Artificial Intelligence Review (2026) 59:165
https://doi.org/10.1007/s10462-026-11573-y

Interpretable Automated Feature Engineering: A Comprehensive Review with a Focus on Dynamic and Stationary Environments

Dilshan Sonnadara[1] · Thanh Tung Khuat[1] · Katarzyna Musial-Gabrys[1] · Bogdan Gabrys[1]

Received: 10 October 2025 / Accepted: 15 April 2026 / Published online: 24 April 2026
© The Author(s) 2026

Abstract
This paper presents a comprehensive review of Automated Feature Engineering (AFE) methods, with a focus on their application in static data environments and explores recent advancements in dynamic Feature Engineering (FE) for evolving data contexts. AFE is essential in modern Machine Learning (ML) pipelines, as it addresses the drawbacks of traditional FE, which often relies heavily on domain expertise and is labor intensive. By automating this process, AFE has the potential to enhance the efficiency, performance, and interpretability of ML workflows. The paper analyzes the methods, strengths, and limitations of different AFE methods. The review also identifies critical research gaps and proposes directions for future advancements. This work serves as a foundational resource for researchers aiming to optimize ML performance through novel AFE strategies.

Keywords Automated Feature Engineering · Interpretability · Adaptivity · Feature Generation · Feature Selection · Machine Learning

Dilshan Sonnadara conducted the research and manuscript preparation. Thanh Tung Khuat, Katarzyna Musial-Gabrys, and Bogdan Gabrys provided supervision, guidance, and critical revisions.

✉ Dilshan Sonnadara
Dilshan.S.Sonnadara@student.uts.edu.au

Thanh Tung Khuat
thanhtung.khuat@uts.edu.au

Katarzyna Musial-Gabrys
katarzyna.musial-gabrys@uts.edu.au

Bogdan Gabrys
bogdan.gabrys@uts.edu.au

[1] Complex Adaptive Systems Laboratory, The Data Science Institute, University of Technology Sydney, Ultimo, NSW 2007, Australia

1 Introduction

Machine Learning (ML) has become an integral part of the modern society, driving innovations across diverse sectors such as healthcare (Habehh and Gohel 2021) and finance (Nazareth and Ramana Reddy 2023). At its core, ML uses algorithms to learn from data, and make informed decisions based on what it has learned. These datasets are typically extensive, potentially including millions of distinct data points (Nichols et al. 2018). As such, the effectiveness of ML models is heavily dependent on the quality of this input data (Pugliese et al. 2021). Feature Engineering (FE) is a critical step in the ML workflow. It involves transforming raw data into a format that is better suited for models, which is essential for improving model performance (Wang et al. 2022b). As ML applications grow in complexity, manually building models has become a bottleneck to scalability. This has led to an increased interest in Automated ML (AutoML), which aims to automate all aspects of the ML pipeline (Kedziora et al. 2020; Scriven et al. 2023; Khuat et al. 2023). In recent years, the evolution of AutoML has been significantly accelerated by advancements in Automated Feature Engineering (AFE). AFE automates the generation and selection of features (Wang et al. 2022c), enhancing the performance and interpretability of ML workflows (Gosiewska et al. 2021). This paper presents a comprehensive review of existing AFE methods, with special attention being paid to those models which generate interpretable features.

Several comprehensive reviews that focus on the methods and tools of FE have been published. For instance, there are many review papers which discuss the topic of Feature Selection (FS) (Miao and Niu 2016; Kumar 2014; Asir et al. 2016). Additionally, there are also papers that discuss the broader aspects of FE itself. For instance, Rawat and Khemchandani (2017) discusses FE methods such as feature transformation (scaling, discretizations, normalization, log odds), Feature Generation (FG) (through binarization), and selection techniques (correlation, ANOVA, chi square). While there exists many papers which review the topic of FE, there is a scarcity of papers which do so for AFE. This study represents the first comprehensive review that examines AFE techniques for static data environments alongside recent developments in dynamic FE for streaming data.

While existing literature, such as Prado and Digiampietri (2020) and Mumuni and Mumuni (2024), reviews AFE techniques, their coverage is restricted. Prado and Digiampietri (2020) focuses on only three AFE models, and is mainly focused on automated FS. This paper on the other hand, reviews over 25 static AFE models, and many more dynamic FE models. Furthermore, Mumuni and Mumuni (2024) mainly concentrates on automated preprocessing tasks such as data cleaning and encoding, missing data management, and data augmentation. Neither study explores the interpretability of AFE models or the challenges of dynamic data environments. This paper is unique, as it delves into methods that enhance both interpretability and adaptability, offering a comprehensive critique of a much larger number of AFE models, in comparison to the other studies.

The paper is structured as follows. Section 2 introduces ML fundamentals, emphasizing the challenges faced within traditional ML workflows and the role of AutoML in overcoming these challenges. This section also focuses on FE, discussing traditional methods and transitioning to AFE. Section 3 explores AFE models designed for static data environments, assessing their methods and limitations, and providing the foundational knowledge needed to move into the next parts. Section 4 covers adaptive FE algorithms that cater for dynamic data environments, focusing on their necessity to handle evolving data features

and distributions. Through these sections, the paper outlines the progression from basic ML concepts to sophisticated AFE applications. It also emphasizes the importance of interpretability throughout the entire review, but focuses specifically on this aspect in Sect. 5. To conclude, Sect. 6 provides an overview of the current gaps which exist in the literature and offers insights into future directions.

This paper aims to serve as a foundational resource for researchers looking to optimize ML performance through AFE strategies. Through a deeper understanding of AFE's potential and limitations, researchers can better use its capabilities in their work. This research can also serve as an important resource for taking the existing work a step further by addressing the gaps which are currently present.

2 Introduction to Machine Learning and feature engineering

This section explores the growing relevance of ML and the important role of FE within its pipeline. It begins by outlining the limitations of traditional ML workflows. The section then introduces AutoML as a solution. Within this context, it underscores the emergence of AFE as a response to the inefficiencies of manual FE. This section aims to provide a comprehensive grounding for subsequent discussions on AFE methods.

2.1 From manual ML pipelines to automated learning systems

ML is the scientific exploration of algorithms and statistical models that enable computer systems to carry out specific tasks without being directly programmed to do so (Mahesh 2020). Despite its widespread adoption, effective ML development remains heavily dependent on skilled practitioners (Scriven et al. 2023). The individuals responsible for building an ML pipeline are expected to have a solid background in statistics and programming to handle numerous tasks (Verma et al. 2021) (as illustrated by the ML development workflow in Fig. 1). For non-expert users, implementing and tuning ML models remains a non-trivial task. AutoML has thus risen as a solution to mitigate these problems.

AutoML refers to the process of automating the complete ML pipeline or even a more complex multi-component, multilevel prediction systems (Salvador et al. 2016b, 2019). It aims to improve accessibility and efficiency, while also reducing potential sources of human error (Salehin et al. 2024; Scriven et al. 2023; Kedziora et al. 2020; Khuat et al. 2023). However, in order for AutoML to reach its full potential, it is also important to focus on the most time intensive step of the pipeline: FE (Khurana et al. 2016).

2.2 Feature engineering

FE can be defined as the process of preparing data to enhance the quality of the features used, aiming to optimize ML models (Wang et al. 2022b; An et al. 2024). It focuses on developing a feature space that accurately represents the core aspects of the problem, thereby allowing models to make more effective predictions (Wang et al. 2022b). This process involves two key approaches: FG and FS (Eldeeb et al. 2022). In the broader literature however, the terms Feature Construction (FC) and Feature Extraction (FExt) are frequently used as well, with FC being used in a similar context to FG. Specifically, Wang et al. (2022b) defines FC as the

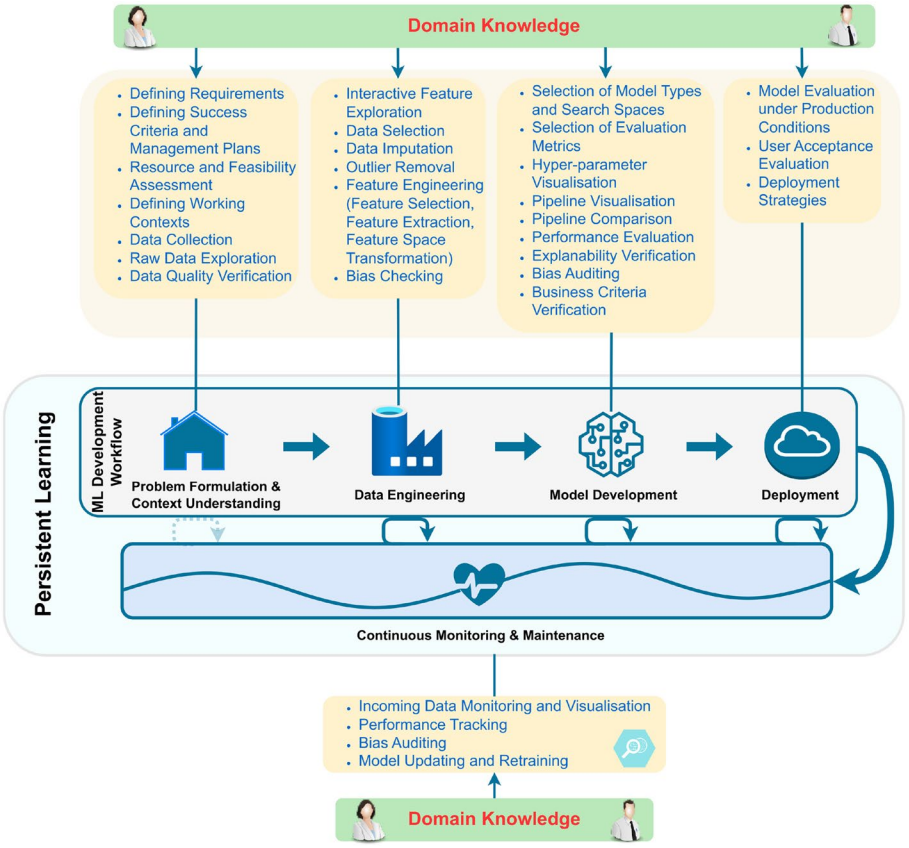

Fig. 1 ML development workflow. Adopted from Khuat et al. (2023)

creation of new features by transforming, aggregating, or decomposing original data using domain knowledge and empirical understanding, while FExt involves projecting the original feature space into a new, often lower-dimensional space through mathematical transformations that yield mutually independent features (essentially transforming the feature space). Figure 2 represents the different components of FE. That is, it shows the proportion of papers which have the terms of FG, FC and FExt in the papers published in the Scopus database as well as the count and percentage of papers which have the term of FS.

2.2.1 Feature selection

FS aims to identify the best subset from a given set of features (Bonidia et al. 2022). The challenge of FS is comparable to finding the optimal subset within a set, a problem that has been proven to be NP hard (Roshanski et al. 2023). Given the impracticality of evaluating every possible subset individually, there is a need for more intelligent methods to find the optimal subset (Zöller and Huber 2021; Khurana et al. 2016). This is what most modern FS algorithms aim to do. FS is crucial, as irrelevant or redundant features can degrade predictive performance (Zöller and Huber 2021; Gada et al. 2021), increase computational cost,

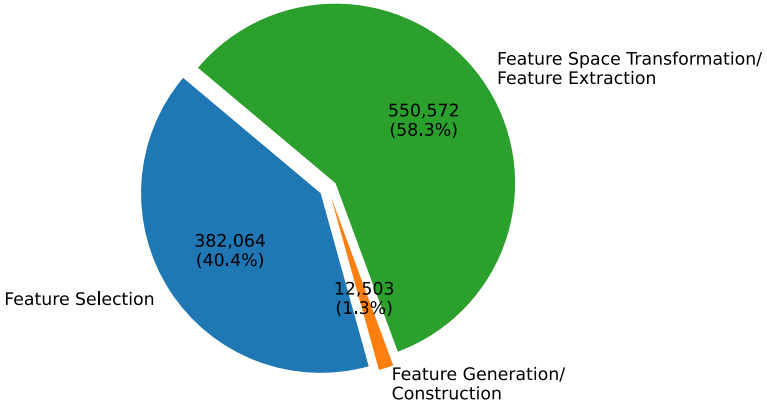

Fig. 2 Pie chart showing the distribution of publications related to three broad categories of FE techniques between 1962 and 2025, based on a search of scientific literature in the Scopus database. The numbers within each pie slice represent the absolute number of publications and their relative percentage of the total. These categories include the number of publications that contain the exact phrase of 'Feature Selection', 'Feature Generation' or 'Feature Construction', and 'Feature space transformation' or 'Feature extraction'

and hinder interpretability (Zöller and Huber 2021), motivating the need for efficient and robust FS strategies (Zöller and Huber 2021; Gada et al. 2021; Real et al. 2020; Mamdouh Farghaly and Abd El-Hafeez 2023; Salazar et al. 2021).

FS can be further categorized as embedded methods, wrapper methods and filter methods.

- **Filter methods**: Uses feature ranking as the evaluation metric for FS, prioritizing features based on their scores from statistical tests that assess their correlation with the target class (Pudjihartono et al. 2022). However, these methods often ignore model-specific effects.
- **Wrapper methods**: Iteratively assesses subsets of features to determine their effect on model performance with a particular dataset, repeating the evaluation until optimal results are achieved. Although they tend to outperform other algorithms in terms of accuracy, this advantage comes at the expense of increased computational demands (Jain and Xu 2023).
- **Embedded methods**: FS is incorporated directly within the ML algorithm itself, allowing the model to select relevant features as part of its learning process (Pudjihartono et al. 2022). However, the selected features are tightly coupled to the chosen model and may not generalize across algorithms.

Offering a comprehensive review of all existing FS methods along with its definitions, advantages and disadvantages is beyond the scope of this paper. Readers interested in a more exhaustive exploration of this subject are encouraged to refer to Theng and Bhoyar (2023), Kamalov et al. (2025), Arun Kumar et al. (2022), and Remeseiro and Bolon-Canedo (2019) for further details.

2.2.2 Feature Generation

FG involves the creation of new features by applying an operation or set of operations to one or more of the original input variables (Eldeeb et al. 2022). In general, this often happens through the use of either unary, binary, or higher order operators (Zöller and Huber 2021). The commonly found operators in performing FG is given in Table 1 (Katz et al. 2016; Eldeeb et al. 2022; Khurana et al. 2016).

It is also important to note that while these are the most common operators found in performing FG, there can be other operators which are specific for certain systems and domains. An example would be the Support Vector Classifier (SVC) based binary operators which is also used at times (Roshanski et al. 2023; Mahajan 2020).

2.2.3 Feature space transformation

Feature space transformation involves transforming the original set of features into a new feature representation, typically without focusing on individual features but rather on the overall structure of the feature space. Unlike FS or FG, which operate at the level of selecting or creating individual features, feature space transformation generates an entirely new feature space through a global transformation.

Often, these transformations aim to capture the essential characteristics of the original data. In the instance that it ends up creating a new space which is of a lower dimension than the original space, it is referred to as dimensionality reduction. Dimensionality reduction

Table 1 Summary of operators used for FG

Type of operator	Operator	Examples	References
Unary operator	Discretization	Breaks a continuous feature into categories	Katz et al. (2016) and Gosiewska et al. (2021)
Unary operator	Normalization	z-normalization, min–max normalization	Katz et al. (2016) and Khurana et al. (2016)
Unary operator	Mathematical transformations	Logarithms, square root, reciprocal, squaring, cubing, sine, cosine, exponential, absolute value, rounding, negation	Horn et al. (2020), Eldeeb et al. (2022), Khurana et al. (2016), Liu et al. (2024a), Roshanski et al. (2023), Kaul et al. (2017) and Eldeeb and Elshawi (2024)
Unary operator	Date & Time extraction	IsWeekend, date, time, day, month, year extraction	Katz et al. (2016) and Khurana et al. (2016)
Binary operator	Arithmetic operations	Addition, subtraction, multiplication, division	Katz et al. (2016), Horn et al. (2020), Eldeeb et al. (2022), Khurana et al. (2016), Liu et al. (2024a), Roshanski et al. (2023), Kaul et al. (2017) and Eldeeb and Elshawi (2024)
Binary operator	Logical operators	Greater than, less than, equality	Roshanski et al. (2023)
Binary operator	Comparison operators	AND, OR, NOT	Roshanski et al. (2023) and Eldeeb and Elshawi (2024)
Higher order operator	Ternary operators	$a\ ?\ b : c$—returns b if a is true, otherwise c	Eldeeb and Elshawi (2024)
Higher order operator	Aggregation functions	MAX, MIN, AVG, MEAN, STDEV, COUNT	Katz et al. (2016), Khurana et al. (2016) and Eldeeb and Elshawi (2024)

techniques seek to represent high dimensional data in a lower dimensional form while preserving as much relevant information as possible.

Well known methods for dimensionality reduction include Principal Component Analysis (PCA), Independent Component Analysis, and t-Distributed Stochastic Neighbor Embedding (Anuragi et al. 2024). These methods captures the essential information from a high dimensional space and maps it to a lower dimensional space by examining the distribution within the reduced space (Thiamchoo and Phukpattaranont 2022).

A comprehensive overview of dimensionality reduction and feature space transformation methods can be found in Velliangiri et al. (2019).

2.3 Automated Feature Engineering

Creating high quality features is a crucial and demanding task that requires data scientists/ML specialists to possess a strong background in ML and statistics, along with substantial domain knowledge and human intuition. This makes FE one of the most labor intensive stages in the development of ML pipelines. It is also a very time consuming step which requires a lot of complicated code to perform well (Anderson et al. 2016; Bonidia et al. 2022). As the demand for ML applications rapidly expands, the number of skilled data scientists struggles to keep up, highlighting the need for automated solutions that can accommodate those without expert knowledge (Khurana et al. 2016).

AFE aims to bring down the burden put on ML experts by streamlining this complex, iterative process, which often involves trial and error and depends heavily on accumulated domain expertise (Eldeeb et al. 2022; Khurana et al. 2016). It aims to automatically create features from the data with the goal of improving the performance of the learning algorithm employed. This is especially crucial in situations where one cannot allocate time and money to develop high quality features (Gosiewska et al. 2021). By integrating automated processes into the FE stage, the efficiency of the ML pipeline can be enhanced, bringing about improvements in decision making to a variety of real world contexts (Gada et al. 2021). This development supports data scientists and also extends the reach and impact of ML across different fields. Through AFE, intensive use of human experts can be reduced, and their skills can be used in other contexts which are more demanding. It can also accommodate those without data science expertise by allowing users to input data in its most raw format, including texts, images and audio.

The key characteristics that an ideal AFE model should possess are given below (Eldeeb et al. 2022).

- **User friendly:** The model should be straightforward to use, minimizing the need for tuning numerous hyperparameters.
- **Scalable:** The model must be capable of managing large datasets with many instances and features, necessitating the use of distributed computing. Ideally, most components of the AFE model should support parallelization.
- **Model agnostic:** The model should generate expressive features that enhance the performance of a broad spectrum of ML models, without being restricted to any specific algorithm.
- **Interpretable:** It should be possible to trace back the generated features to the original features by understanding the transformations applied.

2.3.1 Building an AFE system

In general, majority of the AFE models follow a 3 step process which is repeated many times. Firstly, new features are generated by combining the original ones using either unary, binary or higher order operators, after which all the features are ranked based on their value, and the most informative features are added back into the dataset (Zöller and Huber 2021).

2.3.2 Evaluating the effectiveness of an AFE system

Evaluating the effectiveness of an AFE system typically involves comparing model performance before and after the application of AFE, using an appropriate task specific performance measure such as F1 for classification or Root Mean Square Error (RMSE) for regression (Eldeeb and Elshawi 2024; Eldeeb et al. 2022). In both cases, the evaluation should be performed using the same ML model to ensure that any observed improvement can be attributed to the features generated by AFE rather than the intrinsic properties of a different model. For instance, if a linear regression model is used to assess performance prior to AFE, a linear regression model must again be used after AFE is applied. This consistency is crucial because different models, such as random forests, may internally perform FE as part of their learning process, making it unclear whether improvements in performance are due to the features or the model's internal mechanisms. Thus, it is not possible to compare two different models such as linear regression and random forest when evaluating an AFE framework.

While performance comparisons before and after AFE are informative, they are insufficient on their own to assess the reliability of engineered features. AFE methods can generate high-variance synthetic features that overfit the training data, particularly when large operator search spaces are explored. As a result, rigorous evaluation protocols such as cross-validation should be considered essential when assessing AFE effectiveness, rather than relying on single train–test splits. In addition, regularization mechanisms within the downstream learning model (e.g., L1 or L2 penalties) play a critical role in mitigating the effect of overfitting introduced by feature space explosions. Beyond predictive performance, stability-based criteria—such as the consistency of selected features or feature importance rankings across resampled datasets—can provide valuable complementary insight into the robustness of an AFE system. Without such safeguards, it would be difficult to assess genuine improvements in feature quality.

While existing studies evaluate AFE methods using a variety of criteria, the evaluation strategies adopted across the literature remain highly heterogeneous. In many cases, AFE systems are assessed primarily through predictive performance improvements (Eldeeb and Elshawi 2024; Eldeeb et al. 2022), while other studies additionally consider factors such as interpretability and computational cost. As a result, direct comparisons across studies are often difficult because different datasets, experimental protocols, and evaluation criteria are used. To address this inconsistency and consolidate the evaluation considerations discussed above, a unified evaluation framework for AFE systems has been proposed below.

- **Evaluation across task types.** AFE systems should ideally be evaluated on both regression and classification problems to demonstrate their applicability across different ML settings.

- **Ease of use:** The practical usability of an AFE algorithm should be evaluated based on how easily it can be adopted in practice. This includes factors such as the availability of code (e.g., open-source libraries), straightforward installation procedure (e.g., via pip), and simple implementation workflows (e.g., fit/transform pipelines). Methods that require minimal configuration are generally more accessible to practitioners.
- **Predictive performance improvement.** The effectiveness of an AFE system should be evaluated by comparing predictive performance before and after the application of AFE using the same downstream ML model (Kaul et al. 2017; Khurana et al. 2018; Dor and Reich 2012; Markovitch and Rosenstein 2002; Liu et al. 2024a; Roshanski et al. 2023; Fan et al. 2010).
- **Handling of different data types.** The ability of the AFE system to process different variable types should be evaluated. In particular, the method should be capable of encoding categorical variables.
- **Generality across learning algorithms.** The benefits of the generated features should ideally be evaluated across multiple downstream models (e.g., linear models, tree-based models, and neural networks) to determine whether the improvements are model-specific or generally useful.
- **Dataset compatibility.** The algorithms should be tested across datasets with varying characteristics, including different sample sizes and feature dimensionalities, to identify conditions under which the method may fail to return a result.
- **Feature space management.** Since AFE can potentially generate a very large number of candidate features, it is important to assess how the framework manages feature space growth. This includes evaluating whether FS is used to prevent combinatorial explosion (Kaul et al. 2017; Horn et al. 2020; Roshanski et al. 2023; Fan et al. 2010).
- **Computational efficiency and scalability.** The computational cost of the AFE system should be analysed, including execution time (Lu et al. 2023; Zhang et al. 2023; Markovitch and Rosenstein 2002; Nargesian et al. 2017; Eldeeb et al. 2022; Eldeeb and Elshawi 2024; Liu et al. 2024a; Roshanski et al. 2023) and how the cost scales as the number of features increases (Lu et al. 2023; Eldeeb et al. 2022; Eldeeb and Elshawi 2024).
- **Interpretability** Finally, it is important to assess whether the generated features remain interpretable. Five critical measures of interpretability have been provided in Sect. 5.

