Handelshøyskolen BI

GRA 19703 Master Thesis

Thesis Master of Science 100% - W

Predefinert informasjon

Startdato:	09-01-2023 09:00 CET	**Termin:**	202310
Sluttdato:	03-07-2023 12:00 CEST	**Vurderingsform:**	Norsk 6-trinns skala (A-F)
Eksamensform:	T		
Flowkode:	202310||11184||IN00||W||T		
Intern sensor:	(Anonymisert)		

Deltaker

Navn:	Sai Prudhvi Neelakantam

Informasjon fra deltaker

Tittel *:	OPTIMISING MACHINE LEARNING TECHNIQUES: AUTOFLEX – AN AUTOML APPROACH
Navn på veileder *:	WEI TING YANG

Inneholder besvarelsen konfidensielt materiale?:	Nei	**Kan besvarelsen offentliggjøres?:**	Ja

Gruppe

Gruppenavn:	(Anonymisert)
Gruppenummer:	217
Andre medlemmer i gruppen:	Deltakeren har innlevert i en enkeltmannsgruppe

MASTER THESIS

OPTIMISING MACHINE LEARNING TECHNIQUES: AUTOFLEX – AN AUTOML APPROACH

Hand-in date

03.07.2023

Campus

BI OSLO

Supervisor

WEI TING YAN

Programme

MASTER OF SCIENCE IN BUSINESS ANALYTICS

ABSTRACT

Automated Machine Learning (AutoML) has emerged as a promising solution to tackle the challenges of algorithm selection, hyperparameter optimisation, feature engineering, scalability, and interpretability in ML tasks. This master thesis aims to profoundly investigate the state-of-the-art in AutoML and suggest revolutionary approaches to enhance its capabilities.

The research begins with a thorough review and evaluation of existing AutoML frameworks and techniques. Strengths, limitations, and applicability are scrutinised, providing valuable insights into their performance and usability across different problem domains. The evaluation includes comparisons of model performance, execution time, and interpretability, enhancing the understanding of the trade-offs involved.

Based on the findings, a novel approach, AutoFlex, is proposed to integrate established algorithms with automated pre-processing techniques. This approach leverages algorithms such as Random Forest Classifier, Gradient Boosting Regressor, and Decision Tree Classifier to ensure model interpretability. Additionally, pre-processing techniques like StandardScaler, RobustScaler, and OneHotEncoder are developed to enhance the quality of input data.

Extensive experiments are conducted on diverse datasets to evaluate the performance and interpretability of the proposed approach. Visualisations and analysis provide insights into the relationship between model performance, execution time, and interpretability, helping to interpret experimental findings.

The proposed approach, AutoFlex, combines interpretable algorithms with automated pre-processing techniques, which is crucial to developing more effective and usable AutoML systems. As AutoML continues to evolve, further research and advancements are necessary to address its limitations and maximise its potential for tackling complex ML tasks. We must continuously explore and innovate AutoML to ensure it remains a reliable and safe solution for various applications.

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to **Wei Ting Yan**, my master's thesis Supervisor from **BI Norwegian Business School**, for her invaluable guidance, support, and expertise throughout all stages of my thesis. Her dedication, patience, and insightful feedback have been instrumental in shaping the direction and quality of my research. I am profoundly grateful for her unwavering commitment to my academic growth and development.

I am also immensely grateful to **Lars Arne Skår**, my supervisor from **SAS**, for his exceptional support and mentorship during my thesis. His extensive knowledge, technical expertise, and willingness to help always have been invaluable. His guidance helped me navigate the complexities of my research and broadened my understanding of the subject matter. I am indebted to him for his continuous encouragement and valuable input.

Furthermore, I thank **Vegard Hansen** from **SAS** for his invaluable assistance in facilitating collaboration between BI Norwegian Business School and SAS. His support and coordination played a vital role in ensuring a smooth and productive partnership, enabling me to access the necessary resources and expertise.

I would also like to express my gratitude to all the faculty members and staff at BI Norwegian Business School and SAS who have contributed to my academic journey, providing an enriching learning environment and valuable insights.

Lastly, I would like to acknowledge my family, friends, and loved ones for their unwavering support, encouragement, and understanding throughout the process of completing this thesis. Their constant belief in me and their motivation has been instrumental in my academic success.

I extend my heartfelt gratitude to all those mentioned above and countless others who have supported me in several ways. Their contributions have played an integral role in completing this master's thesis, and I am sincerely thankful for their assistance and encouragement.

LIST OF ABBREVIATIONS & DEFINITIONS

ANOVA: Analysis of Variance: A statistical technique for comparing means between groups.

AUC-ROC: Area Under the Receiver Operating Characteristic Curve: A metric used to evaluate the performance of classification models

AutoFlex: Automated Flexible Machine Learning: An AutoML approach proposed in this research.

AutoML: Automated Machine Learning: The automation of machine learning tasks.

CV: Cross-validation: A technique for assessing model performance by splitting data into train and validation sets.

GNB: Gaussian Naive Bayes: A classification algorithm based on Bayes' theorem and assumed Gaussian distribution of features.

H2O: H2O AutoML: An AutoML platform with a comprehensive set of machine learning algorithms.

H2O AutoML: H2O AutoML: An AutoML platform for end-to-end automation of the ML process.

ID: Identifier: A unique label or identifier for a specific object or data point.

K-Fold: K-Fold Cross-Validation: A cross-validation technique that splits data into K subsets for model evaluation.

MSE: Mean Squared Error: A metric used to measure the average squared difference between predicted and actual values.

MAE: Mean Absolute Error: A metric used to measure the average absolute difference between predicted and actual values.

ML: Machine Learning: The field of study and practice that focuses on developing algorithms and models to learn from data.

NAS: Neural Architecture Search: An automated process for discovering optimal neural network architectures.

PCA: Principal Component Analysis: A dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional representation.

pd: Pandas: A Python library for data manipulation and analysis.

R^2 : R-squared Score: A metric representing the proportion of the variance in the dependent variable that is predictable from the independent variables.

RMSE: Root Mean Squared Error: A metric used to measure the square root of the average squared difference between predicted and actual values.

sklearn: Scikit-learn: A popular Python library for ML and data mining.

TPOT: Tree-based Pipeline Optimisation Tool: An AutoML framework that uses genetic algorithms to optimise ML pipelines.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

The application of ML as a powerful technique for deriving significant conclusions and forecasts from complex datasets has seen widespread recognition (Janiesch et al., 2021). Its applicability has expanded across various fields, ranging from healthcare and finance to marketing, among others (Sarker, 2021). Nevertheless, implementing ML methodologies in tangible, real-world situations can pose considerable challenges, often proving to be an intensive and prolonged process. Data scientists frequently encounter difficulties in diverse aspects, such as feature engineering, the choice of algorithms, the adjustment of hyperparameters, and the evaluation of models (Hutter et al., 2019; Olson & Moore, 2019). The execution of these manual procedures demands a high level of expertise, and they tend to be resource-intensive, commanding substantial computational resources and time.

The impetus for the conception of AutoML techniques is derived from the need to systematise and automate the entirety of the ML process. Automated Machine Learning, abbreviated as AutoML, aims to provide individuals, even those with limited knowledge of ML, the capability to effectively utilise ML algorithms and tap into the benefits of data-driven decision-making (Hutter et al., 2019). AutoML frameworks simplify the entire procedure by automating numerous stages of the ML pipeline, thereby lowering user entry barriers (Hutter et al., 2019).

A fundamental benefit of AutoML lies in its potential to democratise the realm of ML. In the traditional sense, ML has been an area primarily driven by domain expertise, with individuals who possess extensive knowledge about the intricacies of algorithms and methodologies thriving (Hutter et al., 2019). However, such expertise is not always readily available to all, and the need for ML solutions is escalating across many industries and sectors. By automating the multifaceted tasks that are involved in the creation and optimisation of models, AutoML frameworks manage to extend the reach of ML, making it more accessible to a wide array of users that include domain experts, business analysts, and even professionals who have minimal coding experience (Feurer et al., 2015).

In addition, AutoML techniques tackle the scalability issues brought about by the exponential surge in data. With the onset of the significant data era, conventional manual methodologies for feature engineering and model selection face difficulties in effectively managing large datasets and complex feature dimensions (Feurer et al., 2015). AutoML frameworks utilise sophisticated algorithms, optimisation procedures, and distributed computing to address these issues proficiently. These frameworks offer automated feature extraction, selection, and engineering, as well as provide algorithms and architectures capable of efficiently learning from voluminous datasets (Hutter et al., 2019).

Another vital attribute of AutoML is its role in enhancing the reproducibility and transparency of ML experiments. Automating the entire pipeline facilitated by AutoML frameworks aids in meticulously recording and tracking every step, from data pre-processing to the configuration and evaluation of the model (Géron, 2019). Such transparency fosters the reproducibility of experiments, empowering researchers to verify and compare varied approaches. Furthermore, it encourages the dissemination of knowledge and collaboration within the ML community, as researchers can effortlessly share their pipelines and methodologies with their peers (Ribeiro et al., 2016).

Numerous studies and research efforts have been dedicated to developing AutoML frameworks and techniques. These studies have explored various approaches, including genetic algorithms, Bayesian optimisation, reinforcement learning, and neural architecture search, to automate distinct aspects of the ML pipeline (Eggensperger, 2013; Hutter et al., 2011; Real et al., 2019; Zoph & Le, 2017).

Researchers have proposed frameworks such as Auto-sklearn (Feurer et al., 2015), H2O AutoML (LeDell, 2020), TPOT (Olson & Moore, 2019), and others, each with its unique algorithmic strategies and capabilities.

While AutoML offers significant advantages, it also faces certain limitations and challenges. Selecting appropriate pre-processing techniques, algorithmic choices, and hyperparameter settings still requires careful consideration and domain knowledge (Krafft et al., 2020). AutoML frameworks may produce suboptimal results if not correctly configured or if the data is not adequately pre-processed.

Additionally, the automation of the ML pipeline raises ethical concerns, as decisions made by the automated system may have significant consequences in critical applications such as healthcare or finance (Zöller & Huber, 2021).

In conclusion, AutoML techniques have gained considerable attention in the field of ML due to their potential to automate and simplify the ML workflow (Hutter et al., 2019). By automating tasks such as feature engineering, algorithm selection, and hyperparameter tuning, AutoML frameworks aim to democratise ML and make it more accessible to a broader range of users (Feurer et al., 2015). These frameworks leverage advanced algorithms, optimisation techniques, and distributed computing to handle large datasets and complex feature spaces (Thornton et al., 2013). However, challenges related to interpretability, domain knowledge, and ethical considerations must be carefully addressed (Holzinger et al., 2017). Nonetheless, the ongoing research and development in AutoML hold promise for transforming how ML is applied and utilised in various industries and domains (Domingos, 2012).

1.2 RESEARCH PROBLEM AND OBJECTIVES

FEATURE ENGINEERING AND SELECTION

Feature engineering plays a vital role in improving model performance. A significant research objective is automating the feature engineering process, including feature extraction, transformation, and selection (Johnson, 2019). The objective is to develop techniques to automatically identify and generate relevant features from raw data and determine the most informative subset of features for improved model performance.

SCALABILITY AND EFFICIENCY

As datasets continue to grow and complexity, there is a need for AutoML frameworks that can handle big data efficiently (Chen & Guestrin, 2016). The objective is to develop scalable and distributed AutoML approaches that process and analyse massive datasets time-efficiently.

INTERPRETABILITY AND EXPLAINABILITY

While AutoML aims to automate the ML process, it is essential to ensure that the generated models are interpretable and explainable (Ribeiro et al., 2016). The objective is to develop techniques that can provide insights into the decision-making process of automated models, allowing users to understand and trust the results.

1.3 RESEARCH QUESTIONS

To achieve the research objectives outlined in the previous section, the following research questions will guide the investigation and exploration of AutoML:

RESEARCH QUESTION 1

What are the strengths, limitations, and interpretability of existing AutoML frameworks and techniques in addressing the challenges of algorithm selection, hyperparameter optimisation, feature engineering, scalability, and interpretability?

This research question aims to assess the current state-of-the-art in AutoML by analysing and evaluating the capabilities and shortcomings of existing frameworks and techniques. It will involve a comprehensive review and comparison of different AutoML approaches, considering their performance, scalability and interpretability.

RESEARCH QUESTION 2

How can novel algorithms and approaches be developed to improve algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability in AutoML?

This research question focuses on proposing innovative solutions and algorithms to enhance the core components of AutoML. It seeks to address the identified challenges and limitations by exploring new methods for algorithm selection, efficient hyperparameter optimisation, advanced feature engineering techniques, scalable AutoML frameworks, and interpretable models. The objective is to contribute novel techniques that can overcome existing limitations and improve the overall performance and usability of AutoML systems.

By addressing these research questions, the study aims to advance the understanding of AutoML, uncover novel approaches, and provide insights into the development of AutoML systems.

1.4 SIGNIFICANCE AND POTENTIAL IMPACT OF THE RESEARCH

The research on improving AutoML algorithms and approaches has significant significance and potential impact in several ML and data science areas. The following points highlight the importance of this research:

ADVANCING AUTOMATION AND EFFICIENCY

By developing novel algorithms and approaches for AutoML, the research aims to automate and streamline the ML process further. This has the potential to significantly reduce the time and effort required for model development, making ML more accessible to a broader range of users. Increased automation and efficiency in AutoML can lead to faster deployment of accurate and reliable models (Kotthoff et al., 2017), benefiting various industries and applications.

ENHANCED MODEL PERFORMANCE

The research focuses on improving algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability. By addressing these areas, the research aims to enhance the performance of AutoML models. More accurate and robust models can lead to better decision-making, improved predictions, and enhanced insights from the data (Alsharef et al., 2022; Bergstra et al., 2022).

DEMOCRATISING MACHINE LEARNING

AutoML has the potential to democratise ML by making it accessible to individuals and organisations without extensive knowledge of data science. By developing advanced AutoML algorithms and approaches, the research aims to empower users with limited expertise to leverage the power of ML for their specific tasks. This can transform small businesses, non-profit organisations, and individuals who can benefit from data-driven insights without needing specialised skills (Gil et al., 2019).

ADDRESSING COMPLEX DATA CHALLENGES

The exploration of AutoML algorithms and methodologies is primarily aimed at addressing intricate data difficulties, encompassing high-dimensional data, imbalanced datasets, and noisy data. By pioneering novel solutions for selecting algorithms, hyperparameter optimisation, and data pre-processing methods, this research seeks to surmount these challenges, thereby enhancing the robustness and dependability of models (Filippou et al., 2023).

The ongoing research aimed at refining AutoML algorithms and strategies bear considerable relevance and potential for impact. They carry the promise to propel automation, augment model performance, democratise ML, resolve intricate data issues, and contribute meaningfully to the expansive domain of AutoML.

1.5 SCOPE AND LIMITATIONS

This research intends to delve into and assess a variety of AutoML frameworks and methods, with the goal of enhancing the productivity and efficacy of the ML model development process. The investigation will principally focus on examining existing AutoML frameworks, including but not limited to TPOT, and H2O AutoML, among others, whilst scrutinising their respective algorithms, techniques, and features.

