Implications of Data Leakage in Machine Learning Preprocessing: A Multi-Domain Investigation

Mohamed Aly Bouke [1,*], Saleh Ali Zaid[1,2], Azizol Abdullah[1]

[1]Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang 43400, Malaysia.

[2]College of Electronic Technology, Tripoli, Libya.

*Corresponding author(s). E-mail(s): bouke@ieee.org; Phone: +60 1157-53 1822
Contributing authors: saleh.zaid@ieee;org;azizol@upm.edu.my

Abstract - Data leakage during machine learning (ML) preprocessing is a critical issue where unintended external information skews the training process, resulting in artificially high-performance metrics and undermining model reliability. This study addresses the insufficient exploration of data leakage across diverse ML domains, highlighting the necessity of comprehensive investigations to ensure robust and dependable ML models in real-world applications. Significant discrepancies in model performance due to data leakage were observed, with notable variations in F1 scores and ROC AUC values for the Breast Cancer dataset. The Tic-Tac-Toe Endgame dataset analysis revealed the varying impact on models like Ridge, SGD, GaussianNB, and MLP, underscoring the profound effect of data leakage. The German Credit Scoring dataset showed slight enhancements in recall and F1 scores for models like DT and GB without data leakage, indicating reduced overfitting. Additionally, models such as PassiveAggressive, Ridge, SGD, GaussianNB, and Nearest Centroid exhibited shifts in performance metrics, highlighting the intricate response to data leakage. The study also revealed raw data leakage rates, such as 6.79% for Spambase and 1.99% for Breast Cancer. These findings emphasize meticulous data management and validation to mitigate leakage effects, which is crucial for developing reliable ML models.

Keywords: Data Leakage, Machine Learning Models, Model Performance, Overfitting, Data Preprocessing.

1. Introduction

ML has been influential in ushering significant changes across various sectors as a significant subset of Artificial Intelligence. ML continues to serve as a transformative force across industries, from enabling predictive healthcare to augmenting finance risk management and fortifying cybersecurity through threat identification. At the heart of these ML applications lie the algorithms, which, although highly sophisticated, derive their effectiveness mainly from the quality and integrity of the input data [1]–[4].

Data preprocessing is a critical stage in the ML pipeline where raw data undergoes significant transformations to render it suitable for ML algorithms. The preprocessing steps, which include data cleaning, normalization, feature extraction, and feature selection, are pivotal to managing data complexity, rectifying missing or noisy data issues, and converting the data into a more digestible format for ML algorithms. However, precisely within these transformative steps, the potential for a significant risk - data leakage – lurks [5], [6].

In the field of ML, "data leakage" denotes the inadvertent inclusion of information from the target variable or future events into the training dataset. This occurrence can result in artificially inflated performance metrics for ML models during their evaluation phase, as they can access extra information typically unavailable during actual prediction. However, these exaggerated performance metrics mask the model's inability to adapt to new, unseen data.

Therefore, in practical applications where the data differs from that of the test set, the performance of these models often deteriorates, owing to their training on an unrealistic representation of data. Hence, data leakage during the data preprocessing phase is a critical issue, threatening the generalizability, reliability, and validity of ML models [7]–[9].

*Corresponding Author: bouke@ieee.org

Data leakage manifests in various ways, influenced by the preprocessing methods used and the characteristics of the data being handled. Common types include feature engineering leakage, where new features are generated using information from the complete dataset. Similarly, feature selection leakage arises when the criteria for eliminating features are based on the entire dataset. Target leakage is another form where the target variable is, either directly or indirectly, included in the training features. Temporal leakage is a frequent issue in time-series data, occurring when models are trained using future data. Lastly, data preprocessing leakage may happen during normalization and encoding steps if they involve information from the full dataset [10]–[12].

Despite the critical implications of data leakage, the current body of research lacks a comprehensive, cross-domain exploration of this phenomenon across a broad spectrum of preprocessing techniques. Moreover, the extent to which data leakage in preprocessing affects the performance of different ML algorithms has not been sufficiently explored. This research addresses these gaps, offering an exhaustive empirical examination of data leakage during diverse preprocessing stages and its consequential impact on different ML algorithms.

The contribution of this study to the ML field extends beyond its extensive empirical approach. It adds to the ongoing discourse surrounding the development of robust, reliable, and accurate ML models. This work fosters a more nuanced understanding of this critical issue by investigating the oft-obscured aspects of data leakage during preprocessing and elucidating its implications on ML algorithm performance. Furthermore, it opens avenues for designing and implementing improved practices and tools to identify, mitigate, and prevent data leakage in the ML pipeline.

The rest of the paper is organized as follows: Section 2 provides a comprehensive literature review, setting the context for our research by drawing upon existing studies. Section 3 elaborates on our research methodology, detailing our selection of ML models, data collection process, performance metrics used, and the implementation environment. Section 4 presents a detailed analysis of our results, where we thoroughly discuss the implications of our findings in the context of data leakage on the performance of ML models. Finally, in Section 6, we offer our concluding thoughts, summarizing the findings and their potential implications for future research in the field.

2. Literature review

ML plays a crucial role in diverse fields such as healthcare and cybersecurity. Its ability to identify intricate patterns within large datasets is invaluable. However, the effectiveness of ML models heavily relies on having comprehensive, precisely labeled training and testing datasets. Data leaking, which happens when training and testing datasets are not independent and identically distributed, is a major challenge during data preparation. This can result in overfitting, ultimately impacting the model's performance when deployed in real-world scenarios.. [13]. Indeed, researchers have extensively investigated the impact of data leakage on ML model performance.

Dong [14] investigates the crucial issue of data leakage in machine learning, specifically when using data from sports wearable sensors. This leakage, where independent variables used for training models contain information about the dependent variable or related data, significantly undermines the validity and reliability of AI models. To combat this, Dong proposes a novel Bayesian inference-based approach. This method predicts leakage by calculating the reverse probability of unseen variables, effectively identifying and avoiding data leakage. By evaluating the marginal likelihood of observed variables originating from a coupling method, the approach helps determine the probability of leakage. Higher marginal probability indicates a better fit of the data, suggesting a greater likelihood of data leakage. This approach is validated using a sports wearable sensor dataset, demonstrating its effectiveness in detecting data leakage.

Kapoor and Narayanan [15] aim to investigate the widespread problem of data leakage in scientific research that utilizes machine learning techniques, highlighting its effects on the ability to reproduce and rely. Their analysis, spanning 17 fields and 294 studies, reveals the detrimental effects of data leakage. They categorize eight specific types of leakage, ranging from simple data handling errors to more complex discrepancies in the distribution of training and testing data. To address these issues, they suggest integrating model information sheets into the research workflow, which are designed to aid researchers in recognizing and mitigating data leakage, thereby enhancing the integrity of model development. A case study on predicting civil wars demonstrates how data leakage can significantly inflate the perceived effectiveness of sophisticated ML models over conventional logistic regression techniques. When

adjustments for leakage are made, the presumed superiority of these advanced models diminishes, underscoring the importance of meticulous data management and analytical methods in ML research.

Stock et al. [16] address the prevalent issue of data leakage in the ecological application of ML, which leads to misleading results. Data leakage occurs when a model, during its training phase, has access to information that it wouldn't have during its actual application, thereby inflating its predictive accuracy on test datasets but performing poorly on new, unseen data. This problem is magnified in ecology due to the discipline-specific data collection and preprocessing practices that can unwittingly introduce spurious relationships, making models fail when applied outside their training dataset. The authors provide examples illustrating how data leakage has impacted ecological research, including studies on camera trap images, kelp habitat prediction, and chlorophyll concentration prediction, to demonstrate the challenge's scope. They advocate for domain-specific methodologies to prevent data leakage, including sound research design, independent data evaluation, and block cross-validation.

Farokhi and Kaafar [17] delve into the vulnerabilities of ML models to membership inference attacks, where adversaries aim to deduce individuals' data usage in model training. They introduce information-theoretic metrics to model membership information leakage and examine how various aspects of ML models and training algorithms influence this leakage. Their study employs conditional mutual information leakage, quantified through Kullback–Leibler divergence, offering an easier numerical computation method. They establish a direct link between this divergence-based leakage measure and the success probability of adversaries in membership inference, demonstrating how mutual information leakage decreases with larger training dataset sizes and regularization weight. Furthermore, they illustrate that complex models with high sensitivity are more prone to these attacks.

Zhang et al. [18] explore the risk of dataset property leakage in secure multi-party ML, a setting that allows multiple parties to collaboratively build models on pooled data without sharing their data directly. They demonstrate that, even when parties only have black-box access to the final model, a "curious" party can infer the distribution of sensitive attributes in other parties' data with high accuracy. This revelation is significant as it challenges the confidentiality of global dataset properties rather than individual records, raising concerns about the protection of proprietary information or intellectual property contained within the data. Through analyses on various types of datasets, including tabular, text, and graph data, the authors show that this leakage can occur irrespective of the sensitive attribute's direct inclusion in the training data or its correlation with other attributes or the target variable.

Hannun et al. [19] propose a novel method to measure information leakage in machine learning models by leveraging Fisher information. Unlike the worst-case assessments of differential privacy, this method provides a more granular analysis, allowing for the evaluation of leakage on a specific example, attribute, or sub-population level within a dataset. The proposed metric, Fisher information loss, quantifies the amount of information a model leaks about its training data. This is particularly relevant when dealing with sensitive data, where model predictions could potentially reveal training set membership or sensitive attributes. Unlike differential privacy, which is hindered by correlations within data, Fisher information loss is sensitive to these correlations and does not degrade in their presence. This makes it a promising tool for safeguarding individual and subgroup privacy. Furthermore, this approach aligns with fairness in privacy considerations, potentially mitigating "disparate vulnerability" to privacy attacks across different demographic groups. Through empirical validation, the study demonstrates Fisher information loss as a viable measure for assessing information leakage, contributing significantly to privacy-preserving ML.

