Apr 24th)

To capture seasonal variations, we examine pollutant levels over the entire year 2016. Ozone (O_3) levels display a clear seasonal pattern, with higher concentrations in the warmer months (June to August) and lower levels in the colder months (December to February) [23] (Figure 2a). This aligns with the photochemical processes driven by sunlight and higher temperatures. A pattern is also observed in NO_2 levels (Figure 2b). Its levels are higher in colder months, likely due to increased heating and traffic emissions during Winter and Autumn.

Moreover, both year-long pollutant graphs reveal frequent exceedances of the World Health Organization (WHO) recommended levels noted in Section 1.2.1, showing a need for continuous monitoring and regulation of air pollution.

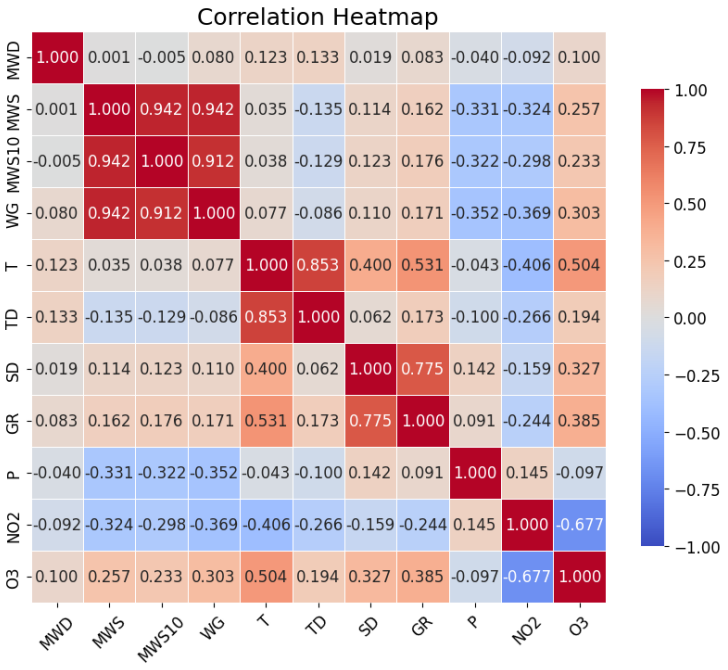


(a) (b)

Figure 2: (a) Ozone (O_3) and (b) Nitrogen dioxide (NO_2) levels levels across the the year 2016 (Jan 1st - Dec 31st)

2.3.3 Correlation Analysis

In addition to analyzing the distribution of the pollutant variables, it is important to understand the relationships between them and other features. A correlation matrix shows how strongly features are related to one another through correlation coefficients, which quantify the degree of linear relationship between two variables. This relationship is expressed in Equation 3b.



(a)

$$r_{XY} = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}} \tag{4}$$

(b)

Figure 3: (a) Correlation matrix between features and predictive variables; (b) Equation for calculating the correlation coefficient.

A good approach to feature selection involves identifying features that have a high correlation with the target variable but a low intercorrelation with other features [24]. This way we avoid multicollinearity, thus each feature contributes unique information to the model.

First, features with very low correlation to the pollutant variables may be discarded, as they will most likely not provide any additional information for predictions. These include the mean wind direction (MWD, corr = -0.092, 0.1) and air pressure (P, corr = 0.15, -0.097). According to Figure 3a, hourly mean wind speed (MWS), mean wind speed during the 10-minute period (MWS10), and maximum wind gust (MW) exhibit very high positive correlations with each other, all around 0.94 to 0.95. For this reason, MWS10 is discarded, as we obtain essentially the same measurements from MWS itself. Temperature (T) and Dew point (TD) also show a strong positive correlation of 0.84, and so do sunshine duration (SQ) and Global radiation (Q) (0.78). These suggest the need for dimensionality reduction techniques, such as PCA, to handle multicollinearity.

The current set of features chosen as possible predictors includes 6 of the initial list: [MWS, WG, T, TD, SD, GR].

2.4 Outliers

In this analysis, outliers in the O_3 and NO_2 time series were identified using a Z-score method. It measures how far data points deviate from the mean in terms of standard deviations. A data point was classified as an outlier if its Z-score exceeded a threshold of three standard deviations away from the rolling mean:

$$Z_i = \frac{x_i - \mu_{\text{rolling}}}{\sigma_{\text{rolling}}},$$

where x_i is the observed value at time i, μ_{rolling} is the rolling mean over a window of 24 hours centered on i, and σ_{rolling} is the rolling standard deviation over the same window.

Our results identified 8 outliers in O_3 and 74 in NO_2. These represent valid high-value measurements, potentially reflecting external factors not captured by our model parameters. Consequently, they were retained in the dataset, as they can act as regularizers, increasing model robustness [25].

2.5 Data Preprocessing

The data is extracted from different sources, transformed into the desired format and loaded into the target destination. A DataSet class is created that handles certain data preprocessing steps but mainly focuses on cleaning, organizing, and loading the data into a CSV file at a specified location.

Since data formats depend on the source organization (KNMI), pollutant data from the Air Quality Monitoring Network and meteorological data from De Bilt Weather Station require alignment. Before combining the O_3 and NO_2 readings with the meteorological information at the corresponding date and hour, all data files need to be loaded into a common format. The DataSet class handles this by loading pollutant data from multiple files and meteorological data from a single file. To match formats, the pollutant data's date format is converted from ISO-8859-15 to YYMMDD and HH, consistent with meteorological data. Additional adjustments, such as omitting certain columns (based on a later selection of features), renaming (based on the English abbreviations proposed in section 2.3.1, and type conversions, ensure a common format for integration.

The datasets include 43848 data points in total. A check for missing values shows 350 for O_3 measurements, 79 for NO_2, and 0 for all the meteorological variables. There is no pattern in the missing values, and a certain variable is not missing a value because of another, therefore they are completely at random. Freeman et al. [17] adjust the imputation techniques based on the number of consecutive null values. Since we are dealing with continuous, hourly measurements, we adopt their approach. In particular, if the gap g is smaller than 8, a linear estimate of the missing data at position n ($0 < n <= g$), X_n, is calculated based on the last and the next provided values that are not null:

$$X_n = X_{n-1} + n\Delta$$

where

$$\Delta = \frac{X_{g+1} - X_0}{g+1}$$

If the gap is larger than 8, the values corresponding to the same hour measured a day before and a day after (if these exist) are averaged:

$$X_i = \frac{X_{i+24} - X_{i-24}}{2}$$

There is no need to discretize any of the variables. All the features are continuous.

The DataSet class allows its instances to use the indexer to get a specific data point, and a length operator. The O_3, NO_2, and meteorological data frames are loaded to separate CSV files or concatenated into a single data frame, which can be loaded to a specified location and used for further analysis.

2.6 Feature Engineering

As part of the feature engineering process, (1) new features are created, (2) the data is split into training, validation, and test sets, (3) the data is normalized, (4) dimensionality reduction is applied, and (5) the final input data is generated using the sliding window approach.

2.6.1 Feature Construction via the Rolling Average Approach

For generating new predictive variables we adopted techniques introduced by Yin et al. (2023) [26]. The authors used lagging and rolling window techniques for air pollution prediction, demonstrating that both short-term fluctuations and long-term dependencies in pollutant data were better captured.

The rolling window method in time series analysis smooths short-term fluctuations and captures longer-term trends by calculating summary statistics (e.g., mean) over a moving window. This technique helps the model focus on underlying patterns rather than transient spikes. Given the frequent short-term fluctuations in meteorological features, this method is well-suited for feature creation. We used window sizes of 4, 12, and 24 hours to calculate rolling means, allowing the model to incorporate both short- and longer-term weather trends.

