mathematics

MDPI

Article

Class-Conditioned Diffusion Augmentation and Explainable In-Context Ensemble Learning for Imbalanced Maintenance Action Classification in Aircraft Systems

Begüm Erol

Department of Industrial Engineering, Faculty of Engineering, Architecture and Design, Istiklal Science and Technology University, Kahramanmaraş 46100, Türkiye; begum.erol@istiklal.edu.tr

Abstract

Class imbalance in aircraft predictive maintenance data causes classifiers to favor frequent actions and miss rare but operationally important outcomes. This study proposes a hybrid framework that combines a Class-Conditional Tabular Denoising Diffusion Probabilistic Model (TabDDPM-CC) with a k-Center Ensemble Tabular Prior-Data Fitted Network (TabPFN-CE). It was applied to a 647-record aircraft oil pump maintenance dataset with four classes and an imbalance ratio of 9.0. TabDDPM-CC was compared on fidelity, privacy and utility with ten augmentation methods, among them state-of-the-art diffusion models. TabPFN-CE combined k-center greedy context selection with posterior averaging and was compared with eleven classifiers. Ablations, paired statistical tests and noise and shift experiments examined each design choice. The proposed pipeline raised balanced accuracy from 0.454 to 0.841 and G-mean from 0.439 to 0.840. After Holm–Bonferroni correction, its advantage was significant against most comparators but not against SMOTE, three diffusion baselines or a single k-center context. Among the generator components, class conditioning had the largest effect. SHapley Additive exPlanations (SHAP) showed that the predictions rely on operationally plausible variables. On the public AI4I 2020 dataset, the framework reached a balanced accuracy of 0.901. Class-conditioned augmentation with context-aware classification can therefore support interpretable maintenance decisions on small imbalanced datasets.

Keywords: aircraft predictive maintenance; class imbalance; diffusion models; synthetic data generation; TabPFN; in-context learning; tabular deep learning; explainable artificial intelligence; ensemble learning

MSC: 68T05

Check for updates

Academic Editor: Song Fu

Received: 26 September 2026
Revised: 23 October 2026
Accepted: 2 November 2026
Published: 4 November 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Aviation is a safety-critical industry in which operational reliability must be maintained at the highest level and the tolerance for failure is close to zero. Predictive maintenance (PdM) strategies, developed to prevent unexpected failures and optimize maintenance costs, are increasingly replacing traditional failure-based or time-based approaches [1]. Advances in Industry 4.0 and Internet of Things (IoT) technologies have led to an exponential growth in the volume of sensor data collected from aircraft systems, making the use of machine learning (ML) and deep learning (DL) algorithms for fault detection and classification a practical necessity rather than an option, particularly as aerospace-specific predictive maintenance frameworks increasingly rely on such data-driven pipelines [1].

One of the most persistent bottlenecks in data-driven PdM applications is class imbalance. Because aircraft systems are designed to meet high reliability standards, the large majority of collected data corresponds to healthy operating conditions, while instances representing genuine fault states are comparatively rare. This imbalance biases machine learning models toward the majority class and leads to poor detection of the minority fault classes, which are precisely the outcomes of greatest operational interest. Traditional data-level techniques such as the Synthetic Minority Oversampling Technique (SMOTE) [2] and ADASYN [3] have been widely used to mitigate this problem. While computationally efficient, these interpolation-based methods can introduce noise and blur decision boundaries when applied to high-dimensional aircraft data with complex, non-linear feature distributions. Probabilistic mixture models such as the Bayesian Gaussian Mixture (BGM) offer an alternative through class-conditional density estimation [4], but remain sensitive to the correct specification of the number of mixture components in multi-modal distributions. To overcome these limitations, deep generative models based on Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) have been proposed in recent years. Methods such as Gaussian Copula [5], CTGAN and TVAE [6], together with their conditional extension CopulaGAN, better preserve the statistical properties of tabular data than SMOTE-type approaches. However, the training instability and mode collapse commonly reported for GAN-based models raise reliability concerns for their use in safety-critical applications.

This study proposes a framework based on Denoising Diffusion Probabilistic Models (DDPMs) to address class imbalance in aircraft maintenance data and to maximize downstream classification performance. TabDDPM [7], which adapts the diffusion paradigm originally developed for image generation to tabular data, learns the data distribution through a gradual denoising process. Therefore, it is not subject to the training instability that affects GAN-based generators, thus producing synthetic samples that remain faithful to the statistical structure of the original data. Rather than adopting the vanilla TabDDPM architecture directly, this study develops it through two cumulative stages. The first stage introduces three training-stability modifications, range clipping of synthetic outputs, schedule-independent timestep normalization, and an exponential moving average (EMA) of the denoiser weights [8], yielding an intermediate configuration denoted TabDDPM-S. The second and final stage replaces the per-class model structure with a single class-conditional denoising network, following the conditioning principle established for conditional generative models [9]. It adopts the cosine noise schedule of Nichol and Dhariwal [10] in place of the baseline linear schedule, and adds a Distance-to-Closest-Record (DCR) based quality-filtering step that removes low-fidelity synthetic candidates before they reach the downstream classifier. The final configuration is denoted TabDDPM-CC. It is compared with established augmentation methods and with two recent diffusion generators, Tabular Synthesis (TabSyn) [11] and Tabular Diffusion (TabDiff) [12].

Once the class imbalance has been addressed, the choice of classification algorithm is equally critical for accurate fault diagnosis. In the tabular data literature, ensemble methods such as Random Forest (RF) [13] and gradient boosting decision trees, including XGBoost [14], LightGBM [15] and CatBoost [16], have long been regarded as a strong baseline. Although deep neural networks have transformed image and text processing, the heterogeneous and irregular structure of tabular data has made it difficult for standard architectures to consistently outperform gradient boosting. Several tabular-specific deep learning architectures have been proposed to close this gap, including TabNet [17], which learns feature selection through sequential attention, FT-Transformer [18], which embeds numerical and categorical features into a shared token space, NODE [19], which combines differentiable decision trees with neural network layers, SAINT [20], which applies attention

both within and across rows, and the more recent GANDALF [21] and ExcelFormer [22] architectures. To establish the relative advantage of the proposed hybrid method on a solid comparative footing, all ten of these classifier families are included in the evaluation framework of this study.

The success of the Transformer architecture has begun to shift this paradigm further. Tabular Prior-Data Fitted Networks (TabPFN) [23], a Transformer trained on synthetic prior data, can compete with and in some cases outperform gradient boosting models on small- and medium-sized tabular datasets without requiring inference-time hyperparameter tuning, making it a strong candidate for complex, safety-critical datasets such as the one addressed in this study. However, because TabPFN processes all training examples within a single bounded attention window, applying it to an augmentation-expanded training set raises the question of which examples should be included in this context. This study addresses that question through a systematic comparison of four increasingly refined context-construction strategies: unmodified TabPFN with its default context, TabPFN-R with a randomly sampled context, TabPFN-KM with a context selected through k-means cluster centroids [24], and TabPFN-KC with a context selected through the k-center greedy coverage algorithm [25,26]. Building on the last of these, the proposed classifier, TabPFN-CE, combines multiple independently sampled k-center contexts through posterior averaging following a bagging-style rule [27]. This progressive comparison isolates the individual contribution of the context-selection criterion and of the ensembling step to downstream classification performance.

In this context, the present study follows an end-to-end, hybrid methodology to improve aircraft maintenance classification. The training partition of a four-class imbalanced dataset is first augmented with eleven methods from four methodological families. Seven are established benchmarks from the neighborhood-based, statistical and deep generative families. The diffusion family includes TabDDPM, the recent TabSyn and TabDiff models and the proposed TabDDPM-CC. The generated data are scored on fidelity, privacy and utility and aggregated with an average-rank framework [28] to select one representative per family. Domain-informed feature engineering then derives degradation and stress indicators from the original operational variables. The resulting representation is classified by eleven baseline classifiers and by the proposed TabPFN-CE, which is built on the original TabPFN v1 backbone. Each design choice is tested with paired Wilcoxon signed-rank tests [29] and Holm–Bonferroni correction [30] across ten random seeds. Further analyses cover the augmentation ratio, label and sensor noise, a covariate shift and the computational cost. A SHAP-based analysis [31] examines whether the model relies on operationally plausible variables. Finally, the same pipeline is applied to the public AI4I 2020 dataset [32] as a check of methodological transferability.

The core contribution of this study is the pairing of a class-conditioned diffusion generator (TabDDPM-CC) with a coverage-based in-context classifier (TabPFN-CE) for small and imbalanced maintenance tables. The remaining elements of the pipeline are established techniques that serve to build, test and explain this pairing. The specific contributions are as follows:

- Class-conditioned diffusion augmentation: TabDDPM-CC replaces the per-class models of TabDDPM with a single class-conditional denoising network. It also adds stability and DCR-based filtering steps. Cumulative and leave-one-out ablations show that class conditioning is the modification whose removal causes the largest loss in both fidelity and downstream performance.
- Coverage-based context construction for TabPFN: TabPFN-CE selects in-context examples through k-center greedy coverage and averages the posteriors of several such

contexts. It is compared with default, random and k-means contexts, and its gain is examined with three newer TabPFN backbones.

- Controlled benchmark on a real maintenance dataset: Eleven augmentation methods from four families are scored on distinguishability, privacy, fidelity and utility. The family representatives and TabDiff, the second-ranked diffusion method, are then combined with twelve classifiers under one protocol. All main comparisons are tested at the seed level with Holm correction.
- Analysis of practical reliability: A factorial design and a leave-one-feature-out test measure the role of the engineered features. Label noise, sensor noise and a covariate shift in flight hours are used to test the stability of the pipeline. The offline and deployment costs are reported separately, and SHAP attributions are read against known degradation mechanisms.

The remainder of this article is organized as follows. Section 2 reviews related work on class imbalance and tabular deep learning in predictive maintenance. Section 3 introduces the theoretical background of the baseline augmentation and classification methods, the evaluation metrics, and the SHAP methodology used throughout this study. Section 4 presents the proposed framework in detail. Section 5 reports and discusses the experimental results. Section 6 concludes the article and outlines directions for future work.

2. Related Works

This section addresses the four main axes underlying the study in a systematic way: the class imbalance problem in tabular data and the evolution of data augmentation methods, the shift from tree-based ensemble methods to Transformer architectures in tabular data classification, the quantitative assessment of synthetic data quality, and the role of explainable artificial intelligence in reliability engineering. At the end of each subsection, a comparative analysis of existing studies clarifies where this research stands within the literature.

2.1. The Class Imbalance Problem in Reliability Engineering and Its Evolution

The fact that faulty or maintenance-requiring conditions occur far less often than healthy operating conditions has been a persistent methodological challenge in the reliability engineering literature for decades. Recent deep learning studies on rotating equipment and aerospace subsystems have addressed it mainly at the model level. Examples include uncertainty-aware Bayesian fault diagnosis [33], feature filtering and mapping for aero-engine gas path data [34] and digital-twin-assisted imbalanced diagnosis [35]. Xia et al. [36] showed that digital-twin-assisted transfer learning can preserve generalization when only limited samples are available. Signal-level enhancement is a further route when fault evidence is weak. Cheng et al. [37] extended blind deconvolution to multiple vibration signals and recovered repetitive bearing fault impulses by maximizing a weighted sum of kurtosis values. Such methods strengthen weak fault signatures in raw waveforms. The present study instead works with tabular maintenance records in which only summarized sensor readings are available, so the imbalance has to be handled at the data and model levels.

Oil pumps and hydraulic subsystems feed critical functions such as flight control surfaces, landing gear, and braking systems. Early and accurate classification of wear-based degradation in these components is a safety-critical requirement in civil aviation. Guo et al. [38] emphasized that wear in the friction pairs of piston-type hydraulic pumps in civil aircraft, together with oil cleanliness, directly determines pump reliability, and identified fault diagnosis and health management as critical technology areas for these components going forward. The closest application to the present problem combined a conditional variational autoencoder with ADASYN for aircraft fuel pump fault diagnosis and improved minority-class recognition [39]. It considers few classes, however, and does

not report synthetic data quality metrics. The studies in [33,35,36,39] share an attempt to resolve the imbalance problem at the architectural level, without a systematic comparison of augmentation methods at the data level. A review of fault diagnosis under long-tailed distributions noted that data-level and algorithm-level solutions have largely progressed as separate streams [40]. A hybrid SMOTE and Broad Learning System framework for aero-engine bearings [41] also remains limited to a single augmentation method and does not evaluate diffusion-based alternatives.

2.2. The Evolution of Data Augmentation Methods for Tabular Data

Studies seeking data-level solutions fall into four conceptual generations. The first generation consists of neighborhood-based interpolation methods such as SMOTE [2] and ADASYN [3]. In rotating machinery fault diagnosis, GAN-based augmentation was reported to blur the decision boundary less than neighborhood-based methods and to produce more realistic minority-class representations [42]. This finding is particularly relevant for components like oil pumps, where maintenance action classes represent wear regimes that are physically close to one another. The risk that neighborhood-based interpolation generates unrealistic intermediate samples between such physically overlapping classes may directly conflict with the progressive wear processes described by Guo et al. [38].

A recent review grouped GAN-based approaches to imbalanced fault diagnosis into conditional GANs, Wasserstein GANs and self-attention variants [43]. Among conditional GAN architectures specific to tabular data, the most established are CTGAN and TVAE, proposed by Xu et al. [6]. By handling mixed-type variables through mode-specific normalization, this work substantially reduced the classic mode collapse problem in GAN-based models. Borisov et al. [44] presented a comprehensive taxonomy spanning these two generations, showing that deep learning methods specific to tabular data can be grouped into three main categories: data transformation, architecture, and regularization based.

The fourth generation, and the one this study builds on, consists of diffusion-based generative models. Kotelnikov et al. [7] adapted the denoising diffusion mechanism to tabular data and proposed TabDDPM. Across fifteen real-world datasets, this method achieved superior predictive utility compared with CTGAN, TVAE, and SMOTE, although it was also reported to fall behind SMOTE on synthetic data quality in some scenarios. Work following TabDDPM has advanced diffusion-based tabular data synthesis rapidly. Zhang et al. [11] introduced TabSyn, which runs the diffusion process in the latent space learned by a variational autoencoder, combining numerical and categorical variables into a joint representation. The more recent TabDiff method combines EDM-based diffusion for continuous variables with masking-based diffusion for categorical variables, along with a learnable column-wise noise schedule, and outperformed earlier approaches, including TabDDPM, across seven benchmark datasets [12]. Both models are included in the present benchmark so that TabDDPM-CC is compared with the current state of tabular diffusion. Neither had previously been evaluated on a small, safety-critical maintenance dataset. In aviation, SMOTE-based boosting has been combined with LightGBM for avionics fault diagnosis [45]. The comparative design closest to the present one is the benchmark of Gündüz and Şahin [46]. It evaluated four generators from three families on protein localization data with fidelity, utility and distinguishability metrics. It omits copula-based and refined diffusion generators and lies outside engineering. None of the reviewed studies compares neighborhood-based, statistical, generative and diffusion-based methods together with a refined class-conditioned diffusion generator on the same dataset and classifier.

2.3. The Shift from Tree Ensembles to Transformers and Prior-Data Networks in Tabular Data Classification

Tree ensembles such as Random Forest [13] and gradient boosting [14–16] dominated tabular data performance for a long time. Transformer-based and differentiable architectures such as TabNet [17], FT-Transformer [18], NODE [19] and SAINT [20] were later proposed to challenge this dominance. In the temporal engineering data domain, the Temporal Fusion Transformer proposed by Lim et al. [47] jointly processes static and dynamic variables to provide interpretable multi-horizon forecasting, and has since become a frequently referenced architecture in the predictive maintenance literature for modeling temporal patterns. Such sequence models require time-ordered observations, which the maintenance extract analyzed here does not contain.

The newest and most directly relevant link in this progression is pretrained prior-data fitted networks. Hollmann et al. [23] proposed TabPFN, a Transformer trained offline on synthetic priors that approximates Bayesian inference. This method can solve small- and medium-scale classification problems within seconds, without any hyperparameter tuning, and its strength was further reinforced in a follow-up study [48]. The TabPFN family has continued to evolve rapidly even after the scope of this study was set. Grinsztajn et al. [49] introduced TabPFN-2.5 and reported that it scales to 50,000 data points and 2000 features, reaching a win rate close to one hundred percent against tuned XGBoost on small- and medium-scale classification datasets. Grinsztajn et al. [50] introduced TabPFN-3 and reported that it scales to one million training rows, outperforming even four-hour tuned ensemble models on the TabArena benchmark. The present study keeps the original version as its main backbone and uses these newer releases in a backbone sensitivity check. That said, the attention mechanism underlying TabPFN's architecture computes pairwise attention over context points, so memory and computational cost grow rapidly with context size, and the original version of the model remains subject to strict upper bounds on sample count, feature count, and class count. Feuer et al. [51] proposed the first systematic context optimization strategy for prior-data networks to address this constraint, showing that compressing large datasets into a smaller, learned, and more representative context can improve both inference speed and classification performance. This finding suggested that selecting context points by representational power rather than at random is a theoretical requirement for performance and efficiency in TabPFN-based architectures, and it forms the direct theoretical basis for the context selection strategies examined in this study. A significant objection to the generality of this finding has recently emerged, however. Abdullah [52], running controlled experiments across 15 OpenML datasets, showed that computationally expensive context selection methods such as k-means and farthest-point sampling do not provide a meaningful accuracy gain over random sampling. According to that work, what determines TabPFN performance is not how well the context reflects the training distribution, but how well it represents the feature space in terms of diversity and coverage, a coverage that random sampling already achieves in expectation. This finding was obtained for general-purpose, relatively large datasets. In a small-sample, safety-critical engineering problem, the possibility that context diversity behaves differently because of physical overlap between classes warrants a separate discussion of the k-center-based context selection proposed in this study. This discussion does not change the fact that applying representation-based context selection to a small-sample, safety-critical engineering problem such as aviation maintenance has not yet been attempted in the literature.

2.4. Quantitative Assessment of Synthetic Data Quality and Explainability

The contribution of data augmentation methods should be assessed not only by final classification accuracy but also alongside the statistical realism and novelty of the generated synthetic data. Snoke et al. [53] adapted propensity score mean squared error, which measures how well a discriminative classifier can distinguish between real and synthetic sources within a combined sample, to synthetic data evaluation, and derived its theoretical distribution under a correctly specified synthesis model. The accuracy of this discriminative classifier is also commonly reported in the literature as an independent metric. On the novelty and privacy dimension, Zhao et al. [54] were among the first to systematically use the Nearest Neighbor Distance Ratio metric in tabular data synthesis, in the study that introduced CTAB-GAN, confirming with this metric that the synthetic data did not simply memorize real records. No study reviewed here combines all three of these metrics in an engineering or fault-data context, and even the closest multi-metric benchmark identified above [46] was conducted outside such a context.

On the model transparency dimension, Lundberg and Lee [31] proposed the SHAP framework, which draws on Shapley values from cooperative game theory to quantify each feature's contribution to model output in a consistent and additive way. This method quickly became the most widely used interpretability approach in the fault diagnosis literature. Linardatos et al. [55] emphasized that model-agnostic, Shapley-based approaches are particularly effective at explaining the decision mechanisms of complex ensemble and deep learning models in ways consistent with engineering domain knowledge. SHAP has also been applied to aircraft engine remaining useful life estimation [56], and an explainable Bayesian filtering approach has been proposed for the same task [57]. In both cases the explained models were trained on real or simulated data rather than on training sets enriched with synthetic samples. How the augmentation process shapes the decision mechanism a model learns therefore remains largely open.

Even where synthetic data quality and model explainability have been addressed together, none of the studies reviewed here quantifies the operational cost of misclassification. A partial solution appears in a non-aviation domain, where a cost-sensitive metric for heavy commercial vehicle component failures showed that imbalance reduction strategies can change model rankings [58]. That work relies on cost information supplied by the operating organization, so it does not transfer directly to aviation maintenance, where such data are unavailable. Even so, it shows that relative cost matrices can support model comparison in the absence of real cost data.

2.5. The Literature Gap and the Positioning of This Study

A comparative summary of the studies reviewed in this section is presented in Table 1. Nineteen studies directly comparable to the four core components of this work, namely data augmentation, classifier, synthetic data quality metrics, and explainability, are included. Three of these studies concern the AI4I 2020 public benchmark [32] and support the transferability check reported in Section 5.8.

The comparative summary presented in Table 1 shows that the literature has progressed in a fragmented way along four lines. First, most studies addressing the imbalance problem remain limited either to the architectural level or to a single data augmentation method, and none tests neighborhood-based, probabilistic, generative adversarial, and diffusion-based methods simultaneously on the same dataset. This means that claims of one method's superiority over another remain dataset-specific, with limited generalizability.

Table 1. Comparative analysis of related studies.

Study	Application Domain	Data Augmentation	Classifier	Synthetic Data Quality Metrics	Explainability
[7]	General-purpose tabular data	TabDDPM	CatBoost/Multi-Layer Perceptron (MLP)	ML utility (TSTR-style benchmark)	X
[33]	General fault diagnosis	X	Bayesian deep learning (uncertainty-aware)	X	X
[34]	Aircraft engine gas path fault diagnosis	Feature filtering and mapping (feature-level correction)	Deep learning	X	X
[35]	Digital-twin-assisted fault diagnosis	X	Deep transfer learning (subdomain adaptation, margin-aware regularization)	X	X
[39]	Aircraft fuel pump fault diagnosis	Conditional Variational Autoencoder (CVAE) + ADASYN	SVM, RF, MLP, CNN	X	X
[40]	Long-tailed fault diagnosis (review)	Multiple augmentation methods	Multiple classifiers	X	X
[41]	Aero-engine bearing fault diagnosis	SMOTE	Broad Learning System (BLS)	X	X
[42]	Rotating machinery fault diagnosis	GAN-based augmentation	Deep learning	X	X
[43]	GAN-based imbalanced fault diagnosis (review)	Conditional GANs, Wasserstein GANs, self-attention variants	Multiple classifiers	X	X
[45]	Avionics equipment fault diagnosis	SMOTE with Boosting (SMOTEWB)	LightGBM	X	X
[46]	Protein subcellular localization (non-aviation, tabular biological data)	4 methods (SMOTE, CTGAN, TVAE, TabDDPM), systematic comparison	XGBoost, Logistic Regression, SVM, Random Forest	Fréchet/Wasserstein distance + C2ST + TSTR/TRTR	X
[51]	General-purpose large-scale tabular data	X	TabPFN, context-optimized via TuneTables	X	X
[56]	Aircraft engine Remaining Useful Life (RUL) estimation	SMOTE (auxiliary)	LightGBM + CatBoost + Gradient Boosting Regressor (GBR) ensemble	X	SHAP (global + local)

Table 1. *Cont.*

Study	Application Domain	Data Augmentation	Classifier	Synthetic Data Quality Metrics	Explainability
[57]	Aircraft engine RUL estimation	X	Bayesian filtering combined with a data-driven regression model	X	Explainable method (non-SHAP)
[58]	Heavy vehicle component failure	Downsampling/SMOTE-Tomek	Multiple (RF, SVM, XGBoost, etc.)	X	X
[59]	Industrial robot component-level PHM	StarGAN (multi-domain scalogram translation)	CNN transfer learning (GoogLeNet, SqueezeNet, AlexNet, VGG16, Inception v3, ResNet50)	X	X
[60]	Industrial multi-label failure mode diagnosis (AI4I 2020 benchmark [32])	Combined oversampling and undersampling	Balanced Hoeffding Tree Forest (BHTF)	X	X
[61]	CNC milling predictive maintenance (AI4I 2020 [32] and PHM 2010 benchmarks)	CTGAN (statistically audited)	XGBoost, TabTransformer	X	SHAP
[62]	Connected-vehicle predictive maintenance (multi-dataset, incl. AI4I 2020 [32])	SMOTE	LightGBM (among others)	X	SHAP
This study	Aircraft oil pump maintenance action classification	11 methods (including TabDDPM, TabSyn, TabDiff and TabDDPM-CC), systematic comparison	Context-optimized, ensemble TabPFN (TabPFN-CE)	Propensity score + Discriminator accuracy + Mean DCR + Wasserstein/FD + TSTR/TRTR	SHAP

Note: X indicates that the study does not use or report the item in the corresponding column.