Kapoor et al. [20] address the imperative for precise reporting standards in ML-based science to tackle issues of validity, reproducibility, and generalizability. They present the reforms checklist, a comprehensive set of reporting guidelines developed through a consensus of 19 researchers across diverse fields, aiming to mitigate common pitfalls in ML-based research. This checklist, comprising 32 questions across eight modules, serves as a tool for researchers during study design, referees during paper review, and journals for enforcing transparency and reproducibility standards. The guidelines emphasize the significance of detailed documentation in ensuring the reliability of ML applications in scientific research, thereby enhancing the credibility and applicability of ML-based scientific claims

Subotić et al. [21] introduce a new approach for statically detecting data leakages in data science code, particularly within the realm of ML. Data leakage, a critical issue where information external to the training dataset influences the model, often results in overly optimistic or ineffective models in practical scenarios. Traditional detection methods

rely on runtime analysis, identifying leakages post-factum through data analysis techniques. This paper presents a novel static analysis method designed to identify potential data leakages during the development phase, thereby serving as a preemptive measure alongside existing runtime methods. Integrated within the NBLyzer static analyzer framework, this lightweight analysis is tailored for interactive data science notebooks, demonstrating its effectiveness through application to real-world benchmarks.

Bouke and Abdullah [22] examines the potential pitfalls of data leakage during the preprocessing stage of machine learning-based intrusion detection systems (IDS). The study highlights how training data unintentionally contaminating test data, a phenomenon known as pattern leakage, can result in overfitting and inflated performance metrics. Through experiments using three widely recognized intrusion detection datasets – UNSW-NB15, NSL-KDD, and KDDCUP99 – the researchers manipulated data to introduce and then mitigate leakage. The study examined six different machine learning models, finding that each model displayed varying degrees of sensitivity to data leakage. Some models exhibited a notable decrease in accuracy after leakage was addressed, indicating that this issue can have a significant impact on performance. These findings underscore the critical importance of meticulously handling data during preprocessing to guarantee the reliability and practical effectiveness of intrusion detection systems (IDS) in real-world applications. The research provides specific recommendations aimed at reducing data leakage and enhancing model robustness.

Gibney [23] discusses concerns raised by Kapoor and Narayanan [15] about a looming reproducibility crisis in ML-based research across various disciplines, citing overestimated predictions due to data analysis errors and inadequate peer review processes. They advocate for a comprehensive definition of reproducibility that includes computational reproducibility and the predictive accuracy of models, highlighting the issue of data leakage where models are inadvertently trained and tested on the same data, leading to inflated accuracy. The researchers propose structured documentation, including "model info" sheets, to mitigate data leakage and enhance research integrity. Despite some disagreements from scrutinized fields, their call for attention has resonated, emphasizing the need for careful application of ML and interdisciplinary collaboration to prevent the crisis and maintain trust in ML research.

Koçak et al. [24] articulate the foundational principles for conducting clinical research on artificial intelligence and ML in healthcare, emphasizing the necessity for valid scientific purpose, high quality datasets, robust reference standards, and input. They detail the critical avoidance of information leakage, the pursuit of an optimal bias-variance tradeoff, appropriate model evaluation, proven clinical utility, transparent reporting, and the advocacy for open science. Highlighting the exponential growth of AI-related publications and the substantial gap in clinician involvement in AI development, the authors argue for enhanced AI literacy among healthcare professionals to bridge the language barrier between clinicians and AI developers. This review aims to foster critical thinking and improve AI comprehension among readers, suggesting a collaborative, multidisciplinary approach in the conceptualization phase of AI healthcare research to ensure its practical translation into clinical settings.

Lastly, Rosenblatt et al. [25] examine the consequences of data leakage in predictive modeling within neuroimaging, focusing on machine learning models that utilize functional and structural connectomes. The study identifies five distinct types of leakage including feature selection, covariate correction, and subject dependency across four datasets and three phenotypes. Their findings reveal that feature selection and the inclusion of repeated subjects lead to a significant overestimation of prediction performance, with smaller datasets amplifying these effects. The findings emphasize the importance of rigorous data handling practices in neuroimaging predictive modeling to ensure the validity and reproducibility of results. This study contributes to understanding data leakage's impact on neuroimaging research, underlining the need for careful design and analysis to avoid inflated performance metrics.

To this end, while ML models have been instrumental in solving complex problems across domains, they are susceptible to pattern leakage, particularly during the data preprocessing phase. The literature underscores the importance of employing strategies such as cross-validation, randomization, and oversampling to enhance the representativeness of datasets.

However, despite the progress made in discerning and mitigating data leakage in ML models, there remains an exigent call for more comprehensive research that spans a broader spectrum of datasets and ML models. Our study addresses this exigency by scrutinizing the ramifications of data leakage in a multitude of contexts and across various ML

models. Specifically, we delve into the influence of two prevalent preprocessing practices—normalization of data and the encoding of categorical variables—on data leakage and model reliability.

Our research stands out for its broad-based scrutiny of data leakage, offering a comprehensive perspective on this pivotal concern. By dissecting the influence of pattern leakage on the reliability of different ML models, we provide critical insights for both the academic and practical realms. Through rigorous empirical analysis and innovative methodological approaches, our study aims to enhance the robustness and reliability of ML models, contributing to the advancement of privacy-preserving ML practices.

3. Materials and Methods

This study employs a rigorous methodological approach to examine the impact of data leakage on ML models during the data preprocessing phase. Utilizing Python—a versatile and powerful programming language favored for data analysis—and Scikit-Learn, a robust library renowned for its effectiveness in building ML models, we have developed a dual-path experimental framework. This framework is designed to simulate common data leakage scenarios on one path, while implementing best practices to prevent such leakage on the other. This bifurcated approach allows for a comprehensive exploration and understanding of the impacts of data leakage, grounded in a meticulously structured experimental setup (refer to Figure 1).

Our experiments focus on two primary forms of data leakage. The first, **Preprocessing Data Leakage**, occurs when information from the test set inadvertently influences the training process through shared preprocessing steps. The second, **Model Leakage**, happens when the training process incorporates information or features that should not be available at the prediction time. These forms of leakage are critical pitfalls in ML workflows, as they can lead to inflated performance metrics and an overestimation of model capabilities. By addressing these issues, our study aims to highlight potential improvements in data handling and model training processes to enhance the reliability and accuracy of ML models.

Moreover, Initially, we gather a collection of diverse datasets, encompassing a broad spectrum of domains to ensure the robustness and generalizability of our findings. This compilation of datasets is represented by:

$$D = \{d_1, d_2, ..., d_n\}$$

where d_i stands for an individual dataset within the collection. Within each dataset, we differentiate between categorical and numerical features, denoted as $C = \{c_1, c_2, ..., c_m\}$ for categorical features and $N = \{n_1, n_2, ..., n_k\}$ for numerical features.

Experiment 1: Preprocessing with Data Leakage

The structure of Experiment 1 is deliberately designed to introduce preprocessing data leakage. This form of data leakage emanates from applying preprocessing steps—such as encoding and scaling—to the entire dataset prior to partitioning it into training and testing sets. Specifically, categorical features are first transformed into a numerical format using the LabelEncoder function, followed by the scaling of both the newly encoded and original numerical features using the StandardScaler function. This sequence of operations results in a processed dataset that is then split into training and testing sets with an 80-20 ratio, thereby inadvertently allowing information from the testing set to influence the training process.

In this experiment, the impact of preprocessing data leakage is evaluated through the training and subsequent performance assessment of a comprehensive suite of 20 ML models on the preprocessed data. This wide selection of models ensures a broad evaluation scope, allowing us to document and analyze the effects of data leakage across different types of models. Performance metrics such as Accuracy, Precision, Recall, F1 Score, and ROC AUC Score are meticulously calculated for each model, providing a nuanced understanding of the model's performance under conditions of data leakage.

The first experiment simulates a scenario where data leakage is present due to preprocessing steps being applied before splitting the data. The categorical features undergo encoding into numerical format using a LabelEncoder function, L, transforming C into C' :

$$C' = L(C)$$

Subsequently, both the newly encoded categorical features and the original numerical features are scaled using a StandardScaler function, S, resulting in N' :

$$N' = S(N \cup C')$$

Following the preprocessing steps, the dataset is divided into training and testing sets, T_{train} and T_{test}, with an 80-20 split. This action, occurring post-preprocessing, introduces data leakage into the experimental setup.

The ML models, collectively represented by $M = \{m_1, m_2, ..., m_{20}\}$, are trained on the preprocessed training set and evaluated on the testing set. The performance of each model is captured through metrics such as Accuracy, Precision, Recall, F1 Score, and ROC AUC Score.

Experiment 2: Preprocessing without Data Leakage

Contrasting with the first, experiment 2 is meticulously architected to prevent data leakage by modifying the sequence of data handling operations. Here, the datasets are preemptively divided into training and testing sets, ensuring that preprocessing steps are conducted independently on each set. This critical adjustment precludes the transfer of information from the testing set to the training set, effectively mitigating the risk of data leakage. The categorical and numerical features of the training and testing sets are then independently encoded and scaled, adhering to best practices in data handling.

Following this rigorous preprocessing regime, the ML models are trained and evaluated on these independently processed sets. This methodological structure allows for an empirical comparison between models trained under conditions of data leakage and those trained without it, highlighting the profound impact that methodological rigor in preprocessing can have on the reliability and performance of ML models.

