VOL 5, NO.1 (March 2025) 111-133
https://jagst.utm.my

JAGST
Journal of Advanced Geospatial Science & Technology

Benchmarking Native Multi-Output, Regressor Chain and TPOT-MTR Models

Hanafi Majid[1,3], Syahid Anuar[2]

[1]Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia
[2]Faculty of Artificial Intelligence, Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia
[3]Malaysia Board of Technologist, 1, Jalan P8G, Presint 8, 62000 Putrajaya, Wilayah Persekutuan Putrajaya, Malaysia

*Corresponding author: hanafimajid@graduate.utm.my

Abstract – Native multi-output expands its capabilities to include multi-target regression using a genetic programming approach. It outperforms state-of-the-art methods on multi-target regression datasets, enhancing prediction accuracy and decision-making in multi-target regression domains. However, it relies heavily on a single target strategy, which may not capture complex interdependencies between multiple targets. Additionally, the systems lack understanding of how linked targets in multi-target regression are handled, making it difficult for practitioners to determine the real connections. The purpose of this study is to evaluate and assess the Native Multi-Output, Regressor Chain and TPOT-MTR model using six relevant public datasets. The methods used were Pearson Correlation and aRRMSE. The research focuses on multi-target regression using AutoML and TPOT, focusing on the TPOT multi-output regression technique. The proposed model is expected to improve correlation and provide customization options, contributing to the advancement of multi-target regression using AutoML techniques. It concludes with a comprehensive analysis of the algorithm's performance in addressing difficulties in multiple target regression and evaluating its ability to identify and employ relevant features for improved predictive correlation. The results show that TPOT-MTR shows stronger correlation performance across most datasets, especially when target relationship capture is vital. However, its performance advantage is reduced in datasets with weakly connected or naturally well-structured targets. In conclusion, TPOT-MTR is a reliable tool for modelling target correlations, especially in multi-target learning tasks. Its performance varies depending on the dataset structure, with Regressor Chain and Native Multi-Output performing better in densely linked datasets. However, it may not always provide the lowest aRRMSE, making it a good substitute for simpler models. Further research can explore more intricate datasets and integrate neural networks.

Keywords – *Multi-Target Regression, AutoML, Machine Learning, Genetic Programming, Multi-Output Regression*

Article History: Received 15 August 2024, Accepted 31 January 2025, Published 28 March 2025

1.0 Introduction

Machine learning has made significant progress in recent years, with Automated Machine Learning (AutoML) being a significant achievement in the field. AutoML has shown advantages in applications like multi-target regression (MTR), which involves simultaneously predicting numerous interconnected output variables. However, the challenge in MTR is effectively handling the relationship between target variables. Conventional methods often fail to consider the specific relationships between each objective, leading to less optimal performance and erroneous forecasts.

AutoML offers a solution by using automation to investigate various models and configurations, minimizing manual labour and ensuring models are fine-tuned to the unique attributes of the data. The Tree-based Pipeline Optimization Tool (TPOT) has shown significant potential in AutoML, focusing on classification problems and genetic programming to create machine-learning pipelines for maximum accuracy. Researchers have expanded TPOT to tackle multi-target regression, creating TPOT-MTR, which uses genetic programming techniques to optimize the entire data preparation pipeline, feature selection, and model training for MTR problems, focusing on handling target correlations.

AutoML is a machine-learning technique used for multi-target regression but doesn't evaluate individual correlations between targets. This gap in correlation issues between targets will cause poor relationships among them and result in less accuracy and performance. Therefore, the proposed TPOT-MTR method combines optimization algorithms and pipeline designs specifically for multi-target regression to address this issue. This innovative approach aims to deliver competitive performance and respect fundamental relationships among targets by considering complex interactions between output variables. TPOT-MTR aims to provide a highly effective solution for complex predictive modelling tasks that require multi-target regression, generating sophisticated and dependable data-driven insights across various fields. This paper presents a comparative analysis of multiple methods, including TPOT-MTR, ERC-SVR, and native multi-output, to provide a better understanding of correlation solutions between targets using different techniques.

Three distinct objectives are established to guide the research effort. These objectives drive the focus and provide a holistic account of multi-target regression using AutoML issues based on the genetic algorithm, relating them to performance-related problems. The following is a summary of the research objectives to ensure that they will be the focus. They are as follows:

a) To conduct a comparative analysis of native multi-output, regressor chain and TPOT-MTR using six public datasets.

b) To highlight key points of performance metrics from multiple methods using six public datasets.

c) To evaluate and assess the method of TPOT-MTR, ERC-SVR and native multi-output model using six public datasets.

This research focuses specifically on multi-target regression using AutoML, particularly on the TPOT multi-output regression technique. The study aims to describe the trained model of multi-target regression using AutoML and TPOT, a supervised machine learning technique. The intention is to provide a clear understanding of the proposed multi-target regression model and its potential benefits for application users in improving their applications' availability. To maintain a clear focus on the research objectives, this study excludes the exploration of other regular machine learning (ML) systems and their features and implementation models.

ML systems' architectural and workflow aspects are also out of scope and will not be discussed in detail. The primary goal is to propose a multi-target regression model using AutoML and TPOT rather than delving into technical aspects related to the architecture and programming of regular ML systems. The research will specifically investigate the utilization of multi-target regression within the AutoML TPOT framework, which is based on a genetic algorithm system. The output of the proposed model is expected to benefit users by improving accuracy and providing the flexibility to customize the model according to their application preferences. Focusing on the multi-target regression model in AutoML TPOT, this research aims to provide insights and a foundation for future improvements in multi-target regression using AutoML techniques.

Overall, the research scope centers on describing and analyzing a trained multi-target regression model using AutoML based on the TPOT technique. The focus is on understanding the potential benefits of this model for application users, improving accuracy, and providing customization options. The research aims to contribute to advancing multi-target regression within the AutoML domain, specifically in the context of TPOT and genetic algorithms.