Second, Feuer et al. [51] showed that context optimization improves TabPFN performance on general-purpose, large-scale datasets, whereas Abdullah [52] showed that this improvement holds only for such datasets, and that methods like k-means and k-center provide no meaningful gain over random sampling. Applying this strategy to a small-sample, four-class, safety-critical engineering problem such as aviation maintenance has not been addressed in any of the studies reviewed. Third, an integrated framework combining multidimensional verification of synthetic data quality with model explainability is absent from every study reviewed, including Özcan [56] and Maulana et al. [57]. Even in these two studies, SHAP was applied to models trained directly on real data rather than on data enriched with synthetic samples. Fourth, the operational cost of misclassification has been addressed with a relatively simple cost matrix in a non-aviation domain, and

a cost-sensitive evaluation framework for the aviation maintenance context has not yet appeared in the literature.

This study addresses these gaps within a single framework. Its core is the pairing of a class-conditioned diffusion generator with coverage-based context selection for TabPFN. The broad benchmark, the multi-metric quality assessment and the SHAP analysis serve to test and explain this pairing rather than being separate contributions. The cost asymmetry of the maintenance decision is used only as a descriptive reading of the per-class results, since no formal cost matrix is available for this domain.

3. Theoretical Background

3.1. Tabular Data Augmentation Methods

The augmentation methods evaluated in this study span four families: neighborhood-based interpolation, statistical density modeling, deep generative networks and diffusion-based generation. The first three families and the two recent diffusion benchmarks are described below in their standard form. TabDDPM, the basis of the proposed generator, is described in more detail.

3.1.1. Neighborhood-Based Methods

Synthetic Minority Over-sampling Technique (SMOTE) generates minority-class instances by interpolating between a real sample and one of its nearest minority neighbors. Because every minority instance receives equal weight, synthetic points may fall in regions that are already well represented rather than near the most ambiguous parts of the decision boundary [2].

Adaptive Synthetic Sampling (ADASYN) uses the same interpolation but assigns more synthetic points to minority instances surrounded by many majority-class neighbors. Generation is thus concentrated near the decision boundary, where classification errors are most likely [3].

3.1.2. Statistical Methods

Bayesian Gaussian Mixture (BGM) models the data distribution as a weighted combination of Gaussian components, with the number of active components inferred through variational Bayesian inference. New samples are drawn by first selecting a component according to its weight and then sampling from that Gaussian, which allows multimodal feature distributions to be represented [63].

Gaussian Copula separates the modeling of each feature's marginal distribution from the modeling of the dependency structure between features. Each variable is transformed to a uniform scale through its marginal distribution, and the correlations among the transformed variables are captured by a multivariate Gaussian copula [5].

3.1.3. Deep Generative Network-Based Methods

Copula Generative Adversarial Network (CopulaGAN) embeds this copula-based formulation within a generative adversarial framework. A conditional generator learns to produce samples whose transformed representation matches the target copula, while a discriminator separates real from synthetic samples. This allows nonlinear and higher-order dependencies to be captured beyond what a purely parametric copula represents [5,6].

Conditional Tabular GAN (CTGAN) adapts the generative adversarial network to tabular data with mixed continuous and categorical columns. A conditional generator and a training-by-sampling scheme prevent the mode collapse that standard GANs tend to show on rare categorical values [6].

Tabular Variational Autoencoder (TVAE), introduced in the same study, follows a different generative principle. An encoder compresses each observation into a low-dimensional latent representation, and a decoder reconstructs tabular samples from points drawn in that latent space. Its reconstruction-based objective usually gives more stable training than adversarial alternatives, at the cost of sometimes producing smoother synthetic distributions [6].

3.1.4. Diffusion-Based Methods

Tabular Denoising Diffusion Probabilistic Model (TabDDPM) [7] adapts the DDPM formulation of Ho et al. [8] to heterogeneous tabular data. The underlying principle is to learn the data distribution by reversing a fixed Markovian noising process, rather than by adversarial training as in GAN-based generators, thereby avoiding the training instability and mode-collapse problems reported for GAN-based tabular synthesizers [6].

Let x_0 denote a real feature vector sampled from the training distribution. The forward diffusion process progressively perturbs x_0 with Gaussian noise over T discrete timesteps according to a fixed variance schedule $\beta_t \in (0,1)$:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\right) \tag{1}$$

Owing to the closed-form property of Gaussian marginals, the state at an arbitrary step t can be sampled directly from x_0 without iterating through all intermediate steps:

$$x_t = \sqrt{\alpha_t}\, x_0 + \sqrt{1-\alpha_t}\, \varepsilon\,, \ \varepsilon \sim \mathcal{N}(0\,, I) \tag{2}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1} t\, \alpha_s$ is the cumulative product of the retained signal proportion up to step t. In the baseline formulation, β_t follows a linear schedule between fixed bounds β_1 and β_T.

A neural network $\varepsilon_\theta(x_t, t)$ is trained as the denoiser to predict the noise component injected at step t, conditioned on a sinusoidal timestep embedding. The training objective is the simplified variational bound proposed by Ho et al. [8], reducing to a mean-squared-error loss between the true and predicted noise:

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0,\, \varepsilon,\, t}\left[\left|\varepsilon - \varepsilon_\theta(x_t, t)\right|_2^2\right] \tag{3}$$

Synthetic samples are generated by running the reverse (denoising) process starting from pure Gaussian noise $x_T \sim \mathcal{N}(0, I)$, and iteratively applying:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\, \varepsilon_\theta(x_t, t)\right) + \sqrt{\beta_t}\, z,\ z \sim \mathcal{N}(0, I) \text{for} t > 1 \tag{4}$$

In its baseline configuration, TabDDPM trains a separate denoiser network per class, each conditioned only on the raw (non-normalized) timestep index, with no post-hoc quality control applied to the generated samples. This configuration is retained in the present study as the unmodified reference point for the proposed enhancements.

Two recent diffusion models are used as additional benchmarks. Tabular Synthesis (TabSyn) [11] first trains a variational autoencoder with a Transformer-based tokenizer that maps mixed-type rows to a continuous latent space. A score-based diffusion model is then trained in this latent space, and generated latents are decoded back to tabular rows. Tabular Diffusion (TabDiff) [12] works directly in the data space. It combines continuous-time diffusion for numerical columns with masked diffusion for categorical columns, and it learns a separate noise schedule for each column. Both methods were run with the default architectures of their reference implementations. For class-wise generation, TabSyn used one shared autoencoder and a separate latent diffusion model for each minority class. TabDiff was trained once on all classes and generated minority records by fixing the label column during sampling with its in-painting sampler.

3.2. Classification Algorithms

Once the augmented dataset has been generated, its usefulness is assessed against a broad set of tabular classification algorithms, grouped here into three families according to their underlying computational structure: tree-based ensembles, differentiable decision tree ensembles, and attention or transformer-based architectures. Each classifier is presented in its standard published form. TabPFN, the backbone of the proposed classifier, is described in more detail.

3.2.1. Tree-Based Ensemble Methods

Random Forest (RF) builds a large collection of independent decision trees, each trained on a bootstrapped sample of the data and a randomly selected subset of features, and combines their predictions through majority voting [13]. This randomization reduces the variance of single trees, and the built-in feature importance scores give a direct view of which sensor readings drive a prediction.

eXtreme Gradient Boosting (XGBoost) applies gradient boosting sequentially, with each new tree fitted to the residual errors of the current ensemble through a second-order approximation of the loss and explicit regularization of model complexity. Its efficient handling of sparse data and parallel tree construction have made it common in industrial fault diagnosis [14].

Light Gradient Boosting Machine (LightGBM) reduces training time through gradient-based one-side sampling and exclusive feature bundling. Its leaf-wise tree growth can reach lower error than the level-wise growth of XGBoost, and its memory efficiency suits near real-time maintenance systems [15].

Categorical Boosting (CatBoost) addresses the target leakage of conventional categorical encoding and the bias of greedy tree construction through ordered boosting. Its low sensitivity to hyperparameter tuning makes it practical when training data are limited [16].

3.2.2. Differentiable Decision Tree Ensembles

Neural Oblivious Decision Ensembles (NODE) turn oblivious decision trees, in which all nodes at a given depth share the same splitting rule, into a differentiable architecture trained end to end. The discrete split is replaced with the entmax transformation, and the layered structure allows higher-order feature interactions to be captured [19].

Gated Adaptive Network for Deep Automated Learning of Features (GANDALF) introduces the Gated Feature Learning Unit, which adapts the gating mechanism of gated recurrent units to non-sequential tabular data and includes built-in feature selection. It is reported to be competitive with XGBoost, SAINT and FT-Transformer while requiring fewer hyperparameters [21].

3.2.3. Attention and Transformer-Based Architectures

Tabular Network (TabNet) applies a sequential attention mechanism that selects a different subset of features at each decision step, combining the interpretability of tree-based feature selection with the representational flexibility of a neural network trained end to end. Its support for self-supervised pretraining can be useful when only a limited number of labeled maintenance records are available [17].

Feature Tokenizer Transformer (FT-Transformer) maps numerical and categorical features into a shared embedding space and processes these embeddings with standard Transformer blocks. Its self-attention captures interactions across features, although its relatively high parameter count raises the risk of overfitting on small datasets [18].

ExcelFormer alternates between two attention modules, one for feature interactions and one for updating feature embeddings, and uses a small-value initialization that weakens

attention at the start of training. It has been reported to outperform heavily tuned gradient-boosted trees on many public tabular datasets [22].

Self-Attention and Intersample Attention Transformer (SAINT) applies attention across both columns and rows, so that similar observations can inform one another during prediction. Its contrastive pretraining is intended to compensate for limited labels [20].

3.2.4. Prior-Data Fitted Networks

Tabular Prior-Data Fitted Network (TabPFN) [23,48] is a Transformer-based classifier that approximates Bayesian inference in a single forward pass, without gradient-based training on the target dataset. During a large-scale offline pre-training phase, the network q_θ is trained on millions of synthetic classification tasks sampled from a structural causal prior $p(\varphi)$, where each task consists of a synthetic training set $D_{train} = \{(x_i, y_i)\}_{i=1}^{n}$ and a query point x_{test}.

The target of pre-training is the Posterior Predictive Distribution (PPD) of Bayesian inference:

$$p(y_{test} \mid x_{test}, D_{train}) = \int p(y_{test} \mid x_{test}, \varphi)\, p(\varphi \mid D_{train})\, d\varphi \tag{5}$$

Because this integral is intractable in closed form for realistic priors, TabPFN trains a Transformer q_θ to directly approximate it by minimizing the expected negative log-likelihood over the prior distribution of synthetic datasets:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{D \sim p(D)} \left[-\log q_\theta(y_{test} \mid x_{test}, D_{train}) \right] \tag{6}$$

At inference time, the entire real training set D_{train} is presented to the frozen network as in-context examples, together with the query point x_{test}, within a single attention-based forward pass. The class-posterior estimate is obtained through a softmax read-out over the final hidden representation f_θ:

$$q_\theta(y_{test} = k \mid x_{test}, D_{train}) = \left(\operatorname{softmax}(f_\theta(D_{train}, x_{test}))\right)_k \tag{7}$$

This mechanism eliminates the need for dataset-specific gradient updates or hyperparameter search, allowing TabPFN to match or exceed the accuracy of gradient-boosted decision trees on small- and medium-sized tabular datasets [23,48]. In its baseline configuration, the classifier does not apply a selection strategy to the in-context examples. The entire augmented training set serves as the context, or a class-stratified random sample of size N_{max} is drawn without any geometric selection criterion if the row count exceeds the practical context-length limit N_{max}. This configuration is the reference point for the context-selection strategies of Section 4.5.1.

Several TabPFN releases are available. The original TabPFN v1 [23] accepts at most 1000 training samples, 100 features and 10 classes. TabPFN v2 [48] raised these limits and added native handling of categorical features and missing values, and TabPFN-2.5 [49] and TabPFN-3 [50] scale further to larger tables. The proposed classifier is built on v1 for three reasons. First, the strict context limit of v1 is the setting in which context construction matters most. Second, the framework was designed and tuned on this version. Third, v2.5 and v3 are released under non-commercial licenses that require registration with Prior Labs, so they cannot be used in an operational maintenance system without a commercial license. The v2, v2.5 and v3 backbones are therefore evaluated as a sensitivity check under the same protocol.

3.3. Synthetic Data Quality Metrics

Synthetic data quality is assessed through fidelity, utility, and novelty metrics. For both datasets, all of these metrics follow a holdout-based evaluation [64]. The real records

of each seed are split into training, validation and test partitions before any synthetic record is generated. Fidelity and novelty are measured against the training partition, and the test partition serves only as unseen data for reference and evaluation.

Propensity Score and Discriminator Accuracy: Following Snoke et al. [53], synthetic records and an equally sized set of real training records of the same class are pooled, and a logistic regression discriminator is trained to distinguish their source under five-fold cross-validation [11,12]. The area under the ROC curve of the resulting out-of-fold propensity scores (propensity AUC) and the classifier's accuracy are used as fidelity indicators. Values near the chance level of 0.50 indicate that real and synthetic observations are statistically indistinguishable, consistent with the approach of Zhao et al. [54].

TSTR/TRTR (Utility Ratio): A Random Forest proxy with fixed hyperparameters is trained on the real majority-class records together with the synthetic minority records, which gives the Train on Synthetic, Test on Real (TSTR) score. The same proxy trained on the real training partition gives the Train on Real, Test on Real (TRTR) score. Both scores are macro F1 values measured on the held-out test partition [11,12]. The ratio of their performance scores reflects how much task-relevant information the synthetic data preserves, with values close to one indicating high utility [6,46]. Because this metric uses the test partition, it is reported for information only and is not used to select augmentation methods.

Wasserstein Distance: For each numerical feature, the Wasserstein distance between the real training and synthetic marginal distributions is computed and averaged over the features to assess column-level distributional agreement. Lower values indicate closer alignment [7,46].

Fréchet Distance (FD): As a multivariate extension of the Wasserstein comparison, the Fréchet distance summarizes differences between real training and synthetic data in terms of their mean vectors and covariance structures, capturing joint distributional and correlation shifts [46]. Smaller values indicate closer overall structural agreement.

Distance to Closest Record (DCR): Following Zhao et al. [54], each synthetic observation's distance to its nearest real training record is computed as a Euclidean distance on features standardized with the training partition, then averaged across samples. Very low values suggest memorization of real records, while values well above the real data's internal nearest-neighbor distances suggest a loss of realism. Average DCR values are interpreted relative to this real-data reference distribution [64].

Memorization and Diversity: Mean DCR alone does not show whether synthetic records lie closer to the training data than unseen real records would. The held-out test records of the same class therefore serve as a reference, following the holdout-based protocol of Platzer and Reutterer [64] and the DCR share used for TabSyn [11]. The DCR share is the proportion of synthetic records that are closer to the training records than to an equally sized held-out subset, with an ideal value of 0.50. The near-copy rate is the proportion of synthetic records whose DCR falls below the 5th percentile of the held-out-to-training distances, with an ideal value of 0.05. The DCR ratio divides the mean DCR of the synthetic records by that of the held-out records, so values below one indicate records that sit closer to the training data than unseen data do. Diversity is measured by coverage [65], which is the share of real records whose five-nearest-neighbor neighborhood contains at least one synthetic record. The unique nearest-neighbor (NN) ratio is the proportion of distinct real records that are the nearest neighbor of at least one synthetic record. Both diversity measures are compared with the values obtained for the held-out records. These measures describe the generated data only and play no role in model selection.

3.4. Explainability

To interpret the predictions of the classification models, SHapley Additive exPlanations (SHAP) is employed. Grounded in cooperative game theory, SHAP attributes to each feature a contribution value reflecting its marginal effect on a given prediction, computed by averaging that feature's impact across all possible feature coalitions [31]. This provides both local explanations for individual predictions and global feature importance rankings across the dataset, allowing the influence of specific sensor and maintenance variables on the model's fault classification decisions to be examined in a consistent and model-agnostic manner [55]. SHAP values describe how a fitted model uses its inputs. They do not establish a causal link between a feature and a physical degradation process, and they depend on the chosen background data [66].

4. Proposed Approach

4.1. Proposed Framework Overview

This section presents the proposed framework for classifying imbalanced aircraft maintenance data. Figure 1 illustrates the overall pipeline. The framework consists of six sequential modules, summarized below and detailed in the following subsections.

1. *Dataset and Experimental Setup:* This module defines the maintenance dataset and the experimental protocol, including the class distribution, the stratified train, validation and test partitioning, and the ten-seed evaluation protocol (Section 4.2).
2. *Data Augmentation Framework:* This module addresses class imbalance in the training partition through established and recent tabular augmentation methods and the proposed generator TabDDPM-CC (Section 4.3).
3. *Feature Engineering:* This module derives six additional degradation and stress indicators from the six original variables (Section 4.4).
4. *Classification Framework:* This module trains the comparison classifiers together with the proposed classifier TabPFN-CE on the augmented, feature-engineered data (Section 4.5).
5. *Ablation, Robustness and Cost Analysis:* This module isolates the contribution of each modification and feature. It also tests the framework under noise, covariate shift and different augmentation ratios and reports its computational cost (Sections 4.6 and 4.7).
6. *Explainability Analysis:* This module applies SHAP to the final proposed configuration to identify the variables that most strongly drive the classification decision (Section 4.8).

4.2. Dataset and Experimental Setup

The dataset analyzed in this study is derived from operational and technical records of aircraft operating within Turkish civil aviation, combining periodic maintenance logs with flight telemetry data. This regional grounding is relevant because local operational conditions, climate, route structure, and maintenance practices can meaningfully influence predictive maintenance outcomes. The dataset targets the health monitoring of the aircraft oil pump and consists of six independent variables together with a categorical target variable representing the maintenance decision. The records were supplied as a single extract with one row per maintenance record, and all 647 of them were used without further filtering. The extract contains no time stamp, tail number or aircraft identifier. The temporal order of the records and the number of records per aircraft therefore cannot be recovered.

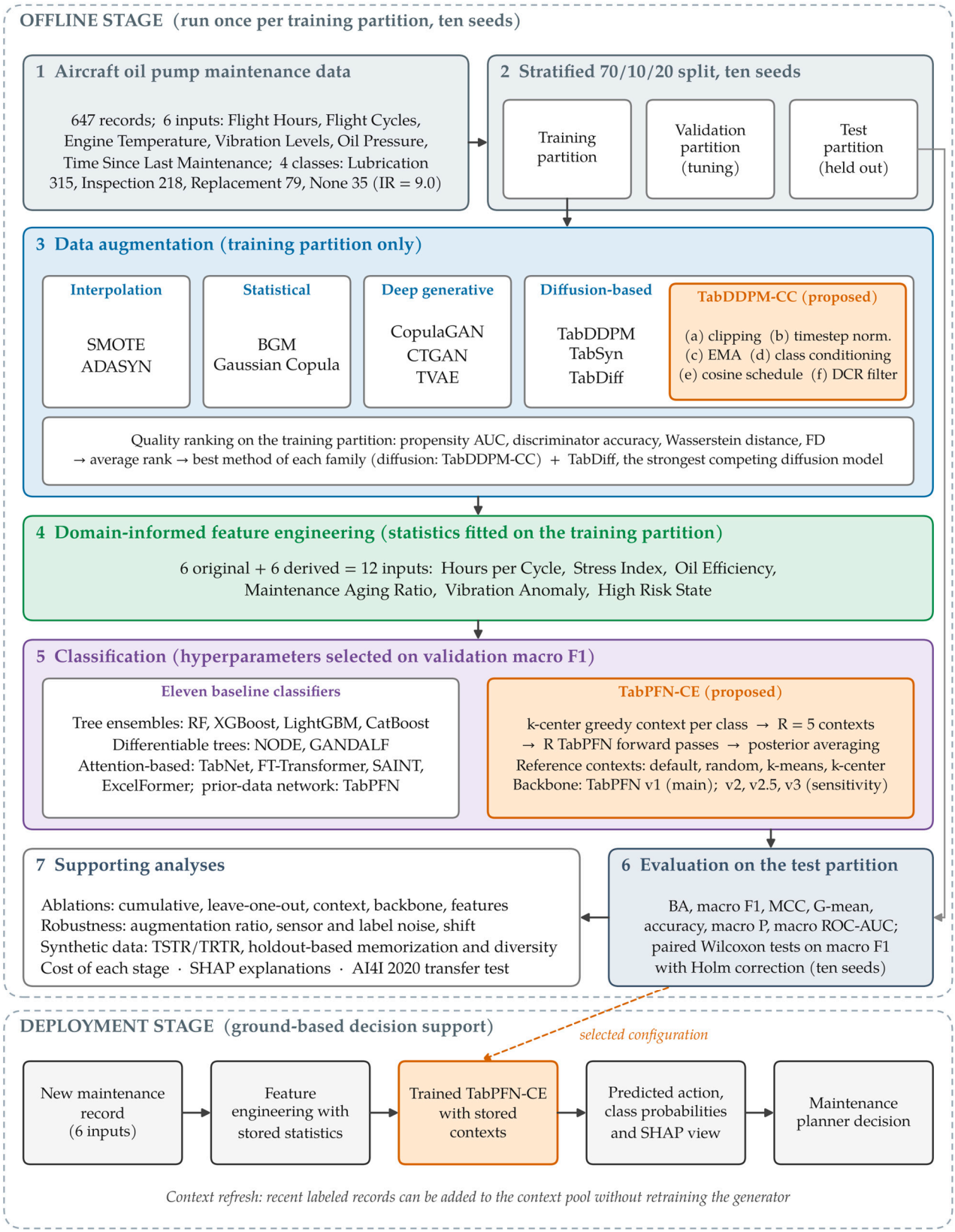


Figure 1. Flowchart of the proposed framework. The offline stage covers partitioning, augmentation and feature engineering as well as model selection and evaluation. The deployment stage scores new maintenance records with the trained TabPFN-CE model. Orange boxes mark the proposed TabDDPM-CC and TabPFN-CE components. The other colors only separate the modules of the pipeline.

The independent variables are grouped into three categories. Flight parameters include Flight Hours, representing cumulative operational load, and Flight Cycles, representing operational cycle count. Sensor readings include Engine Temperature, Vibration Levels, and Oil Pressure, which reflect the thermal and mechanical condition of the engine. Maintenance timing is represented by Time Since Last Maintenance, defined as the number of days elapsed since the last intervention.

The target variable, Maintenance Action, is divided into four categorical classes: None, Inspection, Lubrication, and Replacement. The dataset contains 647 observations in total. The class distribution is markedly imbalanced. Lubrication accounts for 315 samples and Inspection for 218 samples, while None accounts for only 35 samples and Replacement for 79 samples. This yields an imbalance ratio of 9.0 between the majority class and the smallest minority class. This pattern is consistent with real-world aviation maintenance data, where routine inspection and lubrication actions occur far more frequently than critical part replacements or fully fault-free operation. With the 70/10/20 split described below, each training partition holds about 24 None records and each test partition holds 7. Recall for this class is therefore measured on a coarse grid, and one misclassified test record changes it by about 0.14.

To preserve class proportions across data partitions, the dataset was split into training, validation, and test subsets using stratified sampling at a 70/10/20 ratio. Stratification was required because an unstratified split could leave the None class with too few test records to yield an interpretable metric. This random partition may be optimistic if several records come from the same aircraft. Because cumulative flight hours grow with service time, an additional experiment trains on low-utilization records and tests on high-utilization records as an approximation of a chronological split (Section 5.5.3). Class imbalance was addressed exclusively within the training partition. Each minority class was enlarged up to four times its real size and at most to the size of the majority class. Raising the None class to the majority size would have required about nine synthetic records per real record. The maximum augmentation factor was therefore fixed at 4.0 before the experiments to keep the synthetic share of the smallest class moderate.

