International Journal of Innovative Computer Science and IT Research

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

Automated Feature Engineering for High-Dimensional Machine Learning Applications

Dr. Alicja Gosiewska

Assistant Professor, Warsaw University of Technology, Poland

Abstract

High-dimensional machine learning applications frequently involve datasets containing thousands or millions of variables, creating substantial challenges for data representation, computational efficiency, model training, and predictive reliability. Conventional feature engineering depends heavily on domain expertise and manual experimentation, making it difficult to identify meaningful feature transformations and relevant feature subsets within large search spaces. Automated feature engineering provides a systematic approach for generating, transforming, evaluating, and selecting features with limited human intervention. This paper examines an automated feature engineering framework designed specifically for high-dimensional machine learning applications. The proposed conceptual framework integrates automated preprocessing, feature generation, feature transformation, dimensionality reduction, feature selection, search-based optimization, and model-aware evaluation into a coordinated pipeline.

The methodology emphasizes hierarchical search strategies that progressively eliminate irrelevant transformations and redundant variables before computationally expensive model evaluation. The framework also incorporates statistical relevance, predictive contribution, redundancy control, computational cost, and interpretability as complementary evaluation criteria. High-dimensional feature spaces are analyzed according to their structural characteristics, including sparsity, multicollinearity, nonlinear relationships, categorical complexity, and interaction effects. The analysis demonstrates how automated feature engineering can reduce the burden associated with manual feature construction while supporting more efficient machine learning workflows.

Two conceptual tables classify the principal components and optimization mechanisms associated with automated feature engineering. Particular attention is given to preventing data leakage, controlling overfitting, maintaining reproducibility, and balancing predictive performance with computational complexity. The study concludes that automated feature engineering can serve as an important component of scalable machine learning systems when combined with controlled search, robust validation, domain constraints, and transparent feature-selection mechanisms. Future developments should emphasize adaptive search, explainable feature construction, multimodal feature engineering, and integration with increasingly autonomous AutoML systems.

Keywords: Automated Feature Engineering, High-Dimensional Data, Machine Learning, Feature Selection, Feature Construction, AutoML, Dimensionality Reduction, Feature Transformation, Predictive Modeling, Intelligent Computing.

International Journal of Innovative Computer Science and IT Research

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

1. Introduction

Machine learning systems increasingly operate on datasets characterized by large numbers of variables, heterogeneous data structures, complex interactions, and rapidly changing information environments. High-dimensional datasets can arise from text processing, image analysis, sensor networks, genomic computing, financial systems, cybersecurity, healthcare informatics, and enterprise information systems. Although additional variables may contain valuable predictive information, increasing dimensionality can also introduce redundancy, noise, multicollinearity, computational burden, and greater risk of overfitting. Consequently, effective feature engineering has become an essential component of machine learning development.

Feature engineering refers to the transformation of raw variables into representations that improve the ability of machine learning algorithms to identify meaningful patterns. Conventional approaches commonly involve manually selecting variables, transforming distributions, constructing interaction terms, encoding categorical information, and developing domain-specific representations. Such activities can require considerable expertise and experimentation. The growing complexity of modern datasets makes manual feature engineering increasingly difficult because the number of possible transformations and feature combinations can become extremely large.

Automated feature engineering attempts to address these limitations by allowing computational systems to discover useful feature transformations and subsets through systematic search and optimization. Recent research describes automated feature engineering as a central component of modern AutoML pipelines, encompassing feature extraction, construction, transformation, and selection. Automated data-processing research also emphasizes the importance of integrating preprocessing and feature engineering when handling large-scale and heterogeneous machine learning data.

The challenge becomes particularly significant in high-dimensional environments. If a dataset contains thousands of variables, an exhaustive search over transformations, interactions, ratios, aggregations, and nonlinear representations may become computationally infeasible. An automated system therefore requires efficient search strategies capable of prioritizing promising transformations while eliminating unproductive candidates. Such systems must also prevent information leakage and avoid selecting features merely because they perform well on a particular validation sample.