The current set of features after this step includes 6 (original features) + 6 $\times$ 3 (generated features) = 24 features: [MWS, MWS_4, MWS_12, MWS_24, WG, WG_4, WG_12, WG_24, T, T_4, T_12, T_24, TD, TD_4, TD_12, TD_24, SD, SD_4, SD_12, SD_24, GR, GR_4, GR_12, GR_24] where for feature X at time t, X refers to the measurement at time t and X_n refers to the average of the feature over the last n hours.

2.6.2 Train/Test split

For data splitting, we allocated 80% of the data for training and 20% for testing. Within the training set, 75% was used for model training, with the remaining 25% reserved for validation. The dataset was split chronologically to prevent potential leakage, resulting in the following final data distribution:

1. Training set: 60% of the total dataset (full years 2016, 2017, 2018).
2. Validation set: 20% of the total dataset (full year 2020).
3. Test set: 20% of the total dataset (full year 2021).

2.6.3 Normalization

Normalization is applied individually to each data split for model learning stability and convergence. Features with larger ranges or scales can disproportionately influence the model, leading to skewed results or longer training times. By rescaling features to a specific range, models reach faster and improved convergence [27]. The target variables are also normalized, which reduces the gradient magnitude during weight updates, which contributes to a more stable and smooth training process [28]. For our dataset, we applied a Min-Max scaler. It transforms the data by rescaling each feature to a fixed range between 0 and 1. The formula for Min-Max scaling is:

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}},$$

where: X is the original value, X_{min} and X_{max} are the minimum and maximum values of the feature, respectively.

2.6.4 PCA Dimensionality Reduction

PCA is a method that transforms the dataset by rotating it to create new features, known as principal components, that are statistically uncorrelated with each other. It starts with finding a direction of maximum variance (the direction along which data contains the most information). This direction is labeled "Component 1." "Component 2" is then found as the direction that contains the most information while being orthogonal to Component 1, and so on for other components. Afterwards, a subset of these components is chosen, based on how important they are for explaining the data.

As a result of previous steps our dataset now consists of 24 features and 2 target variables (O_3 and NO_2). Moreover, according to Section 2.3.3, some of the variables are strongly correlated. This means that not all of them should be used for training, as it can quickly lead to model overfitting. To decrease the number of non-important features while retaining the variance in the dataset, a Principle Component Analysis was used.

In our data, it was found that 100% of variance was explained by 15 components. Therefore, we reduced our data to this dimension.

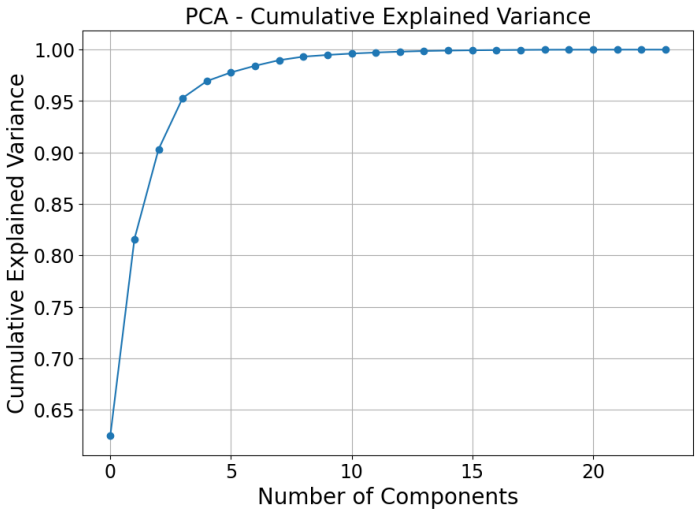


Figure 4: Variance Explained by four PCA components.

2.6.5 Sliding window approach

When working with multivariate time series data, machine learning models learn from sequences of past observations to predict future values. Rather than using individual data points, models rely on windows of time to capture the temporal dependencies between the target variable and related features.

Algorithm 1 Splitting multivariate time series data into input-output sequences

Input: $n\_steps \geq 0, n\_steps\_out \geq 0$

Output: Input-output sequences for both target and feature variables

$X \leftarrow []$; // Empty list to hold input sequences
$y\_target \leftarrow []$; // Empty list for target output sequences
$y\_features \leftarrow []$; // Empty list for feature output sequences
for $i \leftarrow 0$ **to** $length(sequences)$ **do**
  $end\_ix \leftarrow i + n\_steps$; // Calculate end index of the input window
  $out\_end\_ix \leftarrow end\_ix + n\_steps\_out$; // Calculate end index of the output window
  if $out\_end\_ix > length(features)$ **then**
    break ; // End if target window exceeds sequence length
  end if
  $seq\_x \leftarrow sequences[i : end\_ix, :]$; // Get specific slice of the data
  $seq\_q\_target \leftarrow target[end\_ix : out\_end\_ix, 0]$
  $seq\_q\_features \leftarrow sequences[end\_ix : out\_end\_ix, :]$
  Append seq_x to X
  Append seq_q_target to y_target
  Append $seq\_q\_features$ to $y\_features$
end for
return $X, y\_target, y\_features$

As shown in the pseudo-code [6] above, the sliding window method creates consecutive sequences (windows) from the data. This approach captures a fixed number of past observations (input window) to predict future values over a specified number of steps (output window). In this example, an input window of three time steps and an output window of one time step were selected through trial and error. Starting at the beginning of the sequence, the input window spans three past observations, with the output window positioned immediately after to capture the next time step. As the window "slides" forward by one step at a time, each shift creates a new input-output pair, with the process continuing until insufficient data remains to form complete windows. As we move forward through the sequence, the window "slides" one step at a time. Each time we shift by one time step, we redefine the input window to capture the next three steps in the sequence, and the output window shifts to capture the next single step just after these three. This process repeats across the entire sequence until we reach a point where there aren't enough future steps to fill both the input and output windows. At that stage, the windowing process stops, as there's not enough data left to form complete windows. This method produces overlapping input-output pairs, each containing three prior steps as input and the immediate next step as output.

3 Model Training and Evaluation

3.1 Chapter Highlights

This chapter focuses on the training and hyper-parameter tuning of three models: Elastic Net, Long Short-Term Memory (LSTM), and Multi-Layer Perceptron (MLP). Each model is suitable for multivariate time series forecasting tasks like ours. We will explore how these models are trained on the dataset and fine-tune their parameters to optimize performance.

[6] The algorithm was inspired by Machine Learning Mastery [28].

3.2 Linear Regression - Elastic Net

Linear regression models are often used as baselines in predictive modeling [29] due to their simplicity and interoperability [30]. These models assume a linear relationship between the input features and the target variable, by minimizing the difference between predicted and actual values [30]. Despite their simplicity, linear regression models often perform well in many tasks, making them a natural choice for initial benchmarks before moving on to more complex models. However, when applied to real-world data, linear regression has limitations, particularly when the dataset contains a large number of features and it becomes prone to overfitting. Therefore, regularization techniques (Lasso and Ridge) are applied, as described in the scientific paper by Melkumova and Shatskikh (2017) [31].

Ridge regression addresses overfitting by shrinking the coefficients of less important features. This is done by adding an L2 penalty (squared magnitude of coefficients) to the cost function, which forces the model to spread out the coefficient weights rather than allowing a few large ones to dominate the model:

$$\min_{\beta_0,\beta_1} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2, \tag{5}$$

where λ is the regularization parameter, and β_j^2 is the L2 penalty.