The current level of multi-target regression can be found using regular machine learning algorithms. Regression problems, called multi-output or multi-target regression, aim to predict several continuous target variables. The complexity of the issue, the amount and quality of the dataset, the selection of algorithm and hyperparameters, and the evaluation measure employed all affect the degree of performance for multi-target regression.

The strength of this research lies in the proposed model, which facilitates AutoML, and this model will be justified in ensuring the effectiveness of the regression model. The proposed model has several advantages, including greater accuracy and efficiency in error-rated metrics. This will help analysts to:

- Make a good prediction because this proposed model facilitates the AutoML with multi-target regression.
- Chances of rechecking the hyperparameter to tune up the efficiency and accuracy based on the dataset.
- Mitigate potential harmful effects of unexpected issues on a running pipeline.

In developing the model process, gaps will inevitably be bridged by focusing on multi-target regression using AutoML-TPOT and making recommendations to further improve and achieve optimal accuracy in managing efficiency, which is the significance of the research. The significance of this study is demonstrated by the evaluation of the proposed model, which benefits and provides value-added to the multi-target regression mechanism in AutoML-TPOT, specifically based on the genetic programming AutoML system. The prediction-trained model can initiate the following action to make a prediction and inadvertently provide an early warning signal to any proposed systems, thereby preventing unwanted events.

1.1 *Study Gap*

Current AutoML systems are designed for single-target strategy, and current multi-target regression challenges the correlation between targets. TPOT-MTR, an AutoML system, addresses this by expanding its capabilities of correlating between targets to include multi-target regression. Using a genetic programming approach, TPOT-MTR finds the optimal mix of machine learning algorithms and hyperparameters for multi-target regression tasks. It outperforms state-of-the-art methods on various multi-target regression datasets, suggesting its potential to enhance prediction accuracy and facilitate decision-making in domains where multi-target regression is common.

Many steps have been taken to improve multi-target regression by solving association problems, but some study gaps still need to be filled. Existing multi-target regression systems, such as native multi-output, focus heavily on the single-target strategy, which may not always capture the complex interdependencies between multiple targets effectively. Despite native multi-output and regressor chains making predictions more accurate, they do not directly model the statistical relationships between linked targets. This means that it might not work well in datasets where target factors are strongly dependent on each other.

Another significant gap is that AutoML systems don't make it easy to understand how they handle linked targets in multi-target regression. There are numerous current methods, such as native multi-output and regressor chains, that focus on predictive correlation but don't demonstrate how the goal factors interact with each other. Because of this, it is challenging for practitioners to determine if the connections found are genuine or simply errors made by the automatic modelling process. Future studies should investigate methods that integrate domain knowledge with advanced feature engineering techniques to better handle correlated goals.

2.0 Materials and Methods

2.1 Automated Machine Learning (AutoML)

Automated Machine Learning (AutoML) is frequently commended for its capacity to simplify the machine-learning process by decreasing the time and expertise necessary to develop and deploy models (Chauhan et al., 2020). AutoML endeavours to expedite model development while minimizing human intervention by deconstructing the conventional machine-learning pipeline into smaller automated modules. Nevertheless, this automation has drawbacks, notably in complex data preprocessing and feature engineering, which necessitate domain-specific expertise despite its efficiency. Additionally, the quality and nature of the input data are critical factors in the efficacy of AutoML tools, which can automate model selection, hyperparameter tuning, and optimization. This presents a substantial challenge for real-world applications.

AutoML can open up machine learning to businesses that may not have extensive technical expertise, making it easier for them to adopt AI-driven solutions. By streamlining key steps such as selecting models and fine-tuning parameters, AutoML accelerates the onboarding process for various industries, making it easier for anyone to engage with machine learning to address problems. AutoML can enhance model correctness by testing multiple model designs; however, its automated nature may lead to suboptimal selections if not closely monitored. User-friendliness and model development control are balanced in fully automated and semi-automated AutoML systems.

Fully automated systems reduce human input; however, they may not be flexible enough for complex jobs. Semi-automated solutions offer more flexibility but need expertise, making them less accessible to non-experts.

Although AutoML appears to be a viable approach to operationalizing machine learning tools on a broader scale, it still has a long way to go before it can be practically applied. A study by Krauß et al. (2020) discusses the challenges of incorporating AutoML into production environments and its performance, which is still not as effective as that of data science specialists. AutoML can handle some crucial steps in the machine learning process, like data preparation, modelling, deployment, and integration. But it’s not flexible enough for complicated tasks because it can’t adapt to different domains. It doesn’t work well in situations like independent learning and reinforcement learning, where human help is still needed. AutoML also struggles with high-dimensional and diverse data, which is a common issue in real-world applications. This indicates that it requires further improvements to be more easily understood, scalable, and overall better.

A study by Krauß et al. (2020) also highlights the potential and limitations of AutoML while comparing it to human data science experience. It critically examines the factors influencing the success of ML projects in production. Data integration, modelling, and deployment are only a few of the ML pipeline processes that AutoML automates. However, the effectiveness of AutoML still relies heavily on the skill and judgment of production experts. A significant obstacle noted is the lack of agreement on the best method for assessing AutoML systems, as their effectiveness depends much on usability, toolkit breadth, and domain expertise. By grouping AutoML stakeholders into academic, business, and data preparation categories, the scattered nature of AutoML adoption is highlighted, and various user groups are suggested to prioritize different needs. AutoML is limited, even though it has great potential to increase the output of ML projects. It cannot entirely replace human decision-making, especially in jobs that require sophisticated knowledge of industry-specific needs.

Automated Machine Learning (AutoML) and Metalearning are emerging fields that aim to reduce human reliance on machine learning models. They have a long way to go. AutoML automatically searches for neural architectures to speed up model creation, but it can’t handle complex, domain-specific tasks; therefore, it’s only suitable for typical datasets. On the other hand, meta-learning enhances learning speed by leveraging information from multiple datasets. However, it remains challenging to apply in real-life scenarios because it requires substantial computing power and exhibits issues with generalization. A study by Doke and Gaikwad (2021) discusses the ongoing efforts to integrate these two methodologies, demonstrating the potential for enhanced automation. AutoML asserts that it will reduce the necessity for data scientists; however, domain knowledge is still required to ensure that machine learning applications are comprehensible and dependable. This is remarkably accurate in the actual world, where the complexity and variability of data present significant challenges.

