Machine Learning Approaches to Ethical Analysis of Statistics

N. CAMILLERI, N. PORTELLI, and O. GRECH
Department of Artificial Intelligence, University of Malta. (e-mail: {nathan.camilleri.19, nathan.portelli.19, oleg.grech.19}@um.edu.mt)

ABSTRACT In the age of data-driven decision-making, the prominence of Machine Learning (ML) algorithms has soured across various domains, providing unparalleled insights and predictive capabilities. This study explores the effectiveness of three distinct ML techniques, being the Decision Tree, Random Forest and Linear Regression techniques, in forecasting population values. Utilising data from the National Statistics Office (NSO) of Malta, comprising annual population changes across different Local Administrative Units (LAU) from 2005 to 2020, we delve into the unique demographic context the data presents, and significant demographic challenges faced by the country. Traditional statistical methods often fall short in capturing the intricate and non-linear nature of demographic patterns. In response, this research aims to address these limitations by evaluating more advanced techniques for robust and accurate population predictions. By evaluating these algorithms, this research contributes to the advancement of predictive modelling in demography. Through a meticulous examination of metric analysis and visualisations, our findings affirm the superiority of the Random Forest model in this field. This research contributes to the advancement of predictive modelling in demography, overcoming the constraints of conventional forecasting methods and provides valuable insights for policymakers, urban planners, and researchers, fostering informed decision-making processes. This research establishes a foundation for refining predictive modelling strategies not only in demography but also in critical sectors such as work and education. By evaluating these ML approaches, our work seeks to enhance precision and reliability in population predictions, paving the way for a more informed and data-driven future

INDEX TERMS decision tree, random forest, linear regression, machine learning, population prediction

I. INTRODUCTION

In the era of data-driven decision-making, the strategic application of Machine Learning (ML) algorithms has become increasingly prevalent across various domains, offering unprecedented insights and predictive capabilities. One such area of interest is the prediction of population values, a task with far-reaching implications for resource allocation, urban planning, and the development of effective social policies. The island nation of Malta, nestled in the heart of the Mediterranean, provides a unique and compelling context for investigating the efficacy of ML algorithms in forecasting population trends. Malta is in the midst of experiencing a remarkable demographic shift, marked by fluctuations in life expectancy and fertility rates [1]. These changes have posed challenges in managing population growth and distribution effectively [2], placing Malta among EU member states facing some of the most pressing demographic issues, with immigration also being identified as a major concern [3]. Despite its status as the most densely populated country in the EU and one of the most densely populated globally, boasting around 1,649 inhabitants per square kilometre [4], Malta maintains a diverse and dynamic population, featuring a high proportion of foreign residents and migrants [5]. These characteristics pose both challenges and opportunities for population forecasting, making Malta an intriguing case study for applying advanced ML techniques, which may significantly enhance the accuracy and reliability of these predictions, contributing valuable insights for the continued development of the country.

A. PROBLEM DEFINITION

Accurate population predictions are indispensable for informed decision-making in policy, urban development, and resource allocation. Despite the value of traditional statistical methods, their limitations become apparent when attempting to capture the intricate and non-linear nature of demographic patterns. Classical small-area estimation methods, for example, require fitting a complex statistical regression model to survey responses using auxiliary administrative data as predictors. Subsequently, the absent responses are predicted and aggregated the desired geographical level. Unfortunately, these methods face limitations such as data quality issues, model misspecification, and computational complexity [6].

In the specific context of Malta, a country characterised by limited land area and increasing urbanisation, effectively managing population growth and distribution poses significant challenges. These challenges necessitate a shift towards exploring advanced ML techniques for more robust and accurate population predictions. Machine Learning (ML), as a branch of Artificial Intelligence (AI), involves learning from data to make predictions or decisions based on acquired patterns [7]. With the ability to handle large and heterogeneous datasets, uncover hidden non-linear associations, and adapt to changing environments, ML algorithms emerge as a powerful tool for population forecasting [8]. This becomes particularly relevant in situations where conventional methods prove inadequate or impractical.

B. MOTIVATION

The motivation behind this research lies in addressing the limitations of conventional demographic forecasting methods and exploring the potential of ML to improve the precision and reliability of population predictions. While ML has found applications in various applications such as predicting health related outcomes in population health contexts, including forecasting disease risk, mortality, and health-related outcomes [9], there is a notable gap in research when it comes to applying ML to predict population values, particularly in small geographical areas like Malta. By leveraging the power of algorithms such as Decision Tree, Random Forest, and Linear Regression, we aim to fill this gap and contribute to the advancement of predictive modelling in demography. The outcomes of this study have the potential to inform policymakers, urban planners, and researchers, facilitating more informed and effective decision-making processes.

C. AIMS AND OBJECTIVES

This research aims to assess and compare the efficacy of three distinct ML algorithms, Decision Tree, Random Forest, and Linear Regression, in predicting population values for Malta. Specific objectives include:

- Prepare the dataset by conducting data cleaning, transformation, and feature engineering, ensuring it is ready for utilisation in ML algorithms.
- Implement three different ML algorithms for modelling and predicting data within the selected dataset.
- Conduct experiments to compare the results and insights derived from each model, utilising relevant metrics for comprehensive evaluation.

Achieving these objectives will not only contribute valuable insights for population predictions in Malta but will also play a pivotal role in advancing the application of ML techniques to demographic forecasting on a broader scale.

D. PROPOSED SOLUTION

The proposed solution involves employing a combination of Decision Tree, Random Forest, and Linear Regression algorithms to build predictive models tailored to Malta's demographic context. Decision Trees, a type of supervised learning algorithm, partition the data into smaller subsets based on a sequence of rules, resulting in a tree-like structure [10]. Random Forests serve as an ensemble method, amalgamating multiple Decision Trees and consolidating their predictions to diminish variance and enhance model accuracy [11]. Meanwhile, Linear Regression is a straightforward yet widely adopted algorithm which captures the relationship between a dependent and one or more independent variables using a linear equation [12]. Leveraging the strengths of these diverse algorithms, our goal is to craft precise and interpretable models that offer valuable insights into the population dynamics of Malta. Through comprehensive evaluation and comparison, this research endeavours to deepen our understanding of the applicability of ML in demographic forecasting, paving the way for more informed decision-making in the realm of population management.

E. DATASET EXPLANATION

The dataset employed in this study titled 'Total Population by region, district and locality' was obtained as a CSV file from Malta's National Statistics Office (NSO) website[1]. Comprising 3696 records, the dataset provides information on the annual population changes across the different Local Administrative Units (LAU) in Malta and Gozo from 2005 to 2020, categorised by gender into male, female, or total population. The selection of this dataset source was driven by its comprehensive coverage and relevance to the research objectives. An explanation of the columns within the dataset can be found in Table 1.

Column Name	Description
DATAFLOW	Name of the dataset.
CSE_REG_DIST_LOC	Region District Locality, making up Malta's Local Administrative Units (LAU), providing separate data for the total regions, districts within the regions, and localities within the districts.
CSE_SEX	Division of data based on sex, either male, female, or total.
CSE_FREQ	Frequency of data gathering, listed as annual for all records.
TIME_PERIOD	The year pertaining to the record, with "-A1" included to show the time period when the data was extracted.
OBS_VALUE	The number of persons within the LAU and gender subsection within the particular time period.
CSE_OBS_STATUS	Observation Status (This column was empty).
CSE_OBS_NOTE	Observation Note (This column was empty).

TABLE 1. Description of Dataset Columns

The dataset served as the foundation for training and evaluating three models capable of forecasting population changes, based on both gender and LAU.

[1]NSO Statistical Database: https://statdb.nso.gov.mt/.

F. TECHNIQUES

In our population prediction project, we strategically selected three prominent ML techniques, guided by their effectiveness and relevance in the context of demographic forecasting. These techniques, namely:

- Decision Tree: Constructing predictive models by associating observations with target values, Decision Trees offer simplicity and interpretability. They are widely utilised in ML and decision analysis.
- Random Forest: Leveraging the output of multiple Decision Trees, Random Forest reduces overfitting risk and delivers accurate predictions. This approach has been supported by literature findings in the context of population prediction.
- Linear Regression: A high-performance algorithm predicting variable values based on linear equations, Linear Regression has demonstrated effectiveness in previous population prediction studies [13, 14].

The rationale behind selecting these techniques is further elucidated in the subsequent sections. Additionally, various evaluation metrics and methods were devised to comprehensively compare the performance of these chosen ML techniques, detailed in Section II-E.

II. BACKGROUND

In the pursuit of accurate population prediction, this chapter offers a comprehensive overview of the key techniques and measurements central to our project. To lay a solid foundation, we begin with an in-depth examination of crucial preprocessing steps in ML, being rescaling and normalisation, followed by an exploration of cross-validation, dimensionality reduction, feature selection methods, and concluding with a discussion on quantitative measurements. The latter part of the chapter focuses on the specific ML techniques chosen for our population prediction project, being Decision Trees, Random Forest, and Linear Regression. A comprehensive exposition of each technique is provided, accentuating their individual strengths and potential limitations. Additionally, relevant insights from academic literature and related work are incorporated to enrich the understanding of their applications and implications in the context of demographic forecasting. The background chapter thus sets the stage for a thorough exploration of our methodology, presenting the rationale behind our technique selection and setting the stage for the subsequent quantitative assessments undertaken.

A. RESCALING AND NORMALISATION

Rescaling and normalisation are crucial preprocessing steps in ML, especially when dealing with diverse datasets and various algorithms [15]. Rescaling transforms numerical features to a specific range, often between 0 and 1 [15], ensuring proportional contributions during the model training process, preventing certain features from dominating due to their original scale [16]. Conversely, normalisation standardises the scales of features to adhere to a standard distribution, typically with a mean of 0 and a standard deviation of 1 [17]. This practice expedites convergence during the training phase by averting the domination of large-scale features, thus enhancing the learning process [18]. Additionally, it mitigates algorithm sensitivity to varying input scales, culminating in a model that is both more stable and accurate [15].

B. DIMENSIONALITY REDUCTION AND FEATURE SELECTION

In many real-world scenarios, datasets contain a large number of features, some of which may be redundant or irrelevant. Dimensionality reduction and feature selection methods aim to address several challenges. High-dimensional datasets can be computationally expensive and may lead to overfitting [19], prompting the need for techniques like Principal Component Analysis (PCA) to retain essential information while reducing the number of features [20]. Additionally, feature selection plays a crucial role in enhancing model interpretability by identifying the most important features [21]. Furthermore, these methods contribute to improved model performance, particularly in scenarios where the dataset includes noise or irrelevant information, as they facilitate the removal of such extraneous features [22].