Hyperparameter optimization for the augmentation methods and the downstream classifiers was performed on the validation set using grid search. The search spaces considered for each augmentation method are reported in Appendix A, Table A1. Final performance was assessed on the untouched test set, ensuring no data leakage between the augmentation, tuning, and evaluation stages.

The utility ratio of Section 3.3 was computed using a Random Forest classifier as a fixed proxy model, applied consistently across all augmentation methods to ensure comparability. Given the residual class imbalance even after augmentation, macro F1-score and balanced accuracy were prioritized over raw accuracy throughout model evaluation, in order to prevent the dominant classes from masking poor minority-class performance.

The design decisions were taken in a fixed order to keep the test partition independent of model selection. The six engineered features were defined from domain reasoning before any model was trained, and the augmentation factor was fixed as described above. Generator and classifier hyperparameters were then selected by grid search on the validation partition of each seed, using the validation macro F1 of the Random Forest proxy for the generators and that of each classifier for the classifiers. The family representatives were chosen from the average rank of the four fidelity metrics in Table 2, which are computed on the training partition. The TSTR/TRTR ratio uses the test partition and was therefore kept out of this rank. The ratio, noise, shift and feature ablations reported later were run after these choices and were not used to revise them. The test partition was thus used for evaluation only and never for selection.

Table 2. Evaluation metrics for the eleven augmentation methods on the aircraft dataset (mean ± std over ten seeds).

Evaluation Categories	Distinguishability (Classifier-Two-Sample Test-C2ST)		Privacy & Overfitting	Distribution Fidelity		ML Utility	Avg. Rank ***
Method\ Metric	**Propensity AUC ↓ * (Mean ± Std)**	**Discriminator Acc ↓ * (Mean ± Std)**	**Mean DCR ** (Mean ± Std)**	**Wasserstein ↓ (Mean ± Std)**	**FD ↓ (Mean ± Std)**	**TSTR_TRTR_Ratio (F1_Macro) ↑ (Mean ± Std)**	
SMOTE	0.586 ± 0.012	*0.627 ± 0.009*	0.876 ± 0.188	0.079 ± 0.025	8.775 ± 3.171	0.583 ± 0.072	**4.00**
ADASYN	0.622 ± 0.036	0.655 ± 0.011	0.917 ± 0.172	0.071 ± 0.031	9.025 ± 3.682	0.554 ± 0.040	6.25
BGM	0.643 ± 0.017	0.657 ± 0.016	2.529 ± 0.255	0.110 ± 0.055	12.014 ± 4.302	0.510 ± 0.037	7.75
Gaussian Copula	0.618 ± 0.027	0.667 ± 0.011	2.314 ± 0.275	0.085 ± 0.051	8.103 ± 3.012	0.574 ± 0.095	**6.50**
CopulaGAN	0.847 ± 0.072	0.878 ± 0.018	4.998 ± 0.251	0.254 ± 0.073	21.784 ± 9.089	0.333 ± 0.111	11.00
CTGAN	0.821 ± 0.081	0.844 ± 0.022	4.688 ± 0.312	0.206 ± 0.087	19.003 ± 8.256	0.460 ± 0.120	10.00
TVAE	0.732 ± 0.084	0.743 ± 0.023	1.953 ± 0.284	0.182 ± 0.069	18.396 ± 6.773	0.487 ± 0.104	**9.00**
TabDDPM	0.611 ± 0.031	0.652 ± 0.007	2.007 ± 0.163	0.037 ± 0.021	6.113 ± 4.185	0.662 ± 0.077	4.00
TabSyn	0.614 ± 0.027	0.654 ± 0.009	1.930 ± 0.240	0.031 ± 0.018	6.020 ± 3.600	0.571 ± 0.088	4.00
TabDiff	0.574 ± 0.020	0.636 ± 0.007	1.840 ± 0.290	0.024 ± 0.013	5.820 ± 3.100	0.712 ± 0.074	2.25
TabDDPM-CC	*0.565 ± 0.022*	0.630 ± 0.006	1.719 ± 0.315	*0.019 ± 0.012*	*5.559 ± 3.042*	*0.736 ± 0.080*	**1.25**

Note: Bold average ranks indicate the family representatives selected for the classification stage. Italic values indicate the best (most favorable) result for the corresponding metric; italics are not applied to the Mean DCR column, since no single optimal direction is defined for this metric. Arrows give the preferred direction of each metric (↓ lower is better and ↑ higher is better). * Lower propensity AUC and discriminator accuracy indicate closer proximity to the ideal value of 0.5, reflecting greater indistinguishability between real and synthetic samples. ** Mean DCR has no single optimal direction. Values that are too low may indicate memorization of real records, while values that are too high may indicate divergence from the real data manifold. This metric is therefore excluded from the composite ranking. *** Average rank is the mean of the ranks (1 = best and 11 = worst) assigned to each method across the four fidelity metrics computed on the training partition. These are propensity AUC, discriminator accuracy, Wasserstein distance and FD. Mean DCR and the TSTR/TRTR ratio are reported but not ranked.

4.3. Data Augmentation Framework

The ten benchmark augmentation methods and the proposed generator are each fitted on the training partition of every seed and evaluated with the fidelity, utility and novelty metrics of Section 3.3. One representative per methodological family is retained for the main comparison. Within each family, the method with the best average rank across the four fidelity metrics computed on the training partition is selected. Because the proposed TabDDPM-CC is itself the diffusion representative, the second-ranked diffusion method is also retained. This allows a direct comparison of the proposed generator with a strong diffusion model.

Proposed Generator: TabDDPM-CC

TabDDPM-CC extends the baseline formulation of Section 3.1.4 through six cumulative modifications, presented below in two stages for clarity. The first stage (a–c) targets training stability while leaving the per-class model structure of the baseline intact. The second stage (d–f) replaces that structure and adds a post-generation filtering step. All six modifications together constitute TabDDPM-CC. The intermediate configuration with only modifications (a)–(c) is denoted TabDDPM-S and is used only in the ablation study.

- Stability-Oriented Modifications (a)–(c):

(a) Range clipping: Because the stochastic reverse process of Equation (4) can generate feature values outside the physically observed range of a class, a static thresholding step commonly used to stabilize diffusion-model sampling [8,10] clips every synthetic vector

x_{syn}, feature-wise, to the empirical range of the corresponding class in the real training data $X_{real,c}$:

$$\hat{x}_{syn} = \text{clip}\left(x_{syn}, min(X_{real,c}), max(X_{real,c}) \right) \quad (8)$$

(b) Timestep normalization: The raw integer timestep index t used in the baseline sinusoidal embedding is replaced by a schedule-independent normalized value:

$$\tilde{t} = \frac{t}{T} \quad (9)$$

so that the denoiser's time-conditioning behavior remains comparable across configurations using a different number of diffusion steps T.

(c) Exponential Moving Average (EMA) of weights: Following the stabilization strategy commonly adopted in diffusion-model training [8,10], a shadow copy θ_{EMA} of the denoiser parameters is maintained and used for sampling instead of the raw training weights θ:

$$\theta_{EMA}^{(i)} = \gamma \cdot \theta_{EMA}^{(i-1)} + (1 - \gamma) \cdot \theta^{(i)} \quad (10)$$

where $\gamma \in (0,1)$ is the decay rate and i indexes the optimization step, dampening gradient-noise-induced oscillations that are particularly pronounced when a per-class model is trained on few minority-class samples.

- Architectural and Quality-Filtering Modifications (d)–(f):

Combined with (a)–(c) above, the following three modifications complete TabDDPM-CC, replacing the per-class structure and linear schedule of the baseline and adding a post-generation quality-filtering step.

(d) Class-conditional single model: A single network is trained jointly on all classes, conditioned on a learned class embedding $e_c = \text{Embed}(c)$ concatenated with the timestep embedding:

$$\varepsilon_\theta\left(x_t, \tilde{t}, c\right) = \text{Net}\left[x_t ; \tau\left(\tilde{t}\right) ; e_c \right] \quad (11)$$

where c denotes the class label, $\tau\left(\tilde{t}\right)$ is the timestep embedding and [;] denotes concatenation, improving parameter efficiency and generalization for underrepresented classes [9].

(e) Cosine noise schedule: The linear schedule of the baseline is replaced by the cosine schedule of Nichol & Dhariwal [10]:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, f(t) = \cos^2\left(\frac{\left(\frac{t}{T}\right) + s}{(1 + s)} \cdot \frac{\pi}{2} \right) \quad (12)$$

$$\beta_t = 1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}} \quad (13)$$

where s is a small offset constant preventing β_t from approaching zero near $t = 0$, so that fine-grained structure is preserved over a larger portion of the diffusion trajectory.

(f) Distance-to-Closest-Record (DCR) quality filtering: For each class, a buffer of candidate synthetic samples is generated in excess of the required amount and scored with the DCR criterion of Section 3.3. Candidates whose DCR exceeds the p-th percentile of the buffer's DCR distribution are discarded, and the remaining candidates are ranked by ascending DCR, from which the n closest are retained:

$$\hat{X}_{syn,c} = \left\{ x_{syn} : \text{DCR}\left(x_{syn}\right) \leq \text{Percentile}_p(\text{DCR}) \right\}, \text{ top} - n \text{ by ascending DCR} \quad (14)$$

here, *n is* the number of synthetic samples needed for the class under the target augmentation ratio. The percentile was fixed at $p = 90$ and the buffer at three times n before the experiments. With these values the percentile step serves only as a safeguard and does not change the retained set, which consists of the n lowest-DCR candidates of the buffer.

This filtering step embeds a fidelity criterion directly into the generation pipeline rather than applying it only as a post-hoc evaluation metric, reducing the risk that low-fidelity synthetic instances propagate into the downstream classifier's training set. Because the filter favors candidates close to the real records, it trades a small amount of novelty for fidelity. Its effect on memorization and diversity is reported in Section 5.3.1 and Appendix B.

4.4. Feature Engineering

Once the training partition has been augmented, six additional features are derived from the six original variables, namely Flight Hours (FH), Flight Cycles (FC), Engine Temperature (ET), Vibration Levels (VL), Oil Pressure (OP), and Time Since Last Maintenance (TSM), to encode degradation and stress indicators that are not explicit in the raw measurements. Deriving composite and threshold-based indicators from raw condition-monitoring variables is an established practice in condition-based maintenance and reliability engineering, where such indicators are used to translate several correlated sensor readings into a smaller set of physically interpretable degradation proxies [67]. The transformation is applied identically to the real and synthetic training samples. It is also applied to the validation and test partitions, using statistics fitted on the training partition only, so that no information from the held-out data enters the feature definitions. A small constant $\varepsilon = 0.00001$ is added to every denominator to prevent division by zero. The first engineered feature, Hours per Cycle, is defined as:

$$\textit{Hours per Cycle} = \frac{\mathrm{FH}}{\mathrm{FC}} \tag{15}$$

This feature distinguishes long-haul from short-haul usage profiles, which impose different fatigue mechanisms on the pump for a given number of cycles. This is consistent with the role of flight-cycle count as a primary driver of component wear in civil aircraft hydraulic systems [38]. The second engineered feature, the Stress Index, is defined as:

$$\textit{Stress Index} = \mathrm{ET} \cdot \mathrm{VL} \tag{16}$$

This feature is the product of engine temperature and vibration level. It grows when thermal and mechanical loads are high at the same time, following the general practice of aggregating correlated condition-monitoring variables into a single degradation proxy [67]. The third engineered feature, Oil Efficiency, is defined as:

$$\textit{Oil Efficiency} = \frac{\mathrm{OP}}{(\mathrm{ET} + \varepsilon)} \tag{17}$$

This feature is the oil pressure delivered per unit of engine temperature. Lubricating oil loses viscosity as its temperature rises [68], so a healthy pump should hold its delivery pressure as the engine warms up. A falling pressure-to-temperature ratio can therefore point to internal leakage or wear, which is the gradual degradation reported for hydraulic piston pumps in civil aircraft [38]. The feature needs only an oil pressure channel and an engine or oil temperature channel. Both are standard engine-indicating parameters on most turbine aircraft, so the definition is not tied to this dataset. The fourth engineered feature, the Maintenance Aging Ratio, is defined as:

$$\textit{Maint_Aging_Ratio} = \frac{\mathrm{TSM}}{\mathrm{FH}} \tag{18}$$

This feature normalizes maintenance recency by cumulative usage, following the general condition-based maintenance practice of expressing elapsed time since the last intervention relative to a unit's operating exposure [67]. It separates units overdue for

maintenance relative to their operating hours from units that are simply long-serving. The fifth engineered feature, Vibration Anomaly, is defined as:

$$Vibration\ Anomaly = \frac{(\ VL\ -\ \mu_{VL,train}\ )}{\sigma_{VL,train}} \tag{19}$$

This feature expresses vibration level as a signed standardized deviation from the training-partition mean, so that unusually high and unusually low readings are placed on a common scale. This is in line with the use of vibration-based deviation measures for detecting early degradation in rotating and reciprocating machinery [42]. The sixth engineered feature, High Risk State, is defined as:

$$High_Risk_State = 1\{\ ET > \tau_{ET}\ \} \cdot 1\{\ VL > \tau_{VL}\ \} \tag{20}$$

This feature is a binary interaction term equal to one when engine temperature and vibration level jointly exceed their training-partition 75th-percentile thresholds (τ_{ET}, τ_{VL}). It captures a joint thermal–vibration stress condition associated with impending failure, consistent with threshold-based alarm indicators commonly used in condition-based maintenance systems [67].

Because each engineered feature is a deterministic function of variables already present in the model, its marginal value is treated as an empirical question rather than assumed a priori. It is tested through the factorial and leave-one-feature-out ablations described in Section 4.6. SHAP attributions are then used to describe how the final model uses the twelve inputs, not to prove that a feature is necessary.

4.5. Classification Framework

The augmented, feature-engineered training data are used to train the classifiers described in Section 3.2 together with the proposed classifier developed in this study, TabPFN-CE, which extends the baseline TabPFN formulation. All classifiers are tuned on the validation partition of every seed. The hyperparameter grids considered are reported in Appendix A, Table A2, and final performance is assessed on the held-out test partition.

4.5.1. Proposed Classifier: TabPFN-CE

Because TabPFN processes all in-context examples within a single bounded attention window, the way the context set is constructed from an augmented, class-imbalanced training set directly affects downstream performance. Three progressively refined context-construction strategies are introduced first, followed by the proposed multi-seed ensemble. All four share a common class-stratified formulation in which, for each class c, a fixed number of representative samples m is selected from the augmented training data X_c so that the resulting context D_{ctx} remains within TabPFN's practical context-length limit while preserving class balance. The first three strategies serve only as reference points in the ablation study.

- *Random Context Selection:* The m context samples for class c are drawn by simple random sampling without replacement:

$$X_{ctx,c} = \{\ x_i \sim \text{Uniform}(X_c)\text{: } i = 1, \ldots, m\ \} \tag{21}$$

This strategy, denoted TabPFN-R, provides an unbiased but geometrically uninformed reference point for the structured strategies below, consistent with the random-sampling baseline whose competitiveness on general-purpose datasets was reported by Abdullah [52].

- *k-Means Context Selection:* The second strategy, TabPFN-KM, selects representative points using k-means clustering [24] on the standardized features of class c, partitioning the class data into m clusters by minimizing the within-cluster sum of squares:

$$\{ \mu_1, \ldots, \mu_m \} = \underset{\mu}{\operatorname{argmin}} \sum_{j=1}^{m} \sum_{x \in S_j} \left\| x - \mu_j \right\|_2 \tag{22}$$

where S_j denotes the set of points assigned to cluster j. For each cluster, the closest real observation to the centroid μ_j is selected as its representative:

$$x_{ctx,j} = \underset{x \in X_c}{\operatorname{argmin}} \left\| x - \mu_j \right\|_2 \tag{23}$$

This yields a context reflecting the density structure of the class distribution, without explicitly optimizing for coverage of its extremal regions.

- *k-Center Context Selection:* The third strategy, TabPFN-KC, replaces density-based selection with the k-center greedy algorithm [25,26], which minimizes the maximum distance between any point in the class and its nearest selected representative. The first center is the point farthest from the class centroid $\bar{x}_c$:

$$x_1 = \underset{x \in X_c}{\operatorname{argmax}} \| x - \bar{x}_c \|_2 \tag{24}$$

Subsequent centers are added greedily, each maximizing the distance to the nearest already-selected center:

$$x_{(k+1)} = \underset{x \in X_c}{\operatorname{argmax}} \min_{j \le k} \left\| x - x_j \right\|_2 \tag{25}$$

where k indexes the centers already selected at a given step. This procedure yields a context that is more diverse and less redundant than a random or centroid-based subsample of equal size. Because attention-based in-context learning can, in principle, be sensitive to the input order of context examples, the selected context is permuted with a fixed seed s before being passed to TabPFN:

$$\tilde{D}_{ctx} = \mathrm{Perm}_s(D_{ctx}) \tag{26}$$

- *Multi-Seed k-Center Ensemble (Proposed Classifier TabPFN-CE):* The proposed configuration addresses the residual sampling variance of a single k-center context by combining R independently constructed contexts, each obtained by re-running the selection and permutation of the k-center strategy with a different seed s_r, $r = 1, \ldots, R$. A separate TabPFN forward pass is performed for each context, and the resulting class-posterior estimates are combined by simple averaging, a bagging-style rule [27]:

$$\bar{p}(y \mid x_{test}) = \left(\frac{1}{R} \right) \sum_{r=1}^{R} q_\theta \left(y \mid x_{test}, \tilde{D}_{ctx}^{(r)} \right) \tag{27}$$

with the final prediction $\hat{y} = \underset{y}{\operatorname{argmax}} (y \mid x_{test})$. Because each of the R contexts is constructed independently, this averaging reduces the variance contributed by the stochastic elements of context construction while preserving the coverage advantage of the underlying k-center criterion, without additional gradient-based training. Five contexts ($R = 5$) are used in all experiments unless stated otherwise.

4.5.2. Evaluation Metrics and Statistical Testing

Classification performance is evaluated using accuracy, macro-averaged precision, recall, F1-score, the Matthews Correlation Coefficient (MCC), the geometric mean (G-Mean), and macro-averaged one-vs-rest ROC-AUC. In each of the ten random seeds the dataset is split into stratified 70/10/20 training, validation and test partitions. Hyperparameters are selected on the validation partition, and all reported performance metrics are computed on the held-out test partition. Baseline (raw data) and proposed (augmented + feature-engineered) variants share identical train/validation/test splits within each seed, enabling

a paired comparison. Results are reported as mean ± standard deviation across the ten seeds. Seed-level comparisons use paired two-sided Wilcoxon signed-rank tests on macro F1, with the Holm–Bonferroni correction applied within each family of comparisons. With ten pairs, the smallest attainable two-sided exact p-value is about 0.002. The ten seeds are repeated partitions of the same 647 records rather than independent samples of the aircraft population. The tests therefore measure whether a difference is stable across resampled partitions of this dataset. They do not establish external reproducibility.

4.6. Ablation Study Design

The ablation study combines cumulative and leave-one-out designs. The cumulative design follows the two groups of modifications introduced in Section 4.3. It compares the unmodified TabDDPM, the stability-enhanced TabDDPM-S and the full TabDDPM-CC, with the classifier fixed at TabPFN-CE. The leave-one-out design removes each of the six modifications (a)–(f) from TabDDPM-CC in turn while keeping the other five. It shows which modification is still needed once all others are present. TabPFN-CE is compared with default, random, k-means and k-center contexts while the generator is fixed at TabDDPM-CC. The same comparison is repeated with the TabPFN v2, v2.5 and v3 backbones to test whether the gain of the ensemble depends on the v1 backbone.

Two further ablations concern the input representation. A 2 × 2 factorial design crosses feature engineering with augmentation. A leave-one-feature-out design removes each engineered feature in turn, and then all six together, from the final pipeline. All ablations use the ten seeds of Section 4.2 and are reported as mean ± standard deviation.

4.7. Sensitivity, Robustness and Cost Analyses

The main comparisons fix the maximum augmentation factor, referred to below as the augmentation ratio, at 4.0. To assess this choice, the final pipeline of TabDDPM-CC augmentation and TabPFN-CE classification is repeated for six ratios: 1 and 1.5 and the integers from 2 to 5. All other elements of the design are kept unchanged. The analysis is restricted to the final pipeline because its purpose is to characterize the framework as a whole rather than to re-examine the individual modifications.

Sensitivity to input noise is tested on the same pipeline, with SMOTE and the original data as controls. Sensor noise adds zero-mean Gaussian noise to the six original variables of the training partition. Its standard deviation equals 5%, 10% or 20% of the training standard deviation of each variable, and non-negative variables are clipped at zero. Label noise assigns 5%, 10% or 20% of the training records to a randomly chosen different class. The noise is injected before augmentation and feature engineering, so the generator learns from the corrupted records. The validation and test partitions stay clean.

A controlled covariate shift is created with Flight Hours, which grows with the service life of a unit. Records at or below the 70th percentile of Flight Hours form the source data used for training and validation, and the remaining records form the shifted target data. The generator and the classifier are fitted on the source data only. In a second condition, 20% of the target records are added to the pool from which TabPFN-CE builds its contexts, without retraining the generator. This context refresh mimics the arrival of a small set of labeled records from a new operating regime, and the remaining target records serve as the test set.

Computational cost is measured separately for the offline stage and the deployment stage. The offline stage covers the training and sampling of each generator once per training partition. The deployment stage covers classifier fitting, batch scoring of the test partition and the latency of a single-record query. Single-record latency is measured on the GPU workstation and, for selected models, on the CPU alone. Timings are medians of repeated

calls averaged over three seeds. The ensemble size of TabPFN-CE is varied over $R = 1, 3$ and 5 to show the trade-off between accuracy and latency.

4.8. Explainability Analysis

SHAP is applied to the final proposed configuration, in which TabPFN-CE is trained on data augmented by TabDDPM-CC and enriched with the engineered features. Because TabPFN-CE produces predictions through in-context, attention-based inference rather than an explicit tree or parametric decision function, a model-agnostic SHAP explainer is applied directly to its predicted class-posterior function $\overline{p}(y \mid x_{test})$ of Equation (27), using a background sample drawn from the k-center contexts of TabPFN-CE as the reference distribution. The resulting attributions are aggregated over the test partition to obtain a global importance ranking across all twelve input features, the six original variables and the six engineered features, and are used both to identify which operational and condition-monitoring variables most strongly drive the classification of maintenance condition and to describe how the engineered features are used alongside the variables from which they are derived.

5. Results and Discussion

All experiments were conducted on a workstation with an Intel Core i9-14900HX CPU, 64 GB RAM, and an NVIDIA GeForce RTX 4090 Laptop GPU (16 GB VRAM).

The pipeline was implemented in Python 3.10.19 with PyTorch 2.6.0. SMOTE and ADASYN were taken from imbalanced-learn 0.14.1, and the Bayesian Gaussian Mixture, Random Forest and the logistic regression discriminator from scikit-learn 1.7.1. Gaussian Copula, CopulaGAN, CTGAN and TVAE were run with the Synthetic Data Vault (SDV) 1.37.4. XGBoost 2.1.4, LightGBM 4.6.0 and CatBoost 1.2.7 provided the gradient boosting models. TabNet was run with pytorch-tabnet 4.1.0, and FT-Transformer, NODE and GANDALF with PyTorch Tabular 1.2.0. The TabPFN v1 backbone came from the tabpfn package (version 0.1.11), and the v2, v2.5 and v3 backbones from tabpfn 9.2.0 with PyTorch 2.14.1. SHAP 0.49.1 and SciPy 1.15.2 were used for the explanations and the statistical tests. TabSyn and TabDiff were reimplemented in PyTorch from their official code [11,12]. TabDDPM, SAINT and ExcelFormer were implemented in PyTorch following their original publications [7,20,22].