However, not all machine learning issues can be solved with AutoML. Machine learning practitioners' skills and domain knowledge are still crucial for the quality of data used to train models and the quality of the models themselves. AutoML systems can boost ML project productivity, but application-domain knowledge and domain-specific expertise must be incorporated for desired outcomes.

2.2 Multi-Target Regression (MTR)

A generic data-transformation methodology is suggested for MTR feature ranking, which includes two variations of each score. The findings identify the factors that influence the quality of the rankings and demonstrate that both groups of approaches yield significant feature rankings. Petković et al. (2020) suggest a universal technique for sorting MTR features using RReliefF and ensemble-based scoring. The study employs 24 MTR benchmark problems, characterized by characteristics ranging from 6 to 576 and objectives ranging from 2 to 16. The instances used in the study vary in number from 103 to 60607. The results indicate that the ensemble technique is the most suitable for a specific relevance score in ensemble-based feature ranking, and the MTR rankings may be compared to their single target (STR) equivalents.

Using ensembles of regressors, where multiple regression models are trained independently on the same input characteristics and their outputs are combined to predict the numerous targets, is a widely adopted method for MTR in AutoML. Numerous applications, including the prediction of several water quality indices in environmental science, have demonstrated the efficacy of this method. For example, forecasting the synthesis of secondary metabolites in fungi (life sciences), learning habitat models for a variety of species, and predicting forests are all examples of real-world problems that may be phrased as MTR tasks (Breskvar & Dzeroski, 2020).

The article "Multi-Target Regression via Input Space Expansion" by Spyromitros-Xioufis et al. (2016) suggests a novel method for MTR that entails expanding the input feature space to include all potential pairwise interactions between features and then training a single regressor on the expanded feature space. Using satellite data to anticipate various environmental factors, the authors show how good their method is.

Previous studies, such as those (Breskvar & Dzeroski, 2020) propose expanding the functionality of a method called FIRE for learning multi-target regression rules by incorporating random output selections (ROS) into the learning process of tree ensembles. Individual Predictive Clustering Trees (PCT)s are limited to analyzing a small subset of the target variables in such ensembles. The rules gleaned from the tree ensemble similarly narrow in on specific groups of the dependent variables (FIRE-ROS).

The latest research highlights the substantial influence of TPOT-MTR and other AutoML techniques in improving the performance of multi-target regression (MTR) models (Majid, Anuar, and Hassan, 2023). By efficiently capturing the interconnections among the variables of interest, these methods can generate more precise and dependable predictions, leading to progress in MTR and its practical implementations (Abdallah, Grati, and Boukadi, 2023). The increasing demand for advanced predictive models emphasizes the significance of ongoing enhancement and advancement of AutoML tools to tackle the difficulties in MTR (Gawalska et al., 2023). Feature Ranking in MTR (Petković, Džeroski, & Kocev, 2020) presents a universal MTR feature ranking system based on RReliefF and ensemble-based scoring. It has been tested on 24 benchmark sets, showing generalizability. It lacks comparison with deep learning methods, which are becoming more popular in MTR. Future studies aimed at better performance should investigate neural network-driven feature ranking.

Environmental and biological sciences extensively apply multi-target regression, as shown in Breskvar and Dzeroski (2020). It forecasts forest models, species habitats, and water quality metrics. AutoML finds high-dimensional, sparse, or unbalanced data challenging. The black-box character of ensemble models lowers interpretability and calls for explainable artificial intelligence techniques in terms of expanding features and alternative MTR techniques.

Input feature space extension suggested by Spyromitros-Xioufis et al. (2016) would help MTR performance. The drawback is that computational cost rises dramatically, and huge datasets cannot be viable. The FIRE-ROS approach (Breskvar & Dzeroski, 2020) restricts the number of target variables investigated while nevertheless helping to reduce complexity. Although further study should focus on balancing accuracy, efficiency, and scalability, TPOT-MTR (Majid, Anuar, & Hassan, 2023) and AutoML approaches show some potential to address this issue.

2.3 Dataset

The datasets utilized in the experiments are concisely presented below.

2.3.1. Jura

The Jura (Swan, 1998) database contains measurements of 7 heavy metals: chromium, nickel, cadmium, zinc, cobalt, lead, and copper. These measurements are taken at 359 distinct locations within a specific region in Switzerland. Each area is assessed for its land usage, including meadow, forest, tillage, pasture, and the type of rock present, such as Quaternary, Argovian, Portlandian, Sequanian, or Kimmeridgian. In a multi-target regression context, the objective is to forecast the concentration of higher-value metals, which are regarded as the main variables, based on the measurements of lower-value metals used as input variables. This study examines copper, lead, and cadmium as the major focus, but all other metals, land usage type, rock type, and location, are used as predictive inputs.

2.3.2. Slump

The concrete slump database (Yeh, 2007) predicts three attributes of concrete: flow, slump, and compressive strength. These attributes are considered dependent vector variables and are influenced by seven concrete components: blast furnace slag, superplasticizer, cement, water, fly ash, fine aggregate, and coarse aggregate.

2.3.3. Andro

Andro database (Hatzikos et al., 2008) focuses on predicting six future water quality parameters in Thessaloniki, Greece: oxygen levels, pH, temperature, salinity, turbidity, and conductivity. The target variable records are acquired from subaquatic sensors with a sample interval of nine seconds. The measurements are averaged to derive a single record for each variable per day. The typically utilized database is constructed by employing a five-day time window. The attributes pertain to six water quality measurements taken up to five days prior, with a time lag of five days. In other words, the expected values for each variable for the next six days are determined.

2.3.4. Electrical Discharge Machining (EDM)

The Electrical Discharge Machining database (Karalič and Bratko, 1997) is used for a regression job with two outcome variables. This dataset aims to enhance the machining speed by imitating the behaviours of a human operator who oversees two output responses. The output can have three distinct numeric values: -1, 0, or 1. There are sixteen continuous input features available.