In contrast, the second experiment is meticulously designed to prevent data leakage by altering the sequence of data handling operations. The datasets are initially split into training and testing sets, denoted as T_{train} and T_{test}, before any preprocessing:

$$T_{\text{train}}, T_{\text{test}} = \text{train_test_split(D, 0.8)}$$

This preemptive splitting ensures that the preprocessing steps are conducted independently on the training and testing sets, thus averting the leakage of information. The categorical and numerical features within each set are then processed separately:

$$C'_{\text{train}} = L(C_{\text{train}}), \quad C'_{\text{test}} = L(C_{\text{test}})$$
$$N'_{\text{train}} = S(N_{\text{train}}), \quad N'_{\text{test}} = S(N_{\text{test}})$$

Following this independent preprocessing, the models in M are trained on T'_{train} and assessed on T'_{test}, where T'_{train} and T'_{test} symbolize the post-preprocessing training and testing sets, respectively.

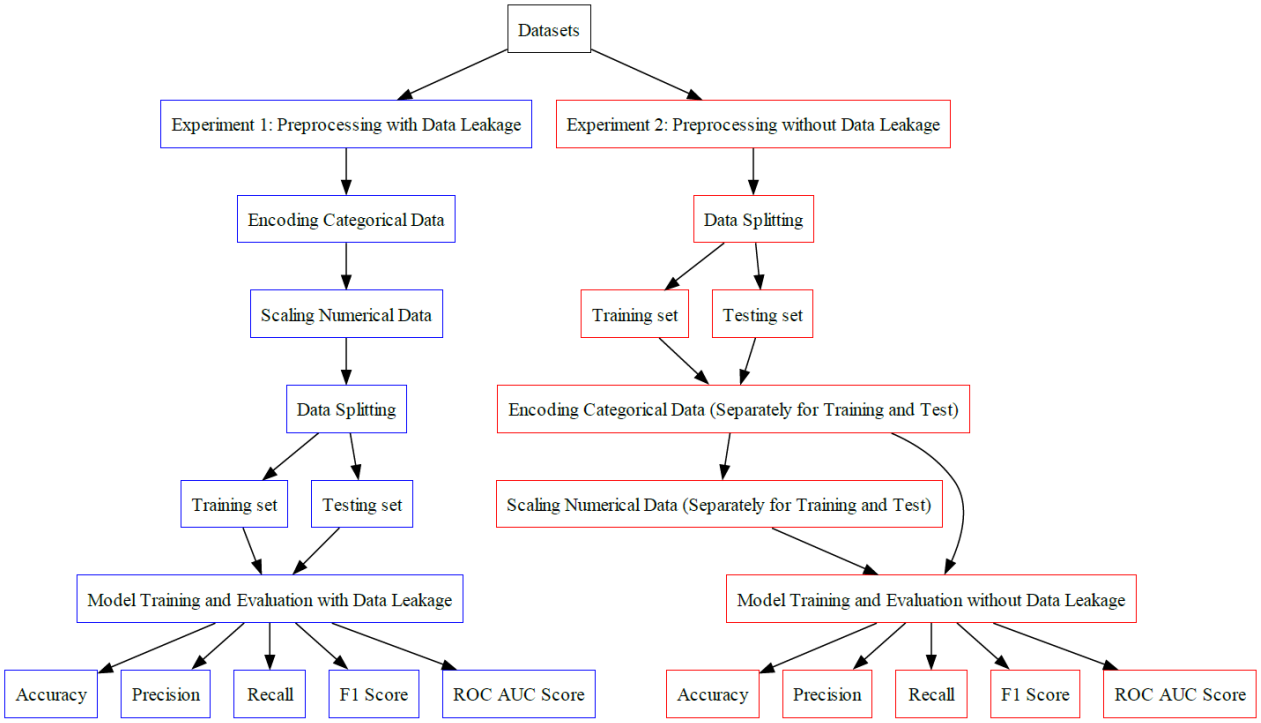

Figure 1 Experimental Methodology.

3.1. Model Selection

Our study utilizes a comprehensive selection of 20 ML models to evaluate the impact of data leakage across a diverse range of algorithmic principles and complexities. We have chosen 20 ML models for our investigation to capture a broad spectrum of algorithms ranging from decision trees, ensemble methods, linear and non-linear classifiers, probabilistic classifiers, neural networks, and support vector machines. Each model selected provides unique value to our study. The models are as follows [26]–[30]:

- **Decision Tree Classifiers (DT)**: are a cornerstone of ML due to their interpretability and simplicity. Their binary tree structure allows for a high level of transparency, enabling us to trace each decision made by the model, which is beneficial for our study of data leakage. They efficiently handle categorical variables and don't require extensive data preprocessing, which aids in isolating the effects of our controlled data leakage.

- **Gradient Boosting Classifier (GB)**: is an ensemble model that optimizes for bias, creating a robust predictive model from a collection of simpler weak models. Given their outstanding performance in various ML tasks, Gradient Boosting Classifier provides an excellent basis for understanding how a high-performing and complex model reacts to data leakage.

- **K-Nearest Neighbors (KNN)**: This model offers a non-parametric technique considered particularly effective when the decision boundary is highly irregular. The ability of KNN to capture non-linear relationships and its simplicity make it an ideal model for studying the effects of data leakage.

- **Support Vector Machines (SVM)**: This algorithm shines in high-dimensional spaces, even when the number of dimensions surpasses the number of samples. SVM's capacity for capturing complex relationships is pivotal to our model selection.

- **Random Forest Classifier (RF)**: Notable for its high accuracy, robustness, and user-friendly nature, Random Forest provides valuable insights into how data leakage impacts ensemble models. It also minimizes overfitting by averaging predictions from individual decision trees.
- **Logistic Regression (LR)**: Despite its relative simplicity, it provides robust predictive power and a probabilistic framework. Its use as a performance baseline in many ML tasks ensures a solid foundation for our study.
- **Linear Discriminant Analysis (LDA)** and **Quadratic Discriminant Analysis (QDA)**: These statistical classifiers are chosen for their basis in solid statistical theory, providing a unique perspective on the effects of data leakage. While LDA assumes identical covariance matrices among the classes, QDA relaxes this assumption, providing a more flexible decision boundary.
- **AdaBoost Classifier**: focuses on boosting the model's performance by adjusting the weights of an observation based on the previous classification. If an observation is classified incorrectly, it becomes more influential in the next iteration, pushing the model to perform better.
- **Bagging Classifier**: leverages the power of ensemble learning to reduce the variance of a base estimator (like decision trees), introducing randomness in its construction procedure and making an ensemble out of it. This variance reduction technique gives us an excellent understanding of how data leakage might bias our models.
- **Extra Trees Classifier**: This ensemble method adds an extra layer of randomness to tree-building over the traditional Random Forest Classifier. This classifier is known for its robustness against overfitting and offers a good comparison point to other ensemble methods.
- **Gaussian Process Classifier**: This model provides a probabilistic, non-parametric approach that estimates predictions and uncertainty. Evaluating the impact of data leakage on such uncertainty measures provides a novel perspective to our study.
- **Passive Aggressive Classifier**: This model belongs to the category of online learning algorithms, making it relevant when online learning is gaining momentum. It provides insights into how data leakage could affect models that are updated successively.
- **Ridge Classifier**: This linear model benefits from L2 regularization, constraining the model's complexity, thus preventing overfitting. The study of the effects of data leakage on regularized models like Ridge Classifier forms an essential part of our analysis.
- **Stochastic Gradient Descent (SGD) Classifier**: Given its efficiency and ease of implementation, SGD is a popular choice for large-scale learning and high-dimensional data. It's an excellent model for examining the impacts of data leakage on large-scale learning scenarios.
- **Gaussian Naive Bayes (GaussianNB)**: Despite its simplicity, GaussianNB performs surprisingly well in many scenarios. Assuming feature independence provides a unique vantage point in our study of data leakage.
- **Nearest Centroid**: This classifier is a simple algorithm that represents each class by the centroid of its members. Its simplicity and straightforward approach make it another good candidate for our experiments.
- **Multi-Layer Perceptron (MLP)**: MLP, an elementary form of neural network, helps us understand the impact of data leakage on deep learning models, offering a lens into the rapidly evolving domain of neural networks.
- **Nu-Support Vector Classification (NuSVC)**: This model provides a slightly different implementation of SVM, allowing more control over the number of support vectors used. It will enable us to compare how slight variations in similar models affect their reaction to data leakage.

- **ExtraTree Classifier**: This model offers another tweak to the general-purpose ensemble method. The ExtraTree Classifier introduces more randomness in selecting cut points, leading to further decorrelation between trees.

Collecting these models provides a wide spectrum of algorithmic design, complexity, and purpose. This diverse selection enables us to comprehensively evaluate and understand the impact of data leakage on the performance of ML models, ensuring the robustness of our study.

3.2. Data Collection

This research employs five distinctive datasets, each providing a unique perspective on diverse domains of study. These datasets were judiciously selected for their specific characteristics and the variety of scenarios they represent. This diversity, in terms of subject areas, attributes, and tasks, facilitates a comprehensive investigation into the influence of data leakage on the performance of ML models. Please note that the focus of this study is on binary classification tasks. Here are the details of each dataset:

- **Spambase** [31]**:** The Spambase dataset, donated on June 30, 1999, is drawn from the field of computer science and aims to classify emails as spam or non-spam. It comprises 4,601 instances with 57 attributes of integer and real types. Most attributes indicate whether a particular word or character frequently occurred in the email, providing a snapshot of the email content without revealing the actual text. The run-length attributes measure the length of sequences of consecutive capital letters, providing insight into the email formatting. This dataset has missing values denoted by the symbol 1. The final attribute is binary and indicates whether the email was classified as spam (1) or not (0). This dataset provides a unique opportunity to study the impact of data leakage on text classification tasks, specifically for spam detection, an ever-evolving challenge in computer science.
- **Breast Cancer Wisconsin (Diagnostic)** [32]**:** This dataset is derived from the healthcare domain, focusing specifically on breast cancer diagnostics. It includes 569 instances and 30 real-valued attributes, each representing a fine needle aspirate (FNA) of a breast mass. The attributes are computed from a digitized image of the FNA and describe characteristics of the cell nuclei present in the image. The objective of this dataset is to correctly classify whether a breast mass is malignant or benign based on the attributes provided. This dataset offers a unique opportunity to study the impact of data leakage on classification tasks in a medical context, where the stakes are particularly high.
- **Tic-Tac-Toe Endgame** [33]**:** This dataset is game-based and specifically focused on tic-tac-toe endgames. It includes 958 instances, each containing a possible board configuration at the end of a tic-tac-toe game, assuming that "x" made the first move. It consists of nine categorical attributes that represent the state of each cell on the game board. The objective, or target, is to determine a "win for x", which is true when "x" has one of the eight possible configurations that result in a "three-in-a-row." This dataset is particularly useful for exploring the impact of data leakage on classification tasks related to game outcomes.
- **German Credit Scoring** [34]**:** This dataset is collected from the financial domain and is used for credit scoring purposes. It includes 1000 instances and 20 attributes, including numerical and categorical variables. The attributes capture various aspects of the customers, such as account balance, credit history, employment duration, instalment rate, personal status, etc. The binary target variable indicates whether the credit was good or bad. This dataset is interesting for our analysis because it represents a real-world scenario where data leakage can have severe financial implications.
- **Student Performance** [35]: This dataset, donated on November 26, 2014, studies student achievement in secondary education (high school) in two Portuguese schools. The dataset includes 649 instances with 33 integer attributes covering various aspects such as student grades, demographic, social, and school-related features. These features were collected using school reports and questionnaires. The dataset is multivariate and can be used for classification and regression tasks. Specifically, Mathematics (mat) and Portuguese language performance can be modeled under binary/five-level classification and regression tasks. The target attribute G3 (final year grade) strongly correlates with attributes G2 and G1 (1st and 2nd-period grades), which makes predicting G3 without G2 and G1 more complex but much more helpful. This dataset gives us

a unique opportunity to investigate the effects of data leakage on predicting educational outcomes, a critical task with profound implications for educational policy and student support.

3.3. Performance Metrics

In understanding model performance within ML studies, it's crucial to engage in detailed analysis using key metrics that provide insightful feedback on a model's effectiveness. This feedback is instrumental in fine-tuning the models to achieve enhanced predictive accuracy. In our study, we employed five widely recognized metrics for model evaluation: Area Under the Receiver Operating Characteristic Curve (ROC AUC), Recall, Accuracy, F1-score, Precision, and the. These metrics were chosen for their ability to give a comprehensive overview of the model's performance from various evaluative perspectives. Let's explore each one [36], [37]:

- **Accuracy**: This metric measures the model's ability to correctly predict outcomes, calculating the proportion of true positives and negatives out of all instances. Accuracy is most effective in scenarios with balanced class distributions but can be misleading in cases of skewed class distributions, as it does not differentiate between false positives and negatives.
- **Recall (Sensitivity)**: Recall assesses the proportion of actual positive instances that the model correctly identifies. It is particularly important in contexts where missing true positives (false negatives) carries a high cost, focusing on minimizing the omission of actual positives.
- **Precision**: This metric calculates the ratio of correctly predicted positive observations to all predicted positives. Precision becomes crucial when the consequences of false positives are significant, aiming to minimize incorrectly identifying negative events as positive.
- **F1-score**: The F1-score is the harmonic mean of precision and recall, balancing these two metrics. It is especially useful in situations with unbalanced class distributions, with a score of 1 representing perfect precision and recall, and 0 being the worst score.
- **ROC AUC**: ROC AUC evaluates the trade-off between true positive rates and false positive rates at various threshold settings. It is particularly beneficial in imbalanced datasets, providing an overall performance measure across all potential classification thresholds.

These metrics are calculated as follows:

$$Accuracy \ = ((TP + TN))/((TP + TN + FP + FN)) \quad (1)$$

$$Recall\ (Sensitivity) = TP/((TP + TN)) \quad (2)$$

$$Precision = TP/((TP + FP)) \quad (3)$$

$$Fscore = 2 \times (\ Precision \times Recall)/(Precision + Recall) \quad (4)$$

Where:

- **TP (True Positive)**: This represents the count of instances accurately identified as belonging to the positive class.
- **TN (True Negative)**: The number of instances correctly classified as belonging to the negative class.
- **FP (False Positive)**: The count of instances wrongly classified as the positive class, also known as Type I error.
- **FN (False Negative)**: The number of instances incorrectly identified as the negative class, referred to as Type II error.

Furthermore, to tackle the possible effects of dataset imbalance on evaluating performance, we integrated the ROC AUC (5) metric into our analysis:

- **ROC AUC (5)**: This measure determines the efficiency of a classifier at different decision points by assessing the area beneath the ROC curve. Its primary utility lies in balancing the rate of accurately detected positive cases (TP) against the rate of erroneously marked positive instances (FP).

Utilizing these metrics, our goal is to assess the impact of pattern leakage during data preparation on our model's performance in scenarios with and without such leakage. This multi-dimensional evaluation strategy ensures that our analysis is not skewed towards any single performance aspect. It offers a holistic understanding of how data leakage affects the model's effectiveness, providing a comprehensive and balanced view of its capabilities under different preprocessing conditions.

4. Results and Discussion

In this section, we delve into the analysis and interpretation of the findings obtained from our rigorous investigation. The objective is to examine how pattern leakage during data preprocessing affects the dependability of different machine learning models in various domain. Our comprehensive investigation involves applying two common preprocessing techniques, data normalization, and encoding categorical features. Further, we consider a broad selection of ML models, which provides a more extensive understanding of data leakage implications in a general context.

We present the baseline performance of the ML models trained and tested on data with data leakage. We then retrain the model without data leakage and compare the observed performance variations. This comparison clearly shows how data leakage can distort a model's performance, providing false assurances about the model's potential in real-world scenarios. This comprehensive analysis and discussion aim to contribute to the field's understanding of data leakage and its implications. The ultimate goal is to equip researchers and practitioners with the knowledge and insights necessary to mitigate data leakage and improve their ML models' reliability.

4.1. Spambase Dataset

The experiments conducted on the Spambase dataset with data leakage (Table 1) demonstrate how various ML models can exhibit seemingly high-performance metrics, which may misleadingly suggest superior predictive capabilities. For example, the DT model showed a remarkable accuracy of 92%, which could easily lead to the false belief that it is exceptionally effective. However, this high performance is problematic as it drastically declines when data leakage is removed, plummeting to just 70%. This stark contrast indicates a significant vulnerability in the DT model: it may be highly prone to overfitting when trained with leaked data. Such discrepancies between training conditions highlight the danger of relying on distorted performance metrics that do not accurately reflect a model's effectiveness in real-world scenarios, potentially misleading practitioners and stakeholders about the model’s true capabilities.

Table 1 Results of Experiments on Spambase with Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	92%	92%	88%	90%	92%
GB	94%	96%	91%	93%	98%
KNN	90%	90%	85%	87%	94%
SVM	93%	95%	89%	92%	98%
RF	96%	98%	93%	95%	99%
LR	92%	93%	87%	90%	97%
LDA	88%	93%	78%	85%	96%
QDA	84%	75%	93%	83%	94%

AdaBoost	94%	93%	92%	92%	98%
Bagging	93%	95%	89%	92%	97%
ExtraTrees	95%	96%	93%	94%	99%
GaussianProcess	91%	92%	87%	90%	93%
PassiveAggressive	81%	75%	82%	79%	NaN
Ridge	88%	93%	77%	85%	NaN
SGD	90%	88%	88%	88%	NaN
GaussianNB	82%	72%	94%	82%	93%
NearestCentroid	89%	90%	85%	87%	NaN
MLP	95%	96%	93%	95%	98%
NuSVC	89%	96%	78%	86%	96%
ExtraTree	88%	86%	85%	86%	88%

As show in Table 2 Removing data leakage paints a more accurate picture of each model's genuine performance and reliability. The sharp reduction in metrics, as seen with the DT model, where accuracy drops from 92% to 70%, underscores the model's sensitivity to data integrity and its tendency to overfit. This drop in performance is crucial as it alerts practitioners to the potential pitfalls of using this model without rigorous testing for overfitting. On the other hand, while maintaining a high precision of 99%, the GB model suffers a significant reduction in recall from 91% to 49%. This suggests that while GB can still correctly identify a high percentage of positive cases, it misses many other positive cases when trained without leakage. This behavior could be detrimental in practical applications were failing to detect true positives can have serious implications, thus emphasizing the risk of over-relying on a single performance metric, such as precision in the presence of data leakage.

The performance of KNN and SVM showcases minimal variation between the scenarios with and without data leakage, highlighting their robustness. KNN's resilience might be attributed to its method of directly leveraging the data's inherent structure, making it less susceptible to the kinds of overfitting that more complex models might suffer from. Similarly, SVM's ability to effectively manage high-dimensional data spaces and mitigate overfitting risks makes it a dependable option in environments where data integrity might be compromised.

Furthermore, RF and LR consistently maintain high-performance metrics, indicating their robust nature against data leakage. RF benefits from its ensemble approach, where multiple decision trees contribute to the final outcome, thus reducing the risk of overfitting any single data artefact. LR’s strong statistical foundation allows it to model relationships with a high degree of reliability, making it suitable for applications requiring stringent data sensitivity.

Conversely, AdaBoost exhibits a dramatic decrease in recall and F1 score when data leakage is rectified, marking it as highly susceptible to leakage effects. This dramatic fluctuation underscores the need for caution when deploying AdaBoost in real-world settings where data leakage can substantially compromise its performance. Ensuring the absence of leakage becomes critical for maintaining the model's effectiveness.

Lastly, models like ExtraTrees and GaussianProcess, which continue to perform robustly even in the absence of data leakage, suggest an inherent resistance to such issues. However, this observation does not negate the importance of thorough data preprocessing. Ensuring data integrity is crucial as even these robust models can occasionally be misled by subtle nuances in leaked data.