C. HYPERPARAMETER TUNING

Hyperparameter tuning is a crucial aspect of ML, involving the systematic adjustment of external configuration settings to optimise a model's performance [23]. These settings, known as hyperparameters, govern aspects such as learning rates and model complexity. The tuning process aims to strike a balance, avoiding overfitting or underfitting and enhancing a model's adaptability to different datasets. Efficient tuning not only improves computational efficiency but also contributes to a model's robustness, making it resilient to variations in data and conditions. Additionally, tuning can enhance interpretability by finding the right balance between model complexity and transparency. Various methods, such as grid search, exist for hyperparameter tuning, each with their own strengths depending on the problem and available resources [24]. The grid search method allows the user to train the model on any possible combination of hyperparameters by setting up a matrix of hyperparameters [25]. Such an approach is typically used when there is a small number of hyperparameters that need to be adjusted.

D. CROSS-VALIDATION

Cross-validation stands as a crucial technique for evaluating the performance of ML models and addressing concerns related to overfitting or underfitting [26]. This method entails dividing the dataset into multiple subsets, training the model on one subset, and assessing its performance on another. This cyclic process is reiterated several times, with the results subsequently averaged [27]. The utility of cross-validation lies in its ability to enhance model assessment, providing a more robust estimate of performance by employing distinct subsets for both training and testing [28]. Furthermore, it contributes to assessing a model's generalisation capability by testing it on various subsets of the data, offering valuable insights into its performance on unseen data [29]. An example of a cross-validation method is K-fold, which facilitates model assessment by employing distinct subsets of the data for both training and testing, contributing to a more reliable performance estimate. Furthermore, it enhances the assessment of a model's generalisation capability by testing it on various subsets of the data, providing valuable insights into its performance on unseen data [29]."

E. QUANTITATIVE MEASUREMENTS

Quantitative measurements are essential for objectively assessing and comparing the performance of ML models in the context of population prediction. They provide a nuanced understanding of the model's predictive capabilities, shedding light on its strengths and potential limitations. These metrics collectively contribute to a comprehensive evaluation, enabling informed decision-making and guiding further model refinement. The metrics employed for this purpose, which were also utilised to assess and compare our results, include the following:

1) Mean Absolute Error

The Mean Absolute Error (MAE) stands as a prevalent metric for assessing the accuracy of regression models. Computed as the average of the absolute differences between predicted and actual values, the MAE offers a straightforward measure of the magnitude of prediction errors. By quantifying the average accuracy of a model's predictions, MAE provides a clear understanding of the model's predictive power, making it a crucial metric in evaluating regression model performance [30]. The formula for MAE is:

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

where:

- n is the number of observations,
- y_i is the actual (observed) value for the i-th observation,
- $\hat{y}_i$ is the predicted value for the i-th observation,

2) Median Absolute Error

The Median Absolute Error (MedAE) is a measure of the dispersion of data points in a dataset. It is calculated as the median of the absolute differences between actual and predicted values. MedAE is less sensitive to outliers compared to mean-based metrics and provides a robust measure of central tendency in prediction errors. The formula for Median Absolute Error is given by:

$$\text{MedAE} = \text{median}\left(\sum_{i=1}^{n}(y_i - \hat{y}_i)\right) \tag{2}$$

where:

- n is the number of observations,
- y_i is the actual value of the dependent variable for the i-th observation,
- $\hat{y}_i$ is the predicted value of the dependent variable for the i-th observation,
- median$(\ldots)$ calculates the median of the set of absolute differences.

3) Maximum Absolute Error

The Median Absolute Error (MaxAE) represents the maximum absolute difference between predicted and actual values, identifying the largest individual prediction error. This metric highlights the extreme cases where the model exhibits significant deviations from actual values. The equation for calculating the Max Error is:

$$\text{MaxAE} = \max\left(\sum_{i=1}^{n}(y_i - \hat{y}_i)\right) \tag{3}$$

where:

- n is the number of observations,
- y_i is the actual value of the dependent variable for the i-th observation,
- $\hat{y}_i$ is the predicted value of the dependent variable for the i-th observation,
- median$(\ldots)$ calculates the max of the set of absolute differences.

4) Mean Squared Error

The Mean Squared Error (MSE) is a commonly used metric to measure the average squared difference between the actual and predicted values in a regression problem. It provides a comprehensive evaluation of prediction accuracy, with larger errors contributing more significantly to the overall score. This metric is valuable for identifying instances where the model exhibits substantial deviations from the actual values [30]. The equation for Mean Squared Error is as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4}$$

where:

- n is the number of data points.
- y_i is the actual (observed) value for the i-th data point.
- $\hat{y}_i$ is the predicted value for the i-th data point.

5) Root Mean Squared Error

The Root Mean Squared Error (RMSE) is a variation of the MSE that is commonly used in regression analysis to measure the average magnitude of the error between predicted and actual values in a more interpretable scale. It aids in easier interpretation and comparison of model performance, providing a measure in the same unit as the target variable [31]. The equation for RMSE is given by:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{5}$$

where:

- n is the number of data points.
- y_i is the actual (observed) value for the i-th data point.
- $\hat{y}_i$ is the predicted value for the i-th data point.

6) R-Squared

The coefficient of determination, R^2, quantifies the proportion of variance that the model explains in the dependent variable. A higher R^2 value signifies a better fit of the model to the data, making this metric pivotal for assessing the overall explanatory power of our models [32]. The R^2 formula is given by:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{6}$$

where:

- n is the number of observations,
- y_i is the actual value of the dependent variable for the i-th observation,
- $\hat{y}_i$ is the predicted value of the dependent variable for the i-th observation,
- $\bar{y}$ is the mean of the observed values of the dependent variable.
- $\sum_{i=1}^{n}$ represents the summation over all observations.

7) Explained Variance

The Explained Variance (EV) is a statistical measure that assesses the extent to which a statistical model explains or accounts for the variability in a dataset. A higher EV indicates a better fit of the model to the data, providing a comprehensive measure of its explanatory capabilities. The formula for explained variance is given by:

$$\text{EV} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{7}$$

where:

- n is the number of observations,
- y_i is the actual value of the dependent variable for the i-th observation,
- $\hat{y}_i$ is the predicted value of the dependent variable for the i-th observation,
- $\bar{y}$ is the mean of the observed values of the dependent variable.

8) Average Execution Time

The time an algorithm takes to execute offers important information about the effectiveness and computing requirements of our solutions as they are put into practice. The computational effort and resource utilisation can be determined with high precision by measuring the elapsed time during algorithm execution. This measure functions as a critical performance indicator, assisting in evaluating the responsiveness and scalability of the method. A careful analysis of the time metric guarantees a complete assessment of the practicality of our methods and helps to optimise the system.

In summary, the selection and interpretation of quantitative measurements are paramount in delineating the efficacy of our ML models, forming the bedrock for informed conclusions and future research directions.

F. VISUALISATION TECHNIQUES

Effective visualisation techniques are crucial for gaining insight into the performance of ML models, particularly in the realm of population prediction. While quantitative metrics offer a numerical understanding of model performance, visualisations provide a more intuitive and interpretative perspective. Such visualisation techniques serve as valuable complements to quantitative metrics, offering a holistic view that aids in identifying patterns, trends, and potential areas for improvement. The visualisation methods chosen for this purpose, which were also utilised to assess and compare the three models, include the following:

1) Scatter Plot

A scatter plot is a two-dimensional data visualisation where values for two different variables are represented by individual data points. Every point on the figure represents a single dataset observation, with the values of the two variables indicated by their vertical and horizontal positions, respectively. When attempting to discern connections or trends between two continuous data, scatter plots come in handy [33].

2) Line Plot

A line plot displays data points connected by straight line segments. It is frequently used to show patterns or trends over an extended period of time. In time-series analysis, line charts are a common option because they are particularly useful for displaying how data evolves over time. Every data point is symbolised by a marker, and trends are indicated by connecting these points with lines [34].

3) Residual Plot

A residual plot graphically illustrates the differences between the actual and predicted values in a regression analysis. Typically, the vertical axis depicts residuals, while the horizontal axis represents the independent variable. These plots offer valuable insights into the goodness of fit in a regression model. Ideally, residuals should exhibit a random dispersion around zero, indicating that the model effectively captures underlying patterns in the data [35].

4) Kernel Density Estimate Plot

A Kernel Density Estimate (KDE) plot visually represents the distribution of observations in a dataset. Utilizing a continuous probability density curve in one or more dimensions, the KDE provides a smooth description of the data. Compared to histograms, KDE plots can offer a clearer and less cluttered representation when visualising multiple distributions. It's important to note that KDE may introduce distortions when applied to non-smooth or constrained underlying distributions [36].

G. DECISION TREES

Decision Trees are a widely utilised methodology in the realm of ML and decision analysis, offering the ability to construct predictive models by associating observations with target values [37]. Structured as inverted trees, they are versatile and applicable to both classification and regression problems. These trees are organized with nodes representing decisions or attribute tests, branches denoting test outcomes, and leaf nodes indicating outcomes or predicted values, as illustrated in Figure 1. The decision-making process is initiated at the root node, branching into internal nodes, ultimately leading to the leaf nodes. Guided by the attributes of the data, this process employs recursive partitioning, leading to node splits based on attribute tests until a predefined stopping criterion is met, such as a specified depth or a minimum number of instances in a node [37].

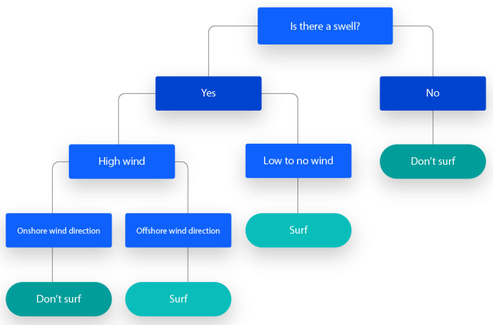


FIGURE 1. Basic Representation of Decision Tree. Reproduced from: [10]

The use of Decision Trees boasts several advantages, simplifying complex relationships and rendering them relatively easy to comprehend. Their non-parametric approach makes them adept at handling missing values and skewed data Additionally, they exhibit robustness to outliers [38]. However, Decision Trees have their shortcomings as well. These include being prone to overfitting, especially with noisy data and can be high-variance estimators. Moreover, their training process may be more resource-intensive compared to alternative algorithms as well as being more expensive to train when compared to other algorithms due to the recursive nature of the method [10, 38].

