M-DEW: Extending Dynamic Ensemble Weighting to Handle Missing Values

Adam Catto*

Icahn School of Medicine at Mount Sinai, New York, NY, USA

Nan Jia*

CUNY Graduate Center, New York, NY, USA

Ansaf Salleb-Aouissi Columbia University, New York, NY, USA

Anita Raja

CUNY Hunter College, New York, NY, USA

Abstract

Missing value imputation is a crucial preprocessing step for many machine learning problems. However, it is often considered as a separate subtask from downstream applications such as classification, regression, or clustering, and thus is not optimized together with them. We hypothesize that treating the imputation model and downstream task model together and optimizing over full pipelines will yield better results than treating them separately. Our work describes a novel AutoML technique for making downstream predictions with missing data that automatically handles preprocessing, model weighting, and selection during inference time, with minimal compute overhead. Specifically we develop M-DEW, a Dynamic missingness-aware Ensemble Weighting (DEW) approach, that constructs a set of two-stage imputation-prediction pipelines, trains each component separately, and dynamically calculates a set of pipeline weights for each sample during inference time. We thus extend previous work on dynamic ensemble weighting to handle missing data at the level of full imputation-prediction pipelines, improving performance and calibration on downstream machine learning tasks over standard model averaging techniques. M-DEW is shown to outperform the state-of-the-art in that it produces statistically significant reductions in model perplexity in 17 out of 18 experiments, while improving average precision in 13 out of 18 experiments.

Data and Code Availability This paper uses six public health datasets: the EEG Eye State (Roesler, 2013), the Diabetic Retinopathy (Antal and Hajdu, 2014), the Wisconsin Breast Cancer Diagnosis and Prognosis (Street et al., 1995), Diabetes 130-US hospitals for years 1999- 2008 (Clore et al., 2014), and the Myocardial Infarction Complications (Golovenkin et al., 2020) datasets from the UCI Machine Learning Repository. The code for the approach paper is available at https://github.com/adamcatto/dynamime

Institutional Review Board (IRB) This research does not involve human subjects, and thus does not require IRB approval.

1. Introduction

Missing value imputation is a crucial preprocessing step for many machine learning problems, notably in supervised machine learning. Consider biomedical and clinical tasks, such as diagnosis, disease progression modeling, and risk stratification. The datasets on which these machine learning models are trained tend to be sparse: many procedures, tests, and measurements are conducted on a small number of samples across the entire dataset. Therefore, successful prediction modeling in these regimes requires methods for handling missing values that are optimal for downstream prediction tasks. While our approach is motivated by clinical informatics problems and we evaluate our approach on healthcare datasets, our approach addresses the missing data problem that is ubiquitous across a plethora of application domains.

ADAM.CATTO@MSSM.EDU

NJIA@GRADCENTER.CUNY.EDU

ANSAFSALLEB@COLUMBIA.EDU

ANITA.RAJA@HUNTER.CUNY.EDU

^{*} These authors contributed equally

We describe the road map leading to our approach below.

It is common knowledge that missing values exist in the health, signal processing, and image recognition fields. Hardware malfunction, machines shutdown and human mistakes are inevitable despite investing much effort to avoid them (Emmanuel et al., 2021). One strategy for handling those data is to simply delete portions of the data that are missing. However, deleting such data can bias the data in non-MCAR regimes (see section 2.1). Furthermore, in real life, a global complete-case analysis is impossible, people may opt instead to delete features with large amounts of missing values. The issue with this approach is two-fold: (1) it is plausible that most features in the dataset are sparse, so only the most commonplace measurements will be preserved, and (2) perhaps those sparse features are particularly important for certain sub-groups of subjects for whom the data are available, leading to worse predictions.

Another approach to missing data is imputation, either by a central tendency statistic on the feature (e.g., mean, median), or as a function of the features which are present in that sample (examples include k-nearest neighbors imputation (Zhang, 2012), multiple imputation via chained equations (White et al., 2011), and random forest-based prediction of feature value (Stekhoven and Bühlmann, 2012)). However, imputation of features which are not stronglycorrelated with other features has serious potential to misrepresent the samples and bias the feature's distribution (White and Carlin, 2010). The same issues are present with imputation based on central tendency (e.g., imputation to the population mean lowers standard deviation).

It is not known a priori what methods are best suited for imputing missing values for a given dataset, especially since certain features may be better predicted by one class of models and covariates vs. others. For instance, one feature may be strongly predicted as an additive model of certain other features, whereas another may be strongly predicted by a nearest-neighbor or tree-like algorithm.

We hypothesize that applying a dynamic ensemble weighting (DEW) approach to the full imputationclassification pipeline will lead to reduced classification errors over the standard soft voting / uniform model averaging (UMA) approach. Specifically, we develop and evaluate M-DEW, a novel approach that learns a dynamic ensemble weighting function for a pool of imputation-classifier pipelines on a persample basis, taking into account missing values. M-DEW is designed to address the knowledge gap in the current state-of-the-art by mitigating the effects of sample misrepresentation by estimating the suitability of each missing-data-handling technique for a given sample. Our approach is designed to offload the problem of selecting imputation and downstream prediction modules by simply supplying a set of imputation-prediction pipelines, and learning an optimal weighting of the pipelines' predictions dynamically for each sample.

Our novel contributions are two-fold: (i) an ensembling technique at the level of joint imputationprediction pipelines, rather than choosing imputation and prediction strategies separately; and (ii) a method for dynamically assigning pipeline contribution weights for each sample during the inference phase, which outperforms simple model prediction averaging. Our approach augments a meta-layer on top of existing pipelines for prediction modeling with missing data, enabling better techniques to ensemble prior approaches. We show that this comes at a relatively small cost on top of uniform averaging, discussed more in Section 3.

We evaluate the capability of our model weighting scheme to correctly order its class probability estimates using standard metrics such as AUROC, but also note that another benefit of learning a model weighting function is to better calibrate the class predictions per-sample, i.e. to reduce the absolute error for each sample. Whereas AUROC measures a pipeline's ability to separate classes, we can measure the capability of a model weighting function (e.g. M-DEW) relative to a baseline (e.g. soft voting) on a sample-by-sample basis by looking at the distribution of per-sample error reductions between weighting schemes. We refer to the sample-wise errors as model perplexity.