2.3.5. ENB

The Energy Building database (Tsanas and Xifara, 2012) focuses on addressing the issue of energy efficiency by forecasting the heating and cooling load requirements of buildings based on eight characteristics, including roof area, total height, and glazing area, among others.

2.3.6. SCM20d

The code SCM20d refers to the supply chain management database compiled from the Trading Agent Competition in the supply chain management competition. The data pretreatment and normalization techniques are explained in depth in reference (Groves and Gini, 2013). The SCM20d dataset pertains to the forecast of the "Product Future" category. Each row in the dataset corresponds to a single day of observation in the tournament, which spans 220 days and consists of eighteen games. The input characteristics represent the recorded prices for a single day in the event. Furthermore, four samples with a time delay are included for each observed component and product to aid in predicting the current trends. The SCM20d dataset provides the average price for each commodity over twenty consecutive days.

3.0 Methodological Steps

The methodological steps will be explained in three steps: selecting the data sources, preparing the chosen model, and lastly, model assessment.

Step 1: Selecting the Data Sources and Approaches

The first stage is finding and choosing pertinent data sources from six public datasets utilized for the analysis. The data's quality, dependability, and completeness must be carefully evaluated to ensure its fit for the research. Ensuring that the dataset is used in current and past studies is also important to compare its results. We also consider data availability, ethical issues, and congruence with the study goals. Organizing and preprocessing the data for additional examination comes next after the data sources are complete.

Step 2: Preparation of the Selected Model

Once the data sources and methods of approach have been chosen, the model ready for analysis must be developed. Data cleansing, addressing missing values, and variable transformation to guarantee they are in the right shape for modelling are among these aspects. After that, the model parameters are chosen based on empirical results or theoretical frameworks; they remain constant across several methods, ensuring their accurate depiction of the topic under examination. Furthermore, features engineering and data normalizing techniques might be used to enhance the model's prediction performance. Finally, the model is trained on the chosen dataset, and the first validation is performed to evaluate its preliminary performance.

```
BEGIN
   # Define MultiOutputTP class
   CLASS MultiOutputTP:
      FUNCTION __init__(self, *args,  kwargs):
         self.args ← args
         self.kwargs ← kwargs

      FUNCTION fit(self, X, y):
         # Ensure X and y are 2D arrays
         Convert X, y to 2D format

         # Ensure X and y have the same number of
rows
         IF number_of_rows(X) ≠ number_of_rows(y):
            THROW ERROR

         # Determine number of target variables (yy)
         yy ← number_of_columns(y)

         # Initialize empty list for regressors
         self.regs ← [ ]

         RETURN self

END
```

Figure 1: Pseudocode for Initial Deployment without TPOT and Predict Function

```
BEGIN
   # Define MultiOutputTP class
   CLASS MultiOutputTP:
      FUNCTION __init__(self, *args,  kwargs):
         self.args ← args
         self.kwargs ← kwargs

      FUNCTION fit(self, X, y):
         # Ensure X and y are 2D arrays
         Convert X, y to 2D format

         # Ensure X and y have the same number of
s
         IF number_of_rows(X) ≠ number_of_rows(y):
            THROW ERROR

         # Determine number of target variables (yy)
         yy ← number_of_columns(y)

         # Initialize empty list for regressors
         self.regs ← [ ]

         # Train individual TPOTRegressor models
for each target
         FOR i FROM 0 TO yy - 1:
            Define REG as TPOTRegressor with:
               generations = 5
               population_size = 50
               verbosity = 2
               random_state = 123
               max_time_mins = None
               max_eval_time_mins = 5
               cv = 10
               scoring = 'neg_mean_squared_error'
            # Prepare training data for target i
            Xi ← concatenate(X with first i columns of
y)
            yi ← column i of y

            # Fit the model and store it
            Append REG.fit(Xi, yi) to self.regs

         RETURN self
END
```

Figure 2: Pseudocode with TPOT and without Predict Function

```
BEGIN
   # Define MultiOutputTP class
   CLASS MultiOutputTP:
      FUNCTION __init__(self, *args,  kwargs):
         self.args ← args
         self.kwargs ← kwargs

      FUNCTION fit(self, X, y):
         # Ensure X and y are 2D arrays
         Convert X, y to 2D format

         # Ensure X and y have the same number of
s
         IF number_of_rows(X) ≠ number_of_rows(y):
            THROW ERROR

         # Determine number of target variables (yy)
         yy ← number_of_columns(y)

         # Initialize empty list for regressors
         self.regs ← [ ]

         # Train individual TPOTRegressor models
for each target
         FOR i FROM 0 TO yy - 1:
            Define REG as TPOTRegressor with:
               generations = 5
               population_size = 50
               verbosity = 2
               random_state = 123
               max_time_mins = None
               max_eval_time_mins = 5
               cv = 10
               scoring = 'neg_mean_squared_error'
            # Prepare training data for target i
            Xi ← concatenate(X with first i columns of
y)
            yi ← column i of y

            # Fit the model and store it
            Append REG.fit(Xi, yi) to self.regs

         RETURN self

      FUNCTION predict(self, X):
         # Initialize output array for predictions
         Define y as empty matrix with dimensions
(X.shape[0], length(self.regs))
         # Generate predictions for each target
variable
         FOR i, REG in enumerate(self.regs):
            y[:, i] ← REG.predict(concatenate(X with
first i columns of y))

         RETURN y
   # Instantiate MultiOutputTP and train model
   Define REG2 as MultiOutputTP(1)
   Fit REG2 on (X_train, y_train)

   # Make predictions
   Define YPRED2 as REG2.predict(X_test)

END
```

Figure 2: Pseudocode with TPOT and Predict Function

The pseudocode for the initial deployment without TPOT and the predicted function are illustrated in Figure 1. The deployment of the iterative algorithm during this phase is the primary objective, as it is designed to continue perusing the next target of outputs. However, external elements should not disrupt this phase, as this will increase the algorithm's complexity and the number of challenges. The algorithm's fundamental structure, including the definition of pipeline fitting classes, should be sufficient. The subsequent phase involves the development of a pseudocode that is both TPOT-enabled and does not include the predict function, as illustrated in

Figure 2. The primary objective of the deployment during this phase was to incorporate TPOT into the algorithm.