Together, these criteria provide a structured guideline for systematically evaluating AFE systems beyond predictive performance alone.

2.3.3 Choosing appropriate models for AFE

Some approaches also propose combining AFE with automated model selection to identify the best performing model-feature combination (Khurana et al. 2016). This is because feature quality and utility are inherently model dependent. Efforts to create universally optimal feature sets that perform well across all models often fall short because the effectiveness of features is tightly linked to the learning structures of specific models. As shown in prior research (Salvador et al. 2016a), effective FE should be closely integrated with model selection as part of a holistic ML pipeline optimization. Rather than seeking universally good features, AFE systems should recognize that different models benefit from different types of feature transformations. AutoML combined with AFE thus offers a promising solution by

jointly optimizing FE steps and model configurations, ensuring that the model is tuned to the specific feature set being used.

2.3.4 Meta knowledge and AFE

Meta knowledge refers to knowledge about knowledge. In the context of ML, it involves understanding and capturing insights about which algorithms, techniques, or processes work well in different situations (Lemke et al. 2015; Ali et al. 2020; Bonidia et al. 2022). By using meta knowledge, ML practitioners can more efficiently choose suitable algorithms and workflows, reducing the need for trial and error. It also helps in optimizing the entire pipeline, from data preprocessing to model selection, speeding up the process and potentially leading to better results (Brazdil 1998; Salvador et al. 2019; Ali et al. 2020).

Meta knowledge can also be applied to AFE (Katz et al. 2016). By understanding patterns in past data and model performance, meta knowledge can guide decisions about which techniques are most likely to improve algorithm performance for a given dataset. For example, if past experiments show that Naive Bayes consistently performs better with discretized features in similar datasets, meta knowledge can inform the decision to discretize numerical features in new datasets (Yang and Webb 2008). By mapping the meta features of each feature in the data set, to the requirements of the ML model to be used, it is also possible to automate part of the FE process (For example, support vector machines would need the categorical features encoded whereas random forest models can handle them) (Zöller and Huber 2021). Additionally, it is also possible to identify which operators to use in the FG process of an AFE model through the use of meta features (Eldeeb and Elshawi 2024).

There are many different meta features which can be generated from a given dataset but they can be grouped into three main categories. Details of this grouping are provided in Table 2.

The primary limitation of using meta features is the lack of reliable meta FG techniques and benchmarking systems suitable for non stationary environments, along with the challenges in systematically collecting and evaluating base model performance data for evolving predictive systems (Ali et al. 2020). Furthermore, this knowledge might not always generalize to new situations, making it somewhat less reliable than direct experimentation. Given that neither approach is perfect, combining both meta knowledge and experimentation may provide better results (Brazdil 1998).

Table 2 Meta-feature groupings

Meta-feature grouping	Definition	Representative examples
Descriptive, statistical, and information theoretic (DSIT)	Focuses on extracting basic characteristics of datasets (Ali et al. 2020)	Meta-features such as the number of instances and variables (Rendell et al. 1987); measures like variability and skewness
Model-based approach	Creates features by analyzing the structure and performance of models generated from the data (Ali et al. 2020)	Features derived from the shape, size, and complexity of decision trees (Bensusan et al. 2000)
Landmarking	Uses the performance of simple and quick algorithms (landmarkers) to describe datasets (Ali et al. 2020)	Performance of a basic Naive Bayes model on the dataset (Pfahringer et al. 2000)

Adapted from Ali et al. (2020)

2.4 Summary

In this section, the foundational concepts of FE were explored, emphasizing its central role in enhancing ML model performance. The discussion highlighted the challenges of traditional FE processes and the rising significance of AFE as a means to overcome these limitations. Key aspects such as FG, FS, feature space transformation, and the principles of building and evaluating effective AFE systems were examined. This section provides the necessary groundwork for understanding AFE methods, setting the stage for the next section, which focuses on existing models for performing AFE.

3 Automated FE models

This section presents an overview of AFE models, which operate under the assumption of a fixed number of data points and features. These models provide a foundational understanding of traditional approaches, setting the stage for the discussion of adaptive FE models that accommodate dynamic and evolving data scenarios.

There are many AFE models which are now widely available, created by different organizations and excelling in various tasks. This can be seen in Fig. 3 which shows an exponential growth in the number of studies which focused on AFE over the years.

As observed in Fig. 3, AFE has been around for decades (Luo et al. 2019; Chen et al. 2019; Zhu et al. 2020; Ouyang et al. 2018; Markovitch and Rosenstein 2002; Dor and Reich 2012), transforming many different areas of study including computer science (Malberg et al. 2024), pharmaceutical research (Yeung et al. 2023), medicine (Björneld and Löwe 2024), finance (Zhang et al. 2021) and many others. These algorithms have been developed using a variety of methods for automating the FE process, with some making use of meta learning approaches (Katz et al. 2016; Nargesian et al. 2017; Harari and Katz 2022; Zhang et al. 2020; Reyes-Amezcua et al. 2022), while others incorporate reinforcement learning (Khurana et al. 2018; Lu et al. 2023; Gaudel and Sebag 2010; Zhang et al. 2019,

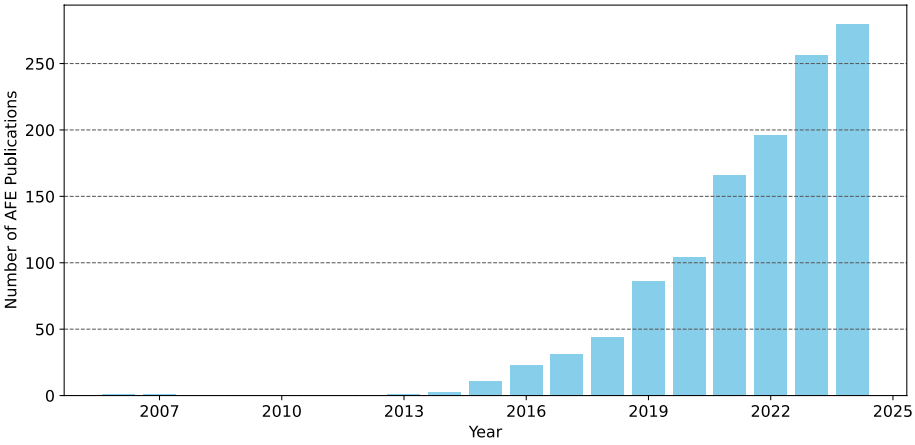

Fig. 3 Number of AFE publications over the years—represents the number of papers which include the exact phrases of AFE, automatic feature engineering, or autonomous feature engineering over the years, as found through the Scopus database

2020; Wang et al. 2022a; He et al. 2025; Liu et al. 2024b; Xiao et al. 2024), or simulated annealing (Ladeira et al. 2021), and others even make use of decision trees (Markovitch and Rosenstein 2002; Fan et al. 2010; Piramuthu and Sikora 2009; Eldeeb et al. 2022; Zhang et al. 2023; Shi et al. 2020; Dumitrescu et al. 2022; Wang et al. 2025b). Furthermore, using regression models (Kaul et al. 2017) or evolutionary algorithms (Ahmed et al. 2020; Veloso de Melo and Banzhaf 2018; Zhu et al. 2022) are common in performing AFE. These methods have been used to develop AFE algorithms for general purpose scenarios (Eldeeb et al. 2022; Eldeeb and Elshawi 2024; Horn et al. 2020), as well as for specific use cases, such as working on relational datasets (Kanter and Veeramachaneni 2015; Fatima et al. 2018; Lam et al. 2017; Zhang et al. 2018; Kassarnig and Wotawa 2018; Zhao et al. 2020; Lam et al. 2021; Qi et al. 2024), or for time series data (Christ et al. 2018; Renault et al. 2023; Roque and Ram 2019; Barandas et al. 2020; De Brabandere et al. 2022). In addition, there are also AFE methods developed in an unsupervised manner, which extract new features without the need for labeled data (Silva Rodrigues et al. 2021a, 2021b; Khobragade and Ponde 2025). A smaller but growing category of AFE methods also relies on external knowledge sources, where entities in the dataset are linked to structured resources such as DBpedia and new features are automatically generated and selected from these external repositories (Harari and Katz 2022; Bouadi et al. 2024, 2025a). However, providing an exhaustive review of the different AFE methods which are available is beyond the scope of this research, as this paper specifically focuses on the methods that promote not only an improvement in the performance, but also in the interpretability. As such, the subsequent paragraphs speak of some of the prominent interpretable AFE methods which are available.

Autofeat for starters, is one of the most widely adopted interpretable AFE methods which are currently present (Horn et al. 2020). It is also available as a python library that offers linear regression models, adopting the style of scikit learn, and featuring AFE. Autofeat follows the generate and select FE strategy, and is capable of creating non linear features which are also interpretable. As the creation of all possible features is impractical, the features are often generated in an intelligent manner. For example, if physical units are provided, Autofeat would not substract a feature representing a temperature from a volume. Using various operators, Autofeat generates a vast number of new, non linear features which have been frequently shown to improve the performance of linear models. Autofeat has also showcased its capability in modeling medical data (An et al. 2024), as well as many others (Nguyen et al. 2024). However, it can only cater to single table datasets, and thus cannot be used in situations where data is coming from many different sources (Eldeeb et al. 2022). A comprehensive overview of the AFE process followed by AutoFeat is given in Fig. 4.

The scalable automatic feature engineering framework (Shi et al. 2020) operates similarly to the Autofeat library, as both follow a two stage process involving FG and FS. However, unlike Autofeat, which exhaustively generates a large pool of transformed features, this method uses the XGBoost model to identify useful feature combinations. It then applies a variety of mathematical operators (similar to those used in Autofeat) to combine only these potentially useful feature, thereby significantly reducing the search space and computational cost. In the selection stage, it ranks and filters the newly created features based on information value, correlation, and importance scores, resulting in a more scalable and efficient process than the exhaustive strategy used by Autofeat. However, this efficiency comes at quite a heavy cost. Since an exhaustive exploration is not conducted, there is a possibility of missing potentially useful feature combinations.

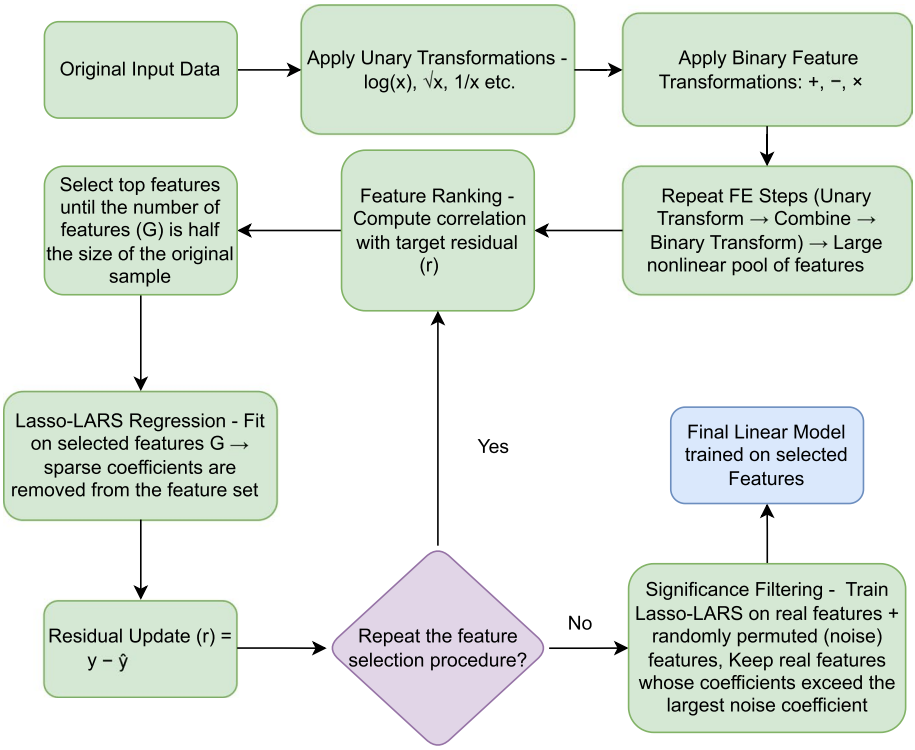

Fig. 4 Overview of AutoFeat AFE process (Horn et al. 2020)

3.1 AFE methods used within an AutoML pipeline

As mentioned in Sect. 2.3.3, effective FE should be closely integrated with model selection as part of a holistic ML pipeline optimization. One of the earlier AFE methods (which also focused on interpretability), namely Cognito (Khurana et al. 2016), does exactly this. That is, the Cognito platform automates the process of model selection as well. This is one of the biggest advantages Cognito has over majority of the other AFE models in existence today. The FE and ML model building phases are very much interconnected, as one directly affects the other, and hence Cognito tackles an important problem here. Additionally, Cognito allows domain experts to specify constraints to guide the FE process, making it adaptable to specific needs [which is a flexibility only offered in a few other methods (Radeev and Vinogradova 2025; Ziegler et al. 2024)]. Cognito performs AFE by exploring a hierarchical transformation tree, where each node represents a dataset with newly constructed features and each edge corresponds to a transformation. The system employs greedy search strategies to incrementally construct and evaluate new features based on their predictive performance. While Cognito does address the drawbacks of Autofeat, a potential downside of this approach is that it might prematurely eliminate some features that could have become useful through further transformations. More modern studies have taken Cognito a step further by wrapping it into an interactive, user friendly Jupyter notebook interface where users can interactively view, accept, or reject feature suggestions (Nargesian et al. 2018).

BigFeat (Eldeeb et al. 2022) is an AFE model which addresses the biggest drawback of Cognito (the problem of prematurely discarding features). The suggested approach in the paper is quite similar to the the scalable automatic feature engineering framework (Shi et al. 2020), and it addresses these shortcomings by maintaining a record of crucial transformations. BigFeat is also created to be domain and model independent, while ensuring computational efficiency, scalability, and interpretability. Furthermore, BigFeat has been compared with Autofeat and shown to perform better in such direct comparison. BigFeat, similarly to Autofeat, functions in a FG and FS manner. However, according to the authors, generating all possible features by combining base features with all operators, results in a combinatorial explosion of the feature space—a scenario that is impractical to manage due to its complexity. BigFeat thus uses a decision tree to capture the features that perform well together (those that occur along the same branch of the tree are assumed to have some form of interaction which is useful in modeling the dependent variable). It is hence, much faster than Autofeat as well. It then uses a variety of operators for the FG process and removes redundant features by applying a unique FS approach in each iteration. A comprehensive overview of the AFE process followed by BigFeat is given in Fig. 5. While studies indicate that BigFeat performs extremely well in automatically constructing features for ML model optimization, it is not perfect. The drawback of BigFeat is that it starts by randomly selecting which operators to apply, and thus is not optimal. As such, the authors have recently proposed another AFE model (BigFeat-FE) (Eldeeb and Elshawi 2024) which uses meta learning to identify the best operators to apply for a given selection of features. Additionally, BigFeat-FE also automates the model selection process, just as Cognito did, further improving on a major drawback of BigFeat. However, both BigFeat and BigFeat-FE have three major limitations. Firstly, the FG process relies on the feature importance scores generated by tree based models (which may not always work for every ML model which is trained subsequently). Secondly, the FS process compares the correlation between parent features and offsprings alone, which, while computationally efficient, may not yield the best results for larger datasets. Lastly, the biggest drawback of BigFeat and BigFeat-FE is that it is only able to cater to the classification problems and fails in the case of regression problems.

Other work has focused on empirically evaluating how AutoFeat integrates into AutoML frameworks such as Auto-Sklearn and TPOT, highlighting both its advantages and limitations when used as part of a holistic pipeline optimization (Eldeeb et al. 2021). Beyond this, several studies have evaluated different AFE approaches in combination with AutoML systems (Shi and Saad 2023; de Winter et al. 2025).

3.2 AFE for linear or interpretable models

The most recent AFE models focus on creating a feature space within which one can achieve considerable performance even when using simple linear classification or regression models, thus maintaining interpretability without a reduction in accuracy. The SAFE framework presented in Gosiewska et al. (2021) for example, is founded on the concept that new features can be derived from a complex black box model and subsequently used to train a straightforward, interpretable model. This simpler model often retains the performance of the complex one. The new features capture the non-linear patterns recognized by the complex model and incorporate them as input features into a linear model. Results from this study indicated that with appropriate FE methods, it is possible to build linear interpretable models which can

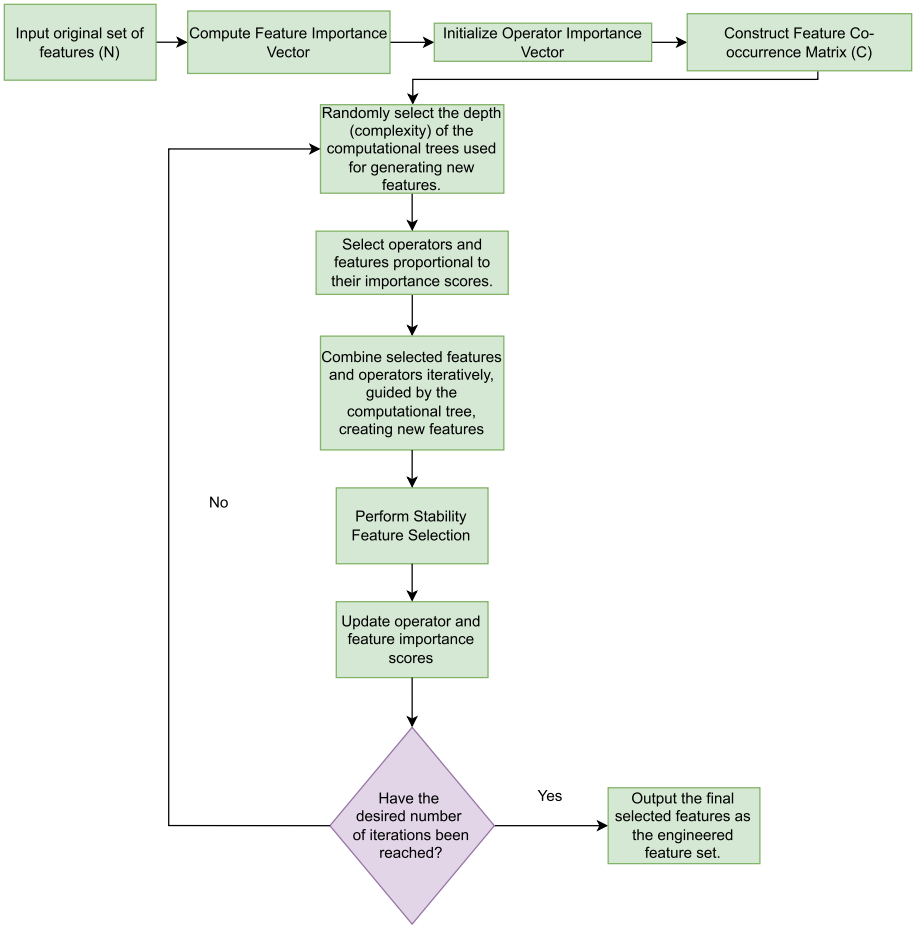

Fig. 5 Overview of BigFeat AFE process. **Feature importance vector:** Feature importance scores are computed using information gain and normalized to obtain the feature importance vector, **Operator importance vector:** A predefined set of operators is initially assigned equal importance scores, forming the Operator importance vector, **Feature co-occurrence matrix (*C*):** A matrix is created to capture how frequently pairs of features occur together within the same decision tree paths, **Stability feature selection:** Pairwise correlations between newly generated and original features are computed. If the correlation between a pair exceeds a predefined threshold, the feature with lower importance is discarded. Finally, the top *N* features with the highest importance scores are retained, **Operator importance update:** The importance score of each operator is increased by one for every successful feature it generates (i.e., features retained after selection). The updated operator scores are then normalized to update the Operator Importance Vector, **Feature importance update:** The feature importance vector is multiplied by the co-occurrence matrix (*C*), increasing the scores of features that frequently appear together in decision tree paths and thereby capturing feature interactions. (Eldeeb et al. 2022)

frequently outperform complex black box models. The algorithm for SAFE works as shown in Fig. 6. While quite useful in scenarios where both the performance and interpretability is quite important, SAFE does not capture interactions between variables. Thus, it is not suitable for modeling data generated from domains where such interactions are important or even critical to identify as part of the process of generating more complex features.