This paper is organized as follows: Section 2 describes the related work, Section 3 explains our proposed missingness-aware dynamic ensemble weighting (M-DEW) approach, Section 4 presents experimental set up and statistical evaluation of M-DEW compared to uniform model averaging (UMA), and Section 5 is the discussion of conclusions and future work.

2. Related Works

Our method develops an ensembling technique to weigh predictions based on missing data patterns. In this section we review the types of missing data that exist, how to handle them with imputation, and existing ensemble learning methods on which we base our novel contributions.

2.1. Missing Data

Missing values are categorized into three bins: (i) missing at random (MAR), (ii) missing completely at random (MCAR), and (iii) missing not at random (MNAR) defined in Mack et al. (2018). MCAR indicates that missingness of a feature does not depend on the values of other features. For instance, oneoff glitches in one sensor isolated from other sensors yield MCAR missingness. MAR indicates that missingness of a feature depends on the values of observed covariates. MAR missing values are generated by some (either deterministic or nondeterministic) function, which could be parameterized by e.g., a quantile function (patients below a certain age would not be screened for various age-related cancers) or logistic regression. MNAR indicates that missingness of a feature can depend on both the values of other variables as well as whether other variables are missing. This may be rule-based, for instance: on a survey, if questions are asked in multiple languages, then if one question in a given language is left blank, the others will likely be as well.

2.2. Data Imputation

A straightforward method of imputing missing values is to use column-wise central-tendency statistics such as the mean. In this case, the mean of the non-missing values of each column in the dataset is calculated, and any missing value in a column c is imputed to the mean of c. However, this approach can severely perturb the distribution of the data, for instance by decreasing variance and artificially concentrating the probability mass function. Furthermore, if a feature can be modeled as a function of other variables, this dependence is washed away and the conditional / joint distributions become corrupted, in addition to an increase in estimation error as compared to, say, parametric estimation based on the values of other features.

Due to this issue with imputation using any central tendency statistic, we may want to approach the problem as a prediction problem using other features. Some options for imputing continuous features are regression and k-nearest neighbors: one can train a

(singular, multivariate) regression model with the column to impute as the target column, and the samples which have the feature present as the training set; the model can be used to predict the column values for those samples which are missing it. For categorical variables, one may consider a similarly simple classifier, such as logistic regression. One particularly useful model for imputing missing tabular data is the MissForest algorithm proposed by Stekhoven and Bühlmann (2012), which first imputes all missing values in a column to the mean of that column, then for each column with missing values fits a random forest model on the observed portion and predicts the missing values, updating them with the predictions; this process is done over a chosen number of iterations or until some convergence criterion is met. MissForest can simultaneously impute continuous and categorical variables. We note that the idea behind the MissForest approach can be extended to other regression and classification backbones, viz. gradient boosting, and MLPs. However, not all features are most appropriately modeled with a random forest; some may be better modeled by linear models, nearest-neighbor methods, etc. Therefore we may prefer to choose estimators other than random forest, or even an ensemble of other kinds of estimators. There have been attempts to use Generative Adversarial Networks (GANS) and Autoencoders (Psychogyios et al., 2022; Lee et al., 2021) to address missing value imputation. While these methods work well, both of them focused on one specific dataset and use a single imputation technique as opposed to our ensemble approach which adapts the imputer for different sub-populations of the data.

2.3. Ensemble Methods

Ensembling (Dietterich et al., 2002) is a class of machine learning techniques that aims to combine multiple predictors into a stronger predictor. For example, decision tree-based ensemble techniques such as random forest (Breiman, 2001), boosting (Schapire, 1999), and bagging (Breiman, 1996) aim to combat the typically high variance associated with single decision trees, by aggregating weaker classifiers trained on smaller sample sets and/or feature spaces. Ensemble methods can also combine multiple strong predictors into yet a stronger predictor; for instance, voting classifiers (Ruta and Gabrys, 2005) take predictions from multiple (strong) predictors and use majority rule to obtain a final prediction. The stacked generalization algorithm (Wolpert, 1992) trains a metaestimator on the set of already-trained prediction models, learning an optimal strategy (e.g. neural network weights or decision tree splits) for combining individual model predictions into a final; this metaestimator can be any estimator that takes a vector as input, such as a logistic regression model or a decision tree. Extreme Gradient Boost (XGBoost) (Chen and Guestrin, 2016) is a popular implementation of gradient tree boosting, which is designed to be spaceand time-efficient.

2.4. Dynamic Ensemble Weighting

Our method builds on previous work in the area of dynamic ensembling (Zhang et al., 2019; Cruz et al., 2020). As a method that is generally applicable to any classification or regression scenario, DEW has been used for time-series forecasting (Du et al., 2022) (Chowell et al., 2020), (Lu et al., 2022), and learning of non-stationary time-series data representations (Yin et al., 2015). In opposition to static ensemble techniques such as random forest, gradient boosting, voting, or stacked generalization, dynamic ensemble techniques choose an ensemble of estimators and a weighting scheme on a sample-by-sample basis. The typical flow of a dynamic ensembling algorithm starts with a pool of estimators and a training set provided as input, upon which the competence of each estimator is assessed per sample; in the inference phase, either a subset of estimators or a weighting scheme is chosen based on the expected competence of each estimator. When only the estimator with the highest expected competence is chosen, this is called dy*namic estimator selection* (e.g. dynamic classifier selection); when a subset of estimators is chosen, this is called *dynamic ensemble selection*; when a weighting scheme is adapted for each sample in the inference phase, this is called *dynamic ensemble weight*ing. Most similar to our work is (Conrov et al., 2016) which is a variant of AdaBoost (Freund et al., 1999) (Schapire, 2013) that is designed to handle missing values; the main difference between Conroy et al. (2016) and our approach is that their approach optimizes a single pipeline for all the data while ours is a meta-ensembling approach, which can take the model proposed in Conroy et al. (2016) as one pipeline input among others. Cruz et al. (2020) provide a more comprehensive overview of dynamic ensembling methods. However, these dynamic ensembling methods do not take into account the properties of missing data. Our work fills this gap by adapting DEW to accept missing values in the inputs. Whereas these other algorithms operate on complete datasets (i.e. datasets without missing values), our algorithm can handle incomplete datasets and utilize missingness patterns to make better predictions.