The Decision Tree framework has given rise to various algorithms, each with its distinctive characteristics. One such algorithm is the Classification and Regression Trees (Classification and Regression Trees (CART)), known for its versatility in handling both classification and regression tasks. Another notable algorithm is the Iterative Dichotomiser 3 (Iterative Dichotomiser 3 (ID3)), which emphasises attribute selection based on information gain. C4.5 [37], an extension of ID3, introduces enhancements in attribute selection, addresses the treatment of missing values, and incorporates tree pruning techniques to mitigate the risk of overfitting. These algorithms differ in their approaches to attribute selection, handling of missing values and numerical attributes, and the implementation of tree pruning mechanisms aimed at preventing overfitting [37].

In their study, Hajirahimova et al. [13] investigated four algorithms, including Decision Trees Regression, and assessed their performance in population forecasting in Azerbaijan. The results indicated a strong correlation coefficient ranging from 0.985 to 0.996. In a parallel study [14], four ML algorithms, including Decision Trees, were employed to forecast the population growth rate of a specific area. All the algorithms demonstrated accuracy levels exceeding 90%. In a separate investigation by Rady et al. [39], focusing on time series forecasting using tree-based methods, Decision Trees exhibited a propensity for closely aligning predicted values with the actual values. Furthermore, a comprehensive comparison of 17 different ML models, aimed at predicting human population growth in countries [40], showcased that Decision Trees secured the second-highest accuracy scores among all models, surpassed only by the Random Forest.

H. RANDOM FOREST

The Random Forest method stands out as a potent ensemble learning technique extensively applied in ML for both classification and regression challenges, showcasing its adaptability across various problem domains. This method harnesses the collective predictive strength of numerous Decision Trees during training, culminating in a robust and accurate mean prediction for regression tasks or a mode prediction for classification scenarios. Unlike a single Decision Tree prone to overfitting, the Random Forest addresses this concern by employing a multitude of decision trees. The ensemble nature ensures that the classifier doesn't overfit, as averaging uncorrelated trees diminishes overall variance and prediction error, bolstering the model's generalisation capabilities. For clarity, Figure 2 offers a basic visualisation of a Random Forest. To unleash the full potential of Random Forest techniques, three pivotal hyperparameters require specification before training, being the node size, number of trees, and the number of features sampled.

Although the Random Forest algorithm can provide more accurate predictions when compared to single classifiers like Decision Trees [41], in practical applications the model is considered to be slow to process due to the computation required for each Decision Tree. This also entails the need for more resources to store data. The trade-off between the potential for increased accuracy and the processing time required needs to be a key consideration in practical applications. While population statistics may benefit from heightened accuracy, other implementations that may demand real-time predictions or large-scale processing, the use of Random Forest models may necessitate careful consideration or a search for alternative algorithms. The interpretability of Random Forest predictions can also be challenging due to the ensemble nature of the model, making it more complex for users to understand the decision-making process [11].

Furthermore, contemporary variations and adaptations of the algorithm, including techniques such as Extremely Randomised Trees and Gradient Boosted Trees (GBT) are continually evolving to enhance the effectiveness of Decision Tree-based models. While some studies such as [39] still position the Random Forest algorithm as superior in predictive performance compared to GBT in certain contexts, these newer variations of the model cannot be ignored. Future studies can be conducted to analyse and compare more of these types of models.

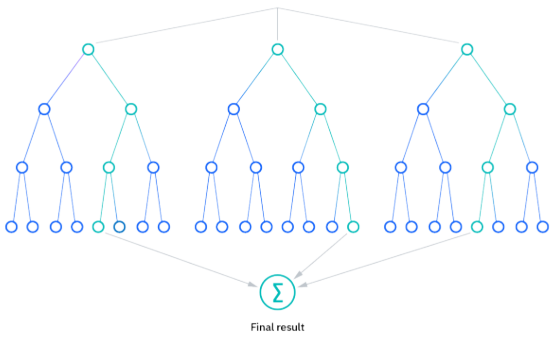


FIGURE 2. Basic Representation of Random Forest. Reproduced from: [11]

With the population prediction problem that is being tackled, one can find various literature which makes use of the RF technique to solve the aforementioned problem. Based on the results obtained by [13] and [14], the RF algorithm was one of the better performers when compared with other ML algorithms that were implemented for comparison. In [40], M. Otoom demonstrated the Random Forest algorithm provided the best results when compared to 17 other ML techniques to predict the population growth rate without feature or historical data. The RF technique outperformed other methods in predicting high, medium, and negative growth rates (0.98%, 0.95%, and 0.96%, respectively), showcasing that RF may be the best ML technique for predicting population growth rates. Hara et al. [42] achieved high accuracy in estimating attendance populations at baseball games using the Random Forest algorithm. Wang, in [41], conducted a comparative study of two techniques, one of which was the Random Forest algorithm, for predicting the population in China. The study revealed that the Random Forest algorithm exhibited superior regression accuracy and a better-fit degree compared to the alternative technique. Therefore, the versatility of Random Forests makes them a suitable choice for handling the complexity and variability in population-related datasets.

I. LINEAR REGRESSION

Linear Regression stands as a foundational machine learning modeling algorithm renowned for its simplicity and efficacy in predicting the value of a dependent variable x based on one or more independent variables y [8, 13, 43]. The methodology of Linear Regression adheres to the principles of Ordinary Least Squares (OLS) to compute the estimate of the coefficients in a linear equation, incorporating one or more independent variables to enhance the prediction accuracy of the dependent variable's value. This process entails fitting a straight line or surface that minimizes the difference between predicted and actual output values. The equation expressing the dependence of x on Y is as follows:

$$y = \beta_0 + \beta_1 x + \epsilon \tag{8}$$

where β_0 represents the y-intercept, β_1 represents the slope, and ϵ is the error term used to account for the variability between x and y [13, 19]. Moreover, the coefficients β_0 and β_1 are estimated using the Least Squares method. This entails minimising the sum of squared differences between the observed and predicted values. The resultant equation produces the best-fit line that minimises the overall prediction error.

According to [13], Linear Regression is a high-performance algorithm known for its strong predictive capabilities and statistical accuracy across different dataset sizes. Its robustness is evident irrespective of the dataset's scale, making it a suitable choice for both small and large datasets. Various studies related to population prediction have implemented and reviewed the results from the use of the Linear Regression model, and have consistently demonstrated its effectiveness within this context [13, 43, 44]. For instance, Hajirahimova et al. [13] employed the Linear Regression technique and achieved notable results in predicting the populations of Azerbaijan. In another study by Çahinarslan et al. [43], the total population of Turkey in 2017 was estimated using various ML techniques, including Linear Regression, yielding a commendable Mean Absolute Percentage Error (MAPE) of 0.0493% across the entire dataset. Similarly, Wang et al. [44] conducted a population forecast study, where Linear Regression was one of their utilised ML techniques, leading to some of the best MAPE scores within the 3, 4, and 5-year prediction ranges.

Linear Regression has spawned several alternative methods and extensions, such as Ridge Regression, which introduces regularisation, mitigates overfitting and enhances the model's performance, and Polynomial Regression, which accommodates non-linear relationships by including polynomial terms, as well as the incorporation of Multiple Linear Regression involving two or more variables used as predictors [8]. In various studies, these models were also tested and compared. While some demonstrated to have a marginally better performance than Linear Regression in the field of population forecasting [43], our deliberate choice of Linear Regression was motivated by its heightened interpretability of relationships and impact of predictors. Additionally, our consideration extended to the specific nature of the dataset, where population statistics are segmented by LAU and gender, which mitigates an analysis of multicollinearity, which would have benefited from the use of Ridge Regression [45]. Nonetheless, our decision to employ Linear Regression does not preclude future explorations from being done to compare different regression algorithms derived from Linear Regression in this domain.

III. DATA PREPARATION

For a comprehensive insight into our implementation structure, refer to Section IV-A. The data preparation process for our analysis occurred within the `NSO_DIS_SEX_YEAR_POP.ipynb` file, where the initial step involved reading the CSV file provided by the NSO website. This raw dataset underwent a comprehensive cleaning and transformation procedure to enhance its quality and usability for subsequent analysis.

In the initial cleaning steps, we removed unnecessary columns such as `DATAFLOW`, `CSE_FREQ: Frequency`, `CSE_OBS_STATUS: Observation Status`, and `CSE_OBS_NOTE: Observation Note`. These columns were either redundant or contained uniform values that did not contribute meaningfully to the analysis. Subsequently, key columns were renamed to improve readability, with `CSE_REG_DIST_LOC: Region District Locality` becoming `District`, `CSE_SEX: Sex` becoming `Sex`, `TIME_PERIOD: Time Period` becoming `Year`, and `OBS_VALUE` becoming `Population`.

Further data cleaning involved refining the `District`, `Sex`, and `Year` columns. In the `District` column, we extracted numerical information from patterns that began with a colon, followed by a space, digits, and ended with a space and a hyphen. This step ensured a standardised representation of district codes. The `Sex` column was simplified by extracting single characters representing 'Total' ('T'), 'Females' ('F'), and 'Males' ('M'). For the `Year` column, we cleaned it by extracting the numerical component that was followed by a hyphen, resulting in a refined representation of the year. Examples for each extraction can be found below:

- District: 'MGC014: 014 - Western' $\rightarrow$ '4'
- Sex: 'M: Males' $\rightarrow$ 'M'
- Year: '2005-A1' $\rightarrow$ '2005'

Additionally, the `District` column was filtered to retain only records pertaining to districts, and the last digit of each district was kept, creating a range from 1 to 6 to represent the Maltese districts. The `Sex` column was further cleaned by dropping records related to the total count and replacing 'M' with '0' and 'F' with '1', resulting in a boolean representation.

In the feature scaling and engineering phase, the `Year` and `Population` columns were feature-scaled using min-max scaling, ensuring values were scaled between 0 and 1. Two new features, `Population_Growth_Rate` and `Average_Population`, were created. `Population_Growth_Rate` represents the percentage change in population from one year to the next, while `Average_Population` calculates the mean population for each district.

Final checks involved handling missing and infinite values. Missing values were replaced with 0, and infinite values were replaced with the maximum finite value in each column to prevent issues during ML algorithm processing. The final cleaned dataset was exported to the `NSO_POPULATION_DATA_CLEANED.csv` file, ready for utilisation in our ML algorithms. This comprehensive data preparation ensured a refined and standardised dataset, suited for subsequent analysis.