Similarly to SAFE, INAFEN (Liu et al. 2024a) is another interpretable AFE model which can aid logistic regression models achieve a performance which is comparable with complex black-box models, while retaining their interpretability. INAFEN first transforms numerical features by training a univariate decision tree per feature to find split thresholds. Each interval is then replaced by the difference between its expected target value and the global expected target, converting nonlinear effects into approximately linear, interpretable values. Categorical features are kept as-is, and together with discretized numerical features are one-hot encoded, after which association rule mining (e.g., Apriori/FP-Growth) generates binary cross-features to capture the interaction. Finally, a high-performing black-box teacher model produces soft labels (predicted probabilities), and logistic regression is

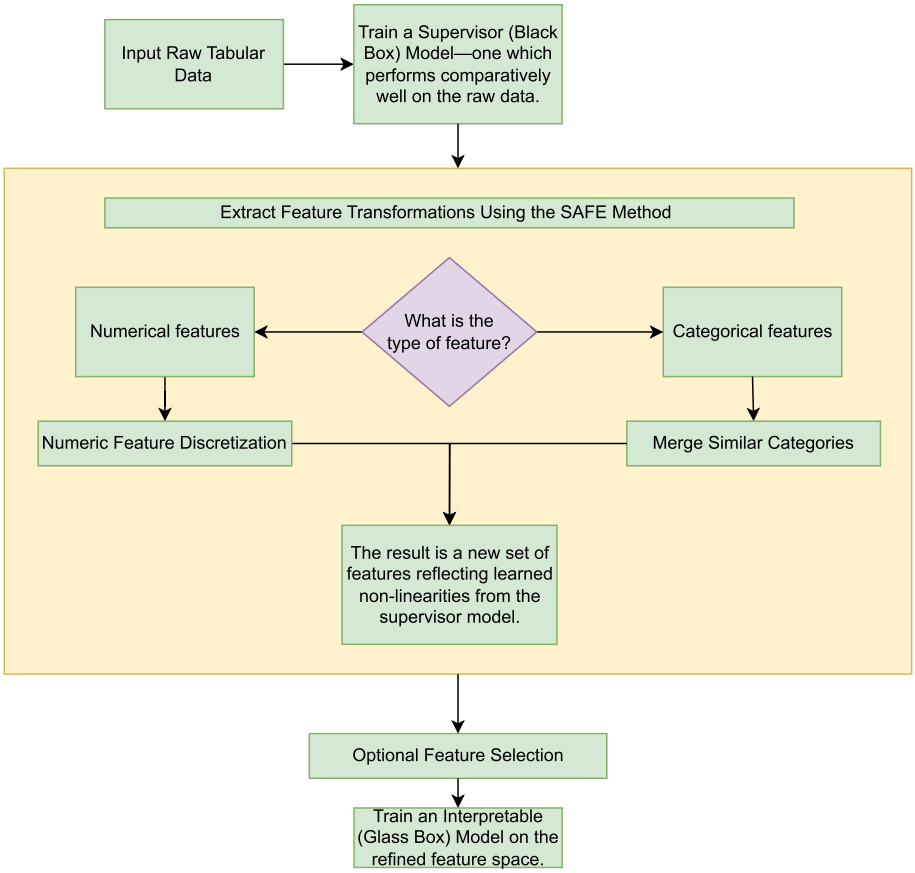

Fig. 6 Overview of SAFE AFE process. **Extract Feature Transformations Using the SAFE Method:** This step identifies meaningful variable transformations guided by the supervisor model's predictions. The process differs based on variable type, **Numeric feature discretization:** The influence of each numerical feature on model predictions is analyzed using partial dependence profiles or accumulated local effect plots. The PELT algorithm is then applied to detect significant change points, which are used to discretize the feature into interpretable bins, **Category merging:** Categories with similar impacts are grouped together to simplify the feature representation. Hierarchical clustering is used to identify and merge categories with similar effects, **Optional feature selection:** The resulting feature space includes both original and transformed features. Feature selection methods such as AIC, BIC, or importance-based rankings may be applied to reduce dimensionality and mitigate overfitting (Gosiewska et al. 2021)

trained on the engineered features using these soft labels, improving accuracy while preserving full model interpretability.

INAFEN, just as SAFE did, converts the non-linear relationships into linear, interpretable ones. However, while SAFE allows the modeling of any linear model, INAFEN is limited to logistic regression modeling. At the same time, INAFEN has an advantage over SAFE, in the sense that it is capable of capturing interactions between variables, while SAFE is not. Furthermore, INAFEN is computationally quite expensive. A key limitation shared by both SAFE and INAFEN arises from their reliance on discretization of continuous variables. While discretization can improve interpretability by converting smooth nonlinear effects into human-readable rules, it may substantially reduce model fidelity when the underlying relationship between a numerical feature and the target is continuous, weakly monotonic, or varies smoothly across ranges. In such cases, hard thresholding can collapse informative intra-bin variation, leading to loss of signal and reduced generalization performance. Moreover, INAFEN introduces additional challenges due to its use of tree-based partitioning to define feature thresholds. Tree–based thresholding can be unstable for noisy or high-variance numerical features, as small perturbations in the training data may lead to different split points being selected. Such threshold instability can result in inconsistent discretization across folds or resampled datasets, producing engineered features that are sensitive to noise rather than robust representations of the underlying relationship.

FACET (Roshanski et al. 2023) is another AFE model which is quite unique in its application. Unlike other AFE models which aim to provide effective feature space for a range of ML models to be subsequently trained on it, FACET focuses on automatically creating features which would enable one to train a much smaller decision tree but with the same level of accuracy that would have been obtained with a larger decision tree trained on the original data set. This reduction in the size of the tree also leads to an increase in the interpretability of the model. FACET uses a variety of mathematical operators and SVC based binary operators for FG. While FACET applies all the chosen operators to all possible combinations of binary variables, it gives the user the flexibility of deciding which operators to use. Hence, FACET is not suitable for use in very complex data environments which have many feature interactions. In terms of FS, FACET evaluates each new feature to decide whether it reduces the model complexity by a significant amount (exceeding a predefined threshold) while maintaining its accuracy (ensuring that the reduction in accuracy falls below a specified threshold). In each iteration, the features that meet this criteria are added back to the feature space. FACET is also found to be domain independent and has shown its capability in modeling decision trees. However, the model does have its limitations. Such as it is, it generates all possible features, and can be computationally expensive. Additionally, it only uses SVC and mathematical binary operators, whereas it could be beneficial to look into more ML model based operators.

While BigFeat, SAFE, and FACET all aim to enrich the feature space, they differ in the extent and nature of the interaction spaces they are able to explore, and none provides exhaustive coverage of complex dependencies. SAFE is inherently limited to capturing transformed marginal effects and does not represent interactions explicitly. FACET can implicitly capture interaction effects when they arise from mathematical or SVC-based operators and contribute to simpler tree structures. However, the discovery of features which capture the interaction effect is not a primary objective of the framework. As a result, higher-order or complex interaction patterns may be included only opportunistically and can

be missed when they do not directly lead to reductions in tree complexity. BigFeat offers the greatest capacity among the three, as higher-order non-linear interactions do emerge through iterative expansion of the feature space. However, the exploration of interaction patterns is guided by tree-based importance signals, favoring feature combinations that co-occur along decision-tree paths, which may lead to under exploration of useful dependencies that are not captured through tree based models. Together, these differences indicate that while each framework captures useful information, none guarantees exhaustive coverage of higher-order dependencies.

3.3 AFE for specific domains

Apart from these general purpose AFE models, there are also specialized AFE algorithms that focus on different types of domains. FELIX (Feature Engineering with LLMs for Interpretability and Explainability) is one such example which automatically identifies relevant features in textual data (Malberg et al. 2024). Another example is found in Yeung et al. (2023) where the author proposes a novel approach to AFE by using shallow ML models to derive interpretable features for peptide retention time prediction. Additionally, BioAutoML (Bonidia et al. 2022) is an AutoML model designed for the automated classification of biological sequences, and also carries out the AFE process as part of its operations. This model works similarly to Cognito and BigFeat-FE, where the modeling process and the AFE process work hand in hand. BioAutoML also addresses one of the main challenges in applying ML to biological sequence data, which is categorical and unstructured. Transforming these sequences into a structured numerical format is essential for ML applications, and the model proposed in this paper achieves this through a sophisticated FE process. Another AFE model used in the medical industry is aKDFE (Automated Knowledge Driven Feature Engineering) (Björneld and Löwe 2024). The AFE process in aKDFE involves iterative application of predefined operations to raw EHR data, creating new informative features. These operations, stored in a database (as defined by domain experts) are applied (based on specific rules and conditions) using the meta data of the features under consideration.

AFE has also been shown to play a critical role in improving robustness and downstream task reliability in high-stakes settings, including crisis-related and adversarial environments. For instance, the MEGA framework demonstrates how engineered graph-based features can enhance infodemic risk management by supporting robust spambot detection and influence estimation during large-scale misinformation events (Hang et al. 2023). Similarly, recent work on sentence-based adversarial attacks against AI-generated text detectors highlights the importance of feature-level representations in both strengthening detection capability and exposing model vulnerabilities under adversarial manipulation (Tu et al. 2025). Together, these studies illustrate how FE can enhance model capability, robustness, and downstream task reliability in high-stakes domains.

3.4 Deep learning and representation learning for AFE

Deep Learning (DL) models are also capable of automatically deriving expressive representations from original input features. This is especially useful in domains such as image processing which can be difficult for humans (Anderson et al. 2016). However, this method has several practical limitations. Firstly, training effective DL models typically requires a sub-

stantial amount of data, which may not be available in many fields (Sarker 2021). Secondly, while DL has shown notable success with data types such as video, images, speech, and natural language, it is less effective with general numerical data, which spans a broad range of domains and often necessitates FE (Eldeeb et al. 2022). Thirdly, DL models involve training numerous parameters (Sarker 2021), a process that can be time intensive and demands significant computational power (Eldeeb et al. 2022). Lastly, the features generated by DL models lack interpretability (Katz et al. 2016), which can make it difficult for practitioners to trust models that rely on these features. To mitigate these drawbacks, it is essential to use interpretable AFE techniques. The quality of the features used is the most important factor in improving the performance of the learning algorithm, and this is evidenced by the performance displayed by DL algorithms which can learn effective representations from the original data. Figure 7 illustrates the architecture of a DL model. The model comprises of multiple hidden layers, where the first hidden layer constructed a new set of feature from the original input variables. The second hidden layer further transforms these features, creating another set of features derived from the first layer. It is from this refined feature space that the model makes its predictions, enhancing the capability to capture and use complex patterns within the data.

DL is just one form of Representation Learning (RL). RL is a technique that allows models to automatically learn and capture the most relevant features from the input data, with a specific goal in mind (Tzelepi et al. 2022). This approach differs from traditional methods, where data scientists had to manually select relevant features, which could be time consuming and sometimes limiting (Khurana et al. 2016). Instead, RL entrusts models to

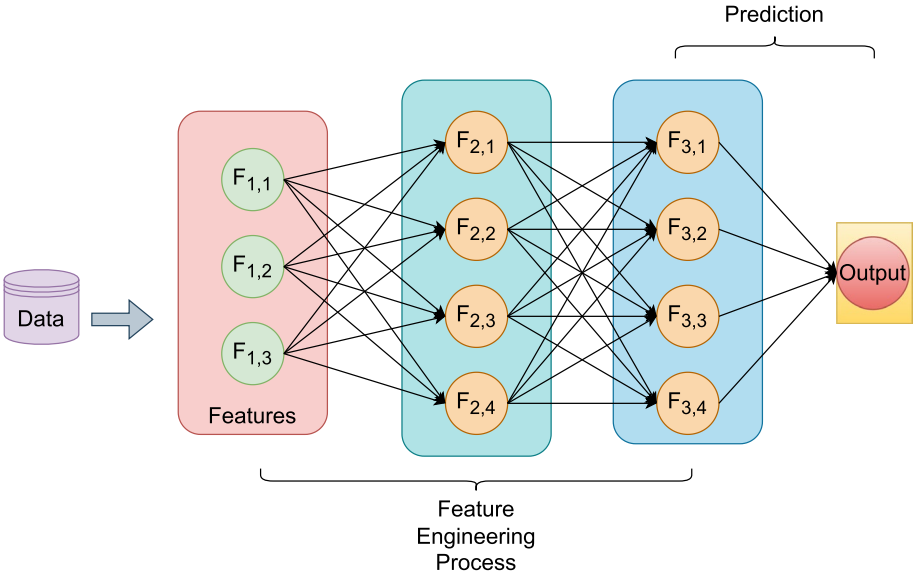

Fig. 7 Illustration of a DL architecture for FE. The input layer represents the original feature space, while subsequent hidden layers generate progressively transformed feature spaces. Each layer captures increasingly abstract representations of the input, optimized toward enhancing predictive performance. The transition between layers can be interpreted as an AFE process, where new features are constructed automatically to better support the learning task. The final output produces the model's prediction

extract complex information autonomously, a capability especially advantageous with high dimensional and sparse data.

3.5 Future of AFE

In recent times, AFE has shifted away from the traditional exhaustive search or decision-tree based approaches, towards LLM driven techniques that promote understanding and external context (Bouadi et al. 2025b; Hollmann et al. 2023; Nam et al. 2024). These methods address majority of the previous drawbacks. It not only generates interpretable features but also semantically meaningful features. Additionally, it also promotes the use of domain knowledge in the AFE process. However, it requires well structured prompts and dataset descriptions, which may defeat one of the purposes of AFE, which is to democratize ML and make it accessible to non experts. Additionally, it may depend on template adherence and additional context beyond the raw data, which cannot always be provided if the relevant expertise is not available.

In addition to these practical limitations, LLM-guided AFE introduces several methodological challenges that remain largely underexplored. One key issue is prompt variance, where small changes in prompt phrasing can lead to substantially different generated features, reducing reproducibility. Closely related is the risk of hallucination, whereby LLMs may generate features that appear useful or semantically meaningful, but are, in reality, spurious representations. Furthermore, many LLM-based approaches are not fully deterministic, meaning that the same prompt can produce different feature suggestions across repeated runs. Finally, aligning LLM-suggested feature definitions with downstream modeling assumptions remains challenging. While certain properties such as feature scaling are largely irrelevant for some model classes (e.g., tree-based methods) but critical for others (e.g., linear or distance-based models), LLM-guided AFE may generate transformations that are misaligned with the assumptions of the chosen downstream model. These challenges highlight the need for careful use of LLM-guided AFE, with particular attention to validation, reproducibility, and alignment with downstream modeling assumptions.

3.6 Source codes—AFE

Practitioners aiming to develop novel algorithms must establish the robustness and efficacy of their methods by benchmarking against existing models. Similarly, those interested in a specific AFE model often require access to its underlying code to understand and use it. Recognizing these needs, we have compiled a list of publicly available code bases associated with the AFE models discussed in this review. These resources, detailed in Table 3, provide the links needed for researchers looking to advance the current state of the art of AFE. It is however important to note that only the general purpose AFE methods have been provided in this table, and that methods which focus on specific domains such as biology (Bonidia et al. 2022) have not been provided in this table.

3.7 Benchmarking datasets—AFE

For ML practitioners focused on developing novel algorithms, another important factor is access to a diverse array of benchmarking datasets. These datasets facilitate the testing and

Table 3 Publicly available codes of existing AFE models

Algorithm	GitHub Link	References	Year(s)
INAFEN	https://github.com/Liumucan/INAFEN	Liu et al. (2024a)	2024
BigFeat	https://github.com/DataSystemsGroupUT/BigFeat	Eldeeb and Elshawi (2024) and Eldeeb et al. (2022)	2024
Felix	https://github.com/simonmalberg/felix	Malberg et al. (2024)	2024
OpenFE	https://github.com/IIIS-Li-Group/OpenFE	Zhang et al. (2023)	2023
AutomaticFE CompactDecisionTrees	https://github.com/anonGHU/AutomaticFE_CompactDecisionTrees	Roshanski et al. (2023)	2023
Catch	https://github.com/Grason-Lu/Catch	Lu et al. (2023)	2023
TSFuse	https://github.com/arnedb/tsfuse	De Brabandere et al. (2022)	2022
SAFE	https://github.com/ModelOriented/SAFE	Gosiewska et al. (2021)	2021
TSFEL	https://github.com/fraunhoferportugal/tsfel	Barandas et al. (2020)	2020
DIFER	https://github.com/PasaLab/DIFER	Zhu et al. (2020)	2020
AutoFeat	https://github.com/cod3licious/autofeat	Horn et al. (2020)	2020
tsfresh	https://github.com/blue-yonder/tsfresh/tree/v0.11.0	Christ et al. (2018)	2018
AutoLearn	https://github.com/saket-maheshwary/AutoLearn	Kaul et al. (2017)	2017
ExploreKit	https://github.com/giladkatz/ExploreKit	Katz et al. (2016)	2016
FeatureTools	https://github.com/alteryx/featuretools	Kanter and Veeramachaneni (2015)	2015

validation of new methods, while ensuring that the developed algorithms can handle a wide range of applications and data characteristics. In this paper, we have also compiled a selection of 20 of the most widely used benchmarking datasets, as cited across numerous studies. These datasets, which span various fields and tasks are thoroughly described along with their characteristics and the specific papers that used them. This compilation is presented in Table 4 for the classification scenario and Table 5 for the regression scenario.

It is important to note that many of the datasets listed in the tables originate from legacy repositories such as the UCI ML Repository and OpenML. These datasets, while widely used and well-documented, are often small in scale and simplistic in structure. They reflect an earlier era of ML research and are frequently employed as illustrative examples across the literature. Although they remain useful for demonstrating the efficiency and interpretability gained by using AFE methods, they may not pose substantial challenges or offer the complexity needed for drawing strong generalizable conclusions about the performance of AFE techniques. Therefore, while these datasets serve a valuable role in establishing baseline functionality, their limitations must be considered when used for benchmarking purposes.

Table 6 provides a unified comparative summary of 20 representative AFE methods as evaluated based on the criteria provided in Sect. 2.3.2.

Table 4 Summary of publicly available benchmarking datasets—Classification

Dataset name	Instances	Features	Subject area	References
Arcene	200	10,001	H&M	Eldeeb et al. (2022), Eldeeb and Elshawi (2024), Kaul et al. (2017)
Hepatitis	155	19	H&M	Lu et al. (2023), Dor and Reich (2012), Nargesian et al. (2017)
EEG-Eye	14,980	14	H&M	Eldeeb et al. (2022), Eldeeb and Elshawi (2024), Gosiewska et al. (2021)
Diabetes	768	9	H&M	Liu et al. (2024a), Khurana et al. (2016), Katz et al. (2016), Gosiewska et al. (2021), Kaul et al. (2017), Nargesian et al. (2017), Dor and Reich (2012), Roshanski et al. (2023)
Fertility	100	9	H&M	Kaul et al. (2017), Lu et al. (2023), Nargesian et al. (2017), Roshanski et al. (2023)
Sonar	208	60	P&C	Eldeeb et al. (2022), Eldeeb and Elshawi (2024), Kaul et al. (2017), Fan et al. (2010), Nargesian et al. (2017)
Ionosphere	351	34	P&C	Khurana et al. (2016), Kaul et al. (2017), Lu et al. (2023), Khurana et al. (2018), Fan et al. (2010), Nargesian et al. (2017), Dor and Reich (2012)
Spambase	4601	57	CS	Gosiewska et al. (2021), Lu et al. (2023), Khurana et al. (2018), Nargesian et al. (2017)
KC1	2109	22	CS	Eldeeb et al. (2022), Eldeeb and Elshawi (2024), Gosiewska et al. (2021)
German Credit Data	1000	20	SS	Liu et al. (2024a), Eldeeb et al. (2022), Khurana et al. (2016), Eldeeb and Elshawi (2024), Katz et al. (2016), Gosiewska et al. (2021), Lu et al. (2023), Khurana et al. (2018), Dor and Reich (2012)
Labor	57	17	SS	Nargesian et al. (2017), Dor and Reich (2012)
Balance	625	4	SS	Markovitch and Rosenstein (2002), Dor and Reich (2012), Nargesian et al. (2017)
Credit-a	690	15	Bu	Lu et al. (2023), Dor and Reich (2012), Nargesian et al. (2017)
Australian Credit Approval	690	14	Bu	Liu et al. (2024a), Khurana et al. (2016), Eldeeb and Elshawi (2024), Katz et al. (2016), Roshanski et al. (2023)
Amazon Employee	32,769	10	Bu	Eldeeb and Elshawi (2024), Lu et al. (2023), Khurana et al. (2018)
Blood	748	4	Bu	Eldeeb et al. (2022), Eldeeb and Elshawi (2024), Roshanski et al. (2023)
Iris	150	4	Bi	Markovitch and Rosenstein (2002), Dor and Reich (2012), Roshanski et al. (2023)
Tic-Tac-Toe	958	9	Ga	Gosiewska et al. (2021), Roshanski et al. (2023), Markovitch and Rosenstein (2002)
Gina	3153	971	Ot	Eldeeb et al. (2022), Eldeeb and Elshawi (2024)
Madelon	2600	501	Ot	Eldeeb et al. (2022), Eldeeb and Elshawi (2024), Gosiewska et al. (2021), Nargesian et al. (2017)

H&M Health and Medicine, *P&C* Physics and Chemistry, *CS* Computer Science, *SS* Social Science, *Bu* Business, *Bi* Biology, *Ga* Games, *Ot* Other

Table 5 Summary of publicly available benchmarking datasets—Regression

Dataset name	Instances	Features	Subject area	References
Boston Housing	506	14	Bu	Horn et al. (2020), Lu et al. (2023), Khurana et al. (2018), Dor and Reich (2012), Khurana et al. (2018)
Wine quality	4898	11	Bu	Horn et al. (2020), Lu et al. (2023), Khurana et al. (2018), Roshanski et al. (2023)
California Housing	20,640	8	Bu	Zhang et al. (2023)
pyrim	74	28	P&C	Dor and Reich (2012)
Concrete Compressive Strength	1030	8	P&C	Horn et al. (2020)
Airfoil	1503	5	P&C	Horn et al. (2020), Khurana et al. (2018), Lu et al. (2023), Khurana et al. (2018)
fri c1 500 50	500	51	Sy	Lu et al. (2023) and Khurana et al. (2018)
fri c3 1000 50	1000	51	Sy	Lu et al. (2023) and Khurana et al. (2018)
fri c4 500 50	500	51	Sy	Lu et al. (2023) and Khurana et al. (2018)
fri c4 1000 50	1000	51	Sy	Lu et al. (2023) and Khurana et al. (2018)
fri c2 1000 25	1000	26	Sy	Lu et al. (2023) and Khurana et al. (2018)
fri c1 1000 25	1000	26	Sy	Lu et al. (2023) and Khurana et al. (2018)
fri c3 1000 25	1000	26	Sy	Lu et al. (2023) and Khurana et al. (2018)
bodyfat	251	15	H&M	Dor and Reich (2012)
Forest Fires	517	12	C&E	Dor and Reich (2012)
Quakes	2178	4	C&E	Dor and Reich (2012)
Servo	167	4	CS	Dor and Reich (2012)
auto93	93	23	Ot	Dor and Reich (2012)
autoPrice	159	16	Ot	Dor and Reich (2012)
autoMpg	398	7	Ot	Dor and Reich (2012)

H&M Health and Medicine, *P&C* Physics and Chemistry, *C&E* Climate and Environment, *CS* Computer Science, *Sy* Synthetic Dataset, *Bu* Business, *Ot* Other

4 Adaptive FE models

The previous section introduced several non adaptive AFE models, which have proven to be highly effective in enhancing the performance and interpretability of ML models. However, these static models often face limitations in evolving data scenarios, where underlying data distributions change over time (a phenomenon also very often referred to in the literature as concept drift). In such cases, the performance of non adaptive models may degrade, necessitating the use of adaptive AFE models. However, unfortunately, to date, there are no adaptive AFE models in existence. Yet, there are many adaptive FE algorithms which include streaming FS and extraction, which are designed to dynamically adjust to changing data distributions, thereby maintaining (and sometimes even improving) performance over time. This section introduces the literature relevant to adaptive FE algorithms, exploring their methods and applications in addressing concept drift.