Automated feature engineering can therefore be viewed not simply as a mechanism for generating more variables but as an optimization problem involving representation quality, predictive utility, redundancy, computational cost, and interpretability. Recent research has demonstrated the potential value of combining automated feature engineering with AutoML, while contemporary reviews continue to identify scalability, interpretability, adaptability, and search efficiency as important research challenges.

2. Objectives

2.1 Primary Objective

The primary objective of this study is to develop a conceptual automated feature engineering framework capable of efficiently processing high-dimensional datasets for machine learning applications. The framework seeks to automate feature transformation, generation, selection, and evaluation while reducing unnecessary computational requirements and limiting the influence of irrelevant or redundant variables.

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

2.2 Development of an Automated Feature Generation Mechanism

The second objective is to establish an automated mechanism for constructing useful features from existing variables. The mechanism considers mathematical transformations, statistical transformations, interaction terms, aggregations, categorical representations, and domain-constrained operations. The purpose is to identify informative representations without requiring extensive manual experimentation.

2.3 Optimization of High-Dimensional Feature Selection

The third objective is to develop an optimization-oriented feature-selection process that identifies relevant variables while controlling redundancy and computational complexity. The proposed approach considers predictive contribution, statistical relevance, correlation, feature importance, and model performance to establish an efficient feature subset.

2.4 Integration of Automated Feature Engineering with Machine Learning

The fourth objective is to examine how automated feature engineering can be integrated with machine learning model development. The framework emphasizes iterative evaluation, cross-validation, search optimization, and model-aware feature assessment to create a coordinated pipeline suitable for complex high-dimensional applications.

3. Review of Literature

3.1 Feature Engineering and High-Dimensional Machine Learning

Feature engineering has traditionally represented one of the most important stages of machine learning development. Guyon and Elisseeff emphasized the importance of feature construction and selection for improving learning performance and reducing irrelevant information. Feature selection methods have subsequently been categorized into filter, wrapper, and embedded approaches, each providing different balances between computational efficiency and predictive optimization.

High-dimensional learning introduces additional difficulties because the number of variables may be substantially larger than the number of observations. This condition can increase model complexity and make conventional statistical assumptions less reliable. Hastie et al. identified regularization and dimensionality reduction as important strategies for learning from high-dimensional data, while James et al. discussed feature selection, transformations, and model complexity as fundamental components of predictive modeling.

Dimensionality reduction methods such as principal component analysis can reduce feature-space complexity by transforming correlated variables into a smaller number of components. However, purely dimensionality-reduction approaches may reduce interpretability because the resulting components do not always correspond directly to meaningful original variables. Automated feature engineering therefore provides an alternative approach by searching for informative representations while preserving potentially useful feature semantics.

3.2 Automated Feature Engineering and AutoML

Automated machine learning has expanded the automation of model selection, hyperparameter optimization, preprocessing, and feature engineering. Hutter et al. established the importance of

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

automated algorithm configuration and optimization, while Feurer et al. demonstrated the broader role of automated machine learning pipelines in constructing predictive systems.

Automated feature engineering extends this concept by treating feature transformation and construction as search problems. Feature engineering systems can generate candidate features through mathematical operations, relational transformations, aggregation procedures, or domain-specific operators. The resulting candidates can then be evaluated using statistical criteria or predictive performance.

Recent research specifically emphasizes the combination of AutoFE and AutoML. de Winter et al. reported that integrating automated feature engineering with AutoML can improve predictive performance without necessarily producing a proportional increase in computational time.

3.3 Search Optimization and Intelligent Feature Selection

Automated feature engineering requires efficient mechanisms for navigating potentially enormous feature spaces. Genetic algorithms, Bayesian optimization, reinforcement learning, heuristic search, and meta-learning have therefore been investigated as mechanisms for discovering effective feature representations.