Table 2 Results of Experiments on Spambase Without Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	70%	65%	65%	65%	69%
GB	78%	99%	49%	66%	98%

KNN	90%	92%	83%	87%	94%
SVM	93%	95%	88%	92%	98%
RF	92%	98%	82%	90%	98%
LR	91%	94%	85%	89%	97%
LDA	88%	94%	75%	84%	96%
QDA	84%	76%	92%	83%	94%
AdaBoost	74%	99%	40%	57%	97%
Bagging	88%	96%	75%	84%	95%
ExtraTrees	94%	98%	88%	93%	98%
GaussianProcess	92%	94%	87%	91%	93%
PassiveAggressive	90%	90%	86%	88%	NaN
Ridge	88%	94%	75%	84%	NaN
SGD	90%	92%	85%	88%	NaN
GaussianNB	83%	74%	93%	83%	94%
NearestCentroid	89%	90%	83%	86%	NaN
MLP	93%	98%	86%	92%	98%
NuSVC	89%	96%	77%	85%	96%
ExtraTree	77%	79%	62%	70%	75%

4.1. Breast Cancer Dataset

The experiments conducted on the Breast Cancer dataset with data leakage (Table 3) demonstrate that many ML models exhibit high-performance metrics, suggesting excellent predictive capabilities. For example, models like DT and SVM displayed accuracy above 94%, which could give clinicians and researchers undue confidence in their predictive accuracy. However, these inflated metrics may not accurately represent the model's real-world effectiveness, as they are likely influenced by the leakage of data between the training and testing phases. This scenario highlights the potential risks associated with data leakage, where models may appear more predictive than they are, leading to possible over-reliance in critical medical decision-making contexts.

Table 3 Results of Experiments on Breast Cancer with Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	94%	91%	93%	92%	94%
GB	95%	93%	93%	93%	100%
KNN	95%	93%	93%	93%	98%
SVM	98%	100%	95%	98%	100%
RF	96%	98%	93%	95%	100%
LR	97%	98%	95%	96%	100%
LDA	96%	98%	91%	94%	99%
QDA	96%	93%	95%	94%	99%
AdaBoost	95%	93%	93%	93%	99%
Bagging	95%	93%	93%	93%	99%
ExtraTrees	96%	98%	93%	95%	100%
GaussianProcess	95%	95%	91%	93%	96%

PassiveAggressive	96%	93%	98%	95%	NaN
Ridge	96%	98%	93%	95%	NaN
SGD	96%	91%	98%	94%	NaN
GaussianNB	96%	98%	93%	95%	100%
NearestCentroid	96%	98%	91%	94%	NaN
MLP	97%	98%	95%	96%	99%
NuSVC	96%	98%	91%	94%	99%
ExtraTree	92%	87%	93%	90%	92%

As depicted in Table 4 when data leakage is controlled, the true capabilities of the models are more accurately reflected, showing some reductions in performance metrics but still maintaining high effectiveness. The DT model, for instance, shows a decrease in accuracy from 94% to 90%. Although this drop might seem modest, it is significant in the context of medical diagnostics, where each percentage point can impact patient outcomes. This demonstrates that while DT is a simple and interpretable model, it can overfit the training data, leading to inflated performance when leakage is present.

Conversely, models like GB and KNN exhibit remarkable resilience to changes in data leakage conditions. GB, for instance, maintains a high level of precision and a slight variation in recall, suggesting that its learning approach, which sequentially corrects the errors of weak learners, is less prone to overfitting. KNN shows only marginal variations in performance, highlighting its stability, which likely stems from its reliance on the data's inherent structure rather than learning from abstract features, making it robust against data leakage.

SVM and LR continue to display almost identical performance in scenarios with and without data leakage, underscoring their robustness and the effectiveness of their methodologies in handling high-dimensional data spaces and maintaining performance consistency, even in potentially compromised data environments. Interestingly, the RF and LDA also show high tolerance to data leakage. RF's ensemble approach, which aggregates predictions from multiple decision trees, helps mitigate the risk of overfitting to leaked data, while LDA's reliance on statistical foundations provides stability against varying data conditions.

The AdaBoost model maintains consistent performance across both scenarios, indicating potential resistance to the impacts of data leakage. However, it's crucial to note that while AdaBoost may not show drastic performance changes, subtle biases introduced during data preprocessing can still influence outcomes, emphasizing the need for careful data handling.

Lastly, the ExtraTrees model demonstrates a slight improvement in performance metrics when data leakage is absent, suggesting that its methodology might benefit from training on data that is free from contamination. This minor improvement is an encouraging sign that rigorous data handling and preventing leakage can enhance model performance by ensuring that the learning process is based solely on the intrinsic properties of the data.

Table 4 Results of Experiments on Breast Cancer Without Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	90%	90%	84%	87%	89%
GB	95%	95%	91%	93%	99%
KNN	96%	95%	93%	94%	98%
SVM	98%	100%	95%	98%	99%
RF	96%	98%	93%	95%	99%
LR	98%	100%	95%	98%	100%

LDA	94%	97%	86%	91%	99%
QDA	93%	89%	93%	91%	97%
AdaBoost	95%	93%	93%	93%	99%
Bagging	96%	98%	91%	94%	96%
ExtraTrees	97%	98%	95%	96%	100%
GaussianProcess	95%	95%	91%	93%	95%
PassiveAggressive	98%	100%	95%	98%	NaN
Ridge	94%	95%	88%	92%	NaN
SGD	96%	100%	88%	94%	NaN
GaussianNB	96%	98%	91%	94%	100%
NearestCentroid	95%	97%	88%	93%	NaN
MLP	97%	98%	95%	96%	99%
NuSVC	96%	98%	91%	94%	99%
ExtraTree	96%	98%	91%	94%	95%

4.2. Tic-Tac-Toe Endgame Dataset

The results from the Tic-Tac-Toe Endgame dataset with data leakage (Table 5) present an interesting scenario where several models exhibit high-performance metrics that may not accurately reflect their true predictive abilities. For example, GB and RF both display near-perfect recall and F1 scores, potentially giving an illusion of flawless performance. However, such metrics could be the result of data leakage, where information from the test set has inadvertently influenced the training process. This skew in data can lead to overly optimistic performance evaluations, which may not hold in a truly independent testing scenario.

Models like LR and LDA show significantly lower performance metrics compared to other models, with accuracies of only 69%. These results may indicate these models' greater sensitivity to the quality of data and their potential inability to handle leaked data effectively.

Table 5 Results of Experiments on Tic-Tac-Toe Endgame with Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	87%	89%	91%	90%	85%
GB	93%	90%	100%	95%	99%
KNN	86%	84%	98%	90%	82%
SVM	88%	84%	100%	92%	97%
RF	94%	91%	100%	95%	100%
LR	69%	69%	96%	80%	61%
LDA	69%	69%	95%	80%	62%
QDA	76%	76%	91%	83%	81%
AdaBoost	78%	77%	94%	85%	85%
Bagging	94%	91%	100%	95%	100%
ExtraTrees	93%	91%	98%	95%	99%
GaussianProcess	79%	76%	99%	86%	93%
PassiveAggressive	64%	72%	73%	72%	NaN
Ridge	69%	69%	96%	80%	NaN

SGD	62%	67%	81%	73%	NaN
GaussianNB	71%	69%	98%	81%	72%
NearestCentroid	60%	74%	60%	66%	NaN
MLP	93%	92%	98%	95%	99%
NuSVC	90%	87%	100%	93%	99%
ExtraTree	78%	84%	82%	83%	76%

On the other hand, when the models are evaluated without data leakage (Table 6), the overall performance metrics are slightly adjusted, indicating a more realistic estimation of each model's ability to generalize. Notably, DT shows a slight improvement in performance, which is uncommon; typically, removing data leakage results in lower performance metrics. This could suggest that DT, in this specific dataset, is robust enough to perform slightly better when not influenced by leaked data.

The performance of models such as GB, RF, and SVM remains stable, even without data leakage, which underscores their robustness and effectiveness in handling this particular game outcome prediction task without being misled by additional data cues. These models demonstrate their capability to maintain high accuracy and recall, crucial for predicting game outcomes reliably.

However, other models like SGD show a notable decrease in performance, which could be indicative of these models' reliance on patterns that may only be present when data leakage occurs. This significant drop in metrics like recall and accuracy for Passive Aggressive—from 73% recall with leakage to 49% without—points to potential overfitting when trained with leaked data.

Table 6 Results of Experiments on Tic-Tac-Toe Endgame Without Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	88%	89%	93%	91%	86%
GB	93%	90%	100%	95%	99%
KNN	83%	80%	99%	89%	83%
SVM	90%	86%	100%	93%	97%
RF	94%	91%	100%	95%	100%
LR	68%	68%	94%	79%	62%
LDA	67%	68%	93%	79%	62%
QDA	76%	76%	92%	83%	81%
AdaBoost	78%	77%	94%	85%	85%
Bagging	95%	94%	99%	96%	99%
ExtraTrees	92%	89%	99%	94%	99%
GaussianProcess	80%	77%	99%	86%	92%
PassiveAggressive	51%	66%	49%	56%	NaN
Ridge	69%	69%	95%	80%	NaN
SGD	56%	66%	66%	66%	NaN
GaussianNB	71%	70%	98%	82%	72%
NearestCentroid	62%	77%	59%	67%	NaN
MLP	93%	92%	97%	95%	99%
NuSVC	91%	87%	100%	93%	99%
ExtraTree	81%	84%	88%	86%	78%

4.3. German Credit Scoring Dataset

The German Credit Scoring dataset's analysis with data leakage (Table 7) illustrates various ML models performing under potentially misleading conditions. For instance, models like SVM and LR display high recall rates, suggesting they can successfully identify the most positive cases. However, these high recall rates might not accurately reflect the models' true capabilities in a strictly controlled environment without leakage, where no information from the test set is available during training. This situation can lead to overly optimistic evaluations of model performance, which is problematic in credit scoring, where decisions directly affect financial outcomes.