3. Approach

The Missingness-aware Dynamic Ensemble Weighting (M-DEW) method we have developed is based on the idea that different methods for handling missing data may work better for certain sub-populations of samples than others. For instance, within some regions of the input space, certain features may be strongly predicted using a nearest-neighbor approach, but in other regions of the input space certain features may be nonlinear functions of subsets of features, perhaps parameterized by a neural network with Rectified Linear Unit (ReLU) activation or a decision tree. The key idea underlying M-DEW is to dynamically assign weights to prediction models trained using different missing-data-handling techniques, based on the data's missingness patterns, in order to more robustly aggregate predictions from these different models.

Algorithm	1	Fit	Dynamic	Ensemble	Weighting
Model					

Procedure: FitM-DEWModel(trainData, trainTargets, valData, valTargets, estimator-Pool)

sampleWiseEstimErrors \leftarrow dict();

for estim in estimatorPool do

estim.fit(trainData, trainTargets);

- valPredictions \leftarrow estim.estimate(valData); # list of predictions, one for each sample
- sampleWiseEstimErrors[estim] \leftarrow |valTargets valPredictions|; # list of errors, one for each sample

end

Return: sampleWiseEstimErrors

Consider an estimator $\mathscr{E} : \mathbb{F}^n \to \mathscr{T}^m$ mapping n-tuples over a field \mathbb{F} to a target space \mathscr{T}^m . Typically, the domain of the input data we receive is $\mathbb{R}^n_{\mathscr{N}}$, where $\mathbb{R}^n_{\mathscr{N}} = \mathbb{R} \cup \{NaN\}$, NaN is the symbol for "missing value". Mechanisms for handling missing data may be thought of as procedures of the form $\mathscr{M} : \mathbb{R}^n_{\mathscr{N}} \to \mathbb{R}^m$ where $m \leq n$. Such procedures include deletion mechanisms – which remove either samples, features, or both which contain missing val-



Figure 1: M-DEW Work Flow Diagram: Phase 1: Imputation and prediction models are fitted to "stage-1" training set. Phase 2: Inference with each imputation-prediction pipeline run on a "stage-2" training set, with pipeline errors stored for each sample. Phase 3: Inference on new samples involves weighting each pipeline's prediction according to its relative competence in the neighborhood of the input sample, i.e. the softmax over pipelines' mean inverse errors.

ues – and imputation mechanisms, which populate the NaN entries with some estimates of what those values might look like, either by central tendency measures or some prediction via other features. As such, they prepare the data to be fed to the estimators for training or inference, such that the new data is compatible with the estimators' domains (i.e., no missing values). In this subsection, we will refer to an estimator as a composite procedure $\mathscr{E} \circ \mathscr{M}$, which first maps from $\mathbb{R}^n_{\mathscr{N}} \to \mathbb{R}^m$ and then from $\mathbb{R}^m \to \{0, 1\}$. Within the scope of this paper, we will strictly use imputers I as opposed to the more general class of \mathscr{M} procedures, thus when we use the term \mathscr{E} for "estimator", we are in fact referring to a composite procedure $\mathscr{E} \circ I : \mathbb{R}^n_{\mathscr{M}} \to \mathbb{R}^m \to \{0, 1\}$.

To fit the model, we first fit a pool of estimators $\mathscr{E}^* = \{\mathscr{E}_1, \cdots, \mathscr{E}_n\}$ which can handle missing data on a training dataset. Then we compile predictions and corresponding errors for each sample in the training set from each estimator. This process is demonstrated in Algorithm 1. This classification error matrix is used in the next phase for determining which pipelines are most competent in different areas of the input space.

In the inference phase (demonstrated in Algorithm 2), for a given sample we impute the sample with each pipeline and perform a k-nearest-neighbor search over the stage-2 imputed training set (see Figure 1), and calculate a normalized error rate for each estimator's predictions in the sample's neighborhood. Algorithm 2 Dynamic Ensemble Weighting Model Prediction

Procedure:	M-DE	WModelPredict	(sample,	train-
	Data,	estimatorPool,	sampleW	ViseEs-
	timErr	ors, nNeighbors)	
$\mathrm{errors} \leftarrow \mathrm{dict}$	z();			

end

weights \leftarrow softmax(normalizedCompetences); prediction \leftarrow dotProduct(weights, estimPredictions); **Return:** prediction

Each estimator is assigned a competence score for the sample's neighborhood as one minus the normalized error; the competence scores reflect each pipeline's

Dataset	# of instances	# of features	types of features	data type	is missing data
EEG Eye State	14980	15	Integer, Real	Multivariate	False
Diabetic Retinopathy	1151	20	Integer, Real	Multivariate	False
Breast Cancer Diagnosis	569	32	Real	Multivariate	False
Breast Cancer Prognosis	198	34	Real	Multivariate	True
Diabetes VCU	100000	55	Integer	Multivariate, Sequential, Time series	True
Myocardial Infarction	1700	124	Real	Multivariate	True

Table	1:	Desc	cription	of f	3 dat	tasets
Table	·	- D CDC	JUDUIOII	OI () au	Jancon

ability to classify the sample with least perplexity. The competence scores are compiled into a vector and the weights per estimator are calculated as the softmax of the vector. It is well-known that ensemble approaches are interpretable (Liang et al., 2022). The competence scores in extending M-DEW can be traced back directly to the pipelines' performances on each of the selected nearest-neighbor samples

Computational Complexity We highlight that our approach comes with minimal overhead in timeand space-complexity. Let n be the sample size of the whole dataset and p be the number of pipelines. The only additional memory requirement is a cached matrix of each baseline classifier's error per sample, which has space complexity O(np + nd); in exact terms, given a dataset split into stage-1 training, stage-2 training, and test sets, for a proportional s_1, s_2, s_3 percentage split, the exact space complexity is $s_2 \cdot n \cdot p$. In our experiments, $s_1 = 0.16$, thus we can approximate the additional space complexity on top of UMA as $\frac{n \cdot p}{6}$. Thus the space complexity is multilinear in the sample size and number of pipelines.