Feature selection is particularly important because automated generation can produce large numbers of candidate variables. Without appropriate selection mechanisms, the feature space may grow rapidly, creating computational overhead and increasing the risk of overfitting. Recursive feature elimination, regularization, mutual information, tree-based importance, and sequential search approaches provide different strategies for controlling feature-space growth.

Recent surveys have identified automated feature generation, feature selection, and system implementation as major dimensions of automated feature engineering research. The literature further indicates that future systems need to address interpretability, scalability, dynamic data environments, and efficient search mechanisms.

4. Methodology

4.1 Research Design and Conceptual Framework

This study adopts a conceptual and analytical research design for examining automated feature engineering in high-dimensional machine learning applications. Rather than reporting fabricated experimental results, the study synthesizes established machine learning, feature-selection, dimensionality-reduction, and AutoML principles into an integrated framework. The framework is organized around sequential stages beginning with data profiling and preprocessing and continuing through feature generation, candidate evaluation, dimensionality control, feature selection, and model validation.

The methodology considers automated feature engineering as an optimization process in which a system searches a defined transformation space to identify representations that maximize predictive usefulness while minimizing redundancy and computational cost. Candidate features are assessed through statistical relevance, predictive contribution, stability, and interpretability. Cross-validation is incorporated into the evaluation stage to reduce the possibility that features are selected because of accidental relationships within a particular training or validation partition.

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

4.2 Automated Data Preprocessing and Feature Generation

The first stage involves automated examination of data types, missing values, categorical variables, numerical distributions, outliers, and feature cardinality. Preprocessing operators are selected according to the characteristics of individual variables. Numerical transformations may include normalization, standardization, logarithmic transformation, power transformation, and discretization, while categorical variables may undergo encoding procedures. The objective is to create a consistent input representation before automated feature generation begins.

The second stage generates candidate features using controlled transformation operators. These may include arithmetic combinations, ratios, differences, logarithmic transformations, polynomial relationships, temporal aggregates, statistical summaries, and interaction terms. To prevent uncontrolled feature-space expansion, transformation depth and operator combinations are restricted. Candidate generation can also incorporate domain constraints that prevent meaningless mathematical operations and preserve application-specific validity.

Key methodological operations include:

- Data-type identification and automated profiling
- Missing-value and outlier assessment
- Numerical and categorical transformation
- Candidate feature generation
- Interaction and nonlinear feature construction
- Transformation-depth control
- Redundancy filtering
- Predictive feature evaluation
- Cross-validation-based selection
- Computational-cost monitoring

4.3 Feature Selection and Dimensionality Optimization

After candidate generation, the framework applies sequential feature-selection mechanisms to eliminate irrelevant and redundant variables. Initial filtering can use low-cost statistical measures such as variance thresholds, correlation analysis, mutual information, and univariate relevance. More computationally expensive methods are subsequently applied to the reduced candidate set. This hierarchical strategy decreases the number of features that require detailed model-based evaluation.

The final feature-selection stage evaluates candidate subsets according to predictive contribution and model complexity. Embedded techniques such as regularization and tree-based importance can identify features that contribute meaningfully to prediction. Wrapper-based evaluation may subsequently compare smaller candidate subsets using cross-validation. The process continues until an acceptable balance is reached between predictive utility, feature-space size, stability, and computational cost.

4.4 Model-Aware Evaluation and Validation

The final stage integrates selected features with machine learning models and evaluates their contribution under controlled validation procedures. Multiple algorithms may be considered because a feature representation that benefits one model may not necessarily provide the same advantage for

another. Model performance can therefore be evaluated using task-appropriate metrics such as accuracy, precision, recall, F1-score, mean absolute error, root mean squared error, or area under the ROC curve. Validation must be conducted without allowing information from the testing data to influence feature construction. Feature-generation and selection operations should occur within the training folds during cross-validation. This procedure helps reduce data leakage and provides a more reliable estimate of generalization performance. Reproducibility should also be supported through fixed search configurations, documented transformation operators, version-controlled pipelines, and recorded feature-selection decisions.