Furthermore, the appropriate hyperparameter was selected and incorporated into the pipelines. The fitting algorithm was manually tested several times to determine the most suitable parameters for the datasets. The pseudocode with TPOT, which includes the predict function, is illustrated in Figure 3 to provide a more comprehensive understanding of the results. This phase should be exhaustive, incorporating all aspects of multi-target regression. The prediction result can also be transferred to the subsequent step, model assessment. It includes and selects all pertinent metrics for the evaluation.

Step 3: Model Assessment

Once the model is prepared, it undergoes rigorous assessment to determine its effectiveness and reliability. Various performance metrics, including mean absolute correlation, average relative root mean square error (aRRMSE), and a Pearson Correlation Heatmap, are used to evaluate the model's predictive capability. Cross-validation techniques will be applied to ensure the model generalizes well to new data and is not overfitting. Additionally, sensitivity analysis can be conducted to examine how variations in input data impact model outcomes. Based on the assessment results, necessary adjustments will be made to improve the model before its final implementation.

3.1 Methods

The instrumentation consists of the tools, frameworks, and techniques used to implement and evaluate the machine learning models. The study utilized publicly available benchmark datasets, including those from the UCI Machine Learning Repository and Kaggle. Preprocessing tools, such as Pandas, NumPy, and Scikit-learn, were used to clean, normalize, and split the dataset into training and testing sets.

Scikit-learn was used to develop and implement different machine learning models. Algorithms to be compared may include supervised models (Decision Trees, Random Forest, SVM, Neural Networks) and unsupervised models (PCA). Performance was measured using standard evaluation metrics such as aRRMSE and Mean Absolute Correlation, depending on the type of machine learning task. Cross-validation techniques such as k-fold cross-validation will be applied to ensure robustness and generalizability. Statistical tools such as Pearson Correlation Analysis may be used to determine significant differences between models. Visualization tools like Matplotlib and Seaborn will be used to represent results graphically.

The study's design and instrumentation framework ensure a systematic and objective comparison of different machine-learning approaches while maintaining reproducibility and reliability in the findings. Table 1 below presents the pseudocode for the native multi-output regressor chain and TPOT-MTR, which are also part of the selected model's preparation. Native multi-output and regressor chains are the existing multi-target regression library retrieved from the scikit-learn, which is also widely used in multi-target regression models. To compare, the proposed TPOT-MTR, as shown in the figure below, utilizes the current TPOT library but is enhanced with a multi-target wrapper, differing from the current native multi-output and regressor chain. The table below outlines the key points regarding the three approaches to simplify the differences.

Table 1: Comparison Table Between Native Multi-Output, Regressor Chain and TPOT-MTR

Feature	MultiOutputRegressor	RegressorChain	TPOT-MTR (Multi-Target Regression with TPOT)
Modelling Targets	Independent models for each target	Sequential models with dependencies	Sequential models with automated feature selection & optimization
Workflow	Parallel	Sequential	Sequential with automated pipeline tuning
Interdependency	Ignores relationships	Model's relationships	Model's relationships dynamically using genetic algorithms
Speed	Faster (no dependencies)	Slower (sequential nature)	Slower due to automated model selection & hyperparameter tuning
Error Propagation	None	Possible	Possible but mitigated through adaptive optimization
Scalability	Scales well for many targets	May struggle with many targets	Handles many targets but may be computationally expensive

Figure 4 below shows pseudocode for a native multi-output design, intended as an automated machine learning (AutoML) tool that utilizes evolutionary algorithms to optimize model selection and hyperparameter tuning; the Multi-Output Regressor wraps the TPOT Regressor. Negative mean squared error (neg_MSE) is the evaluation metric; the model achieves strong performance using a 10-fold cross-validation (cv=10). A TransformedTargetRegressor manages target variable transformations, guaranteeing appropriate scaling of y_train and y_pred. Once trained, the model produces predictions for X_test from X_train and y_training. For challenging multi-target regression projects, this method combines the adaptability of multi-output regression with the force of automated model tweaking.

```
BEGIN
  # Initialize a multi-output regressor with TPOTRegressor
  Define RFG as MultiOutputRegressor(
    TPOTRegressor with parameters:
      generations = 5
      population_size = 50
      verbosity = 2
      random_state = 123
      n_jobs = 1
      max_time_mins = None
      max_eval_time_mins = 5
      cv = 10
      scoring = 'neg_mean_squared_error'
  )

  # Create a TransformedTargetRegressor with target transformation
  Define MODEL as TransformedTargetRegressor(
    regressor = RFG
    transformer = target_transformer
  )

  # Fit the model to the training data
  MODEL = RFG.fit(X_train, y_train)

  # Make predictions on the test set
  Y_PRED = MODEL.predict(X_test)

END
```

Figure 4: Pseudocode for Native Multi Output

Starting with TPOT as the basic model, the Regressor Chain method, as shown below in Figure 5, ignores minor mistakes within a specified margin using epsilon-insensitive loss. Its dual-mode operation guarantees the best performance, and a high repetition limit helps to achieve appropriate convergence. After that, the RegressorChain uses a sequential modelling method wherein an order of [0,1,2] chains forecasts together. This implies that every target variable is expected one after the other, with each next target including the forecast of the prior target as a supplementary element. A TransformedTargetRegressor scales y_train and y_pred2 to ensure the target variables are appropriately converted. Finally, the model is trained using X_train and y_train, and the structural relationships between target variables generate predictions for the X_test.