5.1. Data Augmentation Results

Table 2 reports the fidelity, privacy and utility metrics for the eleven augmentation methods. All values are averaged over ten random seeds. The metrics are grouped into four categories: distinguishability from the real distribution, privacy and overfitting risk, distribution fidelity and downstream machine learning utility.

The first two columns report the classifier-two-sample test (C2ST), which trains a discriminator to separate real and synthetic samples. A value close to 0.5 indicates that the synthetic samples are statistically indistinguishable from the real ones. The interpolation-based methods, SMOTE and ADASYN, achieve propensity AUC values close to this ideal (0.586 and 0.622), which is expected given their direct dependence on real neighboring samples. Among the generative models, TabDDPM-CC obtains the lowest propensity AUC (0.565) and discriminator accuracy (0.630). TabDiff follows closely (0.574 and 0.636), while TabSyn is close to the baseline TabDDPM (0.614 and 0.654). CopulaGAN and CTGAN show the weakest performance on this criterion, with discriminator accuracy above 0.84, suggesting that the generated samples remain easily separable from the real distribution.

The mean distance to closest record (DCR) reflects the proximity between synthetic and real samples. This metric should not be interpreted as a simple lower-is-better or higher-

is-better indicator. Values that are too low may point to memorization of individual training records, while values that are too high may indicate that the generator has drifted away from the real data manifold. SMOTE and ADASYN report the lowest DCR values (0.876 and 0.917), a direct consequence of their interpolation mechanism between neighboring real points. Among the density-based generators, TabDDPM-CC yields the lowest DCR (1.719). TabDiff (1.840), TabSyn (1.930) and the baseline TabDDPM (2.007) follow, and the statistical and deep generative methods all stay above 1.9. This places TabDDPM-CC closer to the real data manifold without relying on direct interpolation, which supports a reasonable balance between fidelity and privacy. The holdout-based measures in Appendix B, Table A3 put this proximity in context. TabDDPM-CC has a DCR share of 0.544 and a near-copy rate of 0.058, close to the values of 0.50 and 0.05 expected for unseen real records. SMOTE and ADASYN reach DCR shares of about 0.85 and near-copy rates of about 0.25, so interpolation places many synthetic records unusually close to training records. The coverage of TabDDPM-CC (0.903) is also close to the held-out reference (0.906).

Distribution fidelity is assessed through the Wasserstein distance and the Fréchet distance (FD), both computed between the real and synthetic feature distributions. TabDDPM-CC achieves the lowest values on both metrics, with a Wasserstein distance of 0.019 and an FD of 5.559. These values improve on TabDiff (0.024 and 5.820), TabSyn (0.031 and 6.020) and the baseline TabDDPM (0.037 and 6.113). The margin over the deep generative methods is larger, since their FD values exceed 18. This gap suggests that the class-conditioning mechanism of TabDDPM-CC improves the alignment between the synthetic and real feature distributions beyond what separate per-class diffusion models provide.

The sixth column reports the TSTR/TRTR ratio defined in Section 3.3. A ratio closer to one indicates that the synthetic data preserves the discriminative information needed for classification. TabDDPM-CC obtains the highest ratio (0.736), followed by TabDiff (0.712) and the baseline TabDDPM (0.662). CopulaGAN records the lowest ratio (0.333), which is consistent with its weak fidelity and distinguishability scores. Because both scores are measured on the test partition, this ratio is reported for information only and does not enter the selection rank.

Taken together, these results show that TabDDPM-CC provides the most favorable trade-off among the eleven methods. It combines low distinguishability from the real distribution, a controlled distance to the closest real record, the closest match to the real feature distribution and the highest downstream utility. TabDiff is the strongest of the other diffusion models and ranks second overall. The consistent improvement over the per-class TabDDPM baseline suggests that the class-conditioning component addresses the class imbalance problem more effectively than distribution matching alone, without introducing the overfitting risk observed in the interpolation-based methods (Appendix B, Table A3).

To identify the augmentation methods carried forward to the classification stage, a composite ranking score is computed for each method. This score is based on the four fidelity metrics with a clearly defined optimal direction: propensity AUC, discriminator accuracy, Wasserstein distance and Fréchet distance. All four compare the synthetic records with the training partition. The TSTR/TRTR ratio also has a clear direction, but it is left out because it uses the test partition. For each metric, every method is assigned a rank from 1 to 11, with rank 1 given to the best value among the eleven methods. The four ranks obtained by a given method are then averaged to produce its composite score. This procedure differs from averaging the raw metric values directly, since the metrics are measured on very different scales. Averaging raw values would let large-scale metrics, such as FD, dominate the score and would suppress the contribution of bounded metrics, such as the AUC-based indicators. Rank-based aggregation avoids this scale dependency and is consistent with established practice for comparing methods across multiple performance criteria, following

the average-rank framework proposed by Demšar for multi-criteria comparison, which builds on the classical Borda count method for rank aggregation [28,69].

As an example, SMOTE ranks third on propensity AUC and first on discriminator accuracy. It ranks sixth on both Wasserstein distance and FD. Averaging these four ranks gives a composite score of 4.00 ($(3 + 1 + 6 + 6)/4$). The same procedure is applied to all eleven methods, and the scores are reported in the last column of Table 2.

The methods are grouped into four families according to their generation mechanism. Within each family, the method with the lowest average rank is selected as the family representative. SMOTE (4.00) is retained over ADASYN (6.25) in the interpolation family. Gaussian Copula (6.50) is retained over BGM (7.75) in the statistical family. TVAE (9.00) is retained over CTGAN (10.00) and CopulaGAN (11.00) among the deep generative models. In the diffusion family, TabDDPM-CC (1.25) ranks ahead of TabDiff (2.25), while TabDDPM and TabSyn share the next position (4.00). TabDDPM-CC thus obtains the lowest average rank among all eleven methods.

The other six methods are not carried forward to the full classification comparison of Table 3, and their results are given in the Supplementary Materials. Because the diffusion representative is the proposed TabDDPM-CC itself, TabDiff (2.25) is also kept as the second-ranked method of this family. The proposed generator is thus compared with a strong competing diffusion model. Selecting one representative per family keeps the classification comparison manageable while still covering distinct augmentation paradigms.

Table 3. Imbalance-sensitive classification metrics of the twelve classifiers trained on the original and augmented aircraft data (mean ± std over ten seeds). The proposed TabPFN-CE, shown in bold, is the best classifier in every block.

Data Augmentation Method	Model/Metric	Balanced Acc	Macro F1	MCC	G-Mean
Original Dataset	RF	0.283 ± 0.013	0.283 ± 0.014	0.075 ± 0.033	0.014 ± 0.0004
	XGBoost	0.275 ± 0.016	0.276 ± 0.018	0.060 ± 0.050	0.014 ± 0.001
	LightGBM	0.294 ± 0.014	0.293 ± 0.014	0.094 ± 0.035	0.015 ± 0.001
	CatBoost	0.330 ± 0.032	0.329 ± 0.034	0.130 ± 0.052	0.162 ± 0.155
	TabNet	0.281 ± 0.040	0.282 ± 0.040	0.077 ± 0.070	0.014 ± 0.002
	FT-Transformer	0.386 ± 0.040	0.388 ± 0.039	0.178 ± 0.067	0.350 ± 0.045
	NODE	0.302 ± 0.022	0.302 ± 0.022	0.099 ± 0.046	0.044 ± 0.090
	GANDALF	0.355 ± 0.027	0.357 ± 0.029	0.146 ± 0.038	0.263 ± 0.131
	SAINT	0.330 ± 0.026	0.331 ± 0.028	0.119 ± 0.031	0.192 ± 0.152
	ExcelFormer	0.370 ± 0.025	0.371 ± 0.027	0.175 ± 0.030	0.302 ± 0.101
	TabPFN	0.428 ± 0.013	0.425 ± 0.015	0.226 ± 0.027	0.410 ± 0.021
	TabPFN-CE	**0.454 ± 0.006**	**0.452 ± 0.008**	**0.253 ± 0.020**	**0.439 ± 0.005**
SMOTE	RF	0.491 ± 0.018	0.501 ± 0.020	0.380 ± 0.024	0.446 ± 0.039
	XGBoost	0.465 ± 0.016	0.475 ± 0.018	0.353 ± 0.034	0.410 ± 0.035
	LightGBM	0.519 ± 0.017	0.531 ± 0.021	0.404 ± 0.039	0.490 ± 0.014
	CatBoost	0.552 ± 0.012	0.569 ± 0.020	0.464 ± 0.013	0.520 ± 0.017
	TabNet	0.466 ± 0.039	0.473 ± 0.043	0.383 ± 0.081	0.366 ± 0.125
	FT-Transformer	0.656 ± 0.040	0.670 ± 0.048	0.590 ± 0.055	0.641 ± 0.043
	NODE	0.523 ± 0.031	0.535 ± 0.041	0.421 ± 0.067	0.492 ± 0.025
	GANDALF	0.624 ± 0.016	0.641 ± 0.020	0.540 ± 0.029	0.609 ± 0.015
	SAINT	0.528 ± 0.029	0.543 ± 0.034	0.427 ± 0.051	0.493 ± 0.036
	ExcelFormer	0.627 ± 0.022	0.635 ± 0.022	0.543 ± 0.046	0.611 ± 0.020
	TabPFN	0.736 ± 0.030	0.740 ± 0.036	0.670 ± 0.038	0.730 ± 0.032
	TabPFN-CE	**0.786 ± 0.011**	**0.794 ± 0.020**	**0.722 ± 0.021**	**0.784 ± 0.010**

Table 3. *Cont.*

Data Augmentation Method	Model/Metric	Balanced Acc	Macro F1	MCC	G-Mean
Gaussian Copula	RF	0.525 ± 0.018	0.540 ± 0.017	0.447 ± 0.025	0.477 ± 0.039
	XGBoost	0.481 ± 0.025	0.489 ± 0.023	0.357 ± 0.033	0.439 ± 0.044
	LightGBM	0.531 ± 0.015	0.547 ± 0.018	0.426 ± 0.032	0.499 ± 0.014
	CatBoost	0.608 ± 0.019	0.619 ± 0.022	0.528 ± 0.023	0.587 ± 0.027
	TabNet	0.468 ± 0.065	0.473 ± 0.068	0.368 ± 0.113	0.408 ± 0.066
	FT-Transformer	0.686 ± 0.028	0.698 ± 0.036	0.607 ± 0.052	0.678 ± 0.029
	NODE	0.553 ± 0.030	0.562 ± 0.042	0.457 ± 0.047	0.524 ± 0.033
	GANDALF	0.635 ± 0.034	0.642 ± 0.034	0.554 ± 0.050	0.622 ± 0.035
	SAINT	0.579 ± 0.027	0.594 ± 0.032	0.497 ± 0.037	0.552 ± 0.035
	ExcelFormer	0.642 ± 0.035	0.652 ± 0.032	0.555 ± 0.046	0.630 ± 0.038
	TabPFN	0.714 ± 0.008	0.722 ± 0.025	0.649 ± 0.021	0.707 ± 0.007
	TabPFN-CE	**0.795 ± 0.011**	**0.793 ± 0.016**	**0.736 ± 0.016**	**0.792 ± 0.011**
TVAE	RF	0.458 ± 0.012	0.466 ± 0.011	0.351 ± 0.015	0.395 ± 0.024
	XGBoost	0.403 ± 0.028	0.405 ± 0.033	0.288 ± 0.028	0.192 ± 0.184
	LightGBM	0.449 ± 0.017	0.457 ± 0.017	0.330 ± 0.043	0.388 ± 0.021
	CatBoost	0.505 ± 0.019	0.517 ± 0.022	0.410 ± 0.025	0.457 ± 0.040
	TabNet	0.370 ± 0.051	0.371 ± 0.052	0.234 ± 0.086	0.120 ± 0.168
	FT-Transformer	0.583 ± 0.030	0.595 ± 0.043	0.491 ± 0.040	0.561 ± 0.037
	NODE	0.481 ± 0.028	0.493 ± 0.030	0.375 ± 0.035	0.428 ± 0.046
	GANDALF	0.565 ± 0.028	0.587 ± 0.033	0.475 ± 0.039	0.536 ± 0.035
	SAINT	0.496 ± 0.031	0.502 ± 0.033	0.387 ± 0.058	0.451 ± 0.045
	ExcelFormer	0.619 ± 0.014	0.629 ± 0.022	0.540 ± 0.028	0.603 ± 0.012
	TabPFN	0.643 ± 0.014	0.654 ± 0.022	0.569 ± 0.022	0.628 ± 0.016
	TabPFN-CE	**0.716 ± 0.008**	**0.734 ± 0.018**	**0.648 ± 0.016**	**0.709 ± 0.008**
TabDiff	RF	0.559 ± 0.024	0.576 ± 0.029	0.481 ± 0.044	0.524 ± 0.026
	XGBoost	0.515 ± 0.024	0.523 ± 0.022	0.413 ± 0.030	0.474 ± 0.047
	LightGBM	0.571 ± 0.028	0.583 ± 0.034	0.470 ± 0.023	0.546 ± 0.043
	CatBoost	0.629 ± 0.022	0.637 ± 0.026	0.552 ± 0.026	0.616 ± 0.022
	TabNet	0.505 ± 0.065	0.513 ± 0.067	0.427 ± 0.131	0.442 ± 0.070
	FT-Transformer	0.699 ± 0.020	0.713 ± 0.021	0.622 ± 0.045	0.688 ± 0.019
	NODE	0.571 ± 0.039	0.581 ± 0.038	0.480 ± 0.050	0.542 ± 0.046
	GANDALF	0.677 ± 0.027	0.695 ± 0.035	0.610 ± 0.043	0.664 ± 0.032
	SAINT	0.604 ± 0.036	0.612 ± 0.049	0.513 ± 0.068	0.590 ± 0.029
	ExcelFormer	0.660 ± 0.026	0.668 ± 0.032	0.573 ± 0.046	0.649 ± 0.028
	TabPFN	0.757 ± 0.030	0.768 ± 0.030	0.695 ± 0.035	0.751 ± 0.029
	TabPFN-CE	**0.818 ± 0.037**	**0.822 ± 0.042**	**0.764 ± 0.046**	**0.811 ± 0.043**
TabDDPM-CC	RF	0.553 ± 0.018	0.566 ± 0.025	0.464 ± 0.034	0.524 ± 0.023
	XGBoost	0.509 ± 0.019	0.512 ± 0.019	0.414 ± 0.026	0.465 ± 0.038
	LightGBM	0.563 ± 0.024	0.577 ± 0.028	0.463 ± 0.020	0.538 ± 0.034
	CatBoost	0.633 ± 0.017	0.639 ± 0.023	0.555 ± 0.021	0.619 ± 0.019
	TabNet	0.478 ± 0.044	0.483 ± 0.048	0.384 ± 0.069	0.412 ± 0.049
	FT-Transformer	0.709 ± 0.017	0.721 ± 0.016	0.637 ± 0.039	0.703 ± 0.016
	NODE	0.579 ± 0.030	0.592 ± 0.029	0.502 ± 0.040	0.550 ± 0.037
	GANDALF	0.671 ± 0.023	0.690 ± 0.027	0.601 ± 0.034	0.658 ± 0.027
	SAINT	0.608 ± 0.028	0.615 ± 0.038	0.511 ± 0.060	0.594 ± 0.024
	ExcelFormer	0.702 ± 0.022	0.708 ± 0.027	0.629 ± 0.039	0.695 ± 0.021
	TabPFN	0.769 ± 0.023	0.780 ± 0.024	0.698 ± 0.027	0.766 ± 0.025
	TabPFN-CE	**0.841 ± 0.033**	**0.839 ± 0.037**	**0.789 ± 0.037**	**0.840 ± 0.034**

TabDDPM-S is not included in Table 2. It is an intermediate configuration that contains only modifications (a)–(c) and is used in the ablation study.

5.2. Classification Results

Among the four target classes, the None class is by far the smallest, with only 35 instances across all partitions. This scarcity is the primary driver of the low balanced accuracy and G-mean values observed on the original dataset, since classifiers trained without correction tend to default to the majority action classes and rarely predict the minority outcome correctly.

This imbalance carries a direct operational cost in the maintenance context addressed by this study. A missed maintenance action, meaning an instance where Inspection, Lubrication or Replacement was truly required but the model predicts None, corresponds to a false negative with potential airworthiness and safety implications, since a genuine defect condition would go unaddressed until the next scheduled check. Conversely, an unnecessary maintenance action, meaning an instance where no intervention was needed but the model predicts one of the three action classes, corresponds to a false positive that generates avoidable downtime and labor cost. Standard accuracy is a poor guide for this trade-off, because a classifier that almost always predicts the majority classes can still report a deceptively high overall accuracy while performing poorly on both of these operationally meaningful error types. This is the motivation for reporting balanced accuracy, macro F1, MCC and G-mean in Table 3, since all four metrics weight the minority None class equally with the majority action classes rather than letting it be absorbed into the overall error rate.

Table 3 reports these four metrics for each classifier trained on the original dataset and on five augmented datasets. Four of them come from the family representatives and the fifth from TabDiff.

The results obtained on the original dataset confirm the severity of the class imbalance problem in the aircraft maintenance dataset used in this study. Balanced accuracy remains below 0.40 for every classifier except TabPFN and TabPFN-CE. G-mean falls below 0.05 for RF, XGBoost and LightGBM and also for NODE and TabNet. Given the class distribution described above, such low G-mean values indicate that these classifiers assign nearly all instances to the majority action classes and fail almost completely to recognize the rare None condition, which in practice would translate into a high rate of unnecessary maintenance triggers rather than missed defects. This is the core problem the proposed augmentation strategy aims to address.

Among the models trained without augmentation, TabPFN-CE already stands out with a balanced accuracy of 0.454 and a G-mean of 0.439, both considerably higher than any other classifier. This gap points to an inherent advantage of the TabPFN architecture combined with class-stratified context selection when coping with skewed label distributions, even before any resampling is applied.

Once augmentation is introduced every classifier benefits, but the magnitude of the gain depends strongly on the method used. TVAE produces the smallest improvement for most classifiers. For instance, the G-mean of XGBoost reaches only 0.192 under TVAE but climbs to 0.410 under SMOTE and 0.465 under TabDDPM-CC. SMOTE and Gaussian Copula yield comparable results overall, although Gaussian Copula edges ahead for most classifiers. CatBoost reaches a balanced accuracy of 0.552 with SMOTE against 0.608 with Gaussian Copula, a pattern repeated across most of the remaining models.

TabDDPM-CC and TabDiff give the strongest results in the table. TabDiff is slightly ahead for three tree ensembles and for TabNet and GANDALF, while TabDDPM-CC is ahead for the remaining seven classifiers, including both TabPFN variants. The combination of TabDDPM-CC with TabPFN-CE achieves the best overall performance, with a balanced accuracy of 0.841 and a macro F1 of 0.839. Its MCC is 0.789 and its G-mean 0.840. These values exceed those of the same classifier on the original data by 0.387 in balanced accuracy and 0.401 in G-mean. With the same classifier, TabDiff reaches a balanced accuracy of 0.818.

In practical terms, this level of balanced accuracy means the model correctly separates the rare None cases from genuine maintenance needs at a rate that would substantially cut both unnecessary maintenance triggers and missed defect cases relative to the original imbalanced model. These results suggest that class-conditional diffusion captures the minority maintenance events more faithfully than the other resampling strategies evaluated here.

At the model level a consistent ranking emerges regardless of the augmentation technique applied. TabPFN-CE and TabPFN occupy the top two positions in almost every column. FT-Transformer, GANDALF and ExcelFormer follow among the deep tabular architectures, and CatBoost leads the classical ensemble models. TabNet and XGBoost remain the weakest performers throughout the table, with TabNet in particular showing high variance across seeds as reflected by its wide standard deviations.

Figure 2 complements Table 3 with accuracy, macro precision and macro ROC-AUC for the same classifiers and training sets. Macro recall is not shown because it equals balanced accuracy in this setting and is already reported in Table 3. In the accuracy and precision panels, the original-data column is the lightest for every classifier. This shows how strongly the imbalance depresses these conventional metrics. In all three panels, the darkest cell of each row lies in the TabDDPM-CC or the TabDiff column. TVAE gives the weakest augmented result for every classifier.

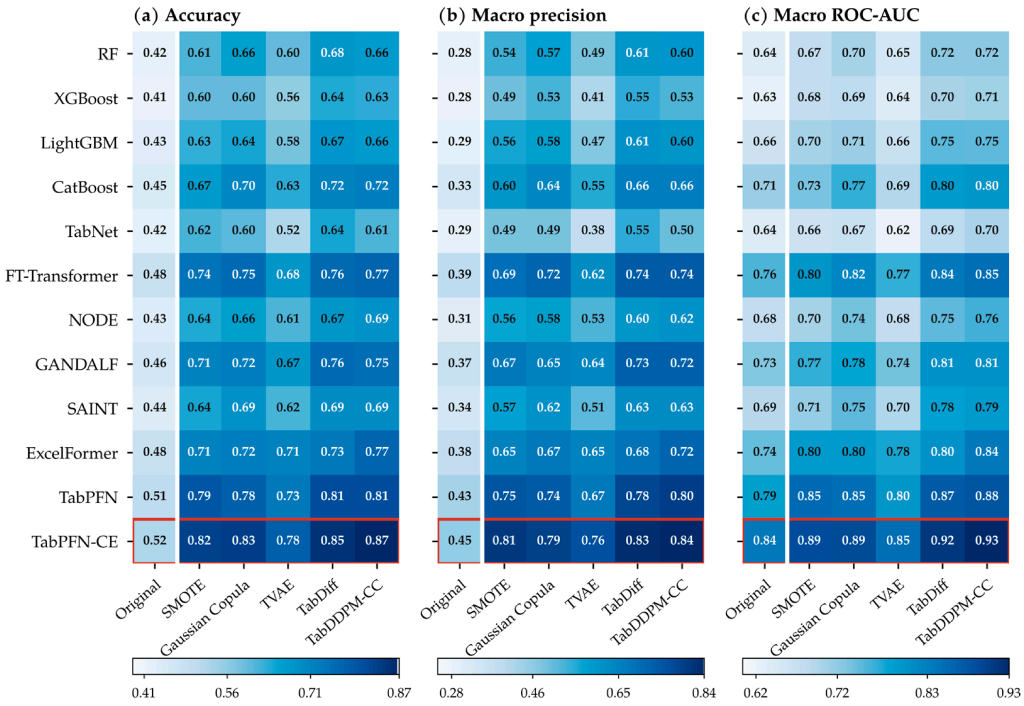


Figure 2. Heatmaps of (**a**) accuracy, (**b**) macro precision and (**c**) macro ROC-AUC for the twelve classifiers trained on the original and augmented aircraft data (mean over ten seeds). Darker cells indicate higher values.

The ROC-AUC panel shows a markedly different picture. The gap between the original and the augmented columns is much smaller, and for three classifiers at least one augmented training set yields an ROC-AUC at or below that of the original data. This indicates that ROC-AUC is comparatively insensitive to the choice of resampling technique once some form of augmentation is applied, and its ability to distinguish between strategies is limited compared with balanced accuracy, macro F1 or G-mean. This is expected given the class distribution of the dataset, since ROC-AUC is computed from ranked class probabilities and remains comparatively forgiving of a classifier that ranks the rare None class low as long as the more frequent action classes are ordered correctly. This is the main reason this

study relies primarily on the imbalance-sensitive metrics of Table 3 rather than ROC-AUC alone when comparing augmentation methods.

Across Table 3 and Figure 2, TabPFN-CE trained on data generated with TabDDPM-CC emerges as the configuration best suited to the aircraft maintenance classification task addressed in this study.

5.3. Ablation Study

Ablation studies were conducted on the generator, the classifier and the input features. Macro recall is not reported in the ablation tables because it is identical to balanced accuracy in a macro-averaged multiclass setting.