5. Analysis

5.1 Components of an Automated Feature Engineering Framework

Automated feature engineering in high-dimensional environments requires multiple coordinated components. The framework must first understand the characteristics of the input data before deciding which transformation operations are appropriate. It should then generate candidate features within a controlled search space and progressively eliminate candidates that provide limited value.

Table 1: Core Components of an Automated Feature Engineering Framework

Component	Primary Function	Contribution to High-Dimensional ML
Data Profiling	Identifies variable types, distributions, missingness, and cardinality	Establishes appropriate transformation strategies
Automated Preprocessing	Standardizes and transforms raw variables	Improves consistency of model inputs
Feature Generation	Creates transformed and interaction features	Expands useful representation space
Candidate Filtering	Removes weak or redundant features	Controls feature-space growth
Feature Selection	Identifies informative variables	Reduces dimensionality
Search Optimization	Explores promising transformation combinations	Improves search efficiency
Model-Aware Evaluation	Measures predictive contribution	Links features to actual learning performance
Validation Control	Prevents leakage and overfitting	Improves reliability and generalization
Interpretability Layer	Records feature construction logic	Supports transparent model development

Interpretation: The components operate as an interconnected pipeline rather than independent processing stages. Data profiling determines the feasible transformation space, while candidate filtering prevents feature generation from producing an uncontrollable number of variables. Feature selection subsequently reduces dimensionality, and model-aware evaluation determines whether retained features provide meaningful predictive value. The interpretability layer is particularly important because automated systems can generate complex feature representations that may be difficult for users to understand. Consequently, automation should include mechanisms for recording feature origins, transformations, and selection criteria.

5.2 Automated Feature Generation and Search Strategies

The effectiveness of automated feature engineering depends strongly on the mechanism used to explore candidate transformations. An exhaustive search is generally impractical for high-dimensional datasets because the number of possible feature combinations increases rapidly with the number of input variables. Search strategies therefore need to prioritize promising regions of the transformation space.
Heuristic approaches can reduce the search burden by eliminating transformations that produce weak statistical relationships or excessive redundancy.

Bayesian optimization can guide exploration toward promising configurations, while evolutionary approaches can search combinations of transformations. Meta-learning can further exploit information from previous datasets to recommend potentially useful operators. Recent research has specifically explored meta-learning to recommend feature-engineering operations within automated machine learning pipelines.

Table 2: Automated Feature Engineering Mechanisms for High-Dimensional Applications

Mechanism	Main Purpose	Major Advantage	Primary Limitation
Filter Selection	Rapid relevance assessment	Computationally efficient	May ignore feature interactions
Wrapper Selection	Evaluate feature subsets using models	Strong predictive alignment	Computationally expensive
Embedded Selection	Select features during model training	Efficient model integration	Model dependent
Genetic Search	Explore feature combinations	Handles complex search spaces	Can require many evaluations
Bayesian Optimization	Guide search toward promising configurations	Efficient exploration	Requires suitable objective design
Reinforcement Learning	Learn transformation decisions	Adaptive sequential search	Training complexity
Meta-Learning	Reuse knowledge across datasets	Reduces search time	Depends on prior meta-data

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

Mechanism	Main Purpose	Major Advantage	Primary Limitation
Dimensionality Reduction	Compress feature representations	Controls feature-space size	May reduce interpretability
Hybrid Search	Combines multiple optimization strategies	Balances efficiency and quality	Greater system complexity

Interpretation: The mechanisms in Table 2 address different aspects of the automated search problem. Filter methods are useful for rapid elimination of obviously weak variables, whereas wrapper and embedded methods provide stronger connections between feature selection and predictive performance. Intelligent search mechanisms become increasingly important when the candidate space is extremely large. Hybrid approaches can combine inexpensive filtering with more sophisticated optimization, creating a hierarchical search process. Such designs can reduce computational requirements while preserving opportunities to discover nonlinear and interaction-based feature representations.