Lasso regression, or L1 regularization, introduces a penalty on the absolute value of the coefficients, which results in some coefficients being reduced to exactly zero. This makes Lasso useful for both shrinkage and feature selection, as it allows the model to select only the most important features while ignoring the rest. However, Lasso has limitations when handling highly correlated features. It arbitrarily picks one feature from a group of correlated ones and discards the others, even if they are important. Lasso can be expressed by the following equation:

$$\min_{\beta_0,\beta_1} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|, \tag{6}$$

where, λ is the regularization parameter, and $|\beta_j|$ is the L1 penalty.

In order to use the strengths of both traditional linear regression models (Lasso and Ridge), we used Elastic Net, which combines L1 and L2 regularization into a single model [32].
It's goal is to minimise the following objective function:

$$\hat{\beta} = \arg\min_{\beta} \left(\frac{(y - Xw)^2}{2 \cdot n\_samples} + \alpha \cdot L1\_ratio \cdot w + 0.5 \cdot \alpha \cdot (1 - L1\_ratio) \cdot w^2 \right), \tag{7}$$

where X is feature matrix, y is target variable, w are the weights of the model, a is regularization strength and $L1\_ratio$ is the balance between L1 regularization and L2 regularisation.

3.3 Multilayer Perceptron (MLP)

While linear regression works well for modeling simple relationships, our task of predicting pollutant levels might require capturing more complex, non-linear dependencies (e.g., seasonality). This is where Multilayer Perceptrons (MLPs) come into play.

MLPs are used for supervised learning tasks, where they learn to map input vectors to output vectors based on training data. The input data, $u \in R^K$, and corresponding output $y \in R^M$, are drawn from a joint distribution $P(U,Y)$, and the MLP's goal is to approximate the mapping from inputs to outputs by minimizing a chosen loss function.

An MLP is composed of an input layer, hidden layers, and an output layer. The input layer has K units that receive input, and the output layer has M units that produce the output. Between these, hidden layers perform internal computations, passing signals forward through a network of neurons connected by weighted synapses. These synaptic weights determine how input signals are processed and how outputs are generated. The weights are adjusted iteratively by backpropagation through time (BPTT), until the MLP function $N : R^K \rightarrow R^M$ matches the desired input-output relationship found in the training data. Training an MLP involves finding the optimal set of weights that minimize the error between the predicted outputs and the true outputs. This is typically achieved by minimizing the loss on the training data [33].

However, MLPs have significant limitations when it comes to time-series data. They process each input independently, without considering the order or temporal relationship between data points. MLPs cannot retain information from previous time steps, making them unable to capture long-term dependencies.

3.4 Long Short-Term Memory (LSTM)

RNNs are widely used for time-series data due to their ability to simulate memory effects through feedback loops [34] [35]. While they retain traces of previous inputs, these effects diminish over time, resulting in the long-term dependency problem. This issue arises when the output at time t relies on input from many steps earlier. RNN struggles to retain information beyond recent steps, leading to inaccuracies [36]. The issue is linked to backpropagation through time (BPTT), where gradients either vanish or explode: small gradients shrink to near zero, while large gradients grow uncontrollably, obstructing the learning of long-term dependencies [37]. Moreover, simple RNNs lack a mechanism to prioritize, reinforce, or forget past information, treating all inputs equally.

LSTMs were proposed as a solution to this problem. Their architecture is a bit more nuanced - LSTM blocks are comprised of 5 specialized neurons that help capture both short-term and long-term patterns. The central neuron with state $c(n)$ carries the activation summed from the other parts: input neuron, input gate, output gate, and forget gate as can be seen in the following update equation

$$c(n+1) = g^{\text{forget}}(n+1) \cdot c(n) + g^{\text{input}}(n+1) \cdot u(n+1),$$

and in the output $y(n)$ of a memory block

$$y(n) = g^{\text{output}}(n) \cdot c(n).$$

The input neuron with state $u(n)$ receives the input to be added to the cell state. The input gate $g^{\text{input}}(n)$ determines how much of the current input should be added to the cell state, while the forget gate $g^{\text{output}}(n)$ determines how much of that should be forgotten. The output gate $g^{\text{output}}(n)$ regulates which information from the cell state will be output, either to another memory block or for external prediction[33]. The dynamics of these weights are learned via BPTT, eliminating vanishing/exploding gradients, enabling LSTMs to capture long-term dependencies effectively [38].

The LSTM architecture suits our task well. As noted in Section 2.3.2, EDA reveals daily and seasonal trends in pollutant data, which align with short- and long-term patterns that LSTMs can capture. By adjusting input, forget, and output gates during training, the model captures complex non-linear relationships between pollutants and meteorological factors.

3.5 Evaluation Metrics

In order to understand how well a model's predictions align with actual outcomes two metrics are used: Mean Squared Error (MSE) and Mean Absolute Error (MAE) [39].

a) Mean Squared Error (MSE)

The Mean Squared Error (MSE) measures the average of the squared differences between the predicted values ($\hat{y}_i$) and actual values (y_i). MSE is particularly useful for identifying large errors as it applies a quadratic penalty to deviations, heavily penalizing predictions far from the actual values. The formula for MSE is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where y_i represents the actual target value, $\hat{y}_i$ represents the predicted target value, n is the total number of observations.

b) Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) is a simpler evaluation metric that measures the average of the absolute differences between the predicted values ($\hat{y}_i$) and the actual values (y_i). It is more robust than MSE in the presence of outliers because it treats all errors equally, without disproportionately penalizing larger deviations. The formula for MAE is:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where y_i represents the actual target value, $\hat{y}_i$ represents the predicted target value, n is the total number of observations.

As discussed in Section 2.4, outliers in our data could skew model performance if using a loss function like Mean Squared Error (MSE), which amplifies extreme values. To address this, we adopted Mean Absolute Error (MAE) for all models (Elastic Net, LSTM, and MLP), as it is more robust to outliers by treating all errors linearly. Additionally, we monitor MSE during training for a better understanding models' performance.

3.6 Model Training

In this section, we outline the steps taken to train our models, including hyperparameter tuning and the optimization of the loss functions to improve predictive performance. The search for the most optimal hyperparameters was done through grid search.

Parameter	Search space
LR	0.00001, 0.0001, 0.001
Num Units in 1st layer	128, 64
Num Units in 2nd layer	64, 32
Dropout rate	0.1, 0.2, 0.3
Batch size	64, 32

(a)

Parameter	Search space
LR	0.00001, 0.0001, 0.001
L1 penalty	0.0001, 0.001, 0.01
L2 penalty	0.0001, 0.001, 0.01
Batch size	64, 32

(b)

Figure 5: (a) Grid search space for the MLP and LSTM models and (b) Elastic Net

Elastic Net: L1 and L2 regularization penalties prevent overfitting by applying regularization to large weights. A range of [0.0001, 0.01] commonly used in the literature to explore a spectrum from minimal to moderate regularization. The learning rate, which controls the speed of weight updates, was tested across [0.00001, 0.001] that is often used to ensure stable training and avoid the risk of overshooting optimal values. Batch sizes of 64 and 32 were chosen to balance memory efficiency and gradient stability. The most optimal values found were: L1 = 0.0001, L2 = 0.001, LR = 0.001, Batch size = 32.