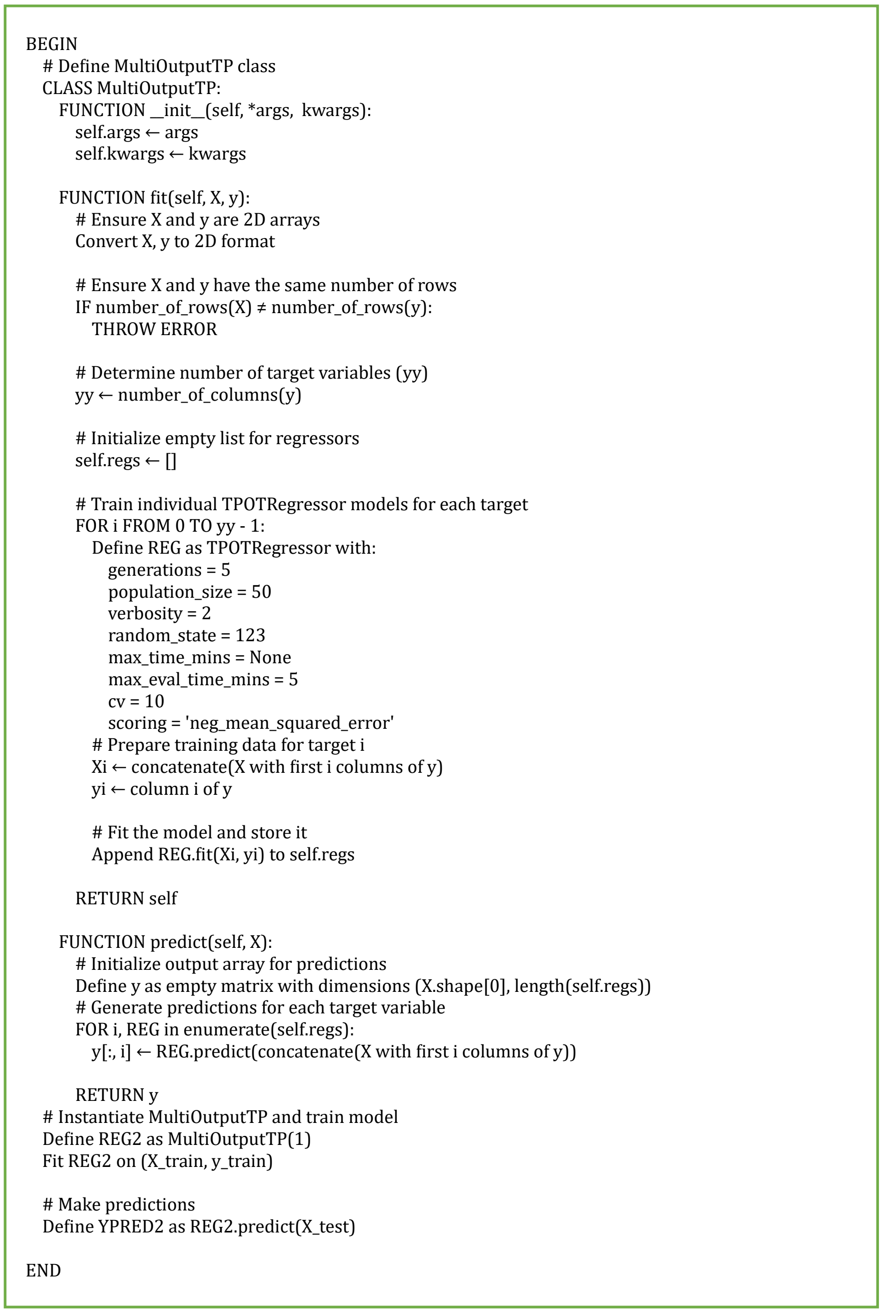
```
BEGIN
# Define the base regressor as LinearSVR
Define REG2 as LinearSVR with parameters:
epsilon = 0.0
tol = 0.0001
C = 1.0
loss = 'epsilon_insensitive'
fit_intercept = True
intercept_scaling = 1.0
dual = True
verbose = 2
random_state = None
max_iter = 100000000

# Initialize RegressorChain using LinearSVR as the base estimator
Define CHAIN as RegressorChain(
base_estimator = REG2
order = [0,1,2]
)

# Fit the regressor chain model to the training data
CHAIN.fit(X_train, y_train)

# Create a TransformedTargetRegressor with target transformation
Define MODEL as TransformedTargetRegressor(
regressor = CHAIN
transformer = target_transformer
)

# Train the transformed model on the training data
MODEL.fit(X_train, y_train)

# Make predictions on the test set
Y_PRED2 = MODEL.predict(X_test)

END
```

Figure 5: Pseudocode for Regressor Chain

The custom multi-output regression method TPOT-MTR (Multi-Output TPOT Regressor), as shown in Figure 6 below, trains independent TPOT Regressor models for each target variable. By traversing all target variables and training separate models, the Multi-OutputTP class is designed to handle this procedure. The approach guarantees that X and y are in 2D format throughout the fit step. It then runs over every column of y, adding previously handled target columns (y[:, :i] to augment the input X. For every target variable, a separate TPOTRegressor is trained; the trained models are kept on hand for future use.

The prediction method starts an empty matrix to hold forecasts. The approach iteratively forecasts each goal sequentially among the trained models. Previously projected values (y[:, :i]) are used as extra characteristics for further predictions, guaranteeing a structured link between target variables. Finally, a multi-transfer learning example called reg2 is developed and trained on