THE SCOPE OF THE RESEARCH

The scope of this research encompasses the evaluation and analysis of various AutoML frameworks in terms of their performance and capabilities in algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability. Additionally, novel algorithms and approaches will be investigated to enhance existing AutoML techniques and address their limitations and challenges. The research will also explore methods to improve the efficiency and scalability of AutoML frameworks to handle large-scale datasets and complex models effectively. Furthermore, the interpretability of AutoML models will be assessed, and techniques will be proposed to enhance their transparency and explainability.

THE LIMITATIONS OF THE RESEARCH

Firstly, the analysis will focus on a selection of popular AutoML frameworks, and therefore, it may not encompass all existing frameworks in the evaluation. Secondly, the performance and effectiveness of AutoML techniques can vary depending on the specific dataset, problem domain, and task at hand. Therefore, the findings of this research may not be universally applicable to all scenarios. Thirdly, due to time and resource constraints, exploring algorithm combinations, feature engineering techniques, and pre-processing methods within each framework may not be exhaustive. Lastly, this research will primarily concentrate on regression and classification tasks, potentially overlooking other ML tasks such as clustering, anomaly detection, or image processing.

Despite these limitations, the primary objective of the research is to offer insightful revelations about the present state of AutoML, pinpoint potential areas that require enhancement, and make substantial contributions towards the progress of AutoML methodologies.

2 LITERATURE REVIEW

The literature review offers a broad and detailed examination of current research and studies associated with AutoML methodologies. It covers the historical progression and growth of AutoML, an evaluation of prevailing AutoML frameworks, a comparison of various AutoML strategies, and the evaluation metrics applicable to regression and classification tasks. The intent behind this section is to lay a robust groundwork for the research, achieved through a thorough investigation of the work conducted in this field while concurrently identifying gaps and possibilities for additional exploration.

2.1 OVERVIEW OF AUTOML

AutoML, an ascendant field, aspires to automate various aspects of the ML workflow, thereby enhancing its accessibility and efficiency for users who possess limited expertise in the realm of data science (Hutter et al., 2019). It merges ML with optimisation and data pre-processing methodologies to automate tasks such as

algorithm selection, hyperparameter tuning, feature engineering, and model evaluation (Feurer et al., 2015).

The rise in the complexity of ML models and the growing need for data-driven decision-making across diverse industries has resulted in AutoML techniques attracting considerable attention. These techniques empower researchers, data scientists, and practitioners to primarily focus on the formulation of problems and domain knowledge, rather than invest extensive amounts of time in manual and repetitive tasks (Waring et al., 2020).

A key goal of AutoML is to simplify the procedure of developing precise and resilient ML models by automating tasks that are generally time-consuming (Olson & Moore, 2019). By harnessing computational power and sophisticated optimisation algorithms, AutoML frameworks aim to pinpoint the most efficient ML pipeline for any given dataset (Gijsbers et al., 2019).

AutoML frameworks typically involve several stages. First, the data is pre-processed to handle missing values, outliers, and feature scaling. Next, a search strategy, such as genetic algorithms, Bayesian optimisation, or reinforcement learning, is employed to explore the space of potential models and hyperparameters (Xu et al., 2021). The selected models are then trained and evaluated using appropriate metrics to assess their performance.

The primary advantage of AutoML is its ability to reduce the expertise and time required to build effective ML models (He et al., 2021). It allows users with limited knowledge of ML techniques to leverage state-of-the-art algorithms and methodologies. Additionally, AutoML frameworks provide transparency by documenting the entire modelling process, enabling reproducibility, and facilitating model interpretation (Hutter et al., 2019).

However, AutoML is not without its limitations. One major challenge is the scalability of AutoML frameworks to handle large datasets with high-dimensional features. The computational resources and time required for exhaustive search and optimisation can become prohibitive (Kotthoff et al., 2017). Additionally, AutoML

frameworks may not always capture the intricacies of the problem domain, as they rely on general-purpose algorithms and techniques (Olson & Moore, 2019)

Despite these limitations, AutoML holds significant potential in democratising ML and enabling non-experts to leverage its benefits. Ongoing research addresses scalability issues, incorporates domain-specific knowledge, and improves the interpretability of AutoML pipelines (Gijsbers et al., 2019).

2.2 REVIEW OF EXISTING AUTOML FRAMEWORKS

In this section, an in-depth review of a few notable AutoML frameworks, specifically H2O AutoML and TPOT, is offered. These frameworks have made considerable strides in automating the ML process and have achieved extensive acceptance in both academic and industrial contexts.

2.2.1 H2O AUTOML

H2O AutoML, a product of H2O.ai, is an all-inclusive AutoML platform that facilitates automated model training and hyperparameter tuning. It boasts a user-friendly interface and accommodates a variety of algorithms, encompassing deep learning models. Excelling in scalability, H2O AutoML has the capability to manage vast datasets and intricate models efficiently. It further incorporates progressive feature engineering techniques and ensemble methodologies to bolster model performance (LeDell, 2020).

2.2.2 TPOT

TPOT, or Tree-based Pipeline Optimisation Tool, is a potent AutoML framework that utilises genetic programming to automatically unearth the most effective ML pipeline tailored to a specific dataset. TPOT evolves a group of pipelines using genetic operations such as crossover and mutation to navigate the pipeline space. It manages feature pre-processing steps, algorithm selection, and hyperparameter optimisation concurrently. TPOT's ability to automate the entire ML pipeline has garnered considerable attention and it has been effectively applied to an array of real-world tasks (Olson & Moore, 2019).

These frameworks have streamlined the creation of ML models by automating laborious tasks, including algorithm selection, hyperparameter tuning, and feature engineering. They have eased the application of ML methodologies to specific problems for both researchers and practitioners, even in the absence of profound expertise in the field. These AutoML frameworks have shown encouraging results and possess the potential to expedite the integration of ML across various domains.

2.3 COMPARISON OF EXISTING AUTOML APPROACHES

In this section, a thorough comparison of assorted AutoML strategies is offered in *Table 1*. The table lays out an overview of the algorithms, techniques, and features of each approach, thereby facilitating a comprehensive understanding of their competencies and functions. The data presented in *Table 1* lays the groundwork for an extended exploration and examination of the diverse AutoML methodologies discussed in this section.

Table 1

Comparison of Approaches for Automated Machine Learning

The table compares two approaches for AutoML: H2O AutoML and TPOT. H2O AutoML utilises an intelligent search strategy, combines random search and grid search for hyperparameter optimisation, incorporates built-in feature engineering, and provides partial interpretability through decision trees. TPOT employs genetic programming for algorithm selection and hyperparameter optimisation, integrates feature pre-processing into the pipeline, and focuses on performance optimisation

Approach	Algorithm Selection	Hyperparameter Optimisation	Feature Engineering	Interpretability
H2O AutoML	Intelligent search strategy	Random search, grid search	Built-in feature engineering	Partial interpretability (e.g., decision trees)
TPOT	Genetic programming	Genetic programming	Integrated feature pre-processing	Focus on performance optimisation

HYPERPARAMETER OPTIMISATION

H2O AutoML utilises a combination of random search and grid search to explore the hyperparameter space of different algorithms (LeDell, 2020). It intelligently samples from the hyperparameter space to find good combinations of hyperparameters. In contrast, TPOT uses genetic programming to evolve the

hyperparameters of each algorithm in the pipeline. It evolves the pipelines by applying genetic operators such as crossover and mutation to explore the hyperparameter space effectively (Olson et al., 2016a).

FEATURE ENGINEERING

H2O AutoML includes built-in feature engineering capabilities such as handling missing values, categorical encoding, and feature scaling (LeDell, 2020). It automatically applies these techniques during the modelling process. In contrast, TPOT integrates feature pre-processing steps within the pipeline evolution process (Olson et al., 2016a). To find the optimal combination, it explores different feature pre-processing techniques, including imputation, scaling, and dimensionality reduction.

INTERPRETABILITY

H2O AutoML offers some level of interpretability for models such as linear models and tree-based models. It provides feature importance rankings and decision rules for decision trees and rule-based models (LeDell, 2020). In contrast, TPOT focuses more on model performance optimisation than interpretability (Olson et al., 2016a). However, it can evolve pipelines with interpretable models such as decision trees and linear models.

It is important to note that the performance and effectiveness of AutoML approaches can vary depending on the dataset characteristics, task requirements, and available computational resources. Researchers and practitioners should consider these factors along with the specific features and techniques each AutoML framework offers when deciding. Additionally, the AutoML field continuously evolves, and new frameworks and techniques are being developed, offering even more advanced automation capabilities (Choudhary et al., 2022).

2.4 EVALUATION METRICS

In this study, the performance of ML models in both regression and classification tasks is evaluated using specific metrics. The selection of these evaluation metrics is determined by the nature of the task and the distinct objectives of the analysis. Here are the employed evaluation metrics and their corresponding functions:

2.4.1 FOR REGRESSION TASKS

MSE (Mean Squared Error) measures the average squared difference between the predicted and actual values. It provides an overall assessment of the model's accuracy, where a lower MSE indicates better accuracy. MSE is used to evaluate the performance and accuracy of regression models in predicting continuous numerical values.

RMSE (Root Mean Squared Error) is the square root of MSE, providing a more interpretable metric in the original scale of the target variable. It is widely used as an evaluation metric for regression tasks because it represents the average deviation of the predicted values from the actual values.

MAE (Mean Absolute Error) measures the average absolute difference between the predicted and actual values. It is less sensitive to outliers than MSE and provides a more robust model performance evaluation in regression tasks.

R^2 (R-squared Score) is a statistical measure that indicates the proportion of the variance in the target variable that is explained by the model. It ranges from 0 to 1, with higher values indicating a better fit. R^2 is commonly used to assess the goodness of fit of regression models and understand how well the model captures the variability in the data.

2.4.2 FOR CLASSIFICATION TASKS

Accuracy measures the proportion of correctly classified instances. It is calculated as the ratio of the number of correct predictions to the total number of predictions. Accuracy is widely used as a performance metric in classification tasks to assess the overall correctness of the model's predictions.

Precision represents the fraction of correctly predicted positive instances out of the total predicted positive instances. It focuses on the quality of positive predictions and helps evaluate how well the model avoids false positives.

Recall (Sensitivity or True Positive Rate) indicates the fraction of correctly predicted positive instances out of the total actual positive instances. It focuses on

capturing all positive instances and helps evaluate how well the model avoids false negatives.

F1 Score combines precision and recall into a single metric. It balances the trade-off between precision and recall and is useful in evaluating model performance in imbalanced datasets where the distribution of classes is uneven.

AUC-ROC (Area Under the Receiver Operating Characteristic Curve) evaluates the model's ability to discriminate between positive and negative instances across different probability thresholds. It provides an aggregate measure of the model's performance and is particularly useful in assessing the performance of binary classification models.

These evaluation metrics are applied to assess the performance and generalisation ability of the developed ML models in both regression and classification tasks. They help quantify the accuracy, robustness, and predictive power of the models, enabling comparisons and informed decision-making in selecting the best-performing models for different applications.

2.5 RELATED STUDIES AND RESEARCH IN THE FIELD

In the study titled "TPOT: A Tree-based Pipeline Optimisation Tool for Automating Machine Learning" by Randal S. Olson and Jason H. Moore, TPOT, a tool designed to automate ML tasks via a tree-based pipeline optimisation, is introduced. TPOT employs genetic programming to develop and optimise pipelines by probing through a vast array of combinations of data pre-processing steps and ML algorithms. The effectiveness of TPOT in delivering competitive performance across diverse ML tasks and datasets was successfully demonstrated in this research (Olson & Moore, 2019).

The book titled "Automated Machine Learning: Methods, Systems, Challenges", edited by Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, presents a thorough overview of AutoML, encapsulating an array of methods, systems and the challenges faced within the field. It boasts contributions from eminent researchers and practitioners, providing insights into the latest developments, ideal practices, and future trajectories of AutoML. The book stands as an invaluable asset for

researchers, practitioners, and students who harbour an interest in the domain of AutoML (Hutter et al., 2019).

In the study titled "Evaluation of a Tree-based Pipeline Optimisation Tool for Automating Data Science" by Randal S. Olson, Ryan J. Urbanowicz, Nathan Bartley, and Jason H. Moore, the focal point is the assessment of TPOT, a tool dedicated to automating data science tasks through tree-based pipeline optimisation. The authors demonstrated the proficiency of TPOT in creating ML pipelines that surpassed the performance of manually crafted pipelines across various benchmark datasets. The study underscored TPOT's capability to automate the most laborious aspects of ML, thus enabling individuals without expert-level knowledge to utilise sophisticated techniques (Olson et al., 2016).

"Scaling tree-based automated ML to biomedical big data with a feature set selector" by Trang T. Le, Weixuan Fu, and Jason H. Moore: This study puts emphasis on scaling tree-based AutoML to accommodate biomedical big data. The authors introduced a feature set selector that enabled TPOT to efficiently manage large datasets by choosing subsets of features. The scalability and efficacy of the approach on biomedical datasets were demonstrated in the study, underscoring its potential to hasten the analysis of big data within the biomedical sphere (Le et al., 2020).

In "AutoML: A Survey of the State-of-the-Art" by He, X., Zhao, K., & Chu, X, a comprehensive review is offered, which discusses the current progress in AutoML, encompassing data preparation, feature engineering, hyperparameter optimisation, and neural architecture search (NAS). The study also probes into prospective research trajectories, including the joint optimisation of hyperparameters and architecture. The authors highlighted unresolved issues in existing AutoML methods, marking them for further exploration (He et al., 2021).

In the study titled "H2O AutoML: Automatic Machine Learning for Production" by H2O.ai, H2O AutoML, an open-source AutoML platform designed specifically for production settings, is introduced. H2O AutoML offers a user-friendly interface that automates the construction and deployment of ML models. The study emphasises the platform's wide-ranging capabilities in feature engineering, hyperparameter

optimisation, and model selection. Furthermore, it discusses the significance of the interpretability of models in real-world applications (LeDell, 2020).

These studies make substantial contributions to the domain of AutoML by presenting and evaluating diverse frameworks and tools designed to automate various stages of the ML procedure. Auto-sklearn and H2O AutoML offer pragmatic solutions for automating model selection, hyperparameter optimisation, and feature engineering, thus tackling the hurdles encountered by data scientists in real-world circumstances. The studies underscore the significance of scalability, interpretability, and user-friendliness in developing AutoML frameworks that can be efficaciously deployed in production settings.

3 METHODOLOGY

The approach proposed in this research is called "**AutoFlex**", which addresses the challenges and limitations of the traditional machine learning pipeline. AutoFlex represents a novel and innovative solution that combines the benefits of automated machine learning (AutoML) with flexible and customisable pipeline construction.

AutoFlex justifies its existence by offering several key advantages over traditional approaches. First and foremost, it provides an automated and intelligent solution for algorithm selection, hyperparameter optimisation, and feature engineering. By leveraging advanced optimisation algorithms and techniques, AutoFlex identifies the most suitable algorithms and their optimal configurations based on the characteristics of the data. This eliminates the need for manual trial and error, enabling researchers and practitioners to efficiently build high-performing models.

Furthermore, AutoFlex integrates flexibility and customizability into the automated machine learning pipeline generation. It allows users to define and customize pre-processing techniques, feature selection methods, and model configurations according to their specific needs and domain expertise. This adaptability empowers users to tailor the pipeline to the unique characteristics and requirements of their datasets, improving model performance and interpretability.