In terms of time complexity, our algorithm operates in two stages: (i) fitting the classification error matrix and (ii) running inference. To fit the classification error matrix requires a running time of O(np)inferences. At inference time on the test set, all that is required is (i) a k nearest neighbor search on the stage-2 training set, (ii) prediction on those k neighbors by p pipelines, and (iii) a mean-reduce operation followed by softmax and scaling of predictions by the calculated weights. For input samples of dimension d, the KNN component has time complexity O(nd+nk), the per-sample inference component has time complexity O(kp), the mean-reduce has time complexity O(pk), and the softmax \rightarrow weight scaling composite is O(p). As the number of test-set samples is linear in the dataset size, the time complexity to run the whole pipeline from training to evaluation after fitting the baseline pipelines is

$$O(np) + O(n) * [O(n(d+k) + O(pk) + O(p)]$$

= $O(np) + O(n^2(d+k) + npk)$
= $O(n(p+n(d+k)) + pk)$
= $O\left(n^2 \left[d+k+\frac{pk}{n}\right]\right)$

The added time complexity is quadratic in the sample size and multilinear in the number of pipelines, nearest-neighbors, and dataset dimension. This presents a tractable cost on top of the standard soft voting. Notably, increasing the size of the ensemble adds only a linear overhead to the whole pipeline, so one can easily scale up the number of pipelines.

One limitation with our approach is nearestneighbor methods on input spaces that contain missing values themselves; in the current M-DEW algorithm, the nearest-neighbor lookup for each pipeline is done on that pipeline's imputed dataset. In this way the M-DEW algorithm does not directly utilize the pattern of missingness intrinsic to a given sample, opting instead to use imputations of that sample and training samples. Utilization of missingness patterns can be explored by constructing distance functions on spaces which allow missing values.

4. Experiments

4.1. Experimental Setup

We evaluate M-DEW on six health-related datasets (described in Table 1): Roesler (2013); Street et al. (1995); Antal and Hajdu (2014); Clore et al. (2014); Golovenkin et al. (2020); Street et al. (1995). Five of these are datasets with < 100,000 samples and one dataset with \geq 100,000 samples. The breast cancer resource (Street et al., 1995) has two datasets: diagnosis and prognosis. All datasets used in this study are openly available on the UCI Machine Learning Repository (Asuncion and Newman, 2007). The criteria for dataset selection were (i) biomedical-related;

	UMA AP	M-DEW AP	UMA AUROC	M-DEW AUROC	% Positive Class	% Samples: M-DEW Error < UMA Error
EEC Eng State MCAD	01 001	91 790	02 200	00 F70	45	71 790
EEG Eye State MCAR	01.201	81.720	83.208	83.373	40	(1.729
EEG Eye State MAR	88.152	88.676	89.328	89.817	45	81.429
EEG Eye State MNAR	82.007	82.389	83.763	84.110	45	72.804
Myocardial Infarction MCAR	84.528	84.300	83.769	83.550	50	62.731
Myocardial Infarction MNAR	89.564	89.510	88.667	88.640	50	72.140
Myocardial Infarction MAR	91.223	91.154	90.321	90.301	50	76.015
Diabetic Retinopathy MNAR	77.370	77.409	72.279	72.414	53	58.471
Diabetic Retinopathy MAR	79.192	79.200	74.285	74.240	53	56.560
Diabetic Retinopathy MCAR	77.349	77.346	72.650	72.657	53	57.168
Breast Cancer Diagnosis MCAR	98.401	98.398	98.593	98.594	37	77.329
Breast Cancer Diagnosis MAR	98.600	98.606	98.935	98.940	37	82.601
Breast Cancer Diagnosis MNAR	98.524	98.546	98.864	98.878	37	77.329
Breast Cancer Prognosis MCAR	30.759	30.852	57.700	57.602	24	62.626
Breast Cancer Prognosis MNAR	36.617	38.053	61.758	61.618	24	62.626
Breast Cancer Prognosis MAR	29.792	29.953	53.487	53.445	24	54.545
Diabetes VCU MCAR	63.324	63.343	67.594	67.608	46	56.125
Diabetes VCU MNAR	66.485	66.551	70.746	70.781	46	59.066
Diabetes VCU MAR	66.850	66.883	71.151	71.169	46	56.185

Table 2: Comparison of M-DEW algorithm to Uniform Model Averaging (UMA) algorithm over six datasets using classification metrics: Average Precision (AP) (Col. 1- 2), Area Under the Receiver-Operator Characteristic Curve (AUROC) (Col. 3-4), % positive class which is percent of samples in positive class i.e. imbalance in dataset (Col. 5) and % of test-set samples on which M-DEW outperformed uniform model averaging UMA (Col. 6).

(ii) contained numerical features (for regression modeling simplicity). A total of 18 experiments were conducted, with 6 datasets and 3 types of missingness. We first create baseline estimators for building the ensemble model for prediction using 4 regression methods as imputers and 2 classifiers for prediction. Next, we construct a corresponding error matrix to capture the competence score and then finally construct the M-DEW ensemble model. More details about imputers and classifiers can be found in the supplementary documents.

4.2. Imputers, Classifiers and Missingness

We used four types of regression models for imputation: a k-nearest neighbor (KNN) imputer, a Bayesian ridge regression imputer, an XGBoost regressor, and a random forest regressor. KNN and linear regression models are the two most straightforward and commonly-used learning algorithms in imputation and have popular implementations in scikitlearn (Kramer and Kramer, 2016). We chose random forest due to the popularity of the MissForest algorithm and included XGBoost as another strong ensembling approach. For all experiments, the XG- Boost models had a max tree depth of 4 and 50 boosting rounds. Each random forest model was similarly built with 50 trees with a max-depth of 4. Downstream of the imputers, we used two types of classifiers to make predictions: XGBoost and random forest. Like the imputers, the XGBoost models had 50 boosting rounds and a max tree depth of 4, and each random forest model was built from 50 trees with a max tree depth of 4. The imputers were implemented using scikit-learn's IterativeImputer and KN-NImputer classes. The Bayesian Ridge regression parameters were set to the scikit-learn default. Details of the implementation of all of our models—imputers, classifiers, and M-DEW models—are available in Appendix A.