Conceptually, an adaptive ML model can be updated through either a replacement strategy, where the old model is discarded and a new one is created based on updated data, or an incremental scheme, where the model evolves in response to new data. Incremental learning can additionally be implemented in 3 ways. Instance based (where the model is updated with each new data point), batch based (where updates occur after accumulating a

Table 6 Evaluation of AFE systems based on unified criteria

Method	TT	EI	PI	CDH	G	DC	FSM	TC	I
AutoCross (Luo et al. 2019)	Classification	×	✓	✓	✓		✓	✓	✓
NFS (Chen et al. 2019)	✓	×	✓		×	✓	✓		
DIFER (Zhu et al. 2020)	✓	✓	✓	×		✓	×		
FICUS (Markovitch and Rosenstein 2002)	Classification	×	✓	✓	✓		✓	✓	
AutoFeat (Horn et al. 2020)	✓	✓	✓	✓	✓		✓		✓
BigFeat (Eldeeb et al. 2022)	Classification	✓	✓		✓	✓	✓	✓	✓
BigFeat-FE (Eldeeb and Elshawi 2024)	Classification	✓	✓		✓	✓	✓	✓	✓
SAFE-ML (Gosiewska et al. 2021)	✓	✓	✓	✓	✓	✓	✓		✓
INAFEN (Liu et al. 2024a)	Classification	✓	✓	✓	×				✓
FACET (Roshanski et al. 2023)	Classification	✓	×		×		✓		✓
SAFE (Shi et al. 2020)	Classification	×	✓		✓		✓	✓	
Cognito (Khurana et al. 2016)	Classification	×	✓		✓	✓	✓		
Semi-Automated FE (Radeev and Vinogradova 2025)	Classification	×	✓	✓	✓		✓		✓
GELFE (de Winter et al. 2025)	Classification	✓	✓	✓	✓		✓		
RAAF (Ladeira et al. 2021)	Classification	×	✓	✓	✓		✓		
ReaGen (Bouadi et al. 2025b)	✓	×	✓	✓	✓	✓	✓	✓	✓
OpenFE (Zhang et al. 2023)	✓	✓	✓	✓	✓		✓	✓	
AutoLearn (Kaul et al. 2017)	Classification	×	✓		✓	✓	✓		
ExploreKit (Katz et al. 2016)	Classification	✓	✓		✓		✓		✓
LFE (Nargesian et al. 2017)	Classification	×	✓		×	✓			✓

TT Task Types—Classification, Regression, or both (✓), *EI* Ease of Implementation—Indicating the availability of the code, *PI* Performance Improvement—Whether AFE is focused on improving ML model performance, *CDH* Categorical Data Handling, *G* Generality—Whether the AFE can optimize for multiple different ML models, *DC* Dataset Compatibility—Can handle/tested on high dimensional datasets, *FSM* Feature Space Management—Whether FS is incorporated in the AFE process, *TC* Time Complexity—Whether the details of the execution time per dataset have been included and/or discussed, *I* Interpretability—Whether the AFE method claims to be interpretable. ✓ = Yes, × = No, = Not specified

predefined number of new data points), or ensemble based (where the old model remains untouched, a new model is created, and the final output is derived by manipulating their weights) (Zliobaite and Gabrys 2014).

While FE is an integral part of building well performing ML models the adaptation of and/or creation of new features spaces has not been commonly considered within the continuously learning ML models (Zliobaite and Gabrys 2014). However, adaptive FE algorithms have been slowly transforming not only the field of data science, but also many others. For example, streaming FS has been found to be useful in both image processing (Wu et al. 2013), and the medical industry (Zhou et al. 2021c). In the following sections we shall examine the current state of the art of this area and provide some discussion with regard to their readiness to be extended towards adapative AFE frameowrks in the near future.

4.1 Dynamic FS—incoming features

While there has been an exponential increase in publications on online FS methods over the years, as illustrated in Fig. 8, one of the earlier approach which is quite influential can be found in Perkins et al. (2003). The paper introduces a FS method that integrates FS and model training within a regularized risk minimization model. The approach, called grafting,

incrementally builds models by iteratively evaluating and adding features that most improve model performance. A descriptive overview of the grafting algorithm is shown in Fig. 9. As this method avoids exhaustive evaluations and retraining from scratch, it is computationally efficient. However, its reliance on a gradient heuristic may occasionally include irrelevant features.

While the authors do not explicitly mention its use for online FS, the incremental nature of the grafting algorithm makes it inherently suitable for such scenarios, and it is also the study which laid the foundation for modern streaming FS models.

Building on their previous work, Perkins and Theiler (2003) extended the grafting technique to the online FS problem, where features are presented sequentially over time rather than all at once. Moreover, when a new feature is added, they recommend reapplying the gradient test to all previously seen features (including those already rejected), optionally capping how many times a feature is reconsidered.

As time went on, other dynamic FS approaches were presented as well. For example, Zhou et al. (2006) which introduced alpha investing, evaluates features sequentially as they arrive, deciding their inclusion based on dynamic thresholds adjusted by either information theoretic (information investing) or statistical (α investing) criteria. This approach focuses on scalability and controlling overfitting, making it efficient for large feature spaces. A detailed overview of the alpha investing algorithm is shown in Fig. 10.

Most modern dynamic FS algorithms now focus on using a concept known as redundancy and relevancy for its operations. Wu et al. (2013) introduced an Online FS (OFS) model, which uses these two concepts. Relevant features are those that provide valuable information for predicting the target variable, while redundant features are those that do not contribute new information because their values are highly correlated with other features in the dataset. Two algorithms, Online Streaming Feature Selection (OSFS) and its enhanced version Fast-OSFS, were developed in Wu et al. (2013). OSFS uses conditional independence tests to assess relevance and redundancy. These algorithms have demonstrated superior prediction accuracy compared to existing methods such as grafting and alpha investing.

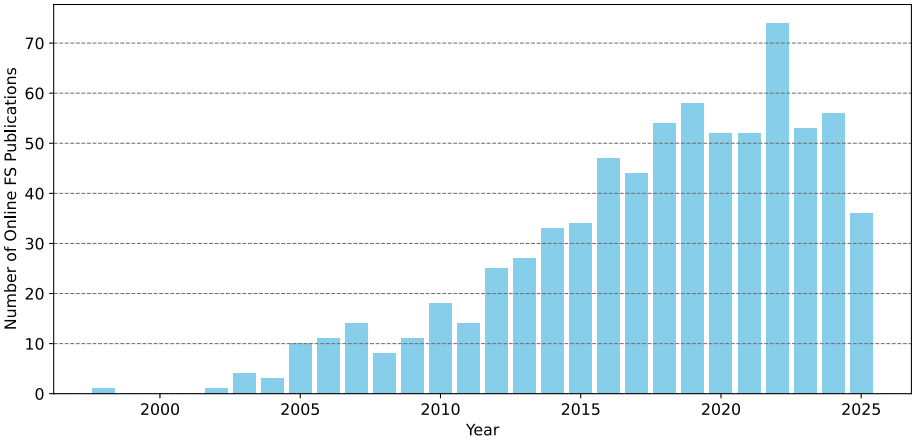

Fig. 8 Number of publications on online FS over the years—Represents the number of papers which include the exact phrases of incremental feature selection, online feature selection, or streaming feature selection, as found through the Scopus database

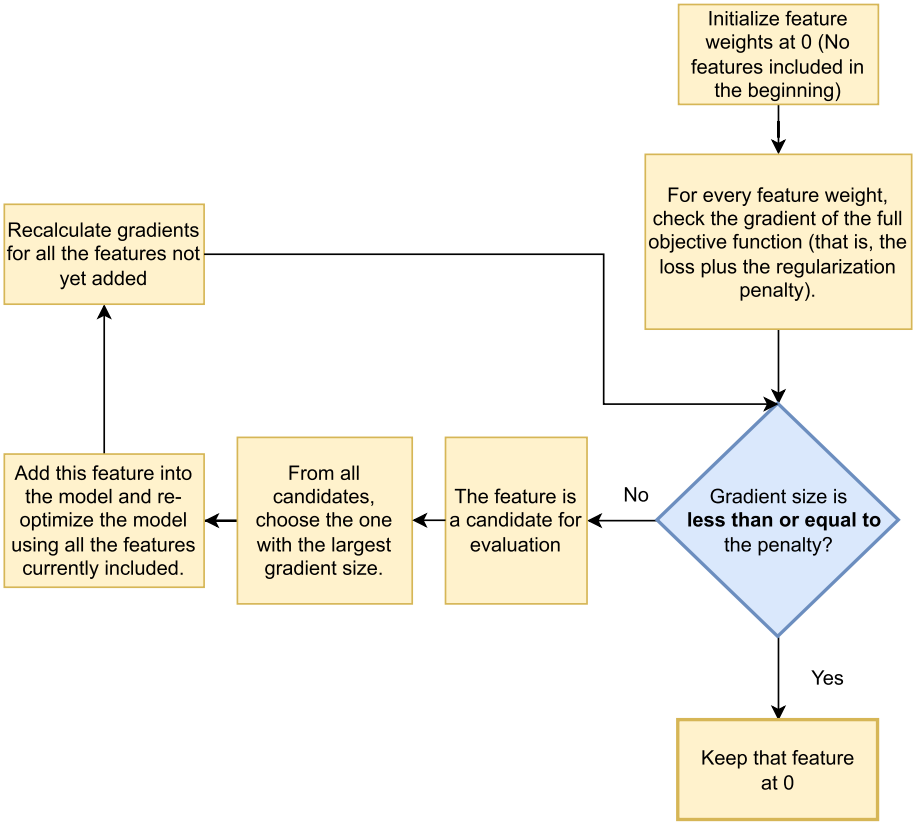

Fig. 9 Overview of grafting algorithm (Perkins et al. 2003)

However, OSFS algorithm can become computationally expensive when the number of selected features grows large, as it re-examines each feature in the current selected set.

The Scalable and Accurate Online Approach for Feature Selection (SAOLA) (Yu et al. 2016b) is another model which is based on the principles of relevancy and redundancy, but is additionally designed to efficiently address the challenges of extremely high dimensional datasets in big data analytics. SAOLA evaluates the relevance of incoming features to the target variable using Mutual Information (MI) and Fisher's Z-test, and discards irrelevant ones. Then it performs pairwise redundancy checks against each feature in the current selected set. SAOLA uses efficient pairwise comparisons instead of exhaustive subset evaluations, making it highly scalable and computationally efficient, especially for datasets with millions of dimensions. However, the algorithm's reliance on pairwise comparisons can result in an approximate rather than the optimal feature subset, potentially overlooking complex feature interactions. Despite these limitations, SAOLA is well suited for big data applications.

The Online Feature Selection for Streaming Features Using Self Adaption Sliding Window Sampling (OSFASW) (You et al. 2019) algorithm introduces an efficient approach to online FS by combining self adaptive sliding window sampling and redundancy, relevancy analysis through conditional independence tests (null-conditional independence

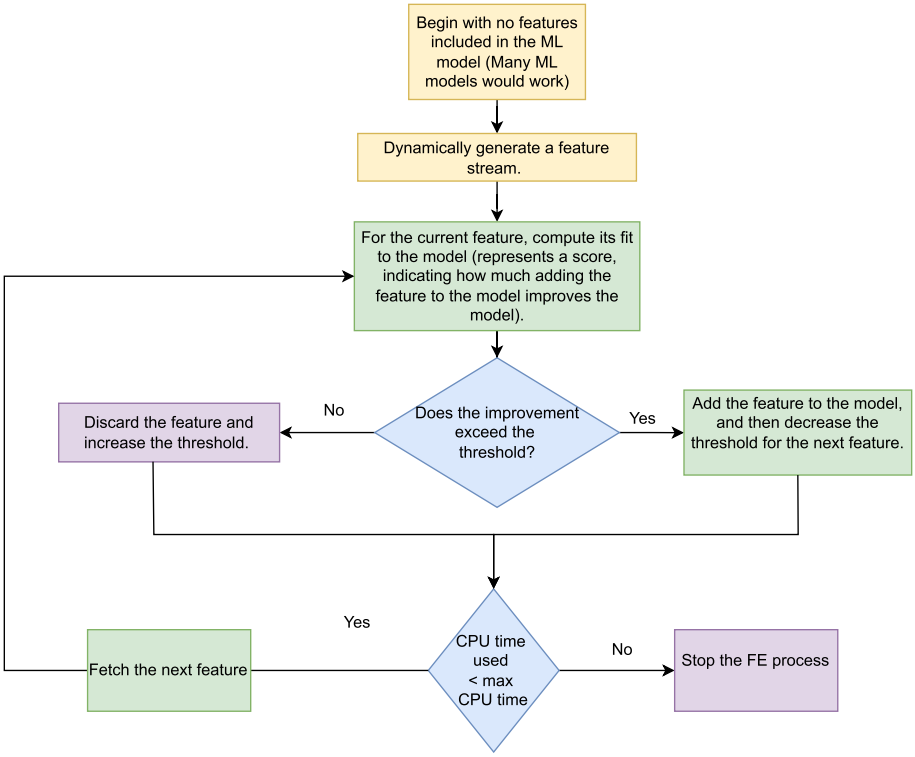

Fig. 10 Overview of alpha investing algorithm (Zhou et al. 2006)

for relevance; conditional independence for redundancy). Compared to previous methods such as alpha investing, OSFS, and SAOLA, OSFASW addresses significant limitations: it improves prediction accuracy and reduces the number of selected features. Furthermore, it manages runtime better than OSFS (but not Alpha-investing/SAOLA). Alpha investing fails to handle redundancy, while OSFS suffers from exponential runtime increases, and SAOLA struggles to optimize the relevance threshold, often selecting too many features. OSFASW overcomes these by using the sliding window to prevent feature overflow, making it suitable for real time applications. While the algorithm achieves high accuracy and efficiency, its computational complexity can rise with a large number of relevant features, and appropriately setting the parameter values is crucial. Nevertheless, OSFASW's represents a significant advancement in online FS. A descriptive overview of the OSFASW algorithm is provided in Fig. 11.

Another unique approach to online FS is found in Wang et al. (2015). In this paper, the authors propose the Online Group Feature Selection (OGFS) algorithm. It selects features dynamically as they arrive in groups, with intra-group discriminative features being selected through spectral analysis and inter-group redundancy elimination happening through sparse regression. Unlike other methods, OGFS uniquely integrates group structure in online FS. The OGFS algorithm often outperformed other OFS methods, such as OSFS and grafting, in terms of accuracy. Furthermore, since features arriving in groups could be more related to each other than those arriving individually, OGFS addresses a major drawback of the previ-

ous algorithms by taking the group structure into account. OFS-Density (Zhou et al. 2019a) is another online streaming FS method using a novel density-based neighborhood rough set approach, which adaptively determines neighbors and selects features with high correlation and dependency as well as low redundancy using a fuzzy equal constraint. Unlike other methods such as grafting, SAOLA or OSFS, it requires no prior parameter specification. While computationally more intensive, it demonstrates superior predictive accuracy on various datasets and avoids issues faced by faster algorithms such as alpha investing, which may underselect features. However, the method's sensitivity to data distribution and increased running time due to the fuzzy constraint remain challenges. OFS-A3M (Zhou et al. 2019b) is another OFS algorithm which can function without the need for domain knowledge, just as OFS-Density does. However, it differs from OFS-Density by using a dynamic gap neighborhood relation instead of fixed neighborhood density, allowing it to adapt to varying instance distributions. Furthermore, OFS-A3M incorporates additional criteria for feature evaluation.

While there are exceptions, the majority of the novel models focus on a relevance and redundancy calculation to determine the online features to select. The OSFSMI and OSFSMI-k methods (Rahmaninia and Moradi 2018) are two such methods, which use MI to incrementally evaluate the relevance and redundancy of features in streaming data. After assessing the relevance of each newly arrived feature, redundancy among features in the selected set is evaluated by computing an effectiveness score, which is a linear combination of relevance and redundancy values. Features with lower effectiveness compared to the newly added feature are iteratively removed. This process involves recalculating the effec-

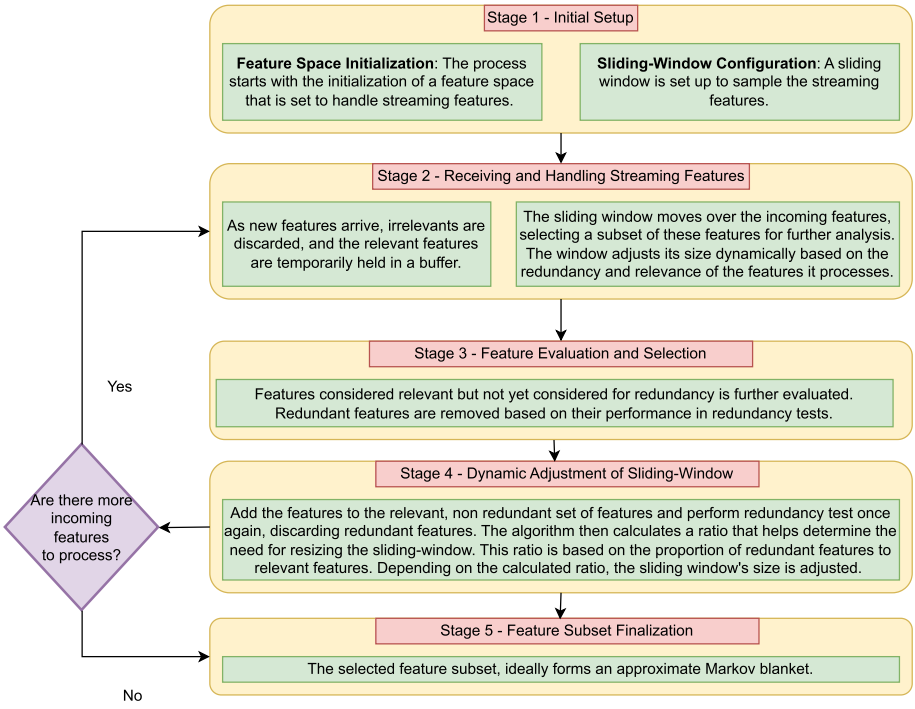

Fig. 11 Overview of OSFASW algorithm (You et al. 2019)

tiveness values of all selected features whenever a feature is added or removed and continues until no more ineffective features remain in the selected set. The OSFSMI-k variant adapts this process to maintain a fixed number of features (k) in the selected set by removing the feature with the lowest effectiveness whenever a new relevant feature arrives after the set reaches size k. Compared to methods such as SAOLA, OSFSMI has higher classification accuracy and more effective feature elimination. Additionally, OSFSMI achieves higher accuracy than Alpha-investing, and it also eliminates redundant features. The proposed methods are scalable due to their reliance on selected feature set size rather than the total number of features and do not require user defined parameters. Furthermore, as OSFSMI and OSFSMI-k evaluate all the existing features (the ones already given in the dataset as well), it is more advantageous than other methods which solely evaluate the newly arriving variables. This is because features in the existing dataset may become less important in light of new arrivals.

One of the major drawbacks observed in most FS algorithms using relevancy and redundancy is the fact that, if the feature is not relevant, then it is discarded. However, there can be instances where, individually the feature is not very well associated with the target, but when combined with other features, it is. This is precisely what Streaming Feature Selection considering Feature Interaction (SFS-FI) aims to address (Zhou et al. 2021a). The algorithm uses an interaction gain metric to measure the relevance and interaction degree of newly arriving features with an already selected subset, dynamically updating the feature set while discarding redundant ones. It evaluates two-way and three-way interactions, which are computationally feasible. Compared to methods such as grafting, alpha investing, OSFS, Fast-OSFS, SAOLA, OFS-A3M and OFS-Density, SFS-FI achieves superior predictive accuracy. However, it is computationally more intensive than SAOLA, Fast-OSFS, Alpha-investing and OSFS.

While SDS-FI provides many advantages, it does not consider the situation when features arrive as a group rather than individual features. In such a case, it is likely that features within a group are closely related to each other and would perform well when considered with each other, rather than the features already selected. As such, to address this problem, Online Group Streaming Feature Selection considering Feature Interaction (OGSFS-FI) (Zhou et al. 2021c) is introduced. The algorithm is designed to handle the dynamic nature of features arriving in a streaming manner, either individually or in groups. Within each incoming group of features, the algorithm identifies pairs of features that exhibit significant interaction using MI. An interaction is quantified by checking whether the combined MI of a feature pair with the target variable exceeds the sum of their individual MI values. Features that enhance the information about the target variable, either individually or through interaction with others, are retained in the selected subset, while others are discarded. After processing the features within a group, the algorithm applies the elastic net regularization method to select features across groups. The elastic net promotes the grouping effect, which tends to select strongly correlated features together, ensuring that inter-group interactions are considered. Compared to other FS algorithms such as alpha investing, OSFS, SAOLA, and OGFS, OGSFS-FI demonstrates superior predictive accuracy. However, it does incurs higher computational costs and is sensitive to hyperparameter tuning.

While the new models gradually improved to address the key issues existing in the original models, one issue still remained. This is the prevalence of missing data. As such, the Latent Factor Analysis Based Online Sparse Streaming Feature Selection Algorithm

(LOSSA) found in Wu et al. (2022) can be taken into account. This is an algorithm designed to address the challenges of online FS in scenarios where streaming features arrive sequentially with missing data. Unlike traditional online streaming FS algorithms, which assume a complete dataset and dynamically expanding feature space, LOSSA incorporates Latent Factor Analysis (LFA) to pre-estimate missing data in sparse streaming features before conducting FS. This approach is crucial because many real-world applications involve incomplete and dynamically generated data, rendering conventional methods ineffective. LOSSA operates in two sequential phases. In the first phase, it handles missing data. Incoming features are buffered into a sparse matrix, where missing values are estimated through low rank matrix approximation. In the second phase, LOSSA conducts FS through relevance and redundancy analyses. The G^2 statistic is used for this. Features that fail either of these tests are discarded. The algorithm is compatible with existing OSFS approaches, enhancing their ability to handle sparse data without significantly altering their mechanisms. Despite its effectiveness in improving FS accuracy in the presence of missing data, LOSSA has certain limitations, including the need for hyperparameter tuning, and additional computational overhead in the LFA preprocessing phase. However, these challenges can be mitigated through parallelization and adaptive parameter selection.