IV. IMPLEMENTATION

In this section, we delve into the implementation details of our population prediction model for Malta, providing an explanation of the three chosen ML techniques: Decision Trees, Random Forest, and Linear Regression. The selection of these algorithms is underpinned by their versatility and wide-ranging applicability in regression tasks. Approaching each ML algorithm from a regression perspective aligns seamlessly with the inherent nature of our prediction task. By employing regression, we effectively model the intricate relationships between various demographic features, enabling accurate predictions of Malta's population dynamics. We also elaborate on our approaches to hyperparameter tuning and cross-validation, crucial aspects in optimising the performance and reliability of our predictive models.

A. STRUCTURE OVERVIEW

The implementation consists of two main phases. In the initial phase, a Jupyter notebook named `NSO_DIS_SEX_YEAR_POP.ipynb` was developed to handle data reading, cleansing, preparation, and subsequent exportation to a CSV file labelled `NSO_POPULATION_DATA_CLEANED.csv`. The data for this process was sourced from the `NSO_DF_TOT_POP_BY_REG_DIST_LOC_1.5.csv` file, originally obtained from the NSO website.

In the subsequent phase, another Jupyter notebook titled `main.ipynb` was implemented, alongside additional classes housing the models or auxiliary functions essential for importation. The `main.ipynb` file reads the data from `NSO_POPULATION_DATA_CLEANED.csv` and contains all the code responsible for training, predicting values, and calculating evaluation metrics through the implemented hyperparameter tuning and cross-validation.

The `Decision Trees`, `Random Forest`, and `Linear Regression` are implemented in the their own classes within the `DecisionTreeRegressor.py`, `RandomForestRegressor.py`, and `LinearRegression.py` files respectively, which all serve as imports in our `main.ipynb` file. Lastly, the `Utils` class, within the `Utils.py` file, encompasses a function for calculating the root mean squared error, an important metric further employed within `main.ipynb`.

B. DECISION TREES

Our implementation of the Decision Trees algorithm is adapted from the code available at [46]. The Decision Tree Regressor is constructed as a recursive binary tree, where each node represents a decision based on a specific feature and threshold. Within the implementation, the `Node` class represents a node in the Decision Tree, distinguishing between decision nodes and leaf nodes. Decision nodes contain information about the splitting feature, threshold, and the subsequent left and right subtrees, while leaf nodes store the predicted value.

The `DecisionTreeRegressor` class serves as the main component of the implementation. It is initialised with parameters specifying the minimum samples required to split a node (`min_samples_split`) and the maximum depth of the tree (`max_depth`). The recursive construction of the Decision Tree is handled by the `build_tree` function, which evaluates splitting conditions based on information gain until specified stopping criteria are met. Important considerations, such as the minimum number of samples required for a split (`min_samples_split`) and the maximum tree depth (`max_depth`), are meticulously assessed. Once a beneficial split is identified, the function proceeds to create left and right subtrees. The decision-making process involves key steps, including calculating variance reduction, determining the optimal split point, and assigning values to leaf nodes based on the mean of target values.

Variance reduction, computed through the `variance_-reduction` function, plays a crucial role in determining the optimal splits during tree construction. Additionally, the `calculate_leaf_value` function computes the predicted value for a leaf node by averaging the target values. The `split` function divides the dataset into left and right child datasets based on a specified feature and threshold, while the `get_best_split` function identifies the best feature and threshold combination, maximising variance reduction. Lastly, the `fit` function initiates the training process by building the Decision Tree, and the `predict` function utilises the trained tree to make predictions on new data.

A summary of the variable names and their descriptions is provided in Table 2.

Variable Name	Description
`feature_index`	Index of the feature used for splitting at a decision node
`threshold`	Threshold value for the splitting feature at a decision node
`left`	Left subtree of a decision node
`right`	Right subtree of a decision node
`var_red`	Variance reduction achieved by a split at a decision node
`value`	Predicted value for a leaf node
`min_samples_split`	Minimum number of samples required to split a node
`max_depth`	Maximum depth of the Decision Tree
`dataset`	Input dataset for building the Decision Tree
`X`	Features of the dataset
`Y`	Target variable of the dataset
`num_samples`	Number of samples in the dataset
`num_features`	Number of features in the dataset
`best_split`	Dictionary storing information about the best split
`curr_var_red`	Current variance reduction during the tree construction process
`max_var_red`	Maximum variance reduction during the tree construction process

TABLE 2. Variable Names and Descriptions for Decision Trees Implementation

C. RANDOM FOREST

Leveraging the code generated by [47], we crafted an implementation of the Random Forest algorithm. The `RandomForestRegressor` class integrates a defined number of Decision Trees through `n_estimators`, each having a maximum depth (`max_depth`) and requiring a minimum number of samples for splitting (`min_samples_split`). The `fit` function utilises the custom Decision Tree implementation, `DecisionTreeRegressor`, explained in Section IV-B. To individually train each tree, we employ a technique called bootstrap sampling. This method entails randomly selecting subsets from the original dataset, enabling each tree to learn from slightly different perspectives of the data. The Decision Tree is then trained using these random subsets. Once a tree is trained, it becomes part of a collection, or a "forest," denoted as `self.trees`. This collection keeps a record of all the individual trees, and their collective contribution forms the final prediction when making forecasts on new data.

In the prediction part of the code, after the Random Forest is trained with multiple Decision Trees, the `predict` function is defined to make predictions on new, unseen data. For each data point in the input dataset `X`, the method creates an array of zeros to store the individual predictions made by each tree in the Random Forest. The code then loops through each tree in the forest, utilising the trained models to make predictions for the entire dataset. The predictions are stored in the previously created array. Finally, the method returns the average of these individual predictions across all trees. This averaging process helps smooth out individual tree idiosyncrasies, resulting in a more stable and reliable overall prediction for the Random Forest. Essentially, by combining the insights from multiple trees, the Random Forest enhances its ability to make accurate predictions on new data.

The defining variables for the Random Forest are elucidated in Table 3.

Name	Description
`n_estimators`	The number of Decision Trees to build the Random Forest
`max_depth`	The maximum number of nodes deep that the Decision Tree can be
`min_samples_split`	The minimum number of samples required to consider splitting a non-leaf Decision Tree node

TABLE 3. Overview of Random Forest Class Variables

D. LINEAR REGRESSION

The Linear Regression model, implemented in the `LinearRegression` class, contains the `fit` function which handles the fitting process of the code and takes input features `X` and target values `y` as parameters for training. The method initialises the weight and bias of the model to zero. The core of the training process involves iteratively updating these weights and bias using a process called gradient descent. For a specified number of iterations, the method calculates predictions based on the current weights and bias. It then computes the gradients, representing the direction and magnitude to adjust the weights and bias for better predictions. These gradients are scaled by the learning rate (`learning_rate`), a hyperparameter that controls the size of each update, to prevent overshooting the optimal values. The weights and bias are updated accordingly in each iteration. This iterative process allows the model to learn from the training data and gradually improve its ability to make accurate predictions.

Conversely, the `predict` method employs the trained weights and bias to generate predictions for new data. This involves a straightforward linear calculation using the dot product of the input features and the learned weights, with the addition of the bias term. The outcome is the predicted values for the target variable.

Our implementation of the `LinearRegression` model is designed to support experimentation with various combinations of hyperparameters, enabling precise tuning of the model's behaviour. Table 4 presents a comprehensive overview of the variables employed in both the training and prediction processes of the model.

E. HYPERPARAMETER TUNING AND CROSS-VALIDATION

Our hyperparameter tuning approach utilises a systematic grid search strategy and was implemented within the `custom_hyperparameter_tuning` function, designed for both hyperparameter tuning and cross-validation for our ML models. As input, the function takes the dataset features (`X`) and target variable (`y`), a grid of hyperparameter combinations (`parameter_grid`) to explore the diverse combinations of hyperparameters in each iteration, the choice of ML algorithm (`algorithm`), and the number of folds for cross-validation (`num_splits`, chosen to be `10`). The function initialises variables to track the best hyperparameters

Variable Name	Description
`learning_rate`	The learning rate, determining the step size in the gradient descent algorithm.
`num_iterations`	The number of iterations for the gradient descent optimisation process.
`weights`	The weights of the Linear Regression model, representing the coefficients for each feature.
`bias`	The bias term of the Linear Regression model, representing the intercept.
`X`	The feature matrix containing input data samples. Rows are samples, columns are features.
`y`	The target values corresponding to the input data samples.
`num_samples`	The number of samples in the input data (`X`).
`num_features`	The number of features in the input data (`X`).
`predictions`	The predicted values computed by the Linear Regression model.
`dw`	The gradient of the weights in the Linear Regression model.
`db`	The gradient of the bias in the Linear Regression model.

TABLE 4. Overview of the variables in the Linear Regression Implementation

(`best_params`) and the corresponding MSE (`best_mse`). It then employs K-Fold cross-validation, iterating through the provided hyperparameter combinations. For each set of hyperparameters, it trains the specified ML model (Decision Tree, Random Forest, or Linear Regression) on each fold of the dataset, calculates the MSE, and averages the results across all folds. The function updates the best hyperparameters only if the calculated MSE is lower than the current best. The choice of MSE as the performance metric is grounded in its widespread use in regression problems, measuring the average squared difference between actual and predicted values, as highlighted by [48]. Ultimately, the function returns the set of hyperparameters that resulted in the lowest MSE across all folds. The choice of algorithm determines the type of ML model instantiated and trained within the function's loop. The overarching goal is to find hyperparameters that optimise the model's performance on the provided dataset.

Subsequently, we apply the best parameters identified during the hyperparameter tuning process to determine the optimal model for the ongoing run. This optimised model is then utilised to make predictions and aggregate population values, along with the MAE, MedAE, MaxAE MSE, RMSE, R2, Explained Variance, and Execution Time metric results that would be later averaged and employed within our experiments, as detailed in Section V. The specific hyperparameters considered for each algorithm are detailed below:

1) Decision Trees

In the case of Decision Trees, we delve into the `min_samples_split` and `max_depth` hyperparameters. `min_samples_split` defines the minimum number of samples needed to split an internal node, influencing the granularity of decision-making. On the other hand, `max_depth` establishes the limit on the tree's depth, impacting its overall complexity and generalisation capability. The hyperparameter ranges under consideration are `min_samples_split` in [2, 3, 4, 5, 10, 15, 20] and `max_depth` in [10, 25, 50, 75, 100, 125, 150, 175, 200].

2) Random Forest

In the Random Forest algorithm, we explore the `n_estimators`, `min_samples_split`, and `max_depth` hyperparameters. `n_estimators` determines the number of trees in the forest, `min_samples_split` sets the threshold for the minimum samples required to split a node, and `max_depth` constrains the maximum depth of each tree. The hyperparameter ranges considered are `n_estimators` in [75, 100, 125], `min_samples_split` in [2, 3, 4, 5, 10, 15, 20], and `max_depth` in [10, 25, 50, 75, 100, 125, 150, 175, 200].