MLP and LSTM: For both the MLP and LSTM models, we followed the same tuning procedure, adjusting the same key hyperparameters with the same parameter ranges (Figure 5a). The learning rate (LR) was tested at range [0.00001, 0.001] to cover a broad range from conservative to moderately fast update speeds, ensuring the model could balance convergence speed with stability. The number of units in the first and second layers was varied (128 and 64 for the first layer; 64 and 32 for the second layer) to control model capacity, with these values chosen to enable the model to capture complex patterns without excessive computational load. To prevent overfitting, we tested dropout rates in range of [0.1, 0.3], as dropout helps make the model more robust by randomly deactivating neurons during training. This range is commonly effective in preventing overfitting while maintaining model flexibility. Lastly, batch sizes of 64 and 32 were chosen based on their balance between memory efficiency and gradient stability. The most optimal values for MLP and LSTM and their architecture are shown in Tables 6a and 6b respectively.

Architecture	Learning configuration
Dense, 64, ReLU	Adam, LR = 0.0001
Dropout, 0.2	Batch size = 32
Dense, 32, ReLU	Epochs = 30

(a) MLP

Architecture	Learning configuration
LSTM, 128 (*)	Adam, LR = 0.00001
Dropout, 0.2	Batch size = 32
LSTM, 64 (*)	Epochs = 30
Dropout, 0.2	

(b) LSTM, (*) act=tanh, rec=sigmoid

Figure 6: Model Architectures and Learning Configurations

3.7 Model Comparison and Selection

For comparison, the results and predictions of all three models are saved, with MAE and MSE errors calculated for each model. CO_2 emissions and energy consumption during training are measured.

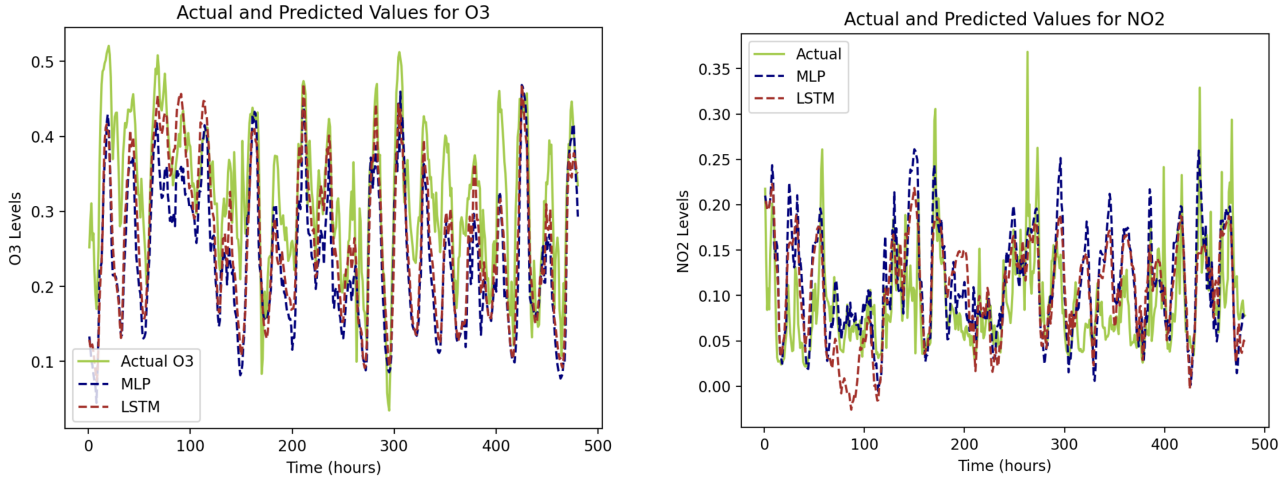
3.7.1 Training Loss and Test set performance

An interactive dashboard was developed, using the Streamlit framework. It allows us to inspect the dynamics of the models' loss during training and the accuracy of their predictions. For each target variable, MAE and MSE errors are displayed and the true versus predicted values are plotted.

The models are compared based on MSE and MAE as performance measures. Table 2 summarizes the results. The Elastic Net model can be discarded as part of further analysis as its performance does not compare to that of the MLP and LSTM models (see Table 2). The MLP model for NO_2 prediction (MAE = 0.0359) seems to slightly outperform the LSTM (MAE = 0.0414). On the other hand, the results for O_3 prediction show just the opposite - the LSTM model (MAE = 0.0664) outperforms the MLP (MAE = 0.0710). Figure 8 compares the training and validation loss across the models. The differences between the MLP and LSTM models are therefore minor, which can also be observed in Figure 7. The graphs show a forecast over a prediction horizon of 480 hours in the month of April.

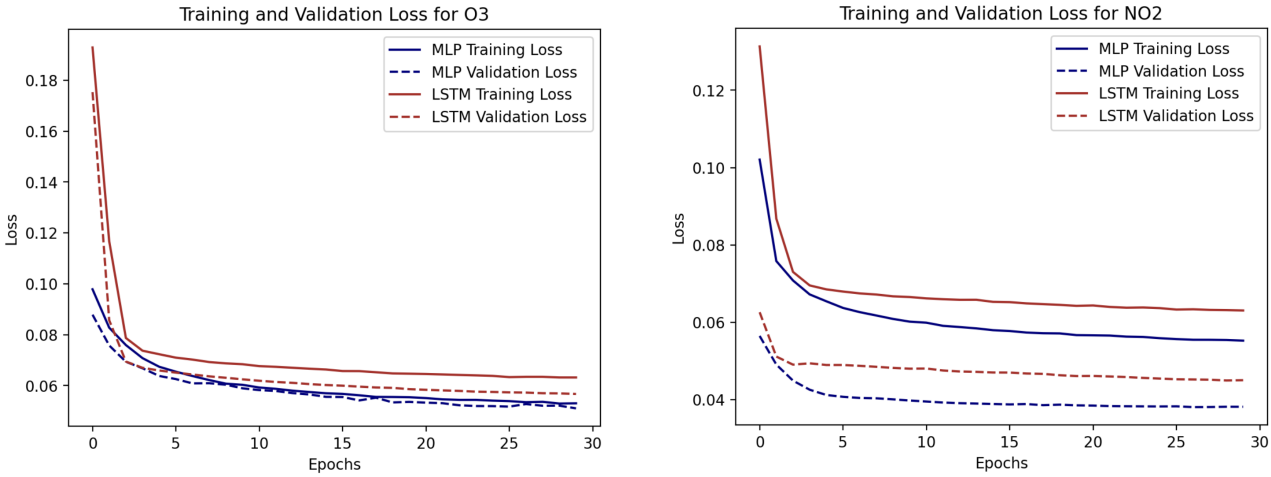
Model	Pollutant	MSE	MAE
Elastic Net	NO_2	0.0086	0.0837
	O_3	0.0163	0.1121
MLP	NO_2	0.0023	0.0359
	O_3	0.0070	0.0710
LSTM	NO_2	0.0027	0.0414
	O_3	0.0067	0.0664

Table 2: Comparing Model Performance



(a) (b)

Figure 7: Performance comparison of the MLP and LSTM models for (a) O_3 and (b) NO_2 prediction



(a) (b)

Figure 8: Training and validation loss of the MLP and LSTM models for (a) O_3 and (b) NO_2

3.7.2 Energy Consumption and CO_2 Emissions

To measure, track and compare the energy consumption and CO_2 emissions of our models, we used the EmissionsTracker from the CodeCarbon library. In evaluating the energy consumption and CO_2 emissions of our three models (Linear Regression, MLP, and LSTM), we observe notable differences. Linear Regression is the most efficient, using under 10 µWh and emitting around 2 mg of CO_2. Its efficiency stems from relying on simple arithmetic operations (see Section 3.2), which are computationally inexpensive. The MLP consumes 23 µWh and emits 8 mg of CO_2. It requires more memory and processing power, as inference depends on the number of layers and neurons—in our case, two layers with 64 and 32 neurons, respectively. The MLP uses more energy than Elastic Net due to computationally intensive operations like matrix multiplications and activation functions (see Section 3.3). The LSTM model has the highest resource usage, consuming about 140 µWh and emitting 35 mg of CO_2. Not only does the inference depend on the number of units and the number of layers, but it also depends on the number of time steps considered. Operations that LSTM involves (gate computations, recurrent connections), are the most resource-demanding (see Section 3.4).