This section reviewed the historical development of online FS methods, emphasizing how modern publications have addressed the limitations inherent in earlier approaches. In recent years, there has been a significant surge in the literature concerning existing online FS techniques. The availability of a vast number of papers highlights the increasing interest and advancements in this area. While this paper has reviewed a majority of the significant contributions in this field, covering the entirety of the available literature is beyond its scope. Readers seeking a broader understanding of the developments in online FS are encouraged to refer to Zaman et al. (2022) and Hu et al. (2018) for an in depth exploration of the topic.

4.1.1 Online FS—publicly available code bases and datasets for streaming features

In the field of online FS, the availability of algorithm implementations plays an important role in advancing research and facilitating practical applications. By collecting and reviewing publicly available codes of existing online FS algorithms, this paper provides a valuable resource for researchers aiming to compare, evaluate, and build upon current methods. This repository is also important for interpretability and reproducibility in computational experiments. Hence, the authors of this paper in Table 7 have provided the links to the code bases of the online FS algorithms (for streaming features) which are publicly available. It is also important to note that LOFS (Yu et al. 2016a) and OSFS (Hu et al. 2018) are two toolboxes that provide implementations of various online FS algorithms.

In addition, a set of benchmarking datasets is also essential for the successful validation of a novel algorithm. Hence, we have compiled Table 8 which provides details of some of the most common datasets used in validating online FS algorithms. Notably, all the datasets listed in this table are classification based. A thorough examination of online FS algorithms published in peer-reviewed journals over the past five years revealed that none of them used regression datasets, highlighting a significant gap in the current literature.

Table 7 Publicly available codes of existing online FS algorithms (for streaming features)

Algorithm	GitHub Link	References	Year(s)
Online feature streaming selection method	https://github.com/pnkx/Online-feature-streaming-selection-method	Yang et al. (2024)	2024
RHDOFS	https://github.com/SCU-Cloud/RHFS	Luo et al. (2023)	2023
OUFSDFC	https://github.com/XuyangAbert/OUFSDFC	Yan et al. (2023)	2023
OSSFS-DD	https://github.com/doodzhou/OSFS	Zhou et al. (2022b)	2022
OSFS-RS	https://github.com/doodzhou/OSFS	Zhou et al. (2022a)	2022
OGSFS	https://github.com/SunY-H/OGSFS	Xu et al. (2022)	2022
OGSFS-FI	https://github.com/doodzhou/OSFS	Zhou et al. (2021c)	2021
SFS-FI	https://github.com/doodzhou/OSFS	Zhou et al. (2021a)	2021
OSFS-ET	https://github.com/doodzhou/OSFS	Zhou et al. (2021b)	2021
zhhu19a	https://github.com/doodzhou/OSFS	Zhou et al. (2019b)	2019
OFS-Density	https://github.com/doodzhou/OSFS	Zhou et al. (2019a)	2019
K-OFSD	https://github.com/doodzhou/OSFS	Zhou et al. (2017)	2017
SAOLA	https://github.com/kuiy/LOFS	Yu et al. (2016b)	2016
Group SAOLA	https://github.com/kuiy/LOFS	Yu et al. (2016b)	2016
OSFS	https://github.com/kuiy/LOFS	Wu et al. (2013)	2013
Fast-OSFS	https://github.com/kuiy/LOFS	Wu et al. (2013)	2013
Alpha Investing	https://github.com/kuiy/LOFS	Zhou et al. (2006)	2006

4.2 Dynamic FS—incoming data points

Dynamic FS for streaming data addresses the challenge of identifying relevant features in real time as data arrives continuously over time. Unlike static FS, which assumes a fixed dataset, dynamic approaches adapt to evolving data distributions, ensuring that the selected features remain relevant and predictive.

The Adaptive Feature Selection with Binary Masking (AFS-BM), found in Turali et al. (2024), is one such adaptive general purpose FS model. The AFS-BM algorithm achieves dynamic FS through a binary mask that iteratively refines the active set of features during model training. The binary mask, represented as a vector of 1 s and 0 s, determines which features are included (1) or excluded (0) at each step. During each iteration, AFS-BM applies the binary mask to the feature set using the Hadamard product, creating a modified dataset. This dataset is used to train the model while keeping the mask fixed, optimizing model parameters to minimize a loss function such as mean squared error or cross-entropy. In the subsequent step, the algorithm evaluates the contribution of individual features by assessing its effect on the loss. If the removal of a feature causes the loss to increase by less than a predefined threshold, the algorithm treats the feature as irrelevant and the corresponding mask entry is updated to 0, allowing the feature to be pruned from the set. The process of masking, loss evaluation, and parameter optimization continues iteratively until the mask stabilizes, meaning no further features are excluded over a set number of iterations. Although the method is motivated by online and non-stationary scenarios, implementing it in real-time streaming contexts is challenging due to computational demands, and this remains an area for future work. Moreover, the iterative nature of the algorithm and the need to evaluate feature contributions make it computationally demanding for very large datasets,

Table 8 Commonly used benchmarking datasets for online FS (streaming features)—Classification Datasets

Dataset name	Instances	Features	Subject area	References
Dorothea	1950	100,000	Biology	Zhou et al. (2006), Yu et al. (2016b), Wu et al. (2013), You et al. (2019)
Gisette	13,500	5000	Computer Science	Zhou et al. (2006), Zhou et al. (2021c)
Arcene	900	10,000	Health and Medicine	Zhou et al. (2006), Rahmaninia and Moradi (2018), Zhou et al. (2019b), Zhou et al. (2021c), Wu et al. (2013), You et al. (2019), Zhou et al. (2019a), Zhou et al. (2021a)
Leukemia	72	7130	Health and Medicine	Yu et al. (2016b), Wang et al. (2015), Zhou et al. (2021c), Wu et al. (2013), You et al. (2019), Zhou et al. (2021a)
Colon	62	2001	Health and Medicine	Wang et al. (2015), Zhou et al. (2019b), Zhou et al. (2021c), Wu et al. (2013), Zhou et al. (2019a), Zhou et al. (2021a), Wu et al. (2022)
SPECTF Heart	267	44	Health and Medicine	Zhou et al. (2006), Wang et al. (2015)
Breast Cancer Wisconsin (Diagnostic) (WDBC)	569	30	Health and Medicine	Zhou et al. (2006), Wang et al. (2015), Zhou et al. (2019b), Zhou et al. (2019a)
Dexter	2600	20,000	Other	Yu et al. (2016b), Zhou et al. (2006), Rahmaninia and Moradi (2018), Wu et al. (2013), You et al. (2019)
Madelon	2600	501	Other	Yu et al. (2016b), Zhou et al. (2006), Rahmaninia and Moradi (2018), Zhou et al. (2021c), Wu et al. (2013), You et al. (2019), Zhao et al. (2023), Zhou et al. (2021a)
Ionosphere	351	34	Physics and Chemistry	Zhou et al. (2006), Wang et al. (2015), You et al. (2019), Zhou et al. (2019a)

and its reliance on hyperparameters requires careful tuning for optimal performance. Furthermore, this method does not consider the interaction between features.

As such, the model in Yelleti et al. (2024) can be taken into account, as it specifically addresses the interaction between features in its methodology by using a wrapper based approach. The algorithm proposed in this paper dynamically updates FS for new data streams by integrating a hybrid optimization algorithm with novelty detection and a feedback mechanism. The FS process begins by initializing parameters and training an initial wrapper model using the BBPSO-L+TNS algorithm, which combines the global exploration capability of Bare Bones Particle Swarm Optimization guided by logistic distribution (BBPSO-L) with the local refinement provided by a Threshold-based Neighborhood Search (TNS). When a new stream arrives, it is partitioned into data islands and processed in parallel, with the wrapper model trained incrementally using the evolved solutions from previous streams as the starting population. Novelty detection metrics, such as modified frobenius norm and cosine similarity, assess differences between consecutive streams, and uses a threshold to identify meaningful changes. If the novelty score exceeds this threshold, incremental retraining is triggered. Otherwise, a feedback mechanism evaluates the current wrapper's performance and skips retraining if sufficient. As such, this also reduces the computational cost, an advantage not displayed in AFS-BM.

While the previous papers focused on general FS for optimizing models, a unique approach which focuses on selecting features specifically for outlier detection can be found in Heigl et al. (2021). This study discusses the Unsupervised Feature Selection for Streaming Outlier Detection (UFSSOD) model. The process begins by assigning scores to features based on their contributions to anomaly detection, using a modified Loda algorithm to compute these scores. Feature scores are updated incrementally through a moving average, with updates happening with each incoming data point, allowing the model to adapt to evolving data trends effectively. The features are then clustered using a one dimensional clustering algorithm. Features from the cluster having the highest average contribution scores are selected, and used for outlier detection.

Apart from the algorithms discussed in detail, numerous other methods have been developed to address specific use cases—for example, handling partially labeled incoming data (Shu et al. 2022), working with both categorical and numerical attributes (Yang et al. 2022a; Wang et al. 2022a; Xu and Liu 2024), and processing datasets where individual feature values are represented as intervals (Feng and Zhang 2026). However, a comprehensive review of all such approaches lies beyond the scope of this paper. Interested readers are referred to Zaman et al. (2022) for a more extensive overview. Notably, a review of journal publications from the past five years reveals that nearly all proposed methods have focused exclusively on the classification setting, highlighting a significant gap in the literature with respect to regression based approaches.

As for publicly available code bases, AFS-BM (Turali et al. 2024) and Incremental Feature Selection by Sample Selection and Feature-based Accelerator (IFS-SSFA) (Yang et al. 2022b) have made their codes accessible. Furthermore, commonly found benchmarking datasets used in online FS for streaming data can be found in Table 9. Note that all the datasets found in this table are classification datasets.

Table 10 provides a comparative overview of the incremental FS methods, highlighting differences in learning setting, data type and supervision level.

4.3 Adaptive feature extraction

Adaptive FExt is a critical component in modern data driven applications, especially in streaming data environments where real time insights are required. It involves the continuous identification and transformation of relevant features from data streams, adapting to evolving data patterns and domain specific requirements. This approach is distinct from traditional FExt methods, which typically rely on static datasets and predefined features, by allowing the model to maintain relevance and accuracy in dynamic scenarios. This section introduces the most popular adaptive FExt algorithms, highlighting their evolution and significance in improving data representation for ML models.

The basis for most incremental FExt algorithms can be traced back to PCA (Jolliffe 1986). While PCA in itself is used for dimensionality reduction and its original algorithm did not account for dynamic data environments, there are many variations of it, such as incremental PCA (IPCA) (Zhao et al. 2006; Tan and 2016; Agrawal and Karmeshu, 2008; Ozawa et al. 2008b, 2009, 2010a) that are uniquely suited for this aspect. Hall et al. (1998) is one of the earlier studies that laid important groundwork for IPCA. The paper proposes an algorithm designed to efficiently update eigenspace models as new observations are introduced without requiring the storage or reprocessing of all prior data. The algorithm

Table 9 Publicly available benchmarking datasets for dynamic FS—streaming data

Dataset name	Instances	Features	Subject area	References
Gisette	13,500	5000	Computer Science	Hochma and Last (2025), Zaman et al. (2024), Zhao et al. (2024a)
Spambase	4601	57	Computer Science	Xu and Ye (2025), Hochma and Last (2025)
Letter Recognition	20,000	16	Computer Science	Shu et al. (2023), Zhao et al. (2024b)
Madelon	4400	500	Other	Hochma and Last (2025), Zaman et al. (2024), Zhao et al. (2024a)
Mushroom	8124	22	Biology	Liang et al. (2024), Zhao et al. (2024a)
Iris	150	4	Biology	Wang et al. (2025a), Shu et al. (2023)
QSAR Androgen Receptor	1687	1024	Physics and Chemistry	Zhao et al. (2024b), Liang et al. (2024)
Ionosphere	351	34	Physics and Chemistry	Zhao et al. (2024b), Shu et al. (2023)
Arcene	900	10,000	Health and Medicine	Zhao et al. (2024b), Zaman et al. (2024)
Adult	48,842	14	Social Science	Zhao et al. (2024b), Liang et al. (2024)

Table 10 Comparative overview of incremental FS methods

Method	Learning setting	Data type	Incremental/online	Supervision
AFS-BM (Turali et al. 2024)	Full Dataset	High-dimensional numeric	No	Supervised
BBPSO-L+TNS (Yelleti et al. 2024)	Batch based	Numeric	Yes	Supervised
UFSSOD (Heigl et al. 2021)	Instance based	Numeric	Yes	Unsupervised
IG-IFS (Shu et al. 2022)	Instance and Batch based	Hybrid (numeric, categorical, missing)	Yes	Semi-supervised
Yang et al. (2022a)	Batch based	Hybrid (numeric, categorical, missing, set valued)	Yes	Supervised
GIFSA (Wang et al. 2022a)	Instance and Batch based	Hybrid (numeric, categorical)	Yes	Supervised
IFS-KNCMNI (Xu and Liu 2024)	Batch based	Hybrid (numeric, categorical, missing)	Yes	Supervised
Feng and Zhang (2026)	Batch based	Interval-valued numeric data with fuzzy decision values	Yes	Supervised

incrementally updates the mean, eigenvectors, and eigenvalues of the eigenspace, addressing a limitation of earlier methods which only catered to static datasets. This approach also dynamically adjusts the dimensionality of the eigenspace. The advantage of this method include its ability to adapt the eigenspace incrementally without having to reconstruct the model from scratch. Hall et al. (2000) takes this approach one step further by allowing entire sets of observations to be processed in a single step. The approach reduces the number of incremental updates needed, enhancing computational efficiency.

In the approaches mentioned thus far, the eigenspace was augmented based on whether the norm of the residual vector (representing the new data point) exceeded a fixed threshold or not. The problems associated with this approach was addressed in Ozawa et al. (2004) where the author proposed to make the decision using an accumulation ratio. The accumulation ratio evaluates the proportion of total variance explained by the current eigenspace after incorporating the new data. If the variance contribution from the new data is significant enough (based on the accumulation ratio), the eigenspace is updated. This method dynamically adjusts the augmentation process, ensuring that the eigenspace grows only when necessary to capture meaningful variance, avoiding redundant dimensions. This was once again updated in Ozawa et al. (2008a) where the author updated the approach to handle groups of data rather than individual observations. Since then, there have been a variety of IPCA algorithms which address the drawbacks of the previous papers (Zhao et al. 2006; Rosas-Arias et al. 2019; Qahtan et al. 2015; Faithfull and Kuncheva 2012; Jain et al. 2016; Fujiwara et al. 2020; Zhang and Teng 2020; Chen and Lin 1994). While IPCA algorithms laid the foundation for incremental FExt algorithms, they face challenges with computational efficiency and time demands, particularly when updating the eigenspace for each incoming data stream. This makes them unsuitable for real time applications that demand immediate responses.

The drawback of the IPCA algorithms is mainly due to the fact that the covariance matrix has to be recalculated for each incoming data stream. Thus, covariance free methods have been introduced to the IPCA process in order to mitigate these problems (Wei et al. 2018; Weng et al. 2003; Zeng and Li 2013). However, this method too has a drawback. This is the inability to accurately estimate the final eigenvectors due to error propagation in the incremental computation process. Furthermore, all forms of IPCA algorithms can only form linear combinations of the original features, thus failing to capture the non linear interactions between variables. Hence, Kernal-IPCA algorithms have been proposed (Takeuchi et al. 2007; Chin and Suter 2007; Zhao et al. 2019; Hallgren and Northrop 2018; Ana Joseph et al. 2014, 2015; Ghashami et al. 2015; Tokumoto and Ozawa 2012; Ozawa et al. 2010b; Tokumoto and Ozawa 2011; Kim et al. 2003, 2004). Yet, these too suffer from the problem of choosing the best kernel type, and higher computational cost.

Recursive PCA (RPCA) (Ebrahimzadeh Hassanabadi et al. 2020) is another form of PCA which has been introduced to continuously updates the principal components (PCs) as new data arrives. RPCA works by incrementally updating the covariance matrix using a weighted combination of the previous covariance and the new data. It then recalculates the PCs using methods such as numerical eigenvalue decomposition, IPCA, or matrix perturbation techniques. Some advanced variants, such as Kalman Filter based RPCA, further enhance performance by handling noisy data. Apart from this, there are also other incremental learning algorithms such as Rank-One Incremental PCA (ROIPCA) (Mitz and Shkolnisky 2023) and incremental linear discriminant analysis (Ana Joseph et al. 2021; Aliyari Ghassabeh and Abrishami Moghaddam 2012; Ohta and Ozawa 2011, 2013; Aa et al. 2012; Ghassabeh and Moghaddam 2007; Ohta and Ozawa 2009; Ozawa and Ohta 2011), as well as incremental (Zeng and Li 2014) partial least squares analysis which are quite useful.

Discrete Cosine Transformation (DCT) (Ahmed et al. 1974) is another algorithm which can be used for FExt. While originally used for static FExt (Nassih et al. 2019; Dabbaghchian et al. 2010) it is now modified in order to handle dynamic data environments. DCT is efficient, adaptable to new data without recomputation, and faster than PCA for streaming

data due to its simplicity and fast implementation (Aydogdu and Ekinci 2020). Hence, it is most suitable for real-time tasks (Wijaya et al. 2016; Cho et al. 2013; Rashidi et al. 2012). A modern example which uses DCT for incremental FExt is found in Aydogdu and Ekinci (2020).

Apart from general purpose incremental FExt algorithms (which works for any ML algorithm), there are also methods specifically optimized to generate features tailored for use with particular learning algorithms. For example, the Incremental Gradient Descent Decision Boundary Feature Extraction for Gaussian Maximum Likelihood Classifier (IGDDBFE) (Woo and Lee 2019) is designed explicitly for use with the Gaussian Maximum Likelihood (GML) classifier, while the Incremental Gradient Descent Decision Boundary Feature Extraction for Neural Networks (IGDDBFE) is designed explicitly for use with the Neural Networks (Woo and Lee 2018), and both incrementally constructs a discriminative feature space. Similarly, the Dual-Generalized Discriminative Common Vectors (Dual-GDCV) method (Diaz-Chito et al. 2018b) is a supervised subspace learning approach that updates the feature space in both incremental and decremental settings while enhancing class separability.

There exists a vast body of literature on incremental FExt, and a comprehensive discussion of all such methods is beyond the scope of this paper. For readers interested in a broader overview, we refer to the review Diaz-Chito et al. (2018a), which surveys incremental FExt methods. Additionally, for an empirical comparison of incremental FExt methods, the readers are referred to Zeng et al. (2015). This paper, although not limited to it, primarily focuses on methods, where the extracted features can be traced back to the original feature space, thereby maintaining interpretability, as exemplified by techniques such as PCA.

In terms of existing code bases, from the papers reviewed in this section, only Hallgren and Northrop (2018) had provided a publicly accessible code. It can be found in IKPCA. Additionally, commonly used benchmarking datasets for dynamic FExt are given in Table 11. However, it should be noted that these are all classification datasets as none of the referred papers had benchamrked their algorithm through the use of regression models.

4.4 The need for meta learning in adaptive AFE

Traditionally, mapping a particular algorithm to a problem has been done by an expert through the use of trial and error. While this is computationally expensive, it works well in the case of offline data. However, in the case of online data, more intelligent methods are required for the recommendation of the specific FE steps needed for each given problem. This is because it is inefficient to try all possible combinations of FE steps whenever there is a concept drift, and hence meta learning can be used to recommend the best FE steps for the newly obtained data. It can also be used to inform the user of a concept drift, if any were to occur, and to filter out the historical data which is most similar to the new data, ensuring that the models are trained only on the relevant observations (Ali et al. 2020).

Meta features for data which is dependent on time should capture key statistical, structural, and dynamic properties that summarize the underlying patterns of the series. These include general statistics such as length, variance, skewness and kurtosis, as well as measures of turning points and step changes. Furthermore, measures of nonlinearity and chaos (as found using the largest Lyapunov exponent) should also be taken into account. Frequency domain features, derived from the Fourier transform, quantify cyclic and seasonal behavior,

Table 11 Commonly used benchmarking datasets—dynamic FExt

Dataset name	Instances	Features	Subject area	References
Spambase	4601	57	CS	Ozawa et al. (2008a), Ana Joseph et al. (2015), Ohta and Ozawa (2013)
Optical-digit	5620	64	CS	Aydogdu and Ekinci (2020), Ana Joseph et al. (2015)
Isolet	7797	617	CS	Ozawa et al. (2008a), Faithfull and Kuncheva (2012), Ana Joseph et al. (2014)
Segmentation	2310	19	CS	Takeuchi et al. (2007), Ohta and Ozawa (2009, 2013), Ozawa et al. (2004, 2008b, 2009, 2010b), Ozawa and Ohta (2011)
Vowel	528	10	CS	Takeuchi et al. (2007), Ozawa et al. (2004, 2009, 2010b), Tokumoto and Ozawa (2012), Agrawal and Karmeshu (2008), Ozawa et al. (2008b)
Waveform	5000	21	P&C	Aydogdu and Ekinci (2020), Ana Joseph et al. (2015), Ohta and Ozawa (2013)
Wine	178	13	P&C	Tan and (2016), Agrawal and Karmeshu, (2008), Ohta and Ozawa (2009), Kim et al. (2003), Kim et al. (2004)
Sonar	208	60	P&C	Ozawa et al. (2004), Faithfull and Kuncheva (2012), Ohta and Ozawa (2009)
Adult	48,842	14	SS	Ozawa et al. (2008a, 2010b), Ana Joseph et al. (2014, 2015)
Zoo	101	16	Bi	Tan and (2016)

P&C Physics and Chemistry, *CS* Computer Science, *SS* Social Science, *Bi* Biology

as well as the strength of their nature. This is obtained through dominant frequencies and power spectrum peaks, and should be considered. Finally, a variety of diversity measures are also to be calculated (Lemke and Gabrys 2010). Collectively, these meta features can enable AFE by providing a compact, descriptive representation of time series data. These features have shown to be of great use in selecting the best forecasting algorithm to employ, based on the characteristics of the data to model. Thus, it should also be possible to use this for the purpose of deciding which FE algorithm to consider based on the incoming data.

As adaptive AFE algorithms have not yet been formally established, researchers aiming to develop such systems may find meta features to be a valuable resource for guiding adaptivity. This is because meta features can reduce the need for exhaustive search and manual tuning, thereby improving the efficiency and responsiveness of AFE systems in dynamic environments.