3) Linear Regression

In Linear Regression, we adjust the `learning_rate` and `num_iterations` hyperparameters. `learning_rate` influences the step size in the optimisation process, and `num_iterations` determines the number of iterations during training. The exploration encompasses `learning_rate` in [0.001, 0.01, 0.1, 1] and `num_iterations` in [50, 100, 250, 500, 750, 1000, 1250, 1500].

To determine the optimal hyperparameter options, we examined the default values predefined by SKLearn for their algorithms and supplemented them with a range of values in proximity. The hyperparameter options undergo iteration for a predetermined number of runs, denoted as `NUMBER_OF_RUNS`. We set the number of runs for all three ML algorithms to 10, which when paired with the 10 k-folds used for our cross-validation, strikes a balance for model optimisation. This choice allows for a comprehensive exploration of a substantial portion of the hyperparameter space while efficiently utilising computational resources, without incurring excessive computing costs.

Ultimately, the code provides a detailed summary, presenting the average performance metrics for each algorithm. This comprehensive analysis unveils the relationship between hyperparameter choices and model performance, offering insights into the strengths and weaknesses of Decision Trees, Random Forest, and Linear Regression in the context of predicting population dynamics in Malta.

V. EXPERIMENTS

This section delves into an exploration of the efficacy of our three ML models, offering a detailed analysis that incorporates various evaluation methods and graphs. These components contribute to a comprehensive comparison of the outcomes obtained. Throughout this investigation, a consistent 80/20 train-test split was employed to ensure a rigorous evaluation of the model's predictive capabilities on unobserved data.

A. HARDWARE USED

It is important to acknowledge that the observed performance metric of average execution time may be influenced by the hardware specifications of the device on which the algorithms were executed. However, these outcomes provide valuable insights into the comparative efficiency of the algorithms. The following hardware specifications were employed during our evaluation:

- CPU: 2.20 GHz Intel Core i9-13900HX
- RAM: 32 GB 4800 MHz SODIMM
- OS: Windows 11.

B. EXPERIMENTS CARRIED OUT

In evaluating the effectiveness and generalisation capability of our models, we conducted experiments using average values for key metrics. These metrics, unravelled in Section II-E include Execution Time, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Median Absolute Error (MedAE), Median Absolute Error (MaxAE), R-squared (R2), and Explained Variance. The experimental results were obtained using average values derived from our cross-validation process, as elaborated in Section IV-E. To ensure robustness and reliability in assessing each metric, we conducted 10 runs, each comprising 10 k-folds.

To provide a comprehensive understanding of model behaviour, visualisations were created using average actual values and average predicted values across all runs. These visualisations include a scatter plot, residual plot, and a KDE plot, elaborated upon in Section II-F. Leveraging the `matplotlib` and `seaborn` packages, these visual tools offer insights into how our models perform across different thresholds and facilitate a detailed exploration of their predictive patterns.

C. RESULTS COMPARISON

In this section, we delve into a comprehensive evaluation of the results obtained from the selected ML techniques: Decision Trees, Random Forest, and Linear Regression. To assess these results thoroughly, we explore various evaluation metrics and charts as outlined in Section V-B. In Table 5, we present the results obtained for our chosen evaluation metrics offering a comprehensive insight into the performance of our ML models, averaged over 10 runs with 10 k-folds each.

The MAE stands as a pivotal metric, depicting the average absolute difference between predicted and actual values. Lower MAE values signify superior predictive accuracy. In our experiments, the Random Forest model demonstrated the most outstanding performance, attaining the lowest average MAE value when compared to both the Decision Tree and Linear Regression models.

Moving on to the MedAE, which captures the median of absolute errors, provides insights into the central tendency of prediction errors. This time, the Decision Tree model emerges as the top performer, displaying a lower average MedAE value than both the Random Forest and Linear Regression, underscoring its robustness and consistency in predictions.

Examining the results from the average MaxAE metric, we find that the Decision Tree model also came out on top when minimising error. The average MaxAE value demonstrate that the Decision Tree model consistently provides predictions with minimal absolute errors compared to the Random Forest and Linear Regression models.

The MSE measures the average squared difference between predicted and actual values. A lower MSE indicates better accuracy. Our findings affirm the dominance of the Random Forest model, as it consistently maintains a lower average MSE value in comparison to the Linear Regression algorithm.

Derived from the square root of the MSE, the RMSE provides a measure of the spread of errors. Once again, the Random Forest model stands out with the lowest average RMSE value, indicating its ability to minimise the impact of larger errors.

Additionally, the R2 score and Explained Variance are crucial for assessing how well the models capture the variance in the data. The Random Forest model exhibits the highest average scores, suggesting that it explains a significant proportion of the variability in the target variable compared to the other models.

Turning our attention to execution time, the Random Forest algorithm takes the longest average time to execute among the three ML algorithms. This observation aligns with expectations, as the Random Forest algorithm involves an ensemble process and generating numerous decision trees, making it computationally more complex and resource-intensive. This insight provides valuable considerations for applications where computational efficiency is a critical factor in model selection.

Evaluation Metric	Decision Tree	Random Forest	Linear Regression
Execution Time (s)	0.111625	6.009703	0.067633
MAE	0.01179	0.01177	0.03631
MedAE	0.00431	0.00486	0.02823
MaxAE	0.08622	0.09202	0.17035
MSE	0.00054	0.00054	0.00258
RMSE	0.02228	0.02201	0.05001
R2	0.98949	0.98966	0.94883
Explained Variance	0.98999	0.99013	0.95074

TABLE 5. Average results obtained across 50 runs

Moreover, the tabulated data in Table 6 offers a comprehensive examination of the average actual and average predicted values derived from the three distinct models over 10 experimental runs consisting of 10 k-folds each. Upon examining the average differences between the actual and predicted values, it is noteworthy that the Random Forest algorithm exhibits the smallest average difference (0.0006), indicating superior predictive accuracy when compared to the Decision Tree and Linear Regression. This suggests that, on average, the Random Forest model provides the closest predictions to the actual values in the given dataset. The Decision Tree model follows closely (0.0008), while Linear Regression trails with a higher average difference (0.0071).

Actual Values	Decision Tree	Random Forest	Linear Regression
0.40070	0.39926	0.40652	0.40957
0.29433	0.28908	0.29073	0.29689
0.32691	0.32109	0.32021	0.32085
0.33624	0.33383	0.33815	0.34761
0.29232	0.29228	0.29255	0.30307
0.30100	0.30487	0.30929	0.29924
0.30598	0.30467	0.31448	0.32143
0.20829	0.20756	0.20818	0.22140
0.28428	0.29170	0.29511	0.31706
0.15349	0.15757	0.15867	0.14583
0.24420	0.23149	0.23336	0.21130
0.19186	0.19292	0.19517	0.20445
0.30271	0.30124	0.30214	0.30923
0.31213	0.31683	0.31799	0.30144
0.18756	0.18776	0.19197	0.19966
0.16955	0.17761	0.17432	0.16059
0.35824	0.36283	0.36879	0.37595
0.29140	0.28938	0.29124	0.29922
0.35017	0.36023	0.36704	0.35369
0.20311	0.19216	0.18902	0.16437
0.41700	0.40259	0.40547	0.41012
0.24089	0.24728	0.24647	0.23923
0.29807	0.29649	0.30000	0.28126
0.21792	0.22056	0.22166	0.23230
0.24217	0.24736	0.24807	0.24670
0.21880	0.21721	0.22315	0.22660
0.31830	0.32871	0.32693	0.28279
0.44925	0.46464	0.46295	0.43303
0.22517	0.22469	0.22524	0.22484
0.23177	0.24430	0.23694	0.24061
0.26034	0.26397	0.26479	0.25574
0.31627	0.32292	0.32722	0.33321
0.44298	0.45125	0.44895	0.42385
0.47538	0.47425	0.46868	0.42975
0.30805	0.32973	0.32314	0.29619
0.33836	0.34554	0.34340	0.36275
0.21474	0.22589	0.22348	0.23825
0.36905	0.37407	0.37418	0.35360
0.19988	0.20181	0.20220	0.19534

TABLE 6. Average Actual vs Average Predicted Values Results

Figure 3 depicts a scatter plot showcasing the average actual and average predicted values of the three distinct ML models. The y-axis represents the average predicted values, while the x-axis denotes the average actual values. Additionally, a best-fit line is plotted, capturing the overall trend across all the average predictions made by the ML algorithms, alongside the average actual results. The scatter plot also reveals that for Linear Regression, the predicted values fall below the actual values, suggesting an underestimation by the models. On the other hand, the predicted values for both Decision Trees and Random Forest are present above the actual values, suggesting overestimation. Among the implementations, the Random Forest stands out as the highest-performing, with projected values closely aligned with the actual values, indicating a higher confidence in its predictions. The Decision Tree, as the second-best performer, exhibits projected values in proximity to those generated by the Random Forest. Conversely, the Linear Regression implementation demonstrates the lowest performance, with predicted values deviating the most from the actual values.

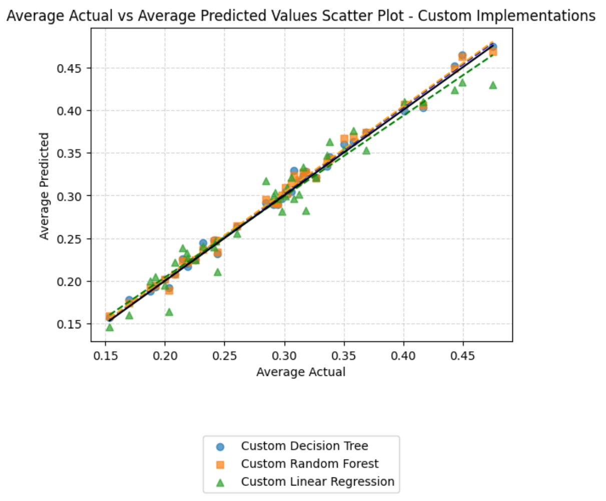


FIGURE 3. **Scatter Plot of Average Actual vs Average Predicted Values**

Figure 4 presents a residual plot featuring residuals and average predicted values for the three ML models. Residuals represent the difference between the actual number of harvested trees and the number predicted by each model. The x-axis displays the average predicted values, while the y-axis depicts the residuals. The plot illustrates how the Random Forest demonstrates a more scattered and evenly distributed pattern of residuals around the horizontal axis, indicating a reduced presence of systematic errors or patterns in prediction discrepancies. This dispersion demonstrates the Random Forest's robustness and versatility in capturing complex relationships in the data. The Decision Tree residuals provide a similar distribution to that of the Random Forest. However, given its slightly higher average difference compared to the Random Forest, the plot indicates areas where the Decision Tree struggles to generalise. In contrast, the greater disparity resulting from the Linear Regression model implies a comparatively lower level of predictive accuracy when compared to both the Random Forest and Decision Tree models. In summary, the graph implies that both the Random Forest and Decision Tree models offer a better fit to the data compared to the Linear Regression model.