5.3 High-Dimensional Optimization Considerations

The primary challenge in high-dimensional feature engineering is maintaining a manageable search space while preserving useful information. Automated systems can initially generate thousands of candidate features, but retaining all candidates can increase training time and produce unstable models. Therefore, the framework should apply multiple stages of filtering rather than depending on a single feature-selection method.

Feature redundancy represents another important consideration. Highly correlated variables may provide overlapping information, while complex interaction terms can reproduce information already captured by simpler features. Automated systems should therefore evaluate both individual relevance and incremental contribution. A candidate feature should be retained when it provides information that is not adequately represented by existing features.

Computational cost should also be treated as an optimization objective. A feature representation that produces a marginal performance improvement at a substantially higher computational cost may not be appropriate for real-time or resource-constrained applications. Consequently, automated feature engineering should consider predictive performance, feature count, processing time, memory consumption, and interpretability simultaneously.

6. Discussion

6.1 Improving Machine Learning Efficiency

Automated feature engineering can significantly reduce the manual effort required to prepare high-dimensional datasets for machine learning. Instead of requiring practitioners to test large numbers of transformations individually, an automated system can systematically explore a predefined search space. This approach is particularly valuable in applications where datasets contain thousands of variables or where domain knowledge is insufficient to manually identify all relevant relationships.

The main efficiency benefit arises from hierarchical processing. Low-cost statistical filtering can eliminate large numbers of unsuitable variables before computationally expensive model-based evaluation occurs. The resulting pipeline allows computational resources to be concentrated on

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

promising feature representations. This is consistent with the broader development of automated data-processing and AutoML systems that seek to automate preprocessing, feature construction, and feature selection as integrated stages.

6.2 Predictive Performance and Generalization

Feature engineering can influence predictive performance by modifying the representation presented to a learning algorithm. Appropriate transformations may expose nonlinear relationships, temporal patterns, interactions, and normalized structures that are difficult for a model to learn from raw variables. Automated feature engineering can therefore expand the range of potentially useful representations available to the learning system.

However, automated search also increases the possibility of overfitting. If a system evaluates a very large number of feature configurations against a limited dataset, some configurations may appear effective due to random variation. Cross-validation, nested evaluation, search-space constraints, and independent testing are therefore essential. Automated feature engineering should optimize not only training or validation performance but also the stability and generalization of selected features.

6.3 Interpretability and Practical Deployment

A major concern associated with automated feature engineering is interpretability. A manually constructed variable can usually be explained through domain knowledge, whereas an automatically generated feature may consist of multiple transformations and interactions. This creates challenges for applications in which decisions must be understandable to users or organizations.

Interpretability can be improved by restricting the transformation vocabulary, maintaining feature lineage, assigning semantic descriptions to generated variables, and recording the sequence of operations used to create each feature. Recent research on interpretable automated feature engineering highlights interpretability and adaptability as important dimensions for future automated systems.

7. Challenges

High-dimensional automated feature engineering faces significant computational challenges. The number of possible transformations and feature combinations increases rapidly as dimensionality grows. Exhaustive search is therefore generally impractical, particularly when every candidate feature set must be evaluated using machine learning models. Efficient filtering, hierarchical search, meta-learning, and resource-aware optimization are necessary to maintain acceptable processing times.

Another challenge concerns overfitting and data leakage. Automated systems may evaluate a very large number of candidate representations, increasing the probability that a feature appears predictive because of accidental relationships. If feature selection occurs before cross-validation or uses information from the testing dataset, the resulting performance estimates can become unreliable. Proper pipeline isolation and nested validation are therefore essential.

Interpretability and reproducibility also remain difficult. Automatically constructed features can involve multiple transformations that are difficult to understand or reproduce. Different search configurations may produce different feature sets, even when applied to the same data. Transparent transformation logs, deterministic settings, feature lineage, and explainable selection criteria are required to make automated feature engineering suitable for dependable machine learning environments.