4.5 Impact of AFE on model fitting during streaming data scenarios

Throughout this review, it was observed that AFE can be of great use during the ML model building phase. However, there is also a significant drawback which must be addressed when using AFE. In the context of AFE applied to streaming data (if such a system were to be created), the dynamic nature of the feature space impacts not only the FE stage but also the subsequent ML modeling and deployment stages (as already observed in Fig. 1). As new features emerge and existing features lose relevance due to changes in data patterns, the original ML model, trained on a static feature set, may no longer be optimal for predicting outcomes. This evolving feature space necessitates continuous adaptation of

the ML model to maintain its predictive performance. A static model may fail to capture critical new relationships in the data or may rely on outdated and less informative features, leading to degraded accuracy over time. Hence, while AFE aids an ML model to improve on its performance over time, it also complicates the modeling stage. For instance, in the case of streaming data, if a previously important feature becomes obsolete and must be discarded, the model's decision-making process, which previously relied on that feature, must be restructured. Similarly, when new relevant features are introduced, the model must adapt to incorporate these into its predictions, further complicating the model building and maintenance process. To address this challenge, mechanisms such as incremental learning or model retraining on updated feature sets can be implemented. These approaches ensure that the ML model evolves alongside the feature space, allowing it to adapt to new data dynamics while maintaining alignment with the most relevant features extracted by the AFE model. Adapting the ML model to the evolving feature space is a complex and extensive research area. However, it lies beyond the scope of this paper and will not be discussed further.

5 Explainable and interpretable Machine Learning

This section examines the growing importance of explainability and interpretability in ML, particularly in domains where ethical and regulatory considerations are important. While advancements in ML algorithms have improved performance, they have also increased complexity, creating challenges for their deployment in highly regulated environments. This section explores key concepts such as interpretability and explainability, as well as the perceived trade off between model performance and its interpretability. Additionally, the role of AFE models to balance interpretability with performance is discussed.

In ML, interpretability is commonly defined as the extent to which humans are able to comprehend the rationale for a model's output (Abdullah et al. 2021). Traditional statistical models often naturally exhibit high interpretability. However, modern ML practices sometimes rely on complex, black box models that sacrifice interpretability for performance. This introduces the need for explainability in ML, which involves an explanation for the decision making processes of these black box models using techniques such as post hoc explanations (Marcinkevics and Vogt 2023). These could involve using methods such as partial dependency plot to explain how the predicted values of a model are influenced by changes in a specific variable. Explainability in ML models is quite difficult to achieve though, as explanations provided might not accurately reflect how these black box models function. Thus, relying on explainability can be quite dangerous, and hence it is better to develop a more interpretable model, if the option is available (Gosiewska et al. 2021). See Fig. 12 for a more clear understanding of the difference between explainability and interpretability.

The common belief is that there is an inevitable trade off between a model's accuracy and its interpretability (Chen et al. 2023). However, this does not universally apply. In many cases, interpretable models can achieve performance comparable to their black box counterparts (Chakraborty et al. 2021; Gosiewska et al. 2021).

Interpretable ML models play a crucial role in critical sectors such as health care by allowing for better auditing capabilities (Garbin and Marques 2022). From a legal standpoint, these features align with regulations such as the EU general data protection regula-

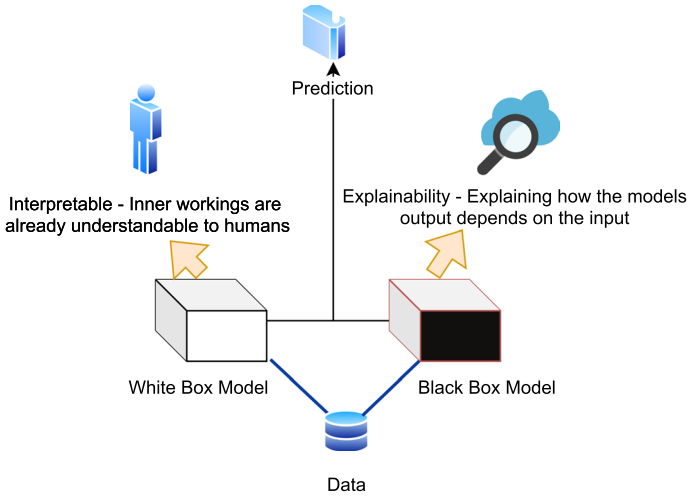

Fig. 12 Understanding Interpretability and Explainability

tion, which supports individuals' rights to explanations of algorithmic decisions (Gosiewska et al. 2021). While this regulation does not outright prohibit black box models and the explanation right is not legally binding, the ethical and social importance of providing explanations is critical.

While a lot of focus on interpretability and explainability is placed on ML models themselves, the FE, as an integral part of ML model development, is critical in achieving these goals. Interpretability in AFE can be assessed using several proxy measures that attempt to capture how easily a human can understand the resulting features and models. These measures generally fall into five broad categories.

- **Traceability of generated features:** One important criterion is whether engineered features can be traced back to the original input variables by understanding the transformations which are applied (Eldeeb et al. 2022). In some AFE systems, the sequence of transformations applied to generate a feature is explicitly defined through the feature names, allowing practitioners to understand how the features were derived (Horn et al. 2020).
- **Feature contribution analysis:** Interpretability can also be assessed through methods that quantify the contribution of features to model predictions. Techniques such as Shapley value-based explanations measure the marginal contribution of each feature to the prediction, allowing practitioners to identify which engineered features influence model behaviour the most. These approaches provide a quantitative way of understanding the importance and impact of generated features within the predictive model.
- **Improved performance of simpler models:** Another way to assess interpretability is by examining whether the engineered feature space enables simpler models to achieve strong predictive performance. In general, most black box models such as deep neural networks tend to have good performance due to better data representations. This is be-

cause they inherently perform FE during training, automatically transforming and combining input variables into richer internal representations that capture useful patterns and interactions in the data. As such, a highly sophisticated AFE model could aid even glass box models such as linear regression to achieve a performance which is comparative with that of the black box model, thus improving interpretability in the pipeline without having to sacrifice the performance. By coming up with similarly strong feature representations through a transparent process, AFE bridges the gap between accuracy and interpretability, offering the benefits of black box performance while retaining model transparency. This was observed even in the previous section through models such as SAFE and INAFEN (Liu et al. 2024a; Gosiewska et al. 2021).

- **Model complexity metrics:** Interpretability is often evaluated through measures of model complexity. For example, in tree-based models, structural properties such as tree depth or number of nodes can be used as proxies for interpretability, with smaller trees generally considered easier to understand. This principle has also been applied in prior AFE systems, where interpretability is assessed by examining whether the generated features enable simpler decision trees to be learned (Roshanski et al. 2023; Markovitch and Rosenstein 2002). Despite these efforts, there is currently no universally accepted quantitative metric for interpretability comparable to commonly used predictive performance metrics such as RMSE or accuracy. Furthermore, comparisons across studies remain difficult because evaluations are typically conducted on different datasets and experimental setups, while the proxy measures used to quantify interpretability also vary across methods. For instance, while the previous studies considered decision trees, Gosiewska et al. (2021) states that interpretability can be measured as the the inverse of the number of model parameters, where for logistic regression the parameter count corresponds to the number of coefficients (including the intercept). Consequently, this highlights the need for a standardized and universally applicable interpretability metric to enable consistent evaluation and comparison across approaches.
- **Domain specific interpretability:** In domain-specific applications, interpretability can also be enhanced when the engineered features have clear semantic meaning within the application domain. Instead of producing abstract mathematical transformations, AFE systems can incorporate domain knowledge to generate features that correspond to meaningful physical or biological relationships. For instance, in the biopharmaceutical industry, AFE can be combined with bio-process knowledge, to generate biophysical features which have meaning (Khuat et al. 2024).

6 Discussions and conclusions

This comprehensive review of AFE methods has emphasized their indispensable role in enhancing ML workflows by addressing the limitations of traditional FE. By examining both adaptive FE algorithms and non adaptive AFE models, the analysis shows the dynamic nature of data environments and the necessity for AFE methods that adapt to changing data characteristics.

The findings reveal that non adaptive AFE models, while robust in stationary environments, often fall short in dynamic scenarios where data evolves over time. Adaptive FE methods, however, demonstrate significant potential to maintain and improve performance

by adjusting to new data inputs and distributions. This adaptability is critical for applications across various fields, including finance, healthcare, and real time system monitoring.

In addition, the evolution of FE from a manual, expertise driven process to an automated, adaptive mechanism has significant implications for the future of ML. While current models provide a strong foundation, there is a need for further research aimed at improving the adaptability and interpretability of these systems. The current research gaps which exist have been summarized in the sections given below, for the benefit of the research community.

6.1 Research gaps in AFE systems

The field of AFE faces several challenges that need to be addressed to enhance its applicability and effectiveness. The current literature reveals several research gaps that can be broadly categorized into four main groups: algorithmic and computational limitations related to feature generation and search strategies, limitations that restrict the applicability of AFE systems across different learning algorithms and tasks, data representation challenges associated with handling heterogeneous feature types, and evaluation limitations arising from inconsistent benchmarking and evaluation practices.

6.1.1 Algorithmic and computational limitations related to feature generation and search strategies

- **Computational inefficiency:** Exhaustive generate-and-select approaches (e.g., AutoFeat (Horn et al. 2020)) create a comprehensive pool of features, which causes scalability and performance issues due to selecting from an excessively large set of generated features (due to the simultaneous use of all operations).
- **Interaction modeling issues:** The iterative construction method builds features incrementally by assessing the incremental improvement each new feature brings. Although more scalable, this method can prematurely discard potentially valuable features, by overlooking complex feature interactions. Most contemporary AFE models such as Cognito (Khurana et al. 2016) employ this iterative approach. Additionally, methods such as SAFE (Gosiewska et al. 2021) cannot model feature interactions during their AFE process. Thus far, only a few methods have considered the interaction of features in the FG process (Overman et al. 2024; Azim et al. 2023).
- **Operator explosion and uncontrolled feature growth:** Many AFE systems rely on the exhaustive application of unary and binary operators over the available feature set. While this strategy maximizes coverage of potential transformations, it leads to a rapid combinatorial explosion in the number of generated features. As a consequence, such systems must resort to aggressive pruning or early selection heuristics, or terminate the exploration after only one or very few iterations. This often prevents the discovery of higher-order or multi-step feature compositions, even when such combinations may be necessary to capture complex relationships in the data. As a result, the effective search depth of many AFE frameworks remains shallow, limiting their ability to explore richer feature interactions. When effective feature selection is not tightly integrated into the generation process, the resulting combinatorial growth quickly becomes computation-

ally prohibitive, leading to excessive memory usage and runtime overhead. Moreover, the inclusion of large numbers of weak or redundant derived features significantly degrades interpretability, making it difficult to attribute model behavior to meaningful feature transformations and undermining the transparency.

6.1.2 Limitations that restrict the applicability of AFE systems across different learning algorithms and tasks

- **Algorithm dependency:** AFE models such as BigFeat and BigFeat-FE (Eldeeb et al. 2022; Eldeeb and Elshawi 2024) try to mitigate both these drawbacks (of the generate and select as well as iterative construction method) using tree based models to identify the features that could potentially become useful in the future, while also avoiding an exhaustive exploration of the feature space. Yet, these models strongly rely on decision trees for this purpose, and this may not work when considering other modeling techniques. This is because the features identified as informative by decision-tree splits are tailored to how tree-based models partition the input space. However, downstream models such as linear regression, SVMs, or neural networks may rely on entirely different types of feature interactions and representations. As a result, features optimized for decision trees may fail to capture the relationships that are most relevant for these alternative models, limiting the transferability and general utility of such AFE approaches.
- **Generalizability:** Autofeat is restricted to single table datasets, making it unsuitable for scenarios with multiple data sources (Eldeeb et al. 2022). Similarly, models such as aKDFE are computationally expensive, domain restricted, cannot handle varying input types and are validated only on limited experiments, reducing their generalizability (Björneld and Löwe 2024).
- **Applicability:** Even advanced frameworks such as BigFeat and BigFeat-FE are designed for classification only and fail on regression tasks, leaving an important research gap (Eldeeb et al. 2022; Eldeeb and Elshawi 2024). Models such as INAFEN are limited to specific modeling techniques, such as logistic regression, which constrains their applicability (Liu et al. 2024a). Similarly, the recently proposed LbR framework is also confined to classification scenarios, as it relies on label-based feature relationships (Wang et al. 2020).

6.1.3 Data representation challenges associated with handling heterogeneous feature types

Many AFE frameworks rely on simplistic encodings of categorical variables, such as ordinal encoding (Eldeeb et al. 2022), before applying generic unary and binary operators. While this enables a uniform treatment of all inputs, it often leads to the application of mathematically invalid transformations (e.g., square root, logarithm, or arithmetic operations) on nominal categories. Such operations impose artificial numerical relationships that do not exist in the original data, resulting in spurious or misleading derived features. These inappropriate transformations can often degrade model performance and compromise interpretability.

6.1.4 Inconsistent benchmarking and evaluation practices

A significant limitation of the current AFE literature is the lack of standardized and consistent evaluation protocols. Many studies benchmark their proposed methods on selectively chosen datasets where performance gains are most pronounced, making cross-method comparison difficult and potentially biased. As different AFE frameworks are evaluated on disjoint sets of datasets, with varying preprocessing pipelines, models, and metrics, it remains unclear whether reported improvements generalize beyond specific experimental settings. To date, there is no comprehensive, large-scale benchmarking study that evaluates multiple AFE methods under a unified experimental protocol, which hinders objective assessment of progress in the field and obscures the true strengths and limitations of competing approaches.

6.2 Research gaps in dynamic FE algorithms

Current adaptive FE algorithms exhibit several research gaps that need to be addressed as well.

- **Trade-off between computational efficiency and accuracy:** Current adaptive FE models face limitations in real-time processing. While some methods handle data streams efficiently, they often fail to scale with large feature sets, leading to a compromise between computational efficiency (You et al. 2019) and predictive accuracy (Wu et al. 2013; Yu et al. 2016b). Robust models that can process real-time streams (for high dimensional data) without sacrificing performance are still lacking.
- **Inability to handle streaming data and features simultaneously:** Existing online FS methods typically address one of two scenarios: Fixed data points with streaming features (Wu et al. 2013, 2022; Yu et al. 2016b; You et al. 2019; Wang et al. 2015; Rahmaninia and Moradi 2018; Zhou et al. 2021a, c), or constant feature sets with streaming data (Turali et al. 2024; Yelleti et al. 2024; Heigl et al. 2021). Few approaches manage both evolving data points and feature streams simultaneously, which is problematic since, in practice, both can vary at the same time.
- **Lack of adaptive AFE systems:** Adaptive learning algorithms usually assume incoming data is already preprocessed, but when this is not the case, model performance is negatively impacted (Zliobaite and Gabrys 2014). Most existing AFE systems are limited to fixed datasets (Eldeeb et al. 2022; Eldeeb and Elshawi 2024; Khurana et al. 2016). Similarly, several adaptive FE algorithms have been proposed to address challenges arising from streaming and evolving data. However, these approaches are often limited to transforming the input feature space rather than performing on the fly construction and reconstruction of new, potentially complex feature sets, which is the core of true AFE. There is no comprehensive AFE framework that automates the entire FE pipeline while also adapting to concept drift, evolving features, and real time streams. This highlights a key opportunity for dynamic, streaming aware AFE.
- **Lack of adaptive FG/construction methods:** In streaming environments, generated features may lose relevance as data evolves. Adaptive FG should dynamically construct new features to capture shifting data characteristics. While terms such as adaptive FG (Jian et al. 2022) and incremental FG (Sun and Ahluwalia 1995) exist, they often diverge

from the traditional definition of FG (the systematic construction of new features by applying mathematical or logical operations to one or more original input variables, with the goal of enhancing the representativeness of the feature set). The lack of true adaptive FG/FC methods in streaming contexts remains a significant gap to be addressed before fully adaptive AFE systems can be developed.

6.3 Contributions of the review paper

While various review papers that examine AFE methods (Prado and Digiampietri 2020; Mumuni and Mumuni 2024) exist, the scope of Prado and Digiampietri (2020) is limited to discussing only three models: AutoLearn, Cognito, and ExploreKit, while Mumuni and Mumuni (2024) mainly looks into automated preprocessing steps, such as data cleaning, labeling, missing data imputation, categorical data encoding, and data augmentation. Furthermore, Prado and Digiampietri (2020) primarily focuses on automated FS rather than the broader scope of AFE. Additionally, neither of the papers address the aspects of interpretability in ML models or the challenges posed by dynamic data scenarios. In contrast, this paper addresses these gaps by exploring AFE methods that prioritize interpretability, as well as FE algorithms that have the capacity to adapt to dynamic data, providing a more holistic review. Additionally, this paper provides an in depth review of over 25 AFE models, offering a more detailed analysis of the field.

Additionally, recent times have seen an explosion of comprehensive studies for AFE methods across different domains. As such, there are also many benchmarking datasets available for evaluating the effectiveness of an AFE model. This could be overwhelming for new researchers entering into the field of ML. Hence, this study compiled some of the most widely used benchmarking datasets as well as code bases for individuals in the research community who would have the need for it.

To conclude, although prior studies have reviewed incremental FS methods (Zaman et al. 2022) and incremental FExt algorithms (Diaz-Chito et al. 2018a), this study is the first to provide a comprehensive review of incremental FE approaches as a whole, offering readers a more integrated understanding of the field.

6.4 Concluding remarks

While AFE models offer promise in handling evolving data scenarios, much research is needed to address these gaps. By developing solutions that effectively tackle all the issues outlined above, the field of AFE can make major progress toward realizing the full potential of autonomous ML modeling in modern data driven applications.

Acknowledgements This research was supported under the Australian Research Council's Industrial Transformation Research Program (ITRP) funding scheme (project number IH210100051). The ARC Digital Bioprocess Development Hub is a collaboration between The University of Melbourne, University of Technology Sydney, RMIT University, CSL Innovation Pty Ltd, Cytiva (Global Life Science Solutions Australia Pty Ltd) and Patheon Biologics Australia Pty Ltd.

Author contributions D.S. conducted the literature review, and prepared the original draft of the manuscript. T.K, K.M, and B.G provided supervision, guidance, and critical revisions to improve the manuscript. All authors read and approved the final version.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. This research was supported under the Australian Research Council's Industrial Transformation Research Program (ITRP) funding scheme (project number IH210100051).

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Abdullah TAA, Zahid MSM, Ali W (2021) A review of interpretable ML in healthcare: taxonomy, applications, challenges, and future directions. Symmetry 13(12):2439. https://doi.org/10.3390/sym13122439

Agrawal R, Karmeshu (2008) Perturbation scheme for online learning of features: incremental principal component analysis. Pattern Recogn 41(5):1452–1460. https://doi.org/10.1016/j.patcog.2007.10.002

Ahmed N, Natarajan T, Rao K (1974) Discrete cosine transform. IEEE Trans Comput C 23(1):90–93. https://doi.org/10.1109/t-c.1974.223784

Ahmed S, Ghosh KK, Singh PK et al (2020) Hybrid of harmony search algorithm and ring theory-based evolutionary algorithm for feature selection. IEEE Access 8:102629–102645. https://doi.org/10.1109/access.2020.2999093

Ali AR, Budka M, Gabrys B (2020) A review of meta-level learning in the context of multi-component, multi-level evolving prediction systems. https://doi.org/10.48550/ARXIV.2007.10818

Aliyari Ghassabeh Y, Abrishami Moghaddam H (2012) Adaptive linear discriminant analysis for online feature extraction. Mach Vis Appl 24(4):777–794. https://doi.org/10.1007/s00138-012-0439-z

An J, Kim IS, Kim KJ et al (2024) Efficacy of automated machine learning models and feature engineering for diagnosis of equivocal appendicitis using clinical and computed tomography findings. Sci Rep. https://doi.org/10.1038/s41598-024-72889-9

Anak Joseph A, Ozawa S (2014) A fast incremental kernel principal component analysis for data streams. In: 2014 International joint conference on neural networks (IJCNN). IEEE, pp 3135–3142, https://doi.org/10.1109/ijcnn.2014.6889940

Anak Joseph A, Anak Pelias Pog EI, Chin KL et al (2021) Online person identification based on multitask learning. Int J Integr Eng. https://doi.org/10.30880/ijie.2021.13.02.014

Anderson MR, Antenucci D, Cafarella MJ (2016) Runtime support for human-in-the-loop feature engineering system. IEEE Data Eng Bull 39:62–84

Anuragi A, Sisodia DS, Pachori RB (2024) Mitigating the curse of dimensionality using feature projection techniques on electroencephalography datasets: an empirical review. Artif Intell Rev. https://doi.org/10.1007/s10462-024-10711-8

Arun Kumar R, Vijay Franklin J, Koppula N (2022) A comprehensive survey on metaheuristic algorithm for feature selection techniques. Mater Today Proc 64:435–441. https://doi.org/10.1016/j.matpr.2022.04.803

Asir D, Appavu S, Jebamalar E (2016) Literature review on feature selection methods for high-dimensional data. Int J Comput Appl 136(1):9–17. https://doi.org/10.5120/ijca2016908317

Aydogdu O, Ekinci M (2020) An approach for streaming data feature extraction based on discrete cosine transform and particle swarm optimization. Symmetry 12(2):299. https://doi.org/10.3390/sym12020299

Azim E, Wang D, Liu K et al (2023) Feature interaction aware automated data representation transformation. https://doi.org/10.48550/ARXIV.2309.17011

Barandas M, Folgado D, Fernandes L et al (2020) Tsfel: Time series feature extraction library. SoftwareX 11:100456. https://doi.org/10.1016/j.softx.2020.100456

Bensusan H, Giraud-Carrier C, Kennedy C (2000) A higher-order approach to meta-learning. In: Proceedings of the ECML'2000 workshop on meta-learning: building automatic advice strategies for model selection and method combination. ECML'2000, pp 109–117

Björneld O, Löwe W (2024) Real-world validation of a framework for automated knowledge driven feature engineering inspired by medical domain experts. Inf Med Unlocked 49:101532. https://doi.org/10.1016/j.imu.2024.101532

Bonidia RP, Santos APA, de Almeida BLS et al (2022) Bioautoml: automated feature engineering and meta-learning to predict noncoding RNAs in bacteria. Brief Bioinform. https://doi.org/10.1093/bib/bbac218

Bouadi M, Alavi A, Benbernou S et al (2024) KRAFT: leveraging knowledge graphs for interpretable feature generation. Springer, Singapore, pp 384–399. https://doi.org/10.1007/978-981-96-0567-5_27

Bouadi M, Alavi A, Benbernou S et al (2025a) Dante: hybrid AI system for context-aware interpretable feature engineering. In: Companion of the 2025 international conference on management of data, SIGMOD/PODS '25. ACM, New York, pp 51–54. https://doi.org/10.1145/3722212.3725092

Bouadi M, Alavi A, Benbernou S et al (2025b) Synergizing large language models and knowledge-based reasoning for interpretable feature engineering. In: Proceedings of the ACM on web conference 2025, WWW '25. ACM, New York, pp 2606–2620. https://doi.org/10.1145/3696410.3714720