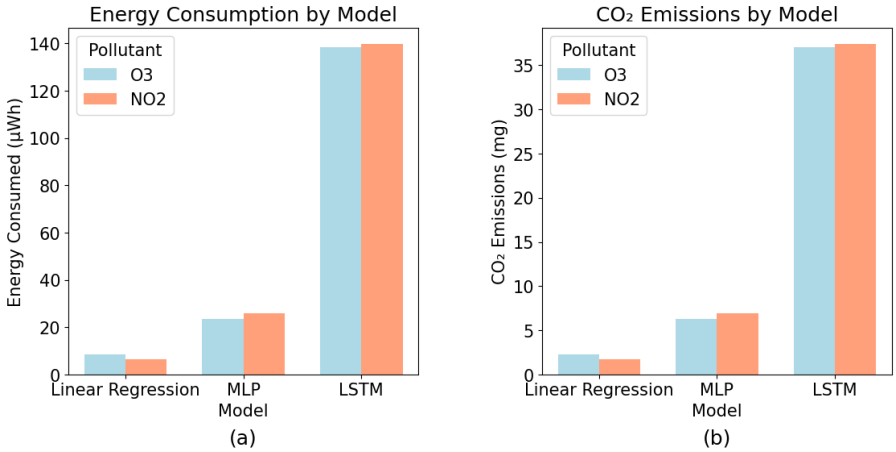


Figure 9: (a) Energy consumption of model and (b) CO_2 emissions by model.

3.7.3 Final Model Selection

For the final model selection, results from 3.7.1 and 3.7.2 are considered. While Linear Regression is resource-efficient, its predictive performance is significantly lower. The LSTM, though accurate, incurs substantial computational and environmental costs, making it less suitable for large-scale deployments. The MLP model offers an optimal balance, providing strong accuracy with considerably lower energy and emissions than the LSTM. Thus, the MLP model is the preferred choice for this application based on both performance and efficiency. The final performance and results on the testing set for the selected model are further described in Section 5.

4 Model Deployment, Monitoring, and Maintenance

4.1 Chapter Highlights

Our air pollution monitoring system automates the prediction of O_3 and NO_2 levels, utilizing HuggingFace for deployment. The processing pipeline and alert mechanisms are included. A protocol for retraining is proposed to maintain accuracy and adapt to changing environmental conditions.

4.2 Deployment Environment

HuggingFace serves as the hosting platform and infrastructure backbone for our web application, providing a stable environment for model deployment and real-time inference. The repository with our models can be found at `https://huggingface.co/senyakk/pollution_MLP/tree/main`, and the web application is accessed at `https://huggingface.co/spaces/MLINPrediction/pollution_prediction`. The application runs on 2 vCPU with 16 GB RAM provided to us for free by the host.

The application has two dashboards, the user and the admin one. The user dashboard introduces the app's purpose, which is to forecast O_3 and NO_2 levels for the next three days. It specifies the last update time, so users know when the forecast data was last refreshed. WHO-recommended safe limits for air quality are displayed as a reference. Below, graphs show predictions for O_3 and NO_2 levels, along with moving averages for each pollutant to illustrate if levels may exceed WHO guidelines. Additionally, users have the option to access a detailed table with forecasted hourly values for both pollutants over the next three days.

The Admin Page provides an overview of the MLP model in use, displaying key performance metrics like MSE and MAE for both pollutant predictions. Graphs show actual versus predicted values for O_3 and NO_2 over the month and prediction horizon selected by the user, along with training and validation loss over epochs, illustrating model convergence. These enable administrators to monitor prediction accuracy, track performance trends, and determine if re-training or tuning is needed to maintain forecast reliability and accuracy.

For automatic air pollution monitoring, a pipeline was designed and implemented (Figure 10). The pipeline's components and functionality are further detailed in Sections 4.3 and 4.4.1.

4.3 Air Pollution Monitoring and Forecasting System

Loading the model, PCA, and scalers:
The pre-trained MLP models for predicting O_3 and NO_2 levels are first loaded and cached to prevent redundant loading. Additionally, the objects needed for the feature engineering steps of the streaming

data are loaded. This includes the PCA and MinMaxScaler fitted on the features from the training data, as well as the MinMaxScalers fitted on the target O_3 and NO_2 values from the training data. The latter will be used for scaling the predictions back to their original range.

Data retrieval with Open-Meteo API:
The data used for predictions is obtained from the KNMI Forecast API every 6 hours. We utilized the Open-Meteo API Python Client with sample code provided at `https://open-meteo.com/en/docs` that retrieves the selected features in specified units and returns a Pandas dataframe. The API client is configured with caching and retry mechanisms, which lowers the risk of temporary network failures affecting the process. The data is then preprocessed, using a modified preprocessing pipeline described below, after which the feature engineering pipeline is applied.

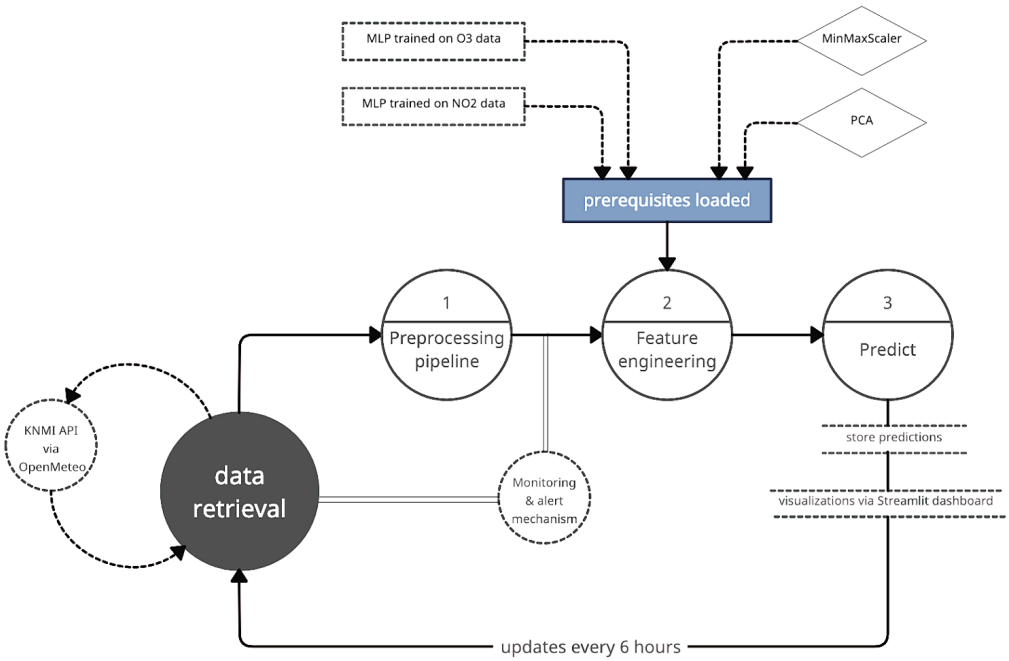


Figure 10: Overview of the air pollution prediction pipeline, deployed via HuggingFace

The preprocessing pipeline:
Since the streaming data does not match the raw training data format perfectly, the preprocessing pipeline was modified to account for these differences. Particularly, a new child class that handles the forecast meteorological dataset was defined. The new pipeline includes the following steps:

1. Features are renamed to abbreviations 'MWS', 'WG', 'T', 'TD', 'SD', and 'GR', and the columns are reordered to match the format of the training data.