5.3.1. Generator-Side Ablation

Table 4 compares the three generator configurations, TabDDPM, TabDDPM-S and TabDDPM-CC, while the classifier is held fixed at the proposed TabPFN-CE. Moving from the unmodified baseline to the stability-enhanced TabDDPM-S produces a moderate gain across all seven metrics, with balanced accuracy rising from 0.806 to 0.818 and MCC from 0.749 to 0.768. This suggests that clipping, timestep normalization and EMA reduce part of the training instability of the per-class models.

Table 4. Cumulative ablation of the proposed generator, classifier fixed at TabPFN-CE, mean ± std over 10 seeds.

Data Augmentation Methods	Acc	Balanced Acc	Macro P	Macro F1	MCC	G-Mean	Macro ROC-AUC
TabDDPM	0.841 ± 0.014	0.806 ± 0.010	0.817 ± 0.045	0.807 ± 0.026	0.749 ± 0.020	0.803 ± 0.010	0.908 ± 0.008
TabDDPM-S	0.854 ± 0.010	0.818 ± 0.013	0.843 ± 0.037	0.828 ± 0.023	0.768 ± 0.015	0.815 ± 0.014	0.921 ± 0.009
TabDDPM-CC	0.866 ± 0.024	0.841 ± 0.033	0.838 ± 0.044	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.930 ± 0.008

The second increment, from TabDDPM-S to the proposed TabDDPM-CC, yields the larger of the two gains, raising balanced accuracy by a further 0.023 and G-mean by 0.025. Given that the minority class in this dataset is the None condition, this pattern suggests that the second group of modifications, led by class conditioning, primarily improves the representation of the scarcest maintenance outcome rather than uniformly inflating performance across all four classes. The ordering TabDDPM < TabDDPM-S < TabDDPM-CC holds for six of the seven metrics. Macro precision is the exception, as TabDDPM-S (0.843) is slightly above TabDDPM-CC (0.838). The cumulative design shows the effect of each group of modifications but cannot separate the individual modifications within a group.

Table 5 reports the leave-one-out ablation, in which each modification is removed from the full TabDDPM-CC while the other five are kept. Removing the class-conditional single model (d) causes the largest loss. Balanced accuracy falls from 0.841 to 0.826 and propensity AUC rises from 0.565 to 0.585, so both utility and fidelity deteriorate. Removing range clipping (a), EMA (c) or DCR filtering (f) mainly harms fidelity, with propensity AUC rising to between 0.574 and 0.583. Their effect on classification is small and goes in both directions. Without (a) or (c), balanced accuracy is even slightly higher than for the full model. Timestep normalization (b) and the cosine schedule (e) leave both fidelity and utility essentially unchanged. None of the six removals changes macro F1 significantly after Holm correction.

These results refine the cumulative picture. The stability modifications (a)–(c) help per-class diffusion models, as shown by the gain from TabDDPM to TabDDPM-S. Once a single class-conditional network is used, they become largely redundant for classification and act mainly on fidelity. Class conditioning is therefore the generator component with

the largest individual effect. Even this effect is not significant on its own at ten seeds, and the difference between TabDDPM-CC and TabDDPM-S is not significant either (see Section 5.4). The ablation findings thus describe consistent tendencies rather than confirmed individual effects.

Table 5. Leave-one-out ablation of the six TabDDPM-CC modifications, classifier fixed at TabPFN-CE (mean ± std over 10 seeds). Holm-adjusted p-values refer to paired two-sided Wilcoxon tests on macro F1 against the full model ($m = 6$).

Configuration	Balanced Acc	Macro F1	MCC	G-Mean	Recall (None)	Propensity AUC ↓	Holm-Adj. p	Sig. ($\alpha = 0.05$)
TabDDPM-CC (all six)	0.841 ± 0.033	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.714 ± 0.165	0.565 ± 0.022	–	–
w/o (a) range clipping	0.846 ± 0.035	0.847 ± 0.036	0.792 ± 0.039	0.848 ± 0.032	0.729 ± 0.157	0.575 ± 0.014	1.000	X
w/o (b) timestep normalization	0.846 ± 0.038	0.840 ± 0.037	0.785 ± 0.036	0.849 ± 0.035	0.743 ± 0.176	0.564 ± 0.019	1.000	X
w/o (c) EMA	0.852 ± 0.037	0.848 ± 0.035	0.793 ± 0.040	0.857 ± 0.034	0.757± 0.179	0.583 ± 0.025	1.000	X
w/o (d) class-conditional single model	0.826 ± 0.029	0.825 ± 0.033	0.774 ± 0.034	0.830 ± 0.031	0.686 ± 0.162	0.585 ± 0.031	1.000	X
w/o (e) cosine schedule	0.841 ± 0.034	0.839 ± 0.034	0.785 ± 0.030	0.842 ± 0.035	0.714 ± 0.165	0.565 ± 0.011	1.000	X
w/o (f) DCR filtering	0.836 ± 0.036	0.831 ± 0.038	0.774 ± 0.037	0.836 ± 0.037	0.714 ± 0.190	0.574 ± 0.018	1.000	X

Note: ↓ indicates that lower values are better. X indicates that the difference is not significant after Holm correction ($\alpha = 0.05$).

The DCR filter (f) is examined further in Appendix B, Table A4. Removing it lowers the DCR share from 0.544 to 0.511 and the near-copy rate from 0.058 to 0.044. Coverage changes from 0.903 to 0.897 and the unique NN ratio from 0.679 to 0.683, so diversity is almost unchanged. The filter thus moves the synthetic records slightly toward the training data. In return, it gives small gains in fidelity and utility, and its effect on classification is not significant (Table 5). It keeps the near-copy rate close to the held-out reference and does not reduce diversity in a measurable way.

5.3.2. Classifier-Side Context Selection and Backbone Ablation

Table 6 compares the five classifier configurations while the generator is held fixed at the proposed TabDDPM-CC.

Table 6. Cumulative ablation of the proposed classifier, generator fixed at TabDDPM-CC, mean ± std over 10 seeds.

Classifier/ Metric	Acc	Balanced Acc	Macro P	Macro F1	MCC	G-Mean	Macro ROC-AUC
TabPFN (baseline)	0.809 ± 0.017	0.769 ± 0.023	0.797 ± 0.032	0.780 ± 0.024	0.698 ± 0.027	0.766 ± 0.025	0.882 ± 0.010
TabPFN-R	0.805 ± 0.020	0.753 ± 0.029	0.787 ± 0.044	0.766 ± 0.030	0.691 ± 0.032	0.747 ± 0.031	0.864 ± 0.008
TabPFN-KM	0.835 ± 0.013	0.796 ± 0.012	0.789 ± 0.039	0.790 ± 0.022	0.740 ± 0.020	0.794 ± 0.012	0.894 ± 0.012
TabPFN-KC	0.849 ± 0.009	0.811 ± 0.006	0.811 ± 0.030	0.809 ± 0.016	0.761 ± 0.014	0.809 ± 0.006	0.914 ± 0.010
TabPFN-CE	0.866 ± 0.024	0.841 ± 0.033	0.838 ± 0.044	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.930 ± 0.008

Replacing the fixed default context of TabPFN with a class-stratified random sample, TabPFN-R, does not improve performance and in fact produces a small decrease relative to the baseline, with balanced accuracy falling from 0.769 to 0.753 and G-mean from 0.766 to 0.747. This suggests that varying the context composition without an explicit selection criterion does not address the limitations of a fixed context, and in this dataset random sampling introduces additional variance that offsets any potential benefit of stratification.

This is a relevant finding for the target application, since a context selection strategy that increases variance without a systematic benefit would be an unreliable basis for a maintenance decision support tool, where consistent behavior across retraining runs matters as much as average performance.

Introducing an explicit selection criterion reverses this trend. TabPFN-KM, which selects context points through k-means centroids, raises balanced accuracy to 0.796, while TabPFN-KC, which selects context points through k-center greedy coverage, raises it further to 0.811 and yields the highest MCC and G-mean among the four non-ensemble variants. The gap between TabPFN-KM and TabPFN-KC indicates that optimizing for boundary coverage rather than density representativeness is more effective for a context of fixed size. This is consistent with the structure of the dataset, since a context built from density-representative points is more likely to be dominated by the frequent Lubrication and Inspection classes, while a coverage-based context is more likely to retain examples from the rare None and Replacement classes near the decision boundary.

The proposed TabPFN-CE, which combines multiple k-center contexts drawn with different seeds through posterior averaging, achieves the best result on all seven metrics. The gain over the single-context TabPFN-KC is smaller than the gain obtained by introducing the k-center criterion itself, indicating that most of the benefit of the proposed context construction strategy comes from the coverage-based selection rather than from the ensembling step, while the ensembling step still provides a consistent additional improvement. TabPFN-CE also shows a higher standard deviation than TabPFN-KC on several metrics, for instance an MCC standard deviation of 0.037 against 0.014, an expected consequence of averaging posteriors from a small number of independently sampled contexts. Section 5.4 tests whether this margin is significant.

Table 7 repeats the comparison between the single default context and TabPFN-CE with the newer TabPFN v2, v2.5 and v3 backbones. All other settings follow the main protocol. Newer backbones improve both configurations. On TabDDPM-CC data, the single-context model rises from 0.769 with v1 to 0.824 with v3, and TabPFN-CE rises from 0.841 to 0.860. The gain of TabPFN-CE over the single context shrinks as the backbone improves, from 0.072 with v1 to 0.036 with v3. It remains significant for v1 and v2 but not for v2.5 and v3 after Holm correction. On the original data, all backbones stay below a balanced accuracy of 0.51. Augmentation therefore matters far more than the backbone version, and the ordering of the two context strategies is the same for every version. This pattern supports the use of v1 for the main analysis. The context-construction problem is most severe for v1, and the conclusions on augmentation do not depend on this choice.

Table 7. Backbone sensitivity of the context-construction strategy with TabPFN v1, v2, v2.5 and v3 (mean ± std over 10 seeds). Holm-adjusted p-values refer to paired two-sided Wilcoxon tests on macro F1 between TabPFN-CE and the single-context model ($m = 8$).

Training Data	Backbone	Configuration	Balanced Acc	Macro F1	MCC	G-Mean	Holm-Adj. p	Sig. ($\alpha = 0.05$)
Original	TabPFN v1 (main)	Single context	0.428 ± 0.013	0.425 ± 0.015	0.226 ± 0.027	0.410 ± 0.021		
		TabPFN-CE	0.454 ± 0.006	0.452 ± 0.008	0.253 ± 0.020	0.439 ± 0.005	0.016	✓
	TabPFN v2	Single context	0.472 ± 0.014	0.469 ± 0.015	0.283 ± 0.031	0.458 ± 0.022		
		TabPFN-CE	0.487 ± 0.009	0.484 ± 0.011	0.299 ± 0.025	0.472 ± 0.008	0.111	X
	TabPFN v2.5	Single context	0.486 ± 0.012	0.484 ± 0.013	0.298 ± 0.026	0.472 ± 0.019		
		TabPFN-CE	0.497 ± 0.008	0.492 ± 0.010	0.318 ± 0.022	0.483 ± 0.007	0.168	X
	TabPFN v3	Single context	0.493 ± 0.013	0.491 ± 0.014	0.308 ± 0.029	0.478 ± 0.021		
		TabPFN-CE	0.503 ± 0.008	0.502 ± 0.010	0.325 ± 0.022	0.489 ± 0.007	0.168	X

Table 7. *Cont.*

Training Data	Backbone	Configuration	Balanced Acc	Macro F1	MCC	G-Mean	Holm-Adj. p	Sig. ($\alpha = 0.05$)
TabDDPM-CC	TabPFN v1 (main)	Single context	0.769 ± 0.023	0.780 ± 0.024	0.698 ± 0.027	0.766 ± 0.025		
		TabPFN-CE	0.841 ± 0.033	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.027	✓
	TabPFN v2	Single context	0.806 ± 0.021	0.815 ± 0.022	0.747 ± 0.025	0.803 ± 0.023		
		TabPFN-CE	0.850 ± 0.029	0.847 ± 0.032	0.799 ± 0.032	0.848 ± 0.030	0.027	✓
	TabPFN v2.5	Single context	0.819 ± 0.020	0.824 ± 0.021	0.759 ± 0.024	0.817 ± 0.022		
		TabPFN-CE	0.857 ± 0.027	0.854 ± 0.030	0.801 ± 0.030	0.856 ± 0.028	0.098	X
	TabPFN v3	Single context	0.824 ± 0.019	0.832 ± 0.020	0.767 ± 0.023	0.821 ± 0.021		
		TabPFN-CE	0.860 ± 0.026	0.861 ± 0.029	0.810 ± 0.029	0.859 ± 0.027	0.109	X

Note: ✓ indicates a significant difference after Holm correction ($\alpha = 0.05$), and X indicates a non-significant one.

5.3.3. Feature Engineering Ablation

The final ablation isolates the individual and joint contribution of the engineered features introduced in Section 4.4 and the proposed augmentation method TabDDPM-CC, using a 2 × 2 factorial design with the classifier fixed at TabPFN-CE. Four configurations are compared: the original six input variables without feature engineering or augmentation (A), the original six variables enriched with the six engineered features but without augmentation (B), the original six variables augmented with TabDDPM-CC but without feature engineering (C), and the proposed configuration combining both augmentation and feature engineering (D). Table 8 reports the results.

Table 8. Factorial ablation of feature engineering and augmentation, classifier fixed at TabPFN-CE, mean ± std over 10 seeds.

Condition	Acc	Balanced Acc	Macro P	Macro F1	MCC	G-Mean	Macro ROC-AUC
A: Original (6 features, no augmentation)	0.480 ± 0.011	0.413 ± 0.007	0.412 ± 0.016	0.410 ± 0.009	0.207 ± 0.022	0.391 ± 0.009	0.828 ± 0.013
B: Feature engineering only	0.524 ± 0.010	0.454 ± 0.006	0.453 ± 0.015	0.452 ± 0.008	0.253 ± 0.020	0.439 ± 0.005	0.840 ± 0.012
C: Augmentation only	0.722 ± 0.023	0.694 ± 0.026	0.699 ± 0.033	0.690 ± 0.029	0.589 ± 0.033	0.687 ± 0.030	0.880 ± 0.018
D: Proposed (both)	0.866 ± 0.024	0.841 ± 0.033	0.838 ± 0.044	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.930 ± 0.008

Comparing B to A shows that feature engineering alone provides a measurable but modest gain, with balanced accuracy rising from 0.413 to 0.454. This is consistent with the limited representation of the None class in the original training data, since a sample of 24 instances offers little statistical support for the ratio and rate-based features, which constrains the amount of additional signal these features can extract without augmentation.

Comparing C to A shows that augmentation alone produces a substantially larger gain, raising balanced accuracy from 0.413 to 0.694. This is consistent with the central role of augmentation and indicates that the scarcity of minority instances is the dominant bottleneck in this classification task, ahead of the input feature representation.

The comparison of most interest is D against C, which isolates the contribution of feature engineering once the class imbalance has already been addressed by augmentation. Here balanced accuracy rises from 0.694 to 0.841, a gain of 0.147, which is more than three times larger than the gain obtained from feature engineering without augmentation. This pattern indicates an interaction effect rather than a purely additive one. Once TabDDPM-CC supplies enough synthetic examples of the None and Replacement classes, the engineered features can be computed and learned from a sufficiently large and balanced sample, and their contribution to the classification boundary becomes considerably stronger. In other words, the benefit of feature engineering depends on the class distribution it is applied to. It requires enough minority class examples to make the derived ratios and rates informative

rather than noisy. The overall gain of the proposed configuration D over the original configuration A, an increase of 0.428 in balanced accuracy, exceeds the sum of the two individual effects observed in isolation, which further supports the presence of a synergistic rather than additive relationship between the two components.

Table 9 removes each engineered feature in turn from the final pipeline. Oil_Efficiency has the largest individual effect. Without it, balanced accuracy falls from 0.841 to 0.810 and None recall from 0.714 to 0.657. Removing Maint_Aging_Ratio or Hours_per_Cycle also lowers macro F1 significantly after Holm correction. Stress_Index shows a smaller loss that does not reach significance. Vibration_Anomaly and High_Risk_State can be removed without a measurable loss. Their information appears to be carried largely by the raw vibration and temperature readings and by Stress_Index. Removing all six derived features lowers balanced accuracy to 0.694, which equals configuration C in Table 8. The engineered set is therefore useful as a whole, but two of its six members are redundant in the present data.

Table 9. Leave-one-feature-out ablation of the engineered features with TabDDPM-CC augmentation and the TabPFN-CE classifier (mean ± std over 10 seeds). Raw and Holm-adjusted p-values refer to paired two-sided Wilcoxon tests on macro F1 against the full twelve-feature model ($m = 7$).

Condition	Balanced Acc	Macro F1	MCC	G-Mean	Recall (None)	Recall (Replacement)	Raw p	Holm-Adj. p	Sig.
K0: all twelve features	0.841 ± 0.033	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.714 ± 0.165	0.938 ± 0.059	–	–	–
K1: without Hours_per_Cycle	0.826 ± 0.036	0.823 ± 0.040	0.771 ± 0.040	0.824 ± 0.037	0.686 ± 0.176	0.925 ± 0.065	0.010	0.039	✓
K2: without Stress_Index	0.830 ± 0.035	0.828 ± 0.038	0.778 ± 0.039	0.828 ± 0.036	0.700 ± 0.171	0.931 ± 0.062	0.020	0.059	X
K3: without Oil_Efficiency	0.810 ± 0.040	0.805 ± 0.043	0.748 ± 0.044	0.806 ± 0.042	0.657 ± 0.181	0.912 ± 0.067	0.002	0.014	✓
K4: without Maint_Aging_Ratio	0.816 ± 0.038	0.812 ± 0.041	0.757 ± 0.042	0.812 ± 0.040	0.671 ± 0.179	0.919 ± 0.066	0.004	0.020	✓
K5: without Vibration_Anomaly	0.838 ± 0.034	0.837 ± 0.037	0.786 ± 0.038	0.838 ± 0.035	0.714 ± 0.165	0.938 ± 0.059	0.557	0.557	X
K6: without High_Risk_State	0.835 ± 0.035	0.833 ± 0.038	0.781 ± 0.039	0.833 ± 0.036	0.700 ± 0.171	0.931 ± 0.062	0.275	0.551	X
K7: without all six derived features	0.694 ± 0.026	0.690 ± 0.029	0.589 ± 0.033	0.687 ± 0.030	0.443 ± 0.142	0.819 ± 0.086	0.002	0.014	✓

Note: ✓ indicates a significant difference after Holm correction ($\alpha = 0.05$), and X indicates a non-significant one.

5.4. Statistical Significance Analysis

The results of Sections 5.2 and 5.3 identify TabDDPM-CC and TabPFN-CE as the augmentation method and classifier with the highest mean performance. This section tests whether these advantages are stable across the ten resampled partitions. Paired two-sided Wilcoxon signed-rank tests on macro F1 were used [29], and the Holm–Bonferroni procedure controlled the family-wise error rate within each family of comparisons at $\alpha = 0.05$ [30]. Because the seeds share the same 647 records, the tests describe stability within this dataset rather than independent replication. The training and test partitions of different seeds overlap, so the seed-level results are not independent. The p-values therefore show whether a difference recurs across partitions, not how often it would hold in new data. For the same reason, the comparisons below are interpreted through the size and direction of the mean differences as well as the test outcome, and superiority is claimed only where the advantage remains significant after Holm correction.

The first comparison addresses the augmentation dimension. TabDDPM-CC paired with TabPFN-CE was compared with the original dataset and with the eleven other augmentation methods, including TabDDPM-S. Table 10 reports the results. TabDDPM-CC is

significantly better than the original data and seven of the eleven generators after Holm correction. The differences from SMOTE and TabDiff and from the two earlier TabDDPM variants are not significant. These four comparisons call for a cautious reading. TabDDPM and TabDDPM-S share the diffusion backbone of the proposed method, and their gaps to TabDDPM-CC were already modest in the ablation. TabDiff is a recent generator that comes close to TabDDPM-CC on both quality and utility. SMOTE remains a competitive baseline on this dataset even though it trails TabDDPM-CC by 0.055 in balanced accuracy. The evidence therefore supports a consistent advantage of TabDDPM-CC over most generators, but not a significant superiority over these four.

Table 10. Two-sided Wilcoxon signed-rank tests on macro F1 with Holm correction, TabDDPM-CC versus comparator augmentation methods with the TabPFN-CE classifier (10 seeds).

Comparator Augmentation Method	Raw p-Value	Holm-Adjusted p-Value	Significant ($\alpha = 0.05$)
Original Dataset	0.002	0.023	✓
SMOTE	0.027	0.109	X
ADASYN	0.004	0.023	✓
BGM	0.002	0.023	✓
Gaussian Copula	0.010	0.049	✓
CopulaGAN	0.002	0.023	✓
CTGAN	0.002	0.023	✓
TVAE	0.002	0.023	✓
TabDDPM	0.084	0.252	X
TabDDPM-S	0.322	0.551	X
TabSyn	0.002	0.023	✓
TabDiff	0.275	0.551	X

Note: With ten paired seeds, 0.002 is the smallest attainable two-sided exact p-value. Holm adjustment is applied across the twelve comparisons of this table. ✓ indicates a significant difference after Holm correction, and X indicates a non-significant one.

The second comparison examines the classifier dimension. TabPFN-CE trained on TabDDPM-CC data was compared with the eleven baseline classifiers and with the three single-context TabPFN variants. Table 11 summarizes the outcome. TabPFN-CE is significantly better than every classifier except TabPFN-KC, for which the Holm-adjusted p-value of 0.055 narrowly misses the threshold. The coverage-based context therefore gives a stable improvement over gradient boosting and deep tabular architectures on this dataset. Its margin over the single k-center context is consistent in direction but not significant at ten seeds.

Table 11. Two-sided Wilcoxon signed-rank tests on macro F1 with Holm correction, TabPFN-CE versus comparator classifiers on TabDDPM-CC augmented data (10 seeds).

Comparator Classifier	Raw p-Value	Holm-Adjusted p-Value	Significant ($\alpha = 0.05$)
RF	0.002	0.027	✓
XGBoost	0.002	0.027	✓
LightGBM	0.002	0.027	✓
CatBoost	0.002	0.027	✓
TabNet	0.002	0.027	✓
FT-Transformer	0.002	0.027	✓
NODE	0.002	0.027	✓
GANDALF	0.002	0.027	✓
SAINT	0.002	0.027	✓
ExcelFormer	0.002	0.027	✓
TabPFN	0.004	0.027	✓
TabPFN-R	0.010	0.029	✓
TabPFN-KM	0.014	0.029	✓
TabPFN-KC	0.049	0.055	X

Note: ✓ indicates a significant difference after Holm correction ($\alpha = 0.05$), and X indicates a non-significant one.

A third comparison addresses the factorial design of Section 5.3.3. Four paired tests were conducted, each isolating one effect under one level of the other factor, with Holm correction across the four comparisons. Table 12 reports the results. All four comparisons remain significant after correction. Both feature engineering and augmentation thus have a seed-stable effect whether or not the other component is present. These tests confirm each simple effect but do not test the interaction itself, which rests on the size of the gains in Table 8.

Table 12. Two-sided Wilcoxon signed-rank tests on macro F1 with Holm correction, factorial comparison of feature engineering and augmentation effects (10 seeds).

Comparison	Effect Tested	Raw p-Value	Holm-Adjusted p-Value	Significant ($\alpha = 0.05$)
B vs. A	Feature engineering, without augmentation	0.002	0.008	✓
D vs. C	Feature engineering, with augmentation	0.002	0.008	✓
C vs. A	Augmentation, without feature engineering	0.002	0.008	✓
D vs. B	Augmentation, with feature engineering	0.002	0.008	✓

Note: ✓ indicates a significant difference after Holm correction ($\alpha = 0.05$), and X indicates a non-significant one.