Volume 02, Issue 04 – Jul-Aug 2026 • E-ISSN: 3067-1108 | P-ISSN: 3067-1094

8. Future Research

Future research should investigate adaptive automated feature engineering systems capable of responding to changing datasets and evolving data distributions. Conventional feature-engineering pipelines generally assume relatively stable data structures, but modern information systems increasingly generate streaming and continuously changing data. Adaptive systems could monitor feature usefulness over time and automatically revise transformation strategies when data characteristics change.

Another important direction is the development of explainable automated feature engineering. Future systems should not only identify high-performing features but also explain why specific transformations were selected. Semantic feature descriptions, transformation graphs, feature lineage, and human-readable explanations could improve trust and facilitate adoption in sensitive machine learning applications.

Future research should also explore stronger integration between automated feature engineering, large-scale AutoML, foundation models, and multimodal learning. High-dimensional applications increasingly combine numerical, textual, temporal, visual, and categorical information. Automated systems capable of discovering meaningful relationships across these modalities could support more comprehensive intelligent information-processing environments while reducing manual pipeline design.

9. Conclusion

Automated feature engineering represents an important direction for addressing the increasing complexity of high-dimensional machine learning applications. Traditional feature engineering depends substantially on human expertise, iterative experimentation, and knowledge of domain-specific relationships. These requirements become increasingly difficult to satisfy when datasets contain thousands or millions of variables. An automated approach can systematically perform preprocessing, feature transformation, feature construction, feature selection, and model-aware evaluation while reducing the manual burden associated with conventional machine learning development. The conceptual framework presented in this study treats feature engineering as an optimization problem in which predictive usefulness, dimensionality, computational cost, redundancy, and interpretability must be considered together.

The proposed framework emphasizes a hierarchical approach to feature-space management. Automated profiling and preprocessing establish an appropriate representation of the input data, while controlled feature generation creates potentially useful transformations and interactions. Statistical filtering and redundancy analysis then reduce the number of candidates before more computationally intensive model-based evaluation is performed. Search mechanisms such as evolutionary optimization, Bayesian optimization, reinforcement learning, and meta-learning can further improve exploration efficiency. Importantly, the framework does not assume that increasing the number of generated features automatically improves machine learning performance. Instead, feature utility must be evaluated according to generalization, stability, computational requirements, and incremental predictive contribution.

The long-term effectiveness of automated feature engineering will depend on its ability to balance automation with transparency and reliable validation. High-dimensional search creates opportunities for overfitting, data leakage, unstable feature selection, and excessive computational consumption. Therefore, automated systems should incorporate cross-validation, feature lineage,

transformation constraints, reproducible search configurations, and interpretable selection criteria. Future systems should extend these capabilities toward adaptive and multimodal environments in which data characteristics continuously evolve. Overall, automated feature engineering can become a foundational component of intelligent machine learning pipelines when designed as a controlled, model-aware, and interpretable optimization process rather than simply as an automated mechanism for generating large numbers of variables.

References

1. Aha, D. W. (1992). Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies, 36*(2), 267–287.
2. Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press.
3. Breiman, L. (2001). Random forests. *Machine Learning, 45*, 5–32.
4. Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering, 40*(1), 16–28.
5. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.
6. De Winter, C., Frasincar, F., de Peuter, B., Matsiiako, V., Ido, E., & Klinkhamer, J. (2025). Automated feature engineering for automated machine learning. *Knowledge-Based Systems, 321*, 113671.
7. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. *Advances in Neural Information Processing Systems, 28*, 2962–2970.
8. Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics, 29*(5), 1189–1232.
9. Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1157–1182.
10. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.
11. He, X., Zhao, K., & Chu, X. (2021). AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems, 212*, 106622.
12. Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. *Learning and Intelligent Optimization*, 507–523.
13. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An Introduction to Statistical Learning: With Applications in R*. Springer.
14. Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1–2), 273–324.
15. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys, 50*(6), 1–45.
16. Liu, H., & Motoda, H. (1998). *Feature Selection for Knowledge Discovery and Data Mining*. Springer.