Furthermore, the DT and Bagging models show relatively low overall performance, with accuracy around 52% to 54%. This indicates that these models might struggle with the dataset complexity or that the leakage does not significantly benefit their performance, unlike other models.

Table 7 Results of Experiments on German Credit Scoring with Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	54%	68%	60%	64%	50%
GB	64%	69%	87%	77%	53%
KNN	63%	68%	85%	76%	46%
SVM	68%	68%	100%	81%	54%
RF	67%	69%	93%	79%	50%
LR	67%	68%	98%	80%	47%
LDA	66%	67%	97%	80%	47%
QDA	60%	66%	84%	74%	47%
AdaBoost	64%	68%	87%	76%	53%
Bagging	52%	65%	63%	64%	51%
ExtraTrees	63%	67%	88%	76%	52%
GaussianProcess	62%	68%	82%	74%	52%
PassiveAggressive	58%	69%	69%	69%	NaN
Ridge	67%	68%	99%	80%	NaN
SGD	56%	68%	68%	68%	NaN
GaussianNB	63%	66%	92%	77%	45%
NearestCentroid	47%	65%	46%	54%	NaN
MLP	54%	64%	72%	68%	46%
NuSVC	61%	67%	85%	75%	46%
ExtraTree	56%	68%	66%	67%	50%

When data leakage is removed (Table 8), the performance of most models remains relatively stable but with slight adjustments that represent a more genuine evaluation of their predictive abilities. Notably, the SVM model maintains its performance metrics (68% accuracy, 100% recall), indicating its robustness and reliability in this application, regardless of the data integrity. This consistency in SVM's performance suggests that it's well-suited to the complexities of credit scoring data.

In contrast, models such as the Decision Tree and Bagging show minimal changes or slight improvements in some metrics like precision and recall, which could indicate that their initial lower performance was more about the model's limitations with the data rather than the effects of leakage.

The ExtraTrees model shows an improvement in performance metrics when data leakage is controlled, with accuracy increasing to 67% and recall to 96%. This improvement might suggest that ExtraTrees is particularly sensitive to the quality of data preprocessing and can perform better when the dataset is well-curated and free from leakage.

Table 8 Results of Experiments on German Credit Scoring Without Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	54%	67%	63%	65%	48%
GB	67%	70%	89%	78%	49%
KNN	60%	66%	84%	74%	46%
SVM	68%	68%	100%	81%	54%
RF	66%	68%	93%	79%	49%
LR	67%	68%	99%	80%	48%
LDA	67%	68%	99%	80%	48%
QDA	61%	67%	85%	75%	47%
AdaBoost	63%	67%	88%	76%	49%
Bagging	56%	68%	66%	67%	50%
ExtraTrees	67%	68%	96%	80%	52%
GaussianProcess	61%	67%	82%	74%	51%
PassiveAggressive	66%	69%	90%	78%	NaN
Ridge	68%	68%	100%	81%	NaN
SGD	62%	66%	89%	76%	NaN
GaussianNB	63%	66%	93%	77%	45%
NearestCentroid	48%	65%	50%	56%	NaN
MLP	59%	67%	76%	71%	45%
NuSVC	61%	67%	84%	75%	47%
ExtraTree	57%	67%	71%	69%	48%

4.4. Student Performance Dataset

The analysis of the Student Performance dataset with data leakage (Table 9) shows several models performing exceptionally well, suggesting high predictive accuracy. For example, models like LR, SVM, and LDA exhibit accuracies and precision rates in the low to mid-90s percentile. These high rates might lead educators and administrators to believe that the models are highly reliable in predicting student performance. However, this could be misleading if the data used in training these models included information inadvertently leaked from the test set, leading to overly optimistic performance estimates.

Moreover, models such as DT and GaussianNB show comparatively lower performance metrics, which might indicate either a lack of robustness to data leakage or simply lower effectiveness with the type of data used in this educational context.

Table 9 Results of Experiments on Student Performance with Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	85%	84%	87%	85%	84%
GB	93%	93%	93%	93%	97%
KNN	78%	77%	82%	80%	86%

SVM	93%	94%	91%	93%	98%
RF	90%	93%	88%	90%	97%
LR	93%	95%	90%	93%	99%
LDA	94%	95%	92%	94%	98%
QDA	93%	95%	90%	93%	98%
AdaBoost	92%	94%	90%	92%	98%
Bagging	90%	92%	89%	90%	96%
ExtraTrees	85%	87%	83%	85%	94%
GaussianProcess	90%	93%	88%	90%	96%
PassiveAggressive	78%	85%	70%	76%	NaN
Ridge	94%	95%	92%	94%	NaN
SGD	91%	88%	94%	91%	NaN
GaussianNB	67%	71%	62%	66%	70%
NearestCentroid	66%	71%	58%	64%	NaN
MLP	91%	91%	90%	91%	98%
NuSVC	93%	94%	91%	93%	98%
ExtraTree	78%	79%	80%	79%	78%

As indicated in Table 10 Removing data leakage provides a more realistic view of each model's ability to predict student performance. Notably, there is a general trend of slight decreases in performance across most models, which is more indicative of their true capabilities. For instance, the DT model shows a decrease in accuracy from 85% to 82%, and its recall drops from 87% to 75%. This reduction could highlight the model's vulnerability to overfitting with leaked data and its true effectiveness when such biases are removed.

On the other hand, models like SVM and LR still maintain high-performance metrics with only minimal changes. This consistency suggests that these models are robust and reliable for predicting student performance, even in the absence of data leakage. Their ability to perform well without reliance on potentially biased training data makes them suitable choices for educational data analysis. Interestingly, the ExtraTrees model shows a decrease in accuracy and recall, indicating it may benefit from more stringent data handling or may be sensitive to the specific way data is structured or processed in the absence of leakage.

Table 10 Results of Experiments on Student Performance Without Leakage.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
DT	82%	88%	75%	81%	82%
GB	89%	91%	88%	89%	96%
KNN	77%	77%	80%	79%	84%
SVM	91%	92%	90%	91%	97%
RF	86%	87%	87%	87%	94%
LR	91%	96%	86%	90%	98%
LDA	91%	96%	87%	91%	98%
QDA	92%	96%	88%	92%	98%
AdaBoost	90%	93%	87%	90%	96%
Bagging	85%	89%	81%	85%	93%
ExtraTrees	84%	84%	86%	85%	93%

GaussianProcess	87%	88%	88%	88%	95%
PassiveAggressive	85%	82%	90%	86%	NaN
Ridge	91%	96%	87%	91%	NaN
SGD	84%	87%	81%	84%	NaN
GaussianNB	65%	68%	64%	66%	69%
NearestCentroid	66%	69%	62%	65%	NaN
MLP	88%	91%	86%	88%	97%
NuSVC	90%	91%	89%	90%	97%
ExtraTree	71%	73%	69%	71%	71%

4.5. Comparative Analysis Across Datasets

Our extensive research into the impact of data leakage across diverse datasets such as Spambase, Breast Cancer, Tic-Tac-Toe Endgame, German Credit Scoring, and Student Performance has yielded critical insights into how data integrity affects ML models. The investigation illustrates a recurrent pattern where data leakage tends to inflate performance metrics, often misleadingly enhancing a model's perceived effectiveness. Such distortions can result in flawed decision-making in practical applications, where models are expected to perform with high accuracy and reliability. In our study, we methodically assessed the performance of of the selected ML algorithms to determine how data leakage during preprocessing might inflate their accuracy. To capture this quantitatively, we employed the following formulas:

$$\text{Average Accuracy with Leakage}_{\text{algorithm}} = \frac{1}{d}\sum_{i=1}^{d} a_{l,i}$$

$$\text{Average Accuracy without Leakage}_{\text{algorithm}} = \frac{1}{d}\sum_{i=1}^{d} a_{nl,i}$$

Where:

- d: This is the number of datasets used in the calculation.
- i: This is an index used in the summation process. It iterates over each dataset from 1 to d, allowing each dataset's accuracy to contribute to the total sum.
- $a_{l,i}$: This parameter represents the accuracy of a given algorithm on the i-th dataset when there is data leakage.
- $a_{nl,i}$: denotes the accuracy of the algorithm on the i-th dataset but without any data leakage.

Figure 2 offers a comprehensive analysis of ML models' performance, contrasting scenarios with and without data leakage. A clear pattern emerges from the Figure 2 is that models generally exhibit inflated accuracy when trained on datasets with leakage.

Our analysis reveals that the DT algorithm's accuracy suffers a substantial reduction of 4.25% when transitioning from conditions with data leakage to those without, suggesting a high sensitivity to overfitting due to leaked data.

For GB and KNN, the accuracy drop is more contained, at 2.3% and 3.5% respectively, indicating a better handling of data leakage. This could be attributed to the sequential nature of error correction in GB and the local data dependency inherent in KNN's methodology.

The SVM and RF algorithms stand out with their negligible leakage impacts of only 1% and 0.5%. Such minor fluctuations in accuracy demonstrate their robustness and the effectiveness of their underlying methodologies in safeguarding against the risks posed by data leakage.

In the realm of algorithms with strong regularization techniques, LR shows a mere 0.25% impact from data leakage, and LDA demonstrates no discernible impact at all, highlighting their potential for consistent performance even in the presence of data leakage.

Conversely, the PA algorithm exhibits a notable vulnerability to the quality of training data, with an accuracy drop reflecting a 2.75% leakage impact. This underlines the algorithm's tendency to overfit when faced with data leakage, making it a less reliable choice in scenarios where data integrity is questionable.

Additionally, the GaussianNB and NC models, with leakage impacts of 1.75% and 3.25%, manifest substantial performance dips when data leakage is accounted for. This indicates that their performance metrics are likely to be inflated when leakage is present, and it emphasizes the need for vigilant data handling and preprocessing when utilizing these models.