X_train and y_training. Leveraging the learnt relationships across several target variables, it provides predictions for X_test once trained.



```
BEGIN
  # Define MultiOutputTP class
  CLASS MultiOutputTP:
    FUNCTION __init__(self, *args,  kwargs):
      self.args ← args
      self.kwargs ← kwargs

    FUNCTION fit(self, X, y):
      # Ensure X and y are 2D arrays
      Convert X, y to 2D format

      # Ensure X and y have the same number of rows
      IF number_of_rows(X) ≠ number_of_rows(y):
        THROW ERROR

      # Determine number of target variables (yy)
      yy ← number_of_columns(y)

      # Initialize empty list for regressors
      self.regs ← []

      # Train individual TPOTRegressor models for each target
      FOR i FROM 0 TO yy - 1:
        Define REG as TPOTRegressor with:
          generations = 5
          population_size = 50
          verbosity = 2
          random_state = 123
          max_time_mins = None
          max_eval_time_mins = 5
          cv = 10
          scoring = 'neg_mean_squared_error'
        # Prepare training data for target i
        Xi ← concatenate(X with first i columns of y)
        yi ← column i of y

        # Fit the model and store it
        Append REG.fit(Xi, yi) to self.regs

      RETURN self

    FUNCTION predict(self, X):
      # Initialize output array for predictions
      Define y as empty matrix with dimensions (X.shape[0], length(self.regs))
      # Generate predictions for each target variable
      FOR i, REG in enumerate(self.regs):
        y[:, i] ← REG.predict(concatenate(X with first i columns of y))

      RETURN y
  # Instantiate MultiOutputTP and train model
  Define REG2 as MultiOutputTP(1)
  Fit REG2 on (X_train, y_train)

  # Make predictions
  Define YPRED2 as REG2.predict(X_test)

END
```

Figure 6: Pseudocode for TPOT-MTR

4.0 Results and Discussion

The results for six datasets are explained further, including error metrics such as Mean Absolute Correlation and aRRMSE.

Table 2: Mean Absolute Correlation Result for Six Datasets

Dataset Name	Native Multi Output	Regressor Chain	TPOT-MTR
Jura	0.658	*0.775*	***0.815***
Slump	*0.546*	*0.593*	***0.580***
Andro	***0.535***	*0.525*	*0.511*
EDM	*0.510*	*0.602*	***0.633***
ENB	*0.990*	*0.992*	***0.994***
SCM20D	*0.649*	*0.666*	***0.681***

Table 2, which presents the Mean Absolute Correlation findings, provides information on the degree of link capture between various target variables across six datasets using each technique. Stronger interdependencies across targets, as indicated by a higher mean absolute correlation value, suggest a more effective modelling strategy for capturing these interactions. TPOT-MTR regularly gets the most outstanding correlation values across all datasets, proving its efficacy in modelling target connections. For the Jura dataset, TPOT-MTR, for instance, obtains a mean absolute correlation of 0.815, surpassing both Native Multi-Output (0.658) and Regressor Chain (0.775). This trend is also shown in the EDM dataset. TPOT-MTR achieves 0.633, compared to Regressor Chain (0.602) and Native Multi-Output (0.510), indicating its ability to capture interdependence among multiple target variables more effectively.

Not all datasets, nevertheless, demonstrate a notable benefit for TPOT-MTR*. While in the Andro dataset, Native Multi-Level leads marginally (0.535) over Regressor Chain (0. 525), Regressor Chain performs better in the Slump dataset, 0.593 against 0.580. These results imply that the advantages of TPOT-MTR may be less noticeable in datasets with smaller target interdependencies. All three techniques perform similarly for datasets with extremely high correlations, including ENB; TPOT-MTR (0.994) shows only a nominal advantage over Regressor Chain (0.992) and Native Multi-Output (0.990). This implies that model selection may not significantly impact performance when target variables are highly correlated.

TPOT-MTR performs better across most datasets, especially when target relationship capture is vital. Its performance advantage, however, diminishes in datasets where targets are either weakly connected or naturally well-structured, suggesting that the optimal approach may depend on the type of dataset rather than a one-size-fits-all solution.

Table 3: aRRMSE Result Using Three Different Approaches

Dataset Name	Native Multi Output	Regressor Chain	TPOT-MTR
Jura	**0.020**	0.021	0.052
Slump	**0.019**	0.021	0.051
Andro	0.017	**0.010**	0.012
EDM	0.120	**0.112**	0.149
ENB	**0.001**	0.003	0.007
SCM20D	**0.000**	**0.000**	0.001

Table 3 shows the aRRMSE result for six datasets using three approaches. Regressor Chain has the lowest ARRMSE (0.010), followed by TPOT-MTR (0.012) and finally, Native Multi-Output (0.017) for the Andro dataset. Here, TPOT-MTR performs well, almost matching the Regressor Chain and well above the Native Multi-Tracker. This implies that TPOT-MTR gains by modelling interactions between targets, making it more competitive in datasets with output dependencies. Regressor Chain (0.112) performs the best in the EDM dataset; Native Multi-Output (0.120) follows, with TPOT-MTR earning the most significant error (0.149). This suggests that TPOT-MTR suffers in this dataset because its evolutionary optimization technique does not adequately capture the fundamental linkages.

Native multi-output routinely gets the best results (0.001 and 0.000, respectively) for datasets with very low ARRMSE values, including ENB and SCM20D. TPOT-MTR remains competitive (0.007 and 0.001), even though it lags somewhat. These results demonstrate that all three approaches achieve nearly ideal performance when the dataset structure allows for highly accurate predictions. Even if TPOT-MTR does not get the lowest ARRMSE, overall, it stays competitive across datasets. In situations like Andro, where it beats Native Multi-Order and closely follows the Regressor Chain, its performance is very noteworthy. On datasets like EDM, where conventional techniques excel, TPOT-MTR's evolutionary approach may struggle to match their accuracy. Notwithstanding these differences, TPOT-MTR appears to be a suitable alternative in several situations, particularly for complex target interactions.

5.0 Conclusion

On performance, the ARRMSE results show another view, nevertheless. Although TPOT-MTR remains competitive in some situations, it does not consistently yield the lowest error. TPOT-MTR has the highest ARRMSE (0.149) in datasets such as EDM; Regressor Chain (0.112) and Native Multi-Output (0.120) perform better here. This suggests that TPOT-MTR's evolutionary optimization strategy may occasionally fail to achieve optimal performance in datasets where simpler models can effectively represent the target-output connections. Still, TPOT-MTR is a good substitute, mainly when information on target correlations is more crucial than the lowest possible error.