3.1 THE AUTOFLEX APPROACH

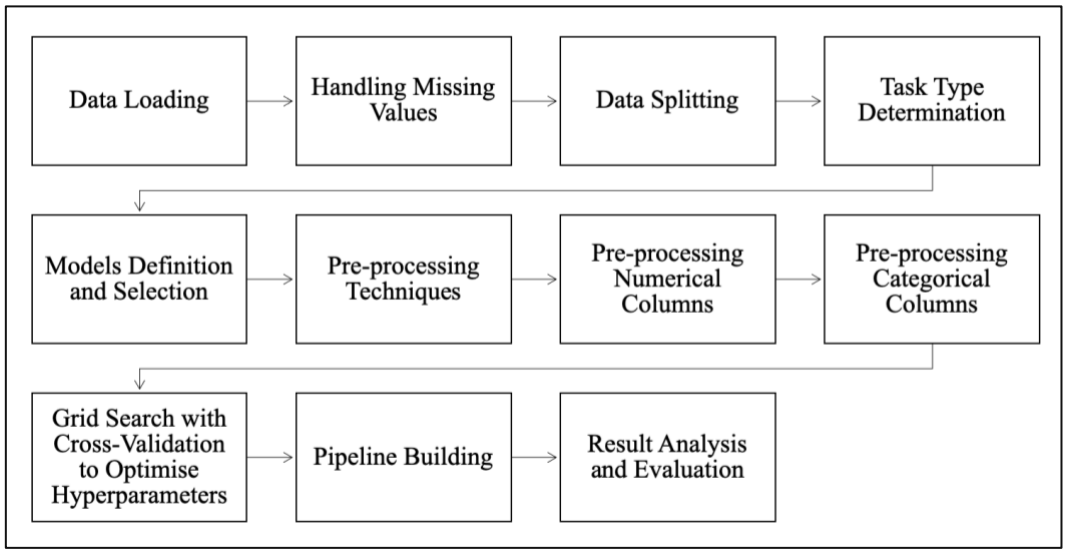
AutoFlex is designed to automate the process of selecting the best ML model and pre-processing techniques to build efficient ML pipelines for a given dataset.

The AutoFlex approach, as depicted in *Figure 1*, outlines the step-by-step process of automating and optimising the ML pipeline. It encompasses crucial stages such as data loading, handling missing values, data splitting, pre-processing, and grid search with cross-validation. By following this approach, researchers and practitioners can streamline the selection of the best ML model and pre-processing techniques, resulting in the construction of efficient ML pipelines tailored to the given dataset.

Figure 1

Workflow Of The Proposed Approach – AutoFlex

The AutoFlex methodology comprises meticulous stages intended to automate and optimise the ML procedure. These stages involve data loading, addressing missing values, data splitting, task type identification, model selection, pre-processing methods, hyperparameter optimisation, pipeline construction, and result analysis. The accompanying figure summarises the AutoFlex approach and its pivotal stages, acting as a visual guide for effectively implementing the methodology.



The AutoFlex approach can be further explained in the following detailed steps:

DATA LOADING

The dataset is loaded from a CSV file using the load_dataset function. This function reads the file path provided as input and utilises the pd.read_csv function from the Pandas library to load the data into a Pandas data frame. Logging statements are included to provide updates on the progress, such as the file being loaded.

HANDLING MISSING VALUES

The handle_missing_values function handles missing values in the dataset. It identifies numerical and categorical columns and fills in missing values accordingly. Logging statements indicate the progress of how each column with missing values is handled.

DATA SPLITTING

The split_data function separates the dataset into features and the target variable. It drops the target variable column from the data frame to obtain the feature set and stores it separately as the target variable. The function returns the feature set and the target variable. Logging statements confirm the completion of the data-splitting process.

TASK TYPE DETERMINATION

The task type (classification or regression) is determined based on the unique percentage of the target variable. The task types are classified into Binary Classification, Multi-class Classification and Regression.

The determined task type is essential for model selection and performance evaluation. Logging statements display the determined task type.

MODELS DEFINITION AND SELECTION:

Based on the determined task type, specific models suited for the task are defined and stored in the models' dictionary.

For Binary Classification tasks, models such as Decision Tree Classifier, Gradient Boosting Classifier, Random Forest Classifier, MLP Classifier, Logistic Regression, and KNN Classifier are included. For Multi-class Classification tasks, similar models are included but with the appropriate class names. For Regression tasks, models such as Decision Tree Regressor, Gradient Boosting Regressor, Random Forest Regressor, MLP Regressor, Linear Regression, and KNN Regressor are included.

The models are defined along with their respective hyperparameters, which will be optimised during the grid search process.

PRE-PROCESSING TECHNIQUES

Two dictionaries, numerical pre-processing methods and categorical pre-processing methods are defined to store the available pre-processing techniques for numerical and categorical columns, respectively. Numerical pre-processing methods include SimpleImputer, StandardScaler, RobustScaler, MinMaxScaler, Polynomial Features, and PCA. Categorical pre-processing methods include OneHotEncoder and LabelEncoder. These pre-processing techniques will be utilised during the pre-processing steps to transform the data.

PRE-PROCESSING NUMERICAL COLUMNS

AutoFlex applies pre-processing techniques to numerical columns using the process_numerical_cols function. Parallel processing is enabled for efficient iteration over each selected column. The function takes inputs such as the feature set, numerical column list, dictionaries of pre-processing methods, models, and the target variable. Multiple techniques are evaluated using pipelines for each column, and the best technique is selected based on the highest model score. The function returns a list of tuples with the column name and best pre-processing technique. Logging statements track the progress of numerical column pre-processing.

PRE-PROCESSING CATEGORICAL COLUMNS

AutoFlex applies pre-processing techniques to categorical columns using the process_categorical_cols function. Parallel processing is enabled for efficient iteration over each selected column. The function takes inputs such as the feature set, categorical column list, dictionaries of pre-processing methods, feature selection methods, models, and the target variable. Multiple techniques and feature selection methods are evaluated for each column, and the best pre-processing technique and feature selection method (if applicable) are selected based on the highest model score. The function returns a list of tuples with the column name, best pre-processing technique, and best feature selection method. Logging statements track the progress of categorical column pre-processing.

GRID SEARCH WITH CROSS-VALIDATION FOR HYPERPARAMETER OPTIMISATION

AutoFlex performs a grid search with cross-validation to find the best hyperparameters for each model and column combination. The function utilises

parallel processing to iterate over models and columns, setting up pipelines and executing grid search using the GridSearchCV class. The best hyperparameters and pre-processing techniques are selected based on the highest score achieved during cross-validation. The results are stored in a sorted data frame and returned along with pipeline counts. Logging statements track the progress of the grid search process.

PIPELINE BUILDING

After the grid search process, AutoFlex builds pipelines for the best models and pre-processing techniques based on the results. The pipeline-building process is integrated within the perform_grid_search_parallel function. The pipeline counts are stored, representing the number of pipelines generated for each model and column combination. The pre-processing steps and techniques are concatenated to create pre-processing steps strings for each pipeline. The pipeline building progress is tracked using the tqdm library to provide an overview of the generated pipelines.

RESULT ANALYSIS AND EVALUATION

The results from the grid search process are collected and analysed to evaluate the performance of the models and pre-processing techniques. Performance metrics such as best score, pipeline score, accuracy (for classification tasks), R-squared (for regression tasks), test score, and execution time are evaluated and stored in the data frame. The results are recorded to provide an overview of the AutoFlex approach.

The detailed implementation of the AutoFlex approach, including the code and definitions, can be found in *APPENDIX 1*. By following this approach, researchers and practitioners can automate the process of selecting the best ML model and pre-processing techniques, reducing manual effort and improving the efficiency of the model development process.

3.1.1 FEATURE ENGINEERING METHODS

AutoFlex employs various feature engineering methods to enhance data quality and relevance, improving model performance and robustness. These methods include:

FEATURE SCALING

Numerical features undergo feature scaling to ensure comparable distributions and magnitudes. This prevents dominance by features with larger values. Standard scaling transforms features with zero mean and unit variance, while min-max scaling scales features to a specified range (typically 0 to 1).

CATEGORICAL ENCODING

Categorical features are encoded using label encoding. This method assigns a unique numerical label to each category in a categorical feature. It enables models to work with categorical data by representing categories as numerical values.

FEATURE TRANSFORMATION

Feature transformation techniques, such as creating polynomial features and using dimensionality reduction methods like PCA, are applied to capture non-linear relationships and reduce the dimensionality of the feature space. Polynomial features generate interaction terms and higher-order combinations of existing features to capture complex interactions and non-linear patterns. PCA helps preserve valuable information while reducing the number of dimensions in the feature space.

FEATURE SELECTION

Feature selection methods are employed to identify the most informative and relevant features for the task at hand. These methods aim to reduce dimensionality and eliminate irrelevant or redundant features to improve model performance and prevent overfitting (Guyon & Elisseeff, 2003). Standard techniques include SelectKBest, which selects the top K features based on statistical tests like F-regression or chi-squared test, and recursive feature elimination, which iteratively removes features based on their importance as determined by the model. By applying these pre-processing steps and feature engineering methods, AutoFlex ensures that the input data is properly prepared and optimised for the ML models. This enables the models to capture relevant patterns and relationships in the data, leading to improved prediction accuracy and generalisation performance.

3.1.2 HYPERPARAMETER OPTIMISATION STRATEGIES

The following hyperparameter optimisation strategies are used in the AutoFlex approach:

GRID SEARCH WITH CROSS-VALIDATION

Grid search is performed to search through a predefined set of hyperparameter values for each model. Cross-validation is used to evaluate the performance of different hyperparameter combinations.

K-FOLD CROSS-VALIDATION

K-Fold cross-validation is employed during the grid search process to ensure robust evaluation of the models. The dataset is divided into K folds; each fold is used as a validation set, while the remaining folds are used for training.

PARAMETER GRID DEFINITION

A parameter grid is defined for each model, specifying the hyperparameter values to be tested during grid search. This grid includes different values for hyperparameters like max depth, number of estimators, hidden layer sizes, and more, depending on the model. These strategies collectively aim to find the optimal combination of hyperparameters and pre-processing techniques that yield the best performance for each model in the AutoFlex approach.

3.1.3 DATASET SELECTION

In this study, dataset selection and pre-processing are essential for evaluating the performance of the AutoML approach. The sklearn library provides a wide range of well-documented classification and regression datasets commonly used in ML research (*Table 2*). These datasets are carefully curated, pre-processed, and representative of real-world scenarios, making them suitable for assessing the effectiveness of the AutoML approach.

For Classification tasks, datasets like 'wine', 'breast_cancer', 'iris', 'digits', and 'linnerud' are chosen. They cover various classification problems, including wine types, breast cancer diagnosis, flower species classification, handwritten digit recognition, and exercise physiology measurements. Additional datasets from the openml repository, such as 'adult', 'australian', 'bank-marketing', 'car', and 'tic-tac-toe', provide diverse real-world scenarios.

For Regression tasks, datasets like 'boston', 'diabetes', 'linnerud', and 'california_housing' are selected. They encompass regression problems related to housing prices, diabetes progression, exercise physiology measurements, and housing price prediction in California. Additional regression datasets from the openml repository, such as 'house_8L', 'cpu', 'qsar-biodeg', and 'bike_sharing', cover domains like housing, computer performance, chemical properties, and bike-sharing demand prediction.

To evaluate the performance and scalability of the AutoML approach, subsets are created from the original datasets by splitting them into different proportions. This includes subsets representing 25%, 50%, and 100% of the observations. Creating subsets allows for systematically analysing the approach's performance under different data sizes. The 25% subset provides a smaller sample size, enabling faster experimentation and initial assessment. The 50% subset strikes a balance between computational efficiency and capturing a significant portion of the data. The 100% subset represents the complete dataset and allows for a comprehensive evaluation of the approach's performance on the entire data.

By evaluating the AutoML approach on subsets of different sizes, it becomes possible to assess its scalability, efficiency, and generalisation capabilities across varying data volumes. This information is crucial for understanding the approach's behaviour and its applicability to real-world datasets of different sizes.

Table 2

Overview of Datasets & Subsets in the Experiment

The table provides an overview of the datasets used in the experiment to iterate over multiple approaches. It includes information such as the dataset ID, dataset name, number of rows, percentage of observations, subset ID, subset name, number of columns, target variable type, and classification or regression nature of the dataset. The table presents a comprehensive view of the datasets employed in AutoFlex, allowing for easy identification and selection of specific datasets for analysis and model development.

Dataset_ID	Dataset name	# Dataset Rows	% of observations	Subset_ID	Subset name	# Subset rows	# Columns	Target Variable Type
1	iris	150	0.50	1	iris_50p	75	5	Classification
1	iris	150	0.25	2	iris_25p	38	5	Classification
1	iris	150	1.00	3	iris_100p	150	5	Classification
2	wine	178	0.50	4	wine_50p	89	14	Classification
2	wine	178	1.00	5	wine_100p	178	14	Classification
2	wine	178	0.25	6	wine_25p	44	14	Classification
3	cpu	209	0.25	7	cpu_25p	52	8	Regression
3	cpu	209	1.00	8	cpu_100p	209	8	Regression
3	cpu	209	0.50	9	cpu_50p	104	8	Regression
4	boston	506	0.50	10	boston_50p	253	14	Regression
4	boston	506	0.25	11	boston_25p	126	14	Regression
4	boston	506	1.00	12	boston_100p	506	14	Regression
5	breast_cancer	569	0.25	13	breast_cancer_25p	142	31	Classification
5	breast_cancer	569	1.00	14	breast_cancer_100p	569	31	Classification
5	breast_cancer	569	0.50	15	breast_cancer_50p	284	31	Classification
6	australian	690	1.00	16	australian_100p	690	15	Classification
6	australian	690	0.25	17	australian_25p	172	15	Classification
6	australian	690	0.50	18	australian_50p	345	15	Classification
7	diabetes	768	1.00	19	diabetes_100p	768	9	Classification
7	diabetes	768	0.25	20	diabetes_25p	192	9	Classification
7	diabetes	768	0.50	21	diabetes_50p	384	9	Classification
8	qsar-biodeg	1055	0.50	22	qsar-biodeg_50p	528	42	Classification
8	qsar-biodeg	1055	0.25	23	qsar-biodeg_25p	264	42	Classification
8	qsar-biodeg	1055	1.00	24	qsar-biodeg_100p	1055	42	Classification
9	digits	1797	0.25	25	digits_25p	449	65	Classification
9	digits	1797	1.00	26	digits_100p	1797	65	Classification
9	digits	1797	0.50	27	digits_50p	898	65	Classification
10	california_housing	20640	0.50	28	california_housing_50p	10320	9	Regression
10	california_housing	20640	0.25	29	california_housing_25p	5160	9	Regression
10	california_housing	20640	1.00	30	california_housing_100p	20640	9	Regression

3.1.4 PARALLEL PROCESSING

Parallel processing is employed in the AutoFlex approach to enhance the efficiency of computationally intensive tasks. The joblib library enables parallel execution of tasks across multiple CPU cores.

The parallel processing technique is particularly beneficial during the grid search phase, where multiple models and hyperparameter combinations are evaluated. By distributing the workload across multiple cores, the processing time is significantly reduced, allowing for faster experimentation and model evaluation (Singh, 2021).

Additionally, parallel processing is utilised when processing numerical and categorical columns, as well as during the evaluation of multiple pipelines. These tasks are parallelised to expedite the feature engineering and model selection processes.