To determine M-DEW's performance on different forms of missing data, we randomly introduced synthetic missing values into the datasets to emphasize the various missingness characteristics discussed in Section 2. This was done for two reasons: (i) not all datasets we selected contain missing values as shown in Table 1, and (ii) to evaluate our model in similar controlled settings, i.e., settings with similar amounts of missingness. In our experiments, for MCAR-type missingness, 30% of values in the dataset were masked at random. For MAR and MNAR, 30%of the columns contained missing values at a rate of 30% of samples, which depended on $\frac{3}{7}$ of the remaining columns via a logistic regression model with randomly assigned weights; in the case of MNAR, the input features to the logistic regression model were masked at random at a rate of 30%, so that the missing values in the columns to be masked depended also on the missing values in the inputs that generated the mask – a feature not present in MAR, which distinguishes MAR from MNAR. A Python port of the R-misstastic library (Mayer et al., 2021) was used to systematically introduce missingness to the data. Details of the implementation of all of our models imputers, classifiers, and DEW models – are available in the appendix.



Figure 2: Violin Plot of AUROC for 8 standard imputer-estimators, baseline UMA pipeline and M-DEW pipeline's performance rankings. All metrics in M-DEW ranked at the first place with narrower range. The wider the shape got in the plots, the more samples filled in

4.3. Evaluation

We compare M-DEW's performance to that of the naive (uniform averaging) ensembling approach using standard classification metrics across six binary classification datasets and three types of missingness (MCAR, MAR, MNAR). We report the average precision score and AUROC score across 18 experiments for M-DEW and UMA in Table 2. Given that a major goal of the M-DEW approach is to reduce the predicted class probability error (i.e. perplexity) compared to the uniform averaging case, we also report the fraction of samples for which M-DEW errors are less than UMA errors, as a measure of the amount of time that M-DEW-calculated weights are directionally correct relative to the baseline.

In addition to reporting the fraction of samples for which M-DEW improved on UMA, to quantify the statistical significance of these sample-wise error improvements, we run a paired t-test over all samples in each dataset and report the p-value.





Figure 3: Frequency of rankings that M-DEW has most 1st place ranking among other models

4.3.1. MAIN RESULTS

As shown in Table 2, M-DEW outperforms UMA on the average precision score in 13/18 trials and on AU-ROC in 11/18 trials. We noted that in cases with more significant class imbalance (breast cancer diagnosis and prognosis), average precision may be a better metric than AUROC, because AUROC can still be quite high while reporting large numbers of false positives; in these cases, M-DEW outperforms UMA on average precision in 5/6 cases, and essentially performs the same in the sixth case. In every experiment, M-DEW classification probability error was globally lower than UMA classification probability error.

In some cases, one of the baseline pipelines may outperform the ensemble methods if it performs more strongly than the other baselines. However, we hypothesized that M-DEW would be robustly among the top-performing models when compared with UMA and the baseline models. To test this, we plotted the relative rankings among the 8 baselines, UMA, and M-DEW on the AUROC metric across the 18 experiments as a violin plot in Figure 2. Indeed, we see that M-DEW is more consistently better-ranked than the other methods, including UMA; M-DEW's ranking is concentrated heavily among the top 2-3. The other pipelines (including UMA) have median rankings higher than M-DEW, and much higher ranking variance, indicating the stability of M-DEW.

To test M-DEW's ability to improve sample-wise calibration, we run a paired t-test for each dataset that compares the class probability errors between M-DEW and UMA. This significance test confirmed the alternative hypothesis that the M-DEW error distribution has a lower mean than the UMA error distribution's mean. 17 out of 18 tests indicate that the DEW has a lower error per-sample.

Dataset	P-Value
EEG Eye State MCAR	< 0.0001
EEG Eye State MAR	< 0.0001
EEG Eye State MNAR	< 0.0001
Myocardial Infarction MCAR	$1.3 imes10^{-07}$
Myocardial Infarction MNAR	$1.5 imes10^{-26}$
Myocardial Infarction MAR	$1.20 imes \mathbf{10^{-31}}$
Diabetic Retinopathy MNAR	$5.0 imes10^{-05}$
Diabetic Retinopathy MAR	0.0041
Diabetic Retinopathy MCAR	0.0017
Breast Cancer Diagnosis MCAR	0.0002
Breast Cancer Diagnosis MAR	< 0.0001
Breast Cancer Diagnosis MNAR	< 0.0001
Breast Cancer Prognosis MCAR	0.0271
Breast Cancer Prognosis MNAR	0.0009
Breast Cancer Prognosis MAR	0.1491
Diabetes VCU MCAR	< 0.0001
Diabetes VCU MNAR	< 0.0001
Diabetes VCU MAR	< 0.0001

Table 3: Results of paired t-test; $p \leq 0.05$ We tested the hypothesis the M-DEW error distribution has a lower mean than the UMA error distribution.

As listed in Table 3, per-sample M-DEW errors are significantly less than the corresponding UMA errors for most samples, in all but one trial. Plots of the histograms of relative error improvements can be seen in Figure 4 and 5. The plots show M-DEW Error minus UMA Error. Therefore, the more dense the plot is to the left of 0, the better M-DEW performs relative to UMA. We can see that especially for the myocardial infarction and EEG eye state experiments, the distributions are visibly concentrated to the left of zero, which is reflected in the results of paired t-test. For the other datasets, the error difference is concentrated closer to zero, yet still shifted left, indicating that while the magnitude of improvement is lower, M-DEW still tends to be directionally correct. Indeed, in every experiment, M-DEW errors are less than UMA errors for the majority of samples. More details and a thorough discussion of calibration analysis and the Brier score can be found in Appendix A.

5. Conclusion and Future Work

We have designed and evaluated an approach to extend the classical notion of dynamic ensemble weighting to datasets with missing data. Whereas previous approaches would have required first imputing and then training classifiers on the imputed dataset, we develop a technique for directly optimizing at the level of joint imputation-prediction pipelines. This approach estimates a soft ordering of pipelines by calculating expected relative competences of each pipeline via the pipelines' performances on nearby samples from the training set. Using relative expected competence scores as weights to scale the pipelines' predictions reduces model perplexity compared with uniform soft voting over pipelines, evidenced by the paired t-test results in Table 3. It improves classification probability error in the majority of samples, leading to better calibrated prediction distributions. This can be useful when calculating disease risk scores or in scenarios where quantifying uncertainty is important.