Additionally significant is the fact that the efficacy of TPOT-MTR varies depending on the dataset structure. With TPOT-MTR achieving 0.994 compared to 0.992 (Regressor Chain) and 0.990 (Native Multi-Output) in datasets with strong natural correlations between targets, such as ENB, the performance variations between all three approaches become insignificant. This implies that the choice of technique could have little effect on performance in densely linked datasets. On the other hand, TPOT-MTR offers no appreciable benefit over the other two methods in datasets with less interdependencies, such as Andro (MAC: 0.511) and Slump (MAC: 0.580). These results indicate that TPOT-MTR is most suitable for datasets where establishing relationships between targets is a priority, rather than solely minimizing prediction errors.

TPOT-MTR generally satisfies its goal of offering a solution for modelling target correlations; hence, it is beneficial for multi-target learning tasks where knowledge of interdependencies is essential. Although it may not always yield the lowest ARRMSE, its ability to model complex connections between outputs makes it a valuable tool when conventional methods are challenging. The results suggest that selecting the appropriate regression technique should depend on the dataset's requirements; if reducing error is the primary goal, Regressor Chain or Native Multi-Order may be preferred. However, TPOT-MTR appears to be an effective and dependable tool when the goal is to analyze and utilize target correlations for enhanced predictions.

Acknowledgement

This study is not funded by any organization.

Conflict of Interest

The authors declare no conflict of interest.

References

Abdallah, Emna Ben, Rima Grati, and Khouloud Boukadi. 2023. “Towards an Explainable Irrigation Scheduling Approach by Predicting Soil Moisture and Evapotranspiration via Multi-Target Regression.” Journal of Ambient Intelligence and Smart Environments 15 (1). doi:10.3233/AIS-220477.

Breskvar, Martin, and Saso Dzeroski. 2021. “Multi-Target Regression Rules with Random Output Selections.” IEEE Access 9. Institute of Electrical and Electronics Engineers Inc.: 10509–22. doi:10.1109/ACCESS.2021.3051185.

Chauhan, Karansingh, Shreena Jani, Dhrumin Thakkar, Riddham Dave, Jitendra Bhatia, Sudeep Tanwar, and Mohammad S. Obaidat. 2020. “Automated Machine Learning: The New Wave of Machine Learning.” In 2nd International Conference on Innovative Mechanisms for Industry Applications, ICIMIA 2020 - Conference Proceedings. doi:10.1109/ICIMIA48430.2020.9074859.

Doke, Ashwini; Gaikwad, Madhava. 2021. “Survey on Automated Machine Learning (AutoML) and Meta learning” 12th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE. doi:10.1109/ICCCNT51525.2021.9579526

Gawalska, Alicja, Natalia Czub, Michał Sapa, Marcin Kołaczkowski, Adam Bucki, and Aleksander Mendyk. 2023. “Application of Automated Machine Learning in the Identification of Multi-Target-Directed Ligands Blocking PDE4B, PDE8A, and TRPA1 with Potential Use in the Treatment of Asthma and COPD.” Molecular Informatics 42 (7). doi:10.1002/minf.202200214.

Groves, William, and Maria Gini. 2013. “Improving Prediction in TAC SCM by Integrating Multivariate and Temporal Aspects via PLS Regression.” In Lecture Notes in Business Information Processing. Vol. 119 LNBIP. doi:10.1007/978-3-642-34889-1_3.

Hatzikos, Evaggelos V., Grigorios Tsoumakas, George Tzanis, Nick Bassiliades, and Ioannis Vlahavas. 2008. “An Empirical Study on Sea Water Quality Prediction.” Knowledge-Based Systems 21 (6). doi:10.1016/j.knosys.2008.03.005.

Karalič, Aram, and Ivan Bratko. 1997. “First Order Regression.” Machine Learning 26 (2–3). doi:10.1023/a:1007365207130.

Krauß, J.; Pacheco, B. M.; Zang, H. M.; Schmitt, R.H.. 2020. “Automated machine learning for predictive quality in production” Procedia CIRP. doi: 10.1016/j.procir.2020.04.039.

Majid, H., S. Anuar, and N.H. Hassan. 2023. “TPOT-MTR: A Multiple Target Regression Based on Genetic Algorithm of Automated Machine Learning Systems.” Journal of Advanced Research in Applied Sciences and Engineering Technology 30 (3). doi:10.37934/araset.30.3.104126.

Petković, Matej, Dragi Kocev, and Sašo Džeroski. 2020. “Feature Ranking for Multi-Target Regression.” Machine Learning 109 (6). doi:10.1007/s10994-019-05829-8.

Rodríguez, Juan J., Mario Juez-Gil, Carlos López-Nozal, and Álvar Arnaiz-González. 2022a. “Rotation Forest for Multi-Target Regression.” International Journal of Machine Learning and Cybernetics 13 (2). doi:10.1007/s13042-021-01329-1.

———. 2022b. “Rotation Forest for Multi-Target Regression.” International Journal of Machine Learning and Cybernetics 13 (2). doi:10.1007/s13042-021-01329-1.

Spyromitros-Xioufis, Eleftherios, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. 2016. “Multi-Target Regression via Input Space Expansion: Treating Targets as Inputs.” Machine Learning 104 (1). Springer New York LLC: 55–98. doi:10.1007/s10994-016-5546-z.

Swan, Andy. 1998. “ G OOVAERTS , P. 1997. Geostatistics for Natural Resources Evaluation . Applied Geostatistics Series. Xiv + 483 Pp. New York, Oxford: Oxford University Press. Price £46.95 (Hard Covers). ISBN 0 19 511538 4. .” Geological Magazine 135 (6). doi:10.1017/s0016756898631502.

Tsanas, Athanasios, and Angeliki Xifara. 2012. “Accurate Quantitative Estimation of Energy Performance of Residential Buildings Using Statistical Machine Learning Tools.” Energy and Buildings 49. doi:10.1016/j.enbuild.2012.03.003.

Yeh, I. Cheng. 2007. “Modeling Slump Flow of Concrete Using Second-Order Regressions and Artificial Neural Networks.” Cement and Concrete Composites 29 (6). doi:10.1016/j.cemconcomp.2007.02.001.