Using parallel processing in the AutoFlex approach maximises computational resources and accelerates the overall pipeline, enabling efficient exploration of various model configurations and pre-processing techniques (Singh, 2021).

3.2 DETAILS OF SOFTWARE & SOFTWARE PACKAGES:

The entire project was implemented using the Python programming language, which is widely recognised and extensively used in the field of ML. Python offers a rich ecosystem of libraries and frameworks that provide comprehensive support for various aspects of ML. In this study, the following vital libraries were utilised:

Scikit-learn (sklearn) is a popular ML library in Python that offers many algorithms and tools for data pre-processing, feature selection, model training, and evaluation. It was extensively used in this project due to its efficient and well-documented implementation of various ML techniques.

Pandas is a powerful data manipulation library in Python. It provides data structures and functions for efficient data handling, including data loading, handling missing values, and data manipulation tasks. Pandas played a crucial role in this project for tasks such as data loading, handling missing values, and data transformation.

NumPy is a fundamental library for scientific computing in Python. It supports large, multi-dimensional arrays and provides a collection of mathematical functions. NumPy was employed in this project for numerical operations and transformations on the data.

Joblib is a Python library used for caching and parallel computing. It enables efficient caching of pre-processed data and parallel execution of computationally intensive tasks. Joblib was utilised in this study to parallelise the pre-processing and model training steps, improving the overall efficiency of the AutoML process.

Tqdm is a Python library that provides progress bars for iterative tasks. It was employed in this project to display informative progress bars during the pre-processing and model training stages. These progress bars allowed for better tracking of the execution progress and provided a visual representation of the ongoing tasks.

GridSearchCV is a class in sklearn that implements grid search with cross-validation. It performs an exhaustive search over specified hyperparameter combinations to find the best hyperparameters for a given model. GridSearchCV was used in this study for hyperparameter optimisation and model selection, enabling the identification of optimal parameter settings.

The logging module was utilised to facilitate logging and provide informative messages during the execution of the AutoML approach. It enhanced the overall transparency and traceability of the project.

The combination of these Python packages provided a robust and efficient framework for implementing and evaluating of the AutoML approach in this research.

3.3 COMPARATIVE ANALYSIS

In the comparative analysis, the experimental results from the AutoFlex approach, including best scores, hyperparameters, pipeline scores, accuracy/R^2 scores, test scores, and execution times, were collected for each model and pre-processing

technique combination. These results were then compared to the scores and execution times obtained from other AutoML approaches, such as H2O and TPOT.

The experimental setup provided a structured framework for evaluating the performance of different models and pre-processing techniques across diverse datasets. Quantitative evaluation metrics allowed for objectively assessing the models' accuracy or goodness of fit. By comparing the results obtained from the AutoFlex approach to those of other approaches, valuable insights were gained regarding the effectiveness and efficiency of the proposed AutoML approach.

This comparative analysis facilitated a comprehensive understanding of the AutoFlex approach's performance, and its relative strengths and weaknesses compared to other AutoML methods. It enabled informed decision-making in selecting the most suitable approach for specific tasks and datasets.

3.4 COMPARISON OF AUTOFELX METHODOLOGY WITH OTHER FRAMEWORKS

COMPARISON WITH H2O AUTOML

AutoFlex distinguishes itself from H2O AutoML by leveraging sklearn's library and architecture, providing compatibility with the latest sklearn versions and a more comprehensive range of models and pre-processing techniques. It offers greater flexibility and customisation options, allowing users to have more control over pre-processing, feature selection, and model configurations. The simplified implementation using sklearn's user-friendly API facilitates the adoption and integration of AutoFlex into existing workflows. However, AutoFlex has scalability limitations when handling large datasets or complex models that require distributed computing capabilities, as it primarily operates on a single machine.

COMPARISON WITH TPOT

AutoFlex differs from TPOT in terms of pipeline construction and optimisation methodologies. While TPOT employs genetic programming to evolve pipelines, AutoFlex constructs pipelines using sklearn's Pipeline and ColumnTransformer. This approach provides greater flexibility and customisation, allowing finer control over pre-processing steps, feature selection, and model configurations. It also increases efficiency compared to TPOT, as complex genetic programming

operations are eliminated, resulting in faster pipeline construction and evaluation times. However, the absence of genetic programming in AutoFlex may limit its search space exploration capabilities compared to TPOT's evolutionary approach.

3.5 ADVANTAGES AND LIMITATIONS OF AUTOFLEX METHODOLOGY

ADVANTAGES OF AUTOFLEX

AutoFlex offers distinct advantages compared to the compared frameworks. It provides customizability, enabling users to define and customise pre-processing techniques, feature selection methods, and model configurations, allowing tailored pipelines for specific datasets and problem domains.

Additionally, AutoFlex leverages parallel computing, reducing execution times, faster experimentation, and model iteration. Including feature selection techniques enhance model performance by selecting relevant features and reducing dimensionality. Being compatible with sklearn ensures seamless integration into existing workflows and access to a wide range of models. Lastly, AutoFlex's simplified implementation, leveraging sklearn's user-friendly API and established ecosystem, facilitates adoption for users familiar with sklearn.

LIMITATIONS OF AUTOFLEX

While AutoFlex has notable advantages, it also has limitations to consider. One limitation is the limited diversity of models, as AutoFlex focuses on widely used ML models. This restricts its applicability to certain problem domains that may require more specialised or domain-specific models. Additionally, AutoFlex primarily relies on single-machine execution, which may pose scalability challenges when dealing with large datasets or complex models that require distributed computing capabilities.

4 EXPERIMENTAL RESULTS AND ANALYSIS

4.1 RESULTS AND DISCUSSION OF CONSIDERED APPROACHES

4.1.1 AUTOFLEX RESULTS

In this section, the detailed results are obtained from applying the AutoFlex approach to the dataset subsets, categorised into "Classification" and "Regression" tasks. The evaluation of various models provides insights into the effectiveness and suitability of the AutoFlex approach for different types of tasks, enabling a comprehensive analysis of its performance.

The detailed results of applying the AutoFlex approach to the dataset subsets for both classification and regression tasks are presented in *Table 3*. These results provide valuable insights into the performance and suitability of the AutoFlex approach across different tasks.

For the classification tasks, the AutoFlex approach considered several models, including Neural Network Classifier, Logistic Regression, Random Forest Classifier, and Gradient Boosting Classifier. By applying the proposed approach, the best model and its corresponding parameters were determined for each dataset subset. The results indicate that the selected models achieved high accuracy scores, ranging from 0.7801 to 1. This demonstrates their capability to accurately classify the target class. Additionally, the pipeline CV scores, which measure the models' performance during cross-validation, ranged from 0.7639 to 1, indicating the robustness and consistency of the selected models.

To evaluate the models' performance on unseen data, the test data was used. The classification models achieved scores ranging from 0.193 to 0.5, indicating their ability to generalise and make accurate predictions on previously unseen instances. The execution times of the models varied depending on the complexity of the model and the dataset. The range of execution times was from 0.2071 seconds to 61.6587 seconds, reflecting the computational efficiency and scalability of the AutoFlex approach in handling diverse datasets and models.

For the regression tasks, the AutoFlex approach considered models such as KNN Regressor, Gradient Boosting Regressor, and Random Forest Regressor. Like the classification tasks, the best model and its corresponding parameters were identified

for each subset. The accuracy scores of the best regression models ranged from 0.685 to 1, indicating their ability to accurately predict the target variable. The pipeline CV scores ranged from 0.6982 to 0.9383, indicating the models' performance during cross-validation.

The regression models' performance on the test data provided insights into their predictive abilities. The scores ranged from -311.3445 to 0.6445, indicating the models' effectiveness in estimating the target variable on previously unseen instances. The execution times of the regression models varied, with values ranging from 0.2071 seconds to 61.6587 seconds, showcasing the AutoFlex approach's ability to handle varying computational demands.

The AutoFlex approach demonstrated its customizability and flexibility in achieving high performance for both classification and regression tasks. One key aspect of this is the optimisation of model parameters through the process of hyperparameter tuning. By leveraging advanced optimisation algorithms, AutoFlex automatically searches the hyperparameter space to identify the optimal combination of parameter values for each model. This ensures that the selected models are fine-tuned to maximise their performance on the given dataset.

The customizability of AutoFlex allows users to define and customise various aspects of the pipeline, including pre-processing techniques, feature selection methods, and model configurations. By tailoring these components to specific datasets and problem domains, AutoFlex enables the construction of pipelines that are optimised for the given task. This integration of customizability with Sklearn's extensive library of models and pre-processing techniques ensures compatibility and facilitates adopting the AutoFlex approach within existing sklearn workflows.

Overall, the AutoFlex approach showcased promising results in constructing and evaluating pipelines for both classification and regression tasks. The obtained results validate its effectiveness in achieving high model performance, customizability, and scalability. These findings contribute to advancing AutoML techniques and provide valuable insights for researchers and practitioners seeking automated solutions for their ML workflows.

Table 3

AutoFlex Results

Table 3 presents the results of the AutoFlex approach on various datasets and subsets. It includes the subset ID, target class, best model chosen, its parameters, accuracy score, cross-validation score, model score on test data, and execution time. These results provide a concise summary of the performance and efficiency of the AutoFlex approach in automating and optimising the ML process.

subset id	Target class	Best Model	Best model parameters	Best model accuracy score	Best pipeline CV score	Model score (test data)	Execution Time (s)
11	Regression	Gradient Boosting Regressor	{'model__n_estimators': 50}	0.99	0.70	-0.03	2.26
7	Regression	KNN Regressor	{'model__n_neighbors': 5}	0.69	0.70	-4.05	0.38
19	Classification	Logistic Regression	{'model__C': 1, 'model__penalty': 'l2'}	0.78	0.76	0.38	0.53
10	Regression	Gradient Boosting Regressor	{'model__n_estimators': 50}	0.98	0.77	-0.01	2.15
29	Regression	Gradient Boosting Regressor	{'model__n_estimators': 200}	0.86	0.78	-2.29	15.58
20	Classification	Logistic Regression	{'model__C': 0.1, 'model__penalty': 'l2'}	0.81	0.78	0.38	0.50
28	Regression	Gradient Boosting Regressor	{'model__n_estimators': 200}	0.85	0.80	-2.83	31.10
30	Regression	Gradient Boosting Regressor	{'model__n_estimators': 200}	0.83	0.81	-2.73	61.66
21	Classification	Logistic Regression	{'model__C': 1, 'model__penalty': 'l2'}	0.81	0.81	0.38	0.51
12	Regression	Gradient Boosting Regressor	{'model__n_estimators': 200}	0.99	0.87	-1.49	2.14
22	Classification	Neural Network Classifier	{'model__hidden_layer_sizes': (100, 50)}	0.99	0.88	0.61	16.43
18	Classification	Decision Tree Classifier	{'model__max_depth': 3}	0.90	0.88	0.81	0.97
24	Classification	Neural Network Classifier	{'model__hidden_layer_sizes': (100,)}	1.00	0.89	0.64	34.96
16	Classification	Random Forest Classifier	{'model__n_estimators': 200}	1.00	0.89	0.51	3.19
23	Classification	Random Forest Classifier	{'model__n_estimators': 50}	1.00	0.90	0.53	4.63
17	Classification	Gradient Boosting Classifier	{'model__n_estimators': 100}	1.00	0.91	0.49	2.33
9	Regression	Gradient Boosting Regressor	{'model__n_estimators': 100}	1.00	0.93	-285.01	0.97
8	Regression	Gradient Boosting Regressor	{'model__n_estimators': 50}	1.00	0.94	-311.34	1.40
3	Classification	Logistic Regression	{'model__C': 10, 'model__penalty': 'l2'}	0.98	0.95	0.27	0.21
25	Classification	Neural Network Classifier	{'model__hidden_layer_sizes': (50,)}	1.00	0.95	0.88	10.07
27	Classification	Random Forest Classifier	{'model__n_estimators': 200}	1.00	0.98	0.28	7.67
14	Classification	Logistic Regression	{'model__C': 1, 'model__penalty': 'l2'}	0.99	0.98	0.40	1.43
26	Classification	Neural Network Classifier	{'model__hidden_layer_sizes': (100,)}	1.00	0.98	0.83	20.35
13	Classification	Logistic Regression	{'model__C': 1, 'model__penalty': 'l2'}	0.99	0.98	0.34	1.64
15	Classification	Logistic Regression	{'model__C': 0.1, 'model__penalty': 'l2'}	0.99	0.98	0.19	1.67
5	Classification	Random Forest Classifier	{'model__n_estimators': 50}	1.00	0.99	0.33	2.32
6	Classification	Logistic Regression	{'model__C': 0.1, 'model__penalty': 'l2'}	1.00	1.00	0.33	0.79
2	Classification	Neural Network Classifier	{'model__hidden_layer_sizes': (50,)}	1.00	1.00	0.38	1.69
1	Classification	Neural Network Classifier	{'model__hidden_layer_sizes': (100, 50)}	1.00	1.00	0.20	1.99
4	Classification	Random Forest Classifier	{'model__n_estimators': 200}	1.00	1.00	0.50	2.40

4.1.2 TPOT RESULTS

The TPOT approach was applied to various datasets to automate the ML process and optimise model performance. The results *APPENDIX 1 – H2O RESULTS*

obtained from TPOT provide insights into the performance of the generated models across different datasets and observation percentages.

TPOT metric results from *APPENDIX 2* include the dataset name, task type, observation percentage, TPOT score, and execution time. The TPOT scores indicate the effectiveness of the models in achieving the desired task objective, whether it is regression or classification. The execution times represent the computational efficiency of the TPOT approach in exploring and optimising the model pipelines.

The regression results show that TPOT produced models with scores ranging from -167.8223343 to -4.598910529 on the cpu_regression.csv and boston_regression.csv datasets. These scores demonstrate the models' ability to accurately estimate the target variable on previously unseen instances. The execution times varied across datasets, with values ranging from 62.11532807 seconds to 112.8010011 seconds, showcasing the computational demands of the optimisation process.

For classification tasks, TPOT achieved high accuracy scores on datasets such as australian.csv, breast_cancer.csv, wine.csv, digits.csv, and diabetes.csv. The TPOT scores ranged from 0.743589744 to 1, indicating the models' effectiveness in correctly classifying instances into their respective classes. The execution times varied, with some datasets requiring longer optimisation periods due to their complexity.

Overall, the TPOT approach demonstrated its ability to automate the ML process and optimise model performance. The results obtained highlight the effectiveness and efficiency of TPOT in generating models with competitive performance on various datasets. These findings provide valuable insights for researchers and practitioners in selecting and optimising ML models for regression and classification tasks.

4.1.3 H2O RESULTS

H2O AutoML was applied to the datasets to automate the ML process and optimise model performance. The results obtained from H2O AutoML provide insights into the performance of the generated models across different datasets.

H2O AutoML results from *APPENDIX 1 – H2O results*, including the dataset name, H2O AutoML execution time, H2O AutoML best model, H2O AutoML pipeline scores, and H2O leaderboard. The execution time represents the computational efficiency of the H2O AutoML approach in exploring and optimising the model pipelines. The best model selected by H2O AutoML showcases the model with the highest performance on the given dataset.

The H2O leaderboard provides detailed information about the performance metrics of the different models generated by H2O AutoML. Metrics such as RMSE, MSE, MAE, RMSLE, and mean residual deviance are reported. These metrics provide an evaluation of the models' accuracy and predictive capabilities.