2. Unit conversions are applied to the features:

   - 'MWS', 'WG': loaded in m/s, converted to 0.1 m/s,
   - 'T', 'TD': loaded in °C, converted to 0.1 °C,
   - 'SD': loaded in s, converted to 0.1 h,
   - 'GR': loaded in W/m^2, converted to J/cm^2.

The feature engineering pipeline:
The feature engineering pipeline is comparable to that applied to the training data; the features are first extracted and modified in the same way by adding rolling windows of sizes 4, 12, and 24. The data is then normalized and reduced in dimensionality using the previously loaded MinMaxScaler and PCA respectively. Finally, sliding windows of size 3 are obtained to ensure the model will predict on input data of the right dimensionality.

The predictions:
The model predictions, based on the 72 sliding windows provided as input data, are used for a 3-day forecast of pollutant levels. The predictions are saved and plotted for further analysis. The application fetches new data and updates the prediction every 6 hours and caches them until the next time.

4.4 Monitoring and Maintenance

4.4.1 Alert Mechanism

To account for failures in data loading, as well as the possibility of the loaded data being unreliable, we added several checks to the processing pipeline. In particular, the following scenarios trigger an alert mechanism.

1. Inability to fetch the data: in case the forecast API fails to return data on the first try, the pipeline includes several retries. If these are not successful, an alert is raised indicating issues with data loading itself.
2. Missing values in the loaded data: the data loaded from the forecast API is inspected for any missing values. The inspection is feature-specific, meaning an alert can indicate issues with the forecasting of a specific weather condition (e.g. wind speed) that need to be resolved. This might arise if certain sensors are down.
3. Check for extreme weather conditions: expected ranges of values are defined for each feature based on feature distributions in the training data. A value outside of this range might indicate some extreme weather condition that does not necessarily lead to poor prediction but is important to note and alert the user to explain an unusual pollutant prediction.
4. Check for out-of-training min-max feature values: a range of possible values is defined for each feature based on the minimum and maximum values of the training data (after it has been cleaned). A value that significantly deviates from this range might indicate some error in the loaded dataframe, caused by a failure in the sensors or forecasting. No matter the cause, the data must be marked as unreliable and cannot be used for prediction. An alert is shown.
5. Data distribution changes may indicate the need for model retraining or signal data unreliability. To detect shifts, new data batches are compared to a reference dataset—the training data. Since the reference data spans multiple years, including different months and seasons, comparing a few days of new data to the entire training set could mistakenly flag natural seasonal variations as shifts. To avoid this, distribution shifts are tracked for a specific time of year, comparing each batch to the corresponding monthly subset of the reference dataset. This way, larger variations that are not explained by seasonal changes would be captured.
    - Covariate shift: $p(X)$ changes, $p(Y|X)$ stays the same, a change in the distribution of feature values suggests a change in the weather patterns from those seen in the training data, detected by Kolmogorov Smirnov test.

- Label shift: p(Y) changes, p(X|Y) stays the same, the relationship between the features and pollutant levels stays the same, but there is a broader trend of pollutant levels changing (not due to shifts in the relationship between the features and pollutant).
- Concept shift: p(Y|X) changes, p(X) stays the same, the relationship between the features and pollutants is changing, performance decreases, suggests the need for retraining the model to capture new dependencies.

6. Other notable changes: the features included in the forecast API might be modified or removed. Modifications can refer to a lower-level change in labels or units, both of which can be fixed and incorporated into the processing pipeline after receiving an alert. This is more difficult to solve for a higher-level change, such as the difference in measurement / forecast style (e.g. moving from average hourly measurements to average measurements over the last 10 minutes of the hour). These need to be carefully analysed after receiving an alert. In the case of a permanent feature removal, there is an option of retraining the model without the feature or finding a new source.

4.4.2 Retrain Protocol Proposal

To ensure predictive accuracy, the deployed model may require periodic retraining. We propose a retraining protocol with criteria for determining when and how retraining should occur.

- Points 5 and 6 from Section 4.4.1 are particularly important for deciding when the model needs retraining. The data is regularly monitored for significant distribution shifts; if the new data distribution is not comparable to that of the old training data, the model performance will start decreasing. Specifically, re-training is triggered when these shifts reach pre-defined thresholds, such as a divergence of more than 10% from historical feature means.
- If the alert mechanism detects changes in the Forecast API, such as a permanent feature removal or alteration in measurement techniques, re-training is triggered to align the model with the new data structure.
- To maintain baseline accuracy, we suggest annual re-training even if no major alerts are triggered. This ensures the model incorporates recent data trends and seasonal variations, which may impact pollution levels over time.
- A consistent decline in model performance also serves as a re-training indicator. Regular performance checks monitor metrics like Mean Absolute Error (MAE) and alignment with expected pollutant trends. If the model's performance decreases by 10% or if predicted patterns deviate notably from observed pollutant variations, re-training is triggered to restore accuracy.

When re-training is triggered, it will be determined whether to fine-tune the model or re-train from scratch. Fine-tuning is chosen for moderate data shifts (within 10-20% deviation from the historical mean), using only recent data from the past 6 months to retain prior knowledge. Full re-training is applied if shifts exceed 20% or if API changes alter data structure (e.g., new feature definitions). In this case, all available data is used, with a fresh split into training, validation, and test sets to re-establish the model baseline.

The re-training pipeline, as described in Sections 2.5 and 2.6, begins with data collection, preprocessing, feature engineering, and splitting, followed by re-training while tracking metrics like MAE and trend accuracy. Models that meet performance thresholds proceed to deployment, replacing the

previous version. Post-deployment monitoring detects shifts or performance drops, with alerts in place. Each re-training instance is documented, noting the trigger, training parameters, and whether fine-tuning or full re-training was applied. Version control enables rollback if necessary.

5 Results

5.1 Chapter Highlights

The results of the selected MLP models for both O_3 and NO_2 prediction are described and discussed. The challenges and limitations of our model are explained, and some improvements are suggested.

5.2 Performance Evaluation

The final models are the MLPs (for NO_2 and O_3) with the architecture and parameters described in Section 3.6 and 3.7. The two models were evaluated on the test set that includes data for the following months in 2021: April, July, October, and December. This provides insight into how well the models handle seasonal variability and their robustness under changing environmental conditions.

Table 3 shows the final performance metrics for the MLP models (MAE and MSE) for each pollutant. These represent prediction accuracy within a normalized range, as they are calculated based on the scaled values of pollutant levels. This allows for a direct comparison between the performance of the two models, as it prevents the metrics from being influenced by the absolute magnitude of pollutant concentrations. The MAE values of 0.0453 for NO_2 and 0.0613 for O_3 suggest that, on average, the NO_2 predictions are more accurate than those for O_3. The same pattern is observed in MSE, suggesting that the model for NO_2 has fewer larger deviations in its predictions. These differences can partially be attributed to the inherently different characteristics in the pollutant trends and underlying causes for variations in their concentrations. We suggest further research should focus on the differences in NO_2 and O_3 interactions with meteorological variables, as well as certain human-based factors (like traffic and industrial emissions) and consider such findings when building and training the different models.