The small overhead required by the algorithm allows it to be used as an AutoML module; instead of performing model selection, one can use all baseline models to form a stronger predictor. M-DEW also preserves any interpretability afforded by the baseline models; it provides a clear, nearest-neighbor approach to assigning competence-based weights, so for each sample one can trace the contributions of each pipeline via that sample's nearest neighbors' competences in M-DEW's second training stage.

In extending the dynamic ensembling approach to handle missing values in the inputs, our work paves the way for future research along these lines. M-DEW makes no assumptions about underlying input structure, as long as there is a suitable distance measure between inputs and suitable objective function for outputs.

We plan to extend the approach to utilize learned representations of the inputs. We are also investigating ways to jointly optimize the imputation and



Figure 4: Relative Error Histograms-EEG Eye State



Figure 5: Relative Error Histograms- Myocardial Infarction

downstream prediction models using differentiable programs such as neural networks with combined loss functions.

6. Acknowledgements

Research reported in this publication was supported by the National Library Of Medicine of the National Institutes of Health under Award Number R01LM013327. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

- Balint Antal and Andras Hajdu. Diabetic Retinopathy Debrecen. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5XP4P.
- Arthur Asuncion and David Newman. Uci machine learning repository, 2007.
- Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45 (1):5–32, 2001.

- Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pages 785– 794, 2016.
- Gerardo Chowell, R Luo, K Sun, Kimberlyn Roosa, Amna Tariq, and C Viboud. Real-time forecasting of epidemic trajectories using computational dynamic ensembles. *Epidemics*, 30:100379, 2020.
- John Clore, Krzysztof Cios, Jon DeShazo, and Beata Strack. Diabetes 130-US hospitals for years 1999-2008. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5230J.
- Bryan Conroy, Larry Eshelman, Cristhian Potes, and Minnan Xu-Wilson. A dynamic ensemble approach to robust classification in the presence of missing data. *Machine Learning*, 102:443–463, 2016.
- Rafael MO Cruz, Luiz G Hafemann, Robert Sabourin, and George DC Cavalcanti. Deslib: A

Mach. Learn. Res., 21(8):1-5, 2020.

- Thomas G Dietterich et al. Ensemble learning. The handbook of brain theory and neural networks, 2(1): 110-125, 2002.
- Liang Du, Ruobin Gao, Ponnuthurai Nagaratnam Suganthan, and David ZW Wang. Bayesian optimization based dynamic ensemble for time series forecasting. Information Sciences, 591:155-175, 2022.
- Tlamelo Emmanuel, Thabiso Maupong, Dimane Mpoeleng, Thabo Semong, Banyatsang Mphago, and Oteng Tabona. A survey on missing data in machine learning. 8(1):140, 2021. ISSN 2196-1115. doi: 10.1186/s40537-021-00516-9. URL https: //doi.org/10.1186/s40537-021-00516-9.
- Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence, 14(771-780): 1612, 1999.
- S.E. Golovenkin, V.A. Shulman, D.A. Rossiev, P.A. Shesternya, S.Yu. Nikulina, Yu.V. Orlova, and V.F. Voino-Yasenetsky. Myocardial infarction complications. UCI Machine Learning Repository, 2020. DOI: https://doi.org/10.24432/C53P5M.
- Oliver Kramer and Oliver Kramer. Scikit-learn. Machine learning for evolution strategies, pages 45–53, 2016.
- Woonghee Lee, Jaeyoung Lee, and Younghoon Kim. Contextual imputation with missing sequence of eeg signals using generative adversarial networks. *IEEE Access*, 9:151753–151765, 2021. doi: 10. 1109/ACCESS.2021.3126345.
- Minfei Liang, Ze Chang, Zhi Wan, Yidong Gan, Erik Schlangen, and Branko Šavija. Interpretable ensemble-machine-learning models for predicting creep behavior of concrete. Cement and Concrete Composites, 125:104295, 2022.
- Hongliang Lu, Hongjun Su, Pan Zheng, Yihan Gao, and Qian Du. Weighted residual dynamic ensemble learning for hyperspectral image classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 15:6912–6927, 2022.

- dynamic ensemble selection library in python. J. Christina Mack, Zhaohui Su, and Daniel Westreich. Types of missing data. In Managing Missing Data in Patient Registries: Addendum to Registries for Evaluating Patient Outcomes: A User's Guide, Third Edition [Internet]. Agency for Healthcare Research and Quality (US), 2018. URL https: //www.ncbi.nlm.nih.gov/books/NBK493614/.
 - David J. C. MacKay. Bayesian interpolation. In C. Ray Smith, Gary J. Erickson, and Paul O. Neudorfer, editors, Maximum Entropy and Bayesian Methods: Seattle, 1991, pages 39–66. Springer Netherlands, 1992. ISBN 978-94-017-2219-3. doi: 10.1007/978-94-017-2219-3_3. URL https://doi. org/10.1007/978-94-017-2219-3_3.
 - Imke Mayer, Aude Sportisse, Julie Josse, Nicholas Tierney, and Nathalie Vialaneix. R-miss-tastic: a unified platform for missing values methods and workflows. arXiv preprint arXiv:1908.04822, 2019.
 - Imke Mayer, Aude Sportisse, Julie Josse, Nicholas Tierney, and Nathalie Vialaneix. R-miss-tastic: a unified platform for missing values methods and workflows, 2021.
 - Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised In Proceedings of the 22nd Inlearning. Conference on Machine Learning ternational '05, pages 625–632. ACM Press, ICML2005.ISBN 978-1-59593-180-1. doi: 10.1145/ 1102351.1102430. URL http://portal.acm.org/ citation.cfm?doid=1102351.1102430.
 - F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . Journal of Machine Learning Research, 12:2825-2830, 2011.
 - John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv. Large Margin Classif., 10, 06 2000.
 - Konstantinos Psychogyios, Loukas Ilias, and Dimitris Askounis. Comparison of missing data imputation methods using the framingham heart In 2022 IEEE-EMBS Internastudy dataset. tional Conference on Biomedical and Health Informatics (BHI). IEEE, September 2022. doi:

10.1109/bhi56158.2022.9926882. URL http://dx. doi.org/10.1109/BHI56158.2022.9926882.