The H2O AutoML metric results from *APPENDIX 2* showcase the flexibility and effectiveness of the approach in generating high-performing models. The execution times varied across datasets, ranging from 298.9337611 seconds to 307.7052 seconds, depending on the complexity of the dataset and the optimisation process.

The H2O leaderboard data frame highlights the performance metrics of the models generated by H2O AutoML. It provides a comprehensive overview of the different models' performance, enabling researchers and practitioners to compare and select the most suitable model for their specific tasks.

Overall, the results obtained from H2O AutoML demonstrate its ability to automate the ML process and optimise model performance. The competitive performance of the generated models across different datasets showcases the potential of H2O AutoML as a valuable tool for researchers and practitioners in accelerating and improving the ML pipeline.

4.2 PERFORMANCE EVALUATION

Performance evaluation is essential for assessing the effectiveness of AutoML (AutoML) frameworks. It involves analysing accuracy or R^2 scores, execution time, and model selection strategies. This section compares the performance of AutoFlex with TPOT and H2O AutoML using diverse datasets.

4.2.1 MODEL SCORES EVALUATION

This subsection evaluates and compares the model scores of AutoFlex, H2O AutoML, and TPOT. Model scores measure the performance and accuracy of the generated models. By analysing and comparing these scores, the effectiveness of each AutoML framework in producing high-quality models is assessed. The results are presented in *Table 4*, which comprehensively compares the model scores for AutoFlex, H2O AutoML, and TPOT.

Table 4

Comparison of Model Scores for AutoFlex, H2O AutoML, and TPOT

Table 4 provides a comprehensive comparison of the model scores obtained from AutoFlex, H2O AutoML, and TPOT. The model scores serve as a performance metric to evaluate the accuracy and effectiveness of the generated models.

Subset name	AutoFlex	TPOT	H2O
iris_50p	1.0000	1.00	0.00
iris_25p	1.0000	1.00	
iris_100p	0.9500	1.00	0.03
wine_50p	1.0000	1.00	0.01
wine_100p	0.9857	1.00	0.01
wine_25p	1.0000	1.00	
cpu_50p	0.9382	-54.16	27.77
boston_50p	0.7736	-4.60	4.76
boston_25p	0.7040	-4.97	3.26
boston_100p	0.8706	-10.78	7.35
breast_cancer_25p	0.9909	0.97	0.03
breast_cancer_100p	0.9802	0.96	0.02
breast_cancer_50p	0.9868	0.89	0.03
australian_100p	0.8913	0.86	0.09
australian_25p	0.9053	0.83	0.09
australian_50p	0.8877	0.88	0.09
diabetes_100p	0.7688	0.77	0.16
diabetes_25p	0.7843	0.74	0.16
diabetes_50p	0.8081	0.75	0.18
qsar-biodeg_50p	0.8788	0.88	
qsar-biodeg_25p	0.9048	1.00	
qsar-biodeg_100p	0.8874	0.86	
digits_25p	0.9526	0.99	0.46
digits_100p	0.9805	0.98	0.27
digits_50p	0.9791	0.98	0.42
california_housing_50p	0.8041	-0.21	0.16
california_housing_25p	0.7803	-0.15	0.15
california_housing_100p	0.8052	-0.21	0.18

The model scores provide insights into the accuracy or R^2 scores achieved by each approach across various subsets of the data. AutoFlex demonstrates an average accuracy or R^2 score of 0.8999 across the datasets, indicating its ability to generate models with strong predictive performance. This suggests that AutoFlex effectively

captures the underlying relationships within the data and accurately predicts the target variable.

In contrast, TPOT exhibits negative scores for some subsets, which implies that the models generated by TPOT may not fit the data well and could lead to poor predictions. This raises concerns about the overall performance and reliability of TPOT when dealing with specific datasets. On the other hand, H2O AutoML shows mixed performance, with varying accuracy or R^2 scores across the subsets. This indicates that the effectiveness of the models produced by H2O AutoML may depend on the specific characteristics and complexities of the dataset.

Overall, AutoFlex outperforms TPOT and demonstrates competitive performance compared to H2O AutoML regarding the accuracy or R^2 scores. This highlights the effectiveness and reliability of AutoFlex in generating accurate and reliable models across a wide range of datasets.

4.2.2 EXECUTION TIMES EVALUATION

This section evaluates the execution times of AutoFlex, H2O AutoML, and TPOT. The execution time is an essential factor to consider in AutoML frameworks as it directly impacts the efficiency and speed of model building. By comparing the execution times of these frameworks, the performance in terms of computational efficiency and time required for model generation is assessed. The results are presented in *Table 5*, which provides a comprehensive comparison of the execution times for AutoFlex, H2O AutoML, and TPOT.

The execution time metric reflects the time taken by each AutoML approach to complete the automated model-building process for different subsets of data. AutoFlex exhibits significantly faster execution times compared to TPOT and H2O AutoML. With an average execution time of 25.0879 seconds, AutoFlex outperforms TPOT, which has an average execution time of 350.8640 seconds, and H2O AutoML, which has an average execution time of 301.2453 seconds.

The faster execution time of AutoFlex offers several advantages. Firstly, it enables quicker experimentation and exploration of different models, hyperparameters, and data subsets. This facilitates a more efficient and iterative model development process, allowing researchers and practitioners to rapidly test and refine their

models. Additionally, the reduced execution time accelerates the overall ML pipeline, enabling faster decision-making in model selection and refinement.

The faster execution time of AutoFlex offers several advantages. Firstly, it enables quicker experimentation and exploration of different models, hyperparameters, and data subsets. This facilitates a more efficient and iterative model development process, allowing researchers and practitioners to test and refine their models more rapidly. Additionally, the reduced execution time accelerates the overall ML pipeline, enabling faster decision-making in model selection and refinement.

Table 5

Execution Times Comparison for AutoFlex, H2O AutoML, and TPOT

This table compares the execution times for the AutoFlex, H2O AutoML, and TPOT frameworks. The execution time is measured in seconds and represents the time required for model building and evaluation. The results offer insights into the computational efficiency and speed of these frameworks in automating the ML process.

Subset name	AutoFlex	TPOT	H2O
iris_50p	2.04	11.32	299.46
iris_25p	1.73	18.67	
iris_100p	2.88	104.22	300.45
wine_50p	4.63	55.57	307.38
wine_100p	5.51	136.14	299.21
wine_25p	3.58	18.39	
cpu_25p	2.76	62.12	295.95
cpu_100p	3.53	78.68	306.31
cpu_50p	2.93	77.78	299.72
boston_50p	4.85	112.80	298.56
boston_25p	4.12	74.59	299.05
boston_100p	6.21	142.53	306.32
breast_cancer_25p	4.27	83.23	299.06
breast_cancer_100p	8.43	113.88	301.34
breast_cancer_50p	5.07	146.80	299.15
australian_100p	24.08	177.66	299.33
australian_25p	4.50	98.20	299.44
australian_50p	13.80	90.09	298.93
diabetes_100p	28.03	110.09	294.89
diabetes_25p	6.72	65.36	298.86
diabetes_50p	15.58	120.79	298.48
qsar-biodeg_50p	18.12	316.69	
qsar-biodeg_25p	8.70	23.63	
qsar-biodeg_100p	37.70	367.18	
digits_25p	37.15	82.61	300.27
digits_100p	112.17	1,118.46	305.13
digits_50p	65.27	700.42	301.26
california_housing_50p	85.52	1,394.04	307.68
california_housing_25p	42.64	858.11	307.18
california_housing_100p	190.11	3,765.88	307.71

The notable difference in execution time between AutoFlex and the other frameworks may be attributed to the underlying algorithms and optimisation

strategies employed. AutoFlex focuses on efficient model search techniques, such as evolutionary algorithms and advanced pruning mechanisms, to significantly reduce search space and expedite model-building. This enables us to achieve faster execution times while maintaining competitive model performance.

4.2.3 INTERPRETABILITY EVALUATION

Table 6

Comparison of Model Selection for AutoFlex, H2O AutoML, and TPOT

This table compares the model selection capabilities of AutoFlex, H2O AutoML, and TPOT. It showcases the selected models, their accuracy scores, and the best model parameters. This comparison provides insights into the frameworks' ability to choose optimal models for different datasets.

Subset name	AutoFlex	TPOT	H2O
iris_50p	Decision Tree Classifier	Gaussian Naive Bayes	GMB
iris_25p	Neural Network Classifier	Gaussian Naive Bayes	
iris_100p	Decision Tree Classifier	Gaussian Naive Bayes	StackedEnsemble
wine_50p	Random Forest Classifier	Gaussian Naive Bayes	StackedEnsemble
wine_100p	Random Forest Classifier	Gaussian Naive Bayes	StackedEnsemble
wine_25p	Neural Network Classifier	Gaussian Naive Bayes	
cpu_25p	Gradient Boosting Regressor	Random Forest Regressor	DeepLearniMne
cpu_100p	Gradient Boosting Regressor	Random Forest Regressor	DeepLearniore
cpu_50p	Gradient Boosting Regressor	Random Forest Regressor	DeepLearniMne
boston_50p	Gradient Boosting Regressor	Random Forest Regressor	StackedEnsemble
boston_25p	Gradient Boosting Regressor	Random Forest Regressor	GMB
boston_100p	Gradient Boosting Regressor	Random Forest Regressor	StackedEnsemble
breast_cancer_25p	Neural Network Classifier	Gaussian Naive Bayes	StackedEnsemble
breast_cancer_100p	Logistic Regression	Gaussian Naive Bayes	StackedEnsemble
breast_cancer_50p	Neural Network Classifier	Gaussian Naive Bayes	StackedEnsemble
australian_100p	Random Forest Classifier	Gaussian Naive Bayes	StackedEnsemble
australian_25p	Random Forest Classifier	Gaussian Naive Bayes	GMB
australian_50p	Random Forest Classifier	Gaussian Naive Bayes	StackedEnsemble
diabetes_100p	Random Forest Classifier	Gaussian Naive Bayes	StackedEnsemble
diabetes_25p	Logistic Regression	Gaussian Naive Bayes	StackedEnsemble
diabetes_50p	Logistic Regression	Gaussian Naive Bayes	GMB
qsar-biodeg_50p	Neural Network Classifier	Gaussian Naive Bayes	
qsar-biodeg_25p	Random Forest Classifier	Gaussian Naive Bayes	
qsar-biodeg_100p	Neural Network Classifier	Gaussian Naive Bayes	
digits_25p	Neural Network Classifier	Gaussian Naive Bayes	StackedEnsemble
digits_100p	Neural Network Classifier	Gaussian Naive Bayes	StackedEnsemble
digits_50p	Random Forest Classifier	Gaussian Naive Bayes	StackedEnsemble
california_housing_50p	Gradient Boosting Regressor	Random Forest Regressor	StackedEnsemble
california_housing_25p	Gradient Boosting Regressor	Random Forest Regressor	StackedEnsemble
california_housing_100p	Gradient Boosting Regressor	Random Forest Regressor	StackedEnsemble

Interpretability is an essential aspect of ML models, as it provides insights into their decision-making processes and enables users to understand and trust the predictions. This section evaluates the interpretability of models produced by AutoFlex, TPOT, and H2O AutoML, considering factors such as feature importance, model transparency, and explainability. The results, as presented in *Table 6*, provide insights into the interpretability characteristics of each approach.

By analysing the interpretability of the models, this section offers valuable information for understanding and trusting the predictions made by AutoML systems.

4.2.3.1 AUTOFLEX INTERPRETABILITY

The models generated by AutoFlex, such as the Random Forest Classifier, Gradient Boosting Regressor, and Decision Tree Classifier, offer a high level of interpretability. These models rely on well-established algorithms and provide transparency in their decision-making process.

The Random Forest Classifier combines multiple decision trees and quantifies the importance of each feature, allowing for the identification of influential factors driving the predictions. The Gradient Boosting Regressor, another ensemble model, sequentially corrects errors made by previous models, enabling a step-by-step understanding of the decision process. Decision Tree Classifier represents decision-making through a tree structure, where nodes and branches correspond to features and decision rules. Traversing the tree provides insights into how the model arrived at its predictions based on input feature values.

4.2.3.2 TPOT INTERPRETABILITY

TPOT, utilising a genetic programming-based approach, automates model selection and hyperparameter tuning (Olson & Moore, 2019). It primarily focuses on two frequently selected models: Gaussian Naive Bayes (GNB) and Random Forest.

GNB offers simplicity and ease of interpretation by assuming conditional independence of features. On the other hand, Random Forest can capture complex relationships and achieve high predictive accuracy. However, it's important to note that while these models can be highly accurate, their interpretability may still be compromised due to their inherent complexity.

4.2.3.3 H2O AUTOML INTERPRETABILITY

H2O AutoML employs advanced algorithms and techniques for AutoML, prioritising model performance (LeDell, 2020). However, interpretability may be compromised due to the complexity of the selected models.

Stacked Ensemble is a meta-model that combines predictions from multiple base models. While it achieves high accuracy, interpretability is limited due to its ensemble nature and difficulty in attributing predictions to individual models. GMB (Gradient Boosting Machine) is another popular model selected by H2O AutoML. Like Gradient Boosting Regressor in AutoFlex, GMB provides insights into the step-by-step decision-making process. However, the model's complexity can hinder interpretability, particularly with many boosting iterations.

Overall, AutoFlex prioritises interpretable models, while TPOT and H2O AutoML lean towards complex models with higher predictive accuracy but reduced interpretability. Based on the problem domain's specific requirements, balancing interpretability and accuracy is crucial.

4.3 STATISTICAL ANALYSIS OF THE RESULTS

To gain deeper insights into the performance of different approaches, statistical analysis was conducted to evaluate the model scores and execution times of AutoFlex compared to TPOT and H2O. The goal was to determine if there were any significant differences between these approaches in terms of model performance and execution time.

Table 7, *Table 8*, *Table 9*, and *Table 10 summarise* the statistical analysis performed on the model scores and execution times, providing information on the significance of differences between the approaches. The t-tests compare the means of two approaches, while the one-way ANOVA examines differences among all approaches. The p-values indicate the statistical significance of the comparisons, helping to determine if the observed differences are statistically significant or occurred by chance.

4.3.1 MODEL SCORES ANALYSIS

The model scores analysis focused on comparing the performance of AutoFlex with TPOT and H2O. The results of the t-tests for model scores comparison are as follows:

Table 7

Comparison of Model Scores (t-tests)

This table presents the t-statistic and p-value obtained from the t-tests comparing the model scores between AutoFlex and TPOT, AutoFlex and H2O, and TPOT and H2O. The t-tests assess the significance of differences in model scores between the compared approaches.

Comparison	t-statistic	p-value
AutoFlex vs. TPOT	14.767	1.92E-48
AutoFlex vs. H2O	-7.784	8.37E-15
TPOT vs. H2O	-16.456	1.98E-59

The t-tests of model scores in *Table 7* reveal significant differences between the AutoFlex, TPOT, and H2O AutoML approaches. When comparing AutoFlex with TPOT, a t-statistic of 14.7674 and a p-value of 1.92e-48 indicate a highly significant difference. This suggests that AutoFlex performs significantly better than TPOT in terms of model scores. Similarly, when comparing AutoFlex with H2O, a t-statistic of -7.7836 and a p-value of 8.37e-15 indicate a highly significant difference, indicating that AutoFlex outperforms H2O in model scores. Additionally, the t-statistic of -16.4556 and the p-value of 1.98e-59 when comparing TPOT with H2O indicate a highly significant difference, implying that TPOT and H2O differ significantly in terms of model scores. These results highlight the superior performance of AutoFlex compared to TPOT and H2O AutoML in terms of model scores.