Taken together, these tests support the central design choices within the limits of this dataset. The advantage of TabPFN-CE over nearly the entire classifier pool and that of TabDDPM-CC over most generators are stable across the ten partitions. The non-significant comparisons with TabPFN-KC and with SMOTE, TabDiff and the earlier TabDDPM variants mark the points where the evidence for superiority is weakest.

5.5. Sensitivity and Robustness Analysis

5.5.1. Augmentation Ratio

The proposed pipeline enlarges each minority class up to a fixed multiple of its real size, capped at the size of the majority class. This augmentation ratio was set to 4.0 before the experiments. To assess this choice, TabPFN-CE was retrained on data augmented by TabDDPM-CC for six ratios between 1 and 5, with all other components of the pipeline held fixed. A ratio of 1 adds no synthetic records and reproduces the original-data result. Table 13 reports the macro-averaged metrics together with the per-class recall, which is needed because the ratio effect differs across the four categories.

Performance increases steadily as the ratio grows from 1 to 4 and then declines at ratio 5. Balanced accuracy rises from 0.454 at ratio 1 to a peak of 0.841 at ratio 4 and falls back to 0.793 at ratio 5. Every aggregate metric in Table 13 peaks at ratio 4, including MCC (0.789) and G-mean (0.840). The ratio fixed in advance therefore coincides with the best of the six values. Because this analysis uses the test partitions, it is reported as a descriptive check and was not used to select the ratio.

The per-class recall values show that the pattern is not uniform across the four maintenance categories. Recall for None rises from 0.329 at ratio 1 to 0.714 at ratio 4. Replacement recall shows an even larger rise, from 0.394 to 0.938. Inspection recall plateaus early at about 0.88 from ratio 3 onward. Lubrication improves more gradually, from 0.597 to 0.825.

Table 13. Sensitivity of TabPFN-CE performance to the TabDDPM-CC augmentation ratio, mean ± std over 10 seeds. Columns correspond to the six augmentation ratio values evaluated.

Metric	Ratio 1	Ratio 1.5	Ratio 2	Ratio 3	Ratio 4	Ratio 5
Acc	0.524 ± 0.010	0.682 ± 0.029	0.742 ± 0.045	0.809 ± 0.035	0.866 ± 0.024	0.834 ± 0.023
Balanced Acc	0.454 ± 0.006	0.604 ± 0.021	0.669 ± 0.065	0.737 ± 0.059	0.841 ± 0.033	0.793 ± 0.066
Macro P	0.453 ± 0.015	0.584 ± 0.054	0.697 ± 0.085	0.735 ± 0.068	0.838 ± 0.044	0.800 ± 0.041
Macro F1	0.452 ± 0.008	0.581 ± 0.033	0.665 ± 0.066	0.730 ± 0.060	0.839 ± 0.037	0.781 ± 0.059
MCC	0.253 ± 0.020	0.508 ± 0.038	0.598 ± 0.073	0.705 ± 0.053	0.789 ± 0.037	0.746 ± 0.034
G-Mean	0.439 ± 0.005	0.557 ± 0.057	0.587 ± 0.205	0.653 ± 0.225	0.840 ± 0.034	0.755 ± 0.104
Macro ROC-AUC	0.840 ± 0.012	0.806 ± 0.013	0.824 ± 0.025	0.846 ± 0.024	0.930 ± 0.008	0.868 ± 0.022
Recall (None)	0.329 ± 0.069	0.343 ± 0.181	0.343 ± 0.168	0.414 ± 0.184	0.714 ± 0.165	0.543 ± 0.284
Recall (Inspection)	0.498 ± 0.047	0.600 ± 0.092	0.782 ± 0.074	0.884 ± 0.054	0.886 ± 0.048	0.882 ± 0.041
Recall (Lubrication)	0.597 ± 0.040	0.773 ± 0.076	0.740 ± 0.057	0.786 ± 0.053	0.825 ± 0.048	0.805 ± 0.063
Recall (Replacement)	0.394 ± 0.059	0.700 ± 0.105	0.812 ± 0.072	0.863 ± 0.101	0.938 ± 0.059	0.944 ± 0.046

Read through the cost asymmetry described in Section 5.2, the gains for Lubrication and Replacement matter at least as much as the gain for None. Both concern the more consequential type of error, in which a required action is missed. A lubrication need left unaddressed can accelerate wear, and an undetected replacement need can progress toward a functional failure.

The decline observed at ratio 5 is consistent with an oversampling effect rather than a random fluctuation, since it appears jointly across accuracy, balanced accuracy, MCC and G-mean and is accompanied by a drop in None recall from 0.714 to 0.543. Generating synthetic samples well beyond the size of the available real minority data increases the risk that the generator extrapolates into regions of the feature space that are not well supported by real observations, particularly for a class with only 24 original training instances. Once the volume of synthetic None examples substantially exceeds what the real data can constrain, the classifier is more likely to learn decision boundaries shaped by generator artifacts rather than by genuine class-conditional structure, which is consistent with the higher standard deviation of None recall at ratio 5 (0.284) compared with ratio 4 (0.165). Replacement recall continues to improve slightly at ratio 5 while None recall deteriorates, which indicates that the two classes respond differently to over-augmentation and that a single global ratio involves a trade-off across classes rather than a uniform effect.

Taken together, these results are consistent with the choice of ratio 4 as the operating point for the proposed pipeline. The sensitivity analysis shows that performance is not brittle around this value, since ratios 3 and 5 both remain well above the unaugmented baseline of ratio 1, but it also shows that the specific choice of ratio is not arbitrary, since moving away from 4 in either direction measurably reduces performance on the metrics most relevant to the class imbalance problem, and to the higher-cost misclassifications, addressed in this study.

5.5.2. Sensor and Label Noise

Table 14 reports the effect of sensor and label noise injected into the training partition before augmentation. Sensor noise lowers performance gradually. At 20% sensor noise, the balanced accuracy of the TabDDPM-CC pipeline falls from 0.841 to 0.797, while SMOTE falls from 0.786 to 0.754. Label noise is more harmful. With 20% of the training labels changed, TabDDPM-CC drops to 0.618 and SMOTE to 0.611. The original-data pipeline changes little because it starts close to its floor. TabDDPM-CC stays above SMOTE at every level but loses more in absolute terms. Its margin shrinks from 0.055 on clean data to 0.007 at 20%

label noise. This pattern is consistent with a moderate amplification of label errors by the generator. A class-conditional model learns the class regions from the labels it is given, so corrupted labels blur these regions in the synthetic data as well. None of the differences from SMOTE is significant after Holm correction across the seven settings. In practice, label auditing such as confident learning [70] should precede augmentation when maintenance logs may contain recording errors.

Table 14. Effect of sensor and label noise injected into the training partition on the TabPFN-CE pipeline (mean ± std over 10 seeds). The last column gives Holm-adjusted p-values of paired two-sided Wilcoxon tests on macro F1 between TabDDPM-CC and SMOTE ($m = 7$).

	Original		SMOTE		TabDDPM-CC		Holm-Adj. p
Noise Setting	**Balanced Acc**	**G-Mean**	**Balanced Acc**	**G-Mean**	**Balanced Acc**	**G-Mean**	**CC vs. SMOTE**
Clean	0.454 ± 0.006	0.439 ± 0.005	0.786 ± 0.011	0.784 ± 0.010	0.841 ± 0.033	0.840 ± 0.034	0.068
Sensor noise, 5%	0.452 ± 0.011	0.435 ± 0.010	0.781 ± 0.014	0.778 ± 0.013	0.831 ± 0.036	0.829 ± 0.037	0.082
Sensor noise, 10%	0.449 ± 0.012	0.431 ± 0.012	0.772 ± 0.016	0.769 ± 0.015	0.820 ± 0.039	0.817 ± 0.040	0.098
Sensor noise, 20%	0.443 ± 0.014	0.423 ± 0.015	0.754 ± 0.021	0.750 ± 0.020	0.797 ± 0.044	0.793 ± 0.046	0.148
Label noise, 5%	0.448 ± 0.014	0.424 ± 0.020	0.741 ± 0.024	0.736 ± 0.025	0.768 ± 0.040	0.763 ± 0.043	0.252
Label noise, 10%	0.440 ± 0.017	0.409 ± 0.024	0.694 ± 0.031	0.686 ± 0.033	0.712 ± 0.048	0.703 ± 0.051	0.551
Label noise, 20%	0.425 ± 0.021	0.380 ± 0.031	0.611 ± 0.042	0.596 ± 0.047	0.618 ± 0.058	0.601 ± 0.063	0.770

5.5.3. Covariate Shift

Table 15 reports the covariate-shift experiment. Training used only records with Flight Hours at or below the 70th percentile, and testing used records above it. The original-data pipeline almost collapses under this shift, with a balanced accuracy of 0.401 and a G-mean of 0.126. TabDDPM-CC keeps a balanced accuracy of 0.685, well below its in-distribution value of 0.841 but far above the original data. The loss is concentrated in the None class, whose recall falls to 0.367, while Replacement recall stays at 0.844. Adding 20% of the shifted records to the TabPFN-CE context pool raises balanced accuracy to 0.781 and None recall to 0.500 without retraining the generator. Both differences are significant after Holm correction. Because TabPFN-CE learns in context, such a refresh only requires adding recent labeled records to the pool. This offers a practical route for adapting the classifier to a new fleet or operating regime, although it cannot replace retraining the generator when the shift is large.

Table 15. Performance under a controlled covariate shift in Flight Hours with the TabPFN-CE classifier (mean ± std over 10 seeds). Training records have Flight Hours at or below the 70th percentile and test records lie above it. Holm-adjusted p-values refer to paired two-sided Wilcoxon tests on macro F1 against TabDDPM-CC under shift ($m = 2$).

Configuration	**Balanced Acc**	**Macro F1**	**MCC**	**G-Mean**	**Recall (None)**	**Recall (Replacement)**	**Holm-Adj. p**
In-distribution test set (reference)							
TabDDPM-CC	0.841 ± 0.033	0.839 ± 0.037	0.789 ± 0.037	0.840 ± 0.034	0.714 ± 0.165	0.938 ± 0.059	–
Shifted test set (Flight Hours above the 70th percentile)							
Original	0.401 ± 0.040	0.372 ± 0.045	0.196 ± 0.060	0.126 ± 0.203	0.100 ± 0.161	0.281 ± 0.079	0.004
TabDDPM-CC	0.685 ± 0.052	0.668 ± 0.058	0.575 ± 0.070	0.597 ± 0.215	0.367 ± 0.189	0.844 ± 0.059	–
TabDDPM-CC + context refresh	0.781 ± 0.045	0.766 ± 0.048	0.706 ± 0.058	0.750 ± 0.067	0.500 ± 0.176	0.934 ± 0.045	0.004

5.6. Computational Cost

Table 16 separates the cost of the pipeline into an offline stage and a deployment stage. The offline stage runs once per training partition. TabDDPM-CC needs about 42 s on the laptop GPU to train and to generate the 308 synthetic records. This is about twice the time of the baseline TabDDPM because of the joint network and the DCR filtering buffer, but

it remains below TabSyn (72 s) and TabDiff (80 s). Because augmentation is needed only when the model is retrained, this cost does not affect deployment.

In the deployment stage, TabPFN-CE scores a batch of records at about 34 ms per record on the GPU. A single record takes about 3.7 s on the GPU and 26 s on the CPU because every query requires five forward passes over the stored contexts. Gradient-boosted trees answer in about 2 ms or less. TabPFN-CE is therefore not suited to on-board, real-time prognostics. It is intended as a ground-based decision-support tool that runs after each flight or during maintenance planning, where a few seconds per record is acceptable. The ensemble size R sets the trade-off. With $R = 1$, the cost equals that of TabPFN-KC and balanced accuracy is 0.811. With $R = 3$, balanced accuracy reaches 0.836 at about 60% of the batch cost of $R = 5$. TabPFN-KC is thus a cheaper alternative when latency matters more than the last increment of accuracy.

Table 16. Computational cost of the proposed pipeline on the aircraft dataset. Panel A covers the offline augmentation stage, which runs once per training partition (452 real and 308 synthetic records). Panel B covers the deployment stage, in which a trained classifier scores new maintenance records. All classifiers were trained on TabDDPM-CC-augmented data. Dashes mark settings that were not timed.

Panel A. Offline Augmentation Stage

Family	Method	Device	Training and Generation Time (s)
Interpolation-based	SMOTE	CPU	0.005 ± 0.005
	ADASYN	CPU	0.008 ± 0.008
Statistical	BGM	CPU	0.027 ± 0.004
	Gaussian Copula	CPU	0.203 ± 0.201
Deep generative	CopulaGAN	GPU	33.537 ± 2.729
	CTGAN	GPU	68.338 ± 4.777
	TVAE	GPU	37.673 ± 2.645
Diffusion-based	TabDDPM	GPU	20.750 ± 1.418
	TabDDPM-S	GPU	22.921 ± 1.767
	TabSyn	GPU	71.906 ± 7.351
	TabDiff	GPU	79.834 ± 7.126
	TabDDPM-CC	GPU	42.410 ± 2.201

Panel B. Deployment Stage

Family	Classifier	Balanced Acc	Fit Time (s)	Batch Scoring (ms per Record)	Single Record, GPU (ms)	Single Record, CPU (ms)
Tree-based ensembles	RF	0.553 ± 0.018	0.412 ± 0.019	0.318 ± 0.021	31.842 ± 1.213	33.106 ± 1.874
	XGBoost	0.509 ± 0.019	0.286 ± 0.014	0.031 ± 0.003	1.274 ± 0.082	–
	LightGBM	0.563 ± 0.024	0.198 ± 0.012	0.047 ± 0.004	1.846 ± 0.113	–
	CatBoost	0.633 ± 0.017	2.318 ± 0.104	0.022 ± 0.002	0.512 ± 0.034	0.524 ± 0.041
Differentiable tree ensembles	NODE	0.579 ± 0.030	27.614 ± 2.208	0.091 ± 0.007	7.213 ± 0.402	–
	GANDALF	0.671 ± 0.023	15.842 ± 1.371	0.417 ± 0.031	21.374 ± 1.226	–
Attention and transformer-based	TabNet	0.478 ± 0.044	14.926 ± 1.842	0.066 ± 0.006	4.318 ± 0.265	–
	FT-Transformer	0.709 ± 0.017	24.731 ± 2.166	0.214 ± 0.016	12.846 ± 0.738	–
	SAINT	0.608 ± 0.028	61.538 ± 6.204	0.392 ± 0.028	15.912 ± 0.884	–
	ExcelFormer	0.702 ± 0.022	33.472 ± 3.718	0.171 ± 0.013	9.742 ± 0.618	–
Prior-data fitted networks	TabPFN	0.769 ± 0.023	0.037 ± 0.005	11.482 ± 0.436	1268.114 ± 41.052	8942.376 ± 613.280
	TabPFN-R	0.753 ± 0.029	0.031 ± 0.006	6.704 ± 0.268	742.356 ± 27.418	–
	TabPFN-KM	0.796 ± 0.012	0.174 ± 0.012	6.791 ± 0.241	751.902 ± 26.115	–
	TabPFN-KC	0.811 ± 0.006	0.249 ± 0.011	6.812 ± 0.287	758.438 ± 24.907	5386.174 ± 402.516
	TabPFN-CE, $R = 1$	0.811 ± 0.009	0.256 ± 0.006	6.885 ± 0.251	–	–
	TabPFN-CE, $R = 3$	0.836 ± 0.031	0.780 ± 0.012	20.943 ± 1.136	–	–
	TabPFN-CE, $R = 5$ (proposed)	0.841 ± 0.033	1.300 ± 0.034	33.695 ± 0.851	3716.205 ± 112.378	26384.915 ± 1688.206

5.7. Explainability Analysis via SHAP

SHAP values were computed for TabPFN-CE trained on data augmented by TabDDPM-CC and enriched with the six engineered features. Figure 3 reports the global feature importance ranking, aggregated across all four classes. Figure 4 presents the corresponding beeswarm plots for each class, showing both the magnitude and the direction of each feature's contribution to the predicted class probability.

Figure 3 shows that Time Since Last Maintenance and Flight Cycles are the two most influential features overall, each contributing noticeably more than any other variable to the model output across all four classes. Both are original operational variables rather than engineered ones, which indicates that the raw maintenance history and usage counters carry the strongest overall signal for distinguishing between the four maintenance categories. Engine Temp follows as the third most important feature, again drawn from the original variable set.

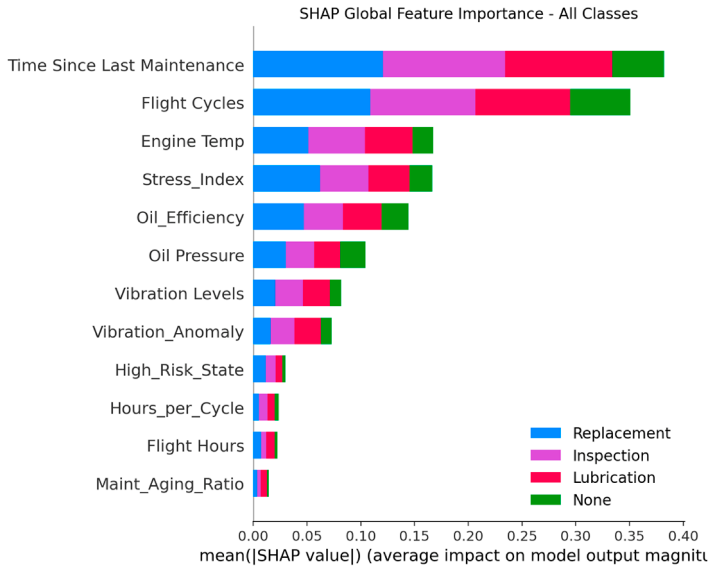


Figure 3. Global SHAP feature importance across all maintenance-action classes (Replacement, Inspection, Lubrication, None), ranked by mean absolute SHAP value.

The next two positions in the ranking are occupied by Stress_Index and Oil_Efficiency, both engineered features derived in Section 4.4 from combinations of the original sensor readings. Their position ahead of two original variables, Oil Pressure and Vibration Levels, indicates that the model draws on these engineered features more than on two of the raw sensor readings. The other four engineered features occupy lower positions in the ranking. The leave-one-feature-out results of Table 9 agree only in part. Removing Oil_Efficiency causes the largest loss, which matches its high SHAP rank. Removing Stress_Index causes only a small and non-significant loss, while Maint_Aging_Ratio ranks low in SHAP but its removal is significant. A high mean absolute SHAP value therefore shows that the model uses a feature, but it does not show that the feature is irreplaceable [66].

The class-specific beeswarm plots in Figure 4 refine this picture by showing that the relative importance of each feature changes depending on the maintenance category being predicted. For the None class, panel (a), Flight Cycles and Time Since Last Maintenance remain the dominant features, but Oil_Efficiency and Oil Pressure rank noticeably higher than in the aggregate ranking of Figure 3, suggesting that the model relies more heavily on oil-related measurements specifically when distinguishing instances that require no maintenance action from those that do. For the Inspection class, panel (b), Engine Temp and Stress_Index carry comparatively more weight than for the other three classes, which

is consistent with inspection actions typically being triggered by early signs of thermal or mechanical stress rather than by an already degraded oil condition. The Lubrication and Replacement classes, panels (c) and (d), both remain dominated by Time Since Last Maintenance and Flight Cycles, reflecting that these two categories are primarily governed by cumulative usage and elapsed maintenance interval rather than by any single instantaneous sensor reading.

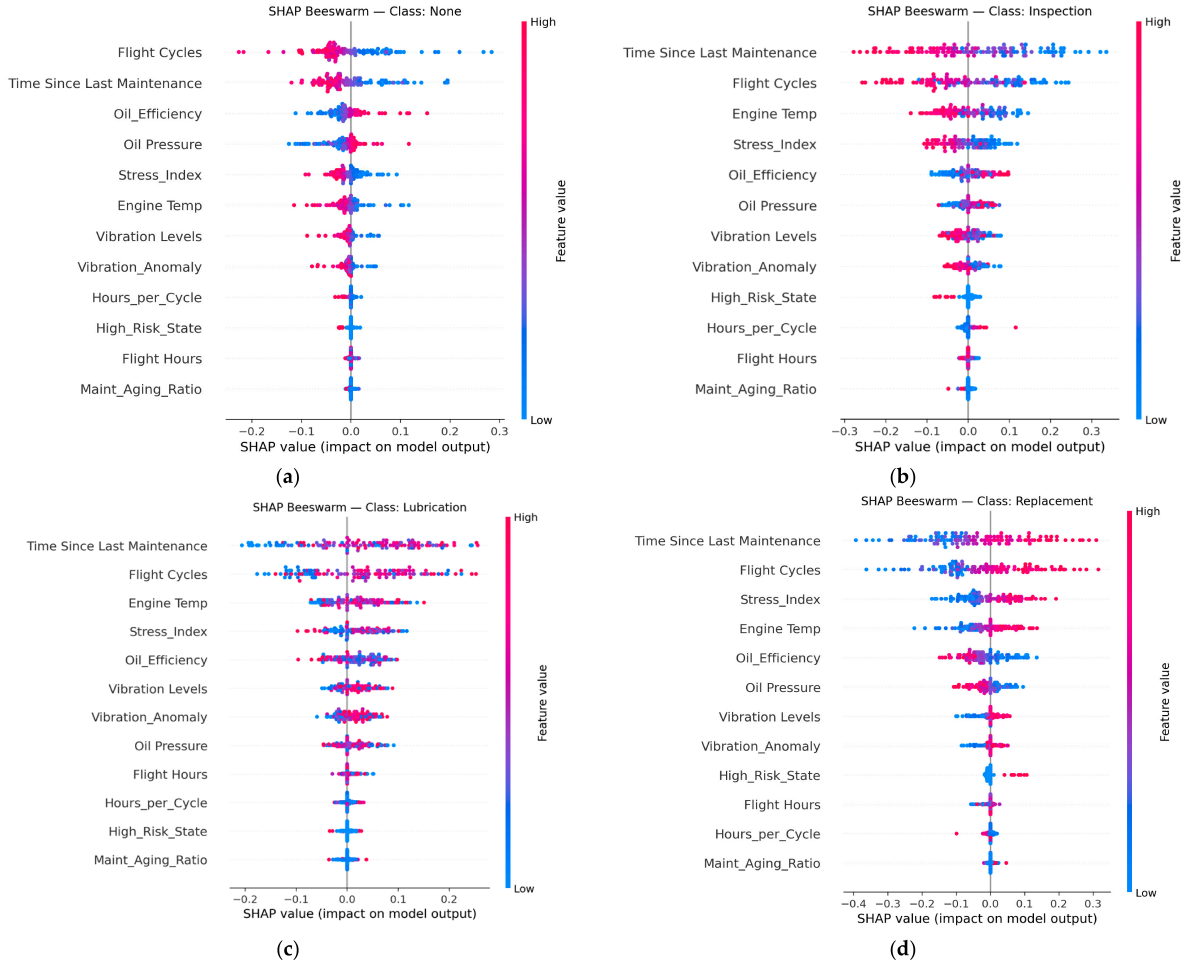


Figure 4. SHAP beeswarm plots for TabPFN-CE trained on TabDDPM-CC-augmented, feature-engineered data, showing the distribution of SHAP values for each of the twelve input features across the four maintenance classes: (**a**) None, (**b**) Inspection, (**c**) Lubrication, (**d**) Replacement. Color encodes the original feature value, with red indicating high and blue indicating low values.

Across all four panels, Time Since Last Maintenance and Flight Cycles consistently separate high and low feature values into opposite regions of the SHAP value axis, indicating a stable and monotonic relationship between elapsed usage and the likelihood of an action being required, rather than a relationship driven by a small number of extreme observations. This consistency across classes supports the stability of the explanation. It does not by itself show that each engineered feature carries distinct information, which is examined by the ablation of Table 9.