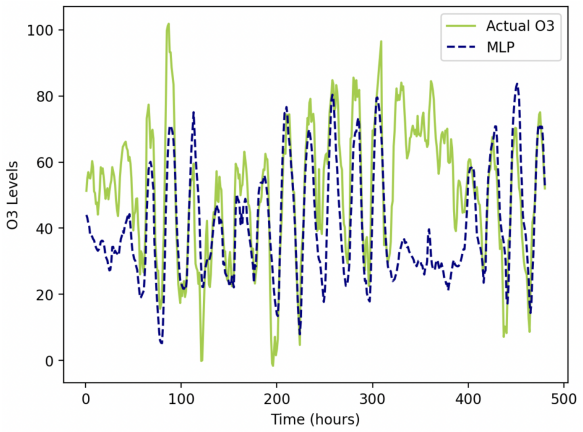
Pollutant/Model	MSE	MAE
NO_2	0.0041	0.0453
O_3	0.0063	0.0613

Table 3: Performance Metrics for the MLP models

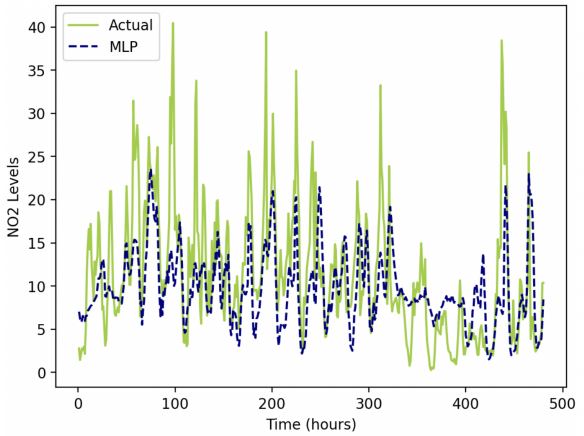
More detailed results of our model along with visualizations of predictions for the mentioned months can be seen on the Admin dashboard in our app. A part is shown in Figure 11, which compares the model predictions with the actual values over the two pollutants and two months (July and December).

Comparing the predictions made for July (a and b) with those made for December (c and d) suggests some seasonal variation in accuracy. In July, the model seems to align more closely with the observed values for both pollutants than it does in December, capturing fluctuations and (especially) peaks more accurately. This might stem from more stable environmental patterns in the training data that more closely match the July trends, or simply a greater prevalence of such data (which will cause the model to generalize better to similar patterns).

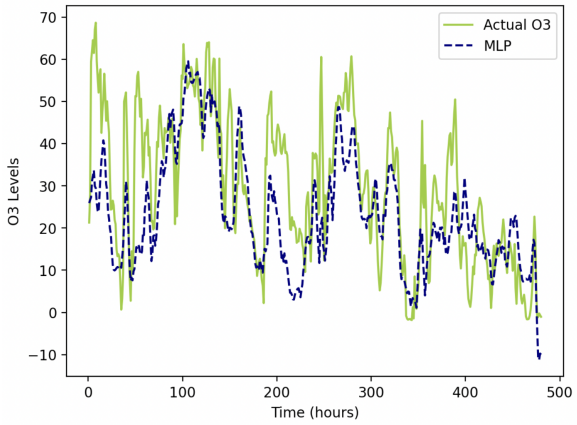
Since our training data *did* equally represent all seasons and months (see 2.6.2), such an approach might not be sufficient, and so we suggest incorporating additional seasonal features to enhance the model's performance throughout the whole year.



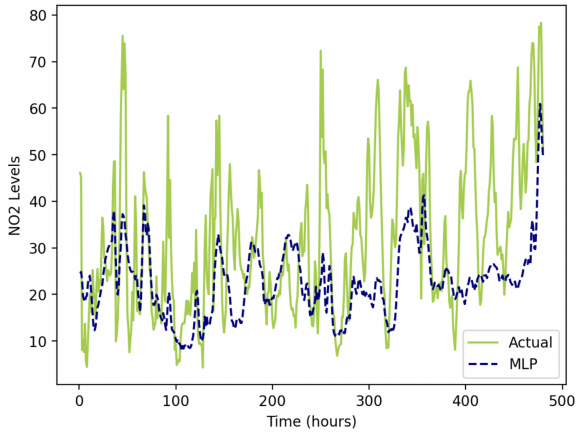
(a) O_3, July



(b) NO_2, July



(c) O_3, December



(d) NO_2, December

Figure 11: Comparison of Model Performance over Pollutant and Month

5.3 Challenges and Limitations

A key challenge in using an MLP model for air pollution forecasting is its lack of interpretability. As a "black-box" model, it's difficult to discern how individual features contribute to predictions [40]. In air pollution forecasting, interpretability is important for validating predictions, especially when triggering alerts or informing decisions based on pollutant levels. Without a clear understanding of the driving factors, stakeholders may struggle to trust the model's outputs. Although post-hoc interpretability tools (e.g., SHAP, LIME) provide some insights [41], they add complexity and may not fully convey the MLP's reasoning, particularly for non-technical users.

In the future, it might be necessary to introduce additional predictive features, which in turn may require a deeper MLP architecture. This, however, introduces challenges with vanishing and exploding gradients during backpropagation, where gradients may shrink, slowing learning, or grow excessively, causing instability (see Section 3.4). These issues hinder convergence and make it difficult to capture complex interactions.

6 Conclusion

We have developed a system based on the public available data, tailored to meet social goals. Throughout the project we have gained many insights, such as learning how to properly preprocess time series data as well as we have shown that LSTMs shouldn't always be preferred choice for time-forecasting as these are computationally expensive and long-to-train models, and it can happen that their performance could be not better compared to the smaller models trained on this task. The resulting application that utilises our system is the first iteration of the model in production.

6.1 Future Work and Recommendations

To maintain the accuracy and adaptability of our air quality forecasting application, it is essential to anticipate both immediate and future developments. In the short term, implementing a protocol (see Section 4.4.2) to handle distribution shifts between new data and the original training data would enhance the application's adaptability. This protocol could store incoming data and trigger re-training when sufficient data is collected and distribution shifts are detected, maintaining model relevance. Automating this process would keep forecasts accurate and responsive to real-world changes over time.

In the long term, exploring interpretable machine learning models, such as decision trees, random forests, or explainable boosting machines (EBMs), could improve future development. These models provide transparency in understanding how features influence predictions, helping users and stakeholders trust the outputs. Additionally, expanding the system to include data from additional monitoring stations would improve spatial accuracy, allowing the model to reflect diverse environmental conditions. This broader data range would offer more reliable air quality insights across locations, supporting local governments and agencies in combating pollution. Finally, developing a mobile-friendly app would further extend the service's reach, allowing users to access real-time air quality forecasts, alerts, and pollution trends on their phones. A globally accessible app would empower individuals worldwide to make informed health and lifestyle decisions based on accurate air quality information.

6.2 Broader Impacts

Creating forecasting system for NO_2 and O_3 benefits for public health, well-being, and environmental awareness. By informing people of current and projected pollution levels, it enables vulnerable groups such as children, the elderly, and people with respiratory conditions to make safer choices about outdoor activities. Additionally, the project promotes environmental awareness about pollution patterns, encouraging actions to improve air quality. For example, government can reduce vehicle access during predicted high-pollution days which could help curb pollutant spikes. Furthermore, insights from historical data can guide the placement of green spaces, pedestrian zones, and bike lanes, contributing to long-term air quality improvement.

While our project provides forecasts of air pollution levels, it is important to recognize that all forecasts inherently carry a degree of uncertainty and cannot guarantee accurate future values. This limitation is exacerbated in our case because we rely on meteorological forecasts as our primary input. Since weather forecasts themselves are subject to variability, the accuracy of our air quality model can be further compromised by changes in weather conditions that are difficult to accurately predict. Therefore, while our model provides useful guidance, users should interpret pollutant forecasts as estimates that may change based on real-time atmospheric changes.