A KDE plot is visualised in Figure 5 where the distribution

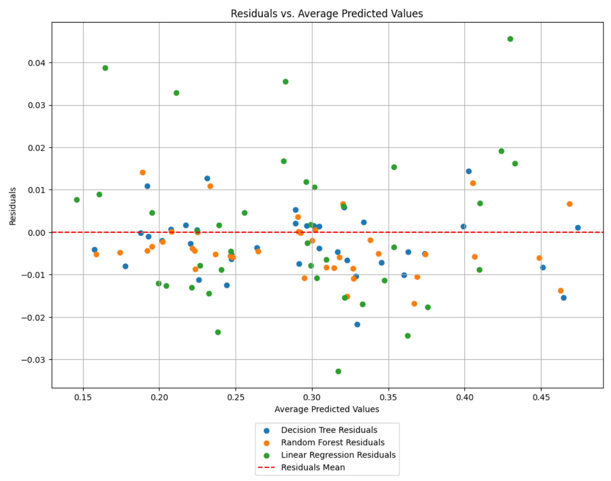


FIGURE 4. **Residual Plot of Residuals vs Average Predicted Values**

of the average actual and average predicted values for the three ML models are plotted. On the x-axis, average actual values are displayed, while the y-axis represents average predicted values. The density of points at a specific coordinate reflects the quantity of data points with that actual-predicted value pair. The plot reveals that, on average, the actual values surpass the predicted values for all three models, indicating that perfect prediction is not achieved. However, the models exhibit a certain level of accuracy, as evidenced by the clustering of points around the diagonal line. Among the models, the Decision Tree model displays the highest MSE, making it the least accurate predictor of average actual values. Nonetheless, it boasts the smallest standard deviation, indicating a higher degree of prediction consistency. In contrast, the Random Forest model, while slightly more accurate than the custom decision tree with a lower MSE, shows a marginally higher standard deviation, implying slightly less consistent predictions. Finally, the Linear Regression model boasts the lowest MSE among the three, making it the most accurate predictor of average actual values. However, it comes with the trade-off of the highest standard deviation, signifying less consistency in its predictions. Despite their greater complexity and the requirement for a larger training dataset, both the Decision Tree and Random Forest models appear to surpass the performance of the Linear Regression model. They demonstrate an enhanced ability to learn the underlying relationship between actual and predicted values with greater accuracy.

In summary, the Random Forest model outperforms the Decision Tree and Linear Regression models across various evaluation metrics, showcasing its efficacy in achieving superior predictive accuracy. These findings highlight the efficacy of ensemble methods like Random Forest in capturing complex relationships within the data, making it a promising choice for regression tasks. It is crucial to assess these findings in the context of specific requirements and objectives.

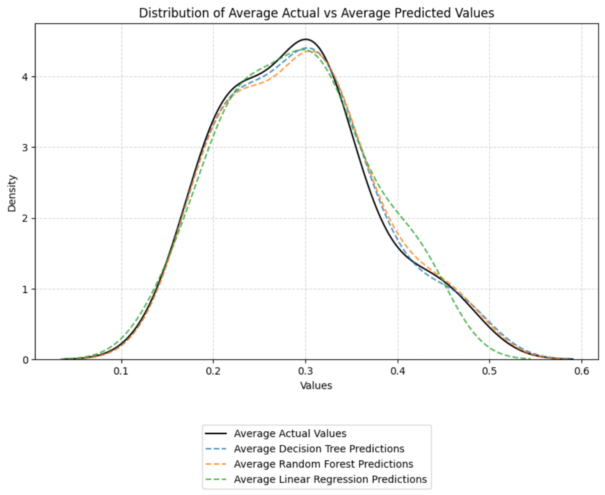


FIGURE 5. **Kernel Density Estimate (KDE) Plot of Average Actual vs Average Predicted Values**

The selection of the most suitable algorithm depends on the dataset's characteristics and the desired trade-offs between interpretability, computational efficiency, and predictive performance.

VI. ETHICAL REVIEW

An ethical review holds paramount importance in any data-driven project, acting as a safeguard against potential pitfalls and ensuring responsible and equitable practices. In this context, we will delve into crucial ethical considerations across various dimensions, including Data Source and Integrity, Data Bias, Transparency and Accountability, Data Privacy, Consent and Opt-In, Fairness and Equity, Social Impact and Long-term Effects, Environmental Impact, and web tool accessibility.

A. DATA SOURCE AND INTEGRITY

The dataset titled 'Total Population by region, district, and locality' was obtained from the NSO Statistical Database, operating under the regulations outlined in the Malta Statistics Authority Act[2]. Recognising that potential biases and inaccuracies may influence our models' outcomes, we believe that NSO's credibility allows us to view the dataset as being derived from a reputable source.

B. DATA BIAS, TRANSPARENCY AND ACCOUNTABILITY

Since the collected data is related to information found on government documents rather than collected through surveys or sampling methodologies, which could introduce inadvertent oversights in the inclusion of specific demographic groups, we are also confident that the data collection method is free from inherent bias and instils a level of confidence in its integrity and transparency.

C. DATA PRIVACY

The chosen dataset comprises demographic information encompassing the entire Maltese population, offering a comprehensive view of population statistics across Malta's LAU, categorised by gender. Notably, the dataset is devoid of any sensitive Personally Identifiable Information (PII) such as names, addresses, or identification numbers, ensuring the privacy and anonymity of individuals.

However, due to the limitation of information present in the population, the dataset may not accurately mirror the demographic diversity, socioeconomic variations, and other pertinent factors intrinsic to Malta's populace, which may differ from one locality or district to another. These additional categorisations may impact the reliability of the dataset as a whole to be used for population prediction.

D. CONSENT AND OPT-IN

Despite these limitations, the data provided is part of government census data collected for official purposes and therefore comprehensive of the entire population rather than a sample or estimate. This makes it a valuable and standardised resource for demographic analysis.

E. FAIRNESS AND EQUITY

An ethical consideration arises from the binary categorisation of individuals into `male` and `female` sections under the `sex` attribute. While Malta introduced a third alternative gender marker in official documentation as part of the LGBTIQ Action Plan[3], the provided dataset fails to acknowledge these individuals. It is important to understand that due to this classification, the diverse spectrum of gender identities is not accurately captured, potentially posing fairness concerns for individuals who do not strictly identify as either.

F. SOCIAL IMPACT AND LONG-TERM EFFECTS

The general application and prevalence of ML models are on the rise in contemporary society, particularly in the realm of forecasting. Given that population trend forecasting significantly influences decision-making and planning for both corporations and governments, the adoption of diverse models can have profound and widespread impacts on societies at large. Malta, facing distinctive challenges related to land usage, overpopulation, and low fertility rates, underscores the importance of population forecasting [5]. Effectively addressing demographic issues, such as an ageing population and the influx of immigrants, hinges on the strategic use of population forecasting to facilitate well-informed decision-making. This involves judicious resource allocation, thoughtful planning in both urban and rural settings, and consideration of various

[2]Malta Statistics Authority Act: https://legislation.mt/eli/cap/422/eng/pdf.

[3]LGBTIQ Action Plan 2015-2017: https://humanrights.gov.mt/en/Documents/Publications/LGBTI\%20Action\%20Plan\%202015-2017.pdf.

factors contributing to the island's growth.

G. ENVIRONMENTAL IMPACT

By providing decision-makers with insights derived from accurate population predictions, it enables proactive planning and resource allocation to address challenges associated with overpopulation. The potential benefits in terms of sustainable urban development, and efficient resource management could potentially reduce the strain on the environment if this aspect is prioritised.

H. WEB TOOL ACCESSIBILITY

The web application developed for this project using Gradio[4] was designed with meticulous attention adhering to universal design principles. This focus ensures accessibility by delivering an interactive and inclusive experience. The user-friendly design of the interface ensures that individuals with diverse abilities can effectively engage with the ML models. The interface incorporates clear labelling, informative descriptions where necessary, and user-friendly input features, enabling efficient manipulation of the models.

VII. WEB PORTAL USAGE GUIDE

The interactive web portal[5], developed using Gradio, offers a user-friendly interface for evaluating ML algorithms such as Decision Trees, Random Forests, and Linear Regression. These algorithms are implemented both as custom solutions and through the widely-used scikit-learn library[6]. The portal interface is structured into two columns. The left-hand side displays the input dataset and user options, while the right-hand side showcases the outputs of the executed models.

A. USER INPUT

In terms of inputs, the first item in the column is a table presenting records from the cleaned 'Total Population by region, district and locality' dataset. This table provides an overview of total populations categorised by district, gender, and year. Users can visualise this information as the input for the models they intend to assess, gaining insights into dataset size and available categorisations.

To initiate the models, users can either utilise the `Run all algorithms/dataset` button, which processes all algorithms on the entire dataset, or selectively choose a single algorithm. Additionally, users have the option to filter results based on specific years and districts for a more targeted assessment. Users may tailor their preferences and set specific parameters for each algorithm to fine-tune the model's behaviour according to their analytical needs. Moreover, Table 7 depicts the key parameters used and a corresponding brief explanation.

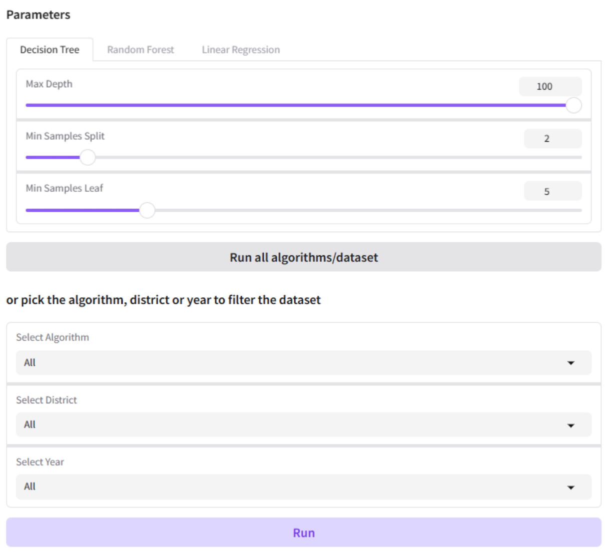
[4]Gradio: https://www.gradio.app/.
[5]Web Portal: https://nathanportelli.github.io/ICS5110-Applied-ML/
[6]scikit-learn: https://scikit-learn.org/stable/.