- Oliver Roesler. EEG Eye State. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C57G7J.
- Dymitr Ruta and Bogdan Gabrys. Classifier selection for majority voting. *Information fusion*, 6(1):63– 81, 2005.
- Robert E Schapire. A brief introduction to boosting. In *Ijcai*, volume 99, pages 1401–1406. Citeseer, 1999.
- Robert E Schapire. Explaining adaboost. In Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik, pages 37–52. Springer, 2013.
- Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28 (1):112–118, 2012.
- W Nick Street, Olvi L Mangasarian, and William H Wolberg. An inductive learning approach to prognostic prediction. In *Machine Learning Proceedings* 1995, pages 522–530. Elsevier, 1995.
- Michael E. Tipping. Sparse bayesian learning and the relevance vector machine. J. Mach. Learn. Res., 1:211-244, sep 2001. ISSN 1532-4435. doi: 10.1162/15324430152748236. URL https://doi-org.ezproxy.gc.cuny.edu/ 10.1162/15324430152748236.
- Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for DNA microarrays. 17(6):520–525, 2001. ISSN 1367-4803. doi: 10.1093/bioinformatics/17.6.520. URL https://doi.org/10.1093/bioinformatics/17.6.520.
- Ian R White and John B Carlin. Bias and efficiency of multiple imputation compared with complete-case analysis for missing covariate values. *Statistics in medicine*, 29(28):2920–2931, 2010.
- Ian R White, Patrick Royston, and Angela M Wood. Multiple imputation using chained equations: issues and guidance for practice. *Statistics in medicine*, 30(4):377–399, 2011.

- David H Wolpert. Stacked generalization. Neural networks, 5(2):241–259, 1992.
- Xu-Cheng Yin, Kaizhu Huang, and Hong-Wei Hao. De2: dynamic ensemble of ensembles for learning nonstationary data. *Neurocomputing*, 165:14–22, 2015.
- Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 694–699, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 158113567X. doi: 10.1145/775047.775151. URL https://doi-org.ezproxy.gc.cuny.edu/10.1145/775047.775151.
- Shichao Zhang. Nearest neighbor selection for iteratively knn imputation. Journal of Systems and Software, 85(11):2541–2552, 2012.
- Zhong-Liang Zhang, Yu-Yu Chen, Jing Li, and Xing-Gang Luo. A distance-based weighting framework for boosting the performance of dynamic ensemble selection. *Information Processing & Management*, 56(4):1300–1316, 2019.

Appendix A. Technical Appendix

In the appendix we provide details for the baseline code (Section 1), hardware information(Section 2), and certainty analysis with reliability diagrams for M-DEW(Section 3).

A.1. Baseline Code

Our baseline is generated with several *sklearn* modules, *Xgboost*, and *R-miss-tastic*.

- Uniform Model Averaging (UMA) This is a technique that gives equal weight to each individual model's prediction when making a final prediction. It serve as a baseline ensemble method.
- R-miss-tastic (Mayer et al., 2019) This method generates missing values from nonmissing datasets with customized distribution of missing data percentage. https: //github.com/R-miss-tastic/website/ tree/master/static/how-to/python
- KNN (Troyanskaya et al., 2001) The method uses the weighted average of its K-nearest neighbors to impute the missing value. https://scikit-learn.org/stable/ modules/generated/sklearn.impute. KNNImputer.html
- Bayesian Ridge (MacKay, 1992; Tipping, 2001) This method is based on Bayesian linear regression and is designed to handle regression problems while taking into account uncertainty in the model parameters. https://scikit-learn.org/stable/ modules/generated/sklearn.linear_model. BayesianRidge.html
- Xgboost (Chen and Guestrin, 2016) Xgboost is applied in both imputation stage and prediction stage. It is the effective and efficient method to deal with missing data. https://github.com/dmlc/xgboost/blob/ 36eb41c960483c8b52b44082663c99e6a0de440a/ doc/index.rst
- Random Forest (Breiman, 2001) This method is applied in both imputation stage and prediction stage. It is a meta estimator that employs averaging to increase predicted accuracy and reduce overfitting after fitting

numerous decision tree classifiers to different dataset subsamples. https://scikit-learn. org/stable/modules/generated/sklearn. ensemble.RandomForestRegressor.html# sklearn.ensemble.RandomForestRegressor https://scikit-learn.org/stable/ modules/generated/sklearn.ensemble. RandomForestClassifier.html

A.2. Hardware Information

• CPU

Model name:	Intel(R)Xeon(R)CPU E5-2687W v4@3.00GHz
CPU(s):	48
Thread(s) per core:	2
CPU max MHz:	3500.0000
L1d cache:	32K
L1i cache:	32K
L2 cache:	256K
L3 cache:	30720K
GPU	
GPUO:	Quadro K620
GPU1:	NVIDIA GeForce

A.3. Relative Error Histograms for datasets

Plots of the histograms of relative error improvements can be seen in Figure 6. The plots show M-DEW Error - UMA Error. Therefore, the more dense the plot is to the left of 0, the better DEW performs relative to UMA.

A.4. Calibration Analysis

Calibration analysis (Platt, 2000; Niculescu-Mizil and Caruana, 2005; Zadrozny and Elkan, 2002) is normally applied after training to statistically measure the model's ability with prediction probability. It is sophisticated when classifying a data point to a correct label is important. Motivated by M-DEW's ability to reduce perplexity, we extend it by adding a further layer of calibration on top.