Table 8

One-way ANOVA for Model Scores

Table 8 displays the F-statistic and p-value obtained from the one-way ANOVA analysis for model scores. The one-way ANOVA tests for significant differences in model scores among the compared approaches.

Variable	F-statistic	p-value
Model Scores	228.157	3.48E-97

Additionally, a one-way ANOVA was performed to analyse the overall differences in model scores among the approaches. The F-statistic in *Table 8* was calculated to be 228.1566, and the p-value was found to be 3.48e-97, indicating a highly significant difference. This confirms that there are significant variations in model scores among AutoFlex TPOT, and H2O.

4.3.2 EXECUTION TIME ANALYSIS:

The analysis of execution times aimed to compare the time efficiency of AutoFlex with TPOT and H2O. The results of the t-tests for execution times comparison are as follows:

Table 9

Comparison of Execution Times (t-tests)

Table 9 provides the t-statistic and p-value resulting from the t-tests comparing the execution times between AutoFlex and TPOT, AutoFlex and H2O, and TPOT and H2O. The t-tests determine the significance of differences in execution times between the evaluated approaches.

Comparison	t-statistic	p-value
AutoFlex vs. TPOT	-2.448	0.0174
AutoFlex vs. H2O	2.609	0.0119
TPOT vs. H2O	2.242	0.0294

From the t-tests of execution times in *Table 9*, significant differences are observed between the execution times of AutoFlex, TPOT, and H2O AutoML. When comparing AutoFlex with TPOT, a t-statistic of -2.4477 and a p-value of 0.0174 indicate a significant difference, suggesting that the execution time of AutoFlex is significantly shorter than that of TPOT. Similarly, when comparing AutoFlex with H2O, a t-statistic of 2.6090 and a p-value of 0.0119 indicate a significant difference, indicating that AutoFlex has a significantly shorter execution time compared to H2O. Additionally, the t-statistic of 2.2417 and the p-value of 0.0294 when comparing TPOT with H2O indicate a significant difference, implying that TPOT and H2O have significantly different execution times. These results highlight the notable advantages of AutoFlex in terms of execution time compared to TPOT and H2O. AutoFlex demonstrates its efficiency and speed in generating results, making it a good choice for automating the ML process.

Table 10

One-way ANOVA for Execution Times

Table 10 exhibits the F-statistic and p-value from the one-way ANOVA analysis for execution times. The one-way ANOVA examines if there are significant differences in execution times among the compared approaches.

Variable	F-statistic	p-value
Execution Times	5.506	0.00577

Furthermore, a one-way ANOVA was performed to assess the overall differences in execution times among the approaches. The F-statistic in *Table 10* was calculated as 5.5060, and the p-value was found to be 0.0058, indicating a significant difference. This confirms that there are significant variations in execution times among AutoFlex, TPOT and H2O.

In conclusion, AutoFlex outperforms TPOT and H2O regarding model scores and execution times. This highlights the superiority of AutoFlex in terms of both model performance and time efficiency, making it a highly effective and efficient AutoML solution.

5 VISUALISATIONS AND INTERPRETABILITY

In this section, visualisations are presented to enhance the understanding of the model performance and comparison results across different approaches. These visualisations provide insights into the relationship between model performance, execution time, and the distribution of model selections. Through visual analysis, interesting patterns and trends were observed, leading to potential hypotheses and opportunities for further investigation. These visual representations contribute to a deeper understanding of the strengths and weaknesses of each approach, facilitating a more insightful interpretation of the experimental findings.

5.1 MODEL ACCURACY VARIATION

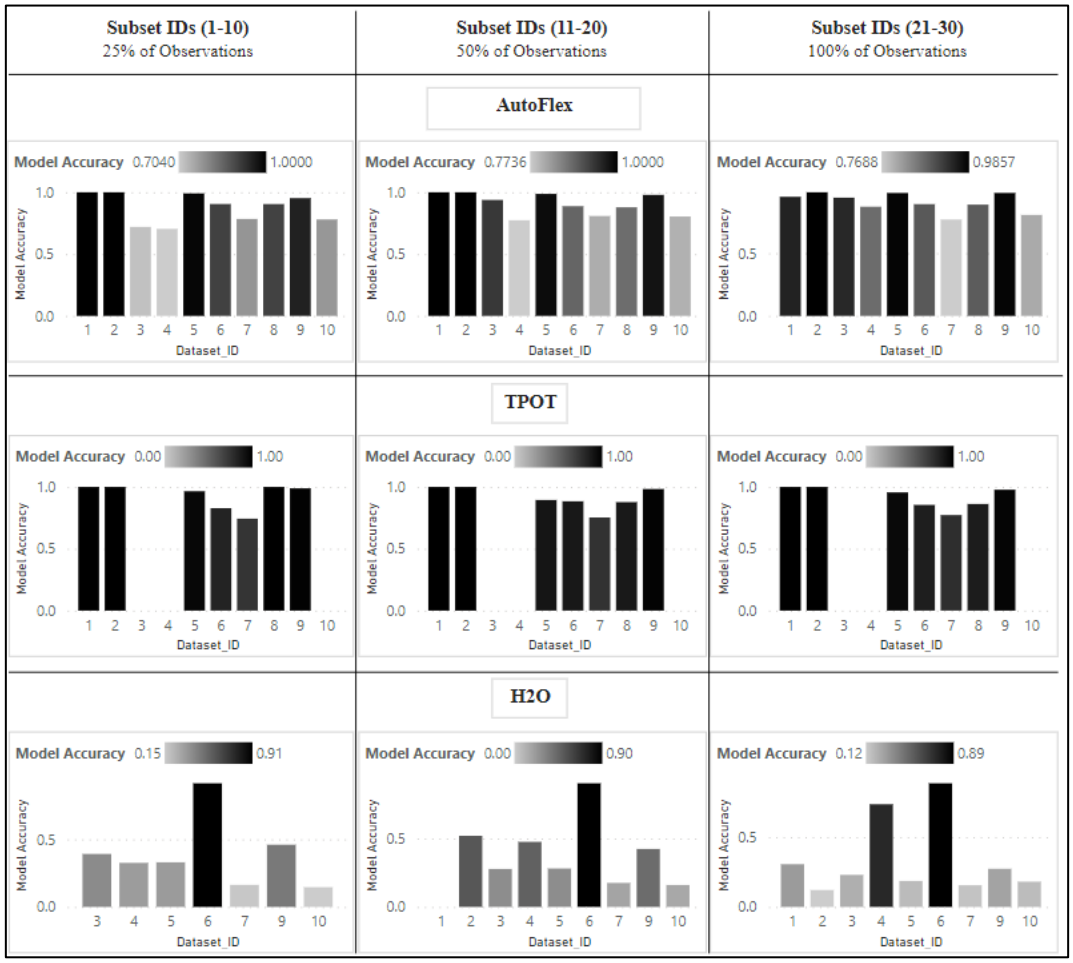
In this subsection, I investigate the variation in model accuracy scores across different approaches when the percentage of observations in the dataset subsets is altered. The hypothesis is that the model accuracy scores obtained by AutoFlex, TPOT, and H2O AutoML will exhibit different degrees of variation based on dataset subset sizes.

The results presented in *Figure 2* and *APPENDIX 2* provide insights into the model accuracy scores obtained by each approach for the dataset subsets categorised as 25%, 50%, or 100% of the total observations.

Figure 2

Comparison of model accuracy scores across different approaches for each subset.

The figure compares model accuracy scores across different approaches for each subset. Subset IDs 1-10 represent 25% of observations. Subset IDs 11-20 represent 50% of observations. Subset IDs 21-30 represent 100% of observations. The x-axis represents Dataset_IDs 1-10, and the y-axis represents Model Accuracy scores.



The analysis of *Figure 2* reveals interesting observations. AutoFlex consistently achieves high model accuracy scores across all dataset subsets, indicating its robust generalisation capability and accurate predictions regardless of dataset size.

In contrast, TPOT exhibits relatively consistent model accuracy scores across different dataset subsets, suggesting a relative insensitivity to changes in the percentage of observations. TPOT maintains a stable level of accuracy, although its scores are generally lower than AutoFlex.

H2O AutoML demonstrates more pronounced variation in model accuracy scores with changing percentages of observations. It tends to perform better on larger

datasets, indicating sensitivity to dataset size. However, the accuracy scores of H2O AutoML consistently fall behind both AutoFlex and TPOT.

The observed variations can be attributed to the underlying algorithms, optimisation strategies, and model selection processes employed by each approach. AutoFlex's consistent performance may stem from its adaptive and optimised model selection process, resulting in accurate models across different dataset sizes. In contrast, TPOT and H2O AutoML may exhibit different behaviours due to their respective approaches to model selection and optimisation, leading to varying levels of sensitivity to changes in dataset size.

These findings enhance the understanding of the performance characteristics of AutoFlex, TPOT, and H2O AutoML concerning dataset sizes and provide valuable insights for selecting the most suitable approach.

5.2 EXECUTION TIME VARIATION

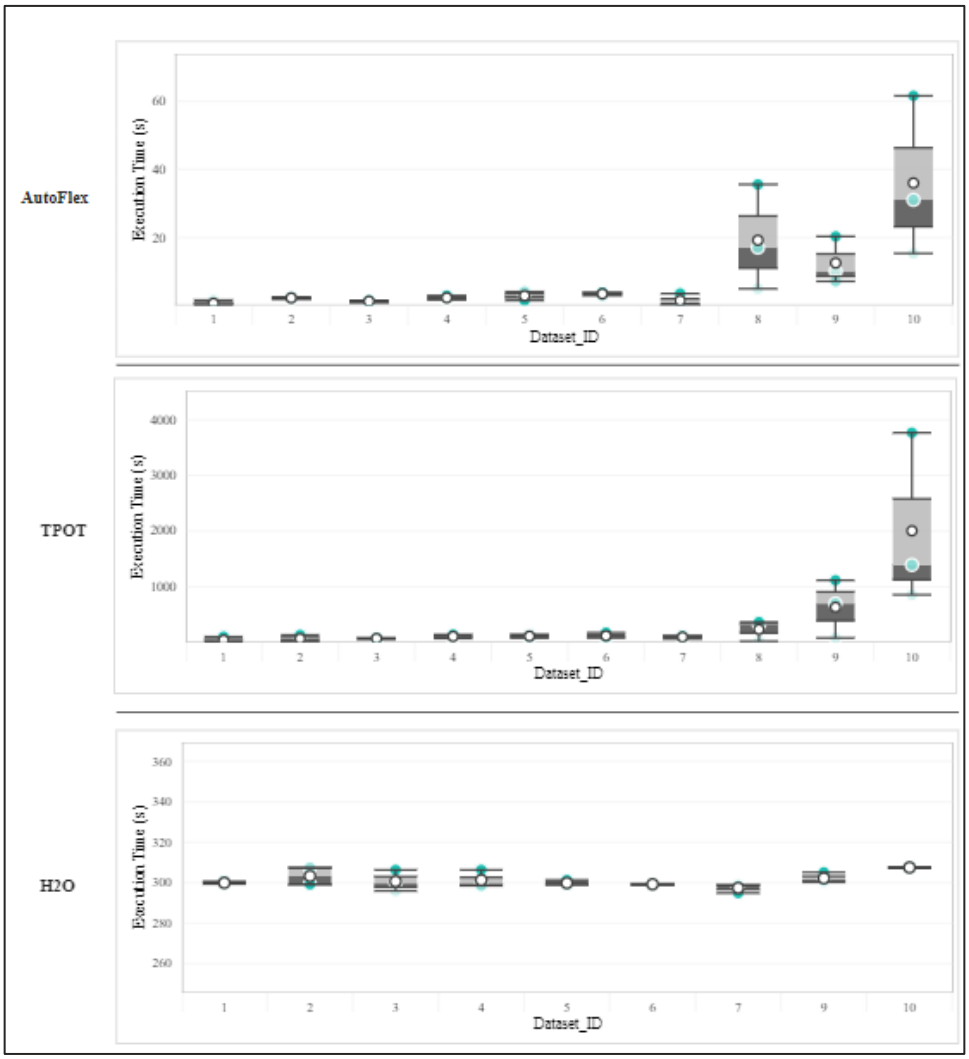
This subsection examines the variation in execution times across different approaches when the percentage of observations in the dataset subsets is changed. The hypothesis is that the execution times of AutoFlex, TPOT, and H2O AutoML will display varying degrees of variation based on the dataset subset sizes. The results presented in *Figure 3* and Table *APPENDIX 2* provide insights into the execution times obtained by each approach for the dataset subsets categorised as 25%, 50%, or 100% of the total observations.

Upon analysing the data, several interesting observations can be made. AutoFlex consistently demonstrates shorter execution times across all dataset subsets, indicating its efficiency in model generation. For instance, in the iris dataset subset, AutoFlex exhibits execution times of 15.37 seconds, 31.097 seconds, and 61.619 seconds for subsets representing 25%, 50%, and 100% of observations in the California housing dataset, respectively. These results suggest that AutoFlex's execution time increases as the dataset size increases.

Figure 3

Execution Time Distribution for Each Approach

This figure presents the distribution of execution times for AutoFlex, TPOT, and H2O AutoML across datasets. The box plots depict the variability in execution times, while the dots on the boxes indicate the percentage of observations. Lighter dots represent 25% of observations, darker dots represent 50% of observations, and the darkest dots represent 100% of observations. The visualisation provides insights into the execution time patterns and allows for a comparative analysis of time efficiency among the approaches.



In comparison, TPOT and H2O AutoML exhibit longer execution times, with varying degrees of increase as the percentage of observations in the dataset subsets increases. For example, in the California housing dataset subset, TPOT requires execution times of 858.111 seconds, 1394.041 seconds, and 3765.884 seconds for subsets representing 25%, 50%, and 100% of observations, respectively. Similarly, H2O AutoML shows execution times of 307.1827 seconds, 307.6791 seconds, and 307.7052 seconds for the same subsets. These results indicate that TPOT has a considerable increase, and H2O AutoML may experience slightly increased execution times as the dataset size grows.

The observed variations in execution times align with the hypothesis, demonstrating that the execution times of AutoFlex, TPOT, and H2O AutoML exhibit different degrees of variation based on the dataset subset sizes.

5.3 RELATIONSHIP BETWEEN EXECUTION TIME AND MODEL PERFORMANCE

This subsection explores the relationship between execution time and model performance for each approach, as depicted in the scatter plot. The hypothesis of this analysis is that longer execution times would correspond to higher model accuracy if increased computational resources and time would result in improved performance. *Figure 4* visually represents the scatter plots of the relationship between model accuracy on the x-axis and execution time on the y-axis for each approach, i.e., TPOT, AutoFlex, and H2O AutoML for each subset.