Parameter	Description
`max_depth`	Utilised in both the implemented Decision Tree and Random Forest algorithms, this parameter determines the maximum depth of the Decision Tree. A higher value enables the model to capture more complex relationships but may elevate the risk of overfitting.
`min_samples_split`	Employed by both the Decision Tree and Random Forest, this parameter represents the minimum number of samples required to split an internal node. Higher values may prevent the model from creating nodes that capture noise, thereby improving generalisation.
`n_estimators`	Specifically relevant to the Random Forest algorithm, this parameter defines the number of Decision Trees in the forest. A higher number generally enhances performance but comes with an increased computational cost.
`learning_rate`	Pertinent to Linear Regression, this parameter scales the contribution of each tree. A lower learning rate requires more trees for the same performance but may improve generalisation.
`num_of_iterations`	Also associated with Linear Regression, this parameter denotes the number of boosting stages to be run. Increasing this value generally improves model performance but may lead to overfitting if set excessively high.

TABLE 7. Overview of key parameters used in our web portal

As shown in Figure 6, these parameters are organised into tabs corresponding to the relevant algorithm, enabling users to specify parameters for each. The input method for these parameters is a slider, ensuring user input remains within reasonable bounds and enhancing visualisation. However, users also have the flexibility to manually input values by directly clicking on the numbers.

The Decision Tree, Random Forest, and Linear Regression algorithms are further categorised into two sections: a custom implementation developed specifically for this project and a scikit-learn implementation widely recognised in the ML community. This categorisation of algorithms into custom and scikit-learn implementations serves a crucial purpose, allowing users to conduct a meaningful performance comparison. By providing both our custom-developed algorithms and the widely-accepted scikit-learn implementations, users can assess how each algorithm variant performs in different scenarios. This comparative analysis offers valuable insights into the strengths and weaknesses of the custom algorithms


Parameters

Decision Tree | Random Forest | Linear Regression

Max Depth: 100

Min Samples Split: 2

Min Samples Leaf: 5

Run all algorithms/dataset

or pick the algorithm, district or year to filter the dataset

Select Algorithm: All

Select District: All

Select Year: All

Run

FIGURE 6. Input Parameters of Web Portal

concerning the well-established scikit-learn counterparts.

Initiating the algorithm evaluation is as simple as clicking the `Run` button, which processes the selected inputs and parameters. After a brief period, the results and evaluations are displayed on the right-hand side of the interface.

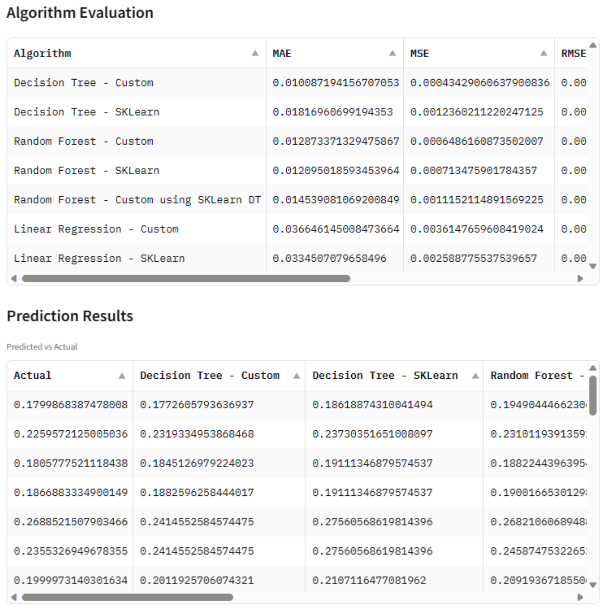
B. OUTPUT

Interpretability is a key aspect of understanding the outputs generated by the algorithms. The tables and graphs presented in the output section play a pivotal role in facilitating interpretation. This emphasis on interpretability ensures that users not only obtain accurate results but also comprehend the rationale behind those results, fostering a more insightful and informed analytical process. To this end, four sections were created to provide users with a comprehensive view of the algorithmic outputs.

The first section of the 'output' column is made up of three tables, divided into tabs for readability. The first table shows a filtered version of the input dataset, based on the user's filtering of districts and years. The second table shows the complete output of `X_test`, and the third table displays the filtered `X_test`, again based on the user's filtered dataset.

The second section provides users with a structured view of the model's performance, displaying algorithm evaluations, and presenting outputs of evaluation metrics (MAE, MedAE, MaxAE, MSE, RMSE, R2 and Explained Variance) in table format.

The third section consists of a table showcasing prediction results, with the first column displaying actual values and the subsequent columns displaying predicted values for each selected algorithm.


Algorithm Evaluation

Algorithm	MAE	MSE	RMSE
Decision Tree - Custom	0.010087194156707053	0.00043429060637900836	0.00
Decision Tree - SKLearn	0.01816960699194353	0.0012360211220247125	0.00
Random Forest - Custom	0.012873371329475867	0.0006486160873502007	0.00
Random Forest - SKLearn	0.012095018593453964	0.000713475901784357	0.00
Random Forest - Custom using SKLearn DT	0.014539081069200849	0.0011152114891569225	0.00
Linear Regression - Custom	0.036646145008473664	0.0036147659608419024	0.00
Linear Regression - SKLearn	0.0334507079658496	0.002588775537539657	0.00

Prediction Results

Predicted vs Actual

Actual	Decision Tree - Custom	Decision Tree - SKLearn	Random Forest -
0.1799868387478008	0.1772605793636937	0.18618874310041494	0.1949044466230
0.2259572125005036	0.2319334953868468	0.23730351651008097	0.2310119391359
0.1805777521118438	0.1845126979224023	0.19111346879574537	0.1882244396395
0.1866883334900149	0.1882596258444017	0.19111346879574537	0.1900166530129
0.2688521507903466	0.2414552584574475	0.27560568619814396	0.2682106068948
0.2355326949678355	0.2414552584574475	0.27560568619814396	0.2458747532265
0.1999973140301634	0.2011925706074321	0.2107116477081962	0.2091936718550

FIGURE 7. Algorithm Evaluation and Prediction Results outputted on Web Portal

The fourth section presents insightful graphical representations generated using `matplotlib`. Users can use these to visually compare actual and predicted values, discern trends, and identify any patterns or discrepancies. These graphical representations include scatter plots, offering a clear comparison between the actual and predicted values as shown in Figure 7, line plots that elucidate the trend of actual values in conjunction with predictions in the same format, and residual plots that highlight the disparities between predicted and actual values. To enhance clarity, scatter plots and line plots are categorised for both custom implementations and scikit-learn algorithms, both collectively and individually. Moreover, dedicated graphs are provided for each algorithm's custom and scikit-learn implementations. As displayed in Figure 8, residual plots are uniquely depicted for each algorithm, with the residual mean line distinctly highlighted in red. These visualisations serve as powerful tools for users to intuitively grasp the performance of the algorithms, fostering a comprehensive understanding of their predictive capabilities.

VIII. CHALLENGES AND LIMITATIONS

A. DATA COLLECTION AND REPRESENTATIVENESS

The dataset used in this study is sourced from government documents, which may introduce limitations in capturing the full demographic diversity and socioeconomic variations present in Malta's populace. The absence or under-representation of specific demographic groups within the dataset could potentially introduce biases and impact the accuracy of population forecasts. Additionally, the dataset's

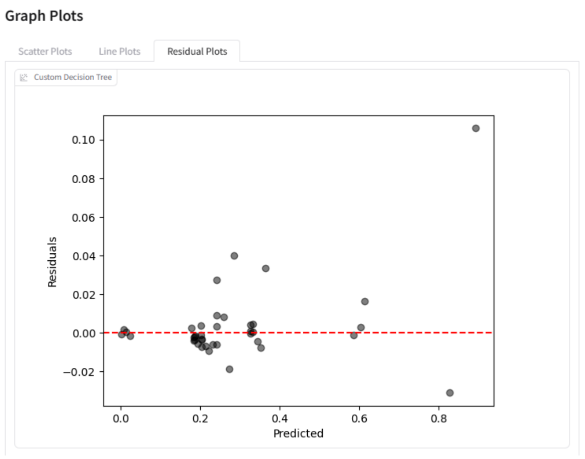


FIGURE 8. Residual Plot on Custom Decision Tree on Web Portal

representativeness at the local or district level may be limited, potentially affecting the reliability of predictions for smaller geographical areas.

B. DATA AVAILABILITY

While the dataset provides valuable insights into the demographic landscape of Malta, challenges related to available data must be acknowledged. The reliability of population forecasts heavily depends on the availability of a wide array of input data. The use of datasets with few attributes may lead to inaccurate predictions. Additionally, the dataset's temporal scope may influence the models' ability to capture dynamic demographic shifts over time. Due to intercensal revisions of the time series by the NSO, we could not use the data from 2021 and 2022. Addressing these challenges requires thorough data validation, cleaning, and augmentation processes to enhance the overall quality and completeness of the dataset.

C. EXTERNAL FACTORS AND ASSUMPTIONS

Beyond inherent limitations in data quality and representativeness, the models developed for population prediction are grounded in certain assumptions. These assumptions include the continuation of historical demographic trends without significant alterations. However, external factors such as changes in government policies, shifts in migration patterns, economic fluctuations, or unexpected events, can introduce substantial deviations from historical patterns. Therefore, while the models provide valuable insights, their reliability diminishes when confronted with substantial external factors, potentially limiting the accuracy of long-term forecasts. Continuous monitoring and adjustment of the models in response to evolving circumstances are essential for maintaining their predictive accuracy and relevance.

IX. FUTURE WORK

Building upon the findings and insights gained from this study, several avenues for future research and development emerge, offering opportunities to further advance the application of ML in demographic forecasting and to address the identified limitations.

A. ADDITIONAL DEMOGRAPHIC FACTORS

Future exploration into the integration of additional demographic factors, such as migration patterns, socioeconomic indicators, and urban development data could be used to enhance the accuracy and richness of population prediction models. By incorporating a broader range of variables, the predictive capabilities of the algorithms can be further refined to capture the multifaceted nature of demographic trends.

B. DYNAMIC MODEL ADAPTATION

While the models themselves would benefit from additional data, a need for dynamic model adaptation arises to account for unforeseen events, policy changes, and evolving demographic trends. Developing adaptive ML frameworks capable of adjusting to real-time demographic shifts can contribute to more responsive and accurate population forecasts.