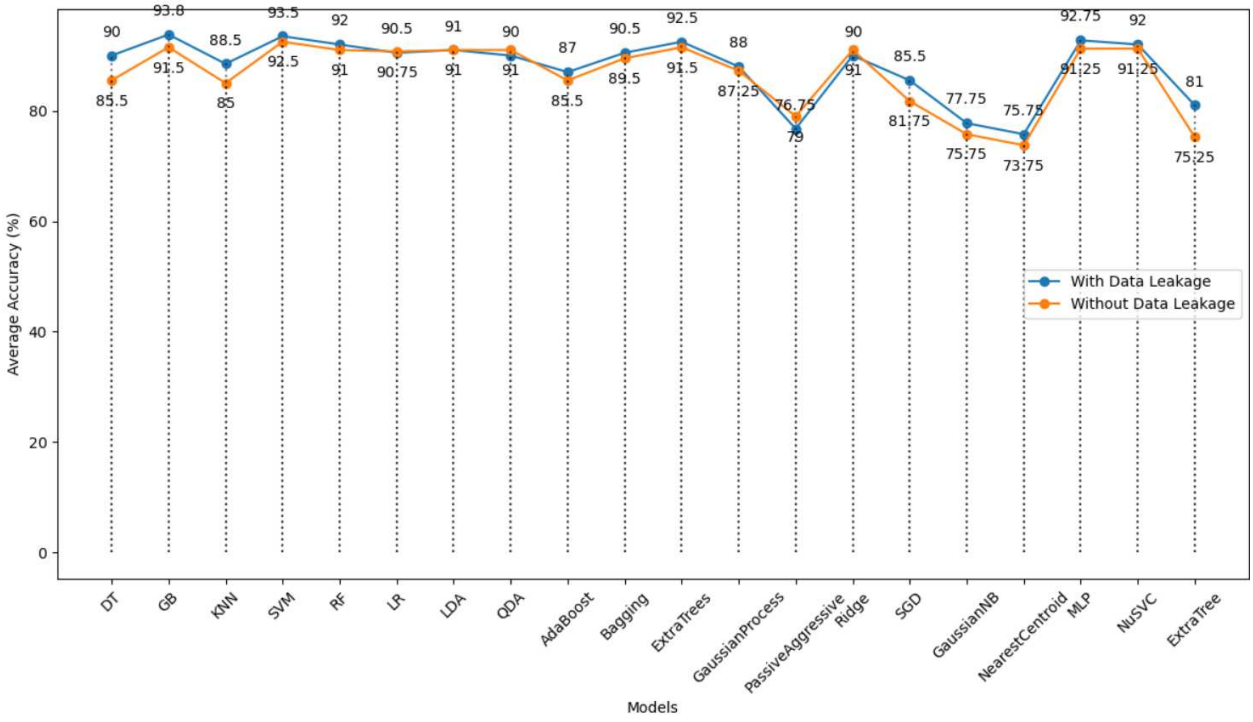

Figure 2 Comparison of Model Accuracy with and Without Data Leakage Across Datasets.

Moreover, Figure 3 provides a visual representation of the absolute raw data leakage percentages across different datasets, reflecting the disparity in model accuracy due to data leakage. The Figure shows that the Spambase dataset experienced the most significant impact from data leakage at 6.79%, followed by the Student Performance dataset at 3.55%, Breast Cancer at 1.99%, Tic-Tac-Toe at 1.92%, and German Credit with the least impact at 0.47%.

These raw datasets leakages are calculated using the raw leakage formula:

$$D_{Raw\ Leakage} = \left(\frac{1}{n} \sum_{i=1}^{n} \text{Accuracy}_{\text{with leakage},i} \right) - \left(\frac{1}{n} \sum_{i=1}^{n} \text{Accuracy}_{\text{without leakage},i} \right)$$

Here, $D_{Raw\ Leakage}$ represents the average raw leakage for a given dataset, n is the number of algorithms, and i indexes each algorithm. The term $\text{Accuracy}_{\text{with leakage},i}$ is the accuracy of the i-th algorithm with data leakage, and $\text{Accuracy}_{\text{without leakage},i}$ is the accuracy of the i-th algorithm without data leakage. This formulation will provide the average change in accuracy, or leakage, per algorithm.

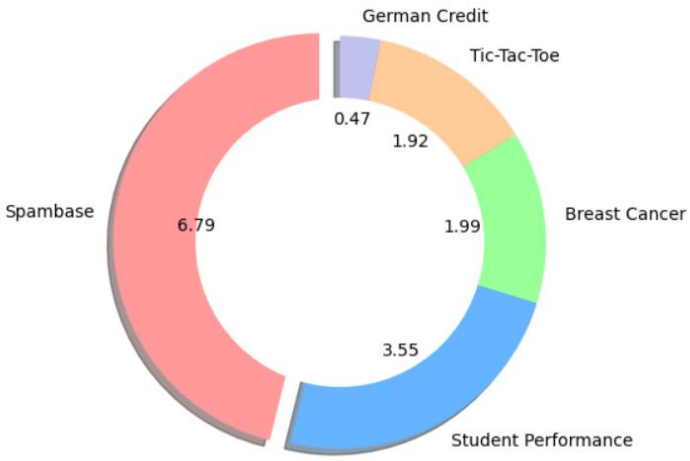

Figure 3 Raw Data Leakage for Each Dataset (%).

Furthermore, The Spambase dataset's notably high percentage suggests that models trained on this dataset may be particularly vulnerable to overfitting due to data leakage, potentially compromising their performance in real-world applications. This level of sensitivity may be indicative of either the dataset's complexity or the models' reliance on features that are not generalizable.

The Student Performance dataset's relatively high leakage impact might suggest that educational data, with its nuanced and non-linear relationships, might be more susceptible to overfitting when leakage occurs. The extent to which data leakage inflates performance metrics in this domain is a cause for concern, particularly when these models are used for critical decisions regarding student evaluation and curriculum adjustments.

In contrast, the German Credit dataset shows a minimal change in model accuracy with and without data leakage, implying that models applied to this dataset are either well-regulated against leakage or that the dataset itself is less prone to the pitfalls of data leakage.

Our investigation into data leakage and its impact on ML model training concludes with significant findings. The meticulously quantified extent of leakage impact emphasizes the importance of considering data leakage during the training phase of ML algorithms. The performance variations observed across different algorithms underscore their varying capacities to handle data with potential integrity issues. This insight is invaluable for practitioners who aim to develop robust, leakage-aware models, ensuring that their ML insights remain reliable and uncorrupted, especially critical in domains where decision-making relies heavily on predictive accuracy.

The moderate leakage impacts observed in the Breast Cancer and Tic-Tac-Toe datasets, while not as pronounced as in the Spambase dataset, still highlight the need for careful data management. To maintain the integrity of the models, practitioners must pay close attention to how they preprocess and handle data across various scenarios.

In essence, understanding the specific characteristics of datasets and how they interact with model vulnerabilities to data leakage is vital. Such awareness enables practitioners to implement targeted measures to counteract the effects of data leakage, ensuring the deployment of effective and trustworthy models. The integrity of data and the rigorous validation of model performance stand as pillars for advancing the field of ML, fostering trust in its applications and outcomes.

5. Key Observations from the Study

Understanding the impact of data leakage on ML model performance is crucial for ensuring the reliability and effectiveness of predictive analytics. Data leakage occurs when information from outside the training dataset inadvertently influences model training, leading to inflated performance estimates. In this study, we investigate the sensitivity of various ML models to data leakage and examine how it affects their performance across different datasets. By analyzing model behavior with and without data leakage, we aim to provide insights into the robustness of models under realistic conditions and offer recommendations for mitigating the risks associated with data leakage in practical applications.

The study revealed varying degrees of sensitivity to data leakage among different ML models. Notably, Decision Trees and AdaBoost exhibited high sensitivity, as evidenced by significant performance drops when data leakage was addressed. For instance, in the Spambase dataset, Decision Trees saw a drastic reduction in accuracy from 92% to 70%. This consistent pattern across datasets suggests a tendency for these models to overfit when trained with leaked data, raising concerns about their reliability in compromised data integrity environments. In contrast, models like SVM and RF demonstrated remarkable robustness in performance regardless of data leakage. Across various datasets, including those with complex structures like Breast Cancer and Tic-Tac-Toe Endgame, SVM and RF maintained consistently high-performance metrics. Their resilience underscores their suitability for scenarios where preventing data leakage presents challenges, making them reliable choices for critical applications.

The impact of data leakage varied significantly across different datasets, reflecting the unique characteristics and challenges inherent in each. Even robust models like SVM and RF exhibited some performance variability in more complex datasets, such as those used for credit scoring and medical diagnostics. This highlights the importance of tailored data preprocessing and model selection strategies adapted to the specific requirements and nuances of each dataset. A notable discrepancy in performance metrics, particularly recall and precision, was observed when data leakage was controlled. For instance, Gradient Boosting's recall dropped dramatically from 91% to 49% in the Spambase dataset upon the removal of data leakage, indicating potential challenges in detecting all relevant instances without the bias introduced by leaked data.

Given the observed sensitivities and disparities, the findings underscore the necessity for rigorous validation mechanisms to ensure that models are evaluated and tested under conditions that closely resemble real-world scenarios. This approach is crucial to prevent the deployment of models that only perform optimally under idealized or biased training conditions. In light of the findings, strategic model selection becomes imperative. Models demonstrating stable performance across various conditions, such as SVM and RF, should be prioritized for applications where data integrity concerns are paramount.

To address these challenges, increased awareness and understanding of data leakage within the ML community are essential. Educational initiatives focusing on the risks and mitigation strategies related to data leakage are crucial. Furthermore, establishing regulatory guidelines that mandate the testing of models for data leakage before deployment can ensure the reliability and trustworthiness of ML applications.