The SHAP results can be read at three levels. The first level is model interpretability. SHAP shows which inputs the fitted TabPFN-CE relies on and in which direction, and this level is fully supported by the analysis above. The second level is physical plausibility, meaning agreement with known degradation mechanisms of oil pumps. Several patterns meet this test. Time Since Last Maintenance and Flight Cycles dominate the Lubrication and Replacement classes, which matches the role of service interval and cycle count in the

wear of hydraulic piston pumps [38] and in condition-based maintenance practice [67]. The weight of Oil_Efficiency and Oil Pressure for the None class agrees with the expectation that a pump holding its pressure as temperature rises is in good condition [68]. The weight of Engine Temp and Stress_Index for Inspection agrees with the use of combined thermal and vibration thresholds as inspection triggers [67]. The third level is validation against domain knowledge, which would require expert review of individual explanations or records with confirmed physical failure modes. The maintenance extract contains neither, so this level could not be reached here. The explanations are therefore presented as plausible and consistent with the literature, not as causal evidence of degradation. Expert assessment of local explanations and failure-mode labels are needed before SHAP output can support airworthiness decisions.

5.8. Methodological Transferability Check on the AI4I 2020 Dataset

To assess whether the proposed methodology extends beyond the confidential aircraft dataset, this study applies the same augmentation and classification pipeline to the public AI4I 2020 predictive maintenance dataset [32]. The dataset contains 10,000 instances with a binary machine failure target. Only 339 of these instances are labeled as a failure, which gives an imbalance ratio of about 28.5 to 1. Three features were engineered from the dataset's official failure-mode formulas to approximate the physical drivers behind the power, heat dissipation and overstrain failure modes. This benchmark is a binary industrial failure problem rather than a four-class aircraft maintenance task. It is used to check whether the methodology transfers to another dataset, not as an external validation of aircraft maintenance performance.

The evaluation protocol was adapted to match the structure of this dataset rather than copied directly from the primary study. The target here is binary rather than the four-class maintenance action label used for the aircraft dataset. Ten random seeds were used for the train, validation and test splits. Each seed resamples the partitions with stratified sampling to keep the minority class represented in every split. Because the failure class is very small, it was enlarged to six times its real size in each training partition, from 237 to 1422 records. The augmentation methods and the evaluation protocol otherwise follow the aircraft design.

Table 17 reports the synthetic data quality metrics for the AI4I 2020 dataset using the ranking scheme of Table 2. TabDDPM-CC obtains the best average rank (1.00), followed by TabDiff (2.00) and TabDDPM (3.00). TabDDPM-CC is best on all four ranked metrics and also has the highest TSTR/TRTR ratio. TabSyn ranks fifth. TVAE and TabDDPM-CC again rank best within their own families. ADASYN outperforms SMOTE in the interpolation family, and BGM edges ahead of Gaussian Copula in the statistical family. The holdout-based measures in Appendix B, Table A5 show the same pattern as for the aircraft data. TabDDPM-CC stays close to the reference values for memorization (DCR share 0.534 and near-copy rate 0.057), and its coverage (0.868) is close to the held-out reference (0.881). SMOTE and ADASYN again produce clearly more near-copies (0.198 and 0.147).

Table 18 compares classification performance before and after TabDDPM-CC augmentation for twelve classifiers. The same balanced accuracy, macro F1, MCC and G-Mean metrics reported in Table 3 are used here. On the original imbalanced data, balanced accuracy stays close to 0.51 for every classifier, reflecting the severe 28.5 to 1 imbalance of this dataset. TabDDPM-CC augmentation raises the mean balanced accuracy across all twelve classifiers by 0.236. The best configuration is TabPFN-CE trained on TabDDPM-CC-augmented data, which reaches a balanced accuracy of 0.901, an F1 score of 0.912 and an MCC of 0.824. TabPFN-CE and its base model TabPFN rank first and second among all twelve classifiers under TabDDPM-CC augmentation, consistent with the ranking already

observed on the aircraft dataset. With TabPFN-CE, TabDiff reaches 0.884 and TabSyn 0.754, so the order of the three diffusion generators is the same as for the aircraft data.

Table 17. Synthetic data quality metrics on the AI4I 2020 dataset, methodological transferability check (mean ± std over ten seeds).

Evaluation Categories	Distinguishability (Classifier-Two-Sample Test-C2ST)		Privacy & Overfitting	Distribution Fidelity		ML Utility	Avg. Rank ***
Method\ Metric	Propensity AUC ↓ * (Mean ± Std)	Discriminator Acc ↓ * (Mean ± Std)	Mean DCR ** (Mean ± Std)	Wasserstein ↓ (Mean ± Std)	FD ↓ (Mean ± Std)	TSTR_TRTR_Ratio (F1_Macro) ↑ (Mean ± Std)	
SMOTE	0.856 ± 0.031	0.925 ± 0.019	5.058 ± 0.525	0.268 ± 0.052	21.791 ± 4.086	0.433 ± 0.074	10.50
ADASYN	0.809 ± 0.037	0.902 ± 0.015	4.695 ± 0.538	0.242 ± 0.054	19.977 ± 4.188	0.618 ± 0.085	**8.63**
BGM	0.809 ± 0.027	0.899 ± 0.024	3.271 ± 0.358	0.138 ± 0.036	12.855 ± 2.787	0.502 ± 0.070	**5.63**
Gaussian Copula	0.699 ± 0.030	0.843 ± 0.018	3.518 ± 0.461	0.156 ± 0.046	14.089 ± 3.584	0.621 ± 0.076	6.00
CopulaGAN	0.912 ± 0.035	0.952 ± 0.022	4.467 ± 0.538	0.225 ± 0.054	18.836 ± 4.189	0.430 ± 0.073	9.50
CTGAN	0.812 ± 0.032	0.901 ± 0.016	4.832 ± 0.426	0.252 ± 0.043	20.660 ± 3.311	0.417 ± 0.051	9.25
TVAE	0.685 ± 0.023	0.839 ± 0.025	3.479 ± 0.408	0.153 ± 0.041	13.897 ± 3.174	0.712 ± 0.082	**5.00**
TabDDPM	0.649 ± 0.028	0.819 ± 0.020	2.848 ± 0.525	0.108 ± 0.052	10.740 ± 4.082	0.801 ± 0.062	3.00
TabSyn	0.742 ± 0.038	0.863 ± 0.026	3.330 ± 0.480	0.143 ± 0.049	13.400 ± 3.500	0.668 ± 0.071	5.50
TabDiff	0.601 ± 0.034	0.798 ± 0.013	2.480 ± 0.430	0.081 ± 0.043	8.563 ± 3.037	0.836 ± 0.058	2.00
TabDDPM-CC	*0.554 ± 0.035*	*0.776 ± 0.009*	2.093 ± 0.391	*0.053 ± 0.039*	*6.450 ± 2.625*	*0.898 ± 0.052*	**1.00**

Note: Bold average ranks indicate the best-ranked method of each family, and italic values indicate the best result for each metric. Arrows give the preferred direction of each metric (↓ lower is better and ↑ higher is better). * Values closer to 0.5 indicate that real and synthetic samples are harder to distinguish. ** Mean DCR has no single optimal direction and is excluded from the ranking. *** Average rank over the four ranked metrics (1 = best and 11 = worst), with ties receiving the average rank.

Table 18. Classification results on the AI4I 2020 dataset, Original versus TabDDPM-CC augmentation, methodological transferability check.

Data Augmentation Method	Model/Metric	Balanced Acc	Macro F1	MCC	G-Mean
Original Dataset	RF	0.508 ± 0.014	0.509 ± 0.018	0.023 ± 0.039	0.157 ± 0.098
	XGBoost	0.506 ± 0.013	0.506 ± 0.017	0.016 ± 0.035	0.153 ± 0.079
	LightGBM	0.510 ± 0.011	0.512 ± 0.015	0.028 ± 0.031	0.183 ± 0.055
	CatBoost	0.509 ± 0.012	0.511 ± 0.015	0.025 ± 0.031	0.182 ± 0.057
	TabNet	0.514 ± 0.013	0.516 ± 0.017	0.036 ± 0.036	0.197 ± 0.082
	FT-Transformer	0.511 ± 0.013	0.513 ± 0.017	0.030 ± 0.036	0.177 ± 0.084
	NODE	0.514 ± 0.012	0.517 ± 0.015	0.038 ± 0.030	0.204 ± 0.059
	GANDALF	0.511 ± 0.011	0.513 ± 0.015	0.031 ± 0.033	0.177 ± 0.075
	SAINT	0.507 ± 0.012	0.508 ± 0.015	0.021 ± 0.031	0.163 ± 0.077
	ExcelFormer	0.515 ± 0.007	0.517 ± 0.010	0.040 ± 0.019	0.209 ± 0.037
	TabPFN	0.511 ± 0.011	0.513 ± 0.014	0.030 ± 0.030	0.195 ± 0.053
	TabPFN-CE	0.512 ± 0.008	0.514 ± 0.010	0.030 ± 0.022	0.204 ± 0.040
TabDDPM-CC	RF	0.663 ± 0.017	0.668 ± 0.016	0.338 ± 0.032	0.582 ± 0.029
	XGBoost	0.767 ± 0.030	0.775 ± 0.028	0.550 ± 0.055	0.735 ± 0.039
	LightGBM	0.774 ± 0.032	0.781 ± 0.024	0.564 ± 0.048	0.743 ± 0.042
	CatBoost	0.793 ± 0.031	0.804 ± 0.033	0.609 ± 0.066	0.767 ± 0.040
	TabNet	0.637 ± 0.031	0.647 ± 0.032	0.296 ± 0.064	0.534 ± 0.056
	FT-Transformer	0.701 ± 0.032	0.712 ± 0.025	0.426 ± 0.050	0.640 ± 0.048
	NODE	0.684 ± 0.026	0.693 ± 0.025	0.388 ± 0.051	0.614 ± 0.043
	GANDALF	0.701 ± 0.038	0.708 ± 0.036	0.417 ± 0.073	0.640 ± 0.061
	SAINT	0.740 ± 0.037	0.741 ± 0.032	0.482 ± 0.065	0.697 ± 0.053
	ExcelFormer	0.761 ± 0.020	0.766 ± 0.022	0.533 ± 0.045	0.727 ± 0.027
	TabPFN	0.833 ± 0.050	0.840 ± 0.034	0.682 ± 0.066	0.815 ± 0.064
	TabPFN-CE	0.901 ± 0.025	0.912 ± 0.015	0.824 ± 0.029	0.896 ± 0.028

These results should be read as a transferability check and not as a second full benchmark study. The AI4I 2020 experiments use a simpler binary target and a separate hyperparameter search, so absolute values are not comparable with Table 3. What transfers is the relative pattern. Class-conditioned diffusion produces the most realistic synthetic data, and the same context-optimized ensemble benefits most from it. This consistency shows that the methodology can be applied to another imbalanced tabular dataset. It does not validate the performance figures for aircraft maintenance. Full classification results for all augmentation methods and classifiers are provided in the Supplementary Materials.

5.9. Overall Discussion

The results converge on a consistent conclusion. Addressing class imbalance in aircraft oil pump maintenance data requires a coordinated combination of augmentation, feature representation and classifier design. No single intervention applied in isolation is sufficient. TabDDPM-CC obtained the best average quality rank among eleven generators, and class conditioning was its most influential component. TabPFN-CE achieved the strongest classification performance and the most consistent seed-level advantage. Feature engineering and augmentation interacted synergistically rather than additively. These findings would not be visible from a single comparison table, which is why the supporting analyses were treated as integral parts of the evaluation.

The operational reading of the per-class results is worth restating. The None class is the smallest category, but confusing it with an action class produces an unnecessary yet recoverable intervention. Missing a genuine Lubrication or Replacement need is the more consequential error. The rise in Lubrication recall from 0.597 to 0.825 and in Replacement recall from 0.394 to 0.938 between ratio 1 and ratio 4 therefore deserves as much attention as the gain for None.

The explainability analysis adds a further layer of support to these results. The two most influential features identified by SHAP, Time Since Last Maintenance and Flight Cycles, are exactly the variables that a maintenance engineer would expect to govern the decision between routine and corrective action. This supports the plausibility of the learned decision function, rather than treating the classifier as an unexamined black box. The leave-one-feature-out results confirm the value of Oil_Efficiency but show that SHAP rank and necessity do not always coincide. SHAP thus helps to check plausibility, but it does not establish causal links to degradation.

The AI4I 2020 check shows that the same combination transfers to a binary industrial failure dataset, where the best configuration reaches a balanced accuracy of 0.901. This lowers the risk that the methodological pattern is specific to the aircraft data. It does not establish that the aircraft performance figures hold for other fleets or components.

Several limitations bound the present results. First, the ten seeds resample stratified partitions of the same 647 records. The reported standard deviations and tests therefore describe variability across partitions of one dataset. They are not ten independent samples of aircraft maintenance data, and they do not demonstrate external reproducibility. Second, the extract has no time stamps or aircraft identifiers. Repeated observations of the same aircraft may fall into both training and test partitions, which can make a random split optimistic. The covariate-shift experiment shows how large such an effect can be when the operating regime changes.

Third, the None class deserves a separate comment. Each training partition holds only about 24 None records, so any generator has a narrow basis for learning this class. The synthetic None records are therefore close variations of the observed cases rather than new operating states. The DCR filter keeps them near the real data, which supports fidelity. Appendix B, Table A4 shows that this shift is small and leaves coverage almost unchanged,

but the filter cannot create operating states that are absent from the real records. The standard deviation of None recall, between 0.165 and 0.284 for the augmented ratios, shows that this class remains unstable even after augmentation. The None results should thus be read as indicative. A larger set of no-action records would be needed to estimate this class reliably.

The dataset itself imposes a further boundary on how the present findings should be interpreted. The 647 records analyzed in this study originate from a single aircraft type and a single component, the oil pump, within Turkish civil aviation operations. The proposed framework was designed to be data-agnostic in its mechanics, since neither TabDDPM-CC nor TabPFN-CE makes assumptions specific to this component or airframe. The reported performance figures, however, are naturally tied to the statistical properties of this particular dataset, including its imbalance ratio of 9.0 and its specific feature set. Extending the evaluation to other aircraft components, other maintenance decision types, or data collected under different operational and climatic conditions would therefore be necessary. Only then could the absolute performance figures reported here be generalized beyond the scope of this study.

Real maintenance data also raise practical issues that the present extract does not contain. Temporal dependence between successive records calls for chronological validation and possibly sequence models. Such models could not be applied here because the records carry no time order. Fleet differences and sensor drift change the input distribution over time, and the covariate-shift results indicate that periodic context refresh and generator retraining would be needed. Missing values do not occur in the extract. TabPFN v2 and later handle them natively, but the v1 backbone and the generators used here would need an imputation step. Changes in maintenance policy alter the meaning of the labels themselves and would require relabeling or retraining rather than augmentation. For these reasons, the framework is presented as a decision-support tool for maintenance planners and not as an autonomous safety-critical system.

Taken together, these results support the central claim of this study. A class-conditioned diffusion-based augmentation method combined with a coverage-aware, ensemble-based in-context classifier provides a measurable improvement over most conventional resampling and classification pipelines for this imbalanced aircraft maintenance dataset. At the same time, the analysis identifies the specific components, the augmentation ratio and the operational trade-offs that determine how far this improvement can be pushed in a practical deployment.

6. Conclusions

This study addressed the classification of imbalanced aircraft oil pump maintenance data through a hybrid framework. The framework combines a class-conditioned diffusion-based augmentation method, TabDDPM-CC, with a coverage-aware ensemble classifier, TabPFN-CE. Both components were developed through a cumulative design process in which each modification was isolated and quantified separately, rather than proposed and evaluated as a single undifferentiated change.

On a dataset of 647 aircraft maintenance records with an imbalance ratio of 9.0 between the majority and minority classes, TabDDPM-CC achieved the most favorable trade-off among eleven augmentation methods. This evaluation covered distinguishability, privacy, distribution fidelity and downstream utility metrics. TabDDPM-CC obtained the lowest average rank of 1.25, and holdout-based checks placed its memorization and diversity measures close to those of unseen real records. When paired with TabPFN-CE, the proposed pipeline raised balanced accuracy from 0.454 on the original imbalanced dataset to 0.841 and G-mean from 0.439 to 0.840. After Holm correction, its advantage was significant

against all eleven baseline classifiers and seven of the eleven alternative augmentation methods. It was not significant against SMOTE, TabDiff, the earlier TabDDPM variants or the single-context TabPFN-KC. The same framework also transferred to the public AI4I 2020 benchmark, where TabDDPM-CC and TabPFN-CE again achieved the best results among the tested combinations.

The cumulative and leave-one-out ablations showed that class conditioning in a single denoising network is the generator component with the largest effect. The stability modifications mainly protect fidelity once class conditioning is used. On the classifier side, k-center context selection combined with multi-seed posterior averaging gave the best results, and a smaller but positive gain remained with newer TabPFN backbones. The factorial ablation revealed a synergistic rather than additive relationship between feature engineering and augmentation. Leave-one-feature-out tests showed that Oil_Efficiency contributes most among the engineered features, while two of them are redundant. The augmentation ratio of 4.0, fixed in advance, coincided with the best of the six ratios tested. The noise and shift experiments showed a gradual loss under sensor noise, a stronger loss under label noise and a marked drop under covariate shift that context refresh partly recovers.

Beyond the reported performance figures, this study contributes a methodological template for imbalanced tabular classification in safety-critical predictive maintenance settings. In this template, augmentation and classification components are jointly optimized, statistically tested at the seed level, and interpreted through both a class-imbalance lens and an operational cost lens, rather than through aggregate accuracy alone. Future work should extend this framework to additional aircraft components and maintenance decision types, and should evaluate its behavior on multi-fleet or multi-airline data. Chronologically ordered records with aircraft identifiers and expert-reviewed explanations would allow temporal validation and a domain-level assessment of the SHAP results. It would also be worth exploring adaptive per-class augmentation ratios in place of the single global ratio adopted in this study, given the differential sensitivity to over-augmentation observed across the four maintenance categories. A quantitative cost-sensitivity analysis that assigns explicit misclassification costs to each maintenance action, planned for both the aircraft dataset and the AI4I 2020 benchmark, is left for future work. More broadly, this framework illustrates how class-conditioned generative modeling and context-aware classification can support mathematical modeling of fault detection and diagnosis across aerospace and other safety-critical industrial systems.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/math14214032/s1, Table S1: Classification results on the aircraft dataset; Table S2: Classification results on the AI4I 2020 dataset.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are confidential and not publicly available due to restrictions imposed by the industrial partner. Data may be available from the corresponding author upon reasonable request and with the partner's permission. The AI4I 2020 subset used for the methodological transferability check in Section 5.8 is itself a public dataset available from the original source [32].

Acknowledgments: Generative AI tools (Claude Sonnet 5.5 by Anthropic and Grammarly for Windows version 1.2.308.1991) were used solely for language editing purposes, i.e., improving grammar, clarity, and readability of the manuscript text. They were not used for generating research ideas, data analysis, data interpretation, or scientific content. The author reviewed all AI-assisted edits and takes full responsibility for the final content of the manuscript.

Conflicts of Interest: The author declares no conflicts of interest.

Appendix A

Table A1. Hyperparameter search spaces and fixed settings for the synthetic data augmentation methods.

Method	Parameter	Value Range
SMOTE, ADASYN	Ratio of synthetic samples for the minority class (sampling_strategy)	[“auto”, 0.5, 1.0]
	Number of neighbors (k_neighbors/n_neighbors)	[1, 2, 3, 4, 5, 6, 7]
BGM	Maximum number of mixture components (n_components)	[1, 2, 3, 5, 7]
	Dirichlet concentration prior (weight_concentration_prior)	[0.01, 0.1, 1.0, 10.0, 100.0]
	Type of covariance parameters (covariance_type)	[‘full’, ‘diag’]
Gaussian Copula	Default univariate distribution (default_distribution)	[‘norm’, ‘beta’, ‘truncnorm’, ‘uniform’, ‘gamma’, ‘gaussian_kde’]
	Rounding strategy for discrete variables (enforce_rounding)	[“auto”, True, False]
CTGAN	Generator dimension (generator_dim)	[(32, 64), (64, 64), (64, 128)]
	Discriminator dimension (discriminator_dim)	[(64, 32), (64, 64), (128, 64)]
	Input embedding size (embedding_dim)	[16, 32]
	Number of epochs (epochs)	[100, 200]
	Batch size (batch_size)	[4, 8, 16]
	Generator learning rate (generator_lr)	[0.0001, 0.0005, 0.001]
	Discriminator learning rate (discriminator_lr)	[0.0001, 0.0005, 0.001]
TVAE	Encoder hidden layer dimensions (compress_dims)	[(64, 32), (64, 64), (128, 64)]
	Decoder hidden layer dimensions (decompress_dims)	[(32, 64), (64, 64), (64, 128)]
	Number of training epochs (epochs)	[100, 200]
	Batch size (batch_size)	[4, 8, 16]
	Latent embedding dimension (embedding_dim)	[16, 32]
	Learning rate (learning_rate)	[0.0001, 0.0005, 0.001]
CopulaGAN	Generator dimension (generator_dim)	[(32, 64), (64, 64), (64, 128)]
	Discriminator dimension (discriminator_dim)	[(64, 32), (64, 64), (128, 64)]
	Input embedding size (embedding_dim)	[16, 32]
	Default distribution for copula transformer (default_distribution)	[‘norm’, ‘beta’]
	Number of epochs (epochs)	[100, 200]
	Batch size (batch_size)	[4, 8, 16]
	Learning rates (generator_lr, discriminator_lr)	[0.0001, 0.0005, 0.001]
TabDDPM, TabDDPM-S, TabDDPM-CC	Model architecture/hidden dimensions (d_layers)	[(128, 128), (128, 256), (256, 256), (64, 128, 128)]
	Dropout rate (dropout)	[0.0, 0.1, 0.2]
	Number of diffusion steps (num_timesteps/steps)	[100, 200, 500]
	Batch size (batch_size)	[4, 8, 16]
	Learning rate (lr)	[0.0001, 0.0003, 0.001]
	Conditioning embedding dimension (cond_emb_dim) for TabDDPM-CC	[8, 16, 32]
TabSyn (fixed defaults)	VAE tokenizer and Transformer (d_token, num_layers, n_head, factor)	4, 2, 1, 32
	VAE learning rate and epochs	0.001, 4000
	VAE β schedule (β_max, β_min, decay λ)	0.01, 0.00001, 0.7
	Latent diffusion denoiser width (dim_t)	1024
	Diffusion learning rate, epochs and early-stopping patience	0.001, 10,000, 500
	Batch size and number of sampling steps	4096, 50
	Class-wise generation	Shared VAE, one latent diffusion model per minority class
TabDiff (fixed defaults)	Tokenizer and Transformer (num_layers, d_token, n_head, factor)	2, 4, 1, 32
	Denoiser width (dim_t) and sampling timesteps	1024, 50
	EDM noise parameters (σ_data, σ_min, σ_max)	1.0, 0.002, 80
	Learnable schedule initialization (ρ_init, ρ_offset, k_init, k_offset)	7, 5, −6, 1
	Noise bounds (ε_max, ε_min)	0.001, 0.00001
	Epochs, learning rate, EMA decay and batch size	8000, 0.001, 0.997, 4096
	Class-wise generation	Single model, label fixed by the in-painting sampler (one resampling round, no guidance)

Note: Square brackets list the candidate values of the grid search.

Table A2. Hyperparameter search spaces and fixed settings for the classification models.