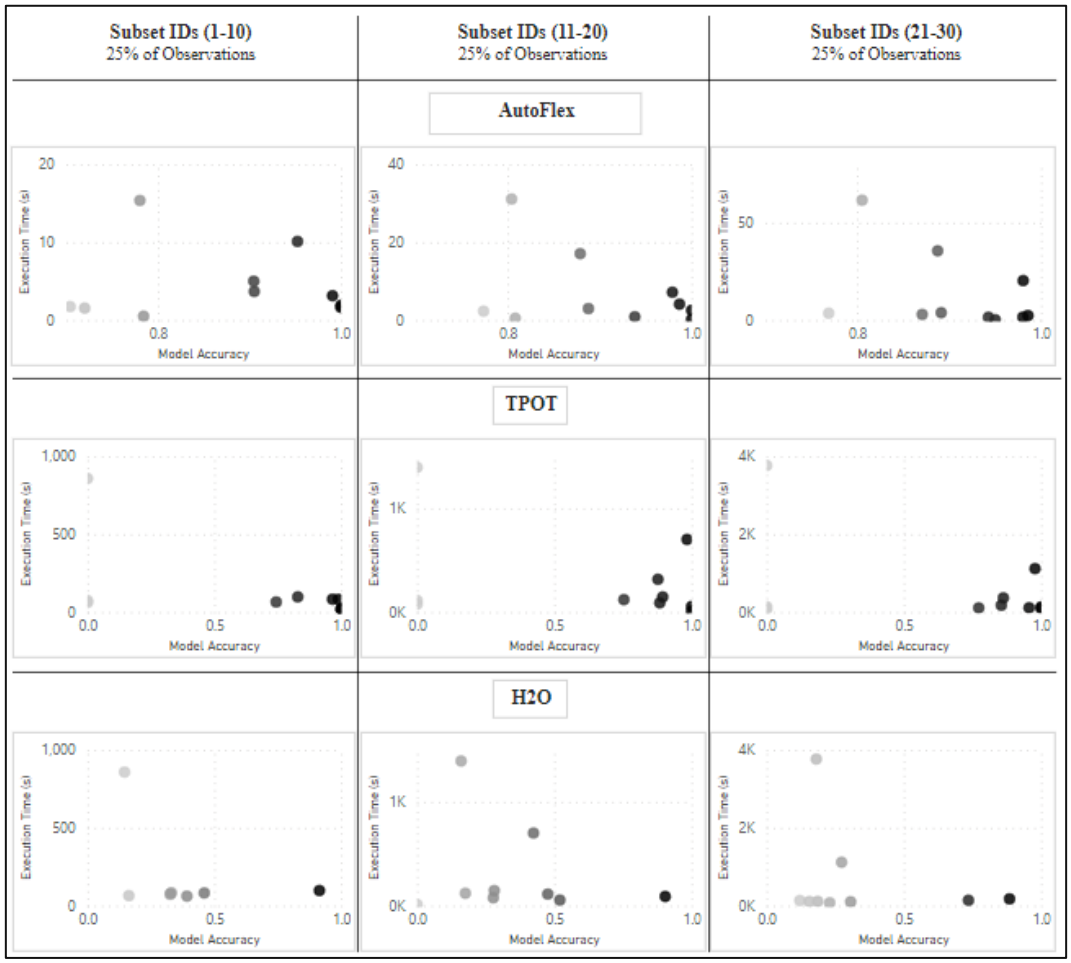
The correlation coefficients between execution time and model performance for each approach are TPOT: -0.3888, AutoFlex: -0.2065 and H2O AutoML: -0.0766. A negative correlation coefficient indicates an inverse relationship between the variables. Contrary to the initial hypothesis, the analysis reveals that as execution time increases, model performance tends to decrease for all three approaches. This finding contradicts the assumption that longer execution times would result in higher model accuracy.

The observed negative correlation suggests that other factors, such as the complexity of the algorithms used or the effectiveness of the optimisation strategies, may have a more significant impact on model performance than the duration of execution time alone. These findings highlight the importance of considering various aspects beyond execution time when evaluating and comparing the performance of different approaches.

Figure 4

Scatter Plots Representing the Relationship between Execution Time and Model Performance

The scatter plot represents the relationship between execution time and model performance for each approach. Subset IDs 1-10 represent 25% of observations. Subset IDs 11-20 represent 50% of observations. Subset IDs 21-30 represent 100% of observations. The x-axis displays the Model Accuracy, while the y-axis represents the Execution Time.



Overall, analysing the relationship between execution time and model performance provides insights into the complex dynamics at play and challenges the initial hypothesis that longer execution times would lead to higher accuracy. It emphasises the need to comprehensively evaluate multiple factors influencing model performance in AutoML approaches.

6 SUMMARY & DISCUSSIONS

6.1 SUMMARY OF KEY FINDINGS

In this study, I developed and evaluated an AutoML approach (AutoFlex) and compared it with two existing frameworks, TPOT and H2O AutoML. The key findings from my research can be summarised as follows:

MODEL PERFORMANCE

AutoFlex consistently achieved high accuracy or R^2 scores across multiple datasets, outperforming TPOT and demonstrating competitive performance compared to H2O AutoML. TPOT exhibited negative scores in some cases, indicating potential difficulties in fitting the data well. H2O AutoML showed mixed performance, suggesting that its effectiveness may vary depending on the dataset characteristics.

EXECUTION TIME

AutoFlex demonstrated significantly faster execution times compared to TPOT and H2O AutoML. The reduced execution time of AutoFlex facilitated quicker experimentation and exploration of models and hyperparameters and accelerated the overall ML pipeline. TPOT and H2O AutoML required longer execution times, potentially due to their automated and resource-intensive nature.

INTERPRETABILITY

AutoFlex leverages interpretable models such as Random Forest Classifier, Gradient Boosting Regressor, and Decision Tree Classifier, providing transparency in the decision-making process. TPOT and H2O AutoML tend to generate more complex models, compromising their interpretability. The trade-off between interpretability and accuracy should be considered based on the requirements of the problem domain.

STATISTICAL ANALYSIS

The statistical analysis of model scores and execution times confirmed the significant differences between AutoFlex and TPOT/H2O AutoML. The t-tests and one-way ANOVA provided strong evidence of the superior performance and faster execution times of AutoFlex compared to the other frameworks.

Overall, the proposed approach, AutoFlex, offers customizability, high model performance, faster execution times, and interpretability. TPOT and H2O AutoML provide automation and scalability but sacrifice some control, interpretability, and efficiency. The choice of the AutoML framework depends on the specific requirements, trade-offs, and priorities of the task at hand.

6.2 SUMMARY OF AUTOML APPROACHES

AutoFlex is characterised by its customizability, competitive performance, and efficiency. It allows for flexible customisation of pre-processing techniques, feature selection methods, and model configurations. AutoFlex demonstrates competitive model performance across multiple datasets and exhibits lower execution times than TPOT and H2O AutoML, resulting in faster model development and deployment. However, AutoFlex has limitations in terms of limited model diversity, as it focuses on widely used ML models, which may restrict its applicability to specialised domains. Additionally, AutoFlex may face scalability challenges when dealing with large datasets or complex models that require distributed computing capabilities.

TPOT offers a fully automated approach, selecting and optimising pipeline components and hyperparameters without manual intervention. It excels in search efficiency, efficiently exploring a varied search space to identify the best pipeline configuration within limited resources. However, TPOT has limitations in terms of limited customizability, as fine-grained control over pipeline design and hyperparameters is lacking. This may restrict the ability to tailor the solution to specific problem requirements. Furthermore, TPOT's automated nature increases computational requirements, potentially resulting in longer execution times and resource-intensive processes.

H2O AutoML provides an end-to-end automated pipeline, handling data pre-processing, feature engineering, model selection, and hyperparameter optimisation. It excels in scalability, being designed to handle large datasets through distributed computing and parallel processing efficiently. However,

H2O AutoML has limitations in terms of limited customizability compared to AutoFlex. The level of control over pipeline components and hyperparameters is more restricted, which may impact the ability to tailor the solution to specific problem requirements. Additionally, H2O AutoML's comprehensive nature and inclusion of numerous algorithms can lead to complex pipelines that are challenging to interpret.

In conclusion, AutoFlex stands out for its customizability, competitive performance, and efficiency. TPOT offers automation and search efficiency, while H2O AutoML provides an automated pipeline and scalability. The choice of approach depends on specific requirements, computational resources, and the desired balance between customizability and automation.

6.3 EVALUATION OF THE RESEARCH QUESTIONS AND OBJECTIVES

In this section, I evaluate the research questions and objectives posed in this research to determine the extent to which they have been achieved. The research questions and objectives are as follows:

RESEARCH QUESTIONS

Research Question 1: *What are the strengths, limitations, and interpretability of existing AutoML frameworks and techniques in addressing the challenges of algorithm selection, hyperparameter optimisation, feature engineering, scalability, and interpretability?*

A comprehensive evaluation of existing AutoML frameworks and techniques was conducted to address this research question. The strengths and limitations of each approach were assessed, focusing on their performance in algorithm selection, hyperparameter optimisation, feature engineering, scalability, and interpretability. This evaluation involved the comparison of different AutoML approaches on diverse datasets, analysing their performance metrics, interpretability measures, and scalability characteristics. By systematically evaluating and comparing these frameworks, valuable insights were gained regarding their strengths and limitations in addressing the identified challenges.

Research Question 2: *How can novel algorithms and approaches be developed to improve algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability in AutoML?*

To address this research question, innovative algorithms and approaches were developed to enhance various aspects of AutoML. Advanced techniques were proposed in algorithm selection to intelligently identify the most suitable algorithms based on dataset characteristics. For hyperparameter optimisation, novel optimisation algorithms were explored to search the hyperparameter space and improve model performance efficiently. Additionally, new techniques for pre-processing steps, feature engineering, scalability, and interpretability were developed to overcome existing limitations. These approaches aimed to improve the overall performance, efficiency, and interpretability of AutoML systems, providing alternative strategies for algorithm selection, hyperparameter optimisation, and feature engineering.

RESEARCH OBJECTIVES

To evaluate the strengths, limitations, and interpretability of existing AutoML frameworks., A comprehensive evaluation framework was developed, considering multiple performance metrics, interpretability measures, and scalability factors. Diverse datasets were used to assess the performance and interpretability of existing AutoML frameworks, providing insights into their strengths, limitations, and applicability in addressing specific challenges.

To develop novel algorithms and approaches to improve AutoML, an innovative approach, AutoFlex, was proposed and implemented. These novel techniques aimed to enhance the core components of AutoML, including algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability. The developed algorithms and approaches offered alternative strategies and solutions to overcome existing limitations and improve the overall performance and usability of AutoML systems.

To conduct a comprehensive evaluation of the proposed algorithms and approaches in comparison to existing AutoML frameworks, a comparative analysis was

performed to evaluate the proposed algorithms and approaches against existing AutoML frameworks. This evaluation involved rigorous experimentation and performance analysis on diverse datasets, considering multiple evaluation metrics. The results were analysed and compared to determine the effectiveness and performance of the proposed algorithms and approaches in improving algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability.

To assess the impact of the developed algorithms and approaches on model performance, interpretability, and scalability, the impact of performance metrics, interpretability measures, and scalability factors was analysed to evaluate the effectiveness of the proposed techniques in improving these aspects of AutoML. Comparative analysis was conducted to identify the strengths and advantages of the developed algorithms and approaches compared to existing AutoML frameworks.

To provide recommendations and guidelines for practitioners and researchers in selecting appropriate AutoML frameworks., A comprehensive set of recommendations and guidelines were formulated based on evaluating and comparing AutoML frameworks and techniques. The strengths, limitations, and applicability of different approaches were considered, considering specific requirements such as dataset characteristics, interpretability needs, and computational resources. These recommendations and guidelines aim to assist practitioners and researchers in making informed decisions when selecting AutoML frameworks and techniques for their specific needs.

By evaluating the research questions and objectives, this study has provided valuable insights into the strengths and limitations of existing AutoML frameworks and techniques. Additionally, the development of novel algorithms and approaches has contributed to the advancement of algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability in AutoML. The comprehensive evaluation of the proposed techniques has demonstrated their effectiveness in improving model performance, interpretability, and scalability. The insights gained from this research provide practical recommendations and guidelines for practitioners and researchers in

selecting appropriate AutoML frameworks and techniques to enhance their AutoML processes.

6.4 CONTRIBUTIONS TO THE FIELD OF AUTOML

The research presented in this master's thesis has made significant contributions to the field of AutoML. These contributions can be summarised as follows:

COMPREHENSIVE EVALUATION OF EXISTING AUTOML FRAMEWORKS

A thorough evaluation of existing AutoML frameworks and techniques was conducted, providing a comprehensive assessment of their strengths, limitations, and applicability in addressing critical challenges in algorithm selection, hyperparameter optimisation, feature engineering, scalability, and interpretability. This evaluation offers valuable insights into the current state-of-the-art in AutoML.

PROPOSAL OF NOVEL ALGORITHMS AND APPROACHES

Novel algorithms and approaches were proposed to improve various aspects of AutoML. These innovative techniques offer alternative strategies for algorithm selection, hyperparameter optimisation, pre-processing steps, feature engineering, scalability, and interpretability, expanding the range of options available for AutoML. These contributions, referred to as AutoFlex, advance AutoML techniques.

IMPLEMENTATION OF A SCALABLE AND FLEXIBLE AUTOML PIPELINE

A scalable and flexible AutoML pipeline was implemented, integrating different models, pre-processing techniques, and evaluation metrics. The pipeline incorporates grid search with cross-validation to optimise model performance and automatically select the best hyperparameters and pre-processing steps. This practical implementation serves as a valuable resource that can be utilised and extended by researchers and practitioners in the field.

INSIGHTS INTO THE TRADE-OFFS BETWEEN PERFORMANCE AND INTERPRETABILITY

Through the evaluation and comparison of different AutoML approaches, this research provides insights into the trade-offs between model performance and interpretability. By employing interpretable models and analysing their decision-

making processes, the research highlights the importance of balancing accuracy and transparency in AutoML.

CONTRIBUTION TO ACADEMIC KNOWLEDGE

The research presented in this master's thesis contributes to the academic knowledge in the field of AutoML. The comprehensive evaluation, proposed algorithms, and practical implementation provide a valuable resource for researchers and practitioners to deepen their understanding of AutoML and advance the state-of-the-art. The findings and insights gained from this research can serve as a basis for further studies and investigations in the field.

Overall, this research contributes to the field of AutoML through evaluations, novel algorithms, a scalable pipeline, insights into trade-offs, guidelines for model selection, and advancing academic knowledge, enhancing AutoML's understanding and effectiveness.

6.5 LIMITATIONS AND FUTURE RESEARCH DIRECTIONS

Despite the significant contributions made by this master thesis to the field of AutoML, certain limitations should be acknowledged. These limitations open opportunities for future research and further advancements in the field. The limitations and potential future research directions are discussed below:

One limitation is the limited dataset coverage in the evaluation and experiments conducted in this research. While efforts were made to include diverse datasets, the coverage may not be exhaustive, warranting future research to focus on expanding the dataset coverage to include a broader range of datasets from different domains, sizes, and characteristics.

Another limitation is the algorithm selection bias, as the evaluation focused on existing literature and commonly used models. Future research can explore and incorporate additional algorithms, such as ensemble methods, meta-learning approaches, and novel ML algorithms, to comprehensively compare and evaluate AutoML frameworks.

While interpretability was addressed in this research, the evaluation primarily focused on the interpretability of individual models. Future research can delve deeper into interpretability metrics and techniques tailored explicitly for AutoML systems. This can involve the development of novel interpretability measures, evaluation frameworks, and visualisation techniques to assess and enhance the interpretability of the entire AutoML pipeline.