Brazdil P (1998) Data transformation and model selection by experimentation and meta-learning. In: Proceedings of the ECML-98 workshop on upgrading learning to meta-level: model selection and data transformation, pp 11–17

Chakraborty D, Basagaoglu H, Winterle J (2021) Interpretable vs. noninterpretable machine learning models for data-driven hydro-climatological process modeling. Expert Syst Appl 170:114498. https://doi.org/10.1016/j.eswa.2020.114498

Chen H, Lin RW (1994) An on-line unsupervised learning machine for adaptive feature extraction. IEEE Trans Circuits Syst II, Analog Digit Signal Process 41(2):87–98. https://doi.org/10.1109/82.281840

Chen X, Qiao B, Zhang W et al (2019) Neural feature search: A neural architecture for automated feature engineering. In: 2019 IEEE international conference on data mining (ICDM). IEEE, pp 71–80. https://doi.org/10.1109/icdm.2019.00017

Chen Z, Xiao F, Guo F et al (2023) Interpretable machine learning for building energy management: a state-of-the-art review. Adv Appl Energy 9:100123. https://doi.org/10.1016/j.adapen.2023.100123

Chin TJ, Suter D (2007) Incremental kernel principal component analysis. IEEE Trans Image Process 16(6):1662–1674. https://doi.org/10.1109/tip.2007.896668

Cho H, Han S, Hwang SY (2013) Design of an efficient real-time algorithm using reduced feature dimension for recognition of speed limit signs. Sci World J. https://doi.org/10.1155/2013/135614

Christ M, Braun N, Neuffer J et al (2018) Time series feature extraction on basis of scalable hypothesis tests (tsfresh—a python package). Neurocomputing 307:72–77. https://doi.org/10.1016/j.neucom.2018.03.067

Dabbaghchian S, Ghaemmaghami MP, Aghagolzadeh A (2010) Feature extraction using discrete cosine transform and discrimination power analysis with a face recognition technology. Pattern Recogn 43(4):1431–1440. https://doi.org/10.1016/j.patcog.2009.11.001

De Brabandere A, Op De Beéck T, Hendrickx K et al (2022) Tsfuse: automated feature construction for multiple time series data. Mach Learn 113(8):5001–5056. https://doi.org/10.1007/s10994-021-06096-2

de Winter C, Frasincar F, de Peuter B et al (2025) Automated feature engineering for automated machine learning. Knowl Based Syst 321:113671. https://doi.org/10.1016/j.knosys.2025.113671

Diaz-Chito K, Ferri FJ, Hernández-Sabaté A (2018) An overview of incremental feature extraction methods based on linear subspaces. Knowl Based Syst 145:219–235. https://doi.org/10.1016/j.knosys.2018.01.020

Diaz-Chito K, Martínez del Rincón J, Rusiñol M et al (2018) Feature extraction by using dual-generalized discriminative common vectors. J Math Imaging Vis 61(3):331–351. https://doi.org/10.1007/s10851-018-0837-6

Dor O, Reich Y (2012) Strengthening learning algorithms by feature discovery. Inf Sci 189:176–190. https://doi.org/10.1016/j.ins.2011.11.039

Dumitrescu E, Hué S, Hurlin C et al (2022) Machine learning for credit scoring: improving logistic regression with non-linear decision-tree effects. Eur J Oper Res 297(3):1178–1192. https://doi.org/10.1016/j.ejor.2021.06.053

Ebrahimzadeh Hassanabadi M, Heidarpour A, Eftekhar Azam S et al (2020) Recursive principal component analysis for model order reduction with application in nonlinear Bayesian filtering. Comput Methods Appl Mech Eng 371:113334. https://doi.org/10.1016/j.cma.2020.113334

Eldeeb H, Elshawi R (2024) Empowering machine learning with scalable feature engineering and interpretable AUTOML. IEEE Trans Artif Intell. https://doi.org/10.1109/tai.2024.3400752

Eldeeb H, Amashukeli S, El Shawi R (2021) An empirical analysis of integrating feature extraction to automated machine learning pipeline. Springer, Cham, pp 336–344. https://doi.org/10.1007/978-3-030-68799-1_24

Eldeeb H, Amashukeli S, ElShawi R (2022) Bigfeat: Scalable and interpretable automated feature engineering framework. In: 2022 IEEE international conference on big data (Big Data). IEEE. https://doi.org/10.1109/bigdata55660.2022.10020768
Faithfull W, Kuncheva L (2012) PCA feature extraction for change detection in multidimensional unlabelled streaming data. In: Proceedings of the 21st international conference on pattern recognition (ICPR2012)
Fan W, Zhong E, Peng J et al (2010) Generalized and heuristic-free feature construction for improved accuracy. In: Proceedings of the 2010 SIAM international conference on data mining. society for industrial and applied mathematics, pp 629–640. https://doi.org/10.1137/1.9781611972801.55
Fatima A, Ali Khan F, Raza A et al (2018) Automated feature synthesis from relational database for data science related problems. In: 2018 International conference on frontiers of information technology (FIT). IEEE, pp 71–75. https://doi.org/10.1109/fit.2018.00020
Feng Z, Zhang X (2026) Supervised incremental feature selection using regularization vector for dynamic multi-scale interval valued datasets. Pattern Recogn 170:111985. https://doi.org/10.1016/j.patcog.2025.111985
Fujiwara T, Chou JK, Shilpika et al (2020) An incremental dimensionality reduction method for visualizing streaming multidimensional data. IEEE Trans Vis Comput Graph 26(1):418–428. https://doi.org/10.1109/tvcg.2019.2934433
Gada M, Haria Z, Mankad A et al (2021) Automated feature engineering and hyperparameter optimization for machine learning. In: 2021 7th International conference on advanced computing and communication systems (ICACCS). IEEE. https://doi.org/10.1109/icaccs51430.2021.9441668
Garbin C, Marques O (2022) Assessing methods and tools to improve reporting, increase transparency, and reduce failures in machine learning applications in health care. Radiology. https://doi.org/10.1148/ryai.210127
Gaudel R, Sebag M (2010) Feature selection as a one-player game. In: Proceedings of the 27th international conference on international conference on machine learning, ICML'10. Omnipress, Madison, pp 359–366
Ghashami M, Perry D, Phillips JM (2015) Streaming kernel principal component analysis. https://doi.org/10.48550/ARXIV.1512.05059
Ghassabeh YA, Moghaddam HA (2007) A new incremental optimal feature extraction method for on-line applications. Springer, Berlin, pp 399–410. https://doi.org/10.1007/978-3-540-74260-9_36
Gosiewska A, Kozak A, Biecek P (2021) Simpler is better: lifting interpretability-performance trade-off via automated feature engineering. Decis Support Syst 150:113556. https://doi.org/10.1016/j.dss.2021.113556
Habehh H, Gohel S (2021) Machine learning in healthcare. Curr Genomics 22(4):291–300. https://doi.org/10.2174/1389202922666210705124359
Hall PM, Marshall D, Martin RR (1998) Incremental eigenanalysis for classification. In: Proceedings of the British machine vision conference 1998. British Machine Vision Association, BMVC 1998, pp 29.1–29.10. https://doi.org/10.5244/c.12.29
Hall P, Marshall D, Martin R (2000) Merging and splitting eigenspace models. IEEE Trans Pattern Anal Mach Intell 22(9):1042–1049. https://doi.org/10.1109/34.877525
Hallgren F, Northrop P (2018) Incremental kernel PCA and the Nyström method. https://doi.org/10.48550/ARXIV.1802.00043
Hang CN, Yu PD, Chen S et al (2023) MEGA: machine learning-enhanced graph analytics for infodemic risk management. IEEE J Biomed Health Inform 27(12):6100–6111. https://doi.org/10.1109/jbhi.2023.3314632
Harari A, Katz G (2022) Automatic features generation and selection from external sources: a dbpedia use case. Inf Sci 582:398–414. https://doi.org/10.1016/j.ins.2021.09.036
He T, Huang X, Du Y et al (2025) FASTFT: accelerating reinforced feature transformation via advanced exploration strategies. In: 2025 IEEE 41st international conference on data engineering (ICDE). IEEE, pp 4120–4133. https://doi.org/10.1109/icde65448.2025.00307
Heigl M, Weigelt E, Fiala D et al (2021) Unsupervised feature selection for outlier detection on streaming data to enhance network security. Appl Sci 11(24):12073. https://doi.org/10.3390/app112412073
Hochma Y, Last M (2025) Fast online feature selection in streaming data. Mach Learn. https://doi.org/10.1007/s10994-024-06712-x
Hollmann N, Müller S, Hutter F (2023) Large language models for automated data science: introducing Caafe for context-aware automated feature engineering. https://doi.org/10.48550/ARXIV.2305.03403
Horn F, Pack R, Rieger M (2020) The autofeat Python library for automated feature engineering and selection. Springer, Cham, pp 111–120. https://doi.org/10.1007/978-3-030-43823-4_10
Hu X, Zhou P, Li P et al (2018) A survey on online feature selection with streaming features. Front Comput Sci 12(3):479–493. https://doi.org/10.1007/s11704-016-5489-3

Jain R, Xu W (2023) Artificial intelligence based wrapper for high dimensional feature selection. BMC Bioinform. https://doi.org/10.1186/s12859-023-05502-x

Jain P, Jin C, Kakade SM et al (2016) Streaming pca: Matching matrix bernstein and near-optimal finite sample guarantees for Oja's algorithm. In: Feldman V, Rakhlin A, Shamir O (eds) 29th Annual conference on learning theory, proceedings of machine learning research, vol 49. PMLR. Columbia University, New York, pp 1147–1164. https://proceedings.mlr.press/v49/jain16.html

Jian Y, Yi J, Zhang L (2022) Adaptive feature generation for online continual learning from imbalanced data. Springer, Cham, pp 276–289. https://doi.org/10.1007/978-3-031-05933-9_22

Jolliffe IT (1986) Principal component analysis and factor analysis. Springer, New York, pp 115–128. https://doi.org/10.1007/978-1-4757-1904-8_7

Joseph A, Jang YM, Ozawa S et al (2012) Extension of incremental linear discriminant analysis to online feature extraction under nonstationary environments. Springer, Berlin, pp 640–647. https://doi.org/10.1007/978-3-642-34481-7_78

Joseph AA, Tokumoto T, Ozawa S (2015) Online feature extraction based on accelerated kernel principal component analysis for data stream. Evol Syst 7(1):15–27. https://doi.org/10.1007/s12530-015-9131-7

Kamalov F, Sulieman H, Alzaatreh A et al (2025) Mathematical methods in feature selection: a review. Mathematics 13(6):996. https://doi.org/10.3390/math13060996

Kanter JM, Veeramachaneni K (2015) Deep feature synthesis: towards automating data science endeavors. In: 2015 IEEE International conference on data science and advanced analytics (DSAA). IEEE, pp 1–10. https://doi.org/10.1109/dsaa.2015.7344858

Kassarnig V, Wotawa F (2018) An approach to automatically extract predictive properties from nominal attributes in relational databases. In: 2018 IEEE international conference on big data (Big Data). IEEE, pp 4932–4939. https://doi.org/10.1109/bigdata.2018.8622359

Katz G, Shin ECR, Song D (2016) Explorekit: automatic feature generation and selection. In: 2016 IEEE 16th international conference on data mining (ICDM). IEEE. https://doi.org/10.1109/icdm.2016.0123

Kaul A, Maheshwary S, Pudi V (2017) Autolearn—automated feature generation and selection. In: 2017 IEEE international conference on data mining (ICDM). IEEE, pp 217–226. https://doi.org/10.1109/icdm.2017.31

Kedziora DJ, Musial K, Gabrys B (2020) Autonoml: Towards an integrated framework for autonomous machine learning. https://doi.org/10.48550/ARXIV.2012.12600

Khobragade K, Ponde P (2025) Generative adversarial networks in feature engineering: an unsupervised approach. IGI Global, pp 87–116. https://doi.org/10.4018/979-8-3693-6900-5.ch004

Khuat TT, Kedziora DJ, Gabrys B (2023) The roles and modes of human interactions with automated machine learning systems: a critical review and perspectives. Found Trends Hum Comput Interact 17(3–4):195–387. https://doi.org/10.1561/1100000091

Khuat TT, Bassett R, Otte E et al (2024) Applications of machine learning in antibody discovery, process development, manufacturing and formulation: current trends, challenges, and opportunities. Comput Chem Eng 182:108585. https://doi.org/10.1016/j.compchemeng.2024.108585

Khurana U, Turaga D, Samulowitz H et al (2016) Cognito: automated feature engineering for supervised learning. In: 2016 IEEE 16th international conference on data mining workshops (ICDMW). IEEE. https://doi.org/10.1109/icdmw.2016.0190

Khurana U, Samulowitz H, Turaga D (2018) Feature engineering for predictive modeling using reinforcement learning. Proceedings of the AAAI conference on artificial intelligence, vol 32(1). https://doi.org/10.1609/aaai.v32i1.11678

Kim BJ, Kim IK, Kim KB (2004) Feature extraction and classification system for nonlinear and online data. Springer, Berlin, pp 171–180. https://doi.org/10.1007/978-3-540-24775-3_22

Kim BJ, Shim JY, Hwang CH et al (2003) Incremental feature extraction based on empirical kernel map. Springer, Berlin, pp 440–444. https://doi.org/10.1007/978-3-540-39592-8_62

Kumar V (2014) Feature selection: a literature review. Smart Comput Rev. https://doi.org/10.6029/smartcr.2014.03.007

Ladeira LZ, Borro LC, Violato RPV et al (2021) RAAF: resource-aware auto featuring. In: 2021 55th annual conference on information sciences and systems (CISS). IEEE, pp 1–6. https://doi.org/10.1109/ciss50987.2021.9400256

Lam HT, Thiebaut JM, Sinn M et al (2017) One button machine for automating feature engineering in relational databases. https://doi.org/10.48550/ARXIV.1706.00327

Lam HT, Buesser B, Min H et al (2021) Automated data science for relational data. In: 2021 IEEE 37th international conference on data engineering (ICDE). IEEE, pp 2689–2692. https://doi.org/10.1109/icde51399.2021.00305

Lemke C, Gabrys B (2010) Meta-learning for time series forecasting and forecast combination. Neurocomputing 73(10–12):2006–2016. https://doi.org/10.1016/j.neucom.2009.09.020

Lemke C, Budka M, Gabrys B (2015) Metalearning: a survey of trends and technologies. Artif Intell Rev 44(1):117–130. https://doi.org/10.1007/s10462-013-9406-y
Liang B, Liu Y, Lu J et al (2024) A group incremental feature selection based on knowledge granularity under the context of clustering. Int J Mach Learn Cybern 15(9):3647–3670. https://doi.org/10.1007/s13042-024-02113-7
Liu M, Guo C, Xu L (2024a) An interpretable automated feature engineering framework for improving logistic regression. Appl Soft Comput 153:111269. https://doi.org/10.1016/j.asoc.2024.111269
Liu Z, Zhang D, Liu H et al (2024b) Visible-hidden hybrid automatic feature engineering via multi-agent reinforcement learning. Knowl-Based Syst 299:111941. https://doi.org/10.1016/j.knosys.2024.111941
Luo C, Wang S, Li T et al (2023) Rhdofs: A distributed online algorithm towards scalable streaming feature selection. IEEE Trans Parallel Distrib Syst 34(6):1830–1847. https://doi.org/10.1109/tpds.2023.3265974
Luo Y, Wang M, Zhou H et al (2019) Autocross: Automatic feature crossing for tabular data in real-world applications. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, KDD '19, 1936–1945, https://doi.org/10.1145/3292500.3330679
Lu G, Wang H, Yang S et al (2023) Catch: Collaborative feature set search for automated feature engineering. In: Proceedings of the ACM Web Conference 2023. ACM, WWW '23, 1886–1896, https://doi.org/10.1145/3543507.3583527
Mahajan P (2020) Automated feature engineering using kernel functions. In: 2020 International Conference on Artificial Intelligence and Signal Processing (AISP). IEEE, 1–3, https://doi.org/10.1109/aisp48273.2020.9073463
Mahesh B (2020) Machine learning algorithms - a review. International Journal of Science and Research (IJSR) 9(1):381–386. https://doi.org/10.21275/art20203995
Malberg S, Mosca E, Groh G (2024) FELIX: Automatic and Interpretable Feature Engineering Using LLMs, Springer Nature Switzerland, 230–246. https://doi.org/10.1007/978-3-031-70359-1_14
Mamdouh Farghaly H, Abd El-Hafeez T (2023) A high-quality feature selection method based on frequent and correlated items for text classification. Soft Comput 27(16):11259–11274. https://doi.org/10.1007/s00500-023-08587-x
Marcinkevics R, Vogt JE (2023) Interpretable and explainable machine learning: a methods-centric overview with concrete examples. WIREs Data Min Knowl Discov. https://doi.org/10.1002/widm.1493
Markovitch S, Rosenstein D (2002) Feature generation using general constructor functions. Mach Learn 49(1):59–98. https://doi.org/10.1023/a:1014046307775
Miao J, Niu L (2016) A survey on feature selection. Procedia Comput Sci 91:919–926. https://doi.org/10.1016/j.procs.2016.07.111
Mitz R, Shkolnisky Y (2023) Roipca: an online memory-restricted PCA algorithm based on rank-one updates. Inf Inference A J IMA 12(4):2851–2872. https://doi.org/10.1093/imaiai/iaad025
Mumuni A, Mumuni F (2024) Automated data processing and feature engineering for deep learning and big data applications: a survey. J Inf Intell. https://doi.org/10.1016/j.jiixd.2024.01.002
Nam J, Kim K, Oh S et al (2024) Optimized feature generation for tabular data via LLMS with decision tree reasoning. https://doi.org/10.48550/ARXIV.2406.08527
Nargesian F, Samulowitz H, Khurana U et al (2017) Learning feature engineering for classification. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence. International Joint Conferences on Artificial Intelligence Organization, IJCAI-2017, 2529–2535, https://doi.org/10.24963/ijcai.2017/352
Nargesian F, Khurana U, Pedapati T et al (2018) Dataset evolver: an interactive feature engineering notebook. In: Proceedings of the thirty-second AAAI conference on artificial intelligence and thirtieth innovative applications of artificial intelligence conference and eighth AAAI symposium on educational advances in artificial intelligence, AAAI'18/IAAI'18/EAAI'18. AAAI Press, New Orleans
Nassih B, Amine A, Ngadi M et al (2019) DCT and hog feature sets combined with BPNN for efficient face classification. Procedia Comput Sci 148:116–125. https://doi.org/10.1016/j.procs.2019.01.015
Nazareth N, Ramana Reddy YV (2023) Financial applications of machine learning: a literature review. Expert Syst Appl 219:119640. https://doi.org/10.1016/j.eswa.2023.119640
Nguyen NH, Abellán-García J, Lee S et al (2024) From machine learning to semi-empirical formulas for estimating compressive strength of ultra-high performance concrete. Expert Syst Appl 237:121456
Nichols JA, Herbert Chan HW, Baker MAB (2018) Machine learning: applications of artificial intelligence to imaging and diagnosis. Biophys Rev 11(1):111–118. https://doi.org/10.1007/s12551-018-0449-9
Ohta R, Ozawa S (2009) An incremental learning algorithm of recursive fisher linear discriminant. In: 2009 International Joint Conference on Neural Networks. IEEE, 2310–2315, https://doi.org/10.1109/ijcnn.2009.5178963
Ohta R, Ozawa S (2011) An improvement of incremental recursive fisher linear discriminant for online feature extraction. IEEJ Trans Electron Inf Syst 131(7):1368–1376. https://doi.org/10.1541/ieejeiss.131.1368

Ohta R, Ozawa S (2013) An improvement of incremental recursive fisher linear discriminant for online feature extraction. Electronics and Communications in Japan 96(4):29–40. https://doi.org/10.1002/ecj.10430

Ouyang R, Curtarolo S, Ahmetcik E et al (2018) Sisso: a compressed-sensing method for identifying the best low-dimensional descriptor in an immensity of offered candidates. Phys Rev Mater 2(8):083802

Overman T, Klabjan D, Utke J (2024) IIFE: interaction information based automated feature engineering. In: 2024 IEEE international conference on data mining (ICDM). IEEE, pp 803–808. https://doi.org/10.1109/icdm59182.2024.00096

Ozawa S, Ohta R (2011) Incremental recursive fisher linear discriminant for online feature extraction. In: 2011 IEEE workshop on evolving and adaptive intelligent systems (EAIS). IEEE, pp 70–76. https://doi.org/10.1109/eais.2011.5945905

Ozawa S, Pang S, Kasabov N (2004) A modified incremental principal component analysis for on-line learning of feature space and classifier. Springer, Berlin, pp 231–240. https://doi.org/10.1007/978-3-540-28633-2_26

Ozawa S, Pang S, Kasabov N (2008a) Incremental learning of chunk data for online pattern classification systems. IEEE Trans Neural Netw 19(6):1061–1074. https://doi.org/10.1109/tnn.2007.2000059

Ozawa S, Matsumoto K, Pang S et al (2008b) An incremental principal component analysis based on dynamic accumulation ratio. In: 2008 SICE annual conference. IEEE, pp 2471–2475. https://doi.org/10.1109/sice.2008.4655080

Ozawa S, Matsumoto K, Pang S et al (2009) Incremental principal component analysis based on adaptive accumulation ratio. Springer, Berlin, pp 1196–1203. https://doi.org/10.1007/978-3-642-02490-0_145

Ozawa S, Pang S, Kasabov N (2010a). Online feature extraction for evolving intelligent systems. In: Evolving intelligent systems: methodology and applications. Wiley-IEEE Press. https://doi.org/10.1002/9780470569962.ch7

Ozawa S, Takeuchi Y, Abe S (2010b) A Fast Incremental Kernel Principal Component Analysis for Online Feature Extraction, Springer Berlin Heidelberg, 487–497. https://doi.org/10.1007/978-3-642-15246-7_45

Perkins S, Theiler J (2003) Online feature selection using grafting. In: International conference on machine learning. https://api.semanticscholar.org/CorpusID:14212255

Perkins S, Lacker K, Theiler J (2003) Grafting: fast, incremental feature selection by gradient descent in function space. J Mach Learn Res 3:1333–1356

Pfahringer B, Giraud-Carrier C (2000) Meta-learning by landmarking various learning algorithms. In: Proceedings of the 17th international conference on machine learning, pp 743–750

Piramuthu S, Sikora RT (2009) Iterative feature construction for improving inductive learning algorithms. Expert Syst Appl 36(2):3401–3406. https://doi.org/10.1016/j.eswa.2008.02.010