CalibratedClassifierCV¹ (Pedregosa et al., 2011) is an effective tool for enhancing the performance of predictive algorithms, particularly in classification tasks.

https://scikit-learn.org/stable/modules/generated/ sklearn.calibration.CalibratedClassifierCV.html

Figure 6: Relative Error Histograms

















DEW error - UMA Error, wisconsin_bc_prognosis MCAR







DEW error - UMA Error, eeg_eye_state MNAR -0.08 -0.06 -0.04 -0.02 0.00 0.02



DEW error - UMA Error, wisconsin_bc_diagnosis MNAR



DEW error - UMA Error, wisconsin_bc_prognosis MNAR 160 140 120 100 80 60 40 20 0 **II** II





50 40 30 20 10

This algorithmic technique aims to improve the reliability of probabilistic predictions made by classifiers, ensuring they are well-calibrated and reflect the true likelihood of class membership. By combining a base classifier with a calibration model, CalibratedClassifierCV transforms the raw predicted probabilities into more accurate estimates, thus providing more meaningful confidence scores.

CalibratedClassifierCV can significantly enhance its predictive capabilities when applied to a new algorithm. The new algorithm can first generate raw predicted probabilities for each class, which are often optimistic and may not accurately represent the true class probabilities. Next, the calibrated model produces more accurate and well-calibrated probabilities, an S-shape curve. For the M-DEW algorithm, We experiment calibration analysis for all six datasets.

There are three groups of reliability diagrams with Sigmoid calibration as Figure ??. In plotting below, the x-axis represents the average predicted probability in each bin. The y-axis is the fraction of positives, i.e., the proportion of samples whose class is the positive class (in each bin). The term "bin" in refers to the number of "bins" or calibration intervals used to transform the raw predicted scores into calibrated probabilities. By dividing the predicted probabilities into bins, we can observe how well the predicted probabilities align with the actual fraction of positive samples within each bin of our model. The rest are the calibrated results(reliability diagrams).

The ideal shape of curves should be close to the Sshape and instead appear to be fluctuating for dataset Wisconsin Breast Cancer diagnosis and prognosis. It usually suggests that the models may have some degree of miscalibration or inconsistency in their predicted probabilities. However, those two datasets are not in S-shape because of sample size or data imbalance. Fluctuations in the calibration curve can also occur when the sample size is small, or there is a significant class imbalance in the data. The calibration curve may not smooth out in such cases due to limited data points in certain regions.



Figure 7: M-DEW's imputing-prediction pipeline. There are total 8 base estimators to construct M-DEW. Random Forest, XGBoosting, and KNN are popular imputation technique

Experiment	UMA	M-DEW
eeg_eye_state MNAR	0.259	0.2601360786916317
eeg_eye_state MCAR	0.2587541559455349	0.2596832891854269
eeg_eye_state MAR	0.2599553776319339	0.26178878397810695
diabetic_retinopathy MAR	0.3020440141903601	0.3031740221628122
diabetic_retinopathy MNAR	0.2891785425641328	0.2902275287724473
diabetic_retinopathy MCAR	0.30269251203479697	0.30352194179421066
wisconsin_bc_prognosis MCAR	0.19828763284593903	0.1984543459655042
wisconsin_bc_prognosis MNAR	0.18263723211754626	0.18393224974616498
wisconsin_bc_prognosis MAR	0.20945584159775288	0.2095635635424871
wisconsin_bc_diagnosis MCAR	0.4237567782008261	0.42418764860161845
wisconsin_bc_diagnosis MNAR	0.4238170852085251	0.4241030127687644
wisconsin_bc_diagnosis MAR	0.4221709207468057	0.42236447517994835
myocardial_infarction MCAR	0.2954491814718572	0.2976526040103881
myocardial_infarction MAR	0.298894038746503	0.3027085660867358
myocardial_infarction MNAR	0.31277641034218473	0.3156143066211194
Diabetes_vcu MAR	0.2630907363015426	0.2637330086475949
Diabetes_vcu MCAR	0.25845379258736195	0.2587403688534198
Diabetes_vcu MNAR	0.26287367651477866	0.2634343516876172

Table 4: Brier Score-a metric for evaluating the precision of probability predictions, particularly in binary classification tasks. During the development of M-DEW, the Brier Score offered trustworthy probability estimates, which is crucial for risk assessment and decision-making procedures. Dew's results outperformed UMA

Dataset	\mathbf{UMA}		M-DEW			
	AUC	Acc.	F 1	AUC	Acc.	F1
Eye-eeg-state MAR	0.894	0.806	0.764	0.899	0.811	0.771
Eye-eeg-state MCAR	0.833	0.750	0.687	0.837	0.754	0.693
Eye-eeg-state MNAR	0.838	0.758	0.697	0.842	0.760	0.701
Diabetic Retinopathy MAR	0.754	0.681	0.681	0.731	0.664	0.664
Diabetic Retinopathy MCAR	0.666	0.633	0.633	0.611	0.595	0.594
Diabetic Retinopathy MNAR	0.732	0.665	0.665	0.733	0.664	0.663
Breast Cancer Wisconsin Diagnostic MAR	0.990	0.947	0.947	0.990	0.947	0.947
Breast Cancer Wisconsin Diagnostic MCAR	0.989	0.954	0.954	0.989	0.953	0.952
Breast Cancer Wisconsin Diagnostic MNAR	0.993	0.949	0.949	0.993	0.947	0.947
Breast Cancer Wisconsin Prognostic MAR	0.544	0.742	0.690	0.546	0.737	0.682
Breast Cancer Wisconsin Prognostic MCAR	0.589	0.758	0.680	0.589	0.758	0.680
Breast Cancer Wisconsin Prognostic MNAR	0.623	0.763	0.709	0.623	0.763	0.709
Myocardial infarction MAR	0.902	0.810	0.795	0.903	0.814	0.801
Myocardial infarction MCAR	0.834	0.740	0.728	0.831	0.738	0.727
Myocardial infarction MNAR	0.892	0.788	0.779	0.892	0.788	0.779
Diabetes 130 hospitals MAR	0.712	0.652	0.572	0.713	0.653	0.575
Diabetes 130 hospitals MCAR	0.677	0.625	0.513	0.677	0.625	0.515
Diabetes 130 hospitals MNAR	0.708	0.649	0.564	0.709	0.650	0.566

Table 5: Standard classification metrics - AUROC (AUC), accuracy (Acc.), F1-score (F1), between UMA and M-DEW. We use AUROC to assess model discrimination between classes, accuracy to measure the overall correctness of predictions, and the F1 score to balance precision and recall, particularly in imbalanced datasets. In most cases, M-DEW performed better than UMA.