6. Conclusion

In conclusion, our research has provided a critical examination of the pervasive issue of data leakage in ML models across a variety of datasets. Through meticulous experimentation and analysis, we have quantified the impact of data leakage, unveiling that its influence on model performance is both significant and varied across different ML algorithms and datasets.

The findings reveal that while some models like SVM and RF exhibit a strong defense against data leakage, with minimal performance degradation, others such as DT and AdaBoost are highly susceptible, showing considerable drops in performance when data leakage is eliminated. This underlines the necessity for practitioners to consider the choice of models carefully, especially in sensitive applications where the integrity of predictions is paramount.

The extent of raw data leakage observed—6.79% for Spambase, 3.55% for Student Performance, 1.99% for Breast Cancer, 1.92% for Tic-Tac-Toe Endgame, and 0.47% for German Credit—highlights the differing levels of

vulnerability across datasets. Such variability calls for tailored strategies in data preprocessing and model selection to mitigate the risk of data leakage and its consequent impact on model reliability.

Our study reinforces the importance of understanding and addressing data leakage to prevent the overestimation of model performance. It also underscores the need for transparent and rigorous validation mechanisms to evaluate ML models accurately. By drawing attention to these findings, we aim to foster a more leakage-conscious culture within the ML community, promoting the development and deployment of reliable, effective models.

Looking forward, it is crucial for the field to continue exploring innovative solutions and best practices for preventing data leakage. Further research into data handling techniques and model robustness can contribute to the enhancement of ML as a reliable tool for decision-making across diverse domains. The implications of our study emphasize the ongoing challenge and necessitate a collective effort to ensure the trustworthiness and accuracy of ML in real-world applications.

Declarations

Funding statement

Not applicable.

Conflict of Interest Statement:

The authors of this study declare that there are no conflicts of interest related to this research work.

Data availability statement

Data included in article/supplementary material/referenced in the article.

Additional information

No additional information is available for this paper.

References:

[1] M. A. Bouke, A. Abdullah, J. Frnda, K. Cengiz, and B. Salah, "BukaGini: A Stability-Aware Gini Index Feature Selection Algorithm for Robust Model Performance," *IEEE Access*, vol. 11, pp. 59386–59396, 2023, doi: 10.1109/ACCESS.2023.3284975.

[2] T. Dokeroglu, A. Deniz, and H. E. Kiziloz, "A Comprehensive Survey on Recent Metaheuristics for Feature Selection," *Neurocomputing*, vol. 494, pp. 269–296, 2022, doi: 10.1016/j.neucom.2022.04.083.

[3] N. Pilnenskiy and I. Smetannikov, "Modern Implementations of Feature Selection Algorithms and Their Perspectives," *Conf. Open Innov. Assoc. Fruct*, pp. 250–256, 2019, doi: 10.23919/FRUCT48121.2019.8981498.

[4] M. A. Bouke, A. Abdullah, K. Cengiz, and S. Akleylek, "Application of BukaGini algorithm for enhanced feature interaction analysis in intrusion detection systems," *PeerJ Comput. Sci.*, vol. 10, p. e2043, Apr. 2024, doi: 10.7717/peerj-cs.2043.

[5] M. Refaat, *Data preparation for data mining using SAS*. Elsevier, 2010.

[6] D. Pyle, *Data preparation for data mining*. Morgan Kaufmann, 1999.

[7] J. Gawlikowski *et al.*, "A Survey of Uncertainty in Deep Neural Networks," 2021, Accessed: Aug. 20, 2022. [Online]. Available: http://arxiv.org/abs/2107.03342

[8] P. Rathore, A. Basak, S. H. Nistala, and V. Runkana, "Untargeted, Targeted and Universal Adversarial Attacks and Defenses on Time Series," Jan. 2021, doi: 10.1109/ijcnn48605.2020.9207272.

[9] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "ML-Leaks: Model and Data Independent

Membership Inference Attacks and Defenses on Machine Learning Models," Jun. 2018, doi: 10.48550/arxiv.1806.01246.

[10] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in Context-Dependent Deep Neural Networks for conversational speech transcription," *2011 IEEE Work. Autom. Speech Recognit. Understanding, ASRU 2011, Proc.*, no. December, pp. 24–29, 2011, doi: 10.1109/ASRU.2011.6163899.

[11] M. A. Bouke and A. Abdullah, "An empirical assessment of ML models for 5G network intrusion detection: A data leakage-free approach," *e-Prime - Adv. Electr. Eng. Electron. Energy*, vol. 8, no. May, p. 100590, 2024, doi: 10.1016/j.prime.2024.100590.

[12] M. Bouke, A. Abdullah, N. Udzir, and N. Samian, "Overcoming the Challenges of Data Lack, Leakage, and Dimensionality in Intrusion Detection Systems: A Comprehensive Review," *J. Commun. Inf. Syst.*, vol. 39, no. 2024, pp. 22–34, Jan. 2024, doi: 10.14209/jcis.2024.3.

[13] J. Brownlee, "Machine Learning Mastery With Python: Data Cleaning, Feature Selection, and Data Transforms in Python," p. 500, 2020.

[14] Q. Dong, "Leakage Prediction in Machine Learning Models When Using Data from Sports Wearable Sensors," *Comput. Intell. Neurosci.*, vol. 2022, 2022, doi: 10.1155/2022/5314671.

[15] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, p. 100804, 2023, doi: 10.1016/j.patter.2023.100804.

[16] A. Stock, E. J. Gregr, and K. M. A. Chan, "Data leakage jeopardizes ecological applications of machine learning," *Nat. Ecol. Evol.*, vol. 7, no. 11, pp. 1743–1745, 2023, doi: 10.1038/s41559-023-02162-1.

[17] F. Farokhi and M. A. Kaafar, "Modelling and Quantifying Membership Information Leakage in Machine Learning," pp. 1–13, 2020, [Online]. Available: http://arxiv.org/abs/2001.10648

[18] W. Zhang, S. Tople, and O. Ohrimenko, "Leakage of dataset properties in multi-party machine learning," *Proc. 30th USENIX Secur. Symp.*, pp. 2687–2704, 2021.

[19] A. Hannun, C. Guo, and L. van der Maaten, "Measuring Data Leakage in Machine-Learning Models with Fisher Information (Extended Abstract)," *IJCAI Int. Jt. Conf. Artif. Intell.*, no. Uai, pp. 5284–5288, 2022, doi: 10.24963/ijcai.2022/736.

[20] S. Kapoor *et al.*, "REFORMS: Reporting Standards for Machine Learning Based Science," 2023, [Online]. Available: http://arxiv.org/abs/2308.07832

[21] P. Subotić, U. Bojanić, and M. Stojić, "Statically detecting data leakages in data science code," in *Proceedings of the 11th ACM SIGPLAN International Workshop on the State Of the Art in Program Analysis*, 2022, pp. 16–22.

[22] M. A. Bouke and A. Abdullah, "An empirical study of pattern leakage impact during data preprocessing on machine learning-based intrusion detection models reliability," *Expert Syst. Appl.*, vol. 230, no. June, p. 120715, Nov. 2023, doi: 10.1016/j.eswa.2023.120715.

[23] B. E. Gibney, "Is AI Fuelling a Reproducibility Crisis in Science?," *Nature*, vol. 608, no. December, pp. 250, 251, 2022.

[24] B. Koçak, R. Cuocolo, D. P. Dos Santos, A. Stanzione, and L. Ugga, "Must-have Qualities of Clinical Research on Artificial Intelligence and Machine Learning," *Balkan Med. J.*, vol. 40, no. 1, pp. 3–12, 2023, doi: 10.4274/balkanmedj.galenos.2022.2022-11-51.

[25] M. Rosenblatt, L. Tejavibulya, R. Jiang, S. Noble, and D. Scheinost, "Data leakage inflates prediction performance in connectome-based machine learning models," *Nat. Commun.*, vol. 15, no. 1, pp. 1–15, 2024, doi: 10.1038/s41467-024-46150-w.

[26] A. Ghodselahi, "A hybrid support vector machine ensemble model for credit scoring," *Int. J. Comput. Appl.*, vol. 17, no. 5, pp. 1–5, 2011.

[27] B. Kalantar, B. Pradhan, S. A. Naghibi, A. Motevalli, and S. Mansor, "Assessment of the effects of training data selection on the landslide susceptibility mapping: a comparison between support vector machine (SVM), logistic regression (LR) and artificial neural networks (ANN)," *Geomatics, Nat. Hazards Risk*, vol. 9, no. 1, pp. 49–69, 2018.

[28] L. Breiman, J. Friedman, R. Olshen, and C. Stone, "Cart," *Classif. Regres. trees*, 1984.

[29] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd*

international conference on knowledge discovery and data mining, 2016, pp. 785–794.

[30] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[31] "UCI Machine Learning Repository: Spambase Data Set." Accessed: May 07, 2022. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/spambase

[32] "Breast Cancer Wisconsin (Diagnostic) - UCI Machine Learning Repository." Accessed: Jul. 10, 2023. [Online]. Available: http://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic

[33] "Tic-Tac-Toe End game Dataset UCI | Kaggle." Accessed: Jul. 10, 2023. [Online]. Available: https://www.kaggle.com/datasets/aungpyaeap/tictactoe-endgame-dataset-uci

[34] "Statlog (German Credit Data) - UCI Machine Learning Repository." Accessed: Jul. 10, 2023. [Online]. Available: https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data

[35] "Student Performance - UCI Machine Learning Repository." Accessed: Jul. 10, 2023. [Online]. Available: https://archive.ics.uci.edu/dataset/320/student+performance

[36] M. L. Gordon, K. Zhou, K. Patel, T. Hashimoto, and M. S. Bernstein, "The disagreement deconvolution: Bringing machine learning performance metrics in line with reality," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–14.

[37] M. Z. Naser and A. Alavi, "Insights into performance fitness and error metrics for machine learning," *arXiv Prepr. arXiv2006.00887*, 2020.