X. CONCLUSION

Recent research indicates that ML has proven to be highly effective in forecasting demographic time series, outperforming traditional demographic methods commonly employed in population projections, which tend to exhibit biases. This study aimed to identify the most effective regressor among the three distinct ML algorithms chosen for predicting the total population. To achieve this, the study evaluates the performance and capabilities of ML regression approaches, namely Decision Trees, Random Forest, and Linear Regression, in population prediction. Data from the NSO website was utilised for the analysis. Eight metrics, including Mean Absolute Error (MAE), Median Absolute Error (MedAE) Median Absolute Error (MaxAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared (R2), Explained Variance, and Execution Time, were employed and averaged over 10 runs having 10 k-folds each to compare the predictive accuracy of the models. Moreover, three visualisation techniques, a Scatter Plot, Residual Plot and a Kernel Density Estimate Plot, were plotted to provide further insight when comparing the ML techniques chosen.

The analysis revealed that all ML models demonstrated high performance, with both R2 and Explained Variance results ranging from 0.95 to 0.99, capturing a sizeable amount of the target variable's variability. The Random Forest model displays lower MAE and RMSE scores when compared to the Decision Tree and Linear Regression models, showing better prediction accuracy. On the other hand, the Decision Tree algorithm held its own, achieving the same MSE score as the Random Forest, and better MedAE and MedAE than both the Random Forest and Linear Regression models. Despite being the slower performing due to its complexity, the Random Forest still proved its overall superiority over the Decision Tree and Linear Regression models.

As a result, utilising the benefits of ML algorithms makes it possible to objectively observe population increase projections for the near future. This in turn makes it easier to eliminate mistakes from the socio-economic activity planning process that pertain to work, education, and other sectors, enabling the creation of more dependable and long-lasting plans. Future studies may expand on this strategy by employing further ML algorithms to estimate additional demographic factors.

REFERENCES

[1] Malta Employers' Association. Malta's demographic challenges, 2017. A Position Paper by the Malta Employers' Association.

[2] Marvin Formosa. Measuring and Modelling Demographic Trends in Malta: Implications for Ageing Policy.

[3] Immigration in Malta: Integration through Education | European Website on Integration. https://migrant-integration.ec.europa.eu/library-document/immigration-malta-integration-through-education_en, December 2023.

[4] Census of Population and Housing 2021: Final Report: Population, migration and other social characteristics. https://nso.gov.mt/themes_publications/census-of-population-and-housing-2021-final-report-population-migration-and-other-social-characteristics-volume-1/. Accessed on 27/12/2023.

[5] Regional Statistics MALTA 2023 Edition. https://nso.gov.mt/themes_publications/regional-statistics-malta-2023-edition/. Accessed on 27/12/2023.

[6] Asian Development Bank. *Introduction to Small Area Estimation Techniques: A Practical Guide for National Statistics Offices*. Asian Development Bank, May 2020.

[7] Osvaldo Simeone. A Very Brief Introduction to Machine Learning With Applications to Communication Systems, November 2018.

[8] Iqbal H. Sarker. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, 2(3):160, March 2021.

[9] Jason Denzil Morgenstern, Emmalin Buajitti, Meghan O'Neill, Thomas Piggott, Vivek Goel, Daniel Fridman, Kathy Kornas, and Laura C. Rosella. Predicting population health with machine learning: A scoping review. *BMJ Open*, 10(10):e037860, October 2020.

[10] What is a Decision Tree | IBM. https://www.ibm.com/topics/decision-trees. Accessed on 26/12/2023.

[11] What is Random Forest? | IBM. https://www.ibm.com/topics/random-forest. Accessed on 26/12/2023.

[12] About Linear Regression | IBM. https://www.ibm.com/topics/linear-regression. Accessed on 27/12/2023.

[13] Institute of Information Technology of the Ministry of Science and Education of the Republic of Azerbaijan, Baku, AZ1141, Azerbaijan, Makrufa Sh. Hajirahimova, and Aybeniz S. Aliyeva. Development of a Prediction Model on Demographic Indicators based on Machine Learning Methods: Azerbaijan Example. *International Journal of Education and Management Engineering*, 13(2):1–9, April 2023.

[14] Mohammad Otoom, Mahdi Jemmali, Yousef Qawqzeh, Khalid Abdus Sattar, and Fayez Alfayez. Comparative Analysis of Different Machine Learning Models for Estimating the Population Growth Rate in Data-Limited Area. *IJCSNS*, 19(12):96, November 2019.

[15] Aniruddha Bhandari. Feature Engineering: Scaling, Normalization, and Standardization (Updated 2023), April 2020.

[16] Kelsy Cabello-Solorzano, Isabela Ortigosa de Araujo, Marco Peña, Luís Correia, and Antonio J. Tallón-Ballesteros. The Impact of Data Normalization on the Accuracy of Machine Learning Algorithms: A Comparative Analysis. In Pablo García Bringas, Hilde Pérez García, Francisco Javier Martínez de Pisón, Francisco Martínez Álvarez, Alicia Troncoso Lora, Álvaro Herrero, José Luis Calvo Rolle, Héctor Quintián, and Emilio Corchado, editors, *18th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2023)*, Lecture Notes in Networks and Systems, pages 344–353, Cham, 2023. Springer Nature Switzerland.

[17] S. Gopal Krishna Patro and Kishore Kumar Sahu. Normalization: A Preprocessing Stage, March 2015.

[18] Yuan Peiwen and Zhu Changsheng. Normalized Activation Function: Toward Better Convergence, September 2022.

[19] Weikuan Jia, Meili Sun, Jian Lian, and Sujuan Hou. Feature dimensionality reduction: A review. *Complex & Intelligent Systems*, 8(3):2663–2693, June 2022.

[20] Fengxi Song, Zhongwei Guo, and Dayong Mei. Feature Selection Using Principal Component Analysis. In *Engineering Design and Manufacturing Informatization 2010 International Conference on System Science*, volume 1, pages 27–30, November 2010.

[21] Features, Reduced: Feature Selection, Dimensionality Reduction and Embeddings. In Pablo Duboue, editor, *The Art of Feature Engineering: Essentials for Machine Learning*, pages 79–111. Cambridge University Press, Cambridge, 2020.

[22] Shichen Liu. Research on computational methods and algorithms for dimensionality reduction and feature selection in high-dimensional data. *Journal of Logistics, Informatics and Service Science*, 10(3):1–12, 2023.

[23] Bernd Bischl, Martin Binder, Michel Lang, Tobias Pielok, Jakob Richter, Stefan Coors, Janek Thomas, Theresa Ullmann, Marc Becker, Anne-Laure Boulesteix, Difan Deng, and Marius Lindauer. Hyperparameter optimization: Foundations, algorithms, best practices and open challenges.

[24] Li Yang and Abdallah Shami. On hyperparameter op-

timization of machine learning algorithms: Theory and practice. *CoRR*, abs/2007.15745, 2020.

[25] Md Riyad Hossain, Douglas Timmer, and Hiram Moya. *Machine learning model optimization with hyper-parameter tuning approach*, volume 21. 2021.

[26] Daniel Berrar. Cross-Validation. January 2018.

[27] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-Validation. In LING LIU and M. TAMER ÖZSU, editors, *Encyclopedia of Database Systems*, pages 532–538. Springer US, Boston, MA, 2009.

[28] 3.1. Cross-validation: Evaluating estimator performance. https://scikit-learn/stable/modules/cross_validation.html. Accessed on 28/12/2023.

[29] Understanding Cross-Validation | Aptech, May 2023. Accessed on 28/12/2023.

[30] Jun Qi, Jun Du, Sabato Marco Siniscalchi, Xiaoli Ma, and Chin-Hui Lee. On mean absolute error for deep neural network based vector-to-vector regression. *IEEE Signal Processing Letters*, 27:1485–1489, 2020.

[31] Tianfeng Chai and R.R. Draxler. Root mean square error (rmse) or mean absolute error (mae)?– arguments against avoiding rmse in the literature. *Geoscientific Model Development*, 7:1247–1250, 06 2014.

[32] Davide Chicco, Matthijs Warrens, and Giuseppe Jurman. The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation. *PeerJ Computer Science*, 7:e623, 07 2021.

[33] Scatter plots and dot plots - IBM Documentation, July 2023.

[34] Line Charts - IBM Documentation, July 2023.

[35] Residual Plot - an overview | ScienceDirect Topics.

[36] seaborn.kdeplot — seaborn 0.13.0 documentation.

[37] Johannes Fürnkranz. Decision Tree. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 263–267. Springer US, Boston, MA, 2010.

[38] Yan-yan SONG and Ying LU. Decision tree methods: Applications for classification and prediction. *Shanghai Archives of Psychiatry*, 27(2):130–135, April 2015.

[39] Time Series Forecasting Using Tree Based Methods. *Journal of Statistics Applications & Probability*, 10(1):229–244, March 2021.

[40] Mohammad Mahmood Otoom. Comparing the Performance of 17 Machine Learning Models in Predicting Human Population Growth of Countries. *International Journal of Computer Science and Network Security*, 21(1):220–225, January 2021.

[41] Shuzhi Wang. Population Prediction Based on Logistic Regression and Random Forest. *Highlights in Science, Engineering and Technology*, 49:496–500, May 2023.

[42] Hiroki Hara, Yoshikatsu Fujita, and Kazuhiko Tsuda. Population estimation by random forest analysis using Social Sensors. *Procedia Computer Science*, 176:1893–1902, January 2020.

[43] Fatih Veli Şahinarslan, Ahmet Tezcan Tekin, and Ferhan Çebi. Application of machine learning algorithms for population forecasting. *International Journal of Data Science*, 6(4):257–270, 2021.

[44] Chian-Yue Wang and Shin-Jye Lee. Regional population forecast and analysis based on machine learning strategy. *Entropy*, 23(6):656, 2021.

[45] K Enwere, E Nduka, and U Ogoke. Comparative analysis of ridge, bridge and lasso regression models in the presence of multicollinearity". *IPS Intelligentsia Multidiscipl J*, 3(1):1–8, 2023.

[46] ML_from_Scratch/decision tree regression.ipynb at master · Suji04/ML_from_-Scratch. https://github.com/Suji04/ML_from_-Scratch/blob/master/decision%20tree%20regression.ipynb.

[47] Machine-Learning-From-Scratch/05 Random Forests at main · AssemblyAI-Examples/Machine-Learning-From-Scratch.

[48] Li Yang and Abdallah Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. 415:295–316.

• • •