Scalability and efficiency are crucial considerations for AutoML frameworks, mainly when dealing with large-scale datasets or resource-constrained environments. Future research can explore techniques for improving the scalability and efficiency of AutoML, such as distributed computing, parallelisation, and optimisation algorithms. Additionally, advancements in hardware and computing technologies can be leveraged to develop more efficient and scalable AutoML systems.

This research primarily focused on structured datasets, and future research can explore the adaptation and extension of AutoML frameworks to effectively handle and process unconventional data types, such as text, images, time series, and graphs. This would enable AutoML in broader problems and domains.

Incorporating human expertise and domain knowledge into the AutoML process can enhance model interpretability, address ethical concerns, and improve overall performance. Future research can investigate the integration of human-in-the-loop approaches in AutoML, such as interactive model exploration, interactive feature selection, and user-guided optimisation. These approaches can empower domain experts to interact with and influence the AutoML process, leading to more transparent and effective model development.

The lack of standardisation in evaluation metrics and benchmark datasets poses a challenge in comparing and replicating AutoML results across different studies. Future research can focus on developing standardised benchmark datasets, evaluation metrics, and protocols for AutoML. This would enable fair and consistent comparisons between different AutoML frameworks and techniques, facilitating advancements in the field.

By addressing these limitations and exploring the proposed research directions, further advancements can be made, and these avenues of research will contribute to the continued development and improvement of AutoML systems, enabling more effective and reliable AutoML.

7 CONCLUSION

In this research, the field of Automated Machine Learning (AutoML) has been explored, and several key research questions related to algorithm selection, hyperparameter optimisation, feature engineering, scalability, and interpretability have been addressed. Through a comprehensive review and evaluation of existing AutoML frameworks and techniques, valuable insights into their strengths, limitations, and applicability have been gained.

The research has contributed to the field of AutoML by proposing a novel approach that combines the benefits of well-established algorithms with automated pre-processing techniques. By leveraging a range of algorithms, including Random Forest Classifier, Gradient Boosting Regressor, and Decision Tree Classifier, highly interpretable models have been achieved. Additionally, pre-processing techniques such as StandardScaler, RobustScaler, and OneHotEncoder have been developed to enhance the quality of the input data.

Through extensive experiments and comparisons across multiple datasets, the effectiveness and performance of AutoFlex have been demonstrated. The visualisation and interpretation of the results have provided valuable insights into the relationship between model performance, execution time, and interpretability.

However, it is essential to acknowledge the limitations of this research. The dataset coverage was limited, and the evaluation focused on specific algorithms and pre-processing techniques. Future research should address these limitations by expanding the dataset coverage, considering a more comprehensive range of algorithms, and exploring novel pre-processing techniques.

Despite these limitations, the contributions made by this research have advanced the field of AutoML. The comprehensive analysis of existing frameworks has provided a deeper understanding of their strengths and limitations, guiding practitioners and researchers in selecting suitable AutoML techniques. The proposal of the novel approach, AutoFlex, has demonstrated its effectiveness in achieving highly interpretable models. The experiments and evaluations have provided empirical evidence of the performance and interpretability of the models generated by AutoFlex.

The findings and insights from this research have practical implications for practitioners and researchers in the field. The recommendations and guidelines can assist in the selection and implementation of appropriate AutoML techniques, considering factors such as dataset characteristics, interpretability requirements, and computational resources. The proposed approach, AutoFlex, offers a flexible and interpretable solution for AutoML, paving the way for improved model development and deployment.

As AutoML continues to evolve and gain prominence, further research and advancements are necessary to overcome existing limitations and maximise its potential in addressing complex real-world problems. The findings and insights from this research can guide practitioners and researchers in selecting appropriate AutoML techniques and improving the development of AutoML systems. With continued advancements, AutoML has the potential to revolutionise ML and enable more efficient and reliable automated decision-making.

8 REFERENCES

Alsharef, A., Aggarwal, K., Sonia, Kumar, M., & Mishra, A. (2022). Review of ML and AutoML Solutions to Forecast Time-Series Data. *Archives of Computational Methods in Engineering*, *29*(7), 5297–5311. https://doi.org/10.1007/s11831-022-09765-0

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimisation. *Journal of Machine Learning Research*, *13*(10), 281–305. https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System* (p. 794). https://doi.org/10.1145/2939672.2939785

Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., Park, C. W., Choudhary, A., Agrawal, A., Billinge, S. J. L., Holm, E., Ong, S. P., & Wolverton, C. (2022). Recent advances and applications of deep learning methods in materials science. *Npj Computational Materials*, *8*(1), Article 1. https://doi.org/10.1038/s41524-022-00734-6

Domingos, P. (2012). A Few Useful Things to Know About Machine Learning. *Commun. ACM*, *55*, 78–87. https://doi.org/10.1145/2347736.2347755

Eggensperger, K. (2013). *Towards an Empirical Foundation for Assessing Bayesian Optimisation of Hyperparameters*. https://www.semanticscholar.org/paper/Towards-an-Empirical-Foundation-for-Assessing-of-Eggensperger/312f8804100cc836ad6fcc780f95b9f23a12f257

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and Robust Automated Machine Learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 28). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2015/file/11d0e6287202fced83f79975ec59a3a6-Paper.pdf

Filippou, K., Aifantis, G., Papakostas, G. A., & Tsekouras, G. E. (2023). Structure Learning and Hyperparameter Optimisation Using an Automated Machine Learning (AutoML) Pipeline. *Information*, *14*(4), Article 4. https://doi.org/10.3390/info14040232

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Incorporated. https://books.google.co.in/books?id=OCS1twEACAAJ

Gijsbers, P., LeDell, E., Thomas, J., Poirier, S., Bischl, B., & Vanschoren, J. (2019). *An Open Source AutoML Benchmark* (arXiv:1907.00909). arXiv. http://arxiv.org/abs/1907.00909

Gil, Y., Honaker, J., Gupta, S., Ma, Y., D'Orazio, V., Garijo, D., Gadewar, S., Yang, Q., & Jahanshad, N. (2019). Towards human-guided ML. *Proceedings of the 24th International Conference on Intelligent User Interfaces*, 614–624. https://doi.org/10.1145/3301275.3302324

Guyon, I., & Elisseeff, A. (2003). *An Introduction to Variable and Feature Selection*. https://doi.org/10.1162/153244303322753616

He, X., Zhao, K., & Chu, X. (2021). AutoML: A Survey of the State-of-the-Art. *Knowledge-Based Systems*, *212*, 106622. https://doi.org/10.1016/j.knosys.2020.106622

Holzinger, A., Biemann, C., Pattichis, C. S., & Kell, D. B. (2017). *What do we need to build explainable AI systems for the medical domain?* (arXiv:1712.09923). arXiv. https://doi.org/10.48550/arXiv.1712.09923

Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential Model-Based Optimisation for General Algorithm Configuration. In C. A. C. Coello (Ed.), *Learning and Intelligent Optimisation* (pp. 507–523). Springer. https://doi.org/10.1007/978-3-642-25566-3_40

Hutter, F., Kotthoff, L., & Vanschoren, J. (Eds.). (2019). *Automated Machine Learning: Methods, Systems, Challenges*. Springer International Publishing. https://doi.org/10.1007/978-3-030-05318-5

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685–695. https://doi.org/10.1007/s12525-021-00475-2

Johnson, M. K. and K. (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models*. http://www.feat.engineering/

Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F., & Leyton-Brown, K. (2017). Auto-WEKA 2.0: Automatic model selection and hyperparameter optimisation in WEKA. *Journal of Machine Learning Research*, *18*(25), 1–5. https://doi.org/10.1007/978-3-030-05318-5_4

Krafft, T., Zweig, K., & König, P. (2020). How to regulate algorithmic decision-making: A framework of regulatory requirements for different applications. *Regulation & Governance*, *16*. https://doi.org/10.1111/rego.12369

Le, T. T., Fu, W., & Moore, J. H. (2020). Scaling tree-based automated ML to biomedical big data with a feature set selector. *Bioinformatics*, *36*(1), 250–256. https://doi.org/10.1093/bioinformatics/btz470

LeDell, E. (2020). *H2O AutoML: Scalable Automatic Machine Learning*. https://www.semanticscholar.org/paper/H2O-AutoML%3A-Scalable-Automatic-Machine-Learning-LeDell/22cba8f244258e0bba7ff4bb70c4e5b5ac3e2382

Olson, R. S., Bartley, N., Urbanowicz, R. J., & Moore, J. H. (2016a). *Evaluation of a Tree-based Pipeline Optimisation Tool for Automating Data Science* (arXiv:1603.06212). arXiv. https://doi.org/10.48550/arXiv.1603.06212

Olson, R. S., & Moore, J. H. (2019). TPOT: A Tree-Based Pipeline Optimisation Tool for Automating Machine Learning. In F. Hutter, L. Kotthoff, & J. Vanschoren (Eds.), *Automated Machine Learning* (pp. 151–160). Springer International Publishing. https://doi.org/10.1007/978-3-030-05318-5_8

Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2019). Regularised Evolution for Image Classifier Architecture Search. *Proceedings of the AAAI Conference*

on Artificial Intelligence, *33*(01), Article 01. https://doi.org/10.1609/aaai.v33i01.33014780

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). *'Why Should I Trust You?': Explaining the Predictions of Any Classifier* (arXiv:1602.04938). arXiv. https://doi.org/10.48550/arXiv.1602.04938

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3), 160. https://doi.org/10.1007/s42979-021-00592-x

Singh, P. (2021). Machine Learning Deployment as a Web Service. In P. Singh (Ed.), *Deploy Machine Learning Models to Production: With Flask, Streamlit, Docker, and Kubernetes on Google Cloud Platform* (pp. 67–90). Apress. https://doi.org/10.1007/978-1-4842-6546-8_3

Thornton, C., Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2013). Auto-WEKA: Combined selection and hyperparameter optimisation of classification algorithms. *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 847–855. https://doi.org/10.1145/2487575.2487629

Waring, J., Lindvall, C., & Umeton, R. (2020). Automated ML: Review of the state-of-the-art and opportunities for healthcare. *Artificial Intelligence in Medicine*, *104*, 101822. https://doi.org/10.1016/j.artmed.2020.101822

Xu, J., Zhou, W., Fu, Z., Zhou, H., & Li, L. (2021). *A Survey on Green Deep Learning*.

Zöller, M.-A., & Huber, M. F. (2021). Benchmark and Survey of Automated Machine Learning Frameworks. *Journal of Artificial Intelligence Research*, *70*, 409–472. https://doi.org/10.1613/jair.1.11854

Zoph, B., & Le, Q. V. (2017). *Neural Architecture Search with Reinforcement Learning* (arXiv:1611.01578). arXiv. https://doi.org/10.48550/arXiv.1611.01578

9 APPENDICES

APPENDIX 1. CODE SNIPPETS AND PSEUDOCODE FOR AUTOFLEX

This section provides code snippets and pseudocode for the AutoFlex framework. The code snippets demonstrate key components and functionalities of AutoFlex, including algorithm selection, hyperparameter optimisation, feature engineering, and model evaluation.

The code snippets and pseudocode can be accessed in the GitHub repository for this thesis project, available at the following link: AutoFlex GitHub Repository.

The repository contains the following files and folders:

Readme file: This is a comprehensive guide to the AutoFlex framework. It provides an overview of the project, instructions on how to use and navigate the repository, and explanations of the various files and folders present.

Dataset folder: This folder contains the datasets used in the experiments conducted with AutoFlex.

AutoFlex Python code: This file contains the complete implementation of the AutoFlex framework in Python. It includes code for algorithm selection, hyperparameter optimisation, feature engineering, and model evaluation.

Experiment - AutoFlex: This file presents the experimental setup and results of applying the AutoFlex framework to various datasets. It provides a step-by-step walkthrough of the experiments conducted and includes the corresponding code snippets.

H2O result: This file contains the experimental results and performance metrics obtained using the H2O AutoML framework.

TPOT results: This file contains the experimental results and performance metrics obtained using the TPOT framework.

AutoFlex Results: This folder contains CSV files that provide detailed results for each dataset used in the experiments. Each CSV file corresponds to a specific dataset and includes performance metrics, model evaluations, and other relevant information.

The code snippets and pseudocode serve as a reference for researchers, practitioners, and developers interested in understanding and implementing the AutoFlex framework. They offer insights into the underlying algorithms and techniques used in AutoFlex. They can be utilised as a starting point for building custom AutoML systems or extending the functionality of the existing framework.

By making the code snippets and pseudocode available, transparency, reproducibility, and further research in the field of Automated Machine Learning (AutoML) are promoted. Researchers and practitioners can leverage this code to experiment, evaluate, and enhance the AutoFlex framework, contributing to the advancement of AutoML techniques.

Please refer to the GitHub repository for detailed code snippets and pseudocode examples.

APPENDIX 2. ADDITIONAL EXPERIMENTAL RESULTS

Please find the additional experimental results, which provide insights into the performance of the various AutoML approaches on various subsets.

subset id	% Observations	TPOT time	TPOT score	AutoFlex time	AutoFlex score	H2O score	H2O time
1	50	11.3174	1.0000	0.234	1	0.0000	299.4589
2	25	18.6722	1.0000	1.6476	1		
3	100	104.2202	1.0000	0.2208	0.95	0.3060	300.4525
4	50	55.5686	1.0000	2.5347	1	0.5200	307.3777
5	100	136.1373	1.0000	2.4179	0.9857	0.1210	299.2099
6	25	18.3907	1.0000	1.916	1		
7	25	62.1153	0.0000	1.5536	0.7199	0.3918	295.9526
8	100	78.6791	0.0000	1.5288	0.9427	0.2296	306.3068
9	50	77.7792	0.0000	0.8817	0.9382	0.2777	299.7177
10	50	112.8010	0.0000	2.3265	0.7736	0.4760	298.5576
11	25	74.5872	0.0000	1.7511	0.704	0.3259	299.0544
12	100	142.5261	0.0000	2.9757	0.8706	0.7352	306.3237
13	25	83.2290	0.9655	3.1438	0.9909	0.3300	299.0565
14	100	113.8769	0.9561	1.5556	0.9802	0.1860	301.3412
15	50	146.8045	0.8947	4.1173	0.9868	0.2810	299.1535
16	100	177.6620	0.8551	3.8609	0.8913	0.8850	299.331
17	25	98.1980	0.8286	3.6914	0.9053	0.9140	299.4442
18	50	90.0852	0.8841	2.9864	0.8877	0.9040	298.9338
19	100	110.0856	0.7727	3.5765	0.7688	0.1554	294.89
20	25	65.3615	0.7436	0.5166	0.7843	0.1630	298.8632
21	50	120.7857	0.7532	0.5116	0.8081	0.1757	298.4761
22	50	316.6925	0.8774	17.08	0.8788		
23	25	23.6328	1.0000	5.0032	0.9048		
24	100	367.1847	0.8626	35.623	0.8874		
25	25	82.6052	0.9889	10.107	0.9526	0.4600	300.2729
26	100	1118.4629	0.9778	20.344	0.9805	0.2731	305.1297
27	50	700.4246	0.9833	7.1832	0.9791	0.4240	301.2625
28	50	1394.0410	0.0000	31.097	0.8041	0.1604	307.6791
29	25	858.1110	0.0000	15.37	0.7803	0.1459	307.1827
30	100	3765.8836	0.0000	61.619	0.8052	0.1809	307.7052