Bibliography

[1] R. J. Estes and M. J. Sirgy, "Global advances in quality of life and well-being: Past, present, and future," *Social Indicators Research*, vol. 141, pp. 1137–1164, 2019.

[2] X. Zhang, L. Han, H. Wei, X. Tan, W. Zhou, W. Li, and Y. Qian, "Linking urbanization and air quality together: A review and a perspective on the future sustainable urban development," *Journal of Cleaner Production*, vol. 346, p. 130988, 2022.

[3] R. Fuller, P. J. Landrigan, K. Balakrishnan, G. Bathan, S. Bose-O'Reilly, M. Brauer, J. Caravanos, T. Chiles, A. Cohen, L. Corra, *et al.*, "Pollution and health: a progress update," *The Lancet Planetary Health*, vol. 6, no. 6, pp. e535–e547, 2022.

[4] "World Health Organisation." `https://shorturl.at/hdAOL`. Accessed: 2024-09-20.

[5] L. van der Net, N. Staats, P. Coenen, J. Rienstra, P. Zijlema, E. Arets, K. Baas, S. van Baren, R. Dröge, K. Geertjes, *et al.*, "Greenhouse gas emissions in the netherlands 1990–2022. national inventory report 2024," 2024.

[6] F. Menezes and G. M. Popowicz, "Acid rain and flue gas: Quantum chemical hydrolysis of no2," *ChemPhysChem*, vol. 23, no. 21, p. e202200395, 2022.

[7] S. Sillman, "Overview: Tropospheric ozone, smog and ozone-nox-voc sensitivity," *Treatise on Geochemistry*, 2003.

[8] D. Nuvolone, D. Petri, and F. Voller, "The effects of ozone on human health," *Environmental Science and Pollution Research*, vol. 25, pp. 8074–8088, 2018.

[9] E. E. Agency, "Europe's air quality status 2024," Jun 2024.

[10] "World Health Organisation." `https://www.who.int/news-room/feature-stories/detail/what-are-the-who-air-quality-guidelines`. Accessed: 2024-09-28.

[11] S. V. Krupa and W. J. Manning, "Atmospheric ozone: formation and effects on vegetation," *Environmental pollution*, vol. 50, no. 1-2, pp. 101–137, 1988.

[12] J. Beck, M. Krzyzanowski, B. Koffi, E. Commission, and E. E. Agency, *Tropospheric Ozone in the European Union: The Consolidated Report*. No. v. 856 in EUi collection, Office for Official Publications of the European Communities, 1999.

[13] X. Gong, S. Hong, and D. A. Jaffe, "Ozone in china: Spatial distribution and leading meteorological factors controlling o3 in 16 chinese cities," *Aerosol and Air Quality Research*, vol. 18, no. 9, p. 2287–2300, 2018.

[14] F. Song, J. Young Shin, R. Jusino-Atresino, and Y. Gao, "Relationships among the springtime ground–level nox, o3 and no3 in the vicinity of highways in the us east coast," *Atmospheric Pollution Research*, vol. 2, no. 3, pp. 374–383, 2011.

[15] A. Yafouz, N. AlDahoul, A. H. Birima, A. N. Ahmed, M. Sherif, A. Sefelnasr, M. F. Allawi, and A. Elshafie, "Comprehensive comparison of various machine learning algorithms for short-term ozone concentration prediction," *Alexandria Engineering Journal*, vol. 61, no. 6, pp. 4607–4622, 2022.

[16] J. Du, F. Qiao, P. Lu, and L. Yu, "Forecasting ground-level ozone concentration levels using machine learning," *Resources, Conservation and Recycling*, vol. 184, p. 106380, 2022.

[17] B. G. Brian S. Freeman, Graham Taylor and J. Thé, "Forecasting air quality time series using deep learning," *Journal of the Air & Waste Management Association*, vol. 68, no. 8, pp. 866–886, 2018. PMID: 29652217.

[18] "KNMI Data Platform." `https://dataplatform.knmi.nl/`. Accessed: 2024-09-17.

[19] "Luchtmeetnet dataset." `https://data.rivm.nl/data/luchtmeetnet/`. Accessed: 2024-09-17.

[20] "Luchtmeetnet." `https://www.luchtmeetnet.nl/`. Accessed: 2024-09-17.

[21] "Weerstations." `https://daggegevens.knmi.nl/klimatologie/uurgegevens`. Accessed: 2024-09-17.

[22] C. M. Kendrick, P. Koonce, and L. A. George, "Diurnal and seasonal variations of no, no2 and pm2. 5 mass as a function of traffic volumes alongside an urban arterial," *Atmospheric Environment*, vol. 122, pp. 133–141, 2015.

[23] Y. Liu, P. Qiu, C. Li, X. Li, W. Ma, S. Yin, Q. Yu, J. Li, and X. Liu, "Evolution and variations of atmospheric vocs and o3 photochemistry during a summer o3 event in a county-level city, southern china," *Atmospheric Environment*, vol. 272, p. 118942, 2022.

[24] F. Ros and R. Riad, *Feature selection*, pp. 27–44. Cham: Springer Nature Switzerland, 2024.

[25] R. Van Malderen, H. De Backer, A. Delcloo, and M. Allaart, "Identifying the origin of anomalous high tropospheric ozone in the ozonesonde data at uccle by comparison with nearby de bilt," *Atmosphere-Ocean*, vol. 53, no. 1, pp. 102–116, 2015.

[26] Y. Yin and H. Liu, "Air quality index prediction based on three-stage feature engineering, model matching, and optimized ensemble," *Air Quality, Atmosphere & Health*, vol. 16, no. 9, pp. 1871–1890, 2023.

[27] M. Fayad, M.-Y. Hachani, A. Mostefaoui, M. A. Merzoug, I. Lajoie, and R. Yahiaoui, "Impact of feature normalization on machine learning-based human fall detection," in *International Conference on Management of Digital*, pp. 147–161, Springer, 2023.

[28] J. Brownlee, "How to use data scaling improve deep learning model stability and performance," Aug 2020.

[29] A. S. Pillai, "A deep learning approach for used car price prediction," *Journal of Science & Technology*, vol. 3, no. 3, pp. 31–50, 2022.

[30] G. James, D. Witten, T. Hastie, R. Tibshirani, and J. Taylor, "Linear regression," in *An introduction to statistical learning: With applications in python*, pp. 69–134, Springer, 2023.

[31] L. Melkumova and S. Y. Shatskikh, "Comparing ridge and lasso estimators for data analysis," *Procedia engineering*, vol. 201, pp. 746–755, 2017.

[32] J. O. Ogutu, T. Schulz-Streeck, and H.-P. Piepho, "Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions," in *BMC proceedings*, vol. 6, pp. 1–6, Springer, 2012.

[33] H. Jaeger, "Neural networks.."

[34] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014.

[35] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649, Ieee, 2013.

[36] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[37] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 28 of *Proceedings of Machine Learning Research*, (Atlanta, Georgia, USA), pp. 1310–1318, PMLR, 17–19 Jun 2013.

[38] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[39] A. Botchkarev, "A new typology design of performance metrics to measure errors in machine learning regression algorithms," *Interdisciplinary Journal of Information, Knowledge, and Management*, vol. 14, p. 045–076, 2019.

[40] J. Hussain, "Deep learning black box problem," 2019.

[41] A. M. Salih, Z. Raisi-Estabragh, I. B. Galazzo, P. Radeva, S. E. Petersen, K. Lekadir, and G. Menegaz, "A perspective on explainable artificial intelligence methods: Shap and lime," *Advanced Intelligent Systems*, p. 2400304, 2024.