Prado FF, Digiampietri LA (2020) A systematic review of automated feature engineering solutions in machine learning problems. In: XVI Brazilian symposium on information systems, SBSI'20. ACM, New York, pp 1–7. https://doi.org/10.1145/3411564.3411610

Pudjihartono N, Fadason T, Kempa-Liehr AW et al (2022) A review of feature selection methods for machine learning-based disease risk prediction. Front Bioinform. https://doi.org/10.3389/fbinf.2022.927312

Pugliese R, Regondi S, Marini R (2021) Machine learning-based approach: global trends, research directions, and regulatory standpoints. Data Sci Manag 4:19–29. https://doi.org/10.1016/j.dsm.2021.12.002

Qahtan AA, Alharbi B, Wang S et al (2015) A pca-based change detection framework for multidimensional data streams: Change detection in multidimensional data streams. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '15. ACM, New York, pp 935–944. https://doi.org/10.1145/2783258.2783359

Qi D, Zheng W, Wang J (2024) FEATAUG: automatic feature augmentation from one-to-many relationship tables. In: 2024 IEEE 40th international conference on data engineering (ICDE). IEEE, pp 1805–1818. https://doi.org/10.1109/icde60146.2024.00146

Radeev N, Vinogradova K (2025) Semi-automated framework for feature engineering in machine learning and data analysis. In: 2025 IEEE 26th International Conference of Young Professionals in Electron Devices and Materials (EDM). IEEE, 1520–1525, https://doi.org/10.1109/edm65517.2025.11096892

Rahmaninia M, Moradi P (2018) Osfsmi: Online stream feature selection method based on mutual information. Appl Soft Comput 68:733–746. https://doi.org/10.1016/j.asoc.2017.08.034

Rashidi S, Fallah A, Towhidkhah F (2012) Feature extraction based DCT on dynamic signature verification. Sci Iran 19(6):1810–1819. https://doi.org/10.1016/j.scient.2012.05.007

Rawat T, Khemchandani V (2017) Feature engineering (FE) tools and techniques for better classification performance. Int J Innov Eng Technol. https://doi.org/10.21172/ijiet.82.024

Real E, Liang C, So D et al (2020) AutoML-zero: evolving machine learning algorithms from scratch. In: Daumé H III, Singh A (eds) Proceedings of the 37th international conference on machine learning, proceedings of machine learning research, vol 119. PMLR, Virtual, pp 8007–8019. http://proceedings.mlr.press/v119/real20a.html

Remeseiro B, Bolon-Canedo V (2019) A review of feature selection methods in medical applications. Comput Biol Med 112:103375. https://doi.org/10.1016/j.compbiomed.2019.103375
Renault A, Bondu A, Lemaire V et al (2023) Automatic feature engineering for time series classification: evaluation and discussion. In: 2023 International joint conference on neural networks (IJCNN). IEEE, pp 1–10. https://doi.org/10.1109/ijcnn54540.2023.10191074
Rendell LA, Sheshu R, Tcheng DK (1987) Layered concept-learning and dynamically variable bias management. In: International joint conference on artificial intelligence. https://api.semanticscholar.org/CorpusID:10111683
Reyes-Amezcua I, Flores-Araiza D, Ochoa-Ruiz G et al (2022) MACFE: a meta-learning and causality based feature engineering framework, Springer, Cham, pp 52–65. https://doi.org/10.1007/978-3-031-19493-1_5
Roque N, Ram N (2019) tsfeaturex: an r package for automating time series feature extraction. J Open Source Soft 4(37):1279. https://doi.org/10.21105/joss.01279
Rosas-Arias L, Portillo-Portillo J, Hernandez-Suarez A et al (2019) Vehicle counting in video sequences: an incremental subspace learning approach. Sensors 19(13):2848. https://doi.org/10.3390/s19132848
Roshanski I, Kalech M, Rokach L (2023) Automatic feature engineering for learning compact decision trees. Expert Syst Appl 229:120470. https://doi.org/10.1016/j.eswa.2023.120470
Salazar R, Neutatz F, Abedjan Z (2021) Automated feature engineering for algorithmic fairness. Proc VLDB Endow 14(9):1694–1702. https://doi.org/10.14778/3461535.3463474
Salehin I, Islam MS, Saha P et al (2024) AUTOML: a systematic review on automated machine learning with neural architecture search. J Inf Intell 2(1):52–81. https://doi.org/10.1016/j.jiixd.2023.10.002
Salvador MM, Budka M, Gabrys B (2016a) Adapting multicomponent predictive systems using hybrid adaptation strategies with auto-weka in process industry. In: Hutter F, Kotthoff L, Vanschoren J (eds) Proceedings of the workshop on automatic machine learning, proceedings of machine learning research, vol 64. PMLR, New York, pp 48–57. https://proceedings.mlr.press/v64/salvador_adapting_2016.html
Salvador MM, Budka M, Gabrys B (2016b) Towards automatic composition of multicomponent predictive systems. In: Lecture notes in computer science. Springer, Cham, pp 27–39. https://doi.org/10.1007/978-3-319-32034-2_3
Salvador MM, Budka M, Gabrys B (2019) Automatic composition and optimization of multicomponent predictive systems with an extended auto-WEKA. IEEE Trans Autom Sci Eng 16(2):946–959. https://doi.org/10.1109/tase.2018.2876430
Sarker IH (2021) Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput Sci. https://doi.org/10.1007/s42979-021-00815-1
Scriven A, Kedziora DJ, Musial K et al (2023) The technological emergence of AUTOML: a survey of performant software and applications in the context of industry. Found Trends Inf Syst 7(1–2):1–252. https://doi.org/10.1561/2900000030
Shi K, Saad S (2023) Automated feature engineering for AUTOML using genetic algorithms. In: Proceedings of the 20th international conference on security and cryptography. SCITEPRESS - Science and Technology Publications, pp 450–459. https://doi.org/10.5220/0012090400003555
Shi Q, Zhang YL, Li L et al (2020) Safe: Scalable automatic feature engineering framework for industrial tasks. In: 2020 IEEE 36th international conference on data engineering (ICDE). IEEE, pp 1645–1656. https://doi.org/10.1109/icde48307.2020.00146
Shu W, Yan Z, Chen T et al (2022) Information granularity-based incremental feature selection for partially labeled hybrid data. Intell Data Anal 26(1):33–56. https://doi.org/10.3233/ida-205560
Shu W, Chen T, Cao D et al (2023) Incremental feature selection based on uncertainty measure for dynamic interval-valued data. Int J Mach Learn Cybern 15(4):1453–1472. https://doi.org/10.1007/s13042-023-01977-5
Silva Rodrigues E, Lima Martins DM, De Lima Neto FB (2021a) Self-organizing transformations for automatic feature engineering. In: 2021 IEEE Symposium series on computational intelligence (SSCI). IEEE, pp 1–7. https://doi.org/10.1109/ssci50451.2021.9659940
Silva Rodrigues E, Mayr Lima Martins D, de Lima Neto FB (2021b) Automatic feature engineering using self-organizing maps. In: 2021 IEEE Latin American conference on computational intelligence (LA-CCI). IEEE, pp 1–6. https://doi.org/10.1109/la-cci48322.2021.9769788
Sun H, Ahluwalia RS (1995) Feature modification in incremental feature generation. Comput Aided Des 27(8):627–635. https://doi.org/10.1016/0010-4485(95)99799-e
Takeuchi Y, Ozawa S, Abe S (2007) An efficient incremental kernel principal component analysis for online feature selection. In: 2007 International joint conference on neural networks. IEEE, pp 2346–2351. https://doi.org/10.1109/ijcnn.2007.4371325
Tan C, Ji G (2016) Semi-supervised incremental feature extraction algorithm for large-scale data stream. Concurr Comput Pract Exp. https://doi.org/10.1002/cpe.3914
Theng D, Bhoyar KK (2023) Feature selection techniques for machine learning: a survey of more than two decades of research. Knowl Inf Syst 66(3):1575–1637. https://doi.org/10.1007/s10115-023-02010-5

Thiamchoo N, Phukpattaranont P (2022) Evaluation of feature projection techniques in object grasp classification using electromyogram signals from different limb positions. PeerJ Comput Sci 8:e949. https://doi.org/10.7717/peerj-cs.949
Tokumoto T, Ozawa S (2011) A fast incremental kernel principal component analysis for learning stream of data chunks. In: The 2011 International joint conference on neural networks. IEEE. https://doi.org/10.1109/ijcnn.2011.6033599
Tokumoto T, Ozawa S (2012) A property of learning chunk data using incremental kernel principal component analysis. In: 2012 IEEE conference on evolving and adaptive intelligent systems. IEEE, pp 7–10, https://doi.org/10.1109/eais.2012.6232796
Tu R, Kang X, Tan CW et al (2025) Sentences based adversarial attack on AI-generated text detectors. IEEE Trans Big Data. https://doi.org/10.1109/tbdata.2025.3600034
Turali MY, Lorasdagi ME, Kozat SS (2024) Afs-bm: enhancing model performance through adaptive feature selection with binary masking. SIViP 18(11):7571–7582. https://doi.org/10.1007/s11760-024-03411-x
Tzelepi M, Nousi P, Passalis N et al (2022) Representation learning and retrieval. Elsevier, Amsterdam, pp 221–241. https://doi.org/10.1016/b978-0-32-385787-1.00015-4
Velliangiri S, Alagumuthukrishnan S, Thankumar Joseph SI (2019) A review of dimensionality reduction techniques for efficient computation. Procedia Comput Sci 165:104–111. https://doi.org/10.1016/j.procs.2020.01.079
Veloso de Melo V, Banzhaf W (2018) Automatic feature engineering for regression models with machine learning: An evolutionary computation and statistics hybrid. Inf Sci 430–431:287–313. https://doi.org/10.1016/j.ins.2017.11.041
Verma A, Lamsal K, Verma P (2021) An investigation of skill requirements in artificial intelligence and machine learning job advertisements. Ind High Educ 36(1):63–73. https://doi.org/10.1177/0950422221990990
Wang J, Wang M, Li P et al (2015) Online feature selection with group structure analysis. IEEE Trans Knowl Data Eng 27(11):3029–3041. https://doi.org/10.1109/tkde.2015.2441716
Wang M, Ding Z, Pan M (2020) Lbr: A new regression architecture for automated feature engineering. In: 2020 International Conference on Data Mining Workshops (ICDMW). IEEE, 432–439, https://doi.org/10.1109/icdmw51313.2020.00066
Wang F, Wei W, Liang J (2022a) A group incremental approach for feature selection on hybrid data. Soft Comput 26(8):3663–3677. https://doi.org/10.1007/s00500-022-06838-x
Wang Z, Xia L, Yuan H et al (2022b) Principles, research status, and prospects of feature engineering for data-driven building energy prediction: a comprehensive review. J Build Eng 58:105028. https://doi.org/10.1016/j.jobe.2022.105028
Wang K, Wang P, Xu C (2022c) Toward efficient automated feature engineering. https://doi.org/10.48550/ARXIV.2212.13152
Wang D, Fu Y, Liu K et al (2022d) Group-wise reinforcement feature generation for optimal and explainable representation space reconstruction. In: Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining, KDD '22. ACM, pp 1826–1834. https://doi.org/10.1145/3534678.3539278
Wang T, Liu S, Yang B (2025a) A novel method for incremental feature selection with fuzzy ß-covering. Fuzzy Sets Syst 512:109379. https://doi.org/10.1016/j.fss.2025.109379
Wang L, Shi Y, Jin Y et al (2025b) OpenFE++: efficient automated feature generation via feature interaction. Society for Industrial and Applied Mathematics, pp 21–30. https://doi.org/10.1137/1.9781611978520.3
Wei Y, Wang H, Wang S et al (2018) Incremental modelling for compositional data streams. Commun Stat Simul Comput 48(8):2229–2243. https://doi.org/10.1080/03610918.2018.1455870
Weng J, Zhang Y, Hwang WS (2003) Candid covariance-free incremental principal component analysis. IEEE Trans Pattern Anal Mach Intell 25:1034–1040
Wijaya IGPS, Husodo AY, Arimbawa IWA (2016) Real time face recognition using dct coefficients based face descriptor. In: 2016 International conference on informatics and computing (ICIC). IEEE, pp 142–147. https://doi.org/10.1109/iac.2016.7905705
Woo S, Lee C (2018) Incremental feature extraction based on decision boundaries. Pattern Recogn 77:65–74. https://doi.org/10.1016/j.patcog.2017.12.010
Woo S, Lee C (2019) Incremental feature extraction based on Gaussian maximum likelihood. In: 2019 34th International technical conference on circuits/systems, computers and communications (ITC-CSCC). IEEE, pp 1–4. https://doi.org/10.1109/itc-cscc.2019.8793458
Wu X, Yu K, Ding W et al (2013) Online feature selection with streaming features. IEEE Trans Pattern Anal Mach Intell 35(5):1178–1192. https://doi.org/10.1109/tpami.2012.197
Wu D, He Y, Luo X et al (2022) A latent factor analysis-based approach to online sparse streaming feature selection. IEEE Trans Syst Man Cybern Syst 52(11):6744–6758. https://doi.org/10.1109/tsmc.2021.3096065

Xiao M, Wang D, Wu M et al (2024) Traceable group-wise self-optimizing feature transformation learning: a dual optimization perspective. ACM Trans Knowl Discov Data 18(4):1–22. https://doi.org/10.1145/3638059
Xu W, Liu C (2024) Incremental reduction of imbalanced distributed mixed data based on k-nearest neighbor rough set. Int J Approx Reason 172:109218. https://doi.org/10.1016/j.ijar.2024.109218
Xu W, Ye W (2025) Incremental feature selection: parallel approach with local neighborhood rough sets and composite entropy. Pattern Recogn 159:111141. https://doi.org/10.1016/j.patcog.2024.111141
Xu J, Sun Y, Qu K et al (2022) Online group streaming feature selection using entropy-based uncertainty measures for fuzzy neighborhood rough sets. Complex Intell Syst 8(6):5309–5328. https://doi.org/10.1007/s40747-022-00763-0
Yan X, Homaifar A, Sarkar M et al (2023) An online unsupervised streaming features selection through dynamic feature clustering. IEEE Trans Artif Intell 4(5):1281–1292. https://doi.org/10.1109/tai.2022.3196637
Yang Y, Webb GI (2008) Discretization for naive-bayes learning: managing discretization bias and variance. Mach Learn 74(1):39–74. https://doi.org/10.1007/s10994-008-5083-5
Yang Y, Chen D, Zhang X et al (2022a) Covering rough set-based incremental feature selection for mixed decision system. Soft Comput 26(6):2651–2669. https://doi.org/10.1007/s00500-021-06687-0
Yang Y, Chen D, Zhang X et al (2022b) Incremental feature selection by sample selection and feature-based accelerator. Appl Soft Comput 121:108800. https://doi.org/10.1016/j.asoc.2022.108800
Yang J, Wang Z, Wang G et al (2024) OSFS-VAGUE: online streaming feature selection algorithm based on vague set. CAAI Trans Intell Technol. https://doi.org/10.1049/cit2.12327
Yelleti V, Ravi V, Krishna PR (2024) Novelty detection and feedback based online feature subset selection for data streams via parallel hybrid particle swarm optimization algorithm. In: Proceedings of the genetic and evolutionary computation conference companion, GECCO '24 Companion. ACM, pp 227–230. https://doi.org/10.1145/3638530.3654298
Yeung D, Spicer V, Zahedi RP et al (2023) Exploring the variable space of shallow machine learning models for reversed-phase retention time prediction. Comput Struct Biotechnol J 21:2446–2453. https://doi.org/10.1016/j.csbj.2023.02.047
You D, Wu X, Shen L et al (2019) Online feature selection for streaming features using self-adaption sliding-window sampling. IEEE Access 7:16088–16100. https://doi.org/10.1109/access.2019.2894121
Yu K, Ding W, Wu X (2016a) LOFS: a library of online streaming feature selection. Knowl Based Syst 113:1–3. https://doi.org/10.1016/j.knosys.2016.08.026
Yu K, Wu X, Ding W et al (2016b) Scalable and accurate online feature selection for big data. ACM Trans Knowl Discov Data 11(2):1–39. https://doi.org/10.1145/2976744
Zaman EAK, Mohamed A, Ahmad A (2022) Feature selection for online streaming high-dimensional data: A state-of-the-art review. Appl Soft Comput 127:109355. https://doi.org/10.1016/j.asoc.2022.109355
Zaman EAK, Ahmad A, Mohamed A (2024) Adaptive threshold optimisation for online feature selection using dynamic particle swarm optimisation in determining feature relevancy and redundancy. Appl Soft Comput 156:111477. https://doi.org/10.1016/j.asoc.2024.111477
Zeng X, Li G (2013) Covariance free incremental principal component analysis with exact mean update. J Comput Inf Syst 9(16):6703–6710. https://doi.org/10.12733/jcisP1170
Zeng XQ, Li GZ (2014) Incremental partial least squares analysis of big streaming data. Pattern Recogn 47(11):3726–3735. https://doi.org/10.1016/j.patcog.2014.05.022
Zeng XQ, Li GZ, Zou HX et al (2015) An empirical comparison of incremental linear feature extraction methods. Int J Wirel Mobile Comput 8(3):249. https://doi.org/10.1504/ijwmc.2015.069387
Zhang X, Teng Z (2020) A subspace type incremental two-dimensional principal component analysis algorithm. J Algorithms Comput Technol. https://doi.org/10.1177/1748302620973531
Zhang J, Fogelman-Soulié F, Largeron C (2018) Towards automatic complex feature engineering. Springer, Cham, pp 312–322. https://doi.org/10.1007/978-3-030-02925-8_22
Zhang J, Hao J, Fogelman-Soulié F et al (2019) Automatic feature engineering by deep reinforcement learning. In: Proceedings of the 18th international conference on autonomous agents and multiagent systems. international foundation for autonomous agents and multiagent systems, Richland, SC, AAMAS '19, pp 2312–2314
Zhang J, Hao J, Fogelman-Soulié F (2020) Cross-data automatic feature engineering via meta-learning and reinforcement learning. Springer, Cham, pp 818–829. https://doi.org/10.1007/978-3-030-47426-3_63
Zhang X, Han Y, Xu W et al (2021) Hoba: a novel feature engineering methodology for credit card fraud detection with a deep learning architecture. Inf Sci 557:302–316. https://doi.org/10.1016/j.ins.2019.05.023
Zhang T, Zhang Z, Fan Z et al (2023) Openfe: automated feature generation with expert-level performance. In: Proceedings of the 40th international conference on machine learning. JMLR.org, ICML'23

Zhao H, Yuen PC, Kwok J (2006) A novel incremental principal component analysis and its application for face recognition. IEEE Trans Syst Man Cybern Part B (Cybernetics) 36(4):873–886. https://doi.org/10.1109/tsmcb.2006.870645
Zhao F, Rekik I, Lee SW et al (2019) Two-phase incremental kernel PCA for learning massive or online datasets. Complexity. https://doi.org/10.1155/2019/5937274
Zhao W, Li X, Rong G et al (2020) DAFEE: a scalable distributed automatic feature engineering algorithm for relational datasets. Springer, Cham, pp 32–46. https://doi.org/10.1007/978-3-030-60239-0_3
Zhao L, Zhang Y, Luo X et al (2023) Selecting heterogeneous features based on unified density-guided neighborhood relation for complex biomedical data analysis. In: 2023 IEEE international conference on bioinformatics and biomedicine (BIBM). IEEE, pp 771–778. https://doi.org/10.1109/bibm58861.2023.10386030
Zhao J, Ling Y, Huang F et al (2024a) Incremental feature selection for dynamic incomplete data using sub-tolerance relations. Pattern Recogn 148:110125. https://doi.org/10.1016/j.patcog.2023.110125
Zhao J, Wu D, Wu J et al (2024b) Consistency approximation: incremental feature selection based on fuzzy rough set theory. Pattern Recogn 155:110652. https://doi.org/10.1016/j.patcog.2024.110652
Zhou J, Foster DP, Stine RA et al (2006) Streamwise feature selection. J Mach Learn Res 7:1861–1885
Zhou P, Hu X, Li P et al (2017) Online feature selection for high-dimensional class-imbalanced data. Knowl Based Syst 136:187–199. https://doi.org/10.1016/j.knosys.2017.09.006
Zhou P, Hu X, Li P et al (2019a) Ofs-density: a novel online streaming feature selection method. Pattern Recogn 86:48–61. https://doi.org/10.1016/j.patcog.2018.08.009
Zhou P, Hu X, Li P et al (2019b) Online streaming feature selection using adapted neighborhood rough set. Inf Sci 481:258–279. https://doi.org/10.1016/j.ins.2018.12.074
Zhou P, Li P, Zhao S et al (2021a) Feature interaction for streaming feature selection. IEEE Trans Neural Netw Learn Syst 32(10):4691–4702. https://doi.org/10.1109/tnnls.2020.3025922
Zhou P, Li P, Zhao S et al (2021b) Online early terminated streaming feature selection based on rough set theory. Appl Soft Comput 113:107993. https://doi.org/10.1016/j.asoc.2021.107993
Zhou P, Wang N, Zhao S (2021c) Online group streaming feature selection considering feature interaction. Knowl Based Syst 226:107157. https://doi.org/10.1016/j.knosys.2021.107157
Zhou P, Zhang Y, Li P et al (2022a) General assembly framework for online streaming feature selection via rough set models. Expert Syst Appl 204:117520. https://doi.org/10.1016/j.eswa.2022.117520
Zhou P, Zhao S, Yan Y et al (2022b) Online scalable streaming feature selection via dynamic decision. ACM Trans Knowl Discov Data 16(5):1–20. https://doi.org/10.1145/3502737
Zhu G, Xu Z, Guo X et al (2020) Difer: Differentiable automated feature engineering. https://doi.org/10.48550/ARXIV.2010.08784
Zhu G, Jiang S, Guo X et al (2022) Evolutionary automated feature engineering. Springer, Cham, pp 574–586. https://doi.org/10.1007/978-3-031-20862-1_42
Ziegler J, Reimann P, Kilian A et al (2024) An approach to integrate domain knowledge into feature engineering to enhance data-driven surrogate models of simulations. Procedia CIRP 130:1276–1281. https://doi.org/10.1016/j.procir.2024.10.239
Zliobaite I, Gabrys B (2014) Adaptive preprocessing for streaming data. IEEE Trans Knowl Data Eng 26(2):309–321. https://doi.org/10.1109/tkde.2012.147
Zöller MA, Huber MF (2021) Benchmark and survey of automated machine learning frameworks. J Artif Intell Res 70:409–472. https://doi.org/10.1613/jair.1.11854

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.