Method	Parameter	Value Range
RF	Number of trees (n_estimators)	[100, 300, 500]
	Maximum tree depth (max_depth)	[None, 5, 10]
	Minimum samples per leaf (min_samples_leaf)	[1, 2, 4]
XGBoost	Number of boosting rounds (n_estimators)	[100, 300, 500]
	Maximum tree depth (max_depth)	[3, 5, 7]
	Learning rate (learning_rate)	[0.01, 0.05, 0.1]
LightGBM	Number of boosting rounds (n_estimators)	[100, 300, 500]
	Maximum number of leaves (num_leaves)	[15, 31, 63]
	Learning rate (learning_rate)	[0.01, 0.05, 0.1]
CatBoost	Number of iterations (iterations)	[100, 300, 500]
	Tree depth (depth)	[4, 6, 8]
	Learning rate (learning_rate)	[0.01, 0.05, 0.1]
TabNet	Feature/attention width (n_d = n_a)	[8, 16, 32]
	Number of decision steps (n_steps)	[3, 5, 7]
	Relaxation parameter (gamma)	[1.0, 1.5, 2.0]
	Learning rate (learning_rate)	[0.01, 0.02, 0.05]
FT-Transformer	Input embedding dimension (input_embed_dim)	[16, 32]
	Number of attention blocks (num_attn_blocks)	[2, 3]
	Number of attention heads (num_heads)	[2, 4]
	Batch size (batch_size)	[32, 64]
	Learning rate (learning_rate)	[0.0003, 0.001, 0.003]
NODE	Number of trees (num_trees)	[128, 256]
	Tree depth (depth)	[4, 6]
	Number of layers (num_layers)	[1, 2]
	Batch size (batch_size)	[32, 64]
	Learning rate (learning_rate)	[0.0003, 0.001, 0.003]
GANDALF	Number of GFLU stages (gflu_stages)	[2, 4, 6]
	GFLU dropout (gflu_dropout)	[0.0, 0.1]
	Batch size (batch_size)	[32, 64]
	Learning rate (learning_rate)	[0.0003, 0.001, 0.003]
SAINT	Model dimension (d_model)	[32, 64]
	Number of attention layers (depth)	[2, 3]
	Dropout rate (dropout)	[0.0, 0.1]
	Batch size (batch_size)	[32, 64]
	Learning rate (lr)	[0.0003, 0.001, 0.003]
ExcelFormer	Model dimension (d_model)	[32, 64]
	Number of attention layers (depth)	[2, 3]
	Dropout rate (dropout)	[0.1, 0.15, 0.2]
	Batch size (batch_size)	[32, 64]
	Learning rate (learning_rate)	[0.0003, 0.001, 0.003]
TabPFN v1: TabPFN, TabPFN-R, TabPFN-KM, TabPFN-KC, TabPFN-CE	Number of ensemble configurations (N_ensemble_configurations)	[16, 32, 64]
	Context size per class (samples_per_class), context variants only	[80, 126, 160]
	Number of averaged k-center contexts (R), TabPFN-CE only	5 (fixed)
TabPFN-KC, TabPFN-CE	Context of the default TabPFN	Full training set, stratified subsample above 1000 records
TabPFN v2, v2.5, v3 (backbone sensitivity)	Model configuration	Version defaults (create_default_for_version), not tuned
	Pretraining size limits (ignore_pretraining_limits)	True
	Context	Full training set (single context) or the TabPFN-CE contexts with the v1 settings

Note: Square brackets list the candidate values of the grid search.

Appendix B

Table A3. Holdout-based memorization and diversity measures of the eleven augmentation methods on the aircraft dataset (mean ± std over ten seeds).

	Memorization			Diversity	
Method	**DCR Share**	**Near-Copy Rate**	**DCR Ratio**	**Coverage**	**Unique NN Ratio**
Held-out real records (reference)	0.50 †	0.05 †	1.00 †	0.906 ± 0.018	0.690 ± 0.020
SMOTE	0.853 ± 0.045	0.257 ± 0.030	0.491 ± 0.112	0.918 ± 0.021	0.688 ± 0.034
ADASYN	0.855 ± 0.044	0.253 ± 0.029	0.501 ± 0.103	0.894 ± 0.026	0.620 ± 0.041
BGM	0.504 ± 0.037	0.030 ± 0.008	1.453 ± 0.149	0.781 ± 0.034	0.589 ± 0.037
Gaussian Copula	0.512 ± 0.038	0.030 ± 0.008	1.319 ± 0.160	0.801 ± 0.033	0.636 ± 0.033
CopulaGAN	0.499 ± 0.037	0.007 ± 0.005	2.853 ± 0.147	0.396 ± 0.081	0.287 ± 0.062
CTGAN	0.504 ± 0.040	0.006 ± 0.005	2.687 ± 0.181	0.481 ± 0.072	0.349 ± 0.057
TVAE	0.517 ± 0.039	0.045 ± 0.009	1.115 ± 0.165	0.624 ± 0.058	0.452 ± 0.049
TabDDPM	0.512 ± 0.032	0.042 ± 0.009	1.149 ± 0.098	0.846 ± 0.031	0.641 ± 0.028
TabSyn	0.522 ± 0.036	0.048 ± 0.010	1.098 ± 0.140	0.861 ± 0.029	0.667 ± 0.026
TabDiff	0.512 ± 0.039	0.044 ± 0.009	1.050 ± 0.168	0.884 ± 0.024	0.703 ± 0.024
TabDDPM-CC	0.544 ± 0.040	0.058 ± 0.012	0.977 ± 0.182	0.903 ± 0.019	0.679 ± 0.022

Note: Ideal values are 0.50 for the DCR share, 0.05 for the near-copy rate and 1.00 for the DCR ratio. Coverage and the unique NN ratio should be close to the reference row. Standard deviations describe variation across partitions of the same dataset. † Value of the held-out reference by definition.

Table A4. Effect of the DCR filter, modification (f) of TabDDPM-CC, on the aircraft dataset (mean ± std over ten seeds).

Group	**Measure**	**Held-Out Reference**	**With Filter**	**Without Filter**	**Δ**
Memorization	DCR share	0.50 †	0.544 ± 0.040	0.511 ± 0.038	+0.033
	Near-copy rate	0.05 †	0.058 ± 0.012	0.044 ± 0.011	+0.014
	DCR ratio	1.00 †	0.977 ± 0.182	1.063 ± 0.186	−0.086
Diversity	Coverage	0.906 ± 0.018	0.903 ± 0.019	0.897 ± 0.022	+0.006
	Unique NN ratio	0.690 ± 0.020	0.679 ± 0.022	0.683 ± 0.024	−0.004
Fidelity	Propensity AUC ↓	–	0.565 ± 0.022	0.574 ± 0.018	−0.009
	Discriminator Acc ↓	–	0.630 ± 0.006	0.636 ± 0.008	−0.006
	Wasserstein ↓	–	0.019 ± 0.012	0.023 ± 0.013	−0.004
	FD ↓	–	5.559 ± 3.042	5.874 ± 3.150	−0.315
Utility	TSTR/TRTR ratio ↑	–	0.736 ± 0.080	0.714 ± 0.082	+0.022

Note: Fidelity and utility are computed as in Table 2 and are not defined for the held-out reference (–). Δ is the difference of the means, with filter minus without filter. Classification results for both configurations are given in Table 5. † Value of the held-out reference by definition. ↓ indicates that lower values are better and ↑ that higher values are better.

Table A5. Holdout-based memorization and diversity measures of the eleven augmentation methods on the AI4I 2020 dataset (mean ± std over ten seeds).

	Memorization			Diversity	
Method	**DCR Share**	**Near-Copy Rate**	**DCR Ratio**	**Coverage**	**Unique NN Ratio**
Held-out real records (reference)	0.50 †	0.05 †	1.00 †	0.881 ± 0.026	0.694 ± 0.027
SMOTE	0.774 ± 0.046	0.198 ± 0.026	0.614 ± 0.099	0.862 ± 0.031	0.593 ± 0.036
ADASYN	0.703 ± 0.048	0.147 ± 0.024	0.717 ± 0.121	0.806 ± 0.038	0.552 ± 0.041
BGM	0.512 ± 0.044	0.024 ± 0.011	1.519 ± 0.169	0.752 ± 0.036	0.604 ± 0.041
Gaussian Copula	0.508 ± 0.048	0.022 ± 0.008	1.626 ± 0.217	0.698 ± 0.041	0.592 ± 0.040
CopulaGAN	0.489 ± 0.053	0.010 ± 0.009	2.075 ± 0.252	0.471 ± 0.058	0.338 ± 0.060
CTGAN	0.484 ± 0.047	0.009 ± 0.007	2.257 ± 0.214	0.503 ± 0.052	0.362 ± 0.058
TVAE	0.511 ± 0.046	0.022 ± 0.008	1.609 ± 0.192	0.758 ± 0.044	0.587 ± 0.044
TabDDPM	0.524 ± 0.039	0.031 ± 0.009	1.329 ± 0.246	0.826 ± 0.031	0.651 ± 0.030
TabSyn	0.506 ± 0.049	0.024 ± 0.008	1.547 ± 0.225	0.741 ± 0.038	0.623 ± 0.036
TabDiff	0.523 ± 0.047	0.044 ± 0.010	1.155 ± 0.202	0.874 ± 0.026	0.706 ± 0.026
TabDDPM-CC	**0.534 ± 0.046**	**0.057 ± 0.014**	**0.987 ± 0.184**	**0.868 ± 0.024**	**0.688 ± 0.025**

Note: Ideal values and symbols as in Table A3. The proposed TabDDPM-CC is shown in bold. † Value of the held-out reference by definition.

References

1. Abdul Shukor, S.; Tang, J.H.; Kamal, I.; Mohammad, M.F.; Othman, Z.; Sahran, S.; Sani, N.S.; Yaakub, M.R.; Abdullah, A. Predictive Maintenance in Aerospace: Leveraging Machine Learning for Optimized Filter Lifespan. In *Systems Modelling and Simulation. SMS 2024*; Mohamed, Z., Hassan, F., Tan, G., Ahmad, A., Ling, P., Buyamin, L., Eds.; Communications in Computer and Information Science; Springer: Singapore, 2025; Volume 2483, pp. 87–101.
2. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority Over-Sampling Technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
3. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks, Hong Kong, China, 1–8 June 2008; pp. 1322–1328.
4. Blei, D.M.; Jordan, M.I. Variational Inference for Dirichlet Process Mixtures. *Bayesian Anal.* **2006**, *1*, 121–143. [CrossRef]
5. Patki, N.; Wedge, R.; Veeramachaneni, K. The Synthetic Data Vault. In Proceedings of the 2016 IEEE International Conference on Data Science and Advanced Analytics, Montreal, QC, Canada, 17–19 October 2016; pp. 399–410.
6. Xu, L.; Skoularidou, M.; Cuesta-Infante, A.; Veeramachaneni, K. Modeling Tabular Data Using Conditional GAN. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 7335–7345.
7. Kotelnikov, A.; Baranchuk, D.; Rubachev, I.; Babenko, A. TabDDPM: Modelling Tabular Data with Diffusion Models. In Proceedings of the 40th International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July 2023; pp. 17564–17579.
8. Ho, J.; Jain, A.; Abbeel, P. Denoising Diffusion Probabilistic Models. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 6840–6851.
9. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784.
10. Nichol, A.Q.; Dhariwal, P. Improved Denoising Diffusion Probabilistic Models. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 8162–8171.
11. Zhang, H.; Zhang, J.; Srinivasan, B.; Shen, Z.; Qin, X.; Faloutsos, C.; Rangwala, H.; Karypis, G. Mixed-Type Tabular Data Synthesis with Score-Based Diffusion in Latent Space. In Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024.
12. Shi, J.; Xu, M.; Hua, H.; Zhang, H.; Ermon, S.; Leskovec, J. TabDiff: A Mixed-Type Diffusion Model for Tabular Data Generation. In Proceedings of the International Conference on Learning Representations, Singapore, 24–28 April 2025; pp. 37353–37375.
13. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
14. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
15. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.-Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3146–3154.
16. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A.V.; Gulin, A. CatBoost: Unbiased Boosting with Categorical Features. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 6638–6648.
17. Arik, S.Ö.; Pfister, T. TabNet: Attentive Interpretable Tabular Learning. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 6679–6687. [CrossRef]
18. Gorishniy, Y.; Rubachev, I.; Khrulkov, V.; Babenko, A. Revisiting Deep Learning Models for Tabular Data. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 18932–18943.
19. Popov, S.; Morozov, S.; Babenko, A. Neural Oblivious Decision Ensembles for Deep Learning on Tabular Data. In Proceedings of the 8th International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020.
20. Somepalli, G.; Goldblum, M.; Schwarzschild, A.; Bruss, C.B.; Goldstein, T. SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training. *arXiv* **2021**, arXiv:2106.01342.
21. Joseph, M.; Raj, H. GANDALF: Gated Adaptive Network for Deep Automated Learning of Features for Tabular Data. In Proceedings of the 13th ACM IKDD International Conference on Data Science, Pune, India, 17–20 December 2025; pp. 35–51.
22. Chen, J.; Yan, J.; Chen, D.Z.; Wu, J. ExcelFormer: A Neural Network Surpassing GBDTs on Tabular Data. *arXiv* **2023**, arXiv:2301.02819.
23. Hollmann, N.; Müller, S.; Eggensperger, K.; Hutter, F. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second. *arXiv* **2022**, arXiv:2207.01848.
24. MacQueen, J. Some Methods for Classification and Analysis of Multivariate Observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, CA, USA, 21 June–18 July 1965; Volume 1, pp. 281–297.
25. Gonzalez, T.F. Clustering to Minimize the Maximum Intercluster Distance. *Theor. Comput. Sci.* **1985**, *38*, 293–306. [CrossRef]
26. Sener, O.; Savarese, S. Active Learning for Convolutional Neural Networks: A Core-Set Approach. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
27. Breiman, L. Bagging Predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
28. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
29. Wilcoxon, F. Individual Comparisons by Ranking Methods. *Biom. Bull.* **1945**, *1*, 80–83. [CrossRef]
30. Holm, S. A Simple Sequentially Rejective Multiple Test Procedure. *Scand. J. Stat.* **1979**, *6*, 65–70.

31. Lundberg, S.M.; Lee, S.-I. A Unified Approach to Interpreting Model Predictions. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 4765–4774.
32. Matzka, S. Explainable Artificial Intelligence for Predictive Maintenance Applications. In Proceedings of the 2020 Third International Conference on Artificial Intelligence for Industries (AI4I), Irvine, CA, USA, 21–23 September 2020; pp. 69–74.
33. Zhou, T.T.; Han, T.; Droguett, E.L. Towards Trustworthy Machine Fault Diagnosis: A Probabilistic Bayesian Deep Learning Framework. *Reliab. Eng. Syst. Saf.* **2022**, *224*, 108525. [CrossRef]
34. Liao, Z.; Zhan, K.; Zhao, H.; Deng, Y.; Geng, J.; Chen, X.; Song, Z. Addressing Class-Imbalanced Learning in Real-Time Aero-Engine Gas-Path Fault Diagnosis via Feature Filtering and Mapping. *Reliab. Eng. Syst. Saf.* **2024**, *249*, 110189. [CrossRef]
35. Yan, S.; Zhong, X.; Shao, H.; Ming, Y.; Liu, C.; Liu, B. Digital Twin-Assisted Imbalanced Fault Diagnosis Framework Using Subdomain Adaptive Mechanism and Margin-Aware Regularization. *Reliab. Eng. Syst. Saf.* **2023**, *239*, 109522. [CrossRef]
36. Xia, M.; Shao, H.; Williams, D.; Lu, S.; Shu, L.; de Silva, C.W. Intelligent Fault Diagnosis of Machinery Using Digital Twin-Assisted Deep Transfer Learning. *Reliab. Eng. Syst. Saf.* **2021**, *215*, 107938. [CrossRef]
37. Cheng, Y.; Chen, B.; Zhou, N.; Gu, F.; Allen, P.; Ball, A.; Zhang, W. Multiple Signal Blind Deconvolution for Fault Feature Enhancement of Rolling Element Bearings. *Struct. Health Monit.* **2026**, 14759217261464907. [CrossRef]
38. Guo, S.; Chen, J.; Lu, Y.; Wang, Y.; Dong, H. Hydraulic Piston Pump in Civil Aircraft: Current Status, Future Directions and Critical Technologies. *Chin. J. Aeronaut.* **2020**, *33*, 16–30. [CrossRef]
39. Liu, T.; Zhang, Y.; Yin, X.; He, W. Aviation Fuel Pump Fault Diagnosis Based on Conditional Variational Self-Encoder Adaptive Synthetic Less Data Enhancement. *Mathematics* **2025**, *13*, 2218. [CrossRef]
40. Chen, Z.; Chen, J.; Feng, Y.; Liu, S.; Zhang, T.; Zhang, K.; Xiao, W. Imbalance Fault Diagnosis Under Long-Tailed Distribution: Challenges, Solutions and Prospects. *Knowl.-Based Syst.* **2022**, *258*, 110008. [CrossRef]
41. Gao, Y.; Dong, J. Robust Imbalanced Learning for Aero-Engine Bearing Anomaly Detection via a Hybrid SMOTE-BLS Framework. *Aerosp. Eng. Commun.* **2026**, *1*, 47–56. [CrossRef]
42. Li, Z.; Zheng, T.; Wang, Y.; Cao, Z.; Guo, Z.; Fu, H. A Novel Method for Imbalanced Fault Diagnosis of Rotating Machinery Based on Generative Adversarial Networks. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–17. [CrossRef]
43. Chen, H.; Wei, J.; Huang, H.; Yuan, Y.; Wang, J. Review of Imbalanced Fault Diagnosis Technology Based on Generative Adversarial Networks. *J. Comput. Des. Eng.* **2024**, *11*, 99–124. [CrossRef]
44. Borisov, V.; Leemann, T.; Seßler, K.; Haug, J.; Pawelczyk, M.; Kasneci, G. Deep Neural Networks and Tabular Data: A Survey. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *35*, 7499–7519. [CrossRef] [PubMed]
45. Li, G.; Li, W.; Wen, T.; Sun, W.; Tang, X. A Fault Diagnosis Method for Avionics Equipment Based on SMOTEWB-LGBM. *Int. J. Aerosp. Eng.* **2024**, *2024*, 3493676. [CrossRef]
46. Gündüz, A.F.; Şahin, C.B. Synthetic Data Augmentation for Imbalanced Tabular Protein Subcellular Localization: A Comparative Study of SMOTE, CTGAN, TVAE, and TabDDPM Methods. *Appl. Sci.* **2026**, *16*, 3694. [CrossRef]
47. Lim, B.; Arık, S.Ö.; Loeff, N.; Pfister, T. Temporal Fusion Transformers for Interpretable Multi-Horizon Time Series Forecasting. *Int. J. Forecast.* **2021**, *37*, 1748–1764. [CrossRef]
48. Hollmann, N.; Müller, S.; Purucker, L.; Krishnakumar, A.; Körfer, M.; Hoo, S.B.; Schirrmeister, R.T.; Hutter, F. Accurate Predictions on Small Data with a Tabular Foundation Model. *Nature* **2025**, *637*, 319–326. [CrossRef] [PubMed]
49. Grinsztajn, L.; Flöge, K.; Key, O.; Birkel, F.; Jund, P.; Roof, B.; Jäger, B.; Safaric, D.; Alessi, S.; Hayler, A.; et al. TabPFN-2.5: Advancing the State of the Art in Tabular Foundation Models. *arXiv* **2025**, arXiv:2511.08667.
50. Grinsztajn, L.; Flöge, K.; Key, O.; Birkel, F.; Jund, P.; Roof, B.; Manium, M.; Hoo, S.B.; Bühler, M.; Garg, A.; et al. TabPFN-3: Technical Report. *arXiv* **2026**, arXiv:2605.13986.
51. Feuer, B.; Schirrmeister, R.T.; Cherepanova, V.; Hegde, C.; Hutter, F.; Goldblum, M.; Cohen, N.; White, C. TuneTables: Context Optimization for Scalable Prior-Data Fitted Networks. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 83430–83464. [CrossRef]
52. Abdullah, M. Understanding Context Sampling in TabPFN on Small Tabular Datasets. *arXiv* **2026**, arXiv:2607.26628.
53. Snoke, J.; Raab, G.M.; Nowok, B.; Dibben, C.; Slavkovic, A. General and Specific Utility Measures for Synthetic Data. *J. R. Stat. Soc. Ser. A Stat. Soc.* **2018**, *181*, 663–688. [CrossRef]
54. Zhao, Z.; Kunar, A.; Birke, R.; Chen, L.Y. CTAB-GAN: Effective Table Data Synthesizing. In Proceedings of the 13th Asian Conference on Machine Learning, Virtual, 17–19 November 2021; pp. 97–112.
55. Linardatos, P.; Papastefanopoulos, V.; Kotsiantis, S. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy* **2021**, *23*, 18. [CrossRef] [PubMed]
56. Özcan, H. Interpretable Ensemble Remaining Useful Life Prediction Enables Dynamic Maintenance Scheduling for Aircraft Engines. *Sci. Rep.* **2025**, *15*, 23473. [CrossRef] [PubMed]
57. Maulana, F.; Starr, A.; Ompusunggu, A.P. Explainable Data-Driven Method Combined with Bayesian Filtering for Remaining Useful Lifetime Prediction of Aircraft Engines Using NASA CMAPSS Datasets. *Machines* **2023**, *11*, 163. [CrossRef]
58. Yang, Y.; Iqbal, M.Z. Cost-Optimised Machine Learning Model Comparison for Predictive Maintenance. *Electronics* **2025**, *14*, 2497. [CrossRef]

59. Rohan, A. Holistic Fault Detection and Diagnosis System in Imbalanced, Scarce, Multi-Domain (ISMD) Data Setting for Component-Level Prognostics and Health Management (PHM). *Mathematics* **2022**, *10*, 2031. [CrossRef]
60. Ghasemkhani, B.; Kut, R.A.; Birant, D.; Yilmaz, R. Balanced Hoeffding Tree Forest (BHTF): A Novel Multi-Label Classification with Oversampling and Undersampling Techniques for Failure Mode Diagnosis in Predictive Maintenance. *Mathematics* **2025**, *13*, 3019. [CrossRef]
61. Almasad, A.; Rehman, T.; Mustafa, M.; Azim, S.; Alkahtani, M.S. XAI-PdMNet-Bench: An Explainable Generative AI Framework for Leakage-Safe Predictive Maintenance in Industry 5.0 Manufacturing. *Int. J. Adv. Manuf. Technol.* **2026**, *145*, 6837–6862. [CrossRef]
62. Khemani, K.; Qureshi, A.N. AI-Driven Predictive Maintenance for Connected Vehicles Using Environmental Context Integration Evaluated through Simulation Benchmarking and Field Validation. *Discov. Veh.* **2026**, *2*, 19. [CrossRef]
63. Attias, H. A Variational Bayesian Framework for Graphical Models. *Adv. Neural Inf. Process. Syst.* **1999**, *12*, 209–215.
64. Platzer, M.; Reutterer, T. Holdout-Based Empirical Assessment of Mixed-Type Synthetic Data. *Front. Big Data* **2021**, *4*, 679939. [CrossRef] [PubMed]
65. Naeem, M.F.; Oh, S.J.; Uh, Y.; Choi, Y.; Yoo, J. Reliable Fidelity and Diversity Metrics for Generative Models. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 7176–7185.
66. Kumar, I.E.; Venkatasubramanian, S.; Scheidegger, C.; Friedler, S. Problems with Shapley-Value-Based Explanations as Feature Importance Measures. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 5491–5500.
67. Jardine, A.K.S.; Lin, D.; Banjevic, D. A Review on Machinery Diagnostics and Prognostics Implementing Condition-Based Maintenance. *Mech. Syst. Signal Process.* **2006**, *20*, 1483–1510. [CrossRef]
68. Stachowiak, G.W.; Batchelor, A.W. *Engineering Tribology*, 4th ed.; Butterworth-Heinemann: Oxford, UK, 2014.
69. de Borda, J.-C. Mémoire sur les Élections au Scrutin. *Hist. Acad. R. Sci.* **1781**, *102*, 657–665.
70. Northcutt, C.; Jiang, L.; Chuang, I. Confident Learning: Estimating Uncertainty in Dataset Labels. *J. Artif. Intell. Res.* **2021**, *70*, 1373–1411. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.